

CodeArts IDE

用户指南

文档版本 01
发布日期 2024-04-12



版权所有 © 华为技术有限公司 2024。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。

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：

<https://www.huawei.com/cn/psirt/vul-response-process>

如企业客户须获取漏洞信息，请参见如下网址：

<https://securitybulletin.huawei.com/enterprise/cn/security-advisory>

目录

1 CodeArts IDE 最佳实践汇总	1
2 基于 CodeArts IDE 快速创建简单的 C++ 工程	2
2.1 使用 CodeArts IDE for C/C++ 开发 OpenGL 示例工程.....	2
3 基于 CodeArts IDE 快速创建简单的 Java 工程	6
3.1 使用 CodeArts IDE for Java 开发简单的 Java 工程.....	6

1 CodeArts IDE 最佳实践汇总

本文汇总了基于CodeArts IDE常见应用场景的操作实践，为每个实践提供详细的方案描述和操作指导。

表 1-1 CodeArts IDE 最佳实践一览表

最佳实践	说明
基于CodeArts IDE快速创建简单的C++工程	本实践向您介绍如何基于CodeArts IDE快速创建简单的C++工程，CodeArts IDE面向开发者提供的智能化可扩展桌面集成开发环境，结合行业和产业开发套件，实现一站式开发体验。
基于CodeArts IDE快速创建简单的Java工程	本实践向您介绍如何基于CodeArts IDE快速创建简单的Java工程，CodeArts IDE for Java是一个JAVA集成开发环境，将文本编辑器和强大的开发工具（如智能代码补全、导航、重构和调试）集成在一起。

2 基于 CodeArts IDE 快速创建简单的 C++工程

2.1 使用 CodeArts IDE for C/C++ 开发 OpenGL 示例工程

1. 功能介绍

CodeArts IDE面向开发者提供的智能化可扩展桌面集成开发环境（IDE），结合行业和产业开发套件，实现一站式开发体验。

- 编码新体验，开发更高效：内置自研C/C++语言开发支持，提供全新的工程加载、语法着色、符号解析、编码重构和运行调试等开发体验，提升开发效率。
- 能力可扩展，生态更开放：支持基于插件的能力扩展，开放的插件标准，开源的插件框架，开放的插件市场，形成更加开放的生态系统。
- 界面可裁剪，体验更优质：支持基于组件的界面剪裁，在精简模式下形成专用工具的优质体验，又可以在需要时升级为全模式的全量IDE工具。

CodeArts IDE for C/C++：基于C/C++语言开发CMake工程，并通过CodeArts IDE完成从工程创建、代码编写、运行调试到发布测试的全过程。基于插件扩展可以将个人开发者作业流集成其中，实现从需求到提交的全部过程，更可在业务中集成提供的的诸多能力，提升应用交付效率。

本实验将指导开发者通过CodeArts IDE for C/C++平台，在本地桌面快速开发一个基于Qt实现的简单项目。通过本实验您将体验到：

- 如何在CodeArts IDE for C/C++上进行基于CMake项目的本地编译构建；
- 在CodeArts IDE上调试和运行；
- 实现一个简单OpenGL demo。

