

智果（AgentArts）智能体平台

托管与运行智能体

文档版本

01

发布日期

2026-09-24



版权所有 © 华为云计算技术有限公司 2026。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址：贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址：<https://www.huaweicloud.com/>

目录

- 1 托管与运行智能体概述..... 1
- 2 入门示例：创建基础对话智能体..... 3
- 3 进阶示例：构建全能出行助手（集成模型/记忆/代码解释器/网关/高德 mcp）..... 19
- 4 运行时..... 54
 - 4.1 运行时概述..... 54
 - 4.2 工作原理..... 56
 - 4.3 入站协议..... 57
 - 4.3.1 HTTP..... 57
 - 4.3.2 MCP..... 61
 - 4.4 权限与访问控制..... 62
 - 4.4.1 委托介绍..... 62
 - 4.4.2 入站身份认证..... 68
 - 4.5 示例：部署第一个 Agent 到运行时..... 73
 - 4.6 部署 Agent..... 84
 - 4.6.1 制作 Agent 镜像..... 84
 - 4.6.2 通过控制台部署..... 86
 - 4.6.3 通过 SDK 部署..... 97
 - 4.7 部署 MCP Server..... 101
 - 4.7.1 什么是 MCP Server 部署..... 101
 - 4.7.2 制作 MCP Server 镜像..... 102
 - 4.7.3 部署到运行时..... 107
 - 4.8 调用运行时..... 111
 - 4.8.1 调用概述..... 111
 - 4.8.2 调用智能体..... 115
 - 4.8.3 会话操作..... 120
 - 4.8.4 执行命令..... 123
 - 4.8.5 文件上传与下载..... 126
 - 4.8.6 自定义路径调用..... 130
 - 4.8.7 调用 MCP 服务..... 135
 - 4.9 会话管理..... 137
 - 4.10 存储配置..... 140
 - 4.11 观测运行时..... 145

4.12 管理运行时.....	146
4.12.1 查看并管理运行时.....	146
4.12.2 管理访问方式.....	148
4.12.3 灰度发布.....	150
4.12.4 更新运行时镜像.....	151
5 记忆库.....	153
5.1 记忆库概述.....	153
5.2 工作原理.....	155
5.2.1 分层记忆架构.....	155
5.2.2 短期记忆.....	155
5.2.3 长期记忆.....	156
5.2.4 长期记忆生成机制.....	157
5.2.5 长期记忆检索机制.....	158
5.2.6 记忆组织与隔离.....	159
5.3 记忆策略.....	159
5.3.1 策略概述.....	159
5.3.2 内置策略.....	161
5.3.2.1 总结策略.....	162
5.3.2.2 语义记忆策略.....	163
5.3.2.3 用户偏好策略.....	164
5.3.2.4 情景记忆策略.....	166
5.3.2.5 程序记忆策略.....	167
5.3.3 自定义策略.....	168
5.4 示例：构建记忆库并集成 SDK 进行对话管理.....	170
5.5 创建记忆库.....	176
5.6 记忆库入站身份认证.....	181
5.7 在 Agent 中集成记忆库.....	183
5.7.1 概述.....	183
5.7.2 环境准备.....	185
5.7.3 连接记忆库.....	187
5.7.4 管理会话.....	188
5.7.5 短期记忆操作.....	189
5.7.6 长期记忆操作.....	190
5.8 安装记忆插件.....	191
5.9 记忆检索.....	194
5.9.1 通过控制台检索.....	194
5.9.2 通过 SDK 检索.....	195
5.10 观测记忆指标.....	196
5.10.1 支持的观测指标.....	196
5.10.2 查看观测指标.....	198
5.10.3 创建告警规则.....	199
5.11 管理记忆库.....	200

5.11.1 查看记忆库列表与详情.....	200
5.11.2 管理记忆策略.....	201
5.11.3 更新 API Key.....	203
5.11.4 管理标签.....	203
5.11.5 删除记忆库.....	204
6 网关.....	205
6.1 网关概述.....	205
6.2 工作原理.....	206
6.3 示例：集成高德 MCP 服务实现旅行攻略制定.....	208
6.4 创建网关.....	214
6.5 创建 Target.....	222
6.5.1 Target 介绍.....	222
6.5.2 创建 MCP Target.....	223
6.5.3 创建推理 Target.....	230
6.5.4 MCP 协议支持的 Target 类型介绍.....	233
6.5.4.1 REST API.....	233
6.5.4.2 MCP.....	240
6.5.4.3 APiG.....	240
6.5.4.4 云服务.....	249
6.5.5 Target 的出站认证配置.....	251
6.6 调测与调用 MCP Target.....	253
6.6.1 概述.....	253
6.6.2 调测网关.....	254
6.6.3 列出网关工具.....	255
6.6.4 调用网关工具.....	257
6.6.5 检索网关工具.....	259
6.7 调用推理 Target.....	259
6.8 管理网关.....	261
7 代码解释器.....	264
7.1 代码解释器介绍.....	264
7.2 示例：基于控制台的代码解释器实践.....	265
7.3 示例：基于 SDK 的代码解释器实践.....	271
7.4 创建代码解释器.....	277
7.5 在 Agent 中集成代码解释器.....	282
7.6 管理代码解释器.....	284
8 浏览器.....	286
8.1 浏览器介绍.....	286
8.2 示例：基于控制台的浏览器工具开发与应用实践.....	288
8.3 示例：基于 SDK 的浏览器工具开发与应用实践.....	294
8.4 创建浏览器.....	299
8.5 在智能体中集成浏览器.....	304

8.6 浏览器会话管理.....	306
8.6.1 启动浏览器会话.....	306
8.6.2 启动会话时配置浏览器代理.....	308
8.6.3 查看浏览器会话.....	310
8.6.4 人工接管浏览器会话.....	311
8.6.5 终止浏览器会话.....	312
8.6.6 浏览器支持的快捷操作.....	316
8.7 浏览器配置文件管理.....	318
8.7.1 新建配置文件.....	318
8.7.2 将会话保存至配置文件.....	320
8.7.3 管理配置文件.....	322
8.8 管理浏览器.....	323
9 知识库.....	325
9.1 知识库介绍.....	325
9.2 知识库类型.....	328
9.3 知识库使用限制.....	329
9.4 创建平台知识库.....	330
9.4.1 创建知识库.....	330
9.4.2 上传文档.....	335
9.4.3 知识库命中测试.....	342
9.4.4 知识库分段.....	344
9.5 接入第三方知识库.....	349
9.5.1 第三方知识库分类.....	349
9.5.2 连接 General 知识库.....	350
9.5.3 连接 KooSearch 知识库.....	358
9.5.4 连接 RAGFlow 知识库.....	366
9.5.5 知识库命中测试.....	373
9.5.6 第三方通用知识库接入规范.....	375
9.6 在低码智能体中使用知识库.....	385
9.6.1 在单智能体中使用知识库.....	385
9.6.2 在工作流中使用知识库.....	389
9.7 管理知识库.....	396
10 模型.....	399
10.1 模型介绍.....	399
10.2 创建模型提供商.....	402
10.3 添加模型服务.....	406
10.4 添加模型代理.....	409
10.5 集成模型提供商.....	411
11 身份认证.....	414
12 网络配置.....	417
13 AgentArts 运行时 SDK.....	420

13.1 SDK 介绍.....	420
13.2 SDK 应用框架.....	422
13.3 快速开始.....	424
13.4 CLI.....	435
13.5 Runtime SDK.....	458
13.6 CodeInterpreter SDK.....	463
13.7 Browser SDK.....	467
13.8 Memory SDK.....	472
13.9 Identity SDK.....	477
13.10 Gateway SDK.....	479
14 常见问题.....	498
14.1 认证鉴权.....	498
14.2 如何获取代码解释器的 API_KEY.....	499
14.3 如何获取浏览器的 API_KEY.....	499
14.4 如何获取浏览器配置文件的 ID.....	500
14.5 如何获取记忆库 ID 和 API Key.....	501
14.6 如何使用记忆库的 API Key.....	502
14.7 获取华为云 MaaS 服务模型 API Key.....	503
14.8 Dockerfile 参数在智能体运行时中如何使用.....	503
14.9 执行 pip install agentarts-sdk 命令安装 AgentArts SDK 时速度较慢或下载超时.....	505
14.10 执行 pip install agentarts-sdk 命令安装 AgentArts SDK 时出现 externally-managed-environment 报错.....	505
14.11 如何将已有的 Agent 应用迁移至 AgentArts 运行时.....	506
14.12 创建网关后调用报错：委托缺少 CSMS/KMS 相关 action 权限.....	510

1 托管与运行智能体概述

AgentArts为专业开发者提供了一条从代码到生产的完整链路——在熟悉的本地环境中使用主流框架开发，通过标准化SDK接入平台能力，以镜像形式一键部署至云端托管环境，并享受生产级的可观测与安全隔离保障。

核心价值：

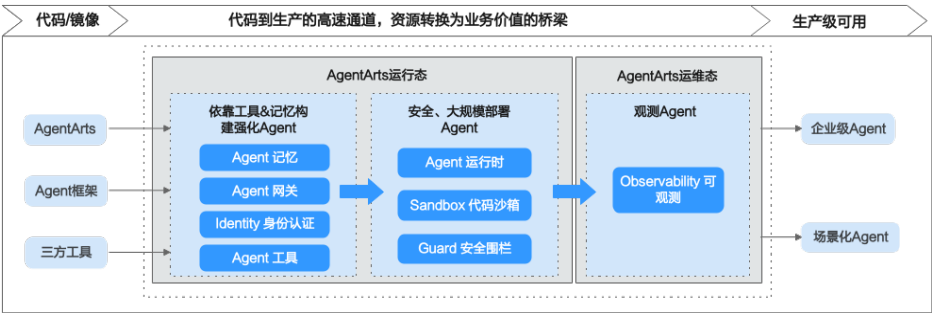
开发自由度高：使用LangChain/LangGraph等主流框架，代码完全自主可控。

平台能力即插即用：记忆、网关、工具集成等组件开箱可用，无需从零搭建。

生产级运行保障：安全隔离、弹性部署、全链路可观测一应俱全。

两种开发模式：依托平台能力配置开发或直接迁移已有代码包，灵活适配不同场景。

产品架构



- AgentArts高代码开发的整体架构由开发态、运行态与运维态三部分构成：
- 本地代码开发：支持基于主流LangChain/LangGraph框架编写的代码、AgentArts平台在线编排产出的资产，以及第三方定制的工具，均可通过标准化的镜像形式无缝接入平台。
 - 运行态：为智能体提供云端托管环境与业务能力扩展组件。
 - 托管环境：包含Agent运行时（Runtime）、代码沙箱（Sandbox）与安全围栏（Guard），支持智能体的弹性部署与安全隔离执行。
 - 扩展环境：包含Agent记忆（Memory）、网关（Gateway）、工具集成（Tools）与身份认证（Identity）。开发者可开箱即用地实现上下文持久化、企业内部工具的安全调用与标准化鉴权。
 - 运维态：提供智能体上线后的可观测能力，保障生产级可用性。

- 可观测：智能体运行时、网关、沙箱工具的日志，开发者可以实时监控智能体运行状态、排查故障。

产品功能

表 1-1 产品功能

功能	说明
记忆库	● 提供长短期记忆、分级存储及多种长期记忆抽取策略，支持自定义built-in策略，完善企业级记忆能力。 ● 通过高码SDK和API接入，支持LangChain等平台的平滑接入。
代码解释器	为Agent提供安全隔离的代码执行环境，Agent可以在代码解释器中执行代码、处理文件、运行命令，所有操作在受控环境中完成。
浏览器	为智能体提供安全隔离的云端浏览器运行环境，使智能体具备访问任意网站、操作页面、获取截图、执行多步表单填写等Web自动化能力。
网关	提供API转换MCP能力，支持多种网关Target集成，灵活适配不同场景需求。
智能体运行时	● 基于容器化技术实现进程级隔离，提供企业级的运行时可靠性及安全保障。 ● 支持基于业务流量（QPS/并发数）的自动扩缩容能力，从容应对流量洪峰。 ● 深度集成身份与访问管理服务，支持身份认证、灵活的网络配置及安全委托能力，保证智能体的权限合规。

2 入门示例：创建基础对话智能体

场景概述

传统开发Agent的方式，是用户在本地使用LangChain或LangGraph等编写智能体脚本，但当需要将其推向生产环境时，往往会卡在繁琐的工程化步骤上：需要手动编写路由、处理请求、构建Docker镜像，并自行寻找服务器部署。

AgentArts提供了一条从代码到生产的完整链路。

本示例作为高代码开发的入门第一课，将暂时剥离复杂的外部工具，带您体验AgentArts最基础但也最核心的“智能体运行时（Runtime）托管能力”。示例中将编写一个纯粹的带有模型的对话智能体，并展示如何通过AgentArts SDK将本地脚本转化为标准的HTTP Web服务，并托管到云端的AgentArts中。

适用场景：

- 存量代码迁移：学习如何将已有的LangChain/LangGraph本地资产无缝接入AgentArts平台。
- 熟悉开发链路：跑通“本地代码开发->SDK标准化封装->云端托管部署”的AgentArts高码开发流程。
- 学习调用MaaS模型：学习如何在智能体中调用华为云MaaS服务提供的模型接口。

环境准备

- 操作系统：**推荐使用Linux Ubuntu 22.04 ARM64**，服务器可访问公网。
本文命令如无特殊说明，均以Ubuntu为例。为保证镜像构建与云端部署的兼容性，建议使用Ubuntu系统。
因此，为了减少部署失败和额外排查成本，建议直接使用Ubuntu 22.04 ARM64制作AgentArts镜像。
若使用CentOS、EulerOS等非Ubuntu系统，请将下命令中的apt替换为yum或dnf，并自行确认Docker 18.06+、Python 3.10+；本文不逐条展开各发行版安装差异。

注意

需使用ARM64系统制作的Agent镜像，使用X86系统制作的镜像在调用智能体运行时调用失败。

可以使用华为云ECS服务购买服务器，购买时需要选择鲲鹏架构（ARM64），系统选择Ubuntu。

云服务器控制台 / 弹性云服务器 / 购买弹性云服务器

<  购买弹性云服务器

快速购买 ① 自定义购买 ②

实例  如何选择ECS实例规格?

规格类型选型 业务场景选型

CPU架构 ③

x86计算

鲲鹏计算

- 安装Python：请确保Python 3.10及以上版本已安装。
大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version检查。如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

非Ubuntu/Debian系统请使用yum或dnf安装Python 3.10+及pip。

- 安装Docker：请确保Docker 18.06及以上版本已安装。如未安装，可以使用如下命令安装：

```
# 查询 Docker 版本
docker --version

# 安装Docker
sudo apt update
sudo apt install docker.io
```

安装完成后，建议立即配置国内镜像加速器。在后续执行agentarts launch构建镜像时，Docker需要拉取python:3.10-slim基础镜像，默认从docker.io官方仓库下载，可能超时失败。提前配置加速器可避免此问题。

执行以下命令创建Docker配置目录：

```
sudo mkdir -p /etc/docker
```

执行以下命令配置国内镜像源（直接复制并回车执行）：

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors":[
    "https://docker.m.daocloud.net",
    "https://dockerproxy.net",
    "https://mirror.baidubce.com"
  ]
}
EOF
```

依次执行以下命令重启Docker服务，使配置生效：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

📖 说明

华为云SWR基础版不支持OCI镜像格式，如果您使用的是Docker 27及以上版本，并且需要处理OCI镜像，可以通过设置环境变量来关闭OCI支持。

```
export DOCKER_BUILDKIT=0
# 或者
export BUILDKIT_USE_OCI_MEDIA_TYPES=0
```

- 执行以下命令安装SDK（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
# 安装依赖包
apt install python3.12-venv

# 创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate

# 安装sdk
pip install agentarts-sdk
```

若配置镜像加速器，后续在执行agentarts launch命令时，仍无法拉取python:3.10-slim，可参考本文结尾的“常见问题”中的“直接修改Dockerfile基础镜像地址”。

📖 说明

- 执行pip install agentarts-sdk命令下载缓慢、超时可以更改为使用如下命令
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
- 如果系统缺少python3-venv包，导致无法创建虚拟环境，请按照命令回显提示安装python3-venv包。

```
root@ecs-6732:~# python3 -m venv myenv
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

apt install python3.12-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

Failing command: /root/myenv/bin/python3
```

- 执行以下命令配置华为云凭证，获取华为云凭证请参考[认证鉴权](#)。
export HUAWEICLOUD_SDK_AK="your-access-key"
export HUAWEICLOUD_SDK_SK="your-secret-key"

核心代码实现

步骤一：编写 Agent 核心逻辑

使用LangGraph框架编写智能体的核心对话逻辑，并配置模型访问密钥。

步骤1 执行如下命令安装langchain、langgraph、langchain-openai。

```
pip install -U langchain langgraph langchain-openai
```

步骤2 在本地创建agent.py文件，编辑代码如下。

示例代码中创建了一个对话智能体，并对接华为云MaaS服务提供的模型。

```
import os
from typing import List, Dict, Any, TypedDict, Annotated
from dotenv import load_dotenv

from langgraph.graph import StateGraph, END
from langgraph.graph.message import add_messages
```

```
from langchain_core.messages import BaseMessage, HumanMessage, SystemMessage
from langchain_openai import ChatOpenAI

# 从 .env 文件加载环境变量
load_dotenv()

# =====
# 第一部分：状态定义与配置
# =====
class AgentState(TypedDict):
    """
    Agent 状态类型定义

    这是 LangGraph 运行时的核心数据结构。
    - messages: 使用 add_messages 确保每次对话的新消息会被安全追加，而不是覆盖。
    """
    messages: Annotated[List[BaseMessage], add_messages]

# =====
# 第二部分：Agent 主类与 workflow 构建
# =====
class BasicAgent:
    """
    基础对话智能体类

    封装了 LLM 的初始化、LangGraph 工作流的构建与执行。
    """
    def __init__(self):
        # 强校验：没有 API Key 直接报错阻断，避免产生难以排查的网络错误
        api_key = os.getenv("MODEL_API_KEY")
        if not api_key:
            raise ValueError("[错误] 未找到 MODEL_API_KEY! 请在 .env 文件中进行配置。")

        # 实例化大模型客户端 (对接华为云MaaS服务提供的模型)
        self.llm = ChatOpenAI(
            model=os.getenv("MODEL_NAME", "deepseek-v4-flash"),
            base_url=os.getenv("MODEL_URL", "https://api.modelarts-maas.com/openai/v1"),
            api_key=api_key,
            temperature=0.7,
        )
        self.graph = self._build_graph()
        self.system_prompt = "你是一个乐于助人的 AI 助手，请简明扼要地回答问题。"

    def _build_graph(self):
        """
        创建 Agent 工作流图

        使用 LangGraph 的 StateGraph 构建极简工作流：
        1. 添加 agent 节点（调用 LLM 生成回复）
        2. 设置起点为 agent 节点
        3. 设置终点（执行完 agent 节点后直接结束，无工具循环）

        Returns:
            编译后的 LangGraph 图
        """
        def model_node(state: AgentState) -> AgentState:
            response = self.llm.invoke(state["messages"])
            return {"messages": [response]}

        workflow = StateGraph(AgentState)
        workflow.add_node("agent", model_node)
        workflow.set_entry_point("agent")
        workflow.add_edge("agent", END)
        return workflow.compile()

    def run(self, user_input: str) -> str:
        """
        执行 Agent 处理用户输入
        """
```

```
"""
messages = [
    SystemMessage(content=self.system_prompt),
    HumanMessage(content=user_input)
]
result = self.graph.invoke({"messages": messages})
return result["messages"][-1].content
```

步骤3 创建.env环境变量配置文件。

```
# 必填：您的华为云 MaaS API 密钥
MODEL_API_KEY=xxxxxxxxxxxxxxxxxxxxx

# 选填：如果您想使用其他模型，取消下方注释并修改
# MODEL_NAME=deepseek-v4-flash
# MODEL_URL=https://api.modelarts-maas.com/openai/v1
```

获取模型API Key的方法请参考[获取华为云MaaS服务模型API Key](#)。

agent.py文件的示例中，使用华为云MaaS服务提供的deepseek-v4-flash模型，如果使用其他模型注意MODEL_NAME的值，需填写为模型接口中model参数的值，可以参考如下方式获取。注意需要选择OpenAI兼容接口。

图 2-1 获取 OpenAI 兼容接口的 model 值



----结束

步骤二：本地功能验证

在本地直接运行智能体，确认智能体可以正常运行。

步骤1 创建main.py文件，用于本地运行智能体，并进行测试。

```
from agent import BasicAgent

def main():
    print("正在初始化 Agent...")
    try:
        agent = BasicAgent()
    except Exception as e:
        print(e)
        return

    print("=" * 50)
    response = agent.run("你好，请用一句话做个自我介绍。")
    print(f"Agent 回复: \n{response}")
    print("=" * 50)

if __name__ == "__main__":
    main()
```

步骤2 将agent.py、.env、main.py上传至服务器中。

如果您使用MobaXterm、Xshell这类客户端软件，可以直接在软件界面上拖拽上传。

```
(venv) root@ecs-ecdd:~/demo1# ll
total 24
drwxr-xr-x  3 root root 4096 May 22 14:37 ./
drwx----- 17 root root 4096 May 22 14:26 ../
-rw-r--r--  1 root root 3172 May 22 14:36 agent.py
-rw-r--r--  1 root root  301 May 22 14:37 .env
-rw-r--r--  1 root root  387 May 22 14:37 main.py
drwxr-xr-x  5 root root 4096 May 22 14:26 venv/
```

步骤3 执行python main.py命令对智能体进行本地测试。

由于在main.py文件中已经设置好了问题“你好，请用一句话做个自我介绍。”，智能体本地运行时调用大模型进行回复。有正常的回复即表示本地测试成功。

```
(venv) root@ecs-ecdd:~/demo1# python main.py
正在初始化 Agent...
=====
Agent 回复：
你好，我是DeepSeek，一个由深度求索公司开发的AI助手，很高兴为你提供帮助！
=====
(venv) root@ecs-ecdd:~/demo1#
```

----结束

步骤三：SDK 封装与本地接口验证

将智能体封装为符合AgentArts平台规范的HTTP服务，并在本地模拟云端接口调用，确认封装正确后再执行后续的部署上云操作。

步骤1 SDK封装，AgentArts平台通过标准HTTP协议与智能体通信，您需要用AgentArts SDK将Agent逻辑包一层，将其暴露为符合平台规范的/invoke接口，平台才能正确调度和管理您的智能体。

创建app.py文件，并上传至服务器中。该文件会将本地的Agent逻辑封装为符合AgentArts平台规范的Web服务。

```
import os
from typing import Dict, Any
from agentarts.sdk import AgentArtsRuntimeApp, RequestContext
from agent import BasicAgent

# 实例化平台运行时应用
app = AgentArtsRuntimeApp()

try:
    my_agent = BasicAgent()
except Exception as e:
    print(f"Agent 初始化失败: {e}")
    my_agent = None

# =====
# AgentArts 平台标准化接入 (核心)
# =====
@app.entrypoint
async def handler(payload: Dict[str, Any], context: RequestContext = None) -> Dict[str, Any]:
    """
```

```
AgentArts 平台标准 HTTP 暴露入口

Args:
    payload: 请求载荷, 包含用户输入 (如 payload.get("message"))
    context: 请求上下文, 包含 session_id 等平台注入信息
"""

if not my_agent:
    return {"response": "服务未正确初始化, 请检查环境变量。", "status": "error"}

query = payload.get("message", "")
try:
    response = my_agent.run(query)
    return {"response": response, "status": "success"}
except Exception as e:
    return {"response": f"执行出错: {str(e)}", "status": "error"}

if __name__ == "__main__":
    # 平台托管时, 必须读取 AGENT_RUN_PORT 环境变量以匹配容器端口契约
    run_port = int(os.getenv("AGENT_RUN_PORT", 8080))
    app.run(port=run_port)
```

步骤2 接口验证, 在正式推送镜像之前, 先在本地模拟云端HTTP调用环境, 验证Agent的接口封装是否正确、通信是否正常, 可以大幅降低因接口问题导致云端部署后才发现错误的调试成本。

执行python app.py启动http server, 执行以下命令调用验证Agent的HTTP接口是否已经被正确封装且能正常通信。

执行python app.py回显效果如下。

```
(venv) root@ecs-ecdd:~/demo1# python app.py
2026-05-22 16:00:07,783 - agentarts.runtime.app - INFO - Starting AgentArts runtime on 127.0.0.1:8080 (http)
INFO: Started server process [10044]
INFO: waiting for application startup.
INFO: Application startup complete.
INFO: Uvicorn running on http://127.0.0.1:8080 (Press CTRL+C to quit)
```

打开一个新的终端窗口 (保持原窗口运行), 使用curl命令进行测试。测试完成后, 可以使用Ctrl + C停止运行的进程。

```
curl --location --request POST 'http://localhost:8080/invocations' \
--header 'Content-Type: application/json' \
--data-raw '{"message": "你好, 请用一句话做个自我介绍。"}'
```

```
root@ecs-ecdd:~# curl --location --request POST 'http://localhost:8080/invocations' \
--header 'Content-Type: application/json' \
--data-raw '{"message": "你好, 请用一句话做个自我介绍。"}'
{"response": "你好, 我是DeepSeek, 一个热情细腻的AI助手, 随时准备为你提供帮助和解答! ☺", "status": "success"}root@ecs-ecdd:~#
```

----结束

步骤四：云端部署

将本地智能体打包为Docker镜像并一键部署至AgentArts云端运行时。

步骤1 部署智能体运行时。

按上述步骤验证完成后, 接下来将智能体打包并托管到AgentArts云端运行时。平台将自动完成镜像构建、推送与部署, 您只需准备好依赖清单和配置文件即可。

首先准备依赖文件requirements.txt, 内容如下:

```
# =====
# 指定国内 PyPI 镜像源
# =====
```

```
--index-url https://repo.huaweicloud.com/repository/pypi/simple
--trusted-host repo.huaweicloud.com

# =====
# AgentArts LangGraph Agent Demo - 依赖清单
# =====
#
# 本项目基于 AgentArts 平台，使用 LangGraph 框架构建 AI Agent
# 依赖分为两部分：基础框架依赖 + 平台 SDK 依赖
#
# 安装方式:
# pip install -r requirements.txt
#
# =====

# =====
# 第一部分：LangGraph & LangChain 核心框架
# =====
#
# LangGraph: AI Agent 工作流编排框架
# - 状态图定义与执行
# - Checkpoint 持久化机制
# - 条件边与节点路由
langgraph>=0.2.0

#
# LangChain: LLM 应用开发工具链
# - 消息类型定义 (HumanMessage, AIMessage, SystemMessage)
# - 工具系统 (@tool 装饰器)
# - LLM 客户端封装
langchain>=0.3.0
langchain-core>=0.3.0
langchain-community>=0.3.0

# =====
# 第二部分：LLM Provider 支持
# =====
#
# OpenAI 兼容接口
openai>=1.0.0
langchain-openai>=0.1.0

#
# Anthropic (Claude) 支持
anthropic>=0.18.0
langchain-anthropic>=0.1.0

# =====
# 第三部分：HTTP & 网络
# =====
#
requests>=2.31.0
httpx>=0.27.0

# =====
# 第四部分：工具与配置
# =====
#
# 环境变量管理
python-dotenv>=1.0.0

#
# 异步支持
aiofiles>=23.0.0

#
# JSON/YAML 支持
pyyaml>=6.0
```



```
# =====  
# 第五部分：AgentArts 平台 SDK（核心依赖）  
# =====  
agentarts-sdk
```

步骤2 执行如下命令配置智能体。

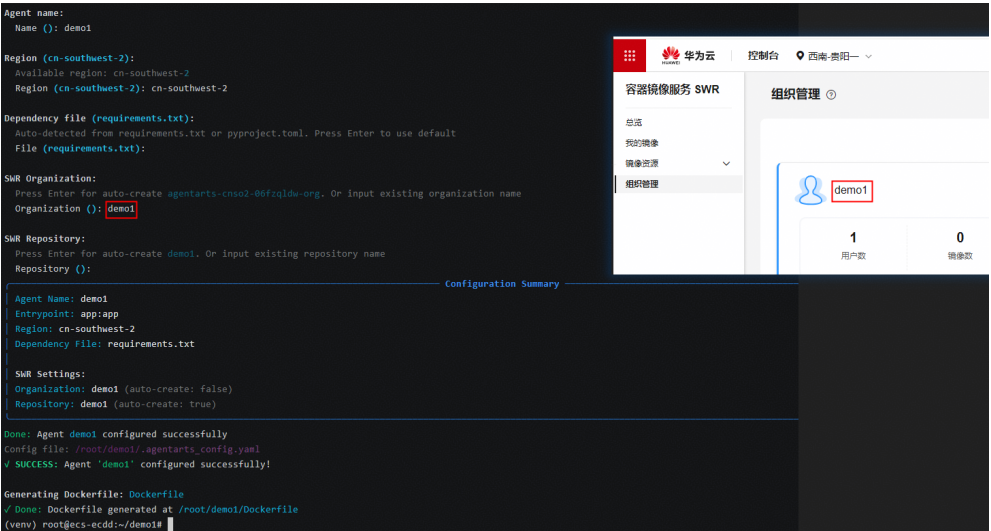
```
agentarts configure --entrypoint app:app
```

Ubuntu系统可忽略此提示，如果您使用的是Huawei Cloud EulerOS 3.0系统构建Agent镜像，在执行完成agentarts configure --entrypoint app:app命令后，建议手动修改下命令生成的Dockerfile文件，再执行后续操作。防止制作好的镜像因缺少依赖包，造成调用Agent时出现沙箱报错。详见[常见问题](#)。

执行后按照操作指引进行配置。

表 2-1 配置参数

参数	说明
Agent name	智能体名称。 以小写字母开头，以小写字母或数字结尾，可以包含小写字母、数字和中划线。
Region	服务部署区域。 cn-southwest-2，仅支持此区域。
Dependency file	依赖文件默认为requirements.txt。
SWR Organization	镜像组织名称（建议自定义的镜像组织名，可以在SWR服务控制台贵阳一region创建）。
SWR Repository	- 不填写（推荐）。 SWR Repository项直接回车留空，工具会自动生成仓库名称，部署agentarts launch时自动创建镜像仓库，避免仓库不存在报错。 - 填写自定义仓库名称。 若需要手动指定仓库，请填写SWR中已预先存在的镜像仓库名称，如未创建，此处请留空不填写。



步骤3 执行命令部署智能体。

agentarts launch

该命令会自动完成以下步骤：

1. 本地构建Docker镜像。
2. 将Docker镜像推送到华为云SWR镜像仓库。
3. 部署到AgentArts运行时托管环境。

[illegible]

步骤4 调用云端Agent进行会话。

agentarts invoke '{"message": "你好，请用一句话做个自我介绍。"}'

```
(venv) root@ecs-ecdd:~/demo1# agentarts invoke '{"message": "你好，请用一句话做个自我介绍。"}'

Invoke Request
Mode: Cloud
Agent: demo1
Session: 92e828af-881b-43e4-a514-9164ac37c565
Endpoint: https://defaultgw-mjswmjggj.cn-southwest-2.huaweicloud-agentarts.com
Auth Type: IAM

Response:
{
  "response": "你好，我是DeepSeek，由深度求索公司创造的AI助手，很高兴为你提供帮助！👉",
  "status": "success"
}
(venv) root@ecs-ecdd:~/demo1#
```

----结束

常见问题

- 使用Huawei Cloud EulerOS 3.0系统制作镜像，invoke命令调用智能体出现沙箱报错。

```
^C(venv) [root@ecs-362eeuros demo02]# agentarts invoke '{"message": "你好，请用一句话做个自我介绍。"}'

Invoke Request
Mode: Cloud
Agent: agentdemo02
Session: 7de74a07-53e2-4f05-b07f-8c3572506a1b
Endpoint: https://defaultgw-mjswmjggj.cn-southwest-2.huaweicloud-agentarts.com
Auth Type: IAM

2026-09-22 18:58:23.661 - agentarts.sdk.service.runtime_client - ERROR - invoke_agent failed: status=500, data={'code': 500, 'data': None, 'message': 'Failed to create sandbox: Timeout waiting for registration (c104319f-d2fd-4df7-836b-381916f79f17)'} error=Failed to create sandbox: Timeout waiting for registration (c104319f-d2fd-4df7-836b-381916f79f17)
invoke_agent failed: status=500, data={'code': 500, 'data': None, 'message': 'Failed to create sandbox: Timeout waiting for registration (c104319f-d2fd-4df7-836b-381916f79f17)'}
error=Failed to create sandbox: Timeout waiting for registration (c104319f-d2fd-4df7-836b-381916f79f17)
> ERROR: invoke_agent failed (HTTP 500): Failed to create sandbox: Timeout waiting for registration (c104319f-d2fd-4df7-836b-381916f79f17)

(venv) [root@ecs-362eeuros demo02]# agentarts invoke '{"message": "你好，请用一句话做个自我介绍。"}'

Invoke Request
Mode: Cloud
Agent: agentdemo02
Session: 732de07e-f62d-454f-a44b-d26d88a1940b
Endpoint: https://defaultgw-mjswmjggj.cn-southwest-2.huaweicloud-agentarts.com
Auth Type: IAM

2026-09-22 19:01:32.949 - agentarts.sdk.service.runtime_client - ERROR - invoke_agent failed: status=403, data={'code': 403, 'data': None, 'message': 'can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\\code\\":\\NA.90001\\",\\message\\":\\NA.90001\\"} warmup health check overtime: warmup takes more than 4m0s\\"}'} error=can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\\code\\":\\NA.90001\\",\\message\\":\\NA.90001\\"} warmup health check overtime: warmup takes more than 4m0s\\"}'
invoke_agent failed: status=403, data={'code': 403, 'data': None, 'message': 'can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\\code\\":\\NA.90001\\",\\message\\":\\NA.90001\\"} warmup health check overtime: warmup takes more than 4m0s\\"}'} error=can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\\code\\":\\NA.90001\\",\\message\\":\\NA.90001\\"} warmup health check overtime: warmup takes more than 4m0s\\"}'
> ERROR: invoke_agent failed (HTTP 403): can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\\code\\":\\NA.90001\\",\\message\\":\\NA.90001\\"} warmup health check overtime: warmup takes more than 4m0s\\"}'
```

处理措施：

- 检查linux服务器是否值ARM架构，如果是X86架构，建议改为ARM架构服务器重新制作镜像。
- 使用EulerOS 3.0系统制作镜像时。在执行完成agentarts configure --entrypoint app:app命令后。需要手动修改下Dockerfile文件，再执行后续的agentarts launch命令，详细操作如下。

- 备份下原有的Dockerfile文件。
cp Dockerfile Dockerfile.bak
- 执行vi Dockerfile命令修改Dockerfile文件，将下面的内容替换进去。

此Dockerfile可以避免非root用户读取site-packages时的权限问题；同时在安装依赖后增加了agentarts sdk的导入校验，安装失败会在构建阶段直接暴露，不会带着异常镜像进入部署流程。

```
# AgentArts Dockerfile
# Generated by AgentArts CLI
# Modified for EulerOS: removed non-root user, added pip source and SDK validation

FROM python:3.10-slim

# Set container environment marker
ENV DOCKER_CONTAINER=true
```

```
ENV HUAWEICLOUD_SDK_REGION=cn-southwest-2

WORKDIR /app

COPY requirements.txt .

# 使用华为云 PyPI 镜像源, 安装后立即验证 agentarts.sdk 可导入
RUN pip install --no-cache-dir -r requirements.txt \
    -i https://repo.huaweicloud.com/repository/pypi/simple \
    --trusted-host repo.huaweicloud.com \
    && python -c "from agentarts.sdk import AgentArtsRuntimeApp; print('SDK OK')"

# 显式复制必要文件, 避免将 venv 等无关文件带入镜像
COPY agent.py app.py main.py .env ./

EXPOSE 8080

CMD ["python", "-m", "app"]
```

- iii. 执行 `vi .dockerignore` 命令, 创建 `.dockerignore` 文件, 并写入以下内容。防止 `venv` 等被 `COPY` .. 打包污染镜像。

```
cat > .dockerignore <<'EOF'
venv/
__pycache__/_
*.pyc
.git/
Dockerfile
Dockerfile.bak
EOF
```

- c. 执行 `agentarts launch` 命令, 部署智能体到运行时。

- 执行 `agentarts launch` 命令时出现 AK/SK 认证报错

执行以下命令配置华为云凭证, 获取华为云凭证请参考[认证鉴权](#)。

```
export HUAWEICLOUD_SDK_AK="your-access-key"
export HUAWEICLOUD_SDK_SK="your-secret-key"
```

```
2026-05-22 17:24:46,221 - agentarts.sdk.metadata_provider - WARNING - Get metadata credentials failed with status: 404
Get metadata: credentials failed with status: 404
Organization 'demo1' query failed: Authentication failed: Could not find valid credentials. Please configure one of the following:
1. AK/SK Authentication: Set HUAWEICLOUD_SDK_AK, HUAWEICLOUD_SDK_SK
2. OIDC Token: Set HUAWEICLOUD_SDK_IDP_ID, HUAWEICLOUD_SDK_ID_TOKEN_FILE, and HUAWEICLOUD_SDK_PROJECT_ID
3. Metadata: Running on the AgentArts runtime
x ERROR: Organization 'demo1' not found
Set organization_auto_create: true in config to auto-create
(venv) root@ecs-ecdd:~/demo1# export HUAWEICLOUD_SDK_AK=HPUA
(venv) root@ecs-ecdd:~/demo1# export HUAWEICLOUD_SDK_SK=Vadm
(venv) root@ecs-ecdd:~/demo1#
```

- 执行 `agentarts launch` 命令出现运行时 Runtime 名称错误

智能体的名称需要以小写字母开头, 以小写字母或数字结尾, 可以包含小写字母、数字和中划线, 长度为 2-48 个字符。

请重新执行 `agentarts configure --entrypoint app:app` 命令进行配置。

```
(creating AgentArts runtime: demo_02...)
2026-05-23 18:07:07,852 - agentarts.sdk.service.runtime_client - ERROR - create_agent failed: status=400, data={'error_code': 'AgentArts.03005400', 'error_msg': 'Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48'}
error-Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48
create_agent failed: status=400, data={'error_code': 'AgentArts.03005400', 'error_msg': 'Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48'}, error-Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48
x ERROR: Failed to create runtime: create_agent failed (HTTP 400): Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48
(venv) root@ecs-ecdd:~/demo_02# agentarts configure --entrypoint app:app
```

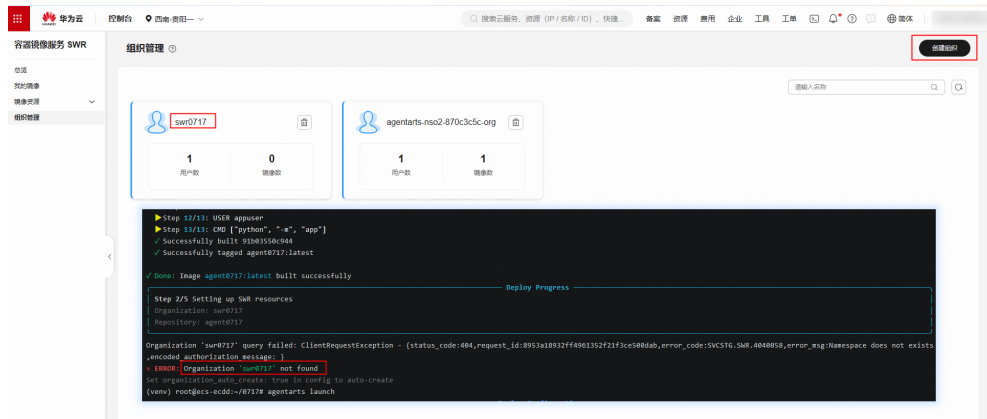
- 执行命令, 出现 `SWR.4040010,error_msg:Repository does not exists,ERROR: Organization 'xxx' not found` 报错。

问题现象及原因

SWR 服务镜像组织不存在, 出现该报错的原因通常是在执行 `agentarts configure --entrypoint app:app` 命令时, 手动指定了一个 SWR 镜像组织名称, 但是未在 SWR 控制台创建同名组织导致。

解决方案

登录**SWR服务控制台**（贵阳一区域），在“组织管理”中手动创建一个与agentarts configure --entrypoint app:app命令中同名的镜像组织。



- 执行agentarts launch命令，在执行requirements.txt步骤中出现Read timed out ... files.pythonhosted.org报错。

问题现象及原因

Docker镜像在构建过程中，尝试从官方PyPI服务器下载在requirements.txt里写的那些Python包，由于网络环境原因造成超时。

```
(venv) root@ecs-3985arm1:~/my-agent# agentarts launch
Deploy Configuration
Agent: my-agent
Mode: cloud
Region: cn-southwest-2

Step 1/5 Building Docker image
Image: my-agent:latest

Building Docker image: my-agent:latest
+ Step 1/13: FROM python:3.10-slim
+ Step 2/13: ENV DOCKER_CONTAINER=true
+ Step 3/13: ENV HUAWICLOUD_SDK_REGION=cn-southwest-2
+ Step 4/13: RUN groupadd -g 1000 appuser && useradd -u 1000 -g 1000 -m -s /bin/bash appuser
+ Step 5/13: WORKDIR /app
+ Step 6/13: RUN chown appuser:appuser /app
+ Step 7/13: COPY requirements.txt .
+ Step 8/13: RUN pip install --no-cache-dir -r requirements.txt
x [9]ERROR: Exception:
x File "/usr/local/lib/python3.10/site-packages/pip/_vendor/urllib3/response.py", line 438, in _error_catcher
x TimeoutError: The read operation timed out
x with self._error_catcher():
x File "/usr/local/lib/python3.10/site-packages/pip/_vendor/urllib3/response.py", line 443, in _error_catcher
x raise ReadTimeoutError(self._pool, None, "Read timed out.")
x pip._vendor.urllib3.exceptions.ReadTimeoutError: HTTPConnectionPool(host='files.pythonhosted.org', port=443): Read timed out.

Error building image:
Error details:
[9]ERROR: Exception:
File "/usr/local/lib/python3.10/site-packages/pip/_vendor/urllib3/response.py", line 438, in _error_catcher
TimeoutError: The read operation timed out
with self._error_catcher():
File "/usr/local/lib/python3.10/site-packages/pip/_vendor/urllib3/response.py", line 443, in _error_catcher
raise ReadTimeoutError(self._pool, None, "Read timed out.")
pip._vendor.urllib3.exceptions.ReadTimeoutError: HTTPConnectionPool(host='files.pythonhosted.org', port=443): Read timed out.
```

解决方案

执行以下命令让其在安装依赖时使用华为云官方的Python镜像加速器。

```
sed -i 's|pip install --no-cache-dir -r requirements.txt|pip install --no-cache-dir -r requirements.txt -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com|g' Dockerfile
```

- 执行agentarts launch命令，出现failed to resolve reference "docker.io/library/python:3.10-slim"类型报错。

问题现象及原因

尝试连接Docker官方镜像仓库（docker.io）下载python:3.10-slim基础环境时，网络连接超时（网络连接被阻断或者限速）。

```
(venv) root@ecs-3985arm:~/my-agent# agentarts launch
Deploy Configuration
Agent: my-agent
Mode: cloud
Region: cn-southwest-2

Deploy Progress
Step 1/5 Building Docker image
Image: my-agent:latest

Building Docker image: my-agent:latest
  - Step 1/13: FROM python:3.10-slim

Error building image:
Last 15 lines of output:
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/

Sending build context to Docker daemon 13.82kB

Step 1/13 : FROM python:3.10-slim
failed to resolve reference "docker.io/library/python:3.10-slim": failed to do request: Head
'https://registry-1.docker.io/v2/library/python/manifests/3.10-slim': dial tcp 47.88.58.234:443: connect: connection refused
```

解决方案一：配置Docker国内镜像加速器（推荐）

执行以下命令在系统配置目录下创建一个名为docker的文件夹。

```
sudo mkdir -p /etc/docker
```

执行以下命令（直接复制下面这段代码并回车执行），更换Docker镜像源，从国内镜像站寻找python:3.10-slim。

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors":[
    "https://docker.m.daocloud.net",
    "https://dockerproxy.net",
    "https://mirror.baidubce.com"
  ]
}
EOF
```

依次执行下面两条命令，重启Docker服务：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

Docker配置完成后，再次执行agentarts launch命令进行Agent部署。

解决方案二：直接修改Dockerfile基础镜像地址

- 在项目根目录下找到Dockerfile（由agentarts launch自动生成或已存在）。
- 编辑Dockerfile，将FROM python:3.10-slim替换为国内镜像地址，例如：

```
sed -i 's|FROM python:3.10-slim|FROM docker.m.daocloud.io/library/python:3.10-slim|g'
Dockerfile
```

或手动修改为：

```
FROM docker.m.daocloud.io/library/python:3.10-slim
```

保存后重新执行agentarts launch命令。

- 执行agentarts launch命令，出现Organization 'agentarts-org1' already exists、Failed to create/get organization 'agentarts-org1'报错。

SWR服务镜像名称已被占用，请重新更换名称，即更换agentarts_config.yaml文件中swr_config配置中organization参数的值、artifact_source参数中的值（在yaml文件中，该值共有2处需要替换）。

可依次执行以下命令进行替换，也可以直接手动执行

nano .agentarts_config.yaml命令进入到yaml文件中进行修改。

```
# 将.agentarts_config.yaml文件中原有的 agentarts-org1 组织名替换为一个不容易重名的名字，比如
org_agent_6731
sed -i 's/organization: agentarts-org1/organization: org_agent_6731/g' .agentarts_config.yaml
```

```
# 修正 artifact_source 里的 URL 中的组织名，这里以 org_agent_6731 为例
sed -i 's|url: swr.cn-southwest-2.myhuaweicloud.com/agentarts-org1/agent_my-agent:latest|url: swr.cn-southwest-2.myhuaweicloud.com/org_agent_6731/agent_my-agent:latest|g' .agentarts_config.yaml
```

- 执行agentarts launch命令，出现SVCSCGTG.SWR.4030017、403 Insufficient permissions报错。

如果使用IAM账户执行agentarts launch命令产生该报错，表示IAM用户没有SWR服务中创建组织或管理资源的权限。请联系主账号授予SWRFullAccessPolicy权限（在IAM新版控制台中授权，IAM控制台“总览”页面右上角处可切换至新版）。

- 执行agentarts invoke命令与智能体进行会话出现invoke_agent failed、Failed to create sandbox且平台运行时日志出现python: exec format error报错。

问题现象及原因

当前AgentArts服务部署于贵阳一区域，在创建智能体镜像时需要使用ARM64机器打镜像，使用X86机器打镜像会出现此报错。

```
2026-07-18 17:21:07.911 - agentarts.sdk.service.runtime.client - ERROR - invoke_agent failed: status=504, data=None error=HTTP 504
invoke_agent failed: status=504, data=None error=HTTP 504
ERROR: invoke_agent failed (HTTP 504): HTTP 504
(venv) root@iZ0j17uei15gdf15hqsiz:~/aa-sdk# agentarts invoke ('message': '你好，请用一句话做个自我介绍。')
Invoke Request
Mode: Cloud
Agent: gpttest
Session: 2a0a348c-1647-4562-a964-b397aa26712d
Endpoint: https://defaultgw-mmydcnjqm.cn-southwest-2.huaweicloud-agentarts.com
Auth Type: IAM
2026-07-18 17:25:36.226 - agentarts.sdk.service.runtime.client - ERROR - invoke_agent failed: status=403, data={'code': 403, 'data': None, 'message': 'Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\'code\': \'NA.90001\', \'message\': \'[NA.90001] warmup health check overtime: warmup takes more than 4m0s\'}" (40a4766e-12e8-4870-825c-c035058ca0f1)'} error=Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\'code\': \'NA.90001\', \'message\': \'[NA.90001] warmup health check overtime: warmup takes more than 4m0s\'}" (40a4766e-12e8-4870-825c-c035058ca0f1)
invoke_agent failed: status=403, data={'code': 403, 'data': None, 'message': 'Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\'code\': \'NA.90001\', \'message\': \'[NA.90001] warmup health check overtime: warmup takes more than 4m0s\'}" (40a4766e-12e8-4870-825c-c035058ca0f1)'}
error=Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\'code\': \'NA.90001\', \'message\': \'[NA.90001] warmup health check overtime: warmup takes more than 4m0s\'}" (40a4766e-12e8-4870-825c-c035058ca0f1)
ERROR: invoke_agent failed (HTTP 403): Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: "{\'code\': \'NA.90001\', \'message\': \'[NA.90001] warmup health check overtime: warmup takes more than 4m0s\'}" (40a4766e-12e8-4870-825c-c035058ca0f1)
(venv) root@iZ0j17uei15gdf15hqsiz:~/aa-sdk#
```

基本信息

版本

日志

标签

基本信息

日志箱

AgentArts

日志流

AgentArts

日志明细

近1小时

选择属性筛选，或输入关键字搜索

时间	级别	内容
2026/07/18 17:35:24 GMT+08:00	INFO	exec /usr/local/bin/python: exec format error

解决方案

请更换为ARM64机器制作智能体镜像。

- 执行agentarts launch命令时出现ERROR: Repository 'xxx' not found报错。

问题现象及原因

执行agentarts launch部署智能体，构建Docker镜像成功，但推送SWR镜像仓库阶段报错：ERROR: Repository 'xxx/xxx' not found。

原因在于通过agentarts configure --entrypoint app:app做交互式配置时，SWR Repository输入的镜像仓库名称在SWR中不存在。

```
Building Docker image: test1111:latest
Step 1/13: FROM python:3.10-slim
Step 2/13: ENV OCKER_CONTAINER=true
Step 3/13: ENV HOME=/root/.ssh:~:~:~:cn-southwest-2
Step 4/13: RUN groupadd -g 30000 appuser 88 useradd -s /bin/bash appuser
Step 5/13: WORKDIR /app
Step 6/13: RUN chown appuser:appuser /app
Step 7/13: COPY requirements.txt .
Step 8/13: RUN pip install --no-cache-dir -r requirements.txt
Step 9/13: CMD ["python", "-m", "app"]
Step 10/13: RUN chown -R appuser:appuser /app
Step 11/13: EXPOSE 8080
Step 12/13: USER appuser
Step 13/13: CMD ["python", "-m", "app"]
Successfully built 30a1111c1c1e
Successfully tagged test1111:latest
Done: Image test1111:latest built successfully
Deploy Progress
Step 1/5: Setting up SWR resources
Organization: test1111
Repository: test1111
SUCCESS: Using existing organization test1111
Repository "test1111/test1111" query failed: ClientRequestException - (status_code=404, request_id=86e4d9d5b0b70e7bf22e4026d07bac2, error_code=SWR376.SW-400006, error_msg: Repository does not exist, encoded_authorization_message: )
ERROR: Repository 'test1111/test1111' not found
```

解决方案

使用nano .agentarts_config.yaml命令进入config文件，将organization_auto_create的值修改为true即可。

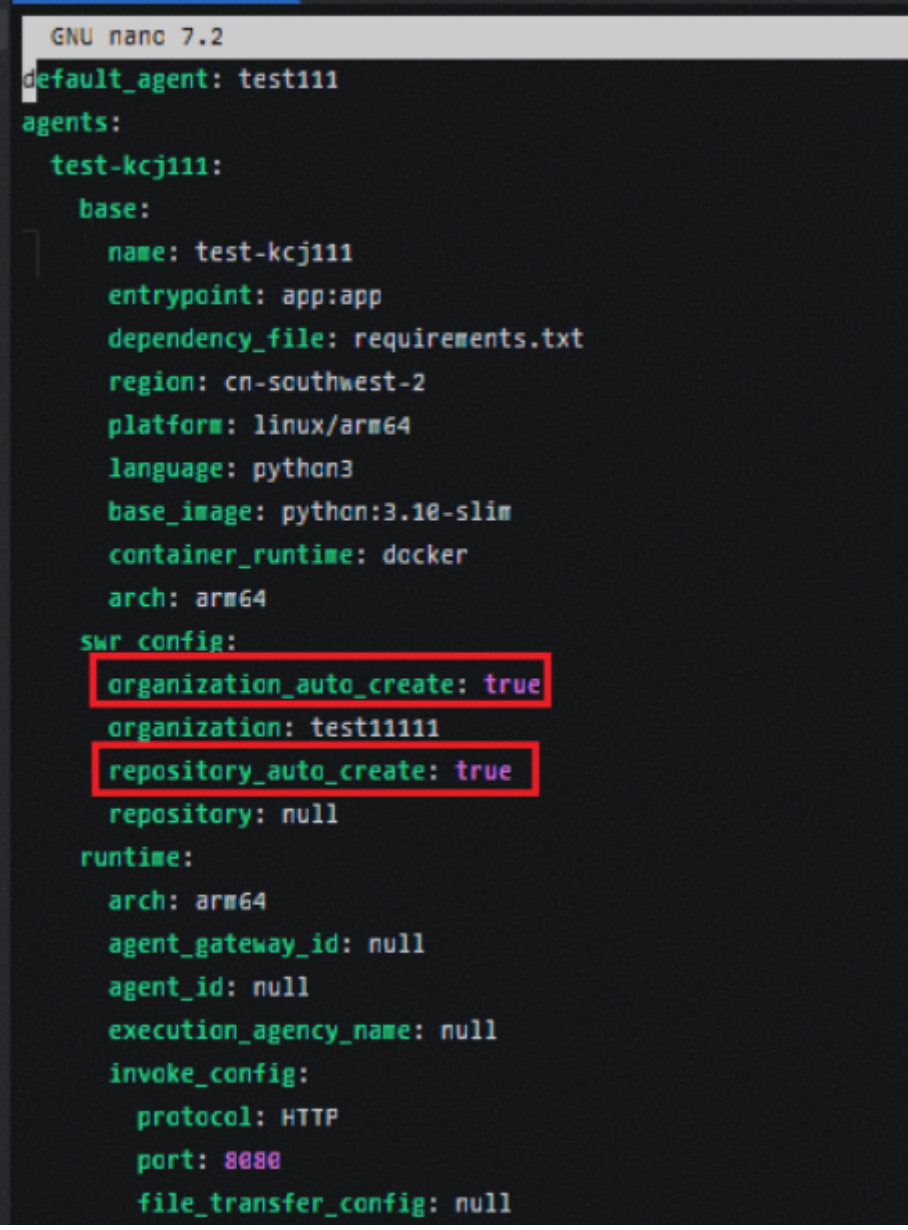
a. 编辑配置文件

nano .agentarts_config.yaml

- b. 在swr_config配置块下，设置repository_auto_create: true，同时填写自定义仓库名称：

```
swr_config:  
  repository: "自定义仓库名"  
  repository_auto_create: true
```

- c. 保存退出，重新执行agentarts launch。



```
GNU nano 7.2  
default_agent: test111  
agents:  
  test-kcj111:  
    base:  
      name: test-kcj111  
      entrypoint: app:app  
      dependency_file: requirements.txt  
      region: cn-southwest-2  
      platform: linux/arm64  
      language: python3  
      base_image: python:3.10-slim  
      container_runtime: docker  
      arch: arm64  
    swr_config:  
      organization_auto_create: true  
      organization: test11111  
      repository_auto_create: true  
      repository: null  
    runtime:  
      arch: arm64  
      agent_gateway_id: null  
      agent_id: null  
      execution_agency_name: null  
      invoke_config:  
        protocol: HTTP  
        port: 8080  
        file_transfer_config: null
```


3 进阶示例：构建全能出行助手（集成模型/记忆/代码解释器/网关/高德 mcp）

场景概述

在[基础示例](#)中，我们已创建一个简易对话智能体，并将其托管部署至云上AgentArts环境。但在真实生产环境下，企业级智能体需要具备持久化记忆、安全沙箱执行、外部系统无缝对接等能力。

本示例将基于LangGraph框架构建一个出行规划智能体，核心实现agent.py的TravelAgent类。该类封装LangGraph ReAct工作流（agent节点完成LLM推理，tools节点负责工具执行）、MCP工具动态发现、云端记忆持久化、代码沙箱调用等特性；同时集成AgentArts平台的Gateway工具调用、代码沙箱执行、云端记忆持久化三大核心能力，部署于AgentArts运行时，对外提供多轮对话式出行规划服务。

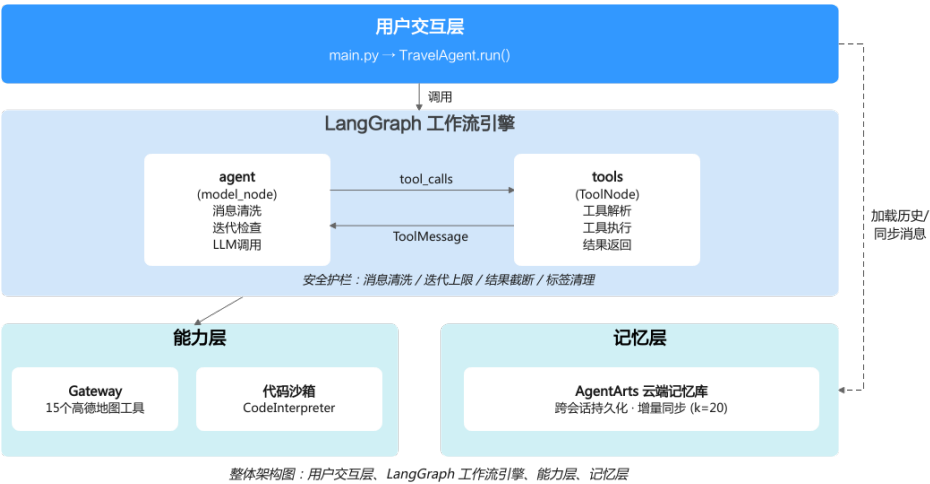
用户输入自然语言指令，例如“查杭州天气”、“帮我算两天行程预算”，Agent会自主选择工具与调用顺序，最终输出结构化回复。

本示例将构建“全能出行助手”智能体，重点演示如下能力：

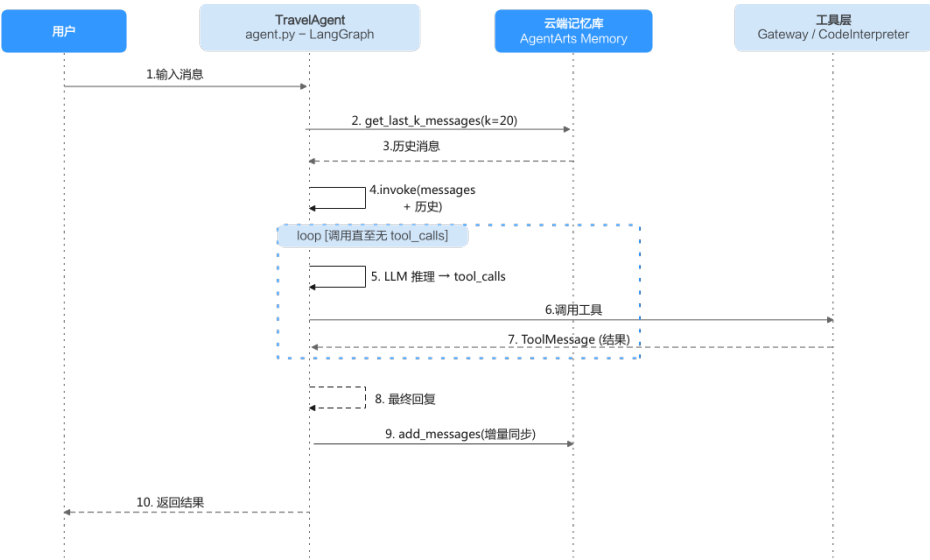
- 记忆库（Memory）：对接AgentArts云端记忆库组件，提供长短期记忆能力，完成跨会话上下文持久化。用户开启新会话时，可加载历史对话记录，接续完成出行规划。
- 网关（Gateway）：作为Agent调用外部工具的统一入口，依托标准MCP协议转换能力对接外部工具；服务启动时动态拉取全部工具定义。示例接入高德地图15个MCP工具，支持天气查询、地理编码、多模式路径规划（驾车 / 步行 / 骑行 / 公交）、周边搜索、距离测量等功能。
- 代码解释器（CodeInterpreter）：依托平台完全隔离的容器环境，创建安全沙箱会话，执行LLM生成的Python代码并返回运算结果。在出行规划场景下，可完成预算核算（门票 + 住宿 + 餐饮 + 交通汇总）、多套方案费用对比、行程增量调整等工作。
- 身份认证（Identity）：与网关协同，完成工具链路的标准化鉴权，保障外部工具调用安全。

架构图与工作流程

整体架构图



工作流程



环境准备

- 操作系统：推荐使用Linux Ubuntu 22.04 ARM64**，服务器可访问公网。
本文命令如无特殊说明，均以Ubuntu为例。为保证镜像构建与云端部署的兼容性，建议使用Ubuntu系统。
因此，为了减少部署失败和额外排查成本，建议直接使用Ubuntu 22.04 ARM64制作AgentArts镜像。
若使用CentOS、EulerOS等非Ubuntu系统，请将下命令中的apt替换为yum或dnf，并自行确认Docker 18.06+、Python 3.10+；本文不逐条展开各发行版安装差异。

⚠ 注意

需使用ARM64系统制作的Agent镜像，使用X86系统制作的镜像在调用智能体运行时调用失败。

可以使用华为云ECS服务购买服务器，购买时需要选择鲲鹏架构（ARM64），系统选择Ubuntu。

云服务器控制台 / 弹性云服务器 / 购买弹性云服务器

<  购买弹性云服务器

快速购买 ① 自定义购买 ②

实例  如何选择ECS实例规格?

规格类型选型 业务场景选型

CPU架构 ③

x86计算

鲲鹏计算

- 安装Python：请确保Python 3.10及以上版本已安装。
大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version检查。如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

- 安装Docker：请确保Docker 18.06及以上版本已安装。如未安装，可以使用如下命令安装：

```
# 查询 Docker 版本
docker --version

# 安装Docker
sudo apt update
sudo apt install docker.io
```

安装完成后，建议立即配置国内镜像加速器。在后续执行agentarts launch构建镜像时，Docker需要拉取python:3.10-slim基础镜像，默认从docker.io官方仓库下载，可能超时失败。提前配置加速器可避免此问题。

执行以下命令创建Docker配置目录：

```
sudo mkdir -p /etc/docker
```

执行以下命令配置国内镜像源（直接复制并回车执行）：

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors":[
    "https://docker.m.daocloud.net",
    "https://dockerproxy.net",
    "https://mirror.baidubce.com"
  ]
}
EOF
```

依次执行以下命令重启Docker服务，使配置生效：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

说明

华为云SWR基础版不支持OCI镜像格式，如果您使用的是Docker 27及以上版本，并且需要处理OCI镜像，可以通过设置环境变量来关闭OCI支持。

```
export DOCKER_BUILDKIT=0
# 或者
export BUILDKIT_USE_OCI_MEDIA_TYPES=0
```

• 安装SDK。

a. 在Linux系统中创建项目目录并进入。

```
mkdir ./test2
cd ./test2
```

b. 执行以下命令安装SDK（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
# 安装依赖包
apt install python3.12-venv

# 创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate

# 安装sdk
pip install agentarts-sdk
```

说明

- 执行pip install agentarts-sdk命令下载缓慢、超时可以更改为使用如下命令
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
- 如果系统缺少python3-venv包，导致无法创建虚拟环境，请按照命令回显提示安装python3-venv包。

```
root@ecs-6732:~# python3 -m venv myenv
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

apt install python3.12-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

Failing command: /root/myenv/bin/python3
```

- 执行以下命令配置华为云凭证，获取华为云凭证请参考[认证鉴权](#)。

```
export HUAWEICLOUD_SDK_AK="your-access-key"
export HUAWEICLOUD_SDK_SK="your-secret-key"
```

智能体开发全流程

本示例涉及的代码样例可通过[代码示例](#)全量下载，也可直接从帮助文档中复制使用。

步骤一：平台组件创建与配置

在AgentArts平台依次创建记忆库、代码解释器和网关，并获取后续代码接入所需的各项配置信息。

步骤1 在AgentArts平台创建记忆库。

1. 登录[AgentArts智能体平台](#)。
2. 在左侧导航栏选择“托管与运行 > 记忆库”，进入“记忆库”界面。
3. 单击“创建记忆库”。填写记忆库名称、描述；勾选“公网访问”；选择“API Key认证”；数据加密配置使用默认设置；勾选全部的“长期记忆提取策略”，其余参数使用默认配置。

图 3-1 创建记忆库

智果（AgentArts）智能体平台 / 记忆库 / 创建记忆库

创建记忆库 ①

基本信息

记忆库名称

memory-muhzgk99

有效字符为 a-z、A-Z、0-9 和 (中划线)，该名称最多可包含 48 个字符。字符数：15/48。

描述 (可选)

请输入

0/1,000

网络配置

☐ 私网访问

开启后，私有网络和子网暂时不支持修改。使用私网访问出站网络，当前账号需具备VPC EIP相关权限 [查看所需权限](#) [查看当前权限](#)

☒ 公网访问

Memory开启公网访问后，存在一定安全风险，建议仅用于开发测试场景。开启后，暂不支持关闭。

权限与访问控制

入站身份认证

IAM 认证

将华为云 IAM 身份配置为入站身份认证，以验证调用者的身份。

API Key 认证

将预设的密钥对（键和值）配置为入站身份认证，以验证请求中携带的密钥合法性。

图 3-2 配置记忆策略

短期记忆

短期记忆（原始事件）过期

为智能体的交互设置短期（原始事件）记忆持续时间。超过指定持续时间的事件将过期且不再存储。介于7~365天之间。

-

90

+

天

长期记忆提取模型配置

模型配置

系统默认

长期记忆提取策略

☒ 总结 Summary

汇总和压缩会话中的主题、任务、历史决策记录等关键信息。

☒ 语义记忆 Semantic Memory

从会话记录中识别和提取事实信息和上下文知识的关键片段，包括但不限于事实性知识、概念等。

☒ 用户偏好 User Preference

从原始会话中提取用户行为模式、关键偏好和选择等信息，为每个用户建立一个持久的、动态更新的用户偏好。

☒ 情景记忆 Episodic Memory

从对话或经历中提取具有时空序列的具体事件、个人体验和场景细节。

☒ 程序性记忆 Procedural Memory

从会话记录中提取操作流程、行动步骤和关键提示等信息，形成可复用的任务执行指南。

4. 记忆库创建完成后，单击记忆库名称，进入“基本信息”界面，获取记忆库ID。

文档版本 01 (2026-09-24)

版权所有 © 华为云计算技术有限公司

23

图 3-3 获取记忆库 ID



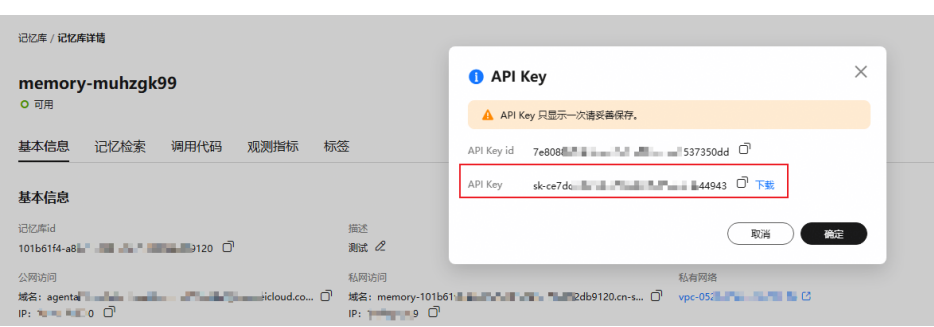
5. 在“API Key”处单击“更新”。

图 3-4 更新 API Key



6. 弹框中单击“确定”后，复制或下载API Key。

图 3-5 复制或下载 API Key



步骤2 在AgentArts平台创建代码解释器。

- 在AgentArts平台左侧导航栏中选择“托管与运行 > 代码解释器”，单击“创建代码解释器”。
- 填写代码解释器相关配置。参考下表进行配置。

表 3-1 代码解释器配置说明

参数	说明
名称	可自定义。

参数	说明
描述	可自定义。
委托（可选）	（可选）授予的代理权限或代理功能，允许代表智能体与外部系统进行通信和交互。 可不填写，如需手动创建，请单击“创建委托”并参考如下配置创建。 - 委托名称：自定义。 - 信任主体类型：选择“云服务”。 - 云服务：搜索 service.WorkloadSandboxMetadata。 - 其余参数使用默认值。 图 3-6 创建委托 
入站身份认证	选择“API Key认证”。
API Key名称	可自定义。
日志记录	开启。
出网网络配置	选择“公网访问”。

图 3-7 创建代码解释器

基本信息

名称

codeinterpreter-nq6ol0h

以小写字母开头，小写字母或数字结尾，中间可包含数字、小写字母、中划线。字符数：24/40。

描述 (可选)

请输入

0/4,096

权限与身份认证

委托 (可选)

请选择

创建委托

入站身份认证

IAM 认证

将华为云 IAM 身份配置为入站身份认证，以验证调用者的身份。

API Key 认证

将预设的密钥对（键和值）配置为入站身份认证，以验证请求中携带的密钥合法性。

身份认证凭证在云资源管理服务（DEW）托管，托管按需计费。[了解 DEW 计费详情](#)

API Key 名称

APIKey-e1ao72dn

可观测配置

日志记录

已开启

启用日志功能后，工具运行过程产生的日志会上报云日志服务(LTS)，日志管理费用由按需收取。[了解 LTS 计费详情](#)

高级配置

出网网络配置

公网访问

私网访问

标签 (可选)

若您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签。建议在TMS中[创建预定义标签](#)

3. 配置完成后，单击“立即创建”创建代码解释器。

在代码解释器列表中，记录名称。

单击代码解释器名称，进入“配置信息”页面，获取域名。

在“配置信息”页面，单击URN链接，在新页面获取代码解释器API Key。

图 3-8 获取代码解释器名称

代码解释器 (4) 配额: 1000

第1步

创建并配置代码解释器

支持配置访问权限和网络，提供安全稳定的沙箱环境，确保代码在可控的隔离环境内执行

第2步

在智能体中集成工具

将工具集成至智能体代码，让智能体调用隔离环境完成代码运行等复杂任务。

第3步

利用可观测数据评估效果

支持日志、基础指标、全链路追踪的工具性能、准确性和可用性

默认按照代码解释器名称搜索

代码解释器名称/ID	状态	创建时间	描述	标签
codeinterpreter-fisz2banshaxiang b0045c0c-92e1-4713-892e-9f964009...	运行中	2026/07/13 20:37:24 GMT+08:00	-	

文档版本 01 (2026-09-24)

版权所有 © 华为云计算技术有限公司

26

图 3-9 获取代码解释器域名



图 3-10 获取代码解释器 API Key



步骤3 在AgentArts平台创建网关，配置高德MCP服务。

1. 高德MCP如下，使用前，请参考[高德开放平台文档](#)，创建并获取API Key。

```
{
  "mcpServers": {
    "amap-maps-streamableHTTP": {
      "url": "https://mcp.amap.com/mcp?key=您在高德官网上申请的key"
    }
  }
}
```

2. 在AgentArts平台左侧导航栏中选择“托管与运行 > 网关”，单击“创建网关”。

3. 填写网关基础信息并配置权限与身份认证。参考下表进行配置。

表 3-2 网关基础信息与权限身份认证配置

参数	说明
名称	可自定义。
描述（可选）	网关的描述信息，可自定义。
MCP 版本	2025-03-26
委托	使用平台的默认值。
入站网关	使用默认网关“defaultgw”即可。

参数	说明
入站身份认证	选择API Key认证。
API Key名称	可自定义。
日志记录	选择开启。
工具检索	开启“语义检索”。
高级配置	选择公网访问，开启会话保持。

图 3-11 填写网关基础信息与权限身份认证

基本信息

名称

gateway-gaodemcp

描述 (可选)

请输入描述内容

MCP 版本

2025-03-26

请选择网关需要支持的MCP版本，创建网关后该版本无法修改。[查看详情](#)

权限与访问控制

委托

AgentArtsGatewayAgency

[创建委托](#)

为保证网关正常运行，所选委托需具备相关权限。[查看所需权限](#)[查看当前权限](#)

入网配置

defaultgw

[前往入网配置](#)

入站身份认证

IAM 认证

将华为云 IAM 身份配置为入站身份认证，以验证调用者的身份。

OAuth 2.0 认证

将JWT（如OAuth令牌）配置为入站身份认证，以验证传入的令牌签名和范围。

API Key 认证

将预设的密钥对（键和值）配置为入站身份认证，以验证请求中携带的密钥合法性。

身份认证凭证在凭据管理服务（DEW）托管，托管按需计费。[了解 DEW 计费详情](#)

API Key 名称

APIKey-171364865

4. 单击“创建Target”，配置高德MCP服务。

表 3-3 高德 Target 配置

参数	说明
名称	填写为gaodemap。
描述（可选）	可自定义。
Target协议	选择“MCP”。
类型	选择“MCP”，使用现成的高德MCP服务，不需要使用直接对接外部REST API的方式。
传输方式	选择“Streamable HTTP”，与高德MCP的调用方式保持一致。
MCP地址	填写：https://mcp.amap.com/mcp

文档版本 01 (2026-09-24)

版权所有 © 华为云计算技术有限公司

28

参数	说明
出站身份认证	选择“API Key”，单击“创建出站身份”，参数配置如下： - 身份名称：填写为gaodemcp。 - 认证类型：选择“API Key”。 - API Key的值：填写高德开发平台中创建的API Key。 出站身份创建完成后，返回“创建Target”页面，选择已创建的出站身份。 - 位置：选择“查询参数”。 - 参数名称：填写为key。 - 前缀：不填。 图 3-12 创建出站身份 Agent Identity / 出站身份 / 创建出站身份 < 创建出站身份 图 3-13 配置出站身份认证

5. 配置完成后，单击“确定”创建Target。Target创建完成后，回到“创建网关”页面，高级配置选择“公网访问”后，单击“创建网关”。
6. 网关创建完成后，回到网关列表页面。单击网关名称，记录网关URL及网关的API Key（注意是网关的API Key，不是高德mcp的key）。

图 3-14 网关列表



7. 调测网关。
- MCP网关配置完成后，建议在“调测”页面进行一次测试，验证MCP工具是否可以正常调用。在网关详情页面单击“调测”标签，确认能够正常返回结果后再进行后续开发。

图 3-15 调测网关



8. 在网关详情页面获取网关URL（即MCP调用URL）。

图 3-16 获取网关 URL



9. 在网关详情页面，单击URN链接。进入Agent Identity页面，获取网关的API Key。

图 3-17 单击 URN 链接

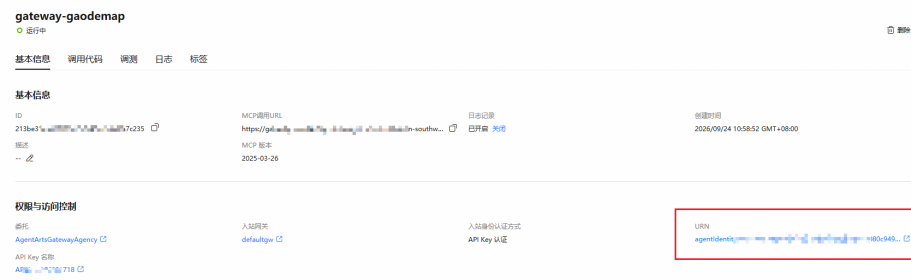
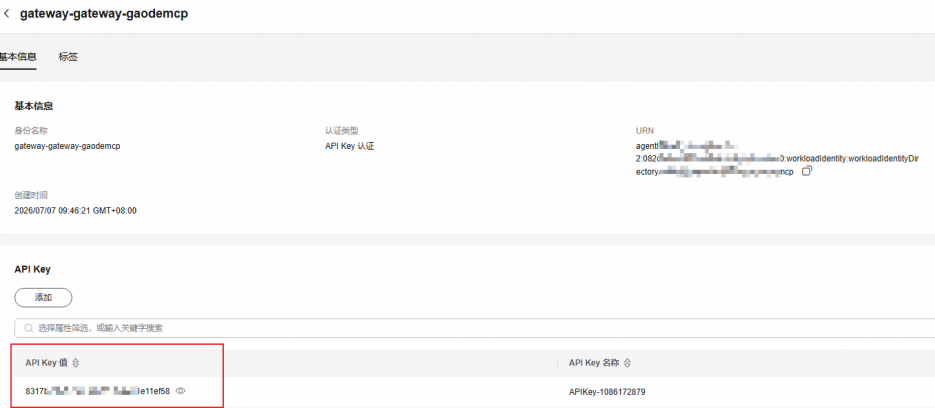


图 3-18 获取网关 API Key



----结束

步骤二：编写 Agent 核心逻辑

使用LangGraph框架编写智能体的核心对话逻辑，并集成记忆库、MCP 网关、代码解释器与华为云MaaS模型。

步骤1 执行如下命令安装langchain、langgraph、langchain-openai。

```
pip install -U langchain langgraph langchain-openai
```

步骤2 创建.env环境变量配置文件。

```
# =====  
# 1. 模型配置 (对接华为云MaaS模型)  
# =====  
MODEL_NAME=deepseek-v4-flash  
MODEL_URL=https://api.modelarts-maas.com/openai/v1  
MODEL_API_KEY=替换为真实的模型API Key  
MODEL_TYPE=openai  
  
# =====  
# 2. AgentArts 记忆库配置  
# =====  
MEMORY_SPACE_ID=替换为记忆库 ID  
HUAWEICLOUD_SDK_MEMORY_API_KEY=替换为记忆库的API Key  
HUAWEICLOUD_SDK_REGION=cn-southwest-2  
  
# =====  
# 3. 网关配置  
# =====  
# 网关地址  
GATEWAY_ENDPOINT=替换为网关的URL  
# 网关的 apikey  
GATEWAY_INBOUND_TOKEN=替换为网关的API Key
```

```
# =====  
# 4. 代码解释器配置  
# =====  
SANDBOX_NAME=在AgentArts平台中创建的代码解释器名称  
AGENTARTS_CODEINTERPRETER_DATA_ENDPOINT="http://您的代码解释器域名"  
HUAWEICLOUD_SDK_CODE_INTERPRETER_API_KEY="您的代码解释器API Key"  
  
# =====  
# 5. Agent 全局配置  
# =====  
MAX_ITERATIONS=50  
SESSION_TTL=3600  
# 本地测试启动 HTTP Server 时绑定的端口  
AGENT_RUN_PORT=8080
```

获取模型API Key的方法请参考[获取华为云MaaS服务模型API Key](#)。

agent.py文件的示例中，使用华为云MaaS服务提供的deepseek-v4-flash模型，如果使用其他模型注意MODEL_NAME的值，需填写为模型接口中model参数的值，可以参考如下方式获取。注意需要选择OpenAI兼容接口。

图 3-19 获取 OpenAI 兼容接口的 model 值



步骤3 在本地创建agent.py文件，编辑代码如下。示例代码中创建了出行助手智能体，并对接记忆库、网关、沙箱、身份认证、华为云MaaS服务提供的模型。

```
=====
AgentArts LangGraph 智能体核心实现
=====
启动时自动从Gateway 拉取 tools/list，动态生成 LangChain Tool 对象。
=====
"""
import os
import uuid
import json
import re
from typing import List, Optional, Dict, Any, Type

import requests
import urllib3
from dotenv import load_dotenv
from pydantic import create_model, Field

urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

from langgraph.graph import StateGraph, END
from langgraph.prebuilt import ToolNode
from langgraph.graph import MessagesState
from langchain_core.messages import BaseMessage, HumanMessage, SystemMessage, AIMessage,
ToolMessage
from langchain_core.tools import StructuredTool
```



```

from langchain_openai import ChatOpenAI

try:
    from agentarts.sdk.memory import MemoryClient
    from agentarts.sdk.memory.inner.config import TextMessage
except ImportError:
    MemoryClient = None
    TextMessage = None
    print("[Warning] 未检测到 agentarts.sdk，将跳过云端记忆同步。")

load_dotenv()

def _apply_reasoning_content_patch():
    import langchain_openai.chat_models.base as _base

    _orig_dict_to_msg = _base._convert_dict_to_message
    def _patched_dict_to_msg(_dict: dict):
        msg = _orig_dict_to_msg(_dict)
        if _dict.get("role") == "assistant":
            reasoning = _dict.get("reasoning_content")
            if reasoning is not None:
                msg.additional_kwargs["reasoning_content"] = reasoning
        return msg
    _base._convert_dict_to_message = _patched_dict_to_msg

    _orig_delta_to_chunk = _base._convert_delta_to_message_chunk
    def _patched_delta_to_chunk(_dict: dict, default_class):
        chunk = _orig_delta_to_chunk(_dict, default_class)
        reasoning = _dict.get("reasoning_content")
        if reasoning is not None:
            chunk.additional_kwargs["reasoning_content"] = reasoning
        return chunk
    _base._convert_delta_to_message_chunk = _patched_delta_to_chunk

    _orig_msg_to_dict = _base._convert_message_to_dict
    def _patched_msg_to_dict(message, **kwargs):
        msg_dict = _orig_msg_to_dict(message, **kwargs)
        if msg_dict.get("role") == "assistant":
            reasoning = getattr(message, "additional_kwargs", {}).get("reasoning_content")
            if reasoning is not None:
                msg_dict["reasoning_content"] = reasoning
        return msg_dict
    _base._convert_message_to_dict = _patched_msg_to_dict

    print("[Patch] 已启用 reasoning_content 透传)")

_apply_reasoning_content_patch()

# =====
# 常量
# =====
_MAX_ITERATIONS = 12
_FORCE_FINAL_REPLY_ITERATION = 10
_TOOL_RESULT_MAX_LENGTH = 8000
_HISTORY_K_MESSAGES = 20
_REQUEST_TIMEOUT = 30
_ACTOR_ID = "user_123"
_ASSISTANT_ID = "travel_agent"
_TOOL_CALL_TAG = "<tool_call>"

_DEFAULT_MODEL_NAME = "deepseek-v4-flash"
_DEFAULT_MODEL_URL = "https://api.modelarts-maas.com/openai/v1"
_DEFAULT_REGION = "cn-southwest-2"
_DEFAULT_MAX_ITERATIONS = 50

_SYSTEM_PROMPT = (
    "你是一个专业的出行规划助手，具备以下能力：\n\n"
    "【 地图服务 - 高德MCP 】\n\n"

```

```

"- 天气查询：帮用户判断出行日期是否合适\n"
"- 地理编码：将地址转为坐标\n"
"- 路径规划：计算驾车/步行/骑行/公交路线\n"
"- 周边搜索：查找餐厅、酒店、景点等\n\n"
"【代码沙箱】\n"
"- 仅在用户明确要求计算费用/预算/总花费时使用\n"
"- 不适用于：查位置、查路线、查天气、展示信息\n\n"
"【云端记忆】\n"
"- 记住用户提到的城市、日期、偏好\n"
"- 跨轮次累积规划，支持增量调整\n\n"
"工作原则：\n"
"1. 规划出行时，先查天气判断可行性\n"
"2. **何时调用 execute_python**：只有当用户明确要求计算费用、预算、总价、多方案费用对比时才调用。"
"查位置、查路线、查天气、展示距离等任务直接用工具返回的结果回答，**不要调用 execute_python**。\n"
" - 对于不确定的价格（如门票、住宿、餐饮），用合理估计值\n"
" - 在 Python 代码中定义这些变量，用代码做加法/乘法计算总费用\n"
" - 禁止直接给出估算数字，必须通过沙箱计算\n"
"3. 记住用户之前提到的所有信息，避免重复询问\n"
"4. 多轮对话中，基于已有上下文做增量调整\n"
"5. 路径规划时，尽量同时查询所有出行方式（驾车、步行、骑行、公交），给用户完整的对比"
)

# =====
# 消息清洗（防止 ModelArts 81001 错误）
# =====
def clean_messages_for_llm(messages: List[BaseMessage]) -> List[BaseMessage]:
    cleaned = []
    expecting_tool_results = False
    for msg in messages:
        if isinstance(msg, AIMessage) and getattr(msg, "tool_calls", None):
            cleaned.append(msg)
            expecting_tool_results = True
        elif isinstance(msg, ToolMessage):
            if expecting_tool_results:
                cleaned.append(msg)
            else:
                print("[MaaS Guard] 检测到孤立的 ToolMessage，已自动过滤。")
        else:
            cleaned.append(msg)
            expecting_tool_results = False
    return cleaned

# =====
# 配置
# =====
class ModelConfig:
    def __init__(self):
        self.name = os.getenv("MODEL_NAME", _DEFAULT_MODEL_NAME)
        self.url = os.getenv("MODEL_URL", _DEFAULT_MODEL_URL)
        self.api_key = os.getenv("MODEL_API_KEY")
        if not self.api_key:
            raise ValueError("[错误] 未找到 MODEL_API_KEY！请在 .env 文件中进行配置。")

class AgentConfig:
    def __init__(self):
        self.model = ModelConfig()
        self.memory_space = os.getenv("MEMORY_SPACE_ID")
        if not self.memory_space:
            print("[Warning] 未配置 MEMORY_SPACE_ID，云端记忆库可能无法正常工作。")
        self.region = os.getenv("HUAWEICLOUD_SDK_REGION", _DEFAULT_REGION)
        self.max_iterations = int(os.getenv("MAX_ITERATIONS", str(_DEFAULT_MAX_ITERATIONS)))

config = AgentConfig()

```

```
# =====
# Gateway 交互层 ( tools/list + tools/call )
# =====
def _mcp_headers() -> Dict[str, str]:
    return {
        "Content-Type": "application/json",
        "Accept": "application/json, text/event-stream",
        "mcp-session-id": str(uuid.uuid4()),
        "Authorization": f"Bearer {os.getenv('GATEWAY_INBOUND_TOKEN', '')}"
    }

def _parse_mcp_response(raw_text: str) -> dict:
    """解析 MCP 响应：兼容 SSE (data: {...}) 和纯 JSON 两种格式"""
    if "data:" in raw_text:
        match = re.search(r'\{.*\}', raw_text, re.DOTALL)
        if match:
            return json.loads(match.group())
    return json.loads(raw_text)

def _mcp_request(method: str, params: dict = None) -> dict:
    """通用 MCP JSON-RPC 请求"""
    gateway_url = os.getenv("GATEWAY_ENDPOINT")
    if not gateway_url:
        raise RuntimeError("未配置 GATEWAY_ENDPOINT")
    payload = {
        "jsonrpc": "2.0",
        "id": str(uuid.uuid4()),
        "method": method,
        "params": params or {}
    }
    resp = requests.post(gateway_url, headers=_mcp_headers(), json=payload, verify=False,
        timeout=_REQUEST_TIMEOUT)
    return _parse_mcp_response(resp.text)

def list_mcp_tools() -> List[dict]:
    """第一步：调 tools/list 拉取 Gateway 上所有工具定义（支持 cursor 分页）"""
    all_tools = []
    cursor = None
    try:
        while True:
            params = {"cursor": cursor} if cursor else {}
            result = _mcp_request("tools/list", params)
            result_data = result.get("result", {})
            all_tools.extend(result_data.get("tools", []))
            cursor = result_data.get("nextCursor")
            if not cursor:
                break
        print(f"[MCP] 从 Gateway 发现 {len(all_tools)} 个工具")
        return all_tools
    except Exception as e:
        print(f"[MCP] tools/list 失败: {e}")
        return all_tools

_TRUNCATION_SUFFIX = "\n...(结果已截断，原始数据过长)"
_MIN_TRUNCATION_RATIO = 0.8

def _truncate_safely(text: str, max_length: int) -> str:
    """安全截断：在 JSON 结构边界处截断，避免破坏数据完整性"""
    if len(text) <= max_length:
        return text
    truncated = text[:max_length]
    last_brace = max(truncated.rfind('{'), truncated.rfind('}'))
    if last_brace > max_length * _MIN_TRUNCATION_RATIO:
        return truncated[:last_brace + 1] + _TRUNCATION_SUFFIX
```

```

return truncated + _TRUNCATION_SUFFIX

def call_mcp_tool(tool_name: str, arguments: Dict[str, Any]) -> str:
    """第四步：模型决定调某工具后，透传到 MCP tools/call"""
    try:
        result = _mcp_request("tools/call", {"name": tool_name, "arguments": arguments})
        text = json.dumps(result.get("result", result), ensure_ascii=False)
        return _truncate_safely(text, _TOOL_RESULT_MAX_LENGTH)
    except Exception as e:
        return f"网关调用异常: {e}"

# =====
# 第二步：把 MCP 工具定义动态转成 LangChain StructuredTool
# =====
_JSON_TYPE_MAP = {
    "string": str,
    "number": float,
    "integer": int,
    "boolean": bool,
    "array": list,
    "object": dict,
}

def _json_schema_to_pydantic(schema: dict, model_name: str = "Args") -> Type:
    """将 MCP inputSchema (JSON Schema) 转为 Pydantic Model 类"""
    properties = schema.get("properties", {})
    required = set(schema.get("required", []))
    fields = {}
    for param_name, param_info in properties.items():
        json_type = param_info.get("type", "string")
        py_type = _JSON_TYPE_MAP.get(json_type, str)
        desc = param_info.get("description", "")
        if param_name in required:
            fields[param_name] = (py_type, Field(..., description=desc))
        else:
            fields[param_name] = (Optional[py_type], Field(None, description=desc))
    return create_model(model_name, **fields) if fields else create_model(model_name)

def _make_tool_caller(tool_name: str):
    """闭包工厂：每个工具绑定自己的 tool_name，防止循环变量捕获问题"""
    def caller(**kwargs):
        clean_args = {k: v for k, v in kwargs.items() if v is not None}
        for k, v in clean_args.items():
            if isinstance(v, (int, float)):
                clean_args[k] = str(v)
        print(f"\n[Gateway] 调用工具: {tool_name} | 参数: {clean_args}")
        return call_mcp_tool(tool_name, clean_args)
    return caller

def build_langchain_tool(mcp_tool_def: dict) -> StructuredTool:
    """将单个 MCP 工具定义转为 LangChain StructuredTool"""
    name = mcp_tool_def["name"]
    desc = mcp_tool_def.get("description", "") or f"MCP tool: {name}"
    schema = mcp_tool_def.get("inputSchema", {})
    args_model = _json_schema_to_pydantic(schema, f"{name}_Args")
    caller = _make_tool_caller(name)
    return StructuredTool.from_function(
        func=caller,
        name=name,
        description=desc,
        args_schema=args_model,
    )

```

```
def discover_mcp_tools() -> List[StructuredTool]:
    """完整的工具发现流程: list -> build"""
    mcp_tools = list_mcp_tools()
    return [build_langchain_tool(t) for t in mcp_tools]

# =====
# 本地工具: 代码解释器（非 MCP，保持独立定义）
# =====
def _execute_python(code: str) -> str:
    from agentarts.sdk.tools import code_session
    print(f"\n[Sandbox] 正在安全沙箱中执行代码...")
    sandbox_name = os.getenv("SANDBOX_NAME")
    api_key = os.getenv("SANDBOX_API_KEY") or
os.getenv("HUAWEICLOUD_SDK_CODE_INTERPRETER_API_KEY")
    if not sandbox_name:
        return "沙箱调用失败: 未在环境变量中配置 SANDBOX_NAME"
    try:
        with code_session(config.region, sandbox_name, api_key=api_key) as client:
            res = client.invoke(operate_type="execute_code", api_key=api_key,
                                arguments={"code": code, "language": "python"})
            return json.dumps(res.get("result"))
    except Exception as e:
        return f"沙箱执行失败: {e}"

execute_python = StructuredTool.from_function(
    func=_execute_python,
    name="execute_python",
    description="在平台沙箱环境中安全执行 Python 代码。Args: code: 待执行的Python代码",
    args_schema=create_model("execute_python_Args", code=(str, Field(..., description="待执行的Python代码"))))

# =====
# LangGraph 工作流
# =====
class AgentState(MessagesState):
    iteration: int

def should_continue(state: AgentState) -> str:
    if state.get("iteration", 0) >= _MAX_ITERATIONS:
        print(f"[Safety] 达到最大迭代次数({_MAX_ITERATIONS}), 强制结束")
        return "end"
    last_message = state["messages"][-1]
    if hasattr(last_message, "tool_calls") and last_message.tool_calls:
        return "continue"
    return "end"

class TravelAgent:
    def __init__(self):
        self.llm = ChatOpenAI(
            model=config.model.name,
            base_url=config.model.url,
            api_key=config.model.api_key,
            temperature=0.1,
        )
        self.mcp_tools = discover_mcp_tools()
        self.all_tools = self.mcp_tools + [execute_python]
        print(f"[Agent] 共加载 {len(self.all_tools)} 个工具 ( MCP: {len(self.mcp_tools)}, 本地: 1 ) ")
        self.graph = self._build_graph()

    def _build_graph(self):
        def model_node(state: AgentState) -> AgentState:
            messages = state["messages"]
            current_iteration = state.get("iteration", 0)
```

```

if current_iteration >= _FORCE_FINAL_REPLY_ITERATION:
    print("[Safety] 接近最大迭代次数，强制生成最终回复")
    final_prompt = SystemMessage(content=(
        "你已达到工具调用次数上限。请基于已有工具返回的结果，"
        "直接给出最终的文字回复，不要再调用任何工具。"
        "如果信息不足，请如实告知用户已获取的部分结果。"
    ))
    safe_messages = clean_messages_for_llm(messages + [final_prompt])
    response = self._llm.invoke(safe_messages)
else:
    llm_with_tools = self._llm.bind_tools(self._all_tools)
    safe_messages = clean_messages_for_llm(messages)
    response = llm_with_tools.invoke(safe_messages)
return {"messages": [response], "iteration": current_iteration + 1}

workflow = StateGraph(AgentState)
workflow.add_node("agent", model_node)
workflow.add_node("tools", ToolNode(self._all_tools))
workflow.set_entry_point("agent")
workflow.add_conditional_edges("agent", should_continue, {"continue": "tools", "end": END})
workflow.add_edge("tools", "agent")
# 不使用 checkpointer，每轮从云端记忆库加载历史
return workflow.compile()

def run(self, user_input: str, session_id: str) -> str:
    thread_config = {"configurable": {"thread_id": session_id}}
    self._ensure_session(session_id)
    history = self._load_cloud_history(session_id)
    messages = self._build_messages(history, user_input)
    prev_count = len(messages) - 1
    result = self.graph.invoke(
        {"messages": messages, "iteration": 0},
        config=thread_config,
        recursion_limit=config.max_iterations
    )
    new_messages = result["messages"][prev_count:]
    self._save_to_cloud_memory(session_id, new_messages)
    return self._clean_response(result["messages"][-1].content)

def _ensure_session(self, session_id: str):
    """确保云端记忆会话存在"""
    if MemoryClient is None or not config.memory_space:
        return
    try:
        MemoryClient().create_memory_session(
            space_id=config.memory_space,
            id=session_id,
            actor_id=_ACTOR_ID
        )
    except Exception as e:
        if "Session id already exists" not in str(e):
            print(f"[Warning] 云端记忆会话同步失败: {e}")

def _build_messages(self, history: List[BaseMessage], user_input: str) -> List[BaseMessage]:
    """构建送入 LLM 的消息列表：系统提示 + 历史 + 当前输入"""
    messages = [SystemMessage(content=_SYSTEM_PROMPT)]
    messages.extend(history)
    messages.append(HumanMessage(content=user_input))
    return messages

def _clean_response(self, response: str) -> str:
    """清理可能幻觉的工具调用标签"""
    if _TOOL_CALL_TAG in response:
        return response[:response.index(_TOOL_CALL_TAG)].strip()
    return response

def _load_cloud_history(self, session_id: str) -> List[BaseMessage]:
    """从云端记忆加载历史消息"""
    if MemoryClient is None or not config.memory_space:

```

```
        return []
    try:
        client = MemoryClient()
        response = client.get_last_k_messages(
            space_id=config.memory_space, session_id=session_id, k=_HISTORY_K_MESSAGES
        )
        msgs = None
        if isinstance(response, list):
            msgs = response
        elif hasattr(response, 'messages'):
            msgs = response.messages
        if msgs is None:
            print(f"[Memory] 无法提取消息列表, {type(response).__name__}")
            return []
        history = []
        for m in msgs:
            role = getattr(m, "role", "")
            content = getattr(m, "content", "")
            if not content and hasattr(m, 'parts') and m.parts:
                part = m.parts[0]
                content = part.get('text', "") if isinstance(part, dict) else getattr(part, 'text', "")
            if role == "user" and content:
                history.append(HumanMessage(content=content))
            elif role == "assistant" and content:
                history.append(AIMessage(content=content))
        if history:
            print(f"[Memory] 从云端加载 {len(history)} 条历史消息")
        return history
    except Exception as e:
        print(f"[Memory] 加载历史失败: {e}")
        return []

def _save_to_cloud_memory(self, session_id: str, new_messages: List[BaseMessage]):
    """增量同步：只写入本轮新增的消息到 AgentArts 记忆库"""
    if MemoryClient is None or TextMessage is None or not config.memory_space:
        return
    try:
        text_messages = []
        for msg in new_messages:
            if isinstance(msg, HumanMessage) and msg.content:
                text_messages.append(TextMessage(role="user", content=msg.content, actor_id=_ACTOR_ID,
assistant_id=_ASSISTANT_ID))
            elif isinstance(msg, AIMessage) and msg.content:
                text_messages.append(TextMessage(role="assistant", content=msg.content,
actor_id=_ACTOR_ID, assistant_id=_ASSISTANT_ID))
        if text_messages:
            MemoryClient().add_messages(
                space_id=config.memory_space,
                session_id=session_id,
                messages=text_messages
            )
            print(f"[Memory] 增量同步 {len(text_messages)} 条新消息到云端")
    except Exception as e:
        print(f"[Memory] 云端同步失败（不影响对话）: {e}")
```

----结束

步骤三：本地功能验证

在本地直接运行智能体，确认智能体可以正常运行。

步骤1 创建main.py文件，用于本地运行智能体，并进行测试。测试脚本为交互式对话，可直接输入问题测试agent.py中集成的基础对话、网关、代码解释器、记忆恢复等能力。

```
"""
出行规划助手 - 交互式场景演示
"""
```



```
import sys
import uuid
from agent import TravelAgent

_EXIT_COMMANDS = {"exit", "quit", "退出"}

def safe_input(prompt="你: "):
    """
    兼容 Cloud Shell 终端编码的输入函数。
    Cloud Shell 可能以 GBK 编码发送中文字符，而 Python 默认按 UTF-8 解码会报 UnicodeDecodeError。
    策略：先尝试 UTF-8，失败后回退到 GBK，再失败则用 replace 模式兜底。
    """
    print(prompt, end="", flush=True)
    raw = sys.stdin.buffer.readline()
    if not raw:
        return ""
    try:
        return raw.decode("utf-8").strip()
    except UnicodeDecodeError:
        try:
            return raw.decode("gbk").strip()
        except UnicodeDecodeError:
            return raw.decode("utf-8", errors="replace").strip()

def print_guide():
    print("\n" + "=" * 60)
    print(" 杭州周末出行规划助手")
    print("=" * 60)
    print("""
我可以帮你：
- 查天气，判断出行是否合适
- 查地点、规划路线
- 计算行程预算
- 记住你的偏好，持续优化方案

建议对话流程：
1. "我周末想去杭州玩，先帮我看看天气"
2. "查一下西湖和灵隐寺的位置"
3. "这两个地方之间怎么走？多远？"
4. "帮我算一下两天行程的总预算"
5. "如果改成三天呢？住宿多一晚"
""")
    print("=" * 60 + "\n")

def main():
    print("正在初始化智能体助手（集成网关、代码解释器与云端记忆）...")
    agent = TravelAgent()
    session_id = str(uuid.uuid4())
    print(f"会话 ID: {session_id}")
    print_guide()
    while True:
        try:
            user_input = safe_input()
            if not user_input:
                continue
            if user_input.lower() in _EXIT_COMMANDS:
                print("再见！祝旅途愉快～")
                break
            print("\n" + "-" * 40)
            response = agent.run(user_input, session_id=session_id)
            print(f"\n助手: {response}\n")
            print("-" * 40 + "\n")
        except KeyboardInterrupt:
            print("\n再见！")
            break
```

```
if __name__ == "__main__":
    main()
```

步骤2 将agent.py、.env、main.py上传至服务器项目目录中。

```
root@ecs-ecdd:~/test2# ll
total 36
drwxr-xr-x  2 root root  4096 May 23 18:32 ./
drwx----- 23 root root  4096 May 23 18:28 ../
-rw-r--r--  1 root root 11235 May 23 18:32 agent.py
-rw-r--r--  1 root root  1813 May 23 18:32 app.py
-rw-r--r--  1 root root  1178 May 23 18:31 .env
-rw-r--r--  1 root root  2862 May 23 18:32 main.py
-rw-r--r--  1 root root  2284 May 23 18:32 requirements.txt
```

步骤3 执行python main.py命令对智能体进行本地测试。由于main.py是交互式对话程序，运行后可直接输入问题，智能体会调用大模型进行回复。有正常的回复即表示本地测试成功。

```
=====
你：我周末想去杭州玩，先帮我看看天气

-----
2026-08-17 16:44:08,449 - agentarts.sdk.memory.inner.dataplane - INFO - DataPlane initialized
2026-08-17 16:44:08,449 - agentarts.sdk.memory.inner.dataplane - INFO - Creating memory session in space: 680e90ab-bbc1-41d1-8004-9d04b78ac876
2026-08-17 16:44:08,681 - agentarts.sdk.memory.inner.dataplane - INFO - Memory session created: 62c36af6-2516-49be-ba9f-f04a3149a240
2026-08-17 16:44:08,682 - agentarts.sdk.memory.inner.dataplane - INFO - DataPlane initialized
2026-08-17 16:44:08,683 - agentarts.sdk.memory.inner.dataplane - INFO - Getting last 20 messages from session: 62c36af6-2516-49be-ba9f-f04a3149a240

[MCP Gateway] 调用工具: target-1hy090it_maps_weather | 参数: {'city': '杭州'}
2026-08-17 16:44:42,494 - agentarts.sdk.memory.inner.dataplane - INFO - DataPlane initialized
2026-08-17 16:44:42,494 - agentarts.sdk.memory.inner.dataplane - INFO - Adding 3 messages to session: 62c36af6-2516-49be-ba9f-f04a3149a240
2026-08-17 16:44:42,677 - agentarts.sdk.memory.inner.dataplane - INFO - Messages added to session: 62c36af6-2516-49be-ba9f-f04a3149a240
[Memory] 增量同步 3 条新消息到云端

助手：以下是杭州近几天的天气预报：

| 日期 | 星期 | 白天天气 | 夜间天气 | 最高温 | 最低温 | 风向 | 风力 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 8月17日 | 周一 | ☁️小雨 | ☁️多云 | 33°C | 26°C | 东风 | 1-3级 |
| 8月18日 | 周二 | ☁️小雨 | ☁️多云 | 34°C | 25°C | 东北风 | 1-3级 |
| 8月19日 | 周三 | ☁️多云 | ☁️阴 | 34°C | 26°C | 东风 | 1-3级 |
| 8月20日 | 周四 | ☁️中雨 | ☁️小雨 | 31°C | 25°C | 东风 | 1-3级 |

---

### 🌤️天气分析

从目前预报来看，杭州这几天**以雨雨雨为主**，气温在 **25°C-34°C** 之间，属于典型的夏季闷热天气：

- **周一、周二**：白天有小雨，气温较高，体感闷热
- **周二**：多云转阴，暂时无雨，是这几天中天气相对较好的
- **周四**：中雨转小雨，降雨较明显

### 📌出行建议

1. **周末（8/22-8/23）的预报暂时还没出**，但从本周趋势看，杭州正处于多雨期，周末大概率也有降雨，建议临近时再关注
2. **务必带好雨具** ☔️ 雨伞/雨衣是必备的
3. 气温高+降雨=闷热潮湿，注意防暑降温，多补水
4. 如果周末确实下雨，可以考虑西湖烟雨游也别有一番风味 🌧️ 或者安排一些室内景点（如浙江省博物馆、中国丝绸博物馆等）

---

你想让我帮你规划杭州的游玩路线吗？可以告诉我从哪个城市出发，以及感兴趣的方向（自然风光/人文历史/美食打卡等），我来帮你安排！

-----

你：查一下西湖和灵隐寺的位置

-----
2026-08-17 16:44:51,129 - agentarts.sdk.memory.inner.dataplane - INFO - DataPlane initialized
2026-08-17 16:44:51,129 - agentarts.sdk.memory.inner.dataplane - INFO - Creating memory session in space: 680e90ab-bbc1-41d1-8004-9d04b78ac876
```

----结束

交互场景与组件调用

问题 1: "我周末想去杭州玩，先帮我看看天气"

- 用户意图：查询目的地天气，判断出行日期是否合适
- 调用组件：网关、记忆库
- 组件作用：
 - MCP Gateway：LLM 决策调用 maps_weather 工具，Gateway 转发请求到高德天气服务，返回杭州未来几天的天气数据（温度、天气状况、风向）。
 - 记忆库：首次对话，MemoryClient().create_memory_session() 在云端创建记忆会话，为跨会话持久化做准备；get_last_k_messages(k=20) 加载 0 条历史（首轮无历史）；对话后增量同步 3 条新消息（用户输入 + AI 工具调用 + AI 最终回复）到云端记忆库。

问题 2: "查一下西湖和灵隐寺的位置"

- 用户意图：获取两个景点的地理坐标和距离
- 调用组件：网关、记忆库
- 组件作用：
 - MCP Gateway：LLM 连续调用 3 次工具——maps_geo（西湖坐标）、maps_geo（灵隐寺坐标）、maps_distance（两点驾车距离 4.4 公里）。LLM 自主决策调用顺序和参数，Gateway 透传执行。
 - 记忆库：对话前 get_last_k_messages(k=20) 从云端加载 3 条历史消息（Turn 1 的对话），LLM 能记住"之前查了杭州天气"的语境；对话后增量同步 4 条新消息到云端记忆库。

问题 3: "这两个地方之间怎么走？多远？"

- 用户意图：获取多种出行方式的路线规划
- 调用组件：网关、记忆库
- 组件作用：
 - MCP Gateway：LLM 一次性调用 4 个路径规划工具——maps_direction_driving（驾车）、maps_direction_walking（步行）、maps_direction_bicycling（骑行）、maps_direction_transit_integrated（公交）。Gateway 并行执行，返回 4 种出行方式的路线数据。安全截断机制检查每个结果长度，超过 8000 字符时会在 JSON 结构边界处截断，避免破坏数据完整性，防止云端记忆写入失败。
 - 记忆库：对话前 get_last_k_messages(k=20) 从云端加载 7 条历史消息（Turn 1-2 的对话），LLM 记住前两轮的景点坐标和天气数据，回复中引用"结合周末雨天情况"给出出行建议；对话后增量同步 3 条新消息到云端记忆库。

问题 4: "帮我算一下两天行程的总预算"

- 用户意图：精确计算两日行程的费用明细
- 调用组件：代码解释器、记忆库
- 组件作用：
 - 代码解释器：LLM 不调用 MCP 工具，直接调用 execute_python，在沙箱中执行 Python 代码——定义住宿（经济型 300 元/晚）、餐饮、交通等变量，用代码做加法计算总费用，返回精确结果。LLM 基于沙箱返回的数值生成预算明细表格。

- 记忆库：对话前 `get_last_k_messages(k=20)` 从云端加载 10 条历史消息（Turn 1-3 的对话），LLM 记住前 3 轮的景点和路线信息，在预算计算中引用之前查询的结果；对话后增量同步 3 条新消息到云端记忆库。

问题 5："如果改成三天呢？住宿多一晚"

- 用户意图：基于已有两日预算做增量调整
- 调用组件：代码解释器、记忆库
- 组件作用：
 - 代码解释器：LLM 生成新的 Python 代码，复用 Turn 4 的变量定义，增加第三天的费用项（+1 晚住宿 300 元 +1 天餐饮 280 元 +1 天交通 40 元 + 额外景点 80 元），沙箱执行增量计算，返回 +700 元的差额。
 - 记忆库：对话前 `get_last_k_messages(k=20)` 从云端加载 13 条历史消息（Turn 1-4 的对话），LLM 记住 Turn 4 的两天预算方案，能做增量调整而非重新计算；对话后增量同步 3 条新消息到云端记忆库，保存完整 5 轮对话上下文，下次开新会话能加载历史继续规划。

步骤四：SDK 封装与本地接口验证

将智能体封装为符合 AgentArts 平台规范的 HTTP 服务，并在本地模拟云端接口调用，确认封装正确后再执行后续的部署上云操作。

步骤1 SDK封装，AgentArts平台通过标准HTTP协议与智能体通信，您需要用AgentArts SDK 将Agent逻辑包一层，将其暴露为符合平台规范的/inocations接口，平台才能正确调度和管理您的智能体。创建app.py文件，并上传至服务器中。该文件会将本地的Agent逻辑封装为符合AgentArts平台规范的Web服务。

```
"""
AgentArts 平台 HTTP 封装层
将 TravelAgent 暴露为符合平台规范的 /inocations 接口
"""
import socket
import uuid
from typing import Dict, Any

# =====
# DNS 补丁：容器内无法解析 AgentArts 公网域名
# 必须在 import agent / sdk 之前执行
# 通过 nslookup <域名> 获取对应 IP，替换下面的 XXX.XXX.XXX.XXX
# =====
_DNS_OVERRIDES = {
    "memory.cn-southwest-2.huaweicloud-agentarts.com": "XXX.XXX.XXX.XXX",
    "gateway-gaodemcp-defaultgw-ge3wkzbthf.cn-southwest-2.huaweicloud-agentarts.com":
    "XXX.XXX.XXX.XXX",
    "defaultgw-ge3wkzbthf.cn-southwest-2.huaweicloud-agentarts.com": "XXX.XXX.XXX.XXX",
}

_orig_getaddrinfo = socket.getaddrinfo

def _patched_getaddrinfo(host, port, family=0, type=0, proto=0, flags=0):
    if host and host in _DNS_OVERRIDES:
        return _orig_getaddrinfo(_DNS_OVERRIDES[host], port, family, type, proto, flags)
    return _orig_getaddrinfo(host, port, family, type, proto, flags)

socket.getaddrinfo = _patched_getaddrinfo

from agentarts.sdk import AgentArtsRuntimeApp, RequestContext
from agent import TravelAgent

app = AgentArtsRuntimeApp()

try:
```

```

my_agent = TravelAgent()
except Exception as e:
    print(f"Agent 初始化失败: {e}")
    my_agent = None

def _resolve_session_id(context: RequestContext = None, payload: Dict[str, Any] = None) -> str:
    raw = None
    if context:
        raw = getattr(context, "session_id", None)
    if not raw and payload:
        raw = payload.get("session_id")
    if not raw:
        raw = str(uuid.uuid4())
    return str(raw)

@app.entrypoint
def invoke(payload: Dict[str, Any], context: RequestContext = None) -> Dict[str, Any]:
    message = payload.get("message", "")
    session_id = _resolve_session_id(context, payload)

    if my_agent is None:
        return {"response": "Agent 初始化失败", "status": "error", "session_id": session_id}

    try:
        response = my_agent.run(message, session_id=session_id)
        return {"response": response, "status": "success", "session_id": session_id}
    except Exception as e:
        return {"response": f"处理失败: {str(e)}", "status": "error", "session_id": session_id}

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=8080)

```

步骤2 接口验证，在正式推送镜像之前，先在本地模拟云端HTTP调用环境，验证Agent的接口封装是否正确、通信是否正常，可以大幅降低因接口问题导致云端部署后才发现错误的调试成本。执行python app.py启动http server，执行以下命令调用验证Agent的HTTP接口是否已经被正确封装且能正常通信。执行python app.py回显效果如下。

```

(venv) root@ecs-ecdd:~/test2# python app.py
2026-05-23 18:37:07,131 - agentarts.sdk.memory.inner.dataplane - INFO - DataPlane initialized
2026-05-23 18:37:07,168 - agentarts.runtime.app - INFO - Starting AgentArts runtime on 127.0.0.1:8080 (http)
INFO: Started server process [7224]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: Uvicorn running on http://127.0.0.1:8080 (Press CTRL+C to quit)

```

打开一个新的终端窗口（保持原窗口运行），使用curl命令进行测试。测试完成后，可以使用Ctrl + C停止运行的进程。

```

curl --location --request POST 'http://localhost:8080/invocations' \
--header 'Content-Type: application/json' \
--data-raw '{"message": "你好，请用一句话做个自我介绍，并查询下杭州的天气。"}'

```

```

Last login: Sat May 23 19:16:57 2026 from 198.19.222.46
root@ecs-ecdd:~# curl --location --request POST 'http://localhost:8080/invocations' \
--header 'Content-Type: application/json' \
--data-raw '{"message": "你好，请用一句话做个自我介绍，并查询下杭州的天气。"}'
{"response": "根据查询结果，杭州的天气情况如下：\n\n**今天（5月23日，周六）**： 小雨，白天温度31°C，夜间温度23°C，东风1-3级\n\n**明天（5月24日，周日）**： 中雨转小雨，白天温度30°C，夜间温度24°C，西南风1-3级\n\n**后天（5月25日，周一）**： 中雨转小雨，白天温度32°C，夜间温度24°C，东南风1-3级\n\n**5月26日（周二）**： 中雨，白天温度30°C，夜间温度23°C，北风1-3级\n\n未来几天 点击放大 降雨，建议出门带好雨具。", "status": "success"}
root@ecs-ecdd:~#

```

----结束

步骤五：云端部署

将本地智能体打包为Docker镜像并一键部署至AgentArts云端运行时。

步骤1 部署智能体运行时。按上述步骤完成后，基本代码开发已经完成，接下来准备将本地创建好的智能体部署托管到AgentArts平台。首先准备依赖文件requirements.txt，内容可参考如下：

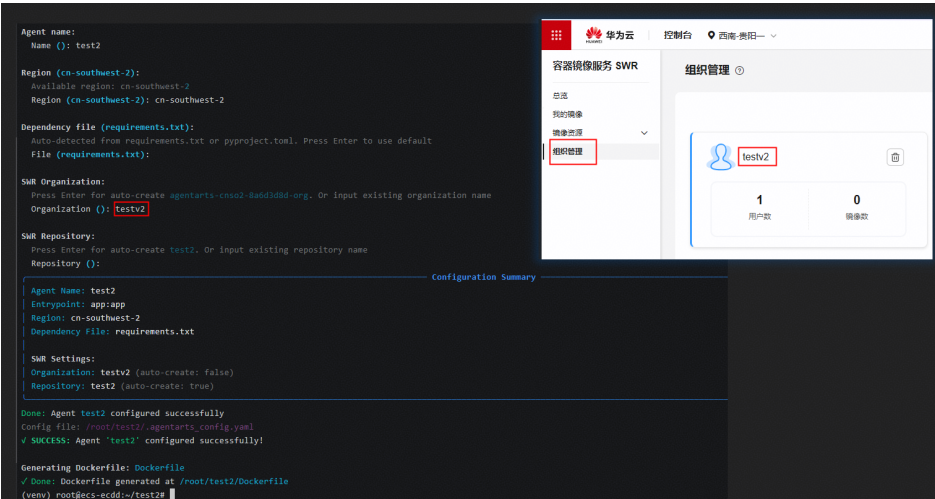
```
# =====  
# AgentArts LangGraph Agent Demo - 依赖清单  
# =====  
#  
# 本项目基于 AgentArts 平台，使用 LangGraph 框架构建 AI Agent  
#  
# 安装方式:  
# pip install -r requirements.txt  
#  
# =====  
  
# 指定国内 PyPI 镜像源  
--index-url https://repo.huaweicloud.com/repository/pypi/simple  
--trusted-host repo.huaweicloud.com  
  
# =====  
# LangGraph & LangChain 核心框架  
# =====  
langgraph>=0.2.0  
langchain>=0.3.0  
langchain-core>=0.3.0  
langchain-openai>=0.1.0  
  
# =====  
# LLM Provider ( ModelArts MaaS OpenAI 兼容接口 )  
# =====  
openai>=1.0.0  
  
# =====  
# HTTP & 网络  
# =====  
requests>=2.31.0  
urllib3>=2.0.0  
  
# =====  
# 工具与配置  
# =====  
python-dotenv>=1.0.0  
pydantic>=2.0.0  
  
# =====  
# AgentArts 平台 SDK  
# =====  
agentarts-sdk>=0.1.0
```

步骤2 执行如下命令配置智能体。

```
agentarts configure --entrypoint app:app
```

执行后按照操作指引进行配置。配置智能体名称（以小写字母开头，以小写字母或数字结尾，可以包含小写字母、数字和中划线）、服务部署区域（使用cn-southwest-2，仅支持此区域）、requirements.txt依赖文件、SWR Organization镜像组织名称（如果使用自定义的镜像组织名，需要在SWR服务控制台贵阳一region创建）。

Ubuntu系统可忽略此提示，如果您使用的是Huawei Cloud EulerOS 3.0系统构建Agent镜像，在执行完成agentarts configure --entrypoint app:app命令后，建议手动修改下命令生成的Dockerfile文件，再执行后续操作。防止制作好的镜像因缺少依赖包，造成调用Agent时出现沙箱报错。详见[常见问题](#)。



步骤3 执行命令部署智能体。

agentarts launch

该命令会自动完成以下步骤：

1. 本地构建Docker镜像。
2. 将Docker镜像推送到华为云SWR镜像仓库。
3. 部署到AgentArts运行时托管环境。


```

(venv) root@ecs-ecdd1-1:test2@ agentarts launch
Agent: test2
Mode: cloud
Region: cn-southwest-2

Deploy Configuration

Step 1/5 Building Docker image
Image: test2:latest

Building Docker image: test2:latest
▶ Step 1/13: FROM python:3.10-slim
▶ Step 2/13: ENV DOCKER_CONTAINER=true
▶ Step 3/13: ENV HUAWAICLOUD_SDK_REGION=cn-southwest-2
▶ Step 4/13: RUN groupadd -g 1000 appuser && useradd -u 1000 -g 1000 -m -s /bin/bash appuser
▶ Step 5/13: WORKDIR /app
▶ Step 6/13: RUN chown appuser:appuser /app
▶ Step 7/13: COPY requirements.txt .
▶ Step 8/13: RUN pip install --no-cache-dir -r requirements.txt -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
▶ Step 9/13: COPY . .
▶ Step 10/13: RUN chown -R appuser:appuser /app
▶ Step 11/13: EXPOSE 8080
▶ Step 12/13: USER appuser
▶ Step 13/13: CMD ["python", "-m", "app"]
✓ Successfully built e4b3db0c00c
✓ Successfully tagged test2:latest

✓ Done: Image test2:latest built successfully

Step 2/5 Setting up SWR resources
Organization: testv2
Repository: test2

✓ SUCCESS: Using existing organization testv2
Repository 'testv2/test2': query failed: ClientRequestException - {status_code:404,request_id:a338a3789b050625f8216a266d19ae04,error_code:SVCSTG.SWR.4048010,ists,encoded_authorization_message: }
✓ Created SWR repository: testv2/test2
✓ SUCCESS: Repository testv2/test2 ready

Step 3/5 Getting SWR credentials
Registry: swr.cn-southwest-2.myhuaweicloud.com

✓ Created SWR secret for region: cn-southwest-2

Logging in to registry: swr.cn-southwest-2.myhuaweicloud.com
✓ Done: Logged in to registry successfully

Step 4/5 Tagging and pushing image
Source: test2:latest
Target: swr.cn-southwest-2.myhuaweicloud.com/testv2/test2:latest

Tagging image: test2:latest -> swr.cn-southwest-2.myhuaweicloud.com/testv2/test2:latest
✓ Done: Image tagged successfully

Pushing image: swr.cn-southwest-2.myhuaweicloud.com/testv2/test2:latest
  ○ Layer exists (skipped)
  ○ Layer exists (skipped)
  ○ Layer exists (skipped)
  ○ Layer exists (skipped)
  ○ Layer exists (skipped)
  ○ Layer exists (skipped)
  ○ Layer exists (skipped)
  ○ Layer exists (skipped)
  ○ Layer exists (skipped)
  ○ Layer exists (skipped)
  ✓ 3e216e841305: Pushed
  ✓ 60b57772b25c: Pushed
  ✓ Digest: sha256:e4b3db0c00c68a95e9816e0f7cf2ada94110d74f057e5382376a99a78ddaa1d
✓ Done: Image pushed successfully
✓ SUCCESS: Image deployed to swr.cn-southwest-2.myhuaweicloud.com/testv2/test2:latest

Step 5/5 Creating AgentArts runtime
Agent: test2
Image: swr.cn-southwest-2.myhuaweicloud.com/testv2/test2:latest

Creating AgentArts runtime: test2...
✓ SUCCESS: Runtime 'test2(v1)' created/updated successfully with ID: 0b0e4ebe-a9bc-4535-9b57-053eabc9392c

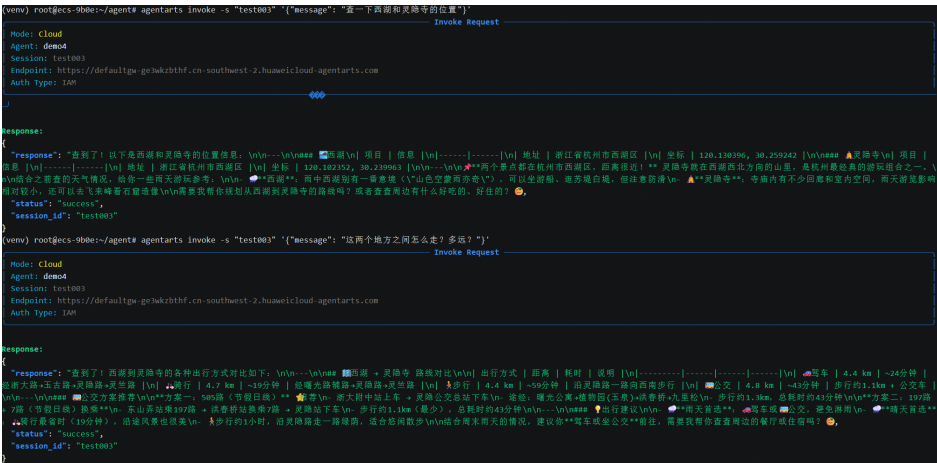
Deployment complete!

Agent Name: test2
Runtime ID: 0b0e4ebe-a9bc-4535-9b57-053eabc9392c
Image: swr.cn-southwest-2.myhuaweicloud.com/testv2/test2:latest
Region: cn-southwest-2
Access Endpoint: defaultgw-wjswnjgqj.cn-southwest-2.huaweicloud-agentarts.com

```

4. 调用云端Agent进行会话。

```
agentarts invoke --agent <你的agent名称> -s "<会话ID>" '{"message": "查一下西湖和灵隐寺的位置"}'
```



---结束

常见问题

- 使用Huawei Cloud EulerOS 3.0系统制作镜像，invoke命令调用智能体出现沙箱报错。



处理措施：

- a. 检查linux服务器是否值ARM架构，如果是X86架构，建议改为ARM架构服务器重新制作镜像。
- b. 使用EulerOS 3.0系统制作镜像时。在执行完成agentarts configure --entrypoint app:app命令后。需要手动修改下Dockerfile文件，再执行后续的agentarts launch命令，详细操作如下。
 - i. 备份下原有的Dockerfile文件。
cp Dockerfile Dockerfile.bak
 - ii. 执行vi Dockerfile命令修改Dockerfile文件，将下面的内容替换进去。

此Dockerfile可以避免非root用户读取site-packages时的权限问题；同时在安装依赖后增加了agentarts sdk的导入校验，安装失败会在构建阶段直接暴露，不会带着异常镜像进入部署流程。

```
# AgentArts Dockerfile
# Generated by AgentArts CLI
# Modified for EulerOS: removed non-root user, added pip source and SDK validation

FROM python:3.10-slim
```

```
# Set container environment marker
ENV DOCKER_CONTAINER=true
ENV HUAWEICLOUD_SDK_REGION=cn-southwest-2

WORKDIR /app

COPY requirements.txt .

