

课程计划安排

第一周 (HTML CSS)

- 1.系统回顾HTML5 CSS样式
- 2.CSS样式模板，HTML+CSS项目

第二周 (JS)

- 1.JS
- 2.jQuery
- 3.JSON
- 4.JS综合应用

第三周课程计划 (UI)

- 1.LayUI/LayUI-Vue/Bootstrapping
- 2.ElementUI
- 3.VUE

网页结构

```
<!DOCTYPE html><!--H5标签-->
<html lang="en"><!--网页根标签-->
<head><!--头部标签-->
    <!--
        ISO-8859-1编码<GBK<UTF-8
    -->
    <meta charset="UTF-8"><!--字符集-->
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <!--浏览器展示内容-->
</body>
</html>
```

网页常用标签

HTML 标题：HTML 标题是通过

标签来定义的

实例：

```
<h1>这是一个标题</h1>
```

HTML段落：HTML 段落是通过标签

来定义的。

```
<p>这是一个段落。</p> <p>这是另外一个段落。</p>
```

HTML 链接：HTML 链接是通过标签 来定义的。

```
<a href="https://www.runoob.com">这是一个链接</a>
```

HTML 图像：HTML 图像是通过标签 来定义的。

```

```

加粗

斜体

下划线

属性: target: 目标窗口打开方式

属性: src: 里面写图片的相对路径或者绝对路径 alt: 图片未加载出来的文字解释 height (高度) 与 width (宽度) 属性用于设置图像的高度与宽度。

-
表格元素中: 行,列,表头

实例：

```
<table border="1">
  <tr>
    <th>Header 1</th>
    <th>Header 2</th>
  </tr>
  <tr>
    <td>row 1, cell 1</td>
    <td>row 1, cell 2</td>
  </tr>
  <tr>
    <td>row 2, cell 1</td>
    <td>row 2, cell 2</td>
  </tr>
</table>
```

div元素:

```
<div>定义了文档的区域, 块盒, 独占一行
```

span元素

[用来组合文档中的行内元素, 内联元素](#)

表单元素:

表单是一个包含表单元素的区域

表单输入元素: 输入类型是由 type 属性定义

实例:

```
<form>
First name: <input type="text" name="firstname"><br>
Last name: <input type="text" name="lastname">
</form>
```

密码:

```
<form>
Password: <input type="password" name="pwd">
</form>
```

单选按钮:

```
<form action="">
<input type="radio" name="sex" value="male">男<br>
<input type="radio" name="sex" value="female">女
</form>
```

复选框:

```
<form>
<input type="checkbox" name="vehicle" value="Bike">我喜欢自行车<br>
<input type="checkbox" name="vehicle" value="Car">我喜欢小汽车
</form>
```

[按钮:](#)

[提交按钮:](#)

```
<form name="input" action="html_form_action.php" method="get">
  Username: <input type="text" name="user">
  <input type="submit" value="Submit">
</form>
```

[和区别:](#)

*button*就是单纯的点击，点击按钮后，出发的事件。

*submit*也是个按钮，他会提交表单，我们需要提交数据的时候用。

*submit*需要有表单时，提交时才会带数据。而*button*默认是不提交任何数据。那么它们的区别就出来了，如果没有表单的话，又想通过提交某些数据给后台进行回应，则需要通过*button*，当然使用*submit*也可以，但是前提要拦截`onclick`事件。当有表单的时候，如果提交的数据很多，那么使用*submit*比*button*要好，可以减少很多数据的获取动作。

[H5中视频音频标签:](#)

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <!--视频标签-->
  <div>
    <video>
      <source src="../first/1.mp4">
    </video>
  </div>
  <!--音频标签-->
  <div>
    <audio controls="controls">
      <source src="../first/1.mp3">
    </audio>
  </div>
</body>
</html>
```

[H5中新的类型:](#)

[颜色选择器](#)

```
<div>
  <input type="color" name="" id="">
</div>
```

[日期选择器:](#)