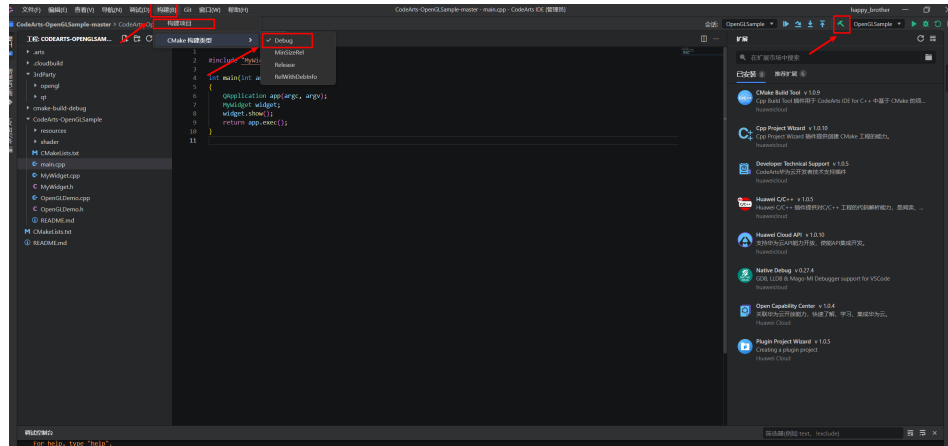
2. 前置条件

- 安装CodeArts IDE for C/C++。
- [示例工程代码获取](#)。

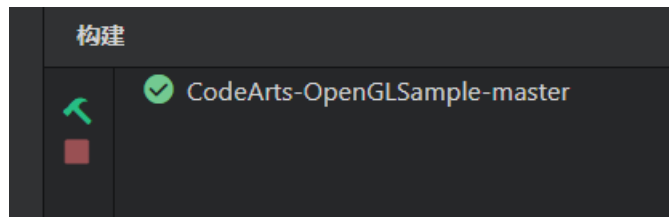
3. 编译构建与运行调试

通过CodeArts IDE客户端以文件夹的形式打开代码包。

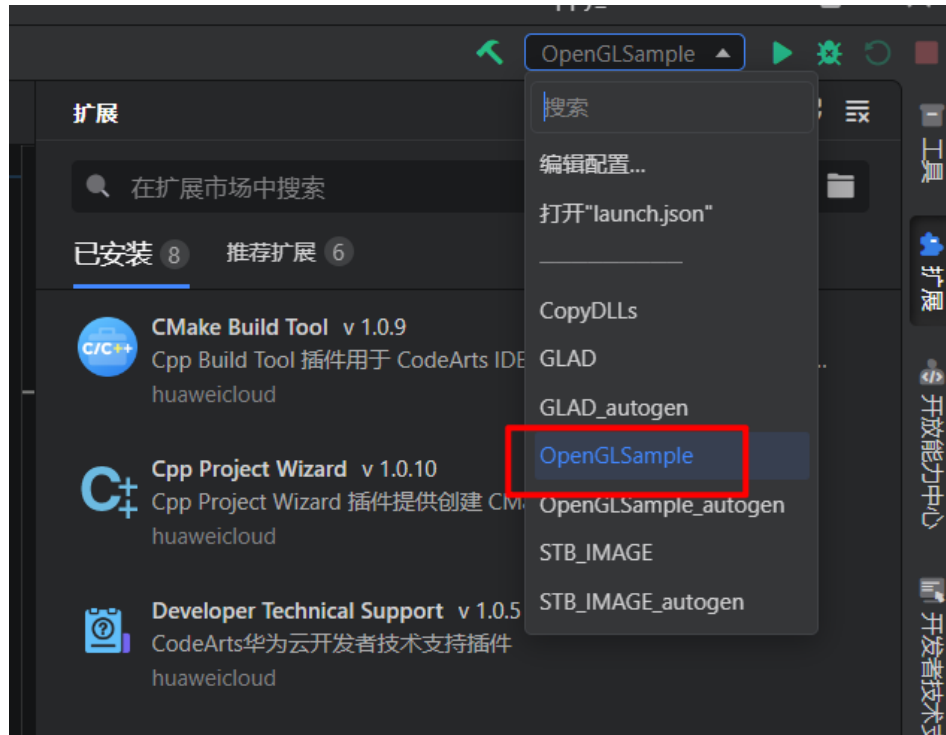
步骤1 在本地CodeArts IDE页面，单击“构建”，选择“CMake 构建类型 > Debug”。



