**HTML5-day01：**

复习：

异步请求 $.get() $.getJSON() $.ajax()

分页显示 —— 最难点

客户端请求：

my\_orders.php?uname=qd&curPage=3

服务器端响应：——pagerObject

{

totalRecord: 32,

pageSize: 5,

pageCount: 7,

curPage: 3,

data: [{},{},{},{},{}]

}

第三阶段：

jQuery：用于简化DOM操作

AJAX： 用于无刷新/提交的局部更新，把动态页面静态化

HTML5新特性：

接下来8天的目标：—— 阶段特点：内容零散杂乱

1. 表单新特性 —— 次重要
2. 视频和音频 —— 次重要
3. Canvas绘图 —— 最重要
4. SVG绘图 —— 最重要
5. 地理定位 —— 了解
6. 拖放API —— 次重要
7. Web Worker —— 了解
8. Web Storage —— 次重要
9. Web Socket —— 了解

今日目标：

1. 新的input type
2. 新的表单元素

表单元素新的属性

1.已经学过的input type

text、password、radio、checkbox、file、button、submit、reset、image、hidden

2.新的input type

提示：下述验证都必须放在一个表单元素中，表单提交时才会触发。 消息弹出在一个小窗口中，不是HTML元素，样式无法定制。

(1)**email**： 提供了邮箱格式的验证，形如：\w{1}@\w{1}

**(2)url**：提供了URL格式的验证，形如：[a-zA-Z]{1}:

**(3)number**：提供了数字格式的验证，形如：数字，提供了三个新的属性可用： max(最大) min(最小) step(步长)

**(4)tel**：在手机中弹出数字键盘，而不是字母键盘

(5)**search**：可能显示一个X用于清空已有输入

(6)**range**：范围选择、滑块选择，提供了三个新的属性可用： max(最大) min(最小) step(步长) value(初始值)

(7)**color**：颜色选择

(8)**date**：日期选择，弹出一个窗口，FF没有实现

**(9)month**：月份选择

(10)**week**：周选择

练习：通过三个滑块设置不同的颜色配比，修改DIV的背景色

3.已经学过的表单元素

提示：用户可以输入/操作的

input textarea select/option button

4.新的表单元素

提示：下述四个新元素仅用于提示，用户无法操作，也无法随着表单输入进行提交。

**(1)datalist**：数据列表，本身不可见，用于为其他输入框提供输入建议/提示：

<datalist id="menu">

<option>京酱肉丝</option>

<option>鱼香肉丝</option>

<option>青椒肉丝</option>

</datalist>

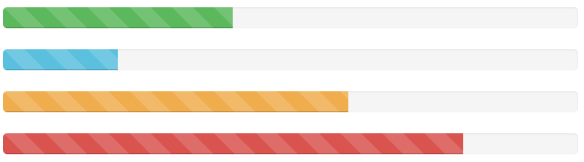
我的午餐：<input type="text" list="menu">

**(2)progress**：进度条

<progress></progress>

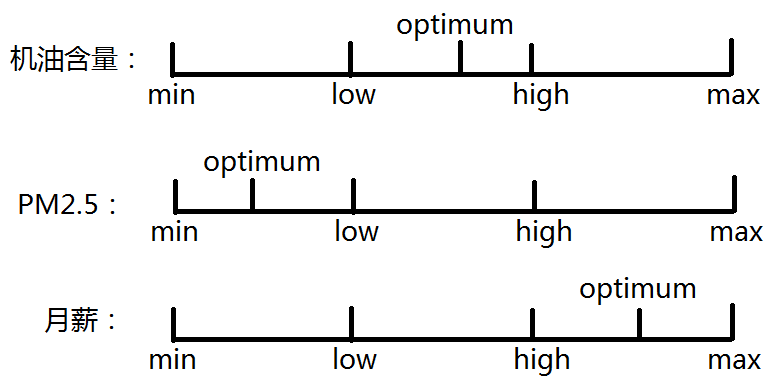
<progress value="10" min="0" max="100"></progress>

练习：使用定时器修改progress的进度值，100%时清除定时器



**(3)meter**：刻度尺

<meter min="0" max="100" low="30" high="70" optimum="50" value="当前值"></meter>



练习：创建三个meter元素，模拟上述三个场景；用三个定时器，不断的修改meter的当前值，观察颜色的变化

结论：

·最优值在上限和下限之间，黄-绿-黄

·最优值在下限之下，绿-黄-红

·最优值在上限之上，红-黄-绿

**(4)output：**输出值，语义标签，样式效果与span无异。

商品单价：￥3.5 购买数量：2 小计：￥7.0

练习：使用output代替span，实现“小计”的自动计算——监听购买数量的值的改变，一改变就要重新计算小计的值

5.已经学过的表单元素的属性

<input id="" class="" title="" style="" name="" value="" type="" checked selected disabled>

6.表单元素的新属性

(1)autocomplete：**on**/off，是否启用输入记录功能，下次实现自动完成效果

(2)autofocus：是否自动获得焦点，HTML中只要出现此属性就会自动申请输入焦点，提示：只有第一个声明此属性的输入域才能获得焦点。

(3)placeholder：占位符，用于在输入框中显示提示性文本。注意：占位符与value没有任何关系。

(4)multiple：是否允许存在多个输入，使用逗号分隔

(5)form：用于把输入域放到表单外部，指定当前输入域所属的表单

<form action="" id="f1"></form>

<input form="f1">

练习：创建一个表单，其中没有任何元素。后面再添加一个用户名输入域，自动获得输入焦点，提示文字在输入框内部；再添加一个URL输入框，允许输入多个URL地址；再添加一个提交按钮，点击后可以将上述所有输入进行提交。

与表单输入验证相关的新属性

(6)required：必填输入域

(7)maxlength：允许的最大字符长度，限制多余的输入

(8)minlength：允许的最小字符长度——并非H5标准属性，不能代替required属性

(9)max：允许的数字的最大值

(10)min：允许的数字的最小值

(11)step：允许的数字的步长，基于min开始的

(12)pattern：指定必须满足的正则表达式

练习：创建一个注册新用户的表单，提交时执行如下验证：

·用户名必填，长度不能小于3位

·密码必填，长度不能小于6位，不能大于9位

·年龄，不能小于20，不能大于60，必须可被5整除

·手机号码，必须满足大陆11位手机号格式

7.如何修改表单新特性中弹出的提示消息——难点&了解

HTML5新特性中有两类可能弹出错误提示消息：

(1)新的input type：

email：邮箱格式非法 ——>typeMismatch:true

url：地址格式非法 ——>typeMismatch:true

number：数字格式非法 ——>badInput:true

(2)表单元素新添的验证属性：

required： ——>valueMissing:true

maxlength： ——>tooLong:true

minlength： ——>tooShort:true

max： ——>rangeOverflow:true

min： ——>rangeUnderflow:true

step： ——>stepMismatch:true

pattern： ——>patternMismatch:true

要想修改上述错误提示信息，需要使用到HTML5元素对象的新属性：

**input.validity (有效性)** ValidityState{

valid: true 当前输入的值是否有效

badInput: false 错误的输入

customError: false 自定义错误

patternMismatch: false 正则表达式不匹配

rangeOverflow: false 值上溢出

rangeUnderflow: false 值下溢出

stepMismatch: false 步长不匹配

tooLong: false 字符串太长

tooShort: false 字符串太短

typeMismatch: false 输入的类型不匹配

valueMissing: false 值缺失

}

每个验证错误都有对应的验证属性，只要对应的验证属性变为true，就说明存在对应的错误。

而每个验证错误都有浏览器默认的错误消息。所有的这些错误消息中，某个时刻只能弹出一个——错误提示消息是在窗口中显示！

所有的错误消息中，“customError”自定义错误对应的错误消息的优先级最高。可使用 input.setCustomValidity('自定义错误') 生成一个自定义错误消息。

课后练习：自定义每一个验证错误提示消息

创建一个注册新用户的表单，要求进行如下输入及验证：

(1)用户名必填，长度不能小于3位

(2)密码必填，长度不能小于6位，不能大于9位

(3)重复密码必填，值必须与密码相同

(4)年龄，不能小于20，不能大于60，必须可被5整除

(5)手机号码，必须满足大陆11位手机号格式

(6)用户邮箱，必须满足EMAIL格式

(7)个人微博地址，必须满足URL格式

面试题——百度

HTML5新标签有哪些？标签新属性？

HTML5废弃的标签有哪些？废弃的标签属性？

**HTML5-day02:**

复习：

HTML5新特性：

表单新特性 —— 次重点

新的input type

email、url、search、number、tel、

range、color、date、month、week

新的表单元素

datalist、progress、meter、output

表单元素的新属性——重要

autocomplete、autofocus、placeholder、multiple、form、

required、maxlength、minlength、max、min、step

自定义自定义消息： input.validity

音频和视频 —— 次重点

Canvas绘图 —— 重点

SVG绘图 —— 重点

地理定位

拖放API

Web Worker

Web Storage

WebSocket

今日目标：

视频和音频 —— 次重点

Canvas绘图 —— 重点&难点

1.HTML5新特性之二——视频和音频

HTML5之前播放声音有两种方式：

(1)<body bgsound="x.mp3">

(2)<embed src="x.swf"></embed>

HTML5之后，播放视频和音频使用：

<video></video> 行内块元素

<audio></audio> 行内块元素

VIDEO标签常用的属性：

src：指定视频源URL

width：指定播放区域的宽

height：指定播放区域的高

controls：是否显示播放控件，默认为false

autoplay：是否自动播放，默认为false

loop：是否循环播放，默认为false

poster：在播放之前显示一张海报(播放之后无法再显示)，注意取消自动播放

preload：如何预加载视频，有三个可选项：

auto：自动预加载视频元数据，并缓存一定的时长

metadata：仅预加载视频的元数据，不缓存视频内容

none：不预加载视频元数据，不缓存视频内容

paused：JS中使用的只读属性，当前是否处于暂停状态

VIDEO对象的常用事件：

onpause： 当刚一暂停时触发此事件

onplay：当刚一开始播放时触发此事件

onplaying：当视频播放进行中触发此事件

VIDEO对象的方法：

play()： 播放视频

pause()：暂停视频

练习：视频播放区域浏览器中水平居中，只要一暂停，上方显示广告图片；只要一播放，去除广告图片。

练习：不使用video自带的播放控件，使用自定义图片(按钮功能)，控制视频的暂停和播放

提示：需要配合video.paused属性和 play()、pause()两个方法来实现

autoplay: true

controls: true

currentSrc: "http://127.0.0.1/HTML5\_day02/resource/birds.mp4"

currentTime: 0

defaultMuted: false

defaultPlaybackRate: 1

loop: true

muted: false

paused: true

poster: ""

preload: "auto"

seeking: false

src: "http://127.0.0.1/HTML5\_day02/resource/birds.mp4"

videoHeight: 0

videoWidth: 0

volume: 1

width: 0

AUDIO标签常用属性：

(1)src：指定视频源URL

(2)width：指定播放区域的宽

(3)height：指定播放区域的高

(4)controls：是否显示播放控件，默认为false

(5)autoplay：是否自动播放，默认为false

(6)loop：是否循环播放，默认为false

(7)preload：如何预加载视频，有三个可选项：

auto：自动预加载视频元数据，并缓存一定的时长

metadata：仅预加载视频的元数据，不缓存视频内容

none：不预加载视频元数据，不缓存视频内容

(8)paused：JS中使用的只读属性，当前是否处于暂停状态

AUDIO标签的常用事件和方法与VIDEO标签一致。

练习：页面中显示一个复选框“□播放背景音乐”控制网页/游戏背景音乐。

2.Canvas绘图技术

使用JS在网页中绘制各种图形(矩形、圆形...)和图像。应用场景：统计图表、在线画板、小游戏等。

使用方法：

<canvas id="c1" width="500" height="400">

您的浏览器不支持Canvas(画布)！

</canvas>

<script>

//Context：绘图上下文 Graphics

var ctx = c1.getContext('2d'); //获得画布上的画笔

//使用画笔绘制图形和图像....

