1.客户端与服务器

1.1. 上网的目的

上网的本质目的是通过互联网的形式来获取和消费资源

1.2. 服务器

上网过程中，负责存放和对外提供资源的电脑，叫服务器

1.3. 客户端

上网过程中，负责获取和消费资源的电脑，叫客户端

2. URL地址

2.1. URL地址的概念

URL（UniformResourceLocator)中文叫统一资源定位符，用于标识互联网上每个资源的唯一存放位置

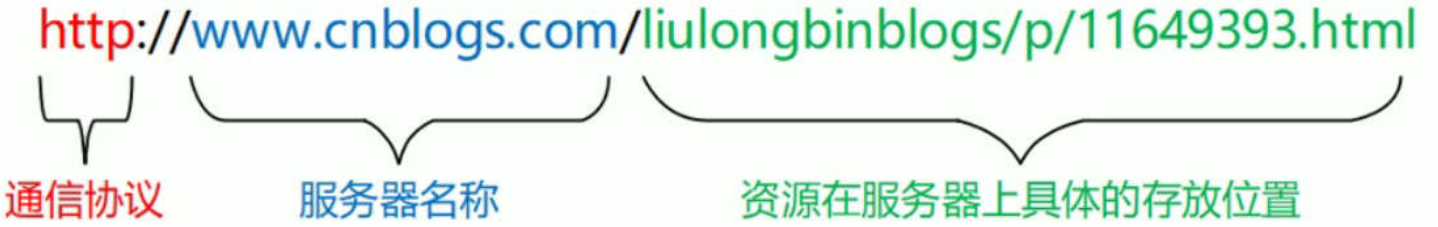
浏览区只有通过URL地址，才能正确的定位资源存放的位置，从而成功访问到对应的资源

