AJAX 原理

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

目录

Contents

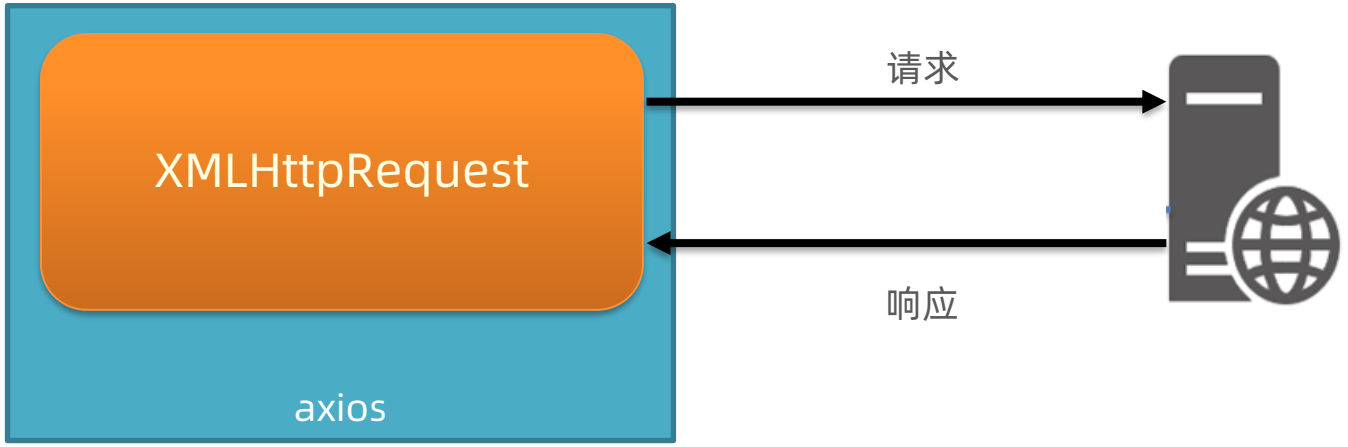
- XMLHttpRequest
- Promise
- 封装简易版 axios
- 案例 - 天气预报

AJAX原理 - XMLHttpRequest

定义：

> `XMLHttpRequest`（XHR）对象用于与服务器交互。通过 XMLHttpRequest 可以在不刷新页面的情况下请求特定 URL，获取数据。这允许网页在不影响用户操作的情况下，更新页面的局部内容。`XMLHttpRequest` 在 AJAX 编程中被大量使用。