步骤2 然后在弹窗中选择“all META”，当编译构建成功后，日志打印finished，并显示成功的图示。



步骤3 构建成功后，在页面的右上角选择调试的目标，然后进行调试和运行。



---结束

4. 关键代码片段

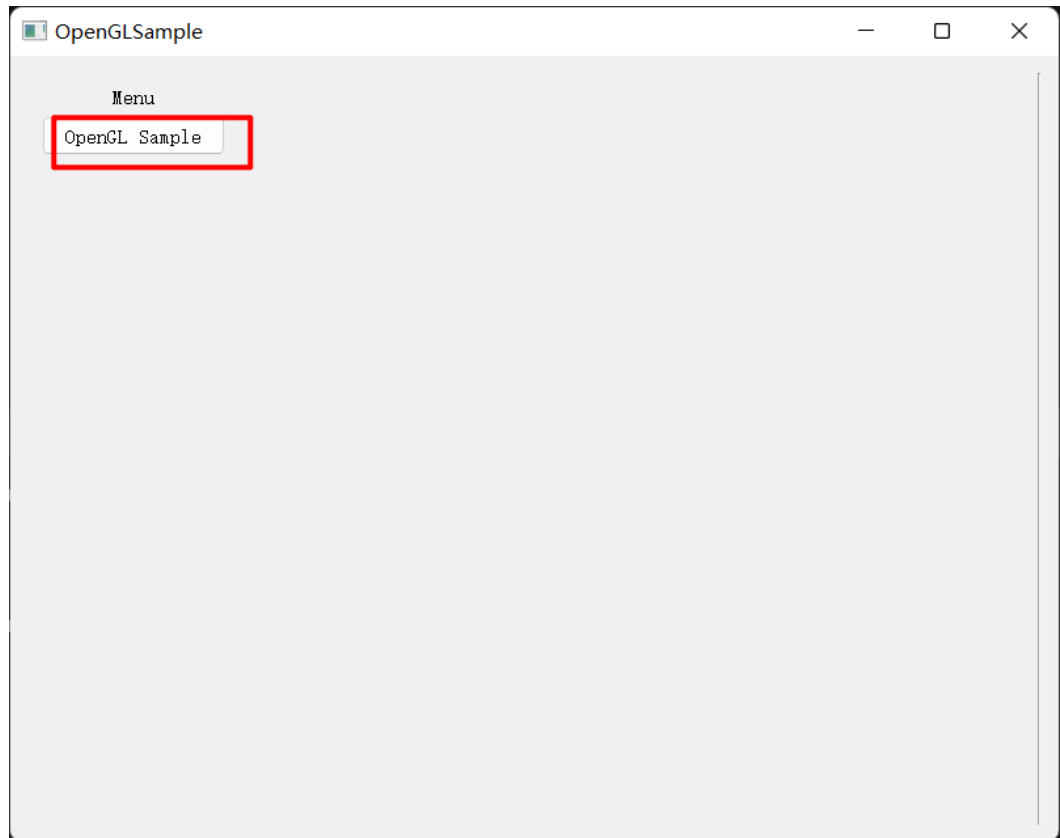
```
// MyWidget类的构造函数，初始化一些成员变量并设置布局
MyWidget::MyWidget(QWidget *parent) : QWidget(parent)
{
    resize(800, 600); // 设置窗口大小为800*600

    // 创建一个垂直布局，用于放置菜单标题和按钮
    m_menuButtonLayout = new QVBoxLayout;
    m_menuButtonLayout->setAlignment(Qt::AlignTop); // 设置布局的对齐方式为顶部对齐
    // 创建一个标签，设置文本为"Menu"，并设置其居中对齐
    m_menuTitle = new QLabel(this);
    m_menuTitle->setText("Menu");
    m_menuTitle->setAlignment(Qt::AlignCenter);
    // 创建一个按钮，设置文本为"OpenGL Sample"，并设置其固定大小
    m_openGLDemoButton = new QPushButton("OpenGL Sample", this);
    m_openGLDemoButton->setFixedSize(140, 30);
    // 将标签和按钮添加到垂直布局中
    m_menuButtonLayout->addWidget(m_menuTitle);
    m_menuButtonLayout->addWidget(m_openGLDemoButton);
    // 创建一个新的窗口部件，并设置其布局为上面创建的垂直布局
    m_menuWidget = new QWidget(this);
    m_menuWidget->setLayout(m_menuButtonLayout);
    // 创建一个帧，设置其形状为垂直线并设置阴影效果
    QFrame* line = new QFrame(this);
    line->setFrameShape(QFrame::VLine);
    line->setFrameShadow(QFrame::Sunken);
    // 创建一个水平布局，用于放置菜单窗口部件和帧
    QHBoxLayout *mainLayout = new QHBoxLayout;
    mainLayout->addWidget(m_menuWidget, 1); // 将菜单窗口部件添加到布局中，并设置其伸缩因子为1
    mainLayout->addWidget(line); // 将帧添加到布局中
    setLayout(mainLayout); // 将主布局设置为窗口的布局
    // 连接按钮的单击信号到槽函数ShowOpenGLDemo
    connect(m_openGLDemoButton, &QPushButton::clicked, this, &MyWidget::ShowOpenGLDemo);
}
// 槽函数，用于显示OpenGL演示
void MyWidget::ShowOpenGLDemo()
```