</script>

data date

text test

content context(上下文)

绘图上下文对象所有的成员：

|  |
| --- |
| fillStyle: "#000000" 填充样式  font: "10px sans-serif" 字体  globalAlpha: 1 全局透明度  imageSmoothingEnabled: true 启用图片平滑  lineCap: "butt" 线条头部  lineJoin: "miter" 线条连接处  lineWidth: 1 线条宽度  shadowBlur: 0 阴影半径  shadowColor: "rgba(0, 0, 0, 0)" 阴影颜色  shadowOffsetX: 0 阴影X轴偏移量  shadowOffsetY: 0 阴影Y轴偏移量  strokeStyle: "#000000" 轮廓/描边样式  textAlign: "start" 文本对齐方式  textBaseline: "alphabetic" 文本基线  arc(): 拱形  arcTo(): 拱形到  beginPath(): 开始一条路径  bezierCurveTo(): 绘制贝塞尔曲线  clearRect(): rectangle 清除矩形范围  clip(): 裁切  closePath(): 闭合路径  createLinearGradient(): 创建线性渐变色  createRadialGradient(): 创建径向渐变色  drawImage(): 绘制图像  ellipse(): 绘制椭圆  fill(): 填充路径  fillRect(): 填充矩形  fillText(): 填充文本  lineTo(): 到指定点绘制一条直线  measureText(): 测量文本的宽度  moveTo(): 移动到指定点  rect(): 绘制矩形路径  resetTransform(): 重置变形  restore(): 恢复  rotate(): 旋转  save(): 保存  scale(): 缩放  stroke(): 勾勒轮廓/描边  strokeRect(): 描边矩形区域  strokeText(): 对文本进行文本  transform(): 变形  translate(): 平移 |

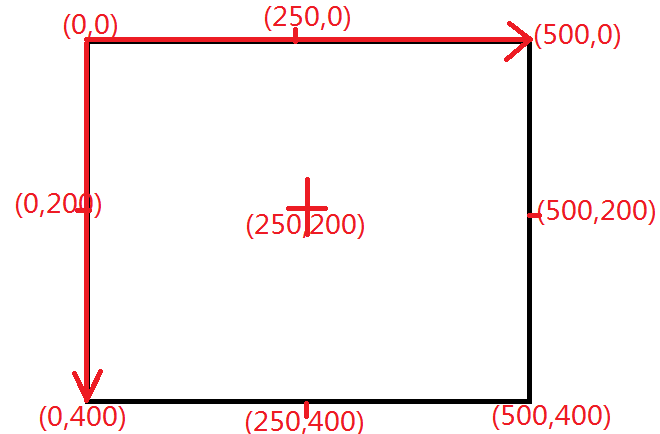
提示：Canvas难点：(1)单词难记 (2)坐标计算

3.使用Canvas绘制矩形——矩形的定位点左上角

ctx.strokeStyle = 颜色/渐变对象;

ctx.fillStyle = 颜色/渐变对象;

ctx.strokeRect(x, y, w, h) 描边一个矩形区域

 ctx.fillRect(x, y, w, h) 填充一个矩形区域

ctx.clearRect(x,y,w,h) 清除一个矩形区域内所有内容

var g = ctx.createLinearGradient(x1,y1,x2,y2);

g.addColorStop(偏移量, 颜色);

练习1)：在500x400的画布的四个角+中央绘制五个矩形(150x100)

练习2)：使用定时器每隔100ms绘制一个不断右移的矩形

练习3)：绘制一个在竖直方向来回移动的矩形

练习4)：绘制一个斜向移动的矩形，到了某个边缘再弹回来

4.使用Canvas绘制文本——文本的定位点在文本基线左侧

ctx.textBaseline = 'alphabetic/middle/bottom/top' 文本基线

ctx.font = '20px SimHei'; 设置大小和字体

ctx.measureText(txt).width 测量一段文本的总宽度

ctx.fillText(txt, x, y) 填充文本

ctx.strokeText(txt, x, y) 描边文本

练习：在500x400画布的四个角/中央都绘制“达内科技123”

且中央的文字使用彩虹色渐变色

5.使用Canvas绘制路径

参考PS中的“钢笔工具”：路径本身不可见！创建完路径有三种用途：

(1)描边 (2)填充(必须闭合) (3)裁剪图片

ctx.beginPath() //开始一条新路径

/\*\*\*\*\*\*绘制路径\*\*\*\*\*\*\*\*/

ctx.moveTo(x, y) 移动到指定点

ctx.lineTo(x,y) 从上一个点到指定点画一条直线

ctx.arc(cx,cy, r, sAngle, eAngle) 绘制一条圆拱形路径

ctx.ellipse(cx,cy, rx, ry, sAngle, eAngle) 绘制一条椭圆拱形路径

ctx.bezierCurveTo(.....) 绘制贝塞尔曲线

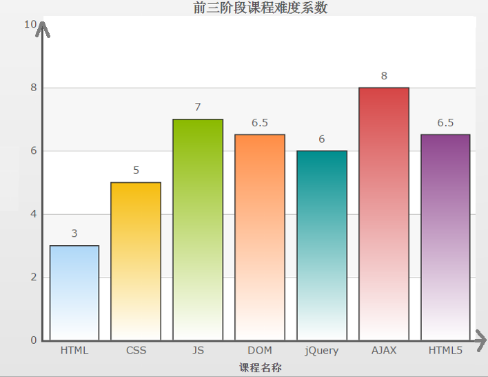
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ctx.closePath() //闭合当前路径

ctx.stroke() //使用当前路径进行描边

ctx.fill() //使用当前路径进行填充

ctx.clip() //使用当前路径进行裁切



6.使用Canvas绘制图像

练习内容：

假设前端页面获取到后台返回了如下的JSON数据：

'[{"label":"HTML",value:3}, {"label":"CSS",value:5},....]'

根据这段JSON字符串，绘制下图：

**HTML5-day03:**

复习：

HTML5新特性：

1. 表单新特性
2. 视频和音频
3. Canvas绘图
4. SVG绘图
5. 地理定位
6. 拖放API
7. Web Worker
8. Web Storage
9. WebSocket

Canvas：画布

Context：绘图上下文/画笔

ctx.strokeStyle = 颜色/渐变色

ctx.fillStyle = 颜色/渐变色

1. 绘制矩形——定位点在左上角

ctx.fillRect(x,y, w, h)

ctx.strokeRect(x,y,w,h)

ctx.clearRect(x,y,w,h)

1. 绘制文本——定位点基线的最左侧

ctx.textBaseline = 'alphabetic/top/bottom/middle'

ctx.font = '30px SimHei'

ctx.measureText(txt).width

ctx.fillText(txt, x, y)

ctx.strokeText(txt, x, y)

1. 绘制路径

ctx.beginPath()

ctx.moveTo(x,y)

ctx.lineTo(x,y)

ctx.closePath()

-----------------

ctx.stroke()

ctx.fill()

ctx.clip()

1. 绘制图像

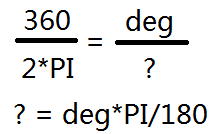
1.使用Canvas绘制路径

绘制圆形路径——圆形定位点在圆心

ctx.beginPath()

ctx.arc(cx,cy, r, sAngle, eAngle, [isCounterClock: false]) //绘制圆拱路径

ctx.stroke()/fill()/clip()

角度转换为弧度制的公式： 

练习1)：500x400的画布上，画圆弧(半径为50)：

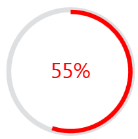
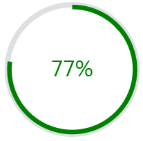
左上角：上半圆 右上角：下半圆

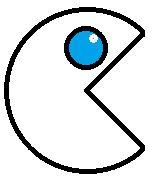
左下角：左半圆 右下角：右半圆

中央：满圆（描边+填充）

练习2)：使用定时器+圆弧，绘制一个圆形进度条

10:35~10:50



练习3)：绘制一个大眼汪汪的吃豆人，形如：

练习4)：绘制一个大眼汪汪的吃豆人，形如：

练习5)： 把练习3封装为一个函数：openMouth() 把练习4封装为一个函数：closeMouth()。使用定时器每隔300ms调用交替的调用这两个函数。

2.绘图图像——图像的定位点在左上角

var img = new Image();

img.src = 'x.png';

img.onload = function(){ //必须等待图片加载完成

//img.width img.height

ctx.drawImage( img, x, y );

ctx.drawImage( img, x, y, w, h ); //绘制图像进行拉伸

}

练习1)：在画布的四个角各绘制一个小飞机，中央绘制一个放大200%的飞机

练习2)：在画布上绘制一个飞来飞去的小飞机

练习3)：小飞机随着鼠标移动而移动

思路：canvas.onmousemove + event.offsetX + setInterval

练习4)：小飞机边飞边发子弹 —— 未完成！

总结：Canvas绘图

1. 绘制矩形

ctx.fillRect(x,y,w,h)

ctx.strokeRect(x,y,w,h)

ctx.clearRect(x,y,w,h)

1. 绘制文本

ctx.fillText(txt,x,y)

ctx.strokeText(txt,x,y)

ctx.measureText(txt).width

1. 绘制路径

ctx.beginPath()

ctx.moveTo(x,y) ctx.lineTo(x,y)

ctx.arc(cx,cy, r, sAngle, eAngle)

ctx.closePath()

----------------------------

ctx.fill() / ctx.stroke() / ctx.clip()

1. 绘制图像

ctx.drawImage(img, x, y)

ctx.drawImage(img, x, y, w, h)

3.补充小难点：如何旋转绘制图像

Canvas（画布）是不能旋转的。

Context（画笔）是可以旋转的。

ctx.rotate( deg \* Math.PI / 180 ) //在原有旋转角度基础上继续旋转deg度——旋转的**轴心是画布的坐标原点**(默认为左上角)

ctx.translate( x, y ) //把画布的坐标原点平移到新的点，记得修改图形/图像的定位点

练习：在画布正中央绘制一个小飞机，使用定时器，让它在原地打转。

Canvas在项目中的常用场景： 游戏、动态走势图、统计图表

4.常用的第三方绘制统计图表的工具

直接使用Canvas绘制统计图表比较麻烦，推荐使用第三方绘图工具库。

**免费的：**

Chart.js

FreeChart

**收费的：**

FusionChart

提示：第三方工具的使用不需要记忆，随查随用即可。

5.Chart.js的使用

官网：http://www.chartjs.org/

定义：Simple yet flexible JavaScript charting for designers & developers。

手册：http://www.chartjs.org/docs/

基础使用方法：

<canvas id="c13" width="500" height="400"></canvas>

<script src="js/Chart.js"></script>

<script>

new Chart(c13, {

type: 'bar', //line/pie/doughnut...

data: {

labels: ['HTML','CSS','JS','DOM','jQuery','AJAX','HTML5'],

datasets: [{

label: '各课程难度系数',

data:[3,5,7,6.5,6,8,7]

}]

}

});