关系：axios 内部采用 XMLHttpRequest 与服务器交互



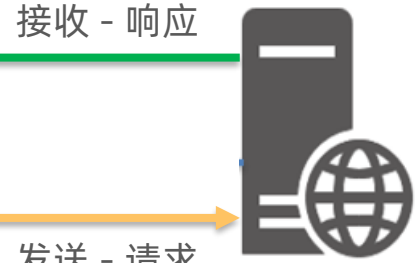
好处：掌握使用 XHR 与服务器进行数据交互，了解 axios 内部原理

使用 XMLHttpRequest

步骤：

1. 创建 XMLHttpRequest 对象
2. 配置请求方法和请求 url 地址
3. 监听 loadend 事件，接收响应结果
4. 发起请求

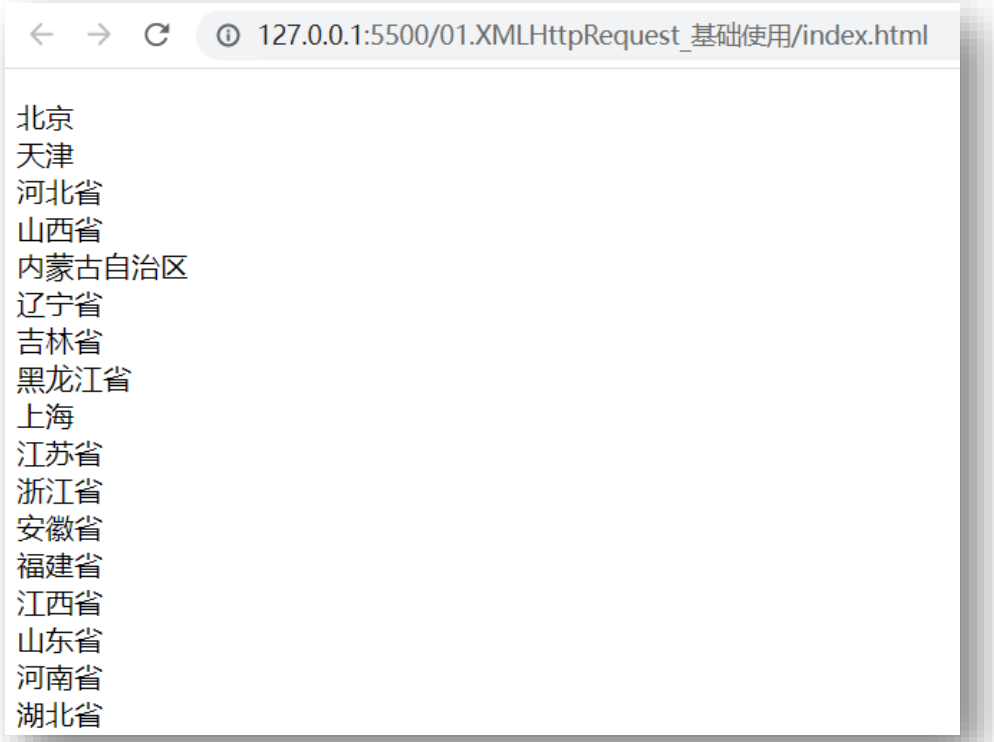
```
const xhr = new XMLHttpRequest()
xhr.open('请求方法', '请求url网址')
xhr.addEventListener('loadend', () => {
    // 响应结果
    console.log(xhr.response)
})
xhr.send()
```

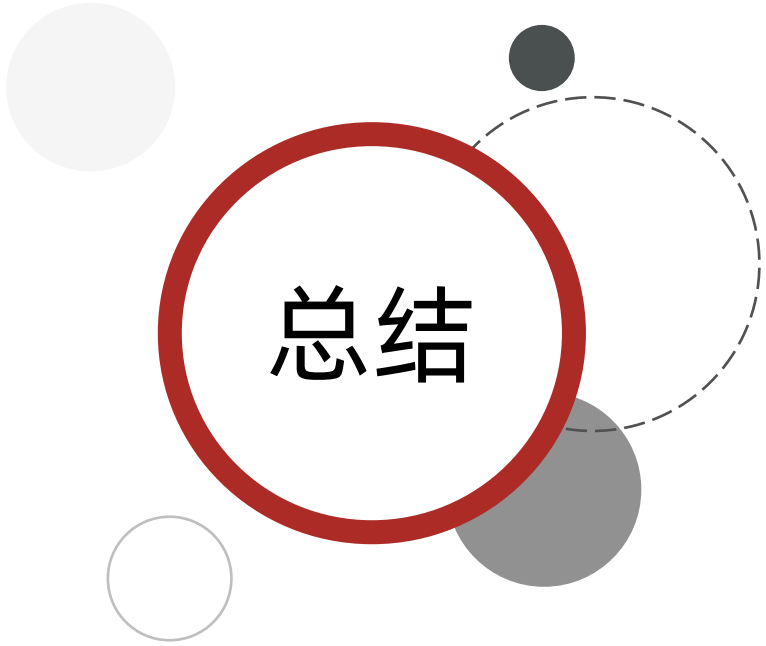


使用 XMLHttpRequest

需求：获取并展示所有省份名字

```
const xhr = new XMLHttpRequest()
xhr.open('GET', 'http://hmajax.itheima.net/api/province')
xhr.addEventListener('loadend', () => {
    console.log(xhr.response)
    // 对响应结果做后续处理
})
xhr.send()
```





