AJAX

URL

—— Uniform Resource Locator

- URL的作用

  用于表示任意资源的位置（互联网）

  http://www.tmooc.cn:80/script/images_new/TMOOC.png

  协议 + 主机 + 文件目录结构 + 文件名称

- 详解

  格式：

  <scheme>://<user>:<pwd>@<host>:<port>/<path>:<params>?<query>#<frag>

1. SCHEME：方案，指定以哪种协议从服务器获取指定资源，方案名不区分大小写

   常见方案：HTTP/ HTTPS/ FTP/ MAILTO/ RTSP/ FILE/ NEWS/ TELNET
2. USER：用户名，访问服务器资源时需要的指定用户名，默认值：anonymouse

   PWD：密码，默认值为EMAIL地址
3. HOST：主机名，资源所在服务器的IP地址或者域名（需要DNS转换为IP地址）
4. PORT：端口号，每项服务在服务器上都对应一个监听端口号
5. PARAMS：参数，某些方案会使用参数来指定输入参数
6. QUERY：查询字符串，某些方案会使用查询字符串传递参数以激活应用程序，以 `?` 隔开
7. FRAG：锚点，指定一个资源中某一部分的名字

协议

HTTP协议

- 什么是 HTTP？

  Hyper Text Transfer Protocol (超文本传输协议)；规范了数据是如何打包以及传输的

- 请求（Request）消息
  - 客户端要带给服务器的数据：

    由三部分组成：请求起始行，请求头，请求主体

  - 请求起始行（请求方法，请求url，协议版本）

    请求方法：

        - GET 表示客户端向服务器要数据时使用

          特点：没有请求主体；靠地址栏传递查询的字符串
        - POST 表示想传递数据给 服务器 时使用

          特点：隐式传递； 有请求主体
        - PUT 表示客户端想放置数据到服务器
        - DELETE 表示客户端想要删除服务器上的数据
        - HEAD 表示客户端只想获取头信息时使用
        - CONNECT 测试连接
        - TRACE 追踪请求路径
        - OPTIONS 选项，保留以后使用
        - 请求url (ex:http://www.jd.com/index.php)
        - 协议版本 http/1.1
    - 请求头
        - host : localhost(本地主机)

          作用：告诉服务器请求哪个主机
        - Connection : keep-alive

          作用：告诉服务器要进行持久连接
        - User-Agent :

          作用：告诉服务器自己（浏览器）的类型 chorme/FF/ ...
        - Accept-Language : zh-cn

          作用：告诉服务器自己接受的自然语言（中文）
        - Accept-Encoding : gzip

          作用：告诉服务器自己接受的压缩文件类型
        - Referer :