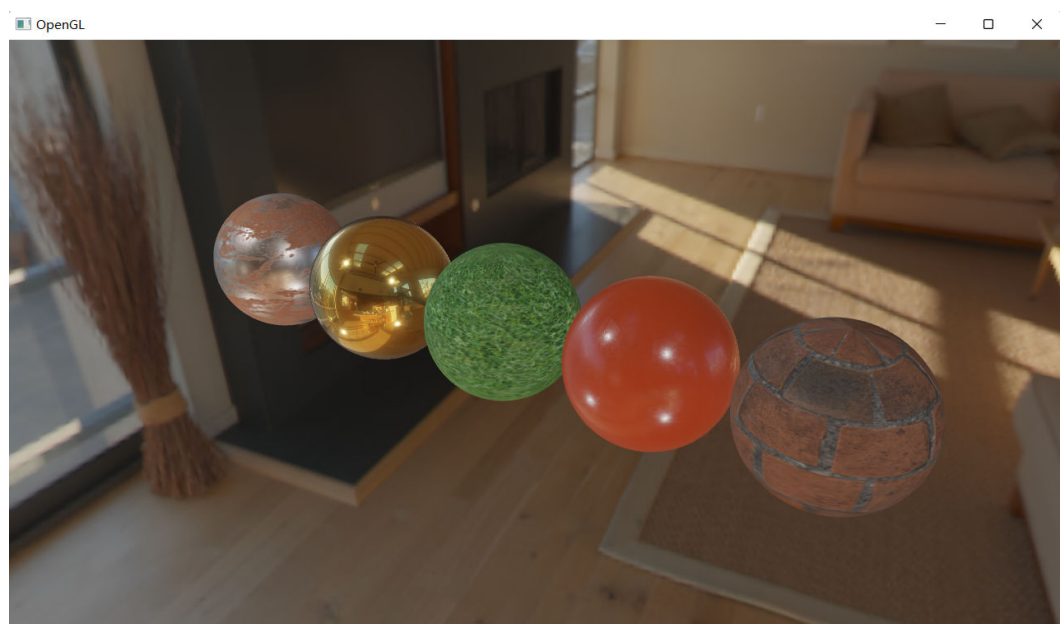


```
{  
    OpenGLDemoShow(); // 调用OpenGL演示显示函数  
    return;  
}
```

5. 运行结果



单击“OpenGL Sample”，结果如下所示：



3 基于 CodeArts IDE 快速创建简单的 Java 工程

3.1 使用 CodeArts IDE for Java 开发简单的 Java 工程

1. 功能介绍

CodeArts IDE for Java是一个JAVA集成开发环境，将文本编辑器和强大的开发工具（如智能代码补全、导航、重构和调试）集成在一起。

- 调试热替换：支持在调试模式下运行程序。当程序挂起时，在“运行”和“调试”视图中检查其输出。找到错误并更正它，然后重新运行程序。在Java上下文中，可通过热代码替换功能来动态修改和重新加载类，无需停止调试会话。
- 强大的编码辅助能力：CodeArts IDE for Java 可以深入理解代码，提供上下文感知的机器学习辅助补全功能，可以补全单条语句或整行代码。对于代码编辑器中的任何符号，开发者可以查看其定义和相关文档。集成开发环境可高亮显示错误，并让开发者直接在代码编辑器中对错误采取行动。“问题”视图会列出当前打开的文件中检测到的所有问题。
- Java构建工具集成：CodeArts IDE for Java 提供对 Maven 和 Gradle 构建系统的内置支持，包括项目和依赖关系解析，以及运行 Maven 目标和 Gradle 任务的能力。

