



离线存储 : Storage

讲师 : 许井龙

微信 : ngsteel
邮箱 : ngsteel@qq.com

HTML5之前的离线存储

Cookie 用于弥补 HTTP 协议的无状态性，服务器可以使用 Cookie 中包含的信息来判断 HTTP 传输中的状态。

Cookie的显著缺陷：

- ✓ Cookie 需要在客户端和服务端来回地传送，繁琐且消耗带宽；

参考：<http://browsercookielimits.squawky.net/>

不同浏览器的Cookiez略微有些差异。

离线存储技术对比表

	大小	生命周期
Cookie	4KB	开发人员自定义
sessionStorage	5MB	浏览器关闭之前
localStorage	5MB	永久，除非用户主动清除。
IndexedDB	不存在大小限制	永久，除非用户主动清除。
WebSQL	注意：W3C已经不再支持这种技术！忘了它！	
Application Cache		

Index.html



HTTP请求-响应 www.baidu.com 56.98.172.66

http://www.baidu.com

会话: sessions 浏览器端的会话 (tab) : 浏览器(tab)打开--- 关闭

考虑浏览器兼容性问题，首先自然是检测浏览器是否支持本地存储。

```
if( window.localStorage ){  
    alert('This browser supports localStorage');  
}  
else{  
    alert('This browser does NOT support localStorage');  
}
```

存储数据的方法就是直接给 window.localStorage 添加一个属性。

```
localStorage.setItem("name","张三"); //设置name为"张三"
```

```
v var b = localStorage.getItem("name"); //获取name的值
```

```
localStorage.removeItem("name"); //清除key为name的值
```

```
localStorage.clear(); //清楚所有键-值对
```

这里最推荐使用的自然是getItem()和setItem()，清除键值对使用removeItem()。如果希望一次性清除所有的键值对，可以使用clear()。另外，HTML5还提供了一个key()方法，可以在不知道有哪些键值的时候使用，如下：

```
var storage = window.localStorage;

function showStorage(){
    for(var i=0; i<storage.length; i++){
        //key(i)获得相应的键，再用getItem()方法获得对应的值
        document.write(storage.key(i)+ " : " + storage.getItem(storage.key(i)) + "<br>");
    }
}
```

sessionStorage 方法

sessionStorage 方法针对一个 session 进行数据存储。当用户关闭浏览器窗口后，数据会被删除。

```
<script type="text/javascript">  
sessionStorage.lastname="Smith";  
document.write(sessionStorage.lastname);  
</script>
```


练习：记录用户的搜索信息

您上次搜索的关键词是：尚硅谷

要求：每次刷刷新页面，显示用户上次搜索的关键词。



H5 history

讲师：许井龙

- **复习H4历史管理**
- **H5历史管理新增的方法及属性**
 - `pushState()`
 - `replaceState()`
 - `window.popstate`
- **兼容性问题**

- DOM window 的 history 存储了用户的访问记录。
- 利用 history 提供了方法和属性，程序开发工程师可以控制浏览器地址栏左侧的返回（back），前进（forward）按钮，实现对已访问页面的重新访问。
- 注意通过 history back() 和 forward() 方法访问的是被缓存的网页，浏览器不会重新去服务器请求。如果网站需要实时刷新，使用该技术处理页面要谨慎。

- `window.history.back()`; 返回上一个页面，类似点击了浏览器地址栏左侧的后退（back）按钮
- `window.history.forward()`; 向前访问一个网页。，类似点击了浏览器地址栏左侧的前进（forward）按钮

`window.history.go(num);`

`window.history`（网页从1~5逐一访问后）

test.html

bbhtml

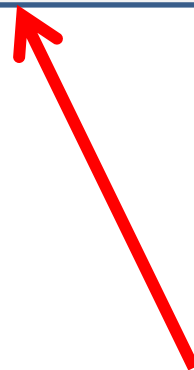
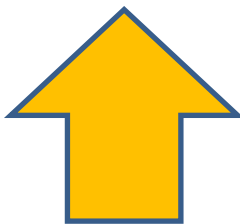
`window.history.go(-1);`

`window.history.go(1);`

当前浏览器显示的页面

`window.history.go(-2);`

`window.history.go(2);`



- 获取当前访问历史记录个数：`window.history.length;`

- HTML5 history 新增了两个方法
 - history.pushState() -- 增加历史记录
 - history.replaceState() --- 修改历史记录
- 上述两个方法与配合[window.onpopstate事件](#)一起使用。


```
var stateObj = { foo: "bar" };  
  
history.pushState(stateObj, "page 2", "bar.html");
```

执行上述脚本代码后，

- 1、浏览器地址栏会发生改变。
- 2、浏览器不会向bar.html发起HTTP请求，仅仅是改变了浏览器地址栏。
- 3、当地址栏像是bar.html时入股我们通过刷新，或者回车键（enter）直接访问该网址，会提示404错误。

pushState () 共包含三个参数。state object , title (所有浏览器暂不支持) , URL (可选) 。

- 参数1 : state object :
 - 一个JS对象，通常是JSON。通过pushState()与历史记录 (history entry) 绑定。当用户点击地址栏后退 (back) , 向前 (forward) 按钮时会触发window.onpopstate , 此时该事件 (event) 的state属性就会包含一个state object副本。
 - state object可以被序列化，Firefox 会把该数据序列化后保留在用户的磁盘上，即便用户重新启动浏览器，数据也不会丢失。序列化的数据存储在本地的磁盘不能超过640kb，当超出该限制，再调用pushState()时就会发生异常。如果你希望保存更多离线数据请使用sessionStorage 或者localStorage.

- 参数二：title：
 - HTML文档的标题（所有浏览器不支持）。实际开发使用document.title来代替。
- 参数三：URL
 - 可选参数。显示在地址栏的网页地址。可以是绝对地址，也可以是相对地址。如果是绝对地址，会把地址栏的URL整体替换掉。
 - **注意：该URL必须与当前网站URL同源（origin）。否则调用pushState()时，会发生异常。如果不指定，默认使用当前地址**

- `history.replaceState()` 与 `history.pushState()` 非常的相似。唯一的区别是，该方法是把当前的历史记录（`history entry`）替换成新的历史记录。

- **1、触发时机：**

- 相同文档中，两个历史记录（ history entry ）切换时。如果历史记录基于 pushState（ ）或者replaceState()创建的，popstate事件的state属性会包含state object的副本。
- 仅仅通过调用history.pushState() 或者 history.replaceState() 不会触发onpopstate 事件，只能通过点击浏览器地址栏的后退(back)按钮或者基于JS调用history.back()才会触发。
- 每个浏览器触发popstate事件有所不同。Chrome（ 34以上版本 ）实在页面加载时就触发，而Firefox不是。

- **2、使用方法：**

```
window.onpopstate = function(event) {  
    event.state; //存储的是 state object 的副本  
};
```

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari
replaceState, pushState	5	4.0 (2.0)	10	11.50	5.0
history.state	18	4.0 (2.0)	10	11.50	6.0

Feature	Android	Firefox Mobile (Gecko)	IE Mobile	Opera Mobile	Safari Mobile
replaceState, pushState	?	?	?	?	?
history.state	?	?	?	?	?

来自mozilla开发者中心

更新时间: **Apr 23, 2016, 2:09:02 PM**

使用H5历史管理实现一站式开发





HTML5 视频&与音频

讲师：许井龙

- 一、H4音频&视频
- 二、H5音频&视频
- 四、H5音频属性及JS API
- 五、H5视频属性及JS API

一、H4音频&视频

- 在H5之前我们需要通过<object>和<embed>嵌入音频或者视频资源。
- 在网络上展示视频、音频、动画，除了使用第三方开发的播放器，Flash应该算是使用最多的工具了。但是这需要用户在其浏览器安装插件。

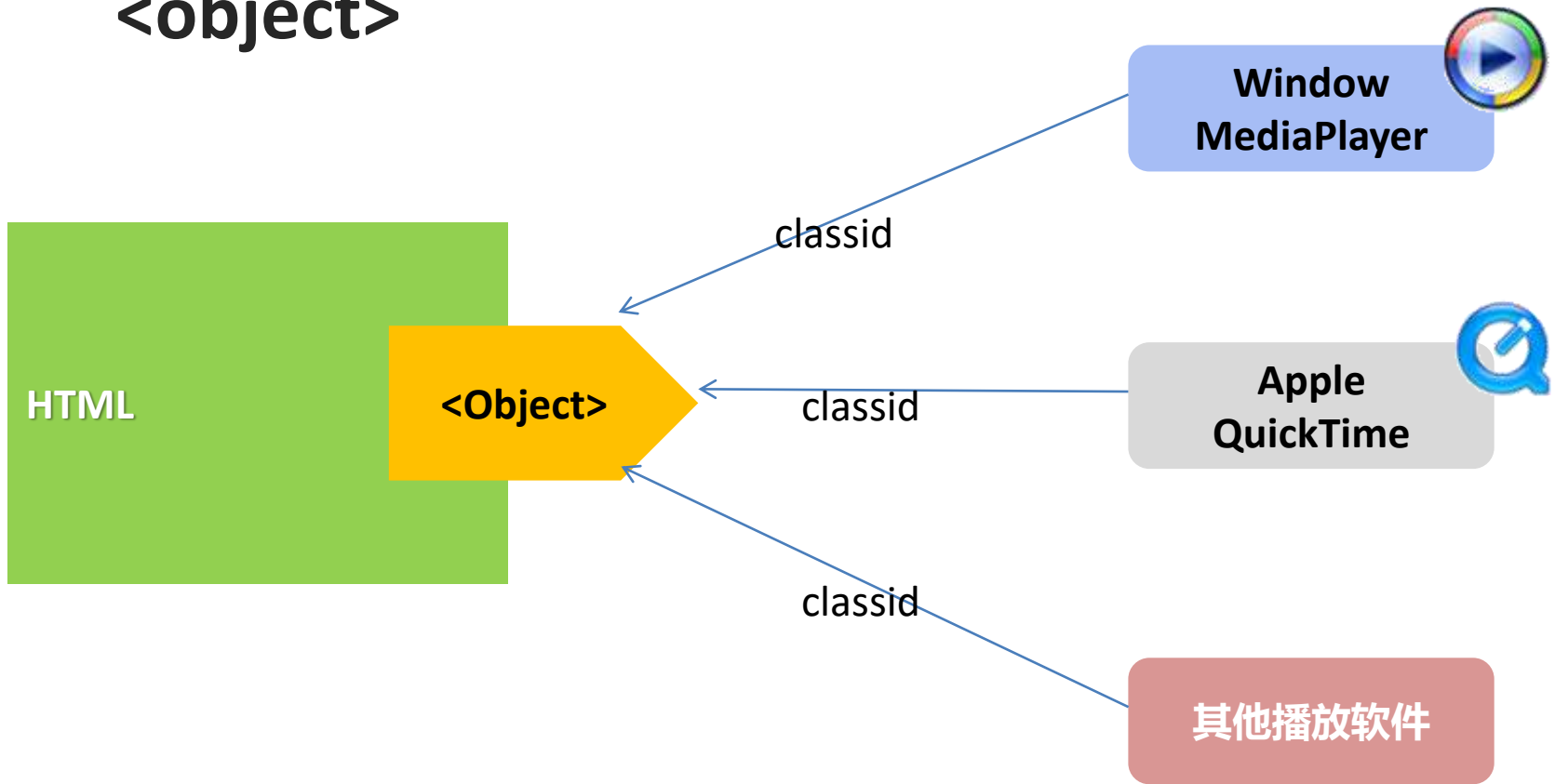
播放音频（支持IE8及其以下浏览器）

```
<object width="280" height="25"  
  classid="CLSID:22d6f312-b0f6-11d0-94ab-0080c74c7e95"  
  type="application/x-oleobject">  
  <param name="FileName" value="media/With_An_Orchid_Yanni.mp3">  
  <param name="ShowControls" value="1">  
  <param name="ShowStatusBar" value="0">  
  <param name="AutoStart" value="1">  
</object>
```

