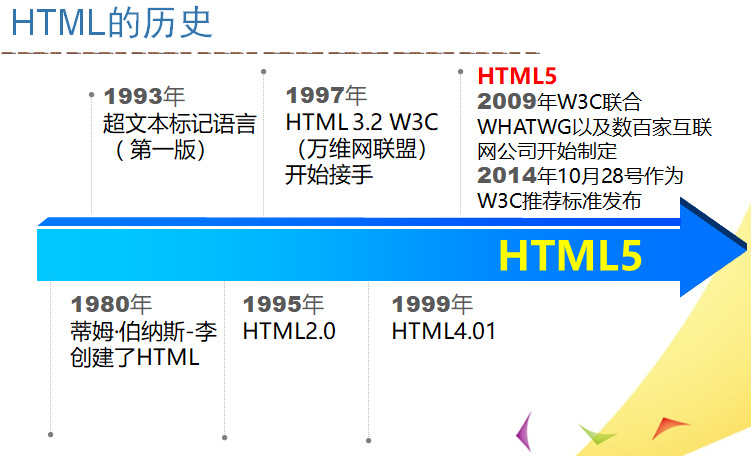
**第一讲 HTML5介绍**

1. 结构标准：HTML（超文本标记语言）

样式标准：CSS

行为标准：W3C DOM、ECMAScript

1. HTML5属于最新的一项Web标准，在原有HTML4的基础上定义了一些新的标签和 JavaScript API。是HTML4标准的一个超集。



1. 设计理念：

·避免不必要的复杂性

如：①文档声明

H4：

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"

H5：

<!DOCTYPE html>

②字符编码

H4：

<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

H5： <meta charset="utf-8" />

·支持已有的内容（可以省略一些元素的标签，当属性值不包括空字符串、<、>、=、单双引号时，可省略引号）

* 可以省略结束标签的元素：
  + li、p、option、tr、td、th、thead、tbody、tfoot
* 可以省略全部标签的元素：
  + html、head、body
* 注意：即使标签被省略了，该元素还是以隐式的方式存在的。例如将body元素的标签省略不写时，它在文档结构中仍是存在的，可以使用 document.body 进行访问。

·求真务实（用新标签代替原来的盒子标签）

<body>

<header></header>

<nav></nav>

<div id="main"></div>

<aside></aside>//侧边栏

<footer></footer>

</body>

//**新添加标签**

·优雅降级

input type="number"

input type="search"

input type="range"

input type="email"

input type="date"

input type="url"

//新添加元素

在浏览器看到自己不理解的type值时，会将type的值解释为text

1. H5优点：
   1. **提升用户体验，加强视觉感受。**

（HTML5的视音频新技术解决了移动端对flash的支持问题。

HTML5中的 CSS3特效样式、Canvas、WebGL的介入，加强了网页的视觉效果，可在网页中呈现三维立体特效 。）

Flash缺点：

·Flash只能通过插件形式在浏览器使用；Flash程序同页面其他内容不能进行任何交互。

·Flash在移动设备的执行效率非常差、不被IOS系统支持、Flash官方取消针对移动设备的更新。

* 1. **多设备、跨平台支持**

（跨平台，适配多终端

应用及时更新

简化标签，更清晰的代码）

* 1. **开放的网络标准**

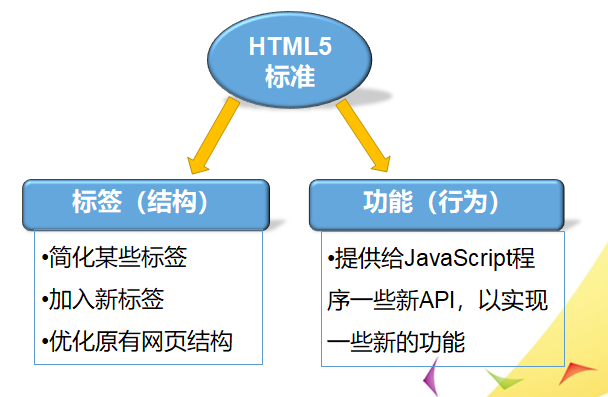
（由W3C组织，数百家互联网公司（包括谷歌、苹果等）参与而制定的标准，技术完全开放

浏览器原生支持HTML5

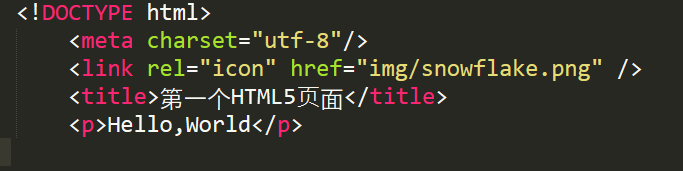
HTML5新增的标签，使搜索引擎更加容易抓取和索引网页，从而驱动网站获得更多的点击流量。）

**第二讲 HTML5标签及特性**

1. 标准

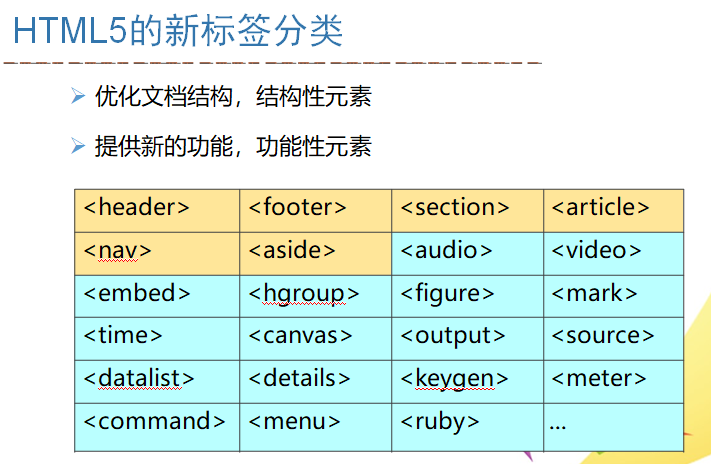


1. 第一个HTML5页面



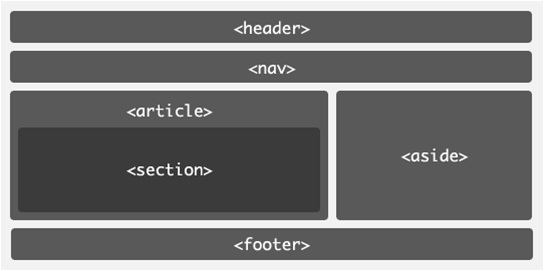
省略了html、head、body标签

3. 新标签分类



1. 新增结构元素（HTML5定义了一组新的语义化标签来描述元素的内容。）

|  |  |
| --- | --- |
| **标签** | **作用** |
| <header> | 定义**头部**，通常包含导航和一些引导信息 |
| <nav> | 定义**导航**内容 |
| <article> | 定义**正文**内容 |
| <section> | 定义文章中的**段** |
| <aside> | 定义**边栏**内容 |
| <footer> | 定义**页脚** |