练习1)：为每个bar指定特定的颜色

练习2)：取消响应式功能

练习3)：Y轴从0开始显示数据

课后练习：

1. 使用Canvas绘制一个随机改变的验证码图片



var str = 'ABCDEFGHJKLMNPQRSTWXY3456789';

var char = str[ 0~字符串长度间的随机数 ];

要求：

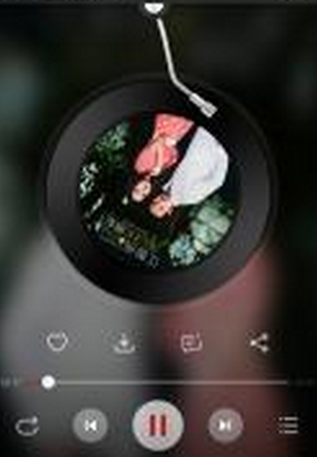
画布背景颜色随机(浅色) ctx.fillRect()

文字内容随机、大小随机，颜色随机(深色)、旋转角度随机。

5条随机干扰线（深色），处于文字上方。

100个杂色点（半径为1为圆），处于文字上方。

2.仿网易云音乐的播放界面



**HTML5-day04:**

复习：

1. 表单新特性
2. 视频和音频
3. Canvas绘图
4. SVG绘图
5. 地理定位
6. 拖放API
7. Web Worker
8. Web Storage
9. WebSocket

Canvas绘图知识点：

使用方法：

<canvas width="500" height="400"></canvas>

var ctx = canvas.getContext('2d');

使用画笔绘制图形图像：

ctx.strokeStyle = 颜色/渐变色;

ctx.fillStyle = 颜色/渐变色;

ctx.lineWidth = 6;

ctx.font = '10px sans-serif';

ctx.textBaseline = 'alphabetic';

绘制矩形：

ctx.strokeRect(x,y, w,h)

ctx.fillRect(x,y,w,y)

ctx.clearRect(x,y,w,y)

绘制文本：

ctx.strokeText(txt, x, y)

ctx.fillText(txt,x,y)

ctx.measureText(txt).width

绘制路径：

ctx.beginPath()

ctx.moveTo(x,y) ctx.lineTo(x,y)

ctx.arc(cx,cy, r, sAngle,eAngle, [false])

ctx.ellipse(cx,cy, rx,ry, sAngle,eAngle, false)

ctx.bezierCurveTo(...)

-----------------------------

ctx.stroke() / ctx.fill() / ctx.clip()

绘制图像：

ctx.drawImage(img, x, y)

ctx.drawImage(img, x, y, w, h)

提示：使用Canvas可以实现把一幅图片实现“高斯模糊”效果的——编写正态分布相关算法

今日目标：

(1)补充Canvas小知识

如何加载多张图片

图像/图形事件绑定

(2)SVG绘图 —— 重点 & 比较容易混淆

(3)two.js —— 了解

1.补充背景知识：

Chrome中：一个窗口可以有6个线程(工人)向服务器发起请求，可以同时工作；往浏览器显示区域中绘制内容只有1个线程(工人)。

Canvas绘图应用中，如果需要绘制多个图片，一般必须等待所有图片加载完成才能开始绘制。图片的加载完成顺序无法预测，必须等待。

2.如何为Canvas上的图像/图形绑定事件监听

只有HTML元素(Canvas)才能进行事件绑定。图形图像无法绑定事件监听。

方法： canvas.onmousemove = function(e){

var x = e.offsetX;

var y = e.offsetY;

if(x>=起点1 && x<=终点1 && y>=起点2 && y<=终点2){

//鼠标放在某个图形/图像上了

}

}

结论：非常不方便！

练习：在画布绘制一个“播放按钮的图片”，鼠标悬停到了图片上方，变为手型光标

前端领域可以使用绘图技术：

1. WebGL
2. Canvas —— HTML5标准中新特性
3. SVG —— HTML5标准中新特性

3.SVG绘图

Scalable Vector Graphic，可缩放的矢量图

**位图：**使用像素点描述图像，善于描述颜色细节变化，放大之后会出现马赛克问题。PS、Canvas

**矢量图**：使用线条、色块描述图像，不善于描述颜色细节变化，可以无限缩放而不失真。AI、SVG

* SVG
  + 矢量图，不依赖分辨率
  + 每个图形都是一个标签，支持事件监听
  + 最适合带有大型渲染区域的应用程序（例如百度地图）
  + 不适合游戏
* Canvas
  + 位图，依赖分辨率
  + 每个图形不是标签，不支持事件绑定
  + 能够以“.png”或“.jpg”格式保存结果图像
  + 最合适图像密集型的游戏

SVG在2002年就是W3C标准，远早于HTML5标准。纳入HTML5标准，SVG技术进行了一些删减，如动画、特效。

**HTML5之前如何使用SVG图形：**

1. 先创建一个.svg文件(是一个XML文档)
2. 再创建一个.html文件，其中使用IMG、IFRAME、EMBED、OBJECT标签引用.svg文件。

**HTML5之后如何使用SVG图形：**

SVG相关标签都已经纳入H5标签库，可以直接在.html中使用。

练习1)：在SVG图像上四个角各放置红橙黄绿四个矩形

练习2)：在SVG图像上放置一个矩形，来回移动——定时器

4.使用SVG绘制矩形

<rect x="200" y="160" width="100"

height="80" fill="rgba(0,0,255,.3)" stroke="#00f"></rect>

|  |
| --- |
| 使用SVG图形图像需要注意：   1. 普通CSS样式对svg图形大多无效，需要自己专用样式，或者声明为属性 2. SVG图形的属性都不是HTML属性，即不能使用 rect.fill = "#f00"来修改该元素的属性——SVG标签根本上属于XML标签，而不是HTML标签！ 可以使用核心DOM操作方法来控制SVG标签的属性： setAttribute() .getAttribute() 3. document.createElement()创建都是标准的HTML标签，不能用于创建SVG元素！document.createElementNS('http://www.w3.org/2000/svg', 'SVG标签名')，使用此方法创建的标签才能被SVG解释器解析。 |

练习3)：SVG中绘制三个矩形，点击某个，它慢慢变大、变淡，直至消失

练习4)：SVG中点击某处，在此处生成一个颜色随机的矩形

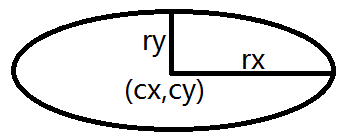
5.使用SVG绘制圆形

<circle cx="" cy="" r="" fill="" stroke=""></circle>

练习1)：在SVG的四个角+中央放置五个圆形，有轮廓、有填充

练习2)：点击SVG的任意位置，生成一个颜色随机的圆形，慢慢变大变淡直至消失.... 蜻蜓点水效果

6.使用SVG绘制椭圆

 <ellipse cx="" cy="" rx="80" ry="40" ....></ellipse>

练习1)：在SVG的四个角+中央放置五个椭圆，有轮廓、有填充

7.使用SVG绘制直线

<line x1="" y1="" x2="" y2=""></line>

练习：绘制如下图标：



提示：可以把多个SVG元素相同的属性放在一个公共的父元素中——称为“元素组”: <g fill="" stroke=""></g>；组员会自动继承小组的属性设置。

8.使用SVG绘制折线

<polyline points="x1,y1 x2,y2 x3,y3 .... "></polyline>

一个点的X和Y用逗号分隔；两个点之间用空格分隔

9. 使用SVG绘制多边形

<polygon points="x1,y1 x2,y2 x3,y3 .... "></polygon>

注意：多边形与折线的区别：多边形会自动闭合起始点，折线不会。

练习：绘制多边形 16:58~17:10



10.使用SVG绘制文本

<text x="" y="" font-size="" font-family="">文本内容</text>

提示：SVG中的文字是可以选中的——标签的innerHTML，Canvas中的fillText()做不到这一点。SVG中的文字内容够容易搜索引擎查询到，排名会靠前。

11.使用SVG绘制图像

<image xlink:href="img/disc.png" x="250" y="200" width="200" height="200"></image>

注意：SVG图形中一旦使用位图，就不能无限缩放了。

12.对SVG图形/图像使用定义特效 —— 了解

定义和使用特效：

<svg>

<defs> <!--定义/声明特效元素-->

<特性元素 id="特效ID"></特效元素>

</defs>

<rect 使用某个特效> <!--使用特效元素-->

</svg>

**(1)渐变色特效： 线性渐变、径向渐变**

<defs>

<linearGradient id="g1" x1="0%" y1="0%" x2="100%" y2="0%">

<stop offset="0%" stop-color="red"></stop>

<stop offset="50%" stop-color="yellow"></stop>

<stop offset="100%" stop-color="green"></stop>

</linearGradient>

</defs>

<rect x="50" y="150" width="400" height="100" fill="url(#g1)"></rect>

**(2)滤镜特效： 模糊滤镜-高斯模糊**

<svg id="s13" width="500" height="400">

<defs>

<filter id="mh">

<feGaussianBlur stdDeviation="3"></feGaussianBlur>

</filter>

</defs>

<text x="100" y="100" filter="url(#mh)">达内科技</text>

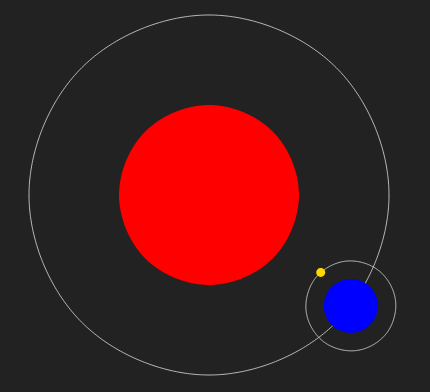
</svg>

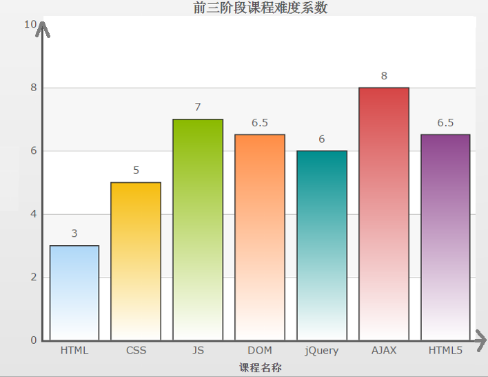
课后练习：

1. 用折线绘制五角星
2. 假设前端页面获取到后台返回了如下的JSON数据：

'[{"label":"HTML",value:3}, {"label":"CSS",value:5},....]'

根据这段JSON字符串，绘制下图：





3)自学two.js工具的使用，理解其作用，仿写官方示例代码，学会使用方法，实现如下效果：

**HTML5-day05:**

复习：

HTML5新特性：

1. 表单新特性
2. 视频和音频
3. Canvas绘图——统计图表、游戏

var ctx = canvas.getContext('2d');

ctx.fillRect() ctx.fillText() ctx.beginPath() ctx.drawImage()

1. SVG绘图——统计图表、图标/LOGO、地图

rect circle ellipse line polyline polygon text image g defs

1. 地理定位
2. 拖放API
3. Web Worker
4. Web Storage
5. WebSocket

今日目标：

1. Two.js —— 掌握
2. HTML5提供的地理定位 —— 了解
3. 百度地图API —— 掌握
4. 拖放API —— 小重点

Web领域可用的绘图技术：

1)WebGL

2)Canvas

3)SVG

1.Two.js

是一个第三方二维绘图函数库，可以在WebGL、Canvas、SVG上绘制图形，从而屏蔽三者在使用上的不同。

官网+手册：http://jonobr1.github.io/two.js/

Two.js is a two-dimensional drawing api geared towards modern web browsers. It is renderer agnostic enabling the same api to draw in multiple contexts: svg, canvas, and webgl.