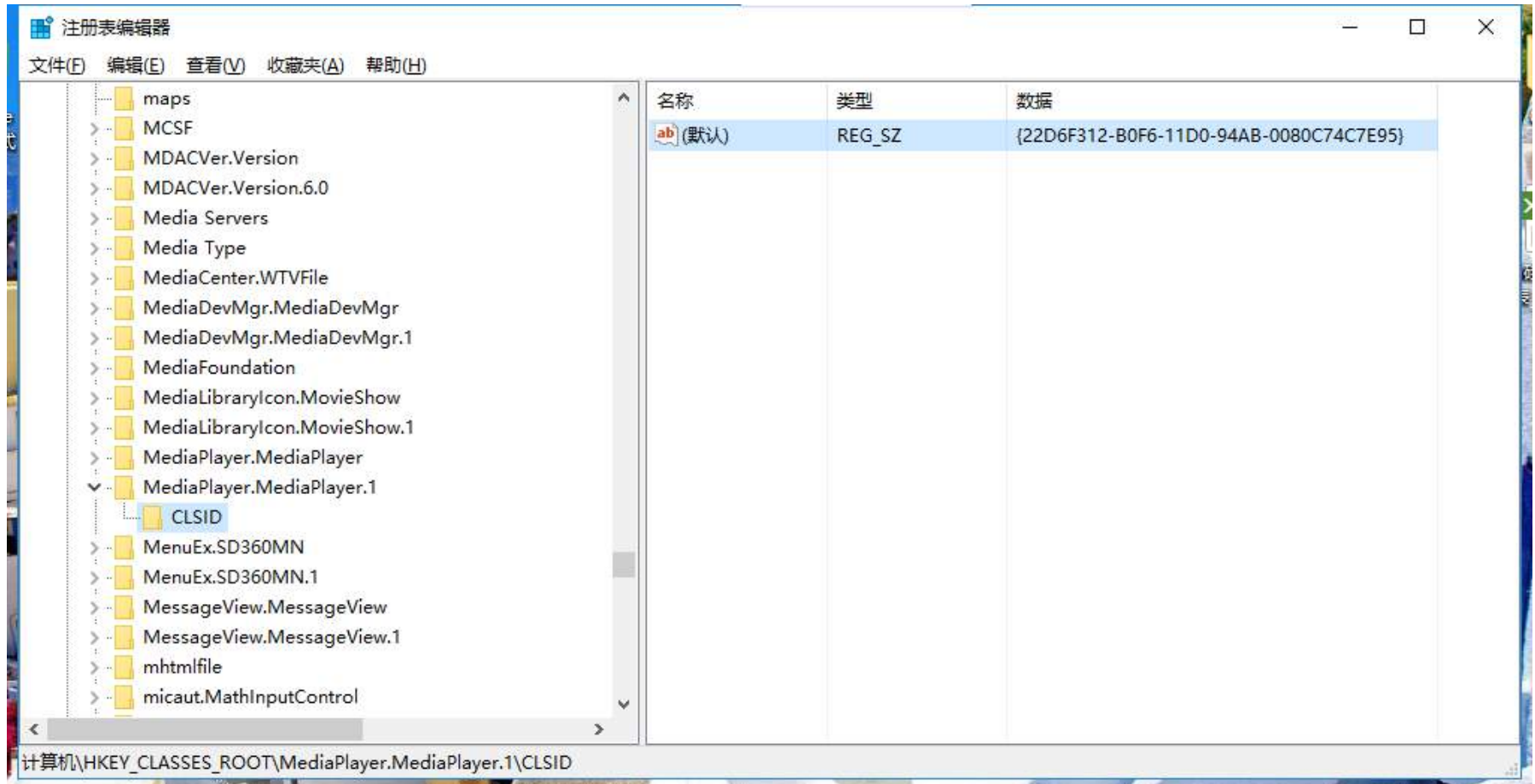
播放视频（支持IE8及其以下浏览器）

```
<object id="myobj" classid="clsid:CFCDA03-8BE4-11cf-B84B-0020AFBBCCFA">  
  <embed src="media/xuwei_wuzhe.mp4"  
    type="video/mp4"  
    width="640"  
    height="320"  
    controller="true"  
    autoplay="false">  
  </embed>  
</object>
```

<object>



关于classid




- H5废止<object>后，
 - 使用audio元素，直接在网页嵌入音频资源，通过CSS进行样式设置，通过Javascript操作该元素及相关属性。
 - 使用video元素，直接在网页嵌入视频资源，与audio元素很多属性相同，具有类似语法，也通过CSS进行样式设置，通过Javascript操作该元素及相关属性。

- **src** : 指定视频资源地址
 - 一般使用<source>代替, 可以指定多个音频格式。至少需要包含ogg免费格式, 以及.mp3或者wav格式。这种方法应该基本覆盖了最新浏览器。
 - 常用的转换软件: Converter
- **controls** : 布尔值。为音频增加控制界面。
- **autoplay** : 布尔值。告诉浏览器自动播放, 一旦设置该属性就会触发加载, 尽管可以使用该属性, 但不要使用它。
- **loop** : 布尔值。告诉浏览器重复播放音频。也应该谨慎使用。
- **preload** : 预加载
 - auto : 建议浏览器加载音频文件。仅仅是建议, 当浏览器检测到当前是移动设备或者连接较慢时, 浏览器可能不要预加载。以及节省流量或者带宽。
 - metadata : 提示浏览器不要预加载音频文件。可以预加载时长(duration)和音轨(tracks) 这样的元数据。
 - none : 不应该下载音频。

metadata

<input type="checkbox"/>	With_An_Orchid_Yanni.OGG	200	media	demo_audio.html:1	36.0 KB	43 ms	
--------------------------	--------------------------	-----	-------	-----------------------------------	---------	-------	---

auto

<input type="checkbox"/>	With_An_Orchid_Yanni.OGG	200	media	demo_audio.html:1	2.1 MB	159 ms	
--------------------------	--------------------------	-----	-------	-----------------------------------	--------	--------	---

- 常用JS方法
 - play() 从当前位置播放
 - pause() 如果音频在播放中，则暂停播放。
- 常用JS事件
 - onloadedmetadata 当音频元数据加载完毕时触发。
 - ontimeupdate 播放过程中实时触发。
 - onvolumechange 声音改变时触发
- 常用JS属性
 - duration 音频总时长（返回未格式化的秒）
 - currentTime 音频已经播放时长（返回未格式化的秒）
 - volume：0~1的任意值。控制音量。
 - muted：布尔值。静音。（ture表示静音，false表示非静音）
 - paused：布尔值。音频文件是否暂停。（ture表示暂停，false表示播放）
 - ended：布尔值。音频文件播放结束（ture表示播放结束，false表示播放中或者暂停）

- **src** : 指定视频资源地址
- **controls** : 布尔值。 为视频增加控制界面
- **preload** : 预加载
 - auto : 建议浏览器预加载视频文件。 仅仅是建议, 当浏览器检测到当前是移动设备或者连接较慢时, 浏览器可能不要预加载。 以及节省流量或者带宽。
 - metadata : 提示浏览器不要预加载视频文件。 可以预加载时长(duration)和音轨 (tracks) 这样的元数据。
 - none : 不预先加载视频。
- **autoplay** : 布尔值。 告诉浏览器自动播放, 一旦设置该属性就会触发加载, 尽管可以使用该属性, 但不要使用它。 尤其是在移动端。
- **loop** : 布尔值。 告诉浏览器重复播放视频。 也应该谨慎使用。
- 子元素 **<source>** : 可以指定多个视频格式。 至少需要包含ogg免费格式, 以及.mp4或者webm格式。 这种方法应该基本覆盖了最新浏览器。
- **poster** : 占位图。 一般视图片的URL。







设备访问：手机触屏事件

讲师：许井龙

微信：ngsteel
邮箱：ngsteel@qq.com

- 触摸接口是一个触摸敏感设备的单点接触点。接触点通常是一个手指或手写笔和设备可能是一个触摸屏或触控板。

- 手机触屏相关事件介绍

- **touchstart** : 触摸开始的时候触发
- **touchmove** : 手指在屏幕上滑动的时候触发
- **touchend** : 触摸结束的时候触发



- **targetTouches** : 位于当前DOM元素上手指的列表。

- Touch.identifier : 返回被触摸对象的唯一标识符。手指在屏幕表面滑动的过程中, 触点具有相同标识符, 这能让你一直的追踪的相同的触摸事件。
- Touch.target : 被触摸的DOM元素。
- Touch.screenX , 返回距离设备屏幕左边缘的X轴坐标
- Touch.screenY , 返回距离设备屏幕上边缘的Y轴坐标
- Touch.clientX , 返回距离浏览器视窗左边缘X轴坐标, 不包括滚动条 (scroll offset)。
- Touch.clientY , 返回距离浏览器视窗上边缘Y轴坐标, 不包括滚动条 (scroll offset)。
- **Touch.pageX , 返回距离HTML文档左边缘的的X轴坐标, 与Touch.clientX不同, 它包括滚动条。**
- **Touch.pageY , 返回距离HTML文档左边缘的的Y轴坐标, 与Touch.clientY不同, 它包括滚动条。**
- radiusX/radiusY/rotationAngle : 画出大约相当于手指形状的椭圆形, 分别为椭圆形的两个半径和旋转角度。该属性暂时处于试验阶段。

www.yahoo.com

www.search.com keyword html5 touch

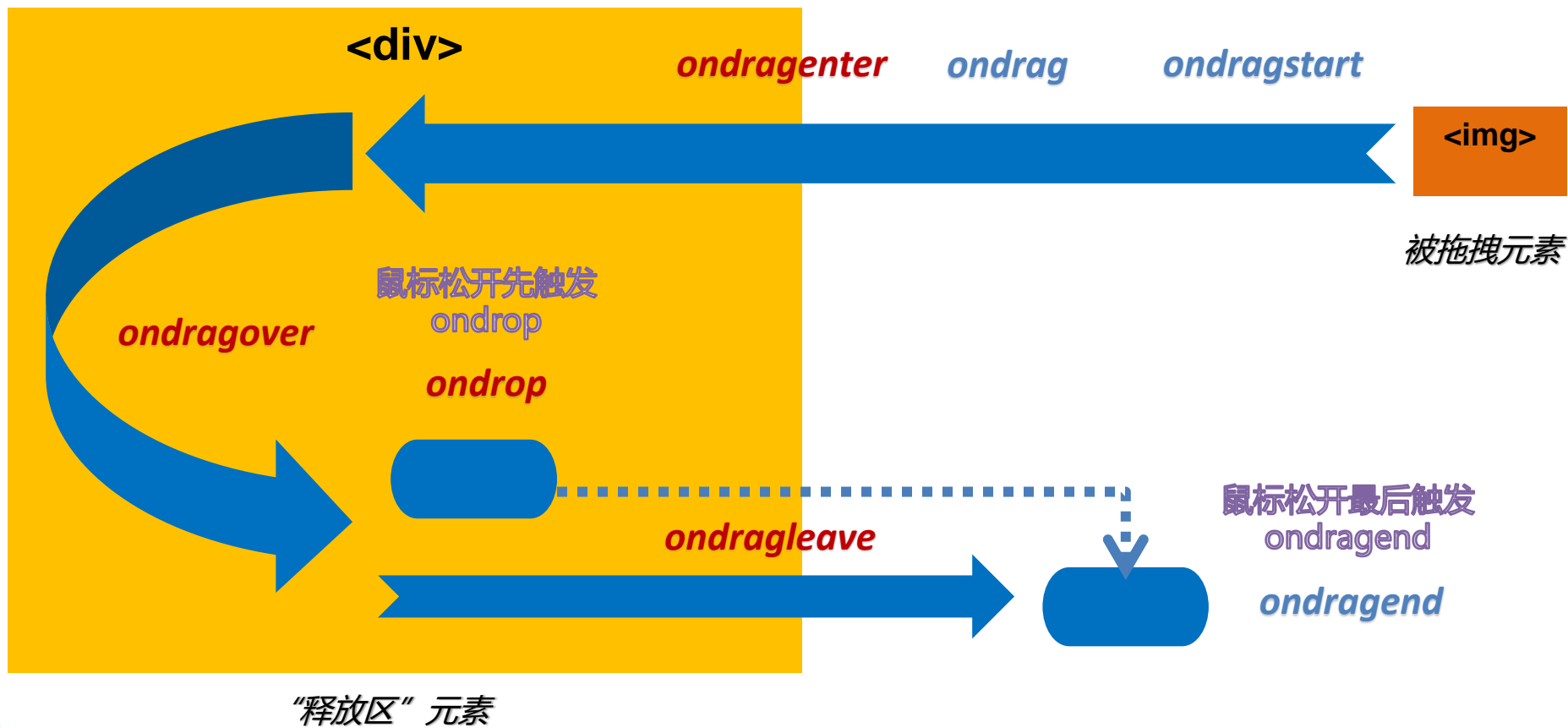
所有属性均为只读, 不可修改



HTML5 拖放 Drag&Drop

讲师：许井龙

通过鼠标拖动一个元素，会触发一系列拖放事件



为元素开启拖拽特性：为该元素设置 `draggable="true"`

示例代码，

```
<div draggable="true" ></div>
```

此时div可以被拖拽

默认具有拖拽特性的元素：``和 ``元素默认开启拖拽属性，可以通过设置`draggable="false"`禁止。

示例代码，

```

```

```
<a href="http://www.atguigu.com" draggable="false" ></a>
```

此时img 和 a 不能够被拖拽

“被拖拽元素”事件：事件对象为被拖拽元素

1. dragstart , 拖拽前触发
2. drag ,拖拽前、拖拽结束之间，连续触发
3. dragend , 拖拽结束触发

“释放区”元素事件：事件对象为目标元素

1. dragenter , 进入目标元素触发，相当于mouseover
2. dragover ,进入目标、离开目标之间，连续触发

必须添加：event.preventDefault();