# 使用华为云 PyPI 镜像源，安装后立即验证 agentarts.sdk 可导入
RUN pip install --no-cache-dir -r requirements.txt \
    -i https://repo.huaweicloud.com/repository/pypi/simple \
    --trusted-host repo.huaweicloud.com \
    && python -c "from agentarts.sdk import AgentArtsRuntimeApp; print('SDK OK')"

# 显式复制必要文件，避免将 venv 等无关文件带入镜像
COPY agent.py app.py main.py .env ./

EXPOSE 8080

CMD ["python", "-m", "app"]
```

- iii. 执行 `vi .dockerignore` 命令，创建 `.dockerignore` 文件，并写入以下内容。防止 `venv/` 等被 `COPY` ... 打包污染镜像。

```
cat > .dockerignore <<'EOF'
venv/
__pycache__
*.pyc
.git/
Dockerfile
Dockerfile.bak
EOF
```

- c. 执行 `agentarts launch` 命令，部署智能体到运行时。

- 执行 `agentarts launch` 命令时 出现 `AK/SK` 认证报错

执行以下命令配置华为云凭证，获取华为云凭证请参考[认证鉴权](#)。

```
export HUAWEICLOUD_SDK_AK="your-access-key"
export HUAWEICLOUD_SDK_SK="your-secret-key"
```

```
2026-05-22 17:24:46,221 - agentarts.sdk.metadata_provider - WARNING - Get metadata credentials failed with status: 404
Get metadata credentials failed with status: 404
Organization 'demo1' query failed: Authentication failed: Could not find valid credentials. Please configure one of the following:
1. AK/SK Authentication: Set HUAWEICLOUD_SDK_AK, HUAWEICLOUD_SDK_SK
2. OIDC Token: Set HUAWEICLOUD_SDK_IDP_ID, HUAWEICLOUD_SDK_ID_TOKEN_FILE, and HUAWEICLOUD_SDK_PROJECT_ID
3. Metadata: Running on the AgentArts runtime
x ERROR: Organization 'demo1' not found
Set organization_auto_create: true in config to auto-create
(venv) root@ecs-ecdd:~/demo1# export HUAWEICLOUD_SDK_AK=HPUA
(venv) root@ecs-ecdd:~/demo1# export HUAWEICLOUD_SDK_SK=Vadm
(venv) root@ecs-ecdd:~/demo1#
```

- 执行 `agentarts launch` 命令出现运行时 `Runtime` 名称错误

智能体的名称需要以小写字母开头，以小写字母或数字结尾，可以包含小写字母、数字和中划线，长度为2-48个字符。

请重新执行 `agentarts configure --entrypoint app:app` 命令进行配置。

```
Creating AgentArts runtime: demo_02...
2026-05-23 18:07:07,852 - agentarts.sdk.service.runtime_client - ERROR - create_agent failed: status=400, data={'error_code': 'AgentArts.03005400', 'error_msg': 'Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48'}, error=Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48
create_agent failed: status=400, data={'error_code': 'AgentArts.03005400', 'error_msg': 'Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48'}, error=Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48
x ERROR: Failed to create runtime: create_agent failed (HTTP 400): Runtime name must start with a lowercase letter, contain only lowercase letters, numbers and hyphens, cannot end with a hyphen, length 2-48
(venv) root@ecs-ecdd:~/demo_02# agentarts configure --entrypoint app:app
```

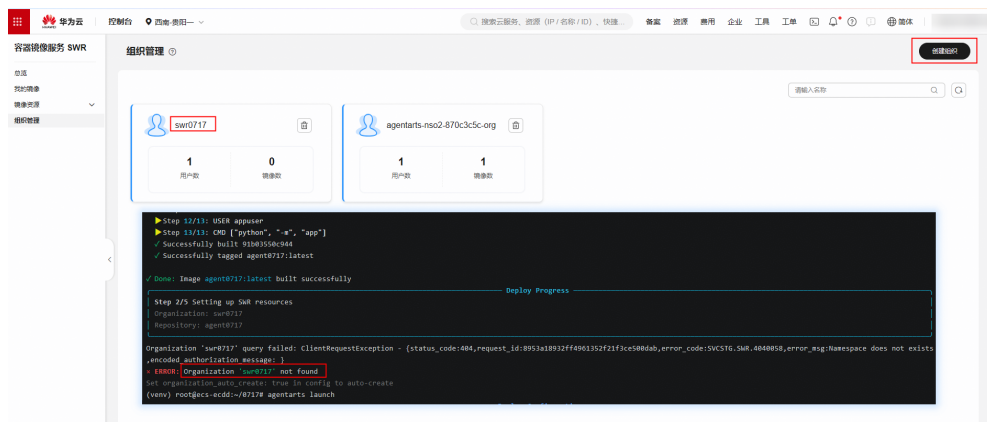
- 执行命令，出现 `SWR.4040010,error_msg:Repository does not exists,ERROR: Organization 'xxx' not found` 报错。

问题现象及原因

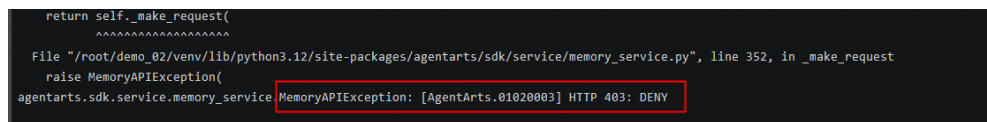
SWR服务镜像组织不存在，出现该报错的原因通常是在执行 `agentarts configure --entrypoint app:app` 命令时，手动指定了一个SWR镜像组织名称，但是未在SWR控制台创建同名组织导致。

解决方案

登录**SWR服务控制台**（贵阳一区域），在“组织管理”中手动创建一个与agentarts configure --entrypoint app:app命令中同名的镜像组织。



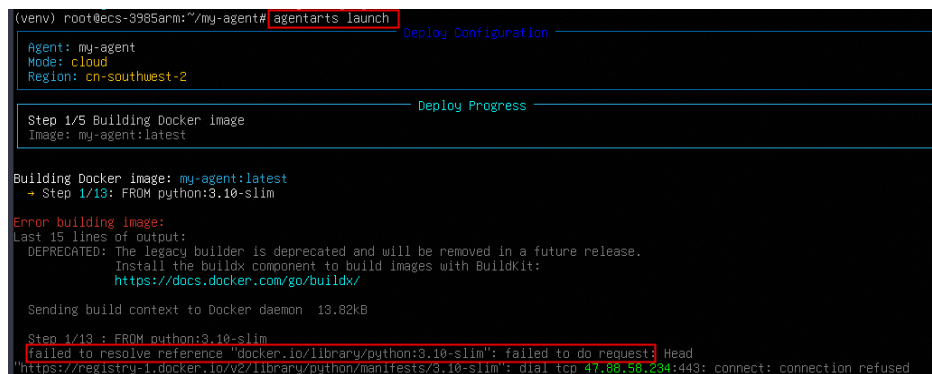
- 执行python main.py或者python app.py命令出现Memory记忆库报错。请检查记忆库ID以及记忆库API Key配置是否正确，建议重新配置该值。



- 执行agentarts launch命令，出现failed to resolve reference "docker.io/library/python:3.10-slim"类型报错。

问题现象及原因

尝试连接Docker官方镜像仓库（docker.io）下载python:3.10-slim基础环境时，网络连接超时（网络连接被阻断或者限速）。



解决方案一：配置Docker国内镜像加速器（推荐）

执行以下命令在系统配置目录下创建一个名为docker的文件夹。

```
sudo mkdir -p /etc/docker
```

执行以下命令（直接复制下面这段代码并回车执行），更换Docker镜像源，从国内镜像站寻找python:3.10-slim。

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors":[
    "https://docker.m.daocloud.net",
    "https://dockerproxy.net",
    "https://mirror.baidubce.com"
  ]
}
EOF
```

依次执行下面两条命令，重启Docker服务：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

Docker配置完成后，再次执行agentarts launch命令进行Agent部署。

解决方案二：直接修改Dockerfile基础镜像地址

- 在项目根目录下找到Dockerfile（由agentarts launch自动生成或已存在）。
- 编辑Dockerfile，将FROM python:3.10-slim替换为国内镜像地址，例如：
sed -i 's|FROM python:3.10-slim|FROM docker.m.daocloud.io/library/python:3.10-slim|g' Dockerfile

或手动修改为：

```
FROM docker.m.daocloud.io/library/python:3.10-slim
```

保存后重新执行agentarts launch命令。

- 执行agentarts invoke与智能体进行会话出现invoke_agent failed、Failed to create sandbox且平台运行时日志出现python: exec format error报错。

问题现象及原因

当前AgentArts服务部署于贵阳一区域，在创建智能体镜像时需要使用ARM64机器打镜像，使用X86机器打镜像会出现此报错。

```
2026-07-18 17:21:07.911 - agentarts.sdk.service.runtime_client - ERROR - invoke_agent failed: status=504, data=None error=HTTP 504
invoke_agent failed: status=504, data=None error=HTTP 504
ERROR: invoke_agent failed (HTTP 504): HTTP 504
(venv) root@IZ0J17ue115gdff15hqsuZ:~/aa-sdk# agentarts invoke {'message': '你好，请用一句话做个自我介绍。'}
Mode: Cloud
Agent: gootest
Session: 2da348c-1647-4562-a964-b397aa26712d
Endpoint: https://defaultgw-mymdcnjqm.cn-southwest-2.huaweicloud-agentarts.com
Auth Type: IAM
2026-07-18 17:25:36.226 - agentarts.sdk.service.runtime_client - ERROR - invoke_agent failed: status=403, data={'code': 403, 'data': None, 'message': 'Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: '{\"code\": \"NA.90001\", \"message\": \"[NA.90001] warmup health check overtime: warmup takes more than 4m0s\\\"}' (40a4766e-12e8-4870-825c-c035058ca0f1)'} error=Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: '{\"code\": \"NA.90001\", \"message\": \"[NA.90001] warmup health check overtime: warmup takes more than 4m0s\\\"}' (40a4766e-12e8-4870-825c-c035058ca0f1)'}
invoke_agent failed: status=403, data={'code': 403, 'data': None, 'message': 'Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: '{\"code\": \"NA.90001\", \"message\": \"[NA.90001] warmup health check overtime: warmup takes more than 4m0s\\\"}' (40a4766e-12e8-4870-825c-c035058ca0f1)'}
error=Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: '{\"code\": \"NA.90001\", \"message\": \"[NA.90001] warmup health check overtime: warmup takes more than 4m0s\\\"}' (40a4766e-12e8-4870-825c-c035058ca0f1)
ERROR: invoke_agent failed (HTTP 403): Failed to create sandbox: can not create sandbox of template in Error status, should in [Initialized Ready] status, detail info: '{\"code\": \"NA.90001\", \"message\": \"[NA.90001] warmup health check overtime: warmup takes more than 4m0s\\\"}' (40a4766e-12e8-4870-825c-c035058ca0f1)
(venv) root@IZ0J17ue115gdff15hqsuZ:~/aa-sdk#
```

基本信息

版本

日志

标签

基本信息

日志流

AgentArts

日志明细

近1小时

选择属性筛选，或输入关键字搜索

时间

级别

内容

2026-07-18 17:35:24 GMT+08:00

INFO

exec /usr/local/bin/python: exec format error

解决方案

请更换为ARM64机器制作智能体镜像。

- ECS DNS 解析失败

问题现象

在ECS上运行python main.py本地测试时，AgentArts公网域名无法解析，报错NameResolutionError: Failed to resolve。涉及的域名包括（具体域名因region和实例不同而异，请在 AgentArts 控制台查看）：

- memory-<region>.huaweicloud-agentarts.com（记忆库服务）
- gateway-<网关名称>-<网关实例ID>.<region>.huaweicloud-agentarts.com（网关）
- defaultgw-<网关实例ID>.<region>.huaweicloud-agentarts.com（代码解释器）

根因分析

AgentArts域名是公网域名，但ECS默认只配置了华为云内网DNS（100.125.x.x）。内网DNS对这些公网域名返回NXDOMAIN（域名不存在）。此外，ECS使用systemd-resolved管理DNS，eth0网卡的+DefaultRoute标志会用DHCP分配的内网DNS覆盖全局DNS配置，导致即使修改了/etc/systemd/resolved.conf也无法生效。

解决方案

绕过systemd-resolved，直接写静态/etc/resolv.conf:

```
# 删除 systemd-resolved 的符号链接
sudo rm /etc/resolv.conf

# 写入静态 DNS 配置（单行命令，不用 heredoc）
echo -e "nameserver 114.114.114.114\nnameserver 8.8.8.8\nnameserver 100.125.1.250\nsearch\nopenstacklocal" | sudo tee /etc/resolv.conf
```

DNS 服务器选择说明：

- 114.114.114.114（中国公共DNS）作为首选——国内访问延迟低，解析速度快
- 8.8.8.8（Google 公共DNS）作为备选——但国内访问有间歇性超时，不能作为首选
- 100.125.1.250（华为云内网DNS）——用于解析华为云内部域名

验证方式：

修改后用 nslookup（不带DNS服务器参数）逐个测试AgentArts域名，全部返回IP地址即为成功。

更多业务场景

场景配置说明

本智能体采用解耦架构设计：工具动态发现、配置通过环境变量注入、System Prompt 独立定义。移植至其他业务场景时，仅需调整少量配置项，核心代码无需改动。

修改类别	具体内容
必须修改	System Prompt — 定义智能体角色、能力描述与工作原则。当前为出行规划场景定制，移植时需根据新场景重写。
必须修改	.env 中的Gateway配置 — 若新场景接入不同的MCP Gateway（如数据库查询工具集），需更新GATEWAY_ENDPOINT与GATEWAY_INBOUND_TOKEN。工具采用动态发现机制，业务代码无需修改。
建议修改	main.py引导文本 — print_guide() 中的场景描述与建议对话流程。属界面展示层，不影响核心逻辑。
建议修改	.env模型配置 — 若新场景对推理能力有更高要求，可调整MODEL_NAME（如 deepseek-v4-flash）。代码层自动适配。
可选修改	追加本地工具 — execute_python 具备通用性，建议保留。如需扩展其他本地工具（如发邮件、文件操作），定义后追加StructuredTool即可。

典型业务场景

- 客服助手
重写System Prompt为客服话术规范 → MCP Gateway替换为工单/订单查询工具
→ 保留execute_python用于退款金额计算 → 其余配置无需调整
- 数据分析助手
重写System Prompt为数据分析流程 → MCP Gateway替换为SQL查询/数据导出
工具 → 保留execute_python用于统计分析 → 其余配置无需调整
- 代码审查助手
重写System Prompt为审查标准 → MCP Gateway替换为代码仓库/CI状态查询工
具 → 保留execute_python用于复杂度/覆盖率计算 → 其余配置无需调整

4 运行时

4.1 运行时概述

智能体运行时是AgentArts提供的安全隔离、框架无关的智能体托管执行环境，专为生产级智能体设计，提供企业级的弹性伸缩与权限治理能力，帮助开发者专注于业务逻辑，无需关心底层基础设施的运维。

用户将Agent代码打包为镜像，部署到运行时后即可通过API调用。

运行时解决的核心问题：

- Agent代码写好了，部署到哪里跑？
- 怎么保证多用户并发时互不干扰？
- 怎么弹性伸缩、版本管理、灰度发布？

运行时提供开箱即用的答案。

功能优势

- 安全隔离运行环境
 - 安全沙箱：基于安全沙箱技术实现microVM隔离，提供企业级运行时可靠性及安全保障。
 - 框架无关性：支持基于Agent框架（LangChain、LangGraph、Google ADK等）开发的智能体部署。
 - 托管执行环境：安全隔离、与框架和模型无关的托管运行环境，确保智能体在独立、受控条件下运行。
- 企业级弹性与伸缩
 - 自动弹性伸缩：支持基于业务流量的自动扩缩容能力。
 - 高并发承载：从容应对流量洪峰，保障服务稳定性。
 - 大规模支持：单账号可创建1000个智能体运行时，单运行时支持1000个版本。
- 精细化权限治理
 - 多维度身份认证：提供IAM、OAuth 2.0、API Key三种认证方式。
 - 安全委托机制：支持IAM委托，使智能体能安全地代表用户与外部系统交互。

- 灵活网络配置：支持公网/私网访问，保障网络环境安全。
- 灵活部署方式
 - 支持控制台可视化部署和SDK命令行部署。
 - 版本化管理，支持多版本管理和历史版本回溯。
 - 平滑镜像更新，更新过程有序可控。
- 完善的可观测性
 - 日志监控：集成华为云云日志服务（LTS），实时查看运行时日志。
 - 健康检查：支持自定义健康检查端点，提供HEALTHY、HEALTHY_BUSY、UNHEALTHY三种状态。
- 灵活访问控制
 - 多访问方式：每个运行时最多创建10个访问方式，支持绑定不同版本。
 - 文件传输：支持开启文件上传下载API接口。
 - 入站协议：支持HTTP和MCP两种入站协议。
- 丰富的运行时配置
 - 环境变量：支持表单编辑和JSON格式两种方式配置环境变量。
 - 启动命令：支持自定义启动命令，最多配置10个命令。
 - 存储扩展：支持SFS Turbo、会话存储和OBS三种存储类型。
 - 标签管理：支持资源标签分类管理，最多添加 20 个标签。
- 完整的API生态
 - RESTful API：提供标准RESTful API支持业务系统集成。
 - 完整SDK：提供Runtime、Tools、Memory、Identity、Gateway等多维度SDK。
 - CLI工具：命令行工具支持部署、调用、状态查询、资源销毁等全生命周期管理。
 - OpenAPI支持：提供完整的OpenAPI接口文档，支持二次开发。

与 Managed Agent 的关系

维度	Managed Agent	运行时
上手方式	配置驱动，零代码	代码驱动，打包镜像
基础设施	全托管	用户管理版本/实例
定制灵活度	通过配置项调整	代码层面完全可控
适用场景	快速验证/标准场景	深度定制/复杂逻辑

镜像制作约束与限制

为了让您的自定义高代码智能体（如基于LangChain开发的代码）能够在AgentArts运行时中被正确调度和访问，您制作的Docker镜像必须满足以下规范：

1. 请使用ARM64系统制作的Agent镜像，使用X86系统制作的镜像在调用智能体运行时调用失败。

2. 兼容AgentRun进站协议
- AgentArts平台通过标准的AgentRun协议与应用程序进行通信。要求如下：
监听指定端口：默认监听8080端口。
实现标准接口：
- a. HTTP协议：需提供 POST /invocations 端点，以适配传统请求响应模式。
b. WebSocket协议（可选）：需提供 /ws 端点，以实现实时双向流式通信。

📖 说明

推荐使用AgentArts SDK。SDK内置了协议转换能力，您只需关注核心业务逻辑，无需手动编写路由和协议接口。
该SDK提供AgentArtsRuntimeApp方法，用于将您的智能体逻辑封装为标准HTTP服务器，并暴露以下接口：

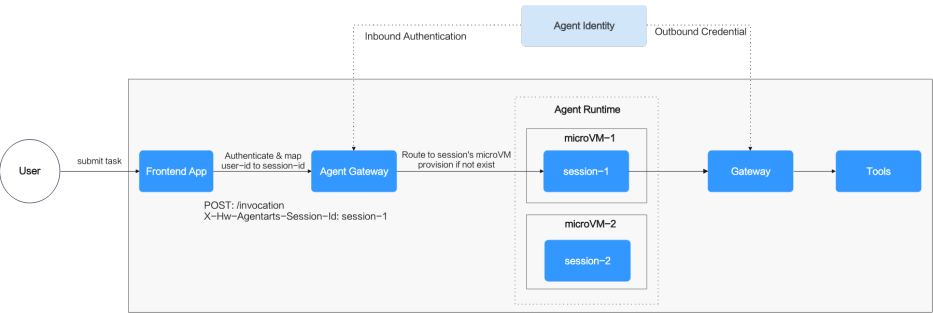
- POST /invocations - 用于智能体调用端点
- WS /ws - 用于流式传输的WebSocket端点

3. 无状态与本地磁盘约束
- 智能体部署在分布式弹性伸缩集群，在沙箱空闲触发资源释放时，本地磁盘数据全部丢失。
- 禁止依赖本地文件系统：安全沙箱随时可能被销毁或水平扩容，写入本地磁盘（如 SQLite、本地日志文件、本地缓存）的数据会丢失。
 - 状态持久化方案：请使用AgentArts提供的记忆库（Memory）组件进行上下文持久化，或使用华为云SFS Turbo弹性文件系统或OBS存储实体文件。

4.2 工作原理

本文介绍AgentArts运行时的内部架构和运行机制，帮助您理解一个请求从进入运行时到返回结果的完整过程。运行时是安全隔离、框架无关的托管执行环境，其核心是入站网关与安全沙箱的分层协作。

整体架构



典型的运行时架构和交互流程：

Agent开发者：

1. 使用任意Agent开发框架完成Agent开发，构建Docker镜像，发布到SWR 镜像仓库。
2. 在AgentArts平台基于Agent镜像创建运行时，获取访问方式。

3. 配置访问方式到前端应用。

Agent用户：

1. 在前端应用（Frontend App）完成业务登录。
2. 通过前端应用提交Agent任务。

前端应用（由开发者开发、部署、配置）：

1. 完成用户任务请求的认证和鉴权，提取user-id，并映射到运行时的session-id。
2. 调用运行时的访问方式，调用/invoke接口，header中需携带session-id。
3. Agent Gateway接收到Frontend的请求后进行认证，提取session-id，按照session-id进行路由：
 - a. 如果session-id存在对应的活跃microVM，直接转发路由。
 - b. 如果session-id找不到对应的microVM，唤醒新的microVM，然后转发路由。

核心概念

智能体运行时（Agent Runtime）：智能体的核心逻辑、决策和编排运行的环境，可用于托管智能体代码或者MCP工具，管理它们的生命周期。

版本（Versions）：智能体运行时的不可变版本，每次编辑运行时配置都会产生新版本。

访问方式（Endpoints）：智能体运行时的访问方式，可指向某一个具体版本，也可按权重指向多个版本，实现蓝绿发布，灰度发布

会话（Sessions）：智能体运行时的一个独立交互上下文，由请求头的session-id唯一标识。由前端应用负责业务对象与Session的映射，如需按用户隔离，配置user-id映射到一个session-id

4.3 进站协议

4.3.1 HTTP

介绍代理应用中HTTP协议的实现要求，基于HTTP搭建REST API接口以适配传统请求响应模式，同时部署WebSocket接口实现实时双向流式通信。

说明

HTTP（/invocations）和WebSocket（/ws）端点可部署在同一个容器中并使用8080端口，单个代理服务可同时兼容传统API交互与实时双向流式通信。

镜像要求

需满足以下容器化应用：

- **Host：**0.0.0.0
- **端口：**基于HTTP代理通信的标准端口8080
- **系统：**ARM64 容器

路径要求

- **/invocations - POST**

主要的代理交互端点，提供JSON输入和JSON/SSE输出。

- **目的：**接收来自用户或应用的请求，并通过代理业务逻辑进行处理。
- **使用场景**
 - 用户互动与对话
 - 与外部系统的API集成
 - 多请求的批量处理
 - 长期运行操作的实时流响应

请求格式示例

```
Content-Type: application/json
{
  "prompt": "What color do you like?"
}
```

响应格式

可以根据使用场景使用以下两种格式进行响应：

- **JSON 响应（非流式）**

目的：为请求提供完整的响应，适用于快速处理。

使用场景

- 简单的问答场景
- 确定性计算
- 快速数据查询
- 状态确认

JSON 响应格式示例

```
Content-Type: application/json
{
  "response": "I like red"
}
```

- **SSE响应（流式传输）**

服务器发送事件（SSE）能够实时传递流式响应。

目的：支持增量响应交付，实现长期运营和提升用户体验。

使用场景

- 实时对话体验
- 渐进式内容生成
- 具有中间结果的长期计算
- 实时数据流与更新

SSE响应格式示例

```
Content-Type: text/event-stream
data: {"event": "response 1"}
```

```
data: {"event": "response 2"}
data: {"event": "response"}
```

- **/ws - WebSocket（可选）**

实现实时双向通信的主要WebSocket连接端点。

目的：接受WebSocket升级请求，并维护流代理交互的持久连接。

使用场景

- 实时对话接口
- 互动代理会话，提供即时反馈
- 双向通信的流式数据处理

连接建立

WebSocket连接以HTTP升级请求开始：

HTTP升级请求示例

```
GET /ws HTTP/1.1
Host: agent-endpoint
Connection: Upgrade
Upgrade: websocket
Sec-WebSocket-Version: 13
Sec-WebSocket-Key: 8ZJk2cV7L9aX3pR5sQ7tW2eY1bN4mD6hG8jK0fS9dA2=
X-Hw-Agentarts-Session-Id: session-uuid
```

WebSocket升级响应示例

```
HTTP/1.1 101 Switching Protocols
Connection: Upgrade
Upgrade: websocket
Sec-WebSocket-Accept: 8ZJk2cV7L9aX3pR5sQ7tW2eY1bN4mD6hG8jK0fS9dA2=
```

消息处理要求

- **连接接受：**调用await websocket.accept()建立连接。
- **消息接收：**支持基于应用需求的文本或二进制消息类型。
- **消息处理：**根据代理的业务逻辑处理接收到的消息。
- **响应发送：**发送适当的响应，使用send_text()或send_bytes()。
- **连接生命周期：**管理连接的建立、维护和终止。

消息格式

- **JSON格式（推荐）**

请求示例

```
{
  "prompt": "What color do you like?"
}
```

响应示例

```
{
  "response": "I like red"
}
```

- **纯文本格式**

使用场景

- 多模态代理交互
- 文件上传与下载
- 压缩数据传输

- 二进制协议数

WebSocket协议过程

- 连接建立

- **HTTP握手**：客户端发送WebSocket升级请求
- **升级响应**：代理接受并返回101交换协议
- **WebSocket激活**：双向通信开始
- **会话绑定**：将连接与会话标识符关联

- 消息交换

- 连续循环**：实现消息监听环路
- 消息处理**：异步处理消息
- 响应生成**：发送合适的回复
- 错误处理**：管理异常和连接问题

- **/ping - GET**

目的：验证代理是否已运行并准备好处理请求。

使用场景：服务监控以检测和修复问题。

响应格式

返回状态代码，显示代理健康状况：

- **类型**：application/json
- **HTTP 状态码**：
 - Agent启动后，ping接口不可用
 - 表现：端口无法连接，或者连接后无法返回应答。
 - 状态：Initing
 - 含义：Agent正在启动，尚未准备好处理请求。
 - Agent启动中，ping接口已经可用，返回Initing状态
 - HTTP返回状态码：200
 - 应答返回的状态：{"status": "Initing"}
 - 含义：Agent正在初始化，暂时未完成，还没有就绪。
 - Agent就绪，初始化完成，ping返回Healthy状态
 - HTTP返回状态码：200
 - 应答返回的状态：{"status": "Healthy"}
 - 含义：Agent健康，当前没有异步后台任务。
 - 空闲超时：如果一个Agent连续处于Healthy (Idle)状态超过空闲超时时间（例如15分钟），则应该发起删除sandbox的流程。
 - Agent启动了后台异步任务，ping返回HealthyBusy状态
 - HTTP返回状态码：200
 - 应答返回的状态：{"status": "HealthyBusy"}
 - 含义：Agent健康，有后台异步任务正在运行。这对于支持 Agent 运行工作时间远超过空闲超时的长任务至关重要。

v. Agent不健康

- 表现：ping接口无法连接，或者无应答，或者返回HTTP 200之外的状态码。
- 状态：不健康
- 含义：Agent不健康，多次健康检查失败则会发起删除sandbox的流程。

示例 Ping 响应格式

```
{  
  "status": "Healthy"  
}
```

4.3.2 MCP

MCP（Model Context Protocol）是一套标准化协议，用于让AI模型安全、高效地对接外部工具、数据源与服务，统一上下文交互与能力调用规范。

协议实现要求

MCP服务器可实现以下协议要求：

- **传输**：流式HTTP传输，默认情况下，使用无状态模式（stateless_http=True）以兼容AgentArts的会话管理和负载均衡。
- **会话管理**：平台会自动添加Mcp-Session-Id头信息，以实现会话隔离。在无状态模式下，服务器需支持无状态操作，以免拒绝平台生成的Mcp-Session-Id头信息。

MCP会话管理：

模型上下文协议（MCP）使用Mcp-Session-Id头来管理会话状态并路由请求。

无状态MCP：

- 平台生成Mcp-Session-Id，并将其包含在请求中，发送至MCP服务器。
- MCP服务器必须接受平台提供的会话ID（不得拒绝）。
- 平台在响应中将相同的Mcp-Session-Id返回给客户端。
- 客户端必须在所有后续的微虚拟机亲和性请求中包含此会话ID。

镜像要求

MCP服务器满足以下规范的容器化要求：

- **Host**：0.0.0.0
- **端口**：MCP服务器通信的标准端口（不同于HTTP协议）8000
- **系统**：ARM64 容器

路径要求

/mcp - POST

目的

接收MCP RPC消息并通过代理的工具能力进行处理，完整传递调用智能体运行时的API负载与标准MCP协议消息。

响应格式

基于JSON-RPC的请求/响应格式，支持将application/json和text/event-stream作为响应的内容类型。

使用场景

- 工具调用与管理：调用和管理外部工具。
- 代理能力发现：发现和使用代理的能力。
- 资源访问与操作：访问和操作外部资源。
- 多步骤代理工作流程：支持多步骤的工作流程。

4.4 权限与访问控制

4.4.1 委托介绍

委托是华为云统一身份认证服务（IAM）提供的一种信任关系机制。如果您在华为云购买了多种云资源，其中一种云资源希望让更专业、高效的公司、账号或者云服务来代运维，您可以通过IAM的委托功能，与代运维公司、账号或者云服务创建委托关系，实现安全高效的资源代运维工作。被委托的账号、公司或云服务可以根据权限代替您进行资源运维工作。

表 4-1 委托类型

类型	说明
普通账号委托	通过委托信任功能，将自己的资源操作权限委托给其他华为云账号，被委托的账号可以根据权限代替您进行资源运维工作。
云服务委托	由于华为云各服务之间存在业务交互关系，一些云服务需要与其他云服务协同工作，需要您创建云服务委托，将操作权限委托给该服务，让该服务以您的身份使用其他云服务，代替您进行一些资源运维工作。

功能优势

- **灵活管理**：支持向账号和云服务委托权限，满足不同场景下的资源管理需求。企业可以根据业务需要，将部分业务委托给更专业的其他企业来完成。
- **安全可靠**：创建委托并授权后，被委托方只能按照权限使用资源，保障了账号、数据和资源的安全。委托方可以随时通过简单操作建立、修改或取消委托关系。

更多关于委托的详细介绍，请参考[委托概述](#)。

AgentArts运行时涉及两个委托，分别用于不同的场景：

- AgentArtsRuntimeDeploymentAgency（服务部署委托）

- 用户运行时委托（智能体身份委托）

AgentArtsRuntimeDeploymentAgency（服务部署委托）

属性	说明
委托名称	AgentArtsRuntimeDeploymentAgency（固定名称）
对应策略	AgentArtsRuntimeDeploymentAgencyPolicy
用途	委托给沙箱服务，用于下载用户镜像、挂载SFS Turbo共享文件存储
是否必须	必须。该委托由服务内部使用，是AgentArts运行时正常工作的前提条件，不可删除
用户感知	对用户不感知，由服务内部自动使用

AgentArtsRuntimeDeploymentAgencyPolicy包含以下权限：

权限 Action	权限级别	说明
swr::createAuthorizationToken	写入	授予共享版仓库生成新的临时登录指令的权限
swr:repo:download	读取	授予共享版仓库下载镜像的权限
sts::createServiceBearerToken	写入	授予权限获取一个绑定至某服务的 Bearer Token
sfsturbo:shares:getShare	读取	授予查询弹性文件系统的权限

各Action详细说明：

swr::createAuthorizationToken

- 功能：生成SWR（容器镜像服务）共享版仓库的临时访问凭证
- 使用场景：AgentArts沙箱服务在拉取用户镜像前，需要先获取SWR的临时登录指令

swr:repo:download

- 功能：从SWR共享版仓库下载容器镜像
- 使用场景：沙箱服务通过该权限下载用户上传的智能体运行时镜像

sts::createServiceBearerToken

- 功能：获取绑定至特定服务的 Bearer Token
- 使用场景：沙箱服务通过STS获取服务级令牌，用于跨服务认证

sfsturbo:shares:getShare

- 功能：查询SFS Turbo弹性文件系统信息
- 使用场景：沙箱服务在挂载SFS Turbo共享存储前，需要查询文件系统的状态和配置信息

用户运行时委托（智能体身份委托）

属性	说明
委托名称	用户自定义（服务开通时自动创建默认委托 DefaultAgentArtsRuntimeAgency）
推荐策略	AgentArtsCoreRunRuntimeIdentityAgencyPolicy + AgentArtsCoreRunRuntimeOpsAgencyPolicy
用途	在智能体运行时内部代表用户身份与其他华为云云服务交互
是否必须	推荐配置。使用默认委托可开箱即用；如需自定义，可在创建运行时手动创建
创建方式	服务开通时自动创建 DefaultAgentArtsRuntimeAgency；也可在创建运行时手动创建，信任主体类型选择“云服务”，云服务搜索 service.WorkloadSandboxMetadata

关于运行时委托的创建和配置，请参考[通过控制台部署智能体运行时](#)。

策略一：AgentArtsCoreRunRuntimeIdentityAgencyPolicy

该策略用于获取智能体运行时出站认证凭据的相关权限。

权限 Action	权限级别	说明
agentIdentity::getResourceApiKey	读取	授予获取与API密钥凭证提供者关联的API密钥的权限
agentIdentity::getResourceOAuth2Token	读取	授予通过OAuth2 两方/三方授权流程获取访问令牌，以访问外部资源的权限
agentIdentity::getResourceStsToken	读取	授予从STS凭证提供者获取IAM临时凭证的权限
csms:secret:getVersion	读取	授予查询指定凭据版本的信息和其明文凭据值的权限
kms:cmk:decryptDataKey	写入	授予解密数据密钥的权限

各 Action 详细说明：

- agentIdentity::getResourceApiKey：从Agent Identity服务获取API Key，用于以API Key认证方式访问外部云服务

- agentIdentity::getResourceOauth2Token：通过OAuth2流程（M2M或USER_FEDERATION）获取访问令牌，用于访问支持OAuth2认证的外部资源
- agentIdentity::getResourceStsToken：从STS凭证提供者获取IAM临时凭证（AK/SK/SecurityToken），用于以华为云用户身份访问其他云服务
- csms:secret:getVersion：查询凭据管理服务中存储的凭据明文值，用于获取预先配置的密钥和证书
- kms:cmk:decryptDataKey：解密凭据管理服务中加密存储的数据密钥，配合CSMS凭据读取使用

策略二：AgentArtsCoreRunRuntimeOpsAgencyPolicy

该策略用于智能体运行时可观测性服务相关权限。

权限 Action	权限级别	说明
apm:application:get	读取	授予获取应用信息数据的权限
aom:metric:list	列举	授予权限以查询指标项
aom:icmgr:get	读取	授予权限以获取采集组件（ICAgent）版本信息

各Action详细说明：

- apm:application:get：从APM（应用性能管理服务）获取应用信息数据，用于智能体运行时的性能监控
- aom:metric:list：从AOM（应用运维管理服务）查询监控指标项，用于采集和上报运行时指标
- aom:icmgr:get：获取ICAgent采集组件版本信息，用于运行时可观测性管理

如何使用两个委托

AgentArtsRuntimeDeploymentAgency

该委托由AgentArts服务内部直接使用，对用户完全透明。用户无需进行任何额外配置或代码编写。

服务会在以下场景中自动使用该委托：

- 下载用户镜像：当用户创建或更新智能体运行时，沙箱服务使用该委托从SWR下载用户指定的容器镜像
- 挂载SFS Turbo：当用户配置了SFS Turbo存储时，沙箱服务使用该委托查询并挂载弹性文件系统

 说明

该委托是AgentArts运行时正常工作的前提条件，请勿删除。如果该委托被删除，将导致运行时创建失败或镜像下载异常。

用户运行时委托（智能体身份委托）

该委托用于在智能体运行时内部代表用户身份访问其他华为云服务。AgentArts运行时SDK提供了 MetadataProvider来自动获取该委托对应的临时凭据，用户可以使用该凭据来访问其他云服务。

工作原理

智能体运行时 → MetadataProvider → 元数据服务 → 获取AK/SK/SecurityToken → 使用凭据访问其他云服务

MetadataProvider是AgentArts SDK凭证提供链中的一环。当智能体运行在AgentArts运行时环境中时，SDK会自动通过元数据服务获取委托对应的临时凭据（AK/SK/SecurityToken），无需用户手动配置AK/SK。

SDK凭证获取优先级

AgentArts SDK的凭证提供链（CredentialProviderChain）按以下顺序尝试获取凭证：

1. EnvCredentialProvider：从环境变量读取（HUAWEICLOUD_SDK_AK / HUAWEICLOUD_SDK_SK）
2. ProfileCredentialProvider：从配置文件读取
3. MetadataProvider：从 AgentArts 运行时元数据服务获取（仅在运行时环境中可用）

代码示例

示例 1：直接使用MetadataProvider获取凭据

```
from agentarts.sdk.utils.metadata import create_credential

# 自动按优先级获取凭证：环境变量 → 配置文件 → 元数据服务
# 在 AgentArts 运行时环境中，会自动通过 MetadataProvider 获取委托凭据
credentials = create_credential()

# 使用获取到的凭据访问其他华为云服务
# credentials 包含 AK、SK 和 SecurityToken
print(f"AK: {credentials.ak}")
print(f"SK: *** (已隐藏)")
print(f"Security Token: {credentials.security_token}")
```

示例 2：使用 @require_sts_token 装饰器获取 STS 临时凭据

```
from agentarts.sdk import require_sts_token

# 通过装饰器自动获取 STS 凭证并注入到函数参数中
@require_sts_token(
    provider_name="huaweicloud-iam", # 凭证提供者名称
    agency_session_name="my-agent-session" # 会话名称
)
def access_cloud_resource(sts_credentials=None):
    """使用 STS 凭证访问华为云服务"""
    print(f"Access Key ID: {sts_credentials.access_key_id}")
    print(f"Secret Access Key: *** (已隐藏)")
    print(f"Security Token: {sts_credentials.security_token}")
    print(f"Expiration: {sts_credentials.expiration}")

    # 在此处使用 sts_credentials 访问其他华为云服务
    # 例如：访问 OBS、RDS、ECS 等

# 调用函数，STS 凭证会自动注入
access_cloud_resource()
```

示例 3：使用 @require_api_key 装饰器获取 API Key

```
from agentarts.sdk import require_api_key

@require_api_key(
```

```
    provider_name="my-api-key-provider" # 凭证提供者名称
)
def call_external_api(api_key=None):
    """使用 API Key 调用外部服务"""
    print(f"API Key: {api_key}")

    # 在此处使用 api_key 访问外部 API
    headers = {"Authorization": f"Bearer {api_key}"}
    # response = requests.get("https://api.example.com/data", headers=headers)

call_external_api()
```

示例 4：使用 @require_access_token 装饰器获取 OAuth2 令牌

```
from agentarts.sdk import require_access_token

@require_access_token(
    provider_name="my-oauth2-provider",
    auth_flow="M2M", # 机器对机器流程，也可使用 "USER_FEDERATION"
    scopes=["read", "write"]
)
def access_oauth_resource(access_token=None):
    """使用 OAuth2 令牌访问受保护的资源"""
    print(f"Access Token: {access_token}")

    # 在此处使用 access_token 访问受 OAuth2 保护的资源
    headers = {"Authorization": f"Bearer {access_token}"}
    # response = requests.get("https://api.example.com/protected", headers=headers)

access_oauth_resource()
```

示例 5：在智能体代码中集成使用

```
from agentarts.sdk import AgentArtsRuntimeApp, RequestContext
from agentarts.sdk import require_sts_token

app = AgentArtsRuntimeApp()

@app.route("/invoke")
@require_sts_token(
    provider_name="huaweicloud-iam",
    agency_session_name="runtime-agent"
)
async def handle_invoke(request, sts_credentials=None):
    """智能体入口函数，自动获取 STS 凭证访问其他云服务"""

    # 使用 sts_credentials 访问用户的其他华为云资源
    # 例如：读取 OBS 中的数据、查询 RDS 数据库等

    ak = sts_credentials.access_key_id
    sk = sts_credentials.secret_access_key
    token = sts_credentials.security_token

    # 示例：使用凭证访问 OBS
    # from obs import ObsClient
    # obs_client = ObsClient(
    #     access_key_id=ak,
    #     secret_access_key=sk,
    #     security_token=token,
    #     server="obs.cn-southwest-2.myhuaweicloud.com"
    # )
    # obs_client.getObject("my-bucket", "my-object")

    return {"status": "success", "message": "资源访问成功"}

if __name__ == "__main__":
    app.run()
```

关键说明

1. 元数据服务地址：MetadataProvider通过元数据服务的 v1/metadata/securitykey 接口获取凭据，该服务仅在AgentArts运行时环境内可访问
2. 凭据有效期：通过元数据服务获取的临时凭据具有有效期（expires_at），SDK会自动处理凭据的刷新
3. 本地开发：在本地开发环境中（非运行时环境），MetadataProvider将无法获取凭据，SDK会提示配置环境变量HUAWEICLOUD_SDK_AK 和 HUAWEICLOUD_SDK_SK
4. 默认委托：如果未自定义委托，服务开通时自动创建的DefaultAgentArtsRuntimeAgency已包含推荐的两个策略，可直接开箱使用

4.4.2 入站身份认证

为什么需要入站身份认证

智能体运行时作为一个独立隔离的托管运行环境，需要对外提供 API 接口以接收外部调用请求。然而，并非所有请求都应该被无条件接受——如果运行时不加验证地处理任何传入请求，将面临以下安全风险：

- 未授权访问：任何人只要知道运行时的访问地址，就可以调用智能体，消耗计算资源，甚至获取敏感数据。
- 身份不可追溯：无法区分请求来自哪个用户或系统，难以进行访问控制和审计。
- 恶意调用：攻击者可以伪造请求，对运行时进行 DDoS 攻击或数据窃取。

入站身份认证的核心目标是：对每一个到达智能体运行时的请求进行身份验证，确保请求来自可信的调用方，并根据调用方身份进行授权控制。

AgentArts 运行时的身份认证理念

在AI智能体场景中，身份认证面临比传统 Web 服务更复杂的挑战：智能体不仅需要被外部系统调用，还可能代表用户身份去访问其他云服务。因此，AgentArts运行时将身份认证分为两个维度：

- 入站认证：验证调用方的身份，确保只有授权的调用方能够访问智能体运行时的 API 接口。
- 出站认证：智能体运行时内部代表用户身份访问其他华为云服务时使用的凭据（通过运行时委托实现）。

入站认证作为运行时的“守门人”，对每个请求独立验证，要求所有访问尝试都必须提供显式的身份证明。无论是来自华为云内部系统的调用，还是来自外部第三方系统的调用，都必须通过入站认证后才能被智能体处理。这种“零信任”的安全模型确保了：

- 每个请求都经过验证：不存在“默认信任”的调用方
- 身份可追溯：通过认证凭据可追踪到具体的调用方
- 访问可控制：可根据不同认证方式配置不同的访问策略
- 合规可审计：所有认证请求都有记录，满足安全合规要求

支持的认证方式

AgentArts运行时入站身份认证支持以下三种方式：

认证方式	认证凭据	请求头格式
IAM 认证（AK/SK 签名认证）	AK + SK通过签名算法生成签名	Authorization: V11-HMAC-SHA256 Access={AK}, SignedHeaders=..., Signature=...
API Key认证	API Key字符串	Authorization: Bearer {API Key}
OAuth 2.0认证	JWT Token	Authorization: Bearer {JWT Token}

IAM 认证（AK/SK 签名认证）

IAM 认证基于华为云的AK/SK签名算法。调用方使用自己的 AK（Access Key）和 SK（Secret Access Key）对请求进行签名，将签名信息放入 Authorization 请求头中。运行时收到请求后，通过验证签名的正确性来确认调用方身份。

签名过程包含以下步骤：

- 1. **构造规范请求：**将 HTTP 方法、URI、查询参数、消息头、消息体等按照规范格式拼接成规范请求字符串，并计算其 SHA-256 哈希值。
- 2. **创建待签字符串：**将签名算法标识、请求时间、规范请求的哈希值拼接成待签字符串。
- 3. **计算签名：**使用 SK 作为密钥，通过 HMAC-SHA256 算法对待签字符串计算签名。
- 4. **添加签名信息到请求头：**将 AK、签名的消息头列表和签名值组装到 Authorization 请求头中。

完整的签名请求示例：

```
GET /v1/runtimes/my-runtime/invocations?input=hello HTTP/1.1
Host: agentarts.cn-southwest-2.myhuaweicloud.com
Content-Type: application/json
X-Sdk-Date: 20260909T033655Z
Authorization: V11-HMAC-SHA256 Access=QTWA***KYUC, SignedHeaders=content-type;host;x-sdk-date,
Signature=f12f84a5ecf9eff3206499c4a55b13d1adad745dc8624a2e31f15c6b381d5b80
```

支持的签名算法

智能体运行时数据面接口（即 /runtimes/{runtime_name}/invocations 等接口）支持以下签名算法：

签名算法	说明
V11-HMAC-SHA256	基于 HMAC-SHA256 的签名算法，使用 SK 作为密钥对待签字符串进行哈希运算，支持 Region 级别的密钥派生
SDK-ECDSA-P256-SHA256	基于椭圆曲线数字签名算法（ECDSA P-256）的签名算法，使用非对称密钥对进行签名

说明

对于智能体运行时数据面接口，不支持对请求 body 体签名，仅对请求头和查询参数进行签名验证。其他 AgentArts 接口仍需对 body 体签名。

约束与限制

- AK/SK 签名认证方式仅支持消息体大小 12MB 以内
- AK/SK 既可以使用永久访问密钥，也可以使用临时访问密钥（STS），使用临时访问密钥时需额外携带 X-Security-Token 请求头
- API 网关会校验请求时间（X-Sdk-Date）与服务器时间的差值，超过 15 分钟的请求将被拒绝，请确保客户端时间同步
- 签名 SDK 仅提供签名功能，与各服务提供的 SDK 不同

API Key 认证

API Key 认证是一种轻量级的认证方式。调用方通过 AgentArts 智能体身份服务获取一个 API Key 字符串，在调用运行时接口时将其放入 Authorization 请求头中即可完成认证。

请求格式：

```
POST /v1/runtimes/my-runtime/invocations HTTP/1.1
Host: agentarts.cn-southwest-2.myhuaweicloud.com
Content-Type: application/json
Authorization: Bearer {你的API Key}

{"input": "你好"}
```

API Key 获取方式

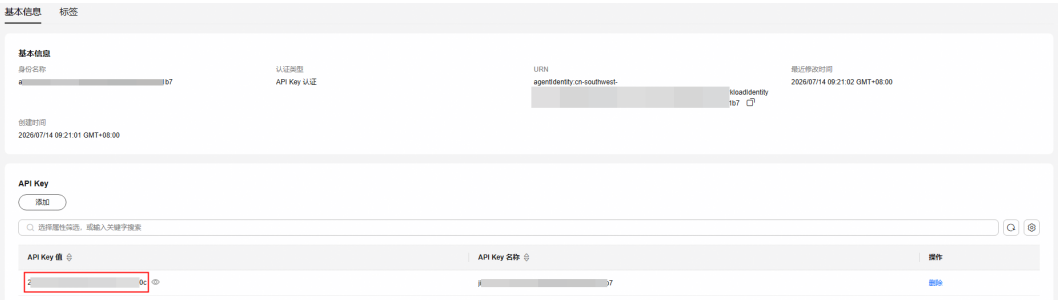
- 步骤1** 进入AgentArts智能体平台，在左侧菜单栏选择“托管与运行 > 智能体运行时”。
- 步骤2** 单击智能体运行时名称，进入智能体详情页面。
- 步骤3** 在“权限与访问控制”区域，访问URN。

图 4-1 访问 URN



- 步骤4** 进入AgentIdentity智能体身份服务，在基本信息页面获取API Key值并复制。

图 4-2 获取 API Key 值



步骤5 添加Authorization的请求头，对应的请求头的值为“Bearer 获取到的API Key”。

----结束

OAuth 2.0 认证

OAuth 2.0认证依赖第三方身份凭据提供商（如 Okta、Cognito 等）签发的JWT（JSON Web Token）作为认证凭据。调用方先从身份提供商处获取JWT Token，然后将Token放入Authorization请求头中调用运行时接口。运行时收到请求后，会验证JWT Token的签名、有效期和声明信息，确认调用方身份。

请求格式：

```
POST /v1/runtimes/my-runtime/invocations HTTP/1.1
Host: agentarts.cn-southwest-2.myhuaweicloud.com
Content-Type: application/json
Authorization: Bearer {JWT Token}

{"input": "你好"}
```

JWT Token获取

JWT Token 由第三方身份提供商签发，获取方式因提供商而异。以GitHub应用为例，可参考GitHub文档中“[生成JSON Web令牌 \(JWT\)](#)”章节获取JWT Token。

OAuth 2.0配置参数

在创建运行时选择 OAuth 2.0 认证后，需要配置以下参数：

参数	说明
Discovery URL	OAuth 发现服务器地址，需输入以 https:// 开头、/.well-known/openid-configuration 结尾的有效URL
允许的受众	用于认证OAuth 2.0 指定的受众是否与 Agent Identity运行时中指定的受众匹配或为其子集，最多100个
允许的客户端	用于认证OAuth 2.0指定的客户端标识符是否被允许访问，最多 100 个
允许的范围	仅当令牌包含此处配置的至少一个必须范围时才允许访问，最多100个
自定义声明	仅当令牌中的特定声明与预定义字符串值相匹配时才允许访问，最多100 个

适用场景

认证方式	适用场景	优势	劣势
IAM 认证（AK/SK 签名认证）	华为云内部系统调用、已接入华为云 IAM 的系统对接	安全性高，签名防篡改，凭据不直接传输，支持临时凭据	签名逻辑复杂，集成成本较高
API Key 认证	快速测试验证、轻量级集成、原型开发	集成简单，仅需一个请求头，开发效率高	安全性相对较低，API Key 明文传输，不适合生产环境高安全场景
OAuth 2.0 认证	对接外部系统、企业级应用集成、已有身份提供商的场景	支持标准化身份联邦，细粒度权限控制，支持用户委托授权	配置复杂，依赖外部身份提供商，需处理 Token 刷新

场景选择建议

- 场景一：华为云内部服务调用**推荐使用**IAM认证**。调用方使用自身的华为云 AK/SK 对请求签名，凭据不直接在网络上传输，安全性最高。适合 已经对接过华为云IAM的服务调用智能体运行时的场景。
- 场景二：开发调试与快速验证**推荐使用**API Key认证**。无需实现复杂的签名逻辑，只需在请求头中携带 API Key 即可快速调用运行时接口，适合开发调试和原型验证阶段。
- 场景三：企业系统对接** 推荐使用**OAuth 2.0认证**。企业已有 IdP（如 Okta、Cognito）时，可通过 OAuth 2.0 联邦认证实现单点登录和统一身份管理，无需为每个系统单独管理 AK/SK 或 API Key。
- 场景四：第三方 SaaS 集成**推荐使用**OAuth 2.0认证**。通过 OAuth 2.0 的授权码流程（3LO），用户可明确授权第三方应用访问智能体运行时，支持细粒度的范围控制和用户同意机制。

入站身份认证的配置方式

入站身份认证在创建智能体运行时配置，配置后不可修改。

配置步骤

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 智能体运行时”。
- 步骤3 单击“托管智能体”，进入运行时创建页面。
- 步骤4 在“权限与访问控制 > 入站身份认证”中选择认证方式：
 - IAM 认证：使用登录管理控制台时使用的IAM用户名进行认证。
 - API Key 认证：通过接口的访问密钥进行身份验证和授权。输入 API Key 的名称后，系统会自动创建并绑定API Key。

- OAuth 2.0 认证：将OAuth 2.0配置为入站身份认证。选择后需配置Discovery URL、允许的受众、允许的客户端、允许的范围等参数。

----结束

 说明

- 认证方式创建后不可修改：智能体运行时在初次创建时选择认证方式后，后续编辑无法修改认证方式。如果认证方式选择错误，需要重新创建一个新的智能体运行时（可使用“复制”功能快速创建相同配置的运行时）。
- 认证方式需与调用方匹配：调用方必须使用与运行时配置一致的认证方式，否则请求将被拒绝。

4.5 示例：部署第一个 Agent 到运行时

概述

将本地Agent代码投入生产，通常意味着一系列繁琐的工程化工作：手动编写路由、处理请求协议、构建Docker镜像，再自行寻找服务器完成部署。

AgentArts提供了一条从本地代码到云端服务的完整链路。本文将引导您将一个本地Agent代码托管到AgentArts运行时。

完成本文后，您将掌握：

- 如何将本地Agent代码封装为符合AgentArts平台规范的HTTP服务。
- 如何将封装好的服务制作成镜像并部署到AgentArts运行时。
- 如何调用已部署的Agent。

AgentArts提供两种部署方式，您可以根据实际情况选择：

表 4-2 Agent 部署方式

类别	方法A：手动制作镜像+控制台部署	方法B：运行时SDK一键部署
适合场景	已有成熟Agent镜像；需要精细控制配置；习惯可视化界面操作。	新项目快速验证；习惯命令行操作。
操作方式	手动构建镜像 → 推送 SWR → 控制台填写配置创建运行时。	通过运行时SDK自动完成镜像构建、推送和部署。

两种方式部署完成后，调用方式完全一致。本文前两步（准备代码、本地验证）为两种方法共用，无需重复操作。

图 4-3 操作流程



前置条件与环境准备

在开始之前，请确认以下准备工作已完成：

账号与服务：

- 已开通AgentArts服务，并在“管理中心 > 授权管理”页面完成依赖云服务授权。

本地环境：

- 操作系统：**推荐使用Linux Ubuntu 22.04 ARM64**，服务器可访问公网。
本文命令如无特殊说明，均以Ubuntu为例。为保证镜像构建与云端部署的兼容性，建议使用Ubuntu系统。
因此，为了减少部署失败和额外排查成本，建议直接使用Ubuntu 22.04 ARM64制作AgentArts镜像。
若使用CentOS、EulerOS等非Ubuntu系统，请将下命令中的apt替换为yum或dnf，并自行确认Docker 18.06+、Python 3.10+；本文不逐条展开各发行版安装差异。

 **注意**

需使用ARM64系统制作的Agent镜像，使用X86系统制作的镜像在调用智能体运行时调用失败。

可以使用华为云ECS服务购买服务器，购买时需要选择鲲鹏架构（ARM64），系统选择Ubuntu。

云服务器控制台 / 弹性云服务器 / 购买弹性云服务器

<

 **购买弹性云服务器**

快速购买 ^①

自定义购买 ^②

实例

+

 如何选择ECS实例规格?

规格类型选型

业务场景选型

CPU架构 ^③

x86计算

鲲鹏计算

- 安装Python：请确保Python 3.10及以上版本已安装。
大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version检查。如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

非Ubuntu/Debian系统请使用yum或dnf安装Python 3.10+及pip。
- 安装Docker：请确保Docker 18.06及以上版本已安装。如未安装，可以使用如下命令安装：

```
# 查询 Docker 版本
docker --version

# 安装Docker
sudo apt update
sudo apt install docker.io
```

安装完成后，建议立即配置国内镜像加速器。在后续执行agentarts launch构建镜像时，Docker需要拉取python:3.10-slim基础镜像，默认从docker.io官方仓库下载，可能超时失败。提前配置加速器可避免此问题。

执行以下命令创建Docker配置目录：

```
sudo mkdir -p /etc/docker
```

执行以下命令配置国内镜像源（直接复制并回车执行）：

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors":[
    "https://docker.m.daocloud.net",
    "https://dockerproxy.net",
    "https://mirror.baidubce.com"
  ]
}
EOF
```

依次执行以下命令重启Docker服务，使配置生效：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

📖 说明

华为云SWR基础版不支持OCI镜像格式，如果您使用的是Docker 27及以上版本，并且需要处理OCI镜像，可以通过设置环境变量来关闭OCI支持。

```
export DOCKER_BUILDKIT=0
# 或者
export BUILDKIT_USE_OCI_MEDIA_TYPES=0
```

- 执行以下命令安装SDK（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
# 安装依赖包
apt install python3.12-venv

# 创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate

# 安装sdk
pip install agentarts-sdk
```

若配置镜像加速器，后续在执行agentarts launch命令时，仍无法拉取python:3.10-slim，可参考本文结尾的“常见问题”中的“直接修改Dockerfile基础镜像地址”。

📖 说明

- 执行pip install agentarts-sdk命令下载缓慢、超时可以更改为使用如下命令
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
- 如果系统缺少python3-venv包，导致无法创建虚拟环境，请按照命令回显提示安装python3-venv包。

```
root@ecs-6732:~# python3 -m venv myenv
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

apt install python3.12-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

Failing command: /root/myenv/bin/python3
```

- 执行以下命令配置华为云凭证，获取华为云凭证请参考[认证鉴权](#)。
export HUAWEICLOUD_SDK_AK="your-access-key"
export HUAWEICLOUD_SDK_SK="your-secret-key"

步骤一：准备 Agent 代码

步骤1 创建项目目录。

在终端执行以下命令，创建项目目录并进入：

```
mkdir my-first-agent && cd my-first-agent
```

步骤2 配置环境变量。

在项目根目录下创建.env文件，用于存放模型调用所需的配置信息。

执行touch .env命令创建.env文件。

执行vi .env命令打开.env文件，写入以下内容，并将model_api_key替换为您实际的模型API Key。

```
# .env

# 必填，您的华为云 MaaS API 密钥
MODEL_API_KEY=model_api_key

# 模型名称
MODEL_NAME=deepseek-v4-flash

# 模型服务地址
MODEL_URL=https://api.modelarts-maas.com/openai/v1

# Agent 服务监听端口（可选，默认 8080）
AGENT_RUN_PORT=8080
```

步骤3 执行如下命令安装langchain、langgraph、langchain-openai。

```
pip install -U langchain langgraph langchain-openai
```

步骤4 编写Agent核心逻辑。

1. 执行touch agent.py命令，在项目根目录创建agent.py文件。
2. 执行vi agent.py命令修改文件，并写入以下内容：

```
# agent.py

import os
from dotenv import load_dotenv
from langchain_openai import ChatOpenAI
from langchain_core.messages import HumanMessage, SystemMessage

# 加载 .env 文件中的环境变量
load_dotenv()

class MyAgent:
    """基础对话智能体"""

    def __init__(self):
        # 初始化大模型客户端
        self.llm = ChatOpenAI(
            model=os.getenv("MODEL_NAME", "deepseek-v4-flash"),
            openai_api_key=os.getenv("MODEL_API_KEY"),
            openai_api_base=os.getenv(
                "MODEL_URL", "https://api.modelarts-maas.com/openai/v1"
            ),
        )
        self.system_prompt = (
            "你是一个友好、专业的 AI 助手，请用简洁清晰的语言回答用户的问题。"
        )

    def chat(self, message: str) -> str:
        """接收用户消息，调用大模型，返回回复内容"""
```

```
messages = [
    SystemMessage(content=self.system_prompt),
    HumanMessage(content=message),
]
response = self.llm.invoke(messages)
return response.content
```

```
# 创建 Agent 单例，供 app.py 引用
agent = MyAgent()
```

3. 验证Agent逻辑是否正常。

在终端执行以下命令，直接调用agent.py进行快速验证，确认模型调用链路没有问题：

```
python3 -c "from agent import agent; print(agent.chat('你好，请用一句话做个自我介绍。'))"
```

如果模型调用正常，终端会输出模型的回复内容。

如果此步骤报错，请优先排查以下问题：

- .env文件中的MODEL_API_KEY是否已替换为实际值。
- 当前网络环境是否可以访问api.modelarts-maas.com。

步骤5 封装为HTTP服务。

AgentArts运行时通过标准HTTP接口与您的Agent通信。本步骤使用AgentArts运行时SDK将Agent逻辑封装为符合平台规范的HTTP服务。

执行touch app.py命令，在项目根目录创建app.py文件。

执行vi app.py命令修改文件，并写入以下内容：

```
# app.py

import os
from typing import Dict, Any
from dotenv import load_dotenv
from agentarts.sdk import AgentArtsRuntimeApp, RequestContext
from agent import agent

load_dotenv()

app = AgentArtsRuntimeApp()

@app.entrypoint
async def handler(payload: Dict[str, Any], context: RequestContext = None) -> Dict[str, Any]:
    """
    AgentArts 平台标准 HTTP 暴露入口。
    - payload: 用户传入的完整请求体（dict 格式）
    - context: 请求上下文，包含 session_id 等平台注入信息
    """
    message = payload.get("message", "")
    if not message:
        return {"response": "请求体中缺少 message 字段", "status": "error"}

    try:
        reply = agent.chat(message)
        return {"response": reply, "status": "success"}
    except Exception as e:
        return {"response": f"执行出错: {str(e)}", "status": "error"}

if __name__ == "__main__":
    port = int(os.getenv("AGENT_RUN_PORT", 8080))
    app.run(host="0.0.0.0", port=port)
```

📖 说明

- AgentArtsRuntimeApp是AgentArts SDK提供的运行时封装类，负责处理平台与Agent之间的通信协议，以及健康检查接口的注册。
- @app.invocation_handler装饰器将handle_invocation函数注册为/invocations接口的处理逻辑，平台每次调用Agent时都会触发此函数。

----结束

步骤二：本地验证

在部署到云端之前，先在本地确认服务启动正常、接口响应符合预期。

步骤1 启动本地服务。

在正式推送镜像之前，先在本地模拟云端HTTP调用环境，验证Agent的接口封装是否正确、通信是否正常，可以大幅降低因接口问题导致云端部署后才发现错误的调试成本。

执行python app.py启动http server，执行以下命令调用验证Agent的HTTP接口是否已经被正确封装且能正常通信。

执行python app.py回显效果如下。

```
(venv) root@ecs-7097:~/demo01/my-first-agent# python app.py
2026-09-07 10:59:35,084 - agentarts.runtime.app - INFO - Starting AgentArts runtime on 0.0.0.0:8080 (http)
INFO: Started server process [23360]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: Uvicorn running on http://0.0.0.0:8080 (Press CTRL+C to quit)
```

步骤2 验证健康检查接口。

打开一个新的终端窗口（保持原窗口运行），执行以下命令，验证 /ping 健康检查接口是否正常：

```
curl http://localhost:8080/ping
```

返回类似{"status": "Healthy"}内容说明服务已正常启动，平台的健康探查可以通过。

步骤3 验证对话接口。

在新终端中执行以下命令，验证/invocations对话接口是否正常：

```
curl -X POST http://localhost:8080/invocations \
-H "Content-Type: application/json" \
-d '{"message": "你好，请用一句话做个自我介绍。"}'
```

收到模型回复后，说明Agent逻辑和HTTP封装均工作正常。

两项验证均通过后，返回第一个终端，按Ctrl+C停止本地服务，继续进行后续的部署操作。

----结束

步骤三（方法 A）：手动制作镜像 + 通过控制台部署

若您选择方法 B（SDK一键部署），请跳转至[步骤三（方法B）：通过SDK一键部署](#)。

步骤1 准备依赖文件。

按上述步骤验证完成后，接下来将智能体打包并部署到AgentArts云端运行时。在构建镜像之前，需要先准备依赖清单文件，镜像构建时将根据此文件在容器内安装所有依赖。

在项目根目录创建requirements.txt，写入以下内容：

```
# =====
# 指定国内 PyPI 镜像源，加速镜像构建时的依赖下载
# =====
--index-url https://repo.huaweicloud.com/repository/pypi/simple
--trusted-host repo.huaweicloud.com

# =====
# 核心依赖
# =====

# LangChain: LLM 应用开发工具链
langchain>=0.3.0
langchain-core>=0.3.0
langchain-openai>=0.2.0

# 环境变量加载
python-dotenv>=1.0.0

# AgentArts SDK：运行时封装与 HTTP 服务标准化
agentarts-sdk
```

步骤2 制作Agent镜像。

1. 执行touch .dockerignore命令，创建.dockerignore文件，防止敏感文件被打包进镜像。

执行vi .dockerignore命令，在文件中写入以下内容：

```
# 虚拟环境（镜像内会重新安装依赖）
venv/

# 环境变量文件（含 API Key，禁止打包进镜像）
.env

# Python 缓存
__pycache__
*.pyc
*.pyo

# Git 相关
.git/
.gitignore

# 部署配置文件
.agentarts_config.yaml
```

2. 执行touch Dockerfile命令，在项目根目录创建Dockerfile，执行vi Dockerfile命令，写入以下内容。

```
# 必须使用 ARM64 架构的基础镜像
FROM python:3.10-slim

WORKDIR /app

# 优先复制依赖声明，利用 Docker 层缓存
COPY requirements.txt .

# 安装依赖（镜像源已在 requirements.txt 中指定）
RUN pip install --no-cache-dir -r requirements.txt

# 复制项目代码
COPY . .

EXPOSE 8080
```

```
ENTRYPOINT ["python", "app.py"]
```

3. 执行以下命令构建镜像。

```
docker build -t my-first-agent:latest .
```

4. 在本地运行容器，确认镜像可以正常启动。

```
docker run --rm -p 8080:8080 --env-file .env my-first-agent:latest
```

打开一个新的终端（保持原窗口运行），依次执行下面的命令，确认容器内的服务接口响应正常。

```
curl http://localhost:8080/ping
curl -X POST http://localhost:8080/invocations \
-H "Content-Type: application/json" \
-d '{"message": "你好，请用一句话做个自我介绍。"}'
```

步骤3 推送镜像到华为云SWR服务。

SWR是华为云的容器镜像服务。AgentArts控制台创建运行时时，会从SWR拉取镜像。本步骤将已构建的本地镜像推送到SWR。

1. 登录[SWR控制台](#)，切换到与AgentArts相同的区域（贵阳一）。

图 4-4 SWR 控制台



2. 创建SWR组织。SWR组织相当于镜像的命名空间，同一账号下组织名全局唯一。输入自定义组织名称，本文示例使用my-org-demo，下方命令中涉及该组织名称时，请替换为您实际使用的值。

3. 获取SWR临时登录指令。

在SWR“总览”页面，单击页面右上角“登录指令”，在弹出的对话框中复制“通用型登录指令”。

4. 在本地终端执行登录指令。

将复制的完整登录指令粘贴到终端执行。

```
# 以下为示例格式，请粘贴您实际获取到的完整指令
sudo docker login -u cn-southwest-2@{xxx} -p {xxx} swr.cn-southwest-2.myhuaweicloud.com
```

终端输出Login Succeeded即表示登录成功。

5. 给本地镜像打上SWR仓库标签。

请将my-org-demo替换为您实际的组织名称

```
sudo docker tag my-first-agent:latest swr.cn-southwest-2.myhuaweicloud.com/my-org-demo/my-first-agent:latest
```

6. 推送镜像到SWR。

请将my-org-demo替换为您实际的组织名称

```
sudo docker push swr.cn-southwest-2.myhuaweicloud.com/my-org-demo/my-first-agent:latest
```

推送完成后，回到SWR控制台，在“我的镜像”页面可查看已上传的镜像。

步骤4 通过AgentArts控制台创建运行时。

登录[AgentArts 控制台](#)，在左侧导航栏单击“智能体运行时”，单击右上角“托管智能体”，参考下表完成配置。

表 4-3 参数说明

参数	配置说明
名称	Agent名称，例如my-first-agent。
描述	Agent的描述。
镜像	选择已推送到SWR的镜像。
权限与访问控制	- 委托：使用系统默认创建的DefaultAgentArtsRuntimeAgency委托。 - 入网配置：使用系统默认网关。 - 入栈协议：选择HTTP协议。 - 入站身份认证：选择IAM认证。
可观测配置	开启。
生命周期配置	用于控制会话的自动终止策略，使用默认配置。
出网网络配置	选择公网访问。
存储配置	可选，用于挂载持久化存储。其中会话存储由平台统一托管；SFS Turbo和OBS仅在“私网访问”模式下支持配置。快速验证阶段无需配置。
启动命令	本示例已在Dockerfile中通过ENTRYPOINT ["python", "app.py"]指定了启动命令，此处无需填写，保持默认即可。
监听端口	监听端口默认为8080，与app.py中的app.run(host="0.0.0.0", port=8080)保持一致，无需修改。
路由配置	选择前缀配置。 这是一个容易忽略但非常关键的配置。AgentArts平台收到调用请求后，会先将路径中的 /runtimes/<运行时名称>/ invocations/ 前缀剥离，再将剩余路径转发到镜像内部的实际接口。若选择“严格匹配”，请求将无法与镜像内的接口路径匹配，调用时会返回404。
文件上传下载	保持默认“未开启”。若您的 Agent 需要处理文件，开启后运行时将额外提供文件上传/下载的API接口，快速验证阶段无需开启。

参数	配置说明								
环境变量	需要添加环境变量。逐一填入以下变量： - 变量名MODEL_API_KEY，值填写您MaaS中获取的模型API Key。 - 变量名MODEL_NAME，值填写deepseek-v4-flash。 - 变量名MODEL_URL，值填写https://api.modelarts-maas.com/openai/v1。 <table><tr><th>键</th><th>值</th></tr><tr><td>MODEL_URL</td><td>https://api.modelarts-maas.com/openai/v1</td></tr><tr><td>MODEL_NAME</td><td>deepseek-v4-flash</td></tr><tr><td>MODEL_API_KEY</td><td>W4BQF4QeXN_Lxk48nR1.....</td></tr></table>	键	值	MODEL_URL	https://api.modelarts-maas.com/openai/v1	MODEL_NAME	deepseek-v4-flash	MODEL_API_KEY	W4BQF4QeXN_Lxk48nR1.....
键	值								
MODEL_URL	https://api.modelarts-maas.com/openai/v1								
MODEL_NAME	deepseek-v4-flash								
MODEL_API_KEY	W4BQF4QeXN_Lxk48nR1.....								
标签	如需通过标签对云资源进行统一管理，可在此添加，快速验证阶段可跳过。								

单击“立即托管”，等待运行时就绪。

页面跳转至托管智能体列表，创建的运行时状态显示为“正常”，表示部署成功。

----结束

步骤三（方法 B）：通过 SDK 一键部署

SDK方式会自动完成镜像构建、推送SWR、部署运行时的全部操作，适合快速验证场景。

- 步骤1 参考前置条件与环境准备安装SDK，并配置华为云凭证。
- 步骤2 准备依赖文件。

在项目根目录创建requirements.txt，写入以下内容：

```
# =====  
# 指定国内 PyPI 镜像源，加速镜像构建时的依赖下载  
# =====  
--index-url https://repo.huaweicloud.com/repository/pypi/simple  
--trusted-host repo.huaweicloud.com  
  
# =====  
# 核心依赖  
# =====  
  
# LangChain: LLM 应用开发工具链  
langchain>=0.3.0  
langchain-core>=0.3.0  
langchain-openai>=0.2.0  
  
# 环境变量加载  
python-dotenv>=1.0.0  
  
# AgentArts SDK: 运行时封装与 HTTP 服务标准化  
agentarts-sdk
```

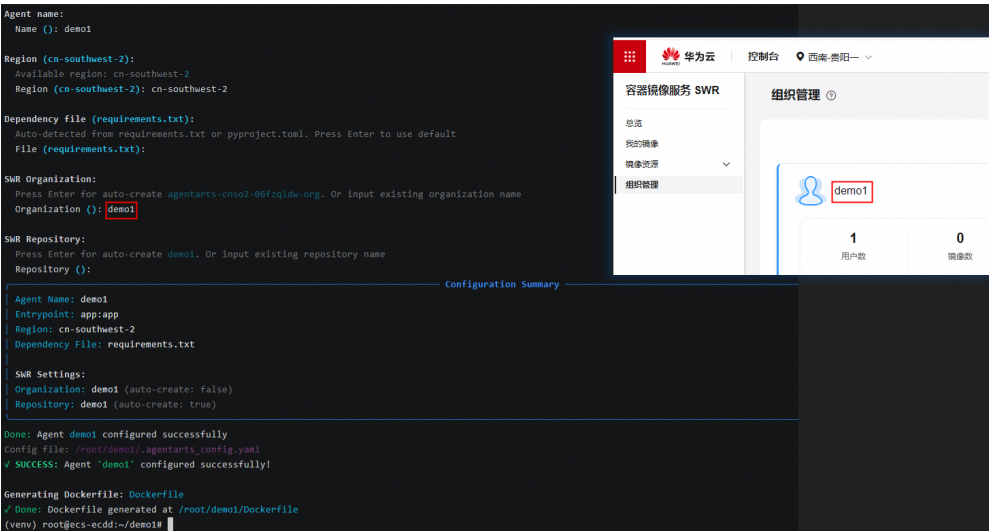
步骤3 初始化部署配置。

执行如下命令配置智能体。
agentarts configure --entrypoint app:app

执行后按照操作指引进行配置。

表 4-4 配置参数

参数	说明
Agent name	智能体名称。 以小写字母开头，以小写字母或数字结尾，可以包含小写字母、数字和中划线。
Region	服务部署区域。 cn-southwest-2，仅支持此区域。
Dependency file	依赖文件默认为requirements.txt。
SWR Organization	镜像组织名称（建议自定义的镜像组织名，可以在SWR服务控制台贵阳一region创建）。
SWR Repository	- 不填写（推荐）。 SWR Repository项直接回车留空，工具会自动生成仓库名称，部署agentarts launch时自动创建镜像仓库，避免仓库不存在报错。 - 填写自定义仓库名称。 若需要手动指定仓库，请填写SWR中已预先存在的镜像仓库名称，如未创建，此处请留空不填写。



步骤4 部署智能体。

agentarts launch

该命令会自动完成以下步骤：

1. 本地构建Docker镜像。
2. 将Docker镜像推送到华为云SWR镜像仓库。
3. 部署到AgentArts运行时托管环境。

----结束

步骤四：调用 Agent

部署完成后，您可以AgentArts运行时SDK提供的invoke命令直接调用您的智能体。

在命令行中执行 `agentarts invoke` 命令，SDK会自动使用您已配置的华为云IAM凭证（AK/SK）进行认证，无需额外参数。

```
agentarts invoke --agent my-first-agent '{"message": "你好，请用一句话做个自我介绍。"}'
```

- `--agent`：指定运行时名称（即您在控制台或部署时设置的名称）。
- 紧随其后的JSON字符串为请求体（需要与您的Agent代码中定义的一致）。

您也可以通过标准 HTTP API 直接调用运行时。调用时需要根据创建运行时选择的“入站身份认证”方式，在请求中携带对应的认证信息。

详细的调用步骤和示例代码请参考：

- [使用API调用运行时接口（API Key 认证）](#)
- [使用API调用运行时接口（IAM 认证）](#)

清理资源

验证完成后，如不再需要该运行时，请及时清理，避免产生不必要的费用。

登录[AgentArts 控制台](#)，在“智能体运行时”列表中，找到创建的运行时，进行删除。

下一步

成功部署第一个运行时后，可以进一步探索：

- 了解镜像制作的完整规范与高级配置 → [制作Agent镜像](#)
- 了解控制台部署的完整配置项说明 → [通过控制台部署智能体运行时](#)
- 了解SDK的完整命令参考 → [AgentArts运行时SDK](#)
- 为Agent集成平台底座组件，增强记忆、工具调用等能力 → [记忆库](#)、[网关](#)、[代码解释器](#)

4.6 部署 Agent

4.6.1 制作 Agent 镜像

本文介绍如何手动制作符合AgentArts智能体运行时规范的Docker镜像，并将其上传至华为云容器镜像服务（SWR），用于后续通过控制台部署智能体运行时。

前提条件

- 参考[环境准备](#)做好环境准备工作。
- 参考[本地开发](#)完成Agent本地代码的编写。

构建镜像

步骤1 已完成[前提条件](#)中的环境准备以及本地代码编写任务。

步骤2 在本地构建镜像。

在 Dockerfile 所在目录执行以下命令：

```
docker build -t agentarts-custom:latest .
```

说明

（可选）如果遇到拉取python:{version}-slim包失败，请修改docker镜像源。

执行以下命令在系统配置目录下创建一个名为docker的文件夹。

```
sudo mkdir -p /etc/docker
```

执行以下命令（直接复制下面这段代码并回车执行），更换Docker镜像源，从国内镜像站寻找python:3.10-slim。

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors":[
    "https://docker.m.daocloud.net",
    "https://dockerproxy.net",
    "https://mirror.baidubce.com"
  ]
}
EOF
```

依次执行下面两条命令，重启Docker服务：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

步骤3 本地运行验证。

```
docker run -d -p 8080:8080 \
-e REGION=cn-southwest-2 \
-e PROJECT_ID=your_project_id \ #鼠标移动至右上角用户名，在下拉列表中单击“我的凭证”，在“我的凭证”页面，在项目列表中查看项目ID。
--name my-agent \
agentarts-custom:latest
```

步骤4 测试健康检查。

```
curl http://localhost:8080/ping
# 预期返回: {"status":"Healthy","time_of_last_update":1784185138}
```

注意：如果健康检查失败，在 AgentArts 上部署时会被判定为启动失败，导致实例不断重启。

----结束

上传镜像

将构建的镜像上传至容器镜像服务。

步骤1 登录容器镜像服务控制台，选择区域，要和AgentArts区域保持一致，否则无法选择到镜像。

步骤2 单击右上角“创建组织”，输入组织名称完成组织创建。请自定义组织名称，本示例使用“demo-886633”，下面的命令中涉及组织名称“demo-886633”也请替换为自定义的值。

步骤3 单击右上角“登录指令”，获取登录访问指令，本文选择复制“通用型登录指令”。
以root用户登录本地环境，输入复制的SWR临时登录指令。

步骤4 上传镜像至容器镜像服务镜像仓库。

1. 使用docker tag命令给上传镜像打标签。
#region和domain信息请替换为实际值，组织名称demo-886633也请替换为自定义的值。

```
sudo docker tag agentarts-custom:latest swr.{region-id}.{domain}/demo-886633/agentarts-custom:latest
```


#此处以华为云cn-southwest-2为例

```
sudo docker tag agentarts-custom:latest swr.cn-southwest-2.myhuaweicloud.com/demo-886633/agentarts-custom:latest
```
2. 使用docker push命令上传镜像。
#region和domain信息请替换为实际值，组织名称demo-886633也请替换为自定义的值。

```
sudo docker push swr.{region-id}.{domain}/demo-886633/agentarts-custom:latest
```


#此处以华为云cn-southwest-2为例

```
sudo docker push swr.cn-southwest-2.myhuaweicloud.com/demo-886633/agentarts-custom:latest
```

步骤5 完成镜像上传后，在容器镜像服务控制台的“我的镜像”页面可查看已上传的自定义镜像。

----结束

4.6.2 通过控制台部署

将本地构建的高代码智能体部署至运行时上，为智能体提供一个安全隔离、与框架和模型无关的托管运行环境，确保智能体在独立、受控的条件下运行。部署完成后，可实时查看智能体托管运行的全链路日志，同时支持离线评估，对智能体的准确性、安全性、流畅性开展综合能力评测。

本文为您介绍如何在控制台上创建智能体运行时，定义智能体的基本信息、来源方式、权限与访问控制等。

前提条件

- 使用智能体运行时前需要先[开通AgentArts服务](#)并进行服务授权，请在“授权管理”页面进行云服务开通及授权，详细信息请参考[授权管理](#)。
- （可选）如出网网络配置为私网访问，请确保已开通虚拟私有云，并已创建VPC、子网、安全组。详细操作请参见[创建虚拟私有云和子网](#)、[创建安全组](#)。
- （可选）管理员可在IAM控制台为创建的IAM子用户授予权限，授权后，子用户即可根据权限使用账号中的云服务资源，相关操作步骤如下。
 - a. [创建IAM用户](#)。
 - b. [给IAM用户授权](#)。
 - 为子用户授予完整的智能体运行时服务权限，需授予AgentArtsCoreRunFullAccessPolicy身份策略。
 - 可根据子用户实际需求[创建自定义身份策略](#)并授予如下权限，授权项功能介绍请参考[表4-5](#)。

表 4-5 授权项

授权项	功能介绍
iam:agencies:pass	授予向云服务传递委托的权限。
iam:agencies:createV5	授予创建信任委托的权限。
vpc:routeTables:update	授予更新路由表权限。
vpc:nativePorts:create	授予原生API创建端口权限。
vpc:nativePorts:delete	授予原生API删除端口权限。
eip:publicIps:associateInstance	授予将弹性公网IP绑定网卡的权限。
eip:publicIps:disassociateInstance	授予将弹性公网IP解绑网卡的权限。
kms:cmk:createDataKey	授予使用KMS密钥生成数据密钥的权限。
kms:cmk:decryptDataKey	授予解密数据密钥的权限。
csms:secret:create	授予创建和恢复凭据的权限。
csms:secret:delete	授予删除指定凭据版本状态的权限。
sfsturbo:shares:getShare	查询弹性文件系统。
sfsturbo:shares:getAllShares	查询弹性文件系统列表。
obs:bucket:headBucket	授予获取桶元数据的权限。
obs:bucket:ListAllMyBuckets	授予查询创建的桶列表的权限。
obs:object:getObject	授予下载非指定版本对象的权限。

约束与限制

- 单个用户创建智能体运行时的数量限制：1000个。
- 单个智能体运行时创建的版本数量限制：1000个。
- 单个智能体运行时创建的访问方式数量限制：10个。
- 在“来源方式”中选择镜像时，当前仅支持在ARM64平台上构建的容器镜像。
- 请使用非latest的镜像tag管理镜像更新，使用时应避免重复推送完全相同的镜像名和标签，如强制更新镜像需要重建或者修改智能体运行时使用的镜像路径。
- 访问智能体运行时关于Session路由约束说明如下：
在智能体运行时中是否配置会话存储的路由规则不一致，请谨慎使用同一Session ID进行跨Endpoint访问，详情如下：
 - 未配置会话存储：新请求会根据指定的Endpoint进行路由，即同一Session ID可以在不同的版本中单独使用。
 - 已配置会话存储：新请求会根据当前是否存在已激活的同一个Session ID的沙箱进行判断。如果不存在已激活的沙箱，则会按照指定的Endpoint进行路由

由；如果存在已激活的沙箱且配置的Endpoint的版本与其一致，则会调用同一个沙箱，如果版本不一致，则会调用超时。

- 使用OBS存储挂载时相关约束及注意事项如下：
 - 托管智能体时如使用OBS存储配置，需创建访问OBS服务的VPC终端节点。
 - i. 进入[VPCEP控制台终端节点列表页](#)。
 - ii. 单击右上角“购买终端节点”，进入购买页。
 - iii. 在“购买终端节点”页面，根据提示配置参数，其他参数保持默认，更多参数详细信息请参见[购买终端节点](#)。

参数	示例	说明
区域	西南-贵阳一	终端节点所在区域。要求与运行时出网VPC、OBS桶所在区域保持一致。
计费方式	按需计费	按需计费是后付费模式，按终端节点的实际使用时长计费，可以随时开通/删除终端节点。
服务类别	按名称查找服务	选择“按名称查找服务”。
服务名称	-	在终端节点服务列表的“名称”列，输入待访问终端节点服务的名称，单击“验证”。 提交工单 获取服务名称，提交工单时同步提供OBS桶名给技术支持人员。将从工单中获得的服务名称填写到“服务名称”中，单击“验证”。
虚拟私有云	-	选择运行时出网VPC。

- iv. 单击右下角的“立即购买”。
- v. 确认终端节点配置，单击“提交”。
- 挂载目录中的文件或文件夹不支持硬链接命令。
- 挂载目录不会准确显示OBS桶的总容量，用户可以通过OBS控制台查看桶容量情况。
- 对象一旦写入即不可变，不支持原地修改。如需变更内容，必须下载完整对象、本地修改后全量重新上传并覆盖原对象。如果文件较大，建议采用分段上传，仅重新上传受影响的分片。
- 一个OBS桶支持挂载到多个Agent或多台云服务器，各挂载进程间互不感知。数据一致性需由用户自行维护，也不能通过OBS锁定文件。建议可通过分布式缓存服务（DCS）的Redis版本实现分布式锁，可避免出现多人对一个文件同时写入的情况。了解分布式缓存服务相关信息，请参见[什么是分布式缓存服务](#)。
- 对于对象存储的多版本功能，AgentArts无法感知到，也没有办法使用多版本的接口。对于开启了多版本控制的桶，AgentArts只能识别当前版本对象和当前版本的删除标记，并且无法对当前版本的对象真正做出修改（对象存储桶里的对象本身是不可修改的，只能覆盖当前版本对象，产生一个新的当前版本对象，原先的对象会变成历史版本对象）。由于上层应用的一些探测性IO（如vim会产生.swp文件），使得多版本桶里将产生大量中间过程对象，且对

象无法被AgentArts删除（删除操作会追加删除标记，而不是真正删除）。了解多版本控制与删除对象相关信息，请参见[多版本控制](#)和[删除对象](#)。

- 对象桶模式采用回写模式（write-back），读和写都会经过客户端本地的文件系统进行中转，通过tmpdir或use_cache参数控制临时文件目录所在的位置。在应用与AgentArts的交互过程中，应用不直接与AgentArts交互，而是先与FUSE交互。应用与FUSE的交互遵循POSIX语义，其中write()只会写到本地文件系统的临时文件，需要在应用调用文件的close()操作时，修改才会真正的上传到OBS。这意味着：
 - 数据在close()返回成功之前不在OBS上。进程崩溃、断电、kill -9等异常退出会导致未关闭文件的数据永久丢失。
 - close() 可能阻塞较长时间（取决于文件大小和网络带宽）。例如100GB文件的close可能需要数分钟。大对象上传场景，可以通过max_dirty_data参数，在写满部分分段大小时就提前触发上传，以提升整体上传效率。
 - 启用use_cache时，已写入但未上传的数据保留在磁盘缓存文件中，下次打开同一文件时有机会恢复。未启用use_cache时，关闭后本地数据立即释放。

基于以上模式，建议在关键数据写入后检查close()返回值，避免close()方法返回错误被吞噬；对持久性要求高的场景使用fsync()定期刷新（需权衡性能）；启用use_cache提供额外的数据恢复机会。

托管智能体

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 智能体运行时”。
- 步骤3 单击“托管智能体”，输入相关配置信息，具体参数请参考[表4-6](#)，配置完成后单击“立即托管”。

表 4-6 参数说明

参数		说明
基本信息	名称	智能体运行时的名称，同一账号下名称不可重复。 命名规则： • 以小写字母开头，小写字母或数字结尾。 • 支持数字、小写字母、中划线。 • 长度为2-48个字符。
	描述	用于对智能体运行时内容和用途的简要说明。 规则：长度不大于4096个字符。

参数		说明
来源方式	镜像	指定一个包含智能体应用程序及其所有依赖项的容器镜像。请选择与所选镜像架构匹配的镜像，否则可能导致运行异常。 单击“选择镜像”，在“我的镜像”、“共享镜像”、“镜像中心”下选择需要的镜像。 - 我的镜像：来源于在容器镜像服务中客户端或页面上上传的镜像，详细操作请参见页面上上传镜像、客户端上传。 - 共享镜像：来源于在容器镜像服务中他人共享的镜像，详情请查看共享镜像。 - 镜像中心：来源于容器镜像服务中的镜像中心。
权限与访问控制	委托	授予的代理权限或代理功能，允许代表智能体与外部系统进行通信和交互。 可使用系统默认创建的DefaultAgentArtsRuntimeAgency委托。 如需手动创建，请单击“创建委托”并参考如下配置创建。 - 委托名称：自定义。 - 信任主体类型：选择“云服务”。 - 云服务：搜索service.WorkloadSandboxMetadata。 - 其余参数使用默认值。 图 4-5 创建委托  委托创建完成后可对委托进行授权，操作具体详情请参考创建信任委托并授权。 说明 使用OBS桶挂载功能时，需为运行时委托配置相应的OBS权限。根据文件系统的访问模式，所需的最小权限如下： - 文件系统只读操作：obs:bucket:headBucket、obs:bucket:listBucket、obs:object:getObject。 - 文件系统读写操作：obs:bucket:headBucket、obs:bucket:listBucket、obs:object:getObject、obs:object:putObject、obs:object:deleteObject。 以上为最小权限要求。如需配置其他权限，请提交工单咨询OBS权限相关事宜。

参数		说明
	入网配置	选择 网络配置 中创建的网关，通过互联网或者VPC内网访问智能体。
	入栈协议	选择入栈协议，定义智能体接收和处理外部请求的方式。关于HTTP以及MCP的详细入站协议请参见 入站协议 。
	入站身份认证	配置入站身份认证方式。在智能体接收到请求时，对请求进行身份验证和授权，确保请求来自可信的来源，并且具有访问特定资源的权限。 支持以下认证方式： • IAM认证：使用登录管理控制台时使用的IAM用户名进行认证。 • OAuth 2.0认证：将OAuth 2.0配置为入站身份认证。选择“OAuth 2.0认证”后，相关参数配置请参考表 4-8。 • API Key 认证：通过接口的访问密钥进行身份验证和授权。输入API Key的名称后，为智能体运行时创建和绑定API Key。
可观测配置	日志记录	• 未开启，托管运行时过程产生的日志无法上报至云日志服务（LTS）。 • 开启后，托管运行时过程产生的日志会上报云日志服务（LTS），日志管理费用按需收取，有关计费相关内容请参考LTS计费说明。
生命周期配置 说明 当空闲会话超时或最大存活时间满足任一条件时，运行时会话将自动终止。	空闲会话超时	智能体运行时会话在设置的空闲会话超时时间内没有交互或活动，系统会自动结束该会话，历史数据会被清除。 最大值为604800秒（7天），您可以选择以秒、分钟、小时或天为单位进行设置，具体数值会根据所选单位的不同而变化。
	最大存活时间	会话持续调用时自动终止前的最大存活时间。 最大值为604800秒（7天），您可以选择以秒、分钟、小时或天为单位进行设置，具体数值会根据所选单位的不同而变化。
高级配置	实例规格	为智能体运行时实例指定CPU和内存规格组合，用户可以根据实际负载特征选择合适的资源规格，从而平衡成本与性能，提升资源利用效率。 当前支持选择2vCPUs 8GiB。如需增加更多规格，请提交工单申请。

参数		说明
	出网网络配置	配置智能体运行时的网络访问方式。 ● 公网访问：智能体可以访问互联网上的公共资源。 ● 私网访问：连接到内部的私有网络，访问内部的资源和服务。 - 选择已配置的VPC，如未配置，请单击“新建VPC”，具体请参考创建虚拟私有云和子网。 - 选择已配置的子网，如未配置，请单击“新建子网”，具体请参考创建虚拟私有云和子网。 - 选择已配置的安全组，如未配置，请单击“新建安全组”，具体请参考创建安全组。 配置后单击刷新。
	存储配置	当前支持SFS Turbo、会话存储、OBS三种类型。 ● 会话存储：用于保存会话上下文、中间结果、用户偏好等，存储资源由后台服务统一托管，每个会话拥有独立存储空间。配置后不支持取消配置，仅支持配置1个。 ● SFS Turbo：用于保存会话过程中产生的文件等，存储资源为用户的SFS Turbo文件系统。该配置仅在私网访问下支持并限制数量为5个。 ● OBS：用于保存会话过程中产生的文件等，存储资源为用户的OBS桶。最多只能选择5个不同的OBS桶，最多添加10个OBS类型的存储，且OBS仅在私网访问下支持配置。 开启后，单击“添加存储”，详细参数配置请参考表 4-7。
	启动命令 (可选)	智能体启动时需要执行的命令，最多添加10个命令，每个命令长度为1-256个字符。 启动命令中如指定端口时需要与监听端口相同。 启动命令、监听端口及环境变量的配置具体可参考Dockerfile参数在智能体运行时中如何使用。
	监听端口	智能体在运行时监听的网络端口，用于接收外部的网络请求。 所配置的监听端口需要与镜像中配置的监听端口（Dockerfile文件中的“EXPOSE”）保持一致。当前仅支持配置一个监听端口。

参数		说明
	路由配置	在调用智能体运行时的时候，支持指定接口路由匹配类型，以实现更灵活的请求路由策略。 - 严格匹配：要求调用URL必须完全符合指定的路径模式，只有当请求的URL路径为/runtimes/{runtime_name}/invocations时，才会匹配成功。一旦选择严格匹配，后续镜像新增的接口无法再修改为前缀匹配。 - 前缀匹配：调用URL的匹配模式支持 /runtimes/{runtime_name}/invocations 以及 /runtimes/{runtime_name}/invocations/{custom_path} 的调用路径。选择前缀匹配后，后续镜像中可以新增、删除接口等。
	文件上传 下载	开启后，智能体运行时将提供文件上传和下载的API接口。例如，用户可通过上传接口将本地文件发送给智能体，待其分析并生成报告后，下载生成的报告。 - 上传文件：通过调用文件上传接口，将本地文件上传至沙箱环境的指定路径。 上传文件时，单文件大小限制为 100MB，多文件总大小限制为 500MB。 - 下载文件：通过调用文件下载接口，获取沙箱环境中生成的结果文件。 配置后不允许修改。 详细操作说明请参考文件上传与下载。
	环境变量 (可选)	智能体运行时传递给应用程序的变量。 - 表单编辑：通过键值对方式配置环境变量。单击“添加变量”，设置键和值。 - JSON格式编辑：通过JSON文件方式配置环境变量。 JSON文件格式要求如下： <pre>{ "键": "值", "Path": "/dir/xxx" }</pre>
	标签 (可 选)	由“标签键”和“标签值”组成，用于标识和分类云资源。 您可以在TMS中创建预定义标签，用于使用同一标签标识多种云资源，创建后可在标签输入框下拉选择同一标签，具体操作请参考创建预定义标签，创建后单击刷新。 单击“添加标签”，选择在TMS中创建的预定义标签，或自定义输入标签键和标签值。 可添加一个或多个标签，最多添加20个标签。

表 4-7 添加存储参数说明

类型	参数	说明
会话存储	挂载路径	输入会话存储将被挂载的具体目录路径。配置后，每个会话在持久化存储中拥有独立的子目录，这些子目录将作为状态存储挂载到该路径。 挂载路径为/home、/mnt、/data、/tmp及其子目录，不能包含“..”，最大长度为4000字符。 存在多个挂载时（包括OBS、SFS Turbo、会话存储之间），禁止使用相同或存在包含关系的路径。
SFS Turbo	文件系统	选择文件系统。如未创建，单击“创建文件系统”，详细操作说明请参考创建文件系统。 配置后单击刷新。 说明 • SFS Turbo和智能体运行时中的出网网络配置必须在同一VPC内，SFS Turbo文件系统使用的安全组需放通NFS协议使用的端口号，具体信息请参考VPC的安全组是否影响高性能弹性文件服务的使用，安全组需要放通运行时出站的子网网段访问SFS Turbo。 • 选择的文件系统需为NFS协议。 • AgentArts部署智能体运行时挂载NFS文件系统的soft/hard的模式为hard模式，关于soft/hard的模式详情信息请查看挂载SFS Turbo文件系统到Linux云服务器。 • 为了您的信息安全，请勿在文件系统中存放敏感数据。
	存储路径	输入文件系统中数据的存储路径。 存储路径以“/”开头，不能包含“..”，最大长度4000字符，默认为/。
	挂载路径	输入文件系统中数据的挂载路径，使得该目录下的文件和目录可以被访问。 • 挂载路径为/home、/mnt、/data、/tmp及其子目录，不能包含相对路径，最大长度为4000字符。 • 存在多个挂载时（包括OBS、SFS Turbo、会话存储之间），禁止使用相同或存在包含关系的路径。 • 同一智能体运行时的多个版本挂载在同一目录时会自动共享。 说明 挂载路径的Linux文件系统权限，继承自SFS Turbo存储路径的原有权限（SFS Turbo根目录权限为任何用户可读写）。此权限不支持通过chmod/chown等命令修改。

类型	参数	说明
	读写权限	选择对挂载文件系统的读写权限。 - 读写：可以读取、写入和修改文件系统中的数据，默认为读写。 说明 Linux文件系统权限和读写权限两者必须同时允许，读写才能成功，如果任一项禁止，读写操作将失败。 - 只读：只能读取文件系统中的数据，不能写入或修改数据。
OBS	OBS桶	选择已创建的OBS桶。如未创建，单击“创建OBS桶”，详细操作说明请参考创建桶。 配置后单击  刷新。 说明 - 挂载路径的OBS文件系统权限，继承自OBS存储路径的原有权限（OBS根目录默认权限为任何用户可读写）。此权限不支持通过chmod/chown等命令修改。 - OBS仅支持桶类型，暂不支持并行文件系统。 - 为了您的信息安全，请勿在文件系统中存放敏感数据。
	存储路径(可选)	输入OBS桶内的子目录路径。 挂载OBS桶内的指定子路径，当配置为空时挂载整个桶根目录；非空时挂载桶内指定子目录，必须以'/'开头，支持大小写字母、数字及！_-.*'()/等字符且不能包含'..'，路径长度为最大1024字符。
	挂载路径	输入OBS桶挂载到容器内的路径。 - 挂载路径为/home、/mnt、/data、/tmp及其子目录，不能包含相对路径，最大长度为4000字符。 - 存在多个挂载时（包括OBS、SFS Turbo、会话存储之间），禁止使用相同或存在包含关系的路径。 说明 挂载路径的Linux文件系统权限继承自OBS桶存储路径的原有权限。此权限不支持通过chmod/chown等命令修改。
	读写权限	选择对挂载OBS桶的读写权限。 - 读写：可以读取、写入和修改OBS桶中的数据，默认为读写。 说明 Linux文件系统权限和读写权限两者必须同时允许，读写才能成功，如果任一项禁止，读写操作将失败。 - 只读：只能读取OBS桶中的数据，不能写入或修改数据。

表 4-8 OAuth 2.0 认证参数配置说明

参数	说明
Discovery URL	OAuth发现服务器（ OAuth discovery server ）只支持公网权威的认证服务器，即服务端证书必须能够校验通过。 需输入身份提供商（例如Okta、Cognito等）提供的Discovery URL，该URL通常可在该提供商的文档中找到。输入以https://开头， /.well-known/openid-configuration结尾的有效URL。
JWT授权配置	
允许的受众	用于认证为OAuth 2.0指定的受众是否在Agent Identity运行中指定的受众匹配或为其子集。 支持添加100个受众。
允许的客户端	用于认证为OAuth 2.0指定的客户端标识符是否被允许访问Agent Identity。 支持添加100个客户端。
允许的范围	仅当令牌包含此处配置的至少一个必须范围时才允许访问自定义声明。 支持添加100个范围。
自定义声明	仅当令牌中的特定声明与预定义字符串值相匹配时才允许访问。 支持添加100个自定义声明。

说明

运行时界面所显示的状态是运行时创建/删除所处的状态，表示操作本身的执行结果，状态“正常”意味着已成功在AgentArts和Miracle进行创建，并非实际运行的实例的状态。



----结束

相关操作

智能体运行时托管部署完成后，支持多维度采集Agent全链路关键指标，可实时观测智能体在托管运行中的使用日志，详细操作请[查看高代码智能体运行时数据信息](#)；同时支持离线评估，对智能体的准确性、安全性、流畅性开展综合能力评测，详细操作请参见[智能体运行时离线评估任务](#)。

更多操作

智能体运行时创建完成后，您还可以执行如下操作。

表 4-9 更多操作

操作	说明
查看智能体运行时基本信息	在智能体运行时页面，单击运行时名称，在“基本信息”页签，查看运行时的基本信息、访问方式及高级配置。
编辑智能体运行时	当修改智能体运行的基本信息、权限与访问控制，或者智能体代码更新并打包成新镜像重新上传后，运行时版本均会更新。 1. 单击智能体运行时列表操作列的“编辑”。 2. 修改运行时的基本信息，权限与访问控制等，然后单击“保存为新版本”。
删除智能体运行时	智能体运行时删除后不可恢复，请谨慎操作。 1. 单击智能体运行时列表操作列的“删除”。 2. 在弹框中确认要删除后输入“DELETE”，然后单击“确定”。

4.6.3 通过 SDK 部署

本文以预置Agent模板为例，介绍如何在本地使用AgentArts SDK的Toolkit工具将本地代码部署到云上AgentArts智能体运行时。

背景介绍

AgentArts SDK包含一套命令行工具，用于简化Agent应用的全生命周期管理，包括初始化、配置、构建、部署和运维。当前Toolkit已预置以下Agent示例模板供您选择，您也可以使用自研代码：

- basic：基础智能体模板，具有最小化的设置。适用于从一个非常简单的起点开始，然后根据需求逐步添加功能的开发者。它提供一个基本的框架，可以帮助开发者快速启动项目，而不需要从零开始构建所有内容。
- langgraph：基于LangGraph的智能体，适用于需要处理复杂状态和逻辑的应用场景。
- langchain：基于LangChain的智能体，适合需要将多个语言处理工具和服务集成到一个应用中的场景。
- google-adk：Google ADK智能体模板，适用于利用Google提供的技术和工具来开发智能体应用的开发者。

前提条件

- 操作系统：**推荐使用Linux Ubuntu 22.04 ARM64**，服务器可访问公网。
本文命令如无特殊说明，均以Ubuntu为例。为保证镜像构建与云端部署的兼容性，建议使用Ubuntu系统。

因此，为了减少部署失败和额外排查成本，建议直接使用Ubuntu 22.04 ARM64制作AgentArts镜像。

若使用CentOS、EulerOS等非Ubuntu系统，请将下命令中的apt替换为yum或dnf，并自行确认Docker 18.06+、Python 3.10+；本文不逐条展开各发行版安装差异。

⚠ 注意

需使用ARM64系统制作的Agent镜像，使用X86系统制作的镜像在调用智能体运行时调用失败。

可以使用华为云ECS服务购买服务器，购买时需要选择鲲鹏架构（ARM64），系统选择Ubuntu。

云服务器控制台 / 弹性云服务器 / 购买弹性云服务器

<  购买弹性云服务器

快速购买 ^① 自定义购买 ^②

实例  如何选择ECS实例规格?

规格类型选型 业务场景选型

CPU架构 ^③

x86计算

鲲鹏计算

- 安装Python：请确保Python 3.10及以上版本已安装。
大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version检查。如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

非Ubuntu/Debian系统请使用yum或dnf安装Python 3.10+及pip。

- 安装Docker：请确保Docker 18.06及以上版本已安装。如未安装，可以使用如下命令安装：

```
# 查询 Docker 版本
docker --version

# 安装Docker
sudo apt update
sudo apt install docker.io
```

安装完成后，建议立即配置国内镜像加速器。在后续执行agentarts launch构建镜像时，Docker需要拉取python:3.10-slim基础镜像，默认从docker.io官方仓库下载，可能超时失败。提前配置加速器可避免此问题。

执行以下命令创建Docker配置目录：

```
sudo mkdir -p /etc/docker
```

执行以下命令配置国内镜像源（直接复制并回车执行）：

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors":[
    "https://docker.m.daocloud.net",
    "https://dockerproxy.net",
    "https://mirror.baidubce.com"
  ]
}
EOF
```

依次执行以下命令重启Docker服务，使配置生效：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

📖 说明

华为云SWR基础版不支持OCI镜像格式，如果您使用的是Docker 27及以上版本，并且需要处理OCI镜像，可以通过设置环境变量来关闭OCI支持。

```
export DOCKER_BUILDKIT=0
# 或者
export BUILDKIT_USE_OCI_MEDIA_TYPES=0
```

- 执行以下命令安装SDK（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
# 安装依赖包
apt install python3.12-venv

# 创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate

# 安装sdk
pip install agentarts-sdk
```

若配置镜像加速器，后续在执行agentarts launch命令时，仍无法拉取python:3.10-slim，可参考本文结尾的“常见问题”中的“直接修改Dockerfile基础镜像地址”。

📖 说明

- 执行pip install agentarts-sdk命令下载缓慢、超时可以更改为使用如下命令
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
- 如果系统缺少python3-venv包，导致无法创建虚拟环境，请按照命令回显提示安装python3-venv包。

```
root@ecs-6732:~# python3 -m venv myenv
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

apt install python3.12-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

Failing command: /root/myenv/bin/python3
```

- 执行以下命令配置华为云凭证，获取华为云凭证请参考[认证鉴权](#)。
export HUAWEICLOUD_SDK_AK="your-access-key"
export HUAWEICLOUD_SDK_SK="your-secret-key"

通过 SDK 部署智能体运行时

步骤1 创建agent。

执行以下命令初始化一个LangGraph项目，支持基本的问答能力。

```
agentarts init -n my-agent -t langgraph
```

执行完成后会在目录下创建如下目录和文件：

```
my_agent/
├── agent.py          # Agent implementation
├── requirements.txt  # Python dependencies
├── Dockerfile
└── .agentarts_config.yaml # Project configuration
```

步骤2 （可选）配置agent。使用交互式配置向导设置部署参数。

```
agentarts config
```

配置向导会引导完成以下配置，按Enter键表示使用默认配置：

- 1. Agent名称（显示已有Agent列表）
- 2. 入口函数（格式：module:function）
- 3. 华为云区域（默认cn-southwest-2）
- 4. 依赖文件（自动检测requirements.txt）
- 5. SWR组织名称
- 6. SWR仓库名称

步骤3 修改.agentarts_config.yaml中环境变量配置。

```
environment_variables:  
  - key: OPENAI_API_KEY  
    value: 您申请的大模型apikey  
  - key: OPENAI_MODEL_NAME  
    value: 您申请的大模型名称  
  - key: OPENAI_BASE_URL  
    value: 您申请的大模型访问地址
```

步骤4 部署agent。

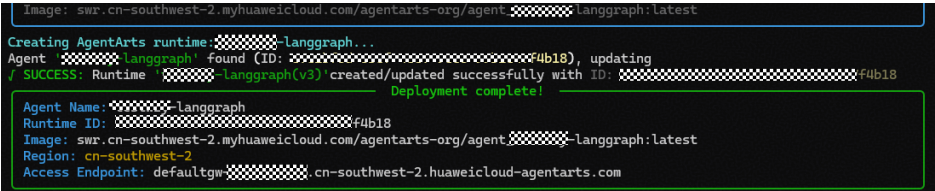
执行以下命令，将应用在本地构建镜像并部署至AgentArts智能体运行时。

```
agentarts launch
```

说明

首次部署耗时较长，大约需要3-5min，请耐心等待。
当回显部署完成时，则表示创建成功。

图 4-6 部署完成



部署完成后登录AgentArts控制台，查看已部署的智能体运行时及详细信息。

图 4-7 查看已部署的智能体运行时



步骤5 查看智能体启动状态以及调试agent。

执行以下命令，对agent进行调试。

```
agentarts invoke '{"message": "你好"}'
```

调用成功结果如下：

```
(venv) [DESKTOP-U64UKKM ~] -langgraph$ agentarts invoke '{"message": "你好"}' --bearer-token   

Invoke Request
Mode: Cloud
Agent: -langgraph
Session: 14ce5514-5932-4db4-80e7-b3fddbb710cd
Endpoint: https://defaultgw.cn-southwest-2.huaweicloud-agentarts.com
Auth Type: API_KEY

Response:
{
  "response": "你好! 🌟 有什么我可以帮你的吗? 无论是学习、工作还是生活中的问题, 都可以随时问我哦!",
  "status": "completed",
  "agent": "-langgraph",
  "model": "qwen3.5-plus"
}
```

----结束

更多操作

关于SDK使用的更多详细操作请参考[AgentArts运行时SDK](#)。

4.7 部署 MCP Server

4.7.1 什么是 MCP Server 部署

智能体运行时支持两种部署对象：Agent和MCP Server。Agent 部署是将一个智能体应用打包为镜像部署到运行时，通过HTTP接口接收用户消息并返回回复。而MCP Server部署是运行时的第二种部署模式，用于将自定义工具服务托管到云端，供其他Agent调用。

本文介绍MCP Server部署模式的概念、适用场景，以及与Agent 部署的区别。

MCP 协议简介

MCP（Model Context Protocol）是由Anthropic提出并开源的协议标准，用于标准化大型语言模型（LLM）应用与外部工具、数据源之间的通信方式。它定义了一套统一的JSON-RPC 2.0消息格式，让AI客户端（如 Claude Desktop、Cursor、Claude Code等）能够发现、调用外部工具，而无需关心底层实现细节。

在AgentArts运行时中，MCP Server部署采用Streamable HTTP传输方式，核心特性包括：

- **单端点**：同一个URL（如 /mcp）处理所有通信，简化路由和负载均衡配置。
- **POST请求**：客户端向服务器发送JSON-RPC请求消息。
- **GET请求（可选）**：客户端建立SSE流，用于接收服务器主动推送的消息（如日志、资源变更通知）。
- **内容协商**：客户端发送Accept: application/json, text/event-stream，服务器根据场景选择返回同步JSON或SSE流。
- **会话管理**：通过Mcp-Session-Id头部实现有状态会话，支持断线重连和消息重传。
- **无状态模式**：对于简单的工具调用场景，可以直接返回application/json，无需维护SSE连接，更适合Serverless和水平扩展。

📖 说明

关于MCP协议在运行时中的完整规范（传输方式、会话管理、镜像要求、路径要求），请参考[MCP](#)。

Agent 部署与 MCP Server 部署说明

智能体运行时支持将以下两种对象打包为镜像并部署：

- Agent：一个完整的智能体应用，接收用户输入，自主推理并调用模型和工具，返回对话回复。
- MCP Server：一个工具服务，不自主推理，而是提供一组可被调用的工具能力（如计算、查询、文件操作等），等待Agent来发现和调用。

Agent部署与MCP Server部署

维度	Agent 部署	MCP Server 部署
进站协议	HTTP	MCP
角色	接收用户输入，自主推理调用	提供工具能力，被 Agent 调用
谁调用它	应用系统/用户	Agent
对外暴露	对话接口（ /invocations ）	工具接口（ /mcp ）
典型代码	基于 LangChain/LangGraph 的对话 Agent	基于 FastMCP 的工具集服务
调用方式	发送消息，获取回复	发现工具 → 调用工具 → 获取结果
端口	默认 8080	默认 8000

使用场景对比

- Agent部署：需要一个完整的对话型智能体，用户发送消息后Agent自主推理、调用模型和工具，返回最终回复。
- MCP Server部署：已经有一个Agent，需要为它提供一组自定义工具能力，这些工具的逻辑由你自己开发并托管到云端。

4.7.2 制作 MCP Server 镜像

一个最小化的MCP Streamable HTTP计算器服务器示例，基于官方MCP SDK的FastMCP实现。适合作为学习MCP协议，快速搭建Streamable HTTP服务的起点。

制作 Docker 镜像

- 参考[环境准备](#)做好环境准备工作。
- 已完成MCP Server代码开发任务，代码示例可参考[代码开发示例](#)。

构建镜像：

```
#进入mcp代码开发路径
cd mcp_test/

# 生成 arm64 镜像
docker build --platform linux/arm64 -t mcp-calculator:latest .
```

本地Docker调测

本地启动Docker容器：

```
docker run -d \
  --name mcp-calculator1 \
  -p 8080:8080 \
  mcp-calculator:latest
```

本地调测Docker：

```
# 方式 1：curl 直接调用 initialize
curl -X POST http://127.0.0.1:8080/mcp \
  -H "Content-Type: application/json" \
  -H "Accept: application/json, text/event-stream" \
  -d '{
    "jsonrpc": "2.0",
    "id": 0,
    "method": "initialize",
    "params": {
      "protocolVersion": "2025-03-26",
      "capabilities": {},
      "clientInfo": {"name": "test", "version": "1.0.0"}
    }
  }'
#调测结果
event: message
data: {"jsonrpc":"2.0","id":0,"result":{"protocolVersion":"2025-03-26","capabilities":{"experimental":{},"prompts":{"listChanged":false},"resources":{"subscribe":false,"listChanged":false},"tools":{"listChanged":false},"serverInfo":{"name":"calculator-mcp-server","version":"1.28.1"}}}}

# 方式 2：curl 调用工具
curl -X POST http://127.0.0.1:8080/mcp \
  -H "Content-Type: application/json" \
  -H "Accept: application/json, text/event-stream" \
  -d '{
    "jsonrpc": "2.0",
    "id": 1,
    "method": "tools/call",
    "params": {
      "name": "add",
      "arguments": {"a": 3, "b": 5}
    }
  }'
#调测结果
event: message
data: {"jsonrpc":"2.0","id":1,"result":{"content":[{"type":"text","text":"8.0"}],"structuredContent":{"result":"8.0","isError":false}}}
```

调测完成后停止并删除容器：

```
docker stop mcp-calculator1
docker rm mcp-calculator1
```

上传镜像

将构建的镜像上传至容器镜像服务。

- 步骤1** 登录容器镜像服务控制台，选择区域，要和AgentArts区域（西南-贵阳一）保持一致，否则无法选择到镜像。
- 步骤2** 单击右上角“创建组织”，输入组织名称完成组织创建。请自定义组织名称，本示例使用“demo-886633”，下面的命令中涉及组织名称“demo-886633”也请替换为自定义的值。
- 步骤3** 单击右上角“登录指令”，获取登录访问指令，本文选择复制“通用型登录指令”。
以root用户登录本地环境，输入复制的SWR临时登录指令。

步骤4 上传镜像至容器镜像服务镜像仓库。

1. 使用docker tag命令给上传镜像打标签。

#region和domain信息请替换为实际值，组织名称demo-886633也请替换为自定义的值。
sudo docker tag mcp-calculator:latest swr.{region-id}.{domain}/demo-886633/mcp-calculator:latest
#此处以华为云cn-southwest-2为例
sudo docker tag mcp-calculator:latest swr.cn-southwest-2.myhuaweicloud.com/demo-886633/mcp-calculator:latest
2. 使用docker push命令上传镜像。

#region和domain信息请替换为实际值，组织名称demo-886633也请替换为自定义的值。
sudo docker push swr.{region-id}.{domain}/demo-886633/mcp-calculator:latest
#此处以华为云cn-southwest-2为例
sudo docker push swr.cn-southwest-2.myhuaweicloud.com/demo-886633/mcp-calculator:latest

步骤5 完成镜像上传后，在容器镜像服务控制台的“我的镜像”页面可查看已上传的自定义镜像。

----结束

部署到华为云 AgentArts

步骤1 部署智能体运行时，部署时“入栈协议”请选择为“MCP”，具体请参考[通过控制台部署](#)。

图 4-8 部署智能体运行时

基本信息

名称

runtime-2gr6pvmv

以小写字母开头，以小写字母或数字结尾，可以包含小写字母、数字和中划线，长度为2-48个字符。

描述 (可选)

请输入

来源方式

镜像架构

ARM

镜像

请输入或选择

选择镜像

请选择或输入与所选镜像架构匹配的镜像，否则可能导致运行异常。镜像地址，如 swr.cn-north-x.mycloud.com/service/ai-assistantv1.0.0

权限与访问控制

委托

D-...

创建委托

为保证智能体运行时正常运行，所选委托需具备相关权限。[查看当前权限](#)

入站网关

d-...

创建网关

入栈协议

HTTP

MCP

工具和代理服务器的模型上下文协议

步骤2 云上AgentArts调测。

1. 列出工具

```
curl -X POST https://${RUNTIME_ENDPOINT} \      # RUNTIME_ENDPOINT的获取方式：在智能体运行时列表单击名称，在详情页面的"访问方式"中的"访问地址"
-H "Content-Type: application/json" \
-H 'X-Hw-Agentarts-Session-Id: {xxxxxxxxxx}' \
-H 'Authorization: Bearer {apikey}' \
-H "Accept: application/json, text/event-stream" \
-d '{"jsonrpc": "2.0", "id": 1, "method": "tools/list"}
```

2. 调用工具

```
curl -X POST https://${RUNTIME_ENDPOINT} \
-H "Content-Type: application/json" \
-H "Accept: application/json, text/event-stream" \
-H 'Authorization: Bearer {apikey}' \
-H "Accept: application/json, text/event-stream" \
-d '{
  "jsonrpc": "2.0",
  "id": 2,
  "method": "tools/call",
  "params": {
    "name": "add",
    "arguments": {"a": 3, "b": 5}
  }
}'
```

如果返回了正确的 JSON-RPC 结果，说明云上部署和调测成功。

----结束

代码开发示例

项目结构

1. 创建项目文件夹：

```
mkdir mcp_test
```

2. 项目下的工程目录文件：

```
mcp_test/
├── mcp_server.py      # 主服务器实现
├── requirements.txt   # Python 依赖
└── Dockerfile        # 自定义镜像构建
```

完整服务器代码

mcp_server.py:

```
import logging
import os
import uvicorn
from mcp.server import transport_security
from mcp.server.fastmcp import FastMCP
from mcp.server.streamable_http_manager import TransportSecuritySettings
from starlette.requests import Request
from starlette.responses import JSONResponse
# 配置日志
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)
transport_security.TransportSecurityMiddleware._validate_host = lambda _self, _host: True
logger.info("MCP Host header validation patched to allow all hosts")
# 创建 MCP 服务器实例
mcp = FastMCP(
    "calculator-mcp-server",
    streamable_http_path="/mcp",
    stateless_http=True,
    transport_security=TransportSecuritySettings(
        enable_dns_rebinding_protection=False,
    ),
)
```

```
logger.info(f"MCP transport_security settings: {mcp.settings.transport_security}")
@mcp.tool()
def add(a: float, b: float) -> float:
    """两个数字相加"""
    logger.info(f"add called: {a} + {b}")
    return a + b
@mcp.tool()
def subtract(a: float, b: float) -> float:
    """两个数字相减"""
    logger.info(f"subtract called: {a} - {b}")
    return a - b
@mcp.tool()
def multiply(a: float, b: float) -> float:
    """两个数字相乘"""
    logger.info(f"multiply called: {a} * {b}")
    return a * b
@mcp.tool()
def divide(a: float, b: float) -> float:
    """两个数字相除，除数不能为0"""
    logger.info(f"divide called: {a} / {b}")
    if b == 0:
        raise ValueError("除数不能为0")
    return a / b
@mcp.custom_route("/ping", methods=["GET"])
async def ping(_request: Request) -> JSONResponse:
    """网关探活 ping 接口"""
    return JSONResponse({"status": "ok", "message": "pong"})
# 导出 Streamable HTTP ASGI 应用
app = mcp.streamable_http_app()
def main() -> None:
    """命令行入口，供 uvicorn / 控制台脚本调用。"""
    port = int(os.getenv("PORT", 8080))
    uvicorn.run(app, host="0.0.0.0", port=port)
if __name__ == "__main__":
    main()
```

Dockerfile:

```
# 先配置docker内置镜像加速拉取基础镜像，构建阶段pip用华为云源
FROM python:3.11-slim
WORKDIR /app
COPY requirements.txt .
# 华为云pip临时源
RUN pip install --no-cache-dir --upgrade pip \
    -i https://mirrors.huaweicloud.com/repository/pypi/simple \
    --trusted-host mirrors.huaweicloud.com \
    && pip install --no-cache-dir -r requirements.txt \
    -i https://mirrors.huaweicloud.com/repository/pypi/simple \
    --trusted-host mirrors.huaweicloud.com
COPY mcp_server.py .
EXPOSE 8080
CMD ["uvicorn", "mcp_server:app", "--host", "0.0.0.0", "--port", "8080", "--proxy-headers", "--forwarded-allow-ips", "*"]
```

requirements.txt:

```
mcp==1.28.1
uvicorn==0.46.0
```

本地验证

参考[环境准备](#)做好环境准备工作。

```
# 1. 安装依赖
pip install -r requirements.txt -i https://mirrors.huaweicloud.com/repository/pypi/simple --trusted-host mirrors.huaweicloud.com

# 2. 启动服务器
python mcp_server.py
```

```
# 执行以下步骤前，请打开一个新的终端窗口（保持原窗口运行）
# 3. 测试 initialize
curl -X POST http://127.0.0.1:8080/mcp \
  -H "Content-Type: application/json" \
  -H "Accept: application/json, text/event-stream" \
  -d '{"jsonrpc":"2.0","id":0,"method":"initialize","params":{"protocolVersion":"2025-03-26","capabilities":{},"clientInfo":{"name":"test","version":"1.0.0"}}}'
# 测试结果
event: message
data: {"jsonrpc":"2.0","id":0,"result":{"protocolVersion":"2025-03-26","capabilities":{"experimental":{},"prompts":{"listChanged":false},"resources":{"subscribe":false,"listChanged":false},"tools":{"listChanged":false},"serverInfo":{"name":"calculator-mcp-server","version":"1.28.1"}}}}

# 4. 测试工具调用
curl -X POST http://127.0.0.1:8080/mcp \
  -H "Content-Type: application/json" \
  -H "Accept: application/json, text/event-stream" \
  -d '{"jsonrpc":"2.0","id":1,"method":"tools/call","params":{"name":"add","arguments":{"a":3,"b":5}}}'
# 测试结果
event: message
data: {"jsonrpc":"2.0","id":1,"result":{"content":[{"type":"text","text":"8.0"}],"structuredContent":{"result":8.0,"isError":false}}}

#停止服务器
ps -ef |grep mcp_server.py
root    12692   5936   0 15:29 pts/0    00:00:00 python mcp_server.py
root    13674   5936   0 15:37 pts/0    00:00:00 grep --color=auto mcp_server.py
(venv) root@ecs-tmp-ggq:~/mcp_test# kill -9 12692
```

4.7.3 部署到运行时

智能体运行时支持部署和运行MCP服务器。本章节介绍如何创建、测试并部署MCP服务器。

步骤1：创建MCP服务器

步骤2：在本地测试MCP服务器

步骤3：将MCP服务器部署到AgentArts

步骤4：调用已部署的MCP服务器

前提条件

- 操作系统：**推荐使用Linux Ubuntu 22.04 ARM64**，服务器可访问公网。
本文命令如无特殊说明，均以Ubuntu为例。为保证镜像构建与云端部署的兼容性，建议使用Ubuntu系统。
因此，为了减少部署失败和额外排查成本，建议直接使用Ubuntu 22.04 ARM64制作AgentArts镜像。
若使用CentOS、EulerOS等非Ubuntu系统，请将下命令中的apt替换为yum或dnf，并自行确认Docker 18.06+、Python 3.10+；本文不逐条展开各发行版安装差异。

⚠ 注意

需使用ARM64系统制作的Agent镜像，使用X86系统制作的镜像在调用智能体运行时
会调用失败。
可以使用华为云ECS服务购买服务器，购买时需要选择鲲鹏架构（ ARM64 ），系
统选择Ubuntu。

云服务器控制台 / 弹性云服务器 / 购买弹性云服务器

<  购买弹性云服务器

快速购买 ① 自定义购买 ②

实例

+ 如何选择ECS实例规格?

规格类型选型 业务场景选型

CPU架构 ①

x86计算 鲲鹏计算

- 安装Python：请确保Python 3.10及以上版本已安装。
大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --
version检查。如未安装，可以使用如下命令安装：

```
sudo apt update  
sudo apt install python3
```

非Ubuntu/Debian系统请使用yum或dnf安装Python 3.10+及pip。

步骤 1：创建 MCP 服务器

步骤1 安装MCP软件包。

```
pip install mcp
```

步骤2 创建MCP服务器。
参考以下代码示例创建名为mcp_test.py的服务器。

```
# mcp_test.py  
from mcp.server.fastmcp import FastMCP  
mcp = FastMCP(host="0.0.0.0", stateless_http=True, port=8000)  
@mcp.tool()  
def add(a: float, b: float) -> float:  
    """Add two numbers"""  
    return a + b  
@mcp.tool()  
def subtract(a: float, b: float) -> float:  
    """Subtract two numbers"""  
    return a - b  
@mcp.tool()  
def multiply(a: float, b: float) -> float:  
    """Multiply two numbers"""  
    return a * b  
@mcp.tool()  
def divide(a: float, b: float) -> float:  
    """Divide two numbers"""  
    if abs(b) < 1e-9:  
        raise ValueError("Division by zero")  
    return a / b  
if __name__ == "__main__":  
    mcp.run(transport="streamable-http")
```

代码解释

- FastMCP：创建可以托管工具的MCP服务器。
- Tools：工具，展示不同类型的操作。
- @mcp.tool（）：将Python函数转换为MCP工具的装饰器。
- stateless_http=True：将服务器配置为无状态模式。

----结束

步骤 2：在本地测试 MCP 服务器

步骤1 启动MCP服务器。

```
python mcp_test.py
```

看到输出信息后，表明服务器正在8000端口上运行。

步骤2 使用curl命令进行客户端测试。

示例如下：

```
# 获取工具列表
curl -X POST http://127.0.0.1:8000/mcp \
-H "Content-Type: application/json" \
-H "Accept: application/json, text/event-stream" \
-d '{"jsonrpc":"2.0","id":1,"method":"tools/list"}'

# 调用工具
curl -X POST http://127.0.0.1:8000/mcp \
-H "Content-Type: application/json" \
-H "Accept: application/json, text/event-stream" \
-d '{
  "jsonrpc": "2.0",
  "id": 2,
  "method": "tools/call",
  "params": {
    "name": "add",
    "arguments": {"a": 3, "b": 5}
  }
}'
```

注意

Streamable HTTP 要求客户端 Accept 头必须同时包含 application/json 和 text/event-stream，否则服务器返回 406 Not Acceptable。

----结束

步骤 3：将 MCP 服务器部署到 AgentArts

步骤1 在本地自主开发MCP并打包成镜像，并将打包的镜像上传至容器镜像服务。

步骤2 部署智能体运行时，部署时“入栈协议”请选择为“MCP”，具体请参考[通过控制台部署](#)。

图 4-9 部署智能体运行时

基本信息

名称

runtime-2gr6pvmy

以小写字母开头，以小写字母或数字结尾，可以包含小写字母、数字和中划线，长度为2-48个字符。

描述 (可选)

请输入

来源方式

镜像架构

ARM

镜像

请输入或选择

选择镜像

请选择或输入与所选镜像架构匹配的镜像，否则可能导致运行异常。镜像地址，如 swr.cn-north-x.mycloud.com/service/ai-assistantv1.0.0

权限与访问控制

委托

D

创建委托

为保证智能体运行时正常运行，所选委托需具备相关权限。[查看当前权限](#)

入站网关

d

创建网关

入栈协议

HTTP

MCP

工具和代理服务器的模型上下文协议

----结束

步骤 4：调用已部署的 MCP 服务器

- 步骤1 在左侧导航栏选择“托管与运行 > 智能体运行时”。
- 步骤2 在列表单击已创建好的智能体运行时名称，进入智能体运行时详情页。
- 步骤3 在“基本信息”区域即可查看“访问域名”。

图 4-10 获取访问域名

访问方式名称	版本	描述	更新时间	操作
Latest	v2	--	2026/04/21 20:04:49 GMT+08:00	编辑

最新版本配置详情

基本信息

版本

v2

访问域名

defaultgw-gyqglyzrgl.cn-southwest-301.huaweicloud-agentarts...

镜像

swr.cn-south...

日志记录

已开启 关闭

权限与访问控制

委托

Defaultgw-gyqglyzrgl.cn-southwest-301.huaweicloud-agentarts...

入站身份认证方式

API Key 认证

URLN

apikey-1201962572

API Key 名称

APIKey-1201962572

- 步骤4 使用API调用智能体运行时
- 使用API调用智能体运行时的操作，请参考[调用智能体运行时](#)。

⚠ 注意

Streamable HTTP 要求客户端Accept头必须同时包含application/json和text/event-stream，否则服务器返回406 Not Acceptable。

----结束

4.8 调用运行时

4.8.1 调用概述

运行时部署成功后，开发者即可通过AgentArts提供的RESTful API与该运行时进行交互。AgentArts的运行时调用能力不仅限于向Agent发送消息获取回复，还涵盖了在会话隔离沙箱中动态执行系统命令、在本地与运行时沙箱环境之间进行大文件传输，以及调用用户在镜像中定义的自定义前缀接口。

运行时 API 一览

调用运行时不仅仅是向智能体发一条消息、拿到一段回复。AgentArts运行时提供了一套规范的RESTful API，覆盖智能体交互中的常见操作：

表 4-10 运行时 API

功能	API标识	接口	说明
向智能体发送并获取消息	ExecuteRuntime	POST /runtimes/{runtime_name}/invocations	最核心的调用接口。将用户输入发送给 Agent，获取 Agent 的回复。
创建会话	StartRuntimeSession	POST /runtimes/{runtime_name}/sessions-start	创建一个独立的会话实例。执行命令、文件上传下载等操作依赖会话提供的沙箱环境。
停止会话	StopRuntimeSession	POST /runtimes/{runtime_name}/sessions-stop	销毁会话实例并释放资源。会话内的文件会被自动清除。
在会话中执行Shell命令	ExecuteRuntimeCommands	POST /runtimes/{runtime_name}/commands	在会话沙箱内执行命令，返回标准输出和错误输出。
上传文件到会话	ExecuteRuntimeUploadFiles	POST /runtimes/{runtime_name}/upload-files	将本地文件上传到会话沙箱的文件系统中。

功能	API标识	接口	说明
从会话下载文件	ExecuteRuntimeDownloadFiles	GET /runtimes/{runtime_name}/download-files	将会话沙箱中的文件下载到本地。
调用自定义路径接口	ExecuteRuntimeWithPrefix	任意 /runtimes/{runtime_name}/invocations/{custom_path}	当Agent暴露了自定义路由时，通过此接口访问非标准路径。该接口支持所有HTTP方法（包括但不限于POST、GET、DELETE、PUT等）。

请求体由您的 Agent 代码决定

您的 Agent 代码定义了它接收什么字段、返回什么字段，平台网关只做鉴权和路由，不对请求体做任何修改。您在调用 API 时传入的 JSON Body，会被原样转发到您的 Agent 容器中。

这意味着：

- 请求体中应该包含哪些字段，完全取决于您编写的 Agent 代码中定义的接收逻辑。
- 响应体的结构也完全由您的 Agent 代码返回值决定。

举例说明，假设您的 Agent 代码如下：