2.Three.js

是一个第三方的三维绘图函数库。

3.HTML5新特性之五——地理定位

Geolocation：用于确定浏览器所在的地理坐标，包括：经度(longitude)、纬度(latitude)、海拔高度(altitude)、速度(speed)，用于实现LBS（Location Based Service）。

百度地图中拾取某一点的经纬度：

http://api.map.baidu.com/lbsapi/getpoint/index.html

HTML5新特性：geolocation，用于返回当前浏览器所在的地理坐标。

手机浏览器：使用内置的GPS模块、手机基站确定定位

PC浏览器：使用IP地址反向推导，精度最差

window.navigator.**geolocation** {

getCurrentPosition(succCB, errCB) 获得当前浏览器所在定位数据

watchPosition() 使用定时器不停的获得定位数据

clearWatch() 清除监视定时器

}

|  |
| --- |
| 项目中可能用到的国内的第三方平台工具：   1. 新浪微博API 2. 腾讯微信API 3. 百度地图API —— 手册最通俗！ 4. 阿里支付API |

API：接口，Application Programming Interface，应用程序编程接口，某个系统/平台提供的供其他应用程序所调用的URL、对象、函数等等。

4.扩展知识：使用百度地图API

**官网**：http://lbsyun.baidu.com/

**注册用户**：https://passport.baidu.com/v2/?reg&regType=1&tpl=mn&u=http://lbsyun.baidu.com/

**登录即可**。

  **手册主页**：http://lbsyun.baidu.com/index.php?title=jspopular

使用步骤：

(1)为自己的网站应用申请一个唯一的密钥

(2)仿照“开发指南”编写示例程序

<script src="http://api.map.baidu.com/api?v=2.0&ak=您的密钥"></script>

<script>

// 创建地图实例

var map = new BMap.Map("container");

// 创建点坐标

var point = new BMap.Point(116.3,39.9);

// 设置中心点坐标和地图级别(1~19)

map.centerAndZoom(point, 15);

</script>

1. 创建一个基本地图应用
2. 添加控件(Control)
3. 添加覆盖物(Overlay)
4. 反向解析：地名->坐标
5. 获得浏览器当前所在地理坐标(代替了navigator.geolocation)

5.HTML5新特性之六——拖放API

Drag & Drop

注意：拖放API在不同的浏览器，甚至在不同版本的Chrome中都细节的不同。

HTML5中，拖放API指七个事件：

**源对象可以触发的事件(3个)：**

src.ondragstart：刚开始被拖动

src.ondrag：被拖动中

src.ondragend：拖动结束

**目标对象可以触发的事件(4个)：**

target.ondragenter：源对象被拖动着进入目标对象

target.ondragover：源对象被拖动在目标对象上方

target.ondragleave：源对象被拖动着离开目标对象

target.ondrop：源对象在目标对象上方释放

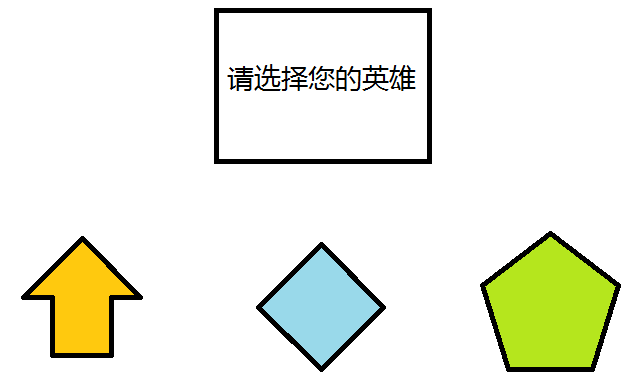
提示：dragover事件的默认后续行为就是触发dragleave，即使用户松手也触发该事件。必须组织其默认行为，才可能触发drop事件！

练习1)：页面创建一个可以随意拖动，边拖动边移动位置的IMG

提示：随着“拖动中”事件的进行，修改img的left/top——必须修改定位方

练习2)：页面创建三个可以随意拖动，边拖动边移动位置的IMG

课后练习： 英雄选择



要求：

拖动某架飞机到目标区域，则在上方显示出该飞机；下方该飞机图片变为灰色；

拖动另一架飞机到目标区域，则显示出该飞机，之前选中的飞机变为彩色。

**HTML5-day06:**

复习：

HTML5新特性：

表单新特性

视频和音频

Canvas绘图

SVG绘图

地理定位

navigator.geolocation {

getCurrentPosition(succCB,errCB)

watchPosition()

clearWatch()

}

拖放API

Web Worker

Web Storage

WebSocket

今日目标：

补充拖放API知识点

Web Worker —— 重要理论

Web Storage —— 重点

WebSocket —— 理论学习

1.HTML5新特性之六——拖放API

源对象可以触发的事件：

ondragstart

ondrag

ondragend

目标对象可以触发的事件：

ondragenter

ondragover

ondragleave

ondrop

拖放源对象和目标对象间如何传递数据：

(1)使用全局变量——污染全局对象

src.ondragstart：为全局变量赋值

target.ondrop：读取全局变量的值

(2)拖放事件的新属性 event.dataTransfer（数据传递对象——拖拉机）

src.ondragstart：

e.dataTransfer.setData(k, v)

target.ondrop：

e.dataTransfer.getData(k)

注意：此方法只能传递string数据。

HTML5标准中：k只能是三种可能之一：text/uri-list、text/html、Files

但不同的浏览器有不同的扩展。

2.扩展知识点：如何在网页中拖放显示一张客户端的图片——了解

(1)必须阻止拖放本地图片到浏览器，默认在新窗口中打开的行为

document.ondragover = function(e){

e.preventDefault();

}

document.ondrop = function(e){

e.preventDefault();

}

(3)为目标div元素绑定ondrop事件

target.ondrop = function(e){

//通过e.dataTransfer对象获取源对象拖动时传递的数据

var f0 = e.dataTransfer.files[0]; //File：文件对象

var fr = new FileReader(); //文件读取器对象

fr.readAsDataURL(f0); //读取文件内容

fr.onload = function(){ //读取完成事件

console.log('客户端文件读取完成');

var img = new Image();

img.src = fr.result;

target.appendChild(img);

}

}

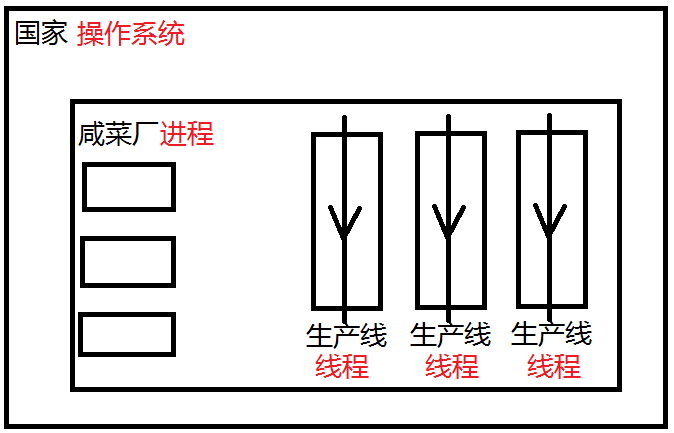
|  |
| --- |
| HTML5新增加的几个文件操作相关对象：  File：表示一个文件  FileList：表示一个文件集合  FileReader：文件读取对象，用于读取文件中的内容  FileWriter：文件写出对象，用于向文件中写出内容 |

3.HTML5新特性之七 —— Web Worker

操作系统：用于管理磁盘和内存，分配CPU时间片，实现程序的并发执行（宏观上是“同时执行”，微观上是“一个一个执行”）。

进程Process：可执行文件被操作系统从硬盘中调入到内存，获得执行，称为进程/任务；进程是操作系统分配内存的基本单位。

线程Thread：线程负责执行代码语句，线程是进程内部的执行代码的基本单位



【面试题】1：进程和线程的关系？

进程是分配内存的基本单位；线程是执行代码的基本单位。线程必须处于某个进程中，一个进程内可以同时存在1到多个线程。

【面试题】2：Chrome中的线程模型是怎么的？

每个Chrome窗口都是一个进程，最多可以使用6个线程发起HTTP请求并接收响应消息。但解析HTML、执行JS、CSS、绘图操作、事件监听由唯一的UI主线程执行。

4.如何执行很耗时的JS任务

方式1：在页面中创建<script src="耗时.js"></script>，

推荐放在body的最后——UI主线程只有一个。

方式2：HTML5中，可以创建一个新的线程来执行耗时的JS任务

new Worker('耗时.js');

由主UI线程创建一个新的“工作线程”，为其分配耗时的JS任务，UI主线程可以继续渲染界面。

|  |
| --- |
| 可以使用Worker加载任意的JS文件吗？ —— 一般都不行  界面渲染任务只能由UI主线程执行，所以任何Worker线程都不能操作DOM树，即window/document/dom对象不能在Worker执行的js中使用。 |

UI主线程给Worker线程传递消息数据：

UI主线程发送消息：

var w1 = new Worker('耗时.js');

w1.postMessage('字符串消息'); //传递消息

Worker接收消息：

onmessage = function(event){

//event.data //主线程发过来的消息

}

Worker线程给UI主线程传递消息数据：

Worker线程发送消息：

postMessage('字符串消息');

UI主线程接收消息：

var w1 = new Work('耗时.js');

w1.onmessage = function(event){

//event.data //对方发过来的数据

}

练习：用户在输入框中输入一个整数，点击“质数判定”按钮，过了一段时间之后，下方P元素中显示一个提示文本：您输入的数字是质数吗：true / false;

5.HTML新特性之八 —— Web Storage —— 重点

服务器端存储：

服务器端文件、内存、数据库。可以存储新闻、留言、商品、订单、评论、进入库记录、.....

客户端存储：

应用：浏览历史记录、内容定制、样式定制、X之内不再登录....