1. dragleave , 离开目标元素触发，相当于mouseout
2. drop , 在目标元素上释放鼠标触发

当有外部文件拖入的时候，必须添加：event.preventDefault();

- 必须在被拖拽元素的ondragstart事件中设置！
- 属性dropEffect：设置拖拽时鼠标的样式，该属性不能单独使用，需要配合effectAllowed一起使用。
- 属性effectAllowed：设置dropEffect鼠标样式生效。

上述两个属性可设置的值包括，none, copy, copyLink, copyMove (默认), link, linkMove, move, all 和 uninitialized。

- 方法 .setDragImage(img, xOffset, yOffset);
 - img: 图片的DOM，
 - xOffset：水平偏移量
 - yOffset：垂直偏移量

以上属性和方法注意必须在：ondragstart监听事件中设置

- 属性files
 - 获取外部拖拽的文件，返回一个filesList列表
 - filesList下有个type属性，返回文件的类型

假设释放区元素的DOM为dropDom

```
dropDom.ondragover = function(ev){
    ev.preventDefault(); // 阻止浏览器默认行为
}
dropDom.ondrop = function(ev){
    ev.preventDefault(); // 阻止浏览器默认行为
    var fileList = ev.dataTransfer.files; //所有被拖拽的网页外的文件
    var len = fileList .length;
    for(var i=0; i<len; i++){
        //使用function，防止变量提升造成异常。
        readFile(dropDom, fileList [i]);
    }
}
function readFile(dropDom, fileObj){
    var fileReader = new FileReader();
    fileReader.onload = function(){
        伪代码 ~~ fileReader.result; // 读取的文件相关信息
        相关代码参考 FileReader ~~~
    }
}
```



HTML5 1px 问题

讲师：许井龙

ngsteel@qq.com

- 1px问题还原
- 设置以下meta标签就会导致移动端出现1px像素问题

```
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=0">
```



HTML5 视口 viewport

讲师：许井龙

ngsteel@qq.com

- 1px问题还原
- 设置以下meta标签就会导致移动端出现1px像素问题

```
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=0">
```



HTML5 语义元素

讲师：许井龙

ngsteel@qq.com

1. 为什么要语义？
2. HTML5新增的布局语义元素
3. article与article嵌套
4. article与section嵌套
5. section与article嵌套
6. IE兼容性
7. time

- 1、HTML元素负责文档内容的结构和含义
- 2、CSS样式负责内容的呈现

我们把字面含义，通俗的表达下。



由于语义化更具可读性，便于团队开发和维护，



没有CSS的情况下，页面也能呈现出很好地内容结构、代码结构



搜索引擎能更好的理解页面中各部分间的关系，可以搜索到更快，更准确的信息

```
<div id="header">
```

```
<div id="nav">
```

```
<div class="article">
```

```
<div class="section">
```

```
<div id="sidebar">
```

```
<div id="footer">
```



```
<header>
```

```
<nav>
```

```
<article>
```

```
<aside>
```

```
<section>
```

```
<footer>
```

header元素

header 元素代表“网页”或“section”的页眉。通常包含h1-h6元素或hgroup，作为整个页面或者一个内容块的标题。也可以包裹一节的目录部分，一个搜索框，一个nav，或者任何相关logo。整个页面没有限制header元素的个数，可以拥有多个，可以为每个内容块增加一个header元素

```
<header>  
  <h1>网站标题</h1>  
  <h2>网站副标题</h2>  
</header>
```

注意事项

1. 可以是“网页”或任意“section”的头部部分；
2. 没有个数限制。
3. **如果hgroup或h1-h6自己就能工作的很好，那就不要用header。**

footer元素

footer元素代表“网页”或“section”的页脚，通常含有该节的一些基本信息，譬如：作者，相关文档链接，版权资料。如果footer元素包含了整个节，那么它们就代表附录，索引，提拔，许可协议，标签，类别等一些其他类似信息。

```
<footer>
```

```
    地址：昌平区平西王府公交站东尚硅谷教学楼 邮箱：info@atguigu.com 微博：weibo.com/u/3272253032
```

```
</footer>
```

注意事项

1. 可以是“网页”或任意“section”的底部部分；
2. 没有个数限制，除了包裹的内容不一样，其他跟header类似。

hgroup元素

hgroup元素代表“网页”或“section”的标题，当元素有多个层级时，该元素可以将h1到h6元素放在其内，譬如文章的主标题和副标题的组合

```
<header>  
  <hgroup>  
    <h1>网站标题</h1>  
    <h2>网站副标题</h2>  
  </hgroup>  
</header>
```

注意事项

1. 如果只需要一个h1-h6标签就不用hgroup
2. 如果有连续多个h1-h6标签就用hgroup
3. 如果有连续多个标题和其他文章数据，h1-h6标签就用hgroup包住，和其他文章元数据一起放入header标签。

nav元素

nav元素代表页面的导航链接区域。用于定义页面的**主要导航部分**。

```
<nav>  
  <ul>  
    <li>HTML 5</li>  
    <li>CSS3</li>  
    <li>JavaScript</li>  
  </ul>  
</nav>
```

注意事项

1. 用在整个页面主要导航部分上。

aside元素

aside元素被包含在article元素中作为主要内容的**附属信息部分**，其中的内容可以是与当前文章有关的相关资料、标签、名词解释等。（特殊的section）

在article元素之外使用作为页面或站点全局的附属信息部分。最典型的是侧边栏，其中的内容可以是日志串连，其他组的导航，甚至广告，这些内容相关的页面。

```
<article>
  <p>文章内容</p>
  <aside>
    <h1>作者简介</h1>
    <p>张三，网络写手。</p>
  </aside>
</article>
```

注意事项

1. aside在article内表示主要内容的附属信息，
- 2. 在article之外则可做侧边栏，没有article与之对应，最好不用。**
3. 如果是广告，其他日志链接或者其他分类导航也可以用

section元素

section元素代表文档中的“节”或“段”，“段”可以是指一篇文章里按照主题的分段；“节”可以是指一个页面里的分组。section通常还带标题，虽然html5中section会自动给标题h1-h6降级，但是最好手动给他们降级

```
<section>
  <h1>section是啥? </h1>
  <article>
    <h2>关于section</h2>
    <p>section的介绍</p>
    <section>
      <h3>关于其他</h3>
      <p>关于其他section的介绍</p>
    </section>
  </article>
</section>
```

注意事项

1. 一张页面可以用section划分为简介、文章条目和联系信息。不过在文章内页，最好用article。**section不是一般意义上的容器元素，如果想作为样式展示和脚本的便利，可以用div。**
2. 表示文档中的节或者段；
3. **article、nav、aside可以理解为特殊的section，所以如果可以用article、nav、aside就不要用section，没实际意义的就用div**

article元素

article元素最容易跟section和div容易混淆，其实article代表一个在文档，页面或者网站中自成一体的内容，其目的是为了让开发者独立开发或重用。譬如论坛的帖子，博客上的文章，一篇用户的评论，一个互动的widget小工具。

```
<article>
  <h1>一篇文章</h1>
  <p>文章内容..</p>
  <footer>
    <p><small>版权：尚硅谷，作者：龙</small>
  </p>
  </footer>
</article>
```

注意事项

1. 一张页面可以用section划分为简介、文章条目和联系信息。不过在文章内页，最好用article。section不是一般意义上的容器元素，如果想作为样式展示和脚本的便利，可以用div。
2. 表示文档中的节或者段；
3. article、nav、aside可以理解为特殊的section，所以如果可以用article、nav、aside就不要用section，没实际意义的就用div

语义元素混搭：article元素嵌套article

```
<article>
  <header>
    <h1>文章标题</h1>
    <p>发布事件<time pubdate datetime="2012-10-03">2012/10/03</time></p>
  </header>
  <p>文章内容..</p>
  <article>
    <h2>评论</h2>
    <article>
      <header>
        <h3>评论者: XXX</h3>
        <p><time pubdate datetime="2012-10-03T19:10-08:00">~1 hour ago</time></p>
      </header>
      <p>哈哈</p>
    </article>
    <article>
      <header>
        <h3>评论者: XXX</h3>
        <p><time pubdate datetime="2012-10-03T19:10-08:00">~1 hour ago</time></p>
      </header>
      <p>哈? 哈? 哈? </p>
    </article>
  </article>
</article>
```

语义元素混搭：article元素嵌套section

<article>

<h1>飞机的组成</h1>

<p>详细描述飞机的组成...</p>

<section>

<h2>机翼</h2>

<p>主要作用是产生升力</p>

</section>

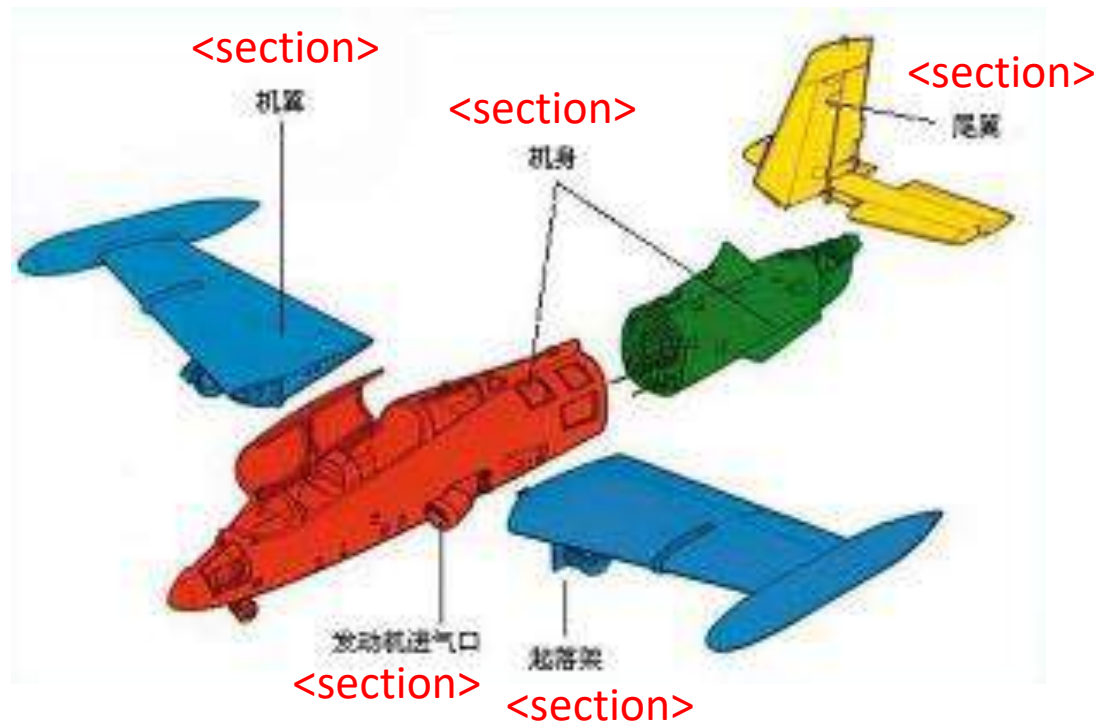
<section>

<h2>尾翼</h2>

<p>起稳定和操纵作用</p>

</section>

</article>



组装后就是一架飞机 <article>

语义元素混搭：section 元素嵌套 article

<section>

<h1>TFBOYS组合</h1>

<article>

<h2>易烱千玺</h2>

<p>2000年11月28日出生于湖南省怀化市...</p>

</article>

<article>

<h2>王俊凯</h2>

<p>1999年9月21日生于中国重庆..</p>

</article>

<article>

<h2>王源</h2>

<p>2000年11月8日出生于重庆..</p>

</article>

</section>



易烱千玺

<article>

王俊凯

<article>

王源

<article>

TFBOYS

<section>

使用新语义元素

- ◆ 没有语义的情况下，就需要是使用DIV了。
- ◆ 注意：千万不要为了语义而语义

使用：html5shiv-master

直接应用源文件即可。

```
<script src="js/html5shiv.js" type="text/javascript"></script>
```