1. header元素

·header元素是一种具有**引导和导航作用**的结构元素，通常用来放置整个页面或页面内一个区块的**标题、搜索表单、logo**等。

·一个网页内不限制只能有一个header元素，可以拥有**多个**，可以为每个内容区域块添加header元素。

<header>

<h1>页面标题</h1>

</header>

1. nav元素

nav元素定义**页面导航的链接组**，其中的导航元素**链接到其他页面或当前页面的其他部分**。

<nav>

<ul>

<li><a href="/">主页</a></li>

<li><a href="/events">开发文档</a></li>

</ul>

</nav>

* + 网页可以拥有多个nav元素，作为页面整体或不同部分的导航。
  + 并不是所有的链接都要放进nav元素，如果文档中有“前后”按钮，则应该把它放到 <nav> 元素中。**？**

1. article元素

article元素定义可以独立被外部引用的内容。可以是一篇博客、一段评论…

<article>

<header>

<h1>苹果</h1>

</header>

<p>苹果，植物类水果，多次花果... </p>

<footer>

<p>著作权归\*\*\*公司所有。</p>

</footer>

</article>

注：一个article元素通常有自己的标题和脚注。

1. section元素

section元素定义文档中的区域。比如章节、页眉、页脚或文档中的其他部分。

<section>

<h1>PRC</h1>

<p>The People's Republic of China was born in 1949...</p>

</section>

1. aside元素 (表示当前页面或文章的附属信息。)

① 包含在article元素**中**，作为**主要内容的附属信息部分**，内容可以是与当前文章有关的相关资料、名词解释等。

② 在article元素之**外**，作为**页面或站点全局的附属信息**。最典型的是**侧边栏**，内容包括友情链接，博客中的文章列表、广告等。

10. footer 元素

* footer元素定义页脚。通常包括其相关区块的脚注信息，如作者、文档的创作日期以及版权信息等。
* 一个页面中可以多个footer元素。可以为article元素或section元素添加footer。

1. 实例



1. 新增的功能

* HTML5中加入了对音频和视频的**原生支持**

优点：

·无需插件

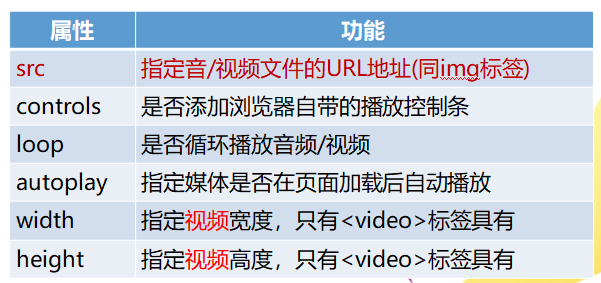
·提供JS脚本控制的API

缺点：

·音视频格式的支持有限

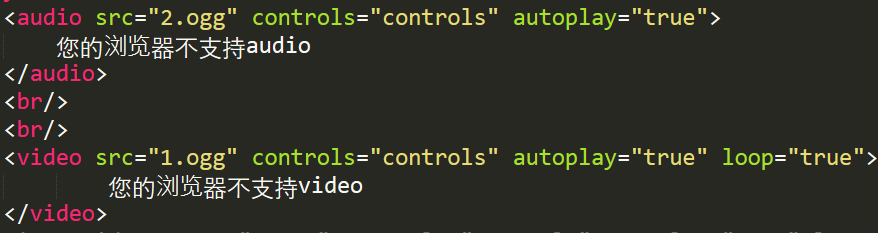
* <video>播放视频
* <audio>播放音频

·属性：



实例：

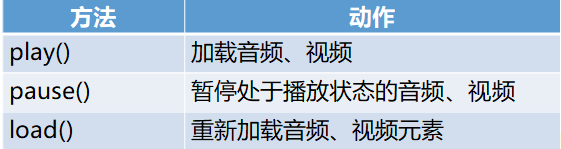
（1）只有不支持才显示标签的内容 **？**



1. 音/视频格式兼容性解决方案: audio/video 组合source 元素。



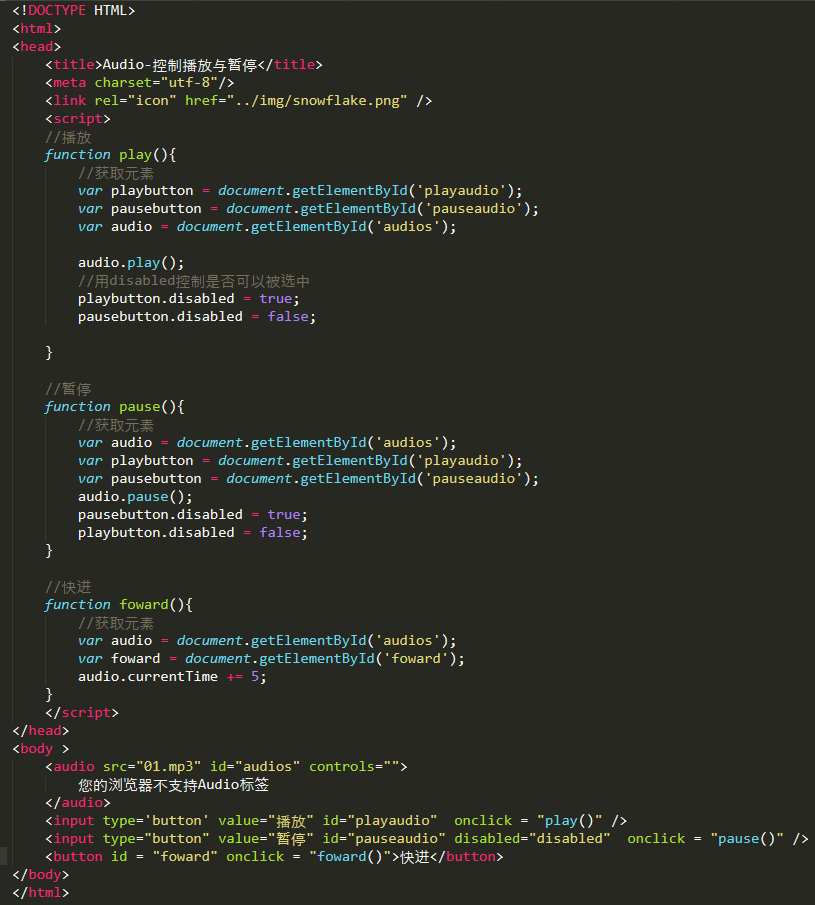
1. 更灵活的控制音/视频







※实例：通过 play( )、pause( )、 currentTime 实现对音频的播放、暂停、快进操作。



1. <embed>
2. <time>

①定义公历的时间

②为机器提供便利

<p>我们在每天早上 <time>8:00</time> 开始上课</p>

<p>我在 <time datetime=“2018-10-1”>国庆节</time> 有个约会</p>

1. <details>与<summary>配合使用可以为details定义标题

<details>

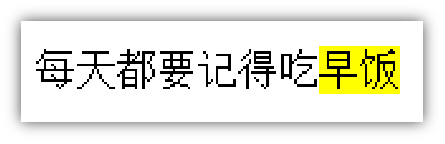
<summary>Copyright 2018.</summary>

<p>版权归H5方向所有.</p>

</details>

1. <mark> 标签定义带有记号的文本。

<p> 每天都要记得吃<mark>早饭</mark> </p>



1. <progress> 标签标示任务的进度（进程）。

|  |  |  |
| --- | --- | --- |
| **属性** | **值** | **描述** |
| max | number | 规定任务一共需要多少工作。 |
| value | number | 规定已经完成多少任务。 |