2.2. URL地址的组成部分

URL地址一般由三部分组成：

- 客户端与服务器之间的通信协议
- 存有该资源的服务器名称
- 资源在服务器上具体的存放位置

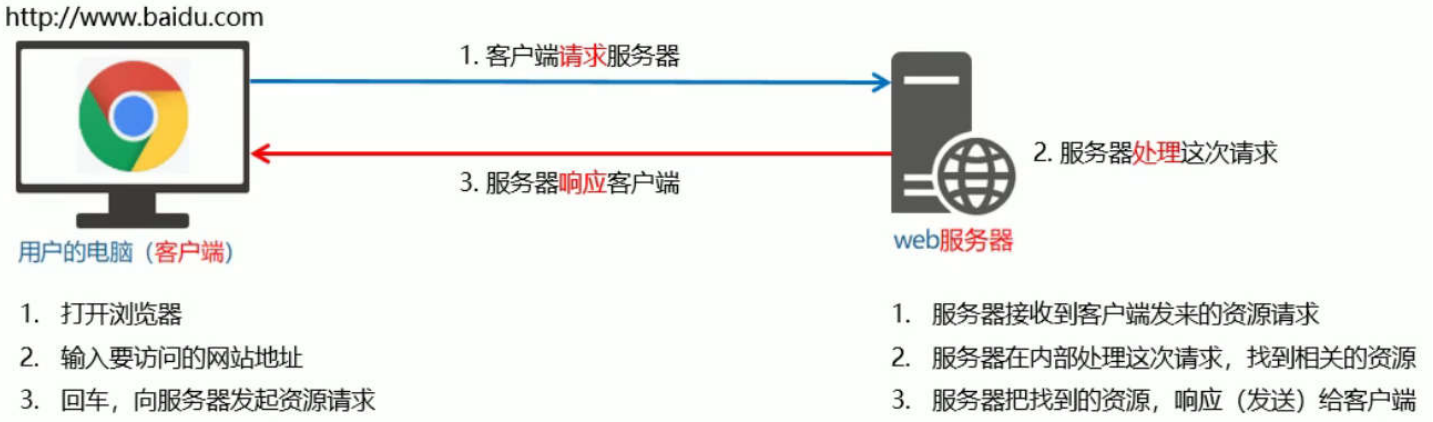
（图：URL组成）



3. 客户端与服务器的通信过程

3.1. 图解客户端与服务器的通信过程

（图：图解通信过程）



注意：

- 客户端与服务器之间的通信过程，分为请求-处理-响应三个步骤
- 网页中的每一个资源都是通过请求-处理-响应三个步骤

3.2. 基于浏览器的开发者工具分析通信过程

- 打开Chrome浏览器
- 打开开发者工具
- 切换到network面板
- 选中doc页签
- 刷新页面，分析客户端与服务器的通信过程

4. 服务器对外提供了哪些资源

4.1. 列举网页中常见的资源

4.2. 数据也是资源

网页中的数据，也是服务器对外提供的一种资源，

4.3. 数据是网页的灵魂

4.4. 网页中如何请求数据

数据也是资源，只要是资源，必然通过请求-处理-响应的方式进行获取

如果要在网页中请求服务器上的数据资源，则需要XMLHttpRequest对象

XMLHttpRequest（简称xhr）是浏览器提供的js成员，通过它，可以请求服务器上的数据资源

最简单的用法： var xhrObj=new XMLHttpRequest()

4.5. 资源的请求方式

客户端请求服务器时，请求的方式有很多种，最常见的两种请求方式是get和post请求

get请求通常用于获取服务器资源
post请求通常用于向服务器提交数据

5. 了解Ajax

5.1. 什么是Ajax

Ajax全称是Asynchronous Javascript And XML(异步JavaScript和XML)

通俗的理解就是在网页中利用XMLHttpRequest对象和服务器进行数据交互的方式就是Ajax

5.2. 为什么要学Ajax

Ajax可以让我们轻松实现网页与服务器站的数据交互

5.3. Ajax的典型应用场景

用户名检测：注册用户时，通过Ajax的形式，动态检测用户名是否被占用

搜索提示：当输入搜索关键字时，通过Ajax的方式，动态加载搜索提示列表

数据分页显示：当点击页码值时，通过Ajax的形式，根据页码值动态刷新表格的数据

数据的增删改查操作

6. jQuery中的Ajax

6.1. 了解jQuery中的Ajax

浏览器中提供的XMLHttpRequest用法比较复杂，所以jQuery对XMLHttpRequest进行了封装，提供了一系列Ajax相关的函数，极大地降低了Ajax的使用难度

jQuery中发起Ajax请求最常用的三个方法：

- $.get()
- $.post()
- $.ajax()

6.2. $.get()函数的语法

jQuery中$.get()函数的功能单一，专门用来发起get请求，从而将服务器上的资源请求到客户端来进行使用

语法：

```
$.get(url,[data],[callback])
```

（图：$.get()参数）

使用$.get()函数发起不带参数的请求时，直接提供请求的URL地址和请求之后的回调函数即可

```
    <script src="lib/jquery.js"></script>
</head>
<body>
    <button id="btnGET">发起不带参数的get请求</button>

    <script>
        $(function(){
            $('#btnGET').on('click',function(){
                $.get('http://www.liulongbin.top:3006/api/getbooks',function(res){
                    console.log(res);
                })
            })
        })
    </script>
```

使用$.get()函数发起不带参数的请求

```
    <script src="lib/jquery.js"></script>
</head>

<body>
    <button id="btnGETINFO">使用get发起带参数的请求</button>

    <script>
        $(function () {
            $('#btnGETINFO').on('click', function () {
                $.get('http://www.liulongbin.top:3006/api/getbooks', {
                    id: 1
                }, function (res) {
                    console.log(res);
                })
            })
        })
    </script>
```

6.3. $.post()函数的语法

jQuery中$.post()函数的功能单一，专门用来发起post请求，从而向服务器提交数据

语法：

```
$.post(url,[data],[callback])
```

三个参数各自的含义如下：

（图：post参数）

参数名	参数类型	是否必选	说明
url	string	是	提交数据的地址
data	object	否	要提交的数据
callback	function	否	数据提交成功时的回调函数

```
    <script src="lib/jquery.js"></script>
</head>
<body>
    <button id="btnPOST">发起post请求</button>

    <script>
        $(function(){
            $('#btnPOST').on('click',function(){
                $.post('http://www.liulongbin.top:3006/api/addbook',{bookname:'海绵宝宝',author
                    console.log(res);
                })
            })
        })
    </script>
```