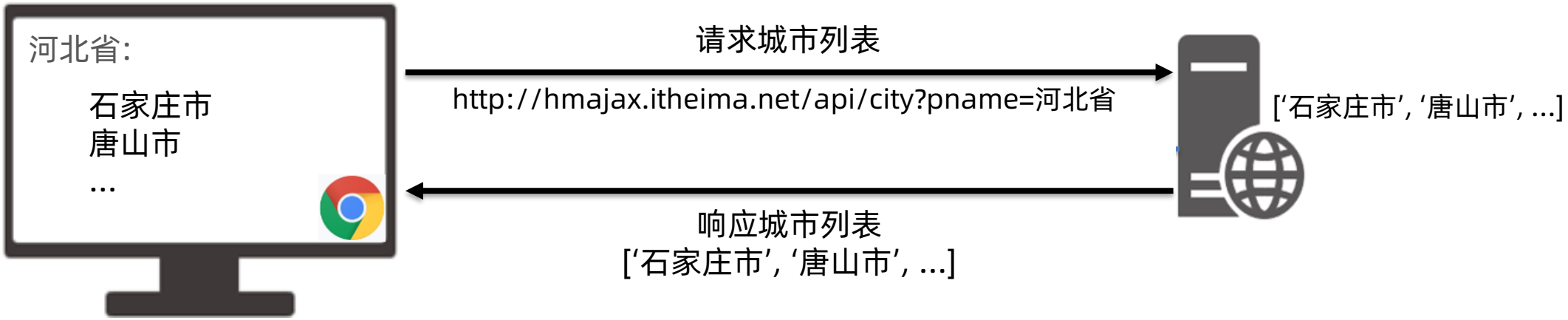
1. AJAX 原理是什么？
   - XMLHttpRequest 对象
2. 为什么学习 XHR？
   - 有更多与服务器数据通信方式
   - 了解 axios 内部原理
3. XHR 使用步骤？
   - 创建 XHR 对象
   - 调用 open 方法，设置 url 和请求方法
   - 监听 loadend 事件，接收结果
   - 调用 send 方法，发起请求

```
let xhr = new XMLHttpRequest()
xhr.open('请求方法', '请求url网址')
xhr.addEventListener('loadend', () => {
    // 响应结果
    console.log(xhr.response)
})
xhr.send()
```

XMLHttpRequest - 查询参数

定义：浏览器提供给服务器的额外信息，让服务器返回浏览器想要的数据

语法：http://xxxx.com/xxx/xxx?参数名1=值1&参数名2=值2



案例 地区查询

需求：输入省份和城市名字，查询地区列表

请求地址：http://hmajax.itheima.net/api/area?参数名1=值1&参数名2=值2


```
// 1. 创建 URLSearchParams 对象
const paramsObj = new URLSearchParams({
    参数名1: 值1,
    参数名2: 值2
})

// 2. 生成指定格式查询参数 字符串
const queryString = paramsObj.toString()
// 结果：参数名1=值1&参数名2=值2
```

XMLHttpRequest - 数据提交

需求：通过 XHR 提交用户名和密码，完成注册功能

核心：

请求头设置 Content-Type：application/json

请求体携带 JSON 字符串

```
const xhr = new XMLHttpRequest()
xhr.open('请求方法', '请求url网址')
xhr.addEventListener('loadend', () => {
    console.log(xhr.response)
})

// 告诉服务器，我传递的内容类型，是JSON字符串
xhr.setRequestHeader('Content-Type', 'application/json')
// 准备数据并转成JSON字符串
const user = { username: 'itheima007', password: '7654321' }
const userStr = JSON.stringify(user)
// 发送请求体数据
xhr.send(userStr)
```

目录

Contents

- XMLHttpRequest
- Promise
- 封装简易版 axios
- 案例 - 天气预报

Promise

定义：

Promise

`Promise` 对象用于表示一个异步操作的最终完成（或失败）及其结果值。

好处：
1. 逻辑更清晰
2. 了解 axios 函数内部运作机制
3. 能解决回调函数地狱问题

语法：

```
// 1. 创建Promise对象
const p = new Promise((resolve, reject) => {
  // 2. 执行异步任务-并传递结果
  // 成功调用： resolve(值) 触发 then() 执行
  // 失败调用： reject(值)  触发 catch() 执行
})
// 3. 接收结果
p.then(result => {
  // 成功
}).catch(error => {
  // 失败
})
```

总结

1. 什么是 Promise?
   - 表示（管理）一个异步操作最终状态和结果值的对象
2. 为什么学习 Promise?
   - 成功和失败状态，可以关联对应处理程序
   - 了解 axios 内部原理
3. Promise 使用步骤?

