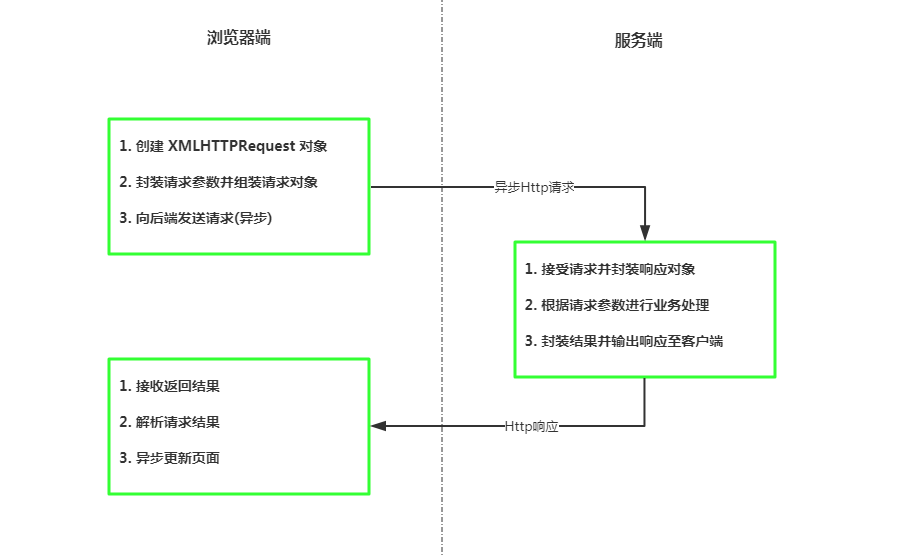
**Ajax 简介**

AJAX = Asynchronous JavaScript and XML（异步的 JavaScript 和 XML），它是一种用于创建快速动态网页的技术，通过浏览器与服务器进行少量数据交换，Ajax 可以使网页实现异步更新，这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新，传统的网页如果需要更新内容，必须要进行跳转并重新加载整个网页。

Ajax 技术使得网站与用户间有了更友好的交互效果，比较常见的借助 Ajax 技术实现的功能有列表上拉加载分页数据、省市区三级联动、进度条更新等等，这些都是借助 Ajax 技术在当前页面即可完成的功能，即使有数据交互也不会跳转页面，整体交互效果有了很大的提升。

**Ajax 工作流程**



Ajax 的整个工作流程如上图所示，用户在进行页面上进行操作时会执行 js 方法，js 方法中通过 Ajax 异步与后端进行数据交互。首先会创建 XMLHttpRequest 对象，XMLHttpRequest 是 AJAX 的基础，之后会根据页面内容将参数封装到请求体中或者放到请求 URL 中，然后向后端服务器发送请求，请求成功后根据后端返回的数据进行解析和部分逻辑处理，最终在不刷新页面的情况下对页面进行局部更新。

**Ajax 调用实践**

在 resources/static 下新建 ajax-test.html，代码如下：

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<title> Ajax 请求测试</title>

</head>

<body class="hold-transition login-page">

<div style="width:720px;margin:7% auto">

<div class="content">

<div class="container-fluid">

<div class="row">

<div class="col-lg-6">

<div class="card">

<div class="card-header">

<h5 class="m-0">接口测试1</h5>

</div>

<div class="card-body">

<input

type="text"

id="info"

class="form-control"

placeholder="请输入info值"

/>

<h6 class="card-title">接口1返回数据如下：</h6>

<p class="card-text" id="test1"></p>

<a href="#" class="btn btn-primary" onclick="requestTest1()"

>发送请求1</a

>

</div>

</div>

<br />

<div class="card card-primary card-outline">

<div class="card-header">

<h5 class="m-0">接口测试2</h5>

</div>

<div class="card-body">

<h6 class="card-title">接口2返回数据如下：</h6>

<p class="card-text" id="test2"></p>

<a href="#" class="btn btn-primary" onclick="requestTest2()"