```
@app.invocation_handler
def handle_invocation(request: dict) -> dict:
    user_msg = request.get("message", "") # 您的代码读取 "message" 字段
    # ... 业务处理 ...
    return {"reply": result, "status": "success"} # 您的代码返回这两个字段
```

将该智能体打包成镜像，部署到AgentArts运行时后，调用API时，那么调用时就必须传 "message" 字段，收到的响应就是 "reply" 和 "status" 字段。如果API参数填写为其他参数，运行时接口虽然能调用，但会因为message实际传参为空，导致智能体无法正常回答问题。

📖 说明

- 如果您团队中的不同Agent使用了不同的字段名（例如有的用message，有的用query），调用方需要分别适配。
- 调用前请与Agent开发者确认接口约定。
- 本文档示例统一使用message作为请求字段名、reply作为响应字段名，仅作演示用途。实际调用时请以您的Agent代码为准。

典型调用场景

根据业务场景的不同，AgentArts运行时支持以下几种典型交互场景：

场景一：基础对话

最简单的场景：向智能体发消息，拿到回复。

```
调用智能体（ExecuteRuntime）  
→ 发送 {"message": "你好"}  
← 收到 {"reply": "你好！有什么可以帮助你？"}
```

基础对话场景下，您只需直接调用ExecuteRuntime接口即可。在请求头中携带X-Hw-Agentarts-Session-Id，同一个Session ID的多次调用在Agent侧可以关联上下文，实现多轮对话。

场景二：涉及文件处理

当您需要让Agent处理本地文件（如分析CSV数据），流程为：

```
① 创建会话（StartRuntimeSession） → 获取沙箱环境  
② 上传文件（ExecuteRuntimeUploadFiles） → 文件进入沙箱  
③ 调用智能体（ExecuteRuntime） → Agent 读取并处理文件  
④ 下载文件（ExecuteRuntimeDownloadFiles） → 取回处理结果  
⑤ 停止会话（StopRuntimeSession） → 释放资源
```

必须先通过StartRuntimeSession创建会话来初始化沙箱环境，上传下载文件才有目标容器可操作。

创建运行时过程中需要开启“文件上传下载”功能。

场景三：涉及命令执行

当您需要在 Agent 的沙箱环境中直接执行 Shell 命令（如运行脚本、安装依赖、检查环境），流程为：

```
① 创建会话（StartRuntimeSession） → 获取沙箱环境  
② 调用智能体（ExecuteRuntime） → Agent 推理决策  
③ 执行命令（ExecuteRuntimeCommands） → 运行 Agent 决策的脚本  
④ 调用智能体（ExecuteRuntime） → Agent 基于命令结果继续推理  
⑤ 停止会话（StopRuntimeSession） → 释放资源
```

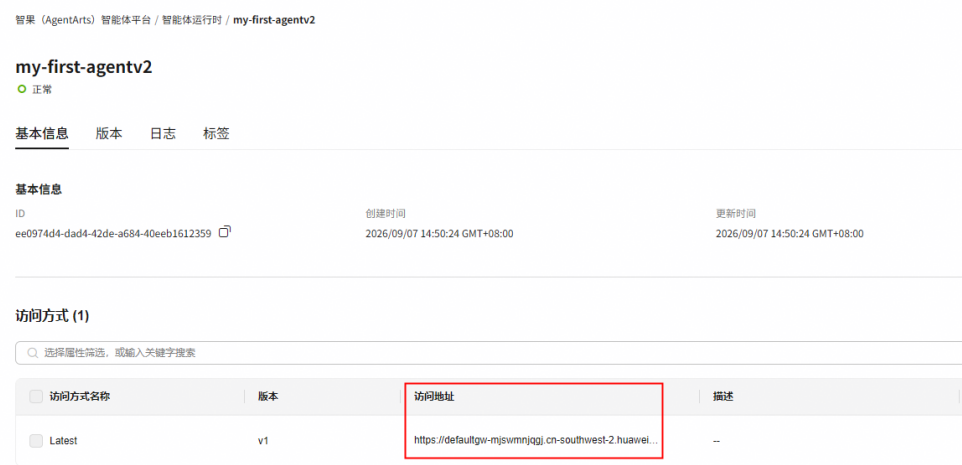
步骤②③④可以多次循环，形成“推理 → 执行 → 再推理”的闭环。

前置条件与调用依赖

在开始调用运行时之前，请确保已完成以下准备：

- 运行时已部署且状态为“正常”。您可以在AgentArts控制台的“智能体运行时”列表中查看运行时状态。
- 已获取“运行时接口”。在运行时详情页的“基本信息”区域可以找到接口地址。

图 4-11 获取运行时接口



- 已配置身份认证凭据。根据创建运行时选择的入站认证方式（IAM认证、API Key 认证或OAuth 2.0），准备好对应的认证信息。
- 如需进行文件上传或下载操作，在托管智能体，创建运行时过程中需要开启“文件上传下载”功能。如果创建时未开启，后续将无法使用文件传输能力。

图 4-12 文件上传下载



关于身份认证

每一个调用请求的Authorization请求头都必须携带有效的身份认证凭据。AgentArts运行时支持三种入站认证方式：

表 4-11 入站身份认证

认证方式	Authorization 格式	使用复杂度	适用场景
API Key	Bearer {api_key}	低，直接在请求头中写入固定凭证。	快速验证、开发调试、内部测试。
IAM 认证（AK/SK）	由签名 SDK 自动生成	高，需要使用签名 SDK对请求进行 V11-HMAC-SHA256签名。	生产环境、企业级集成。
OAuth 2.0	Bearer {OAuth_Token}	高，需要先通过 OAuth流程获取 Token。	第三方应用集成。

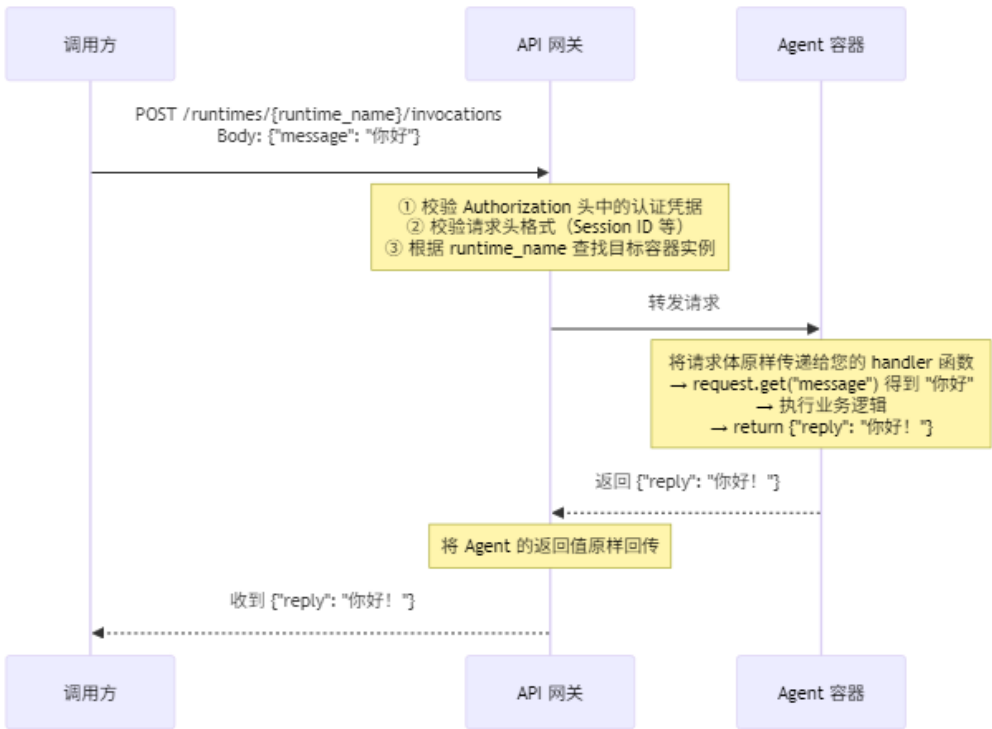
认证方式在部署运行时选定。本章节的示例以API Key认证为主（最简单直观），IAM认证的签名流程请参见[使用API调用运行时接口（IAM认证）](#)。

4.8.2 调用智能体

调用智能体运行时 - ExecuteRuntime，这是最常用的接口，向Agent发送一条消息，Agent处理后返回回复。除此之外，您也可以参考最佳实践了解如何调用运行时API。

工作原理

当您发起调用时，请求经过以下路径到达您的Agent代码：



调用链路说明：

客户端发起接口请求，携带消息体{"message": "你好"}。请求首先到达API网关，网关按序完成三项处理：

1. 身份校验：校验Authorization头中的认证凭据，确认调用方有权限访问该runtime；
2. 格式校验：校验请求头格式（如Session ID等），确保请求符合协议规范；
3. 路由分发：根据路径中的runtime_name查找对应的目标容器实例，并将请求转发过去。

请求到达Agent容器后，平台将请求体原样传递给您Agent中定义的handler函数。handler通过request.get("message")取出用户消息 "你好"，执行业务逻辑后返回结果对象{"reply": "你好!"}。

返回值再经由API网关原样回传给调用方，调用方最终收到响应{"reply": "你好!"}。

网关在整个过程中 不解析、不修改 请求体和响应体的内容，仅负责鉴权和路由。

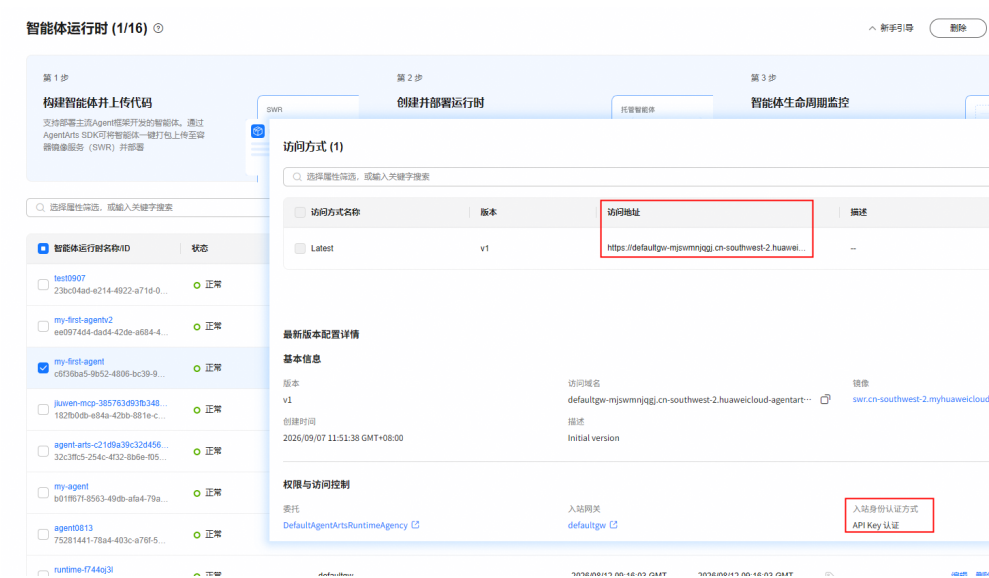
接口说明

表 4-12 接口说明

项目	说明
请求方法	POST
请求路径	/runtimes/{runtime_name}/invocations
runtime_name	您的运行时名称，可在控制台“智能体运行时”列表中获取。
endpoint 参数（可选）	查询参数 ?endpoint=xxx，指定访问方式名称以路由到特定运行时版本。不传时默认使用 Latest。

在运行时列表中找到部署的实例，在基本信息页面可以直接获取完整运行时接口。

图 4-13 获取运行时接口



请求头

表 4-13 请求头参数

请求头	是否必填	说明
Content-Type	是	固定为 application/json。
Authorization	是	身份认证凭据。API Key 方式填写 Bearer {api_key}；IAM 方式由签名 SDK 自动生成；OAuth 2.0 方式填写 Bearer {jwt_token}。

请求头	是否必填	说明
X-Hw-Agentarts-Session-Id	是	会话 ID，用于标识一次会话。您可以自行生成一个字符串（如 UUID），同一个 Session ID 的多次调用在 Agent 侧可关联上下文。由英文、数字、-、_ 组成，不超过 64 个字符。
X-Sdk-Content-Sha256	IAM认证时必填	如果智能体运行时的入站认证类型为IAM认证时，需要指定该Header头为UNSIGNED-PAYLOAD。
X-Hw-Agentgateway-User-Id	否	用户标识，用于区分不同终端用户。由英文、数字、-、_ 组成，不超过 128 个字符。

请求体

请求体为JSON格式，具体字段由您的Agent代码定义。网关不做任何校验，原样转发。

以下是一个常见的请求体示例（假设您的Agent代码使用message字段接收用户输入）：

```
{
  "message": "你好，请介绍一下你自己"
}
```

响应体

响应体同样为JSON格式，结构由您的Agent代码的返回值决定。

调用示例

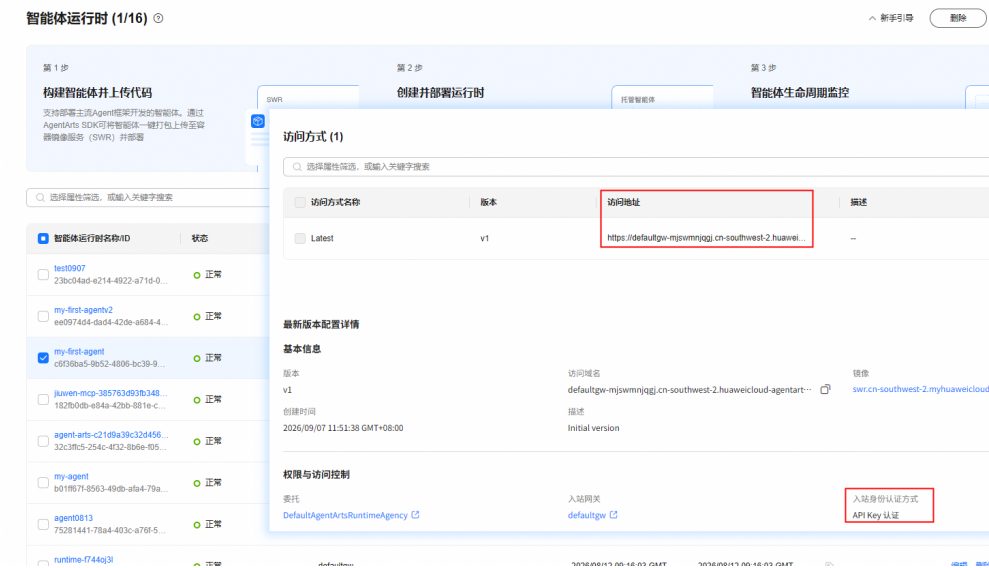
本示例主要演示通过API Key认证调用运行时接口，除了本文档之外，您也可以参考最佳实践了解如何调用运行时API。

最佳实践：[使用API调用运行时接口（IAM认证）](#)

最佳实践：[使用API调用运行时接口（API Key认证）](#)

- 步骤1 通过AgentArts页面部署运行时，请在“入站身份认证”处选择API Key认证。
- 在运行时列表中找到部署的实例，在基本信息页面获取运行时接口。

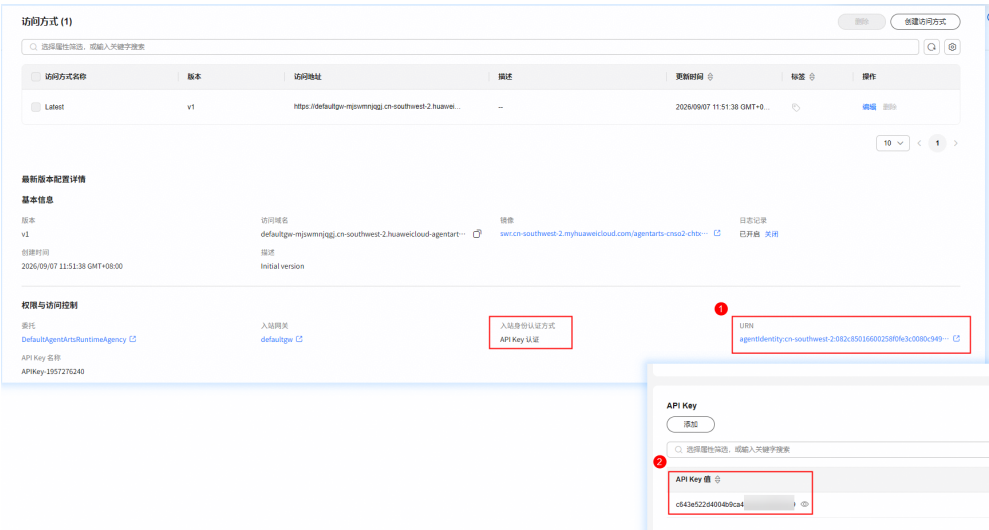
图 4-14 获取运行时接口



步骤2 单击运行时名称，进入基本信息页面。找到URN，单击URN名称，获取API Key的值。

获取到的API Key值后，调用时需在API请求头中拼接为Authorization: Bearer {api_key} 格式（注意Bearer后有一个空格）。

图 4-15 获取 API Key 值



步骤3 调用运行时API。

示例中的"message"是请求体中的一个字段，用于传递您的问题。该字段名称并非固定值，而是由您部署的智能体服务代码中实际定义的入参字段决定的。本示例使用"message"作为演示，实际调用时请根据您的服务代码中定义的字段名进行替换。

示例中的接口地址（API_URL）、Authorization、x-hw-agentarts-session-id请按实际情况填写参数值。

Python调用示例：

```
import requests
import json
```

```
# 替换为真实的API接口
url = "https://{endpoint}/runtimes/{runtime_name}/invocations"

payload = json.dumps({
    "message": "你好"
})

headers = {
    'Content-Type': 'application/json',
    'Authorization': '您的Authorization, 格式为Bearer {api_key}',
    'x-hw-agentarts-session-id': '您的会话ID, 由用户自定义, 由英文、数字、“-”、“_”组成, 不超过64位字符。'
}

response = requests.request("POST", url, headers=headers, data=payload)
print(response.text)
```

cURL调用示例:

针对不同操作系统，命令的转义和换行符有所不同。

- Windows

在CMD（命令提示符）中执行如下命令：

```
curl -X POST "https://{endpoint}/runtimes/{runtime_name}/invocations" ^
-H "Content-Type: application/json" ^
-H "Authorization: 您的Authorization, 格式为Bearer {api_key}" ^
-H "x-hw-agentarts-session-id: 您的会话ID" ^
-d "{ \"message\": \"你好\" }"
```

- Linux/macOS

在Terminal（终端）中执行如下命令：

```
curl -X POST "https://{endpoint}/runtimes/{runtime_name}/invocations" \
-H "Content-Type: application/json" \
-H "Authorization: 您的Authorization, 格式为Bearer {api_key}" \
-H "x-hw-agentarts-session-id: 您的会话ID" \
-d '{
    "message": "你好"
}'
```

----结束

多轮对话

同一个X-Hw-Agentarts-Session-Id下的多次调用，在同一个会话上下文中执行，Agent侧可以基于此会话ID关联多次调用的内容，从而实现多轮对话。

```
SESSION_ID = "23e4567e89b12d3a456426614174000"

# 第一轮
resp1 = call_agent(session_id=SESSION_ID, message="我叫张三")
# → Agent 回复: "你好, 张三! 有什么可以帮助你?"

# 第二轮 (相同 Session ID)
resp2 = call_agent(session_id=SESSION_ID, message="你还记得我叫什么吗? ")
# → Agent 回复: "你叫张三。"

# 第三轮 (换一个新的 Session ID → 新会话, 上下文清空)
resp3 = call_agent(session_id="another-session", message="你还记得我叫什么吗? ")
# → Agent 回复: "抱歉, 我不清楚你的名字。"
```

 说明

X-Hw-Agentarts-Session-Id提供的是会话标识，用于平台将多次请求路由到同一个会话上下文。但上下文是否能保持（即Agent能否记住之前的对话），取决于您的Agent代码是否实现了对话记忆逻辑（例如集成了记忆库组件，关于记忆库的使用，可参考[进阶示例：构建全能出行助手（集成模型/记忆/代码解释器/网关/高德mcp）](#)）。

指定访问方式（版本路由）

如果您为同一个运行时创建了多个访问方式（如dev、staging、prod），可以通过endpoint查询参数指定调用哪个版本：

```
# 调用 dev 版本
POST https://{域名}/runtimes/{runtime_name}/invocations?endpoint=dev

# 调用 prod 版本
POST https://{域名}/runtimes/{runtime_name}/invocations?endpoint=prod

# 不指定时，默认使用 Latest 版本
POST https://{域名}/runtimes/{runtime_name}/invocations
```

访问方式的创建和管理请参见[管理访问方式](#)。

4.8.3 会话操作

什么是会话

会话是AgentArts运行时中的一个逻辑单元，它背后对应着一个独立的安全沙箱环境。

表 4-14 相关概念

概念	说明
会话 ID	一个字符串标识符，用于在请求中指明使用哪个会话。
沙箱	实际的资源容器，包含独立的文件系统、内存空间和进程环境。
创建会话	分配一个沙箱实例，并将其与会话 ID 绑定，完成环境预热。

AgentArts运行时提供了显式的会话生命周期管理，通过[StartRuntimeSession](#)创建会话，通过[StopRuntimeSession](#)停止会话。**理解会话的两种创建方式：**

- **隐式创建：**调用ExecuteRuntime（即/invocations）接口时，在请求头中携带自定义的X-Hw-Agentarts-Session-Id，平台会在该会话不存在时自动创建。这是大多数场景推荐的方式。
- **显式创建：**调用StartRuntimeSession接口，由平台分配并返回session_id。适用于需要精细控制会话生命周期的场景。

调用StartRuntimeSession的本质是：提前向平台申请一个沙箱实例，完成环境预热，然后绑定到您指定的会话ID上。这样后续的第一次实际请求可以立即执行，无需等待沙箱拉起。

什么场景需要显式的创建会话

并非所有场景都需要显式调用StartRuntimeSession接口：

表 4-15 会话接口调用场景

场景	是否需要调用会话接口	原因
基础对话（仅调用ExecuteRuntime）	否	直接在请求头中传入自定义的X-Hw-Agentarts-Session-Id（会话ID），平台会自动创建会话。
执行命令（ExecuteRuntime Commands）	否	也可以直接在请求头中传入自定义会话ID进行隐式创建。但如果需要在执行命令前确保沙箱资源就绪，建议先显式创建。
上传/下载文件	否	同样支持隐式创建。但如果需要上传大文件或批量文件，建议先显式创建会话，确保沙箱资源已分配。
精细控制会话生命周期（如预先分配资源、主动管理会话状态）	是	需要调用 StartRuntimeSession 获取平台分配的会话ID，后续所有操作使用此ID。

如果只是进行对话交互，您完全不需要阅读本章节。直接参考[调用智能体](#)章节即可。仅当您需要执行命令或传输文件，或者需要精细控制会话生命周期时，才需要关注本章节的内容。

创建会话

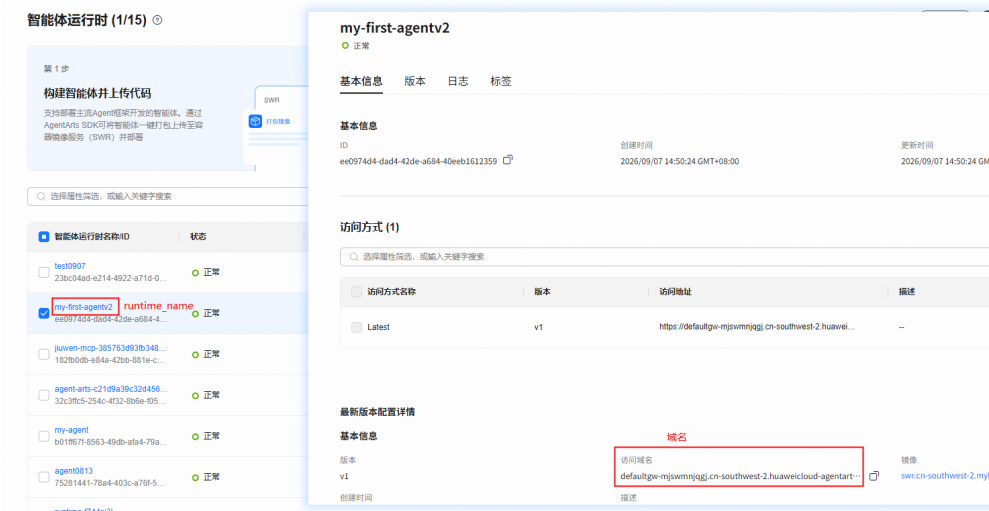
如果需要显式创建会话，调用[StartRuntimeSession](#)接口：

POST https://{域名}/runtimes/{runtime_name}/sessions-start

参数说明：

- 域名：在AgentArts控制台运行时详情页的“基本信息”区域获取。
- runtime_name：运行时名称，在控制台运行时列表中获取。

图 4-16 获取接口参数取值



请求示例

以下示例使用API Key鉴权方式进行演示。

```
curl -X POST "https://{域名}/runtimes/{runtime_name}/sessions-start" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer {api_key}"
```

成功响应

接口调用成功后，会返回一个会话ID。

```
"session_id": "abc123-def456-..."
```

创建成功后，后续的命令执行、文件上传下载等操作，都需要在请求头X-Hw-Agentarts-Session-Id中携带此session_id：

```
curl -X POST "https://{域名}/runtimes/{runtime_name}/commands" \
-H "Authorization: Bearer {api_key}" \
-H "X-Hw-Agentarts-Session-Id: abc123-def456-ghi789" \
-d '{"command": ["ls", "-la"]}'
```

停止会话

使用完毕后，调用StopRuntimeSession释放沙箱资源：

- **请求示例：**

```
POST https://{域名}/runtimes/{runtime_name}/sessions-stop
Headers:
  Authorization: Bearer {api_key}
  X-Hw-Agentarts-Session-Id: {session_id}
```
- **成功响应：**

成功响应后接口会返回200状态码。

说明

- 停止会话后，沙箱实例被销毁，沙箱临时目录中的文件会被自动清除且无法恢复。请在停止前通过DownloadFiles接口下载所有需要保留的结果文件。
- 如果创建运行时过程中，已配置会话存储功能，存储目录中的文件不受影响，会被持久化保留。

4.8.4 执行命令

执行命令接口**ExecuteRuntimeCommands**允许在运行中的会话沙箱内直接执行Shell命令。命令和Agent运行在同一个容器环境中，共享文件系统。

适用场景：

- “推理 → 执行”循环：Agent做推理决策后，通过此接口执行具体操作（如运行脚本），然后将执行结果反馈给Agent继续推理。
- 环境检查与调试：查看沙箱内的文件列表、Python版本、已安装的库等。
- 安装额外依赖：在当前会话中临时安装Python包或其他工具。
- 运行用户脚本：执行预先上传到沙箱中的脚本文件。

接口说明

表 4-16 接口说明

项目	说明
请求方法	POST
请求路径	/runtimes/{runtime_name}/commands
前置条件	必须先有一个有效的会话。支持两种方式： - 在请求头中传入自定义的X-Hw-Agentarts-Session-Id（隐式创建）； - 先调用StartRuntimeSession获取平台分配的session_id（显式创建）。两种方式均可。

请求头

表 4-17 请求头参数

请求头	是否必填	说明
Content-Type	是	固定为application/json。
Authorization	是	身份认证凭据。API Key 方式填写 Bearer {api_key}；IAM 方式由签名 SDK 自动生成；OAuth 2.0 方式填写 Bearer {jwt_token}。
X-Hw-Agentarts-Session-Id	是	会话 ID，用于标识一次会话。您可以自行生成一个字符串（如 UUID），同一个 Session ID 的多次调用在 Agent 侧可关联上下文。由英文、数字、-、_ 组成，不超过 64 个字符。

请求头	是否必填	说明
X-Sdk-Content-Sha256	IAM认证时必填	如果智能体运行时的入站认证类型为IAM认证时，需要指定该Header头为UNSIGNED-PAYLOAD。
Command-Type	否	设置为chunked时启用NDJSON流式响应模式，命令执行过程中实时推送输出。默认为同步模式（等待命令执行完成后返回完整结果）。

请求体

请求体为JSON格式，包含需要执行的命令信息。

```
{
  "command": ["可执行程序", "参数1", "参数2"],
  "timeout": 300
}
```

参数说明：

- command：必填，表示命令及参数数组。第一个元素为可执行文件路径或命令，后续元素为参数。数组元素个数1-1024，每个元素长度1-8192字符。注意command字段是一个数组（而不是字符串）。
- timeout：可选，命令执行超时时间（秒）。范围1-3600，默认300（5分钟）。超时后命令会被强制终止。

常见写法对照

表 4-18 命令写法

想执行的命令	command数组写法
ls -la /workspace/	["ls", "-la", "/workspace/"]
python script.py	["python", "script.py"]
bash test.sh	["bash", "test.sh"]
pip install pandas	["pip", "install", "pandas"]
python -c "print('hello')"	["python", "-c", "print('hello')"]

响应格式

同步模式响应

```
{
  "exit_code": 0,
  "stdout": "命令的标准输出内容",
}
```



```
"stderr": "命令的错误输出内容"
}
```

参数说明：

- **exit_code**：命令退出码。0表示成功，非0表示失败。注意：即使命令执行失败（**exit_code**非0），HTTP状态码仍然返回200，因为“命令已执行”这件事本身是成功的，只是命令结果失败。
- **stdout**：标准输出内容。如果输出超过 100MB，会被按比例截断并添加 TRUNCATED 标记。
- **stderr**：标准错误输出内容。

流式模式响应（Command-Type: chunked）

启用Command-Type: chunked后，响应为NDJSON（Newline Delimited JSON）格式，每行一个JSON对象，命令执行过程中实时推送：

```
{"type": "stdout", "data": "step_1\n"}
{"type": "stdout", "data": "step_2\n"}
{"type": "stderr", "data": "warning: something\n"}
{"type": "exit", "data": {"exit_code": 0}}
```

参数说明：

- **type**：消息类型：stdout（标准输出）、stderr（标准错误）、exit（命令结束，包含最终退出码）。
- **data**：stdout/stderr类型时为输出内容字符串；exit类型时为包含**exit_code**的对象。

调用示例

以下示例使用API Key鉴权方式进行演示。

• 示例一：查看文件列表

```
curl -X POST "https://{域名}/runtimes/{runtime_name}/commands" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer {api_key}" \
-H "X-Hw-Agentarts-Session-Id: {session_id}" \
-d '{
  "command": ["ls", "-la", "/workspace/"]
}'
```

• 示例二：安装Python包并验证

```
# 步骤 1：安装
curl -X POST "https://{域名}/runtimes/{runtime_name}/commands" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer {api_key}" \
-H "X-Hw-Agentarts-Session-Id: {session_id}" \
-d '{
  "command": ["pip", "install", "pandas", "matplotlib"],
  "timeout": 120
}'

# 步骤 2：验证安装结果
curl -X POST "https://{域名}/runtimes/{runtime_name}/commands" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer {api_key}" \
-H "X-Hw-Agentarts-Session-Id: {session_id}" \
-d '{
  "command": ["python", "-c", "import pandas; print(pandas.__version__)"]
}'
```

注意：安装的包仅在当前会话内有效。会话停止后，沙箱环境会被销毁，安装的包不会保留。

- **示例三：Python调用**

```
import requests
import json

url = f"https://{域名}/runtimes/{runtime_name}/commands"

headers = {
    "Content-Type": "application/json",
    "Authorization": f"Bearer {API_KEY}",
    "X-Hw-Agentarts-Session-Id": session_id
}

# 执行命令
payload = {
    "command": ["python", "--version"],
    "timeout": 30
}

try:
    resp = requests.post(url, headers=headers, json=payload, timeout=60)
    result = resp.json()

    if result["exit_code"] == 0:
        print(f"执行成功:\n{result['stdout']}")
    else:
        print(f"执行失败 ( exit_code={result['exit_code']} ) :\n{result['stderr']}")
except requests.exceptions.Timeout:
    print("请求超时")
except requests.exceptions.RequestException as e:
    print(f"网络请求异常: {e}")
```

- **示例四：流式输出模式**

流式模式适合执行日志查看、进度输出频繁的命令，避免长时间等待后一次性返回带来的超时风险。

对于长时间运行的命令（如tail -f、持续输出的脚本），可以在请求头中添加Command-Type: chunked启用NDJSON流式响应，执行过程中实时推送输出：

```
curl -X POST "https://{域名}/runtimes/{runtime_name}/commands" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer {api_key}" \
-H "X-Hw-Agentarts-Session-Id: {session_id}" \
-H "Command-Type: chunked" \
-d '{
  "command": ["bash", "-c", "for i in 1 2 3; do echo step_$i; sleep 1; done"],
  "timeout": 30
}'
```

- **示例五：检查Python版本**

```
curl -X POST "https://{域名}/runtimes/{runtime_name}/commands" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer {api_key}" \
-H "X-Hw-Agentarts-Session-Id: {session_id}" \
-d '{
  "command": ["python", "--version"]
}'
```

4.8.5 文件上传与下载

[ExecuteRuntimeUploadFiles](#)接口，允许将本地文件上传到会话沙箱的文件系统中。上传后，Agent代码可以通过文件路径直接读取。

文件上传接口说明

表 4-19 接口说明

项目	说明
请求方法	POST
请求路径	/runtimes/{runtime_name}/upload-files
关键查询参数	path（必填），表示目标文件路径或目录路径。

表 4-20 Query 参数说明

请求头	是否必填	说明
path	是	目标路径。支持两种写法： - 指定完整文件路径（如/workspace/data.csv），适用于单文件上传。 - 指定目录路径（如/workspace/，以/结尾），文件会自动保存到该目录下。不能使用 /（根目录），不能使用 ..（相对路径回退）。
file_mode	否	设置文件权限（Linux八进制权限，如0644）。默认由镜像的umask决定。
user_id	否	设置文件所属用户（Linux UID）。默认为当前启动镜像的用户UID。
group_id	否	设置文件所属用户组（Linux GID）。默认为当前启动镜像的用户组GID。
endpoint	否	访问方式名称，用于指定运行时版本。默认Latest。

- 仅传文件名时（如path=data.csv），文件会自动保存到用户主目录 ~/ 下。
- 单文件不超过100MB。
- 多文件总计不超过 500MB（仅multipart表单模式）。

表 4-21 请求头参数

请求头	是否必填	说明
Content-Type	是	固定为 application/json。
Authorization	是	身份认证凭据。API Key 方式填写 Bearer {api_key}；IAM 方式由签名 SDK 自动生成；OAuth 2.0 方式填写 Bearer {jwt_token}。

请求头	是否必填	说明
X-Hw-Agentarts-Session-Id	是	会话 ID，用于标识一次会话。您可以自行生成一个字符串（如 UUID），同一个 Session ID 的多次调用在 Agent 侧可关联上下文。由英文、数字、-、_ 组成，不超过 64 个字符。
X-Sdk-Content-Sha256	IAM认证时必填	如果智能体运行时的入站认证类型为IAM认证时，需要指定该Header头为UNSIGNED-PAYLOAD。
X-Hw-Agentgateway-User-Id	否	用户标识，用于区分不同终端用户。由英文、数字、-、_ 组成，不超过 128 个字符。

三种文件上传方式

AgentArts支持三种上传方式，适用于不同场景：

表 4-22 文件上传方式

上传方式	Content-Type	path参数含义	适用场景
单文件流式上传	application/octet-stream	目标文件的完整路径（如 /workspace/data.csv）。	上传单个文件。
tar包上传	application/x-tar	目标目录路径（如 /workspace/）。	一次上传多个文件，打包为 tar 格式，服务端自动解压。
表单上传	multipart/form-data	目标目录路径（如 /workspace/）。	一次上传多个文件，使用标准表单格式。

表单上传和tar上传时，path参数应指定为目录路径（以/结尾），文件会保存到该目录下。

上传文件调用示例

以下示例使用API Key鉴权方式进行演示。

• 示例一：单文件上传（最常用）

```
curl -X POST \
  "https://{域名}/runtimes/{runtime_name}/upload-files?path=/workspace/data.csv" \
  -H "Authorization: Bearer {api_key}" \
  -H "X-Hw-Agentarts-Session-Id: {session_id}" \
  -H "Content-Type: application/octet-stream" \
  --data-binary @./local_data.csv
```

● 示例二：多文件表单上传

```
curl -X POST \
  "https://{域名}/runtimes/{runtime_name}/upload-files?path=/workspace/" \
  -H "Authorization: Bearer {api_key}" \
  -H "X-Hw-Agentarts-Session-Id: {session_id}" \
  -F "file1=@./data.csv" \
  -F "file2=@./config.json"
```

● 示例三：tar包批量上传

```
# 1. 先在本地打包
tar -czf bundle.tar.gz ./data/*

# 2. 上传 tar 包（服务端自动解压到 /workspace/）
curl -X POST \
  "https://{域名}/runtimes/{runtime_name}/upload-files?path=/workspace/" \
  -H "Authorization: Bearer {api_key}" \
  -H "X-Hw-Agentarts-Session-Id: {session_id}" \
  -H "Content-Type: application/x-tar" \
  --data-binary @./bundle.tar.gz
```

下载文件接口说明

ExecuteRuntimeDownloadFiles接口，将会话沙箱中的文件下载到本地。典型用法：下载 Agent 生成的分析报告、表格、代码文件等结果。

表 4-23 接口说明

项目	说明
请求方法	GET
请求路径	/runtimes/{runtime_name}/download-files
关键查询参数	path（必填），表示源文件路径；recursive（可选），表示是否按归档包下载。

表 4-24 Query 参数说明

请求头	是否必填	说明
path	是	沙箱内需要下载的文件路径（URL编码格式）。支持绝对路径和相对路径。
recursive	否	默认false（单文件下载）。设为true时按tar归档包下载，适用于下载整个目录下的所有文件。
endpoint	否	访问方式名称，用于指定运行时版本。默认Latest。

表 4-25 请求头参数

请求头	是否必填	说明
Content-Type	是	固定为 application/json。

请求头	是否必填	说明
Authorization	是	身份认证凭据。API Key 方式填写 Bearer {api_key}; IAM 方式由签名 SDK 自动生成; OAuth 2.0 方式填写 Bearer {jwt_token}。
X-Hw-Agentarts-Session-Id	是	会话 ID，用于标识一次会话。您可以自行生成一个字符串（如 UUID），同一个 Session ID 的多次调用在 Agent 侧可关联上下文。由英文、数字、-、_ 组成，不超过 64 个字符。
X-Sdk-Content-Sha256	IAM认证时必填	如果智能体运行时的入站认证类型为IAM 认证时，需要指定该Header头为 UNSIGNED-PAYLOAD。
X-Hw-Agentgateway-User-Id	否	用户标识，用于区分不同终端用户。由英文、数字、-、_ 组成，不超过 128 个字符。

文件下载调用示例

以下示例使用API Key鉴权方式进行演示。

- 示例一：下载单个文件

```
curl -X GET \
  "https://{域名}/runtimes/{runtime_name}/download-files?path=/workspace/report.pdf" \
  -H "Authorization: Bearer {api_key}" \
  -H "X-Hw-Agentarts-Session-Id: {session_id}" \
  -o report.pdf
```
- 示例二：下载整个目录

```
curl -X GET \
  "https://{域名}/runtimes/{runtime_name}/download-files?path=/workspace/output/&recursive=true" \
  -H "Authorization: Bearer {api_key}" \
  -H "X-Hw-Agentarts-Session-Id: {session_id}" \
  -o output.tar
```

4.8.6 自定义路径调用

ExecuteRuntimeWithPrefix接口，用于通过前缀匹配方式调用已部署的智能体运行时中的自定义接口。使用该接口需确保智能体运行时的路由匹配模式设置为**前缀匹配（PREFIX_MATCH）**。

图 4-17 前缀匹配



使用场景

标准的`ExecuteRuntime`（`POST /invocations`）接口适用于大多数对话场景。但在很多实际场景中，一个Agent不只有一个接口，它可能同时提供对话、健康检查、批量处理等多种功能。这时就需要通过自定义路径调用不同的接口。

- 多功能接口：除了对话接口外，Agent还提供健康检查（`/health`）、状态查询（`/status`）、批量处理（`/batch`）等能力。
- RESTful API：Agent提供了完整的RESTful风格接口，如`GET /users`、`POST /tasks`、`DELETE /tasks/{id}`等。
- MCP Server 集成：部署为MCP Server的运行时需要通过自定义路径接收MCP协议请求。
- 回调通知：Agent需要接收外部系统的Webhook回调。

工作原理

当您发起自定义接口调用时，请求经过以下路径到达Agent代码：

调用链路说明：

1. 客户端发起接口请求，例如`POST /runtimes/{runtime_name}/invocations/chat`，携带请求体`{"input": "hello"}`。
2. 请求首先到达API网关，网关按序完成三项处理：
 - **身份校验**：校验Authorization请求头中的认证凭据，确认调用方有权限访问该运行时。
 - **格式校验**：校验请求头格式（如Session ID等），确保请求符合协议规范。
 - **路由分发**：根据路径中的`runtime_name`查找对应的目标容器实例，并将请求转发过去。
3. 请求到达Agent容器后，平台将请求原样传递给Agent中定义的对应该路由处理函数（如`/chat`）。
4. handler执行业务逻辑后返回结果对象`{"output": "hello"}`。
5. 返回值再经由API网关原样回传给调用方。

网关在整个过程中**不解析、不修改**请求体和响应体的内容，仅负责鉴权和路由。

与标准调用的区别：

标准调用接口（`ExecuteRuntime`）路径为`/runtimes/{runtime_name}/invocations`，只能调用Agent的默认入口。

而自定义接口在前缀匹配模式下支持`invocations/{custom_path}`路径，允许调用Agent代码中定义的任意自定义路由，如`/chat`、`/query`、`/search`等，且支持所有HTTP方法（`POST`、`GET`、`PUT`、`DELETE`等）。

路径映射原理

理解自定义路径调用的方法，需要理解“API接口 → Agent容器路由”的映射关系。当您在Agent代码中注册了多个路由处理器时，比如：

```
from agentarts.sdk import AgentArtsRuntimeApp

app = AgentArtsRuntimeApp()

# 路由 1：标准对话（固定路径 /invocations）
```

```
@app.invocation_handler
def handle_invocation(request: dict) -> dict:
    message = request.get("message", "")
    return {"reply": f"你说了: {message}"}

# 路由 2: 健康检查 (自定义路径 /health)
@app.handler("/health")
def handle_health(request: dict) -> dict:
    return {"status": "ok", "version": "1.0.0"}

# 路由 3: 批量处理 (自定义路径 /batch)
@app.handler("/batch")
def handle_batch(request: dict) -> dict:
    tasks = request.get("tasks", [])
    results = [{"task": t, "done": True} for t in tasks]
    return {"processed": len(tasks), "results": results}
```

那么，调用的API接口和Agent路由的对应关系如下。

表 4-26 对应关系说明

调用的API接口	网关转发到Agent容器的路由	匹配的处理函数
POST /runtimes/{name}/invocations	/invocations	handle_invocation
GET /runtimes/{name}/invocations/health	/health	handle_health
POST /runtimes/{name}/invocations/batch	/batch	handle_batch

API接口中/invocations/后面跟的部分（即custom_path），就是Agent代码中@app.handler()注册的路由路径。

前置条件

使用自定义路径调用前，请确保部署运行时已将URL匹配模式设置为PREFIX_MATCH（前缀匹配）。如果使用默认的精确匹配模式（ACCURATE_MATCH），非 /invocations 路径的请求会被网关直接拒绝。

配置方式：

- 方式一：在.agentarts_config.yaml配置文件中声明：

```
runtime:
  invoke_config:
    url_match_type: PREFIX_MATCH
```
- 方式二：在AgentArts控制台托管智能体，创建运行时过程中，在高级配置中将“路由配置”设置为“前缀匹配”。
前缀匹配支持 /runtimes/{runtime_name}/invocations 以及 /runtimes/{runtime_name}/invocations/{custom_path} 的调用路径；一旦选择前缀匹配，后续镜像中可以新增、删除接口等。

图 4-18 前缀匹配



接口介绍

表 4-27 接口说明

项目	说明
请求方法	支持所有 HTTP 方法（POST、GET、PUT、DELETE 等），请按后端实际接口方法调用。
请求路径	/runtimes/{runtime_name}/invocations/{custom_path}
runtime_name	您的运行时名称，可在控制台"智能体运行时"列表中获得。以小写字母开头，以小写字母或数字结尾，包含小写字母、数字和中划线，长度 2-48 个字符。
custom_path	需要调用的自定义接口URL路径， 不能以/开头 ，需确保运行时路由匹配模式设置为前缀匹配。包含英文字母、数字和特殊字符（- _ . ~ : / ? # [] @ ! \$ & ' () * + , ; = % ），长度1-256个字符。
endpoint 参数（可选）	查询参数?endpoint=xxx，指定访问方式名称以路由到特定运行时版本。不传时默认使用Latest。

表 4-28 请求头参数

请求头	是否必填	说明
Content-Type	是	固定为 application/json。
Authorization	是	身份认证凭据。API Key 方式填写 Bearer {api_key}；IAM 方式由签名 SDK 自动生成；OAuth 2.0 方式填写 Bearer {jwt_token}。
X-Hw-Agentarts-Session-Id	是	会话 ID，用于标识一次会话。您可以自行生成一个字符串（如 UUID），同一个 Session ID 的多次调用在 Agent 侧可关联上下文。由英文、数字、-、_ 组成，不超过 64 个字符。
X-Sdk-Content-Sha256	IAM认证时必填	如果智能体运行时的入站认证类型为IAM 认证时，需要指定该Header头为 UNSIGNED-PAYLOAD。

请求头	是否必填	说明
X-Hw-Agentgateway-User-Id	否	用户标识，用于区分不同终端用户。由英文、数字、-、_ 组成，不超过 128 个字符。
Mcp-Session-Id	否	本次MCP调用对应的会话唯一ID，用于标识同一次会话上下文，实现会话级别的状态保持、日志追踪与问题排查。同一会话上下文内应保持一致。由英文、数字、-、_ 组成，不超过128位字符。
Mcp-Protocol-Version	否	本次MCP协议通信版本，适配 Streamable HTTP传输。客户端可指定版本，未传时服务端默认采用 2025-03-26。支持 2025-03-26、2025-06-18、2025-11-25。

请求体

请求体格式由您的Agent代码定义。网关不做任何校验，原样转发。以下是一个常见的请求体示例（假设您的Agent代码使用message字段接收用户输入）：

```
{  
  "message": "hello"  
}
```

响应体

响应Body为Agent代码返回的结果，格式由您的Agent代码决定。

```
{  
  "output": "hello"  
}
```

表 4-29 错误响应

状态码	说明	响应示例
401	未授权（认证令牌缺失、无效或已过期）	{"code": 401, "message": "Unauthorized"}
404	运行时资源不存在	{"code": 404, "message": "runtime not found"}
500	服务器内部错误	{"code": 500, "message": "internal error"}

调用示例

以下示例使用API Key鉴权方式进行演示。

第一步：在Agent代码中注册路由

```
from agentarts.sdk import AgentArtsRuntimeApp

app = AgentArtsRuntimeApp()

@app.invocation_handler
def handle_invocation(request: dict) -> dict:
    """标准对话接口"""
    message = request.get("message", "")
    return {"reply": f"你说了: {message}"}

@app.handler("/health")
def handle_health(request: dict) -> dict:
    """健康检查接口，供监控系统定时探测"""
    return {"status": "ok", "version": "1.0.0"}

@app.handler("/batch")
def handle_batch(request: dict) -> dict:
    """批量处理接口，一次提交多条任务"""
    tasks = request.get("tasks", [])
    results = [{"task": t, "done": True} for t in tasks]
    return {"processed": len(tasks), "results": results}
```

第二步：部署时选择前缀匹配模式

在控制台部署运行时，将URL的路由匹配模式设置为“前缀匹配”。

第三步：调用各个接口

- 调用标准对话接口（@app.invocation_handler → /invocations）

```
curl -X POST \
  "https://{域名}/runtimes/{runtime_name}/invocations" \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer {API_Key}" \
  -H "X-Hw-Agentarts-Session-Id: session-001" \
  -d '{"message": "你好"}'
```
- 调用健康检查接口（@app.handler("/health") → /invocations/health）

```
curl -X GET \
  "https://{域名}/runtimes/{runtime_name}/invocations/health" \
  -H "Authorization: Bearer {API_Key}" \
  -H "X-Hw-Agentarts-Session-Id: session-001"
```
- 调用批量处理接口（@app.handler("/batch") → /invocations/batch）

```
curl -X POST \
  "https://{域名}/runtimes/{runtime_name}/invocations/batch" \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer {API_Key}" \
  -H "X-Hw-Agentarts-Session-Id: session-001" \
  -d '{"tasks": ["任务A", "任务B", "任务C"]}'
```
- 通过访问方式指定版本
 与标准调用一样，自定义路径调用也支持通过endpoint查询参数指定版本：
 # 调用dev版本的/batch 接口

```
POST https://{网关域名}/runtimes/{runtime_name}/invocations/batch?endpoint=dev
```

4.8.7 调用 MCP 服务

MCP Server部署成功后，通过运行时的访问方式获取MCP Server的端点 URL，然后用MCP 客户端连接并调用。

获取端点 URL

步骤1 登录AgentArts智能体平台。

步骤2 在左侧导航栏选择“托管与运行 > 智能体运行时”。

步骤3 在列表单击已创建好的智能体运行时名称，进入智能体运行时详情页。

步骤4 在“基本信息”区域即可查看“访问域名”。

图 4-19 获取访问域名



---结束

通过 curl 调用

列出工具

```
curl -X POST https://${RUNTIME_ENDPOINT} \ # RUNTIME_ENDPOINT的获取方式：在智能体运行时列表单击名称，在详情页面的“访问方式”中的“访问地址”
-H "Content-Type: application/json" \
-H 'X-Hw-Agentarts-Session-Id: {xxxxxxxxx}' \
-H 'Authorization: Bearer {apikey}' \
-H "Accept: application/json, text/event-stream" \
-d '{"jsonrpc":"2.0","id":1,"method":"tools/list"}'
```

调用工具

```
curl -X POST https://${RUNTIME_ENDPOINT} \
-H "Content-Type: application/json" \
-H "Accept: application/json, text/event-stream" \
-H 'Authorization: Bearer {apikey}' \
-d '{
  "jsonrpc": "2.0",
  "id": 2,
  "method": "tools/call",
  "params": {
    "name": "add",
    "arguments": {"a": 3, "b": 5}
  }
}'
```

如果返回了正确的JSON-RPC结果，说明云上部署和调测成功。

说明

Streamable HTTP要求客户端Accept头必须同时包含application/json和text/event-stream，否则服务器返回406 Not Acceptable。

相关文档

[制作MCP Server镜像](#)

4.9 会话管理

会话（Session）是智能体运行时中隔离执行和状态管理的核心单元。每个会话运行在独立的安全沙箱中，支持会话级状态持久化，使智能体能够在多次调用间保持上下文。通过会话管理，可实现多轮对话、命令执行、文件传输等高级交互能力。

约束与限制

- 会话ID由调用方自行生成，由英文、数字、中划线（-）、下划线（_）组成，不超过 64 个字符。
- 配置会话存储后，不支持取消（关闭），仅支持修改挂载路径。
- 会话存储仅支持配置 1 个。
- 会话存储底层基于OBS 对象存储，通过FUSE挂载到沙箱容器。配置umask后，文件权限由挂载时的umask、uid、gid参数决定，运行时执行chmod/chown命令不会生效。
- 配置会话存储后，请谨慎使用同一会话ID进行跨访问方式访问，版本不一致会导致调用超时。
- 空闲会话超时和最大存活时间取值范围为60 - 604800 秒（1 分钟 ~ 7 天）。

原理介绍

会话隔离机制

每个会话运行在独立的安全沙箱（microVM）实例中，实现进程级、文件系统级和网络级的隔离。同一会话ID的多次调用会被路由到同一个沙箱实例（会话亲和性），不同会话ID的调用彼此隔离。

会话生命周期

会话从创建到销毁经历以下阶段：

表 4-30 会话生命周期

阶段	API 接口	平台动作	说明
创建会话	POST /runtimes/{runtime_name}/sessions-start	创建沙箱和会话	创建独立沙箱实例并生成会话。
会话活跃期	POST /runtimes/{runtime_name}/invocations	调用沙箱	通过会话ID路由到同一沙箱，支持命令执行和文件传输。
停止会话	POST /runtimes/{runtime_name}/sessions-stop	销毁沙箱	销毁沙箱实例，释放计算资源。会话内临时文件自动清除。

 说明

停止会话（销毁沙箱）和删除会话状态（清除持久化存储）是两个独立操作。允许您在沙箱销毁后仍保留会话数据，或按需单独清理存储状态，避免操作失败导致资源泄漏。

可通过控制台配置会话生命周期，详细操作请参见[通过控制台部署](#)中“生命周期配置”。

表 4-31 生命周期参数

参数（控制台）	参数（API/JSON）	描述	取值范围	默认值
空闲会话超时	lifecycleConfig.idleSessionTimeoutSec	会话在设置的空闲时间内没有交互或活动，系统自动结束该会话。	60 - 604800 秒	900 秒（15 分钟）
最大存活时间	lifecycleConfig.maxAliveTimeSec	会话持续调用时自动终止前的最大存活时间。达到此时间后无论是否活跃都会被强制终止。	60 - 604800 秒	86400 秒（24 小时）

 说明

当空闲会话超时或最大存活时间满足任一条件时，运行时会话将自动终止。支持以秒、分钟、小时或天为单位进行配置。

会话路由机制

平台通过 HTTP 请求头中的会话 ID 标识和路由会话。默认请求头为 X-Hw-Agentarts-Session-Id，会话 ID 从请求头（header）中提取。

不同运行时类型的会话 ID 请求头如下：

表 4-32 各运行时类型的会话 ID 请求头

运行时类型	会话 ID 请求头
Agent（默认）	X-Hw-Agentarts-Session-Id
Code Interpreter	X-Hw-Agentarts-Code-Interpreter-Session-Id
Browser	X-Hw-Agentarts-Browser-Session-Id
Gateway（MCP）	X-Hw-Agentarts-Session-Id

跨访问方式的会话路由

在智能体运行时中，是否配置会话存储会影响跨访问方式（Endpoint）的会话路由行为：

- **未配置会话存储**：新请求根据指定的访问方式路由，同一会话ID可在不同版本中独立使用。
- **已配置会话存储**：新请求根据当前是否存在已激活的同一会话ID沙箱进行判断。不存在已激活沙箱时，按指定访问方式路由；存在已激活沙箱且版本一致时，调用同一沙箱；版本不一致时，调用超时。

说明

请谨慎使用同一会话 ID 进行跨访问方式访问。配置会话存储后，版本不一致会导致调用超时。建议在灰度发布验证时使用新的会话 ID。

会话示例

场景一：基础多轮对话

通过会话 ID 实现多轮对话，无需创建独立会话。在请求头中携带X-Hw-Agentarts-Session-Id，同一会话 ID 的多次调用共享沙箱上下文。

```
# 第一轮（会话 ID = session-001）
POST /runtimes/{runtime_name}/invocations
Headers: X-Hw-Agentarts-Session-Id: session-001
Body: {"message": "我叫张三"}
→ {"reply": "你好，张三！有什么可以帮助你？"}

# 第二轮（相同会话 ID）
POST /runtimes/{runtime_name}/invocations
Headers: X-Hw-Agentarts-Session-Id: session-001
Body: {"message": "你还记得我叫什么吗？"}
→ {"reply": "你叫张三。"}

# 第三轮（新会话 ID → 上下文清空）
POST /runtimes/{runtime_name}/invocations
Headers: X-Hw-Agentarts-Session-Id: session-002
Body: {"message": "你还记得我叫什么吗？"}
→ {"reply": "抱歉，我不清楚你的名字。"}

```

说明

X-Hw-Agentarts-Session-Id提供的是会话标识，用于平台将多次请求路由到同一沙箱。但上下文是否能保持（即Agent能否记住之前的对话），取决于Agent代码是否实现了对话记忆逻辑。

场景二：涉及文件处理

当需要让智能体处理本地文件时，需先创建会话获取沙箱环境：

1. **创建会话**：调用 POST /runtimes/{runtime_name}/sessions-start，获取沙箱环境。
2. **上传文件**：调用 POST /runtimes/{runtime_name}/upload-files，将本地文件上传至沙箱。单文件限制 100MB，多文件总大小限制 500MB。
3. **调用智能体**：调用 POST /runtimes/{runtime_name}/invocations，Agent 读取并处理文件。
4. **下载文件**：调用 GET /runtimes/{runtime_name}/download-files，取回处理结果。
5. **停止会话**：调用 POST /runtimes/{runtime_name}/sessions-stop，释放资源。

场景三：涉及命令执行

当需要在智能体沙箱中执行Shell命令时：

1. **创建会话**：获取沙箱环境。
2. **调用智能体**：Agent推理决策，决定需要执行的命令。
3. **执行命令**：调用POST /runtimes/{runtime_name}/commands，在沙箱内执行Shell 命令，返回标准输出和错误输出。
4. **再次调用智能体**：Agent 基于命令结果继续推理。
5. **停止会话**：释放资源。

 说明

步骤②③④可多次循环，形成“推理 → 执行 → 再推理”的闭环。

4.10 存储配置

存储配置用于将外部存储资源挂载到运行时沙箱容器中，实现数据持久化、文件共享和会话状态保存。AgentArts运行时支持SFS Turbo、会话存储和OBS三种存储类型，满足不同场景的数据持久化需求。

原理介绍

存储挂载机制

存储配置在运行时创建时设定，通过Miracle沙箱平台将外部存储资源挂载到沙箱容器内。挂载过程无需自定义挂载代码或特权容器，平台自动完成存储编排和挂载。

版本更新时的配置继承

运行时版本更新时，存储配置支持增量修改（字段继承机制）：

- 未指定某个存储子配置（如sfsTurbo为null）时，继承上一版本的完整配置。
- 指定了某个存储子配置时，使用新配置替换旧配置。
- 会话存储支持字段级继承：未指定的字段（如uid）继承上一版本该字段的值。

存储类型对比

表 4-33 存储类型对比

存储类型	文件特征	读写特征	性能特点	典型场景	最大数量	网络要求
SFS Turbo	适量中大文件	读写混合、多实例共享	中等 IOPS、适中时延	模型文件、共享数据集、配置文件	5 个	私网（VPC）
会话存储	会话级小文件	按会话隔离读写	低时延、按会话隔离	多轮对话上下文、会话临时数据	1 个	无限制

存储类型	文件特征	读写特征	性能特点	典型场景	最大数量	网络要求
OBS	海量大文件	读多写少	高吞吐、低成本	用户上传文件、生成结果文件	10 个（5 个桶）	私网（VPC）

选择建议：

- **模型文件、共享数据：** 优先选择SFS Turbo，支持多实例共享读写。
- **会话上下文持久化：** 选择会话存储，每个会话独立子目录，自动隔离。
- **用户上传/生成文件：** 选择OBS，海量存储、低成本。

使用约束与限制

- SFS Turbo和OBS存储仅在私网访问（VPC网络模式）下支持配置。
- SFS Turbo最多添加5个，OBS最多添加10个（最多5个不同桶），会话存储仅支持1个。
- 所有挂载路径不能包含"..", 防止路径穿越。
- 会话存储配置后不支持取消，仅支持修改挂载路径。
- 会话存储底层基于OBS对象存储（FUSE挂载），配置umask后文件权限由挂载参数决定，运行时chmod/chown命令不生效。
- SFS Turbo文件系统需为NFS协议，且与运行时在同一VPC内。
- 使用OBS桶挂载功能时，需为运行时委托配置相应的OBS权限。根据文件系统的访问模式，所需的最小权限如下：
 - 文件系统只读操作：obs:bucket:headBucket、obs:bucket:listBucket、obs:object:getObject。
 - 文件系统读写操作：obs:bucket:headBucket、obs:bucket:listBucket、obs:object:getObject、obs:object:putObject、obs:object:deleteObject。以上为最小权限要求。如需配置其他权限，请[提交工单](#)咨询OBS权限相关事宜。
- 使用OBS存储挂载时相关约束及注意事项如下：
 - 托管智能体时如使用OBS存储配置，需创建访问OBS服务的VPC终端节点。
 - i. 进入[VPCEP控制台终端节点列表页](#)。
 - ii. 单击右上角“购买终端节点”，进入购买页。
 - iii. 在“购买终端节点”页面，根据提示配置参数，其他参数保持默认，更多参数详细信息请参见[购买终端节点](#)。

参数	示例	说明
区域	西南-贵阳一	终端节点所在区域。要求与运行时出网VPC、OBS桶所在区域保持一致。
计费方式	按需计费	按需计费是后付费模式，按终端节点的实际使用时长计费，可以随时开通/删除终端节点。

参数	示例	说明
服务类别	按名称查找服务	选择“按名称查找服务”。
服务名称	-	在终端节点服务列表的“名称”列，输入待访问终端节点服务的名称，单击“验证”。 提交工单 获取服务名称，提交工单时同步提供OBS桶名给技术支持人员。将从工单中获得的服务名称填写到“服务名称”中，单击“验证”。
虚拟私有云	-	选择运行时出网VPC。

- iv. 单击右下角的“立即购买”。
- v. 确认终端节点配置，单击“提交”。
- 挂载目录中的文件或文件夹不支持硬链接命令。
 - 挂载目录不会准确显示OBS桶的总容量，用户可以通过OBS控制台查看桶容量情况。
 - 对象一旦写入即不可变，不支持原地修改。如需变更内容，必须下载完整对象、本地修改后全量重新上传并覆盖原对象。如果文件较大，建议采用分段上传，仅重新上传受影响的分片。
 - 一个OBS桶支持挂载到多个Agent或多台云服务器，各挂载进程间互不感知。数据一致性需由用户自行维护，也不能通过OBS锁定文件。建议可通过分布式缓存服务（DCS）的Redis版本实现分布式锁，可避免出现多人对一个文件同时写入的情况。了解分布式缓存服务相关信息，请参见[什么是分布式缓存服务](#)。
 - 对于对象存储的多版本功能，AgentArts无法感知到，也没有办法使用多版本的接口。对于开启了多版本控制的桶，AgentArts只能识别当前版本对象和当前版本的删除标记，并且无法对当前版本的对象真正做出修改（对象存储桶里的对象本身是不可修改的，只能覆盖当前版本对象，产生一个新的当前版本对象，原先的对象会变成历史版本对象）。由于上层应用的一些探测性IO（如vim会产生.swp文件），使得多版本桶里将产生大量中间过程对象，且对象无法被AgentArts删除（删除操作会追加删除标记，而不是真正删除）。了解多版本控制与删除对象相关信息，请参见[多版本控制](#)和[删除对象](#)。
 - 对象桶模式采用回写模式（write-back），读和写都会经过客户端本地的文件系统进行中转，通过tmpdir或use_cache参数控制临时文件目录所在的位置。在应用与AgentArts的交互过程中，应用不直接与AgentArts交互，而是先与FUSE交互。应用与FUSE的交互遵循POSIX语义，其中write()只会写到本地文件系统的临时文件，需要在应用调用文件的close()操作时，修改才会真正的上传到OBS。这意味着：
 - 数据在close()返回成功之前不在OBS上。进程崩溃、断电、kill -9等异常退出会导致未关闭文件的数据永久丢失。
 - close() 可能阻塞较长时间（取决于文件大小和网络带宽）。例如100GB文件的close可能需要数分钟。大对象上传场景，可以通过max_dirty_data参数，在写满部分分段大小时就提前触发上传，以提升整体上传效率。

- 启用use_cache时，已写入但未上传的数据保留在磁盘缓存文件中，下次打开同一文件时有机会恢复。未启用use_cache时，关闭后本地数据立即释放。
- 基于以上模式，建议在关键数据写入后检查close()返回值，避免close()方法返回错误被吞噬；对持久性要求高的场景使用fsync()定期刷新（需权衡性能）；启用use_cache提供额外的数据恢复机会。

通过控制台配置存储

- 步骤1** 登录AgentArts智能体平台。
- 步骤2** 在左侧导航栏选择“托管与运行 > 智能体运行时”。
- 步骤3** 单击“托管智能体”，在“高级配置 > 存储配置”中选择存储类型，详情参数可参考表4-34，关于托管智能体的更多详细操作请参见通过控制台部署智能体运行时。

表 4-34 添加存储参数说明

类型	参数	说明
会话存储	挂载路径	输入会话存储将被挂载的具体目录路径。配置后，每个会话在持久化存储中拥有独立的子目录，这些子目录将作为状态存储挂载到该路径。 挂载路径为/home、/mnt、/data、/tmp及其子目录，不能包含“..”，最大长度为4000字符。 存在多个挂载时（包括OBS、SFS Turbo、会话存储之间），禁止使用相同或存在包含关系的路径。
SFS Turbo	文件系统	选择文件系统。如未创建，单击“创建文件系统”，详细操作说明请参考创建文件系统。 配置后单击  刷新。 说明 • SFS Turbo和智能体运行时中的出网网络配置必须在同一VPC内，SFS Turbo文件系统使用的安全组需放通NFS协议使用的端口号，具体信息请参考VPC的安全组是否影响高性能弹性文件服务的使用，安全组需要放通运行时出站的子网网段访问SFS Turbo。 • 选择的文件系统需为NFS协议。 • AgentArts部署智能体运行时挂载NFS文件系统的soft/hard的模式为hard模式，关于soft/hard的模式详情信息请查看挂载SFS Turbo文件系统到Linux云服务器。 • 为了您的信息安全，请勿在文件系统中存放敏感数据。
	存储路径	输入文件系统中数据的存储路径。 存储路径以“/”开头，不能包含“..”，最大长度4000字符，默认为/。

类型	参数	说明
	挂载路径	输入文件系统中数据的挂载路径，使得该目录下的文件和目录可以被访问。 - 挂载路径为/home、/mnt、/data、/tmp及其子目录，不能包含相对路径，最大长度为4000字符。 - 存在多个挂载时（包括OBS、SFS Turbo、会话存储之间），禁止使用相同或存在包含关系的路径。 - 同一智能体运行时的多个版本挂载在同一目录时会自动共享。 说明 挂载路径的Linux文件系统权限，继承自SFS Turbo存储路径的原有权限（SFS Turbo根目录权限为任何用户可读写）。此权限不支持通过chmod/chown等命令修改。
	读写权限	选择对挂载文件系统的读写权限。 - 读写：可以读取、写入和修改文件系统中的数据，默认为读写。 说明 Linux文件系统权限和读写权限两者必须同时允许，读写才能成功，如果任一项禁止，读写操作将失败。 - 只读：只能读取文件系统中的数据，不能写入或修改数据。
OBS	OBS桶	选择已创建的OBS桶。如未创建，单击“创建OBS桶”，详细操作说明请参考创建桶。 配置后单击 刷新。 说明 - 挂载路径的OBS文件系统权限，继承自OBS存储路径的原有权限（OBS根目录默认权限为任何用户可读写）。此权限不支持通过chmod/chown等命令修改。 - OBS仅支持桶类型，暂不支持并行文件系统。 - 为了您的信息安全，请勿在文件系统中存放敏感数据。
	存储路径(可选)	输入OBS桶内的子目录路径。 挂载OBS桶内的指定子路径，当配置为空时挂载整个桶根目录；非空时挂载桶内指定子目录，必须以 '/' 开头，支持大小写字母、数字及！_-.*'()/ 等字符且不能包含 '..'，路径长度为最大1024字符。
	挂载路径	输入OBS桶挂载到容器内的路径。 - 挂载路径为/home、/mnt、/data、/tmp及其子目录，不能包含相对路径，最大长度为4000字符。 - 存在多个挂载时（包括OBS、SFS Turbo、会话存储之间），禁止使用相同或存在包含关系的路径。 说明 挂载路径的Linux文件系统权限继承自OBS桶存储路径的原有权限。此权限不支持通过chmod/chown等命令修改。

类型	参数	说明
	读写权限	选择对挂载OBS桶的读写权限。 - 读写：可以读取、写入和修改OBS桶中的数据，默认为读写。 说明 Linux文件系统权限和读写权限两者必须同时允许，读写才能成功，如果任一项禁止，读写操作将失败。 - 只读：只能读取OBS桶中的数据，不能写入或修改数据。

----结束

通过 API 配置存储

通过创建或更新运行时API，在请求体的storageConfig字段中配置存储：

```
{
  "storageConfig": {
    "sfsTurbo": [
      {
        "sfsTurbold": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
        "sfsPath": "/",
        "mountPath": "/mnt/shared",
        "readOnly": false,
        "userId": 1000,
        "groupId": 1000
      }
    ],
    "sessionStorage": {
      "mountPath": "/home/session"
    },
    "obsBucket": [
      {
        "bucketName": "my-bucket",
        "bucketPath": "/data",
        "mountPath": "/data/obs",
        "readOnly": false
      }
    ]
  }
}
```

4.11 观测运行时

智能体运行时部署后，通过观测功能监控Agent的运行状态。当前运行时的观测能力以日志监控为主，通过开启日志记录，将运行时托管过程中产生的日志上报至云日志服务（LTS），便于排查调用异常和定位问题。

查看智能体运行时日志

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 智能体运行时 ”。
- 步骤3 单击智能体运行时名称，查看智能体运行时详情。

步骤4 在“日志”页签，查看智能体运行时运行过程产生的日志。

如未启用日志功能，可在“日志”页签单击“去启用”开启日志记录。

----结束

离线评估

除日志监控外，运行时还支持离线评估，对智能体的准确性、安全性、流畅性开展综合能力评测。通过离线评估，您可以在发布前验证 Agent 的能力是否达到预期。

说明

离线评估的详细操作请参见[智能体运行时离线评估任务](#)。

4.12 管理运行时

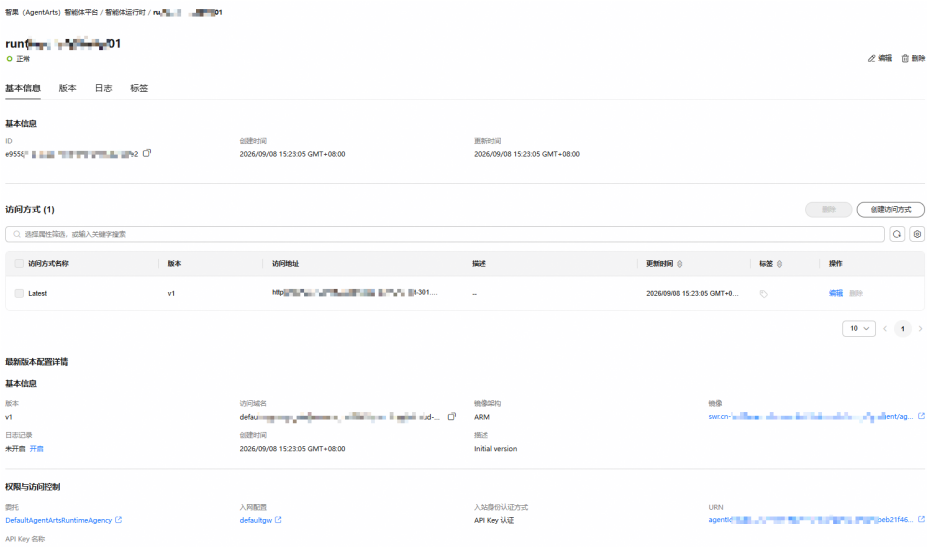
4.12.1 查看并管理运行时

智能体运行时创建完成后，您可以查看并管理运行时的基本信息、版本、标签等。

查看运行时详情

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 智能体运行时”。
- 步骤3 在智能体运行时列表中，单击运行时名称，进入详情页，查看运行时详细信息。

图 4-20 查看运行时详细



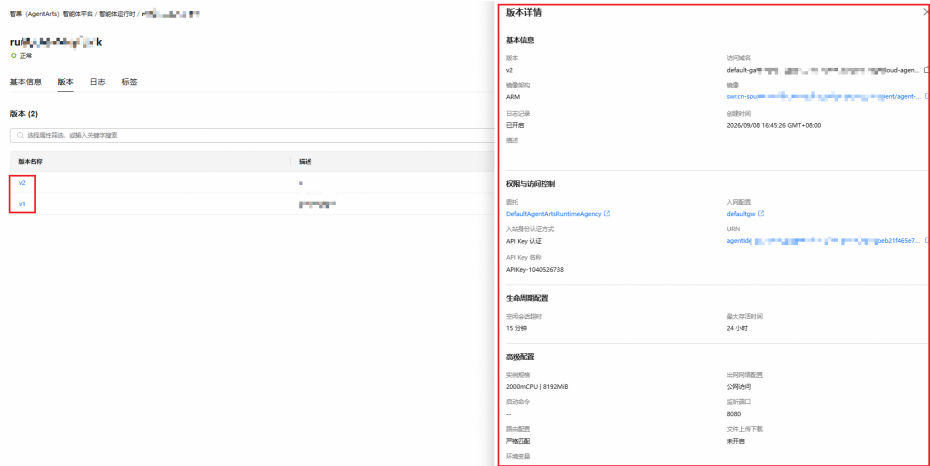
----结束

查看智能体运行时版本

当修改智能体运行时的基本信息、权限与访问控制，或者智能体代码更新并打包成新镜像重新上传后，运行时版本均可更新。

- 步骤1 进入智能体运行时界面。
- 步骤2 单击智能体运行时名称，在“版本”页签，查看运行时的版本列表，单击版本号，查看版本详情。

图 4-21 版本详情



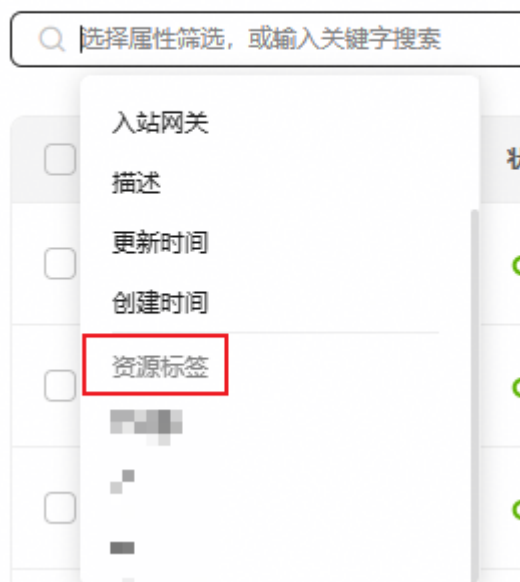
----结束

管理标签

- 步骤1 进入智能体运行时界面。
- 步骤2 在智能体运行时列表中可通过标签过滤查询智能体运行时。

图 4-22 通过标签过滤查询智能体运行时

智能体运行时 (4) [?]



步骤3 单击智能体运行时名称，查看智能体运行时详情。

步骤4 在“标签”页签，查看已添加的标签，单击“编辑标签”，可对标签进行管理（最多支持添加20个标签）。

- 添加标签：单击“添加标签”，可选择TMS中的预定义标签，或自定义输入标签键与值。
- 修改标签：直接修改已有标签内容即可。
- 删除标签：单击目标标签后的“删除”即可。

----结束

4.12.2 管理访问方式

智能体运行时的访问方式用于与已经部署好的智能体运行时进行通信和交互，可以通过创建的访问方式访问指定的智能体运行时版本，如未创建访问方式，则默认使用智能体运行时的Latest访问方式。

创建访问方式

步骤1 登录[AgentArts智能体平台](#)。

步骤2 在左侧导航栏选择“托管与运行 > 智能体运行时”。

步骤3 在列表单击已创建好的智能体运行时名称，进入智能体运行时详情页。

步骤4 在“访问方式”区域，可查看系统已创建的Latest默认访问方式。

Latest默认访问无法编辑与修改，且“Latest访问方式”默认关联运行时的最新版本。

步骤5 单击“创建访问方式”，参数配置请参考[表4-35](#)，配置完成后单击“确定”。

表 4-35 参数配置

参数	说明
名称	访问方式名称。 命名规则：以字母开头，以字母或数字结尾，可以包含字母、数字和中划线，长度为2-48个字符。 系统会默认生成访问方式的名称，可根据需求自定义修改。
灰度策略	通过配置主要版本及次要版本的流量比例，实现新版本的小范围验证。 灰度策略开关： ● 已开启：表示灰度策略已经启用。启用后，可以根据配置的版本权重比例分配流量。至少2个版本才可启用灰度。 ● 已关闭：表示灰度策略未启用。未启用时，所有流量将直接流向主要版本。 版本流量分配： ● 主要版本：选择当前主要运行的版本并配置流量。默认情况下，主要版本及次要版本流量占比各为50，可以通过单击“+”和“-”按钮来调整流量分配，主要版本和次要版本的流量占比之和为100。 ● 次要版本：选择次要运行的版本并配置流量。次要版本通常用于测试新功能或新版本的稳定性，可以通过单击“+”和“-”按钮来调整流量分配。如果次要版本流量为0时，所有流量下发至主要版本。 流量分配预览：展示主要版本和次要版本的流量分配。 配置灰度策略并完成设置后，参考灰度发布，验证灰度环境。
版本	当未开启灰度策略时，选择需要交互的智能体运行时版本。 如智能体运行时版本有更新，可根据需求修改访问方式中的版本。
描述	输入对当前访问方式的描述，方便后续管理和维护。描述内容长度为0到255个字符，不能包含特殊字符如<、>。
标签 (可选)	由“标签键”和“标签值”组成，用于标识和分类云资源。 您可以在TMS中创建预定义标签，用于使用同一标签标识多种云资源，创建后可在标签输入框下拉选择同一标签，具体操作请参考创建预定义标签，创建后单击刷新。 单击“添加标签”，选择在TMS中创建的预定义标签，或自定义输入标签键和标签值。 可添加一个或多个标签，最多添加20个标签。

步骤6 使用已创建的访问方式。

详细请参考[调用智能体运行时](#)中的endpoint参数。

----结束

更多操作

访问方式创建完成后，您还可以执行如下操作。

表 4-36 更多操作

操作	说明
编辑访问方式	1. 在访问方式列表，单击操作列的“编辑”。 2. 修改访问方式的版本以及描述信息。
删除访问方式	Latest默认方式无法删除。 1. 在访问方式列表，单击操作列的“删除”。 2. 在弹框中确认要删除后输入“DELETE”，然后单击“确定”。

4.12.3 灰度发布

灰度发布是一种渐进式版本发布策略，通过将流量按比例分配到不同版本，实现新版本的小范围验证，降低全量发布风险。AgentArts运行时通过访问方式的灰度策略实现版本级别的流量分配。

灰度发布的核心是流量按权重分配。启用灰度策略后，平台将请求按配置的流量比例分发到两个运行时版本：

- **主要版本**：当前主要运行的版本，承载大部分流量。
- **次要版本**：用于测试新功能或新版本稳定性的版本，承载小部分流量。

流量分配通过Miracle沙箱平台的PolicyItem路由规则实现。每个版本对应一个沙箱模板，平台根据权重比例将请求路由到对应版本的沙箱实例。未配置灰度策略时，所有流量路由到单一目标版本（权重为 1）。

 **说明**

如果运行时配置了会话存储，请谨慎使用同一会话ID进行跨访问方式调用。当灰度路由的版本与会话已激活的沙箱版本不一致时，会导致调用超时。建议在灰度发布验证时使用新的会话ID。

约束与限制

- 灰度策略需至少2个版本才可启用。
- 每个访问方式仅支持配置一组灰度策略（1个主要版本 + 1个次要版本）。
- 主要版本和次要版本不能选择同一版本。
- 流量权重为0-100的整数，且两版本权重之和必须等于100。

- Latest默认访问方式无法配置灰度策略，需创建新的访问方式。

操作步骤

步骤1 准备新版本。

将新版本代码打包为镜像，部署为运行时的新版本（如 v2）。具体操作请参见[通过控制台部署](#)中的“编辑智能体运行时”。

步骤2 创建灰度访问方式。

1. 登录AgentArts智能体平台。
2. 在左侧导航栏选择“托管与运行 > 智能体运行时”。
3. 单击运行时名称，进入详情页。
4. 在“访问方式”区域，单击“创建访问方式”。
5. 开启“灰度策略”开关。
6. 选择主要版本和次要版本，配置流量比例（如主要版本 90%，次要版本 10%）。
7. 配置完成后单击“确定”。

步骤3 验证新版本。

通过灰度访问方式调用运行时，小比例流量会路由到新版本。观察新版本的运行状态和日志，确认功能正常。

```
# 调用灰度访问方式（流量按比例分配）
POST https://{域名}/runtimes/{runtime_name}/invocations?endpoint=gray-release
Headers: X-Hw-Agentarts-Session-Id: new-session-id
Body: {"message": "测试新版本功能"}
```

说明

建议使用新的会话ID进行灰度验证，避免与会话存储路由冲突。

步骤4 逐步扩大流量。

如新版本运行正常，逐步调大次要版本的流量比例：

- 第一阶段：主要版本 90% → 次要版本 10%
- 第二阶段：主要版本 70% → 次要版本 30%
- 第三阶段：主要版本 50% → 次要版本 50%
- 第四阶段：主要版本 20% → 次要版本 80%

每个阶段持续观察运行状态、日志和监控指标。

步骤5 全量切换。

确认新版本稳定后，将主要版本切换为新版本，参考[步骤2](#)将主要版本流量设为 100%，完成灰度发布。

----结束

4.12.4 更新运行时镜像

智能体的代码修改后，可以重新打包成新的镜像，将新的镜像上传，确保智能体的更新过程有序、可控。

前提条件

已创建智能体运行时，具体操作请参见[通过控制台部署](#)。

更新运行时镜像

- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 在左侧导航栏选择“托管与运行 > 智能体运行时”。
- 步骤3** 在智能体运行时列表中，单击智能体运行时操作列的“编辑”。
- 步骤4** 在“来源方式”区域，选择目标镜像。
- 步骤5** 单击“确定”。

----结束

5 记忆库

5.1 记忆库概述

什么是记忆库

记忆库是智能体用于存储、管理和检索历史信息的核心组件，可以通过控制台创建并配置不同类型的记忆能力，对交互过程中产生的信息进行提取、整理和保存，使智能体能够在后续交互中按需获取相关历史上下文。

通过短期记忆和长期记忆协同，记忆库帮助智能体实现：

- 单次或近期对话中的上下文保持：通过短期记忆保存近期交互信息，支持连续对话。
- 跨会话的信息复用：通过长期记忆保存用户偏好、历史交互和具有持续价值的信息，支持个性化响应。

核心能力

- 分层记忆：支持短期记忆和长期记忆。短期记忆用于保存近期交互产生的上下文信息，长期记忆用于沉淀具有持续价值的信息，满足近期上下文保持和跨会话信息复用等不同记忆需求。
- 多种记忆策略：支持总结（Summary）、语义记忆（Semantic Memory）、用户偏好（User Preference）、情景记忆（Episodic Memory）和程序性记忆（Procedural Memory）等多种记忆策略，分别用于保留会话摘要、事实与上下文知识、用户偏好、具体经历以及可复用的操作流程等不同类型的信息。不同记忆策略的定义、适用场景和使用方式，参见“[记忆策略](#)”。
- 多维度记忆组织与隔离：支持按照空间、用户、会话等维度对记忆进行组织和隔离，避免不同业务空间、用户或会话之间发生非预期的数据共享，并满足不同粒度下的记忆管理和使用需求。

记忆库由平台统一提供记忆处理和存储能力，开发者无需自行搭建完整的记忆提取、存储和检索链路，可直接在智能体应用中集成使用。

记忆库与知识库的区别

记忆库和知识库都可以为智能体提供模型当前上下文之外的信息，两者的区别如下：

对比项	记忆库	知识库
存储内容	交互产生的动态信息（对话、偏好、经验）	预先准备的静态知识（文档、FAQ、手册）
数据来源	Agent与用户/环境持续交互过程中产生	人工导入、系统同步或外部数据源接入
更新方式	持续增量更新，可新增、修改、删除	通常按文档或数据源进行新增、更新、删除和重建索引
检索时机	可在对话过程中根据业务需求检索和使用相关记忆	用户提问时检索相关段落
典型场景	用户偏好、历史决策、经验复用，如“记住用户上次说了什么”	RAG、知识问答、文档检索，如“查找产品手册内容”
举例	用户偏好咖啡加冰	产品使用说明书

简单说：知识库解决“Agent知道什么”，记忆库解决“Agent记得什么”。

应用场景

记忆库适用于需要保持上下文连续性、复用历史信息或提供个性化体验的智能体场景，例如：

- 智能客服：保留用户历史咨询、问题背景和服务上下文，减少用户重复描述，提高连续服务能力。
- 个人助手：记录用户偏好、习惯和长期关注信息，在后续交互中提供更符合用户需求的响应。
- 企业助手：保留用户在不同会话中的业务背景和历史交互信息，支持连续的业务咨询和任务处理。
- 长任务智能体：记录任务执行过程中的阶段性信息，使智能体能够在后续交互中延续已有任务上下文。

通过记忆库，智能体可以在有限上下文条件下更有效地利用历史信息，提升上下文连续性、信息复用能力和个性化交互体验。

规格与限制

- 短期记忆事件约束与限制：短期记忆按会话隔离，事件在配置的过期时间（7~365 天）后自动清除。
- 长期记忆记录约束与限制：长期记忆持久保存，可通过检索和删除管理。
- 创建记忆库时，公网访问和私网访问至少要选择一种，开启后暂不支持关闭。
- 创建记忆库时，内置策略和自定义策略至少要选择一种；自定义策略最多可创建10个。
- 使用云服务默认密钥（agentarts/default）加密或自定义密钥加密，记忆库创建后不可修改。
- 创建记忆库时默认最多可创建10个记忆库，如需增加更多配额，请[提交工单](#)申请。

- API调用频率限制：超过频率限制的请求会被流控，可在观测指标中查看流控情况。
- 单条消息大小限制：建议单条消息内容不超过平台规定的最大长度。

5.2 工作原理

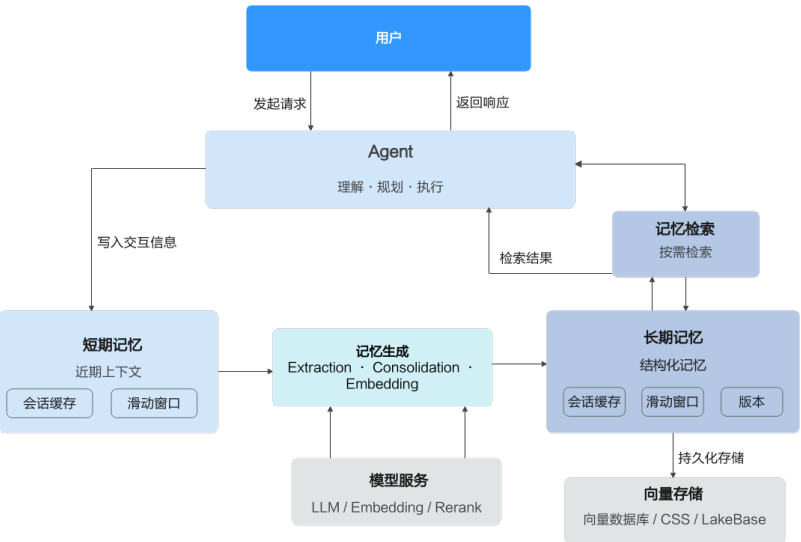
5.2.1 分层记忆架构

记忆库采用短期记忆和长期记忆相结合的分层架构，对智能体交互过程中产生的信息进行分层管理。

短期记忆用于保存近期交互上下文，帮助Agent理解当前会话状态；长期记忆用于沉淀历史交互中形成的记忆信息，支持后续任务中的信息检索和复用。

通过短期记忆与长期记忆协同工作，记忆库能够同时满足实时上下文处理和跨会话信息复用需求。

图 5-1 分层记忆架构



记忆库整体工作流程如下：

1. 用户与Agent交互产生的信息首先进入短期记忆。
2. 当满足长期记忆生成条件时，系统基于短期记忆中的信息生成长期记忆。
3. 后续交互过程中，Agent可根据业务需要检索相关长期记忆，并结合当前上下文完成任务处理。

5.2.2 短期记忆

短期记忆用于存储单次会话中的原始交互消息，包括用户输入、Agent 回复以及 system、tool 等角色产生的消息。

短期记忆以Session为基本管理单元，保存当前会话中的历史交互内容，为后续对话提供上下文信息。

作用

短期记忆主要用于帮助 Agent 保持当前会话内的上下文连续性，使 Agent 能够基于已有消息理解当前任务状态、历史交互内容以及用户后续请求。

特点

- 保存原始消息：短期记忆直接保存交互过程中产生的消息，不对消息内容进行长期记忆抽取或结构化转换。
- 支持多种消息角色：支持user、assistant、system、tool 等不同角色消息。
- 会话级隔离：消息按 Session 进行管理和查询，不支持跨 Session 查询短期记忆。
- 支持分页和排序查询：可按消息顺序或消息时间进行排序，并支持分页获取会话中的消息记录。

生命周期

短期记忆支持配置消息保留期，默认90天，可配置范围为7～365天。超过保留期的消息会由系统自动清理。在过期前，消息可被策略处理并提取为长期记忆。

短期记忆支持的操作请参见[短期记忆操作](#)。

5.2.3 长期记忆

长期记忆用于保存由记忆策略从历史交互消息中提取并生成的结构化记忆记录。

与短期记忆直接保存原始消息不同，长期记忆会根据配置的记忆策略，对历史交互内容进行抽取和整理，形成可持续保存和检索的记忆记录。

作用

长期记忆主要用于支持历史信息复用和跨会话记忆，使应用能够在后续任务处理中检索与当前请求相关的历史信息。

通过长期记忆，可以减少用户重复提供历史背景信息，并为个性化交互、历史经验复用和长期任务处理提供支持。

特点

策略驱动生成：长期记忆根据配置的记忆策略生成，不同策略用于提取不同类型的信息：

- 总结（Summary）：会话主题、任务过程和关键决策等摘要信息。
- 语义记忆（Semantic Memory）：事实信息、概念和上下文知识。
- 用户偏好（User Preference）：用户行为模式、偏好和选择等信息。
- 情景记忆（Episodic Memory）：具有时间、场景和经历特征的具体事件信息。
- 程序性记忆（Procedural Memory）：可复用的任务流程、操作步骤和执行经验。

结构化记忆记录：长期记忆由记忆内容、向量表示和相关元数据共同组成，可用于后续检索。

支持语义检索和相关性排序：可根据查询内容召回相关记忆，并结合相关性阈值和排序配置返回结果。

支持多维过滤：可根据策略、用户、会话、时间等条件筛选长期记忆。

生命周期

长期记忆默认持久化保存，不设置自动过期时间。

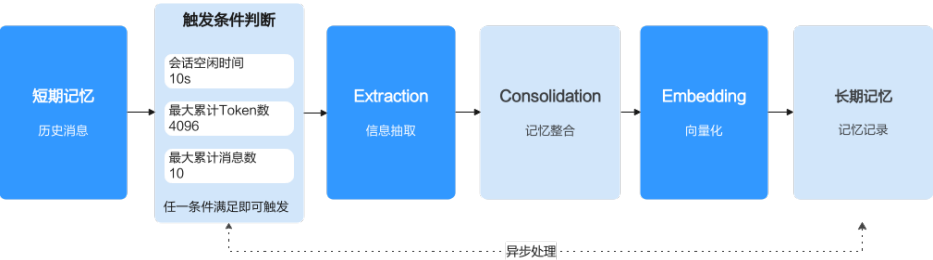
长期记忆会持续保留，直至用户显式删除对应记忆、所属空间被删除并触发关联数据清理，或超出租户配额限制。

长期记忆支持的操作请参见[5.6.6-长期记忆操作](#)。

5.2.4 长期记忆生成机制

长期记忆由系统基于短期记忆中的历史消息异步生成。当满足触发条件时，系统启动长期记忆提取任务，并根据当前启用的记忆策略对相关消息进行处理，最终生成长期记忆记录。

图 5-2 长期记忆生成链路



触发条件

当以下任一条件满足时，系统触发长期记忆提取任务：

触发参数	配置范围	说明
会话空闲时间	10~86,400 秒，默认 10 秒	会话在指定时间内未产生新消息时触发
最大累计 Token 数	1,000~1,073,741,824，默认 4,096	累计消息 Token 数达到阈值时触发
最大累计消息数	3~10,000，默认 10	累计消息数量达到阈值时触发

多个触发条件之间为OR关系，任一条件满足即可触发长期记忆提取。

长期记忆提取采用异步处理机制。触发后，系统需要完成后续信息抽取、记忆整合和向量化等处理，因此提取结果可能不会立即可用。

生成流程

长期记忆生成主要包括以下步骤：

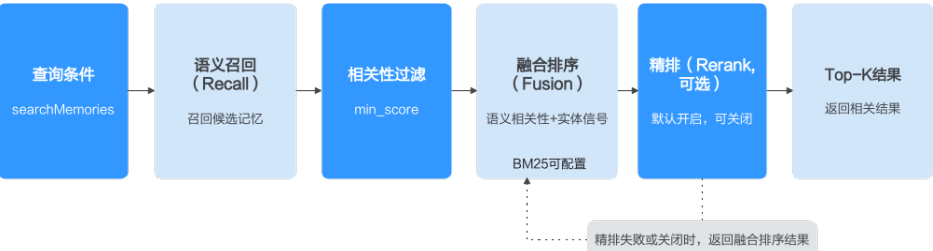
1. 信息抽取（Extraction）：系统根据启用的记忆策略，对短期记忆中的相关消息进行分析，提取符合策略要求的候选记忆内容。
2. 记忆整合（Consolidation）：系统对候选记忆与已有长期记忆进行处理，包括去重、合并和更新等操作。不同记忆策略采用的整合方式可能不同。
3. 向量化（Embedding）：系统对最终生成或更新的记忆内容进行向量化，生成用于后续语义检索的向量表示。
4. 持久化存储：系统将记忆内容、向量表示以及策略、用户、会话等相关元数据保存为长期记忆记录。

不同记忆策略的具体提取内容和处理方式，参见 [5.3 记忆策略](#)。

5.2.5 长期记忆检索机制

长期记忆提供独立的检索能力，支持应用按需查询与复用。提交查询内容和过滤条件后，系统从长期记忆中召回相关候选，依次经过相关性过滤、融合排序，并按配置决定是否精排，最终返回按相关度排序的结果。

图 5-3 长期记忆检索链路



检索流程

1. 召回（Recall）：系统首先基于查询文本进行语义召回，从长期记忆中获取语义相关的候选记忆。
2. 相关性过滤系统：根据min_score过滤低相关候选。min_score取值范围为 0~1，默认值为0.75。
3. 融合排序（Fusion）：系统综合语义相关性以及已启用的辅助相关性信号，对候选记忆计算综合得分并排序。当前支持实体关联信号，并可配置启用 BM25 词法匹配信号。
4. 精排（Rerank，可选）：系统支持使用精排模型对候选结果重新打分和排序。精排默认开启，也可以通过全局配置或单次检索请求关闭。精排未开启时，系统回退至融合排序结果。
5. 结果返回：系统按照top_k截取排序后的结果并返回。top_k支持配置1~100条，默认返回10条。

检索特点

- 语义召回：长期记忆候选集合由语义检索产生。
- 多信号排序：可结合实体关联、BM25词法匹配等信号优化候选排序。
- 可选精排：支持按需开启或关闭Rerank，以进一步提升结果排序精度。
- 多维过滤：支持按策略、用户、助手、会话、记忆类型和时间范围等条件限制检索范围。

📖 说明

返回结果中的score表示最终相关性得分。开启精排时返回精排得分；未开启精排时返回融合排序得分。

5.2.6 记忆组织与隔离

隔离维度	说明
空间（space）	每个记忆库是一个独立空间，对应一个space_id
用户（actor）	按actor_id隔离不同终端用户的记忆，用户A的记忆对用户B不可见
会话（session）	短期记忆按Session隔离；长期记忆也可限在单Session内

短期记忆：以Session为隔离单元，不支持跨Session查询。

长期记忆：支持用户级（跨会话复用）与会话级（会话内使用）两种隔离级别，生成与检索均遵循该级别。

访问控制：空间级密钥可执行全部操作；会话级密钥仅可读、追加消息和检索记忆，禁止创建/删除会话与记忆。

5.3 记忆策略

5.3.1 策略概述

策略概述

记忆策略决定了“从短期记忆中提取什么信息、以什么形式存入长期记忆”。不同策略关注不同维度的信息，提取出的长期记忆内容也不同。

- 可以同时启用多种策略。例如同时启用“总结”和“用户偏好”，每次触发提取时两种策略各自独立运行，分别产生不同类型的长期记忆记录。
- 在创建记忆库和编辑长期记忆提取策略时，至少需选择或者保留一个内置策略或自定义策略。
- 创建后可以在详情页中新增或删除策略。

选型指引

策略	什么时候用
总结 Summary	对话较长，需要压缩保留核心信息。例：客服会话摘要、会议纪要
语义记忆 Semantic Memory	对话中出现关键事实和知识。例：用户提到的项目名称、技术栈

策略	什么时候用
用户偏好 User Preference	需要记住用户的喜好和习惯。例：用户喜欢中文回复、偏好简洁
情景记忆 Episodic Memory	从完整经历中学习，不仅记录“发生了什么”，还沉淀教训与反思。例：上次解决某类问题的完整过程、当时的尝试与最终结论
程序性记忆 Procedural Memory	需要固化操作流程、行动步骤和关键提示，形成可复用的任务执行指南。例：代码仓探索流程、运维操作执行纲要

常见组合推荐

不同策略提取的记忆类型不同，实际使用中通常组合多种策略，让 Agent 从多个维度理解用户。以下按场景给出推荐组合及选择理由：

客服场景

项目	说明
推荐组合	总结 + 用户偏好
选择理由	总结策略压缩每次对话的要点，让Agent快速回溯“之前讨论了什么”；用户偏好策略记住用户的习惯和需求（如偏好联系方式、常用产品），让客服回复更个性化。
典型对话	用户先反馈了订单问题，几天后回来追问进度——Agent靠总结记忆知道上次讨论了什么，靠偏好记忆知道用户的沟通习惯。
注意事项	客服对话量大，总结策略能控制记忆体量，避免每轮对话都产生大量碎片化记忆。

知识助手场景

项目	说明
推荐组合	语义记忆 + 总结
选择理由	语义记忆提取对话中出现的事实知识（如“用户的项目用 Python 3.11”），构建持久知识库；总结策略概括每次咨询的主题，帮助Agent快速定位历史讨论。
典型对话	用户咨询技术方案，多次对话中涉及技术栈选型、架构设计——语义记忆记住每个具体事实，总结记忆记住每次讨论的主题。
注意事项	知识型对话中事实信息密度高，语义记忆是最核心的策略。总结策略作为辅助，帮助Agent快速了解上下文。

个人助理场景

项目	说明
推荐组合	用户偏好 + 情景记忆
选择理由	用户偏好策略记住用户的个人喜好（如饮食偏好、日程习惯）；情景记忆策略记住完整的交互经历（如“上次帮你订机票时遇到了什么问题、如何解决的”），让助理越用越懂用户。
典型对话	用户让助理帮忙规划旅行——偏好记忆知道用户喜欢靠窗座位，情景记忆知道上次订票时遇到的问题和解决方案。
注意事项	情景记忆记录较长、Token消耗更高；成本敏感时可降级为总结策略，但会丢失完整过程回溯能力，需权衡。

编码与运维场景

项目	说明
推荐组合	程序性记忆 + 语义记忆
选择理由	程序性记忆固化操作流程和执行步骤，让 Agent 面对相似任务时直接加载已有经验，避免重复推理；语义记忆提取对话中出现的事实知识（如工具名称、系统配置），构建持久知识库。两者配合，Agent既知道“怎么做”又知道“用什么做”。
典型对话	Agent 首次探索代码仓后，程序性记忆固化探索流程，后续遇到相似任务直接复用；语义记忆记住项目使用的技术栈和关键配置。
注意事项	程序性记忆按用户隔离，一个用户的成功流程不会自动共享给其他用户。程序性记忆支持增量更新，新信息可更新已有流程。

全覆盖场景

项目	说明
推荐组合	总结 + 语义记忆 + 用户偏好 + 情景记忆（四种全部启用）
选择理由	四种策略各有侧重——总结概括对话、语义提取事实、偏好跟踪习惯、情景保留经历。全部启用可以最大限度覆盖记忆维度，适合对记忆精度要求极高的场景。
典型对话	长期个人助理、医疗问诊助手等需要同时记住“讨论了什么、知道了什么、用户喜欢什么、经历了什么”的场景。
注意事项	策略越多，长期记忆提取时调用的 LLM 次数越多（每个启用策略都会执行独立抽取），Token 消耗和费用越高。建议先用 2-3 种策略验证效果，确有需要再增加。

5.3.2 内置策略

5.3.2.1 总结策略

原理

总结策略（Summary）汇总和压缩会话中的主题、任务、历史决策记录等关键信息，将互动的大量详细信息或数据压缩成关键上下文和关键洞察信息，帮助智能体更高效地检索和利用长期记忆中的信息。

适用场景

- 长对话上下文压缩：当对话轮次较多时，通过总结策略压缩为摘要，减少上下文冗余，提高Agent响应效率。
- 客服对话摘要：自动生成客服会话摘要，便于后续回顾和质检。
- 会议纪要生成：从会议讨论中提取关键议题、决策和待办事项，生成结构化纪要。
- 年度工作总结：从长期工作对话中提取年度关键成果和工作要点。

特点

- 提取结果粒度较粗，适合记录一段会话整体“讨论了什么、完成了什么”。
- 通过压缩降低记忆内容长度，但部分原始细节和具体措辞可能不会保留。

提取内容举例

举例一：项目进度讨论

- 用户询问了A项目的进度，Agent回复了三个关键节点。
- 用户要求修改方案的第二部分，Agent完成了修改。

提取为：“讨论了A项目进度（三个节点：需求评审、开发联调、测试验收），修改了方案第二部分”。

举例二：客服会话

- 用户反馈订单延迟问题，Agent排查后确认是物流异常，已重新发货。

提取为：“用户反馈订单延迟，排查原因为物流异常，已重新发货并更新物流单号”。

举例三：会议纪要

- 会议讨论了Q1销售目标、资源分配方案和下阶段重点任务。

提取为：“Q1销售目标为500万，资源向华南区域倾斜，下阶段重点推进产品迭代和客户拓展”。

与语义记忆策略的区别

对比项	总结策略	语义记忆策略
关注维度	整体对话的概括	具体事实和知识
记住的是	“讨论了什么”	“知道了什么”

对比项	总结策略	语义记忆策略
提取粒度	宏观摘要	微观事实
举例	“讨论了A项目进度”	“A项目技术栈为Python 3.11”
信息损失	较大（丢弃细节）	较小（保留事实）

 说明

总结策略和语义记忆策略可以同时启用，互为补充：总结策略负责压缩对话概要，语义记忆策略负责提取其中具体的事实知识。

5.3.2.2 语义记忆策略

原理

从会话中识别并提取具有明确含义、可独立理解和后续复用价值的事实性信息，包括事实、概念、实体属性、实体关系及其他相对稳定的知识。

适用场景

- 知识密集型对话：需要从对话中持续积累事实知识，构建知识图谱式的长期记忆。
- 技术咨询：记住用户提到的技术栈、架构决策、约束条件等具体事实。
- 个人信息管理：记住用户的个人信息、专业背景、教育经历等。
- 需要记住对话中出现的具体事实：避免Agent重复询问已获取的信息，提升交互效率。

特点

- 提取的记忆是独立的事实片段，而非对话摘要，检索时可直接获取具体信息。
- 事实以与上下文无关的格式存储，便于跨会话检索和复用。
- 适合需要精确引用和查证的场景，信息损失较小。
- 需要注意事实的准确性和时效性，过时的事实可能导致Agent给出错误回答。

提取内容举例

举例一：项目信息

用户：“我们项目用的是Python 3.11 + FastAPI”
提取为事实：“用户的项目技术栈为Python 3.11 + FastAPI”

举例二：团队信息

用户：“我们团队有5个人”
提取为事实：“用户团队规模为5人”

举例三：个人背景

用户：“我之前在华为云做了三年后端开发”

提取为事实：“用户工作经历：华为云后端开发，3年”

举例四：业务约束

用户：“我们的系统要求响应时间不超过200ms”

提取为事实：“系统性能要求：响应时间≤200ms”

举例五：实体关系

用户：“张三是我们组的组长，李四是组员”

提取为事实：“张三担任组长，李四为组员，二者同组”

与总结策略的区别

对比项	语义记忆策略	总结策略
关注维度	具体事实和知识	整体对话的概括
记住的是	“知道了什么”	“讨论了什么”
提取粒度	微观事实	宏观摘要
举例	“项目技术栈为 Python 3.11”	“讨论了项目进度”
信息损失	较小（保留事实）	较大（丢弃细节）

与情景记忆策略的区别

对比项	语义记忆策略	情景记忆策略
记住的是	事实（What）	经历（How & Why）
提取内容	静态事实知识	带时空序列的完整事件
举例	“Python 3.11”	“上次遇到 API 报错，先查日志再改参数解决”
作用	Agent 知道什么	Agent 学到了什么
时效性	相对稳定	随经历积累

5.3.2.3 用户偏好策略

原理

从会话中识别并提取用户明确表达或能够可靠识别的偏好、习惯和选择倾向，并按照用户维度持续维护用户偏好信息，为后续交互提供个性化上下文。

适用场景

- 个性化推荐：基于用户偏好提供定制化的内容和服务推荐。
- 定制化服务：根据用户习惯自动调整服务方式，如默认语言、输出格式等。
- 需要跨会话记住用户偏好：用户不必每次对话都重复说明自己的喜好。
- 智能助手/个人助理：长期记住用户的日常习惯和个性化需求。

特点

- 偏好是动态更新的——如果用户后续表达了新的偏好或与旧偏好冲突的选择，会自动更新偏好档案，无需手动修改。
- 按用户（actor_id）隔离，不同用户的偏好互不影响，同一用户的不同会话共享偏好档案。
- 偏好记录带有时间信息，便于追溯偏好变更历史。
- 适合需要个性化服务的场景，是“记住用户是谁”的核心策略。

提取内容举例

举例一：语言与风格偏好

用户：“请用中文回复我，简洁一点”

提取为偏好：“用户偏好中文回复，风格简洁”

举例二：饮食偏好

用户：“帮我订咖啡的时候记得加冰，不要糖”

提取为偏好：“咖啡偏好：加冰、无糖”

举例三：工作习惯

用户：“我习惯早上 9 点开始工作，会议尽量安排在下午”

提取为偏好：“工作时间：9 点开始；会议偏好：安排在下午”

举例四：技术偏好

用户：“代码注释用英文，变量名用驼峰命名法”

提取为偏好：“代码风格：注释英文、变量名驼峰命名”

举例五：偏好更新

- 用户第一次说：“我喜欢深色主题”
- 用户后来改为：“现在改用浅色主题了”

偏好档案自动更新为：“界面偏好：浅色主题”（覆盖原深色主题偏好）

使用建议

- 同一用户的不同会话必须使用相同的actor_id，否则偏好无法跨会话共享。
- 建议使用业务系统中的用户ID作为actor_id，确保用户身份的唯一性和稳定性。
- 偏好策略可与总结策略或语义记忆策略组合使用，既记住用户偏好，也保留对话摘要或事实知识。

5.3.2.4 情景记忆策略

原理

从对话或经历中提取具有时空序列的具体事件、个人体验和场景细节。不只是记住“发生了什么”，而是记住“完整的经过和结果”。

适用场景

- Agent 需要从经验中学习：遇到类似问题时，可检索过往经历复用解决方案。
- 复杂问题排查场景：记住排查经过和根因，避免重复走弯路。
- 希望 Agent 越用越聪明：情景记忆随使用积累，Agent 逐步沉淀经验。
- 个人体验记录：记住用户分享的经历和体验，在后续对话中个性化回应。

特点

- 提取的记忆包含完整的时序信息（背景→过程→结果），而非碎片化事实。
- 适合“从经验中学习”的场景，是 Agent 持续进化的核心策略。
- 检索时按情境相似度匹配，而非精确关键词匹配，适合模糊场景召回。
- 记忆内容较长，检索时可能占用更多上下文窗口。

提取内容举例

举例一：API报错排查

让Agent调试一个API报错问题，Agent先检查了日志 → 发现是参数格式错误 → 修改了参数 → 问题解决。

提取为情景记忆：“遇到API报错时，先检查日志，常见原因是参数格式错误，修改参数格式后可解决”

举例二：部署失败处理

在部署服务时遇到容器启动失败，Agent检查了端口占用 → 发现8080端口被旧进程占用 → 关闭旧进程后重新部署成功。

提取为情景记忆：“容器启动失败时，检查端口占用情况，常见原因是旧进程未释放端口，清理后重新部署可解决”

举例三：用户咨询旅游体验

用户分享了上次去云南旅游的经历：先到大理，再转丽江，行程安排比较紧凑。

提取为情景记忆：“用户云南旅游经历：大理→丽江，行程紧凑”

与语义记忆的核心区别

对比项	语义记忆	情景记忆
记住的是	事实（What）	经历（How & Why）
提取内容	静态事实知识	带时空序列的完整事件

对比项	语义记忆	情景记忆
举例	“Python 3.11 ”	“上次遇到这个问题，是这样解决的”
作用	Agent 知道什么	Agent 学到了什么
时效性	相对稳定	随经历积累
检索匹配	按事实关键词	按情境相似度

 说明

语义记忆提取的是“事实”（静态知识），情景记忆提取的是“经历”（动态过程）。一个回答“是什么”，一个回答“怎么做过的”。两者可同时启用，互为补充。

5.3.2.5 程序记忆策略

原理

从会话记录中提取操作流程、行动步骤和关键提示等信息，形成可复用的任务执行指南。程序性记忆定位为Agent的“流程与经验库”，解决 Agent 每次面对相似任务都从零推理、重复探索、浪费token的问题，同时约束LLM推理边界，提供前置条件检查、步骤拆解、检查点机制和回滚预案等结构化执行纲要。

程序性记忆采用自然语言分条目格式组织，包含以下核心字段：

- 标题（Title）：简洁描述性标题，标识该程序性记忆的主题。
- 适用场景（Use Cases）：触发条件、适用场景和目标类型，描述“什么时候用”。
- 操作步骤（Action Steps）：编号步骤序列，描述“具体做什么”。
- 关键提示（Key Hints）：工具使用模式、决策标准、什么有效、什么该避免。
- 失败模式（Failure Modes）：已知失败场景及恢复策略（可选）。

根据输入信息的质量，程序性记忆支持三个级别的结构化程度：

- Level 1 纯文本摘要：简单经验总结，包含标题、适用场景和关键提示。
- Level 2 自然语言经验：常见操作流程，增加操作步骤和失败模式。
- Level 3 结构化文档：完整的结构化执行纲要，包含全部字段。

适用场景

- Agent 需要固化成功操作流程，避免面对相似任务时重复推理
- 编码智能体需要固化代码仓认知流程、代码审查流程等
- 运维智能体需要结构化执行纲要（前置条件检查、步骤拆解、失败模式处理）
- 需要约束 LLM 推理边界，提供检查点和回滚预案的复杂任务场景

提取内容举例

让Agent探索一个新代码仓并完成功能开发。Agent首次探索后，将认知流程提取为程序性记忆：

提取为程序性记忆：

标题：代码仓探索流程

适用场景：当需要在新代码仓中定位和修改代码时，检测到代码探索意图

操作步骤：

- 先查看README和目录结构，了解项目全貌
- 重点关注src/和tests/目录
- 查找入口文件和配置文件，理解架构设计
- 使用语义搜索定位相关代码
 - 关键提示：先理解整体架构再深入细节；应避免不要在未理解项目结构时直接修改代码
 - 失败模式：目录结构异常 → 检查是否为monorepo，切换到子项目目录

后续遇到相似任务时，Agent直接检索加载该程序性记忆，避免重复推理。

5.3.3 自定义策略

当内置策略（总结、语义记忆、用户偏好、情景记忆、程序性记忆）不满足需求时，可以创建自定义策略来定义自己的记忆提取逻辑。

什么是自定义策略

内置策略的记忆提取逻辑由平台预定义，用户无法修改提取规则和输出格式。自定义策略则将提取逻辑完全交给用户定义——用户编写提示模板（Prompt），告诉模型从对话中提取什么内容、以什么格式输出。平台负责按触发条件调用模型执行提取，并将结果存入长期记忆。

自定义策略可以与内置策略同时启用，互不影响。例如：同时启用语义记忆策略（内置）和一个提取订单信息的自定义策略，Agent既能记住事实知识，又能记住业务数据。

新增自定义策略

步骤1 登录[AgentArts智能体平台](#)。

步骤2 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。

步骤3 单击记忆库名称，查看记忆库详情。

步骤4 在“基本信息”页签，在“长期记忆提取策略”区域，单击“新增自定义策略”，通过基础模型和提示模板自定义记忆处理。

步骤5 配置策略信息后单击“确定”。

- **策略名称：**自定义策略名称，策略名称不可重复。命名规则：包含a-z、A-Z、0-9和（中划线）等有效字符，最多可包含 48 个字符。
- **策略类型：**选择策略类型，然后配置抽取（Extraction）、整合（Consolidation）和反思（Reflection）的提示词模板。提示词模板是自定义策略的核心，决定了模

型从对话中提取什么内容、以什么格式输出结果。提示模板的质量直接决定记忆提取的效果。

请根据页面展示的实际内容配置。

- 抽取 Extraction：从大量数据中提取关键信息和重要特征，可以使用默认用户提示或创建自定义提示。
- 整合 Consolidation：将提取出的信息进行汇总、合并和优化，确保信息的完整性和一致性，可以使用默认用户提示或创建自定义提示。
- 反思 Reflection：系统对过去的经验、行为或数据进行分析和总结，以提取有价值的信息和洞察的过程。

----结束

提示词模板编写原则

原则	说明
明确提取目标	清晰描述要从对话中提取什么类型的信息，避免模糊表述。
定义输出格式	给出明确的JSON结构示例，要求模型严格按格式输出。
提供正例	给出1-2个输入对话和期望输出的示例，帮助模型理解期望。
说明忽略内容	明确指出哪些内容不需要提取，减少噪声记忆。
控制输出长度	单条记忆不宜过长，建议在提示模板中限定输出字段的数量或长度。

以下是一个提取订单信息的自定义策略提示模板示例：

你是一个记忆提取助手。请从以下对话中提取订单相关信息。

提取规则：
1. 只提取明确的订单信息，包括：订单号、商品名称、数量、金额、下单时间。
2. 如果对话中没有订单信息，返回空结果。
3. 忽略与订单无关的闲聊内容。
4. 多个订单分别输出，不要合并。

输出格式（严格按以下 JSON 结构输出，不要添加任何其他文本）：
{
 "orders": [
 {
 "order_id": "订单号",
 "product": "商品名称",
 "quantity": "数量",
 "amount": "金额",
 "order_time": "下单时间"
 }
]
}

示例：
输入对话：用户说"我刚下了一个订单，订单号 A12345，买了 2 台笔记本电脑，一共 12000 元，今天下午下的单。"
期望输出：
{
 "orders": [
 {
 "order_id": "A12345",
 "product": "笔记本电脑",

```
"quantity": "2",  
  "amount": "12000",  
  "order_time": "今天下午"  
}  
]  
}
```

适用场景

行业特定知识提取

内置策略面向通用对话设计，无法识别行业术语和专业概念。自定义策略可以通过提示模板让模型按行业标准提取信息。

行业	提取内容	示例
医疗	症状描述、诊断结论、用药记录	从问诊对话中提取“患者主诉：头痛 3 天；既往用药：布洛芬”
法律	案件类型、法律条款引用、当事人诉求	从法律咨询中提取“案件类型：合同纠纷；引用法条：合同法第 94 条”
金融	交易类型、金额、风险等级	从理财对话中提取“产品类型：基金；申购金额：5 万元”

结构化信息提取

当需要从对话中提取结构化业务数据（而非自然语言摘要）时，自定义策略可以定义精确的字段结构。

场景	提取字段	用途
订单管理	订单号、商品、数量、金额、状态	Agent 记住用户的订单历史，后续查询时直接引用
工单系统	工单号、问题类型、处理状态、责任人	Agent 跟踪用户的问题处理进展
CRM	客户姓名、公司、联系方式、需求标签	Agent 积累客户画像，提供个性化服务

5.4 示例：构建记忆库并集成 SDK 进行对话管理

本示例描述了如何在AgentArts智能体平台上创建记忆库，并通过SDK与记忆库进行交互。记忆库用于存储和管理智能体与用户之间的对话历史，以便于后续的对话管理和个性化服务。

前提条件

- 操作系统：**推荐使用Linux Ubuntu 22.04 ARM64**，服务器可访问公网。
本文命令如无特殊说明，均以Ubuntu为例。为保证镜像构建与云端部署的兼容性，建议使用Ubuntu系统。
因此，为了减少部署失败和额外排查成本，建议直接使用Ubuntu 22.04 ARM64制作AgentArts镜像。
若使用CentOS、EulerOS等非Ubuntu系统，请将下命令中的apt替换为yum或dnf，并自行确认Docker 18.06+、Python 3.10+；本文不逐条展开各发行版安装差异。



注意

需使用ARM64系统制作的Agent镜像，使用X86系统制作的镜像在调用智能体运行时调用失败。

可以使用华为云ECS服务购买服务器，购买时需要选择鲲鹏架构（ARM64），系统选择Ubuntu。

云服务器控制台 / 弹性云服务器 / 购买弹性云服务器

<  购买弹性云服务器

快速购买  自定义购买 

实例  [如何选择ECS实例规格？](#)

规格类型选型 业务场景选型

CPU架构 

x86计算

鲲鹏计算

- 安装Python：请确保Python 3.10及以上版本已安装。
大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version检查。如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

非Ubuntu/Debian系统请使用yum或dnf安装Python 3.10+及pip。

- 安装Docker：请确保Docker 18.06及以上版本已安装。如未安装，可以使用如下命令安装：

```
# 查询 Docker 版本
docker --version

# 安装Docker
sudo apt update
sudo apt install docker.io
```

安装完成后，建议立即配置国内镜像加速器。在后续执行agentarts launch构建镜像时，Docker需要拉取python:3.10-slim基础镜像，默认从docker.io官方仓库下载，可能超时失败。提前配置加速器可避免此问题。

执行以下命令创建Docker配置目录：

```
sudo mkdir -p /etc/docker
```

执行以下命令配置国内镜像源（直接复制并回车执行）：

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors":[
```

```
"https://docker.m.daocloud.net",
"https://dockerproxy.net",
"https://mirror.baidubce.com"
]
}
EOF
```

依次执行以下命令重启Docker服务，使配置生效：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

📖 说明

华为云SWR基础版不支持OCI镜像格式，如果您使用的是Docker 27及以上版本，并且需要处理OCI镜像，可以通过设置环境变量来关闭OCI支持。

```
export DOCKER_BUILDKIT=0
# 或者
export BUILDKIT_USE_OCI_MEDIA_TYPES=0
```

- 执行以下命令安装SDK（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
# 安装依赖包
apt install python3.12-venv

# 创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate

# 安装sdk
pip install agentarts-sdk
```

若配置镜像加速器，后续在执行agentarts launch命令时，仍无法拉取python:3.10-slim，可参考本文结尾的“常见问题”中的“直接修改Dockerfile基础镜像地址”。

📖 说明

- 执行pip install agentarts-sdk命令下载缓慢、超时可以更改为使用如下命令
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
- 如果系统缺少python3-venv包，导致无法创建虚拟环境，请按照命令回显提示安装python3-venv包。

```
root@ecs-6732:~# python3 -m venv myenv
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

apt install python3.12-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

Failing command: /root/myenv/bin/python3
```

- 执行以下命令配置华为云凭证，获取华为云凭证请参考[认证鉴权](#)。
export HUAWEICLOUD_SDK_AK="your-access-key"
export HUAWEICLOUD_SDK_SK="your-secret-key"

构建记忆库并集成 SDK 进行对话管理

- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3** 单击右上角“创建记忆库”。
- 步骤4** 参考[图5-4](#)配置记忆库的基本信息和网络配置。

- 记忆库名称：memory-test
- 网络配置：勾选私有网络，选择业务所在的VPC及子网
公网访问根据业务场景可选择开启
- 入站身份认证：API Key认证
- 数据加密配置：可自定义选择

图 5-4 配置记忆库的基本信息和网络配置

基本信息

记忆库名称

memory-test

有效字符为 a-z、A-Z、0-9 和 (中划线)。该名称最多可包含 48 个字符。字符数：11/48。

描述 (可选)

请输入

0/1,000

网络配置

☒ 私有访问

开启后，私有网络和子网暂时不支持修改。使用私有访问出站网络，当前账号需具备VPC EIP相关权限 [查看当前权限](#)

私有网络

vpc-zyy (11)

Q 新建私有网络

子网

subnet-18e03sdfsdfsdf (11)

Q 新建子网

☐ 公网访问

Memory开启公网访问后，存在一定安全风险，建议仅用于开发测试场景。开启后，暂不支持关闭。

权限与访问控制

入站身份认证

IAM 认证

将华为云 IAM 身份配置为入站身份认证，以验证调用者的身份。

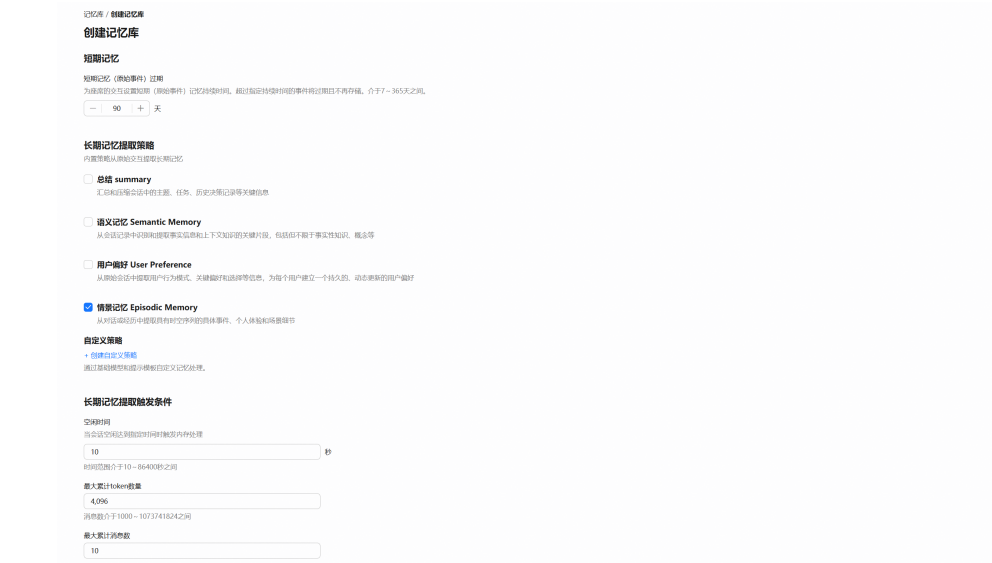
API Key 认证

将预设的密钥对（键和值）配置为入站身份认证，以验证请求中携带的密钥合法性。

步骤5 参考图5-5配置记忆策略。

- 短期记忆：90天。
- 长期记忆提取策略：选择情景记忆 Episodic Memory。
- 长期记忆提取触发条件：
 - 空闲时间：10秒。
 - 最大累计token数量：4096。
 - 最大累计消息数：10。

图 5-5 配置记忆策略

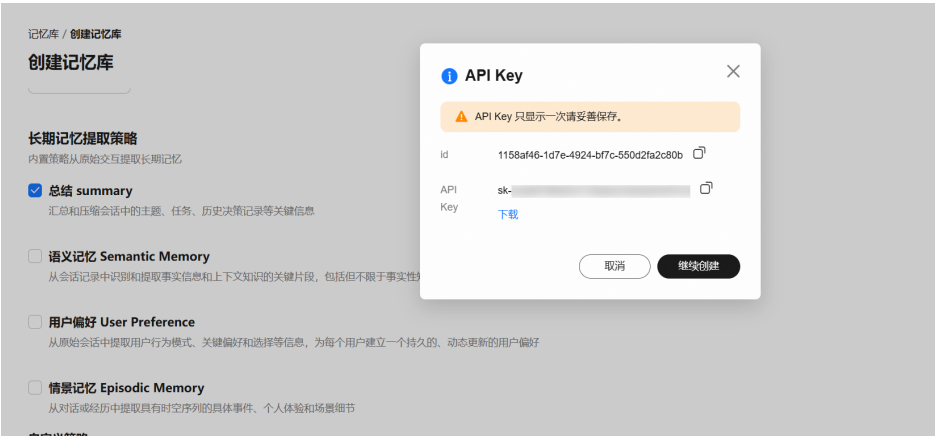


步骤6 单击“创建记忆库”，在“API Key”弹框中，请复制或下载API Key，然后单击“继续创建”。

说明

建议下载保存在本地，API Key一旦丢失只能通过重置方式解决。

图 5-6 复制或下载 API Key



步骤7 返回至记忆库列表，获取记忆库ID（对应space_id）。

图 5-7 获取记忆库 ID



步骤8 在项目目录中创建test.py文件，参考如下示例代码，将SDK集成在应用项目中。请将“space_id”和“api_key”替换为实际使用的值。