- `<progress>` 用来显示一项工作做完成的进度。
 - **元素/DOM属性**：**max** ~ 大于0的浮点型数字，表示完成一项工作总的“工作量”。
 - **元素/ DOM属性**：**value**
 1. 通常情况 value值为0 – max 的任意浮点型数字，表示已经完成了max指定工作量的多少。
 2. 无论是否设定了max的值，如果不指定value的值，进度条是不确定的，这表明一个活动正在进行中，没有迹象显示它需要多长时间。
 3. 如果没有设定max的值，value的取值范围0~1浮点任意浮点数，表示完成工作的进度。
 - **DOM属性（只读）**：**position**，返回一个double值，即value除以max的结果；如果进度条是一个不确定的进度条（比如没有设置max和value），它返回- 1。

- `<address>` 表示联系信息，无特殊属性，使用说明
 1. 要表示联系信息。例如电话，地址，邮箱，网站地址等。如果某一内容与不是地址信息，使用`<p>`元素，而不是`<address>`元素。。
 2. 通常一个`<address>`元素可以放页面的`<footer>`元素中，如果有。

- pre 元素可定义预格式化的文本。被包围在 pre 元素中的文本通常会保留空格和换行符。而文本也会呈现为等宽字体。
- 最佳使用场景：展示程序源代码



1985-04-12T23:20:50.52Z

This represents 20 minutes and 50.52 seconds after the 23rd hour of April 12th, 1985 in UTC.

1996-12-19T16:39:57-08:00（东八区时间） This represents 39 minutes and 57 seconds after the 16th hour of December 19th, 1996 with an offset of -08:00 from UTC (Pacific Standard Time). Note that this is equivalent to 1996-12-20T00:39:57Z in UTC.

1990-12-31T23:59:60Z

This represents the leap second inserted at the end of 1990.

1990-12-31T15:59:60-08:00

This represents the same leap second in Pacific Standard Time, 8 hours behind UTC.

1937-01-01T12:00:27.87+00:20

This represents the same instant of time as noon, January 1, 1937, Netherlands time. Standard time in the Netherlands was exactly 19 minutes and 32.13 seconds ahead of UTC by law from 1909-05-01 through 1937-06-30. This time zone cannot be represented exactly using the HH:MM format, and this timestamp uses the closest representable UTC offset.



元素属性操作 技术方案对比

讲师：许井龙

ngsteel@qq.com

	HTML4 attribut	HTML5 classList 对象	HTML5 dataset 对象
操作范围	操作元素的所有属性，包括元素的自定义属性（无论是否遵循HTML5 data-开头的属性）	仅能操作元素的class属性	只能操作元素的自定义属性，自定义属性规定以 data-开头。
方法特点	新增，或删除属性，原来的属性值被“覆盖”	新增，或删除属性值 不会覆盖原来的属性 。	新增，或删除属性，原来的属性值被“覆盖”
使用场景	兼容性好	只能在高版本浏览器，简单易用。 有 样式切换效果 ，推荐使用toggle()。	只能在高版本浏览器，存储页面程序使用数据。
可设置的值	根据CSS规范要求的值（自定义属性值，参考自定义属性）	类选择器名称，多个类选择器以空格符分隔	只能是字符串或者数字，如果存储json对象，需要基于JSON提供的方法转化格式。



HTML5 Canvas

讲师：许井龙

- 一、关于Canvas
- 二、Canvas API
- 三、实战

- `<canvas>` 最早被Apple引入，用于Mac OS X 的 Dashboard,后来又在Safari和Google Chrome被实现，后来被纳入到HTML 5中。
- `<canvas>`是HTML5新增的元素。这个元素负责在页面设定一个区域。然后就可以通过Canvas提供的API在这个区域描绘2D和3D图形。
- IE9+,Firefox1.5+,Safari2+,Opera9+ , Chrome,IOS版的Safari以及Android版的Webkit都支持`<canvas>`
- 目前该元素对于Canvas 3D支持不够理想。仍处于试验阶段。

- 动态产制图表
- 开发在线与离线游戏
- 制作动画效果
- 与视频，音频交互

- 基本用法：

- 1、设置画布：必须指定<canvas>元素的宽度与高度，也就是说设定“画布”的大小。

```
<canvas id="mycanvas" width="1024" height="768">
```

抱歉，你的浏览器不支持Canvas元素

```
</canvas>
```

注意不要通过CSS的width和height定义canvas的宽高，这是规范的要求，且高度与宽度不能设置px单位（默认宽高 300 * 150 ）

- 2、取得这块“画布”的上下文，只有获得该上下文才可以基于其提供的API进行画图。

```
//获取“画布”的DOM对象
```

```
var mycanvas = document.querySelector("#mycanvas");
```

```
//首先需要判断浏览器是否支持画布
```

```
if(mycanvas.getContext){
```

```
    //获取画布上下文
```

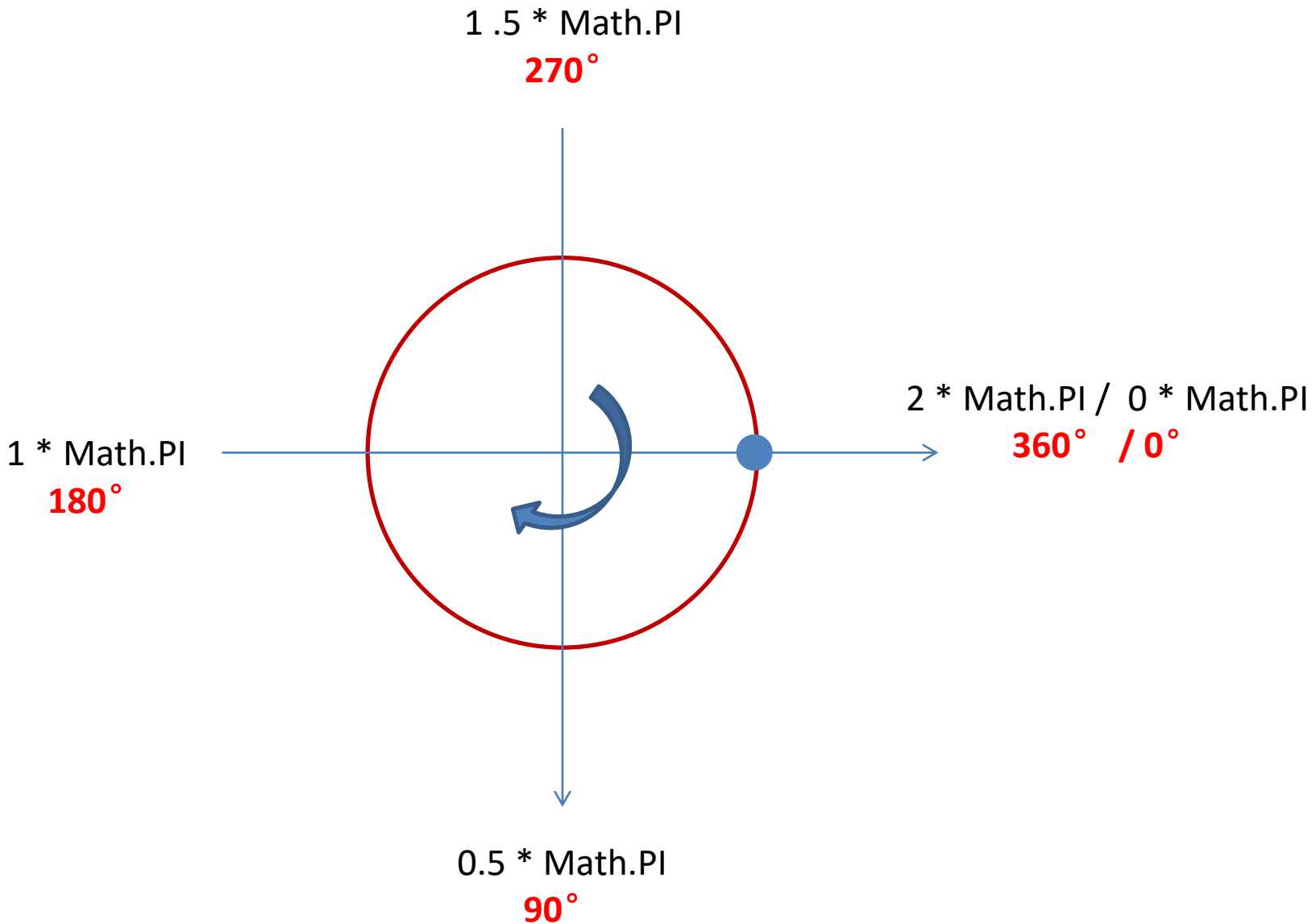
```
    var painting = mycanvas.getContext("2d");
```

```
}
```

- 绘制2D图形：
 - 我们可以利用Canvas上下文对象提供的API绘制简单的2D图形。例如矩形、弧形和路径（线）。
- 描边和填充：2D绘图就两种基本操作 — 描边和填充。
 - 描边：`painting.strokeStyle = red`; 设置图形边缘线的颜色为红色。
 - 填充：`painting.fillStyle = blue`；设置图形的填充色是蓝色。
- 补充说明：
 - 除了颜色，还可以设置“渐变色”和“图像”。如果我们不指定任何值，默认颜色都是黑色。（联想border的默认边框色）
- 设置画笔的宽度
 - `lineWidth`：number。

- 绘制2D矩形：
 - 描边矩形：**strokeRect**(x , y , width, height)
 - x: 距<canvas>元素左上角右侧的距离
 - y: 距<canvas>元素左上角下方的距离
 - width: 矩形的宽度
 - height: 矩形的高度
 - **备注：上述四个值不需要单位，默认就是px**
 - 填充矩形：**fillRect**(x , y , width, height);
 - 参数同描边矩形。
 - 清理出一个矩形区域：**clearRect**(x , y , width, height) , 该方法让我们可以“扣出”一个透明的矩形区域。
 - 参数同描边矩形

- 绘制路径：绘制复杂的形状与线条。
- 1、开始绘制路径：**绘制路径必须调用beginPath()**，表示要开始绘制新路径。然后通过下列方法绘制实际的路径。
- 2、绘制一个圆形路径：`arc(x, y, radius, startAngle, endAngle, counterclockwise)` 以 (x, y) 为圆心绘制一条弧线，`startAngle`表示起始角度，`endAngle`表示结束角度，`counterclockwise`表示起始角度和结束角度是否按逆时针方向计算，**值false表示按顺时针方向计算。**
- 3、绘制结束路径
 - **closePath()**：绘制一条到起点的线。
 - `fill()`: 路径绘制完成后，进行填充。
 - `stroke()`: 路径绘制完成后，进行描边。
 - `clip()`: 依照前面创建的路径进行一次切割，然后所有在clip之后画的图形都只有在被切割的路径内才会显示，路径外的则不得显示。