下载进度：

    <!-- value定义已完成进度 -->

    <progress value="30" max="100">

    </progress>

2

1. 废除的元素

·能使用CSS替代的元素（表线性元素）：big, center, font, s, u等。

HTML5中提倡把呈现性的功能放在CSS样式表中统一编辑。

·不再使用frame框架。

删除frameset、frame、noframes标签，只支持iframe框架。

·只有部分浏览器支持的元素：blink, bgsound等。

**第三讲 表单**

1. Input输入类型
2. 数值输入文本框(name必须写)

    <!-- action写路径，表单提交的目的地址 -->

    <!-- method表示提交时的方式 -->

    <form action="a/index.php" method="GET">

        <!-- 设置数值类型的输入框 -->

        <input type="number" name="num1" min="5" max="100" step="5"/>

        <input type="submit" name="s" action="#"/>

    </form>

图片2

注：外观与text文本框相同，但不能输入数值以外的文字，否则提交时将内容作为空白进行提交。

name : 用来标识表单提交时的key值

min : 当前输入框输入的最小值

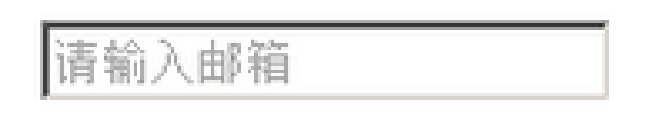
max : 最大值

step：步长，点击增大或者减小时增加减少的步长

② 邮箱输入文本框

<!-- 当表单提交时，会自动校验是否符合邮箱的正则表达式，但不检验该email地址是否存在。 -->

        <input type="email" name="email" placeholder="请输入邮箱" />



③ url输入文本框

<!-- 专门用来输入url地址。当表单在提交前，会自动校验是否符合url网址的规范。 -->

        <input type="url" name="url" placeholder= "输入正确的网址" />

图片4

④ 电话号码输入文本框

<!-- 注：手机中的浏览器遇到tel类型的input元素时，会自动变换触摸屏幕键盘以方便用户输入。 -->

        <input type="tel" placeholder="输入电话" name="p"/>

图片5

⑤ 滑动条输入文本框

        <!-- 用于应该包含一定范围内数字值的输入域。range 类型显示为滑动条。能够设定对所接受的数字的限定。 -->

        <!-- step为移动步长 -->

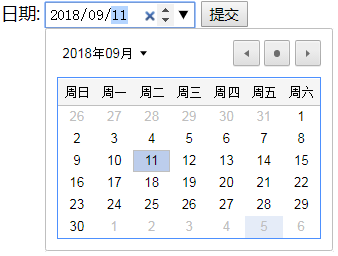
        <input type="range" min="0" max="50" step="8" name="rdemo" value="0" />

图片6

⑥ 日期时间输入文本框

<!-- date类型的input元素以日历的形式方便用户输入。还有其他的type：time、datetime-local、month 、week、datetime。 -->

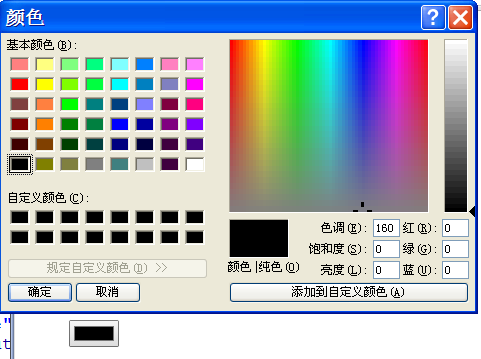
        日期：<input type="date" name="日期">



⑦ 颜色选择文本框

<!-- color类型的input元素用来选取颜色，其提供了一个颜色选择器。 -->

        颜色: <input type="color" name="date1"/>



⑧搜索功能文本框（模糊搜索的应用）

<!-- search类型用于搜索域，比如站点搜索或 Google 搜索。 -->

        <input type="search" name="ser1"/>

1. 新增表单元素

① datalist 元素（为输入框提供可选的列表）

    <form action="" method="get">

        金庸武侠小说中的武功及招式:

        <input type="search" list="names" name="sea1">

        <datalist id = "names">

            <option></option>

            <option>亢龙有悔</option>

            <option>飞龙在天</option>

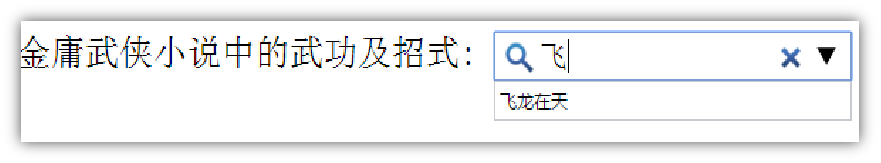
            <option>神龙摆尾</option>

            <option>弹指神通</option>

            <option>九阴真经</option>

        </datalist>

    </form>



1. output元素（用于在浏览器汇总显示计算结果或脚本输 出）

<!-- form元素中设置计算形式 -->

    <form oninput="x.value=parseInt(a.value)+parseInt(b.value)">

        <!-- value为初始值，range为滑杆。 -->

        0<input type="range" id="a" value="50">100

        +<input type="number" id="b" value="50">

        =<output name="x" for="a b"></output>

    </form>

1. form属性（form 属性规定输入域所属的一个或多个表单，属性值为所属表单的 id。）

<form id="testform" method="get" action="">

        姓&nbsp;&nbsp;名：<input type="text" name="name1" autofocus="autofocus" />

        <input type="submit" value="提交"/>

    </form>

    <br/>

    自我介绍：<textarea name="textarea1" form="testform"></textarea>

注：可将表单内的从属元素写在页面的任意位置，然后为该元素指定form属性。如需引用多个表单，使用空格分隔的列表。

④ formaction、formmethod属性

※在HTML5中使用formaction属性实现将表单提交到不同的页面

※使用formmethod属性对每个表单元素分别指定不同的提交方法。

    <form id="testform" method="get" action="serve.jsp">

        <input type="submit" name="s1" value="v1" formaction="s1.jsp">提交到S1

        <input type="submit" name="s2" value="v2" formmethod="post">提交到S2

        <input type="submit" name="s3" value="v3" formmethod="get">提交到S3

    </form>

⑤ required、placeholder属性

required 属性

→布尔属性

→规定输入域在提交之间必须填写

placeholder 属性

→没有值时出现在文本框中的字符串

→获取焦点输入时框中提示信息消失

→以柔和的灰色文本方式显示在域中

⑥ list属性

input 元素的 list 属性与 datalist 元素的 id 属性绑定。

⑦ autofocus 属性

布尔属性，页面加载时表单域自动聚焦

一个页面此属性仅可设置一次

⑧ multiple 属性

用于文件上传控件，设置此属性后，允许上传多个文件。

   <input type="file" multiple="">

⑨ max/min/step 属性

设置最大值/ 最小值/ 数值或日期时间的增量。

1. **地理位置定位**
2. 位置信息来源的分类和特点
3. Geolocation 能获得的具体数据（根据设备情况而定）

latitude 纬度（十进制），例如：38.0441

longitude 经度（十进制），例如：114.51

accuracy 精确度，以米为单位

altitude 海拔，海平面以上以米计

heading 行驶方向，从正北开始以度计

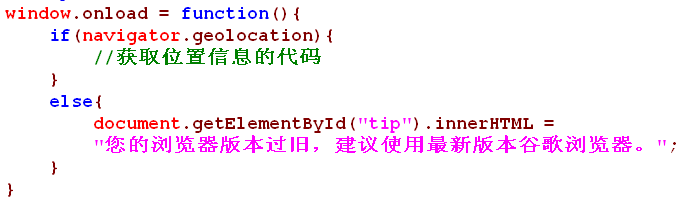
speed 速度，以米/每秒计

（仅在移动端使用）

timestamp 获取位置的时间

…

1. 检测浏览器兼容性（通过判断 geolocation 对象是否存在判断浏览器是否支持。）



1. Geolocation API 存在于navigator对象中，包含 3 个方法：

① getCurrentPosition() //获取当前地理位置

② watchPosition()