```
import time
from agentarts.sdk.memory import MemoryClient
from agentarts.sdk.memory.inner.config import TextMessage
def client_mode_example():
    """Client 模式完整示例（假设已通过Console创建记忆库，并获取spaceId和api-key）本示例仅作参考"""
    space_id = "your_space_id" # 使用页面上复制的记忆库ID
    api_key = "your_api_key" # 使用保存的apikey值
    with MemoryClient(api_key=api_key) as client:
        # 1. 创建会话
        print("\ncreate session...")
        session_data = client.create_memory_session(
            space_id=space_id,
            actor_id="user-001",
            assistant_id="assistant-001"
        )
        session_id = session_data.id
        print(f"Session ID: {session_id}")
        # 2. 添加消息
        print("\nadd chat message...")
        messages = [
            TextMessage(
                role="user",
                content="你好，我想了解机器学习的基础知识",
                actor_id="user-001"
            ),
            TextMessage(
                role="assistant",
                content="机器学习是人工智能的一个分支，主要研究如何让计算机从数据中学习规律。",
                actor_id="assistant-001"
            )
        ]
        client.add_messages(
            space_id=space_id,
            session_id=session_id,
            messages=messages
        )
        print("add message succeed")
        # 3. 等待记忆生成
        print("\nwait for memory...")
        time.sleep(90)
        # 4. 查询记忆
        print("\nsearch memory list...")
        memories = client.list_memories(space_id=space_id, limit=10)
        print(f"find {len(memories.items)} memories")
        # 5. 搜索记忆
        print("\nsearch memories...")
        from agentarts.sdk.memory.inner.config import MemorySearchFilter
        search_results = client.search_memories(
            space_id=space_id,
            filters=MemorySearchFilter(query="机器学习", top_k=3)
        )
        print(f"find {len(search_results.results)} memories")
        return space_id, session_id
if __name__ == "__main__":
    client_mode_example()
```

步骤9 执行如下Python命令启动项目应用，向记忆库批量插入与大型语言模型进行交互时的消息。

python3 test.py

步骤10 进入记忆库界面，在记忆库列表中单击已创建的记忆库名称，在“记忆检索”页签，配置检索条件后单击“确定”，可查看到保存的交互消息已经生成对应的长期记忆。

----结束

5.5 创建记忆库

创建记忆库用于存储和管理用户信息、历史交互和偏好，以便智能体能够提供更个性化、高效和情境适应的响应。

支持在控制台上通过配置长期记忆提取策略、短期记忆等参数来创建记忆库。

前提条件

- 使用记忆库前需要先[开通AgentArts服务](#)并进行服务授权，请在“授权管理”页面进行云服务开通及授权，详细信息请参考[授权管理](#)。
- 如需通过私有网络VPC访问记忆库，请确保已开通虚拟私有云，并已创建私有网络、子网。详细操作请参见[创建虚拟私有云和子网](#)。
- （可选）管理员可在IAM控制台为创建的IAM子用户授予权限，授权后，子用户即可根据权限使用账号中的云服务资源，相关操作步骤如下。
 - a. [创建IAM用户](#)。
 - b. [给IAM用户授权](#)。
 - 为子用户授予完整的记忆库权限，需授予AgentArtsFullAccessPolicy身份策略。
 - 可根据子用户实际需求[创建自定义身份策略](#)并授予如下权限，授权项功能介绍请参考[表5-1](#)。

表 5-1 授权项

授权项	功能介绍
vpc:vpcs:get	授予查询VPC的权限
vpc:subnets:get	授予查询子网的权限
vpcep:endpoints:*	授予终端节点的操作权限
dns:zone:create	授予区域的创建权限
dns:zone:delete	授予区域的删除权限
dns:recordset:create	授予记录集的创建权限
dns:recordset:list	授予记录集的列表权限
dns:recordset:delete	授予记录集的删除权限
kms:cmk:get	授予查看KMS密钥的详细信息的权限
kms:cmk:createGrant	授予给特定密钥创建授权的权限
dns:zone:list	授予区域的列表权限
kms:cmk:list	授予查看用户所有KMS密钥信息的权限
kms:cmk:create	授予创建KMS密钥的权限

创建记忆库

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3 单击右上角“创建记忆库”。
- 步骤4 在“创建记忆库”的页面中，输入记忆库的配置信息，参数说明请参考表5-2。

表 5-2 参数说明

参数		说明
基本信息	记忆库名称	记忆库的名称。 命名规则：包含a-z、A-Z、0-9 和（中划线）等有效字符，最多可包含 48 个字符。 系统会默认生成记忆库的名称，可根据需求自定义修改。
	描述	用于对记忆库内容和用途的简要说明。 规则：长度不大于1000个字符。
高级配置	私网访问	记忆库能够连接到内部的私有网络，访问内部的资源和服务。 开启后，私有网络、子网暂时不支持修改。 选择已配置的VPC和子网，如未配置，请单击“新建私有网络”或“新建子网”并参考 创建虚拟私有云和子网 进行配置，配置后单击  刷新。
	公网访问	开启公网访问后，存在一定安全风险，建议仅用于开发调测场景。开启后，暂不支持关闭。
权限与访问控制	入站身份认证	创建记忆库时需选择入站身份认证方式，支持以下两种方式： ● IAM 认证：将华为云IAM身份配置为入站身份认证，以验证调用者的身份。提供细粒度鉴权能力，适合企业级用户。 ● API Key 认证：将预设的密钥对（键和值）配置为入站身份认证，以验证请求中携带的密钥合法性。系统会自动为记忆库创建和绑定 API Key，适合个人用户和快速验证场景。 关于入站认证的详细介绍请参考 记忆库入站身份认证 。

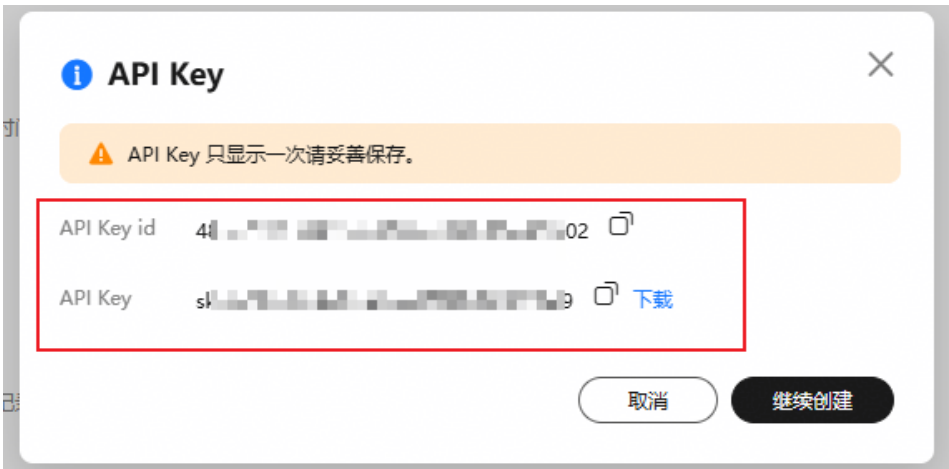
参数		说明
数据加密配置	数据加密配置	选择“默认密钥”或“自定义密钥”。 数据加密功能，可以有效地对会话元数据、消息内容、记忆内容、自定义策略的提示词等敏感数据进行加密保护，满足用户对数据自主加密管控的合规需求，从而降低数据泄露等安全风险。 ● 默认密钥 使用云服务默认密钥（agentarts/default）加密。 ● 自定义密钥 通过自定义创建的KMS密钥进行加密。 - 密钥生成算法：当前支持选择“AES_256”。 AES是一种对称加密算法，AES_256表示使用256位密钥长度的AES算法，提供很高的安全性。 - 密钥名称：在下拉选项中选择已创建的自定义密钥。也可单击“创建密钥”创建一个新的密钥，单击“查看密钥列表”可以查看已有的密钥列表。 说明 密钥名称用于标识具体的加密密钥。在使用过程中，密钥不能被禁用、编辑或冻结，否则会导致记忆库不可用。 记忆库创建后，数据加密功能不可修改。 关于自定义密钥和默认密钥的区别可参考自定义密钥与默认密钥有什么区别？ 自定义密钥以及默认密钥均会收取相关费用，费用说明请参考DEW 计费说明。
短期记忆	短期记忆（原始事件）过期	存储智能体在当前任务或会话中的临时信息和状态，通常在任务完成后清空。 用于设置短期（原始事件）记忆持续时间，超过设置的持续时间的事件过期后不再存储。时间设置可设置为7～365天之间。
长期记忆提取模型配置	模型配置	使用“系统默认”即可。
长期记忆提取策略	总结 summary	勾选后，将互动的大量详细信息或数据压缩成关键上下文和关键洞察信息，帮助智能体更高效地检索和利用长期记忆中的信息。如会议总结、年度工作总结等。
	语义记忆 Semantic Memory	勾选后，使用与上下文无关的格式从原始对话中提取一般的事实性知识、概念和含义，帮助智能体理解和处理抽象信息。如个人信息、专业信息、教育经历等。
	用户偏好 User Preference	勾选后，存储和管理用户在与智能体交互过程中表现出的偏好、习惯和个性化需求，用户偏好帮助智能体更好地理解 and 适应用户的需求。如个人爱好、运动习惯等。

参数		说明
	情景记忆 Episodic Memory	勾选后，保留上下文的格式，从原始对话或经历中提取具有时空序列的具体事件、个人体验和场景细节，帮助智能体记住具体的事件和经历，支持其在类似情境下做出更准确的决策和响应。如旅游中的个人体验、看电影中的场景等。
	程序性记忆 Procedural Memory	勾选后，从会话记录中提取操作流程、行动步骤和关键提示等信息，形成可复用的任务执行指南，帮助智能体固化成功流程与经验，避免面对相似任务时重复推理，约束推理边界。如代码仓探索流程、运维操作执行纲要等。
自定义策略 （可选）	自定义策略	用户可通过选择基础模型和定义提示模板，灵活地处理和生成特定类型的记忆内容，最多可创建10个自定义策略。 单击“创建自定义策略”。 - 策略名称：自定义策略名称，策略名称不可重复。命名规则：包含a-z、A-Z、0-9 和（中划线）等有效字符，最多可包含 48 个字符。 - 策略类型：选择策略类型，然后配置抽取（Extraction）、整合（Consolidation）和反思（Reflection）的提示词模板。请根据页面展示的实际内容配置。 - 抽取 Extraction：从大量数据中提取关键信息和重要特征，可以使用默认用户提示或创建自定义提示。 - 整合 Consolidation：将提取出的信息进行汇总、合并和优化，确保信息的完整性和一致性，可以使用默认用户提示或创建自定义提示。 - 反思 Reflection：系统对过去的经验、行为或数据进行分析 and 总结，以提取有价值的信息和洞察的过程。
长期记忆提取触发条件	空闲时间	在没有接收到新任务或用户交互的一段时间后，触发长期记忆的提取。 请输入具体的时间，时间范围介于10～86400秒之间，单位为秒。
	最大累计token数量	在处理用户输入时，累计的token数量达到预设的最大值时，触发长期记忆的提取。 请输入具体的token数量，数量介于1000～1073741824之间。
	最大累计消息数	在处理用户输入时，累计的消息数量达到预设的最大值时，触发长期记忆的提取。 请输入具体的消息数，消息数介于3～10000之间。

参数		说明
高级配置	标签（可选）	由“标签键”和“标签值”组成，用于标识和分类云资源。 您可以在TMS中创建预定义标签，用于使用同一标签标识多种云资源，创建后可在标签输入框下拉选择同一标签，具体操作请参考 创建预定义标签 ，创建后单击刷新。 单击“添加标签”，选择在TMS中创建的预定义标签，或自定义输入标签键和标签值。 可添加一个或多个标签，最多添加20个标签。

步骤5 单击“立即创建”。

- 选择“IAM 认证”时，记忆库创建成功后直接通过 IAM 临时凭证访问数据面接口。详细的IAM认证方式请参见[记忆库入站身份认证](#)。
- 选择“API Key 认证”时，在“API Key”弹框中，查看或复制API Key id及API Key，API Key只显示一次请妥善保存。单击“继续创建”，则记忆库创建成功，即可在记忆库列表展示已创建的记忆库。
 - API Key ID：[创建记忆库接口](#)时，作为请求参数传入，用于为记忆库绑定对应的鉴权密钥。
 - API Key：数据面API认证，用于调用数据面API时，在请求头中携带进行身份认证，格式为：Authorization: Bearer \${api_key}。



步骤6 单击“继续创建”，则记忆库创建成功，即可在记忆库列表展示已创建的记忆库。

----结束

后续操作

创建记忆库成功后，您可以在智能体中集成记忆库，以保留上下文、用户偏好等，实现跨会话的持久化。详细操作请参见[在Agent中集成记忆库](#)。

更多操作

您还可以对已创建的记忆库执行如下操作。

表 5-3 更多操作

操作	说明
查看并修改记忆库	1. 进入记忆库界面。 2. 在记忆库列表中单击记忆库名称，查看记忆库详情。 3. 在“记忆库详情”页签，支持修改短期记忆持续时间、为记忆库新增内置策略或新增自定义策略、修改长期记忆触发条件等。
删除记忆库	记忆库删除后不可恢复，请谨慎操作。 1. 进入记忆库界面。 2. 单击记忆库列表操作列的“删除”。 3. 在弹框中确认要删除后输入“DELETE”，然后单击“确定”。
更新API Key	1. 单击记忆库名称，进入“基本信息”界面，在“API Key”处单击“更新”。 图 5-8 更新 API Key  2. 弹框中单击“确定”后，复制或下载API Key。
认证方式升级	仅支持从API Key认证升级为IAM认证。 1. 进入记忆库界面，单击记忆库名称进入详情页面。 2. 在“入站身份认证”区域，单击编辑图标，将API Key认证升级为IAM认证（单向升级，升级后不可修改）。升级时可选择是否立即失效原API Key，不立即失效时API Key将在 7 天后自动失效。详细说明请参见认证方式升级。

5.6 记忆库入站身份认证

记忆库作为智能体记忆数据的托管存储服务，为保障数据安全，记忆库需要对每一个到达数据面的请求进行身份验证，确保请求来自可信的调用方，并根据调用方身份进行授权控制。

支持的认证方式

记忆库支持以下两种入站身份认证方式，在创建记忆库时二选一：

认证方式	认证凭据	鉴权能力	适用用户
IAM 认证	IAM临时凭证（AK/SK签名认证）	提供细粒度鉴权能力，支持actor级和session级数据隔离	企业级用户

认证方式	认证凭据	鉴权能力	适用用户
API Key 认证	API Key 字符串	仅提供身份认证能力（无鉴权）	个人用户、快速验证场景

IAM 认证

IAM认证将华为云IAM身份配置为入站身份认证，通过IAM临时凭证（AK/SK）验证调用者身份。IAM认证不仅提供身份验证能力，还支持通过IAM策略进行细粒度鉴权，实现数据级隔离控制。

使用IAM认证方式访问记忆库数据接口时，调用方需携带IAM签名信息（AK/SK 签名认证）：

```
POST /v1/spaces/{space_id}/memories HTTP/1.1
Host: memory.cn-southwest-2.huaweicloud-agentarts.com
Content-Type: application/json
X-Sdk-Date: 20260909T033655Z
Authorization: V11-HMAC-SHA256 Access={AK}, SignedHeaders=content-type;host;x-sdk-date, Signature=...
X-Security-Token: {SecurityToken}

{"content": "用户偏好信息"}
```

API Key 认证

API Key认证是一种轻量级的认证方式。调用方在请求头中携带API Key字符串即可完成身份认证。关于API KEY的获取请参考[如何获取记忆库ID和API Key](#)。

请求格式：

```
POST /v1/spaces/{space_id}/memories HTTP/1.1
Host: memory.cn-southwest-2.huaweicloud-agentarts.com
Content-Type: application/json
Authorization: Bearer {您的API Key}

{"content": "用户偏好信息"}
```

认证方式升级

记忆库支持从API Key认证单向升级为IAM认证，升级后原有API Key将在宽限期结束后失效。

升级规则

- 单向升级：仅支持从API Key认证升级为IAM认证，升级后不可修改。
- 宽限期结束：若未开启立即失效，7天宽限期结束后API Key自动失效，所有请求需通过IAM认证访问。
- 升级后不可逆：认证方式升级为 IAM后，不支持切换回API Key认证。

说明

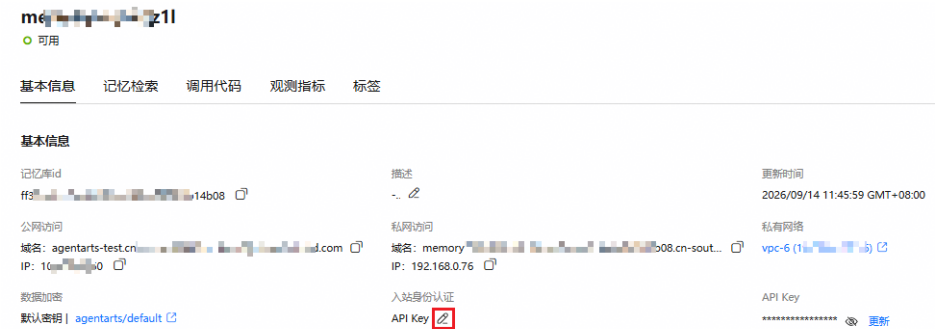
升级后请及时更新相关SDK和接口的认证参数，否则会导致调用失败及数据接口认证失败。

升级操作

步骤1 登录[AgentArts智能体平台](#)。

- 步骤2** 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3** 单击记忆库名称，进入基本信息页面。
- 步骤4** 在“入站身份认证”区域，当前显示为“API Key”，单击右侧的编辑图标 。

图 5-9 升级为 IAM 认证



- 步骤5** 在弹出的“升级为IAM认证”对话框中，配置“API Key 立即失效”开关。
- 开启：原API Key在升级完成时立即失效。
 - 关闭（默认）：API Key认证将在7天后自动失效，宽限期内API Key仍可正常使用，但如果调用了重置API接口则API Key会立刻失效。
- 步骤6** 在确认输入框中输入“CONFIRM”（可单击“一键输入”快速填入），然后单击“确定”。

 说明

- 如果未开启“API Key 立即失效”，请务必在7天宽限期内将所有调用方切换为IAM认证方式，宽限期结束后API Key将自动失效。
- 升级后认证方式不可修改，请确认后再进行操作。

----结束

5.7 在 Agent 中集成记忆库

5.7.1 概述

在智能体中集成记忆库，实现智能体与记忆库的快速对接。记忆库创建完成后，需要通过SDK将记忆能力集成到Agent代码中。集成后Agent具备：

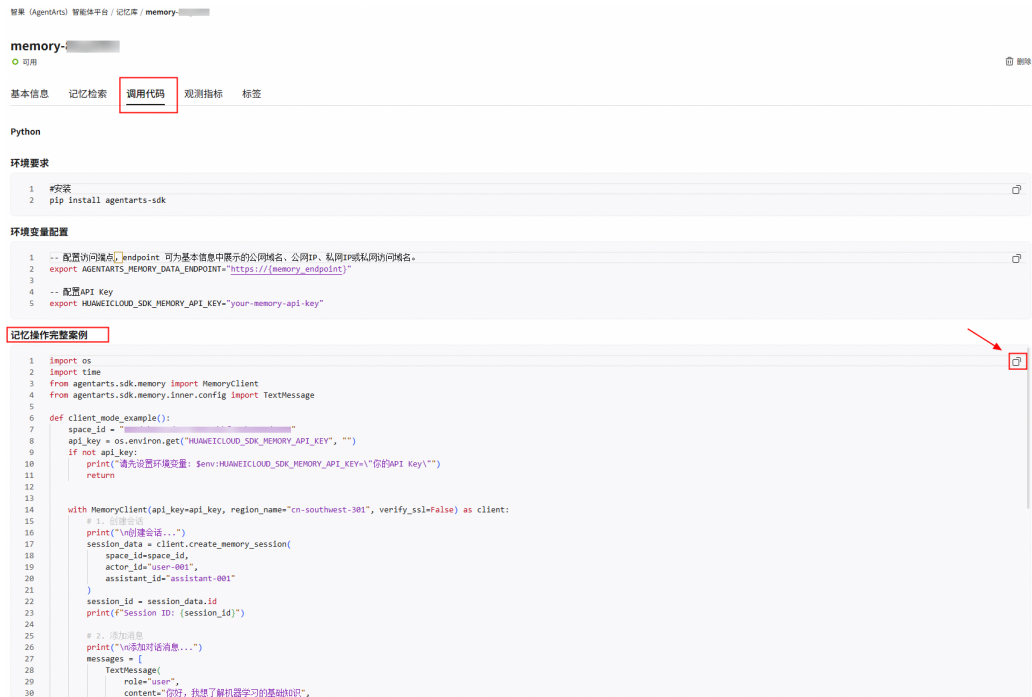
- 在对话中自动存储每轮交互（短期记忆）
- 跨会话检索用户历史和偏好（长期记忆）
- 基于记忆给出个性化响应

完整集成示例

以下示例展示了通过memory SDK将记忆库集成到Agent 中的完整流程：创建会话、添加消息（短期记忆）、等待记忆生成、查询和搜索记忆（长期记忆）。请将“space_id”和“api_key”替换为实际使用的值。内置策略还支持选择语义记忆、用户偏好、情景记忆、程序性记忆等，创建记忆库时可根据业务场景勾选对应的策略。

```
import time
from agentarts.sdk.memory import MemoryClient
from agentarts.sdk.memory.inner.config import TextMessage
def client_mode_example():
    """Client 模式完整示例（假设已通过Console创建记忆库，并获取spaceId和api-key） 本示例仅作参考"""
    space_id = "your_space_id" # 使用页面上复制的记忆库ID
    api_key = "your_api_key" # 使用保存的apikey值
    with MemoryClient(api_key=api_key) as client:
        # 1. 创建会话
        print("\ncreate session...")
        session_data = client.create_memory_session(
            space_id=space_id,
            actor_id="user-001",
            assistant_id="assistant-001"
        )
        session_id = session_data.id
        print(f"Session ID: {session_id}")
        # 2. 添加消息
        print("\nadd chat message...")
        messages = [
            TextMessage(
                role="user",
                content="你好，我想了解机器学习的基础知识",
                actor_id="user-001"
            ),
            TextMessage(
                role="assistant",
                content="机器学习是人工智能的一个分支，主要研究如何让计算机从数据中学习规律。",
                actor_id="assistant-001"
            )
        ]
        client.add_messages(
            space_id=space_id,
            session_id=session_id,
            messages=messages
        )
        print("add message succeed")
        # 3. 等待记忆生成
        print("\nwait for memory...")
        time.sleep(90)
        # 4. 查询记忆
        print("\nsearch memory list...")
        memories = client.list_memories(space_id=space_id, limit=10)
        print(f"find {len(memories.items)} memories")
        # 5. 搜索记忆
        print("\nsearch memories...")
        from agentarts.sdk.memory.inner.config import MemorySearchFilter
        search_results = client.search_memories(
            space_id=space_id,
            filters=MemorySearchFilter(query="机器学习", top_k=3)
        )
        print(f"find {len(search_results.results)} memories")
        return space_id, session_id
if __name__ == "__main__":
    client_mode_example()
```

同样的代码可以在记忆库调用代码下的记忆操作完整案例中进行复制



执行如下 Python 命令启动项目应用，向记忆库批量插入与大型语言模型进行交互时的消息：

python3 test.py

根据打印日志检查执行结果。当触发条件满足时，短期记忆自动通过策略提取为长期记忆；下次该用户对话时，可检索到这些长期记忆并注入上下文。

说明

长期记忆提取是异步的，建议短期记忆写入 3~5 min 后再进行长期记忆查询。

5.7.2 环境准备

前提条件

- 已[创建记忆库](#)，并获取记忆库的API Key。
 - 已安装Python，且版本不低于3.10。
 - 执行以下命令安装SDK（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。
- ```
安装依赖包
apt install python3.12-venv

创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate

安装sdk
pip install agentarts-sdk
```

📖 说明

- 执行pip install agentarts-sdk命令下载缓慢、超时可以更改为使用如下命令  
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
- 如果系统缺少python3-venv包，导致无法创建虚拟环境，请按照命令回显提示安装python3-venv包。

```
root@ecs-6732:~# python3 -m venv myenv
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

apt install python3.12-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

Failing command: /root/myenv/bin/python3
```

配置环境变量

创建记忆库后，复制或下载API Key并妥善保存，用于后续访问记忆库。获取记忆库 ID（对应 space\_id）后，配置以下环境变量：

```
通过 console 创建记忆库后，可以进行记忆操作。
首先需要配置 API Key，配置方式：
方式一：环境变量配置,若不传入,SDK会尝试从环境变量中获取apikey
export HUAWEICLOUD_SDK_MEMORY_API_KEY="your-memory-api-key"

方式二：代码中直接传入
from agentarts.sdk.memory import MemoryClient
client = MemoryClient(api_key="your-memory-api-key")

其次可以配置访问端点
export AGENTARTS_MEMORY_DATA_ENDPOINT="https://{memory_endpoint}"
endpoint为基本信息内展示的公网域名、公网ip、私网ip或私网访问域名
例如：
export AGENTARTS_MEMORY_DATA_ENDPOINT="https://memory-{space_id}.cn-southwest-2.huaweicloud-agentarts.com"

也可通过配置region信息，SDK会自行拼接如下域名"https://memory.{region_name}.huaweicloud-agentarts.com"
client = MemoryClient(region_name="cn-southwest-2")
```

获取记忆库关键参数

后续代码中需要用到的参数：

- space\_id：记忆库 ID，在详情页“基本信息”获取。

图 5-10 获取记忆库 ID



- region\_name：区域，如cn-southwest-2。

- endpoint: 访问端点。
- api\_key: API Key，获取请参见[如何获取记忆库ID和API Key](#)。

### 5.7.3 连接记忆库

#### 创建 MemoryClient

通过 MemoryClient 管理与记忆库的连接，建议使用 with 语句确保连接正确关闭：

记忆库支持两种认证方式，请根据创建记忆库时选择的认证方式选择对应的连接方法：

#### 通过 API Key 认证连接

```
from agentarts.sdk.memory import MemoryClient

with MemoryClient(api_key=api_key) as client:
 # 后续所有操作在此上下文中进行
 pass
```

- api\_key 既可通过环境变量（HUAWEICLOUD\_SDK\_MEMORY\_API\_KEY）传入，也可在代码中直接传入。
- 也可通过配置 region\_name，SDK 会自行拼接访问域名（格式为 https://memory.{region\_name}.huaweicloud-agentarts.com）。  
client = MemoryClient(region\_name="cn-southwest-2")
- 也可通过环境变量配置访问端点 AGENTARTS\_MEMORY\_DATA\_ENDPOINT（公网域名、公网 IP、私网 IP 或私网访问域名）。

#### 通过 IAM 认证连接

当记忆库选择 IAM 认证方式时，通过 IAM 临时凭证（AK/SK/SecurityToken）进行认证，SDK 会自动处理签名：

```
from agentarts.sdk.memory import MemoryClient

with MemoryClient(
 ak="your-ak",
 sk="your-sk",
 security_token="your-security-token" # 使用STS临时凭证时需要
) as client:
 # 后续所有操作在此上下文中进行
 pass
```

- AK/SK 既可通过环境变量（HUAWEICLOUD\_SDK\_AK、HUAWEICLOUD\_SDK\_SK）传入，也可在代码中直接传入。
- 使用 STS 临时凭证时，需额外传入 security\_token 参数，或通过环境变量 HUAWEICLOUD\_SDK\_SECURITY\_TOKEN 传入。

#### 连接失败排查

| 现象      | 排查方向                                                                             |
|---------|----------------------------------------------------------------------------------|
| 连接超时    | endpoint是否正确；网络是否可达（公网/私网）                                                       |
| 401认证失败 | API Key 认证：API Key是否正确；API Key是否已更新或过期。IAM 认证：AK/SK是否正确；SecurityToken是否过期；签名是否正确 |

| 现象      | 排查方向                                                                                  |
|---------|---------------------------------------------------------------------------------------|
| 403鉴权失败 | IAM 认证：是否因 actor 级或 session 级隔离策略被拒绝；检查 IAM 策略中 MemoryActorId 和 MemorySessionId 条件键配置 |
| SSL证书错误 | verify_ssl参数设置                                                                        |

## 5.7.4 管理会话

记忆的读写操作都在会话上下文中进行。每次对话开始前，需要创建一个会话。

### 创建会话

创建一个会话，用于承载一次对话的短期记忆。会话可绑定用户、助手等归属信息。

```
session_data = client.create_memory_session(
 space_id="记忆库ID",
 actor_id="user-001",
 assistant_id="assistant-001"
)
session_id = session_data.id
```

参数说明：

- space\_id：记忆库ID。
- actor\_id：终端用户的唯一标识，用于按用户隔离记忆。
- assistant\_id：Agent的标识。

#### 说明

- 同一个用户的不同会话应使用相同的actor\_id，这样长期记忆才能跨会话共享。
- 不同用户必须使用不同的actor\_id，否则记忆数据会串扰。
- 建议使用业务系统中的用户ID作为 actor\_id。

### 获取会话信息

查询会话的详情或列表，用于恢复会话上下文或管理会话。

```
查询单个会话详情
session_data = client.show_memory_session(
 space_id="记忆库ID",
 session_id="会话ID"
)

查询会话列表
session_list = client.list_memory_sessions(
 space_id="记忆库ID",
 actor_id="user-001", # 可选
 limit=20, # 可选
 offset=0 # 可选
)
```

| 参数       | 说明             | 约束 |
|----------|----------------|----|
| space_id | 记忆库ID（必填，路径参数） | —  |



| 参数       | 说明                                           | 约束                |
|----------|----------------------------------------------|-------------------|
| actor_id | 按Actor ID过滤（可选）                              | 1~64 字符           |
| limit    | 每页数量（可选）                                     | 1~20，默认 20        |
| offset   | 偏移量（可选），从第几条开始返回（跳过前 offset 条），配合 limit 实现分页 | 0~2147483647，默认 0 |

## 关闭会话

关闭（删除）会话。关闭后会话不可再写入或查询消息，会话中的消息按空间配置的保留期异步清理。

```
client.delete_memory_session(
 space_id="记忆库ID",
 session_id="会话ID"
)
```

## 注意事项

关闭会话为逻辑删除，会话内的消息不会立即清除，将按空间配置的保留期异步清理。若业务涉及数据删除合规要求，请知悉该行为。

## 5.7.5 短期记忆操作

短期记忆以“消息（Message）”为单位存储。对话中的用户输入、Agent回复、工具调用和系统消息，均作为消息写入会话，实现对话上下文的持久化。

## 追加消息

在Agent的对话循环中，每轮交互后追加消息。支持一次写入多条消息，便于一次性提交完整对话轮次。

```
from agentarts.sdk.memory.inner.config import TextMessage

messages = [
 TextMessage(
 role="user",
 content="你好，我想了解机器学习的基础知识",
 actor_id="user-001"
),
 TextMessage(
 role="assistant",
 content="机器学习是人工智能的一个分支，主要研究如何让计算机从数据中学习规律。",
 actor_id="assistant-001"
)
]
client.add_messages(
 space_id=space_id,
 session_id=session_id,
 messages=messages
)
```

写入特性：

- 同步写入，写入后立即可读

- 批量追加在单个事务中执行，全部成功或全部失败
- 服务端自动分配递增序号（seq），保证消息顺序
- 写入后消息不可修改、不可删除

## 列出消息

按会话分页查询消息列表，支持排序。

```
result = client.list_messages(
 space_id=space_id,
 session_id=session_id,
 limit=20,
 offset=0,
 order="asc",
 sort_by="seq"
)
```

参数说明：

| 参数         | 说明                                    |
|------------|---------------------------------------|
| space_id   | 记忆库ID                                 |
| session_id | 会话ID                                  |
| limit      | 每页数量，1~20，默认 20                       |
| offset     | 偏移量，默认0，配合limit实现分页                   |
| order      | 排序方向，asc（从旧到新，默认）/ desc（从新到旧）         |
| sort_by    | 排序字段，seq（按序号，默认）/ message_time（按消息时间） |

## 查看单条消息

```
message = client.show_message(
 space_id=space_id,
 session_id=session_id,
 message_id="消息ID"
)
```

## 5.7.6 长期记忆操作

长期记忆由记忆策略从短期记忆中自动提取生成。Agent代码中主要的操作是检索长期记忆并将其注入对话上下文；同时也支持列表查询等管理操作。

### 检索长期记忆

在每次对话开始或用户发送消息时，检索与当前上下文相关的长期记忆：

```
from agentarts.sdk.memory.inner.config import MemorySearchFilter

search_results = client.search_memories(
 space_id=space_id,
 filters=MemorySearchFilter(query="机器学习", top_k=3)
)
print(f"find {len(search_results.results)} memories")
```

参数说明：

| 参数    | 说明                                        |
|-------|-------------------------------------------|
| query | 检索关键词（≤2000 字符），通常使用用户最新消息；不传时按过滤条件返回记忆列表 |
| top_k | 返回最相关的前 N 条，1~100，默认 10                   |

返回结果：返回一个记忆记录列表（results）。列表中的每条记录包含记忆内容、策略类型、时间、相关度分数；列表按相关度分数从高到低排序，len(results) 即返回的记录条数。

 说明

**集成模式：** 将检索到的长期记忆拼接到 Agent 的系统提示词中，作为上下文信息提供给大模型：

```
system_prompt = f"""你是一个智能助手。
以下是关于当前用户的历史记忆：
{search_results}
请基于这些记忆提供个性化的回复。"""
```

查询长期记忆列表

列出指定记忆库中的长期记忆记录：

```
memories = client.list_memories(space_id=space_id, limit=10)
```

删除长期记忆记录

删除一条长期记忆记录，删除后该记忆不再可检索：

```
client.delete_memory(
 space_id=space_id,
 memory_id="记忆ID"
)
```

5.8 安装记忆插件

通过记忆插件，将华为云AgentArts记忆库与Hermes、Claude Code、Codex、OpenCode等主流AI智能体框架无缝对接，安装后，Agent 在对话过程中自动将记忆写入AgentArts Memory，并在每轮对话前搜索并注入相关记忆，实现跨会话记忆持久化。

前提条件

- 已创建记忆库，并获取以下信息：
  - 记忆库ID（Space ID）**：在记忆库详情页获取。
  - API Key**：创建记忆库时生成的API Key，用于数据面API认证。
  - 区域（Region）**：记忆库所在区域，如cn-southwest-2。
- 已安装Python，且版本不低于3.10。

- 已安装agentarts-sdk，版本不低于0.1.5。  

```
pip install agentarts-sdk
```

📖 说明

下载缓慢或超时，可使用华为云镜像源：  

```
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
```
- 已安装并配置好目标智能体框架（Hermes、Claude Code、Codex或OpenCode）。

配置环境变量

安装记忆插件前，需要配置以下环境变量，用于记忆库的认证和访问。

表 5-4 环境变量说明

| 环境变量                           | 说明                               | 是否必填 | 默认值            |
|--------------------------------|----------------------------------|------|----------------|
| HUAWEICLOUD_SDK_MEMORY_API_KEY | AgentArts记忆库的API Key，用于数据面API认证。 | 是    | -              |
| AGENTARTS_MEMORY_SPACE_ID      | AgentArts记忆库ID（Space ID）。        | 是    | -              |
| HUAWEICLOUD_SDK_REGION         | 华为云区域名称。                         | 否    | cn-southwest-2 |

配置方式：

```
export HUAWEICLOUD_SDK_MEMORY_API_KEY="your-memory-api-key"
export AGENTARTS_MEMORY_SPACE_ID="your-space-id"
export HUAWEICLOUD_SDK_REGION="cn-southwest-2"
```

- 📖 说明

安装插件时若使用交互模式，安装器会自动将环境变量写入Shell RC文件（如 ~/.bashrc 或 ~/.zshrc），下次打开终端无需重新设置。Windows系统通过setx命令写入用户环境变量。

支持的平台

| 平台           | target 名称 | 配置目录（global）              | 配置目录（project） | 是否需要Node.js |
|--------------|-----------|---------------------------|---------------|-------------|
| Codex        | codex     | ~/.codex                  | .codex/       | 是           |
| Claude Code  | claude    | ~/.claude                 | .claude/      | 是           |
| Hermes Agent | hermes    | \$HERMES_HOME 或 ~/.hermes | N/A（固定user级）  | 否           |
| OpenCode     | opencode  | ~/.config/opencode        | .opencode/    | 是           |

Windows下Hermes配置目录为 %LOCALAPPDATA%\hermes（由 Hermes 安装器设置 HERMES\_HOME 环境变量）。

## 安装记忆插件

**步骤1** 安装agentarts-sdk后，使用以下命令为指定智能体框架一键安装记忆插件。

```
agentarts memory install [hermes|claude|codex|opencode] --global --yes
```

- **--global**: 全局安装（写入用户主目录配置）。如不指定，则为项目级安装（写入当前工作目录配置）。
- **--yes**: 跳过交互确认，直接安装。

**步骤2** 安装完成后，重启或重新加载智能体框架，插件即生效。

安装器会自动完成以下操作：

- 复制插件脚本文件到目标框架的插件目录。
- 合并框架配置文件（hooks、MCP servers等），保留已有配置。
- 记录安装信息到~/.agentarts/installed.json注册表。

----结束

## 安装示例

### 示例 1：自动检测并交互安装

```
agentarts memory install
```

### 示例 2：安装到 Claude Code（全局）

```
先设置环境变量
export AGENTARTS_MEMORY_SPACE_ID="my-space-12345"
export HUAWEICLOUD_SDK_MEMORY_API_KEY="abcdefghijklmnopqrstuvwxyz123456"
export HUAWEICLOUD_SDK_REGION="cn-southwest-2"

安装
agentarts memory install claude --global
```

### 示例 3：非交互模式安装 Codex（项目级）

```
agentarts memory install codex --yes
```

--yes 模式下：跳过所有交互提示，凭据从环境变量读取（缺失则使用默认值），范围默认 project。

### 示例 4：安装 Hermes 插件

```
agentarts memory install hermes --global
```

Hermes 固定为 global 级安装，--global 参数可选。

### 示例 5：查看已安装的插件

安装记录存储在 ~/.agentarts/installed.json，可直接查看：

```
cat ~/.agentarts/installed.json
```

## 卸载记忆插件

如需移除记忆插件，使用以下命令：

```
交互式选择要卸载的平台
agentarts memory uninstall

直接指定平台
agentarts memory uninstall codex --global --yes
```

卸载操作会：

- 删除插件脚本文件及插件目录。
- 精确剥离配置文件中的插件相关配置（hooks、MCP servers等），保留用户已有其他配置。
- 清理空目录。
- 更新或删除~/.agentarts/installed.json注册表中的安装记录。

### 说明

卸载不会删除记忆库中已存储的记忆数据。如需删除记忆数据，请在AgentArts控制台删除记忆库。

## 5.9 记忆检索

### 5.9.1 通过控制台检索

从记忆库中快速检索已有的记忆，帮助智能体迅速获取用户的历史信息、偏好和过往交互。

#### 前提条件

已创建记忆库并已集成记忆库。

#### 记忆检索

- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3** 在记忆库列表中单击已创建的记忆库名称。
- 步骤4** 在“记忆检索”页签，参考[表5-5](#)配置检索条件。

表 5-5 参数说明

| 参数         | 说明            |
|------------|---------------|
| 用户ID（可选）   | 输入要检索记忆的用户ID。 |
| 会话ID（可选）   | 输入要检索记忆的会话ID。 |
| 日期范围（可选）   | 选择要检索记忆的日期范围。 |
| 记忆策略类型（可选） | 选择要检索的记忆策略类型。 |

| 参数         | 说明                                                                                                                                                                                                                                                 |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 记忆策略名称（可选） | 选择要检索的记忆策略名称。                                                                                                                                                                                                                                      |
| 查询内容（可选）   | 输入要检索的内容。                                                                                                                                                                                                                                          |
| Rerank（可选） | 用于决定是否对检索结果进行重新排序。 <ul style="list-style-type: none"><li>• true：系统会对检索结果进行重新排序，以优化结果的相关性和准确性。</li><li>• false：系统不会对检索结果进行重新排序，直接返回原始的检索结果。</li></ul> 选择“true”通常会提高检索结果的质量，但可能会增加处理时间。选择“false”则会减少处理时间，但可能会影响结果的相关性。根据具体需求选择合适的选项，如不填写，默认按照true处理。 |
| 返回结果数量     | 检索后返回的结果数量，输入值必须在1到100之间。                                                                                                                                                                                                                          |

**步骤5** 单击“确认”后，在搜索框中输入需要检索的问题，平台将返回与该问题相关的文本内容。

----结束

5.9.2 通过 SDK 检索

通过SDK可以在Agent代码中检索长期记忆，并将其注入到对话上下文中。

查询记忆列表

列出指定记忆库中的长期记忆记录：

```
memories = client.list_memories(space_id=space_id, limit=10)
```

集成记忆

```
from agentarts.sdk.memory.inner.config import MemorySearchFilter

search_results = client.search_memories(
 space_id=space_id,
 filters=MemorySearchFilter(query="机器学习", top_k=3)
)
遍历每条记录
for i, r in enumerate(search_results.results):
 record = r["record"]
 print(f"--- 结果 {i+1} ---")
 print(f" 内容: {record['content'][:50]}")
 print(f" 策略类型: {record['strategy_type']}")
 print(f" 时间: {record['created_at']}")
 print(f" 相关度分数: {r['score']}")
```

参数说明：

- query：检索关键词，通常使用用户最新消息。
- top\_k：返回最相关的前N条记忆。
- 返回结果：返回一个记忆记录列表（results）。列表中的每条记录包含记忆内容、策略类型、时间、相关度分数；列表按相关度分数从高到低排序。

集成模式

将检索到的长期记忆拼接到Agent的系统提示词中，作为上下文信息提供给大模型：

```
system_prompt = f"""你是一个智能助手。
以下是关于当前用户的历史记忆：
{search_results}
请基于这些记忆提供个性化的回复。"""
```

说明

SDK检索的详细参数和返回字段说明，参见[长期记忆操作](#)。

5.10 观测记忆指标

5.10.1 支持的观测指标

功能说明

本节定义了上报云监控服务（CES）的监控指标的命名空间，监控指标列表和维度定义，用户可以通过云监控服务提供管理控制台来检索产生的监控指标和告警信息。

命名空间

SYS.AgentArts

监控指标

| 指标ID            | 指标名称    | 指标含义                   | 取值范围 | 单位    | 进制 | 维度                        | 监控周期（原始指标） |
|-----------------|---------|------------------------|------|-------|----|---------------------------|------------|
| requests        | 记忆访问QPS | 当前Memory Space的调用次数    | ≥0   | count | 1  | agentarts_memory_space_id | 1 分钟       |
| throttled_calls | 记忆访问流控  | 当前Memory Space被流控的调用次数 | ≥0   | count | 1  | agentarts_memory_space_id | 1 分钟       |
| req_count_error | 记忆访问失败  | 当前Memory Space调用失败的次数  | ≥0   | count | 1  | agentarts_memory_space_id | 1 分钟       |



| 指标ID                       | 指标名称      | 指标含义                      | 取值范围 | 单位    | 进制 | 维度                        | 监控周期<br>(原始指标) |
|----------------------------|-----------|---------------------------|------|-------|----|---------------------------|----------------|
| avg_latency                | 记忆响应时延    | 当前 Memory Space调用平均时延     | ≥0   | ms    | 1  | agentarts_memory_space_id | 1 分钟           |
| events_requests            | 短期记忆访问QPS | 当前 Memory Space短期记忆调用次数   | ≥0   | count | 1  | agentarts_memory_space_id | 1 分钟           |
| events_request_count_error | 短期记忆访问失败  | 当前 Memory Space短期记忆接口失败次数 | ≥0   | count | 1  | agentarts_memory_space_id | 1 分钟           |
| events_avg_latency         | 短期记忆响应时延  | 当前 Memory Space短期记忆请求时延   | ≥0   | ms    | 1  | agentarts_memory_space_id | 1 分钟           |
| events_count               | 短期记忆数量    | 当前 Memory Space短期记忆条数     | ≥0   | count | 1  | agentarts_memory_space_id | 1 分钟           |
| memory_requests            | 长期记忆访问QPS | 当前 Memory Space长期记忆调用次数   | ≥0   | count | 1  | agentarts_memory_space_id | 1 分钟           |
| memory_request_count_error | 长期记忆访问失败  | 当前 Memory Space长期记忆接口失败次数 | ≥0   | count | 1  | agentarts_memory_space_id | 1 分钟           |
| memory_avg_latency         | 长期记忆响应时延  | 当前 Memory Space长期记忆请求时延   | ≥0   | ms    | 1  | agentarts_memory_space_id | 1 分钟           |

| 指标ID         | 指标名称   | 指标含义                   | 取值范围 | 单位    | 进制 | 维度                        | 监控周期（原始指标） |
|--------------|--------|------------------------|------|-------|----|---------------------------|------------|
| memory_count | 长期记忆数量 | 当前Memory Space长期记忆生成条数 | ≥0   | count | 1  | agentarts_memory_space_id | 1 分钟       |

## 5.10.2 查看观测指标

### 操作场景

您可以通过以下指导，查看记忆库的观测指标。

具体可查看指定时间段内的记忆访问QPS、记忆访问流控、记忆访问失败、记忆响应时延、短期记忆访问QPS和短期记忆访问失败等使用数据信息。

### 查看监控指标

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3 在记忆库列表中单击已创建的记忆库名称。
- 步骤4 在“观测指标”页签，可以选择时间及统计周期。
  - 时间选择：可以选择查看最近15分钟、30分钟、1小时、3小时、12小时、1天、3天、7天的数据。
  - 统计周期：可以选择数据的统计周期，选择1分钟可以看到每分钟的QPS（查询每秒请求数）、响应时延等指标，快速发现系统在某一分钟内的异常情况；选择5分钟，可以看到每5分钟的QPS、响应时延等指标，了解系统在较长时间内的整体表现。
- 步骤5 查看指标。

图 5-11 查看指标



---结束

5.10.3 创建告警规则

操作场景

云监控服务（CES）提供使用告警模板创建告警规则的功能，当您需要监控云服务资源的使用情况或关键操作时，可以创建告警规则。告警规则添加完成后，当监控指标触发设定的阈值或者指定的事件发生时，云监控服务会在第一时间通过消息通知服务实时告知您云上资源异常，以免因此造成业务损失。

创建告警规则

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3 在记忆库列表中单击已创建的记忆库名称。
- 步骤4 在“观测指标”页签，在任一指标卡片上单击“+”，然后在弹框中单击“继续”，进入“创建告警规则”页面。

图 5-12 创建告警规则



**步骤5** 在“创建告警规则”界面，根据界面提示配置参数。

关键参数配置如下：

- 配置告警规则基本信息。
  - 名称：告警规则的名称。系统会随机自动生成一个名称，用户也可以自定义设置。
  - 描述：告警规则描述信息，非必填项。
- 配置告警内容参数。
  - 告警类型：默认为“指标”。
  - 云产品：默认为“SYS.AgentArts”。
  - 资源层级：监控对象的资源层级。当告警类型选择指标时，展示告警规则的资源层级，此处系统已默认，无需填写。
  - 监控范围：告警规则适用的资源范围，默认为“指定资源”。
  - 监控对象：系统已默认展示当前创建告警规则的记忆库ID，如需为多个记忆库添加告警规则，单击“重新选择”，选择多个记忆库。
  - 触发规则：默认为“自定义创建”。
  - 告警策略：触发告警规则的告警策略。

3. 配置告警通知参数。

如果要配置通过邮件、短信、HTTP和HTTPS向用户发送告警通知，则设置“发送通知”为开启。

 **说明**

“告警通知”功能触发产生的告警消息由消息通知服务SMN发送，可能产生少量费用，具体费用请参考[产品价格说明](#)。

更多配置参数信息，请参见[创建告警规则的通知](#)。

4. 配置完成后，单击“立即创建”，完成告警规则的创建。

----结束

## 5.11 管理记忆库

### 5.11.1 查看记忆库列表与详情

在控制台查看已创建的记忆库列表和详情信息，获取记忆库ID、API Key等集成所需的关键参数。

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。

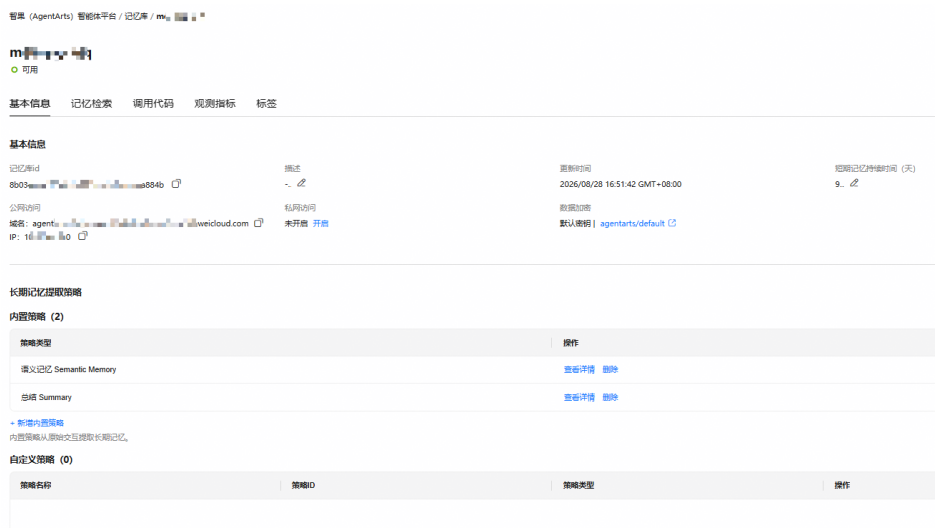
**步骤3** 查看记忆库列表。

记忆库 (10)

| 记忆库名称ID  | 状态 | 记忆库ID | 近期记忆库更新时间 (天) | 创建时间                          | 描述          | 标签 | 操作                                      |
|----------|----|-------|---------------|-------------------------------|-------------|----|-----------------------------------------|
| memory-1 | 可用 | 2     | 90            | 2026/09/28 10:51:42 GMT+08:00 |             |    | <a href="#">查看详情</a> <a href="#">删除</a> |
| memory-2 | 可用 | 1     | 7             | 2026/09/28 10:50:30 GMT+08:00 | Test memory |    | <a href="#">查看详情</a> <a href="#">删除</a> |
| memory-3 | 可用 | 1     | 7             | 2026/09/28 10:44:47 GMT+08:00 | Test memory |    | <a href="#">查看详情</a> <a href="#">删除</a> |
| memory-4 | 可用 | 4     | 90            | 2026/09/27 10:08:15 GMT+08:00 |             |    | <a href="#">查看详情</a> <a href="#">删除</a> |

步骤4 单击记忆库名称，查看记忆库详情。

图 5-13 查看记忆库详情



----结束

5.11.2 管理记忆策略

记忆策略可以在创建记忆库后进行管理，包括新增、删除策略。

新增内置策略

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3 单击记忆库名称，查看记忆库详情。
- 步骤4 在“基本信息”页签，在“长期记忆提取策略”区域，单击“新增内置策略”。
- 步骤5 勾选对应的策略后单击“确定”。

----结束

新增自定义策略

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3 单击记忆库名称，查看记忆库详情。
- 步骤4 在“基本信息”页签，在“长期记忆提取策略”区域，单击“新增自定义策略”，通过基础模型和提示模板自定义记忆处理。
- 步骤5 配置策略信息后单击“确定”。
  - 策略名称：自定义策略名称，策略名称不可重复。命名规则：包含a-z、A-Z、0-9和（中划线）等有效字符，最多可包含 48 个字符。
  - 策略类型：选择策略类型，然后配置抽取（Extraction）、整合（Consolidation）和

反思（Reflection）的提示词模板。请根据页面展示的实际内容配置。

- 抽取 Extraction：从大量数据中提取关键信息和重要特征，可以使用默认用户提示或创建自定义提示。
- 整合 Consolidation：将提取出的信息进行汇总、合并和优化，确保信息的完整性和一致性，可以使用默认用户提示或创建自定义提示。
- 反思 Reflection：系统对过去的经验、行为或数据进行分析和总结，以提取有价值的信息和洞察的过程。

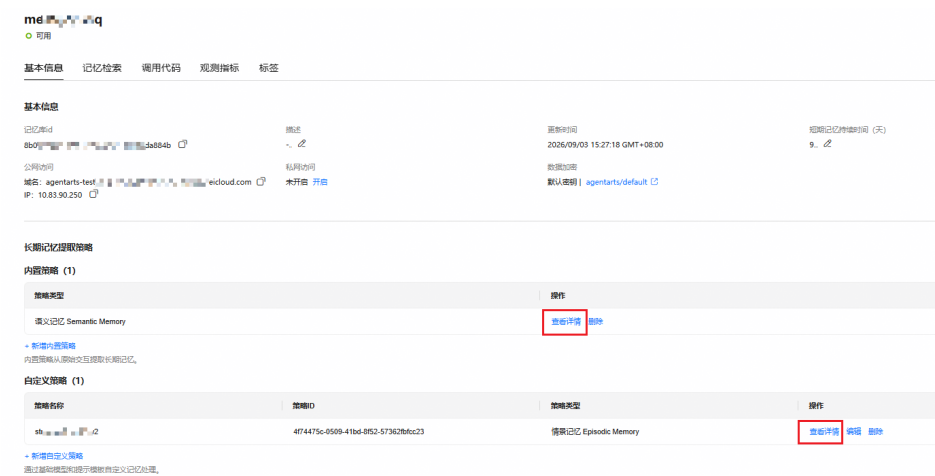
**步骤6** 如需编辑修改自定义策略，单击自定义策略列表操作列的“编辑”，修改自定义策略信息。

----结束

查看策略详情

- 步骤1** 登录AgentArts智能体平台。
- 步骤2** 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3** 单击记忆库名称，查看记忆库详情。
- 步骤4** 在“基本信息”页签，在“长期记忆提取策略”区域，单击内置策略或自定义策略列表操作列的“查看详情”。
- 步骤5** 查看策略详细信息。

图 5-14 查看内置策略详细信息



----结束

删除策略

删除策略时，至少需保留一个自定义策略或内置策略，否则无法删除。

- 步骤1** 登录AgentArts智能体平台。
- 步骤2** 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3** 单击记忆库名称，查看记忆库详情。
- 步骤4** 在“基本信息”页签，在“长期记忆提取策略”区域，单击内置策略自定义策略列表操作列的“删除”，弹框中单击“确定”。

#### 说明

删除策略后，该策略已提取的长期记忆记录不会被自动删除，但后续不会再有新的该类型记忆被提取。

----结束

## 5.11.3 更新 API Key

您可以通过更新操作重新生成API Key，更新后旧API Key立即失效，确保记忆库的访问安全。

#### 说明

更新API Key操作仅适用于使用**API Key 认证**的记忆库。如果记忆库已升级为**IAM 认证**，则不再支持更新或重置API Key。如需将API Key认证升级为IAM认证，请参见[认证方式升级](#)。

### 更新 API Key

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。

**步骤3** 在记忆库列表中单击记忆库名称，进入基本信息界面。

**步骤4** 在“API Key”处单击“更新”。

**步骤5** 在更新API Key弹框中单击“确定”后，复制或下载新的 API Key。

#### 说明

- 更新后旧API Key立即失效。
- 新API Key只显示一次，请妥善保存。
- 更新后需要同步修改所有使用该记忆库的Agent代码中的API Key环境变量，否则会导致认证失败。

----结束

## 5.11.4 管理标签

通过为记忆库添加、编辑和删除标签，您可以对记忆库进行标识和分类，便于在TMS中统一管理和检索云资源。标签由“标签键”和“标签值”组成，值可以为空，最多可添加 20 个标签。

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。

**步骤3** 在记忆库列表中可通过标签过滤查询记忆库。

图 5-15 通过标签过滤查询记忆库



- 步骤4** 单击已创建的记忆库名称，进入记忆库详情页面。
- 步骤5** 在“标签”页签，查看已添加的标签，单击“编辑标签”，可对标签进行管理（最多支持添加20个标签）。
- 添加标签：单击“添加标签”，可选择TMS中的预定义标签，或自定义输入标签键与值。
  - 修改标签：直接修改已有标签内容即可。
  - 删除标签：单击目标标签后的“删除”即可。

----结束

5.11.5 删除记忆库

记忆库删除后不可恢复，请谨慎操作。

- 步骤1** 登录AgentArts智能体平台。
- 步骤2** 在左侧导航栏选择“托管与运行 > 记忆库”，进入记忆库界面。
- 步骤3** 在记忆库列表中，单击操作列的“删除”。
- 步骤4** 在弹框中确认要删除后输入“DELETE”。
- 步骤5** 单击“确定”。

----结束



# 6 网关

## 6.1 网关概述

网关是Agent与外部工具之间的统一接入层。Agent通过网关发现和调用外部工具，不需要关心后端工具的协议差异、认证方式或调用格式。

网关同时支持两类能力：

- 工具调用：**Agent通过网关发现和调用外部工具（MCP 协议），将 REST API、MCP Server、APIG、云服务等自动转换为 Agent 可调用的标准工具。
- 推理路由：**Agent通过网关统一调用多个LLM提供商（推理协议），集中管理模型端点和认证。

### 核心概念

- 网关（Gateway）：**对Agent暴露的统一入口。一个网关可以挂载多个Target，Agent通过网关发现和调用所有Target暴露的工具。
- Target：**网关后面挂载的后端目标（MCP服务、REST API、推理服务等）。一个Target可以暴露一个或多个工具，网关根据Target的协议类型自动路由请求。
- 工具（Tool）：**Target暴露的具体能力，Agent通过工具名称和参数调用。工具信息包括名称（name）、描述（description）和参数定义（inputSchema）。

### 核心能力

| 能力分类      | 能力项      | 简要说明                                       |
|-----------|----------|--------------------------------------------|
| 智能体接入（入站） | 入站认证     | 支持IAM认证、OAuth 2.0认证、API Key认证三种方式，验证调用方身份。 |
|           | 工具自动发现   | Agent通过tools/list自动获取全部工具，无需手动配置工具列表。      |
|           | MCP标准交互  | Agent基于MCP协议完成工具调用，屏蔽后端实现细节。               |
| 网关中转路由    | 异构协议自动转换 | REST、APIG、云服务等后端自动转为MCP标准工具。               |

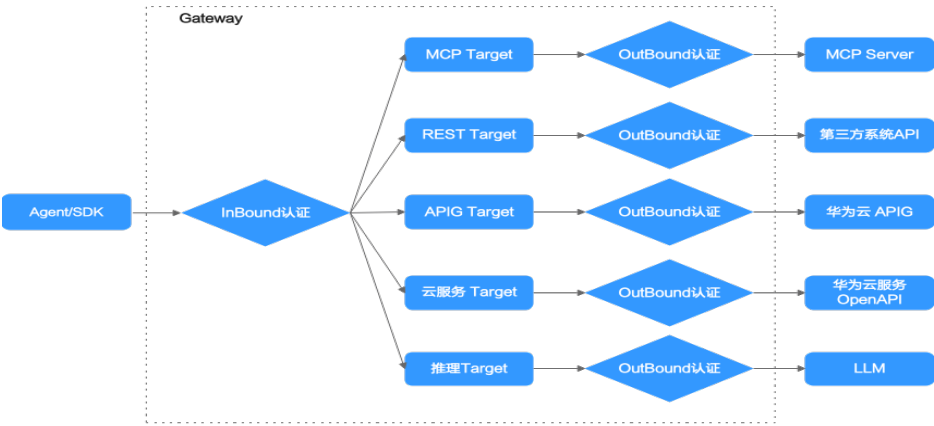
| 能力分类       | 能力项         | 简要说明                                                      |
|------------|-------------|-----------------------------------------------------------|
|            | 多Target聚合管理 | 单网关挂载多个Target，对外聚合输出全部工具集合。                               |
|            | 推理路由        | 推理Target对接多家LLM，网关托管大模型服务商凭证。                             |
| Target出站安全 | 出站凭证托管      | 网关保存后端鉴权信息，Agent代码不泄露后端密钥，支持 IAM认证、OAuth 2.0认证、API Key认证。 |
|            | 协议请求转发      | MCP请求和后端协议双向转换，完成工具调用转发。                                  |
| 运维可观测      | 网关调测        | 支持连通性、工具列表、工具调用的在线调测验证。                                   |
|            | 日志可观测       | 集成华为云云日志服务（LTS），实时查看网关日志。                                 |

6.2 工作原理

网关在Agent与外部系统之间承担统一接入层的角色，根据请求类型自动路由到对应的后端服务。

整体架构

图 6-1 网关原理图



网关的使用原理如下：

- 请求发起  
Agent/SDK：应用程序或客户端通过Agent或SDK发起请求，并将请求发送给Gateway（网关）。
- InBound认证：Gateway接收到请求后，首先进行InBound认证，确保请求的合法性。只有通过认证的请求才能继续进行处理。

3. 请求分发与处理
- MCP Target: 对接标准MCP Server后端，走原生MCP协议。

- REST Target: 对接普通第三方HTTP REST API，把MCP协议调用转成普通REST http请求。

- APIG Target: 对接华为云APIG，把MCP协议调用转成注册在APIG上的REST http请求。

- 云服务 Target: 对接华为云各个云服务 OpenAPI，把MCP协议调用转成华为云服务开放的REST http请求。

- 推理Target: 对接外部 LLM 推理服务，通过 /inference/ 路径路由推理请求到模型提供商。
4. OutBound 认证：网关访问后端服务做的下游认证，支持的认证方式（如API Key、OAuth、IAM等）因Target类型而异。

工具调用模式（MCP 协议）

Agent通过网关发现和调用外部工具：

1. Agent向网关发送tools/list 请求。
2. 网关汇总所有MCP Target暴露的工具，返回统一的工具列表。
3. Agent根据工具描述选择目标工具。
4. Agent向网关发送tools/call请求。
5. 网关路由到对应的Target执行。
6. Target返回结果，网关转发给Agent。

推理路由模式（Inference 协议）

Agent 通过网关统一调用LLM推理服务：

1. Agent向网关的 /inference/ 路径发送推理请求（如 /inference/v1/chat/completions）。
2. 网关根据模型映射确定目标提供商。
3. 网关附加出站认证信息（API Key等）。
4. 网关将请求转发到对应的Inference Target。
5. 提供商返回推理结果，网关转发给 Agent。

认证与鉴权机制

网关涉及入站和出站认证：

| 方向   | 说明                                                                 |
|------|--------------------------------------------------------------------|
| 入站认证 | 验证“谁在调用网关”。Agent 或应用向网关发请求时，网关验证调用方身份（IAM /OAuth 2.0/API Key）。     |
| 出站认证 | 网关“代表调用方访问后端”。网关向Target后端转发请求时，自动附加后端服务所需的认证信息（API Key/OAuth/IAM）。 |

- 入站认证在创建网关时配置。
- 出站认证在创建Target时配置。
- 入站与出站的配置互不影响：入站用IAM认证，出站可以用API Key，完全独立。

## 6.3 示例：集成高德 MCP 服务实现旅行攻略制定

本示例介绍如何通过集成高德MCP服务来实现旅行攻略的制定。

### 集成高德的 MCP 服务实现旅行攻略制定

**步骤1** 在高德开放平台创建并获取API Key。

高德MCP如下，使用前，请参考[高德开放平台文档](#)，创建获取API Key。

```
{
 "mcpServers": {
 "amap-maps-streamableHTTP": {
 "url": "https://mcp.amap.com/mcp?key=您在高德官网上申请的key"
 }
 }
}
```

**步骤2** 在AgentArts平台创建网关。

网关创建完成后，您可以通过控制台调测网关，具体请参考[步骤3](#)，也可以在Python客户端通过调用代码调测网关，具体请参考[步骤4](#)。

1. 在AgentArts平台左侧导航栏中选择“托管与运行 > 网关”，进入“网关”页面，单击“创建网关”。
2. 填写网关基础信息并配置权限与身份认证。参考下表进行配置。

表 6-1 网关基础信息与权限身份认证配置

| 参数        | 说明                   |
|-----------|----------------------|
| 名称        | 可自定义。                |
| 描述（可选）    | 网关的描述信息，可自定义。        |
| MCP 版本    | 2025-03-26           |
| 委托        | 使用平台的默认值。            |
| 入网配置      | 使用默认网关“defaultgw”即可。 |
| 入站身份认证    | 选择API Key认证。         |
| API Key名称 | 可自定义。                |
| 日志记录      | 选择开启。                |

图 6-2 填写网关基础信息与权限身份认证

基本信息

名称

gateway-gaodemap

描述 (可选)

请输入描述内容

MCP 版本

2025-03-26 X

请选择网关需要支持的MCP版本。创建网关后该版本无法修改。[查看详情](#)

权限与访问控制

委托

AgentArtsGatewayAgency

创建委托

为保证网关正常运行，所选委托需具备相关权限。[查看所需权限查看当前权限](#)

入网配置

defaultgw

前往入网配置

入站身份认证

IAM 认证

将华为云 IAM 身份配置为入站身份认证，以验证调用者的身份。

OAuth 2.0 认证

将JWT（如OAuth令牌）配置为入站身份认证，以验证传入的令牌签名和范围。

API Key 认证

将预设的密钥对（键和值）配置为入站身份认证，以验证请求中携带的密钥合法性。

身份认证凭证在凭据管理服务（DEW）托管，托管按需计费。[了解 DEW 计费详情](#)

API Key 名称

APIKey-225419862

3. 单击“创建Target”，配置高德MCP服务。

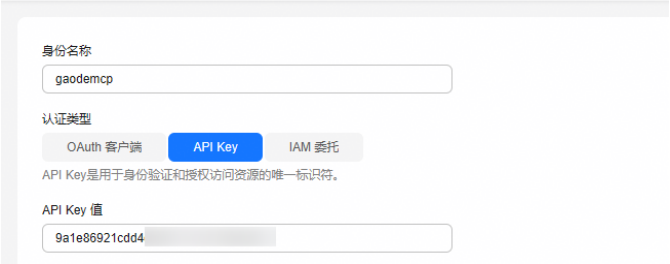
表 6-2 高德 Target 配置

| 参数       | 说明                                           |
|----------|----------------------------------------------|
| 名称       | 可自定义，例如“gaodemap”。                           |
| 描述（可选）   | 可自定义。                                        |
| Target协议 | 选择“MCP”协议。                                   |
| 类型       | 选择“MCP”，使用现成的高德MCP服务，不需要使用直接对接外部REST API的方式。 |
| 传输方式     | 选择“Streamable HTTP”，与高德MCP的调用方式保持一致。         |
| MCP地址    | 填写：https://mcp.amap.com/mcp                  |

文档版本 01 (2026-09-24)

版权所有 © 华为云计算技术有限公司

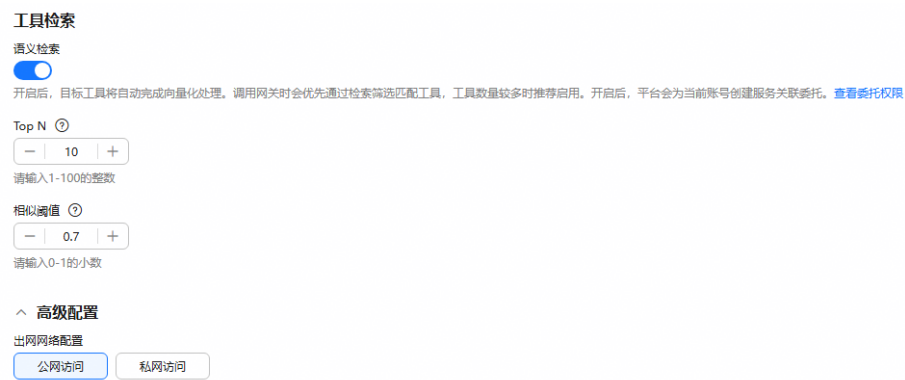
209

| 参数     | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 出站身份认证 | <p>选择“API Key”，单击“创建出站身份”，参数配置如下：</p> <ul style="list-style-type: none"><li>身份名称：可填写为“gaodemcp”。</li><li>认证类型：选择“API Key”。</li><li>API Key的值：填写高德开发平台中创建的API Key。</li></ul> <p>出站身份创建完成后，返回“创建Target”页面，选择已创建的出站身份。</p> <ul style="list-style-type: none"><li>位置：选择“查询参数”。</li><li>参数名称：填写为key。</li><li>前缀：不填。</li></ul> <p><b>图 6-3 创建出站身份</b></p> <p>Agent Identity / 出站身份 / 创建出站身份</p> <p>&lt; 创建出站身份</p>  <p><b>图 6-4 配置出站身份认证</b></p>  |

4. 配置完成后，单击“确定”创建Target。  
Target创建完成后，回到“创建网关”页面。
- a. 开启“语义检索”，其他使用默认即可。

- b. 高级配置选择“公网访问”，开启会话保持。

图 6-5 配置语义检索



5. 单击“创建网关”。

步骤3 通过控制台调测网关。

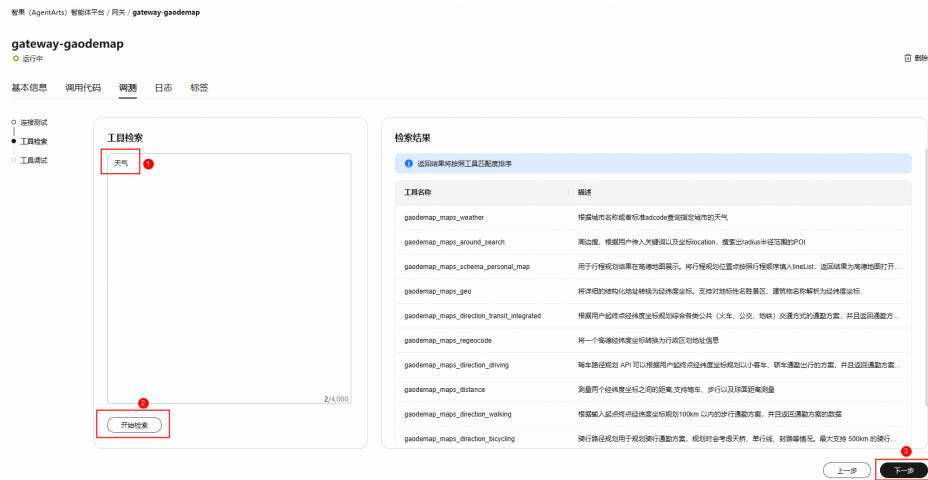
1. 在网关列表中单击已创建的网关名称。
2. 在“调测”页签，单击“连接测试”，待连接成功后单击“下一步”。

图 6-6 调测网关



3. 在“工具检索”页面，检索框中输入“天气”，单击“开始检索”，然后单击“下一步”。

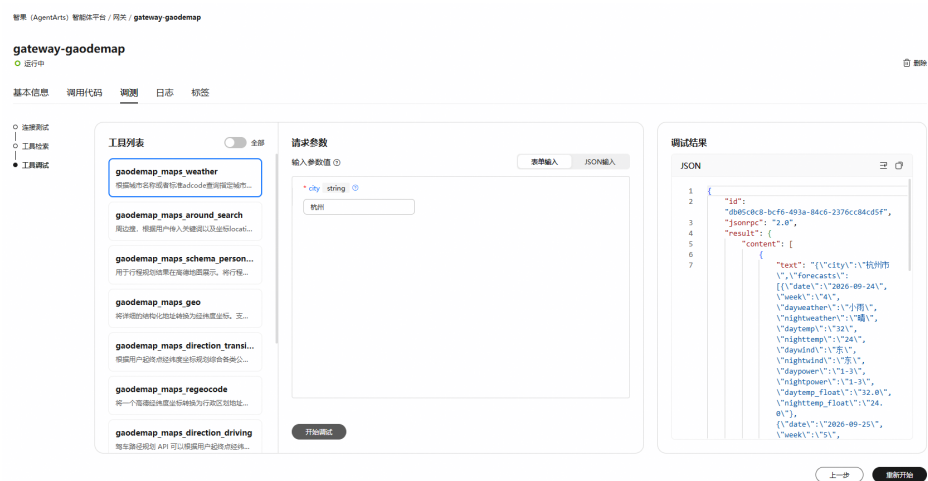
图 6-7 工具检索



4. 进入“工具调试”页面，选择“gaodemap\_maps\_weather”，参考图6-8设置请求参数，单击“开始调试”，即可查看到调试结果。

在工具列表中关于各工具的介绍请参考[高德开放平台MCP Server介绍](#)。

图 6-8 查询天气



步骤4 通过调用代码调测网关。

1. 网关创建完成后，回到网关列表页面，单击网关名称，进入网关详情页面。

图 6-9 网关列表



2. 在网关详情页面获取网关URL（即MCP调用URL）。



图 6-10 获取网关 URL

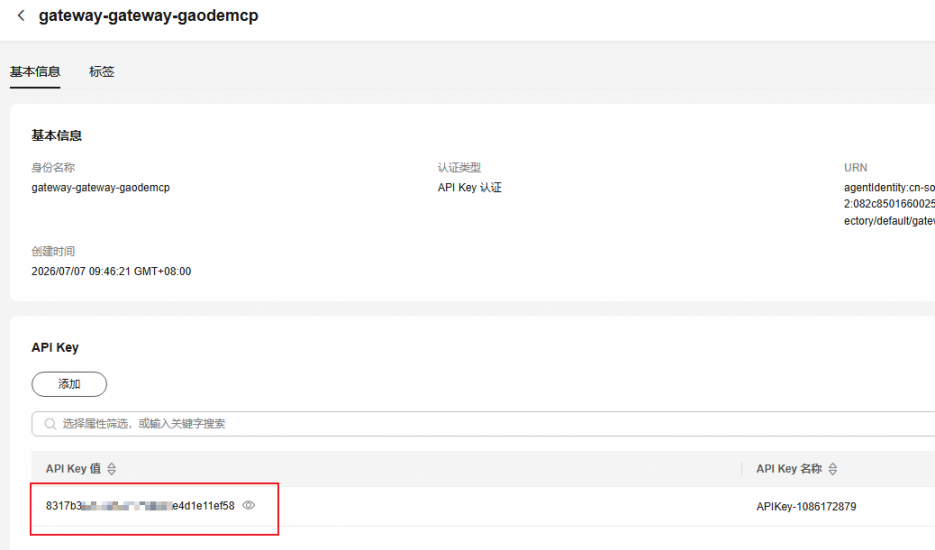


3. 在网关详情页面单击URN链接。进入Agent Identity页面，获取网关的API Key。

图 6-11 单击 URN 链接



图 6-12 获取网关 API Key



4. 参考如下代码示例，将API Key及gateway\_url替换为上述步骤中获取的值，并配置MCP\_SESSION\_ID，配置后在Python客户端运行。

```
import requests
import json
import re
def call_tool(gateway_url, access_token, tool_name, arguments):
 headers = {
 "Content-Type": "application/json",
 "Accept": "application/json, text/event-stream",
 "mcp-session-id": "<MCP_SESSION_ID>", # MCP会话ID，相同的sessionid会使用同一网关资源，调
```

```

用时请配置此参数，例如：4ada96ff-1e16-4e5b-8fb4-b3f0f207a4f3
"Mcp-Protocol-Version": "2025-03-26", # MCP协议版本
"Authorization": f"Bearer {access_token}"
}
payload = {"jsonrpc": "2.0", "id": "call-tool-request", "method": "tools/call", "params": {"name":
tool_name, "arguments": arguments}}
response = requests.post(gateway_url, headers=headers, json=payload) # 若需要忽略ssl校验，可补
充参数verify=False
raw = response.content.decode('utf-8', errors='replace')
ct = response.headers.get('content-type', '')
if 'text/event-stream' in ct:
 for line in raw.strip().split('\n'):
 if line.startswith('data:'):
 try:
 return json.loads(line[5:].strip())
 except json.JSONDecodeError:
 continue
 m = re.search(r'\{.*\}', raw, re.DOTALL)
 if m:
 return json.loads(m.group())
try:
 return json.loads(raw)
except json.JSONDecodeError:
 print(f"[Fail]: {raw[:200]}")
 return None
gateway_url = "<gateway_url>" # 网关接口地址（业务API调用入口）
access_token = "<API_KEY>" # 认证凭证：APIKey认证直接填写密钥；无认证则留空/删除
tool_name = "x_agentarts_gateway_tool_semantic_search" # 语义检索调用的工具名称，固定为
"x_agentarts_gateway_tool_semantic_search"
arguments = {"query": "查询杭州的天气"} # 语义检索调用入参
result = call_tool(gateway_url, access_token, tool_name, arguments)
if result:
 print(json.dumps(result, indent=2, ensure_ascii=False))

```

关于其他的调用方法示例可参考[调测与调用MCP Target](#)。

## 5. 查看调试结果。

图 6-13 调试结果

```

{
 "jsonrpc": "2.0",
 "id": "call-tool-request",
 "result": {
 "content": [
 {
 "type": "text",
 "text": "{\n \"tools\": [\n {\n \"name\": \"target-gaode_maps_weather\", \"description\": \"根据城市名称或者标准adcode查询指\n定城市的天气\", \"inputSchema\": {\n \"type\": \"object\", \"properties\": {\n \"city\": {\n \"description\": \"城市名称或者adcode\", \"t\nype\": \"string\", \"required\": [\"city\"]\n }\n }\n }\n }\n],\n \"isError\": false\n}"
]
],
 "isError": false
 }
}

```

----结束

## 6.4 创建网关

网关作为智能体与外部系统之间的通信中介，不仅负责双向通信和协议转换，还提供了日志记录的功能，确保系统的安全、高效和可扩展性。

创建网关有两种方式：通过控制台创建和通过SDK创建。如果您选择使用SDK创建网关，可以参考[Gateway SDK](#)，本章节主要介绍通过控制台创建网关。

### 前提条件

- 使用网关前需要先[开通AgentArts服务](#)并进行服务授权，请在“授权管理”页面进行云服务开通及授权，详细信息请参考[授权管理](#)。

- （可选）管理员可在IAM控制台为创建的IAM子用户授予权限，授权后，子用户即可根据权限使用账号中的云服务资源，相关操作步骤如下。
  - [创建IAM用户](#)。
  - [给IAM用户授权](#)。
    - 为子用户授予完整的网关权限，需授予AgentArtsFullAccessPolicy身份策略。
    - [创建自定义身份策略](#)并授予如下权限，授权项功能介绍请参考[表6-3](#)。

表 6-3 授权项

| 操作          | 授权项                                           | 功能介绍                     |
|-------------|-----------------------------------------------|--------------------------|
| 创建网关        | iam:agencies:pass                             | 授予向云服务传递委托的权限。           |
|             | vpc:nativePorts:create                        | 授予原生API创建端口权限。           |
|             | vpc:routeTables:update                        | 授予更新路由表权限。               |
|             | eip:publicIps:associateInstance（涉及公网访问时需要）    | 授予将弹性公网IP绑定网卡的权限。        |
| 删除网关        | vpc:routeTables:update                        | 授予更新路由表权限。               |
|             | vpc:nativePorts:delete                        | 授予原生API删除端口权限。           |
|             | eip:publicIps:disassociateInstance（涉及公网访问时需要） | 授予将弹性公网IP解绑网卡的权限。        |
| 创建网关的入站认证功能 | csms:secret:getVersion                        | 授予查询指定凭据版本的信息和其明文凭据值的权限。 |
|             | kms:cmk:decryptDataKey                        | 授予解密数据密钥的权限。             |
| 出站网络使用VPC网络 | vpcep:endpoints:create                        | 授予指定服务创建VPC终端节点的权限。      |
|             | vpcep:endpoints:delete                        | 授予删除终端节点的权限。             |

- 创建网关时需要[创建委托并授权](#)，请确保传入的委托包含以下身份策略权限：

表 6-4 授权项

| 授权项                                   | 功能介绍                                  |
|---------------------------------------|---------------------------------------|
| csms:secret:getVersion                | 授予查询指定凭据版本的信息和其明文凭据值的权限。              |
| kms:cmk:decryptDataKey                | 授予解密数据密钥的权限。                          |
| agentIdentity::getResourceApiKey      | 授予获取与API密钥凭证提供者关联的API密钥的权限。           |
| agentIdentity::getResourceOauth2Token | 授予通过OAuth2两方/三方授权流程获取访问令牌，以访问外部资源的权限。 |
| agentIdentity::getResourceStsToken    | 授予从STS凭证提供者获取IAM临时凭证的权限。              |

约束与限制

一个租户最多可创建10个网关，如需调整配额，请[提交工单](#)申请。

创建网关

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 网关”，进入网关界面。
- 步骤3 单击右上角“创建网关”。
- 步骤4 在“创建网关”的页面中，输入网关的配置信息，参数说明请参考[表6-5](#)。

表 6-5 参数说明

| 参数   |        | 说明                                                                                                                                                                     |
|------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 基本信息 | 名称     | 网关的名称，同一账号下网关名称不可重复。<br>命名规则： <ul style="list-style-type: none"><li>以小写字母开头，小写字母或数字结尾。</li><li>支持小写字母、数字和中划线。</li><li>长度为 2-40 个字符。</li></ul> 系统会默认生成网关的名称，可根据需求自定义修改。 |
|      | 描述     | 用于对网关内容和用途的简要说明。<br>规则：长度不大于1000个字符。                                                                                                                                   |
|      | MCP 版本 | 支持配置MCP协议版本，可以选择一个或多个支持的MCP协议版本（如2025-03-26、2025-06-18、2025-11-25）。<br>关于协议版本的介绍、配置及相关约束等可参考 <a href="#">MCP协议版本介绍</a> 。                                               |

| 参数      |        | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 权限与身份认证 | 委托     | <p>授予的代理权限或代理功能，允许代表智能体与外部系统进行通信和交互。</p> <p>可使用系统默认创建的AgentArtsGatewayAgency委托。</p> <p>如需手动创建，请单击“创建委托”并参考如下配置创建。</p> <ul style="list-style-type: none"><li>委托名称：自定义。</li><li>信任主体类型：选择“云服务”。</li><li>云服务：搜索service.WorkloadSandboxMetadata。</li><li>其余参数使用默认值。</li></ul> <p><b>图 6-14 创建委托</b></p>  <p>委托创建完成后可对委托进行授权，操作具体详情请参考<a href="#">创建委托并授权</a>。</p> |
|         | 入网配置   | <p>选择入站网关，通过互联网或者VPC内网访问网关服务。如下拉框无选项，请单击“<a href="#">创建网络配置</a>”创建新的网络配置。</p> <p>选择后修改网关时入网配置无法修改。</p> <p><b>说明</b></p> <p>入网配置关联VPC内网时，则不支持调测以及语义检索功能。</p>                                                                                                                                                                                                                                                                                      |
|         | 入站身份认证 | <p>配置网关的身份认证方式。入站身份认证是指网关在接收和处理来自智能体或SDK的请求时，对请求进行身份验证，确保只有经过验证的请求才能进入网关，提高系统的安全性和可靠性。</p> <p>支持以下认证方式：</p> <ul style="list-style-type: none"><li>IAM认证：使用登录管理控制台时使用的IAM用户名进行认证。</li><li>OAuth 2.0认证：将OAuth 2.0配置为入站身份认证。选择“OAuth 2.0认证”后，相关参数配置请参考<a href="#">表6-6</a>。</li><li>API Key认证：通过接口的访问密钥进行身份验证和授权。系统会自动生成API Key的名称，可根据需求自定义修改。身份认证凭证在凭据管理服务（DEW）托管，托管按需计费，费用说明请参考<a href="#">DEW 计费说明</a>。</li></ul>                                |

| 参数       |          | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 可观测配置    | 日志记录     | <ul style="list-style-type: none"><li>未开启，网关使用过程产生的日志无法上报至云日志服务。</li><li>开启后，网关使用过程产生的日志会上报云日志服务（LTS），日志管理费用按需收取，有关计费相关内容请参考<a href="#">LTS计费详情</a>。</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 关联Target | 创建Target | <p>单击“创建Target”按钮完成Target创建后，网关将与新创建的Target关联。</p> <p>创建Target详情请参考<a href="#">创建Target</a>。</p> <p>同一个网关中可以同时挂载MCP和推理两种Target，请求各自走独立路径，互不干扰。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 工具检索     | 语义检索     | <p>语义检索是一种基于自然语言处理的技术，用于从大量文本数据中提取和检索相关信息。在网关中启用语义检索功能后，调用网关时会优先通过检索筛选匹配工具，工具数量较多时建议开启。</p> <p>“语义检索”仅针对MCP协议的Target。</p> <p>开启后，将为您自动创建服务关联委托如<a href="#">图6-15</a>，委托中的授权项包含以下action：</p> <p>agentIdentity::getAuthorizerConfiguration：授予查询工作负载身份授权配置的权限。</p> <p><b>图 6-15 服务关联委托</b></p>  <p><b>Top N：</b>用于设置返回的匹配结果数量，从所有工具中选择匹配度排名前N条返回。可以输入1到100之间的整数，表示系统在匹配时返回的前N个最相关的工具或结果。默认值为10，您可以根据需要增加或减少这个数值。数值越大返回越多，但可能包含相关度较低的内容。</p> <p><b>相似阈值：</b>用于设置匹配结果的相似度阈值。可以输入0到0.9之间的数值（精确到小数点后一位），表示系统在匹配时返回的结果与查询内容的相似度必须达到或超过这个阈值。默认值为0.7，您可以根据需要调整这个数值。数值越接近1，表示匹配结果的相似度要求越高。值越高结果越精准，但可能返回更少甚至为空。建议配合Top N数值一起调整。</p> <p>关于语义检索的tools/call的调用示例请参考<a href="#">检索网关工具</a>。</p> |

| 参数   |        | 说明                                                                                                                                                                                                                                                                                                                                                           |
|------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 高级配置 | 私网访问   | 连接内部的私有网络，访问内部的资源和服务。 <ul style="list-style-type: none"><li>选择已配置的VPC，如未配置，请单击“新建VPC”，具体请参考<a href="#">创建虚拟私有云和子网</a>。</li><li>选择已配置的子网，如未配置，请单击“新建子网”，具体请参考<a href="#">创建虚拟私有云和子网</a>。</li><li>选择已配置的安全组，如未配置，请单击“新建安全组”，具体请参考<a href="#">创建安全组</a>。</li></ul> 配置后可单击  刷新。 |
|      | 公网访问   | 能够连接到外部的互联网，访问外部的资源和服务。                                                                                                                                                                                                                                                                                                                                      |
|      | 会话保持   | 开启后，通过同一个mcp-session-id调用网关时，网关将与后端目标（Target）保持会话，用于后续多轮交互。<br>“会话保持”仅针对MCP协议的Target。<br>如果Target的名称、配置信息以及出站身份认证信息发生变更（除描述外），则网关和该Target的会话将会重建。                                                                                                                                                                                                            |
|      | 标签（可选） | 由“标签键”和“标签值”组成，用于标识和分类云资源。<br>您可以在TMS中创建预定义标签，用于使用同一标签标识多种云资源，创建后可在标签输入框下拉选择同一标签，具体操作请参考 <a href="#">创建预定义标签</a> ，创建后单击  刷新。<br>单击“添加标签”，选择在TMS中创建的预定义标签，或自定义输入标签键和标签值。<br>可添加一个或多个标签，最多添加20个标签。                                                                          |

表 6-6 OAuth 2.0 认证参数配置说明

| 参数            | 说明                                                                                                                                                                                        |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Discovery URL | OAuth发现服务器（OAuth discovery server）只支持公网权威的认证服务器，即服务端证书必须能够校验通过。<br>需输入身份提供商（例如Okta、Cognito等）提供的Discovery URL，该URL通常可在该提供商的文档中找到。输入以https://开头，/.well-known/openid-configuration结尾的有效 URL。 |
| JWT授权配置       |                                                                                                                                                                                           |
| 允许的受众         | 用于认证为OAuth 2.0指定的受众是否在Agent Identity运行中指定的受众匹配或为其子集。<br>支持添加100个受众。                                                                                                                       |
| 允许的客户端        | 用于认证为OAuth 2.0指定的客户端标识符是否被允许访问Agent Identity。<br>支持添加100个客户端。                                                                                                                             |



| 参数    | 说明                                             |
|-------|------------------------------------------------|
| 允许的范围 | 仅当令牌包含此处配置的至少一个必须范围时才允许访问自定义声明。<br>支持添加100个范围。 |
| 自定义声明 | 仅当令牌中的特定声明与预定义字符串值相匹配时才允许访问。<br>支持添加100个自定义声明。 |

**步骤5** 单击“确定”。即可在网关列表中展示已创建的网关，单击网关名称，查看网关详情。

----结束

MCP 协议版本介绍

MCP（模型上下文协议）是智能体连接外部工具与数据的通用开放标准。为保障协议兼容性，AgentArts在创建网关时支持配置一个或多个MCP协议版本。网关创建后，调用接口时网关会校验请求中携带的协议版本是否在已配置的支持列表中。

当前网关MCP支持的协议版本及官方文档如下：

- [2025-11-25](#)
- [2025-06-18](#)
- [2025-03-26](#)

📖 说明

- MCP协议版本的协商涉及网关调用方、网关和Target之间的版本协商。
- 网关调用方通过初始化请求与网关协商MCP协议版本。非初始化请求通过Mcp-Protocol-Version请求头传递协商版本；若传递的版本不在网关支持范围内，网关将拦截请求并报错（非初始化请求2025-03-26版本可不传，默认为2025-03-26版本）。
- 网关通过初始化请求与Target协商MCP协议版本，若Target版本高于网关支持的最新版本，将无法调用成功，建议根据Target支持的协议版本合理配置网关版本。

约束与限制

1. 方法调用限制  
当前网关仅支持以下四种MCP方法调用：  
initialize、tools/list、tools/call、ping
2. 批量请求限制  
不支持 JSON-RPC 批量请求与响应（JSONRPCBatchRequest / JSONRPCBatchResponse）。
3. 部分方法下的扩展能力暂不支持：
  - tools/call 方法：不支持 tasks 能力，无法通过tools/call方法调用包含任务列表的工具。
  - tools/list 方法：不支持 tools.annotations（注解信息）、outputSchema（输出模式）、\$schema（JSON Schema）、icons（图标）能力。
4. 响应格式限制



调用网关接口的响应统一以text/event-stream方式返回。

### MCP网关协议版本配置示例

#### 步骤1 创建网关。

1. 在AgentArts平台左侧导航栏中选择“托管与运行 > 网关”，进入“网关”页面，单击“创建网关”。
2. 参考[创建网关](#)，创建“MCP 版本”为“2025-06-18”版本的网关。

图 6-16 填写网关基础信息与权限身份认证

基本信息

名称  
gateway-7ivtyq77

描述 (可选)  
请输入描述内容

MCP 版本  
2025-06-18 X

请选择网关需要支持的MCP版本，创建网关后该版本无法修改。查看详情

权限与访问控制

委托  
AgentArtsGatewayAgency 创建委托

为保证网关正常运行，所选委托需具备相关权限。查看所需权限查看当前权限

入网配置  
defaultgw 前往入网配置

入站身份认证

IAM 认证  
将华为云 IAM 身份配置为入站身份认证，以验证调用者的身份。

OAuth 2.0 认证  
将JWT（如OAuth令牌）配置为入站身份认证，以验证传入的令牌签名和范围。

API Key 认证  
将预设的密钥对（键和值）配置为入站身份认证，以验证请求中携带的密钥合法性。

#### 步骤2 创建Target。

在“关联Target”区域单击“创建Target”，参考[创建MCP Target](#)，分别创建“MCP”类型的TargetA、TargetB、TargetC。Target的MCP协议版本取决于其对应后端服务实际支持的协议版本，本示例中假设各Target的协议版本如下：

- TargetA的MCP协议版本为2025-03-26。
- TargetB的MCP协议版本为2025-06-18。
- TargetC的MCP协议版本为2025-11-25。

#### 步骤3 调用网关。

具体的调用操作请参考[调测与调用MCP Target](#)，当前示例网关支持的MCP版本为2025-06-18，以下为调用网关的场景示例：

- **场景一：**  
当调用方使用2025-03-26版本向网关发起initialize请求，会返回网关支持的协议版本2025-06-18，调用成功。
- **场景二：**  
当调用方使用2025-03-26版本向网关发起tools/list请求，因2025-03-26版本不在网关支持范围内，网关将拦截请求并报错。
- **场景三：**

当调用方使用2025-06-18版本向网关发起tools/list请求，网关调用成功。TargetA和TargetB的工具可正常返回，但由于TargetC版本高于网关支持的最高版本，故无法获取TargetC中的工具。

- **场景四：**  
当调用方使用2025-06-18版本向网关发起TargetA的tools/call请求，TargetA的版本低于网关支持的最高版本，调用成功。
- **场景五：**  
当调用方使用2025-06-18版本向网关发起TargetC的tools/call请求，因TargetC版本高于网关支持的最高版本，网关无法适配，调用失败并报错。

----结束

## 6.5 创建 Target

### 6.5.1 Target 介绍

一个网关可以挂载多个Target，每个Target对应一个后端服务。网关根据Target的协议类型自动路由请求。

#### Target 协议

支持创建MCP及Inference协议的Target。

| Target 协议 | 说明                                                                                      |
|-----------|-----------------------------------------------------------------------------------------|
| MCP       | 接入工具服务。Target可以定义为MCP服务、REST API、APIG 网关、云服务。Agent 通过 tools/list 发现工具，通过 tools/call 调用。 |
| 推理        | 接入 LLM 推理服务。将推理请求（如 /v1/chat/completions）路由到外部模型提供商，实现多模型统一接入和认证管理。                     |

两种协议使用独立的路由路径，互不干扰：

- MCP Target的请求通过/mcp路径路由
- Inference Target的请求通过/inference/路径路由

#### 如何选择

根据使用场景选择Target协议：

| 使用需求                                    | 选择  | 说明                                                 |
|-----------------------------------------|-----|----------------------------------------------------|
| 让Agent调用外部工具（REST API、MCP 服务、APIG、云服务等） | MCP | Agent通过tools/list发现工具，通过tools/call执行工具，网关自动完成协议转换。 |

| 使用需求                                  | 选择        | 说明                                                     |
|---------------------------------------|-----------|--------------------------------------------------------|
| 让Agent通过网关统一调用 LLM（OpenAI、DeepSeek 等） | Inference | 网关统一管理模型提供商的端点和认证，Agent 代码中无需硬编码 API Key。              |
| 既要调用工具，又要调用 LLM                       | 两者都选      | 同一个网关中可以同时挂载 MCP 和 Inference 两种 Target，请求各自走独立路径，互不干扰。 |

## 6.5.2 创建 MCP Target

创建网关时，可以创建并绑定Target。创建Target用于定义和配置一个目标端点或服务，网关将接收到的请求转发到该服务进行处理。

### 前提条件

- 已完成[创建网关](#)。
- 创建Target时，选择REST API类型的Target，且从OBS中获取文件时，用户除了需要具备网关的FullAccessPolicy权限外，还需包含以下身份策略权限：

表 6-7 授权项

| 授权项                           | 功能介绍              |
|-------------------------------|-------------------|
| obs:bucket:listBucket         | 授予获取桶内对象列表的权限。    |
| obs:bucket:listBucketVersions | 授予获取桶内多版本对象列表的权限。 |
| obs:bucket:listAllMyBuckets   | 授予查询创建的桶列表的权限。    |
| obs:object:getObject          | 授予下载非指定版本对象的权限。   |
| obs:object:getObjectVersion   | 授予下载指定版本对象的权限。    |

在IAM控制台为用户授权的操作指导可参考：[给IAM用户授权](#)。

### 约束与限制

一个网关最多可关联10个Target。

### 创建 Target

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 网关”，进入网关界面。
- 步骤3 选择已创建的网关，单击网关名称，在“关联Target”区域，单击“创建Target”。
- 步骤4 输入Target的配置信息，参数说明请参考[表6-8](#)，然后单击“确定”。

表 6-8 参数说明

| 参数       | 说明                                                                                                                                                                            |
|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名称       | Target的名称，同一网关下Target名称不可重复。<br>命名规则： <ul style="list-style-type: none"><li>以字母或数字开头和结尾。</li><li>支持大小写字母、数字和连字符。</li><li>长度为 1-50 个字符。</li></ul> 系统会默认生成Target的名称，可根据需求自定义修改。 |
| 描述       | 用于说明该Target的内容与用途。此描述仅在网关层生效，不会覆盖或影响后端原始工具的描述。<br>规则：长度不大于200个字符。                                                                                                             |
| Target协议 | 选择“MCP”协议。                                                                                                                                                                    |
| 类型       | 选择Target的类型，关于Target类型的介绍请参见 <a href="#">MCP协议支持的Target类型介绍</a> 。                                                                                                             |

| 参数   | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 配置信息 | <p>Target类型不同，对应的配置信息不用，请根据Target类型进行配置。</p> <ul style="list-style-type: none"><li>● REST API：使用OPEN API schema文件调用。<ul style="list-style-type: none"><li>- 使用“OBS”进行OAS文件导入。<br/>使用OpenAPI Specification (OAS) 文件来定义和配置目标端点或服务，并且这些文件存储在对象存储服务（Object Storage Service, OBS）中，并在创建Target时从OBS中导入这些文件。<br/>单击“从OBS中选择”，选择在OBS中创建的桶列表，直至选择到桶中的.json文件后单击“确定”。</li><li>- 使用“自定义”进行OAS文件导入。<br/>从本地文件系统中选择并上传OAS文件，用于定义和配置目标端点或服务。<ul style="list-style-type: none"><li>▪ 单击“从本地导入”，浏览本地文件系统，选择OAS文件并上传。上传后json文件或yaml即可展示在“内联编辑器”中。</li><li>▪ 单击“从OBS中导入”，选择在OBS中创建的桶列表，直至选择到桶中的.json文件后单击“确定”，将OBS中的文件下载下来展示到内联编辑器中。</li></ul></li></ul></li><li>● MCP：使用已有的MCP服务器调用。<ul style="list-style-type: none"><li>- 传输方式<ul style="list-style-type: none"><li>▪ SSE：基于HTTP的协议，用于从服务器向客户端推送实时更新，适用于需要实时更新的应用。</li><li>▪ Streamable HTTP：基于HTTP的流式传输方式，允许服务器向客户端发送连续的数据流，适用于需要实时数据流的应用。</li></ul></li><li>- MCP地址：输入MCP地址。</li></ul></li><li>● APIG：使用注册在云服务 API 网关（APIG）上的 API 作为目标服务，通过 APIG 来调用后端服务。<br/>从APIG导出API接口后，需对涉及body体的接口信息进行修改，具体示例请参见<a href="#">APIG类型的Target请求体配置示例</a>。<ul style="list-style-type: none"><li>- 使用“OBS”进行OAS文件导入。<br/>从 OBS 存储桶中导入 OpenAPI 规范文件（OAS 文件），需使用 Swagger 2.0 或 OpenAPI 3.0 格式的 YAML 或 JSON 文件来描述 API接口。<br/>单击“从OBS中选择”，选择在OBS中创建的桶列表，直至选择到桶中的.json文件后单击“确定”。</li><li>- 使用“自定义”进行OAS文件导入。<br/>从本地文件系统中选择并上传OAS文件，用于定义和配置目标端点或服务。<ul style="list-style-type: none"><li>▪ 单击“从OBS中导入”，选择在OBS中创建的桶列表，直至选择到桶中的.json文件后单击“确定”，将OBS中的文件下载下来展示到内联编辑器中。</li><li>▪ 单击“从本地导入”，浏览本地文件系统，选择OAS文件并上传。上传后json文件或yaml即可展示在“内联编辑器”中。</li></ul></li></ul></li></ul> |

| 参数 | 说明                                                                                                                                                                                                                                                                              |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <ul style="list-style-type: none"><li>云服务：使用华为云提供的云服务作为目标服务，通过华为云服务的开放接口来调用后端服务。</li></ul> <p><b>说明</b><br/>仅支持选择已在APIE上注册的API，如接口未注册则无法选择。</p> <ul style="list-style-type: none"><li>- 云服务：选择具体的华为云服务。</li><li>- 接口：选择了云服务后，系统会根据所选云服务自动加载可用的接口列表，从下拉菜单中选择需要调用的具体接口。</li></ul> |

| 参数     | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 出站身份认证 | <p>配置出站身份认证方式，用于在网关将请求转发到后端服务（Target）时，对请求进行身份验证和授权。不同类型的Target支持的出站认证方式不同，请根据页面展示配置。</p> <p>支持以下认证方式：</p> <ul style="list-style-type: none"><li>● API Key<ul style="list-style-type: none"><li>- 选择出站身份，如下拉框无选项请单击“创建出站身份”，创建“API Key”类型的出站身份。</li><li>- 选择位置。<ul style="list-style-type: none"><li>▪ 标头：在HTTP请求的头部字段中传递API Key。</li><li>▪ 查询参数：在HTTP请求的URL中传递API Key。</li></ul></li><li>- 参数名称（可选）：输入参数名称，在请求中传递API的密钥时使用的具体名称。</li><li>- 前缀（可选）：在请求头中传递API Key时，API密钥值之前的前缀部分，如在授权标头中使用的“Bearer”。</li></ul></li><li>● OAuth<ul style="list-style-type: none"><li>- 选择出站身份，如下拉框无选项请单击“创建出站身份”，创建“OAuth 客户端”类型的出站身份。</li><li>- 选择授权类型。<br/>客户端授权：适用于用户授权应用程序（客户端）访问其资源。</li><li>- 作用域：用于定义可以访问的资源 and 操作的范围，确保只能访问其被授权的资源。输入范围名称，最大可添加100个范围。</li><li>- 自定义参数：在请求中传递的额外参数，可以用于传递特定的信息或满足特定的认证需求。请输入自定义参数信息的键和值，最大可添加255个参数。</li></ul></li><li>● IAM<ul style="list-style-type: none"><li>- 选择出站身份，如下拉框无选项请单击“创建出站身份”，创建“IAM 委托”类型的出站身份，如果无IAM信任委托需<a href="#">创建信任委托</a>。</li></ul></li></ul> |

| 参数 | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | <div><div>图 6-17 创建信任委托</div><div><div>&lt; 创建信任委托 ②</div><div><div><div><div>① IAM新增信任委托，提供更灵活的基于信任策略的配置能力。信任委托只能授予身份策略且仅支持</div><div><div><div>* 委托名称</div><div>iamweituo</div></div><div><div>* 信任主体类型</div><div><div><div><div>普通账号</div><div>将账号内资源的操作权限委托给其他华为云账号。</div></div><div><div>云服务</div><div>将账号内资源的操作权限委托给华为云服务。</div></div><div><div>身份提供商</div><div>将账号内资源的操作权限委托给身份提供商。</div></div><div><div>自定义信任策略</div><div>直接在json中编辑自定义信任策略。</div></div></div></div><div><div>* 云服务</div><div>选择云服务</div></div><div><div>* 最大会话持续时长 ②</div><div>Q AgentIdentity</div></div><div><div>智能体身份服务</div></div><div><div>描述</div><div>将账号内资源的操作权限委托给其他华为账号。</div></div></div></div></div></div><div>使用的IAM信任委托需包含如下action：<br/>sts::setContext：获取STS5委托所需权限，获取临时凭据时携带用户工作负载访问令牌（ workload access token ）信息。</div><div><div>策略内容</div><div><div><div>1 {</div><div>2 "Version": "5.0",</div><div>3 "Statement": [</div><div>4 {</div><div>5 "Action": [</div><div>6 "sts:agencies:assume",</div><div>7 "sts::setContext"</div><div>8 ],</div><div>9 "Effect": "Allow",</div><div>10 "Principal": {</div><div>11 "Service": [</div><div>12 "service.AgentIdentity"</div><div>13 ]</div><div>14 }</div><div>15 ]</div><div>16 }</div><div>17 }</div></div></div></div><div>“云服务”类型的Target，IAM信任委托中需包含所选接口的授权项，具体操作请参考<a href="#">为信任委托授权</a>。</div><div><div>● 无认证：在网关将请求转发到Target时，不对请求进行身份验证和授权。</div></div></div></div></div> |

**步骤5** 创建完成后，即可在列表展示已创建的Target。

Target的状态说明如下：

网关创建完成后，在网关列表中单击网关名称，进入基本信息页面，在“关联Target”区域即可查看Target状态。



关联Target (5/100)

刷新Target信息 创建Target

MCP类型的Target外部工具如有变化，需手动同步MCP信息，更新语义检索工具。

默认按照Target名称搜索

| 名称              | 描述 | 类型  | 更新时间                | 状态   | 操作       |
|-----------------|----|-----|---------------------|------|----------|
| target-status   | -  | MCP | 2026-09-22 11:36:51 | 离线   | 同步 编辑 删除 |
| target-yd2/guar | -  | MCP | 2026-09-22 15:34:50 | 离线   | 同步 编辑 删除 |
| target-B6c3hncs | -  | MCP | 2026-09-22 15:39:34 | 离线   | 同步 编辑 删除 |
| target-akku5j   | -  | MCP | 2026-09-22 15:59:20 | 在线   | 同步 编辑 删除 |
| target-evalip67 | -  | MCP | 2026-09-22 16:03:08 | 就绪可用 | 同步 编辑 删除 |

- 就绪可用状态：Target创建成功后，状态初始化为“就绪可用”。
  - 在线状态：启动沙箱并探测Target连通性成功后，状态更新为“在线”，此时Target可正常调用。当关联该Target的沙箱被释放后，状态将回退至“就绪可用”。
  - 离线状态：启动沙箱并探测Target连通性失败后，状态更新为“离线”，离线状态下Target无法正常调用。
- 离线状态的两种情况以及解决方案：
- 连接超时：Target的连接请求超时，无法在规定时间内建立连接，请检查网关和Target的网络连接性或检查Target服务是否正常运行。
  - 解析不到IP：Target的IP地址无法解析，导致连接失败。
- 请检查Target配置的域名DNS解析是否正常，使用命令行工具（如nslookup或dig）测试Target的域名解析是否正常。

当启用语义检索并同步MCP类型的Target时，触发同步操作（单击操作列的“同步”，或勾选多个Target后单击“同步Target信息”）后，Target状态将呈现以下两种情况：

- 同步中：系统正在同步Target信息。
- 等待同步中：当前Target排队等待同步中，状态提示为“等待同步中”。
- 同步失败：Target信息同步失败，可能是Target后端服务不可达或工具列表获取异常。请检查Target后端服务是否正常运行后重新同步。

----结束

## 同步 Target 信息

手动同步Target信息仅支持MCP协议且为“MCP”类型的Target，REST API、APIG和云服务类型的Target系统会自动同步。

在创建网关时开启语义检索功能后系统会自动同步Target信息，但如果MCP类型的Target外部工具发生变化，需要手动同步MCP信息，以确保语义检索工具能够获取最新的数据。

手动同步Target操作如下：

**步骤1** 在网关列表单击网关名称，进入网关详情页面。

**步骤2** 在网关详情页面的“关联Target”区域。

**步骤3** 同步Target信息。

- 同步单个Target信息：单击Target列表操作列的“同步”，系统提示“同步Target 任务已创建！”。
- 同步多个Target信息：在Target列表勾选多个Target，单击“同步Target信息”，系统提示“同步 Target 任务已创建！”。

----结束

- 同步后Target状态将呈现以下两种情况：
- 同步中：系统正在同步Target信息。
  - 等待同步中：当前Target排队等待同步中，状态提示为“等待同步中”。

6.5.3 创建推理 Target

创建Target时，支持创建推理Target。推理Target用于将LLM推理请求（如 /v1/chat/completions、/v1/messages）路由到外部模型提供商，实现多模型统一接入和认证管理。

推理Target的核心工作原理是将LLM推理请求通过独立路径（/inference/{path}）路由到外部推理提供者，与MCP调用路径（/mcp）隔离，互不干扰。

前提条件

已完成[创建网关](#)。

约束与限制

- 一个网关最多可关联10个Target（包含MCP Target和推理Target）。
- 推理Target地址需以https://开头，确保传输层安全。
- 单个Target的操作最多可添加100个模型。

创建推理 Target

- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 在左侧导航栏选择“托管与运行 > 网关”，进入网关界面。
- 步骤3** 选择已创建的网关，单击网关名称，在“关联Target”区域，单击“创建Target”。
- 步骤4** 输入Target的配置信息，参数说明请参考[表6-9](#)，然后单击“确定”。

表 6-9 参数说明

| 参数       | 说明                                                                                                                                                                                  |
|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名称       | Target的名称，同一网关下Target名称不可重复。<br>命名规则： <ul style="list-style-type: none"><li>• 以字母或数字开头和结尾。</li><li>• 支持大小写字母、数字和连字符。</li><li>• 长度为 1-50 个字符。</li></ul> 系统会默认生成Target的名称，可根据需求自定义修改。 |
| 描述       | 用于说明该Target的内容与用途。此描述仅在网关层生效，不会覆盖或影响后端原始工具的描述。<br>规则：长度不大于200个字符。                                                                                                                   |
| Target协议 | 选择“推理”。<br>推理Target用于调用已定义提供商、模型和参数的LLM。                                                                                                                                            |

| 参数           | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Inference 地址 | 模型提供商的基础URL，用于转发 Inference 请求。必须以 https:// 开头的完整URL。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 模型映射 (可选)    | <p>配置客户端模型ID转发给服务商前的转换逻辑。可要求客户端传入带服务商前缀的完整ID，或使用不带提供商前缀的模型ID，由网关自动补全服务商前缀，支持 . 和 / 两种分隔格式。</p> <ul style="list-style-type: none"><li>● <b>使用完整 ID</b>：客户端请求时需传入带提供商前缀的完整模型ID（如 anthropic/claude-3-opus）。</li><li>● <b>省略前缀，分隔符：.</b>：客户端可省略提供商前缀，网关自动补全。使用点号（.）作为分隔符（如 anthropic.claude-3-opus）。</li><li>● <b>省略前缀，分隔符：/</b>：客户端可省略提供商前缀，网关自动补全。使用斜杠（/）作为分隔符（如 anthropic/claude-3-opus）。</li></ul> <p><b>说明</b><br/>选择“省略前缀”，客户端可以使用不带提供商前缀的模型ID发起请求，网关会自动添加前缀转发到后端。例如配置了模型anthropic/claude-3-opus，客户端可以直接使用claude-3-opus发起请求，网关自动补全为 anthropic/claude-3-opus。</p> |
| 添加操作         | <p>配置Inference地址使用的路径和模型。</p> <p>单击“添加操作”，配置如下信息，最多可添加10个操作。</p> <p><b>操作路径</b>：从下拉框中选择操作请求路径，如 /v1/chat/completions、/v1/messages 等。支持的操作路径请参见<a href="#">支持的操作路径</a>。</p> <p><b>提供商路径覆盖（可选）</b>：提供商实际路径，默认与操作路径相同。如配置了提供商路径覆盖，网关会将请求转发到该路径指定的地址。例如，操作路径为 /v1/messages，提供商路径覆盖配置为 /anthropic/v1/messages，则请求会被转发到 /anthropic/v1/messages。</p> <p><b>模型</b>：定义该操作支持的所有可用模型。模型名称必须与模型提供商定义的格式匹配。支持精确匹配和通配匹配（如 gpt-* 支持前缀通配，*-turbo 支持后缀通配）。可添加多个模型，单个操作最多可添加100个模型。</p>                                                                                              |

| 参数     | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 出站身份认证 | <p>配置出站身份认证方式，用于在网关将请求转发到后端服务（Target）时，对请求进行身份验证和授权。</p> <p>支持以下认证方式：</p> <ul style="list-style-type: none"><li>● API Key：在转发请求时，将API Key附加到请求头或查询参数中。<ul style="list-style-type: none"><li>- 选择出站身份，如下拉框无选项请单击“创建出站身份”，创建“API Key”类型的出站身份。</li><li>- 选择位置。<br/>选择 API Key 的传递位置。<ul style="list-style-type: none"><li>■ 标头：在HTTP请求的头部字段中传递API Key。</li><li>■ 查询参数：在HTTP请求的URL中传递API Key。</li></ul></li><li>- 参数名称（可选）：输入参数名称，在请求中传递API的密钥时使用的具体名称。</li><li>- 前缀（可选）：在请求头中传递API Key时，API密钥值之前的前缀部分，如在授权标头中使用的“Bearer”。填写时不需要增加空格，平台会自动在密钥值前添加空格组装，例如填写“Bearer”，平台最终组装为“Bearer xxx”进行传递。</li></ul></li><li>● OAuth<ul style="list-style-type: none"><li>- 选择出站身份，如下拉框无选项请单击“创建出站身份”，创建“OAuth 客户端”类型的出站身份。</li><li>- 选择授权类型。<br/>客户端授权：适用于用户授权应用程序（客户端）访问其资源。</li><li>- 作用域：用于定义可以访问的资源 and 操作的范围，确保只能访问其被授权的资源。输入范围名称，最大可添加100个范围。</li><li>- 自定义参数：在请求中传递的额外参数，可以用于传递特定的信息或满足特定的认证需求。请输入自定义参数信息的键和值，最大可添加255个参数。</li></ul></li><li>● 无认证：在网关将请求转发到Target时，不对请求进行身份验证和授权。</li></ul> |

**步骤5** 创建完成后，即可在列表展示已创建的Target。

Target的状态说明如下：

网关创建完成后，在网关列表中单击网关名称，进入基本信息页面，在“关联Target”区域即可查看Target状态。

- 就绪可用状态：Target创建成功后，状态初始化为“就绪可用”。
- 在线状态：启动沙箱并探测Target连通性成功后，状态更新为“在线”，此时Target可正常调用。当关联该Target的沙箱被释放后，状态将回退至“就绪可用”。
- 离线状态：启动沙箱并探测Target连通性失败后，状态更新为“离线”，离线状态下Target无法正常调用。

离线状态的两种情况以及解决方案：

- 连接超时：Target的连接请求超时，无法在规定时间内建立连接，请检查网关和Target的网络连接性或检查Target服务是否正常运行。

- 解析不到IP：Target的IP地址无法解析，导致连接失败。  
请检查Target配置的域名DNS解析是否正常，使用命令行工具（如nslookup或dig）测试Target的域名解析是否正常。

----结束

支持的操作路径

推理Target支持以下操作路径，对应不同的模型类型和API协议：

| 操作路径                      | 模型类型                     | API协议       |
|---------------------------|--------------------------|-------------|
| /v1/chat/completions      | 文本对话/图像理解（ completions ） | 标准OpenAI协议  |
| /v1/responses             | 多模态对话（ responses ）       | 标准OpenAI协议  |
| /v1/embeddings            | 文本向量化（ embeddings ）      | 标准OpenAI协议  |
| /v1/rerank                | 文本排序（ rerank ）           | 标准OpenAI协议  |
| /v1/messages              | 文本对话/图像理解（ messages ）    | Anthropic协议 |
| /v1/realtime              | 实时语音对话（ realtime ）       | 标准OpenAI协议  |
| /v1/messages/count_tokens | Token计数（ token_count ）   | Anthropic协议 |

6.5.4 MCP 协议支持的 Target 类型介绍

6.5.4.1 REST API

OpenAPI 是用于描述REST API的广泛使用标准。Gateway支持通过OpenAPI 3.0与Swagger 2.0规范来定义 API Targets。

OpenAPI Targets使Gateway对接后端REST服务，该后端接口由导入的OpenAPI规范进行描述。Gateway将传入的MCP请求转换为这些API的HTTP请求。

关键注意事项和限制

- 在使用 OpenAPI targets时，请注意以下要求和限制：
- 仅支持OpenAPI 3.0和Swagger2.0规范的OpenAPI。
  - OpenAPI 文件必须没有语义错误。
  - servers属性：  
仅能定义一个包含实际端点的有效 URL，仅支持静态 URL，例如：https://api.example.com/v1，且安全限制必须使用 HTTPS 协议，禁止配置以下 IP 网段：

- 127.0.0.0/8
- 169.254.0.0/16
- 0.0.0.0/8
- 255.255.255.255/32
- ::/128
- ::1/128
- fe80::/10

```
{
 "servers": [
 {
 "url": "https://api.example.com/v1"
 }
]
}
```

- method：必须是合法的 HTTP 方法（GET、POST、PUT、DELETE、PATCH、HEAD、OPTIONS）。
- 所有 operations 中必须包含唯一的 operationId，要满足长度限制，operationId 会被用作工具名称。
- \$ref 引用完整性：使用 \$ref 引用 #/components/schemas/ 或 #/components/requestBodies/ 时，components 必须存在。
- 在 paths 对象里，具体的路由（比如 /pet）不能直接用\$ref引用来代替，必须是 Path Item 对象（包含 get、post、put 等方法定义）。

## OpenAPI 功能支持

| 功能支持     | 支持功能                                                                                                                                              | 不支持功能                                                                                                                                                        |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 参数定义     | 支持schema定义                                                                                                                                        | 不支持content定义                                                                                                                                                 |
| path参数   | <ul style="list-style-type: none"><li>• 支持String和Number</li><li>• 支持简单的路径参数定义（示例：/users/{userId}）</li><li>• path中{userId}必须有path参数定义</li></ul>    | <ul style="list-style-type: none"><li>• 不支持Object/Array/Bool/Null</li><li>• 不支持复杂的路径参数序列化器（示例：/users/{;id*}）</li></ul>                                       |
| header参数 | <ul style="list-style-type: none"><li>• 仅支持String</li><li>• 只有schema定义的header参数才会被传递</li></ul>                                                    | <ul style="list-style-type: none"><li>• 不支持Object/Array/Bool/Null/Number</li><li>• 不支持自定义header：content-length、content-type、transfer-encoding、host</li></ul> |
| query参数  | <ul style="list-style-type: none"><li>• 支持String/Number/Bool/Null/Array</li><li>• Array内支持String/Number/Bool/Null，Array必须使用explode=True</li></ul> | 不支持Object                                                                                                                                                    |

| 功能支持     | 支持功能                                                                                                                | 不支持功能                 |
|----------|---------------------------------------------------------------------------------------------------------------------|-----------------------|
| cookie参数 | -                                                                                                                   | 不支持                   |
| 请求体      | <ul style="list-style-type: none"><li>支持application/json格式</li><li>请求体定义application/json下必须使用schema-ref引用</li></ul> | 不支持application/xml等格式 |
| 响应体      | -                                                                                                                   | 不支持                   |

OpenAPI 模式规范

- 在配置 OpenAPI targets时，有关 OpenAPI 规范格式的信息，请参见 [OpenAPI Specification](#)。
- 支持的OpenAPI规范示例
  - openapi 3.0

```
{
 "openapi": "3.0.4",
 "info": {
 "title": "Pet Store API",
 "version": "1.0.0",
 "description": "This is a sample Pet Store Server based on the OpenAPI 3.0 specification."
 },
 "servers": [
 {
 "url": "https://api.example.com/v1"
 }
],
 "paths": {
 "/pet": {
 "post": {
 "tags": [
 "pet"
],
 "summary": "Add a new pet to the store.",
 "description": "Add a new pet to the store.",
 "operationId": "addPet",
 "requestBody": {
 "$ref": "#/components/requestBodies/Pet"
 },
 "responses": {
 "200": {
 "description": "Successful operation",
 "content": {
 "application/json": {
 "schema": {
 "$ref": "#/components/schemas/Pet"
 }
 }
 }
 }
 }
 }
 },
 "/pet/{petId}": {
 "get": {
 "summary": "Find pet by ID.",
 "description": "Returns a single pet.",
 "operationId": "getPetById",
 "parameters": [
```

```
{
 "name": "petId",
 "in": "path",
 "description": "ID of pet to return",
 "required": true,
 "schema": {
 "type": "integer",
 "format": "int64"
 }
},
"responses": {
 "200": {
 "description": "successful operation",
 "content": {
 "application/json": {
 "schema": {
 "$ref": "#/components/schemas/Pet"
 }
 }
 }
 }
}
},
"put": {
 "summary": "Update an existing pet.",
 "description": "Update an existing pet by Id.",
 "operationId": "updatePet",
 "parameters": [
 {
 "name": "petId",
 "in": "path",
 "description": "Pet id to Update",
 "required": true,
 "schema": {
 "type": "integer",
 "format": "int64"
 }
 }
],
 "requestBody": {
 "description": "Update an existent pet in the store",
 "content": {
 "application/json": {
 "schema": {
 "$ref": "#/components/schemas/Pet"
 }
 }
 }
 },
 "required": true
},
"responses": {
 "200": {
 "description": "Successful operation",
 "content": {
 "application/json": {
 "schema": {
 "$ref": "#/components/schemas/Pet"
 }
 }
 }
 }
}
},
"delete": {
 "summary": "Deletes a pet.",
 "description": "Delete a pet.",
 "operationId": "deletePet",
 "parameters": [
```



```
{
 "name": "petId",
 "in": "path",
 "description": "Pet id to delete",
 "required": true,
 "schema": {
 "type": "integer",
 "format": "int64"
 }
},
"responses": {
 "200": {
 "description": "Pet deleted"
 }
}
},
"components": {
 "schemas": {
 "Pet": {
 "required": [
 "name"
],
 "type": "object",
 "properties": {
 "id": {
 "type": "integer",
 "format": "int64",
 "example": 10
 },
 "name": {
 "type": "string",
 "example": "doggie"
 },
 "status": {
 "type": "string",
 "description": "pet status in the store",
 "enum": [
 "available",
 "pending",
 "sold"
]
 }
 }
 }
 },
 "requestBodies": {
 "Pet": {
 "description": "Pet object that needs to be added to the store",
 "content": {
 "application/json": {
 "schema": {
 "$ref": "#/components/schemas/Pet"
 }
 }
 }
 }
 }
}
```

– swagger 2.0

```
{
 "swagger": "2.0",
 "info": {
 "title": "Pet Store API",
 "version": "1.0.0",
 "description": "This is a sample Pet Store Server based on the Swagger 2.0 specification."
 }
}
```

```

},
"host": "api.example.com",
"basePath": "/v1",
"schemes": ["https"],
"paths": {
 "/pet": {
 "post": {
 "tags": ["pet"],
 "summary": "Add a new pet to the store.",
 "description": "Add a new pet to the store.",
 "operationId": "addPet",
 "consumes": ["application/json"],
 "parameters": [
 {
 "name": "body",
 "in": "body",
 "description": "Pet object that needs to be added to the store",
 "required": true,
 "schema": {
 "$ref": "#/definitions/Pet"
 }
 }
],
 "responses": {
 "200": {
 "description": "Successful operation",
 "schema": {
 "$ref": "#/definitions/Pet"
 }
 }
 }
 },
 "get": {
 "summary": "Find pet by ID.",
 "description": "Returns a single pet.",
 "operationId": "getPetById",
 "parameters": [
 {
 "name": "petId",
 "in": "path",
 "description": "ID of pet to return",
 "required": true,
 "type": "integer",
 "format": "int64"
 }
],
 "responses": {
 "200": {
 "description": "successful operation",
 "schema": {
 "$ref": "#/definitions/Pet"
 }
 }
 }
 },
 "put": {
 "summary": "Update an existing pet.",
 "description": "Update an existing pet by Id.",
 "operationId": "updatePet",
 "consumes": ["application/json"],
 "parameters": [
 {
 "name": "petId",
 "in": "path",
 "description": "Pet id to Update",
 "required": true,
 "type": "integer",

```

```
 "format": "int64"
 },
 {
 "name": "body",
 "in": "body",
 "description": "Update an existent pet in the store",
 "required": true,
 "schema": {
 "$ref": "#/definitions/Pet"
 }
 }
],
 "responses": {
 "200": {
 "description": "Successful operation",
 "schema": {
 "$ref": "#/definitions/Pet"
 }
 }
 }
 },
 "delete": {
 "summary": "Deletes a pet.",
 "description": "Delete a pet.",
 "operationId": "deletePet",
 "parameters": [
 {
 "name": "petId",
 "in": "path",
 "description": "Pet id to delete",
 "required": true,
 "type": "integer",
 "format": "int64"
 }
],
 "responses": {
 "200": {
 "description": "Pet deleted"
 }
 }
 }
},
"definitions": {
 "Pet": {
 "type": "object",
 "required": ["name"],
 "properties": {
 "id": {
 "type": "integer",
 "format": "int64",
 "example": 10
 },
 "name": {
 "type": "string",
 "example": "doggie"
 },
 "status": {
 "type": "string",
 "description": "pet status in the store",
 "enum": ["available", "pending", "sold"]
 }
 }
 }
}
```

### 6.5.4.2 MCP

当网关Target配置为MCP类型时，网关遵循标准 MCP 协议（基于 JSON-RPC 2.0 消息格式）与后端 MCP Server 进行交互，无需额外适配转换逻辑，可原生兼容MCP Server的工具调用能力。网关会统一接收智能体与大模型下发的结构化指令及上下文请求，转发至对应的后端MCP Server，由MCP Server完成工具执行操作，并将执行结果标准化回传至网关，最终返回给智能体完成闭环调用。关于MCP Servers的详细介绍请参见[MCP Servers介绍](#)。

MCP 服务器在 AgentArts 中提供工具的功能，支持智能体与模型的交互。在创建网关时，可以定义预配置的 MCP 服务器作为目标，MCP 服务器托管了智能体可以发现和使用的工具。

- 配置同步  
创建Target、更新Target以及删除Target操作会同步网关配置。调用这些操作时，网关会通过MCP的 tools/list能力获取可用工具。
- MCP服务器目标的配置要求
  - 工具功能：MCP服务器必须具备工具功能。
  - 支持的MCP协议版本：包括2024-11-05、2025-06-18、2025-03-26、2025-11-25。

### 6.5.4.3 APIG

API网关（APIG）是为企业和开发者提供的高性能、高可用、高安全的云原生网关服务，帮助企业快速构建、管理和部署API。当网关Target配置为APIG类型时，将以APIG上注册的API作为目标服务。此时，需从APIG导出API文档，并补充完善请求体的详细定义后，方可调用后端服务。

### 约束与限制

- 仅支持OpenAPI 3.0 和Swagger 2.0 规范。
- 响应体需用户自行配置，且必须使用\$ref引用方式定义。
- 当网关出网配置为公网访问时，APIG实例需开启公网入口；当网关出网配置为私网访问时，所配置的VPC需与APIG实例所在VPC网络互通。
- 出站认证支持IAM以及无认证方式。  
使用IAM出站认证时，需确保创建的出站身份认证中，IAM信任委托包含如下 Action 权限：  
sts::setContext：获取STS5委托所需权限，获取临时凭据时携带用户工作负载访问令牌（workload access token）信息。



图 6-19 修改前

