ROMA Exchange

参考

文档版本 01

发布日期 2024-07-03

华为云计算技术有限公司

商标声明


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。

目 录

1 概述

1.1 应用开通定义文件介绍

应用是指实现了某种业务管理的可运行应用程序。

应用提供商将对应的业务管理能力作为商品发布到ROMA Exchange，在用户购买了对应的应用之后，ROMA Exchange需要负责打通应用提供商和购买者之间的交付流程，比如，创建应用账号、告知应用接入环境信息、提供应用包的下载链接等。

考虑到应用交付流程的差异性，ROMA Exchange产品提供了应用开通定义文件能力，一种特殊的适配器，应用提供商在发布应用类商品时，按照规范完成开通定义文件规范，并将定义文件上传到ROMA Exchange，实现不同应用在ROMA Exchange运营平台上的快速接入。

1.2 读者对象

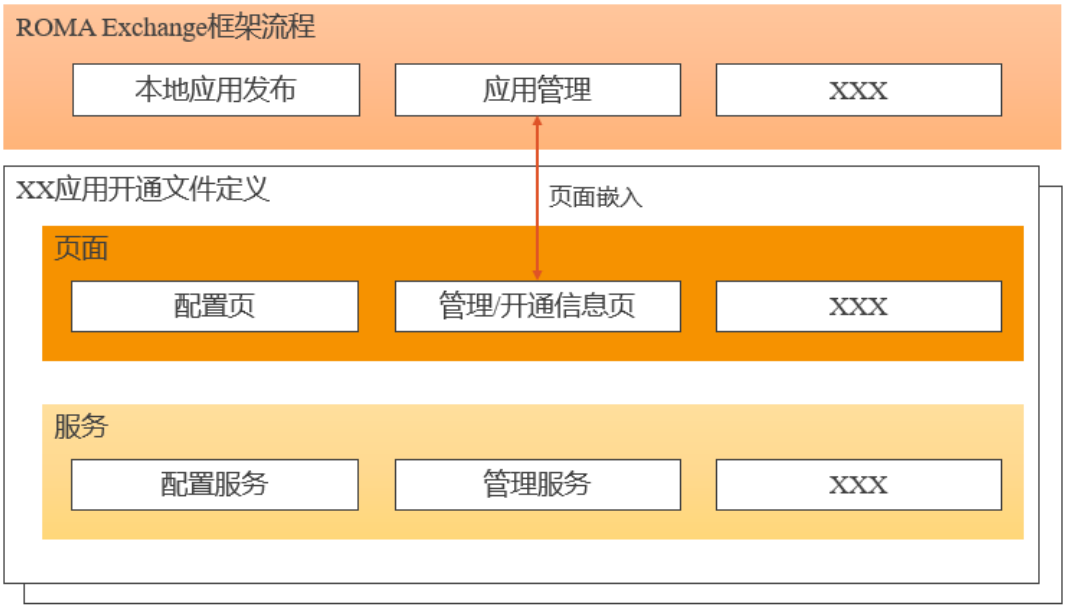
目前开通定义文件仅支持基于Astro轻应用开发平台进行开发。

本文档读者对象是基于Astro轻应用针对ROMA Exchange产品进行应用接入定制开发的人员。

开发人员在开发需求过程中可参考本指导进行开发。

2 架构介绍

ROMA Exchange开通文件架构如下：



开通定义文件架构中，主要包括两部分：

1.页面部分；通过页面部分与ROMA Exchange框架流程进行交互，这部分页面需要遵循开发文件定义规范，以保证与ROMA Exchange框架流程的正确的交互。当前主要包括如下2个页面：

- 管理/开通信息页；在订购者成功订购资产后，通过该页面对资产进行管理，如SaaS账号创建，环境信息回显，应用下载链接显示等；
- 配置页；考虑到应用开通配置信息有变更的场景，则需要在开通文件中定义配置页面，用于后续环境配置修改；