- 绘制路径：
 - `moveTo(x, y)`：将“画笔”移动到 (x, y) 。
 - `lineTo(x, y)` 从一点开始绘制直线，直到 (x, y) 为止。

一、内角和公式：

$$\theta = 180^\circ \cdot (n - 2)$$
 公式中n为多边形的边数。

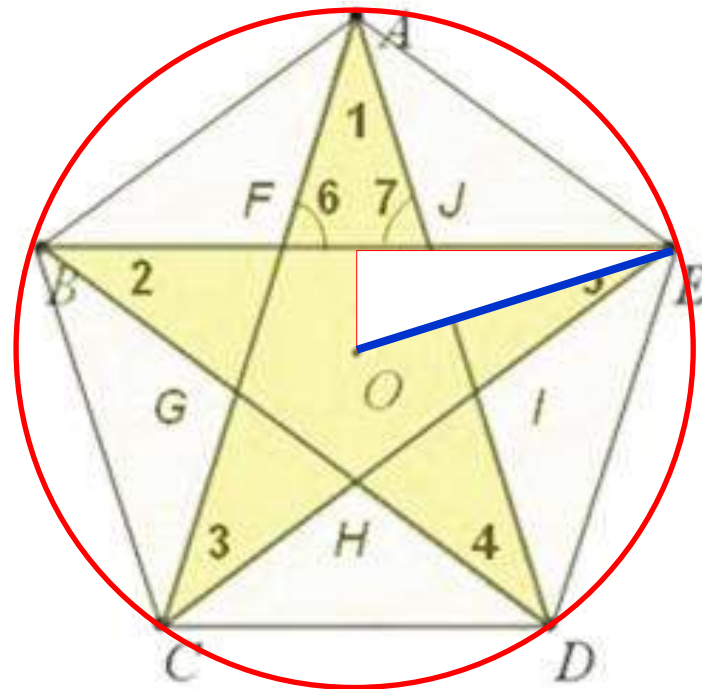
二、五边形的内角和： 540°

三、五边形每个内角： $108^\circ = 540^\circ / 5$

四、五角星的每个角： $36^\circ = (180^\circ - 108^\circ) / 2$

五、角度转弧度：角度/180 * π

六、Math.sin(弧度) 和 Math.cos(弧度)

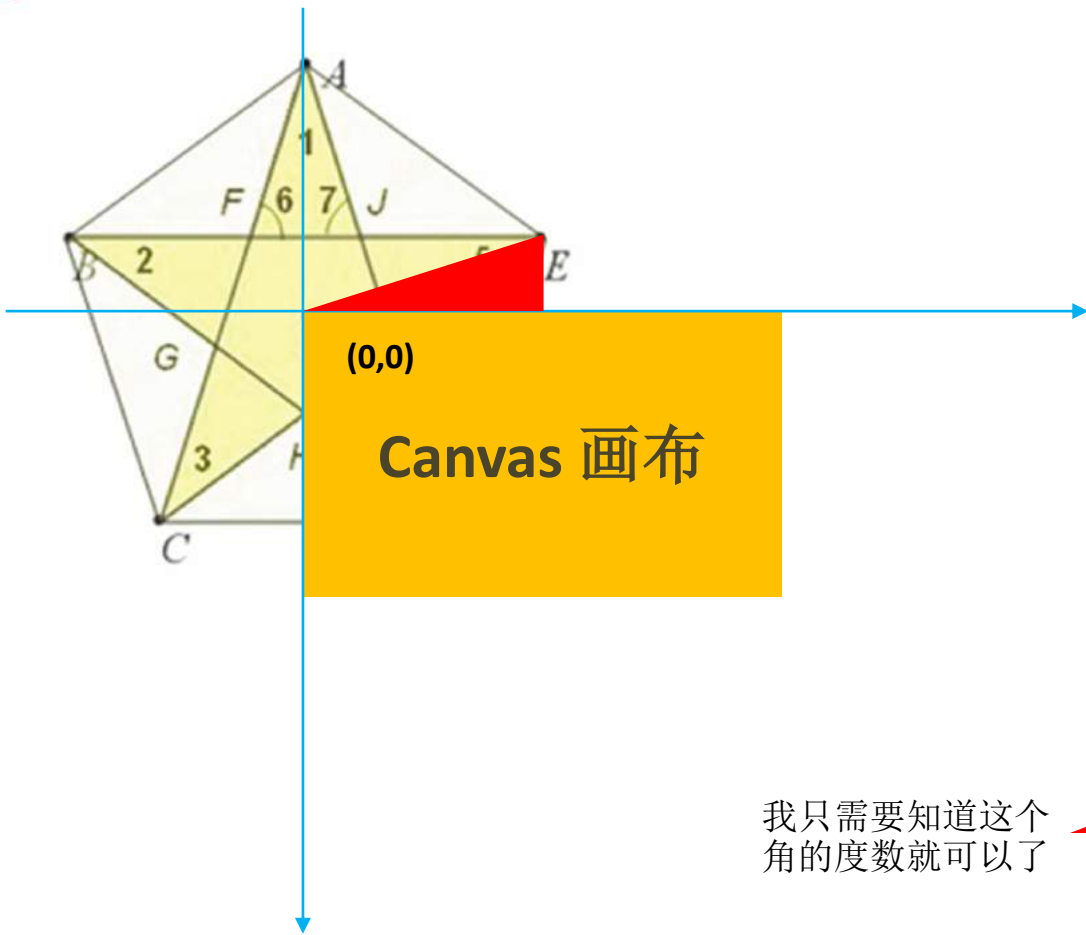


E, A,B,C,D 点的坐标(x, y)规律

$$\text{Math.cos}((18 + 72*i)/180 * \text{Math.PI}) * R + 200 ,$$

$$-\text{Math.sin}((18 + 72*i)/180 * \text{Math.PI}) * R + 200$$

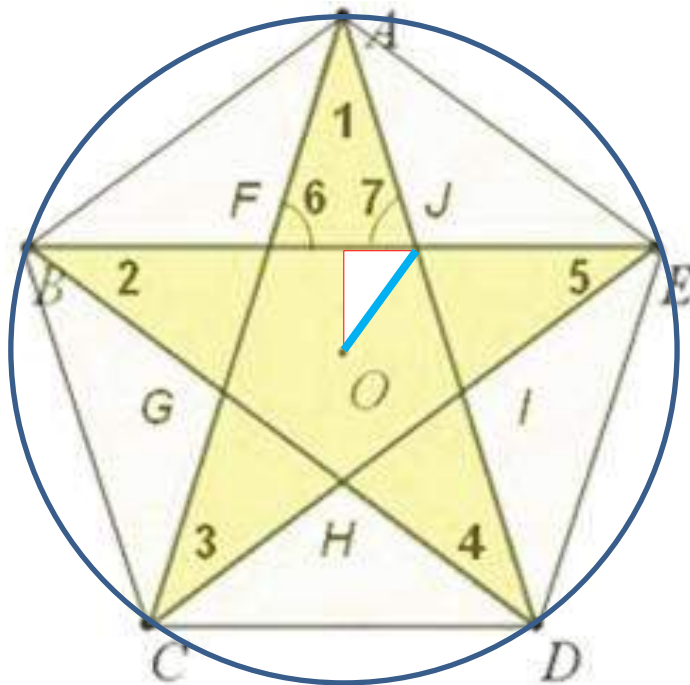
半径 R



斜边
决定了五角
星的大小

我只需要知道这个
角的度数就可以了



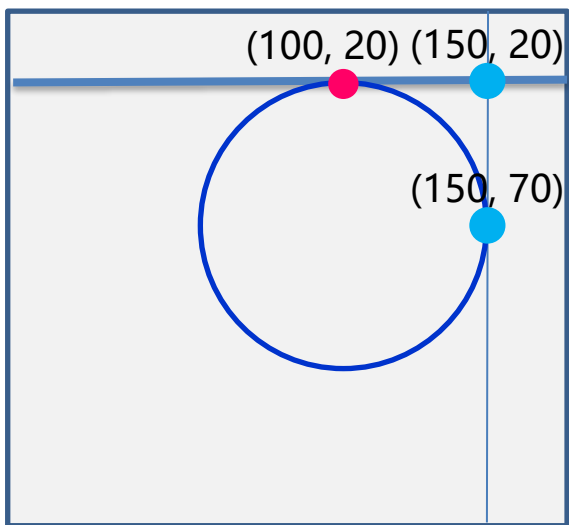


J,F,G,H,I 点的坐标

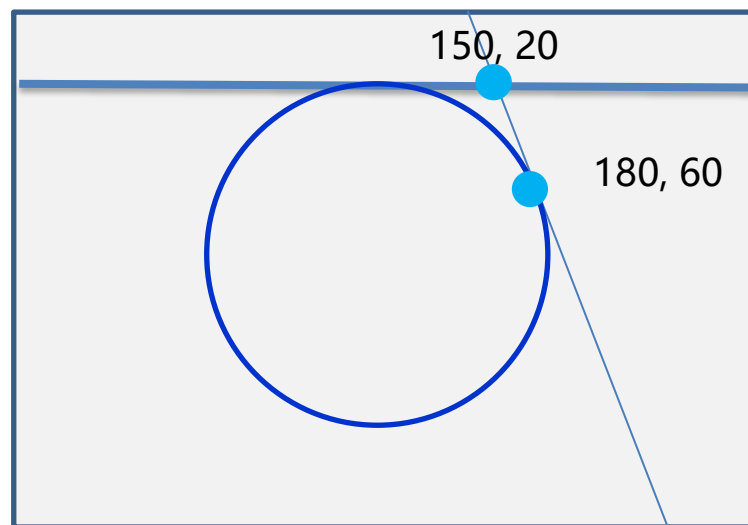
$\text{Math.cos}((54 + 72*i)/180 * \text{Math.PI}) * r + 200 ,$
 $-\text{Math.sin}((54 + 72*i)/180 * \text{Math.PI}) * r + 200$

- 绘制路径-`rect(x, y, width, height)` 从(x, y)开始绘制一个矩形，宽度和高度分别是width, height. 这个方法仅仅是绘制一个矩形的路径。

- 绘制路径-弧线：arcTo (x1, y1, x2, y2, radius) 在画布上创建介于**两个切线之间的弧/曲线**。



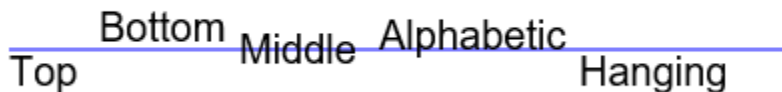
```
ctx.moveTo(20,20); // 创建开始点  
ctx.lineTo(100,20); // 创建水平线  
ctx.arcTo(150,20,150,70,50); // 创建弧  
ctx.lineTo(150,120); // 创建垂直线
```



```
painting.moveTo(20, 20);  
painting.arcTo(150, 20, 180, 60, 70);  
painting.lineTo(200,120);
```

小练习：制作圆角矩形

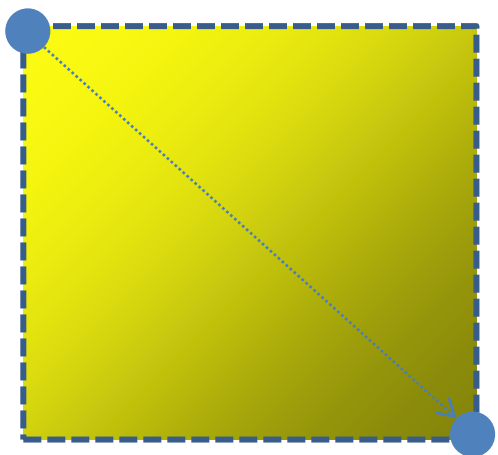
- 绘制文本：文本与图形如影随形。为此，2D绘图上下文也提供了绘制文本的方法。绘制文本的两个主要方法，
 - `fillText(文本字符串, "x坐标", "y坐标", 最大像素宽度[可选])`，需要使用`fillStyle`属性设置字体颜色。该方法比较常用
 - `strokeText(文本字符串, "x坐标", "y坐标", 最大像素宽度[可选])`，需要使用 `strokeStyle`属性设置字体颜色。
- 上述两个方法可以通过下面的属性控制样式，
 - **font**：表示样式、大小和字体。例如，“bold 18px Microsoft YaHei”
 - **textAlign**：表示文本的对齐方式。可能的值start, end, left, right和center。建议使用 start 和 end, 不要使用left 和 right。因为start 和 end语言性更强。
 - **textBaseline**：表示文本的基线。
 - top,
 - hanging,
 - middle,
 - alphabetic：默认值
 - bottom