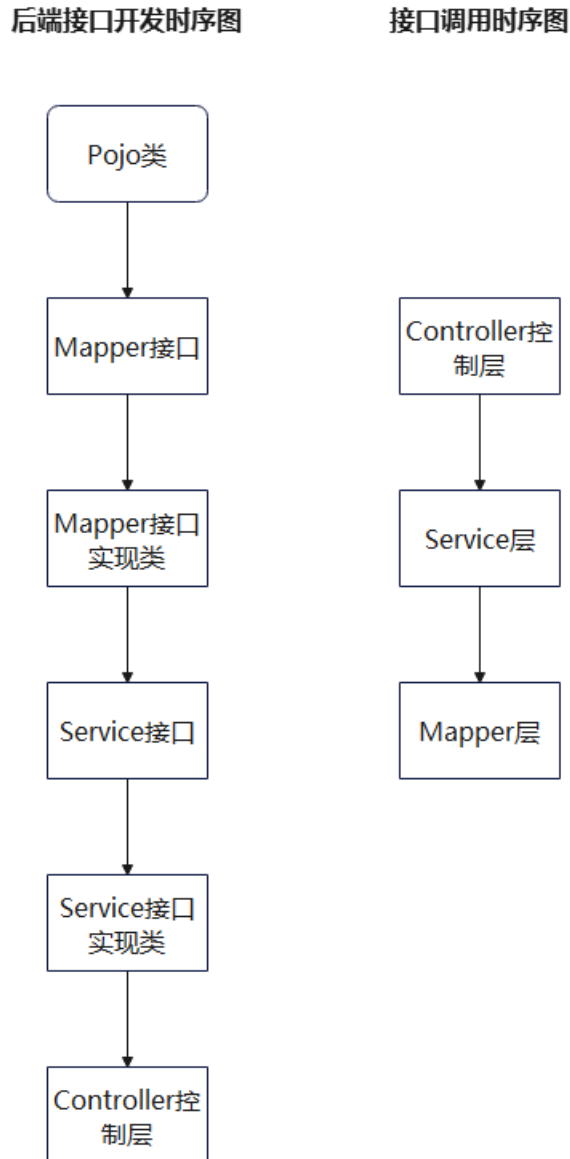
Spring Boot和Thymeleaf是目前非常流行的Java开发框架和模板引擎。Spring Boot提供了快速构建Web应用程序的能力，而Thymeleaf则是一种功能强大且易于使用的模板引擎，可以帮助在服务器端生成动态的HTML页面。在本文中，将使用Spring Boot和Thymeleaf来实现一个完整的用户管理功能，包括增加、删除、修改和查询用户信息。以及使用CodeArts IDE for Java构建，运行Java工程，并且学到Java语言的一些基础知识，例如Java的一些基本类型。通过Java语言创建，读取，并修改本地文件的能力。

2. 前置条件

- 下载并安装CodeArts IDE for Java，了解IDE的基础功能。
- [示例工程代码获取](#)。
- Java语言基础。

- 安装Java JDK 1.8及其以上版本。
- 引入spring-boot-starter-web、spring-boot-starter-thymeleaf和lombok（可选）相关依赖。