```
"paths": {
 "/pet": {
 "post": {
 "description": "Add a new pet to the store.",
 "operationId": "addPet",
 "requestBody": {
 "content": {
 "application/json": {
 "schema": {
 "example": "",
 "type": "object"
 }
 }
 }
 }
 }
 }
},
```

- 修改后（所需配置）：将 "type": "object" 替换为 "\$ref": "#/components/schemas/Pet"，并在 components 中新增 schemas 节点定义具体结构。

图 6-20 修改后的请求体

```
"paths": {
 "/pet": {
 "post": {
 "description": "Add a new pet to the store.",
 "operationId": "addPet",
 "requestBody": {
 "content": {
 "application/json": {
 "schema": {
 "$ref": "#/components/schemas/Pet"
 }
 }
 }
 }
 }
 }
},
```

图 6-21 新增 schemas 节点具体结构

```
{
 "components": {
 "responses": {
 "default": {
 "description": "response example"
 }
 },
 "schemas": {
 "Pet": {
 "type": "object",
 "properties": {
 "id": {
 "type": "integer",
 "format": "int64"
 },
 "name": {
 "type": "string"
 },
 "category": {
 "$ref": "#/components/schemas/Category"
 },
 "photoUrls": {
 "type": "array",
 "items": {
 "type": "string"
 }
 },
 "tags": {
 "type": "array",
 "items": {
 "$ref": "#/components/schemas/Tag"
 }
 },
 "status": {
```

#### 导出的原始 API 示例:

```
{
 "openapi": "3.0.3",
 "info": {
 "description": "This is a sample Pet Store Server based on the OpenAPI 3.0 specification. You can find out more about Swagger at https://swagger.io. In the third iteration of the pet store, we have switched to the design first approach!\nYou can now",
 "title": "Petstore",
 "version": "2026-07-16-09:51:07"
 },
 "servers": [
 {
 "url": "https://gateway-test.cn-north-7.myhuaweicloud.com"
 }
],
 "paths": {
 "/pet": {
 "post": {
 "description": "Add a new pet to the store.",
 "operationId": "addPet",
 "requestBody": {
 "content": {
 "application/json": {
```

```

 "schema": {
 "example": "",
 "type": "object" #【修改点1】：此处需替换为具体的 $ref 引用
 }
 }
 },
 "responses": {
 "default": {
 "description": "response example",
 "x-apigateway-result-failure-sample": "",
 "x-apigateway-result-normal-sample": ""
 }
 },
 "security": [
 {
 "apig-auth-iam": []
 }
],
 "servers": [
 {
 "url": "https://gateway-example.com"
 }
],
 "tags": [
 "pet"
],
 "x-apigateway-backend": {
 "httpVpcEndpoints": {
 "cascade_flag": false,
 "description": "",
 "enableClientSsl": false,
 "enableSmChannel": false,
 "method": "POST",
 "name": "VPC_gjlc",
 "path": "/pet",
 "retryCount": "0",
 "scheme": "http",
 "timeout": 5000
 },
 "type": "HTTP-VPC"
 },
 "x-apigateway-cors": false,
 "x-apigateway-is-send-fg-body-base64": true,
 "x-apigateway-match-mode": "NORMAL",
 "x-apigateway-request-type": "public"
 },
 "put": {
 "description": "Update an existing pet by Id.",
 "operationId": "updatePet",
 "requestBody": {
 "content": {
 "application/json": {
 "schema": {
 "example": "",
 "type": "object" #【修改点2】：此处需替换为具体的$ref 引用
 }
 }
 }
 }
 },
 "responses": {
 "default": {
 "description": "response example",
 "x-apigateway-result-failure-sample": "",
 "x-apigateway-result-normal-sample": ""
 }
 },
 "servers": [
 {

```



```

 "url": "https://gateway-example.com"
 }
],
 "tags": [
 "pet"
],
 "x-apigateway-backend": {
 "httpVpcEndpoints": {
 "cascade_flag": false,
 "description": "",
 "enableClientSsl": false,
 "enableSmChannel": false,
 "method": "PUT",
 "name": "VPC_gjlc",
 "path": "/pet",
 "retryCount": "0",
 "scheme": "http",
 "timeout": 5000
 },
 "type": "HTTP-VPC"
 },
 "x-apigateway-cors": false,
 "x-apigateway-is-send-fg-body-base64": true,
 "x-apigateway-match-mode": "NORMAL",
 "x-apigateway-request-type": "public"
 }
}
},
"components": {
 "responses": {
 "default": {
 "description": "response example"
 }
 },
 "securitySchemes": { #【修改点3】：补充 schemas 定义
 "apig-auth-app": {
 "in": "header",
 "name": "Authorization",
 "type": "apiKey",
 "x-apigateway-auth-type": "AppSigv1"
 },
 "apig-auth-app-header": {
 "in": "header",
 "name": "Authorization",
 "type": "apiKey",
 "x-apigateway-auth-opt": {
 "appcode-auth-type": "header"
 },
 "x-apigateway-auth-type": "AppSigv1"
 },
 "apig-auth-iam": {
 "in": "header",
 "name": "unused",
 "type": "apiKey",
 "x-apigateway-auth-type": "IAM"
 }
 }
}
}
}

```

### 修改后的 API 示例：

修改后的完整配置如下，重点关注requestBody和components.schemas的变化：

```

{
 "openapi": "3.0.3",
 "info": {
 "description": "This is a sample Pet Store Server based on the OpenAPI 3.0 specification. You can find out more about Swagger at https://swagger.io. In the third iteration of the pet store, we have switched to the design first approach!\nYou can now",
 "title": "Petstore",

```

```

 "version": "2026-07-10-00:12:36"
 },
 "servers": [
 {
 "url": "https://gateway-test.cn-north-7.myhuaweicloud.com"
 }
],
 "paths": {
 "/pet": {
 "post": {
 "description": "Add a new pet to the store.",
 "operationId": "addPet",
 "requestBody": {
 "content": {
 "application/json": {
 "schema": {
 "$ref": "#/components/schemas/Pet" #【已修改】：指向具体的 Schema
 }
 }
 }
 },
 "responses": {
 "default": {
 "description": "response example",
 "x-apigateway-result-failure-sample": "",
 "x-apigateway-result-normal-sample": ""
 }
 },
 "security": [
 {
 "apig-auth-iam": []
 }
],
 "servers": [
 {
 "url": "https://gateway-test.cn-north-7.myhuaweicloud.com"
 }
],
 "tags": [
 "pet"
],
 "x-apigateway-backend": {
 "httpVpcEndpoints": {
 "cascade_flag": false,
 "description": "",
 "enableClientSsl": false,
 "enableSmChannel": false,
 "method": "POST",
 "name": "VPC_4ap5",
 "path": "/pet",
 "retryCount": "0",
 "scheme": "http",
 "timeout": 5000
 },
 "type": "HTTP-VPC"
 },
 "x-apigateway-cors": false,
 "x-apigateway-is-send-fg-body-base64": true,
 "x-apigateway-match-mode": "NORMAL",
 "x-apigateway-request-type": "public"
 },
 "put": {
 "description": "Update an existing pet by Id.",
 "operationId": "updatePet",
 "requestBody": {
 "content": {
 "application/json": {
 "schema": {
 "$ref": "#/components/schemas/Pet" #【已修改】：指向具体的 Schema
 }
 }
 }
 }
 }
 }
 }
}

```

```

 }
 }
},
"responses": {
 "default": {
 "description": "response example",
 "x-apigateway-result-failure-sample": "",
 "x-apigateway-result-normal-sample": ""
 }
},
"security": [
 {
 "apig-auth-iam": []
 }
],
"servers": [
 {
 "url": "https://gateway-test.cn-north-7.myhuaweicloud.com"
 }
],
"tags": [
 "pet"
],
"x-apigateway-backend": {
 "httpVpcEndpoints": {
 "cascade_flag": false,
 "description": "",
 "enableClientSsl": false,
 "enableSmChannel": false,
 "method": "PUT",
 "name": "VPC_4ap5",
 "path": "/pet",
 "retryCount": "0",
 "scheme": "http",
 "timeout": 5000
 },
 "type": "HTTP-VPC"
},
"x-apigateway-cors": false,
"x-apigateway-is-send-fg-body-base64": true,
"x-apigateway-match-mode": "NORMAL",
"x-apigateway-request-type": "public"
}
},
"components": {
 "responses": {
 "default": {
 "description": "response example"
 }
 }
},
"schemas": {
 # 【已新增】：补充具体的 Schema 定义
 "Pet": {
 "type": "object",
 "properties": {
 "id": {
 "type": "integer",
 "format": "int64"
 },
 "name": {
 "type": "string"
 },
 "category": {
 "$ref": "#/components/schemas/Category"
 },
 "photoUrls": {
 "type": "array",
 "items": {

```

```

 "type": "string"
 }
 },
 "tags": {
 "type": "array",
 "items": {
 "$ref": "#/components/schemas/Tag"
 }
 },
 "status": {
 "type": "string",
 "enum": [
 "available",
 "pending",
 "sold"
]
 },
 "required": [
 "name",
 "photoUrls"
]
 },
 "Category": {
 "type": "object",
 "properties": {
 "id": {
 "type": "integer",
 "format": "int64"
 },
 "name": {
 "type": "string"
 }
 }
 },
 "Tag": {
 "type": "object",
 "properties": {
 "id": {
 "type": "integer",
 "format": "int64"
 },
 "name": {
 "type": "string"
 }
 }
 }
},
"securitySchemes": {
 "apig-auth-app": {
 "in": "header",
 "name": "Authorization",
 "type": "apiKey",
 "x-apigateway-auth-type": "AppSigv1"
 },
 "apig-auth-app-header": {
 "in": "header",
 "name": "Authorization",
 "type": "apiKey",
 "x-apigateway-auth-opt": {
 "appcode-auth-type": "header"
 },
 "x-apigateway-auth-type": "AppSigv1"
 },
 "apig-auth-iam": {
 "in": "header",
 "name": "unused",
 "type": "apiKey",
 "x-apigateway-auth-type": "IAM"
 }
}

```

```
}
}
}
}
```

#### 6.5.4.4 云服务

当网关Target配置为云服务类型时，目标服务指向华为云服务的开放接口。当前支持50余个华为云服务，通过调用注册在APIE（API Explorer）上的接口实现对接。

#### 认证与授权要求

出站认证支持IAM以及无认证方式。

使用IAM出站认证时，需确保创建的出站身份认证中，IAM信任委托包含如下Action权限：

sts::setContext：获取STS5委托所需权限，获取临时凭据时携带用户工作负载访问令牌（workload access token）信息。

策略内容

```
1 {
2 "Version": "5.0",
3 "Statement": [
4 {
5 "Action": [
6 "sts:agencies:assume",
7 "sts::setContext"
8],
9 "Effect": "Allow",
10 "Principal": {
11 "Service": [
12 "service.AgentIdentity"
13]
14 }
15]
16 }
17 }
```

IAM信任委托中需包含所选接口的授权项，具体操作如下：

**步骤1** 在Target列表单击Target名称，进入Target详情页面。

**步骤2** 单击接口“详情”，在“文档”页签查看接口授权项信息。

图 6-22 Target 页面



图 6-23 在“文档”页签查看接口授权项信息



步骤3 创建IAM信任委托并授权。

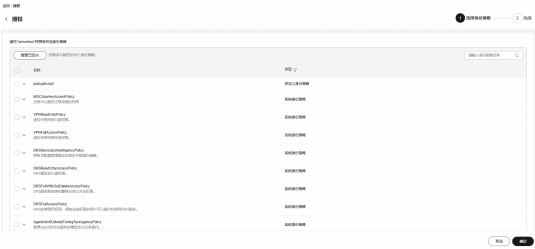
图 6-24 创建信任委托



图 6-25 为创建的信任委托授权



图 6-26 授权项



如果授权中无系统身份策略，可[创建自定义策略](#)后进行授权。

----结束

## 6.5.5 Target 的出站认证配置

出站认证适用于所有类型的Target。网关在将请求转发到Target后端时，自动附加配置好的认证信息，使Agent代码中不出现后端服务的凭证。

为什么需要出站认证

当Agent通过网关调用外部工具或模型时，后端服务通常需要身份验证。如果让Agent代码直接管理这些凭证：

- 凭证硬编码在代码中，存在泄露风险。
- 每次更换后端服务或轮换密钥都需要修改Agent代码并重新部署。
- 多个Agent需要调用同一个后端时，凭证配置重复。

出站认证将这些凭证集中管理在网关侧。Agent只需携带网关的入站认证，网关在转发请求时自动附加后端所需的认证信息。

说明

入站认证和出站认证是两个独立的方向：入站认证验证“谁在调用网关”，出站认证是“网关代表调用方访问后端”。两者互不影响，可以各自选择不同的认证方式。

认证方式

| 认证方式    | 认证流程                                                                                              | 适用场景                                                  |
|---------|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| API Key | 网关在转发请求时，将API Key附加到请求头（如 Authorization: Bearer {api_key} 或自定义头），后端服务校验该Key的合法性。                  | 大多数REST API、外部模型提供商（OpenAI / DeepSeek 等）、需要简单密钥认证的服务。 |
| OAuth   | 网关使用配置的OAuth 2.0凭证（Client ID / Client Secret）向授权服务器获取Access Token，然后将Token附加到转发请求中，Token 过期后自动刷新。 | 需要标准OAuth 2.0授权流程的服务、企业SSO对接、华为云云服务。                  |
| IAM 认证  | 网关使用配置的华为云IAM凭证（AK/SK）对转发请求进行签名，后端服务通过签名验证网关身份。签名过程自动完成，无需手动生成。                                   | 华为云云服务（如 OBS、ECS、ModelArts等）、需要华为云IAM身份认证的后端服务。       |
| 无认证     | 网关直接转发请求，不附加任何认证信息。                                                                               | 内网服务（同一VPC内）、已有其他鉴权机制的后端、测试环境。                        |

创建出站身份

出站身份是一组可复用的认证配置。创建后可以绑定到多个Target上，无需在每个Target中重复填写。

在创建Target的“出站身份认证”中，选择“创建出站身份”，填写认证信息后保存，后续在创建其他Target时，可选择已有的出站身份即可。出站身份认证方式（如API Key、OAuth、IAM等）因Target类型而异，请根据页面选择配置。



图 6-27 创建出站身份

创建Target

API、云服务。

数的LLM。

类型

REST API  
使用OPEN API schema文件

MCP  
使用已有的MCP服务器

APIG  
注册在云服务API网关APIG上的API

云服务  
使用华为云服务开放接口

配置信息

OAS 文件导入

OBS

自定义

Swagger 2.0 / OpenAPI 3.0 格式 (YAML/JSON)

请输入

从OBS中选择

如需使用OBS相关功能，当前账号需具备OBS相关权限[查看当前权限](#)

出站身份认证

出站身份认证

API Key

OAuth

无认证

b

创建出站身份

授权类型

客户端模式

说明

- 一个出站身份可以被多个Target 绑定。
- 修改出站身份的配置后，所有绑定的Target同步生效。

6.6 调测与调用 MCP Target

6.6.1 概述

网关创建并添加Target后，在正式集成到Agent之前，需要验证以下三件事是否正常：

- 网关是否可以正常连接 —— 确认入站认证、网络可达。
- Target的工具是否可以被发现 —— 确认Target后端服务正常、工具列表不为空。
- 工具调用是否返回正确结果 —— 确认出站认证正确、参数格式正确、后端服务响应符合预期。

只有这三步都通过，才能保证Agent集成后不会在调用阶段出现意外错误。

约束与限制

调测功能仅支持创建网关时关联“MCP”类型的Target。

认证机制

网关提供入站认证和出站认证两层安全机制，保障调用方与后端工具之间的安全通信。调测和调用网关前，需要获取对应的认证凭证：

| 认证类型   | 方向        | 作用                                   | 详情                                 |
|--------|-----------|--------------------------------------|------------------------------------|
| 入站身份认证 | 调用方 → 网关  | 验证调用方身份并控制访问权限，确认“谁在调用网关”。           | 参见 <a href="#">认证鉴权</a> 。          |
| 出站身份认证 | 网关 → 后端工具 | 配置网关以特定身份访问后端Target资源，确认“网关代表谁访问后端”。 | 参见 <a href="#">Target的出站认证配置</a> 。 |

两层认证相互独立：入站认证在创建网关时配置，出站认证在创建Target时配置。Agent只需携带入站认证凭证，出站认证由网关自动附加，Agent代码中不出现后端服务凭证。

6.6.2 调测网关

网关创建完成后，支持在控制台调测网关，用于验证网关配置的正确性和连通性。

前提条件

已[创建网关](#)并关联Target。

调测网关

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 网关”，进入网关界面。
- 步骤3 在网关列表单击网关名称，在“调测”页签，首先进行连接测试。
  - 网关 URL：创建网关后生成的网关URL，系统默认写入，无需添加。
  - Access token（可选）：当网关的入站身份认证配置为“OAuth 2.0认证”时，可配置Access token，用于验证调用者的身份。如果未设置Access token，系统将调用默认的调测网关，通常用于内部测试和调试；如果设置了Access token，系统将调用租户网关，用于实际的业务场景。
- 步骤4 单击“测试连接”，连接成功后单击“下一步”。
- 步骤5 （可选）工具检索：在网关中开启语义检索后，可通过关键字筛选匹配的工具。

在工具检索框中输入关键字，单击“开始检索”。系统将根据配置的Top N和相似阈值筛选并展示匹配工具。
- 步骤6 单击“下一步”，进入“工具调试”页面，此时“工具列表”展示Target中的工具。如开启“语义检索”，此时“工具列表”展示检索出的工具，如需展示Target中的全部工具，请打开“全部”开关。

**步骤7** 选择待调试的工具后，“请求参数”区域将展示该Target的tools/call接口信息，请按需填写调试参数。

**步骤8** 单击“开始调试”，即可在右侧区域展示调试结果。

如创建网关时开启了日志功能，则调测网关的日志可以在“日志”页签中查看。

----结束

### 6.6.3 列出网关工具

在请求体中指定tools/list作为请求方法，获取网关提供的所有可用工具列表。返回结果包含工具名称、描述和参数定义等信息，可用于后续工具调用。

请求体示例：

```
{
 "jsonrpc": "2.0",
 "id": "list-tools-request",
 "method": "tools/list",
 "params": {
 "cursor": "<CURSOR>"
 }
}
```

其中cursor为可选参数，用于分页请求，首次请求无需携带。

响应体示例：

```
{
 "jsonrpc": "2.0",
 "id": "list-tools-request",
 "result": {
 "tools": [
 {
 "name": "tool_name",
 "description": "工具描述",
 "inputSchema": {}
 }
],
 "nextCursor": "<NEXT_CURSOR>"
 }
}
```

网关支持分页返回工具列表。当Target响应tools/list请求返回nextCursor字段且非空时，表示该Target有更多工具未返回，需要将nextCursor的值作为下一次请求的cursor参数继续请求，重复此过程直至nextCursor为空，即可获得完整的工具列表，详情请参见[MCP协议分页介绍](#)。

网关会根据不同的Target进行分页，每个Target的工具列表独立分页返回。即每次返回的工具属于同一Target，获取下一个Target的工具需要继续分页请求。每个Target最多返回1000个工具。

- API Key认证的tools/list调用示例

```
import requests
import json
import re

def list_tools(gateway_url, access_token):
 headers = {
 "Content-Type": "application/json",
 "Accept": "application/json, text/event-stream",
 "mcp-session-id": "<MCP_SESSION_ID>", # MCP会话ID，相同的sessionid会使用同一网关资源，调用时请配置此参数，例如：4ada96ff-1e16-4e5b-8fb4-b3f0f207a4f3
 }
```

```

 "Mcp-Protocol-Version": "<MCP_PROTOCOL_VERSION>", # MCP协议版本
 "Authorization": f"Bearer {access_token}"
}
payload = {"jsonrpc": "2.0", "id": "list-tools-request", "method": "tools/list"}
response = requests.post(gateway_url, headers=headers, json=payload) # 若需要忽略ssl校验, 可补充参数verify=False
raw = response.content.decode('utf-8', errors='replace')
ct = response.headers.get('content-type', '')
if 'text/event-stream' in ct:
 for line in raw.strip().split('\n'):
 if line.startswith('data:'):
 try:
 return json.loads(line[5:].strip())
 except json.JSONDecodeError:
 continue
 m = re.search(r'\{.*\}', raw, re.DOTALL)
 if m:
 return json.loads(m.group())
try:
 return json.loads(raw)
except json.JSONDecodeError:
 print(f"[Fail]: {raw[:200]}")
 return None

gateway_url = "<GATEWAY_URL>" # 网关接口地址（业务API调用入口）
access_token = "<API_KEY>" # 认证凭证：APIKey认证直接填写密钥；无认证则留空/删除
result = list_tools(gateway_url, access_token)
if result:
 print(json.dumps(result, indent=2, ensure_ascii=False))

```

- IAM认证的tools/list调用示例

API签名详情请参见[API签名指南](#)：

```

import requests
import json
import re

def list_tools(gateway_url, access_token):
 headers = {
 "Content-Type": "application/json",
 "Accept": "application/json, text/event-stream",
 "mcp-session-id": "<MCP_SESSION_ID>", # MCP会话ID, 相同的sessionid会使用同一网关资源,
 "Mcp-Protocol-Version": "<MCP_PROTOCOL_VERSION>", # MCP协议版本
 "X-Sdk-Date": "<X_SDK_DATE>", # 网关鉴权时间戳
 "X-Sdk-Content-Sha256": "UNSIGNED-PAYLOAD",
 "Authorization": access_token
 }
 payload = {"jsonrpc": "2.0", "id": "list-tools-request", "method": "tools/list"}
 response = requests.post(gateway_url, headers=headers, json=payload) # 若需要忽略ssl校验, 可补充参数verify=False
 raw = response.content.decode('utf-8', errors='replace')
 ct = response.headers.get('content-type', '')
 if 'text/event-stream' in ct:
 for line in raw.strip().split('\n'):
 if line.startswith('data:'):
 try:
 return json.loads(line[5:].strip())
 except json.JSONDecodeError:
 continue
 m = re.search(r'\{.*\}', raw, re.DOTALL)
 if m:
 return json.loads(m.group())
 try:
 return json.loads(raw)
 except json.JSONDecodeError:
 print(f"[Fail]: {raw[:200]}")
 return None

```

```
gateway_url = "<GATEWAY_URL>" # 网关接口地址（业务API调用入口）
access_token = "<AUTHORIZATION>" # 鉴权Authorization
result = list_tools(gateway_url, access_token)
if result:
 print(json.dumps(result, indent=2, ensure_ascii=False))
```

## 6.6.4 调用网关工具

在请求体中指定tools/call作为请求方法，调用网关提供的特定工具。调用时需指定工具名称和入参，工具名称可通过tools/list接口查询获取，入参需严格按照工具定义的参数结构填写。

请求体示例：

```
{
 "jsonrpc": "2.0",
 "id": "call-tool-request",
 "method": "tools/call",
 "params": {
 "name": "<TOOL_NAME>",
 "arguments": {
 "param1": "value1",
 "param2": "value2"
 }
 }
}
```

其中name为待调用的工具名称，通过tools/list接口查询获取；arguments为工具调用入参，需严格按照工具定义的参数结构填写。

响应体示例：

```
{
 "jsonrpc": "2.0",
 "id": "call-tool-request",
 "result": {
 "content": [
 {
 "type": "text",
 "text": "工具调用结果"
 }
]
 }
}
```

- API Key认证的tools/call调用示例

```
import requests
import json
import re

def call_tool(gateway_url, access_token, tool_name, arguments):
 headers = {
 "Content-Type": "application/json",
 "Accept": "application/json, text/event-stream",
 "mcp-session-id": "<MCP_SESSION_ID>", # MCP会话ID，相同的sessionid会使用同一网关资源，调用时请配置此参数，例如：4ada96ff-1e16-4e5b-8fb4-b3f0f207a4f3
 "Mcp-Protocol-Version": "<MCP_PROTOCOL_VERSION>", # MCP协议版本
 "Authorization": f"Bearer {access_token}"
 }
 payload = {"jsonrpc": "2.0", "id": "call-tool-request", "method": "tools/call", "params": {"name": tool_name, "arguments": arguments}}
 response = requests.post(gateway_url, headers=headers, json=payload) # 若需要忽略ssl校验，可补充参数verify=False
 raw = response.content.decode('utf-8', errors='replace')
 ct = response.headers.get('content-type', '')
 if 'text/event-stream' in ct:
 for line in raw.strip().split('\n'):
```

```
 if line.startswith('data:'):
 try:
 return json.loads(line[5:].strip())
 except json.JSONDecodeError:
 continue
 m = re.search(r'\{.*\}', raw, re.DOTALL)
 if m:
 return json.loads(m.group())
 try:
 return json.loads(raw)
 except json.JSONDecodeError:
 print(f'[Fail]: {raw[:200]}')
 return None

gateway_url = "<GATEWAY_URL>" # 网关接口地址 (业务API调用入口)
access_token = "<API_KEY>" # 认证凭证: APIKey认证直接填写密钥; 无认证则留空/删除
tool_name = "<TOOL_NAME>" # 待调用的工具名称, 通过 tools/list 接口查询获取
arguments = {"param1": "value1", "param2": "value2"} # 工具调用入参, 严格按照工具定义的参数结构填写
result = call_tool(gateway_url, access_token, tool_name, arguments)
if result:
 print(json.dumps(result, indent=2, ensure_ascii=False))
```

- IAM认证的tools/call调用示例

```
import requests
import json
import re

def call_tool(gateway_url, access_token, tool_name, arguments):
 headers = {
 "Content-Type": "application/json",
 "Accept": "application/json, text/event-stream",
 "mcp-session-id": "<MCP_SESSION_ID>", # MCP会话ID, 相同的sessionid会使用同一网关资源,
 # 调用时请配置此参数, 例如: 4ada96ff-1e16-4e5b-8fb4-b3f0f207a4f3
 "Mcp-Protocol-Version": "<MCP_PROTOCOL_VERSION>", # MCP协议版本
 "X-Sdk-Date": "<X_SDK_DATE>", # 网关鉴权时间戳
 "X-Sdk-Content-Sha256": "UNSIGNED-PAYLOAD",
 "Authorization": access_token
 }
 payload = {"jsonrpc": "2.0", "id": "call-tool-request", "method": "tools/call", "params": {"name":
tool_name, "arguments": arguments}}
 response = requests.post(gateway_url, headers=headers, json=payload) # 若需要忽略ssl校验, 可补充参数verify=False
 raw = response.content.decode('utf-8', errors='replace')
 ct = response.headers.get('content-type', '')
 if 'text/event-stream' in ct:
 for line in raw.strip().split('\n'):
 if line.startswith('data:'):
 try:
 return json.loads(line[5:].strip())
 except json.JSONDecodeError:
 continue
 m = re.search(r'\{.*\}', raw, re.DOTALL)
 if m:
 return json.loads(m.group())
 try:
 return json.loads(raw)
 except json.JSONDecodeError:
 print(f'[Fail]: {raw[:200]}')
 return None

gateway_url = "<GATEWAY_URL>" # 网关接口地址 (业务API调用入口)
access_token = "<AUTHORIZATION>" # 鉴权Authorization
tool_name = "<TOOL_NAME>" # 待调用的工具名称, 通过 tools/list 接口查询获取
arguments = {"param1": "value1", "param2": "value2"} # 工具调用入参, 严格按照工具定义的参数结构填写
result = call_tool(gateway_url, access_token, tool_name, arguments)
if result:
 print(json.dumps(result, indent=2, ensure_ascii=False))
```

## 6.6.5 检索网关工具

当您拥有大量工具并且需要为特定用例找到合适工具时，语义搜索尤其重要。开启语义检索后，可以使用tools/call调用网关提供的x\_agentarts\_gateway\_tool\_semantic\_search工具进行工具检索。

```
import requests
import json
import re

def call_tool(gateway_url, access_token, tool_name, arguments):
 headers = {
 "Content-Type": "application/json",
 "Accept": "application/json, text/event-stream",
 "mcp-session-id": "<MCP_SESSION_ID>", # MCP会话ID，相同的sessionid会使用同一网关资源，调用时
 # 请配置此参数，例如：4ada96ff-1e16-4e5b-8fb4-b3f0f207a4f3
 "Mcp-Protocol-Version": "<MCP_PROTOCOL_VERSION>", # MCP协议版本
 "Authorization": f"Bearer {access_token}"
 }
 payload = {"jsonrpc": "2.0", "id": "call-tool-request", "method": "tools/call", "params": {"name":
 tool_name, "arguments": arguments}}
 response = requests.post(gateway_url, headers=headers, json=payload) # 若需要忽略ssl校验，可补充参数
 verify=False
 raw = response.content.decode('utf-8', errors='replace')
 ct = response.headers.get('content-type', '')
 if 'text/event-stream' in ct:
 for line in raw.strip().split('\n'):
 if line.startswith('data:'):
 try:
 return json.loads(line[5:].strip())
 except json.JSONDecodeError:
 continue
 m = re.search(r'\{.*\}', raw, re.DOTALL)
 if m:
 return json.loads(m.group())
 try:
 return json.loads(raw)
 except json.JSONDecodeError:
 print(f"[Fail]: {raw[:200]}")
 return None

gateway_url = "<GATEWAY_URL>" # 网关接口地址（业务API调用入口）
access_token = "<API_KEY>" # 认证凭证：APIKey认证直接填写密钥；无认证则留空/删除
tool_name = "x_agentarts_gateway_tool_semantic_search" # 语义检索调用的工具名称，固定为
"x_agentarts_gateway_tool_semantic_search"
arguments = {"query": "value"} # 语义检索调用入参，value表示检索提示词
result = call_tool(gateway_url, access_token, tool_name, arguments)
if result:
 print(json.dumps(result, indent=2, ensure_ascii=False))
```

## 6.7 调用推理 Target

创建推理Target后，通过网关端点及 /inference/ 路径发起推理请求。

### 前提条件

调用推理Target需进行入站和出站认证：

- 入站身份认证：调用方访问网关时的认证和鉴权，用于验证调用方身份并控制访问权限。详细说明请参考[创建网关](#)中的入站身份认证。
- 出站身份认证：网关访问后端服务时的身份认证，用于配置网关以特定身份访问后端资源。详情请参见[创建推理Target](#)中的出站认证。

请求路径格式

https://{gateway\_endpoint}/inference/{path}

其中 {path} 为标准API路径，如 /v1/chat/completions、/v1/messages 等。关于支持的请求路径，请参见[支持的操作路径](#)。

gateway\_endpoint获取方式：

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 网关”，进入网关界面。
- 步骤3 单击网关名称，查看网关URL。

图 6-28 调用 URL



----结束

API Key 认证

当网关入站身份认证配置为API Key时，调用方需在请求头中携带API Key名称和对应的密钥值。

以下以API Key认证为例，调用OpenAI协议的文本对话接口：

```
POST https://{gateway_endpoint}/inference/v1/chat/completions HTTP/1.1
Content-Type: application/json
{API_Key_Name}: {api_key_value}
{
 "model": "gpt-4",
 "messages": [
 { "role": "user", "content": "Hello, how are you?" }
]
}
```

IAM 认证

当网关入站身份认证配置为IAM时，调用方需使用华为云IAM用户凭据进行签名认证，在请求头中携带Authorization签名信息。

以下以IAM认证为例，调用Anthropic协议的文本对话接口：

```
POST https://{gateway_endpoint}/inference/v1/messages HTTP/1.1
Content-Type: application/json
Authorization: SDK-HMAC-SHA256 Credential={access_key}/{date}/region/agentarts, SignedHeaders=host;x-sdk-date, Signature={signature}
{
 "model": "claude-3-opus",
```



```
"messages": [
 {
 "role": "user",
 "content": "Hello, how are you?"
 }
]
```

## 6.8 管理网关

网关创建完成后，您可以对网关执行如下操作：

| 操作       | 说明                                                                                                                                                                               |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 查看网关URL  | 1. 在网关列表中单击网关名称。<br>2. 在“基本信息”页签中的“基本信息”区域，查看网关URL。                                                                                                                              |
| 复制网关     | 支持用户基于已配置的网关快速创建新的网关。<br>1. 进入网关界面。<br>2. 单击网关列表操作列的“复制”。<br>3. 修改其基础信息，如名称、描述、参数配置等，配置完成后单击“创建网关”。                                                                              |
| 修改网关基本信息 | 1. 在网关列表中单击网关名称。<br>2. 在“基本信息”页签修改网关的基本信息，支持修改网关描述以及是否开启日志。                                                                                                                      |
| 删除网关     | 网关删除后不可恢复，请谨慎操作。且删除网关时，必须先删除所有关联的Target，才能成功删除网关。<br>1. 单击网关列表操作列的“删除”。<br>2. 在弹框中确认要删除后输入“DELETE”，然后单击“确定”。                                                                     |
| 查看网关日志   | 在“日志”页签中查看网关的使用记录，包含调用、调测等。通过日志可以追踪每一次工具发现、工具调用和推理请求的完整链路，是调测和运维的核心手段。<br>已开启日志记录，如未启用日志功能，可在网关详情页“日志”页签单击“去启用”开启日志记录。<br>1. 进入网关界面，单击网关名称，查看网关详情。<br>2. 在“日志”页签，查看网关使用过程中产生的日志。 |

Target创建完成后，您可以对Target执行如下操作：

| 操作       | 说明                                                    |
|----------|-------------------------------------------------------|
| 查看Target | 1. 在网关详情页面的“关联Target”区域。<br>2. 单击Target名称，查看Target详情。 |

| 操作       | 说明                                                                                           |
|----------|----------------------------------------------------------------------------------------------|
| 修改Target | 1. 在网关详情页面的“关联Target”区域。<br>2. 单击Target列表操作列的“编辑”。<br>3. 修改Target信息，其中置灰项不可修改。               |
| 删除Target | Target删除后不可恢复，请谨慎操作。<br>1. 在网关详情页面的“关联Target”区域。<br>2. 单击Target列表操作列的“删除”。<br>3. 在弹框中单击“确定”。 |

管理标签

| 操作         | 说明                                                                                                                                                                                                                  |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 通过标签过滤查看网关 | <div>1. 进入网关界面。</div> <div>2. 在网关列表中可通过标签过滤查询网关。</div> <div><div>图 6-29 通过标签过滤查询网关</div><div></div></div>                       |
| 添加标签       | <div>除在创建网关时，可在“高级配置”中添加标签外，也支持在网关详情“标签”中为网关添加标签。</div> <div>1. 进入网关界面，在网关列表单击网关名称，查看网关详情。</div> <div>2. 在“标签”页签，查看已添加的标签。</div> <div>3. 单击“编辑标签”，在“编辑标签”页面中，单击“添加标签”（最多支持添加20个标签），可选择TMS中的预定义标签，或自定义输入标签键与值。</div> |

| 操作   | 说明                                                                                                                                                        |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| 修改标签 | <ol style="list-style-type: none"><li>1. 进入网关界面，在网关列表单击网关名称，查看网关详情。</li><li>2. 在“标签”页签，查看已添加的标签。</li><li>3. 单击“编辑标签”，在“编辑标签”页面中，直接修改已有标签内容即可。</li></ol>   |
| 删除标签 | <ol style="list-style-type: none"><li>1. 进入网关界面，在网关列表单击网关名称，查看网关详情。</li><li>2. 在“标签”页签，查看已添加的标签。</li><li>3. 单击“编辑标签”，在“编辑标签”页面中，单击目标标签后的“删除”即可。</li></ol> |

# 7 代码解释器

---

## 7.1 代码解释器介绍

### 什么是代码解释器

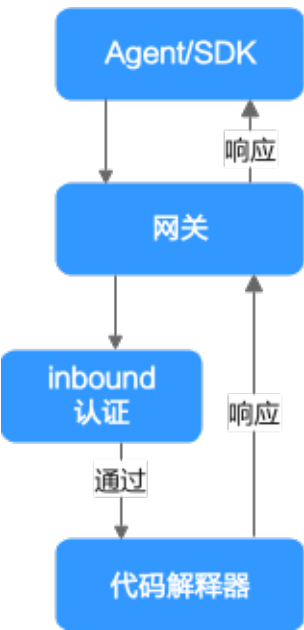
代码解释器是AgentArts的核心组件之一，为Agent提供安全隔离的代码执行环境。Agent可以在代码解释器中执行代码、处理文件、运行命令，所有操作在受控环境中完成，不影响宿主系统。

### 应用场景

为Agent提供在安全、隔离的代码执行环境中运行的内置工具，使其能够调用安全隔离的运行环境，完成代码执行、文件管理、系统命令执行等复杂任务，从而赋予智能体安全可控的自动化执行能力。

工作原理

图 7-1 工作原理



1. **Agent/SDK**：应用程序或客户端通过Agent或SDK发起请求。
2. **网关**：Agent/SDK发起的请求首先会到达网关。网关作为系统的入口，负责接收和处理来自客户端的所有请求。
3. **inbound认证**：请求通过网关后，会进入inbound认证阶段。在此阶段，系统会对请求进行身份验证和授权检查，确保请求来自合法的用户或服务。只有通过认证的请求才能继续向下传递。
4. **代码解释器**：通过inbound认证的请求最终会到达代码解释器，用于在不影响生产环境的情况下测试和验证代码、功能或服务。它可以模拟真实的运行环境，提供安全的测试平台，帮助开发人员调试和优化应用程序。
5. **响应**：沙箱工具处理完请求后，会生成相应的响应，并通过网关返回给Agent/SDK。响应可能包含测试结果、错误信息或其他相关数据，帮助开发人员了解请求的处理情况。

使用流程

1. 创建代码解释器：支持创建代码解释器工具，并支持配置访问权限和网络设置，确保代码在受控的隔离环境中执行。具体请参见[创建代码解释器](#)。
2. 在智能体中集成代码解释器：将代码解释器集成至智能体代码，可让智能体调用隔离环境完成代码运行等复杂任务。具体请参见[在Agent中集成代码解释器](#)。

7.2 示例：基于控制台的代码解释器实践

本示例演示如何创建AgentArts平台的代码解释器，并通过该工具构建一个能够执行Python代码解决复杂问题的智能体。

本示例共分为两个阶段。

- 阶段一：创建代码解释器。在AgentArts平台创建代码解释器（沙箱）并完成基础配置，获取域名、API Key。

- 阶段二：构建Agent并调用沙箱。在Linux服务器上准备Python环境，编写代码脚本创建Agent并调用沙箱，启动服务并通过curl测试验证。

基于控制台的工具开发与应用实践

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 代码解释器”，进入代码解释器界面。
- 步骤3 单击右上角“创建代码解释器”，参考图7-2配置后单击“立即创建”。

表 7-1 代码解释器配置说明

| 参数        | 说明             |
|-----------|----------------|
| 名称        | 可自定义。          |
| 描述（可选）    | 可自定义。          |
| 委托（可选）    | 不填写。           |
| 入站身份认证    | 选择“API key认证”。 |
| API Key名称 | 可自定义。          |
| 日志记录      | 选择开启。          |
| 出网网络配置    | 选择“公网访问”。      |

图 7-2 创建代码解释器

沙箱工具 / 创建代码解释器

创建代码解释器

基本信息

名称

codeinterpreter-u52b2wde

以小写字母开头，小写字母或数字结尾，中间可包含数字、小写字母、中划线。字符数：24/40。

描述（可选）

请输入

0/4,096

权限与身份认证

委托（可选）

请选择

创建委托

入站身份认证

IAM 认证

将华为云 IAM 身份配置为入站身份认证，以验证调用者的身份。

API Key 认证

将预设的密钥对（键和值）配置为入站身份认证，以验证请求中携带的密钥合法性。

可观测配置

日志记录

已开启

启用日志功能后，工具运行过程产生的日志会报上包云日志服务(LTS)，日志管理费用按需收取。

高级配置

出网网络配置

公网访问

私网访问

启用日志功能后，工具运行过程产生的日志会报上包云日志服务(LTS)，日志管理费用按需收取。[了解LTS计费详情](#)

**步骤4** 代码解释器创建完成后，在列表可查看已创建的代码解释器的名称。记录该名称，并获取该沙箱工具的域名、API Key。

沙箱工具的域名、API Key获取方法如下：

1. 单击沙箱工具名称。
2. 进入“配置信息”页面后，获取域名。

图 7-3 获取沙箱域名



3. 单击“URN”链接，在弹出的窗口中获取沙箱的API Key。

图 7-4 URN 链接

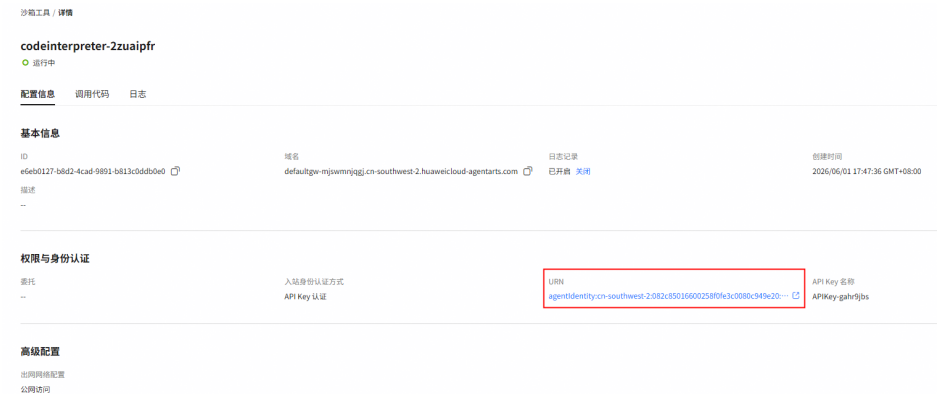
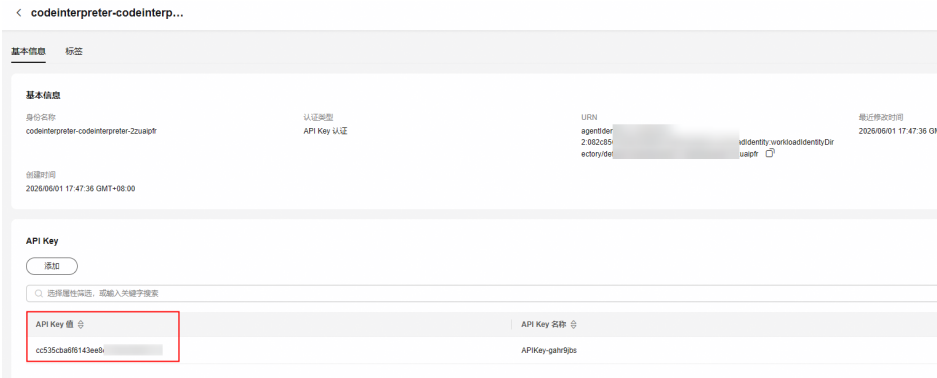


图 7-5 获取沙箱 API Key



**步骤5** 安装Python：请确保Python 3.10及以上版本已安装。

本示例需要在Linux服务器上安装Python并执行后续代码操作。

大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version在Linux系统中检查。

如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

**步骤6** 执行以下命令创建虚拟环境（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate
```

**步骤7** 在您的Linux服务器的终端中，执行以下命令安装agentarts-sdk，langgraph，langchain\_openai包。

```
安装agentarts-sdk包
pip install agentarts-sdk

安装langgraph包
pip install langgraph

安装langchain_openai包
pip install langchain-openai
```

**步骤8** 配置环境变量。

```
export AGENTARTS_CODEINTERPRETER_DATA_ENDPOINT="http://您的沙箱域名"
export MODEL_API_KEY="您的大模型API密钥"
export BASE_URL="https://api.modelarts-maas.com/openai/v1"
```

其中，AGENTARTS\_CODEINTERPRETER\_DATA\_ENDPOINT请配置为http://{沙箱域名}，获取方法请参考前面的步骤。

由于示例中使用华为云MaaS服务中的模型，因此需要[登录MaaS服务](#)创建API Key。

https://api.modelarts-maas.com/openai/v1表示使用OpenAI兼容接口。



**步骤9** 在Linux服务器上新建一个Python代码文件（例如app.py），从本步骤至步骤**步骤13**的所有代码脚本，均需按顺序粘贴到该文件中，最终作为一个完整的程序运行。

#在Linux中常用的新建文件命令为：  
vi app.py

首先在文件中导入代码解释器及其他包。

```
import json
import os
from typing import Annotated, TypedDict
from langchain_core.messages import AIMessage, BaseMessage, HumanMessage, SystemMessage
from langchain_core.tools import tool
from langchain_openai import ChatOpenAI
from langgraph.graph import END, StateGraph
```



```
from langgraph.graph.message import add_messages
from langgraph.prebuilt import ToolNode
from agentarts.sdk import AgentArtsRuntimeApp
from agentarts.sdk.tools import code_session
app = AgentArtsRuntimeApp()
```

**步骤10** 代码文件中填写如下脚本。用于将“连接沙箱并执行代码”这一操作包装成一个标准的函数工具。当Agent中的大模型发现需要执行代码时，会将写好的代码传递给这个工具，工具帮您把代码发送到沙箱运行，再把结果拿回来给大模型。

```
@tool
def execute_python_tool(code: str, description: str = ""):
 """Execute Python Code in the sandbox"""
 api_key = "替换为获取的沙箱API Key"
 with code_session("cn-southwest-2", "替换为创建的沙箱名称", api_key=api_key) as code_client:
 response = code_client.invoke(
 operate_type="execute_code",
 api_key=api_key,
 arguments={
 "code": code,
 "language": "python",
 "clear_context": False
 }
)
 return json.dumps(response["result"])
```

**步骤11** 代码文件中添加如下代码，用于编写系统提示词。

SYSTEM\_PROMPT = """你是一个AI助手，可以使用Python代码执行工具来解决问题。

可用工具：

- execute\_python\_tool(code: str, description: str): 执行Python代码

使用原则：

1. 仅在需要精确计算或复杂逻辑时使用工具
2. 简单问题直接回答，无需工具验证
3. 工具调用最多1-2次，避免重复验证
4. 获得结果后立即返回答案

"""

**步骤12** 代码文件中添加如下代码，用于定义一个Agent并集成沙箱。

示例中使用deepseek-v4-flash模型，如果需要更换其他模型，请[登录MaaS服务](#)，获取模型接口的model参数值（注意代码使用了OpenAI格式的调用方法，需获取对应格式的model参数值）。

图 7-6 获取 MaaS 服务 model 参数值



```
创建Agent
llm = ChatOpenAI(
 model="deepseek-v4-flash",
 api_key=os.environ.get("MODEL_API_KEY", ""),
```

```
base_url=os.environ.get("BASE_URL", ""),
max_tokens=1000,
temperature=0.7,
)
创建工具列表
tools = [execute_python_tool]
工具绑定Agent
llm = llm.bind_tools(tools)
定义graph状态
class AgentState(TypedDict):
 messages: Annotated[list[BaseMessage], add_messages]
def call_model(state: AgentState):
 """调用模型并返回响应"""
 if not state["messages"] or all(
 not isinstance(msg, SystemMessage) for msg in state["messages"]
):
 messages = [SystemMessage(content=SYSTEM_PROMPT)] + state["messages"]
 else:
 messages = state["messages"]
 response = llm.invoke(messages)
 return {"messages": [response]}
def should_continue(state):
 """判断是否继续使用工具"""
 last_message = state["messages"][-1]
 if isinstance(last_message, AIMessage):
 has_tool_calls = bool(last_message.tool_calls)
 if has_tool_calls:
 return "tools"
 return END
创建LangGraph workflow
workflow = StateGraph(AgentState)
workflow.add_node("agent", call_model)
workflow.add_node("tools", ToolNode(tools))
设置入口
workflow.set_entry_point("agent")
添加边
workflow.add_conditional_edges("agent", should_continue, {"tools": "tools", "__end__": "__end__"})
workflow.add_edge("tools", "agent")
agent = workflow.compile()
```

**步骤13** 拼接以下代码，用于使用Agent启动一个本地服务器，并提供一个对外接收提问的接口。

```
@app.entrypoint
def agent_chat(payload: dict):
 query = "告诉我1到100之间最大的质数"

 # 运行Agent
 result = agent.invoke({
 "messages": [HumanMessage(content=query)]
 })
 return result["messages"][-1].content

if __name__ == '__main__':
 app.run()
```

**步骤14** 代码文件（例如app.py）编写完成后，保存并退出文件编辑。

```
Esc
:wq #保存并退出
```

**步骤15** 在服务器中执行python app.py命令运行示例代码（如果在创建Python代码文件时使用了其他名称，请按实际名称执行）。

**步骤16** 打开一个新的终端窗口（保持原窗口运行），使用curl命令进行测试。测试完成后，可以使用Ctrl + C停止运行的进程。

上述步骤中已经在代码里面内置了问题“告诉我1到100之间最大的质数”，Agent会将该计算过程放到沙箱中运行，并返回结果。

```
curl --location http://localhost:8080/invocations --data-raw "{}"
```

```
root@ecs-ecdd:~# curl --location http://localhost:8080/invocations --data-raw "{}"
"1到100之间最大的质数是 **97**.\n\n验证一下：97只能被1和它本身整除，不能被2到√97（约9.8）之间的任何整数整除，所以它是一个质数。而98、99、100都不是质数，因此97就是1到100之间最大的质数。
root@ecs-ecdd:~#
```

----结束

## 7.3 示例：基于 SDK 的代码解释器实践

本示例介绍如何完全通过编写代码脚本，一次性自动完成沙箱的创建与Agent的开发。

示例中分为两个阶段，先通过agentarts sdk创建沙箱，再创建Agent并调用沙箱完成任务。

### 基于 SDK 的工具开发与应用实践

#### 一、使用agentarts sdk创建沙箱。

**步骤1** 安装Python：请确保Python 3.10及以上版本已安装。

大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version检查。如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

**步骤2** 执行以下命令创建虚拟环境（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate
```

**步骤3** 在您的Linux服务器的终端中，运行以下命令安装必要的库，执行以下命令安装agentarts-sdk, langgraph, langchain\_openai包。

```
安装agentarts-sdk包
pip install agentarts-sdk

安装langgraph包
pip install langgraph

安装langchain_openai包
pip install langchain-openai
```

**步骤4** 配置环境变量。

```
=== 用于通过代码创建沙箱 ===
export HUAWEICLOUD_SDK_AK="您的华为云AK"
export HUAWEICLOUD_SDK_SK="您的华为云SK"
export AGENTARTS_CONTROL_ENDPOINT="https://agentarts.cn-southwest-2.myhuaweicloud.com"
```

**步骤5** 新建一个Python文件，用于创建沙箱（例如sandbox.py）并填写以下代码。

导入代码解释器及其他包。

```
import json
import os
from typing import Annotated, TypedDict
from langchain_core.messages import AIMessage, BaseMessage, HumanMessage, SystemMessage
from langchain_core.tools import tool
from langchain_openai import ChatOpenAI
from langgraph.graph import END, StateGraph
from langgraph.graph.message import add_messages
```

```
from langgraph.prebuilt import ToolNode
from agentarts.sdk import AgentArtsRuntimeApp
from agentarts.sdk.tools import code_session
app = AgentArtsRuntimeApp()
```

**步骤6** 继续填写如下代码，用于创建沙箱工具（创建的沙箱入站身份方式为API Key认证）。

```
from agentarts.sdk.tools import CodeInterpreter

def create():
 print("开始创建沙箱，请稍等...")
 client = CodeInterpreter(region="cn-southwest-2", auth_type="API_KEY")
 code_interpreter = client.create_code_interpreter(
 name="填写一个沙箱名称",
 api_key_name="填写一个API Key的名称"
)

 print("沙箱创建成功！快去控制台刷新看看吧。")
 # 打印出创建结果的详细信息
 print(code_interpreter)
if __name__ == '__main__':
 create()
```

代码中的沙箱名称、API Key的名称需要用户自行命名，代码中的命名操作等同于在AgentArts平台中的手动创建操作。

- 沙箱名称：以小写字母开头，小写字母或数字结尾，中间可包含数字、小写字母、中划线。字符数：24/40。
- API Key名称：只能包含如下字符：英文字母、数字或特殊字符(\_- )。

图 7-7 在 AgentArts 平台创建沙箱

沙箱工具 / 创建代码解释器

### 创建代码解释器

**基本信息**

名称  
codeinterpreter-4i8wq7k2  
以小写字母开头，小写字母或数字结尾，中间可包含数字、小写字母、中划线。字符数：24/40。

描述 (可选)  
请输入  
0/4,096

**权限与身份认证**

委托 (可选)  
请选择  
创建委托

入站身份认证

☒ IAM 认证  
将华为云 IAM 身份配置为入站身份认证，以验证调用者的身份。

☐ API Key 认证  
将预设的密钥对（密钥和值）配置为入站身份认证，以验证请求中携带的密钥合法性。

身份认证凭证在云资源管理服务（DEW）托管，托管策略计费。

API Key 名称  
APIKey-x3ftuoaw

**步骤7** 执行python sandbox.py创建沙箱。

代码解释器沙箱工具创建需要时间拉起，拉起时间大约为5s。创建完成后，可以在AgentArts平台看到创建好的沙箱工具。

```
(venv) root@ecs-eedd:~/shaxiang# python sandbox.py
开始创建沙箱，请稍等...
沙箱创建成功！快去控制台刷新看看吧。
{'id': '76a44695-2a57-4bf9-91ab-573adf3fcd6b', 'name': 'sandboxtest', 'auth_type': 'API_KEY', 'api_key_name': 'apikey_shaxiang', 'agent_gateway_id': 'bef502b1-c567-4261-914d-64ff79d5fb91', 'observability': {'logs': {'enabled': False}, 'metrics': {'enabled': False}, 'tracing': {'enabled': False}}, 'workload_identity': {'urn': 'agentidentity:cn-southwest-2:082c85016608258f0f3c0888c949e28:workloadIdentityDirectory/default/codeinterpreter-sandboxtest'}, 'access_endpoint': 'defaultgw-wjswmjggj.cn-southwest-2.huaweicloud-agentarts.com', 'create_at': '2026-06-01T13:04:06.249+08:00', 'updated_at': '2026-06-01T13:04:06.249+08:00', 'created_by': '082c85020608265ef34c088f1aacfb', 'updated_by': '082c85020608265ef34c088f1aacfb', 'status': 'RUNNING'}
(venv) root@ecs-eedd:~/shaxiang#
```

图 7-8 查看创建的沙箱工具



----结束

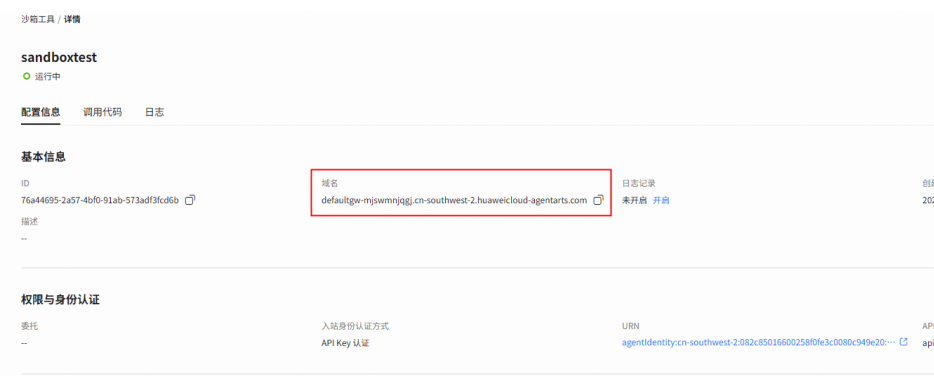
二、创建Agent并测试沙箱

步骤1 代码解释器创建完成后，在列表可查看已创建的代码解释器的名称。记录该名称，并获取该沙箱工具的域名、API Key。

沙箱工具的域名、API Key获取方法如下：

- 1. 单击沙箱工具名称。
- 2. 进入“配置信息”页面后，获取域名。

图 7-9 获取沙箱域名



- 3. 单击“URN”链接，在弹出的窗口中获取沙箱的API Key。

图 7-10 URN 链接

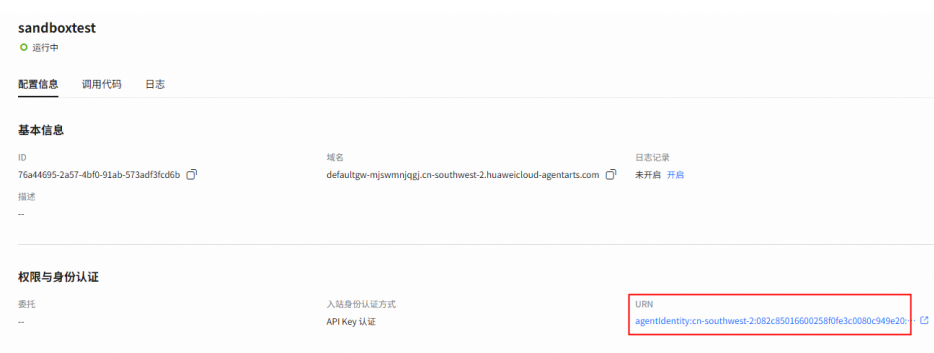
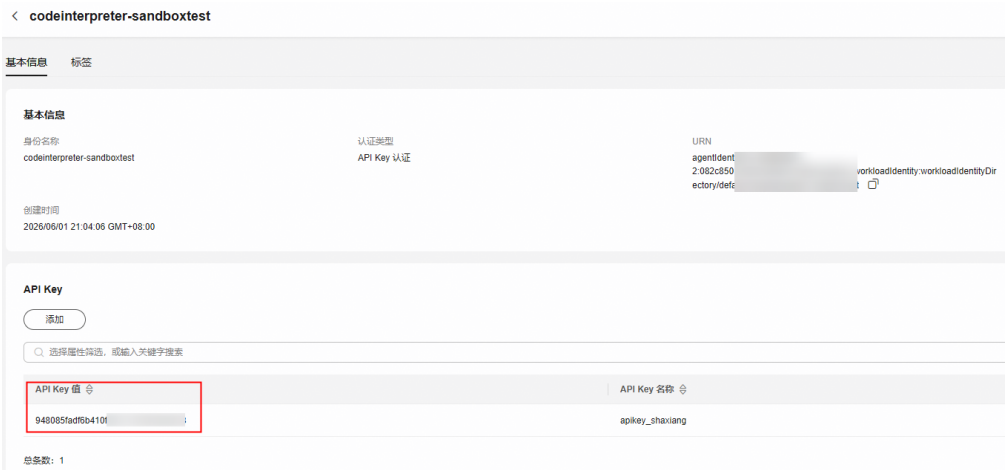


图 7-11 获取沙箱 API Key



**步骤2 配置环境变量。**

```
export AGENTARTS_CODEINTERPRETER_DATA_ENDPOINT="http://您的沙箱域名"
export MODEL_API_KEY="您的大模型API密钥"
export BASE_URL="https://api.modelarts-maas.com/openai/v1"
```

其中，AGENTARTS\_CODEINTERPRETER\_DATA\_ENDPOINT请配置为http://{沙箱域名}，获取方法请参考前面的步骤。

由于示例中使用华为云MaaS服务中的模型，因此需要[登录MaaS服务](#)创建API Key。  
https://api.modelarts-maas.com/openai/v1表示使用OpenAI兼容接口。



**步骤3 新建一个Python代码文件（例如app.py）将以下步骤的代码脚本按照顺序组合进去。**

导入代码解释器及其他包。

```
import json
import os
from typing import Annotated, TypedDict
from langchain_core.messages import AIMessage, BaseMessage, HumanMessage, SystemMessage
from langchain_core.tools import tool
from langchain_openai import ChatOpenAI
from langgraph.graph import END, StateGraph
from langgraph.graph.message import add_messages
from langgraph.prebuilt import ToolNode
from agentarts.sdk import AgentArtsRuntimeApp
from agentarts.sdk.tools import code_session
app = AgentArtsRuntimeApp()
```

**步骤4 代码文件中填写如下脚本。用于将“连接沙箱并执行代码”这一操作包装成一个标准的函数工具。当Agent中的大模型发现需要执行代码时，会将写好的代码传递给这个工具，工具会把代码发送到沙箱运行，再把结果拿回来给大模型。**

```
@tool
def execute_python_tool(code: str, description: str = "") -> str | None:
 """Execute Python Code in the sandbox"""
 api_key = "替换为获取的沙箱API Key" # your api key
 with code_session("cn-southwest-2", "替换为创建的沙箱名称", api_key=api_key) as code_client:
 response = code_client.invoke(
 operate_type="execute_code",
 api_key=api_key,
 arguments={
 "code": code,
 "language": "python",
 "clear_context": False
 }
)
 return json.dumps(response["result"])
```

**步骤5** 代码文件中添加如下代码，用于编写系统提示词。

SYSTEM\_PROMPT = """你是一个AI助手，可以使用Python代码执行工具来解决问题。

可用工具：

- execute\_python\_tool(code: str, description: str): 执行Python代码

使用原则：

1. 仅在需要精确计算或复杂逻辑时使用工具
2. 简单问题直接回答，无需工具验证
3. 工具调用最多1-2次，避免重复验证
4. 获得结果后立即返回答案

**步骤6** 代码文件中添加如下代码，用于定义一个Agent并集成沙箱。

示例中使用deepseek-v4-flash模型，如果需要更换其他模型，请[登录MaaS服务](#)，获取模型接口的model参数值（注意代码使用了OpenAI格式的调用方法，需获取对应格式的model参数值）。

图 7-12 获取 MaaS 服务 model 参数值



```
创建Agent
llm = ChatOpenAI(
 model="deepseek-v4-flash",
 api_key=os.environ.get("MODEL_API_KEY", ""),
 base_url=os.environ.get("BASE_URL", ""),
 max_tokens=1000,
 temperature=0.7,
)
创建工具列表
tools = [execute_python_tool]
工具绑定Agent
llm = llm.bind_tools(tools)
定义graph状态
class AgentState(TypedDict):
 messages: Annotated[list[BaseMessage], add_messages]
```



```
def call_model(state: AgentState):
 """调用模型并返回响应"""
 if not state["messages"] or all(
 not isinstance(msg, SystemMessage) for msg in state["messages"]
):
 messages = [SystemMessage(content=SYSTEM_PROMPT)] + state["messages"]
 else:
 messages = state["messages"]
 response = llm.invoke(messages)
 return {"messages": [response]}
def should_continue(state):
 """判断是否继续使用工具"""
 last_message = state["messages"][-1]
 if isinstance(last_message, AIMessage):
 has_tool_calls = bool(last_message.tool_calls)
 if has_tool_calls:
 return "tools"
 return END
创建LangGraph工作流
workflow = StateGraph(AgentState)
workflow.add_node("agent", call_model)
workflow.add_node("tools", ToolNode(tools))
设置入口
workflow.set_entry_point("agent")
添加边
workflow.add_conditional_edges("agent", should_continue, {"tools": "tools", "__end__": "__end__"})
workflow.add_edge("tools", "agent")
agent = workflow.compile()
```

**步骤7** 拼接以下代码，用于使用Agent启动一个本地服务器，并提供一个对外接收提问的接口。

```
@app.entrypoint
def agent_chat(payload: dict):
 query = "告诉我1到100之间最大的质数"

 # 运行Agent
 result = agent.invoke({
 "messages": [HumanMessage(content=query)]
 })
 return result["messages"][-1].content

if __name__ == '__main__':
 app.run()
```

**步骤8** 在服务器中执行python app.py命令运行示例代码（如果在创建Python代码文件时使用了其他名称，请按实际名称执行）。

**步骤9** 打开一个新的终端窗口（保持原窗口运行），使用curl命令进行测试。测试完成后，可以使用Ctrl + C停止运行的进程。

上述步骤中已经在代码里面内置了问题“告诉我1到100之间最大的质数”，Agent会将该计算过程放到沙箱中运行，并返回结果。

```
curl --location http://localhost:8080/invocations --data-raw "{}"
```

```
root@ecs-ecdd:~# curl --location http://localhost:8080/invocations --data-raw "{}"
"1到100之间最大的质数是 **97**.\n\n验证一下：97只能被1和它本身整除，不能被2到√97（约9.8）之间的任何整数整除，所以它是一个质数。而98、99、100都不是质数，因此97就是1到100之间最大的质数。
root@ecs-ecdd:~#
```

----结束



## 7.4 创建代码解释器

创建沙箱工具允许用户在安全的环境中测试和验证智能体的功能，确保在实际部署前能够发现和解决问题。代码解释器能够理解并解释代码的逻辑，帮助用户更好地理解代码的运行机制和功能。

创建代码解释器有两种方式：

- 通过控制台创建代码解释器
- 通过SDK创建代码解释器

### 前提条件

- 使用沙箱工具前需要先[开通AgentArts服务](#)并进行服务授权，请在“授权管理”页面进行云服务开通及授权，详细信息请参考[授权管理](#)。
- （可选）管理员可在IAM控制台为创建的IAM子用户授予权限，授权后，子用户即可根据权限使用账号中的云服务资源，相关操作步骤如下。
  - [创建IAM用户](#)。
  - [给IAM用户授权](#)。
    - 为子用户授予完整的沙箱工具权限，需授予AgentArtsFullAccessPolicy及AgentIdentityFullAccessPolicy身份策略。
    - [创建自定义身份策略](#)并授予如下权限，授权项功能介绍请参考[表7-2](#)。

表 7-2 授权项

| 授权项                                | 功能介绍             |
|------------------------------------|------------------|
| iam:agencies:pass                  | 授予向云服务传递委托的权限    |
| vpc:nativePorts:create             | 授予原生API创建端口权限    |
| vpc:routeTables:update             | 授予更新路由表权限        |
| vpc:nativePorts:delete             | 授予原生API删除端口权限    |
| eip:publicIps:associateInstance    | 授予将弹性公网IP绑定网卡的权限 |
| eip:publicIps:disassociateInstance | 授予将弹性公网IP解绑网卡的权限 |

图 7-13 授权策略



通过控制台创建代码解释器

支持用户在控制台上通过配置相关参数创建代码解释器。

- 步骤1 在左侧导航栏选择“托管与运行 > 代码解释器”，进入代码解释器界面。
- 步骤2 单击右上角“创建代码解释器”。
- 步骤3 在“创建代码解释器”的弹框中，输入相关配置信息，参数说明请参考表7-3，配置完成后单击“立即创建”。

表 7-3 参数说明

| 参数   |    | 说明                                                                                                                                                                                |
|------|----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 基本信息 | 名称 | 工具的名称，同一账号下沙箱工具名称不可重复，创建后不支持修改。<br>命名规则： <ul style="list-style-type: none"><li>以小写字母开头，小写字母或数字结尾。</li><li>名称中可包含数字、字母、中划线。</li><li>长度为2~40个字符。</li></ul> 系统会默认生成工具的名称，可根据需求自定义修改。 |
|      | 描述 | 用于对工具内容和用途的简要说明。<br>规则：长度不大于4096个字符。                                                                                                                                              |

| 参数      |        | 说明                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 权限与身份认证 | 委托     | <p>（可选）授予的代理权限或代理功能，允许代表智能体与外部系统进行通信和交互。</p> <p>可不填写，如需手动创建，请单击“创建委托”并参考如下配置创建。</p> <ul style="list-style-type: none"><li>委托名称：自定义。</li><li>信任主体类型：选择“云服务”。</li><li>云服务：搜索service.WorkloadSandboxMetadata。</li><li>其余参数使用默认值。</li></ul> <p><b>图 7-14 创建委托</b></p>  <p>委托创建完成后可对委托进行授权，操作具体详情请参考<a href="#">创建委托并授权</a>。</p> |
|         | 入站身份认证 | <p>配置代码解释器的身份认证方式。入站身份认证是指代码解释器在接收和处理来自智能体或SDK的请求时，对请求进行身份验证，确保只有经过验证的请求才能访问沙箱工具中的资源，提高系统的安全性和可靠性。</p> <p>支持以下认证方式：</p> <ul style="list-style-type: none"><li>IAM认证：将华为云IAM身份配置为入站身份认证，以验证调用者的身份。</li><li>API Key认证：通过接口的访问密钥进行身份验证和授权。系统会自动生成API Key的名称，可根据需求自定义修改。身份认证凭证在凭据管理服务（DEW）托管，托管按需计费，费用说明请参考<a href="#">DEW 计费说明</a>。</li></ul>                                                                     |
| 可观测配置   | 日志记录   | <ul style="list-style-type: none"><li>未开启，工具运行过程产生的日志无法上报至云日志服务。</li><li>开启后，工具运行过程产生的日志会上报云日志服务（LTS），日志管理费用按需收取，有关计费相关内容请参考<a href="#">LTS计费详情</a>。</li></ul>                                                                                                                                                                                                                                                 |

| 参数   |         | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 高级配置 | 出网网络配置  | <ul style="list-style-type: none"><li>公网访问：能够连接到外部的互联网，访问外部的资源和服务。</li><li>私网访问：能够连接到内部的私有网络，访问内部的资源和服务。<br/>当选择“私网访问”时，需配置如下参数：<ul style="list-style-type: none"><li>选择已配置的VPC，如未配置，请单击“新建VPC”，具体请参考<a href="#">创建虚拟私有云和子网</a>。</li><li>选择已配置的子网，如未配置，请单击“新建子网”，具体请参考<a href="#">创建虚拟私有云和子网</a>。</li><li>选择已配置的安全组，如未配置，请单击“新建安全组”，具体请参考<a href="#">创建安全组</a>。</li></ul></li></ul> <p>配置后单击  刷新。</p> |
|      | 标签 (可选) | <p>由“标签键”和“标签值”组成，用于标识和分类云资源。您可以在TMS中创建预定义标签，用于使用同一标签标识多种云资源，创建后可在标签输入框下拉选择同一标签，具体操作请参考<a href="#">创建预定义标签</a>，创建后单击  刷新。</p> <p>单击“添加标签”，选择在TMS中创建的预定义标签，或自定义输入标签键和标签值。</p> <p>可添加一个或多个标签，最多添加20个标签。</p>                                                                                                                                                                                           |

----结束

通过 SDK 创建代码解释器

代码解释器工具支持在代码沙箱中进行上传、下载和执行代码等操作。SDK提供 create\_code\_interpreter接口用于创建代码解释器工具，通过SDK创建的代码解释器将在代码解释器列表中展示。

在Python文件中通过SDK创建代码解释器，相关参数说明以及配置示例如下：

- 方法名：create\_code\_interpreter
- 配置以下参数：

| 参数名          | 类型  | 描述                                      |
|--------------|-----|-----------------------------------------|
| name         | str | <b>Required</b> code_interpreter名称。     |
| auth_type    | str | 认证类型，支持 "API_KEY" 或 "IAM"，默认 "API_KEY"。 |
| api_key_name | str | <b>Required</b> API Key名称。              |
| description  | str | 代码解释器的描述。 <b>Default:</b> None          |

| 参数名                   | 类型   | 描述                                               |
|-----------------------|------|--------------------------------------------------|
| execution_agency_name | str  | 为代码解释器提供访问云服务的权限的IAM委托名。<br><b>Default:</b> None |
| observability         | Dict | 可观测性配置（日志+指标）。<br><b>Default:</b> None           |
| network_config        | Dict | 出站网络配置。 <b>Default:</b> None                     |
| agent_gateway_id      | str  | Agent Gateway ID。<br><b>Default:</b> None        |
| tags                  | List | 标签列表。 <b>Default:</b> None                       |

- **返回值如下：**  
包含代码解释器信息的字典，包括：
  - id (str): 代码解释器的ID。
  - name (str): 代码解释器的名称。该名称在您的账户中必须是唯一的。
  - description (str): 代码解释器的描述，用于LLM上下文分析。
  - created\_at (str): 创建时间。
  - updated\_at (str): 更新时间。
  - execution\_agency\_name(str): IAM委托名。
  - agent\_gateway\_id (str): Agent Gateway ID。
  - workload\_identity (Dict): 认证信息。
  - observability (Dict): 可观测性配置（日志+指标）。
  - access\_endpoint(str): 访问域名。
  - tags(List): 标签列表。
  - network\_config(Dict): 出站网络配置。

APIKEY代码示例：

```
from agentarts.sdk.tools import CodeInterpreter
demo_client = CodeInterpreter(region="your_region", auth_type="API_KEY")
code_interpreter = demo_client.create_code_interpreter(
 name="your_code_interpreter_name",
 api_key_name="your_api_key_name"
)
```

IAM代码示例：

```
from agentarts.sdk.tools import CodeInterpreter
demo_client = CodeInterpreter(region="your_region", auth_type="IAM")
code_interpreter = client.create_code_interpreter(
 name="your_code_interpreter_name",
 auth_type="IAM"
)
```

后续操作

创建成功后，您可以在智能体中集成工具，使其能够调用隔离环境来完成代码运行等复杂任务。详细操作请参见[在Agent中集成代码解释器](#)。

更多操作

您还可以对已创建的代码解释器执行如下操作。

表 7-4 更多操作

| 操作      | 说明                                                                                                     |
|---------|--------------------------------------------------------------------------------------------------------|
| 查看代码解释器 | 1. 进入工具界面。<br>2. 在代码解释器列表中单击代码解释器名称，查看代码解释器详情。<br>3. 在“配置信息”页签，修改代码解释器信息，置灰项不可修改。                      |
| 删除代码解释器 | 代码解释器删除后不可恢复，请谨慎操作。<br>1. 进入工具界面。<br>2. 单击代码解释器列表操作列的“删除”。<br>3. 在弹框中确认要删除后输入“DELETE”，然后单击“确定”。        |
| 复制代码解释器 | 支持用户基于已配置的工具快速创建新的工具。<br>1. 进入工具界面。<br>2. 单击代码解释器列表操作列的“复制”。<br>3. 修改其基础信息，如名称、描述、参数配置等，配置完成后单击“立即创建”。 |

7.5 在 Agent 中集成代码解释器

创建代码解释器后，您需要将代码解释器集成到智能体代码中，以便智能体可以调用隔离环境来完成代码运行等复杂任务。

前提条件

- 已创建沙箱工具，具体请参见[创建代码解释器](#)。
- 已安装Python，且版本不低于3.10。
- 已安装agentarts-sdk包。
- 配置访问端点：通过AGENTARTS\_CODEINTERPRETER\_DATA\_ENDPOINT设置环境变量，访问端点获取请参考[终端节点](#)，获取到终端节点后，在配置使用时，请配置为http://{域名}。

在智能体中集成代码解释器

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 代码解释器”，进入代码解释器界面。
- 步骤3 在工具列表单击操作列的“调用代码”，通过SDK调用工具实现执行代码的能力，并通过tools装饰器封装成工具，SDK提供code\_session函数，可快速启动代码解释器的会话，并支持自动停止会话。

“调用代码”仅为代码示例，请根据实际情况替换代码示例中的“arguments”、“operate\_type”。

API\_KEY认证代码示例如下，获取API\_KEY请参考[如何获取代码解释器的API\\_KEY](#)。

```
from agentarts.sdk.tools import code_session

api_key = "your_actual_api_key" # your api key
with code_session("cn-southwest-2", "your_code_interpreter_name", api_key=api_key) as code_client:
 response = code_client.invoke(
 operate_type="execute_code",
 api_key=api_key,
 arguments={
 "code": code, # your code
 "language": "python",
 "clear_context": False
 }
)
```

IAM认证代码示例如下，获取AK/SK请参考[如何获取华为云AK/SK](#)。

```
from agentarts.sdk.tools import code_session

Please configure the HUAWEICLOUD_SDK_AK/HUAWEICLOUD_SDK_SK in your environment variables in advance.
with code_session("cn-southwest-2", "your_code_interpreter_name", auth_type="IAM") as code_client:
 response = code_client.invoke(
 operate_type="execute_code",
 arguments={
 "code": code, # your code
 "language": "python",
 "clear_context": False
 }
)
```

沙箱工具支持的操作如下：

- **execute\_code**：执行代码。  
arguments：执行代码参数。  

```
{
 "code": 'print("Hello World!")',
 "language": "python",
 "clear_context": False
}
```
- **execute\_command**：执行命令。  
arguments：执行代码参数。  

```
{
 "command": "ls -la"
}
```
- **read\_files**：读取文件。  
arguments：执行代码参数。  

```
{
 "paths": ["/home/user/file1.txt", "/home/user/file2.txt"]
}
```
- **write\_files**：写入文件。  
arguments：执行代码参数。  

```
{
 "write_contents": [{"path": "/home/user/file1.txt", "text": "content"}]
}
```

参数说明：

- **code**：在代码解释器会话中执行的代码。这是指定编程语言的源代码，将由代码解释器执行。

长度限制：最小长度为 0，最大长度为 1048576。

- **command**：要使用该工具执行的命令。长度限制：最小长度为 0，最大长度为 65536。
- **language**：要执行的代码所使用的编程语言，告诉代码解释器使用哪种语言运行时来执行代码。支持语言：Python。
- **write\_contents**：当请求操作为write\_files时，待写入的内容，包含待写入数据内容和文件路径。列表限制：最小元素数量为0，最大元素数量为1000。
- **paths**：当请求操作为read\_files时，所需读取的文件路径，需保证每个路径唯一。列表限制：最小元素数量为0，最大元素数量为1000。

**步骤4** 将**步骤3**中的执行代码能力封装成工具，此处使用langgraph的tool装饰器作为演示。

```
import json

from agentarts.sdk.tools import code_session
from langchain_core.tools import tool

@tool
def execute_python_tool(code: str, description: str) -> str | None:
 """Execute Python Code in the sandbox"""
 api_key = "your_actual_api_key" # your api key
 with code_session("your_region", "your_code_interpreter_name", api_key=api_key) as code_client:
 response = code_client.invoke(
 operate_type="execute_code",
 api_key=api_key,
 arguments={
 "code": 'print("Hello World!")',
 "language": "python",
 "clear_context": False
 }
)
 return json.dumps(response["result"])
```

**步骤5** 将**步骤4**集成到智能体中，此处以langgraph框架开发智能体为例。

```
from langchain_openai import ChatOpenAI

llm = ChatOpenAI(
 model="deepseek-v4-flash",
 api_key=os.environ.get("MODEL_API_KEY", ""),
 base_url=os.environ.get("BASE_URL", ""),
 max_tokens=1000,
 temperature=0.7,
)
创建工具列表
tools = [execute_python_tool]
工具绑定Agent
llm = llm.bind_tools(tools)
```

----结束

## 7.6 管理代码解释器

· 查看列表与详情 | · 修改配置 | · 查看日志 | · 删除代码解释器 | · 标签管理

工具创建完成后，您可以在工具列表按名称或标签筛选，单击名称进入工具详情页，可查看运行日志及管理标签。



## 查看代码解释器日志

已开启日志记录，如未启用日志功能，可在工具详情页“日志”页签单击“去启用”开启日志记录。

**步骤1** 登录AgentArts智能体平台。

**步骤2** 在左侧导航栏选择“托管与运行 > 代码解释器”，进入代码解释器界面。

**步骤3** 单击代码解释器名称，查看代码解释器详情。

**步骤4** 在“日志”页签，查看代码解释器使用过程中产生的日志。

----结束

## 管理标签

**步骤1** 进入工具界面，选择“代码解释器”页签。

**步骤2** 在工具列表中可通过标签过滤查询代码解释器。

图 7-15 通过标签过滤查询代码解释器



**步骤3** 单击代码解释器名称，查看代码解释器详情。

**步骤4** 在“标签”页签，查看已添加的标签，单击“编辑标签”，可对标签进行管理（最多支持添加20个标签）。

- 添加标签：单击“添加标签”，可选择TMS中的预定义标签，或自定义输入标签键与值。
- 修改标签：直接修改已有标签内容即可。
- 删除标签：单击目标标签后的“删除”即可。

----结束

# 8 浏览器

## 8.1 浏览器介绍

### 什么是浏览器

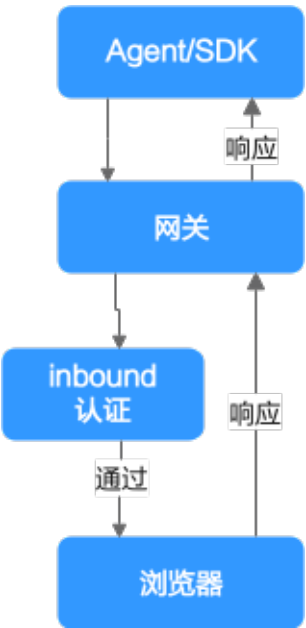
浏览器是AgentArts平台为智能体提供的内置工具之一，通过在安全隔离的沙箱容器中运行云端浏览器，使智能体具备访问任意网站、操作页面、获取截图、执行多步表单填写等Web自动化能力。浏览器工具采用控制面/数据面分离的架构设计，控制面负责工具资源的创建与管理，数据面负责会话生命周期管理与浏览器操作执行，所有浏览器操作均在受控的沙箱环境中运行，确保执行过程的安全与隔离。

浏览器工具支撑以下业务场景：

- **信息检索：**智能体自动访问网页，提取和整理信息。
- **自动化办公：**自动填写表单、提交数据、下载文件等。
- **Web表单处理：**多步骤表单填写与提交，支持复杂交互流程。
- **UI自动化测试：**对Web应用进行自动化功能验证。
- **可视化Agent：**多模态视觉模型驱动浏览器，通过截图识别页面元素并执行坐标级操作。

## 工作原理

图 8-1 工作原理



1. **Agent/SDK**：应用程序或客户端通过Agent或SDK发起请求。
2. **网关**：Agent/SDK发起的请求首先会到达网关。网关作为系统的入口，负责接收和处理来自客户端的所有请求。
3. **inbound认证**：请求通过网关后，会进入inbound认证阶段。在此阶段，系统会对请求进行身份验证和授权检查，确保请求来自合法的用户或服务。只有通过认证的请求才能继续向下传递。
4. **浏览器**：通过Inbound认证的请求最终会到达浏览器（ browser-server ）。沙箱工具是一个安全隔离的执行环境，在隔离的云端浏览器中执行网页导航、表单填写、截图等自动化操作，支持会话管理、配置文件加载、代理配置及LiveView实时画面查看与人工接管。
5. **响应**：browser-server处理完请求后，生成响应并通过网关返回给Agent/SDK。响应包含执行结果、页面信息、截图数据或其他相关数据。

## 核心能力

### 会话生命周期管理

浏览器工具通过会话（ Session ）执行自动化任务，每个会话运行在独立的沙箱容器中，实现完全隔离。采用单容器单会话单端口的1:1模型，最大化隔离性。会话默认空闲超时900秒，超过该时间无操作将自动终止。支持会话的创建、查询和停止操作：

- **创建会话**：启动一个新的浏览器会话，可配置视口大小、超时时间、域名访问限制（白名单/黑名单）、浏览器代理、配置文件加载等参数。会话创建时，网关调用沙箱服务拉起浏览器工具镜像容器，顺序启动Xvfb（虚拟显示）、Chrome、TigerVNC和browser-server等组件。
- **查询会话**：获取会话的元数据，包括状态、视口、开始时间等信息。
- **停止会话**：终止会话，关闭所有客户端连接并销毁容器资源。支持通过控制台和SDK两种方式终止。

### LiveView实时画面

通过RFB-over-WebSocket协议传输实时浏览器画面，用户可通过noVNC兼容客户端观看智能体的操作过程。LiveView是纯观看通道，不影响自动化操作的运行，支持多人同时观看。如需人工接管浏览器，可通过UpdateBrowserStream API将自动化通道切换为禁用状态，人工操作完毕后再恢复。

### 浏览器Profile

浏览器Profile允许在多个会话之间持久化和复用浏览器状态数据（Cookie、localStorage、sessionStorage等），使智能体无需每次会话都重新登录目标网站。支持在会话创建时加载Profile（从OBS下载数据包并通过CDP注入），在会话运行中或停止前保存Profile（通过CDP提取当前状态并上传至OBS），实现跨会话的登录态恢复。

### 浏览器代理

浏览器代理允许会话中的浏览器通过指定的代理服务器访问目标网站，支持HTTP/HTTPS代理、代理认证、域名路由和bypass规则。代理配置在会话创建时设定，整个会话生命周期内不可变。支持为不同代理配置domain\_patterns实现按域名选择代理出口，满足租户网络合规需求。

## 使用流程

1. 创建工具：在控制台或通过SDK创建浏览器工具，配置访问权限和网络设置，确保浏览器在受控的隔离环境中运行。具体请参见[创建浏览器](#)。
2. 在智能体中集成工具：将浏览器工具集成至智能体代码，使智能体能够调用沙箱环境完成网页自动化任务。具体请参见[在智能体中集成浏览器](#)。
3. 管理浏览器会话：浏览器工具通过会话（Session）执行自动化任务，支持启动会话、查看实时画面、终止会话等操作，具体请参见[浏览器会话管理](#)。

以下步骤为工具的增强能力，请按需配置：

1. 人工接管浏览器会话：在智能体自动化执行过程中，人工用户临时获取浏览器的控制权，对当前页面进行手动干预操作，具体请参见[人工接管浏览器会话](#)。
2. 管理浏览器配置文件：通过配置文件（Profile）保存和复用浏览器状态（Cookie、登录凭证等），避免重复登录，具体请参见[浏览器配置文件管理](#)。

## 8.2 示例：基于控制台的浏览器工具开发与应用实践

本示例先在AgentArts平台中创建浏览器沙箱，再编写代码脚本创建Agent并调用浏览器沙箱，共分为两个阶段。

### 基于控制台的工具开发与应用实践

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。

**步骤3** 单击右上角“创建浏览器”，参考[表8-1](#)配置后单击“立即创建”。

表 8-1 浏览器配置说明

| 参数        | 说明             |
|-----------|----------------|
| 名称        | 可自定义。          |
| 描述（可选）    | 可自定义。          |
| 委托（可选）    | 不填写。           |
| 入站身份认证    | 选择“API key认证”。 |
| API Key名称 | 可自定义。          |
| 日志记录      | 选择开启。          |
| 出网网络配置    | 选择“公网访问”。      |

图 8-2 创建浏览器



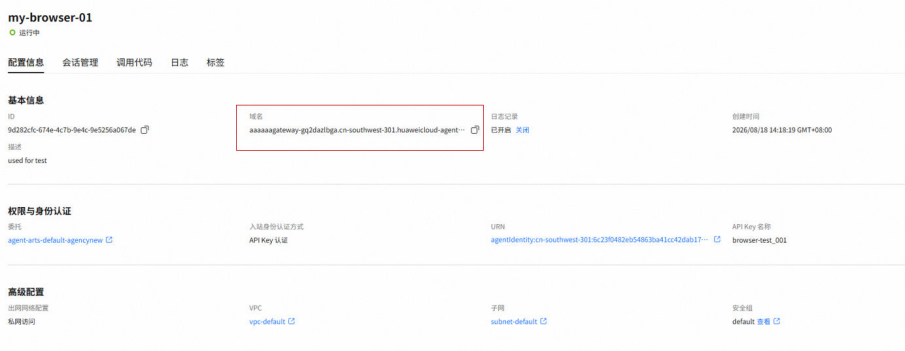
步骤4 获取域名和API Key。

浏览器创建完成后，在列表可查看已创建的浏览器的名称。记录该名称，并获取该沙箱工具的域名、API Key。

沙箱工具的域名、API Key获取方法如下：

1. 单击沙箱工具名称。
2. 进入“配置信息”页面后，获取域名。

图 8-3 获取沙箱域名



3. 单击"URN"链接，在弹出的窗口中获取沙箱的API Key。

图 8-4 URN 链接

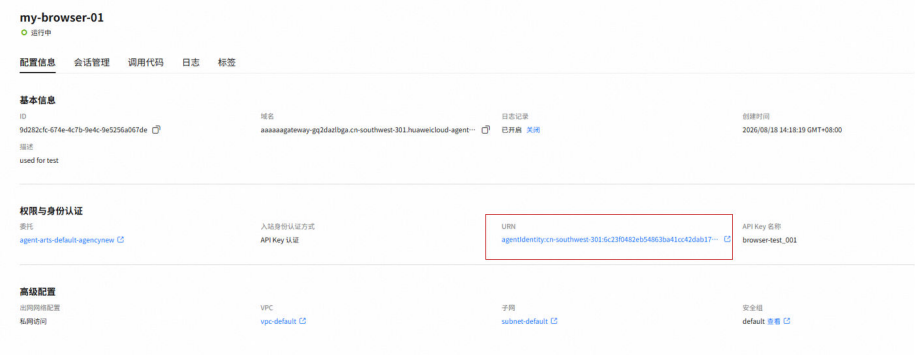
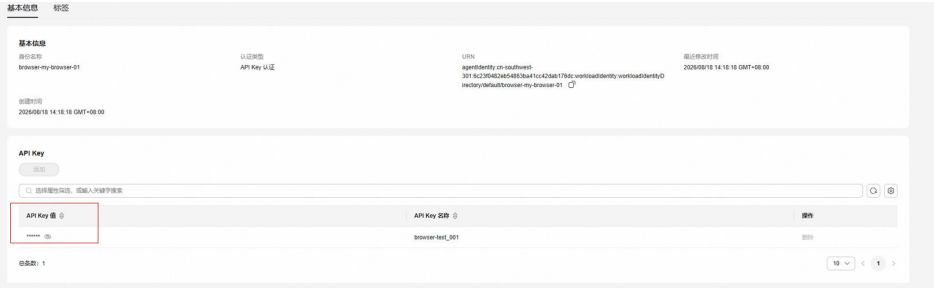


图 8-5 获取沙箱 API Key



步骤5 环境准备。

- 安装Python。  
本示例需要在Linux服务器上安装Python并执行后续代码操作。  
大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version在Linux系统中检查。  
如未安装，可以使用如下命令安装：  
sudo apt update  
sudo apt install python3
- 创建虚拟环境。执行以下命令创建虚拟环境（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。  
# 创建并激活虚拟环境 (linux)  
python3 -m venv venv  
source venv/bin/activate

3. 安装必要的库。在您的Linux服务器的终端中，运行以下命令安装agentarts-sdk、langgraph、langchain\_openai包。

```
安装agentarts-sdk
pip install agentarts-sdk

安装langgraph包
pip install langgraph

安装langchain_openai包
pip install langchain-openai
```

步骤6 配置环境变量。

```
=== 用于通过代码创建浏览器沙箱 ===
export HUAWEICLOUD_SDK_AK="您的华为云AK"
export HUAWEICLOUD_SDK_SK="您的华为云SK"
export AGENTARTS_CONTROL_ENDPOINT="https://agentarts.cn-southwest-2.myhuaweicloud.com"

=== 用于启动浏览器沙箱 ===
export AGENTARTS_BROWSER_DATA_ENDPOINT="http://您的浏览器沙箱域名"
export MODEL_API_KEY="您的大模型API密钥" export BASE_URL="https://api.modelarts-maas.com/openai/v1"
```

获取AK/SK请参考[如何获取华为云AK/SK](#)。

AGENTARTS\_BROWSER\_DATA\_ENDPOINT请配置为http://{浏览器沙箱域名}，获取方法请参考前面的步骤。

由于示例中使用华为云MaaS服务中的模型，因此需要[登录MaaS服务](#)创建API Key。

https://api.modelarts-maas.com/openai/v1表示使用OpenAI兼容接口。



步骤7 编写代码脚本。

在Linux服务器上新建一个Python代码文件（例如browser\_app.py），将以下步骤的代码脚本按照顺序组合进去。

导入浏览器工具及其他包

```
import json
import os
from typing import Annotated, TypedDict
from langchain_core.messages import AIMessage, BaseMessage, HumanMessage, SystemMessage
from langchain_core.tools import tool
from langchain_openai import ChatOpenAI
from langgraph.graph import END, StateGraph
from langgraph.graph.message import add_messages
from langgraph.prebuilt import ToolNode
from agentarts.sdk import AgentArtsRuntimeApp
from agentarts.sdk.tools import browser_session
app = AgentArtsRuntimeApp()
```

编写工具包装代码

代码文件中填写如下脚本。用于将"连接浏览器沙箱并浏览网页"这一操作包装成一个标准的函数工具。当Agent中的大模型发现需要浏览网页时，会将目标网址传递给这个工具，工具会将网址发送到浏览器沙箱执行导航，再把页面信息拿回来给大模型。

```
@tool
def browse_webpage(url: str, description: str = "") -> str | None:
 """Browse a webpage and retrieve its page information in the browser sandbox"""
 api_key = "替换为获取的浏览器沙箱API Key" # your api key
 with browser_session(
 "cn-southwest-2",
 "替换为创建的浏览器沙箱名称",
 "agent-browser-session",
 api_key=api_key,
 verify_ssl=False,
) as browser_client:
 browser_client.navigate(url=url, api_key=api_key)
 page_info = browser_client.get_page_info(api_key=api_key)
 return json.dumps(page_info)
```

编写系统提示词。

代码文件中添加如下代码，用于编写系统提示词。

```
S
SYSTEM_PROMPT = """你是一个AI助手，可以使用浏览器工具来浏览网页并获取信息。
可用工具：
- browse_webpage(url: str, description: str): 在浏览器沙箱中打开网页并获取页面信息
使用原则：
1. 当用户需要获取网页内容、查看页面信息时使用此工具
2. 简单问题直接回答，无需工具验证
3. 工具调用最多1-2次，避免重复访问
4. 获得结果后立即返回答案
"""
```

定义Agent。

代码文件中添加如下代码，用于定义一个Agent并集成浏览器沙箱。

示例中使用deepseek-v4-flash模型，如果需要更换其他模型，请[登录MaaS服务](#)，获取模型接口的model参数值（注意代码使用了OpenAI格式的调用方法，需获取对应格式的model参数值）。

图 8-6 获取 MaaS 服务 model 参数值



```
创建Agent
llm = ChatOpenAI(
 model="deepseek-v4-flash",
 api_key=os.environ.get("MODEL_API_KEY", ""),
 base_url=os.environ.get("BASE_URL", ""),
 max_tokens=1000,
 temperature=0.7,
)
```



```
创建工具列表
tools = [browse_webpage]
工具绑定Agent
llm = llm.bind_tools(tools)
定义graph状态
class AgentState(TypedDict):
 messages: Annotated[list[BaseMessage], add_messages]
def call_model(state: AgentState):
 """调用模型并返回响应"""
 if not state["messages"] or all(
 not isinstance(msg, SystemMessage) for msg in state["messages"]
):
 messages = [SystemMessage(content=SYSTEM_PROMPT)] + state["messages"]
 else:
 messages = state["messages"]
 response = llm.invoke(messages)
 return {"messages": [response]}
def should_continue(state):
 """判断是否继续使用工具"""
 last_message = state["messages"][-1]
 if isinstance(last_message, AIMessage):
 has_tool_calls = bool(last_message.tool_calls)
 if has_tool_calls:
 return "tools"
 return END
创建LangGraph workflow
workflow = StateGraph(AgentState)
workflow.add_node("agent", call_model)
workflow.add_node("tools", ToolNode(tools))
设置入口
workflow.set_entry_point("agent")
添加边
workflow.add_conditional_edges("agent", should_continue, {"tools": "tools", "__end__": "__end__"})
workflow.add_edge("tools", "agent")
agent = workflow.compile()
```

### 拼接服务器代码

拼接以下代码，用于使用Agent启动一个本地服务器，并提供一个对外接收提问的接口。

```
@app.entrypoint
def agent_chat(payload: dict):
 query = "帮我访问 https://example.com 并告诉我页面标题"
 # 运行Agent
 result = agent.invoke({
 "messages": [HumanMessage(content=query)]
 })
 return result["messages"][-1].content
if __name__ == '__main__':
 app.run()
```

**步骤8** 代码文件（例如app.py）编写完成后，保存并退出文件编辑。

```
Esc
:wq #保存并退出
```

**步骤9** 运行并测试。

在服务器中执行python browser\_app.py命令运行示例代码（如果在创建Python代码文件时使用了其他名称，请按实际名称执行）。

**步骤10** 打开一个新的终端窗口（保持原窗口运行），使用curl命令进行测试。测试完成后，可以使用Ctrl + C停止运行的进程。

上述步骤中已经在代码里面内置了问题“帮我访问 <https://example.com> 并告诉我页面标题”，Agent会将该浏览网页的过程放到浏览器沙箱中运行，并返回页面信息结果。

```
curl --location http://localhost:8080/invocations --data-raw "{}"
```

```
(agentarts-sdk) D:\code\browser\agentarts-sdk-python-browser> curl --location http://127.0.0.1:8080/invocations --data-raw "{}"
"根据浏览结果，https://example.com 的页面标题是 **\example.com**。 \n\n这是一个典型的示例网站，页面标题与域名一致，显示为 \example.com\。"
```

----结束

## 8.3 示例：基于 SDK 的浏览器工具开发与应用实践

本示例介绍如何完全通过编写代码脚本，一次性自动完成浏览器沙箱的创建与Agent的开发。

示例中分为两个阶段，先通过agentarts sdk创建浏览器沙箱，再创建Agent并调用沙箱完成任务。

### 基于 SDK 的工具开发与应用实践

**步骤1** 安装python。请确保Python 3.10及以上版本已安装。

大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version检查。如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

**步骤2** 创建虚拟环境。

执行以下命令创建虚拟环境（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate
```

**步骤3** 安装必要的库。在您的Linux服务器的终端中，运行以下命令安装agentarts-sdk、langgraph、langchain\_openai包。

```
安装agentarts-sdk包
pip install agentarts-sdk
安装langgraph包
pip install langgraph
安装langchain_openai包
pip install langchain-openai
```

**步骤4** 配置管理面环境变量。

```
=== 用于通过代码创建浏览器沙箱 ===
export HUAWEICLOUD_SDK_AK="您的华为云AK"
export HUAWEICLOUD_SDK_SK="您的华为云SK"
export AGENTARTS_CONTROL_ENDPOINT="https://agentarts.cn-southwest-2.myhuaweicloud.com"
```

**步骤5** 创建浏览器沙箱脚本。

新建一个Python文件，用于创建浏览器沙箱（例如browser\_sandbox.py）并填写以下代码。

导入浏览器工具及其他包：

```
import json
import os
from typing import Annotated, TypedDict
from langchain_core.messages import AIMessage, BaseMessage, HumanMessage, SystemMessage
from langchain_core.tools import tool
from langchain_openai import ChatOpenAI
from langgraph.graph import END, StateGraph
from langgraph.graph.message import add_messages
```

```
from langgraph.prebuilt import ToolNode
from agentarts.sdk import AgentArtsRuntimeApp
from agentarts.sdk.tools import browser_session
app = AgentArtsRuntimeApp()
```

继续填写如下代码，用于创建浏览器沙箱（创建的浏览器入站身份方式为API Key认证）：

```
from agentarts.sdk.tools import Browser
def create():
 print("开始创建浏览器沙箱，请稍等...")
 client = Browser(region="cn-southwest-2", auth_type="API_KEY")
 browser = client.create_browser(
 name="填写一个浏览器名称",
 auth_type="API_KEY",
 api_key_name="填写一个API Key的名称",
 description="浏览器沙箱工具描述",
 execution_agency_name="填写IAM委托名称",
 agent_gateway_id="填写Agent Gateway ID",
 network_config={"network_mode": "PUBLIC"},
)
 print("浏览器沙箱创建成功！快去控制台刷新看看吧。")
 # 打印出创建结果的详细信息
 print(browser)
if __name__ == '__main__':
 create()
```

代码中的参数说明：

- **浏览器名称**：以小写字母开头，小写字母或数字结尾，中间可包含数字、小写字母、中划线。字符数：2-40。
- **API Key名称**：只能包含英文字母、数字、下划线（\_）或中划线（-），长度1-64。
- **execution\_agency\_name**：IAM委托名称，请根据您的实际配置填写。
- **agent\_gateway\_id**：Agent Gateway的ID，可在AgentArts平台的网关管理页面获取。
- **network\_config**：网络模式，PUBLIC为公网访问，VPC为私网访问。

#### 步骤6 执行创建脚本。

执行python browser\_sandbox.py创建浏览器沙箱。

浏览器沙箱工具创建需要时间拉起，拉起时间大约为5s。创建完成后，可以在AgentArts平台看到创建好的浏览器沙箱工具。

```
(agentarts-sdk) D:\code\browser\agentarts-sdk\python-browser> cd /code/browser/agentarts-sdk/python-browser/ && python browser_sandbox.py
开始创建浏览器沙箱，请稍等...
2026-09-18 15:15:07.772 - agentarts.sdk.tools.browser.browser_client - INFO - Creating browser with name: test-1
浏览器沙箱创建成功！快去控制台刷新看看吧。
{'id': '41e1242e-1136-4659-a0d2-58f1b24a7b', 'name': 'test-1', 'description': '浏览器沙箱工具描述', 'auth_type': 'API_KEY', 'api_key_name': 'api-key-test', 'agent_gateway_id': '646a067-810b-4232-acc2-24b0b0e2b04', 'observability': {}, 'network_config': {'network_mode': 'PUBLIC'}, 'workload_identity': {'urn': 'agentidentity:cn-southwest:301:621868262600:iam::646a067:workloadidentity/default/browser-test-1'}, 'access_endpoint': 'https://api.browser.cn-southwest-301.amazonaws.com', 'created_at': '2026-09-18T02:15:07.564000000', 'updated_at': '2026-09-18T02:15:07.564000000', 'status': 'running'}
```

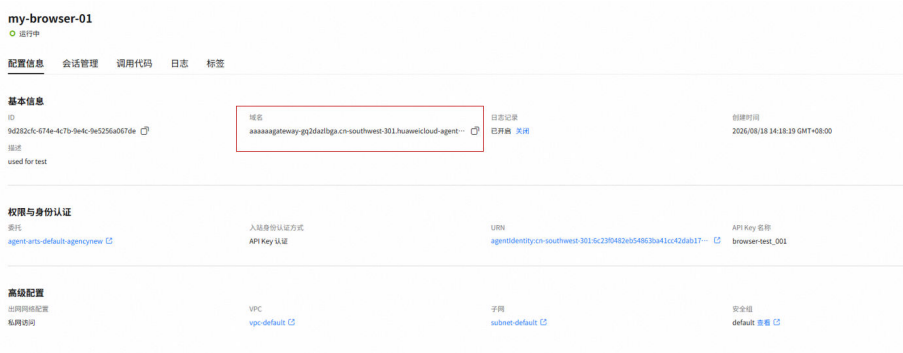
#### 步骤7 获取浏览器沙箱信息。

浏览器沙箱创建完成后，在列表可查看已创建的浏览器沙箱的名称。记录该名称，并获取该沙箱工具的域名、API Key。

沙箱工具的域名、API Key获取方法如下：

1. 单击沙箱工具名称。
2. 进入"配置信息"页面后，获取域名。

图 8-7 获取沙箱域名



3. 单击"URN"链接，在弹出的窗口中获取沙箱的API Key。

图 8-8 URN 链接

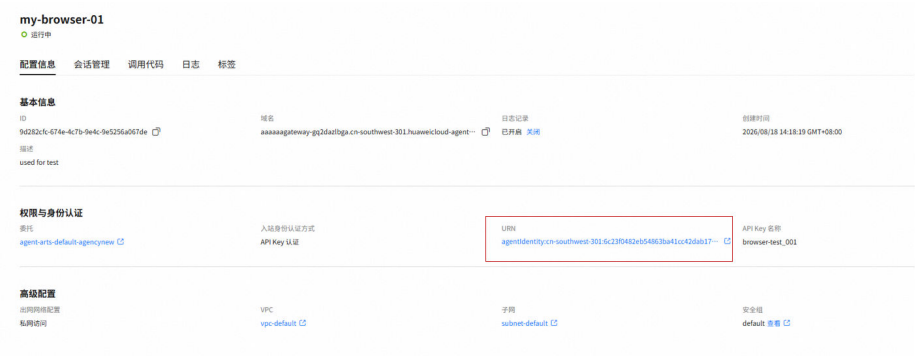
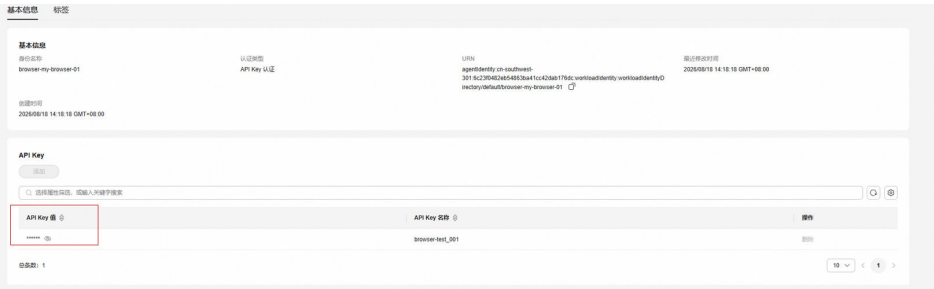


图 8-9 获取沙箱 API Key



步骤8 配置环境变量。

```
export AGENTARTS_BROWSER_DATA_ENDPOINT="http://您的浏览器沙箱域名"
export MODEL_API_KEY="您的大模型API密钥"
export BASE_URL="https://api.modelarts-maas.com/openai/v1"
```

其中，AGENTARTS\_BROWSER\_DATA\_ENDPOINT请配置为http://{浏览器沙箱域名}，获取方法请参考前面的步骤。

由于示例中使用华为云MaaS服务中的模型，因此需要[登录MaaS服务](#)创建API Key。

https://api.modelarts-maas.com/openai/v1表示使用OpenAI兼容接口。

步骤9 新建一个Python代码文件（例如browser\_app.py）将以下步骤的代码脚本按照顺序组合进去。

导入浏览器工具及其他包：

```
import json
import os
from typing import Annotated, TypedDict
from langchain_core.messages import AIMessage, BaseMessage, HumanMessage, SystemMessage
from langchain_core.tools import tool
from langchain_openai import ChatOpenAI
from langgraph.graph import END, StateGraph
from langgraph.graph.message import add_messages
from langgraph.prebuilt import ToolNode
from agentarts.sdk import AgentArtsRuntimeApp
from agentarts.sdk.tools import browser_session
app = AgentArtsRuntimeApp()
```

#### 步骤10 编写工具包装代码。

代码文件中填写如下脚本。用于将"连接浏览器沙箱并浏览网页"这一操作包装成一个标准的函数工具。当Agent中的大模型发现需要浏览网页时，会将目标网址传递给这个工具，工具会将网址发送到浏览器沙箱执行导航，再把页面信息拿回来给大模型。

```
@tool
def browse_webpage(url: str, description: str = "") -> str | None:
 """Browse a webpage and retrieve its page information in the browser sandbox"""
 api_key = "替换为获取的浏览器沙箱API Key" # your api key
 with browser_session(
 "cn-southwest-2",
 "替换为创建的浏览器沙箱名称",
 "agent-browser-session",
 api_key=api_key,
 verify_ssl=False,
) as browser_client:
 browser_client.navigate(url=url, api_key=api_key)
 page_info = browser_client.get_page_info(api_key=api_key)
 return json.dumps(page_info)
```

#### 步骤11 编写系统提示词。

代码文件中添加如下代码，用于编写系统提示词。

```
SYSTEM_PROMPT = """你是一个AI助手，可以使用浏览器工具来浏览网页并获取信息。
可用工具：
- browse_webpage(url: str, description: str): 在浏览器沙箱中打开网页并获取页面信息
使用原则：
1. 当用户需要获取网页内容、查看页面信息时使用此工具
2. 简单问题直接回答，无需工具验证
3. 工具调用最多1-2次，避免重复访问
4. 获得结果后立即返回答案
"""
```

#### 步骤12 定义Agent。

代码文件中添加如下代码，用于定义一个Agent并集成浏览器沙箱。

示例中使用deepseek-v4-flash模型，如果需要更换其他模型，请[登录MaaS服务](#)，获取模型接口的model参数值（注意代码使用了OpenAI格式的调用方法，需获取对应格式的model参数值）。

图 8-10 获取 MaaS 服务 model 参数值



```
创建Agent
llm = ChatOpenAI(
 model="deepseek-v4-flash",
 api_key=os.environ.get("MODEL_API_KEY", ""),
 base_url=os.environ.get("BASE_URL", ""),
 max_tokens=1000,
 temperature=0.7,
)

创建工具列表
tools = [browse_webpage]
工具绑定Agent
llm = llm.bind_tools(tools)
定义graph状态
class AgentState(TypedDict):
 messages: Annotated[list[BaseMessage], add_messages]
def call_model(state: AgentState):
 """调用模型并返回响应"""
 if not state["messages"] or all(
 not isinstance(msg, SystemMessage) for msg in state["messages"]
):
 messages = [SystemMessage(content=SYSTEM_PROMPT)] + state["messages"]
 else:
 messages = state["messages"]
 response = llm.invoke(messages)
 return {"messages": [response]}
def should_continue(state):
 """判断是否继续使用工具"""
 last_message = state["messages"][-1]
 if isinstance(last_message, AIMessage):
 has_tool_calls = bool(last_message.tool_calls)
 if has_tool_calls:
 return "tools"
 return END
创建LangGraph工作流
workflow = StateGraph(AgentState)
workflow.add_node("agent", call_model)
workflow.add_node("tools", ToolNode(tools))
设置入口
workflow.set_entry_point("agent")
添加边
workflow.add_conditional_edges("agent", should_continue, {"tools": "tools", "__end__": "__end__"})
workflow.add_edge("tools", "agent")
agent = workflow.compile()
```

### 步骤13 拼接服务器代码。

拼接以下代码，用于使用Agent启动一个本地服务器，并提供一个对外接收提问的接口。

```
@app.entrypoint
def agent_chat(payload: dict):
 query = "帮我访问 https://example.com 并告诉我页面标题"
```

```
运行Agent
result = agent.invoke({
 "messages": [HumanMessage(content=query)]
})
return result["messages"][-1].content
if __name__ == '__main__':
 app.run()
```

**步骤14** 运行并测试。

在服务器中执行python browser\_app.py命令运行示例代码（如果在创建Python代码文件时使用了其他名称，请按实际名称执行）。

**步骤15** 打开一个新的终端窗口（保持原窗口运行），使用curl命令进行测试。测试完成后，可以使用Ctrl + C停止运行的进程。

上述步骤中已经在代码里面内置了问题"帮我访问 <https://example.com> 并告诉我页面标题"，Agent会将该浏览网页的过程放到浏览器沙箱中运行，并返回页面信息结果。

```
curl --location http://localhost:8080/invocations --data-raw "{}"
```

```
(agentarts-sdk) D:\code\browser\agentarts-sdk-python-browser> curl --location http://127.0.0.1:8080/invocations --data-raw "{}"
"根据浏览结果，https://example.com 的页面标题是 **\`example.com\`**。\\n\\n这是一个典型的示例网站，页面标题与域名一致，显示为 \`example.com\`。"
```

----结束

## 8.4 创建浏览器

浏览器为智能体提供安全隔离的云端浏览器运行环境，用于智能体访问互联网网页，实现网页导航、表单填写、信息提取与复杂Web交互等网页交互操作。

创建浏览器有两种方式：

- [通过控制台创建浏览器](#)
- [通过SDK创建浏览器](#)

### 前提条件

- 使用沙箱工具前需要先[开通AgentArts服务](#)并进行服务授权，请在“授权管理”页面进行云服务开通及授权，详细信息请参考[授权管理](#)。
- （可选）管理员可在IAM控制台为创建的IAM子用户授予权限，授权后，子用户即可根据权限使用账号中的云服务资源，相关操作步骤如下。
  - a. [创建IAM用户](#)。
  - b. [给IAM用户授权](#)。
    - 为子用户授予完整的沙箱工具权限，需授予AgentArtsFullAccessPolicy及AgentIdentityFullAccessPolicy身份策略。
    - [创建自定义身份策略](#)并授予如下权限，授权项功能介绍请参考[表8-2](#)。

表 8-2 授权项

| 授权项               | 功能介绍          |
|-------------------|---------------|
| iam:agencies:pass | 授予向云服务传递委托的权限 |



| 授权项                                | 功能介绍             |
|------------------------------------|------------------|
| vpc:nativePorts:create             | 授予原生API创建端口权限    |
| vpc:routeTables:update             | 授予更新路由表权限        |
| vpc:nativePorts:delete             | 授予原生API删除端口权限    |
| eip:publicIps:associateInstance    | 授予将弹性公网IP绑定网卡的权限 |
| eip:publicIps:disassociateInstance | 授予将弹性公网IP解绑网卡的权限 |

图 8-11 授权策略



通过控制台创建浏览器

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3 单击右上角“创建浏览器”。
- 步骤4 在“创建浏览器”的弹框中，输入相关配置信息，参数说明请参考表8-3，配置完成后单击“立即创建”。



表 8-3 参数说明

| 参数      |    | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 基本信息    | 名称 | 浏览器名称，同一账号下沙箱工具名称不可重复，创建后不支持修改。<br>命名规则： <ul style="list-style-type: none"><li>以小写字母开头，小写字母或数字结尾。</li><li>名称中需包含数字、字母、中划线。</li><li>长度为2~40个字符。</li></ul> 系统会默认生成浏览器的名称，可根据需求自定义修改。                                                                                                                                                                                                                                                                    |
|         | 描述 | 用于对浏览器内容和用途的简要说明。<br>规则：长度不大于4096个字符。                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 权限与身份认证 | 委托 | <p>（可选）授予的代理权限或代理功能，允许代表智能体与外部系统进行通信和交互。</p> <p>可使用系统默认创建的AgentArtsBrowserAgency委托。</p> <p>如需手动创建，请单击“创建委托”并参考如下配置创建。</p> <ul style="list-style-type: none"><li>委托名称：自定义。</li><li>信任主体类型：选择“云服务”。</li><li>云服务：搜索service.WorkloadSandboxMetadata。</li><li>其余参数使用默认值。</li></ul> <p><b>图 8-12 创建委托</b></p>  <p>委托创建完成后可对委托进行授权，操作具体详情请参考<a href="#">创建委托并授权</a>。</p> |

| 参数    |        | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       | 入站身份认证 | <p>配置浏览器的身份认证方式。入站身份认证是指浏览器在接收和处理来自智能体或SDK的请求时，对请求进行身份验证，确保只有经过验证的请求才能访问沙箱工具中的资源，提高系统的安全性和可靠性。</p> <p>支持以下认证方式：</p> <ul style="list-style-type: none"><li>• IAM认证：将华为云IAM身份配置为入站身份认证，以验证调用者的身份。</li><li>• API Key认证：通过接口的访问密钥进行身份验证和授权。系统会自动生成API Key的名称，可根据需求自定义修改。身份认证凭证在凭据管理服务（DEW）托管，托管按需计费，费用说明请参考<a href="#">DEW 计费说明</a>。</li></ul>                                                                                                                                                             |
| 可观测配置 | 日志记录   | <ul style="list-style-type: none"><li>• 未开启，工具运行过程产生的日志无法上报至云日志服务。</li><li>• 开启后，工具运行过程产生的日志会上报云日志服务（LTS），日志管理费用按需收取，有关计费相关内容请参考<a href="#">LTS计费详情</a>。</li></ul>                                                                                                                                                                                                                                                                                                                                     |
| 高级配置  | 出网网络配置 | <ul style="list-style-type: none"><li>• 公网访问：能够连接到外部的互联网，访问外部的资源和服务。</li><li>• 私网访问：能够连接到内部的私有网络，访问内部的资源和服务。</li></ul> <p>当选择“私网访问”时，需配置如下参数：</p> <ul style="list-style-type: none"><li>- 选择已配置的VPC，如未配置，请单击“新建VPC”，具体请参考<a href="#">创建虚拟私有云和子网</a>。</li><li>- 选择已配置的子网，如未配置，请单击“新建子网”，具体请参考<a href="#">创建虚拟私有云和子网</a>。</li><li>- 选择已配置的安全组，如未配置，请单击“新建安全组”，具体请参考<a href="#">创建安全组</a>。</li></ul> <p>配置后单击  刷新。</p> |
|       | 标签（可选） | <p>由“标签键”和“标签值”组成，用于标识和分类云资源。</p> <p>您可以在TMS中创建预定义标签，用于使用同一标签标识多种云资源，创建后可在标签输入框下拉选择同一标签，具体操作请参考<a href="#">创建预定义标签</a>，创建后单击  刷新。</p> <p>单击“添加标签”，选择在TMS中创建的预定义标签，或自定义输入标签键和标签值。</p> <p>可添加一个或多个标签，最多添加20个标签。</p>                                                                                                                                                                                                 |

----结束

通过 SDK 创建浏览器

浏览器沙箱工具支持在隔离环境中进行网页浏览、页面截图、鼠标点击、键盘输入等操作。SDK提供create\_browser接口用于创建浏览器沙箱工具，通过SDK创建的浏览器沙箱将在浏览器沙箱列表中展示。

在Python文件中通过SDK创建浏览器沙箱，相关参数说明以及配置示例如下：

方法名：create\_browser

配置以下参数：

| 参数名                   | 类型   | 描述                                        |
|-----------------------|------|-------------------------------------------|
| name                  | str  | Required 浏览器沙箱名称。                         |
| auth_type             | str  | 认证类型，支持 "API_KEY" 或 "IAM"，默认 "API_KEY"。   |
| api_key_name          | str  | Required API Key名称。                       |
| description           | str  | 浏览器沙箱的描述。<br>Default: None                |
| execution_agency_name | str  | 为浏览器沙箱提供访问云服务的权限的IAM委托名。<br>Default: None |
| observability         | Dict | 可观测性配置（日志+指标）。<br>Default: None           |
| network_config        | Dict | 出站网络配置。<br>Default: None                  |
| agent_gateway_id      | str  | Agent Gateway ID。<br>Default: None        |
| tags                  | List | 标签列表。<br>Default: None                    |

返回值如下：

包含浏览器沙箱信息的字典，包括：

- id (str): 浏览器沙箱的ID。
- name (str): 浏览器沙箱的名称。该名称在您的账户中必须是唯一的。
- description (str): 浏览器沙箱的描述，用于LLM上下文分析。
- created\_at (str): 创建时间。
- updated\_at (str): 更新时间。
- execution\_agency\_name (str): IAM委托名。
- agent\_gateway\_id (str): Agent Gateway ID。
- workload\_identity (Dict): 认证信息。
- observability (Dict): 可观测性配置（日志+指标）。
- access\_endpoint (str): 访问域名。
- tags (List): 标签列表。

- network\_config (Dict): 出站网络配置。

API KEY代码示例

```
from agentarts.sdk.tools import Browser
demo_client = Browser(region="cn-southwest-2", auth_type="API_KEY", verify_ssl=False)
browser = demo_client.create_browser(
 name="your_browser_name",
 api_key_name="your_api_key_name",
 network_config={"network_mode": "PUBLIC"}
)
```

IAM代码示例

```
from agentarts.sdk.tools import Browser
demo_client = Browser(region="cn-southwest-2", auth_type="IAM", verify_ssl=False)
browser = demo_client.create_browser(
 name="your_browser_name",
 auth_type="IAM",
 network_config={"network_mode": "PUBLIC"}
)
```

更多操作

您还可以对已创建的浏览器执行如下操作。

表 8-4 更多操作

| 操作    | 说明                                                                                       |
|-------|------------------------------------------------------------------------------------------|
| 查看浏览器 | 1. 在浏览器列表中单击浏览器名称，查看浏览器详情。<br>2. 在“配置信息”页签，仅可开启或关闭“日志记录”，其余配置项不可修改。                      |
| 删除浏览器 | 浏览器删除后不可恢复，请谨慎操作。<br>1. 单击浏览器列表操作列的“删除”。<br>2. 在弹框中确认要删除后输入“DELETE”，然后单击“确定”。            |
| 复制浏览器 | 支持用户基于已配置的浏览器快速创建新的浏览器。<br>1. 单击浏览器列表操作列的“复制”。<br>2. 修改其基础信息，如名称、描述、参数配置等，配置完成后单击“立即创建”。 |

8.5 在智能体中集成浏览器

创建沙箱工具后，您需要将沙箱工具集成到智能体代码中，以便智能体可以调用隔离环境来完成代码运行等复杂任务。

前提条件

- 已创建沙箱工具，具体请参见[创建浏览器](#)。
- 已安装Python，且版本不低于3.10。

- 已安装agentarts-sdk包。
- 配置访问端点：通过AGENTARTS\_BROWSER\_DATA\_ENDPOINT设置环境变量，访问端点获取请参考[终端节点](#)，获取到终端节点后，在配置使用时，请配置为http://{域名}。

## 在智能体中集成浏览器

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。

**步骤3** 通过SDK调用浏览器工具。

在工具列表单击操作列的“调用代码”，通过SDK调用工具实现浏览器自动化的能力。SDK提供browser\_session函数，可快速启动浏览器的会话，并支持自动停止会话。

"调用代码"仅为代码示例，请根据实际情况替换代码示例中的"type"、"action"。

API\_KEY认证代码示例如下，获取API\_KEY请参考[如何获取浏览器的API\\_KEY](#)。

```
from agentarts.sdk.tools import browser_session
api_key = "your_actual_api_key" # your api key
with browser_session("cn-southwest-2", "your_browser_name", "your_session_name", api_key=api_key) as browser_client:
 response = browser_client.invoke(
 type="navigate",
 api_key=api_key,
 action={
 "url": "https://example.com"
 }
)
```

IAM认证代码示例如下，获取AK/SK请参考[如何获取华为云AK/SK](#)。

```
from agentarts.sdk.tools import browser_session
Please configure the HUAWEICLOUD_SDK_AK/HUAWEICLOUD_SDK_SK in your environment variables in advance.
with browser_session("cn-southwest-2", "your_browser_name", "your_session_name", auth_type="IAM") as browser_client:
 response = browser_client.invoke(
 type="navigate",
 action={
 "url": "https://example.com"
 }
)
```

了解沙箱工具支持的操作，请参考[浏览器支持的快捷操作](#)。

**步骤4** 将浏览器能力封装成工具。

将浏览器自动化能力封装成工具，此处使用langgraph的tool装饰器作为演示。

```
import json
from agentarts.sdk.tools import browser_session
from langchain_core.tools import tool
@tool
def navigate_tool(url: str) -> str | None:
 """Navigate to a URL in the browser"""
 api_key = "your_actual_api_key" # your api key
 with browser_session("your_region", "your_browser_name", "your_session_name", api_key=api_key) as browser_client:
 response = browser_client.invoke(
 type="navigate",
 api_key=api_key,
```

```
 action={
 "url": url
 }
)
 return json.dumps(response)
```

**步骤5** 将工具集成到智能体中。

将封装好的工具集成到智能体中，此处以langgraph框架开发智能体为例。

```
from langchain_openai import ChatOpenAI
import os
llm = ChatOpenAI(
 model="deepseek-v4-flash",
 api_key=os.environ.get("MODEL_API_KEY", ""),
 base_url=os.environ.get("BASE_URL", ""),
 max_tokens=1000,
 temperature=0.7,
)
创建工具列表
tools = [navigate_tool]
工具绑定Agent
llm = llm.bind_tools(tools)
```

----结束

## 8.6 浏览器会话管理

### 8.6.1 启动浏览器会话

浏览器会话 (Session) 是浏览器沙箱中执行自动化任务的基本单元，每个会话相互隔离。本章介绍如何通过SDK启动浏览器会话。

在启动一个新的浏览器会话时，您可以指定加载某个已存在的配置文件，使会话在创建后立即恢复至该配置文件保存的状态。加载的配置文件包括 Cookie、LocalStorage、SessionStorage、登录凭证等浏览器状态数据，使 Agent 无需每次会话都重新登录目标网站。

#### 约束与限制

- 会话默认空闲超时900秒，超过该时间无操作将自动终止。
- 会话最大持续时间为24小时。
- 单次invoke请求体上限为10 MB。
- 一个会话创建时只能加载一个配置文件ID；多会话可共用同一配置文件但相互隔离。
- 域名访问限制

创建会话时，您可以配置域名访问白名单和黑名单，限制浏览器会话可访问的网络范围，满足租户合规与防越权要求。

- 白名单 (allowed\_domains)：仅允许访问白名单中的域名，其余域名一律拦截。
- 黑名单 (blocked\_domains)：禁止访问黑名单中的域名，其余域名正常放行。

**示例：**租户希望Agent只能访问内部知识库 \*.tenant.com，同时禁止访问广告和社交平台，可将 \*.tenant.com 加入白名单，将广告和社交平台等加入黑名单。

白名单和黑名单可单独使用，也可组合使用。当同时配置时，请求需同时满足“在白名单中”且“不在黑名单中”才允许访问。

前提条件

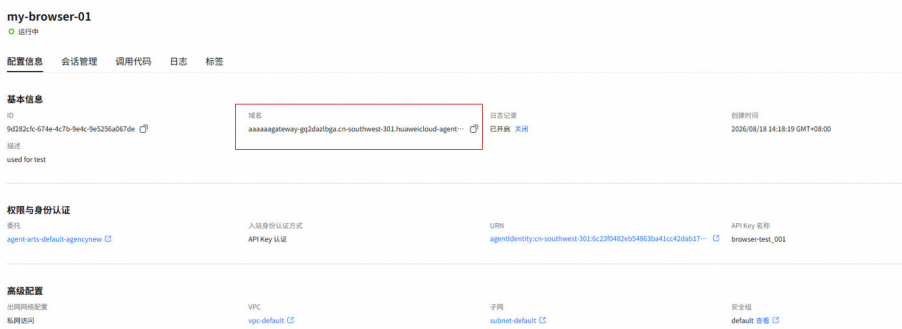
在会话中加载配置文件时，已创建至少一个浏览器配置文件（Profile），具体请参见[新建配置文件](#)。