//（在一个特定的时间段）持续监视当前地理位置

③clearWatch() //清除监视

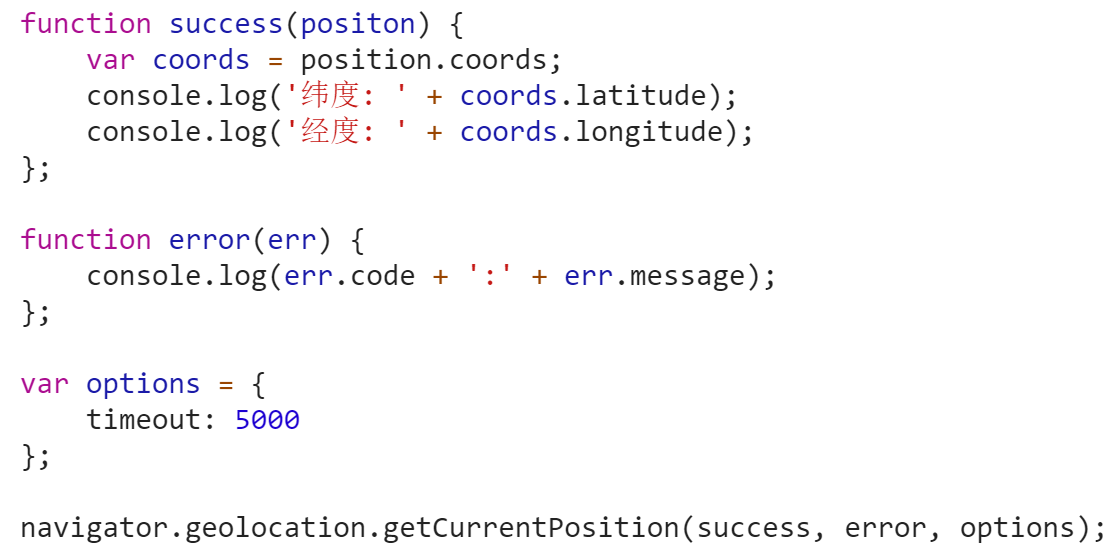
1. 单次获取位置数据

getCurrentPosition(onSuccess, onError, options)

参数1：获取数据成功后执行的回调函数，使用 Position 对象作为唯一的参数

参数2（可选）：获取数据失败时执行的回调函数，使用 PositionError 对象作为唯一的参数

参数3 （可选）：可选参数列表，可通过该对象参数设定最长可接受的定位返回时间（不宜太短，否则结果会直接返回定位失败）、等待请求的时间和是否获取高精度定位



1. **成功回调函数：**定位信息成功时返回一个 positon 对象给成功回调函数，此对象包含属性:

|  |  |
| --- | --- |
| **属性** | **描述** |
| coords.latitude | 十进制数的纬度 |
| coords.longitude | 十进制数的经度 |
| coords.accuracy | 位置精度 |
| coords.altitude | 海拔，海平面以上以米计 |
| coords.altitudeAccuracy | 位置的海拔精度 |
| coords.heading | 方向，从正北开始以度计 |
| coords.speed | 速度，以米/每秒计 |
| timestamp | 响应的日期/时间 |

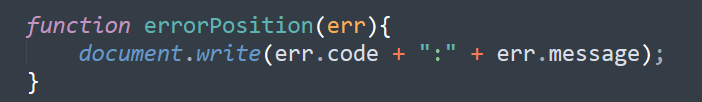
**用法举例：position.coords.accuracy**

1. **失败回调函数**

定位信息失败时返回一个 error 对象给失败回调函数。

属性:

|  |  |
| --- | --- |
| **属性** | **描述** |
| code | 错误编码：  UNKNOWN\_ERROR（错误编号0）- 未知错误  PERMISSION\_DENIED（错误编号1） - 用户不允许地理定位  POSITION\_UNAVAILABLE （错误编号2） - 无法获取当前位置  TIMEOUT （错误编号3） - 操作超时 |
| message | 错误信息 |



1. options可选参数列表

enableHighAcuracy — 布尔值，表示是否启用高精确度模式(GPS)启用此模式，浏览器在获取位置信息时需耗费更多的时间。

timeout — 整数，表示等待响应的最大时间，否则触发 errorCallback，默认为 0 毫秒，表示无穷时间。

maximumAge — 整数/常量，表示是否使用最近缓存的位置数据。默认为 0 毫秒，表示必须在每次请求时查找一个新位置。



1. 持续性获取位置数据方法

watchPosition(onSuccess, onError, options)

参数同 getCurrentPosition 相同

监测用户位置，位置发生改变时即调用成功回调函数

清除监控：该方法会返回一个 ID，如要取消监听可以通过 clearWatch(watchId) 传入该 ID 实现取消的目的。

只针对移动设备

1. **离线应用**
2. 创建离线应用
3. 创建缓存清单文件（manifest 文件）
4. 在 html 标记中指定使用缓存文件

<html manifest="cacheData.appcache">

1. Manifest文件（三部分）

（1）CACHE MANIFEST （没网也可以获取）

在此标题下列出的文件将在首次下载后**进行缓存**，写在第一行，必须有该部分

（2）NETWORK（必须有网）

在此标题下列出的文件**不会被缓存**，需要与服务器的连接，离线不可用

（3）FALLBACK（备用资源）

提供了获取不到缓存资源时的**备选资源路径**（比如 404 页面）

4.Manifest文件控制缓存

CACHE MANIFEST

#2019-3-15 v1.0.2//控制版本更新

style.css

NETWORK:

app.js

FALLBACK:

login.html info.html

5.离线控制API

（1）提供对象

（2）兼容性

1. 状态

|  |  |
| --- | --- |
| **Status值** | **缓存状态** |
| 0 | UNCACHED（未缓存） |
| 1 | IDLE（空闲状态） |
| 2 | CHECKING（检查中） |
| 3 | DOWNLOADING（下载中） |
| 4 | UPDATEREADY（更新就绪） |
| 5 | OBSOLETE（过期） |

1. 事件
2. 离线事件监听
3. 更新缓存方法：
4. **Web存储**
5. HTML5 Web 存储的优点：

存储空间更大，浏览器至少提供 5M 的空间

数据仅仅存储在本地，不会自动向服务器发送

提供丰富的接口，方便的操作数据

独立的存储空间，每个域都有各自存储空间，不会造成混乱

1. Web Storage API
2. 检测浏览器是否支持
3. 存储数据项

localStorage.setItem("key", value)

localStorage["key"] = value

localStorage.key = value

1. 读取数据项

① localStorage.getItem(“key”)

② localStorage["key"]