3. 开发时序图



4. 关键代码片段

4.1 后端实现过程

pojo类：（@Data是lombok注解，不需要写get、set方法了。）

```
@Data
public class User {
    private String uuid;
    private Integer id;
    private String name;
    private Integer age;
```

```
private String department;
public User(){
public User(Integer id, String name, Integer age, String department) {
    this.uuid = UUID.randomUUID().toString();
    this.id = id;
    this.name = name;
    this.age = age;
    this.department = department;
}
}
```

mapper接口:

```
public interface IUserMapper {
    public List<User> getUserList();
    public boolean addUser(User user);
    public boolean updateUser(User user);
    public boolean deleteUser(String uuid);
}
```

mapperImpl实现类:

```
@Component
public class UserMapperImpl implements IUserMapper {
    @Override
    public List<User> getUserList() {
        return UserData.userList;
    }
    @Override
    public boolean addUser(User user) {
        user.setUuid(UUID.randomUUID().toString());
        return UserData.userList.add(user);
    }
    @Override
    public boolean updateUser(User user) {
        Stream<User> userStream = UserData.userList.stream().filter(userItem -> {
            if (user.getId().equals(userItem.getId())) {
                if (user.getId() != null) {
                    userItem.setId(user.getId());
                }
                if (!StringUtils.isEmpty(user.getName())) {
                    userItem.setName(user.getName());
                }
                if (!StringUtils.isEmpty(user.getDepartment())) {
                    userItem.setDepartment(user.getDepartment());
                }
                if (!StringUtils.isEmpty(user.getAge())) {
                    userItem.setAge(user.getAge());
                }
            }
            return true;
        });
        return false;
    }
    return userStream.count() >= 1;
}
    @Override
    public boolean deleteUser(String uuid) {
        int sizeBeforeDelete = UserData.userList.size();
        List<User> deletedUsers = UserData.userList.stream().filter(user ->
        uuid.equals(user.getUuid())).collect(Collectors.toList());
        deletedUsers.forEach(UserData.userList::remove);
        return sizeBeforeDelete != UserData.userList.size();
    }
}
```

service接口:

```
public interface IUserService {
    public List<User> getUserList();
    public boolean addUser(User user);
}
```

```
public boolean updateUser(User user);  
public boolean deleteUser(String id);  
}
```

serviceImpl实现类:

```
@Service  
public class UserServiceImpl implements IUserService {  
    @Autowired  
    IUserMapper userMapperImpl;  
    @Override  
    public List<User> getUserList() {  
        return userMapperImpl.getUserList();  
    }  
    @Override  
    public boolean addUser(User user) {  
        return userMapperImpl.addUser(user);  
    }  
    @Override  
    public boolean updateUser(User user) {  
        return userMapperImpl.updateUser(user);  
    }  
    @Override  
    public boolean deleteUser(String id) {  
        return userMapperImpl.deleteUser(id);  
    }  
}
```

controller控制层: 用户管理接口Controller:

```
@RestController  
@RequestMapping("/user")  
public class UserController {  
    @Autowired  
    IUserService userServiceImpl;  
    @GetMapping("/getUserList")  
    public List<User> getUserList() {  
        return userServiceImpl.getUserList();  
    }  
    @PostMapping("/addUser")  
    public boolean addUser(User user) {  
        return userServiceImpl.addUser(user);  
    }  
    @PostMapping("/updateUser")  
    public boolean updateUser(User user) {  
        return userServiceImpl.updateUser(user);  
    }  
    @PostMapping("/deleteUser")  
    public boolean deleteUser(String uuid) {  
        return userServiceImpl.deleteUser(uuid);  
    }  
}
```

用户欢迎界面Controller:

```
@Controller  
public class WelcomePageController {  
    @RequestMapping("/")  
    public String user() {  
        return "user";  
    }  
}
```

用户数据:

```
public class UserData {  
    public static List<User> userList = new ArrayList<>();  
  
    public static void initUserData() {  
        userList.add(new User(1, "张三", 20, "人力资源部"));  
    }  
}
```

```
        userList.add(new User(2, "李四", 18, "后勤部"));
        userList.add(new User(3, "王五", 21, "研发部"));
        userList.add(new User(4, "赵六", 25, "产品部"));
    }
}
```

启动类：（每次启动调试需要初始化用户数据）

```
@SpringBootApplication
public class UserManagerApplication {

    public static void main(String[] args) {
        UserData.initUserData(); // 初始化用户数据
        SpringApplication.run(UserManagerApplication.class, args);
    }
}
```