客户端存储实现方法：

Cookie：浏览器兼容性最好，最长只能保存4KB数据

Flash：依赖于Flash播放器

HTML5 Web Storage：HTML5新特性，最长度不能超过8MB

IndexedDB：保存在客户端的“索引数据库”

HTML5 Web Storage分为两个对象：

window.sessionStorage：会话级存储

键值对形式的字符串，存储在浏览器当前进程的内存空间中，一个页面中保存的数据可以供后续的所有页面使用；一旦浏览器窗口关闭、操作系统重启，sessionStorage中的数据都会消失；其它的浏览器进程不能访问当前浏览器进程中的sesionStorage数据。

sessionStorage.setItem(key, value); //保存一个数据

var v = sessionStorage.getItem(key); //读取一个数据

sessionStorage.removeItem(key); //删除一个数据

sessionStorage.clear() //清除当前内存中所有的数据

sessionStorage.length //数据的个数

sessionStorage.key(index) //获取第index个key

练习：login.html中，实现用户登录，提示“登录成功”；

可以使用js或者超链接跳转到其他页面，都能够显示出“欢迎回来：xxx”。 若客户端直接访问其他页面而没有登录，则跳转到登录提示页面(need\_login.html)。

总结：sessionStorage的用途——在某个浏览器窗口先后打开的多个页面间传递数据。

window.localStorage：本地存储，跨会话级存储

键值对形式的字符串，存储在本地的文件系统(磁盘的文件)中，这些数据在浏览器关闭后，甚至操作系统重启后，仍然可以继续使用。

localStorage.setItem(key, value); //保存一个数据

var v = localStorage.getItem(key); //读取一个数据

localStorage.removeItem(key); //删除一个数据

localStorage.clear() //清除当前内存中所有的数据

localStorage.length //数据的个数

localStorage.key(index) //获取第index个key

练习：在style\_list.html中让用户选择自己的网站风格，如：青青草原、芭比公主、地下重金属等，根据用户的选择，修改body的背景、文字颜色。跳转到下一个页面，仍然采用上一页面选择的风格，即使重启浏览器仍然可以看到之前选择的风格。

如何监视localStorage中存储的数据发生了改变：

window.onstorage = function(){

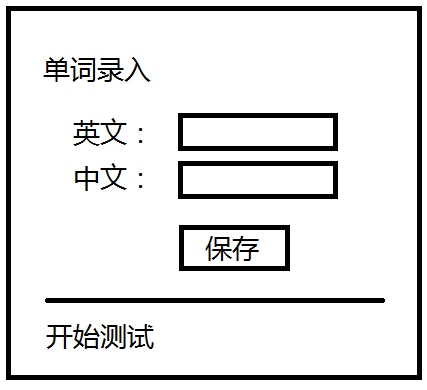
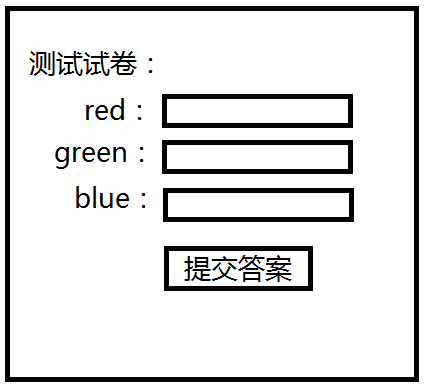
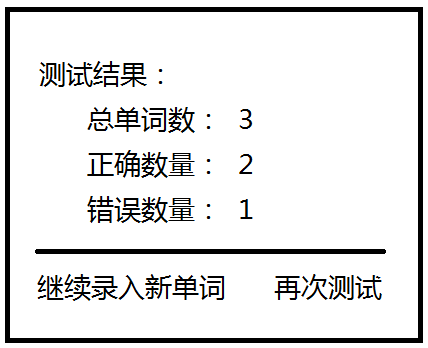
//localStorage对象中存储的数据发生改变了....

}

课后练习： 单词测试系统

用户可以在save.html中不停的录入新单词；

进入test.html开始测试，需要对之前录入过的所有单词进行测试；

提交答案后，在result.html中显示出测试成绩。

**HTML5-day07：**

复习：

HTML5新特性：

表单新特性 —— 次重点

视频和音频 —— 次重点

Canvas绘图 —— 重点

SVG绘图 —— 重点

地理定位

拖放API

Web Worker

Web Storage —— 次重点

WebSocket

Web Worker：

作用：用于创建一个新的线程，在UI主线程之外执行很耗时的JS任务

主UI线程：

var w1 = new Worker('x.js')

w1.postMessage('字符串消息')

w1.onmessage = function(e){

//e.data

}

Worker线程：

onmessage = function(e){

//e.data

//开始计算...

postMessage('字符串消息');

}

Web Storage：

作用：在客户端存储当前客户端的专有数据

sessionStorage：

把数据保存在当前浏览器窗口对应的进程的内存中，在当前窗口先后打开的多个页面中共享数据。

localStorage：

把数据保存在磁盘文件中，可以永久存储

localStorage.setItem(k,v)

localStorage.getItem(k)

localStorage.removeItem(k)

localStorage.clear()

localStorage.length

localStorage.key(index)

今日目标：

1)WebSocket —— 了解

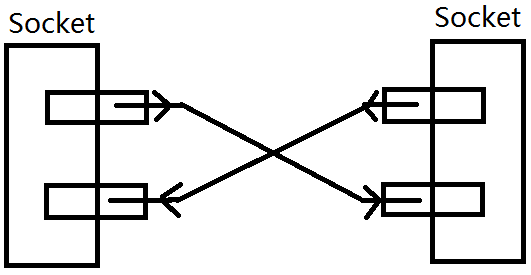
2)HTML5阶段项目

1.WebSocket

HTTP协议：基于“请求-响应模型”的协议，没有请求，就没有响应；一个请求，只会对应一个响应。用于呈现网页，没有问题；但是有些应用中，显得力不从心： 实时应用、聊天室....

“心跳请求”——使用 “定时器+XHR”——解决效果并不好。

WebSocket协议：基于“广播模型”的协议，只要客户端连接到服务器，就一直保持连接/永久连接，一方不需要发消息，对方可以主动的给出消息；一方发一个消息，对方可以给出多个应答消息。

Socket：套接字，插座

浏览器作为客户端使用WebSocket协议进行通信步骤：

var ws = new WebSocket('ws://127.0.0.1:9999'); //创建连接

ws.onopen = function(){ //连接成功

ws.onmessage = function(e){ //接收消息

//e.data

}

ws.send(); //发送消息

}

注意：WebSocket服务器端编写比较麻烦！可以使用PHP、Java、C#、NodeJS等编写。

总结：HTML5 WebSocket用于创建使用WS协议的客户端应用，连接WS协议的服务器端应用。

3.HTML5阶段项目——“京东用户中心”

知识点： jQuery + AJAX + Canvas + 定时器

功能点：

异步登录验证

菜单切换

我的订单(分页)

消费统计(Canvas原生版)

消费统计(FusionChart版)

幸运抽奖(Canvas小游戏)

4.复习：AJAX

原生AJAX：

var xhr = new XMLHttpRequest();

xhr.onreadystatechange = function(){}

xhr.open('GET','x.php', true);

xhr.send(null);

jQuery封装的AJAX：

$(...).load('x.php')

$.get('x.php', {}, succFn)

$.post('x.php', {}, succFn)

$.getScript('x.php')

$.getJSON('x.php', {}, succFn)

$.ajax({

type: 'GET',

url: 'x.php',

data: {},

beforeSend: fn,

success: fn,

error: fn,

complete: fn

})

5.复习：Canvas绘图

<canvas id="c1" width="" height=""></canvas>

var ctx = c1.getContext('2d');

ctx.fillStyle = 颜色/渐变;

ctx.strokeStyle = 颜色/渐变;

//绘制矩形

ctx.fillRect(x, y, w, h)

ctx.strokeRect(x,y,w,h)

ctx.clearRect(x,y,w,h)

//绘制文本

ctx.textBaseline = 'bottom'

ctx.font = '10px SimHei'

ctx.measureText(txt).width

ctx.fillText(txt, x, y)

ctx.strokeText(txt,x,y)

//绘制路径

ctx.beginPath()

ctx.moveTo(x,y) ctx.lineTo(x,y)

ctx.arc()

ctx.ellipse()

ctx.bezierCurveTo()

ctx.fill() / ctx.stroke() / ctx.clip()

//绘图图像

var img = new Image();

img.srs = 'x.png';

img.onload = function(){

ctx.drawImage(img, x, y)

ctx.drawImage(img, x, y, w, h)

}

提示：项目中绘制统计表使用原生代码很麻烦！

推荐使用第三方的工具：

(1)Chart.js （免费）

(2)FusionCharts （收费）

5.FusionCharts的使用

官方：http://www.fusioncharts.com/

手册：http://www.fusioncharts.com/dev/getting-started/installation.html

使用步骤：

Installing FusionCharts Suite XT for your application

Converting your data to a JSON or XML format. FusionCharts Suite XT accepts both data formats, and can read it as a string, or from a file, local or remote

Including the FusionCharts Suite XT JavaScript library in your HTML page

Creating a container <div> for the chart

Using the new FusionCharts() constructor to create the chart instance, and then calling the render()method

new FusionCharts({}).render();

练习内容： 京东用户中心项目——“消费统计”

Canvas原生版

SVG原生版

FusionCharts版

**HTML5-day08：**

复习：

见思维导图

1.HTML5阶段项目——京东用户中心

功能点5：幸运抽奖

按订单总金额给出抽奖次数——每￥100有一次抽奖次数；

抽奖后，把摇奖结果保存到数据库中，剩余抽奖次数-1；

分析：

(1)计算指定用户的订单总金额：

SELECT SUM(金额) FROM 订单 WHERE 用户名='XX';

(2)查询已抽奖次数

SELECT COUNT(抽奖记录ID) FROM 抽奖表 WHERE ...;

(3)异步请求当前登录用户剩余抽奖次数，大于>0则，“开始抽奖”按钮可以点击

(4)使用Canvas绘图让圆盘旋转

(5)异步提交抽奖结果给服务器，保存到数据库；同时剩余抽奖次数-1

难点：(1)两个PHP页面的编写 (2)圆盘旋转，控制转速

2.控制圆盘的转速

var deg = 0;

var timer = setInterval(function(){

deg += 5; 旋转的角度应该是一个变量——随着时间推进而改变

ctx.rotate(deg\*Math.PI/180);

....

}, 42)

0 = a\*0\*0 + b\*0 + c => c: 0

15 = a\*duration/2\*duration/2 + b\*duration/2

0 = a\*duration\*duration + b\*duration

y = ax^2 + bx

当x:0 y: 0

当x:duration/2 y:15

当x:duration y:0

-a\*duration = b

60 = -a\*duration\*duration

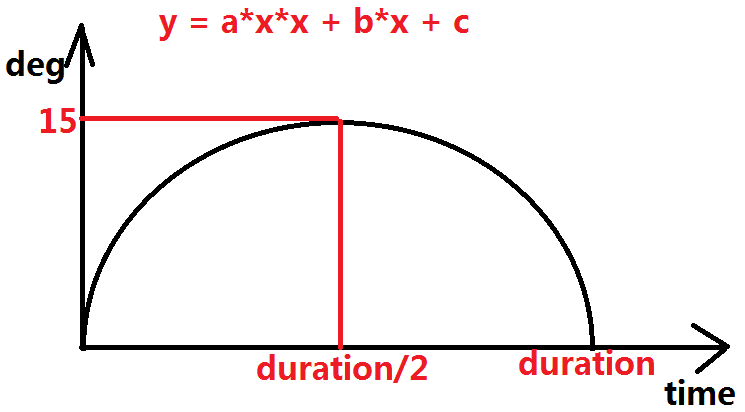
a = - 60/(duration\*duration)

b = 60/duration

//旋转速度函数：根据总持续时间和当前已旋转的时间，获取对应的旋转速度

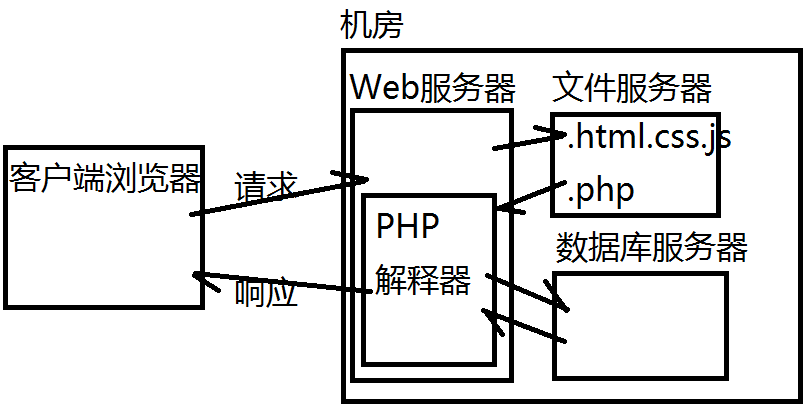
function speedFn(duration, last){

var deg = 0;

 deg = -60\*last\*last/(duration\*duration)+60\*last/duration;

return deg;