>发送请求2</a

>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</body>

</html>

ajax-test 页面上分别定义了两个按钮，在点击时分别会分别触发 onclick 点击事件，在点击事件的实现逻辑中进行页面内容的局部更新。

**Ajax 调用逻辑**

由于原生 Ajax 实现比较繁琐，实际开发中一般都会使用 jQuery 封装的 Ajax 方法，或者其他库封装的 Ajax 方法。

在调用前首先在 html 代码中引入 jQuery 库，之后根据点击事件分别定义两个事件触发的 js 方法：requestTest1() 和 requestTest2()，在方法中分别使用 Ajax 向后端发送请求，在请求成功后将响应结果赋值到对应的 div 中，代码如下：

<!-- 引入jQuery -->

<script src="https://cdn.staticfile.org/jquery/1.12.0/jquery.min.js"></script>

<!-- 定义两个点击事件并实现 Ajax 调用逻辑 -->

<script type="text/javascript">

function requestTest1() {

var info = $('#info').val();

$.ajax({

type: 'GET', //方法类型

dataType: 'text', //预期服务器返回的数据类型

url: 'api/test1?info=' + info, //请求地址

contentType: 'application/json; charset=utf-8',

success: function (result) {

//请求成功后回调

$('#test1').html(JSON.stringify(result));

},

error: function () {

//请求失败后回调

$('#test1').html('接口异常，请联系管理员！');

},

});

}

function requestTest2() {

$.ajax({

type: 'GET', //方法类型

dataType: 'json', //预期服务器返回的数据类型

url: 'api/test2',

contentType: 'application/json; charset=utf-8',

success: function (result) {

$('#test2').html(JSON.stringify(result));

},

error: function () {

$('#test2').html('接口异常，请联系管理员！');

},

});

}

</script>

方法一中会首先获取用户在 input 框中输入的字段，之后将其拼接到请求的 URL 中，最后发送 Ajax 请求并完成回调，方法二中也是类似，用户点击发送请求的按钮后，会触发 onclick 点击事件并调用 requestTest2() 方法，在请求完成后进入 success 回调方法，并将请求结果的内容放到 div 中显示。

**后端接口实现**

**Controller包下面创建**AjaxRequestController

@RestController  
@RequestMapping*(*"/api"*)*public class AjaxRequestController *{* @RequestMapping*(*value = "/test1",method = RequestMethod.*GET)* public String test1*(*String info*)  
 {* if *(*StringUtils.*isEmpty(*info*))* return "输入info";  
 return "你输入的内容是："+info;  
 *}* @RequestMapping*(*value = "/test2",method = RequestMethod.*GET)* public List*<*Users*>* test2*()  
 {* ArrayList*<*Users*>* list = new ArrayList*<>()*;  
 Users user1 = new Users*()*;  
 user1.setId*(*1*)*;  
 user1.setName*(*"十一"*)*;  
 user1.setPassword*(*"12121"*)*;  
 Users user2 = new Users*()*;  
 user2.setId*(*2*)*;  
 user2.setName*(*"十二"*)*;  
 user2.setPassword*(*"21212"*)*;  
 Users user3 = new Users*()*;  
 user3.setId*(*3*)*;  
 user3.setName*(*"十三"*)*;  
 user3.setPassword*(*"31313"*)*;  
 list.add*(*user1*)*;  
 list.add*(*user2*)*;  
 list.add*(*user3*)*;  
 return list;  
 *}  
}*

控制器中分别实现两个方法并设置请求地址映射来处理前端发送的异步请求，test1() 方法中会根据前端传入的 info 字段值重新返回给前端一个字符串，请求方式为 GET，test2() 方法中会返回一个集合对象给前端，请求方式为 GET。

在前后端对接时需要确定好请求路径、请求方法、返回结果的格式等，比如后端定义的请求方法为 GET，那么在 Ajax 设置时一定要将 type 设置为 GET，不然无法正常的发起请求，test1() 方法和 test2() 方法中返回的结果格式也不同，所以在前端进行 Ajax 调用时 dataType 也需要注意，比如本案例中分别返回的是字符串类型和集合类型，那么在 Ajax 请求中需要将 dataType 分别设置为 text 和 json，不然前端在 Ajax 调用后会直接进入 error 回调中，而不是 success 成功回调中。

**测试**