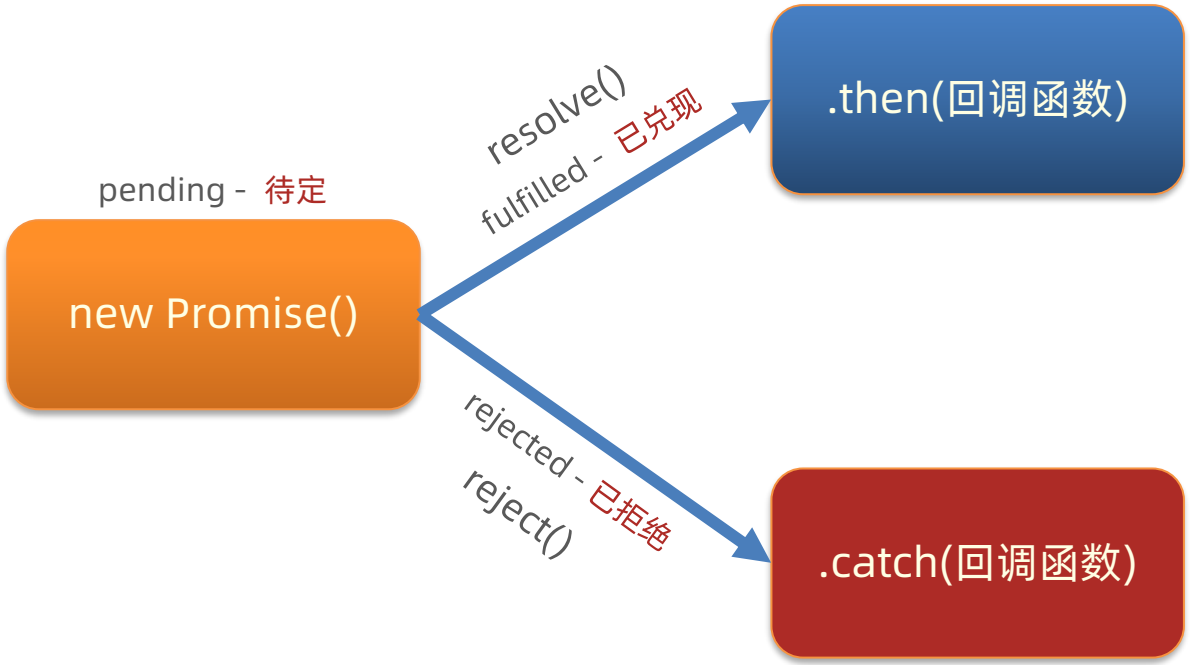
```
// 1. 创建Promise对象
const p = new Promise((resolve, reject) => {
  // 2. 执行异步任务-并传递结果
  // 成功调用：resolve(值) 触发 then() 执行
  // 失败调用：reject(值)  触发 catch() 执行
})
// 3. 接收结果
p.then(result => {
  // 成功
}).catch(error => {
  // 失败
})
```

Promise - 三种状态

作用：了解Promise对象如何关联的处理函数，以及代码执行顺序

概念：一个Promise对象，必然处于以下几种状态之一

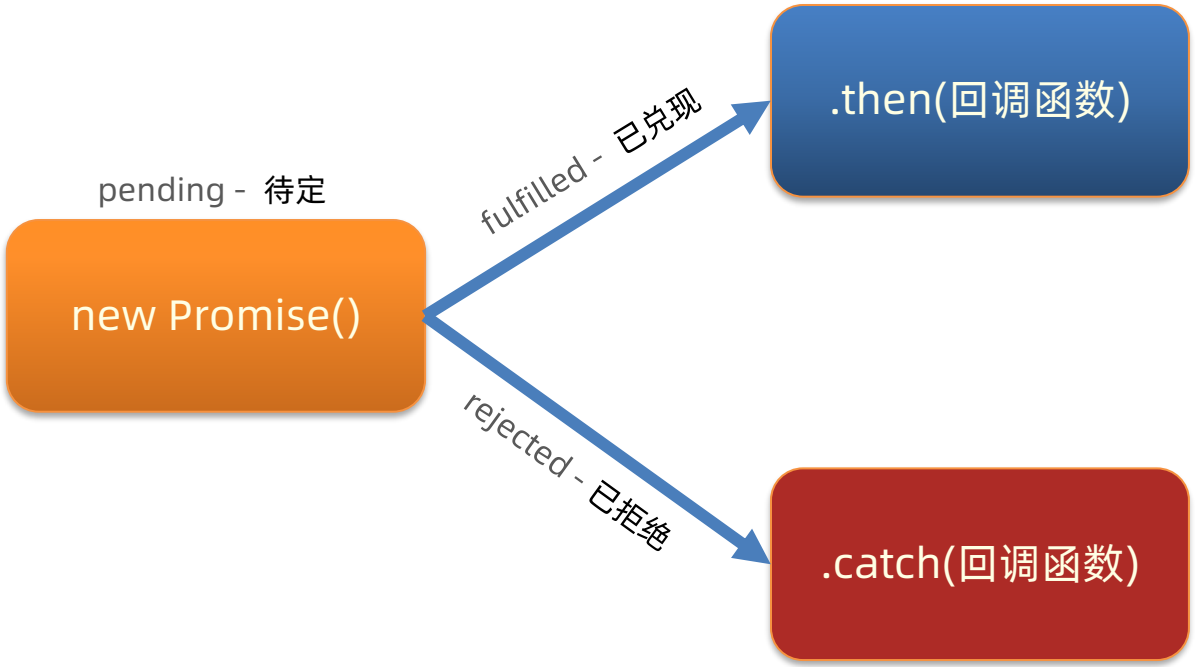
- ✓ 待定（pending）：初始状态，既没有被兑现，也没有被拒绝
- ✓ 已兑现（fulfilled）：意味着，操作成功完成
- ✓ 已拒绝（rejected）：意味着，操作失败