③ localStorage.key

window.onload = function(){

            //如果不存在localStorage

            if(!window.localStorage){

                alert("您的浏览器不支持localStorage");

            }

            //支持,开始读写操作

            else{

                localStorage.setItem("key0", 666);//存数据项最常用方法

                localStorage["key1"] = 777;//存数据项方法二

                localStorage.key2 = "第三个值";///存数据项方法三

                alert(localStorage.getItem("key0"));//666 读取数据项方法一

                alert(typeof(localStorage.getItem("key0")));//String

                alert(localStorage["key1"]);//读取数据项方法二

                alert(localStorage.key2);//读取数据项方法三

            }

        }

1. 删除数据项

localStorage.removeItem(key)

1. 清除全部数据

localStorage.clear( )

（6）获得键名

localStorage.key(index)

（7）获得数据项的个数

localStorage.length

1. **数据通信**
2. Web应用的数据通信方式

HTTP协议

一种无状态的、无连接的、单向的应用层协议。

采用了请求/响应模型，Request = Response 。

通信请求只能由客户端发起，服务端对请求做出应答处理。

Ajax

由JavaScript发起请求，局部更新页面/应用。

缺陷：通信只能由客户端发起服务器总是被动的。

1. ajax轮询，即让浏览器隔几秒就发送一次请求，询问服务器是否有新信息。
2. 长轮询，采取的是阻塞模型，即客户端发起连接后，如果没消息，就一直不返回Response给客户端，直到有消息才返回。返回完之后，客户端再次建立连接，周而复始。
3. WebSocket是HTML5提供的一种浏览器与服务器间进行**全双工通讯的网络通讯协议**。
4. WebSocket特点：

（1）建立在 TCP 协议之上，服务器端的实现比较容易。

（2）与 HTTP 协议有着良好的兼容性。默认端口也是80和443，握手阶段采用 HTTP 协议，不容易屏蔽，能通过各种 HTTP 代理服务器。

（3）数据格式比较轻量，性能开销小，通信高效。

（4）可以发送文本，也可以发送二进制数据。

（5）没有同源限制，客户端可以与任意服务器通信。

（6）协议标识符是ws（如果加密，则为wss），服务器网址就是 URL。

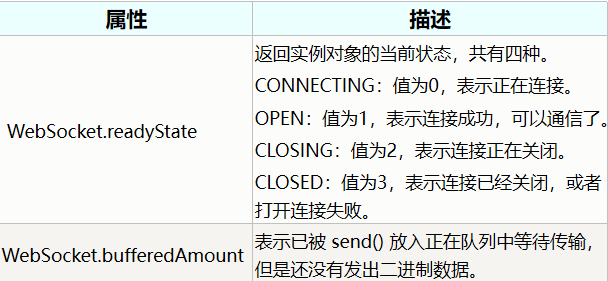
1. 使用WebSocket
2. WebSocket构造函数

实例化：

var Socket = new WebSocket(url, [protocol] );

注：第一个参数 url, 指定连接的 URL。第二个参数 protocol 是可选的，指定了可接受的子协议。

属性:



1. 实例对象的 bufferedAmount 属性可以用来判断发送是否结束。

var socket = new WebSocket(url, [protocol] );

var data = new ArrayBuffer(10000000);//二进制

socket.send(data);

if (socket.bufferedAmount === 0) {

// 发送完毕

}

else {

// 发送还没结束

}

2. 的的的
3. **文件**
4. file与filelist

File对象指用户选择的文件

Filelist对象指用户选择的文件列表

1. File对象的属性（file对象继承了Blob对象）
2. Blob对象的属性

slice( ) 可以访问到字节内部的原始数据块。

size属性 表示一个Blob对象的字节长度

type属性 表示Blob的MIME类型，如果是未知类型，则返回一个空字符串。

※通过类型过滤文件（type属性）

image/开头的，后面紧跟图像的类型。

audio/\* 表示音频文件 HTML5

video/\* 表示视频文件 HTML5

image/\* 表示图片文件

1. FileReader（把文件读入内存，并且读取文件中的数据。）
2. 检测兼容性

// if ( typeof FileReader == 'undefined' )

if ( ! **window.FileReader** )

{

alert( " 您的浏览器未实现 FileReader 接口 " );

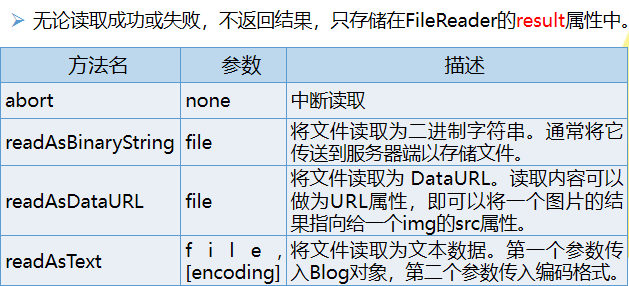
}

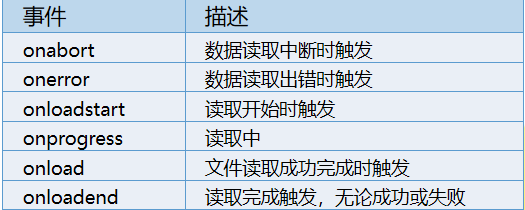
else

{

var reader = **new FileReader();**

}





（2）实现文本文件的读取**8-6**

使用 FileReader 对象的 readAsText()方法实现文本文件的预览。

需要注意：txt文件的编码格式需要设置为**UTF-8**。

**第九讲**

1. DataTransfer对象
2. 源对象与目标对象
3. 方法

①设置数据（向DataTransfer对象存入数据）

**setData(format, data)**

Format:格式

Data：数据

②获取数据（从DataTransfer对象读数据）

**getData(format)**

注：现在支持拖动处理的MIME的类型有：“text/plain（文本文字）”、“text/html”、“text/xml”、“text/uri-list（URL列表，每个URL为一行）

1. 属性

①files ：拖拽文件列表

file.length可以表示文件长度

②dropEffect ：表示拖放操作的视觉效果，即鼠标形状。

③types

④effectAllowed

1. 步骤

（1）将需要拖放的对象元素的draggable属性设为true ，即 draggable="true"

注： img元素与a元素默认允许拖放。

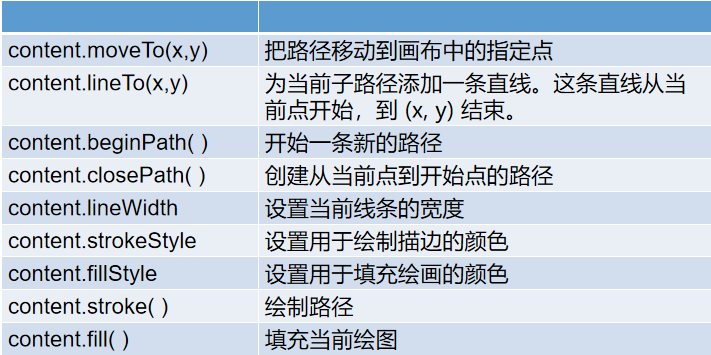
1. 设置拖动什么 ondragstart 和 setData()
2. 放置元素到何处 ondragover
3. 进行放置 ondrop 和 getData()

3.拖放相关事件※



**第十讲**

1. 方法



1. 路径封闭（canvas 之中只能有一条路径存在，称之为“当前路径” (current path) 。）

（1）beginPath( )