启动会话

- 步骤1 确保已安装agentarts-sdk包，且Python版本不低于3.10。
- 步骤2 配置环境变量 AGENTARTS\_BROWSER\_DATA\_ENDPOINT，值为 http://{浏览器沙箱域名}。
- 浏览器域名获取如下：
1. 单击浏览器名称。

2. 进入“配置信息”页面后，获取域名。

图 8-13 获取沙箱域名



- 步骤3 使用 browser\_session 上下文管理器启动会话，会话在代码块执行完毕后自动终止。
- 获取API\_KEY请参考[如何获取浏览器的API\\_KEY](#)。
- 获取配置文件ID请参考[如何获取浏览器配置文件的ID](#)。

API\_KEY认证代码示例如下：

```
from agentarts.sdk.tools import Browser
创建浏览器客户端
client = Browser(region="cn-southwest-2", auth_type="API_KEY", verify_ssl=False)
启动会话并加载配置文件
client.start_session(
 browser_name="your_browser_name",
 session_name="your_session_name",
 api_key="your_api_key",
 profile_configuration={"profile_id": "your_profile_id"}
)
```

IAM认证代码示例如下：

```
from agentarts.sdk.tools import Browser
创建浏览器客户端（使用IAM认证）
client = Browser(region="cn-southwest-2", auth_type="IAM", verify_ssl=False)
启动会话并加载配置文件
client.start_session(
 browser_name="your_browser_name",
```

```
session_name="your_session_name",
profile_configuration={"profile_id": "your_profile_id"}
)
```

----结束

会话状态说明

| 状态  | 说明                       |
|-----|--------------------------|
| 运行中 | 会话正在执行自动化任务，可查看实时画面、可终止。 |
| 已终止 | 会话已结束，实时画面与终止操作均不可用。     |

8.6.2 启动会话时配置浏览器代理

浏览器代理用于在浏览器会话中配置代理服务器，使浏览器通过指定的代理服务器访问外部网络。该功能适用于以下场景：

- 企业内网访问：Agent需要通过企业代理服务器访问内部知识库或业务系统。
- 安全合规要求：租户要求所有浏览器流量必须经过指定的安全代理进行审计和过滤。
- 网络隔离环境：浏览器沙箱处于隔离网络中，需通过代理访问互联网。

约束与限制

- 每个会话最多可配置5个代理。
- 每个代理最多支持100个域名模式。
- Bypass模式使用操作系统原生代理绕过列表，最多支持配置100条域名、IP地址规则。
- 仅支持HTTP/HTTPS/SOCKS5三种代理协议。
- 代理配置在会话创建后不可变，如需修改代理配置，需重新创建会话。

配置方式

浏览器代理在创建会话时配置，创建后不可修改。如需修改代理配置，需终止当前会话并重新创建。

在 browser\_session 中通过 proxies 和 bypass\_patterns 参数传入代理配置。

```
from agentarts.sdk.tools import Browser
创建浏览器客户端
client = Browser(region="cn-southwest-2", auth_type="API_KEY", verify_ssl=False)
配置单个默认代理，并设置 bypass 直连域名
proxy_configuration = {
 "proxies": [
 {
 "server": "proxy.example.com", # 代理服务器地址
 "port": 8080, # 代理服务器端口
 "type": "http" # 代理类型：http / https / socks5
 }
],
 "bypass": {
 "domain_patterns": ["localhost", "127.0.0.1"] # 这些域名直连，不走代理
 }
}
```



```
}
启动会话并加载代理配置
client.start_session(
 browser_name="your_browser_name",
 session_name="your_session_name",
 api_key="your_api_key",
 proxy_configuration=proxy_configuration
)
```

## 配置示例

### 场景一：单代理配置

企业内网有一个代理服务器，Agent 需要通过该代理访问所有外部网站，但本地地址不走代理。

```
proxy_configuration = {
 "proxies": [
 {
 "server": "proxy.example.com", # 代理服务器地址
 "port": 8080, # 代理服务器端口
 "type": "http", # 代理类型: http / https / socks5
 }
],
 "bypass": {
 "domain_patterns": ["localhost", "127.0.0.1"] # 这些域名直连，不走代理
 }
}
```

### 场景二：多代理配置

为不同域名指定不同代理，未命中域名模式的请求走默认代理，同时配置 bypass 直连域名。

```
配置多代理域名路由 + bypass 直连
proxy_configuration = {
 "proxies": [
 {
 "server": "proxy-a.example.com", # 代理服务器地址
 "port": 8080, # 代理服务器端口
 "type": "http", # 代理类型
 "domain_patterns": ["example.com", ".example.com"] # 匹配 example.com 及其子域名
 },
 {
 "server": "proxy-b.example.com", # 代理服务器地址
 "port": 1080, # 代理服务器端口
 "type": "socks5", # 代理类型
 "domain_patterns": ["internal.corp", ".internal.corp"] # 匹配 internal.corp 及其子域名
 },
 {
 "server": "proxy-default.example.com", # 默认代理（无 domain_patterns）
 "port": 8080, # 代理服务器端口
 "type": "http", # 代理类型
 }
],
 "bypass": {
 "domain_patterns": ["localhost", "127.0.0.1", ".local"] # 这些域名直连，不走任何代理
 }
}
```

### 场景三：需要认证的代理

代理服务器需要用户名密码认证时，通过 credential\_id 引用平台密钥管理服务中预存的凭据。

```
proxy_configuration = {
 "proxies": [

```

```
{
 "server": "auth-proxy.example.com", # 代理服务器地址
 "port": 8443, # 代理服务器端口
 "type": "https", # 代理类型
 "credential_id": "your_credential_id" # 凭据 ID，引用平台密钥管理服务中的凭据，明文传递用户名
 密码
}
],
"bypass": {
 "domain_patterns": ["localhost", "127.0.0.1"] # 这些域名直连，不走代理
}
}
```

### 8.6.3 查看浏览器会话

本章介绍如何查看运行中会话的实时执行画面，包含进入实时会话、查看浏览器会话详情等。

#### 查看浏览器会话详情

会话详情信息包括会话的基本信息（会话ID、状态、时间、配置文件等）和代理信息（代理域、旁域匹配规则等）。无论会话处于“运行中”还是“已终止”状态，均可查看会话详情信息。

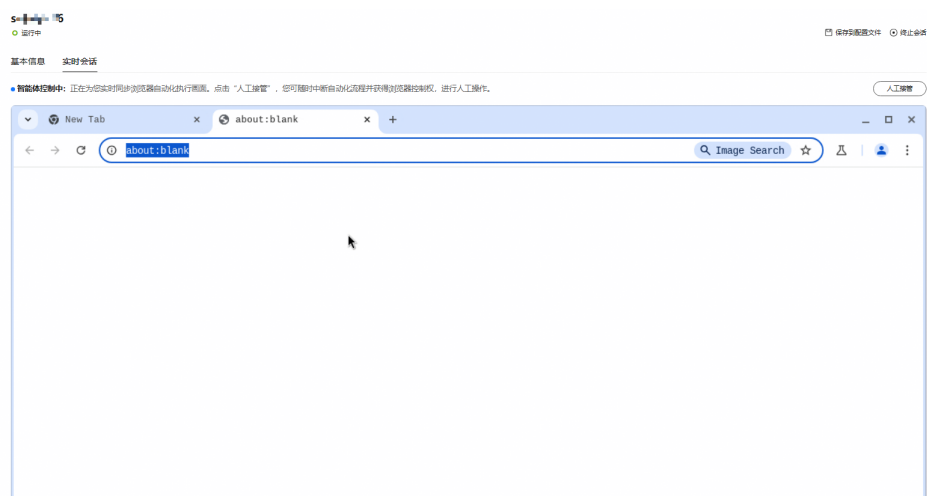
- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3** 单击已创建的浏览器名称。
- 步骤4** 在“会话管理”页签，单击会话列表中的会话名称，进入会话详情页。
- 步骤5** 在“基本信息”页签，会话详情页可查看以下信息：
- **基本信息**：展示会话名称、ID、会话状态、会话开始时间、会话上次更新时间、配置文件等。
  - **代理信息**：展示代理与旁域匹配规则，以及代理域表格（包含服务器、端口、代理协议、凭证ID、域名匹配规则）。代理信息详细说明请参考[启动会话时配置浏览器代理](#)。
- 结束

#### 查看实时会话

仅支持查看状态为“运行中”的会话。

- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3** 单击已创建的浏览器名称。
- 步骤4** 在“会话管理”页签，单击“运行中”的会话列表中操作列的“查看实时会话”。
- 步骤5** 默认展示“实时会话”标签页，实时同步浏览器自动化执行画面。
- **智能体控制模式**：  
当会话由智能体自动执行时，展示智能体控制中：正在为您实时同步浏览器自动化执行画面。单击“人工接管”，您可随时中断自动化流程并获得浏览器控制权，进行人工操作。

图 8-14 智能体控制模式



- 人工接管模式：  
单击“人工接管”后，自动化流程暂停，此时您可以在浏览器中进行页面导航、编辑URL等操作。详细操作请参考[人工接管浏览器会话](#)。

----结束

8.6.4 人工接管浏览器会话

人工接管是指在智能体自动化执行过程中，人工用户临时获取浏览器的控制权，对当前页面进行手动干预操作。该功能适用于以下场景：

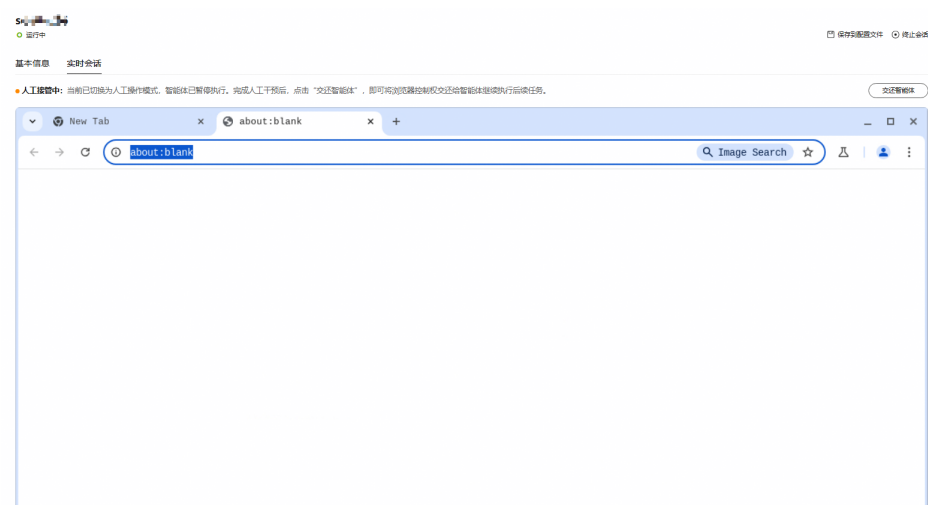
- **辅助调试：**自动化流程执行到某一步骤时，用户需要手动验证页面状态或调整浏览器操作。
- **处理验证码：**目标网站弹出人机验证（如验证码、滑块），需要人工完成验证后继续自动化流程。
- **手动导航：**用户需要手动跳转到特定页面或输入搜索内容，再交还智能体继续执行。
- **异常干预：**自动化流程出现异常或卡顿，用户手动修正后恢复执行。

进入人工接管模式后，页面顶部显示提示信息：“人工接管中：当前已切换为人工操作模式，智能体已暂停执行。完成人工干预后，单击“交还智能体”，即可将浏览器控制权交还给智能体继续执行后续任务。”

人工接管浏览器会话

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3 单击已创建的浏览器名称。
- 步骤4 在“会话管理”页签，单击“运行中”的会话列表中操作列的“查看实时会话”。
- 步骤5 在“智能体控制中”提示栏右侧，单击“人工接管”。
- 步骤6 自动化流程立即暂停，页面切换为人工接管模式，显示“人工接管中”提示信息。
- 步骤7 进入人工接管模式后，您可以像使用普通浏览器一样操作沙箱中的浏览器，比如页面导航、编辑URL、搜索等操作。

图 8-15 人工接管浏览器会话



----结束

交还智能体

完成人工干预后，可将浏览器控制权交还给智能体，恢复自动化执行。

- 步骤1 在“人工接管中”提示栏右侧，单击“交还智能体”。
- 步骤2 浏览器控制权交还给智能体，自动化流程从当前页面状态继续执行。

交还智能体后，智能体将基于人工操作后的当前页面状态继续执行后续任务。如果人工操作改变了页面状态（如导航到新页面、提交了表单），请确保当前页面状态符合后续自动化流程的预期。

----结束

8.6.5 终止浏览器会话

本章介绍如何终止正在运行的浏览器会话，提供从会话详情页、会话列表以及SDK等方式终止浏览器会话。

通过控制台在会话列表终止

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3 单击已创建的浏览器名称。
- 步骤4 在“会话管理”页签，单击“运行中”的会话列表中操作列的“终止会话”。
- 步骤5 会话立即终止，状态变更为“已终止”。

----结束

通过控制台在会话详情页终止

- 步骤1 在“会话管理”页签，单击“运行中”的会话列表中操作列的“查看实时会话”。

- 步骤2 进入会话详情页面。
- 步骤3 单击“终止会话”按钮。
- 步骤4 会话立即终止，状态变更为“已终止”。
- 结束

通过 SDK 终止浏览器会话

浏览器支持通过SDK终止浏览器会话。SDK提供stop\_session接口用于终止当前浏览器会话，释放浏览器资源。

方式一：直接调用stop\_session方法

在Python文件中通过SDK直接调用stop\_session方法终止浏览器会话。

- 方法名： stop\_session
- 参数说明：

| 参数名     | 类型  | 描述                                                                                             |
|---------|-----|------------------------------------------------------------------------------------------------|
| api_key | str | API Key认证密钥。当使用API_KEY认证方式时，如果未提供此参数，SDK会尝试从环境变量HUAWEICLOUD_SDK_BROWSER_API_KEY获取。Default：None |

- 返回值：  
bool: 成功终止会话或无活跃会话时返回True。
- 行为说明：
  - 如果当前没有活跃会话（ session\_id为空 ），方法会立即返回True（ no-op操作 ）。
  - 成功终止会话后，SDK会清空会话相关状态（ session\_id、browser\_name、automation\_endpoint、live\_view\_endpoint ）。
  - 支持两种认证方式： API\_KEY和IAM（ AK/SK ）

API KEY代码示例

```
from agentarts.sdk.tools import Browser
创建浏览器客户端
client = Browser(region="cn-southwest-2", auth_type="API_KEY", verify_ssl=False)
启动会话
client.start_session(
 browser_name="your_browser_name",
 session_name="your_session_name",
 api_key="your_api_key"
)
执行浏览器操作...
终止会话
success = client.stop_session(api_key="your_api_key")
print(f"会话终止结果: {success}")
```

IAM代码示例

```
from agentarts.sdk.tools import Browser
创建浏览器客户端（使用IAM认证）
```

```
client = Browser(region="cn-southwest-2", auth_type="IAM", verify_ssl=False)
启动会话
client.start_session(
 browser_name="your_browser_name",
 session_name="your_session_name"
)
执行浏览器操作...
终止会话（IAM认证无需传递api_key）
success = client.stop_session()
print(f"会话终止结果: {success}")
```

方式二：使用上下文管理器自动终止

推荐使用browser\_session上下文管理器，它会在退出时自动终止会话，无需手动调用stop\_session。

• browser\_session 参数说明：

| 参数名                   | 类型        | 描述                                                         |
|-----------------------|-----------|------------------------------------------------------------|
| region                | str       | Required 区域名称，如 "cn-southwest-2"。                          |
| browser_name          | str       | Required 浏览器沙箱名称。                                          |
| session_name          | str       | Required 客户端指定的会话名称，只能包含字母、数字、下划线和中划线，长度1-128。             |
| session_id            | str       | 客户端指定的会话ID，不传则由服务端生成。Default: None                         |
| viewport              | Dict      | 视口配置，如 {"width": 1920, "height": 1080}。Default: None       |
| profile_configuration | Dict      | 配置档案配置，格式为 {"profile_id": "your_profile_id"}。Default: None |
| allowed_domains       | List[str] | 允许访问的域名列表。与 blocked_domains 互斥。Default: None               |
| blocked_domains       | List[str] | 禁止访问的域名列表。与 allowed_domains 互斥。Default: None               |
| proxy_configuration   | Dict      | 代理配置。Default: None                                         |
| session_timeout       | int       | 会话超时时间（秒），默认900秒（15分钟）。Default: 900                        |

| 参数名        | 类型       | 描述                                                                           |
|------------|----------|------------------------------------------------------------------------------|
| auth_type  | str      | 认证类型，支持 "API_KEY" 或 "IAM"。<br>Default: "API_KEY"                             |
| api_key    | str      | API_KEY模式下的认证密钥。也可通过环境变量 HUAWEICLOUD_SDK_BROWSER_API_KEY设置。<br>Default: None |
| verify_ssl | bool/str | SSL证书验证。True验证、False跳过、或字符串指向CA包路径。<br>Default: True                         |

- 支持两种认证方式：API\_KEY和IAM（AK/SK）

API KEY代码示例

```
from agentarts.sdk.tools import browser_session
api_key = "your_actual_api_key" # your api key
with browser_session(
 "cn-southwest-2",
 "your_browser_name",
 "your_session_name",
 api_key=api_key,
 profile_configuration={"profile_id": "your_profile_id"} # 加载配置文件
) as browser_client:
 response = browser_client.navigate(
 url="https://example.com",
 api_key=api_key
)
退出with块时，会话自动终止
```

IAM代码示例

```
from agentarts.sdk.tools import browser_session
Please configure the HUAWEICLOUD_SDK_AK/HUAWEICLOUD_SDK_SK in your environment
variables in advance.
with browser_session(
 "cn-southwest-2",
 "your_browser_name",
 "your_session_name",
 auth_type="IAM",
 profile_configuration={"profile_id": "your_profile_id"} # 加载配置文件
) as browser_client:
 response = browser_client.navigate(url="https://example.com")
退出with块时，会话自动终止
```

注意事项

- 资源释放：终止会话会释放浏览器沙箱资源，建议在完成所有浏览器操作后及时终止会话
- 幂等性：多次调用stop\_session是安全的，如果没有活跃会话，方法会返回True
- 状态清空：终止会话后，Browser实例的会话相关状态会被清空，如需再次使用需重新启动会话
- 认证方式：
  - API\_KEY认证：需要传递api\_key参数或设置环境变量 HUAWEICLOUD\_SDK\_BROWSER\_API\_KEY

- IAM认证：需要配置环境变量HUAWEICLOUD\_SDK\_AK和HUAWEICLOUD\_SDK\_SK
- 5. 超时机制：即使未主动终止会话，会话也会在超时后自动终止（默认超时时间为900秒/15分钟）

## 8.6.6 浏览器支持的快捷操作

浏览器沙箱会话支持以下自动化操作，可通过SDK的invoke方法调用，也可在实时会话页面通过人工接管模式手动执行。

### 导航类操作

**navigate**：导航到指定URL。

action参数：

```
{ "url": "https://example.com"}
```

**go\_back**：返回上一页。

action参数：

```
{}
```

**go\_forward**：前进到下一页。

action参数：

```
{}
```

**refresh**：刷新当前页面。

action参数：

```
{}
```

### 鼠标类操作

**mouse\_click**：鼠标点击。

action参数：

```
{ "x": 312, "y": 482, "button": "left", "click_count": 1}
```

参数说明：x、y为坐标；button可选值为"left"、"right"、"middle"，默认"left"；click\_count为点击次数，默认1。

**mouse\_move**：鼠标移动。

action参数：

```
{ "x": 312, "y": 482}
```

**mouse\_drag**：鼠标拖拽。

action参数：

```
{ "start_x": 100, "start_y": 100, "end_x": 200, "end_y": 200, "button": "left"}
```

参数说明：button可选值为"left"、"right"、"middle"，默认"left"。

**mouse\_scroll**：鼠标滚动。



action参数：

```
{ "x": 500, "y": 300, "delta_x": 0, "delta_y": -100}
```

参数说明：delta\_x为水平滚动量，delta\_y为垂直滚动量，默认均为0。

## 键盘类操作

**key\_press**：按键。

action参数：

```
{ "key": "Enter", "presses": 1}
```

参数说明：presses为按键次数，范围1-100，默认1。

**key\_type**：输入文本。

action参数：

```
{ "text": "Hello, World!"}
```

**key\_shortcut**：按键组合。

action参数：

```
{ "keys": ["Control", "c"]}
```

参数说明：keys为按键列表，最多5个按键。

## 页面信息操作

**get\_page\_info**：获取当前页面信息。

action参数：

```
{}
```

**screenshot**：截图。

action参数：

```
{ "format": "jpeg", "quality": 80, "full_page": false}
```

参数说明：format可选"png"或"jpeg"，默认"jpeg"；quality为JPEG压缩质量，范围1-100，默认80；full\_page为是否全页截图，默认false。

## 标签页管理类操作

**list\_tabs**：列出所有标签页。

action参数：

```
{}
```

**switch\_tab**：切换标签页。

action参数：

```
{ "tab_id": "tab-123"}
```

**close\_tab**：关闭标签页。

action参数：

```
{ "tab_id": "tab-123"}
```

参数说明：tab\_id为可选，不提供则关闭当前标签页。

**new\_tab**：打开新标签页。

action参数：

```
{ "url": "https://example.com"}
```

参数说明：url为可选，不提供则打开空白页。

其他操作

**wait**：等待。

action参数：

```
{ "duration": 2.5}
```

参数说明：duration为等待时长（秒），范围0.1-30。

8.7 浏览器配置文件管理

8.7.1 新建配置文件

配置文件（Profile）用于保存浏览器会话的运行状态，包括但不限于 Cookie、本地存储（LocalStorage / sessionStorage）、登录凭证等。通过配置文件，您可以：

- 将当前会话状态持久化保存，避免重复登录或重复初始化。
- 在新的会话中加载已有配置文件，快速恢复到上次保存的状态。
- 对同一业务场景创建多个配置文件，按需切换使用。

会话状态不会自动保存，如需持久化当前浏览器的状态（Cookie、登录凭证等），请手动在实时会话页面单击“[保存到配置文件](#)”，或[通过SDK将会话保存至配置文件](#)。

通过控制台新建配置文件

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3 单击右上角“配置文件”，然后单击“新建配置文件”。
- 步骤4 输入相关配置信息，参数说明请参考[表1 参数说明](#)，配置完成后单击“确定”。

表 8-5 参数说明

| 参数 | 说明                                                                                                 |
|----|----------------------------------------------------------------------------------------------------|
| 名称 | 配置文件名称。<br>系统默认生成格式为profile- <a href="#">&lt;随机字符串&gt;</a> （例如profile-6amd1kwy2），您可自行修改为具有业务含义的名称。 |

| 参数      | 说明                                                                                                                                                                                                                                                                                                 |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 描述      | 对配置文件用途的简要说明，最多支持 4,096 个字符。                                                                                                                                                                                                                                                                       |
| 标签 (可选) | <p>由“标签键”和“标签值”组成，用于标识和分类云资源。</p> <p>您可以在TMS中创建预定义标签，用于使用同一标签标识多种云资源，创建后可在标签输入框下拉选择同一标签，具体操作请参考<a href="#">创建预定义标签</a>，创建后单击刷新。</p> <p>单击“添加标签”，选择在TMS中创建的预定义标签，或自定义输入标签键和标签值。</p> <p>可添加一个或多个标签，最多添加20个标签。</p> |

----结束

通过 SDK 新建配置文件

浏览器配置文件用于保存浏览器会话状态（cookie、localStorage、sessionStorage），可在创建会话时加载，实现状态持久化。SDK提供 create\_browser\_profile接口用于创建浏览器配置文件。

创建浏览器配置文件

在Python文件中通过SDK创建浏览器配置文件，相关参数说明以及配置示例如下：

- 方法名：create\_browser\_profile
- 参数说明：

| 参数名         | 类型         | 描述                                                                   |
|-------------|------------|----------------------------------------------------------------------|
| name        | str        | Required 配置文件名称。必须以小写字母开头，小写字母或数字结尾，中间可包含数字、小写字母、中划线。字符数：2-40。       |
| description | str        | 配置文件描述，最大4096个字符。<br>Default: None                                   |
| tags        | List[Dict] | 标签列表，最多20个，key必须唯一。格式：[{"key": "标签键", "value": "标签值"}]。Default: None |

API KEY代码示例

```
from agentarts.sdk.tools import Browser
创建浏览器客户端
client = Browser(region="cn-southwest-2", auth_type="API_KEY", verify_ssl=False)
创建浏览器配置文件
profile = client.create_browser_profile(
 name="your_profile_name",
 description="浏览器配置文件描述",
 tags=[{"key": "env", "value": "dev"}]
)
```

```
print(f"配置文件ID: {profile['id']}")
print(f"配置文件名称: {profile['name']}")
```

### IAM代码示例

```
from agentarts.sdk.tools import Browser
创建浏览器客户端（使用IAM认证）
client = Browser(region="cn-southwest-2", auth_type="IAM", verify_ssl=False)
创建浏览器配置文件
profile = client.create_browser_profile(
 name="your_profile_name",
 description="浏览器配置文件描述",
 tags=[{"key": "env", "value": "dev"}]
)
print(f"配置文件ID: {profile['id']}")
print(f"配置文件名称: {profile['name']}")
```

## 8.7.2 将会话保存至配置文件

在会话执行过程中或执行完成后，您可以将当前浏览器的状态保存到配置文件，以便后续会话加载复用。

保存操作将覆盖目标配置文件的现有状态，原有数据将被替换，请谨慎操作。

数据大小限制：单个配置文件（Profile）的数据包上限为50MB，超出将无法保存。

### 通过控制台将会话保存至配置文件

- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3** 单击已创建的浏览器名称。
- 步骤4** 在“会话管理”页签，单击会话列表中操作列的“查看实时会话”。
- 步骤5** 在“实时会话”页面，查看浏览器页面，右上角单击“保存到配置文件”。
- 步骤6** 在列表中选中目标配置文件，单击“确定”，当前会话状态将覆盖写入该配置文件。

----结束

### 通过 SDK 将会话保存至配置文件

浏览器沙箱工具支持通过SDK将当前会话状态（cookie、localStorage、sessionStorage）保存到指定的配置文件中，以便下次创建会话时恢复状态。SDK提供save\_profile接口用于保存会话状态。

- **前提条件**

在调用save\_profile方法前，必须已经启动浏览器会话。

- **保存会话状态**

在Python文件中通过SDK保存浏览器会话状态，相关参数说明以及配置示例如下：

- **方法名：**save\_profile
- **配置以下参数：**

| 参数名        | 类型  | 描述                                                                                              |
|------------|-----|-------------------------------------------------------------------------------------------------|
| profile_id | str | Required 要保存到的配置文件ID。由控制面创建并返回。                                                                 |
| api_key    | str | API Key认证密钥。当使用API_KEY认证方式时，如果未提供此参数，SDK会尝试从环境变量HUAWEICLOUD_SDK_BROWSER_API_KEY获取。Default: None |

- 返回值如下：  
包含保存结果的字典，包括：
  - profile\_id (str): 已保存的配置文件ID。
  - saved\_at (str): 保存时间。
  - size\_bytes (int): 保存数据大小（字节）。
  - cookie\_count (int): 保存的cookie数量。
  - origin\_count (int): 保存的源数量。
- 行为说明：
  - 必须在活跃会话中调用（会话终止后无法保存）
  - 保存的是当前会话的浏览器状态（cookie、localStorage、sessionStorage）
  - 保存后，下次创建会话时通过profile\_configuration={"profile\_id": "..."}加载该配置，即可恢复状态

API KEY代码示例

```
from agentarts.sdk.tools import Browser
创建浏览器客户端
client = Browser(region="cn-southwest-2", auth_type="API_KEY", verify_ssl=False)
启动会话（前提条件）
client.start_session(
 browser_name="your_browser_name",
 session_name="your_session_name",
 api_key="your_api_key"
)
执行浏览器操作
client.navigate(url="https://example.com", api_key="your_api_key")
保存当前会话状态到配置文件
result = client.save_profile(
 profile_id="your_profile_id",
 api_key="your_api_key"
)
print(f"保存结果: {result}")
终止会话
client.stop_session(api_key="your_api_key")
```

### IAM代码示例

```
from agentarts.sdk.tools import Browser
创建浏览器客户端（使用IAM认证）
client = Browser(region="cn-southwest-2", auth_type="IAM", verify_ssl=False)
启动会话（前提条件）
client.start_session(
 browser_name="your_browser_name",
 session_name="your_session_name"
)
执行浏览器操作
client.navigate(url="https://example.com")
保存当前会话状态到配置文件
result = client.save_profile(profile_id="your_profile_id")
print(f"保存结果: {result}")
终止会话（IAM认证无需传递api_key）
client.stop_session()
```

### 注意事项

1. **前提条件**：调用save\_profile前必须已经启动会话，否则会抛出异常
2. **调用时机**：必须在活跃会话中调用save\_profile，会话终止后无法保存
3. **profile\_id**：必须是通过控制面已创建的配置文件ID
4. **保存内容**：当前会话的浏览器状态，包括cookie、localStorage、sessionStorage
5. **状态恢复**：下次创建会话时通过profile\_configuration={"profile\_id": "..."}加载配置，即可恢复之前保存的状态
6. **认证方式**:
  - API\_KEY认证：需要传递api\_key参数或设置环境变量HUAWEICLOUD\_SDK\_BROWSER\_API\_KEY
  - IAM认证：需要配置环境变量HUAWEICLOUD\_SDK\_AK和HUAWEICLOUD\_SDK\_SK

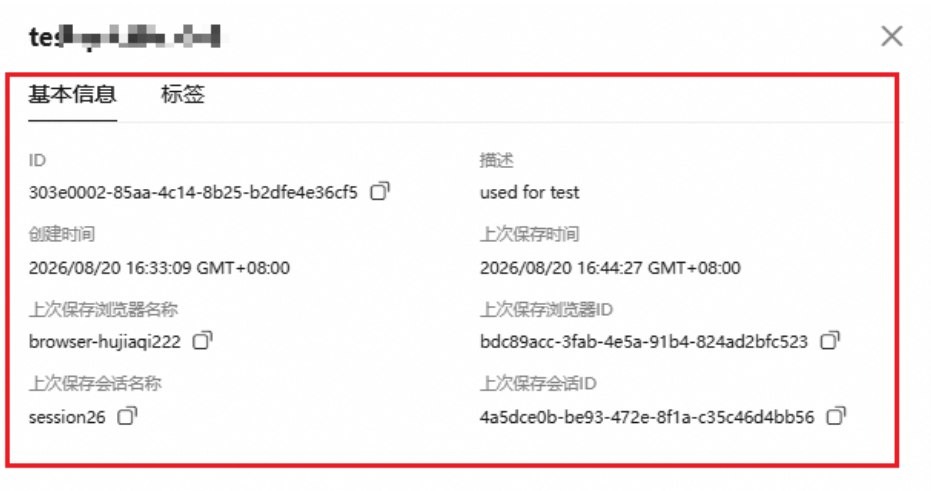
## 8.7.3 管理配置文件

配置文件创建完成后，您可以查看配置文件、删除配置文件，帮助用户高效管理浏览器会话状态的持久化配置。

### 查看配置文件详情

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3 单击右上角“配置文件”。
- 步骤4 展示当前用户下的所有配置文件列表。
- 步骤5 单击配置文件名称，在“基本信息”页签查看配置文件详情。

图 8-16 查看配置文件详情



- 步骤6** 在“标签”页签，查看已添加的标签，单击“编辑标签”，可对标签进行管理（最多支持添加20个标签）。
- 添加标签：单击“添加标签”，可选择TMS中的预定义标签，或自定义输入标签键与值。
  - 修改标签：直接修改已有标签内容即可。
  - 删除标签：单击目标标签后的“删除”即可。
- 结束

删除配置文件

- 步骤1** 在“浏览器”页面，单击右上角“配置文件”。
- 步骤2** 展示当前用户下的所有配置文件列表。
- 步骤3** 单击操作列的“删除”。
- 步骤4** 在弹框中确认要删除后输入“DELETE”，然后单击“确定”。
- 结束

8.8 管理浏览器

浏览器创建完成后，您可以在浏览器列表按名称或标签筛选，单击名称进入浏览器详情页，可查看运行日志及管理标签。

查看浏览器日志

已开启日志记录，如未启用日志功能，可在工具详情页“日志”页签单击“去启用”开启日志记录。

- 步骤1** 登录AgentArts智能体平台。
- 步骤2** 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3** 单击浏览器名称，查看浏览器详情。

**步骤4** 在“日志”页签，查看浏览器使用过程中产生的日志。

----结束

## 管理标签

**步骤1** 进入浏览器界面，在浏览器列表中可通过标签过滤查询浏览器。

图 8-17 通过标签过滤查询浏览器



**步骤2** 单击浏览器名称，查看浏览器详情。

**步骤3** 在“标签”页签，查看已添加的标签，单击“编辑标签”，可对标签进行管理（最多支持添加20个标签）。

- 添加标签：单击“添加标签”，可选择TMS中的预定义标签，或自定义输入标签键与值。
- 修改标签：直接修改已有标签内容即可。
- 删除标签：单击目标标签后的“删除”即可。

----结束



# 9 知识库

## 9.1 知识库介绍

### 功能概述

知识库是组织、存储及管理知识的系统，涵盖文档、图片等信息的分类整理，可以帮助用户高效管理大量的信息。在智能体中添加知识库，使其能够利用知识库中的信息与用户进行交互，可以显著提升智能体的准确度和专业度。

AgentArts支持管理平台知识库和第三方知识库。

平台知识库功能对文本文档、FAQ（Frequently Asked Questions，常见问题解答）等数据进行向量化存储、知识检索，支持为应用、工作流提供检索增强能力。无论是文本文档、演示文稿，还是电子表格文件，用户都可以轻松地将数据导入知识库，无需额外的转换或格式处理。

图 9-1 文档处理流程



知识库支持导入如表9-1所示的本地文档：

表 9-1 支持的文档格式

| 文档类型 | 文档格式及限制                                                                                                                                                                                                                                                                                                                                                                               |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 知识文档 | <ul style="list-style-type: none"><li>支持上传常见的文档格式：<ul style="list-style-type: none"><li>文档/文本类：doc、docx、pdf、ofd、txt、md、wps。</li><li>表格/数据类：xls、xlsx、csv、et。</li><li>演示/幻灯片类：ppt、pptx、dps。</li><li>图片/设计类：jpg、jpeg、png、gif、bmp、webp、tiff、tif、psd、ico、pcx。<br/>AgentArts支持用户根据需求，将图片类的文件直接存储，或是提取图片中的文字信息进行存储。</li></ul></li><li>不同套餐包对文件大小限制不同，具体请参考<a href="#">知识库使用限制</a>。</li></ul> |
| FAQ  | <ul style="list-style-type: none"><li>支持按照模板上传文件，文件格式仅支持xlsx、xls、docx、doc。</li><li>不同套餐包对文件大小限制不同，具体请参考<a href="#">知识库使用限制</a>。</li><li>Excel单个文件最大支持100,000条数据。</li><li>FAQ文件中不允许空行，空行后的数据将被忽略。</li></ul>                                                                                                                                                                            |

费用说明

- 平台知识库：不同套餐包的[知识库存储空间容量](#)不同，具体可参考[计费说明](#)。
- 第三方知识库：AgentArts侧不计费，但第三方系统的部署与运维费用需用户自行承担，具体计费以第三方为准。

应用场景

- 文旅服务场景：**旅游平台可以在知识库中上传景区介绍、游玩路线、周边餐饮和住宿信息、交通时刻表、门票价格及优惠政策，以及当地特色体验项目。当游客查询“周末亲子游景区推荐”时，智能体会结合适合的年龄段和游玩时长等信息生成定制化方案；当查询“景区门票如何预约”时，会直接调取预约要求与流程。
- 科研辅助场景：**科研人员可以在知识库中上传中英文文献摘要、实验数据记录、参考文献目录和行业标准与研究方法。撰写论文时，智能体通过向量召回相关文献摘要，提取核心结论与数据支撑；设计实验方案时，快速调取同类实验的参数设置和研究方法，对比优劣并优化实验设计，提升研究效率。
- 智能问答助手场景：**企业可以将产品文档、产品使用手册、用户高频咨询的问题合集、产品参数规格表、常见故障排查指南等信息上传至知识库。智能体能够通过精准匹配知识库中的相关信息准确解答用户的问题。

知识与记忆对比

知识：是指Agent通过数据输入获得的信息和理解。这些信息可以是非结构化的（如文档、文章）。

记忆：是指Agent在与用户交互过程中存储和检索的特定信息。这些信息通常是与特定用户或会话相关的，用于提供个性化的服务和上下文相关的响应。

### 相同点

- 信息存储  
知识和记忆都涉及信息的存储。无论是知识库中的数据还是会话中的临时信息，都是为了在需要时能够快速访问和使用。
- 支持决策  
知识和记忆都支持Agent在与用户交互时做出更准确和有效的决策。知识提供广泛的信息基础，而记忆提供具体的上下文信息。
- 动态性  
知识和记忆都是动态的，需要定期更新和维护。知识库需要定期更新以保持信息的准确性和时效性，而记忆需要在会话过程中不断更新以反映用户的最新交互。
- 提高用户体验  
知识和记忆都旨在提高用户体验。知识库提供全面的信息支持，帮助用户快速找到答案；记忆则提供个性化的服务，使交互更加自然和连贯。

### 不同点

- 存储方式  
知识：通常存储在外部介质中，如数据库、知识图谱、文档等。这些信息是结构化或非结构化的，可以长期保存和共享。  
记忆：通常存储在Agent的内部状态中，如会话状态、用户状态等。这些信息通常是临时的，会随着会话的结束或用户的注销而清除。
- 信息类型  
知识：涵盖广泛的信息，包括事实、原理、技能、方法等。这些信息是通用的，可以应用于多个用户和场景。  
记忆：涉及特定用户或会话的信息，如用户的查询历史、偏好、上下文信息等。这些信息是个性化的，特定于某个用户或会话。
- 更新频率  
知识：需要定期更新和维护，以保持信息的准确性和时效性。更新频率可能较低，但每次更新都涉及大量的信息。  
记忆：在每次交互过程中不断更新，以反映用户的最新行为和需求。更新频率较高，但每次更新涉及的信息量较小。
- 传递性  
知识：知识库可以共享给多个Agent，提高整体的服务质量。  
记忆：通常是特定于某个用户或会话的，不直接传递给其他用户或会话。记忆信息的传递通常需要通过用户标识和会话标识来实现。
- 应用场景  
知识：广泛应用于常见问题解答、产品信息、政策和法规、最佳实践等。这些信息是通用的，适用于多个用户和场景。  
记忆：主要应用于多轮对话、个性化推荐、用户识别、上下文理解等。这些信息是个性化的，特定于某个用户或会话。
- 个体差异  
知识：不同Agent的知识水平和获取途径可能不同，但知识本身是客观的，适用于所有用户。

记忆：不同用户的记忆内容和方式存在显著差异，受用户的特定需求和历史行为影响。

选型建议

| 类型     | 适用场景                        | 优势          | 劣势         |
|--------|-----------------------------|-------------|------------|
| 平台知识库  | 无现成知识库，希望开箱即用               | 集成度高，管理便捷   | 数据需上传至平台   |
| 第三方知识库 | 已有自建知识库（如KooSearch、RAGFlow） | 数据自主可控，无需迁移 | 需自行维护第三方服务 |

相关操作

- [创建平台知识库](#)。
- [接入第三方知识库](#)。
- 如何配置智能体记忆，请参考[为应用添加记忆](#)。

9.2 知识库类型

费用说明

- 平台知识库：不同套餐包的**知识存储空间容量**不同，具体可参考[计费说明](#)。
- 第三方知识库：AgentArts侧不计费，但第三方系统的部署与运维费用需用户自行承担，具体计费以第三方为准。

知识库类型

AgentArts支持以下两种类型知识库的管理：

- **平台知识库**：在AgentArts内直接创建并管理的知识库。支持上传文本文档、FAQ文档等文件，设置文档标签，并对其进行向量化存储、知识检索。适用于从零开始构建知识库，或数据量在平台限制范围内的场景。
- **第三方知识库**：将第三方系统中的知识库接入到AgentArts中。当前支持对接第三方知识库General、KooSearch和RAGFlow。适用于企业已有现成的知识库系统（如RAGFlow），希望直接复用其数据而无需重复上传的场景。接入后，Agent在检索时会直接调用第三方知识库的接口返回结果，数据不存储在AgentArts中。

选型建议

| 对比维度 | 平台知识库           | 第三方知识库          |
|------|-----------------|-----------------|
| 数据存储 | 存储在AgentArts平台。 | 存储在第三方系统。       |
| 使用场景 | 无现成知识库，需上传文档管理。 | 已有企业知识库，希望直接对接。 |
| 配置难度 | 低（上传即用）。        | 中（需配置连接参数）。     |

| 对比维度 | 平台知识库             | 第三方知识库                 |
|------|-------------------|------------------------|
| 计费模式 | 占用套餐包存储额度。        | AgentArts侧免费，第三方侧费用自理。 |
| 数据安全 | 数据存储在AgentArts平台。 | 数据保留在第三方系统，不传入平台。      |

相关文档

- [创建平台知识库](#)
- [接入第三方知识库](#)
- [创建第三方知识库时为什么查不到知识库列表？](#)
- [第三方知识库连接中的已启用/已停用状态如何更改？](#)

9.3 知识库使用限制

为保障平台资源合理分配和系统稳定性，知识库功能设有以下使用限制。

当前AgentArts团队标准版、团队高级版、企业专业版、企业旗舰版未开放，如果需要购买，请[提交工单](#)咨询购买相关事宜。

表 9-2 知识库使用限制

| 资源    | 描述                                         | 体验版        | 团队标准版      | 团队高级版      | 企业专业版      | 企业旗舰版      |
|-------|--------------------------------------------|------------|------------|------------|------------|------------|
| 知识库数量 | 单租户最多可创建平台知识库数量。                           | 5个知识库      | 10个知识库     | 20个知识库     | 50个知识库     | 100个知识库    |
|       | 单租户最多可创建第三方知识库连接数量（例如，General、RAGFlow）。    | 5个第三方知识库连接 | 5个第三方知识库连接 | 5个第三方知识库连接 | 5个第三方知识库连接 | 5个第三方知识库连接 |
|       | 单租户最多可接入第三方知识库数量（例如，RAGFlow平台下的知识库1、知识库2）。 | 50个第三方知识库  | 50个第三方知识库  | 50个第三方知识库  | 50个第三方知识库  | 50个第三方知识库  |
|       | 单个智能体最多可添加知识库数量。                           | 1个知识库      | 3个知识库      | 3个知识库      | 3个知识库      | 3个知识库      |
|       | 单个工作流最多可添加知识库数量。                           | 1个知识库      | 3个知识库      | 3个知识库      | 3个知识库      | 3个知识库      |
| 知识库文件 | 单个知识库上传的知识文档或FAQ文档总数量。                     | 500个文档     | 500个文档     | 500个文档     | 500个文档     | 500个文档     |

| 资源    | 描述                                                                                                                           | 体验版     | 团队标准版   | 团队高级版   | 企业专业版   | 企业旗舰版   |
|-------|------------------------------------------------------------------------------------------------------------------------------|---------|---------|---------|---------|---------|
|       | 上传知识文档、FAQ文档的单个文件大小限制。                                                                                                       | 20MB    | 60MB    | 60MB    | 60MB    | 60MB    |
|       | OBS接入文件的单个文件大小限制。                                                                                                            | 20MB    | 128MB   | 128MB   | 128MB   | 128MB   |
| 知识库容量 | 单租户知识库存储空间容量。                                                                                                                | 1GB     | 5GB     | 50GB    | 200GB   | 1200GB  |
| 召回数量  | 支持检索召回的最大文本切片数量。 <ul style="list-style-type: none"><li>召回指从知识库中检索并返回相关内容片段的过程。</li><li>文本切片指系统将文档自动切分成的片段，检索时按片返回。</li></ul> | 50个文本切片 | 50个文本切片 | 50个文本切片 | 50个文本切片 | 50个文本切片 |

 **注意**

达到上述限制时，相关操作将失败并提示限制信息。如需提升配额，请购买团队标准版、团队高级版、企业专业版或企业旗舰版。

选型建议

如果您有以下需求，建议选择**团队标准版、团队高级版、企业专业版或企业旗舰版**：

当前AgentArts**团队标准版、团队高级版、企业专业版、企业旗舰版**未开放，如果需要购买，请[提交工单](#)咨询购买相关事宜。

- 单个上传文件超过 20MB（**团队标准版、团队高级版、企业专业版或企业旗舰版**支持最大60MB）。
- 需要在单个智能体/工作流中关联多个知识库（**体验版**仅支持 1 个）。
- 需要更大的存储空间。

9.4 创建平台知识库

9.4.1 创建知识库

AgentArts提供的知识库功能对文本文档、FAQ等数据进行向量化存储、知识检索，支持为应用、工作流提供检索增强能力。

本文将详细介绍如何创建知识库，包含模型配置、解析配置和拆分配置等。

图 9-2 创建平台知识库流程

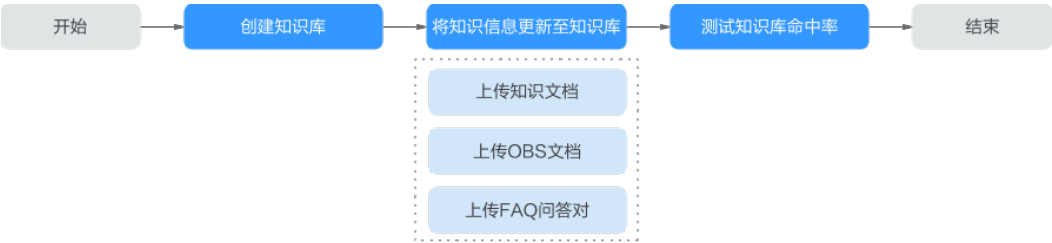


表 9-3 创建平台知识库流程

| 序号 | 流程环节                                                               |          | 说明                                                                                                                           |
|----|--------------------------------------------------------------------|----------|------------------------------------------------------------------------------------------------------------------------------|
| 1  | 创建知识库                                                              |          | AgentArts提供的知识库功能对文本文档、FAQ等数据进行向量化存储、知识检索，支持为应用、工作流提供检索增强能力。                                                                 |
| 2  | 将知识信息更新至知识库。<br>说明<br>AgentArts支持通过多种方式将知识信息更新至知识库，用户可根据需求选择合适的方式。 | 上传知识文档   | 在创建知识库时，您可以选择上传本地的知识文档到知识库。                                                                                                  |
|    |                                                                    | 上传OBS文档  | 在创建知识库时，支持通过对象存储服务（Object Storage Service, OBS）接入知识文档。<br>OBS是华为云提供的对象存储服务，具备数据存储能力。AgentArts知识库能够与OBS对接，将OBS中存储的文档数据上传至知识库。 |
|    |                                                                    | 上传FAQ问答对 | 知识库支持通过上传FAQ文档来批量导入FAQ问答对，也支持在线创建。                                                                                           |
| 3  | 测试知识库命中率                                                           |          | AgentArts通过对创建的知识库进行命中率测试，可以评估知识库的效果和准确性。                                                                                    |

费用说明

创建知识库及上传文档会占用知识库存储容量，知识库存储容量具体可参考[计费说明](#)。

文档导入过程中的向量化处理可能会产生模型调用费用，请关注配额及计费详情。

AgentArts体验版不计费。

AgentArts团队标准版/团队高级版/企业专业版/企业旗舰版：AgentArts侧不计费，接入MaaS模型即服务的模型，具体计费请参考[MaaS模型即服务价格](#)。

当前AgentArts团队标准版、团队高级版、企业专业版、企业旗舰版未开放，如果需要购买，请[提交工单](#)咨询购买相关事宜。

前提条件

已[开通AgentArts服务](#)。

约束与限制

不同套餐包可以创建平台知识库数量不同，详细信息请参考[知识库使用限制](#)。

创建知识库

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏中选择“托管与运行 > 知识库”。
- 步骤3 在“知识库”页面，单击“创建知识库 > 创建平台知识库”。
- 创建平台知识库：适用于数据在AgentArts平台直接管理的场景，支持上传文档和FAQ。
  - 接入第三方知识库：适用于企业已有现成知识库（如RAGFlow），希望直接对接复用的场景。具体操作请参考[接入第三方知识库](#)。
- 步骤4 在“创建知识库”的“基本信息”页面中，参见表9-4配置知识库信息。

图 9-3 创建知识库（以体验版为例）

1 基本信息

填写基本信息和解析配置项

2 上传文档

可批量上传文档资料

智果（AgentArts）智能体平台 / 知识库 / 创建知识库

创建知识库

基本信息

名称AgentArts平台知识库

知识库名称支持以中文、字母、数字开头，内容允许包含中文、字母、数字、中划线、下划线，长度1~50的字符串。

描述（可选）AgentArts平台知识库

知识库图标

支持上传png、jpeg、gif、jpg的图片，大小不能超过200 KB

解析配置

向量模型pangu\_embedding

检索模型pangu\_rerank

文档解析（可选）

☐ OCR增强

☐ 页面页脚解析

☒ 目录类解析

分块策略（可选）

自动分块

长度分块

层级分块

自动分块：按照系统默认预设的规则和分块策略切分，适用于格式规范的通用文档。详细说明请参考[知识库分块](#)

取消

创建并上传

立即创建

表 9-4 参数说明

| 参数   |        | 说明                                                                 | 示例             |
|------|--------|--------------------------------------------------------------------|----------------|
| 基本信息 | 名称     | 用于标识知识库。<br>由1~50个字符组成，包含中文、字母、数字、中划线、下划线，且必须以中文、字母、数字开头。          | AgentArts平台知识库 |
|      | 描述（可选） | 用于对知识库内容和用途的简要说明。它提供了关于知识库的详细信息，帮助用户了解知识库的内容和使用场景。<br>由0~100个字符组成。 | AgentArts平台知识库 |

文档版本 01 (2026-09-24)

版权所有 © 华为云计算技术有限公司

332



| 参数   |       | 说明                                                                                                                                                                                                                                                                                                                                                                                            | 示例              |
|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
|      | 知识库图标 | <p>系统默认知识库图标，可单击自动生成图标，也可以自定义图标。</p> <p>1. 鼠标移动至图标上，单击鼠标左键。</p> <p>2. 上传已准备好的知识库图标。</p> <p>支持jpg、jpeg、png、gif格式图片，且不大于200KB。</p>                                                                                                                                                                           | 默认              |
| 解析配置 | 向量模型  | <p>创建后，不支持修改。</p> <p>向量模型是一种将文本、图像等非结构化数据转换为数值向量的模型。例如，在文本处理阶段，用于对文本文档切片后的片段进行向量化表示；在知识检索阶段，根据用户输入的信息对切片进行召回。</p> <p>向量模型用于在海量的知识库中，快速识别和用户输入信息语义相近的词或句子，进行信息的初步筛选，解决“大海捞针”的效率问题。</p> <p><b>体验版：</b>系统预置向量模型pangu_embedding。</p> <p><b>团队标准版、团队高级版、企业专业版或企业旗舰版：</b>可以自定义使用MaaS模型即服务的向量模型。</p>                                                                                                 | pangu_embedding |
|      | 精排模型  | <p>精排模型是一种用于对检索结果进行精细排序的模型。针对用户输入的信息，对向量模型召回的切片进行从高到低的相关度排序，把相关度最高的前几个信息（例如Top 10）呈现给用户。</p> <p>精排模型用于进一步提升系统搜索的相关性准确度。</p> <p><b>体验版：</b>系统预置精排模型pangu_rerank。</p> <p><b>团队标准版、团队高级版、企业专业版或企业旗舰版：</b>可以自定义使用MaaS模型即服务的文本排序。</p> <p><b>说明</b></p> <p><b>体验版</b>用户在创建知识库后，无法修改精排模型的配置。</p> <p><b>团队标准版、团队高级版、企业专业版或企业旗舰版</b>用户在创建知识库后，可以通过“高级设置”选项来修改精排模型。具体操作步骤，请参考<a href="#">管理知识库</a>。</p> | pangu_rerank    |

| 参数 |          |        | 说明                                                                                                                                                                                                                                                         | 示例   |
|----|----------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|
|    | 文档解析（可选） | OCR增强  | <ul style="list-style-type: none"><li>未开启，不可调用OCR服务进行智能文档识别。</li><li>开启后，即可调用OCR服务进行智能文档识别，如表格解析或扫描文件等。</li></ul>                                                                                                                                          | 未开启  |
|    |          | 页眉页脚解析 | <ul style="list-style-type: none"><li>未开启，解析结果中不包含页眉页脚。</li><li>开启后，解析结果中包含页眉页脚。</li></ul>                                                                                                                                                                 | 开启   |
|    |          | 目录页解析  | <ul style="list-style-type: none"><li>未开启，解析结果中不包含目录页。</li><li>开启后，解析结果中包含目录页。</li></ul>                                                                                                                                                                   | 未开启  |
|    |          | 图片解析   | <ul style="list-style-type: none"><li>未开启，则在文档中遇到图片默认跳过，不处理图片。</li><li>开启后，根据需要选择“提取图片文本”或者“仅保留原图”。<ul style="list-style-type: none"><li>提取图片文本：识别图片内文字。</li><li>仅保留原图：仅提取图片保存，不会识别图片内容，便于问答图文展示。</li></ul></li></ul>                                      | 未开启  |
|    | 分段策略（可选） |        | <p>系统默认自动分段。支持如下分段策略，分段介绍及分段策略配置请参考<a href="#">知识库分段</a>。</p> <ul style="list-style-type: none"><li>自动分段：按照系统默认预设的规则和分隔符切分。适用于格式规范的通用文档。</li><li>长度分段：基于内容的长度来决定如何进行分段。适用于对分段长度有严格要求的场景。</li><li>层级分段：根据内容的结构层次来进行分段。适用于带有明确标题层级的结构化文档（如说明书、书籍）。</li></ul> | 自动分段 |

**步骤5** 配置完成后，单击“创建并上传”或“立即创建”。

- 创建并上传：进入“创建知识库”的“上传文档”页面，为知识库添加文档信息，具体操作请参考[上传文档](#)。
- 立即创建：页面返回知识库列表。

创建完成后，可以在“知识库”页签中查看创建完成的知识库，“类型”为“平台知识库”，“状态”为“已启用”。单击知识库名称进入知识库详情页进行后续配置。

----结束

## 相关文档

创建平台知识库应用实践，请参考[创建语文知识库](#)。

## 常见问题

- **自定义的向量模型一直正常使用，突然在列表中无法被选中或消失了。**  
登录[云搜索服务控制台](#)，在“模型管理”页面，检查该模型的“连通性”是否正常。只有状态显示为“正常”的模型，才能在AgentArts中被正常选择和使用。
- **单击“创建模型”时，页面跳转失败怎么办？**  
当前登录的账号没有[云搜索服务 CSS](#)的权限，请联系主账号管理员添加对应的权限。需要添加的权限项为：css:cluster:list，具体操作请参考[给IAM用户授权](#)。
- **为什么无法创建知识库？**  
套餐包知识库数量有限制，超过限制后无法再创建。用户可以删除闲置知识库后再创建。

## 9.4.2 上传文档

在创建知识库时，您可以选择本地上传知识文档，或通过对象存储服务（Object Storage Service, OBS）接入知识文档。本文将详细说明如何选择本地上传知识文档。

上传文档适用于文件数量少、单次上传的场景。

在创建知识库后，需上传文档至知识库，平台支持知识文档、FAQ问答对、FAQ文档三种形态的文档。

- **上传知识文档：**知识文档上传并解析完成后，用户可根据需要新增/编辑切片，以提升知识库的检索效率。如果用户对知识库内文档的解析切片效果不满意，可修改知识库模型配置、解析配置和拆分配置信息，并对知识库内文档重新解析拆分。
- **上传FAQ问答对：**FAQ（Frequently Asked Questions，常见问题解答）问答对是指常见问题及其对应的答案，用于快速解决用户可能遇到的问题。  
知识库支持通过上传FAQ文档来批量导入FAQ问答对。FAQ文档上传并解析完成后，用户可根据需要新增/编辑切片，以提升知识库的检索效率。

## 费用说明

文档导入过程中的向量化处理可能会产生模型调用费用，请关注配额及计费详情，具体可参考[计费说明](#)。

AgentArts**体验版**不计费。

AgentArts**团队标准版/团队高级版/企业专业版/企业旗舰版**：AgentArts侧不计费，接入MaaS模型即服务的模型，具体计费请参考[MaaS模型即服务价格](#)。

当前AgentArts**团队标准版、团队高级版、企业专业版、企业旗舰版**未开放，如果需要购买，请[提交工单](#)咨询购买相关事宜。

## 选型建议

- **知识文档：**适用于非结构化的长文档（如产品手册、操作指南、技术白皮书），系统会自动进行切片和向量化处理。
- **FAQ问答对/文档：**适用于结构化的问答数据（如Excel表格中的Q&A），能够精准匹配用户提问，适合客服场景。

## 前提条件

- 已[开通AgentArts服务](#)。
- 已完成[创建知识库](#)。

## 上传知识文档

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。

**步骤3** 在“知识库”页面，单击知识库列表所需知识库名称，进入该知识库详情页面。

**步骤4** 单击“添加知识”，“上传方式”选择“知识文档上传”。

- 知识文档上传：支持本地批量导入知识文档。
- OBS上传：可通过OBS路径批量导入文档。
- FAQ问答对：可通过导入文档上传，或在线创建问答对。

**步骤5** 在“上传文档”区域，单击“点此上传”，在弹出的对话框中，选择要上传的文档。

当上传的文档由“上传中”变为“上传成功”，表示文档已成功上传。

### 说明

- 支持上传常见的文档格式：
  - 文档/文本类：doc、docx、pdf、ofd、txt、md、wps。
  - 表格/数据类：xls、xlsx、csv、et。
  - 演示/幻灯片类：ppt、pptx、dps。
  - 图片/设计类：jpg、jpeg、png、gif、bmp、webp、tiff、tif、psd、ico、pcx。AgentArts支持用户根据需求，将图片类的文件直接存储，或是提取图片中的文字信息进行存储。
- 不同套餐包对文件大小限制不同，具体请参考[知识库使用限制](#)。

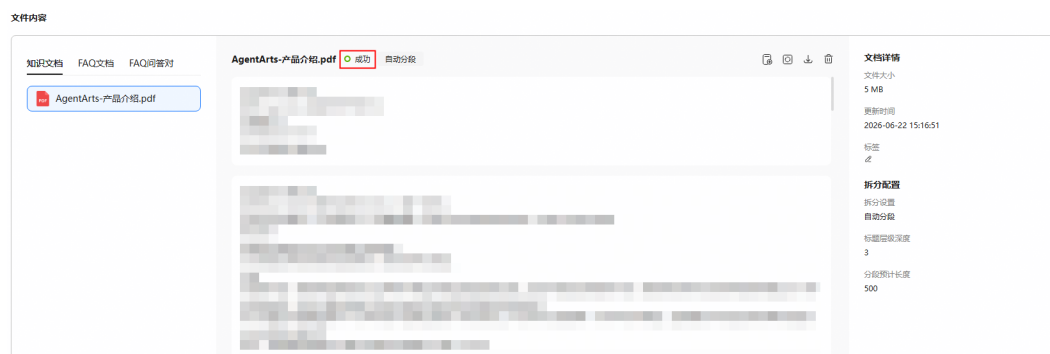
**步骤6** 单击“完成”。

返回知识库详情页面，文件上传后将自动进行解析，解析时间视文件大小和数量而定。如果状态长时间显示为“解析中”或“失败”，请检查文件格式是否损坏或联系技术支持工程师。

待文件状态为“成功”，即完成文件解析。

上传的知识文档，显示在知识库详情页面的“文件内容 > 知识文档”中。

图 9-4 文件解析成功



“文件内容”区域说明：

- 左侧：知识库文档的类别及上传的文档列表。
- 中间：知识文档的切片信息。
- 右侧：文档详情。

**步骤7** （可选）在“知识库详情”页面右上方，单击“... > 标签”，可添加文档标签（如科普、教育、科技等）用于对文档进行分类。

**步骤8** （可选）在“文档详情”区域，单击，为文档设置标签，可在工作流“知识检索”节点中配置按照标签检索文档。

----结束

## 上传 OBS 文档

通过OBS接入文档适用于文件数量较多、文件较大（超过本地上传限制），或数据已存储在OBS桶中的场景。

### 注意

需拥有已实名认证的华为账号，且该账号具有OBS桶的访问权限。

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。

**步骤3** 在“知识库”页面，单击知识库列表所需知识库名称，进入该知识库详情页面。

**步骤4** 单击“添加知识”，“上传方式”选择“OBS上传”。

- 知识文档上传：支持本地批量导入知识文档。
- OBS上传：可通过OBS路径批量导入文档。
- FAQ问答对：可通过导入文档上传，或在线创建问答对。

**步骤5** 配置OBS信息，如[图9-5](#)所示，参数配置请参考[表9-5](#)。

### 警告

OBS配置信息保存之后将无法修改。如果配置错误，需删除当前知识库并重新创建。一个知识库仅支持接入一次OBS文件，接入后不可编辑。

图 9-5 配置 OBS 信息



表 9-5 OBS 配置参数说明

| 参数    | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | 示例       |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|
| OBS桶名 | 选择存储知识文档的OBS桶。接入后不支持修改。                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | test01   |
| OBS路径 | 知识文档在OBS中的具体路径。接入后不支持修改。 <ul style="list-style-type: none"><li>接入的OBS目录下子目录层级不超过三层，如接入目录为path1，则path1最大目录层级为path1/path2/path3/path4/xxx.txt。</li><li>支持上传常见的文档格式：<ul style="list-style-type: none"><li>文档/文本类：doc、docx、pdf、ofd、txt、md、wps。</li><li>表格/数据类：xls、xlsx、csv、et。</li><li>演示/幻灯片类：ppt、pptx、dps。</li><li>图片/设计类：jpg、jpeg、png、gif、bmp、webp、tiff、tif、psd、ico、pcx。AgentArts支持用户根据需求，将图片类的文件直接存储，或是提取图片中的文字信息进行存储。</li></ul></li><li>不同套餐包对文件大小限制不同，具体请参考<a href="#">知识库使用限制</a>。</li></ul> | RMSLogs/ |

- 步骤6** 配置完成后，单击“立即执行”。
- 执行时间取决于文件数量和大小，请耐心等待。如果状态显示“执行失败”，请检查OBS桶权限是否正确配置或查看执行日志。
- 当执行状态显示为“执行成功”，即完成OBS接入知识文档。
- 步骤7** 单击“完成”。
- 返回知识库详情页面。
- 结束

上传 FAQ 问答对

- 步骤1** 登录AgentArts智能体平台。
- 步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。
- 步骤3** 在“知识库”页面，单击知识库列表所需知识库名称，进入该知识库详情页面。
- 步骤4** 单击“添加知识”，“上传方式”选择“FAQ问答对”。
- 知识文档上传：支持本地批量导入知识文档。
  - OBS上传：可通过OBS路径批量导入文档。
  - FAQ问答对：可通过导入文档上传，或在线创建问答对。
- 步骤5** 选择创建方式。

- “选择创建方式”选择“文档导入”。

单击“[点此上传](#)”，在弹出的对话框中选择符合“Excel模板”或“Word模板”要求的FAQ文档上传。

当上传的文档由“上传中”变为“上传成功”，表示文档已成功上传。

 **说明**

- 支持按照模板上传文件，文件格式仅支持xlsx、xls、docx、doc。
- 不同套餐包对文件大小限制不同，具体请参考[知识库使用限制](#)。
- Excel单个文件最大支持100,000条数据。
- FAQ文件中不允许空行，空行后的数据将被忽略。

- “选择创建方式”选择“在线创建”。

单击“添加问答对”，填写FAQ问答对信息，如图9-6所示，参数配置请参考表9-6。

图 9-6 FAQ 问答对

选择创建方式

文档导入

在线创建

问题列表

+ 添加问题 1/10

小儿肥胖该怎么治

问题详情

问题

小儿肥胖该怎么治

9/1,000

相关问题1

小儿肥胖的治疗需要综合考虑饮食、运动和行为习惯等多方面因素。建议咨询专业的儿科医生或营养师，根据孩子的具体情况制定个性化的治疗计划。

66/1,000

相关问题2

请输入

0/1,000

答案

孩子一旦患上肥胖和长期肥胖需要通过运动和饮食来改变孩子的情况。要让孩子做一些这个年龄的能做的运动，如游泳、慢跑等。要给孩子多吃一些蔬菜、猕猴桃、苹果、橙子等食物。禁止孩子吃高热量、高脂肪的食物，像蛋糕、干果、油炸饼干等，并严格控制孩子的零食。不要让孩子暴饮暴食。多运动对改变孩子肥胖都是有帮助的。在治疗了小儿肥胖病期间如果情况严重，建议家长先带孩子去医院检查孩子肥胖程度和原因再针对性的治疗。

190/1,000

### 表 9-6 参数说明

| 参数   | 说明                                                             | 示例                                                                 |
|------|----------------------------------------------------------------|--------------------------------------------------------------------|
| 标准问题 | <p>必选参数。</p> <p>用户可能提出的问题的最直接、最清晰的形式。</p> <p>由1~1000个字符组成。</p> | 小儿肥胖该怎样医治                                                          |
| 相似问题 | <p>可选参数。</p> <p>与标准问题意思相近或相关的一系列问题。</p> <p>由1~1000个字符组成。</p>   | 小儿肥胖的治疗需要综合考虑饮食、运动和行为改变等多方面因素。建议咨询专业的儿科医生或营养师，根据孩子的具体情况制定个性化的治疗计划。 |



| 参数 | 说明                                        | 示例                                                                                                                                                                                             |
|----|-------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 答案 | 必选参数。<br>针对标准问题提供的详细解答。<br>由1~10000个字符组成。 | 孩子一旦患上肥胖症家长要先通过运动和饮食来改变孩子的情况，要让孩子做一些他这个年龄段能做的运动，如游泳，慢跑等，要给孩子多吃一些像苹果，猕猴桃，胡萝卜等食物，禁止孩子吃高热量，高脂肪的食物，像蛋糕，干果，曲奇饼干等，严格地控制孩子的饮食，不要让他暴饮暴食，多运动对改变孩子肥胖都是有好处的，在治疗小儿肥胖期间如果情况严重，建议家长先带孩子去医院检查孩子肥胖症的原因再针对性的治疗。 |

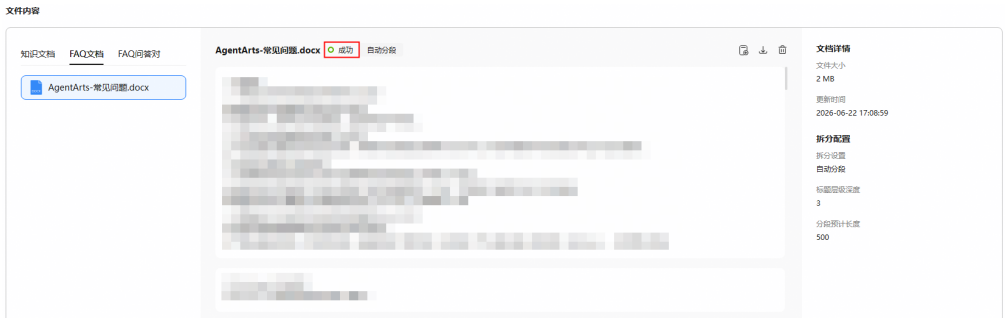
**步骤6** 单击“完成”。

返回知识库详情页面，文件上传后将自动进行解析，解析时间视文件大小和数量而定。如果状态长时间显示为“解析中”或“失败”，请检查文件格式是否损坏或联系技术支持工程师。

待文件状态为“成功”，即完成文件解析。

- 上传的FAQ文档，显示在知识库详情页面的“文件内容 > FAQ文档”中。

**图 9-7** 文件解析成功



“文件内容”区域说明：

- 左侧：知识库文档的类别及上传的文档列表。
  - 中间：知识文档的切片信息。
  - 右侧：文档详情。
- 添加的FAQ问答对，显示在知识库详情页面的“文件内容 > FAQ问答对”中，左侧显示问题，右侧显示问答对。  
单击内容，可以修改。



图 9-8 FAQ 问答对



----结束

常见问题

为什么文档上传失败？

- 可能原因：
  - 文档格式不符合格式要求。
  - 文档大小超过限制。
- 处理措施：
  - 支持上传常见的文档格式：
    - 文档/文本类：doc、docx、pdf、ofd、txt、md、wps。
    - 表格/数据类：xls、xlsx、csv、et。
    - 演示/幻灯片类：ppt、pptx、dps。
    - 图片/设计类：jpg、jpeg、png、gif、bmp、webp、tiff、tif、psd、ico、pcx。  
AgentArts支持用户根据需求，将图片类的文件直接存储，或是提取图片中的文字信息进行存储。
  - 不同套餐包对文件大小限制不同，具体请参考[知识库使用限制](#)。

为什么文档解析失败？

- 可能原因：文档格式损坏或OCR模型不可用。
- 处理措施：请确认文件格式正常，如使用了基于KooSearch的模型，请在KooSearch服务侧检查OCR模型状态并修复。

相关操作

在知识库详情页面，在“文件内容 > 知识文档”、“文件内容 > FAQ文档”、“文件内容 > FAQ问答对”页签，单击指定文件或问题，可以查看文档或问题详情。在文档或问题详情页面，支持的其他操作请参考[表9-7](#)。

表 9-7 相关操作

| 操作        | 说明                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 新增切片      | <p>单击，新增切片。</p> <p><b>说明</b><br/>仅知识文档、FAQ文档支持新增切片。</p> <ul style="list-style-type: none"><li>● <b>切片标题：</b>用于快速了解每个切片的内容，便于在大量切片中进行查找和管理。由1~1000个字符组成。<br/>示例：<br/>1 什么是AgentArts</li><li>● <b>切片内容：</b>通过切片内容，用户可以详细阅读和理解每个知识点或信息。由1~6000个字符组成。<br/>示例：<br/>包含了“1 什么是AgentArts”及其子章节“AgentArts的使用限制”的内容。</li></ul> |
| 重新对文档解析拆分 | <p>单击，在弹出的对话框中，单击“重试”即进行文档重新解析切片。重试任务在知识库详情页面的“... &gt; 日志”中查看。</p> <p>解析时间视文件大小和数量而定。如果状态长时间显示为“解析中”或“失败”，请检查文件格式是否损坏或联系技术支持工程师。</p> <p><b>说明</b><br/>重新解析和向量化处理，该过程会消耗计算资源，请谨慎操作。<br/>仅知识文档支持重新解析拆分。</p>                                                                                                           |
| 下载知识文档    | <p>单击，可以下载知识文档、FAQ文档。</p> <p><b>说明</b><br/>仅知识文档、FAQ文档支持下载文档。</p>                                                                                                                                                                                                                                                  |
| 删除知识文档    | <p><b>警告</b><br/>删除后数据将无法恢复，请谨慎操作。删除知识文档、FAQ文档、FAQ问答对将影响依赖该知识库的智能体回答效果。</p> <p>单击，可以删除知识文档、FAQ文档、FAQ问答对。</p>                                                                                                                                                                                                       |
| 编辑文档切片    | <ul style="list-style-type: none"><li>● 鼠标移动至待编辑的切片信息区域，单击，可以修改知识文档、FAQ文档切片。</li><li>● 单击，可以编辑FAQ问答对。</li></ul>                                                                                                                 |
| 删除文档切片    | <p>鼠标移动至待删除的切片信息区域，单击，可以删除知识文档、FAQ文档切片。</p>                                                                                                                                                                                                                                                                        |
| 修改标签      | <p>在“文档详情”区域，单击，可以修改标签。</p> <p><b>说明</b><br/>仅知识文档支持标签功能。</p>                                                                                                                                                                                                                                                      |

9.4.3 知识库命中测试

AgentArts支持对创建的知识库进行命中率测试，以评估知识库的效果和准确性。

建议在知识库创建完成后、正式用于智能体调用前，先进行命中测试验证效果。

本文将详细介绍知识库创建完成后如何执行命中测试操作。

## 典型场景

在智能客服场景中，通过命中测试验证知识库能否准确召回用户常见问题的相关文档，确保正式上线后检索效果符合预期。

## 费用说明

知识库检索召回依赖于创建知识库时设置的向量模型和精排模型。

AgentArts**体验版**不计费。

AgentArts**团队标准版/团队高级版/企业专业版/企业旗舰版**：AgentArts侧不计费，接入MaaS模型即服务的模型，具体计费请参考[MaaS模型即服务价格](#)。

**当前AgentArts团队标准版、团队高级版、企业专业版、企业旗舰版未开放，如果需要购买，请[提交工单](#)咨询购买相关事宜。**

## 前提条件

已完成[上传文档](#)。

## 命中测试

系统将输入的问题转化为向量，并在知识库中查找与之语义最相近的文档切片，根据相似度计算匹配分值。

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。

**步骤3** 在“知识库”界面，单击需要测试的知识库操作列的“命中测试”；或当知识库以卡片形式展示时，将鼠标悬浮在需要命中测试的知识库卡片上，单击“命中测试”。

**步骤4** 在命中测试页面左侧文本框中输入问题，单击“命中测试”。

在页面右侧将根据不同的检索方式（语义检索、关键词检索、混合检索、FAQ检索），展示多条匹配的内容，并按照匹配分值降序排列。

- 语义检索：理解语义，召回相关内容。适用于自然语言提问。
- 关键词检索：精确匹配，结果可控。适用于精确匹配特定名词。
- 混合检索：结合语义检索和关键词检索的优势。适用于需要兼顾语义理解和精确匹配的场景。
- FAQ检索：直接返回预设答案。适用于标准问答对场景。

**步骤5** 用户可以根据分值与匹配到的信息数量来评估当前知识库是否满足需求。

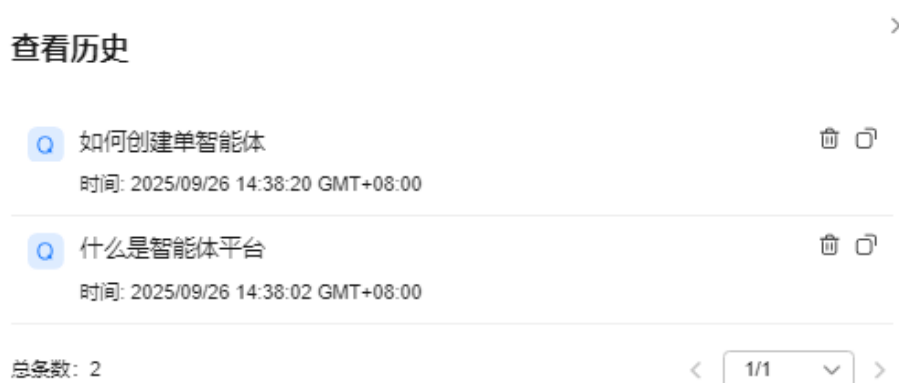
匹配分值范围为0~1，分值越接近1表示相关性越高。

如果测试结果为空或分值过低，建议检查文档切片设置是否合理，或尝试更换提问关键词。

系统将根据知识库配置自动选择检索方式，或您可以切换检索方式进行对比测试。

**步骤6** 单击右上角的“查看历史”，可以查看用户输入的历史问题。在历史记录右侧，单击可删除或复制该记录。

图 9-9 查看历史



----结束

## 常见问题

### 如何优化知识库检索效果？

1. 优化文档结构：无结构的文档，转化为有层次结构的文档。
2. AgentArts团队标准版、团队高级版、企业专业版或企业旗舰版支持更换接入的其他模型。
3. 修改知识库的分段设置。

### 为什么检索不到已上传的文档？

- 可能原因：
  - 阈值设置太高。
  - 检索词语、模式不合适。
  - 精排模型不可用。
- 解决措施：
  - **调整阈值大小**。
  - 修改检索词语、模式。
  - 在**KooSearch服务**中确认精排模型正常。

## 9.4.4 知识库分段

### 知识库分段介绍

对知识库进行分段设置主要是为了提高信息的组织性、可读性和检索效率。具体来说，分段设置的优势包括：

- 提高可读性：将长篇内容分成多个段落，每个段落集中讨论一个主题或一个方面，可以使读者更容易理解和消化信息。长篇大论往往会让读者感到疲惫，而分段则可以提供自然的停顿点，帮助读者更好地吸收信息。

- 增强组织性：通过分段，可以将相关的信息归类在一起，使知识库的内容结构更加清晰。这种结构化的组织方式不仅有助于作者在编写时保持逻辑清晰，也方便读者快速找到他们感兴趣的内容。
- 提高检索效率：对于大型知识库而言，分段设置可以与索引、标签等机制结合使用，帮助用户更快地定位到特定的信息。例如，搜索引擎可以更容易地识别出每个段落的关键内容，从而提高搜索结果的相关性和准确性。
- 便于维护和更新：分段的知识库内容更容易进行维护和更新。当需要修改或添加信息时，可以针对特定的段落进行操作，而不会影响到其他部分的内容，这有助于保持知识库的准确性和时效性。
- 适应不同的阅读习惯：不同的人有不同的阅读习惯和偏好。分段设置可以满足不同用户的需求，比如一些用户可能喜欢快速浏览标题和段落首句来获取概览，而另一些用户则可能更倾向于深入阅读每个段落的详细内容。

合理的分段能平衡上下文完整性与检索精准度。切片过大可能引入噪声，切片过小可能丢失上下文关联。

## 分段策略

如果您不确定如何选择，推荐优先使用自动分段；如果文档具有明确的书籍目录结构，推荐使用层级分段。

图 9-10 分段原始文档部分截图



表 9-8 分段策略

| 分段类型 | 自动分段                                                                                                                                                                                                      | 长度分段                                                                                                      | 层级分段                                                                                                        |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| 分段原则 | AgentArts可对上传的内容进行自动分段，支持复杂布局的文件处理，例如： <ul style="list-style-type: none"><li>可识别段落。</li><li>可识别页眉/页脚/脚注等非重点内容。</li><li>支持跨页的段落合并。</li><li>支持解析表格中的图片信息。</li><li>支持解析文档中的表格内容（目前，仅支持解析带线框的表格内容）。</li></ul> | 根据用户需求，灵活配置分段标识符、分段最大长度、分段重叠度等参数，还能通过设置文本预处理规则，在分段前对文本进行特定处理。                                             | 根据文档的目录结构、章节划分等层级信息，将内容划分为不同层级的文本单元。                                                                        |
| 分段优点 | 系统预设，无需额外配置，提高使用效率。                                                                                                                                                                                       | 严格控制分段长度，节约大模型会话时的Tokens。                                                                                 | 结构清晰，便于用户理解，提高检索效率。                                                                                         |
| 分段缺点 | 自动分段的效果依赖文档质量。                                                                                                                                                                                            | 配置复杂，不适用于所有文档，需要用户对文档内容和参数规则有较深的理解。                                                                       | 需要文档具有明确的层级结构，对于结构不规范的文档难以适用。                                                                               |
| 分段效果 | <b>图 9-11 效果良好</b><br>                                                                                                 | <b>图 9-12 效果差</b><br> | <b>图 9-13 效果良好</b><br> |
| 适用场景 | 适用于大部分分段场景，通常文档结构较为规范，是使用情况最多的分段类型。                                                                                                                                                                       | 适用于对分段长度有严格要求的场景。                                                                                         | 适用于结构层次分明的知识体系，例如技术手册、法律条文、标准规范等。这些文档通常具有明确的结构层次，需要按照章节、小节等进行组织和检索。                                         |

分段策略配置

修改分段策略后，系统将对知识库内文档重新进行解析和向量化处理，该过程会消耗计算资源，请谨慎操作。

 **警告**

修改分段策略将导致知识库内现有文档切片失效，系统将自动重新切分。如有重要的人工编辑切片，请提前备份或确认。

- 自动分段：系统默认自动分段。选择该策略时会自动利用句号、分号、问号、感叹号等标点符号作为分段依据，将文档分割成一个个独立的句子或段落，无其他配置项。
- 长度分段：选择该分段策略，完成以下配置。
  - 分段标识符：分段方式为遇到所选符号即截断，符号之间没有优先级，最终分割后合并到预计最大长度。  
支持中文句号、英文句号、中文感叹号、英文感叹号、中文问号、英文问号、空格、中文逗号、英文逗号。

 **注意**

长度分段配置中，如果未命中分段标识符，分段将会失败。

- 分段预计长度：分段的最大长度。文档的正文如果大于设定的最大长度，则截取最大长度的片段为新文档，随后回溯分段重叠字符，继续向后检查，直到文档结束。取值范围为1~6000，默认值为500。  
例如，设置长度500，当段落长800时，将切分为两段，第一段为0~497/502，第二段为498/503~800。
- 层级分段：选择该分段策略，完成以下配置。
  - 层级解析模型：
    - 自动解析：自动识别和解析具有层级结构的数据或信息。示例请参考[自动解析示例](#)。
    - 规则解析：支持添加自定义层级规则。示例请参考[规则解析示例](#)。
  - 标题层级深度：指设置的切分标题级别，例如，文本包含最多5级标题，选择的标题层级深度为3，则会分别将所有3级标题下的内容合并成文本块，文本块作为一个整体执行后续切分操作。输入值必须在1到10之间。
  - 标题保存方式：指标题信息在切片中的保存形式，影响检索结果的展示逻辑和索引构建方式。
    - 保存多标题组合：多级标题用空格组合。  
1级标题 2级标题 3级标题 ... 最后一级标题  
文本
    - 保存最后一级标题：仅最后一级标题。  
最后一级标题  
文本
  - 跨标题合并：根据需求开启或者关闭。
    - 开启“跨标题合并”功能：当不同标题下的段落文字较少时，平台会自动将其合并到指定的分段长度，有助于生成更加全面的内容。
    - 关闭“跨标题合并”开关：不会自动合并不同标题下的内容。
  - 分段标识符：分段方式为遇到所选符号即截断，符号之间没有优先级，最终分割后合并到预计最大长度。自定义分段中如果未命中分段标识符，分段将



会失败。支持中文句号、英文句号、中文感叹号、英文感叹号、中文问号、英文问号、空格、中文逗号、英文逗号。

- 分段预计长度：分段的最大长度。文档的正文如果大于设定的最大长度，则截取最大长度的片段为新文档，随后回溯分段重叠字符，继续向后检查，直到文档结束。

取值范围为1~6000，默认值为500。

配置完成后，您可以在知识库详情页的“切片管理”中查看分段效果，或通过“检索测试”验证检索精准度。

自动解析示例

图 9-14 配置

分段策略（可选）  

自动分段 长度分段 层级分段

层级解析模型  

自动解析 规则解析

标题层级深度  

3

标题保存方式  

保存多标题组合 保存最后一级标题

跨标题合并  

开启

分段标识符  

中文句号。 英文句号。 中文感叹号！ 英文感叹号！ 中文问号？ 英文问号？

分段预计长度  

1 6000 500

图 9-15 自动解析效果

文件内容

知识文档 FAQ文档 FAQ问答对

语文知识（新）.docx

语文知识（新）.docx 成功 层级分段

语文知识（新）  
1 第一部分：基础知识 1.1 下列词语中，字形和加点字的读音全部正确的一项是（ ）  
A. 蜿蜒 (wān) 蜿蜒 (wān) 人声鼎沸 (dǐng) 人声鼎沸 (dǐng)  
B. 蜿蜒 (wān) 蜿蜒 (wān) 人声鼎沸 (dǐng) 人声鼎沸 (dǐng)  
C. 蜿蜒 (wān) 蜿蜒 (wān) 人声鼎沸 (dǐng) 人声鼎沸 (dǐng)  
D. 蜿蜒 (wān) 蜿蜒 (wān) 人声鼎沸 (dǐng) 人声鼎沸 (dǐng)  
答案：B  
解析：  
A项，“蜿蜒”应为“蜿蜒”，  
C项，“蜿蜒”的“蜿”正确读音应为“wān”，  
D项，“蜿蜒”的“蜿”正确读音应为“wān”，  
B项全部正确。

语文知识（新）  
1 第一部分：基础知识 1.2 下列句子中，加点的成语使用恰当的一项是（ ）  
A. 他创作计算机非常熟练，已经到了为所欲为的程度。  
B. 这部小说情节曲折，人物形象生动，确实引人入胜。  
C. 同学们经常向老师请教问题，这种不耻下问的精神值得提倡。  
D. 同学们在考试中作弊，这种行为是不道德的。  
答案：B  
解析：  
A项，“为所欲为”应为“为所欲为”，  
C项，“不耻下问”应为“不耻下问”，  
D项，“不耻下问”应为“不耻下问”，  
B项全部正确。

文档版本 01 (2026-09-24)

版权所有 © 华为云计算技术有限公司

348



规则解析示例

图 9-16 配置



图 9-17 规则解析效果



相关操作

知识库分段用于[创建知识库](#)，为知识库设置“拆分配置”。

9.5 接入第三方知识库

9.5.1 第三方知识库分类

当您的企业数据已存储在现有的知识库系统中时，无需将数据迁移至AgentArts，通过接入第三方知识库可直接调用现有数据，实现知识复用。接入完成后，AgentArts智能体在运行时可通过标准接口调用第三方知识库中的数据，无需数据迁移。调用时，AgentArts将检索请求发送至第三方知识库，获取检索结果后返回给智能体使用。

支持将第三方系统中的知识库接入到AgentArts中。目前支持的知识库类型请参考[表 9-9](#)。

表 9-9 第三方知识库分类说明

| 知识库类型     | 说明                                                                                                                                  |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------|
| General   | 通用外部知识库。用户需要先将知识库按 <a href="#">第三方通用知识库接入规范</a> 进行适配，再接入。接入操作请参考 <a href="#">连接General知识库</a> 。<br>适用于自研知识库或非标准接口的系统，需要开发者自行开发适配接口。 |
| KooSearch | 可以直接接入KooSearch知识库。接入操作请参考 <a href="#">连接KooSearch知识库</a> 。<br>适用于已部署了该开源项目的用户，可实现零代码快速对接。                                          |
| RAGFlow   | 可以直接接入RAGFlow知识库。接入操作请参考 <a href="#">连接RAGFlow知识库</a> 。<br>适用于已购买或部署了对应产品的用户，可实现零代码快速对接。                                            |

选型建议

| 分类                    | 适用场景                                          | 接入方式                                      | 复杂度 |
|-----------------------|-----------------------------------------------|-------------------------------------------|-----|
| General               | 如果您使用的是自研或其他第三方知识库，请选择General类型并参考接入规范进行适配开发。 | 先按 <a href="#">第三方通用知识库接入规范</a> 进行适配，再接入。 | 中   |
| KooSearch/<br>RAGFlow | 如果您已拥有KooSearch或RAGFlow知识库，推荐直接选择对应类型以实现快速对接。 | 直接接入。                                     | 低   |

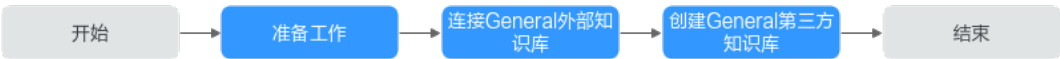
9.5.2 连接 General 知识库

适用于企业已有现成知识库系统（如自研知识库、RAGFlow等），希望直接对接复用，避免数据迁移的场景。

目前业界有诸多知识库种类，但并没有一个统一的规范。对于用户自开发的知识库平台，AgentArts提供了[第三方通用知识库接入规范](#)。知识库的开发者参考此规范进行适配后，即能够作为第三方知识库接入AgentArts平台。

本文将介绍如何在AgentArts对接第三方General知识库。

图 9-18 连接 General 知识库流程



费用说明

调用第三方知识库接口可能产生第三方平台的相关费用，请确保第三方服务账户状态正常。

约束与限制

警告

删除外部知识库连接前，请确认无智能体正在使用该连接下的知识库，否则将导致智能体问答失败。

当资源数量达到上限时，请删除不再使用的资源以释放配额：

- 第三方知识库连接：**进入“第三方知识库连接”页面进行删除。具体操作请参考[管理知识库](#)中的“删除外部知识库连接信息”。删除后，基于该连接接入的知识库将不可用。
- 第三方知识库：**进入“知识库”页面进行删除。具体操作请参考[管理知识库](#)中的“取消接入”。

关于知识库的更多约束与限制，请参考[知识库使用限制](#)。

步骤一：准备工作

- 用户拥有一个自开发的知识库系统（下文统称第三方知识库），并且已经按[第三方通用知识库接入规范](#)完成适配。知识库的服务接口发布到公网IP，并且可访问。
- 获取已搭建的第三方知识库连接信息：
  - 获取服务器接口地址，即第三方知识库接口的发布地址IP，例如，http://10.0.0.0:8080。
  - 获取鉴权信息密钥，该密钥是第三方知识库平台用户的认证信息，由第三方知识库提供。例如，123456789。

说明

- 用户填入时不需要在密钥开始处拼接Bearer，AgentArts平台会负责拼接操作。
- 获取知识库详情页面链接，登录第三方知识库平台后，进入知识库详情页面，复制地址栏的URL，将其中的知识库ID替换为{{id}}。以下图RAGFlow为例，其知识库详情页面链接为：http://10.0.0.0/dataset?id={{id}}。该链接信息用于接入第三方知识库后，AgentArts用户可根据此链接跳转第三方知识库平台页面。如果第三方知识库只有后端服务器，没有web或前端页面，可填写http://xxx.com作为占位。

图 9-19 RAGFlow 知识库详情页



## 步骤二：连接 General 第三方知识库

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。

**步骤3** 在“知识库”页面，单击“第三方知识库连接”。

**步骤4** 在“第三方知识库连接”界面，单击“新建知识库连接”。

**步骤5** 在“创建知识库连接”页面，设置连接General知识库基本信息，参数配置请参考[表 9-10](#)。

图 9-20 连接外部知识库

创建知识库连接

×

基本信息

选择类型 ?

KooSearch

用于对接华为云企业搜索服务：KooSearch

RAGFlow

用于对接开源项目RAGFlow

General

用于对接用户自开发知识库平台

知识库名称

General

描述（可选）

General知识库

图标

支持上传png、jpeg、gif、jpg的图片，大小不能超过200 KB

连接信息

服务地址

http://

×

知识库服务地址

鉴权信息密钥（连接信息若变动需重新填写） ?

认证鉴权，在http/https的请求header中添加的用户鉴权信息密钥

知识库详情页链接

http://

×

知识库详情页面链接，如https://xxx?id={id}。请注意使用占位符{id}表示知识库ID，否则无法跳转到对应的知识库页面

测试连接

取消

确定

表 9-10 参数说明

| 参数   |        | 说明                                                                                                                                                                                                 | 示例                           |
|------|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------|
| 基本信息 | 选择类型   | <ul style="list-style-type: none"><li>General：用于对接用户自研知识库平台，需自行实现检索接口。</li><li>KooSearch：用于对接华为云企业搜索服务，开箱即用。</li><li>RAGFlow：用于对接开源项目RAGFlow，快速对接复用。</li></ul>                                     | General                      |
|      | 知识库名称  | 用于标识知识库。<br>由1~50个字符组成，包含中文、字母、数字、中划线、下划线，且必须以中文、字母、数字开头。                                                                                                                                          | General                      |
|      | 描述（可选） | 用于简要说明知识库内容和用途。它提供了关于知识库的详细信息，帮助用户了解知识库的内容和使用场景。<br>由0~255个字符组成。                                                                                                                                   | General<br>知识库               |
|      | 图标     | 系统默认知识库图标，可单击  自动生成图标，也可以自定义图标。<br>1. 鼠标移动至图标上，单击鼠标左键。<br>2. 上传已准备好的知识库图标。<br>支持jpg、jpeg、png、gif格式图片，且不大于200KB。 | 默认                           |
| 连接信息 | 服务地址   | 能够访问检索接口及查询列表接口的地址，以https://或http://开头。<br>填入按 <a href="#">获取服务器接口地址，即第三方知识库接口的发布地址...</a> 获取的信息。                                                                                                  | http://<br>10.0.0.0:<br>8080 |

| 参数 |               | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | 示例                                |
|----|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|
|    | 鉴权信息<br>密钥    | <p>在http/https的请求header中添加的用户鉴权信息密钥。</p> <p>填入按<a href="#">获取鉴权信息密钥，该密钥是第三方知识库平台用户的...</a>获取的信息。</p> <p><b>注意</b><br/>请确保鉴权密钥来源安全，不要在非安全环境下泄露。</p> <p><b>关于密钥加密：</b>鼠标移动至“鉴权信息密钥”右侧的上，可查看密钥的加密方式。平台默认使用<b>KMS加密 + 默认密钥</b>方式对认证密钥进行加密存储，防止信息泄露。</p> <ul style="list-style-type: none"><li>加密机制：使用用户选择的密钥派生DEK进行加密，默认密钥为kms-agentarts/default。</li><li>费用说明：实例不收费，每月提供20,000次免费调用额度。解密时支持缓存，但敏感数据较多时可能超出免费次数。如需更换加密方式，具体请参考<a href="#">更换密钥的加密方式</a>。超出后按实际调用次数计费，详见<a href="#">密码安全中心计费说明</a>。</li></ul> | 123456789                         |
|    | 知识库详情<br>页面链接 | <p>第三方General知识库详情页面的链接，可通过该页面直接访问General知识库的详情页面。</p> <p>填入按<a href="#">获取知识库详情页面链接，登录第三方知识库平台后，...</a>获取的信息。</p> <p><b>注意</b><br/>需要使用占位符{{id}}表示知识库ID，否则无法跳转到对应的知识库页面。</p>                                                                                                                                                                                                                                                                                                                                                                                                             | http://10.0.0.0/dataset?id={{id}} |

- 步骤6 单击“测试连接”，弹出“测试成功”提示。
- 如果显示“第三方知识库连接失败，请检查连接地址和认证信息”，请参考[连接外部知识库失败如何排查？](#)。
- 步骤7 测试成功后，单击“确定”。
- 页面返回第三方知识库连接列表，连接创建成功后“状态”为“已停用”，完成[步骤三：创建General第三方知识库](#)后“状态”变为“已启用”。
- 结束

步骤三：创建 General 第三方知识库

在完成General知识库连接后，您可以将General中的知识库接入到AgentArts中进行使用。

已完成[步骤二：连接General第三方知识库](#)。

- 步骤1 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。

**步骤3** 在“知识库”页面，单击“创建知识库 > 接入第三方知识库”。

- 创建平台知识库：适用于数据在AgentArts平台直接管理的场景，支持上传文档和FAQ。具体操作请参考[创建平台知识库](#)。
- 接入第三方知识库：适用于企业已有现成知识库（如RAGFlow），希望直接对接复用的场景。

**步骤4** 在“接入第三方知识库”界面中，选择已接入的知识库，在右侧勾选知识库。

#### 说明

如果列表中没有需要的第三方知识库，则可以单击“新建”，新建知识库连接，具体配置请参考[表9-10](#)。



图 9-21 接入第三方知识库



**步骤5** 单击“确定”，第三方知识库创建完成。

创建完成的知识库，显示在知识库列表，“类型”为“General”，“状态”为“已启用”。

----结束

相关操作

在“第三方知识库连接”页面，支持的其他操作请参考[表9-11](#)。

表 9-11 相关操作

| 操作           | 说明                                                                         |
|--------------|----------------------------------------------------------------------------|
| 编辑第三方知识库连接信息 | 在对应的“操作”列下，单击“编辑”，可以编辑外部知识库连接信息。<br>如遇连接失败，请参考 <a href="#">各个章节</a> 的常见问题。 |
| 删除第三方知识库连接信息 | <b>注意</b><br>需要先取消接入后，才可以删除。<br>在对应的“操作”列下，单击“删除”，可以删除外部知识库连接信息。           |

常见问题

操作过程中如遇到以下问题，可参考对应解决方案：

- [创建第三方知识库时为什么查不到知识库列表？](#)
- [外部知识库连接中的已启用/已停用状态如何更改？](#)
- [连接外部知识库失败如何排查？](#)
  - 接口返回401/403：检查鉴权信息密钥（API Key）是否正确。
  - 连接超时：检查URL是否正确。
  - 请检查General服务是否支持在公网使用API访问。

相关文档

- 连接外部知识库后，请参考[知识库命中测试](#)校验知识库是否可用。
- 连接外部知识库后，可以对外部知识库进行管理，具体请参考[管理知识库](#)。

9.5.3 连接 KooSearch 知识库

为什么要连接外部知识库？

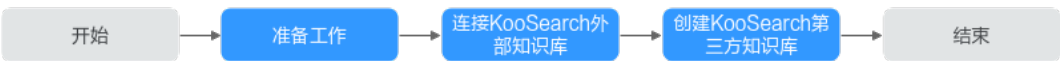
当企业已在KooSearch等平台构建了成熟的知识库，通过外部知识库连接可直接复用现有数据，无需重复上传，同时可利用KooSearch的联网增强能力获取公域知识。

AgentArts支持连接外部知识库平台，以使用户访问和利用外部的知识资源。

KooSearch可以帮助企业搭建企业级RAG，提升用户获取企业私有知识的效率，同时提供联网增强功能，提供互联网公域知识问答能力。主要用于智能客服应用、数字人、数字员工、AI助手、AI搜索等知识问答场景。

本文将介绍如何在AgentArts对接第三方KooSearch知识库。

图 9-22 连接 KooSearch 知识库流程



费用说明

通过AgentArts调用KooSearch知识库会消耗KooSearch的API调用次数及流量，请关注KooSearch服务的资源使用情况。

约束与限制



警告

删除外部知识库连接前，请确认无智能体正在使用该连接下的知识库，否则将导致智能体问答失败。

当资源数量达到上限时，请删除不再使用的资源以释放配额：

- **第三方知识库连接：**进入“第三方知识库连接”页面进行删除。具体操作请参考[管理知识库](#)中的“删除外部知识库连接信息”。删除后，基于该连接接入的知识库将不可用。
- **第三方知识库：**进入“知识库”页面进行删除。具体操作请参考[管理知识库](#)中的“取消接入”。

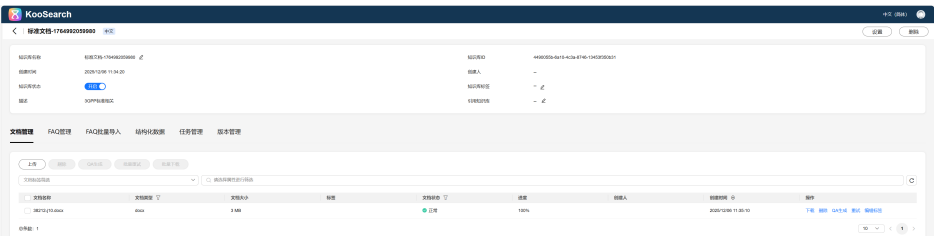
关于知识库的更多约束与限制，请参考[知识库使用限制](#)。

步骤一：准备工作

获取第三方KooSearch知识库的连接信息：

1. 确保需要连接的第三方KooSearch知识库平台已经开通了华为云KooSearch服务，并购买了相关资费套餐。第三方KooSearch的拥有者需要发布[搜索知识库](#)和[获取知识库列表](#)的两个API接口供后续使用者在AgentArts平台调用。发布API接口请参考[通过API使用KooSearch实现搜索问答](#)。
2. 在KooSearch中创建一个知识库，并上传相关文档。

图 9-23 KooSearch 中的知识库



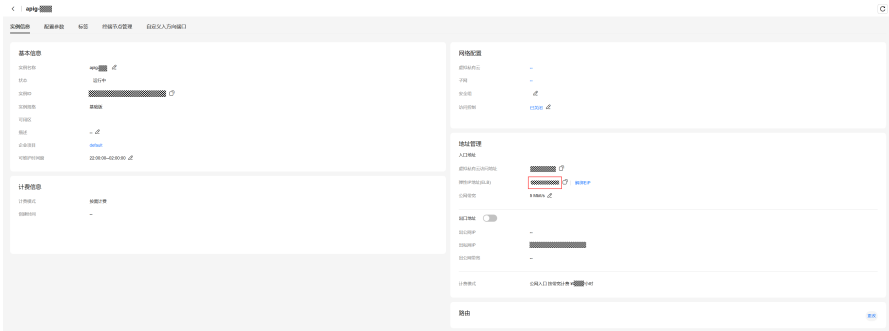
3. 获取KooSearch服务地址：可以参考[KooSearch接口获取](#)APIG公网地址，也可以参考以下步骤：
  - a. 登录KooSearch绑定的APIG服务，在“apig接口 > 实例管理”中单击实例名称。

图 9-24 实例名称



- b. 在“实例信息”页签中找到并复制“弹性IP地址”，该弹性IP地址就是KooSearch的服务地址。

图 9-25 弹性 IP 地址



4. 获取项目ID请参考[获取KooSearch服务的项目ID](#)。

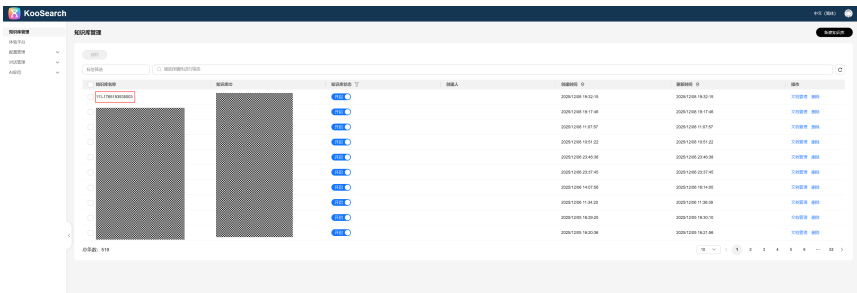
5. 获取应用ID请参考[获取KooSearch服务的应用ID](#)。

6. 获取APIG凭证请参考[获取KooSearch服务的APIG凭证](#)的APP认证部分。

7. 获取KooSearch知识库详情页面链接。

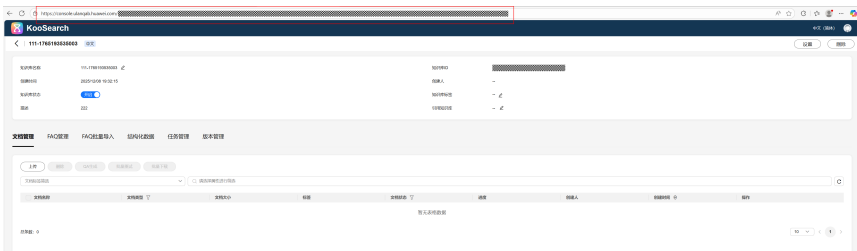
a. 登录KooSearch主页，单击需要接入的知识库，进入KooSearch知识库详情页面。

图 9-26 KooSearch 知识库列表



- b. 复制浏览器地址栏中的链接，该链接就是KooSearch知识库详情页面的链接。

图 9-27 知识库详情页面链接



步骤二：连接 KooSearch 第三方知识库

- 步骤1 登录[AgentArts智能体平台](#)。

步骤2 在左侧导航栏中选择“托管与运行 > 知识库”。

步骤3 在“知识库”页面，单击“第三方知识库连接”。

步骤4 在“第三方知识库连接”界面，单击“新建知识库连接”。

**步骤5** 在“创建知识库连接”页面，设置连接KooSearch知识库基本信息，参数配置请参考[表 9-12](#)。

图 9-28 连接外部知识库

创建知识库连接

基本信息

选择类型 ?

KooSearch

用于对接华为云企业搜索服务：KooSearch

RAGFlow

用于对接开源项目RAGFlow

General

用于对接用户自开发知识库平台

知识库名称

KooSearch

描述（可选）

KooSearch知识库

图标

支持上传png、jpeg、gif、jpg的图片，大小不能超过200 KB

连接信息

服务地址

https://

APIG接口的地址

项目id

用户登录账号的项目id，可以在个人信息中“我的凭证”处查看

应用id

KooSearch知识库平台的应用id，在KooSearch平台查看

APIG凭证 ?

KooSearch平台接口绑定的APIG凭证的AppCode，用于认证鉴权

知识库详情页面链接

https://

知识库详情页面链接，如https://xxx?id={id}。注意使用占位符{id}表示知识库ID，否则无法跳转到对应的知识库页面

测试连接

取消

确定

表 9-12 参数说明

| 参数   |        | 说明                                                                                                                                                                                                 | 示例                    |
|------|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------|
| 基本信息 | 选择类型   | <ul style="list-style-type: none"><li>General：用于对接用户自研知识库平台，需自行实现检索接口。</li><li>KooSearch：用于对接华为云企业搜索服务，开箱即用。</li><li>RAGFlow：用于对接开源项目RAGFlow，快速对接复用。</li></ul>                                     | KooSearch             |
|      | 知识库名称  | 用于标识知识库。<br>由1~50个字符组成，包含中文、字母、数字、中划线、下划线，且必须以中文、字母、数字开头。                                                                                                                                          | KooSearch             |
|      | 描述（可选） | 用于简要说明知识库内容和用途。它提供了关于知识库的详细信息，帮助用户了解知识库的内容和使用场景。<br>由0~255个字符组成。                                                                                                                                   | KooSearch知识库          |
|      | 图标     | 系统默认知识库图标，可单击  自动生成图标，也可以自定义图标。<br>1. 鼠标移动至图标上，单击鼠标左键。<br>2. 上传已准备好的知识库图标。<br>支持jpg、jpeg、png、gif格式图片，且不大于200KB。 | 默认                    |
| 连接信息 | 服务地址   | 能够访问检索接口及查询列表接口的地址，以https://或http://开头。                                                                                                                                                            | https://xxx.com       |
|      | 项目id   | 登录KooSearch知识库平台账号的项目ID。                                                                                                                                                                           | 0503dda8970xxx<br>xxx |
|      | 应用id   | KooSearch知识库平台的应用ID。                                                                                                                                                                               | 9ae90c5e53xxxxx<br>x  |

| 参数 |           | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | 示例                               |
|----|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------|
|    | APIG凭证    | <p>KooSearch平台接口绑定的APIG凭证的AppCode，用于认证鉴权。</p> <p><b>注意</b><br/>请妥善保管APIG凭证，避免泄露。建议定期更换凭证以确保安全。</p> <p><b>关于密钥加密：</b>鼠标移动至“APIG凭证”右侧的上，可查看密钥的加密方式。平台默认使用<b>KMS加密 + 默认密钥</b>方式对认证密钥进行加密存储，防止信息泄露。</p> <ul style="list-style-type: none"><li>加密机制：使用用户选择的密钥派生DEK进行加密，默认密钥为kms-agentarts/default。</li><li>费用说明：实例不收费，每月提供20,000次免费调用额度。解密时支持缓存，但敏感数据较多时可能超出免费次数。如需更换加密方式，具体请参考<a href="#">更换密钥的加密方式</a>。超出后按实际调用次数计费，详见<a href="#">密码安全中心计费说明</a>。</li></ul> | b099xxx<br>xxxxx                 |
|    | 知识库详情页面链接 | <p>第三方KooSearch知识库详情页面的链接，可通过该页面直接访问KooSearch知识库的详情页面。</p> <p>注意需要使用占位符{{id}}表示知识库ID，否则无法跳转到对应的知识库页面。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                 | https://xxxxx.com/xxxx?id={{id}} |

- 步骤6** 单击“测试连接”，弹出“测试成功”提示。

如果显示“第三方知识库连接失败，请检查连接地址和认证信息”，请检查KooSearch服务是否支持在公网使用API访问。
- 步骤7** 测试成功后，单击“确定”。

页面返回第三方知识库连接列表，连接创建成功后“状态”为“已停用”，完成[步骤三：创建KooSearch第三方知识库](#)后“状态”变为“已启用”。

----结束

步骤三：创建 KooSearch 第三方知识库

在完成KooSearch知识库连接后，您可以将KooSearch中的知识库接入到AgentArts中进行使用。

已完成[步骤二：连接KooSearch第三方知识库](#)。

- 步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。

**步骤3** 在“知识库”页面，单击“创建知识库 > 接入第三方知识库”。

  - 创建平台知识库：适用于数据在AgentArts平台直接管理的场景，支持上传文档和FAQ。具体操作请参考[创建平台知识库](#)。



- 接入第三方知识库：适用于企业已有现成知识库（如RAGFlow），希望直接对接复用的场景。

**步骤4** 在“接入第三方知识库”界面中，选择已接入的知识库，在右侧勾选知识库。

 **说明**

如果列表中没有需要的第三方知识库，则可以单击“新建”，新建知识库连接，具体配置请参考表9-12。

**图 9-29** 接入第三方知识库



**步骤5** 单击“确定”，第三方知识库创建完成。

创建完成的知识库，显示在知识库列表，“类型”为“KooSearch”，“状态”为“已启用”。

----结束

相关操作

在“第三方知识库连接”页面，支持的其他操作请参考[表9-13](#)。

表 9-13 相关操作

| 操作           | 说明                                                                         |
|--------------|----------------------------------------------------------------------------|
| 编辑第三方知识库连接信息 | 在对应的“操作”列下，单击“编辑”，可以编辑外部知识库连接信息。<br>如遇连接失败，请参考 <a href="#">各个章节</a> 的常见问题。 |
| 删除第三方知识库连接信息 | <b>注意</b><br>需要先取消接入后，才可以删除。<br>在对应的“操作”列下，单击“删除”，可以删除外部知识库连接信息。           |

常见问题

- [创建第三方知识库时为什么查不到知识库列表？](#)
- [外部知识库连接中的已启用/已停用状态如何更改？](#)
- [连接KooSearch失败如何排查？](#)
  - 接口返回401/403：检查APIG凭证是否正确。
  - 连接超时：检查URL是否正确。

相关文档

- 连接外部知识库后，请参考[知识库命中测试](#)校验知识库是否可用。
- 连接外部知识库后，可以对外部知识库进行管理，具体请参考[管理知识库](#)。

9.5.4 连接 RAGFlow 知识库

AgentArts支持连接外部知识库，通过连接外部知识库，可以显著地扩展内部知识库的知识范围，引入更多领域和更广泛的信息资源，从而提高知识库的全面性和深度。

RAGFlow适用于已自建开源RAGFlow知识库系统的场景。

本文将介绍如何在AgentArts中对接第三方RAGFlow知识库。

图 9-30 连接 RAGFlow 知识库流程



**工作原理：**连接成功后，AgentArts智能体在检索知识时，会将检索请求转发至RAGFlow服务，RAGFlow返回检索结果后由智能体进行后续处理。

**典型场景：**某企业已使用RAGFlow构建了包含产品手册、技术文档的知识库，现需在AgentArts中开发智能客服智能体，可直接连接现有RAGFlow知识库，无需重复上传文档。

费用说明

调用外部RAGFlow知识库接口会产生RAGFlow服务侧的计算资源消耗及网络流量费用，请关注第三方服务状态。

约束与限制

 警告

删除外部知识库连接前，请确认无智能体正在使用该连接下的知识库，否则将导致智能体问答失败。

当资源数量达到上限时，请删除不再使用的资源以释放配额：

- **第三方知识库连接：**进入“第三方知识库连接”页面进行删除。具体操作请参考[管理知识库](#)中的“删除外部知识库连接信息”。删除后，基于该连接接入的知识库将不可用。
- **第三方知识库：**进入“知识库”页面进行删除。具体操作请参考[管理知识库](#)中的“取消接入”。

关于知识库的更多约束与限制，请参考[知识库使用限制](#)。

步骤一：准备工作

获取第三方RAGFlow知识库的连接信息：

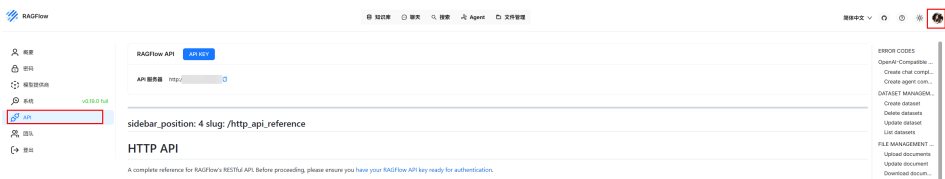
1. 在RAGFlow里创建一个知识库，并上传相关文档。

图 9-31 RAGFlow 中的知识库



2. 在RAGFlow“个人中心 > API”中获取RAGFlow的连接信息。

图 9-32 RAGFlow 连接信息



3. 在知识库列表中选中第三方知识库，单击 复制知识库地址，在新浏览器窗口的地址栏中粘贴该链接并访问，即可直接跳转到第三方知识库的详情页。

图 9-33 复制知识库地址

| 知识库                                                                                                       |              |                    |    |                                |          |                  |                       |                     |                                                            |
|-----------------------------------------------------------------------------------------------------------|--------------|--------------------|----|--------------------------------|----------|------------------|-----------------------|---------------------|------------------------------------------------------------|
| 知识库                                                                                                       |              | 外部知识库连接            |    |                                |          |                  |                       |                     |                                                            |
| <div><div>新建知识库</div><div><div>全部数据</div><div><input type="text" value="请输入名称、来源进行搜索"/></div></div></div> |              |                    |    |                                |          |                  |                       |                     |                                                            |
| 图标                                                                                                        | 知识库名称        | 类型                 | 来源 | 知识库ID                          | 描述       | 状态               | 更新人                   | 更新时间                | 操作                                                         |
|                          | D-Engine引擎   | <a href="#">默认</a> | -- | 86e92f95ec5467928c64e246a3d4   | 引擎描述     | <span>已启用</span> | op_svc_modularversaff | 2025-09-11 18:24:23 | <a href="#">详情</a> <a href="#">申请测试</a> <a href="#">更多</a> |
|                          | D-Engine引擎详情 | <a href="#">默认</a> | -- | 36740b656549595931078d2ee63    | 针对引擎相关内容 | <span>已启用</span> | op_svc_modularversaff | 2025-09-11 18:09:56 | <a href="#">详情</a> <a href="#">申请测试</a> <a href="#">更多</a> |
|                          | 测试引擎         | <a href="#">默认</a> | -- | fe73a71a8b341c0959b6d307087c25 | 测试引擎     | <span>已启用</span> | op_svc_modularversaff | 2025-09-08 17:48:22 | <a href="#">详情</a> <a href="#">申请测试</a> <a href="#">更多</a> |
|                          | yyd908       | <a href="#">默认</a> | -- | 30eebf6d42e691591552153bb6d5   | 1        | <span>已启用</span> | op_svc_modularversaff | 2025-09-08 14:07:04 | <a href="#">详情</a> <a href="#">申请测试</a> <a href="#">更多</a> |
|                          | 测试引擎         | <a href="#">默认</a> | -- | 27aa3c6b376cc80b359df6dc29795  | 测试引擎     | <span>已启用</span> | op_svc_modularversaff | 2025-09-08 14:05:33 | <a href="#">详情</a> <a href="#">申请测试</a> <a href="#">更多</a> |
|                          | 人才库          | <a href="#">默认</a> | -- | 08d5464241177a32c0456a0120a0   | --       | <span>已启用</span> | op_svc_modularversaff | 2025-09-06 19:01:47 | <a href="#">详情</a> <a href="#">申请测试</a> <a href="#">更多</a> |

图 9-34 RAGFlow 知识库详情页

## 步骤二：连接 RAGFlow 第三方知识库

### 步骤1 登录AgentArts智能体平台。

**步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。

**步骤3** 在“知识库”页面，单击“第三方知识库连接”。

**步骤4** 在“创建知识库连接”页面，设置连接RAGFlow知识库基本信息，参数配置请参考[表9-14](#)。

图 9-35 连接外部知识库

创建知识库连接

基本信息

选择类型 ?

KooSearch

用于对接华为云企业搜索服务：KooSearch

RAGFlow

用于对接开源项目RAGFlow

General

用于对接用户自开发知识库平台

知识库名称

RAGFlow

描述（可选）

RAGFlow知识库

图标





支持上传png、jpeg、gif、jpg的图片，大小不能超过200 KB

连接信息

服务地址

https://

RAGFlow服务地址

APIKey ?