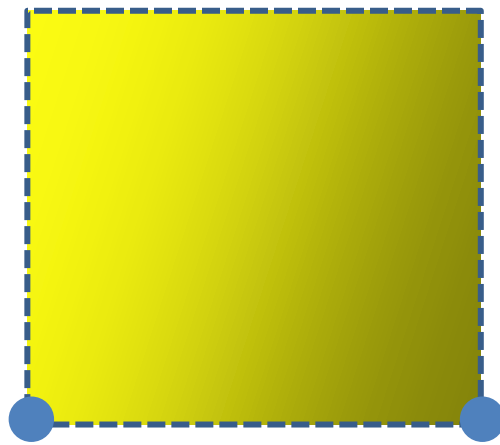
匆匆那年我们 究竟说了几遍 再见之后再拖延
可惜谁有没有 爱过不是一场 七情上面的雄辩
匆匆那年我们 一时匆忙撂下 难以承受的诺言
只有等别人兑现
不怪那吻痕还 没积累成茧
拥抱着冬眠也没能 羽化再成仙
不怪这一段情 没空反复再排练
是岁月宽容恩赐 反悔的时间
如果再见不能红着眼 是否还能红着脸
就像那年匆促 刻下永远一起 那样美丽的谣言
如果过去还值得眷恋 别太快冰释前嫌
谁甘心就这样 彼此无挂也无牵
我们要互相亏欠 要不然凭何怀缅

- **线性渐变** : `createLinearGradient(startX, startY, endX, endY)`
 - (startX , startY) -- 起点坐标
 - (endX , endX) -- 终点坐标
- **实际运用**
 - 对角线渐变举例 : `createLinearGradient(20, 20, 100, 100)`
 - 水平渐变举例 : `createLinearGradient(20, 100, 100, 100)`

起点
(20, 20)



终点
(100, 100)



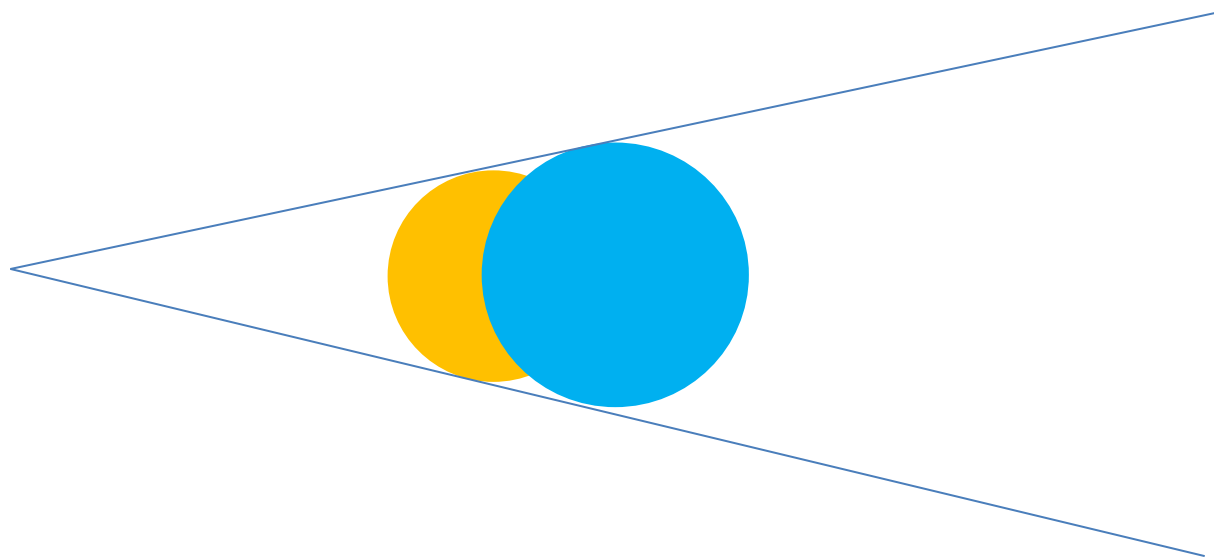
起点
(20, 100)

终点
(100,, 100)

- `gradient.addColorStop(number, color);`
 - `number` : 色标位置 (0, ~ 1 之间的任意数字)
 - `color` : CSS颜色

注意：为了让渐变的区域覆盖矩形区域，需要让矩形和渐变区域“匹配”

- 径向渐变：`createRadialGradient(x1, y1, radius1, x2, y2, radius2)`
 - $(x1, y1)$ ：起点圆的圆心
 - `radius1`：起点圆的半径
 - $(x2, y2)$ ：终点圆的圆心
 - `radius2`：终点圆的半径



- shadowOffsetX : 形状或路径Y轴方向阴影偏移量。默认0
- shadowOffsetY : 形状或路径x轴方向阴影偏移量。默认0
- shadowBlur : 模糊像素, 默认0, 即不模糊
- shadowColor : 阴影颜色

- drawImage (imgObj, x, y)
- drawImage (imgObj, x, y, imageWidth, imageHeight)

实例代码：

```
var img = new Image();
```

```
img.src = "img/logo.png"
```

```
img.onload = function(){
```

```
    ctx.drawImage(img,0,0);
```

```
};
```






HTML



HTML5 FileReader

讲师：许井龙

ngsteel@qq.com

- 一 . FileReader概述
- 二 . 属性
- 三 . 方法
- 四 . 事件

- 图片上传预览
- 文本编辑预览

出于安全因素，浏览器网页无法直接显示用户磁盘上的图片文件或者文本内容

如果实现上传预览功能，我们需要读取用户选中的文件，然后对齐进行处理，转换为浏览器可以识别的格式。

通过FileReader接口提供的异步API，浏览器可以把文件读入内存，并且读取文件中的数据。

- readyState
 - EMPTY : 0 : 尚未加载数据.
 - LOADING : 1 : 数据正在加载.
 - DONE : 2 数据已经读取完毕.
- result : 返回文件的内容。此属性仅在读操作完成后才有效，并且数据的格式取决于使用哪一方法来启动读操作。

- readAsDataURL()



- loadstart
- **progress**
- abort
- **error**
- **load**
- loadend



HTML5 Geolocation API

讲师：许井龙

- 一、关于地理定位
- 二、获取当前地理坐标
- 三、实时获取地理坐标
- 四、实战：百度地图

一、关于地理定位

- 通过H5提供的地理定位API，我们可以在用户授权后（共享地理位置），获取设备的地理位置。
- 该API由window.navigator.geolocation提供。

- IE9+, Firefox3.5+ , Opera10.6+ , Safari5+ Chrome , IOS版本的Safari,Android Webkit支持地理信息API。
- 目前由于google在中国大陆地区使用受限。所有基于Google定位的浏览器都无法获取地理位置信息。

```
▼ PositionError {code: 2, message: "Network location provider at 'https://www.googleapis.com/' : No response received."} ⓘ  
  code: 2  
  message: "Network location provider at 'https://www.googleapis.com/' : No response received."
```

- 在IE10及其以上浏览器可以使用地理信息相关功能。

- geolocation提供了两个获取地理信息的方法。
 - `getCurrentPosition ()`
 - `watchPosition ()`
 - `clearWatch()`

二、获取当前地理坐标

- `getCurrentPosition(function1(position){}, function2(err){}, optionObj);`
- 该方法可获取设备当前位置信息。有3个参数：
- 参数一：回调函数。如果成功获取位置信息，该回调函数会接收到一个 `position` 对象。 `position` 对象有两个属性：`coords` 和 `timestamp`（获取定位的时间戳）。 `coords` 又包含如下属性，
 - `longitude`: 经度坐标
 - `latitude`: 纬度坐标
 - `accuracy`: 经纬度坐标的精度（单位 米，值越大定位约不准）。
 - `altitude`: 海拔高度（单位 米）
 - `altitudeAccuracy`: 海拔高度精度（单位 米，值越大精度越差），
 - `heading`: 指南针方向。0°正北，90°正东，180°正南，270°正西。
 - `speed`: 速度（单位 米/秒）

二、获取当前地理坐标

- 参数二：回调函数（可选），当设备无法获取地理信息时，该函数被调用。此时会接收到一个err对象。该对象有message和code属性。
 - message: 描述错误的原因。
 - code: 错误状态码
 - 0 未知错误
 - 1 用户拒绝共享地理位置信息。
 - 2 位置无效
 - 3 访问超时
- 参数三：选项对象（可选），一个JSON对象包含三个属性。
 - timeout: number（可选）。等待返回位置信息的最长时间（单位毫秒），
 - enableHighAccuracy: boolean（可选）。true表示尽可能使用精确的位置信息。除非确实非常需要精确的信息，否则建议使用false。如果设置true，数据返回时间慢，还会导致移动设备耗用更多电量。
 - maximumAge: number（可选）。上一次取得坐标的有效时间（单位毫秒），如果时间到，重新获取新的坐标信息。如果不需要平凡更新用户位置信息，建议设置为infinity。从始至终都使用第一次获取的位置信息。

- `watchPosition(function1(position){}, function2(err){}, optionObj) ;`
- 等待系统发出位置改变信号（不会主动轮询），该函数返回一个标识符。用于跟踪监控操作。`clearWatch()`通过该标识符取消监控操作（类似`setTimeout()`和`clearTimeout()`）。
- 该函数参数及作用与`getCurrentPosition()`完全相同。

百度地图官网

<http://lbsyun.baidu.com/index.php?title=jspopular>

注意：自v1.5版本起，您需先申请密钥（ak）才可使用。

实战：
集成百度地图至自己的页面。



尚硅谷

www.atguigu.com

```
C:\Users\admin>tracert www.baidu.com

通过最多 30 个跃点跟踪
到 www.a.shifen.com [119.75.218.70] 的路由:

  1    3 ms    3 ms    2 ms  XiaoQiang [192.168.31.1]
  2   21 ms   7 ms    5 ms  1.64.45.113.in-addr.arpa [113.45.64.1]
  3    8 ms   11 ms   3 ms  124.205.97.50
  4    5 ms    5 ms   3 ms  124.205.97.50
  5   24 ms    4 ms   3 ms  37.165.241.218.in-addr.arpa [218.241.165.37]
  6    5 ms    5 ms   3 ms  17.177.197.14.in-addr.arpa [14.197.177.17]
  7   19 ms    4 ms   6 ms  222.254.160.168.in-addr.arpa [168.160.254.222]
  8    *      4 ms   3 ms  192.168.0.50
  9    5 ms    4 ms   4 ms  10.34.240.22
 10    9 ms    7 ms   3 ms  119.75.218.70

跟踪完成。
```

查询结果	192.168.31.1	局域网 ?
查询结果	113.45.64.1	北京市, 电信通
查询结果	124.205.97.50	北京市, 电信通
查询结果	124.205.97.50	北京市, 电信通
查询结果	218.241.165.37	北京市, 电信通
查询结果	14.197.177.17	北京市, 电信通
查询结果	168.160.254.222	北京市, 中国科学技术信息研究所
查询结果	192.168.0.50	局域网 ?
查询结果	10.34.240.22	局域网, 对方和您在同一内部网 ?
查询结果	119.75.218.70	北京市, 北京百度网讯科技有限公司BGP节点