```
<div>
  <label> 日期选择器</label>
  <input type="date" id="" value="">
</div>
```

邮箱选择器：

```
<div>
  <label> email</label>
  <input type="text" name="email">
</div>
```

数字选择器：

```
<div>
  <label> 范围</label>
  <input type="range" name="range" min="0" max="100" value="90">
</div>
```

下拉框：

```
<div>
  <input list="lists">
  <datalist id="lists">
    <option value="1">
    <option value="2">
  </datalist>
</div>
```

*input*输入框属性: *autofocus*:获取输入框焦点 *autocomplete="off"*不记录输入框输入记录
placeholder:输入框提示信息***(有*value*值时*placeholder*不显示)**

required:必选选项

size: 文本框长度

maxlength: 允许输入的最大字符长度

```
<div>
  <input type="text" autofocus autocomplete="off">
  <input type="text" placeholder="off">
</div>
```

Canvas (画布)

*canvas*标签用于定义图形，其必须使用脚本 (JS) 来绘制，*canvas*标签相当于画布的容器。在JS脚本中可以通过*Canvas*绘制盒子、线以及添加图像。

```
<canvas id="mycanvas" width="" height="" style="border:1px solid red;">
</canvas>
<script>
  var c=document.getElementById("mycanvas");
  var ctx=c.getContext("2d");
  ctx.fillStyle="#FF0000";
  ctx.fillRect(0,0,300,120);
</script>
```

```
</script>
```



画线

moveTo(x,y) 定义线条开始坐标
lineTo(x,y) 定义线条结束坐标

```
<canvas id="mycanvas" width="" height="" style="border:1px solid red;">
</canvas>
<script>
var c=document.getElementById("mycanvas");
var ctx=c.getContext("2d");
ctx.moveTo(0,0);
ctx.lineTo(300,150);
ctx.stroke();
</script>
```

画圆

arc(x,y,r,start,stop)

```
<canvas id="mycanvas" width="" height="" style="border:1px solid red;">
</canvas>
<script>
var c=document.getElementById("mycanvas");
var ctx=c.getContext("2d");
ctx.beginPath();
ctx.arc(95,50,40,0,2*Math.PI);
ctx.stroke();
</script>
```

SVG

SVG用于定义可缩放矢量图形，H5中支持内联SVG，HTML中使用

标签定义SVG的容器，SVG可以绘制框 圆 文本等操作。

1、WWW(万维网) 的标准

2、相比(JPEG、GIF、ICO)图像文件来说，SVG图像可以通过文本编辑器来创建和修改

4、SVG可以伸缩

My first SVG

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <circle cx="100" cy="50" r="40" stroke="black"
    stroke-width="2" fill="red" />
</svg>
```

外部样式：

```
<svg width="300px" height="200px" version="1.1" baseProfile="full"
  xmlns="http://www.w3.org/2000/svg">
  <rect width="100%" height="100%" fill="red"></rect>
</svg>
```

内联样式：

内联样式

2、object

```
<object type="image/svg+xml" data="./svg/svg_filter.svg"></object>
```

3、iframe

```
<iframe src="svg/svg_filter.svg"></iframe>
```

rect:定义矩形

属性:width和height用于定义矩形的大小

CSS的fill属性定义矩形的填充（背景颜色），浏览器对rgb的方式解析更加友好

stroke-width:定义矩形边框的宽度

stroke:定义矩形边框的颜色

fill-opacity:设置填充颜色透明度(0-1)

opacity:设置整体颜色透明度(0-1)