RAGFlow的api访问密钥

知识库详情页面链接

http://

知识库详情页面链接，如https://xxxx?id={{id}}。注意使用占位符{{id}}表示知识库ID，否则无法跳转到对应的知识库页面

测试连接

取消

确定

表 9-14 参数说明

| 参数   |        | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | 示例              |
|------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| 基本信息 | 选择类型   | <ul style="list-style-type: none"><li>General：用于对接用户自研知识库平台，需自行实现检索接口。</li><li>KooSearch：用于对接华为云企业搜索服务，开箱即用。</li><li>RAGFlow：用于对接开源项目 RAGFlow，快速对接复用。</li></ul>                                                                                                                                                                                                                                                                                                       | RAGFlow         |
|      | 知识库名称  | 用于标识知识库。<br>由1~50个字符组成，包含中文、字母、数字、中划线、下划线，且必须以中文、字母、数字开头。                                                                                                                                                                                                                                                                                                                                                                                                             | RAGFlow         |
|      | 描述（可选） | 用于简要说明知识库内容和用途。它提供了关于知识库的详细信息，帮助用户了解知识库的内容和使用场景。<br>由0~255个字符组成。                                                                                                                                                                                                                                                                                                                                                                                                      | RAGFlow 知识库     |
|      | 图标     | 系统默认知识库图标，可单击  自动生成图标，也可以自定义图标。<br>1. 鼠标移动至图标上，单击鼠标左键。<br>2. 上传已准备好的知识库图标。<br>支持jpg、jpeg、png、gif格式图片，且不大于200KB。                                                                                                                                                                                                                                                                    | 默认              |
| 连接信息 | 服务地址   | 能够访问检索接口及查询列表接口的地址。<br>需填写RAGFlow服务的API基础地址，如 https://ragflow.example.com。                                                                                                                                                                                                                                                                                                                                                                                            | https://xxx.com |
|      | APIKey | 用于访问第三方RAGFlow知识库的鉴权密钥。<br><b>注意</b><br>请妥善保管鉴权密钥，避免泄露。建议定期更换密钥以确保安全。<br><b>关于密钥加密：</b> 鼠标移动至“APIKey”右侧的  上，可查看密钥的加密方式。平台默认使用 <b>KMS加密+默认密钥</b> 方式对认证密钥进行加密存储，防止信息泄露。 <ul style="list-style-type: none"><li>加密机制：使用用户选择的密钥派生DEK进行加密，默认密钥为kms-agentarts/default。</li><li>费用说明：实例不收费，每月提供20,000次免费调用额度。解密时支持缓存，但敏感数据较多时可能超出免费次数。超出后按实际调用次数计费，详见<a href="#">密码安全中心计费说明</a>。</li></ul> | sk-xxxxxxxx     |

| 参数 |               | 说明                                                                                     | 示例                                           |
|----|---------------|----------------------------------------------------------------------------------------|----------------------------------------------|
|    | 知识库详情<br>页面链接 | 第三方RAGFlow知识库详情页面的链接，可通过该页面直接访问RAGFlow知识库的详情页面。注意需要使用占位符{{id}}表示知识库ID，否则无法跳转到对应的知识库页面。 | http://xxxxx.com/knowledge/dataset?id={{id}} |

**步骤5** 单击“测试连接”，弹出“测试成功”提示。

如果显示“第三方知识库连接失败，请检查连接地址和认证信息”，请检查RAGFlow服务是否支持在公网使用API访问。

**步骤6** 测试成功后，单击“确定”。

页面返回第三方知识库连接列表，连接创建成功后“状态”为“已停用”，完成**步骤三：创建RAGFlow第三方知识库**后“状态”变为“已启用”。

----结束

### 步骤三：创建 RAGFlow 第三方知识库

在完成RAGFlow知识库连接后，您可以将RAGFlow中的知识库接入到AgentArts中进行使用。

已完成**步骤二：连接RAGFlow第三方知识库**。

**步骤1** 登录**AgentArts智能体平台**。

**步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。

**步骤3** 在“知识库”页面，单击“创建知识库 > 接入第三方知识库”。

- 创建平台知识库：适用于数据在AgentArts平台直接管理的场景，支持上传文档和FAQ。具体操作请参考**创建平台知识库**。
- 接入第三方知识库：适用于企业已有现成知识库（如RAGFlow），希望直接对接复用的场景。

**步骤4** 在“接入第三方知识库”界面中，选择已接入的知识库，在右侧勾选知识库。

#### 说明

如果列表中没有需要的第三方知识库，则可以单击“新建”，新建知识库连接，具体配置请参考**表9-14**。

图 9-36 接入第三方知识库



**步骤5** 单击“确定”，第三方知识库创建完成。

创建完成的知识库，显示在知识库列表，“类型”为“RAGFlow”，“状态”为“已启用”。

----结束

相关操作

在“第三方知识库连接”页面，支持的其他操作请参考[表9-15](#)。



表 9-15 相关操作

| 操作           | 说明                                                                         |
|--------------|----------------------------------------------------------------------------|
| 编辑第三方知识库连接信息 | 在对应的“操作”列下，单击“编辑”，可以编辑外部知识库连接信息。<br>如遇连接失败，请参考 <a href="#">各个章节</a> 的常见问题。 |
| 删除第三方知识库连接信息 | <b>注意</b><br>需要先取消接入后，才可以删除。<br>在对应的“操作”列下，单击“删除”，可以删除外部知识库连接信息。           |

常见问题

- [创建第三方知识库时为什么查不到知识库列表？](#)
- [外部知识库连接中的已启用/已停用状态如何更改？](#)
- [连接RAGFlow失败如何排查？](#)
  - 接口返回401/403：检查APIKey是否正确。
  - 连接超时：检查URL是否正确。

相关文档

- 连接外部知识库后，请参考[知识库命中测试](#)校验知识库是否可用。
- 连接外部知识库后，可以对外部知识库进行管理，具体请参考[管理知识库](#)。

9.5.5 知识库命中测试

AgentArts通过对创建的知识库进行命中率测试，以评估知识库的效果和准确性。  
本文将详细介绍第三方知识库创建完成后如何执行命中测试操作。

工作原理

命中测试通过语义检索技术，将用户输入的问题与知识库中的文档片段进行语义相似度计算，返回相似度最高的多条结果。

典型场景

- 验证知识库是否正确导入：输入已知存在的文档内容关键词，检查是否能召回。
- 评估检索质量：输入用户实际提问，观察召回结果的相关性。

费用说明

第三方知识库检索召回，会产生资源消耗，请关注第三方资源使用情况。

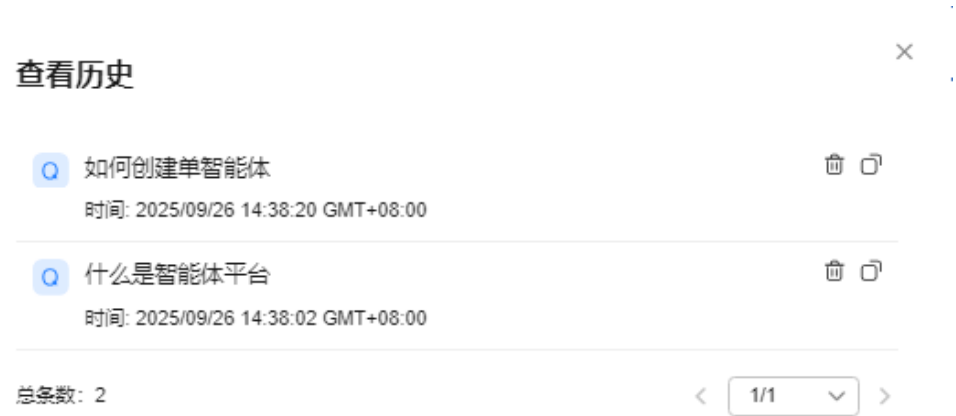
前提条件

已完成[连接General知识库](#)、[连接KooSearch知识库](#)、[连接RAGFlow知识库](#)。

## 命中测试

- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 在左侧导航栏中选择“托管与运行 > 知识库”。
- 步骤3** 在“知识库”界面，单击需要测试的知识库操作列的“命中测试”；或当知识库以卡片形式展示时，将鼠标悬浮在需要命中测试的知识库卡片上，单击“命中测试”。
- 步骤4** 在命中测试页面左侧文本框中输入问题，单击“命中测试”。
- 在命中测试页面右侧进行语义检索，展示多条匹配的内容，并按照匹配分值降序排列。
- 步骤5** 用户可以根据分值与匹配到的信息数量来评估当前知识库是否满足需求。
- 匹配分值范围为0~1，分值越接近1表示相关性越高。
- 如果测试结果为空或分值过低，建议检查第三方知识库服务是否正常运行，或尝试更换提问关键词。
- 步骤6** 单击右上角的“查看历史”，可以查看用户输入的历史问题。在历史记录右侧，单击可删除或复制该记录。

图 9-37 查看历史



----结束

## 常见问题

- **如何优化知识库检索效果？**  
优化文档结构：将无结构的文档转化为有层次结构的文档。
- **为什么检索不到已上传的文档？**
  - 可能原因：阈值设置太高；检索词语不合适。
  - 解决措施：调整阈值大小；更换提问关键词。

## 9.5.6 第三方通用知识库接入规范

### 前言

当您拥有企业内部已有的知识库系统（如内部Wiki、私有文档库），且不希望将数据迁移至AgentArts平台时，可遵循本规范开发适配接口。

### 典型场景

企业智能体需要检索内部保密文档，通过实现本规范定义的API接口，AgentArts可直接调用您的知识库服务进行问答增强。

### 工作原理

您需开发并部署两个标准HTTP接口，AgentArts智能体在对话过程中，当需要检索外部知识时，将依次调用您的接口：

1. 调用[获取知识库列表](#)接口，获取可用的知识库列表。
2. 根据用户问题，调用[检索知识库](#)接口，获取相关文档切片。
3. 将检索结果注入提示词，增强问答能力。

### 费用说明

通过AgentArts调用第三方知识库接口，可能会产生公网流量费用，具体取决于您的第三方服务部署环境。该调用过程将消耗智能体的运行时长资源。

### API 接口规范

对于API接口适配，开发者需要新增两个接口，一个是[获取知识库列表](#)，一个是[检索知识库](#)。开发者按照[接口：获取知识库列表](#)、[接口：检索知识库](#)规范提供对外接口，并在接口内实现密钥鉴权、数据查询、知识检索等业务逻辑，最后按照规范返回对应数据。

#### 接口：获取知识库列表

- URI  
GET /knowledge-bases
- 查询Query参数

表 9-16 查询 Query 参数

| 参数   | 是否必选 | 参数类型   | 描述                                                                                              |
|------|------|--------|-------------------------------------------------------------------------------------------------|
| name | 否    | String | <b>参数解释：</b><br>知识库名称，用于获取知识库列表时筛选知识库名称。<br><b>取值范围：</b><br>长度不超过64个字符。<br><b>默认取值：</b><br>不涉及。 |

| 参数     | 是否必选 | 参数类型    | 描述                                                                                                                   |
|--------|------|---------|----------------------------------------------------------------------------------------------------------------------|
| offset | 否    | Integer | <b>参数解释：</b><br>当前请求的分页偏移量，表示从第几条开始获取数据。默认值为0，表示从第0条开始获取数据。<br><b>取值范围：</b><br>0～65535。<br><b>默认取值：</b><br>0。        |
| limit  | 否    | Integer | <b>参数解释：</b><br>当前请求返回的分页最大数据条数，表示一次请求返回的数据条数。默认值为10，表示一页最多展示10条数据。<br><b>取值范围：</b><br>1~100。<br><b>默认取值：</b><br>10。 |

- 请求Header参数

表 9-17 请求 Header 参数

| 参数            | 是否必选 | 参数类型   | 描述                                                                                                                                       |
|---------------|------|--------|------------------------------------------------------------------------------------------------------------------------------------------|
| Authorization | 是    | String | <b>参数解释：</b><br>认证鉴权使用的api-key密钥，拼接“Bearer ”后的字符串。<br>示例：Bearer d59*****9C3，用于第三方知识库的鉴权。<br><b>取值范围：</b><br>不涉及。<br><b>默认取值：</b><br>不涉及。 |

- 响应参数

表 9-18 响应参数 200

| 参数                  | 参数类型           | 描述                                                                                                          |
|---------------------|----------------|-------------------------------------------------------------------------------------------------------------|
| total               | Integer        | <b>参数解释：</b><br>知识库数据总数。<br><b>取值范围：</b><br>不涉及。                                                            |
| knowledge_base_list | Array of 表9-19 | <b>参数解释：</b><br>知识库数据信息列表。<br><b>注意</b><br>第三方知识库只返回可用的知识库，即在当前时刻可被外部引用并支持检索的知识库数据。<br><b>取值范围：</b><br>不涉及。 |

表 9-19 KnowledgeBaseInfo

| 参数                | 参数类型   | 描述                                             |
|-------------------|--------|------------------------------------------------|
| knowledge_base_id | String | <b>参数解释：</b><br>知识库ID。<br><b>取值范围：</b><br>不涉及。 |
| name              | String | <b>参数解释：</b><br>知识库名称。<br><b>取值范围：</b><br>不涉及。 |
| description       | String | <b>参数解释：</b><br>知识库描述。<br><b>取值范围：</b><br>不涉及。 |

表 9-20 响应参数 400、500 等

| 参数         | 参数类型   | 描述                                           |
|------------|--------|----------------------------------------------|
| error_code | String | <b>参数解释：</b><br>错误码。<br><b>取值范围：</b><br>不涉及。 |

| 参数        | 参数类型   | 描述                                            |
|-----------|--------|-----------------------------------------------|
| error_msg | String | <b>参数解释：</b><br>错误描述。<br><b>取值范围：</b><br>不涉及。 |

- 请求示例  
GET http://10.0.0.0:8090/knowledge-bases?name=123&offset=0&limit=10
- 响应示例

```
{
 "total": 1,
 "knowledge_base_list": [
 {
 "knowledge_base_id": "001",
 "name": "一个通用第三方知识库的名称 / this is a knowledgeBase name",
 "description": "一个描述 / a description"
 }
]
}
```

接口：检索知识库

- URI  
POST /knowledge-bases/retrieve
- 请求Header参数

表 9-21 请求 Header 参数

| 参数            | 是否必选 | 参数类型   | 描述                                                                                                                                       |
|---------------|------|--------|------------------------------------------------------------------------------------------------------------------------------------------|
| Authorization | 是    | String | <b>参数解释：</b><br>认证鉴权使用的api-key密钥，拼接“Bearer ”后的字符串。<br>示例：Bearer d59*****9C3，用于第三方知识库的鉴权。<br><b>取值范围：</b><br>不涉及。<br><b>默认取值：</b><br>不涉及。 |

- Body参数

表 9-22 Body 参数

| 参数                 | 是否必选 | 参数类型            | 描述                                                                                                              |
|--------------------|------|-----------------|-----------------------------------------------------------------------------------------------------------------|
| knowledge_base_ids | 是    | Array of String | <b>参数解释：</b><br>知识库ID的列表，用于指定搜索时需要检索的知识库。<br><b>取值范围：</b><br>列表长度大于等于1，即至少指定检索一个知识库。<br><b>默认取值：</b><br>不涉及。    |
| query              | 是    | String          | <b>参数解释：</b><br>用户的搜索内容，即需要在知识库中检索的问题或关键信息。<br><b>取值范围：</b><br>长度不超过4096个字符。<br><b>默认取值：</b><br>不涉及。            |
| method             | 是    | String          | <b>参数解释：</b><br>搜索方式。语义检索：“doc”。<br><b>取值范围：</b><br>不涉及。<br><b>默认取值：</b><br>不涉及。                                |
| offset             | 否    | Integer         | <b>参数解释：</b><br>当前请求的分页偏移量，表示从第几条开始获取数据。默认值为0，表示从第0条开始获取数据。<br><b>取值范围：</b><br>0 ~ 65535。<br><b>默认取值：</b><br>0。 |

| 参数                   | 是否必选 | 参数类型               | 描述                                                                                                                                                                                                                                           |
|----------------------|------|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| limit                | 否    | Integer            | <b>参数解释：</b><br>当前请求返回的分页最大数据数量，表示一次请求返回的数据条数。默认值为10，表示一页最多展示10条数据。<br><b>取值范围：</b><br>1~100。<br><b>默认取值：</b><br>10。                                                                                                                         |
| top_k                | 否    | Integer            | <b>参数解释：</b><br>检索返回结果的最大召回数量。<br><b>取值范围：</b><br>1-1024。<br><b>默认取值：</b><br>50。                                                                                                                                                             |
| search_thres<br>hold | 否    | Number             | <b>参数解释：</b><br>检索时的相关度得分阈值，低于该阈值的结果将被过滤。默认值0.0时表示不设置相关度得分阈值。<br><b>取值范围：</b><br>0.0 - 1.0。<br><b>默认取值：</b><br>0.0。                                                                                                                          |
| extra_params         | 否    | Array of 表<br>9-23 | <b>参数解释：</b><br>额外的自定义参数，用于传递第三方知识库所需的其他业务参数。<br><b>示例：</b> <pre>"extra_params": [   { "key": "tenant_id", "value": "enterprise-001" },   { "key": "data_region", "value": "cn-north" } ]</pre> <b>取值范围：</b><br>不涉及。<br><b>默认取值：</b><br>不涉及。 |



表 9-23 KnowledgeBaseExtraParam

| 参数    | 是否必选 | 参数类型   | 描述                                               |
|-------|------|--------|--------------------------------------------------|
| key   | 是    | String | <b>参数解释：</b><br>额外参数主键名。<br><b>取值范围：</b><br>不涉及。 |
| value | 是    | String | <b>参数解释：</b><br>额外参数的值。<br><b>取值范围：</b><br>不涉及。  |

- 响应参数

表 9-24 响应参数 200

| 参数                 | 参数类型                           | 描述                                                 |
|--------------------|--------------------------------|----------------------------------------------------|
| total              | Integer                        | <b>参数解释：</b><br>检索结果总数。<br><b>取值范围：</b><br>不涉及。    |
| search_result_list | Array of <a href="#">表9-25</a> | <b>参数解释：</b><br>检索结果的信息列表。<br><b>取值范围：</b><br>不涉及。 |

表 9-25 SearchChunkInfo

| 参数                | 参数类型   | 描述                                                   |
|-------------------|--------|------------------------------------------------------|
| knowledge_base_id | String | <b>参数解释：</b><br>该切片所属的知识库ID。<br><b>取值范围：</b><br>不涉及。 |
| file_id           | String | <b>参数解释：</b><br>该切片所属的文件ID。<br><b>取值范围：</b><br>不涉及。  |

| 参数       | 参数类型   | 描述                                                                               |
|----------|--------|----------------------------------------------------------------------------------|
| chunk_id | String | <b>参数解释：</b><br>切片的唯一标识ID。<br><b>取值范围：</b><br>不涉及。                               |
| content  | String | <b>参数解释：</b><br>检索返回的文本内容，即实际注入提示词的知识片段。<br><b>取值范围：</b><br>不涉及。                 |
| score    | Number | <b>参数解释：</b><br>该切片与检索query的相关度得分，将根据此得分决定是否使用该切片。<br><b>取值范围：</b><br>0.0 - 1.0。 |
| title    | String | <b>参数解释：</b><br>文本标题，推荐返回文件名称，用于在引用来源时展示给用户。<br><b>取值范围：</b><br>不涉及。             |

表 9-26 响应参数 400、500 等

| 参数         | 参数类型   | 描述                                            |
|------------|--------|-----------------------------------------------|
| error_code | String | <b>参数解释：</b><br>错误码。<br><b>取值范围：</b><br>不涉及。  |
| error_msg  | String | <b>参数解释：</b><br>错误描述。<br><b>取值范围：</b><br>不涉及。 |

- 请求示例  
POST `http://10.0.0.0:8090/knowledge-bases/retrieve`
- 响应示例

```
{
 "knowledge_base_ids": [
 "001"
],
 "query": "检索",
 "method": "doc",
```

```

"offset": 0,
"limit": 10,
"top_k": 50,
"search_threshold": 0.01
}
{
"total": 1,
"search_result_list": [
{
"knowledge_base_id": "001",
"file_id": "101",
"chunk_id": "201",
"content": "一个通用第三方知识库的检索切片内容 / this is a retrieve chunk content",
"score": 0.8,
"title": "检索切片的标题 / retrieve chunk title"
}
]
}

```

## API 接口实现案例（Java）

```

@Slf4j
@RestController
public class CommonThirdpartyKnowledgeBaseController {
 /**
 * list knowledgeBase function
 *
 * @param name
 * @param offset
 * @param limit
 * @param authorization
 * @return ListKnowledgeBaseResponse
 */
 @GetMapping("/knowledge-bases")
 public ListKnowledgeBaseResponse listKnowledgeBase(
 @RequestHeader("Authorization") String authorization,
 @RequestParam(required = false) String name,
 @RequestParam(required = false) Integer offset,
 @RequestParam(required = false) Integer limit) {
 /**
 * 对密钥认证鉴权
 * Verify and authenticate the key
 */
 if (!authorization.startsWith("Bearer ")) {
 log.error("invalid authorization");
 throw new RuntimeException("invalid authorization");
 }
 String apiKey = authorization.substring("Bearer ".length());
 /**
 * then do something for authorization
 */

 /**
 * 根据业务实现代码逻辑，构造返回结果
 * Construct and return results based on your business implementation and code logic
 */
 ListKnowledgeBaseResponse response = new ListKnowledgeBaseResponse();
 response.total = 1;
 KnowledgeBaseInfo knowledgeBaseInfo = new KnowledgeBaseInfo();
 knowledgeBaseInfo.knowledge_base_id = "001";
 knowledgeBaseInfo.name = "一个通用第三方知识库的名称 / this is a knowledgeBase name";
 knowledgeBaseInfo.description = "一个描述 / a description";
 response.knowledge_base_list = List.of(knowledgeBaseInfo);
 return response;
 }
}

```

```

* @param authorization
* @param request
* @return RetrieveKnowledgeBaseResponse
*/
@PostMapping("/knowledge-bases/retrieve")
public RetrieveKnowledgeBaseResponse retrieveKnowledgeBase(
 @RequestHeader("Authorization") String authorization,
 @RequestBody RetrieveKnowledgeBaseRequest request) {
 /**
 * 对密钥认证鉴权
 * Verify and authenticate the key
 */
 if (!authorization.startsWith("Bearer ")) {
 log.error("invalid authorization");
 throw new RuntimeException("invalid authorization");
 }
 String apiKey = authorization.substring("Bearer ".length());
 /**
 * then do something for authorization
 */

 /**
 * 根据业务实现代码逻辑，构造返回结果
 * Construct and return results based on your business implementation and code logic
 */
 RetrieveKnowledgeBaseResponse response = new RetrieveKnowledgeBaseResponse();
 response.total = 1;
 SearchChunkInfo searchChunkInfo = new SearchChunkInfo();
 searchChunkInfo.knowledge_base_id = "001";
 searchChunkInfo.file_id = "101";
 searchChunkInfo.chunk_id = "201";
 searchChunkInfo.content = "一个通用第三方知识库的检索切片内容 / this is a retrieve chunk content";
 searchChunkInfo.title = "检索切片的标题 / retrieve chunk title";
 searchChunkInfo.score = 0.8f;
 response.search_result_list = List.of(searchChunkInfo);
 return response;
}

/*
 * the response structure of the listKnowledgeBase function
 */
@Data
public class ListKnowledgeBaseResponse {
 Integer total;
 List<KnowledgeBaseInfo> knowledge_base_list;
}

@Data
public class KnowledgeBaseInfo {
 String knowledge_base_id;
 String name;
 String description;
}

/*
 * the request structure of the retrieveKnowledgeBase function
 */
@Data
public static class RetrieveKnowledgeBaseRequest {
 List<String> knowledge_base_ids;
 String query;
 String method;
 Integer offset;
 Integer limit;
 Integer top_k;
 Float search_threshold;
 List<KnowledgeBaseExtraParam> extra_params;
}

```

```
}

@Data
public static class KnowledgeBaseExtraParam {
 String key;
 String value;
}

/*
 * the response structure of the retrieveKnowledgeBase function
 */
@Data
public class RetrieveKnowledgeBaseResponse {
 Integer total;
 List<SearchChunkInfo> search_result_list;
}

@Data
public class SearchChunkInfo {
 String knowledge_base_id;
 String file_id;
 String chunk_id;
 String content;
 Float score;
 String title;
}
}
```

## 常见问题

如果接口调用失败，请检查鉴权头是否正确配置。常见报错排查请参见[第三方接入故障排除](#)。

## 9.6 在低码智能体中使用知识库

### 9.6.1 在单智能体中使用知识库

支持在AgentArts中添加引用知识库，以根据用户意图来检索召回对应的知识切片。

通过关联知识库，智能体可在回答用户问题前，先从私有知识库中检索相关信息，从而基于私有数据生成准确回答，有效减少模型幻觉。

**典型使用场景：**

- 企业内部知识问答：员工可通过智能体查询公司制度、流程规范等。
- 产品文档查询：基于产品手册回答用户技术问题。
- 私有数据问答：基于企业专属数据生成回答，避免通用模型的局限性。

## 费用说明

使用知识库进行对话时，检索召回的内容将作为上下文输入大模型，这会增加Token消耗，请关注用量成本。检索召回的切片数量（topk设置）直接影响Token消耗，建议根据实际需求调整。

## 前提条件

- 如果需要在单智能体中使用本地知识库，请确保已[创建平台知识库](#)且知识库是启用状态。

- 如果需要在单智能体中使用第三方知识库，请确保已[接入第三方知识库](#)且知识库是启用状态。

## 配置知识库

**步骤1** 登录[AgentArts智能体平台](#)。

**步骤2** 单击左侧导航栏“开发中心 > 智能体管理”，单击左上角“单智能体”页签，进入单智能体应用管理界面。

**步骤3** 单击目标单智能体应用，在“单智能体配置”页签中，在“知识库”模块，单击 $+$ 。

**步骤4** 在“添加知识库”窗口，选择知识库类型，勾选目标知识库右侧的☐进行添加，单击“确定”完成知识库添加。

图 9-38 添加知识库



**步骤5** 单击“知识库”模块的 $\equiv$ 按钮，弹出配置弹窗。

图 9-39 高级配置

检索策略

**语义检索** 使用向量检索技术检索知识文档

**关键词检索** 使用倒排检索技术检索知识文档

**混合检索** 使用向量检索和关键词检索混合检索知识文档

启用FAQ

☒

FAQ直出阈值

00.51

0.900

相关度阈值

00.51

0.500

topk召回数量

12550

3

步骤6 参见表9-27设置参数。

表 9-27 参数说明

| 参数   | 说明                                                                                                                                                                                                                                                                                                      | 示例   |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|
| 检索策略 | 检索策略，文档检索的方式，有三种： <ul style="list-style-type: none"><li>语义检索，使用向量检索技术检索，对文档及结构化数据中知识进行检索，召回与用户意图相关性高的切片内容，推荐在需要结合上下文相关性、并对用户意图进行理解场景中使用。</li><li>关键词检索，使用倒排检索技术，对文档及结构化数据中知识进行检索，召回与Query关键词匹配度高的切片内容，推荐在需要用户提问关键词匹配度高的场景中使用。</li><li>混合检索，使用向量检索和关键词检索两种策略混合检索知识库，推荐在需要兼顾用户意图理解及关键词匹配度场景中使用。</li></ul> | 语义检索 |

| 参数       | 说明                                                                                                                                                                                             | 示例    |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|
| 相关度阈值    | 超过相关度阈值的搜索结果会提交给大模型进行总结，否则被过滤，可以参考知识库中命中测试的相关度分值调整该阈值。<br><b>取值范围：</b> 0~1。<br><b>默认值：</b> 0.5。                                                                                                | 0.500 |
| topk召回数量 | 召回相关性排名前topk的切片数量，如topk召回数量为5，则相关性排名前5的切片将被召回提交给大模型总结。<br><b>取值范围：</b> 1~50。<br><b>默认值：</b> 3。                                                                                                 | 3     |
| FAQ直出阈值  | FAQ检索超过阈值的结果将直接提交给大模型总结，不再进行文档检索。如果没有超过阈值的结果，将进行文档检索。<br><b>取值范围：</b> 0~1。<br>启用FAQ功能后，系统将优先检索FAQ数据。如果未命中结果，则会继续查询切片内容，可能会带来一定的性能开销。当FAQ检索结果超过预设阈值时，将直接提交给大模型进行总结，不再进行文档检索。如果未超过阈值，则将继续进行文档检索。 | 0.900 |
| 查看来源     | 添加知识库并开启此功能后，可以在预览调试界面中查看搜索结果的详细来源信息，包括上下文内容和文件名称。有助于更快速、准确地定位和理解搜索结果。                                                                                                                         | 开启    |
| 查看图片     | 开启此功能后，当知识库支持图片检索时，可查看检索结果中的图片信息。<br><b>说明</b><br>此功能仅在知识库包含图片数据且支持图片检索时生效。                                                                                                                    | 开启    |

**步骤7** 单击其他位置退出弹窗，完成配置。

配置完成后，在“知识库”模块中可看到已添加的知识库卡片。您可以在单智能体界面的“预览调试”区域进行对话测试，验证知识库是否生效。在预览调试区域输入与知识库内容相关的问题，观察回答是否引用了知识库信息。开启“查看来源”功能后，可查看引用的具体切片来源。

----结束

常见问题

为什么检索不到知识库内容？

- 可能原因：阈值设置太高；检索词语、策略不合适；精排模型不可用。
- 解决措施：调整阈值大小；修改检索词语、策略；在KooSearch服务中确认精排模型正常。



#### 如何提升知识库检索准确率？

- 优化文档结构：无结构的文档，转化为有层次结构的文档。
- 团队标准版、团队高级版、企业专业版或企业旗舰版支持更换接入的其他模型。
- 当前AgentArts团队标准版、团队高级版、企业专业版、企业旗舰版未开放，如果需要购买，请[提交工单](#)咨询购买相关事宜。
- 修改知识库的分段设置。

### 9.6.2 在工作流中使用知识库

支持在AgentArts中添加引用知识库，以根据用户意图来检索召回对应的知识切片。

在工作流中，“知识检索”节点作为独立的处理单元，将检索结果输出给下游的“LLM”节点，使大模型能够基于检索到的私有知识生成回答。适用于需要结合工具调用、多轮复杂逻辑的场景。

**工作原理：**知识检索节点接收用户问题，通过向量嵌入或关键词匹配在知识库中检索相关内容，按相关度排序后返回TopK个知识切片，供下游LLM节点生成回答。

#### 典型场景：

- 企业内部知识问答：员工提问“如何申请年假？”，系统检索员工手册相关条款
- 技术文档助手：开发者提问“如何配置API网关？”，系统检索技术文档对应章节。

### 费用说明

召回的知识切片将作为上下文传递给后续节点，切片数量和大小会影响大模型的Token消耗，请合理设置TopK和阈值。切片数量越多、内容越长，传递给大模型的上下文Token数越多，调用费用越高。建议根据实际需求设置TopK（推荐3-5），避免召回过多无关内容。

### 前提条件

- 如果需要在 workflow 中使用平台知识库，请确保已[创建平台知识库](#)且知识库是启用状态。
- 如果需要在 workflow 中使用第三方知识库，请确保已[接入第三方知识库](#)且知识库是启用状态。

### 配置知识库

- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 单击左侧导航栏“开发中心 > 智能体管理”，单击左上角“工作流”页签，进入工作流应用管理界面，单击您创建的工作流。
- 步骤3** 单击“添加节点”并选择“知识检索”节点。
- 步骤4** 通过单击该节点打开节点配置页面。
- 步骤5** 配置输入参数，请参见[表9-28](#)设置参数。

图 9-40 输入参数

 知识检索

 | 

根据输入内容，从指定的知识库中检索并召回最相关的知识片段，以列表形式返回。检索结果可传入大模...

---

输入参数 ⓘ ^

| 参数名称  | 类型 | 值                                |
|-------|----|----------------------------------|
| query | 输入 | <input type="text" value="请输入"/> |

---

知识库 ⓘ ^ 

  
暂无知识库，请选择

---

输出参数 ⓘ ^

 output\_list Array<>

document\_name String

subtitle String

content String

score Number

knowledge\_base\_id String

file\_id String

knowledge\_base\_type String

表 9-28 参数说明

| 参数名称 | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | 示例                      |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------|
| 输入参数 | <p>输入参数固定只有1个，参数名称为 query且不可修改，类型是字符串，表示待知识检索的问题。通常引用开始节点的值，该变量决定了检索的搜索词。</p> <p>设置参数值时，支持“引用”和“输入”两种方式。</p> <ul style="list-style-type: none"><li>引用：支持用户选择工作流中已包含的前置节点的输出变量值以及全局配置中的记忆变量，限制String类型，适用于需要从前置节点输出中获取知识检索问题的场景。</li></ul> <p>系统参数列举如下：</p> <ul style="list-style-type: none"><li>“conversation_history”：存储当前对话的历史消息记录。</li><li>“current_time”：获取当前的系统时间戳或时间字符串。</li><li>“user_id”：识别当前用户的唯一标识符。</li><li>“conversation_id”：当前工作流的唯一对话标识。</li><li>“dialogue_count”：当前对话的轮次数计数。</li></ul> <ul style="list-style-type: none"><li>输入：支持用户输入固定值，适用于知识检索问题固定的场景。</li></ul> | <p>类型：输入</p> <p>值：1</p> |

| 参数名称 | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | 示例                                                                                  |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
| 输出参数 | <p>知识检索节点的输出是一个名为 <code>output_list</code> 的对象数组，表示所有满足检索要求的知识切片。数组中对象有七个属性：</p> <ul style="list-style-type: none"><li>• <code>document_name</code>，知识切片所在的知识文档名称。</li><li>• <code>subtitle</code>，知识切片子标题。</li><li>• <code>content</code>，知识切片的内容。</li><li>• <code>score</code>，知识切片的匹配度得分，<code>output_list</code> 中的元素按照得分由高到低排序。</li><li>• <code>knowledge_base_id</code>，文件所属知识库ID。</li><li>• <code>knowledge_base_type</code>，文件所属知识库类型。</li><li>• <code>file_id</code>，文件ID。</li></ul> <p>后续节点引用该输出参数，可以引用 <code>output_list</code>，此时将获取全量的检索结果，包括文档名、切片子标题、切片内容和分数。也可以直接引用切片的属性，比如 <code>content</code>，此时将获取 <code>output_list</code> 中第一条记录的切片内容。</p> |  |

- 步骤6 在知识库区域单击  按钮，进入知识库添加页面。
- 步骤7 在“添加知识库”窗口，选择知识库类型，勾选目标知识库右侧的 ☐ 进行添加，单击“确定”。

图 9-41 添加知识库



- 步骤8** 在知识库区域单击  按钮，弹出检索参数配置页面。
- 步骤9** 配置检索参数，完成后，单击其他位置退出弹窗。

图 9-42 配置检索参数

检索策略 ?

 **语义检索** 使用向量检索技术检索知识文档

 **关键词检索** 使用倒排检索技术检索知识文档

 **混合检索** 使用向量检索和关键词检索混合检索知识文档

启用FAQ ?

☒

FAQ直出阈值 ?

00.51

0.900

相关度阈值 ?

00.51

0.500

topk召回数量 ?

12550

3

表 9-29 参数说明

| 参数名称 | 说明                                                                                                                                                                                                                                                                                                  | 示例   |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|
| 检索策略 | 文档检索的方式，有三种： <ul style="list-style-type: none"><li>语义检索：使用向量检索技术检索，对文档及结构化数据中知识进行检索，召回与用户意图相关性高的切片内容，推荐在需要结合上下文相关性、并对用户意图进行理解的场景中使用。</li><li>关键词检索：使用倒排检索技术，对文档及结构化数据中知识进行检索，召回与Query关键词匹配度高的切片内容，推荐在需要用户提问关键词匹配度高的场景中使用。</li><li>混合检索：使用向量检索和关键词检索两种策略混合检索知识库，推荐在需要兼顾用户意图理解及关键词匹配度场景中使用。</li></ul> | 语义检索 |

| 参数名称     | 说明                                                                                                                                                                                     | 示例    |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|
| 相关度阈值    | 超过相关度阈值的搜索结果会提交给大模型进行总结，否则被过滤，可以参考知识库中命中测试的相关度分值调整该阈值。<br>取值范围：0~1。<br>默认值：0.5。                                                                                                        | 0.100 |
| topk召回数量 | 召回的相关性排名前topk的切片数量，如topk召回数量为5，则相关性排名前5的切片将被召回提交给大模型总结。<br>取值范围：1~50。<br>默认值：3。                                                                                                        | 3     |
| FAQ直出阈值  | FAQ检索超过阈值的结果将直接提交给大模型总结，不再进行文档检索。如果没有超过阈值的结果，将进行文档检索。<br>取值范围：0~1。<br>启用FAQ功能后，系统将优先检索FAQ数据。如果未命中结果，则会继续查询切片内容，可能会带来一定的性能开销。当FAQ检索结果超过预设阈值时，将直接提交给大模型进行总结，不再进行文档检索。如果未超过阈值，则将继续进行文档检索。 | 0.100 |
| 查看图片     | 开启此功能后，当知识库支持图片检索时，可查看检索结果中的图片信息。                                                                                                                                                      | 开启    |

**步骤10** 在知识检索配置页面单击“确定”，完成知识检索节点配置。

配置完成后，单击工作流编辑器右上角的“试运行”或使用节点的  功能，验证知识库是否能正确召回内容。

----结束

常见问题

- **为什么知识检索节点返回结果为空？**
  - 可能原因：知识库异常、阈值设置不合理。
  - 处理措施：检查知识库：状态正常，命中测试通过；重新设置检索阈值。
- **如何在工作流中引用知识检索节点的输出？**

后续节点引用该输出参数，可以引用output\_list，此时将获取全量的检索结果，包括文档名、切片子标题、切片内容和分数。也可以直接引用切片的属性，比如content，此时将获取output\_list中第一条记录的切片内容。

图 9-43 结束节点引用知识检索节点的输出参数示例



9.7 管理知识库

知识库创建完成后，在“知识库”界面，您可以通过分类筛选或搜索功能来查找知识库。支持的其他操作请参考表9-30。

知识库支持以列表和卡片形式展示。单击搜索框右侧  按钮，知识库将以列表形式展示。单击搜索框右侧  按钮，知识库将以卡片形式展示。

表 9-30 相关操作

| 操作    | 说明                                                                                                                                                                                                                                                                 |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 查看知识库 | <ul style="list-style-type: none"><li>当知识库以列表形式展示时，单击知识库列表所需知识库名称，进入知识库详情页面。</li><li>当知识库以卡片形式展示时，单击知识库卡片，进入知识库详情页面。</li></ul>                                                                                                                                     |
| 启用知识库 | <ul style="list-style-type: none"><li>当知识库以列表形式展示时，找到“状态”是“已停用”的知识库，单击操作列“启用”，可以启用知识库。操作完成后，该知识库的“状态”将更新为“已启用”。</li><li>当知识库以卡片形式展示时，找到“状态”是“已停用”的知识库，将鼠标悬浮在需要启用的知识库卡片上，单击“启用”，可以启用知识库。操作完成后，该知识库的“状态”将更新为“已启用”。</li></ul> <p>只有“状态”是“已启用”的知识库才能应用、工作流中引用该知识库。</p> |



| 操作    | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 停用知识库 | <p><b>警告</b></p> <p>停用已经被应用、工作流引用的知识库，会导致检索结果返回空值，请谨慎操作。</p> <p>删除前，请确保该知识库未被业务引用且不再需要。可通过下方“查看引用”确认引用关系。</p> <ul style="list-style-type: none"><li>当知识库以列表形式展示时，找到“状态”是“已启用”的知识库，单击操作列“停用”，可以停用知识库。操作完成后，该知识库的“状态”将更新为“已停用”。</li><li>当知识库以卡片形式展示时，找到“状态”是“已启用”的知识库，将鼠标悬浮在需要停用的知识库卡片上，单击“停用”，可以停用知识库。操作完成后，该知识库的“状态”将更新为“已停用”。</li></ul>                                                                                                                                                                  |
| 命中测试  | <p>用于验证知识库的检索效果，检查提问能否召回相关知识点。测试完成后，将显示召回的知识点列表及相似度得分，您可以根据结果调整分段策略或检索参数。</p> <ul style="list-style-type: none"><li>当知识库以列表形式展示时，单击操作列“命中测试”，可以测试知识库命中率。</li><li>当知识库以卡片形式展示时，将鼠标悬浮在需要命中测试的知识库卡片上，单击“命中测试”，可以测试知识库命中率。</li></ul> <p><b>只有“状态”是“已启用”的知识库才能进行命中测试</b>，详细操作请参见<a href="#">知识库命中测试</a>或<a href="#">知识库命中测试</a>。</p> <p><b>注意</b></p> <p>测试过程会调用模型进行语义检索，消耗模型Token。Token消耗量与提问长度和召回结果数量相关，建议在测试前了解模型计费说明。</p>                                                                                  |
| 修改配置  | <p>修改配置可以修改“名称”、“描述（可选）”、“知识库图标”、“精排模型”、“文档解析（可选）”、“分段策略（可选）”。</p> <p>“精排模型”、“文档解析（可选）”、“分段策略（可选）”修改配置后，通常需要重新解析或拆分数据才能生效。配置修改并重新解析后，可通过<a href="#">知识库命中测试</a>或<a href="#">知识库命中测试</a>验证新配置效果，或查看知识库详情页的解析状态确认是否生效。</p> <ul style="list-style-type: none"><li>当知识库以列表形式展示时，单击操作列“更多 &gt; 修改配置”，可以编辑知识库。</li><li>当知识库以卡片形式展示时，将鼠标悬浮在需要设置的知识库卡片上，单击“修改配置”，可以编辑知识库。</li></ul> <p><b>只有“状态”是“已停用”的平台知识库才可修改配置。</b></p> <p><b>说明</b></p> <p>修改模型配置可能影响解析计费规格。不同模型规格的解析单价不同，修改为更高规格模型会增加解析成本。建议提前了解模型计费说明。</p> |

| 操作    | 说明                                                                                                                                                                                                                                                                                                                           |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 编辑知识库 | <p>编辑知识库可以修改“名称”、“描述（可选）”、“知识库图标”。编辑完成后，知识库列表/卡片将显示更新后的名称、描述和图标。</p> <ul style="list-style-type: none"><li>当知识库以列表形式展示时，单击操作列“更多 &gt; 编辑”，可以编辑知识库。</li><li>当知识库以卡片形式展示时，将鼠标悬浮在需要编辑的知识库卡片上，单击“编辑”，可以编辑知识库。</li></ul> <p><b>只有“状态”是“已停用”的平台知识库才可编辑。</b></p>                                                                    |
| 查看引用  | <p>可以查看当前知识库被哪些智能体和工作流引用。具体有以下三种查看方式：</p> <ul style="list-style-type: none"><li>当知识库以列表形式展示时，单击操作列“更多 &gt; 查看引用”，可以查看知识库被哪些智能体和工作流引用。</li><li>当知识库以卡片形式展示时，将鼠标悬浮在需要查看的知识库卡片上，单击“更多 &gt; 查看引用”，可以查看知识库被哪些智能体和工作流引用。</li><li>在知识库详情页面，单击“... &gt; 引用列表”，可以查看知识库被哪些智能体和工作流引用。</li></ul> <p>单击引用列表中的智能体或工作流名称，可跳转到具体的应用详情页面。</p> |
| 取消接入  | <ul style="list-style-type: none"><li>当知识库以列表形式展示时，单击操作列“更多 &gt; 取消接入”，可以取消接入外部知识库。</li><li>当知识库以卡片形式展示时，将鼠标悬浮在需要取消接入的知识库卡片上，单击“取消接入”，可以取消接入外部知识库。</li></ul> <p><b>只有“状态”是“已停用”的外部知识库才可取消接入。</b></p>                                                                                                                         |
| 删除知识库 | <p><b>警告</b><br/>删除知识库属于高危操作，数据将被永久删除且不可恢复。删除前，请确保该知识库未被业务引用且不再需要。</p> <ul style="list-style-type: none"><li>当知识库以列表形式展示时，单击操作列“更多 &gt; 删除”，可以删除知识库。</li><li>当知识库以卡片形式展示时，将鼠标悬浮在需要删除的知识库卡片上，单击“更多 &gt; 删除”，可以删除知识库。</li></ul> <p><b>只有“状态”是“已停用”的平台知识库才可删除。</b></p>                                                          |

10

模型

10.1 模型介绍

模型管理为用户提供自定义模型服务的统一接入与全生命周期管理能力，实现模型提供商、模型服务和模型代理的一站式配置与运维。

通过模型管理，可以将外部模型提供商（如OpenAI、Anthropic 等）的LLM推理服务接入AgentArts平台，由网关统一管理推理请求的路由、认证和负载均衡，使智能体、记忆库等组件按需调用已配置的模型服务。

核心概念

模型管理包含三个核心概念：模型提供商、模型服务和模型代理。

表 10-1 核心概念说明

| 概念    | 说明                                                                      |
|-------|-------------------------------------------------------------------------|
| 模型提供商 | 模型服务的管理实体，定义统一的访问端点（base_url）、出站身份认证和网络配置。一个模型提供商下可配置多个模型服务，也可关联多个模型代理。 |
| 模型服务  | 具体的模型配置，从属于模型提供商，包含模型名称、API 接口协议和模型类型。                                  |
| 模型代理  | 连接模型提供商与网关的桥梁，提供统一的推理服务入口。模型代理配置入站身份认证和入站网关，将推理请求路由到关联的模型提供商。           |

三者之间的关系如下：

1.

创建模型提供商，配置访问端点和出站认证。
2.

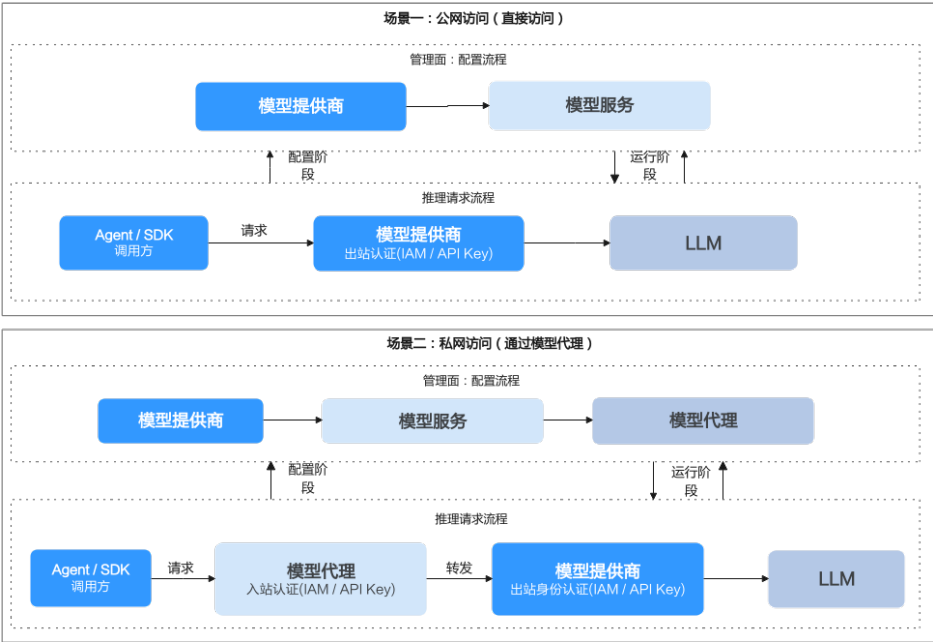
在模型提供商下添加模型服务，定义具体的模型名称、协议和类型。
3.

关联模型代理，将模型提供商与网关关联，由模型代理提供统一的推理服务访问入口。

工作原理

模型管理支持两种访问方式：私网访问（通过模型代理）和公网访问（直接访问），根据模型提供商的网络配置决定。

图 10-1 工作原理



公网访问（直接访问）

适用于网络配置为“公网”的模型提供商，无需创建模型代理，调用方直接通过 base\_url 访问后端。

1. 调用方发起请求：Agent或SDK直接向模型提供商的base\_url发起推理请求。
2. 出站身份认证：模型供应商下的模型服务的鉴权方式。
3. 访问后端：请求被转发到模型提供商后端（如OpenAI、Anthropic等），返回推理结果。

私网访问（通过模型代理）

适用于网络配置为“私网”的模型提供商，需创建模型代理作为访问入口。

1. 调用方发起请求：Agent或SDK通过 /inference/{path} 向模型代理发起推理请求。
2. 入站认证：模型代理验证调用方身份，确保只有授权的调用方才能访问后端模型，防止未授权访问。入站认证在创建模型代理时配置。
3. 转发到模型提供商：模型代理将请求转发到关联的模型提供商。
4. 出站身份认证：模型供应商下的模型服务的鉴权方式。
5. 访问后端：请求被转发到模型提供商后端（如OpenAI、Anthropic等），返回推理结果。

使用流程

1. 创建模型提供商：配置模型提供商的访问地址（base\_url）、出站身份认证和网络配置。详细操作请参见[创建模型提供商](#)。

2. 添加模型服务：在模型提供商下添加具体的模型服务，定义模型名称、API接口协议和模型类型。详细操作请参见[添加模型服务](#)。
3. 创建模型代理：创建模型代理并关联模型提供商，配置入站身份认证和入站网关，请求通过/inference/路径路由到模型提供商。详细操作请参见[添加模型代理](#)。
4. 调用模型服务：通过模型代理的访问地址调用已配置的模型服务。详细操作请参见[集成模型提供商](#)。

支持的 API 接口协议

模型服务支持三种API接口协议，不同协议对应不同的模型类型和请求路径。

表 10-2 API 接口协议说明

| API接口协议         | 说明                                          | 支持的模型类型                           |
|-----------------|---------------------------------------------|-----------------------------------|
| 标准OpenAI协议      | 兼容OpenAI API标准的协议，适用于OpenAI、DeepSeek等模型提供商。 | 文本对话/图像理解、多模态对话、文本向量化、文本排序、实时语音对话 |
| Anthropic协议     | 兼容Anthropic API标准的协议，适用于Anthropic等模型提供商。    | 文本对话/图像理解、Token计数                 |
| 华为云MaaS标准API V1 | 华为云Model as a Service 标准协议，适用于华为云 MaaS 服务。  | 文本对话/图像理解、文本向量化、文本排序              |

支持的模型类型

每种模型类型对应一个标准请求路径，网关根据模型类型自动路由推理请求。

表 10-3 模型类型与请求路径映射

| 模型类型                     | 说明                   | 请求路径                 | 对应API协议     |
|--------------------------|----------------------|----------------------|-------------|
| 文本对话/图像理解（ completions ） | 文本生成和图像理解对话          | /v1/chat/completions | 标准OpenAI协议  |
| 多模态对话（ responses ）       | 多模态对话交互              | /v1/responses        | 标准OpenAI协议  |
| 文本向量化（ embeddings ）      | 将文本转换为向量表示           | /v1/embeddings       | 标准OpenAI协议  |
| 文本排序（ rerank ）           | 对文本进行相关性排序           | /v1/rerank           | 标准OpenAI协议  |
| 文本对话/图像理解（ messages ）    | 基于 Anthropic 协议的文本对话 | /v1/messages         | Anthropic协议 |
| 实时语音对话（ realtime ）       | 实时语音交互               | /v1/realtime         | 标准OpenAI协议  |

| 模型类型                       | 说明          | 请求路径                          | 对应API协议     |
|----------------------------|-------------|-------------------------------|-------------|
| Token计数<br>( token_count ) | 计算输入Token数量 | /v1/messages/<br>count_tokens | Anthropic协议 |

## 10.2 创建模型提供商

模型提供商是模型服务的管理实体，定义统一的访问端点（base\_url）、出站身份认证和网络配置。通过创建模型提供商，可以将外部模型服务（如OpenAI、Anthropic等）接入AgentArts平台，由网关统一管理推理请求的路由和认证。

一个模型提供商下可配置多个模型服务，也可关联多个模型代理，实现多模型统一接入和一站式管理。

### 前提条件

- 已[开通AgentArts服务](#)并完成服务授权。
- 需具备agentarts:customModelProvider:create权限。如需为IAM 子用户授权，请参考[给IAM用户授权](#)。

### 约束与限制

- 一个模型提供商最多可添加10个模型服务，如需调整配额，请[提交工单](#)申请。
- 一个模型提供商最多可关联5个模型代理，如需调整配额，请[提交工单](#)申请。

### 创建模型提供商

- 步骤1** 登录[AgentArts智能体平台](#)。
- 步骤2** 在左侧导航栏选择“托管与运行 > 模型”，进入模型界面。
- 步骤3** 单击“创建模型提供商”，参数说明请参考[表10-4](#)，然后单击“确定”。

表 10-4 参数说明

| 参数       | 说明                                                                                    |
|----------|---------------------------------------------------------------------------------------|
| 名称       | 模型提供商的名称。<br>命名规则：以小写字母开头，以小写字母或数字结尾，可以包含小写字母、数字和中划线，长度为2-40个字符。系统会默认生成名称，可根据需求自定义修改。 |
| 描述       | 用于说明该模型提供商的内容与用途。                                                                     |
| base_url | 模型地址，以 https:// 开头的完整 URL，长度为1-2048个字符。                                               |
| 模型服务     | 添加该提供商下的模型服务，最多可添加10个。单击“添加模型服务”，详细操作请参见 <a href="#">添加模型服务</a> 。                     |
| 模型代理     | 关联已有的模型代理，最多可关联5个，单击“添加模型代理”，详细操作请参见 <a href="#">添加模型代理</a> 。                         |

| 参数     | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 出站身份认证 | <p>模型供应商下的模型服务的鉴权方式，支持API Key和IAM两种认证方式。</p> <ul style="list-style-type: none"><li>IAM认证<br/>选择出站身份，如下拉框无选项请单击“创建出站身份”，创建“IAM 委托”类型的出站身份，如果无IAM信任委托需<a href="#">创建信任委托</a>。</li></ul> <p><b>图 10-2 创建信任委托</b></p> <div><div>&lt; 创建信任委托 ②</div><div><div><div>IAM新增信任委托，提供更灵活的基于信任策略的配置能力。信任委托只能授予身份策略且仅支持</div><div><div><div>* 委托名称</div><div>iamweituotuo</div></div><div><div>* 信任主体类型</div><div><div><div><input type="radio"/> 普通账号<br/>将账号内资源的操作权限委托给其他华为云账号。</div><div><input checked="" type="radio"/> 云服务<br/>将账号内资源的操作权限委托给华为云服务。</div><div><input type="radio"/> 身份提供商<br/>将账号内资源的操作权限委托给身份提供商。</div><div><input type="radio"/> 自定义信任策略<br/>直接在json中编辑自定义信任策略。</div></div></div><div><div>* 云服务</div><div>选择云服务</div></div><div><div>* 最大会话持续时长 ②</div><div>AgentIdentity</div></div><div><div>描述</div><div>智能体身份服务</div></div></div></div></div><p>使用的IAM信任委托需包含如下action：</p><p>sts::setContext：获取STS5委托所需权限，获取临时凭据时携带用户工作负载访问令牌（workload access token）信息。</p><p>策略内容</p><pre>1  { 2    "Version": "5.0", 3    "Statement": [ 4      { 5        "Action": [ 6          "sts:agencies:assume", 7          "sts::setContext" 8        ], 9        "Effect": "Allow", 10       "Principal": { 11         "Service": [ 12           "service.AgentIdentity" 13         ] 14       } 15     ] 16   } 17 }</pre><ul style="list-style-type: none"><li>API Key：在转发请求时，将 API Key 附加到请求头或查询参数中。</li></ul></div></div> |



| 参数     | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|        | <ul style="list-style-type: none"><li>- 选择出站身份，如下拉框无选项请单击“创建出站身份”，创建“API Key”类型的出站身份。</li><li>- 选择位置。<br/>选择 API Key 的传递位置。<ul style="list-style-type: none"><li>▪ 标头：在HTTP请求的头部字段中传递API Key。</li><li>▪ 查询参数：在HTTP请求的URL中传递API Key。</li></ul></li><li>- 参数名称（可选）：输入参数名称，在请求中传递API的密钥时使用的具体名称，默认为Authorization。</li><li>- 前缀（可选）：在请求头中传递API Key时，API密钥值之前的前缀部分，如在授权标头中使用的“Bearer”。填写时不需要增加空格，平台会自动在密钥值前添加空格组装，例如填写“Bearer”，平台最终组装为“Bearer xxx”进行传递。</li></ul>                                                                                   |
| 出网网络配置 | <p>访问模型提供商后端的网络。</p> <ul style="list-style-type: none"><li>• 公网访问<br/>能够连接到外部的互联网，访问外部的资源和服务。</li><li>• 私网访问<br/>连接内部的私有网络，访问内部的资源和服务。<ul style="list-style-type: none"><li>- 选择已配置的VPC，如未配置，请单击“新建VPC”，具体请参考<a href="#">创建虚拟私有云和子网</a>。</li><li>- 选择已配置的子网，如未配置，请单击“新建子网”，具体请参考<a href="#">创建虚拟私有云和子网</a>。</li><li>- 选择已配置的安全组，如未配置，请单击“新建安全组”，具体请参考<a href="#">创建安全组</a>。</li></ul></li></ul> <p>配置后可单击  刷新。</p> <p>配置后如修改模型供应商信息，则已关联模型代理不可更改出网网络配置。</p> |
| 标签（可选） | <p>由“标签键”和“标签值”组成，用于标识和分类云资源。</p> <p>您可以在TMS中创建预定义标签，用于使用同一标签标识多种云资源，创建后可在标签输入框下拉选择同一标签，具体操作请参考<a href="#">创建预定义标签</a>，创建后单击  刷新。</p> <p>单击“添加标签”，选择在TMS中创建的预定义标签，或自定义输入标签键和标签值。</p> <p>可添加一个或多个标签，最多添加20个标签。</p>                                                                                                                                                                                                                                           |

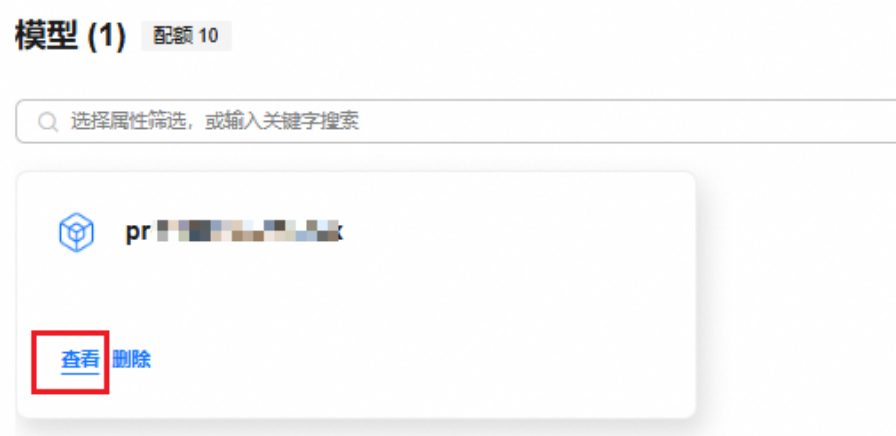
----结束

## 查看模型提供商

模型提供商创建完成后，即可在模型页面展示提供商卡片，单击卡片上的“查看”，查看模型提供商详情。



图 10-3 查看模型提供商



在“基本信息”页签，修改基本信息，添加修改删除模型服务或添加解绑模型代理。

图 10-4 修改模型提供商



在“标签”页签，查看已添加的标签，单击“编辑标签”，可对标签进行管理（最多支持添加20个标签）。

- 添加标签：单击“添加标签”，可选择TMS中的预定义标签，或自定义输入标签键与值。
- 修改标签：直接修改已有标签内容即可。
- 删除标签：单击目标标签后的“删除”即可。

删除模型提供商

删除后无法恢复，请谨慎操作。

如需删除模型提供商，单击卡片上的“删除”，在弹框中确认要删除后输入“DELETE”，然后单击“确定”。

图 10-5 删除模型提供商



说明

模型提供商如有关联的资源（模型服务或模型代理），则无法执行删除操作。请先删除已关联的模型服务和模型代理。

10.3 添加模型服务

模型服务是具体的模型配置，从属于模型提供商。每个模型服务定义了模型名称、API 接口协议和模型类型，是推理请求路由的核心依据。

通过添加模型服务，用户可以在同一个模型提供商下配置多个不同协议和类型的模型，实现多模型统一接入。一个模型提供商最多可添加10个模型服务，如需调整配额，请[提交工单](#)申请。

在创建模型提供商页面或模型提供商详情页面，可以添加模型服务。

添加模型服务

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 模型”，进入模型界面。
- 步骤3 单击已创建的模型提供商卡片上的“查看”或“创建模型提供商”，在“模型服务”区域，单击“添加模型服务”。
- 步骤4 输入模型服务的配置信息，参数说明请参考[表10-5](#)。

表 10-5 模型服务参数说明

| 参数      | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 模型服务名称  | 模型服务名称。<br>命名规则：支持中英文、数字及符号 _./-，须以中英文或数字开头结尾，长度2-64个字符。<br>系统会默认生成名称，可根据需求自定义修改。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 描述      | 用于说明该模型服务的内容与用途。长度不大于1000个字符。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 模型名称    | 模型名称，接口调用请求体中的model字段的取值。<br>由英文、数字及符号（连字符、点、下划线、星号、问号和@）组成，支持 * 的前缀通配以及后缀通配，长度为1-256个字符。<br>例如：gpt-4、claude-3-opus、gpt-*。<br>模型名称支持通配匹配，用于灵活匹配客户端请求中的model字段。 <ul style="list-style-type: none"><li>● <b>精确匹配</b>：模型名称与请求中的model字段完全一致时匹配。例如配置了model=gpt-4，请求model=gpt-4时匹配。</li><li>● <b>前缀通配</b>：模型名称以 * 结尾时，匹配所有以指定前缀开头的模型。例如配置了model=gpt-*，请求model=gpt-4-turbo时匹配。</li><li>● <b>后缀通配</b>：模型名称以 * 开头时，匹配所有以指定后缀结尾的模型。例如配置了model=*-turbo，请求model=gpt-4-turbo时匹配。</li><li>● <b>精确优先</b>：当精确匹配和通配匹配同时可以匹配时，优先使用精确匹配。</li></ul> <b>说明</b><br>通配匹配时，网关会将客户端请求体中的完整model值传递给后端，而不是通配符。例如配置了model=gpt-*，客户端请求model=gpt-4-turbo时，后端收到的model为gpt-4-turbo。 |
| API接口协议 | 选择 API 接口协议，支持以下三种： <ul style="list-style-type: none"><li>● 标准OpenAI协议：兼容OpenAI API标准的协议。选择后，模型类型可选项为：文本对话/图像理解、多模态对话、文本向量化、文本排序、实时语音对话。</li><li>● Anthropic协议：兼容Anthropic API标准的协议。选择后，模型类型可选项为：文本对话/图像理解、Token计数。</li><li>● 华为云MaaS标准API V1：华为云MaaS标准协议。选择后，模型类型可选项为：文本对话/图像理解、文本向量化、文本排序。</li></ul>                                                                                                                                                                                                                                                                                                                                               |
| 模型类型    | 根据所选的API接口协议，选择模型类型。每种模型类型对应一个标准请求路径，请参考 <a href="#">模型类型与请求路径</a> 。可勾选一个或多个模型类型。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

**步骤5** 单击“确定”，完成模型服务的添加。

添加完成后，模型服务将显示在模型提供商详情页面的“模型服务”表格中。

----结束

模型类型与请求路径

每种模型类型对应一个标准请求路径，网关根据模型类型自动路由推理请求。选择不同的 API 接口协议时，可选择的模型类型不同。

表 10-6 模型类型与请求路径映射

| API接口协议          | 模型类型                     | 说明                   | 请求路径                      |
|------------------|--------------------------|----------------------|---------------------------|
| 标准OpenAI协议       | 文本对话/图像理解（ completions ） | 文本生成和图像理解对话          | /v1/chat/completions      |
| 标准OpenAI协议       | 多模态对话（ responses ）       | 多模态对话交互              | /v1/responses             |
| 标准OpenAI协议       | 文本向量化（ embeddings ）      | 将文本转换为向量表示           | /v1/embeddings            |
| 标准OpenAI协议       | 文本排序（ rerank ）           | 对文本进行相关性排序           | /v1/rerank                |
| 标准OpenAI协议       | 实时语音对话（ realtime ）       | 实时语音交互               | /v1/realtime              |
| Anthropic协议      | 文本对话/图像理解（ messages ）    | 基于 Anthropic 协议的文本对话 | /v1/messages              |
| Anthropic协议      | Token计数（ token_count ）   | 计算输入 Token 数量        | /v1/messages/count_tokens |
| 华为云MaaS标准 API V1 | 文本对话/图像理解（ completions ） | 文本生成和图像理解对话          | /v1/chat/completions      |
| 华为云MaaS标准 API V1 | 文本向量化（ embeddings ）      | 将文本转换为向量表示           | /v1/embeddings            |
| 华为云MaaS标准 API V1 | 文本排序（ rerank ）           | 对文本进行相关性排序           | /v1/rerank                |

更多操作

模型服务创建完成后，您还可以执行如表10-7所示更多操作。

表 10-7 更多操作

| 操作     | 说明                                                                                                            |
|--------|---------------------------------------------------------------------------------------------------------------|
| 修改模型服务 | <div><div>1. 在模型提供商详情页面的“模型服务”表格中，单击操作列的“编辑”。</div><div>2. 修改模型服务的配置信息。</div><div>3. 单击“确定”，完成更新。</div></div> |

| 操作     | 说明                                                                                                                |
|--------|-------------------------------------------------------------------------------------------------------------------|
| 删除模型服务 | 若模型提供商已经关联模型代理，则该提供商下必须至少保留一条模型服务，不允许删除最后一个模型服务。<br><br>1. 在模型提供商详情页面的“模型服务”表格中，单击操作列的“删除”。<br>2. 在确认对话框中，单击“确定”。 |

## 10.4 添加模型代理

模型代理是连接模型提供商与网关的桥梁，提供统一的推理服务入口。模型代理配置入站身份认证和入网配置，将推理请求通过 /inference/ 路径路由到关联的模型提供商。

在创建模型提供商页面或模型提供商详情页面，可以创建模型代理。

### 前提条件

已创建至少一个模型提供商，并配置了模型服务。

### 约束与限制

一个模型提供商最多可创建5个模型代理，如需调整配额，请[提交工单](#)申请。

### 关联模型代理

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 模型”，进入模型界面。
- 步骤3 单击已创建的模型提供商卡片上的“查看”或“创建模型提供商”，在“模型代理”区域，单击“添加模型代理”。
- 步骤4 输入模型代理的配置信息，参数说明请参考[表10-8](#)。

表 10-8 模型代理参数说明

| 参数     | 说明                                                                                       |
|--------|------------------------------------------------------------------------------------------|
| 模型代理名称 | 模型代理的名称。<br>命名规则：以小写字母开头，以小写字母或数字结尾，可以包含小写字母、数字和中划线，长度为2-40个字符。<br>系统会默认生成名称，可根据需求自定义修改。 |
| 描述（可选） | 用于说明该模型代理的内容与用途。长度不大于1000个字符。                                                            |

| 参数        | 说明                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 入站身份认证    | <p>配置入站身份认证方式，用于验证调用者的身份。支持以下两种认证方式：</p> <ul style="list-style-type: none"><li>• IAM认证：将华为云IAM身份配置为入站身份认证，以验证调用者的身份。</li><li>• API Key认证：将预设的密钥对（键和值）配置为入站身份认证，以验证请求中携带的密钥合法性。选择后需配置 API Key 名称。</li></ul>                                                                                                                                                                                               |
| API Key名称 | <p>当入站身份认证选择“API Key认证”时填写。用于指定在 HTTP请求头中传递API Key的键名，调用方需在请求头中携带该键名和对应的密钥值。系统会自动生成API Key的名称，可根据需求自定义修改。</p>                                                                                                                                                                                                                                                                                            |
| 入网配置      | <p>选择入站网关，通过互联网或者VPC内网访问模型代理。如下拉框无选项，请单击“<a href="#">前往网络配置</a>”创建新的网络配置。</p>                                                                                                                                                                                                                                                                                                                            |
| 委托        | <p>可使用系统默认创建的AgentArtsGatewayAgency委托。如需手动创建，请单击“创建委托”并参考如下配置创建。</p> <ul style="list-style-type: none"><li>• 委托名称：自定义。</li><li>• 信任主体类型：选择“云服务”。</li><li>• 云服务：搜索service.WorkloadSandboxMetadata。</li><li>• 其余参数使用默认值。</li></ul> <p><b>图 10-6 创建委托</b></p>  <p>委托创建完成后可对委托进行授权，操作具体详情请参考<a href="#">创建委托并授权</a>。</p> |
| 标签（可选）    | <p>由“标签键”和“标签值”组成，用于标识和分类云资源。您可以在TMS中创建预定义标签，用于使用同一标签标识多种云资源，创建后可在标签输入框下拉选择同一标签，具体操作请参考<a href="#">创建预定义标签</a>，创建后单击刷新。</p> <p>单击“添加标签”，选择在TMS中创建的预定义标签，或自定义输入标签键和标签值。</p> <p>可添加一个或多个标签，最多添加20个标签。</p>                                                                                                                                                                                                  |

**步骤5** 单击“确定”，完成模型代理的创建和关联。

----结束

## 解绑模型代理

当模型代理关联的所有模型提供商都被解绑后，模型代理将自动删除。

**步骤1** 在模型代理列表中，选择待操作的模型代理，单击“解绑”。

**步骤2** 在确认对话框中，单击“确定”。

----结束

## 10.5 集成模型提供商

通过集成模型提供商，智能体可以按需快速调用已配置的模型服务，实现LLM推理能力的统一接入和调用。

模型提供商创建完成后，可以通过以下两种方式访问模型服务：

- **通过模型代理访问**：适用于关联了模型代理的模型提供商，通过模型代理的访问端点统一调用。
- **直接访问**：适用于公网网络配置的模型提供商，通过base\_url直接调用。

### 通过模型代理访问

模型提供商关联模型代理后，通过模型代理的访问端点（endpoint）调用模型服务。模型代理提供统一的入站身份认证和请求路由，调用方无需关心后端模型提供商的认证和网络配置。

#### 请求路由

推理请求通过 /inference/ 前缀路径路由到模型代理，与MCP请求的/mcp路径隔离。

#### 请求路径格式：

```
https://{model_proxy_endpoint}/inference/{path}
```

其中 {path} 为标准API路径，如 /v1/chat/completions、/v1/messages 等。

model\_proxy\_endpoint获取方式：

**步骤1** 登录AgentArts智能体平台。

**步骤2** 在左侧导航栏选择“托管与运行 > 模型”，进入模型界面。

**步骤3** 单击模型卡片上的“查看”，进入模型详情页面。

**步骤4** 在“模型代理”区域查看获取。



模型代理(5/5)

| 选择属性筛选，或输入关键字搜索 |    |                         |            |        |
|-----------------|----|-------------------------|------------|--------|
| 模型代理名称          | 描述 | 访问URL                   | 入站身份认证     | 入网配置   |
| mc-...          | -- | https://model-proxy-... | API Key 认证 | de-... |
| mc-...          | -- | https://model-proxy-... | API Key 认证 | ga-... |
| mc-...          | -- | https://model-proxy-... | API Key 认证 | ga-... |
| mc-...          | -- | https://model-proxy-... | API Key 认证 | ga-... |
| mc-...          | -- | https://model-proxy-... | IAM 认证     | ga-... |

----结束

### 请求路由流程：

1. 调用方通过模型代理端点发起推理请求，请求路径为 `/inference/{path}`。例如，调用OpenAI协议的文本对话接口，请求路径为 `/inference/v1/chat/completions`，请求体中携带model字段（如 `model=gpt-4`）。
2. 网关从请求体提取model字段，在已注册的模型列表中进行匹配。
3. 模型提供商附加出站认证信息，模型代理将请求转发到模型提供商后端。
4. 模型提供商返回响应，网关将响应返回给调用方。

### 认证方式

调用方需根据模型代理配置的入站身份认证方式，在请求中携带认证信息：

- **IAM 认证：**使用华为云IAM用户凭据进行认证。调用方需在请求中携带IAM用户的访问密钥（AK/SK）。
- **API Key 认证：**使用密钥对进行认证，调用方需在请求头中携带API Key名称和对应的密钥值。

### 说明

请求路径中的 `{path}` 必须与模型服务的API接口协议对应的请求路径一致。例如，OpenAI协议的文本对话模型使用 `/v1/chat/completions`，Anthropic协议的文本对话模型使用 `/v1/messages`。关于模型类型与请求路径的映射关系，请参见[支持的模型类型](#)。

以下以API Key认证为例，通过模型代理调用OpenAI协议的文本对话接口：

```
POST https://{model_proxy_endpoint}/inference/v1/chat/completions HTTP/1.1
Content-Type: application/json
{API_Key_Name}: {api_key_value}
{
 "model": "gpt-4",
 "messages": [
 {
 "role": "user",
 "content": "Hello, how are you?"
 }
]
}
```

以下以IAM认证为例，通过模型代理调用Anthropic协议的文本对话接口：

```
POST https://{model_proxy_endpoint}/inference/v1/messages HTTP/1.1
Content-Type: application/json
Authorization: SDK-HMAC-SHA256 Credential={access_key}/{date}/region/agentarts, ...
{
 "model": "claude-3-opus",
 "messages": [
 {
 "role": "user",
 "content": "Hello, how are you?"
 }
]
}
```



```
]
}
```

## 直接访问

对于公网网络配置的模型提供商，可以不关联模型代理，通过base\_url直接调用模型服务。直接访问时，调用方需自行管理模型提供商的认证信息。

### 说明

- 公网网络配置的模型提供商可以不关联模型代理，直接通过base\_url访问，也可以关联模型代理通过模型代理访问。
- 私网网络配置的模型提供商必须关联模型代理，通过模型代理的VPC网络访问。

以下以 API Key 认证为例，通过base\_url直接调用OpenAI协议的文本对话接口：

```
POST {base_url}/v1/chat/completions HTTP/1.1
Content-Type: application/json
Authorization: Bearer {api_key}
{
 "model": "gpt-4",
 "messages": [
 { "role": "user", "content": "Hello, how are you?" }
]
}
```

以下通过base\_url直接调用Anthropic协议的文本对话接口：

```
POST {base_url}/v1/messages HTTP/1.1
Content-Type: application/json
x-api-key: {api_key}
{
 "model": "claude-3-opus",
 "messages": [
 { "role": "user", "content": "Hello, how are you?" }
]
}
```

# 11 身份认证

身份认证是华为云IAM服务推出的专为Agent提供安全身份标识、安全凭证管理的能力，AgentArts使用该能力，为智能体运行时构建进站认证与出站认证双维度的安全防护体系，确保智能体能够安全地调用各类第三方外部工具和华为云服务资源。

在AI智能体场景中，身份认证面临比传统Web服务更复杂的挑战：智能体不仅需要被外部系统调用，还可能需要代表用户身份去访问其他云服务。因此，AgentArts与Agent Identity服务协同，将身份认证分为进站认证和出站认证两个维度：

- **进站身份认证：**验证调用方的身份，确保只有授权的调用方能够访问智能体运行时的API接口。进站认证作为运行时的“守门人”，对每个请求独立验证，要求所有访问尝试都必须提供显式的身份证明。
- **出站身份认证：**智能体运行时内部代表用户身份访问其他华为云服务或第三方资源时使用的凭据。出站身份的凭据由智能体身份平台统一托管、加密存储，并在Agent调用时按需注入，避免密钥在代码中硬编码。

## 支持的认证方式

AgentArts进站身份认证及出站身份认证支持以下三种方式：

| 认证方式              | 认证凭据            | 请求头格式                                                                                 |
|-------------------|-----------------|---------------------------------------------------------------------------------------|
| IAM认证（AK/SK 签名认证） | AK+SK通过签名算法生成签名 | Authorization: V11-HMAC-SHA256<br>Access={AK},<br>SignedHeaders=...,<br>Signature=... |
| API Key认证         | API Key字符串      | Authorization: Bearer {API Key}                                                       |
| OAuth 2.0认证       | JWT Token       | Authorization: Bearer {JWT Token}                                                     |

## 使用身份认证的组件

## 网关

网关提供入站认证鉴权和出站身份认证两层安全机制，保障调用方与后端工具之间的安全通信。调用网关或调用网关Target需要获取对应凭证：

- **入站身份认证**：调用方访问网关时的认证和鉴权，用于验证调用方身份并控制访问权限。支持IAM认证、API Key、OAuth 2.0认证，详细说明请参考[创建网关](#)中的入站身份认证。
- **出站身份认证**：网关访问后端工具服务时的身份认证，用于配置网关以特定身份访问后端资源，支持IAM认证、API Key、OAuth 2.0认证。详细说明请参考[Target的出站认证配置](#)。

## 运行时

智能体运行时是AgentArts提供的安全隔离、框架无关的智能体托管执行环境，专为生产级智能体设计。

**入站身份认证**：在创建智能体运行时配置，支持IAM认证、OAuth 2.0认证、API Key认证三种方式，用于验证调用方身份并控制访问权限。当前入站身份认证主要在托管智能体时使用，用于托管HTTP以及MCP入站协议的运行时。认证方式配置后不可修改，调用方必须使用与运行时配置一致的认证方式，否则请求将被拒绝。详细说明请参考[入站身份认证](#)。

## 浏览器

浏览器是AgentArts平台为智能体提供的内置工具之一，通过在安全隔离的沙箱容器中运行云端浏览器，使智能体具备网页导航、表单填写等Web自动化能力。创建浏览器时需配置入站身份认证，确保只有经过验证的请求才能访问浏览器沙箱工具中的资源。

**入站身份认证**：在创建浏览器时配置，支持IAM认证和API Key认证两种方式。入站身份认证是指浏览器在接收和处理来自智能体或 SDK 的请求时，对请求进行身份验证，确保只有经过验证的请求才能访问沙箱工具中的资源，提高系统的安全性和可靠性。调用浏览器时需在请求头中携带对应的认证凭据（API Key 或 AK/SK 签名）。详细说明请参考[创建浏览器](#)中的入站身份认证。

## 代码解释器

代码解释器是 AgentArts 的核心组件之一，为 Agent 提供安全隔离的代码执行环境。创建代码解释器时需配置入站身份认证，确保只有经过验证的请求才能访问沙箱工具中的资源。

**入站身份认证**：在创建代码解释器时配置，支持IAM认证和API Key认证两种方式。入站身份认证是指代码解释器在接收和处理来自智能体或 SDK 的请求时，对请求进行身份验证，确保只有经过验证的请求才能访问沙箱工具中的资源，提高系统的安全性和可靠性。调用代码解释器时需在请求头中携带对应的认证凭据（API Key或AK/SK签名）。详细说明请参考[创建代码解释器](#)中的入站身份认证。

## 记忆库

记忆库是智能体用于存储、管理和检索历史信息的核心组件，使智能体能够在后续交互中按需获取相关历史上下文。创建记忆库时需配置入站身份认证，确保只有经过验证的请求才能访问记忆库资源。

**入站身份认证：**记忆库支持IAM认证和API Key认证两种方式。确保只有授权的智能体或应用能够读写记忆库中的短期记忆、长期记忆及用户偏好等数据，详细说明请参见[记忆库入站身份认证](#)。

# 12 网络配置

为了通过互联网或者VPC内网访问智能体，需要在AgentArts智能体开发平台中进行网络配置，通过网络配置，可以管理智能体的入站网络访问方式，支持公网访问和私网（VPC）访问两种类型。

在网络配置界面，系统会默认创建一个名为“defaultgw”的网络配置，您也可以根据业务需要自行创建网络配置。

- 默认网络配置：默认网络配置通常已开启公网访问（未关联VPC），适合快速创建运行时或网关进行功能验证，无需额外配置即可使用。



- 自定义网络配置：可创建关闭公网访问并关联VPC的网络配置，基于VPC内网通信，满足企业网络隔离的安全合规要求；也可为生产环境单独创建网络配置，按业务隔离公网入口，独立管控公网访问开关与关联VPC，杜绝不同环境流量混用。

网络配置创建完成后，您可以在[托管运行时](#)、[创建网关](#)及[添加模型代理](#)时使用。

约束与限制

- 最多可配置5个入站网关。
- 系统会默认创建一个名为“defaultgw”的默认网络配置，默认网络配置可编辑修改描述、是否开启公网访问和关联VPC，不可删除。

配置入站网关

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“治理与安全 > 网络配置”。
- 步骤3 在“网络配置”页面，可以看到系统默认网关。  
默认网关可编辑修改描述、是否开启公网访问和关联VPC，不可删除。
- 步骤4 单击“创建网关”，参数配置请参考表12-1，配置完成后单击“确定”。

表 12-1 参数说明

| 参数   | 说明                                                                                        |
|------|-------------------------------------------------------------------------------------------|
| 名称   | 网关的名称。<br>以字母开头，可以包含小写字母、数字和中划线，长度为2-48个字符。                                               |
| 描述   | （可选）网关的描述信息，提供网关的详细说明，便于管理和理解网关的用途和配置。<br>最多255个字符。                                       |
| 公网访问 | 是否启用公网访问功能。<br>启用后可以从公网访问网关，系统将自动分配公网域名。<br><b>说明</b><br>启用公网访问后，存在将资源暴露在公网中的安全风险，需要谨慎使用。 |

- 步骤5 为网关关联VPC，关联后单击“确定”。
- VPC提供了逻辑隔离的网络环境，确保网关和其关联的资源在一个独立的网络中运行，在VPC中，可以创建多个子网，满足不同的业务需求。同一网关不可重复关联同一VPC，同一个VPC可以被多个网关关联。
- 选择已创建的网关，在“关联VPC”区域单击“关联VPC”。
- 选择已配置的VPC，然后选择子网，如未配置，请单击“新建VPC”或“新建子网”，详细配置操作请参考[创建虚拟私有云和子网](#)。
- 配置后单击 刷新。
- 步骤6 关联后在VPC列表状态展示为“可用”时，则表示关联成功。

图 12-1 关联成功

The screenshot shows the 'VPC (2)' section of the console. A search bar at the top contains the text '选择属性筛选，或输入关键字搜索'. Below it is a table with columns: 'VPC名称ID', '子网名称ID', '状态', and '操作'. The first row shows a VPC named 'vpc-12a9' with a status of '可用' (Available), which is highlighted with a red rectangular box. The '操作' column contains a '删除' (Delete) button.

----结束

## 更多操作

网关创建完成后，您可以执行如下所示更多操作。

### 表 12-2 更多操作

| 操作         | 说明                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 修改网关基本信息   | <p>在入网配置页面，选择已创建的网关。</p> <ul style="list-style-type: none"> <li>修改网关描述：右上角单击“编辑”，修改网关描述。</li> <li>修改公网访问设置：在“公网访问”处可修改是否开启公网访问。</li> </ul> <p><b>图 12-2 修改公网访问设置</b></p> <p>The screenshot shows the configuration page for a gateway. At the top, there are two tabs: '默认' (Default) and 'g-xxxxxx3'. The 'g-xxxxxx3' tab is selected. Below the tabs, the gateway name 'ga-xxxxxx' is shown with a green dot and the text '运行中' (Running). Underneath, the ID '8b326d67-xxxxxx-xxxxxx-xxxxxx-xxxxxx' is displayed. At the bottom, there is a section for '公网访问' (Public Network Access) with a toggle switch currently set to '已开启' (Enabled). A red box highlights this toggle switch.</p> |
| 删除网关关联的VPC | <p>在VPC列表单击操作列的“删除”，在弹框中单击“确定”。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 删除网关       | <p>在入网配置页面，选择已创建的网关，在右上角单击“删除”，在弹框中单击“确定”。</p> <p><b>说明</b></p> <p>网关删除后无法恢复，请谨慎操作。</p> <ul style="list-style-type: none"> <li>删除网关前需先删除网关关联的VPC才可删除成功。</li> <li>删除网关前需删除调用网关的智能体运行时才可删除成功。</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

# 13 AgentArts 运行时 SDK

## 13.1 SDK 介绍

在开发智能代理应用时，开发者经常面临接口不统一、开发工具链缺失、最佳实践缺乏封装以及扩展性不足等问题，导致开发效率低下。AgentArts SDK通过以下方式解决上述痛点：

- **标准化接口**：提供统一的API规范，简化平台集成。
- **开发工具链**：内置CLI工具，支持快速创建、调试和部署。
- **最佳实践封装**：内置错误处理、日志记录、性能优化等最佳实践。
- **灵活扩展**：支持自定义中间件、插件和工具集成。

华为云AgentArts SDK是一个专为华为云AgentArts平台设计的面向开发者的Python开发工具包。该SDK支持使用Langchain/LangGraph、GoogleADK等开源高码Agent框架开发的智能体，实现一键部署到华为云，从而实现零基础设施管理及完全可控的云端部署。它通过本地代码开发Agent和能力增强（沙箱工具、记忆库、网关）的核心优势，帮助用户将大模型能力快速转化为可落地的业务智能体。

### 使用场景

1. **纯托管模式**：您已经在本地使用LangChain、LangGraph等开源框架开发了一个成熟的Agent，目前只需要一个安全、高可用的生产环境将其运行起来。
2. **能力增强模式**：您的Agent正在开发中，且需要使用复杂的记忆管理、安全的执行沙箱或对接企业内网API。您希望直接调用云上组件，拒绝重复造轮子。

### SDK 版本说明

[表1](#)提供了AgentArts SDK支持的SDK列表，您可以在GitHub仓库查看SDK更新历史、获取安装包以及查看指导文档。



表 13-1 SDK 列表

| 编程语言   | Github地址                             | 发布时间    | 版本号    | 说明                                                                                                                                                                                                                                                                                                 |
|--------|--------------------------------------|---------|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python | <a href="#">agentarts-sdk-python</a> | 2026-08 | v0.1.6 | 新增记忆插件                                                                                                                                                                                                                                                                                             |
|        |                                      | 2026-08 | v0.1.5 | <ul style="list-style-type: none"><li>长期记忆增加过滤、分页查询接口；修复记忆存储并发读写异常。</li><li>适配Python3.13；更新底层华为云鉴权 sdk 版本。</li><li>CLI工具优化：agentarts dev 本地调试支持热重载；修复镜像构建时 SWR 仓库域名拼接 bug。</li><li>完善网关协议兼容性，支持服务端推送通知、工具列表增量更新，优化网关会话生命周期管理。</li><li>上下文对象新增 trace_id、span_id 全链路追踪字段，日志携带链路 ID，便于问题排查。</li></ul> |
|        |                                      | 2026-07 | v0.1.4 | <ul style="list-style-type: none"><li>新增集成测试套件</li><li>CLI增加环境检测能力</li></ul>                                                                                                                                                                                                                       |
|        |                                      | 2026-06 | v0.1.3 | <ul style="list-style-type: none"><li>新增代码解释器的SSL证书校验能力。</li><li>重构智能体身份认证底层依赖，将原有鉴权修改为agent identity unionsdk。</li><li>新增Runtime运行时CLI命令，支持沙箱执行命令、文件上传功能。</li><li>新增CLI配置命令，用于快速设置环境变量。</li></ul>                                                                                                 |
|        |                                      | 2026-05 | v0.1.2 | 适配SWR组织名称，使其可使用随机字符串。                                                                                                                                                                                                                                                                              |
|        |                                      | 2026-05 | v0.1.1 | 增加安全选项。                                                                                                                                                                                                                                                                                            |
|        |                                      | 2026-04 | v0.1.0 | v0.1.0是agentarts-sdk首次版本发布，提供了agent开发、部署和生命周期管理，以及记忆、身份、内置工具和mcp网关等运行时能力。                                                                                                                                                                                                                          |

## 支持的区域

西南-贵阳一（cn-southwest-2）

## 13.2 SDK 应用框架

### 框架概述

SDK应用框架基于Starlette Web框架实现，采用装饰器模式将用户函数封装为HTTP端点。这种设计使得开发者无需关心底层网络通信，只需专注于业务逻辑的实现。

### 核心入口类: AgentArtsRuntimeApp

AgentArtsRuntimeApp是SDK的核心类，继承自Starlette框架，提供AI代理部署所需的所有Web服务功能。

**文件位置：** src/agentarts/sdk/runtime/app.py

**基本用法：**

```
from agentarts.sdk import AgentArtsRuntimeApp, RequestContext

app = AgentArtsRuntimeApp()

@app.entrypoint
async def handler(payload: dict, context: RequestContext = None) -> dict:
 """Agent 入口函数"""
 message = payload.get("message", "")
 return {"response": f"Received: {message}"}

if __name__ == "__main__":
 app.run()
```

### 装饰器详解

#### @app.entrypoint - 主入口装饰器

用于注册主处理函数，处理所有到达/inocations端点的请求。

```
@app.entrypoint
def handle_request(payload, context):
 """
 处理Agent调用请求

 Args:
 payload: 请求载荷，包含用户输入
 context: 请求上下文，包含用户ID、会话ID等信息

 Returns:
 响应数据，可以是普通对象、生成器或异步生成器
 """
 prompt = payload.get("prompt")
 # 处理逻辑
 return {"response": f"处理: {prompt}"}
```

**支持的返回类型：**

- 普通对象：自动序列化为JSON响应
- 同步生成器：支持流式响应（SSE）
- 异步生成器：支持异步流式响应

### @app.ping - 健康检查装饰器

用于注册自定义的健康检查处理器。

```
@app.ping
def custom_ping():
 """返回自定义的ping状态"""
 return "Healthy" # 或返回 PingStatus.HEALTHY
```

#### PingStatus枚举值：

- HEALTHY：健康状态
- HEALTHY\_BUSY：忙碌状态
- UNHEALTHY：不健康状态

### @app.websocket - WebSocket装饰器

用于注册WebSocket处理函数，支持双向实时通信。

```
@app.websocket
async def handle_websocket(websocket, context):
 """处理WebSocket连接"""
 await websocket.accept()
 while True:
 try:
 data = await websocket.receive_text()
 # 处理消息
 await websocket.send_text(f"Echo: {data}")
 except Exception:
 break
```

### @app.async\_task - 异步任务装饰器

用于注册异步任务，SDK会自动追踪任务健康状态。

```
@app.async_task
async def background_task():
 """后台异步任务"""
 # 任务执行期间，ping状态会自动设置为HEALTHY_BUSY
 # 任务完成后，自动恢复为HEALTHY
 await asyncio.sleep(10)
```

## 请求上下文

SDK提供两种上下文对象用于获取请求信息：

### AgentArtsRuntimeContext

基于contextvars的线程安全运行时上下文，用于在请求处理过程中存储和获取上下文信息。

```
from agentarts.sdk.runtime.context import AgentArtsRuntimeContext
设置上下文
AgentArtsRuntimeContext.set_user_id("user123")
AgentArtsRuntimeContext.set_session_id("session456")
AgentArtsRuntimeContext.set_workload_access_token("token789")
获取上下文
user_id = AgentArtsRuntimeContext.get_user_id()
session_id = AgentArtsRuntimeContext.get_session_id()
token = AgentArtsRuntimeContext.get_workload_access_token()
```

### RequestContext

请求级别的上下文对象，包含请求的详细信息。

```
@app.entrypoint
def handle_request(payload, context: RequestContext):
```

```
"""
context包含以下属性:
- user_id: 用户ID
- session_id: 会话ID
- request_id: 请求ID
- workload_access_token: 工作负载访问令牌
"""

print(f"User: {context.user_id}")
print(f"Session: {context.session_id}")
return {"status": "ok"}
```

框架请求处理流程



13.3 快速开始

环境准备

- 操作系统：**推荐使用Linux Ubuntu 22.04 ARM64**，服务器可访问公网。  
本文命令如无特殊说明，均以Ubuntu为例。为保证镜像构建与云端部署的兼容性，建议使用Ubuntu系统。  
因此，为了减少部署失败和额外排查成本，建议直接使用Ubuntu 22.04 ARM64制作AgentArts镜像。  
若使用CentOS、EulerOS等非Ubuntu系统，请将下命令中的apt替换为yum或dnf，并自行确认Docker 18.06+、Python 3.10+；本文不逐条展开各发行版安装差异。

**注意**

需使用ARM64系统制作的Agent镜像，使用X86系统制作的镜像在调用智能体运行时调用失败。

可以使用华为云ECS服务购买服务器，购买时需要选择鲲鹏架构（ARM64），系统选择Ubuntu。

云服务器控制台 / 弹性云服务器 / 购买弹性云服务器

<  购买弹性云服务器

快速购买 ① 自定义购买 ②

实例  如何选择ECS实例规格?

规格类型选型 业务场景选型

CPU架构 ③

x86计算

鲲鹏计算

- 安装Python：请确保Python 3.10及以上版本已安装。  
大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version检查。如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

非Ubuntu/Debian系统请使用yum或dnf安装Python 3.10+及pip。

- 安装Docker：请确保Docker 18.06及以上版本已安装。如未安装，可以使用如下命令安装：

```
查询 Docker 版本
docker --version

安装Docker
sudo apt update
sudo apt install docker.io
```

安装完成后，建议立即配置国内镜像加速器。在后续执行agentarts launch构建镜像时，Docker需要拉取python:3.10-slim基础镜像，默认从docker.io官方仓库下载，可能超时失败。提前配置加速器可避免此问题。

执行以下命令创建Docker配置目录：

```
sudo mkdir -p /etc/docker
```

执行以下命令配置国内镜像源（直接复制并回车执行）：

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
 "registry-mirrors":[
 "https://docker.m.daocloud.net",
 "https://dockerproxy.net",
 "https://mirror.baidubce.com"
]
}
EOF
```

依次执行以下命令重启Docker服务，使配置生效：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

## 📖 说明

华为云SWR基础版不支持OCI镜像格式，如果您使用的是Docker 27及以上版本，并且需要处理OCI镜像，可以通过设置环境变量来关闭OCI支持。

```
export DOCKER_BUILDKIT=0
或者
export BUILDKIT_USE_OCI_MEDIA_TYPES=0
```

- 执行以下命令安装SDK（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
安装依赖包
apt install python3.12-venv

创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate

安装sdk
pip install agentarts-sdk
```

若配置镜像加速器，后续在执行agentarts launch命令时，仍无法拉取python:3.10-slim，可参考本文结尾的“常见问题”中的“直接修改Dockerfile基础镜像地址”。

## 📖 说明

- 执行pip install agentarts-sdk命令下载缓慢、超时可以更改为使用如下命令  
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
- 如果系统缺少python3-venv包，导致无法创建虚拟环境，请按照命令回显提示安装python3-venv包。

```
root@ecs-6732:~# python3 -m venv myenv
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

apt install python3.12-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

Failing command: /root/myenv/bin/python3
```

- 执行以下命令配置华为云凭证，获取华为云凭证请参考[认证鉴权](#)。  
export HUAWEICLOUD\_SDK\_AK="your-access-key"  
export HUAWEICLOUD\_SDK\_SK="your-secret-key"

## 本地开发

- 步骤1** 执行如下命令安装langchain、langgraph、langchain-openai。

```
pip install -U langchain langgraph langchain-openai
```

执行此命令时，如出现下载缓慢，可以更改使用如下命令。

```
pip install -U langchain langgraph langchain-openai -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
```

- 步骤2** 执行以下命令初始化一个langgraph项目，支持基本的问答能力（请提前安装langchain、langgraph，可以使用pip install -U langchain langgraph命令安装）。

```
agentarts init -n my-agent -t langgraph
```

执行命令后，根据提示输入Region信息。

```
(myenv) root@ecs-6732:~# agentarts init -n my-agent -t langgraph

Region:
 Default: cn-southwest-2
 Region (cn-southwest-2): cn-southwest-2

Creating project: my-agent
Template: langgraph
Region: cn-southwest-2
Path: .

Created: agent.py
Created: requirements.txt
Created: .agentarts_config.yaml
Created: Dockerfile
[SUCCESS: Project 'my-agent' created successfully!

Project structure
my-agent/
├── agent.py # Agent implementation
├── requirements.txt # Dependencies
├── .agentarts_config.yaml # Configuration
└── Dockerfile # Docker build file

Step 1: Navigate to project directory
 cd my-agent
Step 2: Install dependencies
 pip install -r requirements.txt
Step 3: Edit agent.py to implement your agent logic
Step 4: Deploy to Huawei Cloud
 agentarts deploy
(myenv) root@ecs-6732:~#
```

执行完成后，会在当前目录下创建名为my-agent的项目文件夹。请先执行以下命令进入刚刚创建的项目目录：

```
cd my-agent
```

此时目录下会包含如下目录和文件：

```
my_agent/
├── .agentarts_config.yaml # Project configuration (表示# 项目的配置文件)
├── agent.py # Agent implementation (表示Agent 核心业务代码)
├── Dockerfile # (表示云端部署所需的镜像构建文件)
└── requirements.txt # Python dependencies (表示Python 依赖包列表)
```

执行cat << 'EOF' > agent.py命令，对agent.py文件的内容进行覆盖重写。其中，agent.py代码如下：

代码输入完毕后，新起一行，输入EOF后再按回车可退出编辑页面。

```
import os
from typing import Dict, Any, TypedDict, Annotated
from langgraph.graph.message import add_messages
from agentarts.sdk import AgentArtsRuntimeApp, RequestContext
app = AgentArtsRuntimeApp()

class State(TypedDict):
 messages: Annotated[list, add_messages]
 query: str
 response: str

class LangGraphAgent:
 def __init__(self):
 self.model_name = os.environ.get("MODEL_NAME", "deepseek-v4-flash")
 self.graph = None

 def _build_graph(self):
```

```
from langgraph.graph import StateGraph, END
from langchain_openai import ChatOpenAI
from langchain_core.messages import HumanMessage, AIMessage

llm = ChatOpenAI(
 model=self.model_name,
 api_key=os.environ.get("MODEL_API_KEY"),
 base_url=os.environ.get("MODEL_URL", "https://api.modelarts-maas.com/openai/v1")
)

async def process_node(state: State) -> Dict[str, Any]:
 query = state.get("query", "")
 messages = state.get("messages", []) or [HumanMessage(content=query)]
 response = await llm.ainvoke(messages)
 return {
 "messages": [AIMessage(content=response.content)],
 "response": response.content,
 }

workflow = StateGraph(State)
workflow.add_node("process", process_node)
workflow.set_entry_point("process")
workflow.add_edge("process", END)
return workflow.compile()

async def run(self, query: str) -> Dict[str, Any]:
 graph = self._graph or self._build_graph()
 self._graph = graph
 result = await graph.ainvoke({"messages": [], "query": query, "response": ""})
 return {"response": result.get("response", "")}

_agent = LangGraphAgent()

@app.entrypoint
async def handler(payload: Dict[str, Any], context: RequestContext = None) -> Dict[str, Any]:
 query = payload.get("message", "")
 return await _agent.run(query)

if __name__ == "__main__":
 app.run()
```

**步骤3** 使用 `vi .agentarts_config.yaml` 命令打开配置文件，修改环境变量。

执行以下命令进入文件编辑：

```
vi .agentarts_config.yaml
```

打开文件后，找到 `environment_variables` 部分。默认生成的配置如下（变量值为空）：

```
environment_variables:
- key: OPENAI_API_KEY
 value: null
- key: OPENAI_MODEL_NAME
 value: null
- key: OPENAI_BASE_URL
 value: null
```

需要将上述环境变量修改为以下内容：

- 将 `OPENAI_API_KEY` 修改为 `MODEL_API_KEY`，并将 `value` 填写为华为云MaaS服务的API Key值。获取方法请参考[获取华为云MaaS服务模型API Key](#)。
- 将 `OPENAI_MODEL_NAME` 修改为 `MODEL_NAME`，`value` 填写为 `deepseek-v4-flash`。



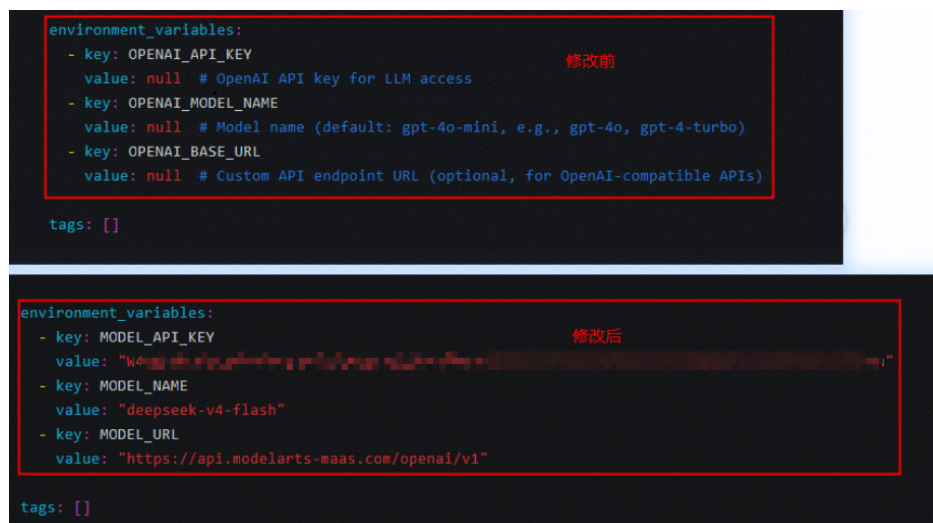
- 将OPENAI\_BASE\_URL修改为MODEL\_URL，value填写为https://api.modelarts-maas.com/openai/v1。

修改后的environment\_variables部分如下所示：

```
environment_variables:
- key: MODEL_API_KEY
 value: "请替换为华为云MaaS服务中模型的API Key"
- key: MODEL_NAME
 value: "deepseek-v4-flash"
- key: MODEL_URL
 value: "https://api.modelarts-maas.com/openai/v1"
```

修改完成后，按Esc键进入命令模式，输入:wq并按Enter键保存退出。

图 13-1 环境变量修改前后对比



**步骤4** 在my-agent目录下，依次执行以下命令安装依赖并启动本地测试服务器：

```
安装项目所需的依赖包
pip install -r requirements.txt

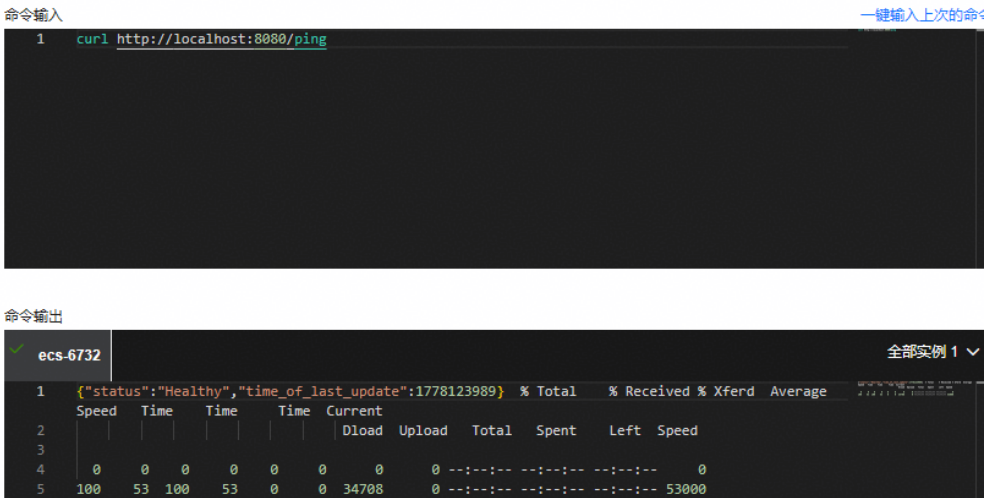
启动本地开发服务（该命令会自动读取 yaml 配置并启动服务）
agentarts dev
```

服务启动后，可以打开一个新的终端窗口（保持原窗口运行），使用 curl 命令进行测试：

- 健康检查调用示例  
curl http://localhost:8080/ping

检查成功结果示例

```
{"status":"Healthy","time_of_last_update":1778123989}
```

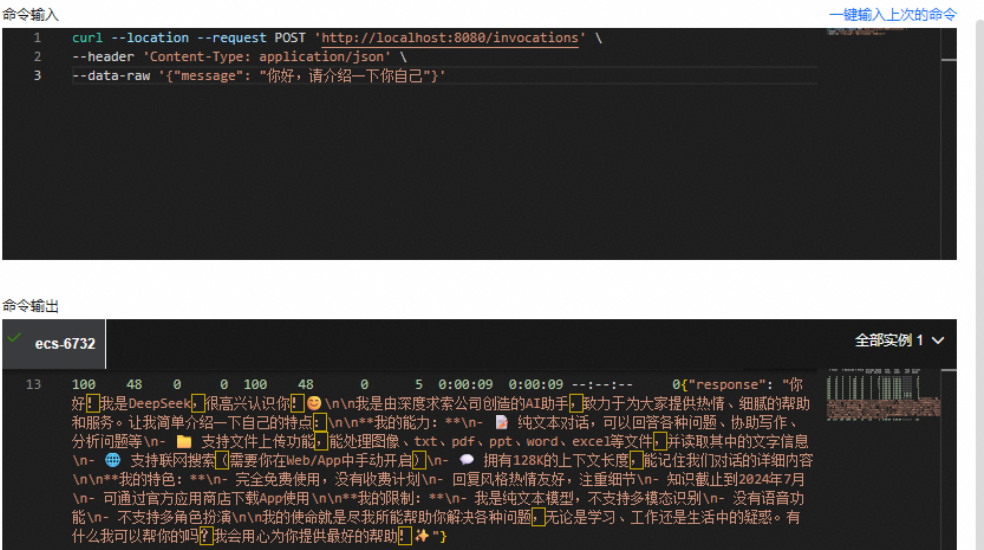


- Agent对话调用示例

```
curl --location --request POST 'http://localhost:8080/invocations' \
--header 'Content-Type: application/json' \
--data-raw '{"message": "你好, 请介绍一下你自己"}'
```

调用成功结果示例

```
{"response": "你好! 我是 **xxx**"}
```



----结束

云端部署

- 步骤1 当本地测试无误后, 通过Ctrl+C停止正在运行中的本地脚本。将Agent部署到华为云时, 首先设置部署区域。
- ```
agentarts config set region cn-southwest-2
```
- ```
(myenv) root@ecs-6732:~/my-agent# agentarts config set region cn-southwest-2
Done: Set region = cn-southwest-2 for agent my-agent
```
- 也可执行agentarts config命令进行交互式配置, 配置向导会引导您完成以下设置:  
(按“回车”使用默认值)



## 📖 说明

（可选）如果遇到拉取python:{version}-slim包失败，请修改docker镜像源。

执行以下命令在系统配置目录下创建一个名为docker的文件夹。

```
sudo mkdir -p /etc/docker
```

执行以下命令（直接复制下面这段代码并回车执行），更换Docker镜像源，从国内镜像站寻找python:3.10-slim。

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
 "registry-mirrors":[
 "https://docker.m.daocloud.net",
 "https://dockerproxy.net",
 "https://mirror.baidubce.com"
]
}
EOF
```

依次执行下面两条命令，重启Docker服务：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

## 步骤3 调用云端Agent进行会话。

```
agentarts invoke '{"message": "Hello"}'
```

```
(venv) root@ecs-6732:~/my-agent# agentarts invoke '{"message": "Hello"}'
Invoke Request
Mode: Cloud
Agent: my-agent
Session: 7a29911c-ab64-431c-ae74-26706a1c2e9d
Endpoint: https://defaultgw-heywcyzvhf.cn-southwest-2.huaweicloud-agentarts.com
Auth Type: IAM

Response:
{
 "response": "Hello! How can I help you today?"
}
(venv) root@ecs-6732:~/my-agent#
```

----结束

## 常见问题

- 执行pip install agentarts-sdk命令，出现  
HTTPSConnectionPool(host='files.pythonhosted.org', port=443): Read  
timed out报错。  
请重新执行pip install agentarts-sdk命令进行安装。
- 执行agentarts launch命令，出现failed to resolve reference "docker.io/  
library/python:3.10-slim"类型报错。  
问题现象及原因  
尝试连接Docker官方镜像仓库（docker.io）下载python:3.10-slim基础环境时，网  
络连接超时（网络连接被阻断或者限速）。

```
(venv) root@ecs-3985arm:~/my-agent# agentarts launch
Deploy Configuration
Agent: my-agent
Mode: cloud
Region: cn-southwest-2

Deploy Progress
Step 1/5 Building Docker image
Image: my-agent:latest

Building Docker image: my-agent:latest
- Step 1/13: FROM python:3.10-slim
Error building image:
Last 15 lines of output:
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/

Sending build context to Docker daemon 13.82kB

Step 1/13 : FROM python:3.10-slim
failed to resolve reference "docker.io/library/python:3.10-slim": failed to do request: Head
'https://registry-1.docker.io/v2/library/python/manifests/3.10-slim': dial tcp 47.88.58.234:443: connect: connection refused
```

### 解决方案一：配置Docker国内镜像加速器（推荐）

执行以下命令在系统配置目录下创建一个名为docker的文件夹。

```
sudo mkdir -p /etc/docker
```

执行以下命令（直接复制下面这段代码并回车执行），更换Docker镜像源，从国内镜像站寻找python:3.10-slim。

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
 "registry-mirrors":[
 "替换为国内可用的Docker镜像源地址"
]
}
EOF
```

依次执行下面两条命令，重启Docker服务：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

Docker配置完成后，再次执行agentarts launch命令进行Agent部署。

### 解决方案二：直接修改Dockerfile基础镜像地址

- 在项目根目录下找到Dockerfile（由agentarts launch自动生成或已存在）。
- 编辑Dockerfile，将FROM python:3.10-slim替换为国内镜像地址，例如：  
sed -i 's|FROM python:3.10-slim|FROM docker.m.daocloud.io/library/python:3.10-slim|g' Dockerfile

或手动修改为：

```
FROM docker.m.daocloud.io/library/python:3.10-slim
```

保存后重新执行agentarts launch命令。

- 执行agentarts launch命令，在执行requirements.txt步骤中出现Read timed out ... files.pythonhosted.org报错。

问题现象及原因

Docker镜像在构建过程中，尝试从官方PyPI服务器下载在requirements.txt里写的那些Python包，由于网络环境原因造成超时。



```
(venv) root@ecs-3985arm:~/my-agent# agentarts launch
Deploy Configuration
Agent: my-agent
Mode: cloud
Region: cn-southwest-2

Deploy Progress
Step 1/5 Building Docker image
Image: my-agent:latest

Building Docker image: my-agent:latest
+ Step 1/13: FROM python:3.10-slim
+ Step 2/13: ENV DOCKER_CONTAINER=true
+ Step 3/13: ENV HUAWECLOUD_SDK_REGION=cn-southwest-2
+ Step 4/13: RUN groupadd -g 1000 appuser && useradd -u 1000 -g 1000 -m -s /bin/bash appuser
+ Step 5/13: WORKDIR /app
+ Step 6/13: RUN chown appuser:appuser /app
+ Step 7/13: COPY requirements.txt .
+ Step 8/13: RUN pip install --no-cache-dir -r requirements.txt
x [9]ERROR: Exception:
x File "/usr/local/lib/python3.10/site-packages/pip/_vendor/urllib3/response.py", line 438, in _error_catcher
x TimeoutError: The read operation timed out
x with self._error_catcher():
x File "/usr/local/lib/python3.10/site-packages/pip/_vendor/urllib3/response.py", line 443, in _error_catcher
x raise ReadTimeoutError(self._pool, None, "Read timed out.")
x pip._vendor.urllib3.exceptions.ReadTimeoutError: HTTPConnectionPool(host='files.pythonhosted.org', port=443): Read timed out.

Error building image:
Error details:
[9]ERROR: Exception:
x File "/usr/local/lib/python3.10/site-packages/pip/_vendor/urllib3/response.py", line 438, in _error_catcher
x TimeoutError: The read operation timed out
x with self._error_catcher():
x File "/usr/local/lib/python3.10/site-packages/pip/_vendor/urllib3/response.py", line 443, in _error_catcher
x raise ReadTimeoutError(self._pool, None, "Read timed out.")
x pip._vendor.urllib3.exceptions.ReadTimeoutError: HTTPConnectionPool(host='files.pythonhosted.org', port=443): Read timed out.
```

## 解决方案

执行以下命令让其在安装依赖时使用华为云官方的Python镜像加速器。

```
sed -i 's|pip install --no-cache-dir -r requirements.txt|pip install --no-cache-dir -r requirements.txt -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com|g' Dockerfile
```

配置完成后，再次执行agentarts launch命令进行Agent部署。

- **执行agentarts launch命令，出现Organization 'agentarts-org1' already exists、Failed to create/get organization 'agentarts-org1'报错。**

SWR服务镜像名称已被占用，请重新更换名称，即更换agentarts\_config.yaml文件中swr\_config配置中organization参数的值、artifact\_source参数中的值（在yaml文件中，该值共有2处需要替换）。

可依次执行以下命令进行替换，也可以直接手动执行nano .agentarts\_config.yaml命令进入到yaml文件中进行修改。

```
将agentarts_config.yaml文件中原有的 agentarts-org1 组织名替换为一个不容易重名的名字，比如 org_agent_6731
sed -i 's/organization: agentarts-org1/organization: org_agent_6731/g' .agentarts_config.yaml
```

```
修正 artifact_source 里的 URL 中的组织名，这里以 org_agent_6731 为例
sed -i 's|url: swr.cn-southwest-2.myhuaweicloud.com/agentarts-org1/agent_my-agent:latest|url: swr.cn-southwest-2.myhuaweicloud.com/org_agent_6731/agent_my-agent:latest|g' .agentarts_config.yaml
```

- **执行agentarts launch命令，出现SVCSTG.SWR.4030017、403 Insufficient permissions报错。**

如果使用IAM账户执行agentarts launch命令产生该报错，表示IAM用户没有SWR服务中创建组织或管理资源的权限。请联系主账号授予SWRFullAccessPolicy权限（在IAM新版控制台中授权，IAM控制台“总览”页面右上角处可切换至新版）。

- **执行agentarts launch命令弹窗要求修改配置repository及swr。**

需手动编辑 .agentarts\_config.yaml文件中的配置。

- 使用编辑器打开 .agentarts\_config.yaml 文件。
- 修改 swr\_config 部分，在 .agentarts\_config.yaml 文件中，找到 swr\_config 部分。修改以下字段：
  - organization: 组织名称
  - repository: 仓库名称

- organization\_auto\_create: 是否自动创建组织
- repository\_auto\_create: 是否自动创建仓库
- 执行agentarts launch命令，出现ReadTimeoutError: Read timed out报错。  
需修改Dockerfile中的pip install命令，添加国内镜像源和超时时间。
  - a. 执行以下命令，打开Dockerfile：  

```
vi Dockerfile
```
  - b. 在文件中找到如下行：  

```
RUN pip install --no-cache-dir -r requirements.txt
```
  - c. 将该行替换为以下内容（添加镜像源和超时时间）：  

```
RUN pip install --no-cache-dir -i https://repo.huaweicloud.com/repository/pypi/simple --timeout 100 -r requirements.txt
```
  - d. 按Esc键进入命令模式，输入:wq并按Enter键保存退出。

## 13.4 CLI

### CLI 概述

AgentArts Starter toolkit是SDK提供的命令行工具，用于在终端中管理Agent的部署、调用、状态查询和资源销毁等操作。

### 环境准备

- 操作系统：**推荐使用Linux Ubuntu 22.04 ARM64**，服务器可访问公网。  
本文命令如无特殊说明，均以Ubuntu为例。为保证镜像构建与云端部署的兼容性，建议使用Ubuntu系统。  
因此，为了减少部署失败和额外排查成本，建议直接使用Ubuntu 22.04 ARM64制作AgentArts镜像。  
若使用CentOS、EulerOS等非Ubuntu系统，请将下命令中的apt替换为yum或dnf，并自行确认Docker 18.06+、Python 3.10+；本文不逐条展开各发行版安装差异。

#### ⚠ 注意

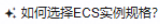
需使用ARM64系统制作的Agent镜像，使用X86系统制作的镜像在调用智能体运行时调用失败。

可以使用华为云ECS服务购买服务器，购买时需要选择鲲鹏架构（ARM64），系统选择Ubuntu。

云服务器控制台 / 弹性云服务器 / 购买弹性云服务器

<  购买弹性云服务器

快速购买  自定义购买 

实例  

规格类型选型 业务场景选型

CPU架构 

x86计算



- 安装Python：请确保Python 3.10及以上版本已安装。

大多数Linux发行版（如Ubuntu）都预装了Python，您可以先通过python3 --version检查。如未安装，可以使用如下命令安装：

```
sudo apt update
sudo apt install python3
```

非Ubuntu/Debian系统请使用yum或dnf安装Python 3.10+及pip。

- 安装Docker：请确保Docker 18.06及以上版本已安装。如未安装，可以使用如下命令安装：

```
查询 Docker 版本
docker --version
```

```
安装Docker
sudo apt update
sudo apt install docker.io
```

安装完成后，建议立即配置国内镜像加速器。在后续执行agentarts launch构建镜像时，Docker需要拉取python:3.10-slim基础镜像，默认从docker.io官方仓库下载，可能超时失败。提前配置加速器可避免此问题。

执行以下命令创建Docker配置目录：

```
sudo mkdir -p /etc/docker
```

执行以下命令配置国内镜像源（直接复制并回车执行）：

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
 "registry-mirrors":[
 "https://docker.m.daocloud.net",
 "https://dockerproxy.net",
 "https://mirror.baidubce.com"
]
}
EOF
```

依次执行以下命令重启Docker服务，使配置生效：

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

### 说明

华为云SWR基础版不支持OCI镜像格式，如果您使用的是Docker 27及以上版本，并且需要处理OCI镜像，可以通过设置环境变量来关闭OCI支持。

```
export DOCKER_BUILDKIT=0
或者
export BUILDKIT_USE_OCI_MEDIA_TYPES=0
```

- 执行以下命令安装SDK（建议在Python虚拟环境中安装，以避免与系统包产生冲突）。

```
安装依赖包
apt install python3.12-venv

创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate
```

```
安装sdk
pip install agentarts-sdk
```

若配置镜像加速器，后续在执行agentarts launch命令时，仍无法拉取python:3.10-slim，可参考本文结尾的“常见问题”中的“直接修改Dockerfile基础镜像地址”。



📖 说明

- 执行pip install agentarts-sdk命令下载缓慢、超时可以更改为使用如下命令  
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
- 如果系统缺少python3-venv包，导致无法创建虚拟环境，请按照命令回显提示安装python3-venv包。

```
root@ecs-6732:~# python3 -m venv myenv
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

apt install python3.12-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

Failing command: /root/myenv/bin/python3
```

- 执行以下命令配置华为云凭证，获取华为云凭证请参考[认证鉴权](#)。  
export HUAWEICLOUD\_SDK\_AK="your-access-key"  
export HUAWEICLOUD\_SDK\_SK="your-secret-key"
- 可通过以下命令获取详细命令信息。  
agentarts --help  
通过以下命令查看调试日志  
agentarts --verbose

命令列表

表 13-2 命令列表

| 命令                                | 说明                                                                                     |
|-----------------------------------|----------------------------------------------------------------------------------------|
| <b>init - 初始化agent项目</b>          | 命令用于初始化一个新的AgentArts项目。该命令会创建完整的项目结构，包括Agent实现代码、依赖文件和配置文件，帮助开发者快速开始Agent应用开发。         |
| <b>configure/config - 配置Agent</b> | 命令用于管理AgentArts项目配置。支持创建、更新、查看和管理Agent配置信息，包括基本配置、SWR配置和运行时配置等。                        |
| <b>dev - 本地开发调试Agent</b>          | 命令用于在本地启动开发服务器，方便开发者快速测试和调试 Agent 应用。支持热重载、自定义端口和环境变量配置。                               |
| <b>deploy/launch- 部署Agent</b>     | deploy和launch命令用于将Agent部署到华为云 AgentArts平台。该命令会构建Docker镜像、上传到SWR（容器镜像服务），并创建或更新Agent实例。 |
| <b>invoke - 调用Agent</b>           | 用于调用已部署的Agent，向其发送请求并获取响应。支持同步调用和流式调用两种模式，适用于测试Agent功能、调试业务逻辑以及生产环境调用。                 |
| <b>destroy - 销毁资源</b>             | 用于从华为云AgentArts平台删除Agent及其相关资源。该命令会永久删除Agent实例、配置信息和相关资源，执行前会提示确认，防止误操作。               |
| <b>runtime - 运行时操作</b>            | 用于操作华为云 AgentArts 运行时，提供会话管理、文件传输、命令执行等数据平面操作能力。                                       |

| 命令             | 说明                    |
|----------------|-----------------------|
| Gateway - 网关管理 | 命令用于mcp网关组件的生命周期管理操作。 |

init - 初始化 agent 项目

命令用于初始化一个新的AgentArts项目。该命令会创建完整的项目结构，包括Agent实现代码、依赖文件和配置文件，帮助开发者快速开始Agent应用开发。

表 13-3 可选参数

| 参数         | 简写 | 说明      | 默认值                  |
|------------|----|---------|----------------------|
| --name     | -n | 项目名称    | myagent-{随机字符}       |
| --template | -t | 项目模板类型  | 交互式选择                |
| --path     | -p | 项目创建路径  | 当前目录                 |
| --region   | -r | 华为云区域   | cn-southwest-2       |
| --swr-org  | 无  | SWR组织名称 | agentarts-{随机字符}-org |
| --swr-repo | 无  | SWR仓库名称 | agent_{Agent名称}      |

表 13-4 模板类型

| 模板名称       | 说明                         |
|------------|----------------------------|
| basic      | 基础模板，适合快速开始。               |
| langgraph  | LangGraph框架模板，支持状态工作流。     |
| langchain  | LangChain框架模板，支持工具集成。      |
| google-adk | Google ADK框架模板，支持Gemini模型。 |

执行效果

执行init命令后，会在指定路径下创建以下项目结构：

```
{project_name}/
├── .agentarts_config.yaml # AgentArts配置文件
├── agent.py # Agent实现代码
├── Dockerfile # Docker构建文件
└── requirements.txt # 项目依赖文件
```

- .agentarts\_config.yaml: AgentArts配置文件，包含部署和运行配置。
- agent.py: Agent的主要实现文件，包含入口函数和业务逻辑。
- Dockerfile: Docker镜像构建文件，用于容器化部署。
- requirements.txt: 项目依赖列表，包括SDK和框架包。

使用示例

- 示例 1: 交互式创建项目

```
agentarts init
```

执行后会提示：

- 输入项目名称
- 选择模板类型（输入数字选择）
- 输入华为云区域（默认cn-southwest-2）

- 示例 2: 使用LangGraph模板创建项目

```
agentarts init --name my-agent --template langgraph
```

- 示例 3: 指定路径和区域创建项目

```
agentarts init -n my-agent -t langchain --path ./projects --region cn-southwest-2
```

- 示例 4: 完整参数示例

```
agentarts init \
--name my-agent \
--template langgraph \
--path ./my-projects \
--region cn-southwest-2 \
--swr-org my-organization \
--swr-repo my-repository
```

configure/config - 配置 Agent

命令用于管理AgentArts项目配置。支持创建、更新、查看和管理当前路径下的Agent配置信息，包括基本配置、SWR配置和运行时配置等。

 说明

configure是config命令的别名，两者功能完全相同。

表 13-5 config 参数(交互式配置)

| 参数                | 简写 | 说明        | 是否必填                |
|-------------------|----|-----------|---------------------|
| --name            | -n | Agent名称   | 否，交互式提示             |
| --entrypoint      | -e | Agent入口函数 | 是，必须指定              |
| --region          | -r | 华为云区域     | 否，默认cn-southwest-2  |
| --dependency-file | -d | 依赖文件路径    | 否，自动检测              |
| --swr-org         | 无  | SWR组织名称   | 否，默认agentarts-org   |
| --swr-repo        | 无  | SWR仓库名称   | 否，默认agent_{Agent名称} |

表 13-6 子命令列表

| 子命令                | 说明              |
|--------------------|-----------------|
| config (无子命令)      | 交互式配置 Agent     |
| config list        | 列出所有已配置的 Agent  |
| config get         | 获取配置值或 Agent 详情 |
| config set         | 设置配置值           |
| config set-default | 设置默认 Agent      |
| config remove      | 删除 Agent 配置     |

执行agentarts config后，会依次提示输入：

1. Agent名称（显示已有Agent列表）
2. 入口函数（格式：module:function）
3. 华为云区域（默认cn-southwest-2）
4. 依赖文件（自动检测requirements.txt）
5. SWR组织名称
6. SWR仓库名称

配置完成后：

- 保存配置到 .agentarts\_config.yaml
- 自动生成Dockerfile（如果不存在）

使用示例

- 示例 1: 交互式配置

```
agentarts config
```

按提示输入配置信息，适合首次配置或不确定参数时使用。

- 示例 2: 快速配置

```
agentarts config --name my-agent --entrypoint agent:create_app
```

仅提供必要参数，其他使用默认值。

- 示例 3: 完整参数配置

```
agentarts config \
 --name my-agent \
 --entrypoint agent:create_app \
 --region cn-southwest-2 \
 --dependency-file requirements.txt \
 --swr-org my-organization \
 --swr-repo my-repository
```

- 示例 4: 列出所有Agent

```
agentarts config list
```

- 示例 5: 设置默认Agent

```
agentarts config set-default my-agent
```

dev - 本地开发调试 Agent

dev 命令用于在本地启动开发服务器，方便开发者快速测试和调试 Agent 应用。该命令支持热重载、自定义端口和环境变量配置。

参数解释

| 参数       | 简写 | 说明                       | 默认值                    |
|----------|----|--------------------------|------------------------|
| --port   | -p | 服务器端口                    | 8080                   |
| --host   | -h | 服务器主机地址                  | 0.0.0.0                |
| --reload | -  | 启用热重载（代码变更自动重启）          | False                  |
| --config | -c | 配置文件路径                   | .agentarts_config.yaml |
| --env    | -e | 环境变量（格式：KEY=VALUE），可多次使用 | 从配置文件读取                |

• 环境变量配置

方式一：命令行参数

使用 --env 或 -e 参数直接传递环境变量：

```
agentarts dev --env OPENAI_API_KEY=sk-xxx --env OPENAI_MODEL_NAME=deepseek-v4-flash
```

方式二：配置文件

在 .agentarts\_config.yaml 中配置环境变量：

```
runtime:
 environment_variables:
 - key: OPENAI_API_KEY
 value: "your-api-key"
 - key: OPENAI_MODEL_NAME
 value: "deepseek-v4-flash"
 - key: OPENAI_BASE_URL
 value: ""
```

优先级

命令行 --env 参数优先级高于配置文件，相同变量会使用命令行传入的值。

执行效果

启动开发服务器后，会显示以下信息：

```
Development Server
Host: 0.0.0.0
Port: 8080
Config: .agentarts_config.yaml
Entrypoint: agent:create_app
Auto-reload: enabled
Environment Variables:
 OPENAI_API_KEY=sk-xxxx****xxxx (CLI)
 OPENAI_MODEL_NAME=deepseek-v4-flash (CLI)

Invocation Endpoint: POST http://0.0.0.0:8080/invocations
Health Check: GET http://0.0.0.0:8080/ping
```

使用示例

示例 1: 基本启动

```
agentarts dev
```

示例 2: 指定端口

```
agentarts dev --port 3000
```

示例 3: 启用热重载

```
agentarts dev --reload
```

示例 4: 传递环境变量

```
agentarts dev --env OPENAI_API_KEY=sk-xxx --env OPENAI_MODEL_NAME=deepseek-v4-flash
```

示例 5: 使用简写参数

```
agentarts dev -p 3000 -e OPENAI_API_KEY=sk-xxx -e OPENAI_MODEL_NAME=deepseek-v4-flash
```

示例 6: 指定配置文件

```
agentarts dev --config ./config/agentarts_config.yaml
```

示例 7: 完整参数示例

```
agentarts dev \
--port 3000 \
--host 127.0.0.1 \
--reload \
--env OPENAI_API_KEY=sk-xxx \
--env OPENAI_MODEL_NAME=deepseek-v4-flash \
--env OPENAI_BASE_URL=https://api.openai.com/v1
```

- **端点说明**  
开发服务器启动后，提供以下端点：

| 端点           | 方法   | 说明         |
|--------------|------|------------|
| /invocations | POST | Agent 调用入口 |
| /ping        | GET  | 健康检查       |

调用示例

```
curl -X POST http://localhost:8080/invocations \ -H "Content-Type: application/json" \ -d
'{"message": "Hello, Agent!"}'
```

健康检查示例

```
curl http://localhost:8080/ping
```

- **热重载**  
启用 --reload 参数后，当检测到代码文件变更时，服务器会自动重启：  
agentarts dev --reload  
适用于开发阶段快速迭代。

deploy/launch- 部署 Agent

deploy和launch命令用于将Agent部署到华为云AgentArts平台。该命令会构建Docker镜像、上传到SWR（容器镜像服务），并创建或更新Agent实例。

📖 说明

launch是deploy命令的别名，两者功能完全相同。

表 13-7 参数解释

| 参数                      | 简写 | 说明                             | 默认值       |
|-------------------------|----|--------------------------------|-----------|
| --agent                 | -a | Agent名称                        | 使用默认Agent |
| --mode                  | -m | 部署模式（local/<br>cloud）          | cloud     |
| --tag                   | -t | Docker镜像标签                     | latest    |
| --local-port            | -l | 本地端口映射（本地模式）                   | 从配置文件读取   |
| --swr-org               | 无  | SWR组织（覆盖配置）                    | 从配置文件读取   |
| --swr-repo              | 无  | SWR仓库（覆盖配置）                    | 从配置文件读取   |
| --description           | -d | Agent描述（覆盖配置）                  | 从配置文件读取   |
| --skip-build            | -  | 使用配置文件中的镜像URL创建agent，跳过构建镜像的步骤 | 默认构建并推送镜像 |
| --skip-ssl-verification | -k | 跳过SSL证书验证                      | 默认开启      |

## 部署流程

执行deploy命令会依次执行以下步骤：

1. 配置验证
  - 检查 .agentarts\_config.yaml配置文件
  - 验证必要配置项是否完整
  - 确认Agent入口函数存在
2. 镜像构建
  - 读取Dockerfile构建配置
  - 构建包含Agent代码的Docker镜像
  - 标记镜像版本（latest和时间戳版本）
3. 镜像推送
  - 登录华为云SWR服务
  - 推送镜像到指定组织和仓库
  - 验证镜像推送成功
4. Agent创建/更新
  - 调用AgentArts控制平面API
  - 创建新的Agent或更新现有Agent

- 配置运行时参数和环境变量
5. 部署验证
- 检查Agent状态
  - 验证服务可用性
  - 输出访问端点信息

使用示例

- 示例 1: 基本部署  
agentarts deploy  
使用默认Agent和配置进行部署。
- 示例 2: 指定Agent部署  
agentarts deploy --agent my-agent  
或使用简写：  
agentarts deploy -a my-agent

invoke - 调用 Agent

用于调用已部署的Agent，向其发送请求并获取响应。支持同步调用和流式调用两种模式，适用于测试Agent功能、调试业务逻辑以及生产环境调用。

表 13-8 参数解释

| 参数                      | 简写  | 说明                | 默认值       |
|-------------------------|-----|-------------------|-----------|
| payload                 | -   | JSON格式的请求数据（位置参数） | 必填        |
| --agent                 | -a  | Agent名称           | 使用默认Agent |
| --region                | -r  | 华为云区域             | 从配置文件读取   |
| --mode                  | -m  | 调用模式（local/cloud） | cloud     |
| --port                  | -p  | 本地模式端口            | 8080      |
| --endpoint              | -e  | 指定端点名称            | 无         |
| --session               | -s  | 会话ID（用于有状态Agent）  | 自动生成UUID  |
| --bearer-token          | -bt | Bearer认证令牌        | 无         |
| --timeout               | -   | 请求超时时间（秒）         | 900       |
| --skip-ssl-verification | -   | 跳过 SSL 证书验证       | 默认开启      |
| --user-id               | -u  | OAuth2 出站凭据的用户ID  | 无         |



| 参数            | 简写 | 说明                                                             | 默认值 |
|---------------|----|----------------------------------------------------------------|-----|
| --custom-path | -  | 附加到 / invocations 的自定义路径（例如，'stream' -> / invocations/stream ） | 无   |

使用示例

- 示例 1: 基本调用  
agentarts invoke '{"message": "你好"}'
- 示例 2: 指定Agent调用  
agentarts invoke '{"message": "你好"}' --agent my-agent
- 示例 3: 使用简写参数  
agentarts invoke '{"message": "你好"}' -a my-agent
- 示例 4: 指定区域调用  
agentarts invoke '{"message": "你好"}' --agent my-agent --region cn-southwest-2
- 示例 5: 本地模式调用  
agentarts invoke '{"message": "你好"}' --mode local

destroy - 销毁资源

用于从华为云AgentArts平台删除Agent及其相关资源。该命令会永久删除Agent实例、配置信息和相关资源，执行前会提示确认，防止误操作。

📖 说明

此操作不可逆，删除后无法恢复Agent数据和配置。

表 13-9 参数解释

| 参数                      | 简写 | 说明          | 默认值       |
|-------------------------|----|-------------|-----------|
| --agent                 | -a | Agent名称     | 使用默认Agent |
| --region                | -r | 华为云区域       | 从配置文件读取   |
| --yes                   | -y | 跳过确认提示      | false     |
| --skip-ssl-verification | -k | 跳过 SSL 证书验证 | 开启ssl认证   |

使用示例

- 示例 1: 交互式删除默认Agent  
agentarts destroy  
执行后会提示确认，输入y确认删除。
- 示例 2: 删除指定Agent  
agentarts destroy --agent my-agent

## runtime - 运行时操作

用于操作华为云AgentArts运行时，提供会话管理、文件传输、命令执行等数据平面操作能力。

### 会话管理流程

1. 创建会话  
agentarts runtime start-session --agent my-agent # 输出: {"data": {"session\_id": "session-xxx"}}
2. 执行操作（使用会话 ID）  
# 上传文件  
agentarts runtime upload-files -a my-agent -s session-xxx -f data.txt  
# 执行命令  
agentarts runtime exec-command "pip install pandas" -a my-agent -s session-xxx  
# 下载文件  
agentarts runtime download-files -a my-agent -s session-xxx -p /home/user/result.txt
3. 停止会话  
agentarts runtime stop-session --agent my-agent --session session-xxx

### 会话作用

- 有状态操作：upload-files、download-files、exec-command 可使用会话保持状态。
- 文件传输：同一会话中的文件操作共享文件系统状态。
- 命令执行：同一会话中的命令共享执行环境（如已安装的包）。

子命令概览以及各命令详情如下所示：

| 子命令            | 说明                    | 是否需要会话id |
|----------------|-----------------------|----------|
| start-session  | 创建运行时会话，获取 session_id | 否        |
| invoke         | 调用 Agent 发送 JSON 请求   | 可选       |
| exec-command   | 在运行时中执行命令             | 可选       |
| upload-files   | 上传文件到运行时              | 是        |
| download-files | 从运行时下载文件              | 是        |
| stop-session   | 停止运行时会话               | 是        |

- **start-session**

#### 命令用途

创建运行时会话，返回 session\_id 用于后续有状态操作（如文件传输、命令执行）。

#### 参数解释

| 参数                      | 简写  | 说明                    | 默认值     |
|-------------------------|-----|-----------------------|---------|
| --agent                 | -a  | Agent 名称（必填）          | 无       |
| --region                | -r  | 华为云区域                 | 从配置文件读取 |
| --bearer-token          | -bt | Bearer 认证令牌           | 无       |
| --endpoint              | -e  | 指定端点名称                | 无       |
| --skip-ssl-verification | -k  | 跳过SSL证书验证             | false   |
| --user-id               | -u  | 用户 ID（用于 OAuth2 出站凭据） | 无       |

响应格式