HTML



HTML5 新特性

讲师：许井龙

ngsteel@qq.com

- 一、新的Javascript选择器
- 二、DOM的classList属性（侧边栏）
- 三、data- 自定义属性（懒加载）
- 四、CSS3 attr() 属性函数（打印）
- 五、CSS3 calc() 计算函数（布局）
- 六、CSS3 rgba() 颜色函数（透明色）
- 七、CSS3 counter() 计数函数
- 八、JSON序列化与反序列化

- document.querySelector(selectors)
 - document.querySelectorAll(selectors)
 - document.getElementsByClassName(classvalue)
-
- selectors : CSS选择器
 - classvalue : 元素 class 属性值

- 获取元素所有的class名
- 属性：length --- class的长度
- 方法
 - add(value) 添加class方法
 - contains(value) 元素是否包含value指定的值
 - remove(value) 删除class方法
 - toggle(value) 切换class方法

- 点击某一按钮，显示侧边栏。

- 自定义属性：
 - 以data-开头的属性。是将自定义的数据存储到页面或应用程序中，**注意这些数据不是用来呈现给用户的。**
 - 属性定义：dataset。它是DOM对象的一个新增属性。

HTML代码

```
<div id="myattr" data-md-width="140px" data-bg-width="180px"></div>
```

JS代码

```
var myattr = document.querySelector("#myattr");  
document.write(myattr.dataset.bgWidth); //读取值  
myattr.dataset.bgWidth = "26px"; //设置新值
```

注意：自定义属性data-后的名字可以使用-分割，在基于dataset获取该属性是，-后面的单词首字母大写。

- 懒加载 (lazy loading)

- attr() 获取元素属性值，通常与CSS的content属性搭配使用。
- 注意：attr() 对content以外的属性的支持是实验性的。在本教材编写时，高级用法仍未获得支持。

```
@media print {  
    /*最简单的使用方式*/  
    a:after{  
        content: " [" attr(href) "]" ;  
    }  
}
```

`尚硅谷`

/* 指定属性类型 */

attr(src url);

attr(data-count number);

attr(data-width px);

/* 设置回退方案 */

attr(data-count number, 0);

attr(src url, "");

attr(data-width px, inherit);

attr(data-something, 'default');

- CSS3函数calc()可以用在任何一个需要“尺寸”的地方，该函数可以动态计算“尺寸”。
- 使用方法：
 - line-height: calc(1em + 2px);
 - width: calc(100% - 100px);
 - font-size: calc(16px * 2)
 - height: calc(100% / 2)
 - 你还可以在一个calc()内部嵌套另一个calc()。例如，width: calc(100% - calc(800px + 200px)) 目前仅欧朋浏览器支持calc嵌套。
- 注意事项：
 - 必须在运算符左右加空格。
 - 在低版本的浏览器使用，需要增加浏览器厂商前缀。

三列布局：中间自适应，两边固定

- rgba() 颜色透明度
- 语法：rgba(<rgb-component>#{3}, <alpha-value>)

- rgba() 元素颜色透明度
- opacity 元素及内容、子元素透明度
- transparent 透明色

- counter-reset: **counter-name** ;
 - 相当于定义了一个变量，且初始值为0
 - 该声明设置counter()元素的父元素中。
- counter-increment: **counter-name** ;
 - 使用前自增1.
- **content**: counter(**counter-name**);
 - 把自增后的数据给content属性
- **content**: counters(**counter-name**, ".");
 - 二级自增

- `var str = JSON.stringify(jsonObj);`
 - 序列化JSON对象
- `var jsonObj = JSON.parse(str);`
 - 反序列化JSON对象



HTML5 WebSocket

讲师：许井龙

- 一、关于HTTP
- 二、Websocket的诞生
- 三、Websocket Client API
- 四、实战：简版智能机器人

一、关于HTTP

- HTTP101 (即HTTP1.0 和 HTTP1.1)
- HTTP1.0 每个请求需要一个建立一个单独的连接。
- HTTP1.1 改进。多个请求可以共用一个连接。这种改进减少了连接数量,降低请求延时。比如我们可以重用之前访问网页的连接,再读取该网页的图片,JS脚本等。

▼ Request Headers [view source](#)

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

Accept-Encoding: gzip, deflate, sdch, br

Accept-Language: zh-CN,zh;q=0.8

Cache-Control: max-age=0

Connection: keep-alive

Cookie: PSTM=1467337067; BAIDUID=4391077FDF802D7058FB58D14535E26C:FG=1; BIDUPSID=283A7!DdjUHZUulozRmV-aWtSSVlxYmpMaGxvY3V0OC1XcnhvMmllocUJYQVFBQUFBjCQAAAAAAAAAAAAEAAACKtgQoyftAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAKL5eFei-XhXc; plus_cv=1::m:3f72b740; ispeed=1; ispD=1437_20516_20537_17944_20415_20455_18560_15625_11605_20592; BD_UPN=12314753; sugstor

Host: www.baidu.com

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)

1.1、服务器端：keep-live

- 1、如果服务器内存充裕，KeepAlive = On还是Off对系统性能影响不大。
- 2、如果服务器上静态网页(Html、图片、Css、Js)居多时，建议打开KeepAlive。
- 3、如果的服务器处理的多为动态请求(因为连接数据库，对文件系统访问较多)，KeepAlive 关掉，会节省一定的内存，节省的内存正好可以作为文件系统的Cache(vmstat命令中cache一列)，降低I/O压力。

Request Headers

[view source](#)

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

Accept-Encoding: gzip, deflate, sdch

Accept-Language: zh-CN,zh;q=0.8

Cache-Control: max-age=0

Connection: keep-alive

Cookie: user-key=834315f7-c1fe-46a9-91c5-61f45ff8bdf0; _tp=aH9xbAuJp5gC68KWQ8HqOw%3D%3D; unick=%E8%AE%B8%E4%8A%95%E9%BE%99; _pst=u_6e07dd5508d9c; TrackID=1fCjzm6isHbaGUBtYwgSv906HuPaSki96eQ1Y6vAimSZ467KD461kkak3xLCVD83H34XkExafIo4r8Z02ZH80pdaOwerbFqxbmA1-HuF3eHo; pinId=dUk6G9Y_Ysf0kDTHM1iYfw; pin=u_6e07dd5508d9c; __jdv=122270672|direct|-|none|-; cn=1; ipLocation=%u5317%u4EAC; areaId=1; ipLoc-djd=1-72-2839-0; __jda=122270672.0x09562da8b8ce0328.1465397903949.1467121111.1467166740.16; __jdb=122270672.1.0x09562da8b8ce0328|16.1467166740; __jdc=122270672; thor=0E3755448D53A42A4EEC8A299C71A11F20C888595AF266C8ADF06BFD7F285A6B1AB69D97C8ADBFBFD9D28D3EFA33033EBE3A7FBB9AD85220C5802DF251B351F230371301CB6FDF721295105891F8AF9B6461535833D5D0E0D11D5F8AD10F749C012B784A315A32F917E7C2D45640186441E3461B468F7AD3578863A212A76D9A96A305A950D4708DAFD8C9AF961A4D5EB5B03DE92FA7B3DF7A88979FA0F8DB89; __jdu=0x09562da8b8ce0328

Host: www.jd.com

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/50.0.2661.102 Safari/537.36

京东首页请求头（基于HTTP1.1）

```
▼ Response Headers    view source
Age: 1
Cache-Control: max-age=20
Content-Encoding: gzip
Content-Length: 43018
Content-Type: text/html; charset=gbk
Date: Wed, 29 Jun 2016 02:25:37 GMT
Expires: Wed, 29 Jun 2016 02:25:47 GMT
ser: 101.107
Server: JDWS
Vary: Accept-Encoding
Via: BJ-Y-NX-104(HIT), http/1.1 BJ-GWBN-1-JCS-162 ( [cRs f ] )
```

京东首页响应头（基于HTTP1.1）

- 从上面的例子，我们可以看出，仅请求头的Cookie就有将近1200多字节。这个例子说明，不管服务器端是否向客户端（例如浏览器），这些信息都会随着每次请求与响应来回在网络上传输。

1.3、保持会话状态

- 由于HTTP是无状态，每个请求之间彼此独立。服务器不“关心”两个请求是否来自同一个客户端（浏览器）。如果需要保持状态，开发人员需要在每次请求和响应时加入冗余的信息，已区分哪些请求来自同一个客户端。

1.4、HTTP “半双工协议”

- HTTP是“半双工”的协议。即请求必须在响应完成后进行，反之响应必须在请求完成后。请求与响应不能同步进行。这种工作方式非常低效。

```
<script type="text/javascript">
  //添加脚本
  var xmlHttp;
  if(window.XMLHttpRequest){
    xmlHttp = new XMLHttpRequest();
  } else {
    xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
  }

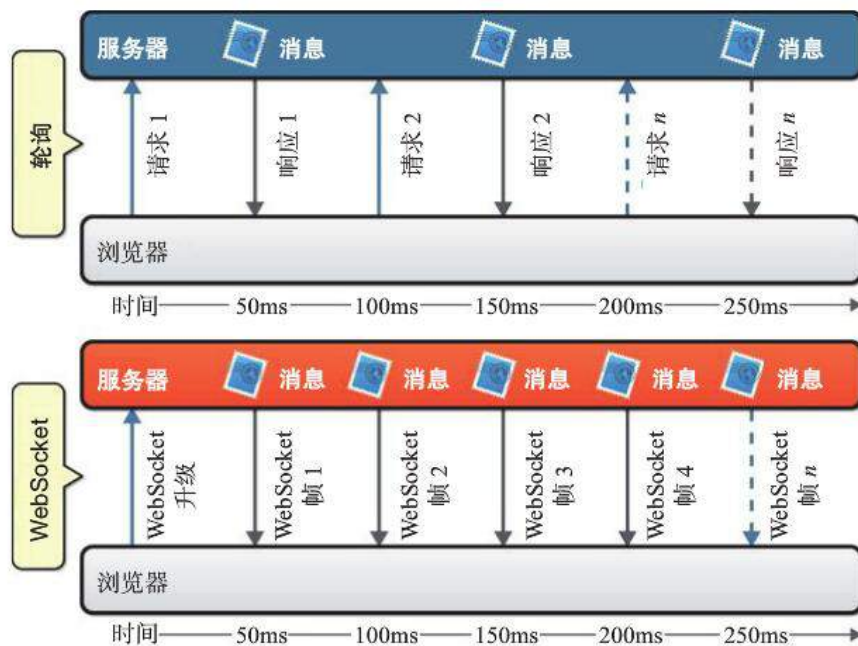
  xmlHttp.open("GET", "http://www.baidu.com", true);
  xmlHttp.send();
</script>
```

XMLHttpRequest cannot load <https://www.baidu.com/>. No 'Access-Control-Allow-Origin' header is present on the requested resource. Origin 'null' is therefore not allowed access.