2. 服务部分；这部分提供页面所需要的能力；

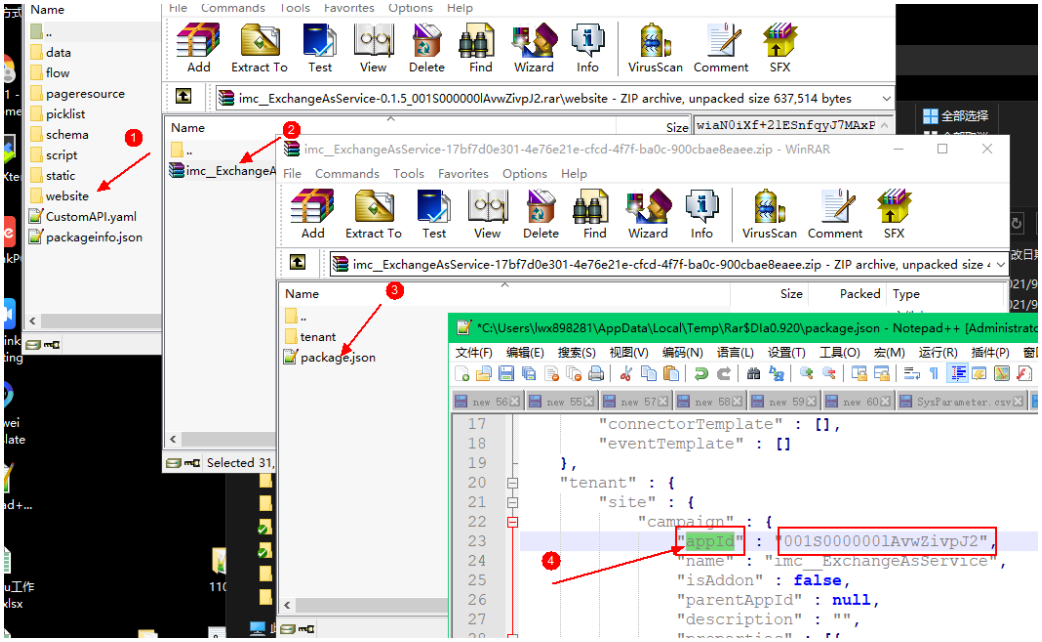
2.1 交互流程

前提：适配器内部与业务平台的交互配置已完成

1.1 应用发布

1、 用户选择交付方式为在线开通，上传定义文件。

说明：上传的适配器包名称需要有站点信息。格式如：{application}_{siteId}.zip



1. application：适配器名称，建议有版本号。【例如：imc__ExchangeAsService-0.1.5 】
2. siteId：适配器在Astro轻应用平台的站点id，获取方式如上图：