}



3.云服务器的使用

亚马逊云

阿里云

百度云

腾讯云

主机屋

新浪云，“免费”的云服务器+免费的二级域名

新浪云服务器的使用步骤：

注册新浪微博账号

注册地址： http://weibo.com

使用微博账号开通新浪云账号

开通地址： http://sae.sina.com.cn ——没有云豆

邀请地址： http://t.cn/R4WcToI ——赠送云豆

注意：安全邮箱和安全密码——用户上传网站代码！！

创建一个Web应用

登录地址： http://sae.sina.com.cn

注意：Web服务器选择PHP5.6！！

上传Web应用中的页面代码

登录后，点击应用的名称，开始管理指定应用

点击“应用”=>“代码管理”，可以上传/删除代码

第一次使用需要选择“代码仓库”的类型，可以选择SVN

临时编辑代码：

批量上传代码：

使用MySQL服务器

“数据库与缓存服务”=> “共享型MySQL”

注意：新浪云中的MySQL不允许删除数据库、创建新的数据库，每个Web应用都对应唯一的一个数据库。

注意：PHP中 mysqli\_connect( ) 的参数都必须进行修改！！！

**SECTION IVAngularJS-day01:**

1、框架

优秀的可以重复使用的代码，单独提取封装好，完成复杂的功能，提高代码质量，保证良好的浏览器兼容。

学习框架，理解框架、动手查阅手册、快速编码。

2、学习安排

3天理论知识学习 + 2天的综合项目。

3、设计原则和模式

七大设计原则：

①YAGNI (不写不需要的代码)

You Aren't Gonna Need It

②KISS（代码越简单越好）

（Keep it simple and stupid）

③DRY(不要重复ctr+c ctr+v，封装和重用)

（Don't Repeat Yourself）

④High Cohesion，Low Coupling

（高内聚、低耦合）

内聚：一个组件内的不同部分的关系

耦合：不同组件的关系

⑤SRP 单一责任原则

Single Responsibility Principle

⑥OCP（Open Closed Principle）

对扩展开放、对于修改关闭

⑦LCP 迪米特法则/最少知识法则

封装的功能单位要小，只与自己相关。

设计模式（23种）：

①创建型模式，共五种：工厂方法模式抽象工厂模式、单例模式、建造者模式、原型模式。

②结构型模式，共七种：适配器模式、装饰器模式、代理模式、外观模式、桥接模式、组合模式、享元模式。

③行为型模式，共十一种：策略模式、模板方法模式、观察者模式、迭代子模式、责任链模式、命令模式、备忘录模式、状态模式、访问者模式、中介者模式、解释器模式

MVC

为什么需要MVC？

①代码规模越来越大，切分职责是大势所趋

②为了复用代码，很多逻辑都是一样的，封装

③为了方便后期的维护，修改一块功能不影响其他功能

什么是MVC？

M：Model 模型数据 业务数据

V：View 视图 业务数据的表现

C：Controller 控制器 负责控制器数据的增删改查

当用户触发事件的时候，view层会发送指令给controlller，接着控制器去通知model层更新数据，直接显示在view层，这就是MVC的工作原理。

https://www.wikipedia.org/

angularjs.org

3、AngularJS

①什么是AngularJS？

诞生在09年，被google收购的一个js框架，使用它来构建以数据操作为主的SPA（Single Page Application）应用

②四大特征

使用MVC模式

双向数据绑定

依赖注入

模块化设计

③使用ng的表达式、指令

AngularJS中如果执行表达式（由数据变量和运算符）

语法：{{表达式}}

练习：

ng-init初始化一个对象数组，3个对象

，对象中包含name、score、age

输出到table中。

①对象数组

[{},{},{}]

②table

thead tbody

**AngularJS-day02:**

1、指令

是由ng提供的，可以用在HTML中，扩展属性来使用、扩展标签.

ngApp指令：用来指定一个AngularJS应用的范围，同时也是ng应用自动载入并启动的。

ngInit指令:声明一些变量

ngBind指令：将指定的表达式的值输出到当前元素的innerHTML

<ANY ng-bind="表达式"></ANY>

代替{{}}语法，可以防止闪烁的问题

ngRepeat 为HTML添加循环功能

ngIf 为HTML添加选择/判断功能

语法： <ANY　ng-if='表达式'> 123</ANY>

用途：当表达式是true，当前元素会挂到DOM树，否则会被删掉

练习：对象数组，有3个对象，每个对象有name、score、age，

要求使用ngRepeat循环以及ngIf判断来实现一个table：

仅显示成绩大于80分的数据

分析:

ng-init初始化数据

在tr通过ng-repeat遍历对象数组 判断，

stuList

<table>

<thead/>

<tbody>

<tr ng-repeat="tmp in stuList">

<td>

tmp.name

</td>

</tr>

</tbody>

</table>

2、MVC

面试题：

ng的四大特征：

采用MVC模式

双向数据绑定

依赖注入

模块化设计

步骤：

①声明自定义的模块

var app = angular.module('模块名',[依赖列表]);

②注册模块

ng-app=‘模块名’

③声明控制器

app.controller('控制器名',function(){})

④使用控制器

ng-controller='控制器名'

⑤数据操作

$scope

ng-init可以用来完成一些数据的初始化，通过控制器中$scope对象完成初始化

建议：$scope

实现菜品列表，点击按钮，加载更多

$scope.dishList=[]

$scope.add = function(){

$scope.dishList.push()}

ng-repeat

ng-click、ng-src

其他的一些指令：

ng-app

ng-init

ng-controller

ng-bind

ng-if

ng-repeat

ng-click

ng-src

ng-show/ng-hide

ng-checked

ngStyle ngClass

3、双向数据绑定

方向1：模型数据到视图的绑定。

一旦绑定之后，模型数据的修改，将自动更新view

实现方法：{{model}}+常用的指令

功能：

对象user={name:'Lucy',age:39};

放置2个按钮，

第一个按钮点击修改模型数据{name:'jerry',age:20},

第二个按钮点击恢复数据user={name:'Lucy',age:39}

p用来显示名称 p显示年龄

方向2：view绑定model

实现方法：ng-model指令

$scope.$watch('value',function(){...})

留言：textarea 城市：select 性别：单选框

**AularJS-day03:**

1、过滤器

过滤器用在表达式，实现对表达式结果的筛选、过滤、格式化，达到更好表现效果。

用法：{{表达式 | filter1 | filter2:param}}

| 是管道，表示传递数据的管道。

currency：实现货币的格式化

{{price | currency:'￥'}}

date 实现日期的格式化

{{newdate | date:'yyyy-MM-dd'}}

lowercase/uppercase 实现大小写的转换

orderBy 实现排序，指定增序还是降序

{{tmp | orderBy:'名称':true}}

limitTo 限制显示的个数。

练习：对象数组，5个对象：age name score

按照age降序、

按照name增序、

按照score降序只显示前三名

分析:

1、定义对象数组

2、显示结合orderBy以及limitTo

2、ng所提供的方法

angular.forEach(set,function(value,key){})

angular.uppercase(name);

angular.lowercase(name);

angular.toJson(jsonObj);将传入的json对象序列化为一个json字符串

angular.fromJson(jsonStr),将json字符串反序列化为一个json对象

3、服务

服务就是一个单例对象。

功能：提供数据、方法、对象。

划分：内置服务和自定义服务

（$http/$location/$window/$scope/$rootScope/$interval/$timeout）

在使用服务的时候，需要将要用到的服务注入进来。

app.controller('myCtrl',function($scope,$location){})

$rootScope以及$scope之间的关系

具体请看图片。

$http 跟服务器交互

$http.get('url').success(function(data){})

实现：通过$http 从php文件中读取数据

分析：

apache服务跑起来

在xampp下的htdocs目录中：php文件，html文件

访问通过在地址栏localhost访问

双向数据绑定的原理：

1、{{num}}

ng就在$scope给num设置一个watcher

$scope.$watch('num',function(){})

2、watcher的回调什么时候会执行？？？

ng会周期性的运行一个函数，$digest循环。

会触发各个watcher，ng的scope作对比判断值是否发生变化

3、什么时候执行$digest循环？？

ng指令都会触发循环；

$scope.$digest

$scope.$apply() ==>$rootScope.$digest

$timeout

$interval 每隔2s，num++

4、依赖注入

文件压缩：

会删除所有的注释，并删除空白字符，

尽可能的简化变量名、方法名、形参名（混淆），所有的数据、关键字都不会做任何改变。

压缩方式1：

java.exe -jar yuicompressor-2.3.jar test.js > test.min.js

压缩方式2：

在webStorm中，点击左上角的file，找到settings，选中tools中的fileWatcher中，点击+号，配置program为下载的yui-compressor.jar

依赖注入：

什么是依赖？

function driver(car){

car.start();

car.stop();

}

app.controller('myCtrl',function($scope){

})

依赖注入的三种方式：

推断式依赖注入

标记式依赖注入

行内式依赖注入

**AularJS-day04:**

1、依赖注入

根据依赖注入服务，标记的方式的不同，划分：

推断/猜测、标记、内联式/行内

①推断式

实现点击按钮，弹窗Hello Injector

如何自定义服务？

app.factory('$show', function ($window) {

return{

name:'custom',

show: function (text) {

$window.alert(text);

}

}

})

②标记式依赖注入

$inject 设置一个字符数组

//设置回调函数的$inject 注入标记（字符串数组）

ctrFunc.$inject = ["$scope","$show","$print"]

自定义两个服务，一个服务弹窗显示HelloAngular，另一个服务用来通过控制台打印Hello Injector

③行内式（内联式）依赖注入

在构建一个ng应用的时候，允许将一个字符串数组 作为对象的参数。

注意：在这个数组中，最后一个必须是函数体

练习：

针对行内式依赖注入demo，把js代码单独抽成一个js文件，script引入这个文件经过压缩后的文件，去验证压缩后，是否可以使用ng!!