4.2 前端实现过程(thymeleaf模板+html+ajax)

user.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>User Manager</title>
    <script src="https://cdn.bootcdn.net/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
    <script>
        $(function () {
            queryData();
        });

        function queryData() {
            // 发送 AJAX 请求
            $.ajax({
                url: 'http://localhost:8080/user/getUserList',
                type: 'GET',
                dataType: 'json',
                success(res) {
                    var html = "";
                    for (var i = 0; i < res.length; i++) {
                        html += '<tr><td>' + res[i].id + '</td>' + '<td>' + res[i].name + '<td>' + res[i].age + '<td>' +
res[i].department + '</td><td> <button onclick="del(\' + res[i].uuid + '\')">删除</button></td></tr>';
                    }
                    // console.log(html);
                    $('#tt').html(html);
                },
                error: function (xhr, status, error) {
                    console.error(error); // 处理错误情况
                }
            });
        }

        function add() {
            const id = document.getElementById('InputId').value;
            const name = document.getElementById('InputName').value;
            const age = document.getElementById('InputAge').value;
            const department = document.getElementById('InputDepartment').value;

            if (!validateId(id)) {
                return;
            }

            $.ajax({
                url: `http://localhost:8080/user/addUser?id=${id}&name=${name}&age=${age}&department=${
department}`,
                type: 'post',
                dataType: 'json',
            });
        }
    </script>
</head>

<body>
    <div style="text-align: center;">
        <h2>用户管理</h2>
        <hr/>
        <div style="display: flex; justify-content: space-around; align-items: center;">
            <div style="border: 1px solid #ccc; padding: 5px; width: 150px; text-align: center;">
                添加用户
            </div>
            <div style="border: 1px solid #ccc; padding: 5px; width: 150px; text-align: center;">
                删除用户
            </div>
        </div>
        <hr/>
        <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;">
            <thead>
                <tr>
                    <th>ID</th>
                    <th>姓名</th>
                    <th>年龄</th>
                    <th>部门</th>
                    <th>操作</th>
                </tr>
            </thead>
            <tbody>
                <tr>
                    <td>1</td>
                    <td>张三</td>
                    <td>20</td>
                    <td>研发部</td>
                    <td><button onclick="del('1')">删除</button></td>
                </tr>
            </tbody>
        </table>
        <div style="margin-top: 20px; text-align: center;">
            <input type="text" value="ID" style="width: 50px; margin-right: 10px;"/>
            <input type="text" value="姓名" style="width: 50px; margin-right: 10px;"/>
            <input type="text" value="年龄" style="width: 50px; margin-right: 10px;"/>
            <input type="text" value="部门" style="width: 50px; margin-right: 10px;"/>
            <input type="button" value="添加" style="margin-right: 10px;"/>
            <input type="button" value="删除" style="margin-right: 10px;"/>
        </div>
    </div>
</body>
</html>
```

```
        success: function (res) {
            queryData();
            if (res == true) {
                alert('添加成功');
            } else {
                alert('添加失败');
            }
        },
        error: function (xhr, status, error) {
            console.error(error);
        }
    });
    document.getElementById('InputId').value = "";
    document.getElementById('InputName').value = "";
    document.getElementById('InputAge').value = "";
    document.getElementById('InputDepartment').value = "";
}

function update() {
    const id = document.getElementById('updateInputId').value;
    const name = document.getElementById('updateInputName').value;
    const age = document.getElementById('updateInputAge').value;
    const department = document.getElementById('updateInputDepartment').value;

    if (!validateId(id)) {
        return;
    }

    $.ajax({
        url: `http://localhost:8080/user/updateUser?id=${id}&name=${name}&age=${age}&department=${department}`,
        type: 'post',
        dataType: 'json',
        success: function (res) {
            queryData();
            if (res == true) {
                alert('修改成功');
            } else {
                alert('修改失败');
            }
        },
        error: function (xhr, status, error) {
            console.error(error);
        }
    });
    document.getElementById('updateInputId').value = "";
    document.getElementById('updateInputName').value = "";
    document.getElementById('updateInputAge').value = "";
    document.getElementById('updateInputDepartment').value = "";
}

function del(uuid) {
    $.ajax({
        url: `http://localhost:8080/user/deleteUser?uuid=${uuid}`,
        type: 'post',
        dataType: 'json',
        success: function (res) {
            queryData();
            if (res == true) {
                alert('删除成功');
            } else {
                alert('删除失败');
            }
        },
        error: function (xhr, status, error) {
            console.error(error);
        }
    });
}
```