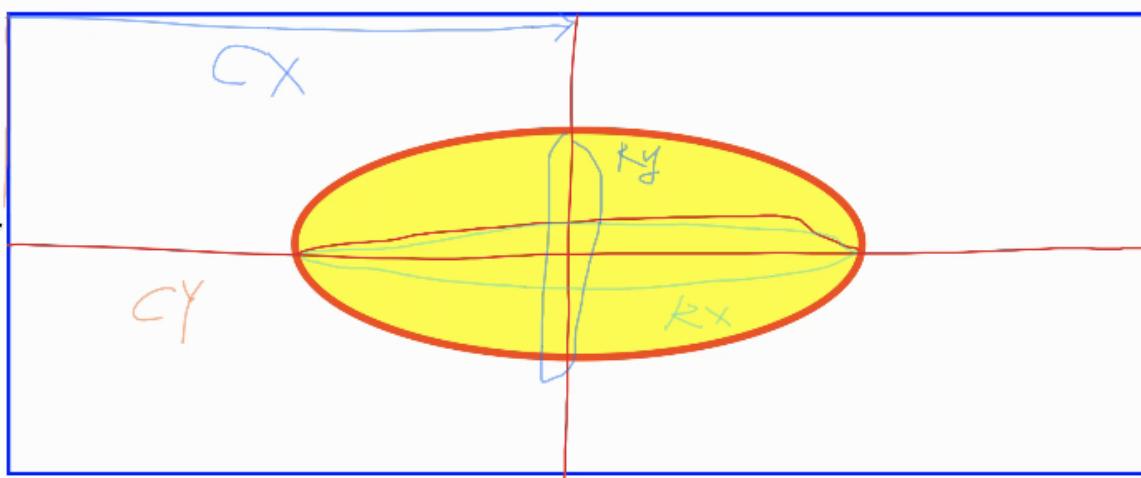
```
<svg width="300px" height="200px" version="1.1" baseProfile="full"
  xmlns="http://www.w3.org/2000/svg">
  <rect width="100%" height="100%" fill="red"></rect>
</svg>
```

```

<svg width="300" height="120" style="border:1px solid red">
  <circle cx="150" cy="60" r="50" stroke="black"
    stroke-width="2" fill="red"></circle>
</svg>
<!--
  circle: 定义圆
  cx: x轴偏移值
  cy: Y轴偏移值
  r: 圆半径
  stroke: 边框颜色
  stroke-width: 边框宽度
  fill: 填充色
-->

```

椭圆(ellipse)



```

<svg width="300px" height="120px" style="border:1px solid red ;">
  <ellipse cx="150" cy="60" rx="75" ry="30"
    style="fill: yellow; stroke:red; stroke-width: 2;"></ellipse>
</svg>

```

SVG画奥迪车标

```

<div>
<svg width="400" height="300" style="border:3px solid red ; ">
  <circle cx="60" cy="60" r="50" stroke="black" stroke-width="2"
    fill="transparent"/>
  <circle cx="120" cy="60" r="50" stroke="black" stroke-width="2"
    fill="transparent"/>
  <circle cx="180" cy="60" r="50" stroke="black" stroke-width="2"
    fill="transparent"/>
  <circle cx="240" cy="60" r="50" stroke="black" stroke-width="2"
    fill="transparent"/>
</svg>
</div>

```

SVG画三角形



```
<svg width="200px" height="200px">
    <!-- <polygon points="0,0 0,150 100,150" style="fill:#ffffff;
stroke:#008c8c; stroke-width:3;" />
    <polygon points="100,0 200,0 200,150" style="fill:#ffffff;
stroke:#008c8c; stroke-width:3;" /-->
    <line x1="0" y1="0" x2="150" y2="60" style="stroke:#008c8c ;"></line>
    <line x1="0" y1="0" x2="0" y2="60" style="stroke:#008c8c ;"></line>
    <line x1="0" y1="60" x2="150" y2="60" style="stroke:#008c8c ;"></line>
</svg>
```

折线画三角形



```
<svg width="200px" height="200px">
    <polygon points="0,0 0,150 100,150" style="fill:#ffffff; stroke:#EC5C97;
stroke-width:3;" />
    <polygon points="100,0 200,0 200,150" style="fill:#ffffff;
stroke:#EC5C97; stroke-width:3;" />
</svg>
```