验证，行内式依赖注入可以解决js文件被压缩后，服务无法被实例化以及使用的问题。

推断式：尽量不要这样写

标记式

行内式：建议

$injector

angular在依赖注入时，注入器$injector,快速的定位到应用中需要注入的各种服务，定位和创建，get和has方法。

//通过注入器 判断是否存在$custom服务

var injector = angular.injector(['ng','myApp']);

//判断参数中的服务是否存在

if(injector.has('$custom'))

{

console.log('服务存在');

//获取参数中的实例化服务对象

var custom = injector.get('$custom');

custom.print();

}

练习：在案例基础上，按钮，点击按钮调用$custom中的方法。（使用注入器的get和has）

分析：可以注入$injector 代替angular.injector方法

angularjs

采用MVC的设计模式

双向数据绑定

依赖注入

模块化设计

2、模块化

ng有两种：

第1种 ng自身封装好的模块

第2种：自定义的模块

自定义的模块之间如何通讯？

angular.module('myApp02',['ng','myApp01']);

面试题：ng模块都包含哪些组件？？

①controller：用于维护模型数据

②directive：数据相关

③service：在不同的控制器之间提供某种函数服务

④filter：刷选、格式化、排序

⑤function：

。。

如何自定义一个指令？？

var app = angular.module(...);

app.directive(name,fn)

新指令的命名必须使用驼峰式，由两部分构成，前部分是前缀，使用项目名/公司名;后半部分是指 指令的作用，用于说明他的功能。

app.directive('tsHello',..);

<ts-hello></ts-hello>

restrict:'EACM'

E===>Element

A==>Attribute

C==>Class

M==>Comment

service:

$http/$timeout/$interval/$scope/$rootScope/$location/$window...

app.factory(name,fn);

app.service(name,fn);

app.constant/value

使用factory创建2个服务，

$outfun(方法) $outarr（返回数组）

3、学习使用路由模块

多页应用和单页应用

SPA（SinglePageApplication）的基本工作原理？？

①url

http://127.0.0.1/index.html#/路由地址

②浏览器解析index.html,解析路由地址

③查找路由词典，找到对应的代码片段

#/start ==> tpl/start.html

#/main ==> tpl/main.html

⑤获取当前url中路由地址所对应的真是模板页面的地址

⑥发起异步ajax请求，获取模板页面，插入DOM数，实现刷新。

实现SPA的步骤：

①创建一个完整的页面 index.html

(引入angular.js angular-route.js)

②创建自定义模块，声明依赖于ngRoute和ng

③在index.html的body使用ngView，盛放代码片段

④创建代码片段

⑤使用ngRoute提供的对象配置路由词典

app.config(function($routeProvider){

$routeProvider.when('/路由地址'，{templateUrl：‘模板页面URL’})

})

⑥测试SPA

模板之间的跳转：

①直接修改地址栏中的路由地址

.../index.html#/路由地址

②超链接

<a href='#/路由地址'></a>

③使用js

$location.path('/路由地址');

每日一练：

index.html

代码片段：

login.html button=>main a=>register

register.html a=>login

main.html button=>login

**JqueryMobile-day01:**

jQueryMobile-01：

前言：data属性:

data-role="page" 基础页面

data-role="header" 页头

class="ui-content" 正文

data-role="footer" 页尾

data-position='fixed'

data-transition='slide/pop/slideup...' 设置过场动画

data-direction='reverse' 设置相反的过场动画

data-rel='back' 实现原路返回

data-corners='false' 取消圆角

data-shadow='false' 取消阴影

data-inline='true' 在一行内显示

data-theme='a/b' 设置主题效果

data-mini='true' 调整迷你按钮的大小

data-icon='user/bullets/home' 设置元素的图标

data-iconpos='right/top/bottom' 调整图标相对文本的位置

data-position='right' 调整面板打开的方向

data-display=‘reveal/push/overlay’ 调整面板打开的模式（揭开、推开、覆盖）

data-dismissible='false' 点击panel外侧就不会再关掉面板

data-rel='close' 点击链接 会关闭面板

一、认识jQueryMobile

JQM是一个为触控优化的，基于H5的框架，设计网站和应用，同时支持在线...

二、开始学习JQM

1、5分钟时间快速过一下introduction

准备工作：

1、JQM的page，引入jQuery、jQueryMobile(css/js/images)

2、<meta name="viewport" content="width=device-width,initial-scale=1,user-scalable=no">

3、page:header、content、footer

data-role="page"

data-role="header"

class="ui-content"

data-role="footer"

2、第一个JQM页面

3、在一个HTML页面，实现多个page

方法1：

<a href="#pageID">单页面跳转到pageID</a>

<a href="demo03.html">跨页面跳转到...</a>

注意点：page的ID是不允许重复的！！！

方法2：

$.mobile.changePage('demo2.html',{transition:'flip/turn/slide/none'});

过场动画的设置：

<a href="demo03.html" data-transiton='slide'>jump</a>

data-direction='reverse'

取消过场动画：

data-transition='none'

通过js的方式指定跳转时的过场动画：

$.mobile.changPage('demo03.html',{transition:'flip'})

原路返回：

<a href="" data-rel='back'></a>

4、按钮

1、三种实现方式

<button>button按钮</button>

<a href="#" data-role="button">a按钮</a>

<input type="button" value="input按钮"/>

使用场景：a 按钮 用在导航/跳转，input、button 用在表单中

data-corners="false" 取消圆角

data-shadow="false" 取消阴影

data-inline="true" 在一行显示

data-theme="a/b" 设置主题效果

data-mini="true" 设置迷你按钮

data-icon="user/mail/lock/home/..."

data-iconpos="right/top/bottom" 调整图标相对于文本的位置

data-iconpos="notext" 设置无文本按钮

练习：

在同一行显示5个按钮：

后退按钮（右边）、前进按钮（左边）、

下载按钮（上边）、删除按钮（只有图标）、

上传按钮（下边）

5、grid栅格布局

行和列：

ui-grid-solo (一行只能放一列) ui-block-a (100%)

ui-grid-a (一行放两列) ui-block-a/ui-block-b (50%)

ui-grid-b (一行放三列) ui-block-a/ui-block-b/ui-block-c (33%)

ui-grid-c (一行放四列) ui-block-a/ui-block-b/ui-block-c/ui-block-d (25%)

... ...

注意点：

ui-block-a 都会重新起一行

6、实现一个面板：

1、<div id="panel-1" data-role="panel"></div>

2、面板应该放在page中header之前或footer之后

3、在<div class="ui-content"></div>中设置一个a标签

<a href="#panel-1">打开面板</a>

设置面板打开的方向

data-position="right"

设置面板打开的模式

data-display="overlay/reveal/push"

设置面板是否会在点击面板外侧时被关闭

data-dismissible="false"

当设置了data-dismissible="false"时，设置一个链接用于关闭面板

<a href="" data-rel="close">

<button data-icon="delete" data-iconpos="notext" data-inline="true"></button>

</a>

**JqueryMobile-day02:**

jQueryMobile-day02：

前言：

data-fullscreen="true" 实现全屏效果

data-position="fixed" 固定header或footer

data-role="navbar" 导航条(使用无序列表)

class=ui-bar/ui-body ui-btn-left/ui-btn-right ui-btn-active

data-role='listview' 添加jQM中列表的样式

data-inset='true' 添加嵌入效果（有圆角）

data-filter='true' 支持搜索

data-filter-placeholder="" 修改默认的搜索提示

data-autodividers='true' 自定添加分隔栏

data-count-theme='a/b' 设置计数气泡的主题。

data-role='collapsible'

data-collapsed='false' 调整默认的折叠状态

data-collapsed-icon

data-expaned-icon

data-role='collapsibleset' 折叠组件集合(手风琴组件)

data-role='table' 指定为jQM中的table

data-mode='reflow/columntoggle' 设置表格模式

data-priority='1~6' 设置优先级

data-column-btn-text='str'; 设置文本

1、body bar

分块显示内容：

class="ui-bar"

class="ui-body"

2、toolbar 工具条

全屏效果

data-full-screen="true"

data-position="fixed"

将上述两个属性同时指定给header和footer，就能实现全屏效果

在header中添加按钮，只有一个button，如果想在右边，class='ui-btn-right'

两个按钮，默认一个在左边一个在右边。

3、navbar 导航条

无序列表外部添加一个容器 指定data-role='navbar'

练习：

在footer中，放置一个navbar ，主页、个人中心、设置（图标在文本的上方）。

当导航条中的按钮 添加图标时，图标会默认在文本的上方。

4、listview 列表

ul/ol data-role="listview"

ul/ol data-inset="true" 添加一个嵌入的效果

ul/ol data-filter="true" JQM会给列表添加一个搜索框，提示支持搜索行为

ul/ol data-filter-placeholder="请输入..." 自定义提示内容

li data-role="list-divider" 手动添加分隔栏

ul/ol data-autodividers="true" 自动添加分隔栏

计数气泡：

<ul data-role="listview" data-count-theme="b" data-inset="true" data-filter="true" data-autodividers="true" data-divider-theme="b">

<li>

<a>Input

<span class='ui-li-count'>2</span>

</a>

</li>

</ul>

修改气泡的主题：在ul/ol中添加属性：data-count-theme='a/b/c...'

练习：使用jQM中的listview来实现当前的页面效果

在li中 将img作为第一个元素就可以了。

5、collapsible 折叠组件

<div data-role='collapsible'>

文本

<ANY />

</div>

修改默认的折叠状态：

data-collapsed='false/true' 默认展开/折叠

修改折叠或展开的图标

data-collapsed-icon='gear/...'

data-expanded-icon='bullets/...'

修改图标的位置

data-iconpos='left/right'

data-role='collapsibleset' 折叠组件集合（手风琴组件）

6、table 表格

表格的使用过程中需要设置以下属性：

data-role='table'

必须设置thead tbody

class='ui-responsive'

data-mode='reflow/columntoggle' table为回流式表格/列切换式表格

data-priority="1/2/3/4/5/6" 1是优先级最高的，6是优先级最低的。

没有设置过data-priority的列头是不允许被移除或者隐藏掉。

1 --> 6 优先级越来越低（越来越容易被隐藏）

修改列切换式表格 按钮的文本：data-column-btn-text 设置

例：

1、回流式表格

<table data-role="table" class="ui-responsive">

...

</table>

2、列切换式表格

<table data-role="table"data-mode="columntoggle" data-column-btn-text="请选择要显示或隐藏的列" class="ui-responsive" >

<thead>

<tr>

<th>I'm critical and can't be removed</th>

<th data-priority="1">I'm very important</th>

<th data-priority="3">I'm somewhat</th>

<th data-priority="5">I'm less important</th>

</tr>

</thead>

<tbody>

...

</tbody>

</table>

7、表单

input type=checkbox data-role='flipswitch'

(data-on/off-text)

8、事件

bind('tap',function(){})

每日一练：

练习：

左滑动，右边滑出一个panel

右滑动，左边滑出一个panel

$('#panel-1').panel（'open'）;

【课上典型例题】：

12-表单组件综合练习.html

<div class="ui-content">

<!--TODO 父容器-->

<div class="ui-field-contain">

<label for="name">全名：</label>

<input class="form-control" type="text" id="name" name="name" placeholder="你的姓名是？"><br><br><br>

<label for="search">需要查找什么？</label>

<input type="search" id="search" name="search" placeholder="搜索内容"><br><br><br>

<label for="date">今天的日期：</label>

<input type="date" id="date" name="date"><br><br><br>

<label for="color">选择喜欢的颜色：</label>

<select name="colorshoose" id="colorshoose">

<option>红色</option>

<option>绿色</option>

<option>蓝色</option>

</select><br><br><br>

<label for="switch">切换开关：</label>

<input data-role="flipswitch" type="checkbox" id="switch"><br><br><br>

<div data-role="controlgroup">

<legend>选择喜欢的电影：</legend>

<label for="m1">蜘蛛侠</label>

<input type="checkbox" id="m1">

<label for="m2">变形金刚</label>

<input type="checkbox" id="m2">

<label for="m3">碟中谍</label>

<input type="checkbox" id="m3">

</div>

</div>

</div>

13-事件的绑定.html

<script>

$(document).ready(function(){

$('h3').bind('tap',function(){

alert('已被点击...');

});

$('h3').bind('taphold',function(){

alert('已被长按...');

});

$('div').bind('swipeleft',function(){

alert('左滑动...');

});

})

</script>

<div class="ui-content">

<h3>Hello jQueryMobile</h3>

</div>

**JqueryMobile-day03:**

jQueryMobile day04：

1、前端框架：

1、jQuery： js函数库，简化DOM操作

2、jQueryUI： HTML组件库，用于PC端应用

3、BootStrap： css样式库，html组件库，响应式布局(偏向于pc端)