a步骤：解压适配器包进入website目录

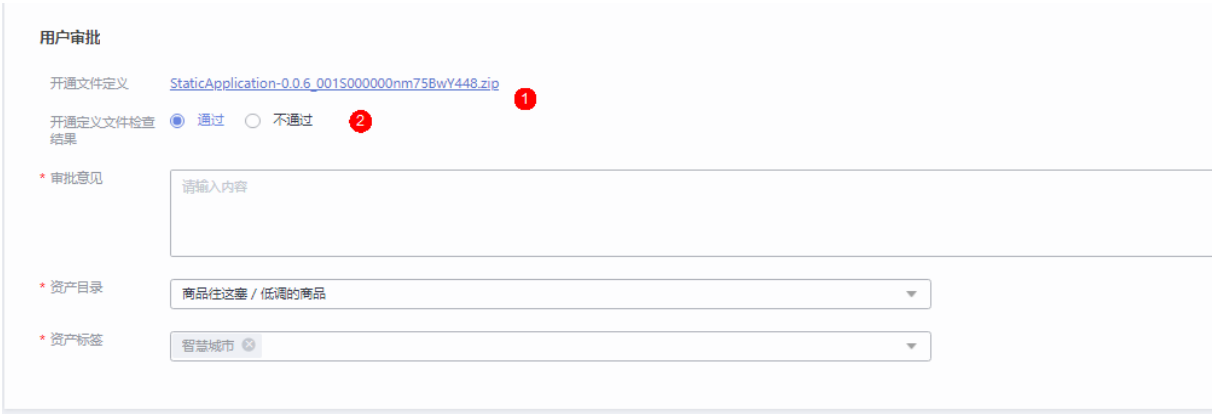
b步骤：解压包website包，找到package.json文件

c步骤：打开package.json文件

d步骤：package.json文件中找到appId对应的值为站点Id。

1. {application}_{siteId}.zip：适配器名称和站点用短横线连接，后缀为.zip。

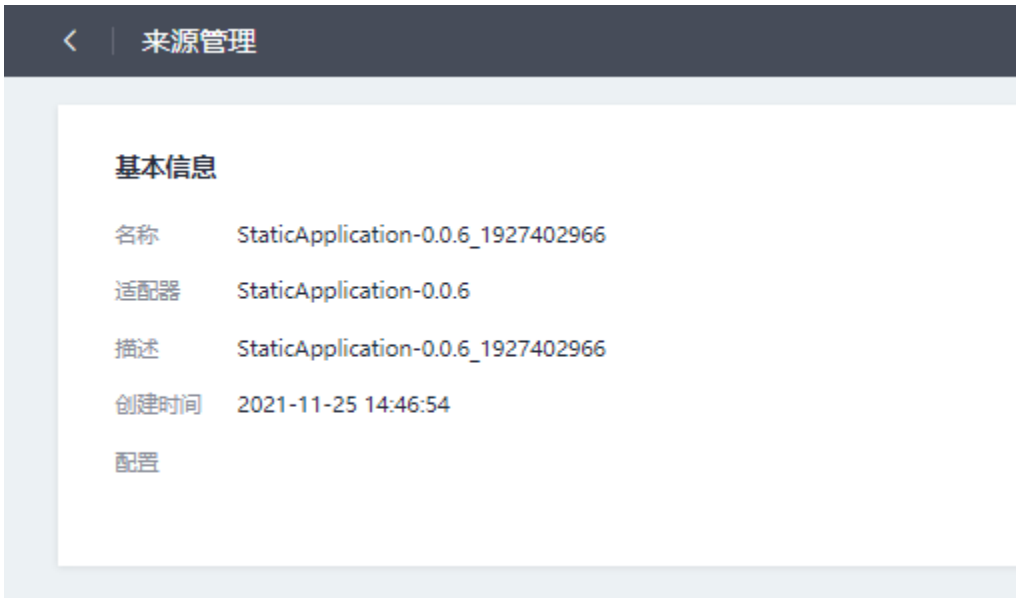
2、用户上架发布后，管理员审批：



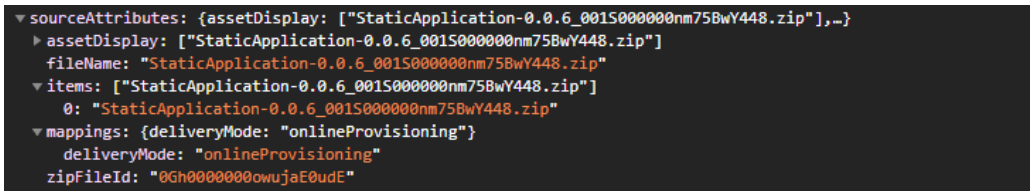
开通定义文件有附件链接给管理员下载检查，通过后管理员将开通定义文件检查为通过，填写审批意见、资产目录、资产标签。提交审批后后台会生成适配器数据，来源数据，商品级来源属性数据，安装适配器软件包应用。

1. 生成适配器详情：适配器类型为本地应用分类

1. 生成的来源详情：来源配置为空，适配器通过内部配置与业务平台实现交互，来源做配置后台默认为出空配置。



1. 管理员审批通过完成后台生成商品来源属性数据。



- assetDisplay：给页面组件展示资产显示元素
- fileName：上传适配器软件包名称
- items：显示在基本信息页面的资产对象
- mappins：高级搜索时对该商品标记交付方式为在线开通
- zipFileId：上传适配器软件包生成的附件唯一标识

1.2 订阅管理

1、 用户选择已订阅资产，单击“管理”，弹出适配器部署页面

2、 适配器页面操作（用于查看开通类密钥或安装部署资产包）

通过来源ID查询来源信息接口获取适配器页面参数进行适配器内部页面跳转(需要适配器内部页面跳转的才加，如deploy)