//points内写需要三角形的三点坐标

文本

```
<svg width="400" height="400" style="border: 1px solid red;">
<!--
    transform="rotate(40,30,20)" 旋转操作, 只针对块元素有效
-->
<text x="100" y="200" fill="red" transform="rotate(40,30,20)" style="font-size: 30px;">---开始学习---SVG</text>
<text x="100" y="200" fill="red" style="font-size: 30px;">---项目实训---
<!--构造换行操作-->
<tspan x="100" y="240">---开始学习---</tspan>
<tspan x="100" y="280">---前端课程---</tspan>
</text>
</svg>
```

超链接

```
<a xlink:href="http://www.baidu.com" target="_blank">
```

path (路径)

path主要是d属性，d属性可以构造不同类型的图形，包裹前面讲到线、圆、块等操作，其主要通过对应的属性参数来绑定操作：

M:画笔起始位置

L：画直线(x,y)坐标

H:一个参数，注明在X轴移动到的位置〔水平直线〕

V:一个参数，注明在y轴移动到的位置(垂直直线)

Z:从当前点画一条直线到路径的起点

画正方形：

```
<svg width="200px" height="200px">
  <path d="M 50 50 H 100 V 100 H 50 L 50 50 "></path>
</svg>
```

用M命令在坐标50, 50)创建起点

通过H命令在水平方向移动300(350,50)

通过V命令在垂直方向移动308 (350,350)

在通过H命令在水平方向移动到X轴为50的地方

最后在通过L 50 50回到起点位置

fill:填充

stroke:描边颜色

stroke-width:描边宽度

以上所有的命令均允许大小写字母

大写字母表示:绝对路径，绝对的参照点svg的原点(0,0)

小写字母表示:相对路径，相对的参照点是上一个位置

弧形操作 贝塞尔曲线

Q(q):二次贝塞尔曲线

T(t):平滑贝塞尔曲线

C(c): sanc贝塞尔曲线

S(s):平滑三次贝塞尔曲线

```
<svg width="200" height="200" style="border:3px solid red ;">
  <path d="
```

```

    M 10,10
    Q 100 70, 190 10
    " stroke="pink" stroke-width="55" fill="none" >
  </path>
</svg>

<path d="
M 10,10
L 100 70
L 190 10
" stroke="#888" stroke-width="5" fill="none" >
</path>

```

弧形

A命令用于画弧形，可以理解是圆或椭圆的一部分。

语法:

A参数1 参数2 参数3 参数4 参数5 参数6 参数7

说明:

参数1：椭圆的X轴半径

参数2：椭圆的Y轴半径

参数3：椭圆相当于坐标轴的旋转角度

参数4：标记绘制的弧形是大弧形(1)还是小弧形(0)

参数5：标记弧形是向顺时针(1)还是逆时针(0)

参数6：弧形终点X轴坐标

参数7：弧形终点Y轴坐标

弧形画圆

```

<svg width="500" height="500" style="border:3px solid red ;">

  <path d="
M 100,100
m -75 0
a 75,75 0 1 0 150,0
a 75,75 0 1 0 -150 ,0
" stroke="#888" stroke-width="5" fill="none" >
</path>
</svg>

```

```

<!--g标记group的简写，用来分组，其能把多个元素放在一组，对g标记实施样式会渲染到分组内的所有元素上-->
<g fill="transparent" stroke-width="10" stroke="red">
  <circle cx="100" cy="100" r="50"></circle>
  <circle cx="170" cy="100" r="50"></circle>
  <circle cx="240" cy="100" r="50"></circle>
  <circle cx="310" cy="100" r="50"></circle>
</g>

```

```

<!--defs模版标记
defs元素用于预定一个元素使其能够在SVG图像中重复使用，和g结合使用
-->
<defs>
  <g id="shape" fill="transparent" stroke-width="2" stroke="blue">
    <circle cx="100" cy="200" r="10"></circle>
  </g>
</defs>

```