```
{
 "data": {
 "session_id": "session-xxxx-xxxx-xxxx"
 }
}
```

使用示例

- **示例1：基本会话创建**  
agentarts runtime start-session --agent my-agent
- **示例2：指定区域**  
agentarts runtime start-session -a my-agent -r cn-southwest-2
- **示例3：使用 Bearer Token**  
agentarts runtime start-session -a my-agent -bt your-token
- **示例4：指定端点**  
agentarts runtime start-session -a my-agent -e my-endpoint
- **示例5：跳过 SSL 验证**  
agentarts runtime start-session -a my-agent --skip-ssl-verification

• invoke

命令用途

调用Agent发送JSON请求，获取响应。支持自定义路径扩展。

注意: 此命令与顶层agentarts invoke命令功能相同。

参数解释

| 参数         | 简写 | 说明                 | 默认值       |
|------------|----|--------------------|-----------|
| payload    | 无  | JSON 请求数据（位置参数，必填） | 无         |
| --agent    | -a | Agent名称            | 使用默认Agent |
| --region   | -r | 华为云区域              | 从配置文件读取   |
| --endpoint | -e | 指定端点名称             | 无         |

| 参数                      | 简写  | 说明                                                          | 默认值   |
|-------------------------|-----|-------------------------------------------------------------|-------|
| --session               | -s  | 会话 ID（用于有状态 Agent）                                          | 无     |
| --bearer-token          | -bt | Bearer 认证令牌                                                 | 无     |
| --timeout               | 无   | 请求超时时间（秒）                                                   | 900   |
| --skip-ssl-verification | -k  | 跳过 SSL 证书验证                                                 | false |
| --user-id               | -u  | 用户 ID（用于 OAuth2 出站凭据）                                       | 无     |
| --custom-path           | 无   | 自定义路径追加到 /invocations                                       | 无     |
| --skip-ssl-verification | -   | 跳过 SSL 证书验证                                                 | 默认开启  |
| --user-id               | -u  | OAuth2 出站凭据的用户 ID                                           | 无     |
| --custom-path           | -   | 附加到 /invocations 的自定义路径（例如，'stream' -> /invocations/stream） | 无     |

**custom-path 参数说明**

- custom-path 允许在 /invocations 路径后追加自定义路径段：
- 有效格式：字母、数字、连字符、下划线、斜杠（用于嵌套路径）
  - 无效字符：!、@、#、\$、%、^、&、\*、(、)、空格
  - 路径遍历：不允许 .. 序列

示例：

- --custom-path stream → /invocations/stream
- --custom-path api/v2 → /invocations/api/v2

**配置要求**

使用 --custom-path 需要在配置文件中设置 url\_match\_type: PREFIX\_MATCH并部署到云端（或直接在控制台配置）：

```
runtime:
 invoke_config:
 url_match_type: PREFIX_MATCH
```

若配置为 ACCURATE\_MATCH（默认），使用 --custom-path会因后端期望精确路径匹配而失败。

**使用示例**

- **示例1：基本调用**  
agentarts runtime invoke '{"message": "hello"}' --agent my-agent
- **示例2：使用会话 ID**  
agentarts runtime invoke '{"message": "hello"}' -a my-agent -s session-xxx
- **示例3：使用自定义路径**  
agentarts runtime invoke '{"message": "hello"}' -a my-agent --custom-path stream
- **示例4：设置超时时间**  
agentarts runtime invoke '{"message": "hello"}' -a my-agent --timeout 60

• **exec-command**

**命令用途**

在运行时环境中执行命令（如 ls、cat、pip install 等）。

**参数解释**

| 参数                      | 简写  | 说明                                                                                                   | 默认值              |
|-------------------------|-----|------------------------------------------------------------------------------------------------------|------------------|
| command                 | 无   | 要执行的命令<br>（位置参数，必填）                                                                                  | 无                |
| --agent                 | -a  | Agent 名称（必填）                                                                                         | 无                |
| --session               | -s  | 会话 ID                                                                                                | 无                |
| --chunked               | 无   | 启用流式响应<br>（application/x-ndjson）                                                                     | false            |
| --region                | -r  | 华为云区域                                                                                                | 从配置文件读取          |
| --bearer-token          | -bt | Bearer 认证令牌                                                                                          | 无                |
| --endpoint              | -e  | 指定端点名称                                                                                               | 无                |
| --skip-ssl-verification | -k  | 跳过 SSL 证书验证                                                                                          | false            |
| --user-id               | -u  | 用户 ID（用于 OAuth2 出站凭据）                                                                                | 无                |
| --timeout               | 无   | 请求超时时间（秒） <ul style="list-style-type: none"><li>• 默认超时: 60 秒</li><li>• 最大超时: 300 秒，配置要求为正数</li></ul> | 60（最大 300, 要求>0） |

**Chunked 模式说明**

启用 --chunked 后：

- 请求头: Command-Type: chunked
- 响应格式: application/x-ndjson（每行一个 JSON 对象）

使用示例

- **示例1：基本命令执行**  
agentarts runtime exec-command "ls -la" --agent my-agent --session session-xxx
- **示例2：流式输出**  
agentarts runtime exec-command "ls -la" -a my-agent -s session-xxx --chunked
- **示例3：设置超时时间**  
agentarts runtime exec-command "pip install pandas" -a my-agent -s session-xxx --timeout 120
- **示例4：使用 Bearer Token**  
agentarts runtime exec-command "cat /home/user/data.txt" -a my-agent -s session-xxx -bt your-token

- **upload-files**

命令用途

上传本地文件到运行时环境。

参数解释

| 参数                      | 简写  | 说明                    | 默认值     |
|-------------------------|-----|-----------------------|---------|
| --agent                 | -a  | Agent 名称（必填）          | 无       |
| --session               | -s  | 会话 ID（必填）             | 无       |
| --files                 | -f  | 本地文件路径（可多次指定，必填）      | 无       |
| --path                  | -p  | 远程目录路径                | /tmp    |
| --file-user-id          | 无   | 文件所有者用户 ID            | 1000    |
| --file-group-id         | 无   | 文件所有者组 ID             | 1000    |
| --file-mode             | -m  | 文件权限（八进制格式）           | 0644    |
| --region                | -r  | 华为云区域                 | 从配置文件读取 |
| --bearer-token          | -bt | Bearer 认证令牌           | 无       |
| --endpoint              | -e  | 指定端点名称                | 无       |
| --skip-ssl-verification | -k  | 跳过 SSL 证书验证           | false   |
| --user-id               | -u  | 用户 ID（用于 OAuth2 出站凭据） | 无       |
| --timeout               | 无   | 请求超时时间（秒）             | 900     |

文件路径格式：

简单格式

文件上传到 /home/user/{filename}:

```
agentarts runtime upload-files -a my-agent -s session-xxx -f local_file.txt
```

指定远程目录

使用 --path / -p 参数指定远程目录，路径必须以 / 结尾:

```
agentarts runtime upload-files -a my-agent -s session-xxx -f local_file.txt -p /app/data/
```

多文件上传

多次使用 -f 参数，所有文件上传到同一目录:

```
agentarts runtime upload-files -a my-agent -s session-xxx -f file1.txt -f file2.txt -f file3.txt
```

多文件自定义路径

```
agentarts runtime upload-files -a my-agent -s session-xxx \ -f file1.txt -f file2.txt -f file3.txt \ -p /app/data/
```

文件大小限制

单个文件最大 100MB，超过将报错拦截。

配置要求

使用 upload-files 需要在配置文件中启用文件传输并部署到云端（或直接在控制台开启）：

```
runtime:
 invoke_config:
 file_transfer_config:
 enabled: true
```

若配置为 false（默认），上传操作将失败。

使用示例

- **示例1：单文件上传**  
agentarts runtime upload-files --agent my-agent --session session-xxx -f data.txt
- **示例2：多文件上传**  
agentarts runtime upload-files -a my-agent -s session-xxx -f file1.txt -f file2.txt
- **示例3：自定义文件权限**  
agentarts runtime upload-files -a my-agent -s session-xxx -f script.sh --file-mode 0755
- **示例4：自定义所有者**  
agentarts runtime upload-files -a my-agent -s session-xxx -f data.txt --file-user-id 1001 --file-group-id 1001

● download-files

命令用途

从运行时环境下载文件或目录。

参数解释

| 参数          | 简写 | 说明            | 默认值   |
|-------------|----|---------------|-------|
| --agent     | -a | Agent 名称（必填）  | 无     |
| --session   | -s | 会话 ID（必填）     | 无     |
| --path      | -p | 远程文件/目录路径（必填） | 无     |
| --output    | -o | 本地输出路径        | 使用文件名 |
| --recursive | 无  | 下载目录为 tar 归档  | false |

| 参数                      | 简写  | 说明                    | 默认值     |
|-------------------------|-----|-----------------------|---------|
| --region                | 无   | 华为云区域                 | 从配置文件读取 |
| --bearer-token          | -bt | Bearer 认证令牌           | 无       |
| --endpoint              | -e  | 指定端点名称                | 无       |
| --skip-ssl-verification | -k  | 跳过 SSL 证书验证           | false   |
| --user-id               | -u  | 用户 ID（用于 OAuth2 出站凭据） | 无       |
| --timeout               | 无   | 请求超时时间（秒）             | 900     |

配置要求

使用 download-files 需要在配置文件中启用文件传输并部署到云端：

```
runtime:
 invoke_config:
 file_transfer_config:
 enabled: true
```

若配置为 false（默认），下载操作将失败。

使用示例

- **示例1：下载单个文件**  
agentarts runtime download-files --agent my-agent --session session-xxx --path /home/user/data.txt
- **示例2：指定输出路径**  
agentarts runtime download-files -a my-agent -s session-xxx -p /home/user/data.txt -o ./local\_data.txt
- **示例3：下载目录**  
agentarts runtime download-files -a my-agent -s session-xxx -p /home/user/project --recursive
- **示例4：使用 Bearer Token**  
agentarts runtime download-files -a my-agent -s session-xxx -p /home/user/data.txt -bt your-token

• stop-session

命令用途

停止运行时会话，释放会话资源。

参数解释

| 参数             | 简写  | 说明           | 默认值     |
|----------------|-----|--------------|---------|
| --agent        | -a  | Agent 名称（必填） | 无       |
| --session      | -s  | 会话 ID（必填）    | 无       |
| --region       | -r  | 华为云区域        | 从配置文件读取 |
| --bearer-token | -bt | Bearer 认证令牌  | 无       |



| 参数                      | 简写 | 说明                    | 默认值   |
|-------------------------|----|-----------------------|-------|
| --endpoint              | -e | 指定端点名称                | 无     |
| --skip-ssl-verification | -k | 跳过 SSL 证书验证           | false |
| --user-id               | -u | 用户 ID（用于 OAuth2 出站凭证） | 无     |

使用示例

- **示例1：基本会话停止**  
agentarts runtime stop-session --agent my-agent --session session-xxx
- **示例2：简写参数的基本会话停止**  
agentarts runtime stop-session -a my-agent -s session-xxx
- **示例3：使用 Bearer Token**  
agentarts runtime stop-session -a my-agent -s session-xxx -bt your-token

Gateway - 网关管理

- 创建网关  
agentarts gateway create

表 13-10 创建网关参数说明

| 参数                         | 类型          | 是否必选 | 默认值               | 说明                            |
|----------------------------|-------------|------|-------------------|-------------------------------|
| --name                     | string      | 否    | gateway-{8 位随机后缀} | 网关名称                          |
| --description              | string      | 否    | -                 | 网关描述                          |
| --protocol-type            | string      | 否    | mcp               | 网关协议类型                        |
| --agency-name              | string      | 否    | -                 | 用于指定网关委托身份的委托名称               |
| --authorizer-type          | string      | 否    | iam               | 授权器类型（custom_jwt、iam、api_key） |
| --protocol-configuration   | JSON string | 否    | -                 | 协议配置（JSON 格式）                 |
| --authorizer-configuration | JSON string | 否    | -                 | 授权器配置的 JSON 字符串               |

| 参数                               | 类型          | 是否必选 | 默认值 | 说明             |
|----------------------------------|-------------|------|-----|----------------|
| --log-delivery-configuration     | JSON string | 否    | -   | 日志投递配置的JSON字符串 |
| --outbound-network-configuration | JSON string | 否    | -   | 网络配置的JSON字符串   |
| --tags                           | JSON string | 否    | -   | 资源标签（JSON 格式）  |

授权器配置要求：

- 当authorizer-type = 'custom\_jwt' 时，必须提供custom\_jwt\_authorizer配置。
- 当authorizer-type = 'api\_key' 时，必须提供key\_auth配置。

示例：

```
agentarts gateway create --name "my-gateway" --description "测试网关"
```

● 获取指定的网关详情

```
agentarts gateway get <gateway-id>
```

表 13-11 获取网关详情参数说明

| 参数           | 类型     | 是否必选 | 说明   |
|--------------|--------|------|------|
| <gateway-id> | string | 是    | 网关ID |

响应信息包含：

- gateway\_id：网关唯一标识符
- name：网关名称
- description：网关详细描述
- status：网关当前状态
- protocol\_type：网关协议类型（mcp）
- protocol\_configuration：网关协议的自定义设置
- authorizer\_type: 授权器类型
- agency\_name：委托名称
- authorizer\_configuration：授权器配置
- endpoint\_url：网关访问的URL端点
- log\_delivery\_configuration：日志投递配置
- workload\_identity：工作负载身份标识符
- agent\_gateway\_id：AgentGateway ID，关联底层 AgentGateway 实例
- outbound\_network\_configuration：出站网络配置

- tags: 资源标签列表
- created\_at: 网关创建时间戳
- updated\_at: 网关最后更新时间戳
- 更新指定的网关实例  
agentarts gateway update <gateway-id> [参数]

表 13-12 更新网关参数说明

| 参数                           | 类型          | 是否必选 | 说明             |
|------------------------------|-------------|------|----------------|
| <gateway-id>                 | string      | 是    | 网关ID           |
| --description                | string      | 否    | 网关描述           |
| --log-delivery-configuration | JSON string | 否    | 日志投递配置的JSON字符串 |
| --protocol-configuration     | JSON string | 否    | 协议配置（JSON格式）   |
| --tags                       | JSON string | 否    | 资源标签（JSON格式）   |

授权器配置要求：

- 当authorizer-type = 'custom\_jwt' 时，必须提供custom\_jwt\_authorizer配置
- 当authorizer-type = 'api\_key' 时，必须提供key\_auth配置

 说明

更新操作为增量更新，只传入需要更新的字段，未传入的字段将保持原值不变。更新网关时，至少填入一个参数。

示例：

```
agentarts gateway update "gateway-123" --description "更新后的描述"
```

- 删除指定的网关实例  
agentarts gateway delete <gateway-id>

表 13-13 删除网关参数说明

| 参数           | 类型     | 是否必选 | 说明   |
|--------------|--------|------|------|
| <gateway-id> | string | 是    | 网关ID |

响应信息包含删除操作的状态和相关操作日志信息。

- 获取网关实例列表  
# 列出所有网关  
agentarts gateway list  
  
# 按条件过滤  
agentarts gateway list [参数]

表 13-14 网关列表参数说明

| 参数                  | 类型      | 是否必选 | 默认值 | 说明                             |
|---------------------|---------|------|-----|--------------------------------|
| --name              | string  | 否    | -   | 按名称过滤网关，支持模糊匹配                 |
| --status            | string  | 否    | -   | 按状态过滤网关（如 "active"、"inactive"） |
| --gateway-id        | string  | 否    | -   | 按网关ID过滤，支持多个ID（最多20个），用逗号分隔    |
| --tag-key-exists    | string  | 否    | -   | 按标签键存在过滤（逗号分隔）                 |
| --tag-key-matches   | string  | 否    | -   | 按标签键值对过滤-键（逗号分隔）               |
| --tag-value-matches | string  | 否    | -   | 按标签键值对过滤-值（逗号分隔）               |
| --tag-match-policy  | string  | 否    | -   | 标签匹配模式（ALL/ANY）                |
| --offset            | integer | 否    | 0   | 分页偏移量，最小 0，最大 100000           |
| --limit             | integer | 否    | 50  | 每页大小，最小 1，最大 100               |

- 在指定的网关下创建新的Target实例  
agentarts gateway create-target <gateway-id> [参数]

表 13-15 创建 Target 参数说明

| 参数            | 类型     | 是否必选 | 默认值             | 说明       |
|---------------|--------|------|-----------------|----------|
| <gateway-id>  | string | 是    | -               | 网关ID     |
| --name        | string | 否    | target-{8位随机后缀} | Target名称 |
| --description | string | 否    | -               | Target描述 |

| 参数                                  | 类型          | 是否必选 | 默认值 | 说明                     |
|-------------------------------------|-------------|------|-----|------------------------|
| --target-configuration              | JSON string | 是    | -   | Target的配置，以JSON字符串形式传入 |
| --credential-provider-configuration | JSON string | 否    | -   | 凭证提供者配置的JSON字符串        |

凭证提供者配置要求：

- 如果未提供，credential\_provider\_type默认为GATEWAY\_IAM\_ROLE，表示使用网关IAM角色。
- 当credential\_provider\_type='none' 时，表示无认证。
- 当credential\_provider\_type='oauth' 时，表示使用OAuth 2.0，必须提供oauth\_credential\_provider。
- 当credential\_provider\_type='api\_key' 时，表示使用API密钥，必须提供api\_key\_credential\_provider。

示例：

```
agentarts gateway create-target "gateway-123" \ --target-configuration '{"mcp_server\":"{\"endpoint\":"\"https://example.com/mcp\\",\"server_type\":"\"sse\\\"}}'
```

● 获取Target详情

```
agentarts gateway get-target <gateway-id> <target-id>
```

功能说明：此命令用于获取指定网关下特定Target的详细信息，包括名称、描述、状态、端点、超时时间等所有相关配置信息。如果Target不存在，系统会返回错误提示。

表 13-16 获取 Target 详情参数说明

| 参数           | 类型     | 是否必选 | 默认值  | 说明             |
|--------------|--------|------|------|----------------|
| <gateway-id> | string | 是    | 网关ID | 网关id           |
| <target-id>  | string | 是    | 目标ID | 网关关联的Target id |

● 更新指定网关下的特定Target实例

```
agentarts gateway update-target <gateway-id> <target-id> [参数]
```

表 13-17 更新 Target 参数说明

| 参数           | 类型     | 是否必选 | 说明        |
|--------------|--------|------|-----------|
| <gateway-id> | string | 是    | 网关ID      |
| <target-id>  | string | 是    | Target ID |
| --name       | string | 否    | Target名称  |

| 参数                                  | 类型          | 是否必选 | 说明               |
|-------------------------------------|-------------|------|------------------|
| --description                       | string      | 否    | Target描述         |
| --target-configuration              | JSON string | 否    | Target的配置JSON字符串 |
| --credential-provider-configuration | JSON string | 否    | 凭证提供者配置的JSON字符串  |

示例：

```
agentarts gateway update-target "gateway-123" "target-456" --description "更新后的Target描述"
```

- 删除指定网关下的Target实例  
agentarts gateway delete-target <gateway-id> <target-id>

表 13-18 删除 Target 参数说明

| 参数           | 类型     | 是否必选 | 说明        |
|--------------|--------|------|-----------|
| <gateway-id> | string | 是    | 网关ID      |
| <target-id>  | string | 是    | Target ID |

- 查询指定网关下的所有Target实例列表  
agentarts gateway list-targets <gateway-id> [参数]

表 13-19 列出网关 Target 参数说明

| 参数           | 类型      | 是否必选 | 默认值 | 说明                   |
|--------------|---------|------|-----|----------------------|
| <gateway-id> | string  | 是    | -   | 网关ID                 |
| --offset     | integer | 否    | 0   | 分页偏移量，最小 0，最大 100000 |
| --limit      | integer | 否    | 50  | 每页大小，最小 1，最大 100     |

13.5 Runtime SDK

场景介绍

Runtime SDK是SDK的核心组成部分，提供Agent运行时所需的所有功能，包括请求处理、响应生成、上下文管理等。

## 原理优势

1. 开箱即用，一键部署
  - **零配置部署**：一行命令即可将本地Agent部署到云端Runtime，无需手动构建镜像、配置网络。
  - **CLI工具完备**：agentarts dev本地运行、agentarts launch一键部署、agentarts invoke本地运行。
  - **模板丰富**：内置多种Agent框架模板（Langchain/LangGraph、GoogleADK等），开箱即用。
2. 框架无关，极致兼容
  - **多框架支持**：原生支持Langchain/LangGraph、GoogleADK等主流框架。
  - **统一入口**：无论使用何种框架，都可以通过统一的SDK接口进行管理和部署。
  - **平滑迁移**：现有Agent代码只需少量适配即可接入。
3. 云端托管，Serverless体验
  - **无需运维**：云端自动托管运行环境，无需关心服务器、容器、网络配置。
  - **弹性伸缩**：根据流量自动扩缩容，从零到大规模并发自适应。
  - **按量付费**：仅在实际调用时计费，空闲时自动释放资源。
4. 企业级安全隔离
  - **会话隔离**：每个用户/会话独立运行环境，防止数据串扰。
  - **权限控制**：与IAM深度集成，支持精细化的访问控制。
  - **安全防护**：基于云原生的安全机制，保障Agent和数据安全。
5. 可观测性内置
  - **实时监控**：开箱即用的健康状态监控和指标展示。
  - **日志追踪**：自动收集和展示运行日志，快速定位问题。
  - **调用洞察**：完整的请求/响应日志和性能分析。
6. 丰富的生态集成
  - **多协议支持**：原生支持HTTP、WebSocket、MCP等协议。
  - **云服务集成**：深度集成华为云SWR（镜像仓库）、IAM（身份认证）、APM（应用监控）等服务。

## 核心特性

### 1. AgentArtsRuntimeApp类

#### - 初始化参数

```
app = AgentArtsRuntimeApp(
 debug=False, # 调试模式
 lifespan=None, # 生命周期管理
 middleware=None, # 中间件列表
 protocol="http", # 协议类型: "http" 或 "https"
)
```

#### - run() 方法

启动ASGI服务器：

```
app.run(host="0.0.0.0", port=8080)
```

参数说明：

host: 绑定地址，默认在Docker环境中为 0.0.0.0，本地为 127.0.0.1

port: 绑定端口，默认 8080

**\*\*kwargs:** 传递给uvicorn的其他参数

2. 装饰器

- @app.entrypoint

注册Agent主入口函数，处理 /invocations端点的请求。

基本用法

```
@app.entrypoint
def handler(payload: dict) -> dict:
 """同步处理函数"""
 return {"result": payload["message"].upper()}

@app.entrypoint
async def async_handler(payload: dict) -> dict:
 """异步处理函数"""
 result = await some_async_operation(payload)
 return result
```

使用请求上下文

```
@app.entrypoint
async def handler(payload: dict, context: RequestContext = None) -> dict:
 """带上上下文的处理函数"""
 session_id = context.session_id if context else None
 request_id = context.request_id if context else None

 return {
 "response": "OK",
 "session_id": session_id,
 "request_id": request_id,
 }
```

流式响应-返回生成器以实现流式输出（SSE格式）

```
@app.entrypoint
async def streaming_handler(payload: dict) -> AsyncGenerator:
 """流式响应处理函数"""
 message = payload.get("message", "")

 for char in message:
 await asyncio.sleep(0.1)
 yield {"chunk": char}

@app.entrypoint
def sync_streaming_handler(payload: dict) -> Generator:
 """同步流式响应"""
 for i in range(10):
 yield {"count": i}
```

- @app.ping

注册健康检查处理函数，处理 /ping端点的请求。

```
from agentarts.sdk.runtime.model import PingStatus

@app.ping
def health_check() -> PingStatus:
 """自定义健康检查"""
 if is_healthy():
 return PingStatus.HEALTHY
 else:
 return PingStatus.UNHEALTHY
```

表 13-20 PingStatus 状态值

| 状态           | 说明             |
|--------------|----------------|
| HEALTHY      | 服务健康，无正在执行的任务。 |
| HEALTHY_BUSY | 服务健康，有任务正在执行。  |



| 状态        | 说明     |
|-----------|--------|
| UNHEALTHY | 服务不健康。 |

```
强制设置状态
设置维护模式app.force_ping_status(PingStatus.UNHEALTHY) # 恢复正常
app.force_ping_status(None)
```

```
- @app.websocket

注册WebSocket处理函数，处理 /ws端点的连接。
from starlette.websockets import WebSocket

@app.websocket
async def ws_handler(websocket: WebSocket, context: RequestContext = None):
 """WebSocket 处理函数"""
 await websocket.accept()

 try:
 while True:
 data = await websocket.receive_json()
 response = await process_message(data)
 await websocket.send_json(response)
 except WebSocketDisconnect:
 print("Client disconnected")
```

```
WebSocket示例：聊天应用

@app.websocket
async def chat_handler(websocket: WebSocket, context: RequestContext = None):
 await websocket.accept()
 session_id = context.session_id if context else "default"

 try:
 while True:
 message = await websocket.receive_text()
 response = await agent.chat(session_id, message)
 await websocket.send_text(response)
 except WebSocketDisconnect:
 await agent.end_session(session_id)
```

```
- @app.async_task

注册异步后台任务，任务执行状态会被自动追踪。
@app.async_task
async def background_job(payload: dict):
 """后台异步任务"""
 await asyncio.sleep(10)
 result = await process_data(payload)
 return result

@app.entrypoint
async def handler(payload: dict, context: RequestContext = None):
 # 启动后台任务
 asyncio.create_task(background_job(payload))

 # 检查是否有运行中的任务
 if app.has_running_tasks():
 print("有后台任务正在执行")

 return {"status": "accepted"}
```

3. RequestContext
- RequestContext是请求数据的上下文，包含请求的元信息。

表 13-21 属性

| 属性         | 类型            | 说明      |
|------------|---------------|---------|
| request_id | Optional[str] | 请求唯一标识符 |
| session_id | Optional[str] | 会话标识符   |
| request    | Optional[Any] | 原始请求对象  |

- 使用示例

```
@app.entrypoint
async def handler(payload: dict, context: RequestContext = None):
 if context:
 print(f"Request ID: {context.request_id}")
 print(f"Session ID: {context.session_id}")

 # 访问原始请求对象
 if context.request:
 headers = context.request.headers
 client_host = context.request.client.host

 return {"status": "ok"}
```

4. AgentArtsRuntimeContext

- AgentArtsRuntimeContext是全局上下文管理器，基于Python的contextvars实现，支持异步安全的上下文访问。
- 核心特性
  - 协程安全：每个异步任务都有独立的上下文视图。
  - 全局访问：在调用栈的任何位置都可以访问上下文数据。
  - 无需实例化：直接使用类方法访问。
- 可用方法

| 方法                               | 说明            |
|----------------------------------|---------------|
| get_session_id()                 | 获取会话ID        |
| set_session_id(value)            | 设置会话ID        |
| get_request_id()                 | 获取请求ID        |
| set_request_id(value)            | 设置请求ID        |
| get_user_id()                    | 获取用户ID        |
| set_user_id(value)               | 设置用户ID        |
| get_workload_access_token()      | 获取工作负载访问令牌    |
| set_workload_access_token(value) | 设置工作负载访问令牌    |
| get_user_token()                 | 获取用户令牌        |
| set_user_token(value)            | 设置用户令牌        |
| get_oauth2_callback_url()        | 获取OAuth2回调URL |

| 方法                             | 说明            |
|--------------------------------|---------------|
| set_oauth2_callback_url(value) | 设置OAuth2回调URL |
| clear()                        | 清除所有上下文变量     |

- 使用示例

```
from agentarts.sdk.runtime.context import AgentArtsRuntimeContext

@app.entrypoint
async def handler(payload: dict, context: RequestContext = None):
 # 方式一：通过 context 参数获取
 session_id = context.session_id if context else None

 # 方式二：通过全局上下文获取
 session_id = AgentArtsRuntimeContext.get_session_id()
 request_id = AgentArtsRuntimeContext.get_request_id()
 user_id = AgentArtsRuntimeContext.get_user_id()

 # 在深层调用中使用
 result = await process_with_context()
 return {"session_id": session_id}

async def process_with_context():
 """在任意函数中访问上下文"""
 session_id = AgentArtsRuntimeContext.get_session_id()
 # 使用 session_id 进行业务处理
 return {"processed": True, "session": session_id}
```

13.6 CodeInterpreter SDK

场景介绍

支持CodeInterpreter生命周期管理和Session生命周期管理。

原理优势

用户空间已存在代码解释器时，SDK支持快速拉起代码解释器会话

```
api_key = "your_actual_api_key" # your api key
with code_session("your_region", "your_code_interpreter_name", api_key=api_key) as code_client:
 response = code_client.invoke(
 operate_type="your_operate_type",
 api_key=api_key,
 arguments={your_arguments}
)
```

前提条件

- 已开通AgentArts。
- 登录控制台获取AK/SK，和API\_KEY信息，详情请参考[如何获取华为云AK/SK](#)和[如何获取代码解释器的API\\_KEY](#)。
- 已安装Python，且版本不低于3.10。查看Python版本的命令示例：python --version。

操作步骤

步骤1 执行如下命令进行包安装。

```
安装agentarts-sdk包
pip install agentarts-sdk

安装langgraph包
pip install langgraph

安装langchain_openai包
pip install langchain-openai
```

步骤2 配置环境变量：

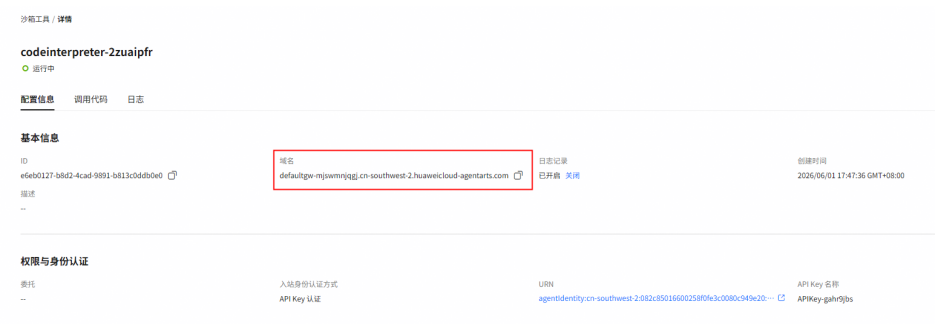
```
配置环境变量。
=== 用于通过代码创建沙箱 ===
export HUAWEICLOUD_SDK_AK="您的华为云AK"
export HUAWEICLOUD_SDK_SK="您的华为云SK"
export AGENTARTS_CONTROL_ENDPOINT="https://agentarts.cn-southwest-2.myhuaweicloud.com"
export AGENTARTS_CODEINTERPRETER_DATA_ENDPOINT="http://您的沙箱域名"
export HUAWEICLOUD_SDK_CODE_INTERPRETER_API_KEY="您的沙箱 API Key"
```

获取AK/SK请参考[如何获取华为云AK/SK](#)。

沙箱工具的域名、API Key获取方法如下：

- 1. 单击沙箱工具名称。
- 2. 进入“配置信息”页面后，获取域名。

图 13-2 获取沙箱域名



- 3. 单击“URN”链接，在弹出的窗口中获取沙箱的API Key。

图 13-3 URN 链接

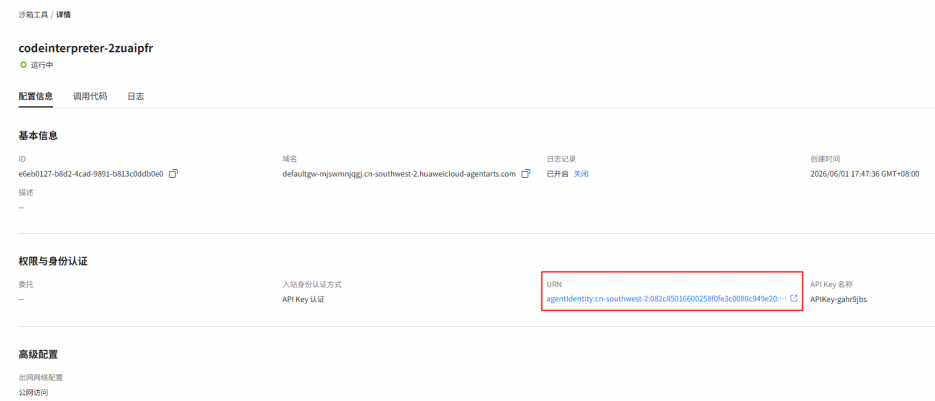
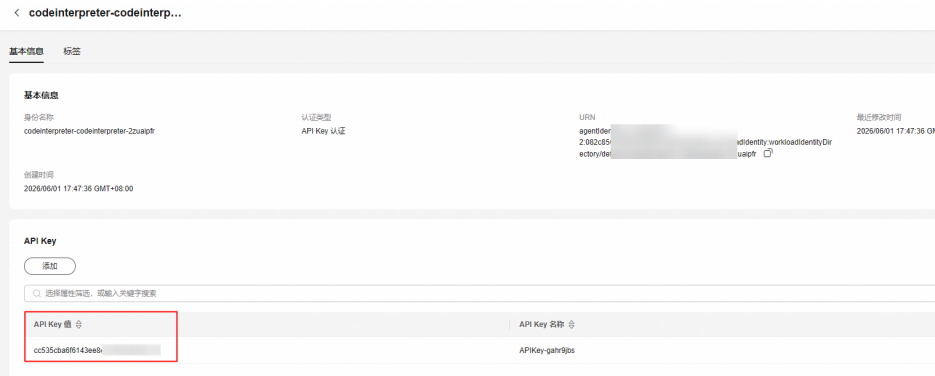


图 13-4 获取沙箱 API Key



步骤3 创建代码解释器。

```
from agentarts.sdk.tools import CodeInterpreter

auth_type为API_KEY时创建示例
client = CodeInterpreter(region="your_region",auth_type="API_KEY")
code_interpreter = client.create_code_interpreter(
 name="your_code_interpreter_name",
 api_key_name="your_api_key_name"
)

auth_type为IAM时创建示例
demo_client = CodeInterpreter(region="your_region", auth_type="IAM")
code_interpreter = client.create_code_interpreter(
 name="your_code_interpreter_name",
 auth_type="IAM"
)
```

步骤4 配置工具，这里以代码执行工具为例。

- API\_KEY的配置工具示例如下，用户自定义region信息和需要使用的代码解释器，代码解释器名称和API\_KEY一一对应。

```
import json

from agentarts.sdk.tools import code_session
from langchain_core.tools import tool

@tool
def execute_python_tool(code: str, description: str = "") -> str | None:
 """Execute Python Code in the sandbox"""
 api_key = "your_actual_api_key" # your api key
 with code_session("cn-southwest-2", "your_code_interpreter_name", api_key=api_key) as code_client:
 response = code_client.invoke(
 operate_type="execute_code",
 api_key=api_key,
 arguments={
 "code": code,
 "language": "python",
 "clear_context": False
 }
)
 return json.dumps(response["result"])
```

- IAM认证的配置工具示例：

```
import json

from agentarts.sdk.tools import code_session
from langchain_core.tools import tool

@tool
def execute_python_tool(code: str, description: str = "") -> str | None:
```

```
"""Execute Python Code in the sandbox"""
with code_session("cn-southwest-2", "your_code_interpreter_name", auth_type="IAM") as
code_client:
 response = code_client.invoke(
 operate_type="execute_code",
 arguments={
 "code": code,
 "language": "python",
 "clear_context": False
 }
)
 return json.dumps(response["result"])
```

**步骤5** 集成[步骤4](#)中配置的工具，以如下执行代码为例。

```
from langchain_openai import ChatOpenAI

创建Agent
llm = ChatOpenAI(
 model="deepseek-v4-flash",
 api_key=os.environ.get("MODEL_API_KEY", ""),
 base_url=os.environ.get("BASE_URL", ""),
 max_tokens=1000,
 temperature=0.7,
)
创建工具列表
tools = [execute_python_tool]
工具绑定Agent
llm = llm.bind_tools(tools)
```

----结束

通过SDK可调用的管理面方法请参考[表13-22](#)，详细使用指导请参考[工具使用方法指引](#)。

表 13-22 管理面方法

| 方法                      | 说明        |
|-------------------------|-----------|
| create_code_interpreter | 创建代码解释器   |
| list_code_interpreters  | 查询代码解释器列表 |
| update_code_interpreter | 更新代码解释器   |
| get_code_interpreter    | 获取代码解释器详情 |
| delete_code_interpreter | 删除代码解释器   |

通过SDK可调用的数据面方法请参考[表13-23](#)，详细使用指导请参考[工具使用方法指引](#)。

表 13-23 数据面方法

| 方法            | 说明          |
|---------------|-------------|
| start_session | 启动代码解释器会话   |
| get_session   | 获取代码解释器会话详情 |

| 方法               | 说明               |
|------------------|------------------|
| stop_session     | 停止当前代码解释器会话      |
| invoke           | 调用代码解释器会话        |
| execute_code     | 在代码解释器中执行代码      |
| execute_command  | 在代码解释器中执行命令      |
| upload_file      | 上传文件到代码解释器       |
| upload_files     | 上传多个文件到代码解释器     |
| download_file    | 从代码解释器下载文件       |
| download_files   | 从代码解释器下载多个文件     |
| install_packages | 在代码解释器中安装Python包 |
| clear_context    | 清除代码解释器上下文       |
| code_session     | 上下文管理器           |

## 13.7 Browser SDK

### 场景介绍

支持Browser生命周期管理、Browser Profile生命周期管理和Session生命周期管理。

### 原理优势

用户空间已存在浏览器时，SDK支持快速拉起浏览器会话。

```
api_key = "your_actual_api_key" # your api key
with browser_session("your_region", "your_browser_name", "your_session_name", api_key=api_key) as browser_client:
 response = browser_client.invoke(
 type="your_type",
 api_key=api_key,
 action={your_action}
)
```

### 前提条件

- 已开通AgentArts。
- 登录控制台获取AK/SK，和API\_KEY信息，详情请参考[如何获取华为云AK/SK](#)和[如何获取浏览器的API\\_KEY](#)。
- 已安装Python，且版本不低于3.10。查看Python版本的命令示例：python --version。

### 操作步骤

**步骤1** 执行如下命令进行包安装。

```
安装agentarts-sdk包
pip install agentarts-sdk
安装langgraph包
pip install langgraph
安装langchain_openai包
pip install langchain-openai
```

步骤2 配置环境变量：

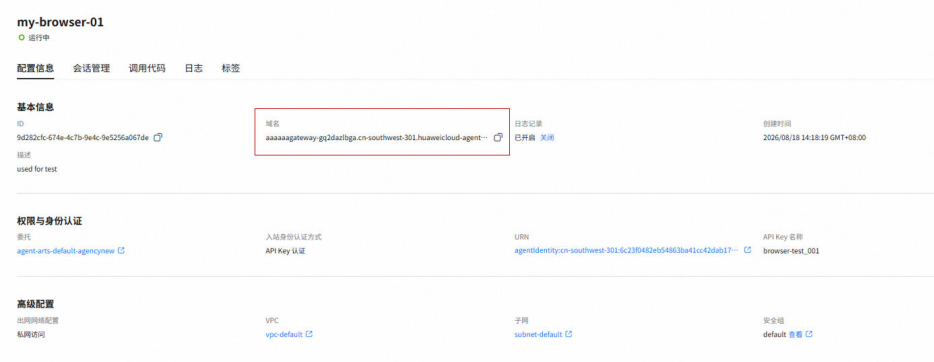
```
=== 用于通过代码创建沙箱 ===
export HUAWEICLOUD_SDK_AK="您的华为云AK"
export HUAWEICLOUD_SDK_SK="您的华为云SK"
export AGENTARTS_CONTROL_ENDPOINT="https://agentarts.cn-southwest-2.myhuaweicloud.com"
export AGENTARTS_BROWSER_DATA_ENDPOINT="http://您的沙箱域名"
export HUAWEICLOUD_SDK_BROWSER_API_KEY="您的沙箱 API Key"
```

获取AK/SK请参考[如何获取华为云AK/SK](#)。

沙箱工具的域名、API Key获取方法如下：

- 1. 单击沙箱工具名称。
- 2. 进入“配置信息”页面后，获取域名。

图 13-5 获取沙箱域名



- 3. 单击“URN”链接，在弹出的窗口中获取沙箱的API Key。

图 13-6 URN 链接

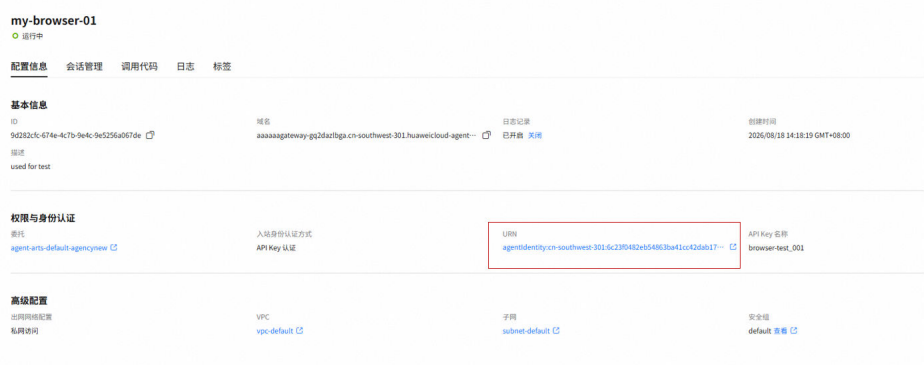
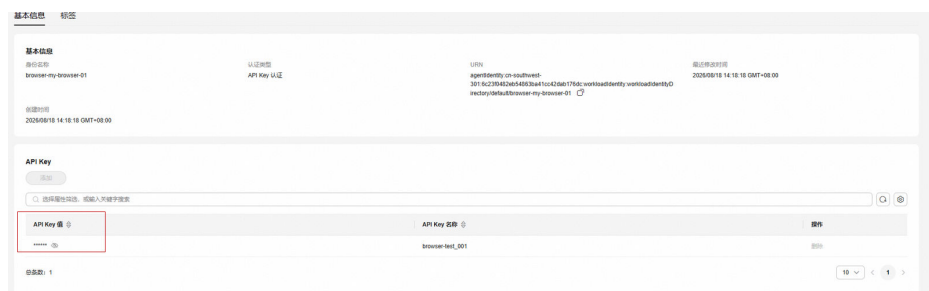




图 13-7 获取沙箱 API Key



### 步骤3 创建浏览器。

```
from agentarts.sdk.tools import Browser
auth_type为API_KEY时创建示例
client = Browser(region="your_region", auth_type="API_KEY")
browser = client.create_browser(
 name="your_browser_name",
 api_key_name="your_api_key_name"
)
auth_type为IAM时创建示例
client = Browser(region="your_region", auth_type="IAM")
browser = client.create_browser(
 name="your_browser_name",
 auth_type="IAM"
)
```

### 步骤4 配置工具，这里以导航工具为例。

- API\_KEY的配置工具示例如下，用户自定义region信息和需要使用的浏览器，浏览器名称和API\_KEY一一对应。

```
import json
from agentarts.sdk.tools import browser_session
from langchain_core.tools import tool
@tool
def navigate_tool(url: str) -> str | None:
 """Navigate to a URL in the browser"""
 api_key = "your_actual_api_key" # your api key
 with browser_session("cn-southwest-2", "your_browser_name", "your_session_name",
 api_key=api_key) as browser_client:
 response = browser_client.invoke(
 type="navigate",
 api_key=api_key,
 action={
 "url": url
 }
)
 return json.dumps(response)
```

- IAM认证的配置工具示例：

```
import json
from agentarts.sdk.tools import browser_session
from langchain_core.tools import tool
@tool
def navigate_tool(url: str) -> str | None:
 """Navigate to a URL in the browser"""
 with browser_session("cn-southwest-2", "your_browser_name", "your_session_name",
 auth_type="IAM") as browser_client:
 response = browser_client.invoke(
 type="navigate",
 action={
 "url": url
 }
)
 return json.dumps(response)
```

### 步骤5 集成配置的工具，以如下导航为例。

```
from langchain_openai import ChatOpenAI
创建Agent
llm = ChatOpenAI(
 model="deepseek-v4-flash",
 api_key=os.environ.get("MODEL_API_KEY", ""),
 base_url=os.environ.get("BASE_URL", ""),
 max_tokens=1000,
 temperature=0.7,
)
创建工具列表
tools = [navigate_tool]
工具绑定Agent
llm = llm.bind_tools(tools)
```

----结束

通过SDK可调用的管理面方法请参考[表13-24](#)，详细使用指导请参考[工具使用方法指引](#)。

表 13-24 管理面方法

| 方法                     | 说明        |
|------------------------|-----------|
| create_browser         | 创建浏览器     |
| list_browsers          | 查询浏览器列表   |
| update_browser         | 更新浏览器     |
| get_browser            | 获取浏览器详情   |
| delete_browser         | 删除浏览器     |
| create_browser_profile | 创建浏览器配置   |
| list_browser_profiles  | 查询浏览器配置列表 |
| get_browser_profile    | 获取浏览器配置详情 |
| delete_browser_profile | 删除浏览器配置   |

通过SDK可调用的数据面方法请参考[表13-25](#)，详细使用指导请参考[工具使用方法指引](#)。

表 13-25 数据面方法

| 方法            | 说明             |
|---------------|----------------|
| start_session | 启动浏览器会话        |
| get_session   | 获取浏览器会话详情      |
| stop_session  | 停止当前浏览器会话      |
| invoke        | 调用浏览器操作        |
| save_profile  | 保存当前浏览器会话状态到配置 |
| update_stream | 更新浏览器会话流状态     |

| 方法                      | 说明                        |
|-------------------------|---------------------------|
| take_control            | 获取浏览器人工控制权（禁用自动化）         |
| release_control         | 释放人工控制权（启用自动化）            |
| generate_automation_url | 生成自动化流的WebSocket URL和请求头  |
| generate_live_view_url  | 生成实时查看流的WebSocket URL和请求头 |
| mouse_click             | 鼠标点击                      |
| left_mouse_click        | 鼠标左键点击                    |
| right_mouse_click       | 鼠标右键点击                    |
| double_mouse_click      | 鼠标双击                      |
| mouse_move              | 鼠标移动                      |
| mouse_drag              | 鼠标拖拽                      |
| mouse_scroll            | 鼠标滚动                      |
| key_press               | 按键                        |
| key_type                | 输入文本                      |
| key_shortcut            | 按键组合                      |
| navigate                | 导航到指定URL                  |
| go_back                 | 返回上一页                     |
| go_forward              | 前进到下一页                    |
| refresh                 | 刷新当前页面                    |
| get_page_info           | 获取当前页面信息                  |
| screenshot              | 截图                        |
| wait                    | 等待                        |
| list_tabs               | 列出所有标签页                   |
| switch_tab              | 切换标签页                     |
| close_tab               | 关闭标签页                     |
| new_tab                 | 打开新标签页                    |
| browser_session         | 上下文管理器                    |

## 13.8 Memory SDK

### 场景介绍

Memory SDK提供两种使用模式：

- **Client模式**：功能完整的客户端，适合需要控制所有操作的应用场景。
- **Session模式**：基于绑定的会话进行记忆管理，适合特定用户/对话场景。

### 原理优势

- Client模式完整案例

```
#!/usr/bin/env python3
"""
Client模式完整使用案例
创建Space -> 创建Session -> 发送Message -> 查询Memory -> 搜索Memory
"""
import os
import time
from agentarts.sdk.memory import MemoryClient
from agentarts.sdk.memory.inner.config import TextMessage
确保环境变量已设置
assert os.getenv('HUAWEICLOUD_SDK_AK'), "请设置 HUAWEICLOUD_SDK_AK"
assert os.getenv('HUAWEICLOUD_SDK_SK'), "请设置 HUAWEICLOUD_SDK_SK"
def client_mode_example():
 """Client模式完整示例"""
 print("=== Client模式完整示例 ===")
 # 1. 创建Space
 with MemoryClient() as client:
 print("1. 创建测试Space")
 space = client.create_space(
 name=f"client_test_space_{int(time.time())}",
 message_ttl_hours=168,
 description="Client模式测试专用空间",
 memory_strategies_builtin=["semantic", "user-preference", "episodic"]
)
 space_id = space.id
 print(f"✓ Space创建成功: {space_id}")
 print(f"API Key ID: {space.api_key_id}")
 os.environ["HUAWEICLOUD_SDK_MEMORY_API_KEY"] = space.api_key
 # 2. 创建会话
 print("\n2. 创建会话")
 session_data = client.create_memory_session(
 space_id=space_id,
 actor_id="client-test-user",
 assistant_id="client-test-assistant"
)
 session_id = session_data.id
 print(f"✓ Session创建成功: {session_id}")
 # 3. 发送消息（使用TextMessage对象）
 print("\n3. 发送对话消息")
 messages = [
 TextMessage(
 role="user",
 content="你好，我想了解电商推荐算法，能够根据用户行为进行个性化推荐的算法有哪些？",
 actor_id="client-test-user"
),
 TextMessage(
 role="assistant",
 content="电商推荐算法主要包括：1) 协同过滤算法，基于用户行为相似性推荐；2) 基于内容的推荐，分析商品特征；3) 深度学习算法，能更准确捕捉用户偏好。推荐组合使用多种算法提升准确率。",
 actor_id="client-test-assistant"
)
]
```

```

),
 TextMessage(
 role="user",
 content="我对机器学习的监督学习特别感兴趣，深度学习和传统机器学习有什么区别？",
 actor_id="client-test-user"
)
]
add_result = client.add_messages(
 space_id=space_id,
 session_id=session_id,
 messages=messages
)
print(f"✓ 已添加 {len(add_result.items)} 条消息")
4. 等待记忆系统处理消息
print("\n4. 等待记忆系统处理消息...")
time.sleep(30)
5. 列出记忆
print("\n5. 查询记忆列表")
memories = client.list_memories(
 space_id=space_id,
 limit=10
)
print(f"✓ 发现 {len(memories.items)} 条记忆")
for i, memory in enumerate(memories.items[:3]):
 print(f" {i+1}. {memory.content[:50]}...")
 print(f" 策略: {memory.strategy_type}")
6. 搜索记忆 (使用MemorySearchFilter对象)
print("\n6. 搜索相关记忆")
from agentarts.sdk.memory.inner.config import MemorySearchFilter
search_results = client.search_memories(
 space_id=space_id,
 filters=MemorySearchFilter(query="机器学习", top_k=3)
)
print(f"✓ 找到 {len(search_results.results)} 条相关记忆")
for i, result in enumerate(search_results.results):
 score = result.get('score', 0)
 content = result.get('record', {}).get('content', "")[:60]
 print(f" {i+1}. [{score:.2f}] {content}...")
7. 获取特定记忆详情
if memories.items:
 print("\n7. 获取记忆详情")
 memory_id = memories.items[0].id
 memory_detail = client.get_memory(space_id, memory_id)
 print(f"✓ 记忆ID: {memory_detail.id}")
 print(f" 内容: {memory_detail.content[:80]}...")
 print(f" 策略: {memory_detail.strategy_type}")
 return space_id, session_id
if __name__ == "__main__":
 client_mode_example()

```

- **Session模式完整案例**

```

#!/usr/bin/env python3
"""
Session模式完整使用案例
创建Space -> 绑定会话 -> 发送Message -> 查询Memory -> 搜索Memory
"""
import os
import time
from agentarts.sdk.memory import MemoryClient
from agentarts.sdk.memory.session import MemorySession
from agentarts.sdk.memory.inner.config import TextMessage
确保环境变量已设置
assert os.getenv('HUAWEICLOUD_SDK_AK'), "请设置 HUAWEICLOUD_SDK_AK"
assert os.getenv('HUAWEICLOUD_SDK_SK'), "请设置 HUAWEICLOUD_SDK_SK"
def session_mode_example():
 """Session模式完整示例"""
 print("=== Session模式完整示例 ===")
 # 1. 创建Space
 with MemoryClient() as client:
 print("1. 创建测试Space")

```

```

space = client.create_space(
 name=f"session-test-space-{int(time.time())}",
 message_ttl_hours=168,
 description="Session模式测试专用空间",
 memory_strategies_builtin=["semantic", "user_preference", "episodic"]
)
space_id = space.id
print(f"✓ Space创建成功: {space_id}")
os.environ["HUAWEICLOUD_SDK_MEMORY_API_KEY"] = space.api_key
2. 创建并绑定会话
print("\n2. 创建并绑定会话")
session_obj = MemorySession(
 space_id=space_id,
 actor_id="session-test-user",
 assistant_id="session-test-assistant"
)
session_id = session_obj.session_id
print(f"✓ Session创建并绑定成功: {session_id}")
3. 发送对话消息（使用TextMessage对象）
print("\n3. 模拟用户与AI助手对话")
messages = [
 TextMessage(
 role="user",
 content="我是一个数据分析师，主要使用Python和SQL进行数据处理"
),
 TextMessage(
 role="assistant",
 content="数据分析师是很有前景的职业！Python方面，pandas、numpy、matplotlib是核心
库。"
),
 TextMessage(
 role="user",
 content="我最感兴趣的是可视化和机器学习方向，有什么推荐的学习路径吗？"
),
 TextMessage(
 role="assistant",
 content="可视化推荐学习matplotlib、seaborn、plotly；机器学习可以从scikit-learn入门，然后学习深度学习框架如TensorFlow或PyTorch。"
),
 TextMessage(
 role="user",
 content="我对Python数据分析很熟练，但机器学习经验还比较少，应该从哪里开始？"
)
]
add_result = session_obj.add_messages(messages)
print(f"✓ 已添加 {len(messages)} 条对话消息")
4. 等待记忆系统处理
print("\n4. 等待记忆系统生成...")
time.sleep(30)
5. 查询消息（在绑定会话上下文中）
print("\n5. 查询当前会话的消息")
messages_response = session_obj.list_messages(limit=10)
print(f"✓ 会话中有 {messages_response.total} 条消息")
6. 获取会话中的记忆列表
print("\n6. 查看生成的记忆")
memories = session_obj.list_memories(limit=10)
memory_list = memories.items
print(f"✓ 总共发现 {len(memory_list)} 条记忆")
for i, memory in enumerate(memory_list):
 content = memory.content[:50]
 strategy = memory.strategy_type or 'unknown'
 print(f" {i+1}. [{strategy}] {content}...")
7. 在会话上下文中搜索记忆（使用MemorySearchFilter对象）
print("\n7. 搜索Python相关记忆")
from agentarts.sdk.memory.inner.config import MemorySearchFilter
search_results = session_obj.search_memories(
 filters=MemorySearchFilter(query="Python", top_k=3)
)
print(f"✓ 找到 {len(search_results.results)} 条相关记忆")

```

```
for i, result in enumerate(search_results.results):
 score = result.get('score', 0)
 content = result.get('record', {}).get('content', '')[:60]
 print(f" {i+1}. [{score:.2f}] {content}...")
8. 获取特定记忆详情
if memory_list:
 print("\n8. 获取第一条记忆详情")
 memory_id = memory_list[0].id
 memory_detail = session_obj.get_memory(memory_id)
 print(f"✓ 记忆ID: {memory_detail.id}")
 print(f" 内容: {memory_detail.content[:80]}...")
 print(f" 策略: {memory_detail.strategy_type}")
 print(f" 来源策略: {memory_detail.strategy_id or 'N/A'}")
 return space_id, session_id
if __name__ == "__main__":
 session_mode_example()
```

## 前提条件

认证鉴权：

- 华为云访问密钥，用于调用管理面创建space接口认证。获取华为云凭证请参考[如何获取华为云AK/SK](#)。

```
export HUAWEICLOUD_SDK_AK="您的AK"
export HUAWEICLOUD_SDK_SK="您的SK"
```

- 创建space后，会返回apikey，可将其设置为环境变量，用于调用数据面接口认证。

```
export HUAWEICLOUD_SDK_MEMORY_API_KEY="您的数据面API密钥"
```

也可以在代码中传入。

```
from agentarts.sdk.memory import MemoryClient
```

```
client = MemoryClient(api_key="your-memory-api-key")
```

## 参数详解

认证信息

- 管理面（控制面）**：使用AK/SK认证，通过HUAWEICLOUD\_SDK\_AK和HUAWEICLOUD\_SDK\_SK环境变量。
- 数据面**：使用API\_KEY认证，SDK内部自动处理。

请求流程

- 使用AK/SK创建Space（控制面）。
- SDK自动创建API Key并返回api\_key（仅创建时可见）和api\_key\_id。
- SDK内部自动使用API Key进行数据面操作。
- 配置数据面访问端点（私网访问场景下，通过环境变量AGENTARTS\_MEMORY\_DATA\_ENDPOINT配置，例如https://memory-{space\_id}.cn-southwest-2.huaweicloud-agentarts.com）。

类型化返回值

SDK使用类型化对象作为返回值，便于代码提示和类型检查：

| 返回类型      | 说明      | 重要属性                                              |
|-----------|---------|---------------------------------------------------|
| SpaceInfo | Space信息 | id, name, api_key, api_key_id, status, created_at |

| 返回类型                 | 说明        | 重要属性                                                          |
|----------------------|-----------|---------------------------------------------------------------|
| SpaceListResponse    | Space列表   | items (SpaceInfo列表), total, limit, offset                     |
| SessionInfo          | Session信息 | id, space_id, actor_id, assistant_id, created_at              |
| MessageInfo          | 消息信息      | id, session_id, role, parts, actor_id, created_at             |
| MessageListResponse  | 消息列表      | items (MessageInfo列表), total, limit, offset                   |
| MessageBatchResponse | 批量消息响应    | items (MessageInfo列表)                                         |
| MemoryInfo           | 记忆信息      | id, space_id, content, strategy_type, strategy_id, created_at |
| MemoryListResponse   | 记忆列表      | items (MemoryInfo列表), total, limit, offset                    |
| MemorySearchResponse | 搜索结果      | results (列表, 含record和score), total, query                     |

类型化请求对象

SDK提供类型化的请求对象用于构建参数：

| 请求类型               | 用途                                            |
|--------------------|-----------------------------------------------|
| TextMessage        | 文本消息，参数：role, content, actor_id, assistant_id |
| ToolCallMessage    | 工具调用消息，参数：id, name, arguments                 |
| ToolResultMessage  | 工具结果消息，参数：tool_call_id, content, asset_ref    |
| MemorySearchFilter | 记忆搜索过滤，参数：query, top_k, min_score等            |
| MemoryListFilter   | 记忆列表过滤，参数：strategy_type, actor_id, sort_by等   |

记忆使用建议

- 始终检查环境变量是否正确设置。
- 根据场景选择Client或Session模式。
- 理解记忆生成的延迟时间。



- 合理设置Space的TTL时间。
- 及时清理测试数据。

## 13.9 Identity SDK

### 场景介绍

Identity SDK提供华为云身份认证服务的管理能力，支持工作负载身份、凭证提供者、访问令牌等管理功能。

### 原理优势

IdentityClient是身份认证服务的高级客户端。

**文件位置：**src/agentarts/sdk/services/identity/identity\_client.py

```
from agentarts.sdk.service.identity import IdentityClient
client = IdentityClient(region="cn-southwest-2")
```

### 操作步骤

#### 步骤1 创建工作负载身份。

```
workload = client.create_workload_identity(
 name="my-workload",
)
print(f"URN: {workload.urn}")
```

#### 步骤2 创建访问令牌。

```
使用用户Token创建
token = client.create_workload_access_token(
 workload_name="my-workload",
 user_token="user-oauth-token"
)
使用用户ID创建
token = client.create_workload_access_token_for_user_id(
 workload_name="my-workload",
 user_id="user-123"
)
print(f"Access Token: {token}")
```

#### 步骤3 配置凭证提供者管理。

- API Key凭证提供者

```
provider = client.create_api_key_credential_provider(
 name="my-api-key-provider",
 api_key="your-api-key"
)
print(f"Provider URN: {provider.urn}")
```

- OAuth2凭证提供者

```
GitHub OAuth2
provider = client.create_oauth2_credential_provider(
 name="github-provider",
 vendor="GithubOAuth2",
 client_id="your-client-id",
 client_secret="your-client-secret"
)
Google OAuth2
provider = client.create_oauth2_credential_provider(
 name="google-provider",
 vendor="GoogleOAuth2",
 client_id="your-client-id",
```

```
 client_secret="your-client-secret"
)
 # Microsoft OAuth2
 provider = client.create_oauth2_credential_provider(
 name="microsoft-provider",
 vendor="MicrosoftOAuth2",
 client_id="your-client-id",
 client_secret="your-client-secret"
)
```

- STS凭证提供者

```
provider = client.create_sts_credential_provider(
 name="my-sts-provider",
 agency_urn="agency-urn:your-agency",
 tags=[{"key": "env", "value": "prod"}]
)
```

#### 步骤4 使用客户端进行凭证获取。

- 获取OAuth2 Token

```
token = client.get_resource_oauth2_token(
 provider_name="github-provider",
 scopes=["user:email", "read:user"],
 workload_access_token="agent-identity-token"
)
print(f"Token: {token}")
```

- 获取API Key

```
api_key = client.get_resource_api_key(
 provider_name="my-api-key-provider",
 workload_access_token="workload-token"
)
print(f"API Key: {api_key}")
```

- 获取STS Token

```
sts_token = client.get_resource_sts_token(
 provider_name="my-sts-provider",
 workload_access_token="workload-token",
 agency_session_name="session-name"
)
print(f"Access Key: {sts_token.access_key_id}")
print(f"Secret Key: {sts_token.secret_access_key}")
print(f"Security Token: {sts_token.security_token}")
```

#### 步骤5 推荐使用装饰器来处理凭据生命周期。它们会自动将令牌或密钥注入到您的函数参数中。

```
from typing import Optional
from agentarts.sdk import require_access_token, require_api_key, require_sts_token
from agentarts.sdk.identity.types import StsCredentials

OAuth2 Access Token 注入 (异步)
@require_access_token(
 provider_name="google",
 scopes=["https://www.googleapis.com/auth/userinfo.email"],
 auth_flow="USER_FEDERATION"
)
async def fetch_google_data(access_token: Optional[str] = None):
 print(f"Using OAuth2 token: {access_token}")

M2M Access Token 注入 (异步)
@require_access_token(
 provider_name="my-company-api",
 auth_flow="M2M"
)
async def call_internal_service(access_token: Optional[str] = None):
 print(f"Using M2M OAuth2 token: {access_token}")

API Key 注入 (同步)
@require_api_key(provider_name="openai")
def call_llm(api_key: Optional[str] = None):
```

```
print(f"Using API Key: {api_key}")

STS Token 注入
@require_sts_token(provider_name="huaweicloud-iam", agency_session_name="example-session")
async def access_huawei_resource(sts_credentials: Optional[StsCredentials] = None):
 # 直接访问 SDK 对象字段
 print(f"AK: {sts_credentials.access_key_id}")
 print(f"SK: {sts_credentials.secret_access_key}")
 print(f"SecurityToken: {sts_credentials.security_token}")
```

----结束

## 13.10 Gateway SDK

### 场景介绍

Gateway SDK提供管理网关及相关操作的命令行工具，支持网关和Target的创建、查询、更新、删除等功能。

### 前提条件

- 已开通AgentArts。
- 登录控制台获取AK/SK，详情请参考[认证鉴权](#)。

```
export HUAWEICLOUD_SDK_AK="your-access-key"
export HUAWEICLOUD_SDK_SK="your-secret-key"
```
- 已安装Python，且版本不低于3.10。查看Python版本的命令示例：

```
python --version
```

### 网关管理

- 创建网关**

```
def create_gateway(
 self,
 name: Optional[str] = None,
 description: Optional[str] = None,
 protocol_type: Optional[str] = "mcp",
 authorizer_type: Optional[str] = "iam",
 agency_name: Optional[str] = None,
 protocol_configuration: dict[str, Any] | None = None,
 authorizer_configuration: Optional[CoreGatewayAuthorizerConfiguration] = None,
 log_delivery_configuration: Optional[CoreGatewayLogDeliveryConfigurationRequestBody] = None,
 outbound_network_configuration: Optional[CoreGatewayOutboundNetworkConfiguration] = None,
 tags: list[dict[str, str]] | None = None
) -> RequestResult
```

表 13-26 创建网关参数说明

| 参数            | 类型            | 是否必选 | 默认值                | 说明      |
|---------------|---------------|------|--------------------|---------|
| name          | Optional[str] | 否    | gateway-{8个随机字符后缀} | 网关名称    |
| description   | Optional[str] | 否    | -                  | 网关的详细描述 |
| protocol_type | Optional[str] | 否    | mcp                | 网关协议类型  |

| 参数                             | 类型                                                       | 是否必选 | 默认值                  | 说明                                               |
|--------------------------------|----------------------------------------------------------|------|----------------------|--------------------------------------------------|
| authorizer_type                | Optional[str]                                            | 否    | iam                  | 授权器类型，<br>可选值：<br>custom_jwt<br>、iam、<br>api_key |
| agency_name                    | Optional[str]                                            | 否    | AgentArtsCoreGateway | 网关委托身份的委托名称                                      |
| protocol_configuration         | Optional[CoreGatewayProtocolConfiguration]               | 否    | -                    | 网关协议的自定义设置                                       |
| authorizer_configuration       | Optional[CoreGatewayAuthorizerConfiguration]             | 否    | -                    | 授权器配置                                            |
| log_delivery_configuration     | Optional[CoreGatewayLogDeliveryConfigurationRequestBody] | 否    | -                    | 日志投递配置                                           |
| outbound_network_configuration | Optional[CoreGatewayOutboundNetworkConfiguration]        | 否    | -                    | 出站网络配置                                           |
| tags                           | Optional[CoreGatewayTag]                                 | 否    | -                    | 资源标签列表                                           |

授权器配置要求：

- 当authorizer\_type = 'custom\_jwt' 时，必须提供custom\_jwt\_authorizer配置。
- 当authorizer\_type = 'api\_key' 时，必须提供key\_auth配置。

- 查询网关信息

```
def get_gateway(self, gateway_id: str) -> ShowCoreGatewayResponse
```

表 13-27 查询网关参数说明

| 参数         | 类型  | 是否必选 | 说明                  |
|------------|-----|------|---------------------|
| gateway_id | str | 是    | 要查询的 Gateway 的唯一标识符 |

- 更新指定的网关信息

```
def update_gateway(
 self,
 gateway_id: str,
 description: Optional[str] = None,
 protocol_configuration: Optional[CoreGatewayProtocolConfiguration] = None,
 tags: list[dict[str, str]] | None = None,
 log_delivery_configuration: Optional[CoreGatewayLogDeliveryConfigurationRequestBody] = None,
) -> RequestResult
```

表 13-28 更新网关参数说明

| 参数                         | 类型                                                       | 是否必选 | 说明                  |
|----------------------------|----------------------------------------------------------|------|---------------------|
| gateway_id                 | str                                                      | 是    | 要更新的 Gateway 的唯一标识符 |
| description                | Optional[str]                                            | 否    | 网关的详细描述             |
| protocol_configuration     | Optional[CoreGatewayProtocolConfiguration]               | 否    | 网关协议的自定义设置          |
| log_delivery_configuration | Optional[CoreGatewayLogDeliveryConfigurationRequestBody] | 否    | 日志投递配置              |

说明

更新操作为增量更新，只传入需要更新的字段，未传入的字段将保持原值不变。

- 删除指定的网关信息

```
def delete_gateway(self, gateway_id: str) -> DeleteCoreGatewayResponse
```

表 13-29 更新网关参数说明

| 参数         | 类型  | 是否必选 | 说明                  |
|------------|-----|------|---------------------|
| gateway_id | str | 是    | 要删除的 Gateway 的唯一标识符 |

- 获取网关列表

```
def list_gateways(
 self,
 name: Optional[str] = None,
 status: Optional[str] = None,
 gateway_id: Optional[str] = None,
 tag_key_exists: list[str] | None = None,
 tag_key_matches: list[str] | None = None,
 tag_value_matches: list[str] | None = None,
 tag_match_policy: str | None = None,
 limit: Optional[int] = None,
 offset: Optional[int] = None) -> ListCoreGatewaysResponse
```

表 13-30 获取网关实例列表参数说明

| 参数                | 类型            | 是否必选 | 默认值 | 说明                          |
|-------------------|---------------|------|-----|-----------------------------|
| name              | Optional[str] | 否    | -   | 按名称过滤网关，支持模糊匹配              |
| status            | Optional[str] | 否    | -   | 按状态过滤网关                     |
| gateway_id        | Optional[str] | 否    | -   | 按网关ID过滤，支持多个ID（最多20个），用逗号分隔 |
| tag-key-exists    | list[str]     | 否    | -   | 按标签键存在过滤（逗号分隔）              |
| tag-key-matches   | list[str]     | 否    | -   | 按标签键值对过滤-键（逗号分隔）            |
| tag-value-matches | list[str]     | 否    | -   | 按标签键值对过滤-值（逗号分隔）            |
| tag-match-policy  | list[str]     | 否    | -   | 标签匹配模式（ALL/ANY）             |
| limit             | Optional[int] | 否    | 50  | 返回结果的最大数量                   |
| offset            | Optional[int] | 否    | 0   | 返回结果的偏移量                    |

- 在指定的网关下创建新的Target实例

```
def create_gateway_target(
 self,
 gateway_id: str,
 name: Optional[str] = None,
 description: Optional[str] = None,
 target_configuration: Optional[CoreGatewayTargetConfiguration] = None,
 credential_provider_configuration: Optional[CoreGatewayCredentialProviderConfiguration] = None)
-> CreateCoreGatewayTargetResponse
```

表 13-31 创建 Target 参数说明

| 参数         | 类型            | 是否必选 | 默认值               | 说明         |
|------------|---------------|------|-------------------|------------|
| gateway_id | str           | 是    | -                 | 网关ID       |
| name       | Optional[str] | 否    | target-{8个随机字符后缀} | Target服务名称 |

| 参数                                | 类型                                                   | 是否必选 | 默认值 | 说明                          |
|-----------------------------------|------------------------------------------------------|------|-----|-----------------------------|
| description                       | Optional[str]                                        | 否    | -   | Target服务描述                  |
| target_configuration              | Optional[CoreGatewayTargetConfiguration]             | 是    | -   | Target服务配置，mcp_server内容必须提供 |
| credential_provider_configuration | Optional[CoreGatewayCredentialProviderConfiguration] | 否    | -   | 凭证提供者配置                     |

凭证提供者配置要求：

- 如果不提供，credential\_provider\_type默认为none，表示无认证。
- 当credential\_provider\_type='oauth' 时，必须提供oauth\_credential\_provider。
- 当credential\_provider\_type='api\_key' 时，必须提供api\_key\_credential\_provider。

- 查询指定Gateway下的特定Target实例的详细信息

```
def get_gateway_target(self, gateway_id: str, target_id: str) -> ShowCoreGatewayTargetResponse
```

表 13-32 查询 Target 信息参数说明

| 参数         | 类型  | 是否必选 | 说明                    |
|------------|-----|------|-----------------------|
| gateway_id | str | 是    | Target所属Gateway的唯一标识符 |
| target_id  | str | 是    | 要查询的Target的唯一标识符      |

- 更新指定网关下的特定Target实例

```
def update_gateway_target(
 self,
 gateway_id: str,
 target_id: str,
 name: Optional[str] = None,
 description: Optional[str] = None,
 target_configuration: Optional[CoreGatewayTargetConfiguration] = None,
 tags: list[dict[str, str]] | None = None,
 credential_provider_configuration: Optional[CoreGatewayCredentialProviderConfiguration] = None)
-> UpdateCoreGatewayTargetResponse
```

表 13-33 更新 Target 参数信息

| 参数                                | 类型                                                   | 是否必选 | 说明                     |
|-----------------------------------|------------------------------------------------------|------|------------------------|
| gateway_id                        | str                                                  | 是    | Target所属 Gateway的唯一标识符 |
| target_id                         | str                                                  | 是    | 要更新的Target的唯一标识符       |
| name                              | Optional[str]                                        | 否    | Target服务名称             |
| description                       | Optional[str]                                        | 否    | Target服务描述             |
| target_configuration              | Optional[CoreGatewayTargetConfiguration]             | 否    | Target服务配置             |
| credential_provider_configuration | Optional[CoreGatewayCredentialProviderConfiguration] | 否    | 凭证提供者配置                |

- 删除指定网关下的特定Target实例  
def delete\_gateway\_target(self, gateway\_id: str, target\_id: str) -> DeleteCoreGatewayTargetResponse

表 13-34 删除 Target 参数说明

| 参数         | 类型  | 是否必选 | 说明                     |
|------------|-----|------|------------------------|
| gateway_id | str | 是    | Target所属 Gateway的唯一标识符 |
| target_id  | str | 是    | 要删除的Target的唯一标识符       |

- 查询指定网关下的所有Target列表  
def list\_gateway\_targets(  
    self,  
    gateway\_id: str,  
    limit: Optional[int] = None,  
    offset: Optional[int] = None) -> ListCoreGatewayTargetsResponse

表 13-35 查询指定网关下的 Target 列表参数说明

| 参数         | 类型  | 是否必选 | 默认值 | 说明                         |
|------------|-----|------|-----|----------------------------|
| gateway_id | str | 是    | -   | 要查询 Targets的 Gateway的唯一标识符 |



| 参数     | 类型            | 是否必选 | 默认值 | 说明          |
|--------|---------------|------|-----|-------------|
| limit  | Optional[int] | 否    | 50  | 每页返回的最大结果数量 |
| offset | Optional[int] | 否    | 0   | 返回结果的偏移量    |

参数说明

- 请求参数

表 13-36 CoreGatewayProtocolConfiguration

| 参数  | 类型                                  | 说明         |
|-----|-------------------------------------|------------|
| mcp | CoreGatewayMcpProtocolConfiguration | 网关协议的自定义设置 |

表 13-37 CoreGatewayMcpProtocolConfiguration

| 参数                    | 类型                              | 说明                                                                          |
|-----------------------|---------------------------------|-----------------------------------------------------------------------------|
| search_configuration  | CoreGatewaySearchConfiguration  | 搜索操作配置                                                                      |
| session_configuration | CoreGatewaySessionConfiguration | 会话管理配置                                                                      |
| supported_versions    | Array                           | MCP网关支持的协议版本列表，支持选择的MCP版本：2025-03-26、2025-06-18、2025-11-25。默认值：[2025-03-26] |

表 13-38 CoreGatewaySearchConfiguration

| 参数          | 类型                             | 说明                                              |
|-------------|--------------------------------|-------------------------------------------------|
| search_type | CoreGatewaySearchConfiguration | 搜索类型，该字段用于指定网关如何处理搜索操作。                         |
| top_n       | integer                        | 当search_type为semantic时，该字段可选。用于指定语义检索场景返回的工具数量。 |

| 参数              | 类型     | 说明                                                       |
|-----------------|--------|----------------------------------------------------------|
| score_threshold | string | 当search_type为semantic时，该字段可选。语义检索场景，只有工具的相似得分不低于该值时才会返回。 |

表 13-39 CoreGatewaySessionConfiguration

| 参数      | 类型   | 说明             |
|---------|------|----------------|
| enabled | bool | 是否开启会话管理，默认关闭。 |

表 13-40 CoreGatewayAuthorizerConfiguration

| 参数                    | 类型                                          | 说明                                             |
|-----------------------|---------------------------------------------|------------------------------------------------|
| custom_jwt_authorizer | CoreGatewayCustomJWTAuthorizerConfiguration | 搜索类型，该字段用于指定网关如何处理搜索操作。                        |
| key_auth              | CoreGatewayKeyAuthAuthorizerConfiguration   | 当search_type为semantic时，该字段可选。用于指定语义检索场景返回的工具数量 |

表 13-41 CoreGatewayCustomJWTAuthorizerConfiguration

| 参数               | 类型     | 说明                              |
|------------------|--------|---------------------------------|
| discovery_url    | String | JWT 发现 URL。                     |
| allowed_audience | Array  | 允许的 JWT 受众（aud）列表，用于验证令牌的预期接收者。 |
| allowed_clients  | Array  | 允许的客户端 ID 列表，用于限制哪些客户端可以访问。     |
| allowed_scopes   | Array  | 允许的 JWT 作用域（scope）列表，用于控制访问权限   |
| custom_claims    | Array  | 自定义声明验证规则列表，用于验证 JWT 中的自定义声明。   |

表 13-42 CoreGatewayKeyAuthAuthorizerConfiguration

| 参数       | 类型    | 说明        |
|----------|-------|-----------|
| api_keys | Array | API 密钥列表。 |

表 13-43 CoreGatewayLogDeliveryConfigurationRequestBody

| 参数      | 类型      | 说明        |
|---------|---------|-----------|
| enabled | Boolean | 是否启用日志上报。 |

表 13-44 CoreGatewayOutboundNetworkConfiguration

| 参数           | 类型                   | 说明                         |
|--------------|----------------------|----------------------------|
| network_mode | String               | 网络模式。取值范围：<br>public   vpc |
| vpc_config   | CoreGatewayVpcConfig | VPC网络配置。                   |

表 13-45 CoreGatewayVpcConfig

| 参数                 | 类型     | 说明                |
|--------------------|--------|-------------------|
| vpc_id             | String | VPC ID，VPC网络的唯一标识 |
| subnet_id          | String | 子网ID，VPC子网的唯一标识。  |
| security_group_ids | Array  | 安全组ID列表。          |

表 13-46 CoreGatewayTag

| 参数    | 类型     | 说明   |
|-------|--------|------|
| key   | String | 标签键。 |
| value | String | 标签值。 |

表 13-47 CoreGatewayTargetConfiguration

| 参数                    | 类型                                                | 说明            |
|-----------------------|---------------------------------------------------|---------------|
| mcp_server            | CoreGatewayMcpServerTargetConfiguration           | MCP Server 配置 |
| openapi               | CoreGatewayOpenApiTargetConfiguration             | OpenAPI 配置    |
| dedicated_gateway_api | CoreGatewayDedicatedGatewayApiTargetConfiguration | 专享网关API       |
| huawei_cloud_api      | CoreGatewayHuaweiCloudApiTargetConfiguration      | 华为云API        |

表 13-48 CoreGatewayMcpServerTargetConfiguration

| 参数          | 类型     | 说明                                                                                                                                 |
|-------------|--------|------------------------------------------------------------------------------------------------------------------------------------|
| endpoint    | String | MCP服务器端点URL                                                                                                                        |
| server_type | String | MCP服务器类型，取值范围： <ul style="list-style-type: none"><li>sse: 使用Server-Sent Events长连接</li><li>streamable_http: 使用可流式 HTTP 请求</li></ul> |

表 13-49 CoreGatewayOpenApiTargetConfiguration

| 参数      | 类型                          | 说明                                    |
|---------|-----------------------------|---------------------------------------|
| payload | String                      | OpenAPI 规范目标配置，支持内联文档或从OBS获取规范文档      |
| obs     | CoreGatewayObsConfiguration | OBS（对象存储服务）配置，用于从 OBS 获取 OpenAPI 规范文档 |

表 13-50 CoreGatewayDedicatedGatewayApiTargetConfiguration

| 参数      | 类型                          | 说明                                    |
|---------|-----------------------------|---------------------------------------|
| payload | String                      | OpenAPI 规范目标配置，支持内联文档或从 OBS 获取规范文档    |
| obs     | CoreGatewayObsConfiguration | OBS（对象存储服务）配置，用于从 OBS 获取 OpenAPI 规范文档 |

表 13-51 CoreGatewayHuaweiCloudApiTargetConfiguration

| 参数      | 类型     | 说明                                  |
|---------|--------|-------------------------------------|
| payload | String | OpenAPI 规范文档内容（JSON 或 YAML 格式的内联内容） |

表 13-52 CoreGatewayObsConfiguration

| 参数          | 类型     | 说明                                    |
|-------------|--------|---------------------------------------|
| bucket_name | String | OBS 桶名称                               |
| object_key  | String | OBS 对象键名（文件路径）：示例：specs/petstore.yaml |

表 13-53 CoreGatewayCredentialProviderConfiguration

| 参数                       | 类型                            | 说明                                                                                                                                                                                                                                                                                                                           |
|--------------------------|-------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| credential_provider_type | String                        | 凭证提供者类型；<br>取值范围： <ul style="list-style-type: none"><li>• iam：使用网关 IAM 角色，当target_type取值为openapi时，不支持</li><li>• oauth：使用 OAuth 2.0，当target_type取值为dedicated_gateway_api以及huawei_cloud_api时，不支持</li><li>• api_key：使用 API 密钥，当target_type取值为dedicated_gateway_api以及huawei_cloud_api时，不支持</li><li>• none：无认证 默认取值：不涉及</li></ul> |
| credential_provider      | CoreGatewayCredentialProvider | 凭证提供者配置                                                                                                                                                                                                                                                                                                                      |

表 13-54 CoreGatewayCredentialProvider

| 参数                          | 类型                                  | 说明             |
|-----------------------------|-------------------------------------|----------------|
| api_key_credential_provider | CoreGatewayApiKeyCredentialProvider | API KEY凭证提供者配置 |
| oauth_credential_provider   | CoreGatewayOAuthCredentialProvider  | OAuth凭证提供者配置   |
| iam_credential_provider     | CoreGatewayIamCredentialProvider    | Iam凭证提供者配置     |

表 13-55 CoreGatewayApiKeyCredentialProvider

| 参数            | 类型     | 说明      |
|---------------|--------|---------|
| provider_name | String | 凭证提供者名称 |

| 参数                        | 类型     | 说明                                                                                                                      |
|---------------------------|--------|-------------------------------------------------------------------------------------------------------------------------|
| credential_location       | String | 凭证传递位置；<br>取值范围： <ul style="list-style-type: none"><li>header：通过HTTP请求头传递API密钥</li><li>query：通过URL查询参数传递API密钥</li></ul> |
| credential_parameter_name | String | 凭证参数名称                                                                                                                  |
| credential_prefix         | String | 凭证前缀（如 "Bearer"）                                                                                                        |

表 13-56 CoreGatewayOAuthCredentialProvider

| 参数                 | 类型     | 说明                                                                                                                                           |
|--------------------|--------|----------------------------------------------------------------------------------------------------------------------------------------------|
| provider_name      | String | 凭证提供者名称                                                                                                                                      |
| grant_type         | String | OAuth 授权类型；<br>取值范围： <ul style="list-style-type: none"><li>client_credentials：适用于服务器到服务器通信</li><li>authorization_code：适用于需要用户授权的应用</li></ul> |
| scopes             | Array  | OAuth作用域列表                                                                                                                                   |
| default_return_url | String | 默认返回URL                                                                                                                                      |
| custom_parameters  | Object | 自定义参数，键值对形式。                                                                                                                                 |

表 13-57 CoreGatewayIamCredentialProvider

| 参数            | 类型     | 说明      |
|---------------|--------|---------|
| provider_name | String | 凭证提供者名称 |

- 响应参数

表 13-58 ListCoreGatewaysResponse

| 字段       | 类型                       | 说明         |
|----------|--------------------------|------------|
| gateways | List[CoreGatewaySummary] | 网关详情列表     |
| size     | Integer                  | 当前页返回的网关数量 |
| total    | Integer                  | 网关总数量      |

表 13-59 CoreGatewaySummary

| 字段              | 类型     | 说明                                                                                                              |
|-----------------|--------|-----------------------------------------------------------------------------------------------------------------|
| gateway_id      | String | 网关的唯一标识符                                                                                                        |
| name            | String | Target名称                                                                                                        |
| status          | String | Target状态。取值范围：<br>active: 运行中   inactive: 已停用   creating: 创建中   failed: 创建或运行失败   deleting: 删除中   updating: 更新中 |
| inactive_reason | String | Target离线状态原因。                                                                                                   |
| protocol_type   | String | 网关协议类型。                                                                                                         |
| authorizer_type | String | 授权器类型。                                                                                                          |
| agency_name     | string | 委托名称，用于指定网关使用的委托身份。                                                                                             |
| tags            | Array  | 资源标签列表。取值范围：数组长度为 0-20。                                                                                         |
| created_at      | String | 创建时间戳                                                                                                           |
| updated_at      | String | 更新时间戳                                                                                                           |

表 13-60 ShowCoreGatewayResponse

| 字段          | 类型     | 说明                           |
|-------------|--------|------------------------------|
| gateway_id  | String | 网关的唯一标识符                     |
| name        | String | 网关名称。取值范围：长度为 2-40 个字符，      |
| description | String | 网关的详细描述。取值范围：长度为 1-1000 个字符。 |



| 字段                             | 类型     | 说明                                                                                                          |
|--------------------------------|--------|-------------------------------------------------------------------------------------------------------------|
| status                         | String | Target状态。取值范围：active: 运行中   inactive: 已停用   creating: 创建中   failed: 创建或运行失败   deleting: 删除中   updating: 更新中 |
| inactive_reason                | String | Target离线状态原因。                                                                                               |
| protocol_type                  | String | 网关协议类型。                                                                                                     |
| protocol_configuration         | Object | 网关协议的自定义设置                                                                                                  |
| authorizer_type                | String | 授权器类型。                                                                                                      |
| agency_name                    | String | 委托名称，用于指定网关使用的委托身份。                                                                                         |
| endpoint_url                   | String | 访问网关的 URL 端点。                                                                                               |
| log_delivery_configuration     | Object | 日志上报配置。                                                                                                     |
| workload_identity              | Object | 工作负载身份标识，用于代理身份认证。                                                                                          |
| agent_gateway_id               | String | AgentGateway ID，关联底层 AgentGateway 实例。                                                                       |
| outbound_network_configuration | Object | 网络模式。                                                                                                       |
| tags                           | Array  | 资源标签列表。取值范围：数组长度为 0-20。                                                                                     |
| created_at                     | String | 创建时间戳                                                                                                       |
| updated_at                     | String | 更新时间戳                                                                                                       |

表 13-61 ListCoreGatewayTargetsResponse

| 字段      | 类型                             | 说明         |
|---------|--------------------------------|------------|
| targets | List[CoreGatewayTargetSummary] | 网关详情列表     |
| size    | Integer                        | 当前页返回的网关数量 |
| total   | Integer                        | 网关总数量      |

表 13-62 CoreGatewayTargetSummary

| 字段              | 类型     | 说明                                                                                                                                                                                               |
|-----------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| target_id       | String | target唯一标识符                                                                                                                                                                                      |
| gateway_id      | String | 所属网关的标识符                                                                                                                                                                                         |
| name            | String | target名称                                                                                                                                                                                         |
| status          | String | target状态。取值范围：<br>creating: 创建中  <br>updating: 更新中  <br>ready: 就绪可用   failed:<br>失败   deleting: 删除中  <br>synchronize_pending:<br>等待同步中  <br>synchronizing: 同步中  <br>active: 在线   inactive:<br>离线 |
| inactive_reason | String | Target离线状态原因。                                                                                                                                                                                    |
| target_type     | String | Target类型。取值范<br>围：mcp_server: MCP<br>服务器   openapi: 基于<br>OpenAPI 规范的 REST<br>API   APIG: 注册在云服<br>务 APIG上的 API   云服<br>务：华为云提供的云服<br>务，且接口注册在APIE<br>上                                          |
| created_at      | String | 创建时间戳                                                                                                                                                                                            |
| updated_at      | String | 更新时间戳                                                                                                                                                                                            |

表 13-63 ShowCoreGatewayTargetResponse

| 字段          | 类型     | 说明          |
|-------------|--------|-------------|
| target_id   | String | Target唯一标识符 |
| gateway_id  | String | 所属网关的标识符    |
| name        | String | Target名称    |
| description | String | Target描述    |

| 字段                                    | 类型     | 说明                                                                                                                                                                                               |
|---------------------------------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| status                                | String | Target状态。取值范围：<br>creating: 创建中  <br>updating: 更新中  <br>ready: 就绪可用   failed:<br>失败   deleting: 删除中  <br>synchronize_pending:<br>等待同步中  <br>synchronizing: 同步中  <br>active: 在线   inactive:<br>离线 |
| inactive_reason                       | String | Target离线状态原因。                                                                                                                                                                                    |
| target_type                           | String | Target类型。取值范围：<br>mcp_server: MCP<br>服务器   openapi: 基于<br>OpenAPI 规范的 REST<br>API   APIG: 注册在云服<br>务 APIG上的 API   云服<br>务：华为云提供的云服<br>务，且接口注册在APIE<br>上                                          |
| target_configuration                  | Object | Target配置                                                                                                                                                                                         |
| credential_provider_conf<br>iguration | Object | 凭证提供者配置                                                                                                                                                                                          |
| created_at                            | String | 创建时间戳                                                                                                                                                                                            |
| updated_at                            | String | 更新时间戳                                                                                                                                                                                            |

快速入门

步骤1 创建网关。

```
Client模式
result = client.create_gateway(
 name="my-gateway",
 description="我的第一个MCP网关"
)
gateway_id = result.data.get("gateway_id")

CLI模式
agentarts gateway create --name "my-gateway" --description "我的第一个MCP网关"
```

步骤2 创建Target。

```
Client模式
target_result = client.create_gateway_target(
 gateway_id=gateway_id,
 target_configuration={
 "mcp_server": {
 "endpoint": "https://echo.example.com",
 "server_type": "sse"
 }
 }
)
```

```
target = target_result.data.get("target")
target_id = target.get("target_id")

CLI模式
agentarts gateway create-target <gateway-id> \
--target-configuration '{"mcp_server":{"endpoint":"https://example.com/mcp"},"server_type":"sse"}'
\}}
```

### 步骤3 查询信息。

```
Client模式
gateway_info = client.get_gateway(gateway_id)
target_info = client.get_gateway_target(gateway_id, target_id)

CLI模式
agentarts gateway get <gateway-id>
agentarts gateway get-target <gateway-id> <target-id>
```

----结束

## 示例代码

```
from typing import Optional, List
from agentarts.sdk.gateway import GatewayClient

初始化客户端
client = GatewayClient()
创建MCP Gateway
result = client.create_gateway(
 name="production-echo-gateway",
 description="生产环境 Echo 服务网关",
 authorizer_type="iam"
)
gateway = result.data
print(f"创建的网关ID: {gateway.get('gateway_id')}")

更新网关描述
result = client.update_gateway(
 gateway_id=gateway.get('gateway_id'),
 description="更新后的网关描述"
)
updated_gateway = result.data
print(f"更新后的描述: {updated_gateway.get('description')}")

在网关下创建Target (MCP服务器类型)
result = client.create_gateway_target(
 gateway_id=gateway.get('gateway_id'),
 name="echo-service-target",
 description="Echo服务目标",
 target_configuration={
 "mcp_server": {
 "endpoint": "https://echo.example.com",
 "server_type": "sse"
 }
 }
)
target = result.data.get("target")
target_id = target.get("target_id")
print(f"目标创建成功, ID: {target_id}")

在网关下创建Target (OpenAPI内联文档类型)
result = client.create_gateway_target(
 gateway_id=gateway.get('gateway_id'),
 name="petstore-api",
 description="PetStore API目标",
 target_configuration={
 "openapi": {
 "payload": '{"info":{"description":"Search & Content Management API (OpenAPI 3.0.0 compliant)"
```

```

\,"title\":"Search & Content API - APIs Scene\","version\":"1.0.0\","openapi\":"3.0.0\","servers\":[{"url
\":"https://api.example.com/v1\"}],\"paths\":[{\"/api/resource/1\":{\"post\":{\"operationId
\":"createResource1\", \"summary\":"Create resource 1\", \"requestBody\":{\"required\":true, \"content\":
{\"application/json\":{\"schema\":{\"$ref\":\"#/components/schemas/ContentBase\"}}}, \"responses\":
{ \"201\":{\"description\":"Created success\", \"content\":{\"application/json\":{\"schema\":{\"type\":\"object
\", \"properties\":{\"id\":{\"type\":\"integer\"}, \"title\":{\"type\":\"string\"}}}}}}, \"components\":{\"schemas
\":"ContentBase\":{\"type\":\"object\", \"required\":[\"title\"], \"properties\":{\"title\":{\"type\":\"string
\"}, \"isPublic\":{\"type\":\"boolean\"}, \"tags\":{\"type\":\"array\"}, \"items\":{\"type\":\"string\"}, \"metadata\":
{ \"type\":\"object\", \"properties\":{\"createTime\":{\"type\":\"string\"}, \"operator\":{\"type\":\"string
\"}}}}}}}}"}
 }
 }
)
target = result.data.get("target")
target_id = target.get("target_id")
print(f"OpenAPI目标创建成功, ID: {target_id}")

在网关下创建Target (OpenAPI OBS文档类型)
result = client.create_gateway_target(
 gateway_id=gateway.get('gateway_id'),
 name="petstore-api-obs",
 description="PetStore API目标 (OBS) ",
 target_configuration={
 "openapi": {
 "obs": {
 "bucket_name": "api-specs-bucket",
 "object_key": "specs/petstore.yaml"
 }
 }
 }
)
target = result.data.get("target")
target_id = target.get("target_id")
print(f"OpenAPI OBS 目标创建成功, ID: {target_id}")

查询网关信息
result = client.get_gateway(gateway.get('gateway_id'))
gateway_info = result.data
print(f"网关状态: {gateway_info.get('status')}")

查询target信息
result = client.get_gateway_target(gateway.get('gateway_id'), target.get('target_id'))
target_info = result.data.get("target")
print(f"目标状态: {target_info.get('status')}")

列出网关下的所有target
result = client.list_gateway_targets(gateway.get('gateway_id'))
targets = result.data
print(f"网关下有 {targets.get('total')} 个目标")

更新Target描述
result = client.update_gateway_target(
 gateway_id=gateway.get('gateway_id'),
 target_id=target.get('target_id'),
 description="更新后的Echo服务目标描述"
)
updated_target = result.data.get("target")
print(f"更新后的描述: {updated_target.get('description')}")

删除Target
client.delete_gateway_target(gateway.get('gateway_id'), target.get('target_id'))
print("目标已删除")

删除Gateway
client.delete_gateway(gateway.get('gateway_id'))
print("网关已删除")

```

# 14 常见问题

## 14.1 认证鉴权

### 如何获取华为云 AK/SK

AK/SK认证就是使用AK/SK对请求进行签名，在请求时将签名信息添加到消息头，从而通过身份认证。

- AK(Access Key ID)：访问密钥ID。与私有访问密钥关联的唯一标识符；访问密钥ID和私有访问密钥一起使用，对请求进行加密签名。
- SK(Secret Access Key)：与访问密钥ID结合使用的密钥，对请求进行加密签名，可标识发送方，并防止请求被修改。

如果之前没有生成过AK/SK，可参考以下步骤来获取。

**步骤1** 登录控制台“[我的凭证 > 访问密钥](#)”，获取AK和SK。AK、SK最多可添加2个。

图 14-1 获取 AK、SK



图 14-2 AK、SK 示例

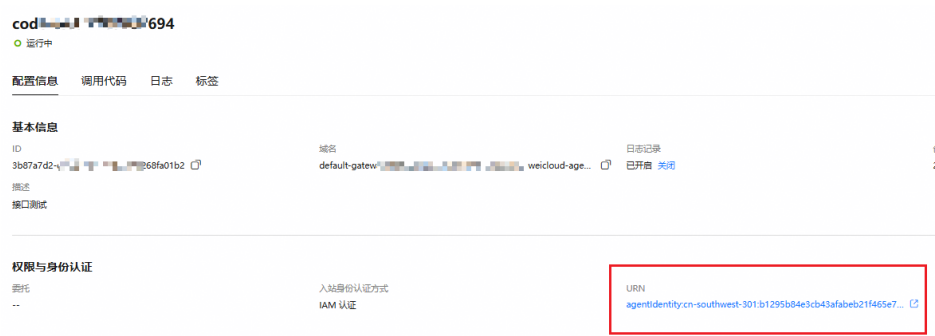
|   | A         | B             | C                 | D | E | F |
|---|-----------|---------------|-------------------|---|---|---|
| 1 | User Name | Access Key Id | Secret Access Key |   |   |   |
| 2 |           | HPUAKA        | 6irFsl1UjTok0k    |   |   |   |
| 3 |           |               |                   |   |   |   |
| 4 |           | AK            | SK                |   |   |   |
| 5 |           |               |                   |   |   |   |
| 6 |           |               |                   |   |   |   |
| 7 |           |               |                   |   |   |   |

----结束

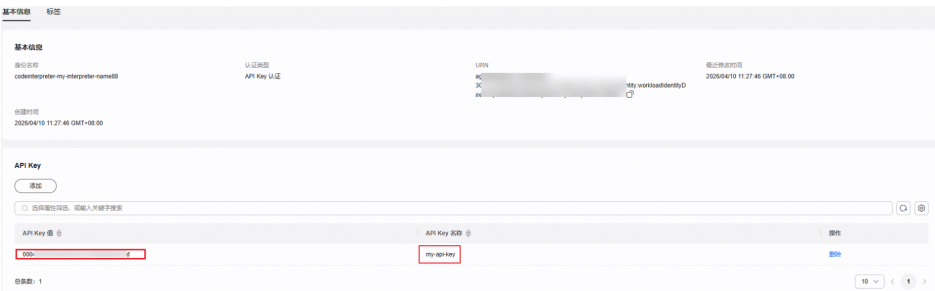
## 14.2 如何获取代码解释器的 API\_KEY

API\_KEY认证就是在请求时将API\_KEY信息添加到消息头，从而通过身份认证。

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 代码解释器”，进入“代码解释器”界面。
- 步骤3 单击代码解释器名称，进入“配置信息”界面。
- 步骤4 单击URN链接，进入基本信息页签。



- 步骤5 在API KEY模块即可查找API KEY值以及API KEY名称。

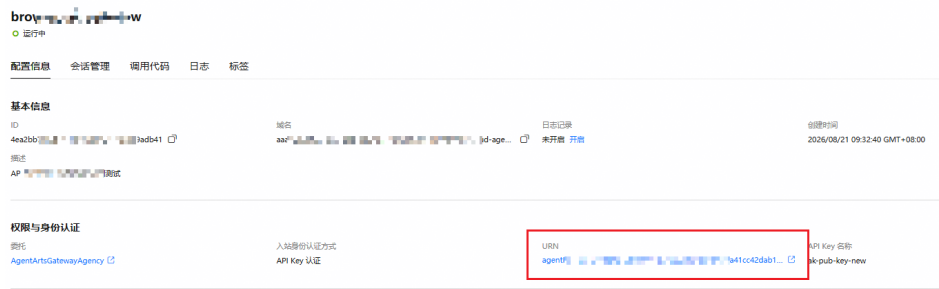


----结束

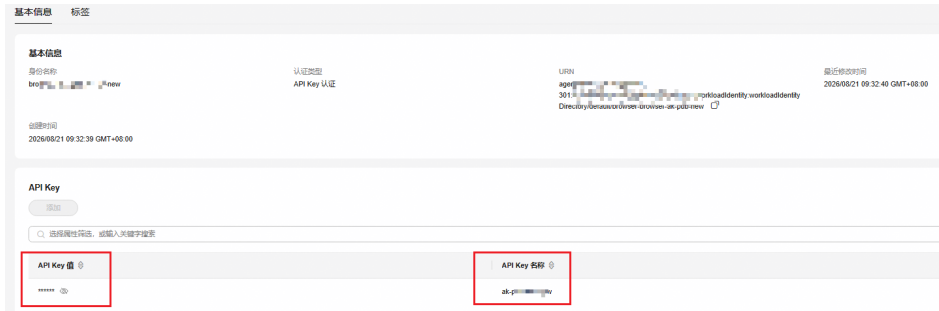
## 14.3 如何获取浏览器的 API\_KEY

API\_KEY认证就是在请求时将API\_KEY信息添加到消息头，从而通过身份认证。

- 步骤1 登录AgentArts智能体平台。
- 步骤2 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3 单击浏览器名称，进入“配置信息”界面。
- 步骤4 单击URN链接，进入基本信息页签。



**步骤5** 在API KEY模块即可查找API KEY值以及API KEY名称。



----结束

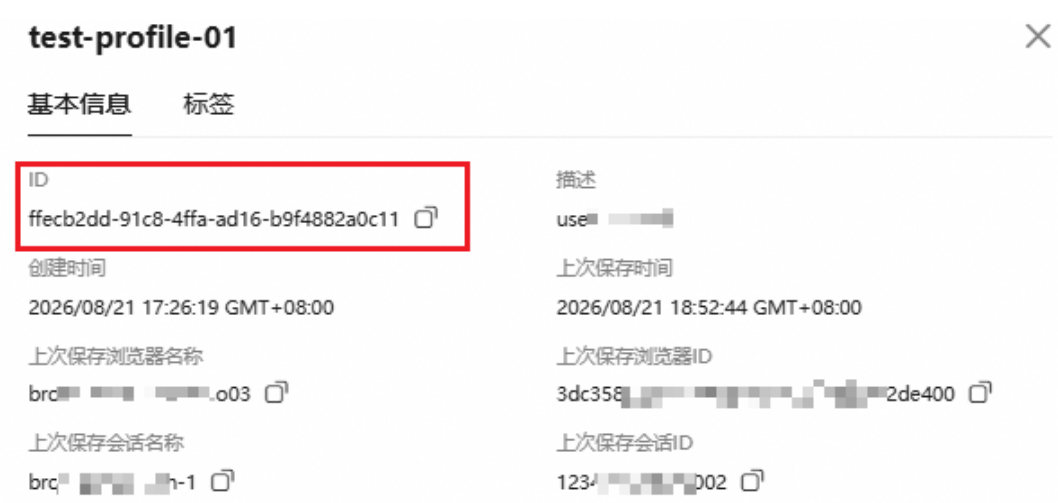
## 14.4 如何获取浏览器配置文件的 ID

在会话中加载配置文件时，需要获取配置文件的ID。

- 步骤1** 登录AgentArts智能体平台。
- 步骤2** 在左侧导航栏选择“托管与运行 > 浏览器”，进入浏览器界面。
- 步骤3** 单击右上角“配置文件”。
- 步骤4** 展示当前用户下的所有配置文件列表。
- 步骤5** 单击配置文件名称，在“基本信息”页签查看配置文件详情。



图 14-3 获取配置文件 ID



----结束

14.5 如何获取记忆库 ID 和 API Key

API\_KEY认证就是在请求时将API\_KEY信息添加到消息头，从而通过身份认证。  
在[创建记忆库](#)后，需要复制或下载API Key，以便在SDK客户端进行鉴权使用。  
API Key如丢失，需重新申请更新，请根据以下操作执行。

- 步骤1 登录[AgentArts智能体平台](#)。
- 步骤2 在左侧导航栏选择“托管与运行 > 记忆库”，进入“记忆库”界面。
- 步骤3 按页面提示创建记忆库。
- 步骤4 单击记忆库名称，进入“基本信息”界面，获取记忆库ID。

图 14-4 获取记忆库 ID



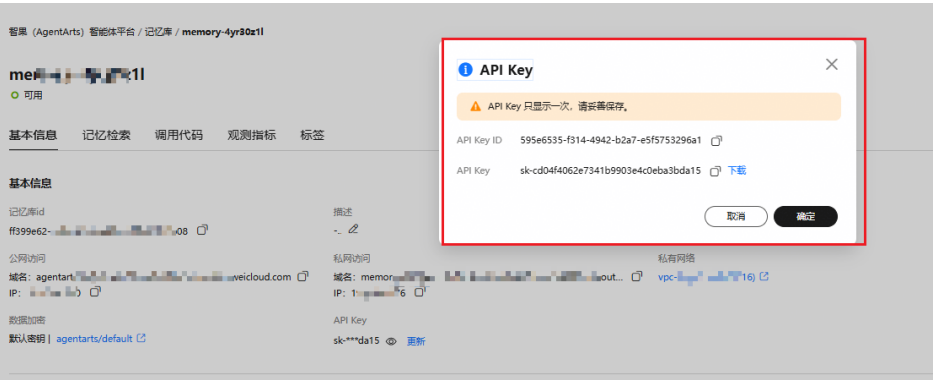
- 步骤5 在“API Key”处单击“更新”。

图 14-5 更新 API Key



步骤6 弹框中单击“确定”后，复制或下载API Key。

图 14-6 复制或下载 API Key



----结束

## 14.6 如何使用记忆库的 API Key

在调用数据面接口涉及鉴权时，请求的Header中需要包含对应记忆库的API-Key。传入Header的鉴权方式如下：

使用Authorization头部进行鉴权。Authorization头部内容格式为 Bearer <api-key>，其中 <api-key> 是您的API密钥，如Bearer sk-f6855361556a4a54881565132e46a2f4

如果没有正确传入Authorization头部或传入的API密钥无效，则报错NOT EXIST APIKEY。

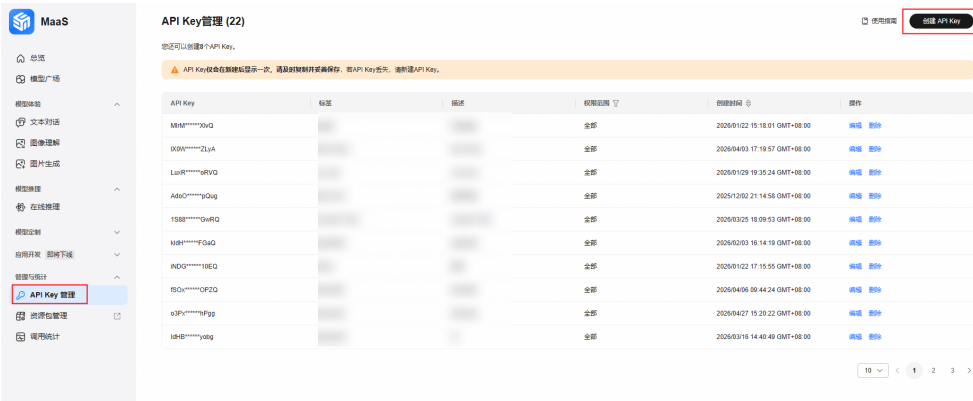
```
1 {
2 "error_code": "AgentArts.01020003",
3 "error_msg": "NOT EXIST APIKEY"
4 }
```

## 14.7 获取华为云 MaaS 服务模型 API Key

MaaS服务提供基于昇腾适配的主流大模型，覆盖DeepSeek、GLM、Kimi、Qwen等模型。

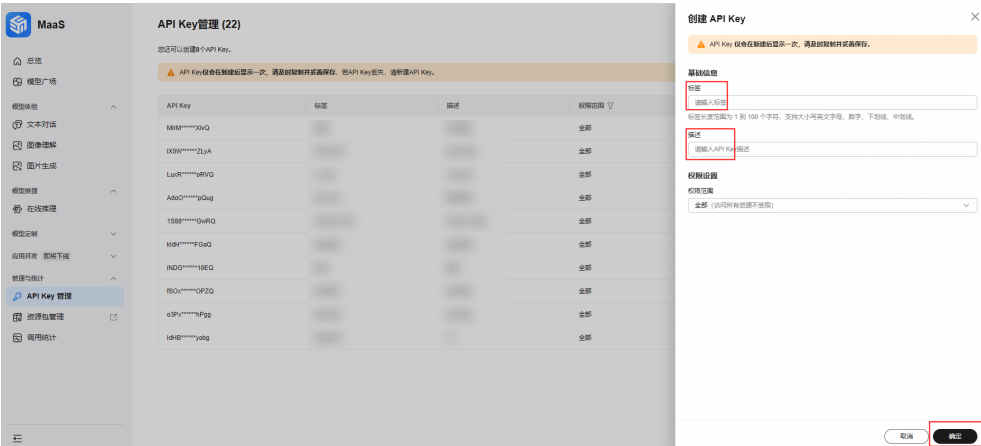
- 步骤1 登录华为云MaaS服务控制台。
- 步骤2 在左侧导航栏中选择“API Key管理”，单击“创建 API Key”。

图 14-7 API Key 管理



- 步骤3 根据页面提示，填写标签、描述后，单击“确定”创建API Key。

图 14-8 创建 API Key



----结束

## 14.8 Dockerfile 参数在智能体运行时中如何使用

### 应用场景

如果您了解容器引擎的使用，明白定制镜像时，一般使用Dockerfile来完成。Dockerfile是一个文本文件，其内包含了一条条的指令，每一条指令构建镜像的其中一层，因此每一条指令的内容，就是描述该层应该如何构建。

本章节将介绍Dockerfile文件的一些配置如何对应到智能体运行时中去使用。

Dockerfile 参数在智能体运行时中的使用

下面通过一个例子来说明它们之间的关系，这样您就可以更好地了解和熟悉智能体运行时。

```
FROM ubuntu:16.04
ENV VERSION 1.0
EXPOSE 80
ENTRYPOINT ["/entrypoint.sh"]
CMD ["start"]
```

上面是一个Dockerfile文件，包含一些常见的参数ENV、EXPOSE、ENTRYPOINT、CMD，这些参数在智能体运行时中可以按如下方法配置。

- ENV为环境变量，可以在创建智能体运行时的高级配置中设置“ENV VERSION 1.0”指令，如下所示。

监听端口

8080

智能体运行时监听的端口号，用于接收外部请求。默认为8080，可选端口为1-65535。

环境变量 (可选)

表单编辑JSON 格式编辑

VERSION1.0

+ 添加变量

您还可以添加49个变量

- ENTRYPOINT与CMD对应智能体运行时中高级配置的启动命令。

VPC

v1

新建VPC

子网

su

新建子网

安全组

Sys

新建安全组

启动命令 (可选)

./entrypoint.sh;start

镜像启动命令，多个命令以英文分号分隔，每个命令长度为1-256个字符。

监听端口

8080

- EXPOSE即暴露某个端口，智能体运行时中如果要对外暴露端口，则需要和在创建智能体运行时配置的监听端口一致。

## 14.9 执行 `pip install agentarts-sdk` 命令安装 AgentArts SDK 时速度较慢或下载超时

### 问题描述

在使用 `pip install agentarts-sdk` 命令安装 AgentArts SDK 时，可能会遇到下载速度较慢或下载超时的问题。

### 解决方案

通常是由于网络连接问题或没有使用国内镜像源导致的。

- 检查网络连接  
确保您的网络连接正常，可以尝试访问其他网站或使用其他网络环境进行测试。
- 使用国内镜像源安装  
为了加快下载速度，建议使用国内的镜像源进行安装，例如使用华为云镜像源下载：

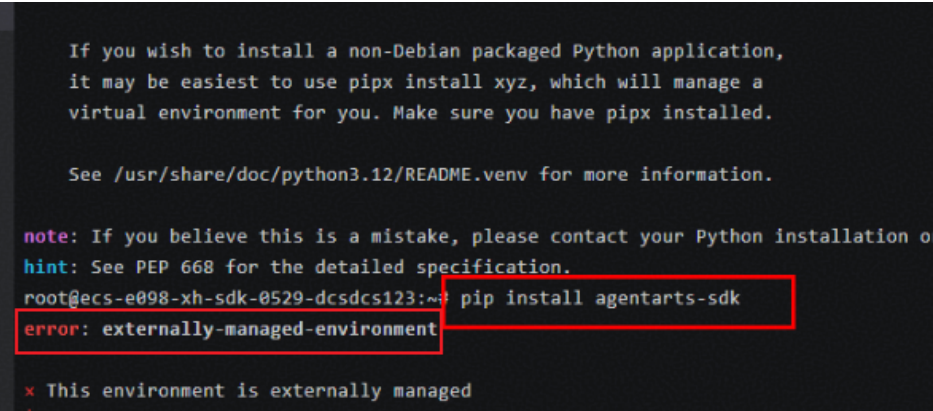
```
pip install agentarts-sdk -i https://repo.huaweicloud.com/repository/pypi/simple --trusted-host repo.huaweicloud.com
```

## 14.10 执行 `pip install agentarts-sdk` 命令安装 AgentArts SDK 时出现 `externally-managed-environment` 报错

### 问题描述

执行 `pip install agentarts-sdk` 命令安装 AgentArts SDK 时出现 `externally-managed-environment` 报错。

图 14-9 出现 externally-managed-environment 报错



解决方案

建议使用虚拟环境进行安装，以避免与系统全局环境发生冲突。

```
创建并激活虚拟环境 (linux)
python3 -m venv venv
source venv/bin/activate
```

14.11 如何将已有的 Agent 应用迁移至 AgentArts 运行时

背景

如果您已有一个完整的Agent应用，希望将其运行在AgentArts平台上，但不希望基于AgentArts SDK进行重新开发。AgentArts支持轻量迁移模式：只需对原有应用进行少量微调，即可部署到 AgentArts 运行时。

迁移步骤

步骤1 添加健康检查接口

AgentArts底座沙箱会通过健康探查接口检测您的Agent是否就绪。您需要在原有应用中新增一个GET /ping接口。

| 项目    | 说明                  |
|-------|---------------------|
| 请求方法  | GET                 |
| 路径    | /ping               |
| 响应状态码 | 200                 |
| 响应体格式 | JSON                |
| 触发时机  | 镜像启动完成后，沙箱会周期性调用此接口 |

响应体格式

```
{
 "status": "Initing | Healthy | HealthyBusy"
}
```

| 状态值         | 含义       | 说明                |
|-------------|----------|-------------------|
| Initing     | 初始化中     | 应用正在启动，依赖服务或模型加载中 |
| Healthy     | 进程健康且空闲  | 应用已就绪，可接受请求       |
| HealthyBusy | 进程健康且非空闲 | 应用运行中但正忙于处理其他请求   |

镜像启动完成后，进程处于健康状态时，接口必须返回200状态码且status字段为Healthy，沙箱才会认为实例就绪。

代码示例

Python (Flask)

```
from flask import Flask, jsonify

app = Flask(__name__)

新增健康检查接口
@app.route('/ping', methods=['GET'])
def ping():
 return jsonify({"status": "Healthy"}), 200

保留您原有的业务接口
@app.route('/chat', methods=['POST'])
def chat():
 # 您原有的业务逻辑
 ...

if __name__ == '__main__':
 app.run(host='0.0.0.0', port=8080)
```

Python (FastAPI)

```
from fastapi import FastAPI

app = FastAPI()

新增健康检查接口
@app.get("/ping")
async def ping():
 return {"status": "Healthy"}

保留您原有的业务接口
@app.post("/chat")
async def chat():
 # 您原有的业务逻辑
 ...
```

Node.js (Express)

```
const express = require('express');
const app = express();

// 新增健康检查接口
app.get('/ping', (req, res) => {
 res.status(200).json({ status: 'Healthy' });
});
```

```
// 保留您原有的业务接口
app.post('/chat', (req, res) => {
 // 您原有的业务逻辑
});

app.listen(8080, '0.0.0.0');
```

### Java (Spring Boot)

```
@RestController
public class AgentController {

 // 新增健康检查接口
 @GetMapping("/ping")
 public ResponseEntity<Map<String, String>> ping() {
 return ResponseEntity.ok(Collections.singletonMap("status", "Healthy"));
 }

 // 保留您原有的业务接口
 @PostMapping("/chat")
 public ResponseEntity<?> chat(@RequestBody Map<String, Object> request) {
 // 您原有的业务逻辑
 ...
 }
}
```

- 健康检查接口应确保在应用完全启动（依赖服务就绪、模型加载完成等）后再返回 Healthy 状态。
- 如果应用尚未就绪，可返回 {"status": "Initing"}，沙箱会继续重试。

### 步骤2 打包镜像并上传至华为云 SWR。

1. 登录容器镜像服务控制台，选择区域，要和AgentArts区域保持一致，否则无法选择到镜像。
  2. 单击右上角“创建组织”，输入组织名称完成组织创建。请自定义组织名称，本示例使用“demo-886633”，下面的命令中涉及组织名称“demo-886633”也请替换为自定义的值。
  3. 单击右上角“登录指令”，获取登录访问指令，本文选择复制“通用型登录指令”。
- 以root用户登录本地环境，输入复制的SWR临时登录指令。
4. 上传镜像至容器镜像服务镜像仓库。

- a. 使用docker tag命令给上传镜像打标签。

```
#region和domain信息请替换为实际值，组织名称demo-886633也请替换为自定义的值。
sudo docker tag agentarts-custom:latest swr.{region-id}.{domain}/demo-886633/agentarts-
custom:latest
#此处以华为云cn-southwest-2为例
sudo docker tag agentarts-custom:latest swr.cn-southwest-2.myhuaweicloud.com/demo-886633/
agentarts-custom:latest
```

- b. 使用docker push命令上传镜像。

```
#region和domain信息请替换为实际值，组织名称demo-886633也请替换为自定义的值。
sudo docker push swr.{region-id}.{domain}/demo-886633/agentarts-custom:latest
#此处以华为云cn-southwest-2为例
sudo docker push swr.cn-southwest-2.myhuaweicloud.com/demo-886633/agentarts-
custom:latest
```

5. 完成镜像上传后，在容器镜像服务控制台的“我的镜像”页面可查看已上传的自定义镜像。

### 步骤3 部署智能体运行时

部署智能体运行时，具体请参考[通过控制台部署](#)。



部署时必须确保路由配置为“前缀匹配”，以便请求能正确转发到镜像内部的自定义接口。

图 14-10 路由配置

可观测配置

日志记录

未开启

启用日志功能后，Agent运行时产生的日志会上报云日志服务(LTS)，日志管理费用按需收取。[了解LTS计费详情](#)

高级配置

出网网络配置

公网访问

私网访问

存储配置 (可选)

未开启

如需使用SFS Turbo等类型存储配置，需要开启私网访问。

启动命令 (可选)

+ 添加启动命令

您还可以增加10个镜像启动命令。每个命令长度为1-256个字符，如果命令中有指定端口需与监听端口一致。

监听端口

8080

智能体运行时监听的端口号，用于接收外部请求。默认为8080，可选端口为1-65535。

路由配置

严格匹配

前缀匹配

调用URL的匹配模式。支持/runtimes/{runtime\_name}/invocations以及/runtimes/{runtime\_name}/invocations/{custom\_path}的调用路径。

文件上传下载

步骤4 调整调用方式，

部署完成后，原有的接口调用路径需要按照新规则进行调整。

路径转换规则

旧路径：POST https://旧的平台域名/<接口路径>

新路径：POST https://<AgentArts运行时访问域名>/runtimes/<运行时名称>/invocations/<接口路径>

转换示例

| 场景            | 旧调用方式                           | 新调用方式                                                              |
|---------------|---------------------------------|--------------------------------------------------------------------|
| 对话接口          | POST https://旧域名/chat           | POST https://<运行时访问域名>/runtimes/<运行时名称>/invocations/chat           |
| completions接口 | POST https://旧域名/v1/completions | POST https://<运行时访问域名>/runtimes/<运行时名称>/invocations/v1/completions |
| 自定义接口         | POST https://旧域名/my/custom/api  | POST https://<运行时访问域名>/runtimes/<运行时名称>/invocations/my/custom/api  |

完整调用示例

假设：

- 运行时访问域名为 `https://example.cn-southwest-2.huaweicloud-agentarts.com`
- 运行时名称为 `my-agent-runtime`
- 原有接口为 `POST /chat`

调整后的调用方式：

```
curl -X POST https://example.cn-southwest-2.huaweicloud-agentarts.com/runtimes/my-agent-runtime/
invocations/chat \
-H "Content-Type: application/json" \
-d '{
 "messages": [
 {
 "role": "user",
 "content": "你好"
 }
]
}'
```

----结束

## 常见问题

1. 原有的业务接口需要修改吗？  
不需要。原有的业务接口代码无需改动，只需确保路由配置为前缀匹配，AgentArts会将 `/runtimes/<运行时名称>/invocations/` 前缀剥离后转发到您的实际接口。
2. 部署后调用返回404？  
请检查：
  - a. 部署时路由配置是否为**前缀匹配**模式。
  - b. 调用路径是否已按新规则调整，包含 `/runtimes/<运行时名称>/invocations/` 前缀。
  - c. 镜像中的服务是否已正确启动（可通过健康检查状态确认）。

## 14.12 创建网关后调用报错：委托缺少 CSMS/KMS 相关 action 权限

### 问题描述

在调用网关过程中，返回报错缺少相关action策略，完整报错信息如下：

```
{
 "jsonrpc": "2.0",
 "id": 0,
 "error": {
 "code": -32603,
 "message": "send error: http upstream error: http request failed: outbound authentication failure: get
api key failed, status code: 403 Forbidden, body: ErrorResponse { error_code:
\\\"AgentIdentityTokenVault.1007\\\", error_msg: \\\"User: sts::xxx:assumed-
agency:AgentArtsGatewayAgency/xxx is not authorized to perform: csms:secret:getVersion on resource:
csms:regionId:xxx:secretName:apikeycredentialprovider-a452e54f because no identity-based policy allows
the csms:secret:getVersion action.\\\" }"
 }
}
```

### 原因分析

网关实际运行在Agent的沙箱环境中，当网关需要访问其他云服务时，需要在创建网关时选择相关的委托（Agency），缺少相应权限时会报此错误。主要原因如下：

1. **入站和出站身份认证**：网关调用涉及入站和出站身份认证，该模块通过调用 AgentIdentity 实现。因此，创建网关时选择的委托中需要包含 Identity 调用相关策略，所需权限请参考 [创建委托并授权](#)。
2. **密钥凭据权限**：csms:secret:getVersion 和 kms:cmk:decryptDataKey 涉及密钥凭据权限，需自行补充。如果在调用场景中需要其他 action，也可以自行加入。

解决方案

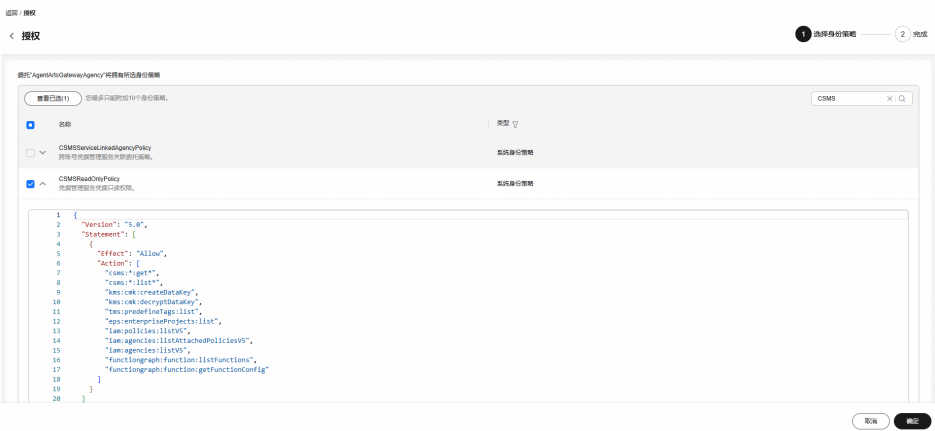
方案一：为已有委托补充 CSMS 权限

如果在网关创建时已选择 AgentArtsGatewayAgency 委托，可通过以下步骤补充授权：

- 步骤1** 进入统一身份认证服务 IAM，选择“委托”。
- 步骤2** 在委托列表选择 AgentArtsGatewayAgency 委托，单击操作列的“授权”。



- 步骤3** 在策略列表中搜索“CSMS”，选择 CSMSReadOnlyPolicy 策略（该策略包含 csms:secret:getVersion 和 kms:cmk:decryptDataKey 权限）。



- 步骤4** 单击“确定”完成授权。

-----结束

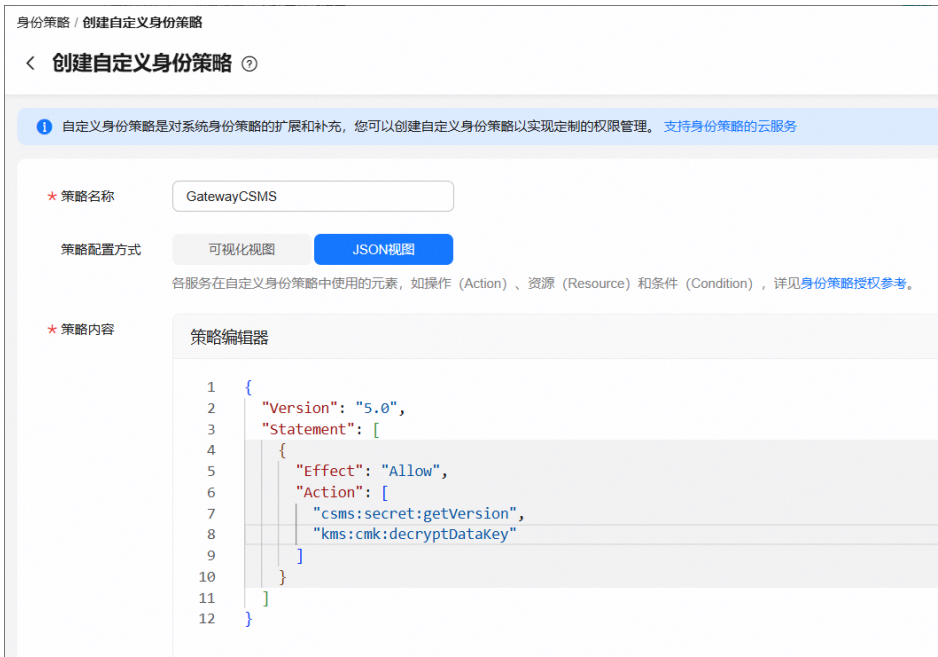
方案二：创建自定义身份策略并关联委托

- 步骤1** 进入统一身份认证服务 IAM，选择“身份策略”。
- 步骤2** 单击“创建自定义身份策略”。



**步骤3** 参考下图配置所需的action权限（如csms:secret:getVersion、kms:cmk:decryptDataKey等），然后单击“确定”。

```
{
 "Version": "5.0",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "csms:secret:getVersion",
 "kms:cmk:decryptDataKey"
]
 }
]
}
```

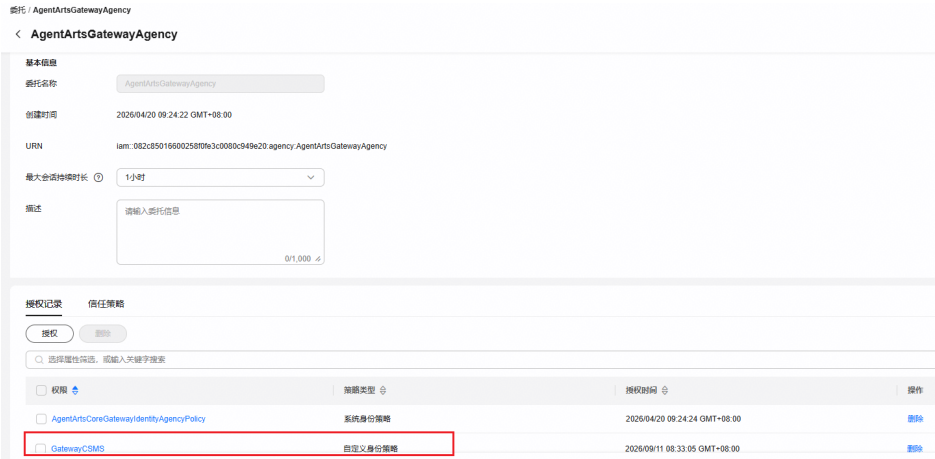


**步骤4** 在左侧菜单栏选择“委托”，在委托列表选择AgentArtsGatewayAgency委托，单击操作列的“授权”。

**步骤5** 在策略列表中，选择**步骤3**中创建的自定义策略。



**步骤6** 单击“确定”完成授权后，即可在“基本信息 > 授权记录”中查看。



----结束