第 4 章

HTML5 高级开发标签

- **标记：<canvas>**
- html5新标签，由 Apple 引入。
- 可用于绘制图形，制作照片，创建动画，甚至可进行实时视频处理或渲染。
- <canvas> 只是**图形容器**，必须使用**脚本**来绘制图形。
- 有属性、方法和事件，包括绘图方法；
- 有多种绘制路径、矩形、圆形、字符、添加图像的方法。
- <canvas> ➜ 创建矩形区域，默认：300×150

◆ **语法**

```
<canvas id=# width=# height=# >
    浏览器不支持此标签时，提示文字
</canvas>
```

- 默认：**width**=300，**height**=150
- 可用CSS定义大小，但绘制时图像会伸缩适应其尺寸

✓ **canvas.getContext(contextID)**

- 返回画布绘图环境
- **contextID="2d"** → 二维绘图，唯一合法值

◆ 绘制矩形

– 运用JS命令绘制。

```
cans = document.getElementById(#);
ctx = cans.getContext("2d");
```

➢ 填充矩形

```
ctx.fillRect(x, y, width, height);
```

– x, y	➜ 矩形起点坐标
– width, height	➜ 矩形的宽、高

- 描边矩形

```
ctx.strokeRect(x, y, width, height);
```

- x, y → 矩形起点坐标
- width, height → 矩形的宽、高

- 清除矩形

```
ctx.clearRect(x,  y, width, height);
```

- 清除指定的矩形区域
- 这块区域会完全透明

4-2

◆ 绘制线条（图形）

(1) 创建起点

```
ctx.beginPath();
```

– 新建一条路径，图形绘制命令被指向到路径上生成路径

```
ctx.moveTo(x, y);
```

– 定义线条开始坐标：将笔触移动到指定坐标(x,y)上

(2) 绘制线条（路径）

ctx.lineTo(x, y);

- 绘制一条从当前位置到指定位置(x,y)的直线

ctx.arc(x, y, radius, startAngle, endAngle, anticlockwise);

- 画一个以（x,y）为圆心，以radius为半径的圆/弧，从startAngle开始到endAngle结束。

anticlockwise=ture|false	➜ 逆时针 | 顺时针（默认）

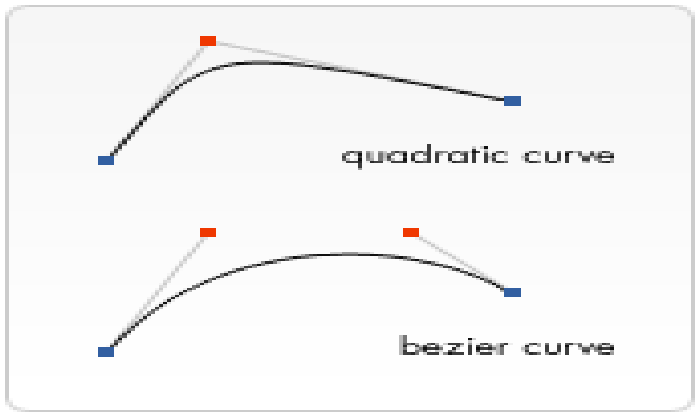
- 角度单位：弧度
- 弧度=(Math.PI/180)×角度

ctx.quadraticCurveTo(cp1x, cp1y, x, y);

- 绘制二次贝塞尔曲线
- cp1x,cp1y 为控制点一，(x,y)为结束点

ctx.quadraticCurveTo(cp1x, cp1y, cp2x, cp2y, x, y);

- 绘制三次贝塞尔曲线
- 2个控制点，(x,y)为结束点



(3) 封闭路径

```
ctx.closePath();
```

- 若绘制封闭图形 → 闭合路径
- 图形绘制命令又重新指向到上下文中。

(4) 渲染图形

– 路径生成后，可描边或填充路径区域，→渲染图形

```
ctx.stroke();
```

– 通过线条来绘制图形轮廓

```
ctx.fill();
```

– 填充路径的内容区域，→生成实心图形

4-3

◆ 修饰

➢ 色彩

ctx.fillStyle = color	设置图形的填充颜色
ctx.strokeStyle = color	设置图形轮廓的颜色

– 通过线条来绘制图形轮廓

➢ 透明度

ctx.globalAlpha = #	设置图形的填充颜色
# = [0,1]	0=完全透明, 1=完全不透明

– ctx.fillStyle = rgba(255,0,0,0.5);