- 1、无法实时真正意义上的长连接
- 2、每次请求-响应都会附加非常多的信息。
- 3、无法保持会话状态。
- 4、半双工通讯方式，效率低下。
- 5、无法实现跨域访问。

- 如何解决HTTP1.0 和 HTTP1.1低效及实时性问题
 - **轮询 (polling)** : 定期发起一个请求，服务器每次都会响应，不管是否由信息更新。如果信息变化由规律，使用轮询当然是最好的方法，但是很多信息变化是没有规律的。这会导致我们打开很多无用的连接。
 - **长轮询** : 客户端向服务器发送一个请求，并创建一个连接，该连接一直会处于打开状态，直到服务器有信息更新或者连接超时。常见的技术为**Comet**。该技术的本质是延长了服务器响应时间。它有时候也被称为“挂起GET”或者“搁置POST”。由于服务会挂起很多请求连接，如果访问过度，会导致服务器宕机。
 - **流化** : 本质是服务器永远不会响应请求，它只是更新响应。看起来实时更新信息的问题已经得到了妥善解决。但是任何商业网站一定会在其主站前加入防火墙或者代理服务器，而这些服务器可能缓存响应。导致响应更新的延迟。例如，Flash Socket。缺点是客户端必须安装Flash插件；非HTTP协议，无法自动穿越防火墙。

- 实现长连接
- 减少无用信息传递，节省带宽。
- 全双工（Full Duplex）：请求与响应可以同步进行。
- 减少延迟：一旦建立WebSocket连接，只要服务器有可用消息就会推送到客户端。它与轮询由本质区别，只有一个连且实时“推送”信息。
- 可以跨源（origin）请求第三方服务
- 简单易用



- 1、如果开发实时通讯的应用，websocket是你最佳的选择，例如聊天、写文档编辑，大型多人在线游戏，股票交易软件。该技术尤其适合HTML5游戏开发，因为在线有限对响应实时性要求非常高。
- 2、如果你需要全双工通讯，那么websocket绝对是最佳选择，但是如果客户端只是被动的接受服务端推送的消息（例如天气预报、新闻），那么使用服务器推送事件（Server-Sent, Event, SSE）提供的接口也是个不错的选择。

- WebSocket由Client 和 Server两部分组成。
 - Client API (应用程序编程接口) , 包括事件、方法和特性。

- 构造函数
- `var websocket = new WebSocket("ws://echo.websocket.org/ ")`
- 该构造函数由一个必要的参数URL（提供websocket的服务地址，该地址必须以ws://开头）和一个可选参数（protocols），该可选参数是一个字符数组。
- 执行完此代码后，http协议升级为websocket协议。一旦建立连接（`onopen()`）消息就可以在客户端和服务端来回传送。客户端可以通过异步事件监听连接生命周期的每个阶段。

- open
- message
- error
- close

WebSocket是纯事件驱动的，应用程序只需要监听WebSocket对象上的事件，就可以处理输入数据和控制连接状态的改变。

- 当服务器响应了Websocket连接请求，open事件触发并建立一个连接。open对应的回调函数是onopen().此时Websocket已经准备好发送和接收数据。

```
websocket.onopen = function(event){  
    console.log("已经建立了websocket连接。");  
}
```

- 当服务器推动消息时，触发message方法。此时会回调onmessage()。以下代码是接收文本消息

```
websocket.onmessage = function(event){  
  var msg = event.data;  
  if(typeof msg === "string"){  
    console.log(msg);  
  }  
}
```

- 除了接收文本消息，还可以处理二进制数据，例如Blob消息或者ArrayBuffer消息。处理前需要判断二进制格式类型。



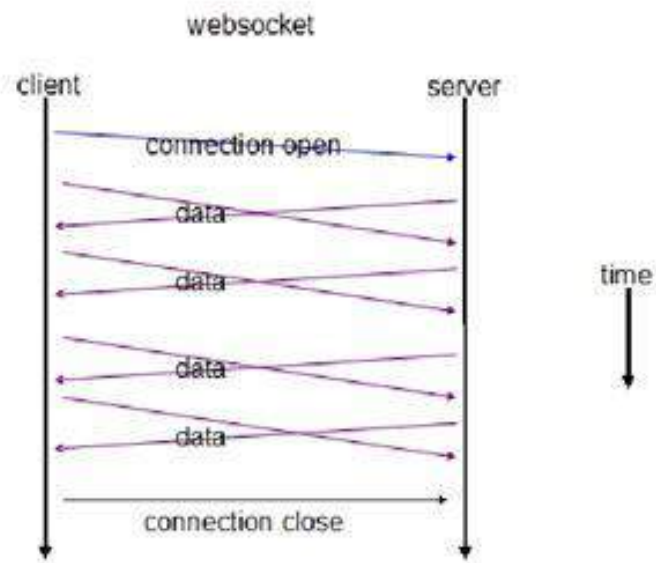
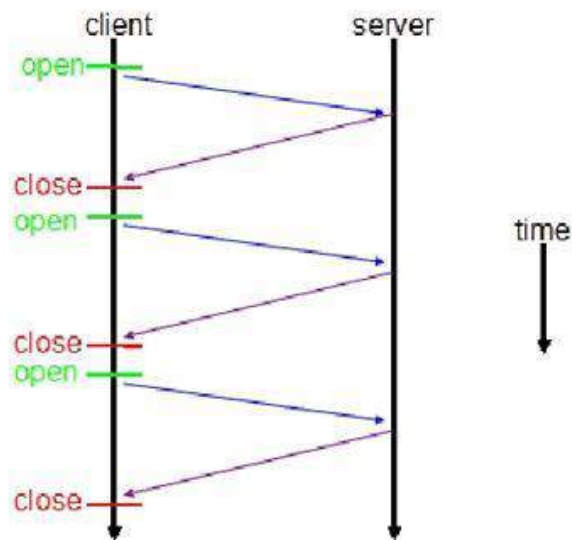
一、传统的请求-响应模式

- 1、轮询：客户端（浏览器）基于JS脚本，在设定指定的时间间隔后，不间断的向服务器发送请求，来保持客户端和服务端的数据同步。问题是，不能保证每次请求时，服务器端的数据都会有变化，随之带来是很多无谓请求，浪费带宽和效率低下。
- 2、基于 Flash：Adobe-Flash 基于自己的 Socket 实现完成数据交换，然后暴露出相应的接口为 JavaScript 调用，从而达到实时传输目的。此方式比轮询要高效。由于Flash在PC端安装率高，应用场景非常广泛。但在移动互联网终端上 Flash 的支持并不好。
 - IOS 系统不支持Flash。
 - Android 在2012年之前虽然支持Flash，但是对移动设备的硬件配置要求较高。2012 年 Adobe 官方宣布不再支持 Android4.1+系统。

- 当服务器端数据发生变化时，通知客户端数据更新。
 - 可以实现跨域访问（不受同源策略的影响）
 - 建立长连接。
- 节省带宽（这是移动端的普遍心声）

二、WebSocket 机制

- WebSocket 是 HTML5 一种新的协议。它实现了浏览器与服务器**全双工通信**，能更好的节省服务器资源和带宽并达到实时通讯，它建立在 TCP 之上，同 HTTP 一样通过 TCP 来传输数据，但是它和 HTTP 最大不同是：
 - WebSocket 是一种双向通信协议，在建立连接后，WebSocket 服务器和 Browser/Client Agent 都能主动的向对方发送或接收数据，就像 Socket 一样；
 - WebSocket 需要类似 TCP 的客户端和服务器端通过握手连接，连接成功后才能相互通信。





HTML5 简介

讲师：许井龙

微信：ngsteel（承神之佑）

- 一．什么是HTML5?
- 二．HTML5的发展历史
- 三．HTML5的前景
- 四．浏览器支持
- 五．HTML5带给我们的惊喜
- 六．课程目标
- 七．开发及测试工具

- 万维网的核心语言、标准通用标记语言下的一个应用超文本标记语言（HTML）的第五次重大修改。

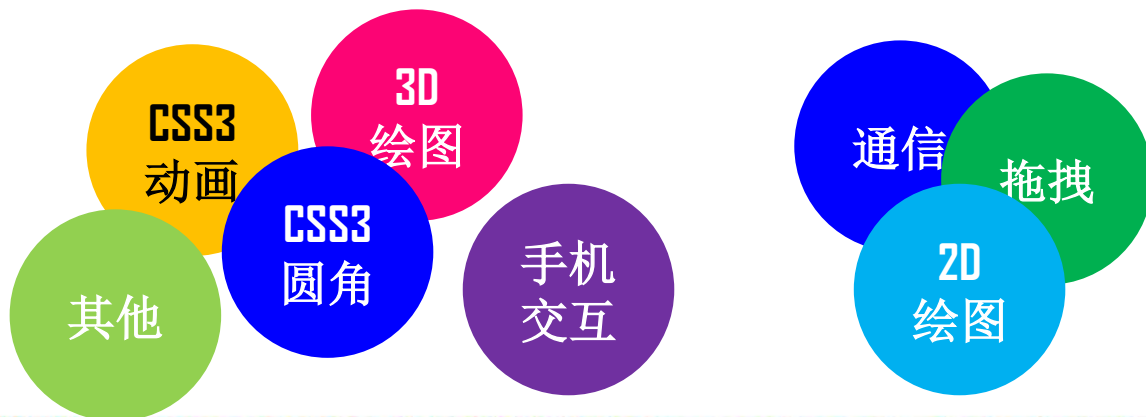
- 标准通用标记语言下的一个应用HTML标准自1999年12月发布的HTML4.01后，后继的HTML5和其它标准被束之高阁，为了推动Web标准化运动的发展，一些公司联合起来，成立了一个叫做 Web Hypertext Application Technology Working Group（Web超文本应用技术工作组 -WHATWG）的组织。WHATWG 致力于 Web 表单和应用程序，而W3C（World Wide Web Consortium，万维网联盟）专注于XHTML2.0。在2006年，双方决定进行合作，来创建一个新版本的HTML。
- HTML5草案的前身名为 Web Applications 1.0，于2004年被WHATWG提出，于2007年被W3C接纳，并成立了新的HTML工作团队。
- HTML 5 的第一份正式草案已于2008年1月22日公布。虽然HTML5仍处于完善之中。然而，大部分现代浏览器已经具备了某些HTML5支持。
- 2012年12月17日，万维网联盟（W3C）正式宣布凝结了大量网络工作者心血的HTML5规范已经正式定稿。根据W3C的发言稿称：“HTML5是开放的Web网络平台的奠基石。”
- 2013年5月6日，HTML 5.1正式草案公布。该规范定义了第五次重大版本，新功能不断推出，以帮助Web应用程序的作者，努力提高新元素互操作性。

- HTML5将会**取代**1999年制定的**HTML 4.01、XHTML 1.0**标准，以期能在互联网应用迅速发展的时候，使网络标准达到符合当代的网络需求，为桌面和移动平台带来无缝衔接的丰富内容。
- 不就将来，W3C将致力于开发用于实时通信、电子支付、应用开发等方面的标准规范，还会创建一系列的隐私、安全防护措施。

- 支持Html5的浏览器包括Firefox（火狐浏览器），IE9及其更高版本，Chrome（谷歌浏览器），Safari，Opera等；国内的遨游浏览器（Maxthon），以及基于IE或Chromium（Chrome的工程版或称实验版）所推出的360浏览器、搜狗浏览器、QQ浏览器、猎豹浏览器等国产浏览器同样具备支持HTML5的能力。

HTML5不仅仅是用来做网页的！

1. 减少对PS依赖，完全可以使用纯CSS样式，制作有规则的图形。
2. 减少对Flash的依赖，仅仅使用CSS样式，就能制作动画效果。
3. 打包成手机应用，可以与原生APP相媲美。易于调试、极大的企业降低人工成本及软件维护成本。
4. 开发桌面或手机游戏（canvas 2D，3D），而无需安装任何程序或插件。
5. 标准的Socket通信接口，轻松实现聊天室的开发。
6. 随着规范不断的更新，我们期待HTML5给我们带来更多惊喜。



1. 计划课时：

– 14天

2. 技术要点

1. 掌握基本的PS切图方法，能够基于UI设计图（.psd），设计开发PC端网页。

✓ 相关布局知识及布局技巧

✓ 高度还原设计图

2. 能够使用CSS3新特性编写网页特效

✓ CSS3 新增选择器

✓ 边框：圆角(border-radius)、阴影(box-shadow)、图片边框

✓ 过渡（transition）及动画（animation）

✓ 2D, 3D变换

✓ 背景：多背景，背景大小，背景原点，背景裁剪，渐变函数

✓ 文本处理

✓ 表单

✓ 伸缩盒模型

3. 熟悉并掌握HTML5新增API。

✓ 离线存储

✓ Web Socket

✓ Canvas

✓ 拖拽

✓ 文件读取

✓ 历史记录



- 编码工具
 - Webstorm 2016 (本次课程使用)
 - HBuilder
- 调试及调试工具
 - Chrome 54 以上版本
 - IETester
- 图形软件
 - Photoshop CS6 及其以上版本

- 晨考时间：
 - 8:40 ~ 9:00
- 阅卷要求：
 - 组长评判本组学员分数，汇总后发给班长。
 - 班长汇总分数后，第二天早上发给任课老师。

- 负责晨读的学员，今天晚上提前把单词预览一遍。
- 晨读时间：8:30 ~ 8:40

相识是缘分，龙哥爱你们！