4、AngularJS(ng)： mvc框架，数据操作为主的SPA应用。主要将数据和业务隔开

5、jQueryMobile(JQM)： HTML组件库，用于移动端的网页应用等

2、能否配对组合使用？

1、jQ + jQUI ok

2、jQ + BootStrap ok

3、jQ + ng ok

4、jQ + JQM ok

5、jQUI + Bootstrap 不建议

6、jQUI + ng ok

7、jQUI + JQM 不建议

8、Bootstrap + ng ok

9、Bootstrap + jQM 很少使用(易引起冲突)

10、ng + JQM ok

3、重构项目注意点：

1、SPA方案的选择：

SPA：ng原理：由 ngRoute + 路由词典 完成

SPA：JQM原理：html -> page id

2、导航方式：

ng： $location.path();

JQM: $.mobile.changePage();

3、动态添加数据的问题：

案例：ng应用，把model数据绑定到页面，通过按钮，动态添加元素进行数据绑定

解决方案：编译链接，进入$digest循环

$(document).on('precreatepage',function(){

//获取当前元素对应的作用域对象

var scope = $('#container').scope();

//获取注入器对象

var injector = $('#container').injector();

//调用注入器，注入$compile服务，为程序提供$compile服务

injector.invoke(function($compile){

//编译并链接DOM节点

$compile($('#container'))(scope);

scope.$digest();

});

})

4、准备工作：

1、将rar压缩包解压到xampp/htdocs中

2、将xampp的apache和sql运行起来

3、确保工程是可以执行的(localhost)

4、框架的搭建

5、JQM ==> 引入必要的文件

由于抛弃了ngRoute的方案，必须要删掉相关代码，否则会冲突

可以先注释掉之前的代码，然后再去写新代码

验证：kaifanla.html ==> main.html ==> detail.html ==> order.html ==> myOrder.html

/\* TODO 新增样式，实现垂直、水平居中且文本内容也居中 \*/

.align-center{

position: fixed;

top:50%;

left:50%;

transform: translate(-50%,-50%);

text-align: center;

}

**经典面试题：**

**Day01:**

**<1>.memory-函数记忆**

/\*

1、函数记忆

让函数记住处理过的数据

为什么：避免重复执行复杂的运算

如何使用：

1、用闭包将函数处理过的参数和处理结果保存起来

\*/

//面试题1：实现一个函数，判断n是不是质数

function isPrime1(n){

if(n<=3){

return true; //是质数

}else{

for(var i=2;i<=Math.sqrt(n);i++){

if(n%i == 0){

return false; //不是质数

}

}

return true;

}

}

console.log(isPrime1(12)); //false 不是质数

console.log(isPrime1(17)); //true 是质数

//用闭包的方式实现函数记忆

var isPrime2 = (function(){

//哈希对象负责记忆计算过的结果

var hash={};

return function(n){

if(n<3){

return true;

}else{

if(hash[n]!==undefined){

return hash[n];

}else{

for(var i=2;i<=Math.sqrt(n);i++){

if(n%i==0){

return hash[n]=false;

}

}

return hash[n]=true;

}

}

}

})();

//测试比较效率

for(var i = 0,arr = [];i<1000000;i++){

arr.push(Math.floor(Math.random()\*1000));

}

//第一个

console.time("isPrime1");

for(var i0;i<arr.length;i++){

isPrime1(arr[i]);

}

console.timeEnd("isPrime1");

//第二个

console.time('isPrime2');

for(var j=0;j<arr.length;j++){

isPrime2(arr[j]);

}

console.timeEnd("isPrime2");

**<2>.repeat-函数重复**

var arr = ["a","b","a","c","b","a"];

//1、unique1看着简单，实际上效率并不高

function unique1(arr){

for(var i=0,unique=[];i<arr.length;i++){

//如果unique中不包含当前元素值

if(unique.indexOf(arr[i])==-1){

//就将当前元素值压入unique

unique.push(arr[i]);

}

}

return unique;

}

//2、哈希hash

function unique2(arr){

for(var i=0,hash={};i<arr.length;i++){

//如果unique中不包含当前元素值

if(hash[arr[i]]===undefined){

//就将当前元素值压入unique

hash[arr[i]]=1;

}

}

//提取hash中所有key

var i = 0;

var keys=[];

for(keys[i++] in hash);

return keys;

}

//3、正则的效率最低

function unique3(arr){

var reg = /(^|,,)([^,]+)(,,\2)\*/g;

return arr.sort().join(",,").replace(reg,"$1$2").split(",,");

}

console.log(String(unique1(arr)));

console.log(String(unique2(arr)));

console.log(String(unique3(arr)));

//测试

for(var i=0,nums=[];i<100000;i++){

nums.push(Math.floor(Math.random()\*1000));

}

console.time("unique1");

unique1(nums);

console.timeEnd("unique1");

//第二个

console.time("unique2");

unique2(nums);

console.timeEnd("unique2");

//第三个

console.time("unique3");

unique3(nums);

console.timeEnd("unique3");

**<3>.chars-统计字符串中重复出现的字符个数**

var str = "helloworld";

//统计每种字符出现的次数

function calc(str){

//遍历字符串中每个字符，提示声明hash数组

for(var i= 0,hash={};i<str.length;i++) {

//如果hash中已经包含当前字符

if(hash[str[i]] !== undefined) {

//就给hash中当前字符的数量+1

}

//否则

//就向hash中添加新元素key为当前字符，值为1

}

}

console.log(calc(str));

//统计每种字符出现的次数

function getMost(str){

var most = str.split("").sort().join("").match(/([a-z])\1\*/g).sort(function(a,b){return b.length- a.length})[0];

console.log("出现次数最多的字符是："+most);

}

console.dir(getMost(str));

**<4>.sort-插入排序**

var arr=[3,1,2,5,4,6];

//插入排序

function insertSort(arr){

//i从1位置开始，遍历数组中每个元素

for(var i=1;i<arr.length;i++){

//将i位置的值存在t中

var t=arr[i];

//定义变量p，值为i-1

var p=i-1;

while(arr[p]>t&&p>=0){

//将p位置的值赋值给p+1位置

arr[p+1]=arr[p];

p--;

}

//将t放在p+1的位置

arr[p+1]=t;

}

}

//insertSort(arr);

//console.log(String(arr));

//快速排序

function quickSort(arr){

//如果arr的length<=1

if(arr.length<=1){

return arr;//直接返回arr

}

//计算中间位置:

var c=Math.floor(arr.length/2);

var left=[];

var right=[];

var center=arr.splice(c,1)[0];

for(var i=0;i<arr.length;i++){

if(arr[i]>=center){

right.push(arr[i]);

}else{

left.push(arr[i]);

}

}

return quickSort(left).concat(

center,quickSort(right));

}

arr=quickSort(arr);

console.log(String(arr));

**Day02:**

**<1>.array-数组降维**

/\* TODO 数组降维 \*/

var arr = [

[0,0,0,0],

[0,0,0,0],

[0,0,0,0],

[0,0,0,0]

];

/\* TODO 三种方法 \*/

//1、数组遍历

for(var i=0,result=[];i<arr.length;i++){

for(var j=0;j<arr[i].length;j++){

result.push(arr[i][j]);

}

}

console.log(result);

//2、concat

for(var i=0,result=[];i<arr.length;i++){

result = result.concat(arr[i]);

}

console.log(result);

//3、apply+concat

var result=Array.prototype.concat.apply([],arr);

console.log(result);

/\*三种方法均输出：

Array[16]

0: 0

1: 0

2: 0

3: 0

4: 0

5: 0

6: 0

7: 0

8: 0

9: 0

10: 0

11: 0

12: 0

13: 0

14: 0

15: 0

length: 16

\_\_proto\_\_: Array[0]

\* \*/

/\*即：

\* [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

\* \*/

**<2>.function-函数相关**

/\*TODO 第一题\*/

var f = function(){

var a = b = 1; //var a = b = 1;

}

f();

console.log(b); //1

console.log(a); //报错

/\*TODO 第二题\*/

var f = function(){var a = b =1;}

setTimeout(f,0);

console.log(b); //报错，因为会先执行console.log(b);

/\*TODO 第三题\*/

var a,b = 0,fn = function(){ var a = b = 2;}

fn();

console.log(a); //undefined

console.log(b); //2

/\*TODO 第四题\*/

// TODO 函数currying化

// 函数可反复调用，且每次调用时都可追加新的参数

function add(n){

return function(m){ //

n += m; //将m追加到了n上

arguments.callee.toString = function(){ return n; };

return arguments.callee; //调用函数自己

}

}

//实现以下功能

// alert(add(1)(2)(3)/\*.toString()\*/); //6

// alert(add(1)(2)(3)(4)/\*.toString()\*/); //10

/\*TODO 第五题 递归\*/

var emp = {

work:function(){

var sum = 0;

for(var i = 0;i<arguments.length && arguments[0]>0;i++){

sum += arguments[i]+arguments.callee(--arguments[i]);

}

return sum;

}

}

console.log(emp.work(3,2,1)); //10

**<3>.closure-闭包**

/\*TODO 第一题 闭包\*/

function fun(n,o){

console.log(o);

return {

fun:function(m){

return fun(m,n);

}

}

}

var a = fun(0); //undefined，因为o未赋值，0付给了n

a.fun(1); //0

a.fun(2); //0

a.fun(3); //0

var b = fun(0).fun(1).fun(2).fun(3);

//undefined //0 //1 //2

var c = fun(0).fun(1);

//undefined //0

c.fun(2); //1

c.fun(3); //1

**<4>.oop-面向对象**

// TODO 第一题

window.a = 300;

function fn1(){

this.a = 100;

this.b = 200;

return function(){

alert(this.a);

}.call(arguments[0]);

}

function fn2(){

this.a = new fn1();

}

var a = new fn1().b; //300

//{a:100,b:200}.b

//a=200

var v = new fn1(fn2()); //undefined

//200 [object Object]

//fn2(); a={a:100,b:200}

/\*输出结果：

\* 300

\* 200

\* [object Object]

\* \*/

// TODO 第二题

function Human(){

var c = function(){var a = 200;};

return c;

} //如果构造函数有return，即返回了一个引用类型的对象，则不再创建新对象

var kang = new Human(); //不会再创建对象

var feng = new Human();

var x = Human();

console.log(feng==x); //false

console.log(kang==feng); //false

// TODO 第三题

var arr = [];

function fun(){

for(var i=0;i<4;i++){

var x = {};

x.no = i;

x.text = arr[i];

x.fun = function(){alert(i);};

arr.push(x);

}

}

fun();

arr[0].fun(); //4

arr[1].fun(); //4

arr[2].fun(); //4

arr[3].fun(); //4

// TODO 第四题

var setObj = function(o){

o.name = "xaioming";

o = {};

o.name = "xiaohong";

};

var p = {name:"xixi",age:32};

setObj(p);

console.log(p); //Object {name: "xaioming", age: 32}

// TODO 第五题--包装类型对象，（默认情况下用完即释放）

var str = "Hello";

str.len = 100; //相当于 new String(str).len

console.log(str.len); //undefined

// new String(str).len

str = new String("Hello");

str.len = 100;

console.log(str.len); //100

// TODO 第六题

var number = 2; //4

var obj = {

number : 4,

fn1:(function(){

this.number \*= 2;

number\*=2;

var number = 3;

return function(){

this.number\*=2;

number\*=3;

alert(number);

}

})()

};

var fn1 = obj.fn1;

alert(number); //4

fn1(); //9

obj.fn1(); //27

alert(window.number); //8

alert(obj.number);

// TODO 第七题

var x = 5,o = {

c:10,

doit:function doit(){

var x = 20;

setTimeout(function(){

alert(this.x)

},10)

}

};

alert(o.doit()); //undefined //5