{

adaptTypes: [] //适配类型

adapterId: "" //适配器ID

adapterName: "" //适配器名称

adapterParams: "" //适配器的页面参数

adapterType: "" //适配器类型

configParams: "" //配置参数

createdate: "" //来源创建时间

description: "" //来源描述

id: "" //来源ID

isSupportCrossSource: false //是否支跨来源

name: "" //来源名称

}

3、完成后下次进入管理界面，直接显示执行结果

（下图为案例适配器内部deploy页面，管理操作后相应业务完成后可展示deploy页面信息）

3 开发流程说明

3.1 开发和编译

前提：开发租户需要订购基线资产

1. 创建空白应用

输入标签和名称后单击创建即可，APP命名规范为****Adapter。名称需要保证唯一性。

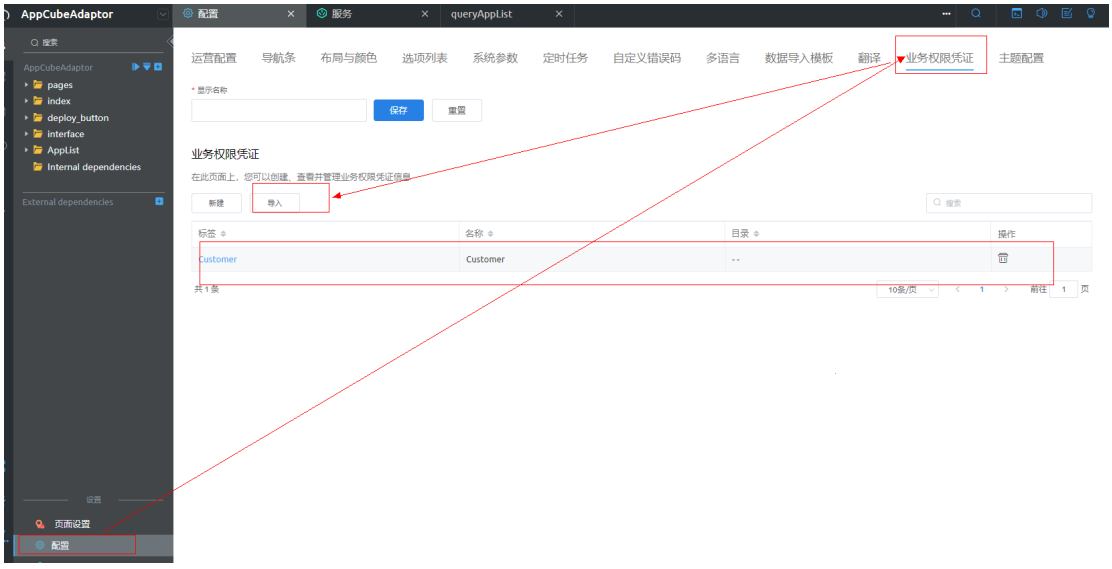
2、进入APP开发页面和接口

参考第4章节

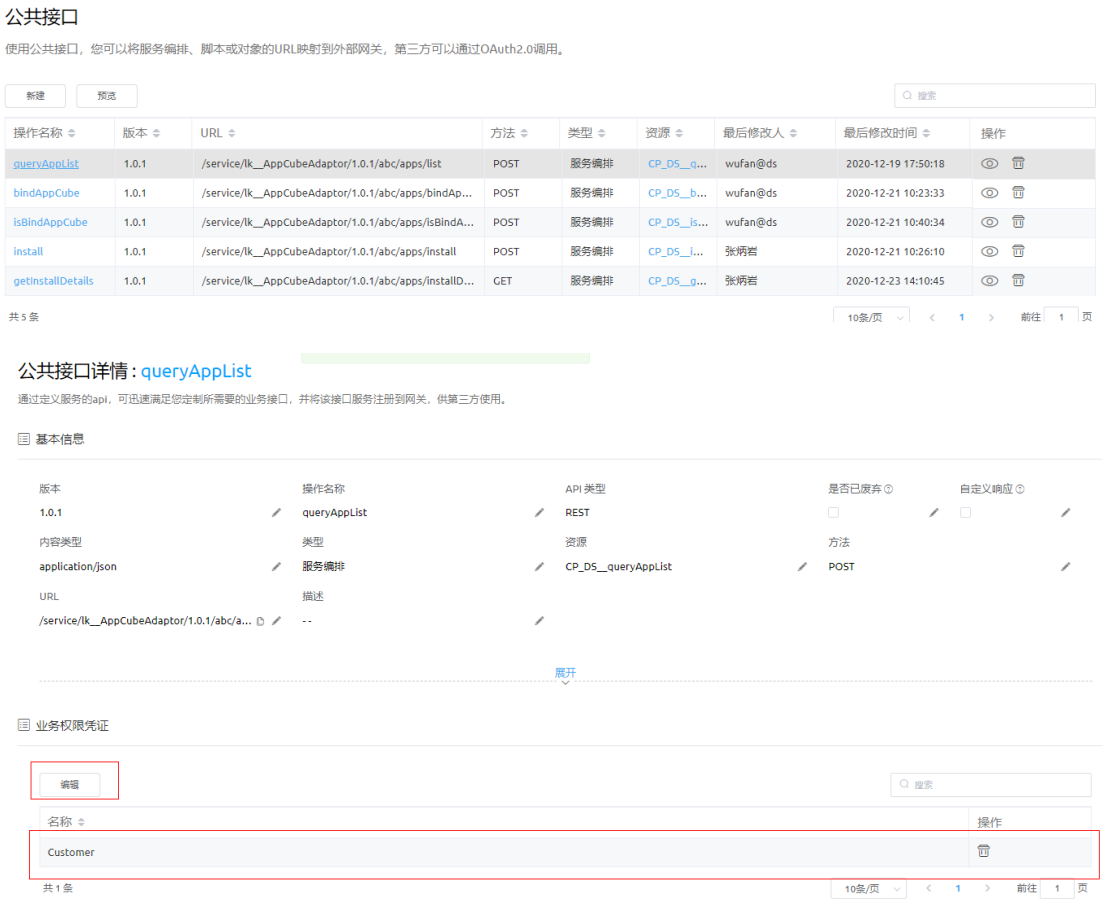
配置接口业务权限

接口开发完后，需要给每个接口配置业务权限

1）先导入“Customer”业务权限凭证



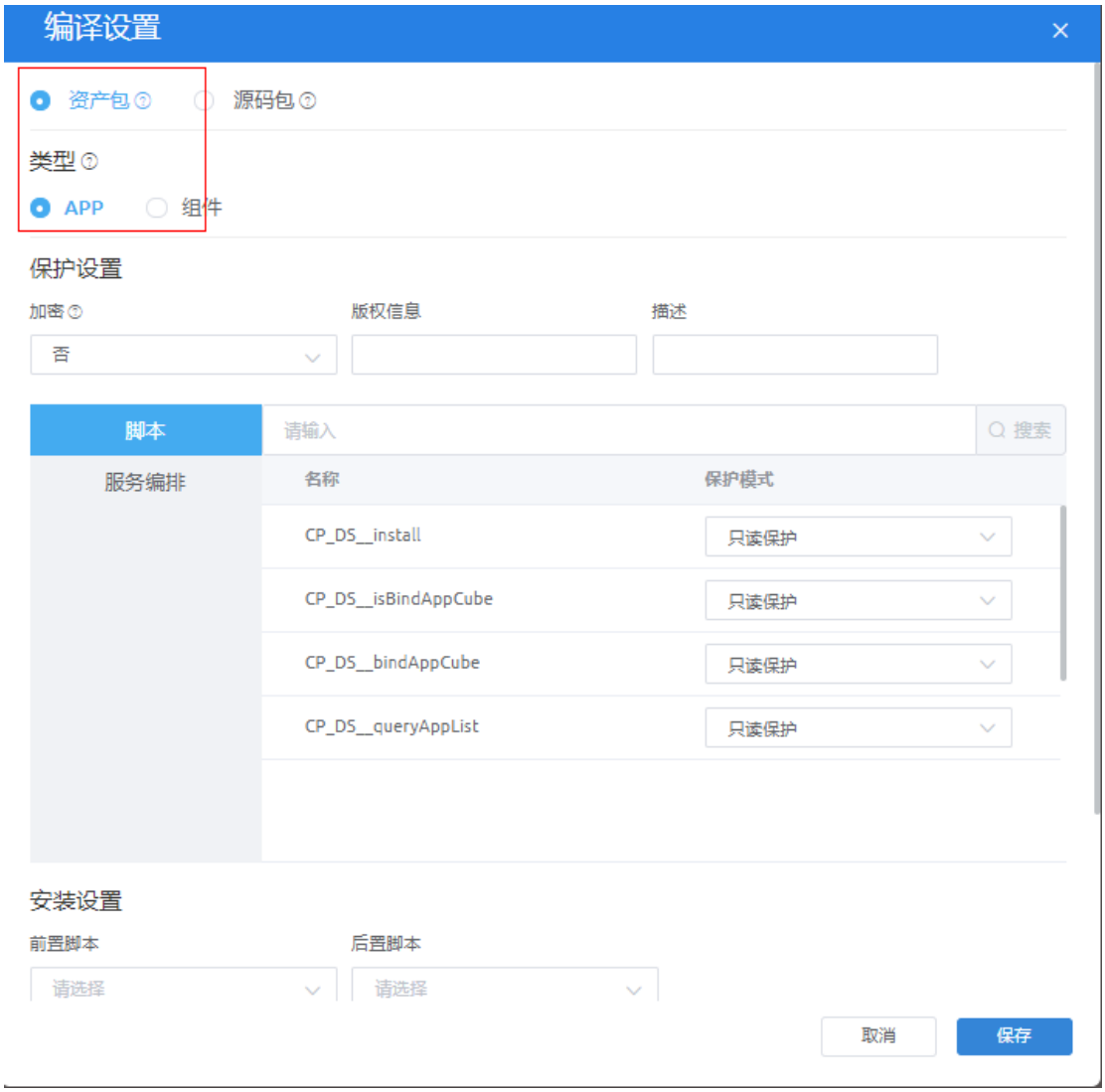
2）给每个公共接口加上 “Customer”业务权限凭证



3、编译设置选择资产包

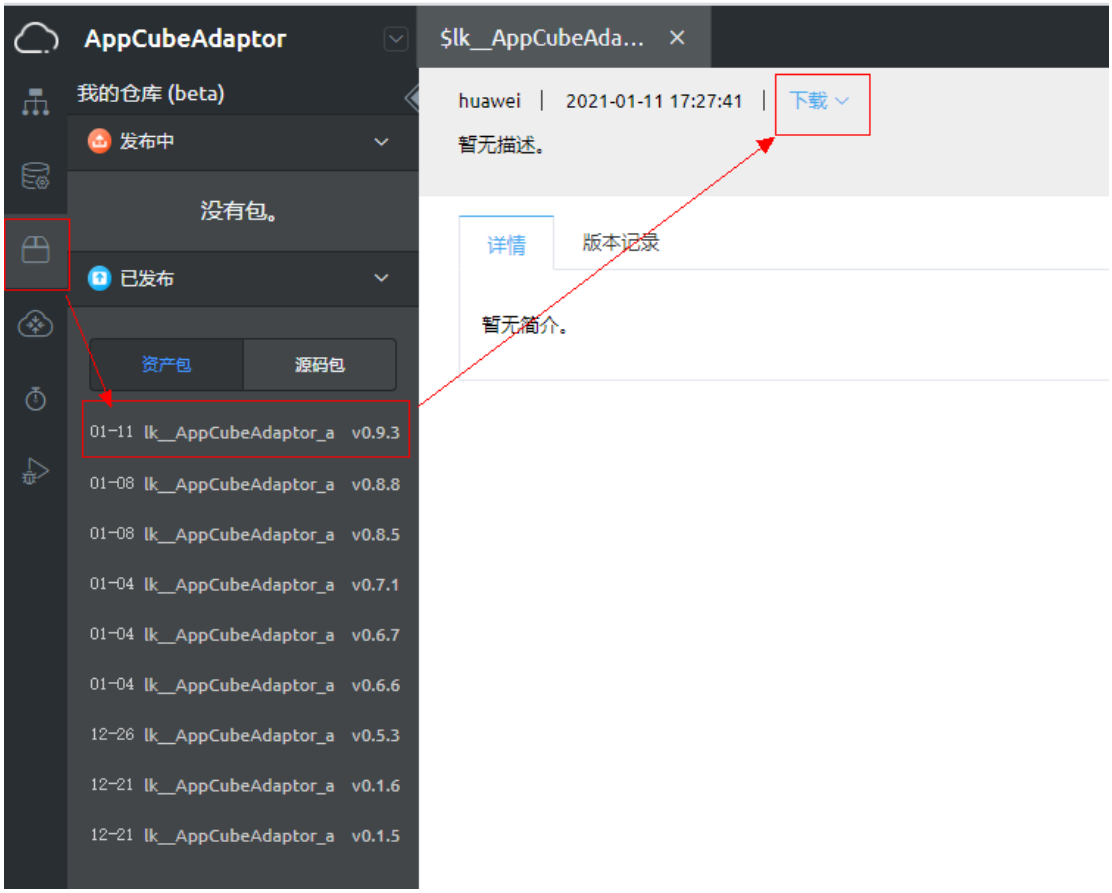
选择左下角“设置”