          作用：告诉服务器请求来自哪个页面
    - 请求主体
        - Form data

- 响应（Response）消息

  响应：服务器端将结果返回给客户端，

  由三部分组成：响应起始行，响应头，响应主体
    - 响应起始行

      响应协议版本号 HTTP/1.1

      响应的状态码

          作用：告诉浏览器，服务器端响应的状态是什么

          1xx : 100-199, 提示信息

2xx : 200 : ok 成功

3xx : 301 : Moved Permanently 永久性重定向

302 : See Other 临时重定向

304 : Not Modified 命中了缓存

4xx : 404 : Not Found 请求资源不存在

403 : Forbidden 权限不够

404 : Method not Allowed 请求方法不允许

5xx : 500 : 服务器运行出错（服务器内部错误）

- 响应头（Response Header）
  - Date

    作用：服务器告诉浏览器，服务器的响应时间：默认是格林尼治时间
  - Connection

    作用：告诉浏览器已经启动了持久连接
  - Content-Type

    作用：告诉浏览器响应回来的内容类型

    取值：

    text/html：html文本

    text/plain：普通文本

    text/css：css样式

    application/javascript：js脚本代码

    application/xml：xml字符串

    application/jason：json格式的字符串

    images/jpeg;
- 响应主体

  Response

缓存

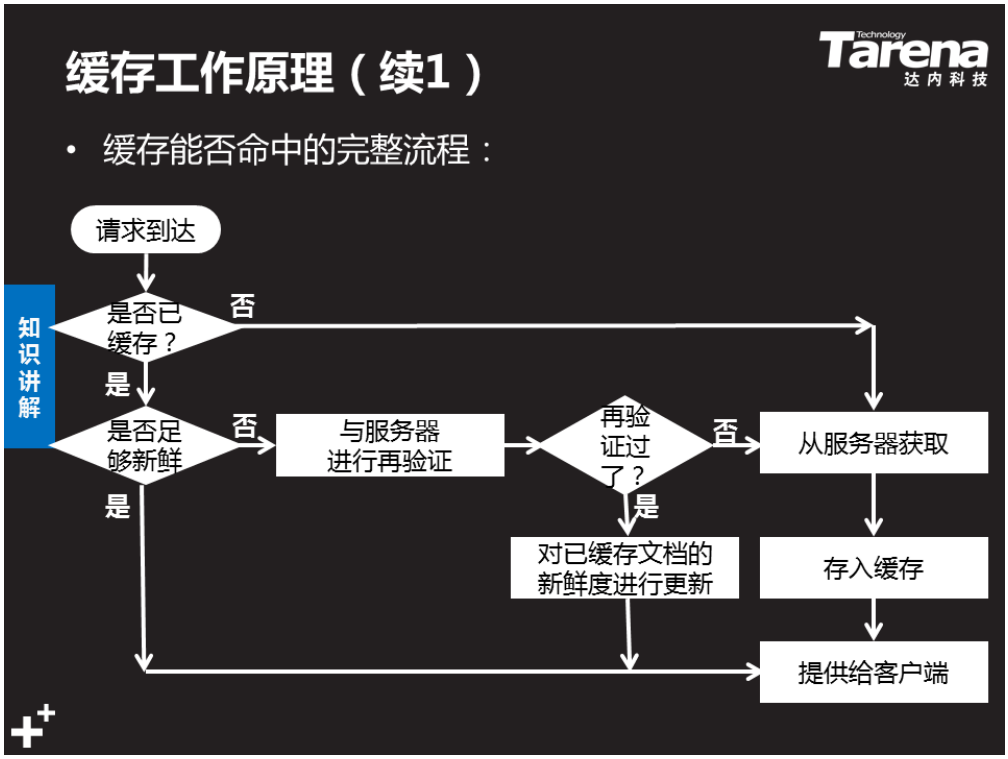
- 什么是缓存

  客户端将服务器响应回来的数据进行自动保存，当再次访问时，直接使用保存的数据。
- 缓存优点

  减少了冗余数据的传输，节省客户端的流量

  可以节省服务器的带宽，减轻服务器的压力

  降低了由于远距离而造成的延时加载



- 与缓存相关的消息头
  - Cache-Control消息头

    作用：从服务器将文档传递到客户端之时起，此文档处于新鲜的秒数

    语法：

    Cache-Control:max-age = 处于新鲜的秒数

    Cache-Control:max-age = 3600;
  - Expires消息头

    作用：指定缓存过期的确切时间

    语法：

    Expires:Thu,18 Nov 2017 07:00:00

  在网页中设置缓存：

  <meta http-equiv="Cache-Control" content="max-age=3600">

Ajax

- 什么是ajax?

  Asynchronous Javascript And Xml

  异步的 js 和 数据格式

  本质：使用js提供的异步对象（XMLHttpRequest），

  异步的向服务器发送请求，并接收响应回来的数据（数据的格式是xml）
- 异步请求的步骤

    - 创建ajax（xhr）对象
    - 绑定监听事件
    - 打开连接
    - 发送请求
- 创建异步对象（xhr）
    - 标准创建

```
var xhr = new XMLHttpRequest();
```

    - IE8以下创建

```
var xhr = new ActiveXObject("Microsoft.XMLHttp");
```

    - 通过window.XMLHttpRequest来判断浏览器是否支持标准创建。

      如果浏览器不支持标准创建，那么window.XMLHttpRequest的值为null

```
if(window.XMLHttpRequest){
  var xhr=new XMLHttpRequest();
}else{
  var xhr=new ActiveXObject("Microsoft.XMLHttp");
}
```

异步对象 xhr

- **readyState** 属性

  作用：用于表示异步对象（xhr）的请求状态

  取值：

  0 - 4 表示5个状态
  0：请求尚未初始化
  1：已经打开到服务器连接，正在发送请求中
  2：接收响应头
  3：接收响应主体
  4：接收响应数据成功