➢ 渐变

ctx.createLinearGradient(x1, y1, x2, y2)

- 渐变 → 起点=(x1,y1)，终点=(x2,y2)

ctx.createRadialGradient(x1, y1, r1, x2, y2,r2)

- 起点： (x1,y1) 为原点，半径为 r1 的圆
- 终点： (x2,y2) 为原点，半径为 r2 的圆

gradient.addColorStop(position,color)

- 创建gradient 对象后 →上色
- **position**=[0,1.0] →渐变中颜色的相对位置。0.5=中间
- **color** = #F00、rgba(0,0,255,1)

4-5

◆ 绘制文本

- 使用 canvas 绘制文本 ➜ 特效文字。

```
ctx.fillText(text, x, y[, maxWidth])
```

- 在指定(x,y)位置填充指定文本（实心文本）
- **maxWidth** ➜ 绘制的最大宽度，可选

```
ctx.strokeText(text, x, y[, maxWidth])
```

- 在指定的(x,y)位置绘制文本边框（空心文本）

```
ctx.font = "48px serif "
```

- 定义字体

◆ Geolocation

- 地理定位，HTML5 Geolocation API，允许用户在web应用程序中共享位置，能够享受位置感知服务。
- 位置信息：经度、纬度
- Geolocation API 不指定设备使用哪种技术来定位的，只用于检索位置信息的API，不能保证数据的精确性。
- 位置信息，来源包括：

➢ IP地址定位

- 查找用户IP地址，检索其注册的物理地址
- **优点**：任何地方都可以，在服务器端处理；**缺点**：定位不准确。

- **卫星定位**
  - 通过卫星信号定位，GPS、格洛纳斯、伽利略、北斗。
  - **优点**：比较准确
  - **缺点**：室内效果不好，需要硬件设备支持
- **wi-fi定位**
  - 通过三角距离（多个wifi接入点）计算得到。
  - **优点**：准确，简单快捷，可室内定位
  - **缺点**：适合大城市，无接入点地区不可用

- **手机定位**
  - 通过用户到基站的三角距离确定，通常和WIFI、GPS结合使用。
  - **优点**：非常精确，简单快捷，可室内定位
  - **缺点**：需要基站
- **自定义定位**
  - 用户输入地理位置信息。

◆ **Geolocation API**

– 存在于navigator对象中，包含3个方法：

① **getCurrentPosition**

– 获取当前位置

② **watchPosition**

– 监听用户，位置改变时捕获

③ **clearWatch**

– 取消watch

4-8

➢ getCurrentPosition

– 单次位置请求。

navigator.geolocation.getCurrentPosition(successCallback, [errorCallback], [options])

successCallback	必选参数，位置请求成功后处理
errorCallback	可选参数，位置请求错误后处理
options	可选参数，设置数据搜集方式
enableHighAccuracy	true|false, →是否启用高精度模式
Timeout	超时数，否则→errorCallback
maximumAge	整数，重新获取位置信息的时间间隔

✓ **successCallback**

– 表示获取到的用户数据位置，包含2个属性。

coords	**accuracy**	精确度
	latitude	纬度
	longitude	经度
	altitude	海拔
	altitudeAcuracy	海拔高度的精确度
	heading	朝向
	speed	速度
timestamp		时间戳

✓ **errorCallback**

– 表示出现错误，返回的错误代码，包含2个属性。

message		错误信息
code	**unknow_error**	未知错误，以在message中查找
	permission_denied	用户拒绝浏览器获取位置信息的请求
	position unavalablf	网络不可用或者连接不到卫星
	timeout	获取超时时