开始一条新的路径（路径开始点），写在moveTo之前。

（2）closePath( ) //自动变成密闭图形

创建从当前点到开始点的路径（路径结束点）。

关闭一条打开的子路径。写在stroke之前。

写在fill之前或之后填充效果相同，因为fill也会自动密闭。

1. 绘制弧和圆

<!DOCTYPE html>

<html>

<head>

<title></title>

<meta charset="utf-8">

<style>

canvas{

border: 1px solid rgb(155, 31, 31);

background-color: rgb(240, 203, 155);

}

</style>

</head>

<body>

<canvas id="Canvas1" width="500" height="500">

您的浏览器不支持canvas。

</canvas>

<canvas id="Canvas2" width="800" height="800">

您的浏览器不支持canvas。

</canvas>

</body>

<script>

/\*

//画三角形

var canvas = document.getElementById("Canvas1");

var context = canvas.getContext("2d");

context.beginPath();

context.moveTo(0,0);//设置起点

context.lineTo(0,300);//设置终点

context.lineTo(300,300);

context.lineTo(0,0);

context.lineWidth = 5;//线条宽度

context.strokeStyle = "red";//线条颜色

//实心填充，必须先写fillstyle后写fill，否则会只填充黑色，fillstyle设置的颜色无效。

context.fillStyle="blue";

context.fill();

context.closePath();

context.stroke();//绘制

\*/

/\*

//绘制一段圆弧

var canvas = document.getElementById("Canvas1");

var context = canvas.getContext("2d");

//context.beginPath();

context.arc(300,300,200,0.5\*Math.PI,1.8\*Math.PI,true);

//圆心的横坐标，纵坐标，半径,起始角，结束角是否按照逆时针方向绘图。

context.lineWidth = 5;

context.strokeStyle = "#005588";

//context.closePath(); //闭合圆弧

context.stroke();

\*/

/\*

//同心圆

var canvas = document.getElementById("Canvas1");

var context = canvas.getContext("2d");

for(var i = 0;i <= 8;i++){

context.strokeStyle = "#32CD32";

context.beginPath();

context.arc(0,400,i\*50,0,1.5\*Math.PI,true);

context.stroke();

}

\*/

/\*

//太极图

var canvas = document.getElementById("Canvas2");

var con = canvas.getContext("2d");

//绘制白色半圆

con.beginPath();

con.arc(200,200,80,1.5\*Math.PI,0.5\*Math.PI);

con.fillStyle = "white";

con.closePath();

con.fill();

//绘制黑色半圆

con.beginPath();

con.arc(200,200,80,1.5\*Math.PI,0.5\*Math.PI,true);

con.fillStyle = "black";

con.closePath();

con.fill();

//绘制白色小圆

con.beginPath();

con.arc(200,160,40,1.5\*Math.PI,0.5\*Math.PI,true);

con.fillStyle = "white";

con.closePath();

con.fill();

//绘制黑色小圆

con.beginPath();

con.arc(200,240,40,1.5\*Math.PI,0.5\*Math.PI);

con.fillStyle = "black";

con.closePath();

con.fill();

//绘制黑色小点

con.beginPath();

con.arc(200,160,3,2\*Math.PI,0,true);

con.fillStyle = "black";

con.closePath();

con.fill();

//绘制白色小点

con.beginPath();

con.arc(200,240,3,2\*Math.PI,0);

con.fillStyle = "white";

con.closePath();

con.fill();

// con.stroke();

\*/

/\*

//绘制矩形1

var canvas = document.getElementById("Canvas2");

var con = canvas.getContext("2d");

con.rect(100,100,400,400);//空心

//矩形左上角的横纵坐标，宽高。

con.lineWidth = 5;

con.fill();

con.stroke();

\*/

//绘制矩形2

var canvas = document.getElementById("Canvas2");

var con = canvas.getContext("2d");

con.fillRect(100,200,800,400);//填充一个实心矩形

con.clearRect(200,400,100,100);//清空给定矩形内的指定像素

</script>

</html>

第二十二讲 Bootstrap 全局CSS（一）

1. 文字排版
2. <h1>到<h6>均可用，在此基础上提供了.h1到.h6 class

<h2>h2. Bootstrap heading</h2>

<p class="h2">我是一个段落</p> /\*这两行效果一样\*/

1. 标题内可以添加副标题

<h1>h1. Bootstrap heading<small>副标题</small></h1>



1. Bootstrap将全局font-size设置为14px，line-height为1.428。

这些属性直接赋给<body>和所有段落元素。

<p>元素被设置了等于1/2行高的底部外边距（即10px）。

1. 通过添加.lead可以让段落更强调的突出显示。

          <p class="lead">

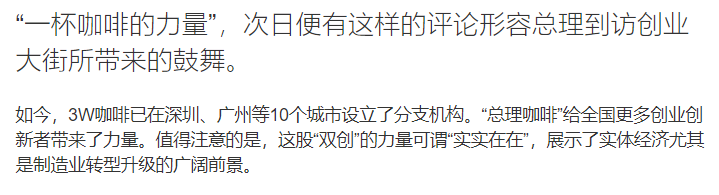
            “一杯咖啡的力量”，次日便有这样的评论形容总理到访创业大街所带来的鼓舞。

          </p>

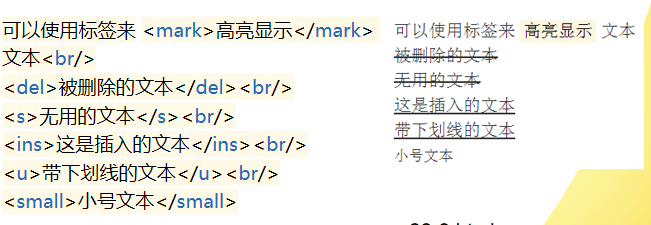
          <p>

            如今，3W咖啡已在深圳、广州等10个城市设立了分支机构。“总理咖啡”给全国更多创业创新者带来了力量。值得注意的是，这股“双创”的力量可谓“实实在在”，展示了实体经济尤其是制造业转型升级的广阔前景。