注意：Promise对象一旦被兑现/拒绝
就是已敲定了，状态无法再被改变

总结

1. Promise 对象有哪 3 种状态？
   - 待定 pending
   - 已兑现 fulfilled
   - 已拒绝 rejected
2. Promise 状态有什么用？
   - 状态改变后，调用关联的处理函数



案例 使用Promise + XHR 获取省份列表

需求：使用 Promise 管理 XHR 获取省份列表，并展示到页面上

步骤：

1. 创建 Promise 对象
2. 执行 XHR 异步代码，获取省份列表
3. 关联成功或失败函数，做后续处理

目录

Contents

- XMLHttpRequest
- Promise
- 封装简易版 axios
- 案例 - 天气预报

封装_简易axios_获取省份列表

需求：基于 Promise + XHR 封装 myAxios 函数，获取省份列表展示

步骤：

1. 定义 myAxios 函数，接收配置对象，返回 Promise 对象
2. 发起 XHR 请求，默认请求方法为 GET
3. 调用成功/失败的处理程序
4. 使用 myAxios 函数，获取省份列表展示


```
function myAxios(config) {
  return new Promise((resolve, reject) => {
    // XHR请求
    // 调用成功/失败的处理程序
  })
}

myAxios({
  url: '目标资源地址'
}).then(result => {

}).catch(error => {

})
```

封装_简易axios_获取地区列表

需求：修改 myAxios 函数支持传递查询参数，获取"辽宁省"，"大连市"对应地区列表展示

步骤：

1. myAxios 函数调用后，判断 params 选项
2. 基于 URLSearchParams 转换查询参数字符串
3. 使用自己封装的 myAxios 函数展示地区列表

```
function myAxios(config) {
  return new Promise((resolve, reject) => {
    // XHR请求 - 判断params选项，携带查询参数
    // 调用成功/失败的处理程序
  })
}

myAxios({
  url: '目标资源地址',
  params: {
    参数名1: 值1,
    参数名2: 值2
  }
})
```

封装_简易axios_注册用户

需求：修改 myAxios 函数支持传递请求体数据，完成注册用户功能

步骤：

1. myAxios 函数调用后，判断 data 选项
2. 转换数据类型，在 send 方法中发送
3. 使用自己封装的 myAxios 函数完成注册用户功能

```
function myAxios(config) {
  return new Promise((resolve, reject) => {
    // XHR请求 - 判断data选项，携带请求体
    // 调用成功/失败的处理程序
  })
}

myAxios({
  url: '目标资源地址',
  data: {
    参数名1: 值1,
    参数名2: 值2
  }
})
```

目录

Contents

- XMLHttpRequest
- Promise
- 封装简易版 axios
- 案例 - 天气预报

案例 天气预报

步骤：

1. 获取北京市天气数据，展示
2. 搜索城市列表，展示
3. 点击城市，显示对应天气数据

案例 天气预报 - 搜索城市列表

需求：根据关键字，展示匹配城市列表

步骤：

1. 绑定 input 事件，获取关键字
2. 获取展示城市列表数据

案例 天气预报 - 展示城市天气

需求：展示用户搜索查看的城市天气

步骤：

1. 检测搜索列表点击事件，获取城市 code 值
2. 复用获取展示城市天气函数

传智教育旗下高端IT教育品牌