选择资产包、APP类型



4、先单击编译后单击发布：

5、在我的仓库下载资产包



3.2 安装和配置

安装：

用户上架后，管理台检查通过审批提交后，后台自动安装上架时上传的适配器软件包应用到环境中。

配置：

应用开通文件为ROMA Exchange适配器（流水线式）框架，该适配器不需要通过适配器配置页定义适配器引入参数（如来源配置），适配器内部可以通过系统参数或配置文件形式来配置来实现适配器与业务平台的交互。

4 开发指导

4.1 前台

4.1.1 开通定义文件组成

在管理上，不同应用的差异性主要体现在：来源参数配置中、对已订阅资产进行管理操作时，这些接入点要在适配器中开发。

因此，适配器里一般包含参数配置页、资产对象选择页和部署页三个页面，采用iframe的方式嵌入到Roma Exchange框架的相关页面中，并通过PostMessage \AddEventListener接口实现适配器与Roma Exchange框架页面的数据通信。

适配器为Astro轻应用中创建的APP，创建步骤：

1、在Astro轻应用开发环境首页的“项目”页签下单击“行业应用”，再单击“创建空白行业应用”。

2、在提示框中填写应用标签和名称。

3、创建高级页面

首先在APP中新建两个目录用来存放高级页面和脚本，建议分别命名为“pages”和“service”。

在pages中新建：参数配置页（config）、资产对象选择页（item）和管理页（deploy）三个高级页面。

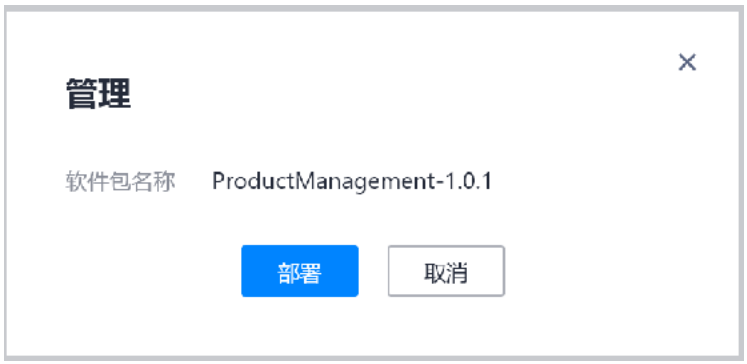
4、适配器Widget创建

分别创建四个widget组件，为了避免组件名称冲突，建议命名规范为：适配器_页面名_Widget（如：ROMAConnect_Config_Widget），然后在左侧菜单栏的“资产”中单击“组件”，提交新组件（具体开发参考平台相关规范）。

4.1.1.1 管理页(deploy)

用户订阅资产后，在“已订阅资产”中可以进行“管理”操作，实现对部署类资产的安装部署或授权开通类资产的获取密钥。

下图是Astro轻应用适配器中的管理页（deploy）



4.1.2 页面交互事件

适配器与Roma Exchange框架有多次交互，主要通过PostMessage和AddEventListener接口实现数据通信。对于适配器页面来说，有多次发送消息事件和接收消息事件，其中每次发送数据之后，ROMA Exchange都会根据监听到的事件名触发对应的动作。

以Astro轻应用适配器的资产对象选择页为例，五次发送消息事件和一次接收消息事件分别介绍如下：

1. 触发数据传输事件

   发布者在资产发布页单击资产对象时，适配器资产对象选择页需要获取ROMA Exchange框架传来的“资产来源标识（sourceId）”、“资产来源参数（sourceAttributes）”。

   其中，“资产来源标识”用来调接口获取资产对象数据；“资产来源参数”包含了从草稿中获取的数据，用于数据回显；“地址信息”包含了适配器鉴权页的跳转地址，用于未授权情况下的页面跳转。

   因此，适配器需执行触发动作给ROMA Exchange，ROMA Exchange监听到该事件时，就会触发数据传递动作。

代码示例：

```
created () {
let eventObj = {
eventName: 'PARENT_EVENT'
params: {
message: 'loading'
}
};
parent.postMessage(JSON.stringify(eventObj),'*');
},
```

   a. 传递页面高度数据

      资产对象选择页渲染前，会传递一个页面高度和宽度的数据给ROMA Exchange框架，用以调整弹窗的高度和宽度。

代码示例：

```
mounted () {
let div = document.getElementById("***AdaptorConfigWidget");
```

3. let eventObj = {
4. eventName: 'PAGELOAD_EVENT',
5. params: {
6. height: div.offsetHeight,
7. width: div.offsetWidth
8. }
9. };
10. parent.postMessage(JSON.stringify(eventObj),'*');
11. },

    a. 确定事件后传递数据

       选择资产对象之后单击确认按钮，适配器会给ROMA Exchange框架传递选择的资产对象数据

       代码示例：

       confirmFun () {

12. let eventObj = {
13. eventName: 'CONFIRM_EVENT',
14. params: {
15. sourceAttributes:{
16. items: this.items,
17. sourceId: this.sourceId,
18. assetData***: this.assetData***
19. }

    a. }

20. };
21. parent.postMessage(JSON.stringify(eventObj),'*');
22. },

    a. 取消事件后传递信息

       单击取消按钮后，适配器会给ROMA Exchange上架框架传递信息，以提示框架进行弹窗关闭操作。

       代码示例：

       cancelFun () {

23. let eventObj = {
24. eventName: 'CONCEL_EVENT',
25. params: {
26. message: 'Cancel!'