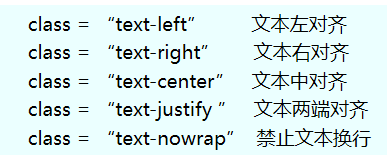
          </p>



1. 内联文本元素



1. 文本对齐



1. 改变大小写



1. 缩写：文本底部的虚线框，鼠标移至上面时会变成带有“问号”的指针。Title内容为鼠标悬停上面显示的内容。

<abbr title="World Wide Web">WWW</abbr>万维网



1. 文本颜色

**.text-muted：提示，使用浅灰色（#999）**

**.text-primary：主要，使用蓝色（#428bca）**

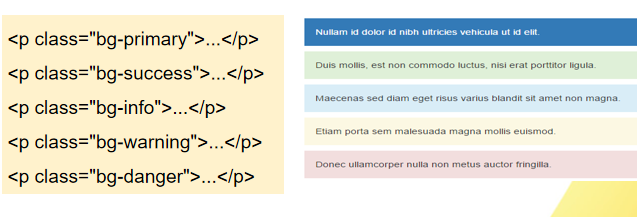
**.text-success：成功，使用浅绿色(#3c763d)**

**.text-info：通知信息，使用浅蓝色（#31708f）**

**.text-warning：警告，使用黄色（#8a6d3b）**

**.text-danger：危险，使用褐色（#a94442）**

1. 文本背景颜色



1. 列表

（1）Bootstrap 支持有序列表、无序列表和定义列表。

有序列表ol：有序列表是指以数字或其他有序字符开头的列表。

无序列表ul：无序列表是指没有特定顺序的列表。

如果不想显示列表项符号，使用 class .list-unstyled 来移除样式。

<ul class = " list-unstyled " >……</ul>



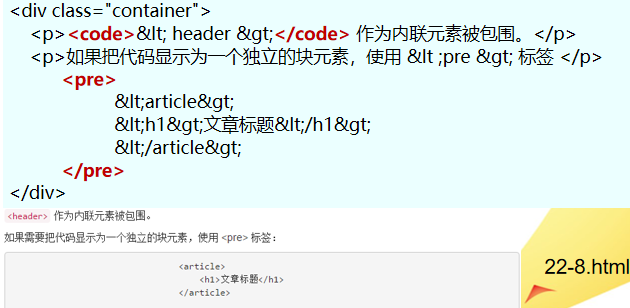
1. 代码

Bootstrap 允许以两种方式显示代码：

·<code> 标签。内联显示代码。

·<pre> 标签。代码需要被显示为一个独立的块元素或者当代码有多行时，应该使用 <pre> 标签。

·使用 <pre> 和 <code> 标签时，确保代码中的开始和结束标签使用字符实体： &lt; 和 &gt;



1. 图片

1.响应式图片.img-responsive

2.图片形状

圆角图片：<img src="..." class="img-rounded">

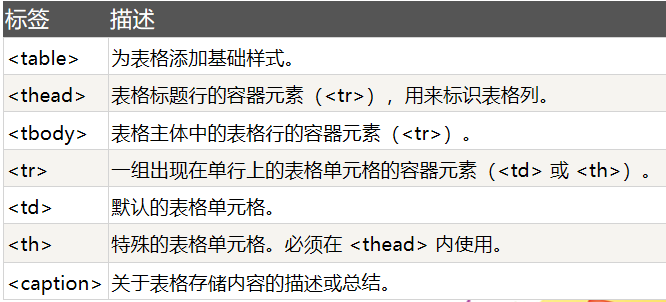
圆形图片：<img src="..." class="img-circle">

边框圆角：<img src="..." class="img-thumbnail">

注意：Internet Explorer 8 不支持 CSS3 中的圆角属性。

第二十三讲 Bootstrap 全局CSS（二）

1. Bootstrap表格



1.为任意<table>标签添加.table可以为其赋予基本的样式：少量的内补（padding）和水平方向的分隔线。

2..table-striped 给表格内的每一行增加斑马条纹样式。

3.table-condensed 使表格更加紧凑，单元格中的内边距均会减半。

<table class="table table-striped table-hover table-condensed">

                <thead>

                    <tr>

                        <th>#</th>

                        <th>姓名</th>

                    </tr>

                </thead>

                <tbody>

                <tr>

                    <td>1</td>

                    <td>张三</td>

                </tr>

                <tr>

                    <td>2</td>

                    <td>李四</td>

                </tr>

                <tr>

                    <td>3</td>

                    <td>王五</td>

                </tr>

                <tr>

                    <td>4</td>

                    <td>赵六</td>

                </tr>

                <tr>

                    <td>5</td>

                    <td>田七</td>

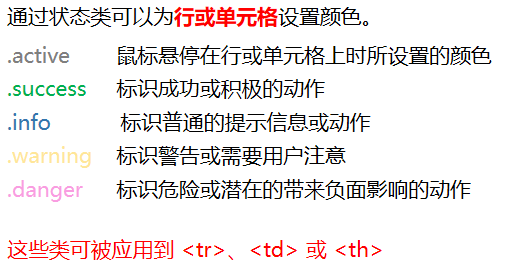
                </tr>

            </tbody>

            </table>



1. table-border给每个单元格加边框
2. table-hover鼠标悬停浅灰背景
3. 状态类



1. 响应式表格※

**<div class = "table-responsive">**

**<table class="table"> ... </table>**

**</div>**

**在小屏幕设备上（小于768px）水平滚动。当屏幕大于 768px 宽度时，水平滚动条消失。**

**垂直方向的内容截断：响应式表格使用了 overflow-y: hidden 属性，将超出表格底部和顶部的内容截断。**

1. 表单
2. 支持原生H5的input类型，包括：text、password、datetime、datetime-local、date、month、time、week、number、email、url、search、tel 和 color。
3. 创建方法：

· 向父 <form> 元素添加 role="form"。

· 把标签和控件放在一个带有 .form-group 类的 <div> 中。以获取最佳间距。

· 向文本元素 <input>、<textarea> 和 <select> 添加 class ="form-control" 。

<form>

                <div class="form-group">

                    <label for="InputEmail1">邮箱地址</label>

                    <input class="form-control" type="text" id="InputEmail1" width="100px" value="123@163.com">

                </div>

</form>

1. form-control 的元素：

1、宽度变成了100%

2、设置了一个浅灰色（#ccc）的边框

3、具有4px的圆角

4、设置阴影效果，并且元素得到焦点之时，阴影和边框效果会有所变化

5、设置了placeholder的颜色为#999

4. 为<form>元素添加.form-inline 可使其**内容左对齐**并表现为内联控件。（缩成一行）

注：

（1）需要设置宽度

在Bootstrap中，input、select和textarea默认被设置为100%宽度。为了使用内联表单，需要为表单控件设置宽度。

（2）需要设置label

如果没有为每个输入控件设置label，屏幕阅读器将无法正确识读。可以通过为label设置.sr-only将其隐藏。

1. 通过为表单添加 .form-horizontal 类，并联合使用 Bootstrap 栅格类，可以将 label 标签和控件组水平并排布局。（一行一行展开）



1. 添加相应的 .has-feedback 类即可针对校验状态为输入框添加额外的图标。

             <div class="form-group has-success has-feedback">

             <label class="control-label" for="inputSuccess2">输入成功</label>

             <input type="text" class="form-control" id="inputSuccess2" >

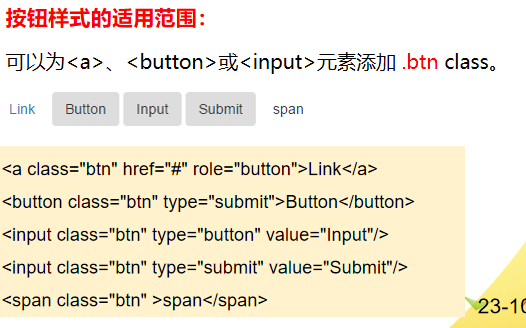
                <span class="glyphicon glyphicon-heart form-control-feedback" ></span>