➢ watchPosition

– 重复性位置请求。

```
navigator.geolocation.watchPosition(successCallback,
[errorCallback], [options])
```

successCallback	必选参数，位置变化，服务器就会调用
errorCallback	可选参数，位置请求错误后处理
options	可选参数，设置数据搜集方式
enableHighAccuracy	true|false，→是否启用高精度模式
Timeout	超时数，否则→errorCallback
maximumAge	整数，重新获取位置信息的时间间隔

- **clearWatch**

– 取消正在进行的watchPosition调用。

```
navigator.geolocation.clearWatch(watchID)
```

◆ 语义化标记

- HTML5之前，采用DIV+CSS布局页面→文档结构不够清晰，不利于搜索引擎爬取。
- 解决：新增语义化标签→通过标签名就能判断出其语义的标签。
- 内容结构化(内容语义化)，选择合适的标签(代码语义化)，便于阅读、书写、解析。

➢ 引入语义化标签的好处

- 比原标签有更加丰富的含义，方便开发与维护
- 搜索引擎能更方便的识别页面的每个部分
- 方便其他设备解析（如移动设备、盲人阅读器等）。

HTML5新增的语义化标签	
标签名	描述
header	标记头部区域的内容（用于整个页面或页面中的一块区域）
footer	标记脚部区域的内容（用于整个页面或页面中的一块区域）
section	Web页面中的一块区域
article	独立的文章内容
aside	相关内容或者引文
nav	导航类辅助内容
hgroup	组合网页或区段的标题
figure	对元素进行组合

- <header>
  - 页眉，通常放在页面或者某个区块元素的顶部。
  - 包含：标题、简介等，起到导航作用。

```
<header>
    ---
</header>
```

  - 文档中可以包含多对<header>标签。

- <nav>

– 导航，可连接到网站其他页面，或当前页面的其它部分。

```
<nav>
    ---
</nav>
```

– 文档中可以包含多对<nav>标签。

– 只用于页面的主要导航部分。

- <aside>
  - 侧边栏，包含的不是页面主要内容、具有独立性。
  - 是对页面的补充。

```
<aside>
    ---
</aside>
```

  - 一般用在页面、文章的侧边栏、广告、友情链接等区域。

- <footer>
  - 页脚，一般放在底部。
  - 包含版权信息、联系方式等信息

```
<footer>
    ---
</footer>
```

  - 可以在任意需要的区块底部使用。

- <article>
  - 文章，一段独立的内容。
  - 可被独立发布或者重新使用。

```
<article>
    ---
</article>
```

  - 通常情况下，<article>元素包括标题、正文和脚注。

- <section>
  - 文档中区段，是一个主题性的内容分组。
  - 通常用于对页面进行分块或者对文章等进行分段。

```
<section>
    ---
</section>
```

  - 通常包含一个头部<header>、可能还会包含一个尾部<footer>。

- <hgroup>

– 网页标题的组合，用于对网页或区段的标题进行组合。

```
<hgroup>
    ---
</hgroup>
```

- <time>
  - 定义日期和时间。
  - 公历时间（24 小时制）或日期，时间和时区偏移是可选的。

✓ 每天早上 <time>9:00</time>开始营业。

✓ <time datetime="1949-10-01">中国人民共和国成立！ </time>

- <figure>
  - 表示独立的流内容（图像、图表、照片、代码等）。

```
<figure>
    ---
</figure>
```

- **<input>**
  - 新增属性 → 详见第5章。

- **<button>**

- 按钮。

```
<button onclick=# >
    ---
</button>
```

- **onclick =** 点击按钮触发的事件

- <button> 与 <input type="button" > 相比，提供了更为强大功能和更丰富内容

- 在 button 元素内部，可以放置内容，比如文本或图像。

本章作业

◆ **课后作业**		**第122页**
1	**选择题：**	(1)(2)(3)(4)
2	**简答题：**	(1)(2)
3	**填空题：**	(1)(2)
◆ **上机练习**		
➢	设计页面：	实验1、2
➢	设计页面：	案例 4.5：创建魔方玩具效果