    a. }

27. };
28. parent.postMessage(JSON.stringify(eventObj),'*');
29. },

    a. 传递提示信息

当有警示信息或者异常情况时，适配器会传递提示信息给外层框架，以在外层展示提示信息。

代码示例;

messageFun () {

30. let eventObj = {
31. eventName: 'MESSAGE_EVENT',
32. params: {
33. message: 'Warning!'

a. }

34. };
35. parent.postMessage(JSON.stringify(eventObj),'*');
36. },

a. 接收数据

在触发事件传输事件之后，ROMA Exchange会传递资产来源的相关数据给适配器。

代码示例;

created() {
const DICT = {
DATA_EVENT: dataFun,
}
function dataFun (e) {
this.sourceId = e.sourceId
}
window.addEventListener("message", function (e) {
var event = JSON.parse(e.data)
DICT[event.eventName](event.params)
})

37. },

因此，适配器中资产对象选择页需要约定以下几个事件：

页面事件类型	**事件名称**	**事件解释**
Item消息发送事件	PARENT_EVENT	触发数据传输
	PAGELOAD_EVENT	发送高度
	CONFIRM_EVENT	确定
	CANCEL_EVENT	取消
	MESSAGE_EVENT	提示消息
Item消息接收事件	DATA_EVENT	接收数据

其他页面可以根据需求选择事件，但是事件名称应与上表中的一致。

4.2 后台

4.2.1 数据接口

接口详情请参考第五章节《 接口和样式参考.docx 》

1、**createSourceAttribute**

接口用于存储资产来源属性：资产上架页，订购页都可能用，比如Astro轻应用资产上架页用于存储用户鉴权信息。

2、**querySourceAttribute**

接口用于用户查询来源属性值，资产上架页，订阅页都可能用。

3、**filterSourceAttribute**

资产上架过滤已被使用的来源属性：以Astro轻应用资产为例，因为不允许同一个资产对象多次被上架，此接口用于过滤已被上架过的资产对象，资产上架页调用。

4、**querySubscriptionSourceAttribute**

接口用于业务用户根据订阅实例号查询商品和用户属性，订阅页调用，查询订阅自定义属性。同时对订阅实例号与当前用户信息进行校验

5、**notifySubscriptionStatus**

接口用于适配器通知订阅实例状态，订阅页调用。

变更：为适配多来源底座，在通知订阅状态时，新增sourceId参数作为某一来源下的订阅实例状态，用于区分多来源部署状态划分。

6、**recordLog**

记录接口日志及操作日志。

7、**uploadOBSObject**

存入OBS对象，适用于ServiceStage类来源部署时，存入私仓中。

8、**downloadOBSObject**

获取OBS对象，适用于ServiceStage类来源发布时，从私仓中下载。

9、**createOBSBucket**

创建OBS桶，适用于serviceStage类安装时使用，创建的OBS桶信息存储进用户级来源属性信息内，每次进行安装是，校验当前用户当前来源下的‘User’级属性信息内是否存在桶信息，存在则直接使用，不存在则创建OBS桶，将桶信息存储进当前来源User级来源属性信息内。

10、**fileDownload**

附件下载接口，屏蔽基线接口校验，直接从租户公仓下载资源。

11、**交互脚本（用于资产管理各个阶段调用适配器逻辑）**

命名规范：适配器名称__**Interaction**

注意：交互脚本命名取适配器命名，否则在安装适配器时无法自动获取脚本脚本，需手动添加适配器交互脚本数据。

格式定义参考：

①提交资产上架申请

exportfunction submitAsset(sourceAttributes:object, sourceId:string):void{

// 实现资产上架时校验或其他操作

}

② 撤销/驳回资产上架申请

exportfunction cancelAsset(sourceAttributes:object, sourceId:string):

void{

// 实现资产上架驳回时校验或其他操作

}

③ 资产下架成功

exportfunction unshelfAsset(sourceAttributes:object, sourceId:string):

void{

// 实现资产下架成功时校验或其他操作

}

④ 资产上架申请成功

exportfunction submitAssetSuccess(sourceAttributes:object, sourceId:string, subjectId:string):

object{

//实现资产上架申请成功是需要触发的操作，新增入参subjectId为来源用户，因审批人发生变更，无法获取发布者信息，故添加入参。

}

⑤资产提交订阅申请

exportfunction subscribeAsset(sourceAttributes:object, sourceId:string):

void{

// 资产提交订阅申请时校验或其他操作

}

⑥ 撤销/驳回资产订阅申请

exportfunction cancelSubscribeAsset(sourceAttributes:object, sourceId:string):

void{

//撤销/驳回资产订阅申请时校验或其他操作

}

⑦删除资产申请

```
exportfunction deleteAsset(sourceAttributes:object, sourceId:string):void{
// 实现资产删除时删除相关附件或其他操作
}
```