                <!-- form-control-feedback的作用是定位，不然会另起一行 -->

            </div>



1. 按钮
2. 多种方式添加按钮



1. 按钮样式



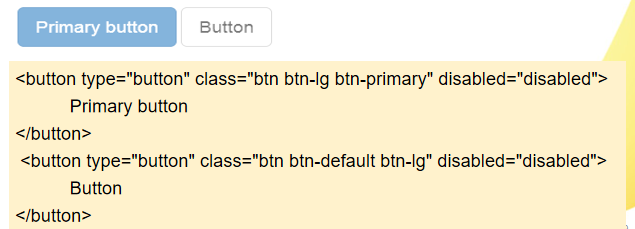
1. 尺寸

①btn-lg

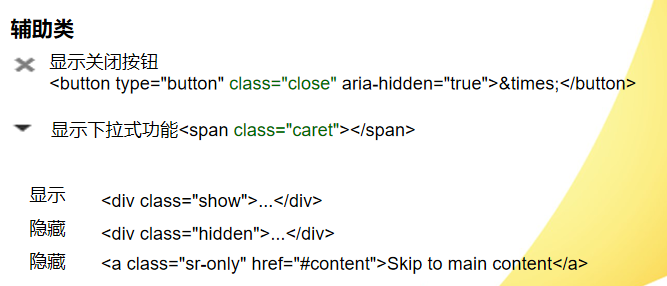
②btn-sm

③btn-xs

1. 禁用



1. 其它按钮



第二十四讲 Bootstrap组件

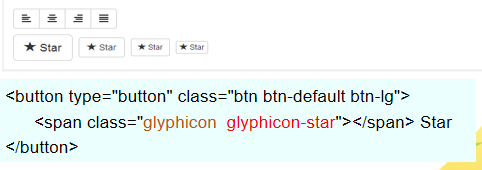
1. 图标（.glyphicon）

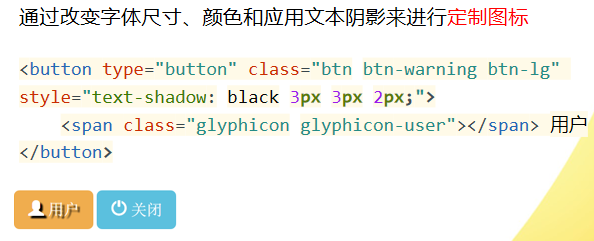
1、引用基类 .glyphicon

2、引用图标的对应类 glyphicon-\*

3、不能和其他组件联合使用，需要外部嵌套 span 标签

4、图标是响应式的





1. 按钮组（.btn-group）
2. 默认横向堆叠，（.btn-group-vertical）可以实现垂直堆叠。

         <div class="btn-group-vertical btn-group-lg">

         <button type="button" class="btn btn-default" >

             Left

         </button>

             <button type="button" class="btn btn-info" >

             Middle

         </button>

                <button type="button" class="btn btn-warning" >

             Right

         </button>

         <button type="button" class="btn btn-danger" >

             Right

         </button>

         </div>

1. 导航(响应式、页面缩小后呈现汉堡按钮)

创建一个默认的导航栏的步骤如下：

·向 <nav> 标签添加 class .navbar、.navbar-default。

·添加 role="navigation"，有助于增加可访问性。

·向 <div> 元素添加一个标题类 .navbar-header，内部包含了带有 class navbar-brand 的 <a> 元素。使文本看起来更大一号。

·添加带有 class .nav、.navbar-nav 的无序列表为导航栏添加链接。

    <div class="container">

        <p></p>

        <nav class="navbar navbar-inverse" >

            <div class="container-fluid">

                <div class="navbar-header">

                <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#bs-1">/\*data-target指示要切换到哪一个元素,datatoggle告诉js对按钮的操作\*/

                    <span class="sr-only">导航开关</span>

                    <span class="icon-bar"></span>

                    <span class="icon-bar"></span>

                    <span class="icon-bar"></span>

                </button>

                <a class="navbar-brand" href="#">CEO</a>

                <!-- navbar-brand的a元素使文本看起来更大一号。 -->

                </div>

                <div class="collapse navbar-collapse" id="bs-1">

                <ul class="nav navbar-nav">

                    <li class="active"><a href="#">Link</a></li>

                    <li><a href="#">Link1</a></li>

                    <li><a href="#">Link2</a></li>

                    <li><a href="#">Link3</a></li>

                    <li><a href="#">Link4</a></li>

                </ul>

            </div>

        </nav>

    </div>

1. 固定在顶部

<nav class="navbar navbar-default navbar-fixed-top" role="navigation">

...

</nav>

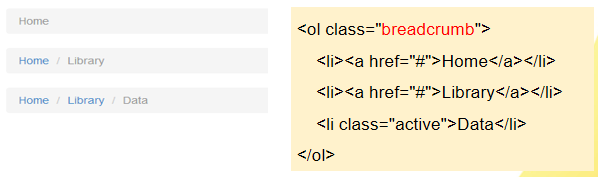
1. 固定在底部

<nav class="navbar navbar-fixed-bottom" role="navigation">

...

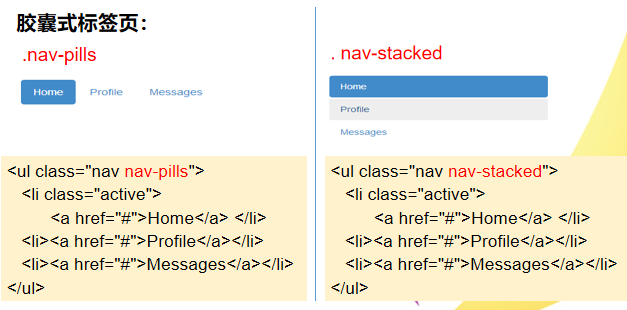
</nav>

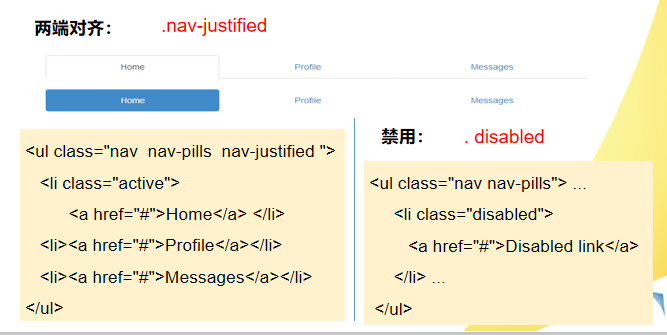
1. 面包屑

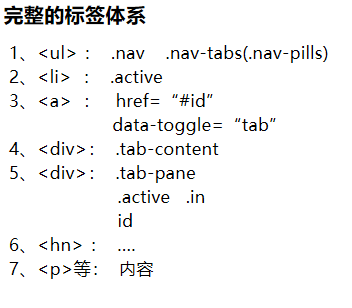


1. 标签页





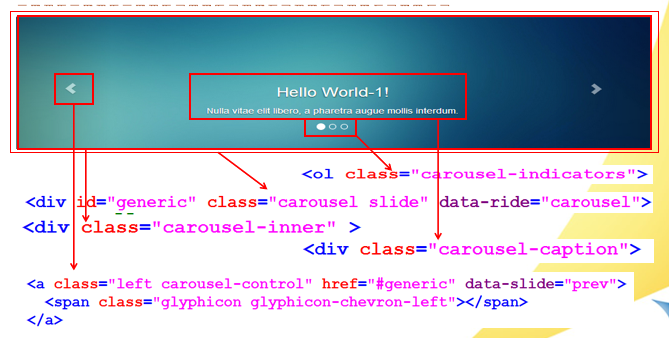




1. 分页
2. 标签和徽章
3. 缩略图
4. 多媒体
5. 进度条
6. 列表组

第二十五讲 Bootstrap插件

1. 模态框
2. 滚动监听
3. 折叠
4. 轮播





JS操作：