6.4. ajax()函数

ajax()的语法相比于get()和post()函数，jQuery中提供的ajax()函数，是一个功能比较综合的函数，它允许我们对Ajax请求进行更详细的配置

$.ajax()函数的基本语法如下：

```
$.ajax({
    type:'',    //请求的方式，例如：get和post
    url:'',     //请求的URL地址
    data:{},    //请求要携带的数据
    success: function(res){}    //请求成功之后的回调函数
})
```

使用$.ajax()发起get请求

使用$.ajax()发起get请求时，只需要将type属性的值设置为GET即可

```
$.ajax({
    type:'GET',   //请求的方式，例如：get和post
    url:'',    //请求的URL地址
    data:{},   //请求要携带的数据
    success: function(res){}   //请求成功之后的回调函数
})

    <script src="lib/jquery.js"></script>
</head>

<body>
    <button id="btnGET">发起get请求</button>

    <script>
        $(function () {
            $('#btnGET').on('click', function () {
                $.ajax({
                    type: 'GET', //get大写小写都可以，尽可能写成大写
                    url: 'http://www.liulongbin.top:3006/api/getbooks',
                    data: {
                        id: 1
                    },
                    success: function (res) {
                        console.log(res);
                    }

                })
            })
        })
    </script>
```

使用.ajax()发起post请求使用.ajax()发起post请求时，只需要将type的属性值设置为POST即可

7. 接口

7.1. 接口的概念

使用Ajax请求数据时，被请求的URL地址，就叫做数据接口（简称接口）。同时，每个接口必须有请求方式

7.2. 分析接口的请求过程

通过get方式请求接口的过程
请求-处理-响应

通过post方式请求接口的过程
请求-处理-响应

7.3. 接口测试工具

什么是接口测试工具
为了验证接口能否正常被访问，常常使用接口测试工具，来对数据接口进行检测

好处：接口测试工具能让我们在不写任何代码的情况下，对接口进行调用和测试

常用工具：PostMan

7.4. 使用postman测试get接口

步骤：

- 选择请求的方式
- 填写请求的URL地址
- 填写请求的参数
- 点击send按钮发起get请求
- 查看服务器响应的结果

7.5. 使用postman测试post接口

步骤：

- 选择请求的方式
- 填写请求的URL地址
- 选择body面板并勾选数据格式
- 填写要发送到服务器的数据
- 点击send按钮发起post请求
- 看服务器响应的结果

8. 案例-图书管理

8.1. 渲染UI结构

8.2. 案例要用到的库和插件

- css:bootstrap.css
- Javascript:jquery.js
- vscode 插件：Bootstrap 3 Snippets

9. 案例-聊天机器人

9.1. 演示案例要完成的效果

实现步骤：

- 梳理案例的代码结构
- 将用户输入的内容渲染到聊天窗口
- 发起请求获取聊天的消息
- 将聊天机器人的聊天内容转为语音
- 通过audio播放语音
- 使用回车键发送消息

9.2 梳理案例的代码结构

- 梳理页面的UI布局
- 将业务代码抽离到chat.js中
- 了解resetui()函数的作用

9.3. 将用户输入的内容渲染到聊天窗口中

```
$(function () {
    // 初始化右侧滚动条
    // 这个方法定义在scroll.js中
    resetui()
})

// 为发送按钮绑定鼠标的点击事件
$('#btnSend').on('click', function () {
    var text = $('#ipt').val().trim()
    if (text.length <= 0) {
        return $('#ipt').val('')
    }

    // 如果用户输入了聊天内容，则将内容追加到聊天页面上显示
    $('#talk_list').append('<li class="right_word"><img src="img/person02.png" /> <span>' + te

    // 重置滚动条的位置
    resetui();

})
```

9.4. 发起请求获取聊天的消息