```
function validateId(id) {
  if (id == "") {
    alert("Id值不能为空");//空值校验
    return false;
  }
  return true;
}
</script>
</head>
<body>
  <div class="header">
    <!--<button onclick="queryData()">查询</button-->
    <div class="items">
      <input type="number" value="" placeholder="请输入id" id="InputId" />
      <input type="text" value="" placeholder="请输入名称" id="InputName">
      <input type="number" value="" placeholder="请输入年龄" id="InputAge">
      <input type="text" value="" placeholder="请输入部门" id="InputDepartment">
      <button onclick="add()">添加</button>
    </div>
    <div class="items">
      <input type="number" value="" placeholder="请输入id" id="updateInputId" />
      <input type="text" value="" placeholder="请输入名称" id="updateInputName">
      <input type="number" value="" placeholder="请输入年龄" id="updateInputAge">
      <input type="text" value="" placeholder="请输入部门" id="updateInputDepartment">
      <button onclick="update()">修改</button>
    </div>
  </div>
  <div class="table">
    <table>
      <thead>
        <tr>
          <th>用户Id</th>
          <th>姓名</th>
          <th>年龄</th>
          <th>部门</th>
          <th>操作</th>
        </tr>
      </thead>
      <tbody id="tt">
      </tbody>
    </table>
  </div>
</body>
<style>
  table {
    margin: 20px;
    border-collapse: collapse;
  }

  th {
    background-color: #cccccc;
  }

  tr {
    text-align: center;
    border: 1px solid #ccc;
  }

  td {
    padding: 20px;
    border: 1px solid #ccc;
  }

  button {
    background-color: white;
    width: 70px;
    border-radius: 20px;
  }
</style>
```



```
border: 1px solid #ccc;
}

button :hover {
  background-color: #cccccc;
}

.header {
  display: flex;
  justify-content: center;
  flex-wrap: wrap;
}

.items {
  display: grid;
  margin-left: 10px;
}

.table {
  display: flex;
  justify-content: center;
}
</style>
</html>
```

4.3 相关配置文件:

相关的依赖信息如下，引入spring-boot-starter-web、spring-boot-starter-thymeleaf和lombok（可选）相关依赖：

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>
  <dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <scope>annotationProcessor</scope>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-thymeleaf</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
</dependencies>
```

application.properties:

```
spring.application.name=user-manager
server.port=8080
```

application.yml

```
thymeleaf:
  prefix:
    classpath: /templates # 访问template下的html文件需要配置模板，映射
```

4.4 代码解析

在html代码中,主要利用onclick单击事件和jquery框架中的ajax交互。 jquery中ajax的语法格式:

```
$.ajax({  
  url: 接口名,  
  type: 'get',  
  dataType: 'json',  
  success: function(res) {  
    queryData();  
  },  
  error: function(xhr, status, error) {  
    console.error(error); // 处理错误情况  
  }  
});
```

5. 输出结果示例

单击“构建”并运行，实现截图如下。



The screenshot shows a web browser at localhost:8080. At the top, there are two forms for adding and modifying data. Each form has four input fields: '请输入id', '请输入名称', '请输入年龄', and '请输入部门'. Below each form is a button: '添加' (Add) and '修改' (Modify). Below the forms is a table with the following data:

用户Id	姓名	年龄	部门	操作
1	张三	20	人力资源部	删除
2	李四	18	后勤部	删除
3	王五	21	研发部	删除
4	赵六	25	产品部	删除