  注意：当readyState的值为4的时候，才表示所有的响应都已经接收完毕。
- **status** 属性

  作用：表示服务器的响应状态码

  取值：200

  当status的值为200时，表示服务器已经正确的处理了请求以及给出了响应。
- **onreadystatechange** 事件

  作用：当 xhr 的readyState属性值发生改变时，要自动激发的操作

```
xhr.onreadystatechange = function(){
    if(xhr.readyState == 4 && xhr.status == 200){
      //接收响应回来的结果
      var result = xhr.responseText;
      console.log(result);
    }
}
```

- **open()**

  作用：打开连接（创建请求）

  语法：`xhr.open(method,url,isAsyn);`

  - **method** 请求方法（get/post）string 类型
  - **url** 请求地址 string类型
    isAsyn 指定采用同步（false）还是异步(true)方式发送请求 boolean类型
- **send()**

  作用：发送请求

  语法：`xhr.send(body)`

  - **body**：请求主体

    如果说没有请求主体，body处值为null.

    ```
    如果说有请求主体，body处放请求主体数据。
    ```

- 使用 `GET` 提交，发送请求数据（带参数）

  http://127.0.0.1:8080/user_login.php?uname=dangdang&upwd=123456
- 使用 post 发送请求

  注意两点：

  1. post的请求的数据放在请求主体中

     xhr.send("uname="+value1+"&upwd="+uvalue2);

  2. 在发送请求之前，需要手动修改请求消息头

     xhr.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

JSON格式

- 什么是 JSON?

  JavaScript Object Notation (js对象表现方式)

  js对象表示法，即以js对象的格式表现出来的 **字符串**
- JSON 语法

用一对 {} 来表示一个对象

对象的属性名称，必须用 **双引号** 引起来，值如果是字符串的话，也必须用 **双引号**

EX:

普通字符串： var Tom = "汤姆";

JSON字符串：var Tom = '{"name":"汤姆", "age":20}';

- JSON数组

  普通数组： var arr = ["金三胖", "川普", "习大大"];

  JSON数组：var arr = '["金三胖", "川普", "习大大"]';

```
var arr = '[
  { "name" : "金三胖", "height" : 155, "age" : 40 },
  { "name" : "川普", "height" : 180, "age" : 65 },
  { "name" : "习大大", "height" : 180, "age" : 60 }
]';
```

json_encode()

语法：通过 json_encode() 将数组转换为json格式的字符串，并返回转换后的结果；js -> json

ex:

$array=["习大大","金三胖","川普"];

$str=json_encode($array);

echo $str;

JSON_parse()

使用 JSON.parse() 来将 json 字符串解析为 js对象或数组；json -> js

EX:

var obj = JSON.parse("jsonString");

XML

Ajax: Asynchronous Javascript And Xml

什么是XML

eXtensible Markup Language **可扩展标记语言**

XML的标记没有被预定义过，需要自行定义

XML的宗旨：数据传递，非数据显示

XML的语法规范

XML可以保存在 .xml 的文件中，也可以以字符串的形式出现在其他文件中

XML的声明：最顶端

`<?xml version="1.0" encoding="utf-8"?>`

XML标记的语法

1. **XML标记必须成对出现**

```
<persion>  // erro
<persion></persion> // right
```

2. **XML标记严格区分大小写，开始和结束标记必须一致**
3. **XML标记允许被嵌套，注意顺序**

```
<student>
  <firstName></firstName>
  <lastName></lastName>
</student>
```

4. **每个标记都允许定义属性，格式与HTML一致，但属性值必须用 双引号 引起来**

`<student num="10"></student>`

5. **每个XML文件，必须要有一个根元素**

使用AJAX请求XML文件

遵循AJAX的请求步骤

1. 创建xhr
2. 绑定事件

   在绑定事件的回调函数中获取xml返回数据，

   使用xhr.responseXML来获取相应数据

   返回的是xml文档对象
3. 打开链接

   xhr.open("get", "student.xml", true);
4. 发送请求

1. 解析XML文档象的内容
    1. 核心方法

       elem.getElementsByTagName("标记");

       elem：获取范围的对象

       返回值：返回一个包含指定元素的"类数组"

2. 在PHP中返回XML格式的字符串
    1. 必须增加响应消息头

       header("Content-Type:application/xml");

    2. 按照xml的语法结构，去拼接xml字符串，再响应给客户端

       $xml = "<?xml version='1.0' encoding='utf-8'?>";

       $xml .= "<StudentList>";

           ... ...

       $xml .= "</StudentList>" ;