交互脚本方法存在返回值时，根据业务需要在基线触发交互处，接收返回数据，并进行处理操作。

7，**基线触发脚本：CP_DS__offeringPublishApplication**

交互方式：

```
//调用适配器交互方法
const reservedField2 = dataParse['offering']['reservedField2'];
const sourceAttributes = dataParse['sourceAttributes'];
if(reservedField2 && sourceAttributes){
const sources =newQueryAssetSourcesList().run({ id: reservedField2 }).sources;
if(sources.length ==0){
throw new I18nError('CP_DS__NotFoundSource');
}
const interactionScript = sources[0].interactionScript;
if(interactionScript
&&Object.keys(require(interactionScript)).indexOf('submitAssetSuccess')!=-1){
const sourceAttributesTmpl
=require(interactionScript).submitAssetSuccess(sourceAttributes, reservedField2,
owner);
if(sourceAttributesTmpl){
dataParse['sourceAttributes']= sourceAttributesTmpl;
}
}
}
```

此交互方式触发的适配器方法为submitAssetSuccess，并使用了该方法的返回值，回写进dataParse['sourceAttributes']。其余方法的触发调用与此类似，只需要变更需要触发的方法名称即可。

4.2.2 定制接口

1、查询允许授权的资产来源

POST /service/CP_DS__DigitalStoreService/1.0.1/asset-sources/auth-list

业务用户管理查看允许配置授权的资产来源的授权信息。

请求消息体

表 4-1 请求消息体参数说明

参数说明	数据类型	必选/可选	描述
skip	Number	O	起始页，默认0
limit	Number	O	每页记录数，默认10
adaptType	String	O	适配类型

响应消息体

参数说明	数据类型	必选/可选	描述
sources	Source[]	M	适配器
count	Number	M	来源总数

Source结构体

参数说明	数据类型	必选/可选	描述
id	String	M	来源ID
name	String	M	来源名称
description	String	O	来源描述
adapterParams	String	M	适配器的页面参数
configParams	String	M	适配器来源的配置参数
lastModifiedDate	DateTime	O	最后修改时间
authorized	Number	M	授权状态 1-授权 0-未授权

2、取消授权

POST /service/CP_DS__DigitalStoreService/1.0.1/asset-sources/cancel-auth

接口用于查询适配器来源是否已经被发布使用或存在订阅实例，如果被发布使用，不能取消授权。

取消授权操作是将CP_DS__AssetSourceAttribute表中subjectType为“User”的attributes值清空。

请求消息体

表 4-2 请求消息体参数说明

参数说明	数据类型	必选/可选	描述
id	String	M	

3、判断用户关联的适配器来源是否授权

POST /service/CP_DS__DigitalStoreService/1.0.1/asset-sources/auth-judge/{id}

查询用户关联的适配器来源，在CP_DS__AssetSourceAttribute表中subjectType为“User”的attributes值是否为空，不为空标识已经授权（true），反之，未授权（false）。

响应消息体

参数说明	数据类型	必选/可选	描述
isAuth	Boolean	M	是否授权

4.3 参考样例

4.3.1 StaticApplication 适配器开发样例

以StaticApplication适配器开发流程为例：

整体流程：

用户A进入资产上架页面 --> 选择本地应用 --> 基本信息配置中交付方式 --> 选择在线开通 -->上传定义文件（StaticApplication软件包） --> 用户A发布上架 -->管理台上架审批-->通过a / 不通过b

a--> 后台生成适配器来源信息，安装StaticApplication软件包。

b--> 正常驳回业务，后台不生成适配器来源信息，不安装StaticApplication软件包。

具体流程和接口调用：

首先用户进入资产发布页，交付方式选择在线开通，上传定义文件（StaticApplication软件包）

1， **upload**接口用于上架StaticApplication软件包，fileName为软件包名称scene上传主场景为Offering，subScene上传子场景为AdapterPack。其中fileName上传的适配器包名称需要遵循格式：{application}_{siteId}.zip，否则上传接口报错如下：

2， offerings保存草稿接口或submit发布接口，入参报文中offering结构体offeringAttribute集合中属性设置：


```
],
{
        "attributeCode":"deliveryMode_0Gba000000mwYjyXmIIC",
        "attributeValue":"onlineProvisioning",
        "Status":"Valid"
],

                {
                        "attributeCode":"localApplicationFile_0Gba000000mwYjyXmIIC",
                        "attributeValue":"{\"assetDisplay\":[\"StaticApplicationDemo-
0.1.3_001S000000nm75BwY448.zip\"],\"fileName\":\"StaticApplicationDemo-0.1.3_001S000000nm75BwY448.zip\",\"items\":
[\"StaticApplicationDemo-0.1.3_001S000000nm75BwY448.zip\"],\"zipFileId\":\"0Gh0000000p0EIOJitv6\"}",
                        "Status":"Valid"
```


其中：attributeCode为localApplicationFile_0Gba000000mwYjyXmIIC时

attributeValue约定形式为：

{ "assetDisplay":[appName], "fileName": appName, "items": [appName], "zipFileId":上传软件包后台附件id。调用**upload**接口返回值}

3，管理台审批通过后，后台安装上传的软件包。

5，用户订阅资产，适配器将保存一份订阅者信息存起来用于安装时候展示。

6， 安装/部署已订阅资产，用户单击部署触发软件包适配器内部业务**展示部署页面**