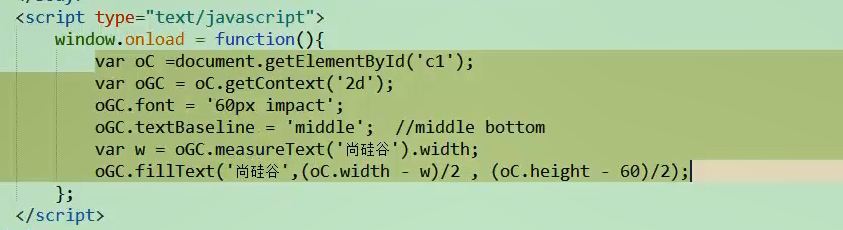
# Attr和prop

* Attr和prop的区别
  + Attr是html的预定义属性或者自定义属性
  + Prop是js的原生属性
  + 每一个预定义的attribute在编译时都会生成一个js的prop
  + 非布尔值属性
    - 不管什么情况，非布尔值属性（name,class）的property和attribute都是同步的
  + 布尔值属性
    - 改变property时，不会同步attribute
    - 在没有动过property（如html没有加attr的标签）时，attribute会同步property
    - 在动过property时，attribute不会同步property
  + 浏览器在执行的时候只认property
  + 用户操作的也是property
  + 所以如果操作布尔值属性的话，最好用jqury的prop方法操作
* Html5的小功能
* 新增classlist属性直接操作class的数量
  + Js获取classlist来直接对标签的class进行增删
  + Classlist.add
  + Classlist.remove
  + Classlist.toggle
* 新增HTMLelement.dataset.xx 来获取自定义的以data-\*开头的html属性的property
* Contenteditable=“true” 标签可以直接修改标签内内容，
  + 注意：此属性是非布尔值属性
* Html中的语义化标签
* Header（放hpgroup和h1-h6）
* Hpgroup
* Nav
* Section (放主要内容)
* Article（几乎不用）
* Aside(几乎不用)
* Footer

# Canvas

* Canvas是h5新增的元素，可以通过js中的脚本来绘制图形
* 画布默认宽高 ,定宽高不要在css上定制，要在html上
* Canvas想要绘制内容需要获取画笔（getcontext）
  + 拿画笔之前判断画笔是否存在，否则可能报错
  + Var canvas = document.queryselector(“#canvas”) 获取画布
  + If （canvas.getcontext）{var pen = canvas.getcontext(“2d”)} 获取画笔
* Canvas绘制矩形
* Canvas画矩形的三种方式：x,y参照画布的原点
  + Fillrect(x,x,width,height)
  + Strkerect(x,y,width,height)
  + Clearrect(x,y,width,height)
* Strokerect边框2像素的问题
  + Css中不支持小数点的像素，0.5的像素会判断成1像素
  + pen.strokerect(100,100,100,100) 会出现在（100,100）的位置，画1px的边框，1px的边框上部分在99.5，下部分在100.5的位置，导致像素取整为2像素
    - 解决方法：pen.fillrec(100.5,100.5,100,100)
* Canvas的样式
  + Fillstyle 填充颜色
  + Strokestyle 边框颜色
  + Linewidth 边框长度
  + Linejoin 设定线条与线条接合间样式，跟填充样式无关
    - Round 圆角
    - Bevel 斜角
    - Miter 直角
  + Canvas必须先设置样式才绘制，跟js渲染浏览器的异步不一致
* canvas绘制路径
* 绘制路径有下面属性
  + Moveto()抬起画笔，创建一个新点，如果有多个moveto（），就会多次抬起画笔画出多个图形
  + Lineto() 可以有多个创建多个点
  + Closepath() 完成结束点到起始点的绘制
  + Fill() 会自动合并路径，并填充
  + Stroke（） 会绘制边框，不会自动合并路径
  + Beginpath()
    - 每次使用moveto,lineto的时候，都会在路径容器里加上绘制过的路径，用beginpath可以清空当前容器里的路径
    - 画完一段路径时，调用beginpath时，清楚整个路径容器 ，来保持规范
  + Linecap（） ：绘制每一条线段的前后两段末端的图片
    - Round
    - Square （增加一段厚度为线段一半的矩形在线段前后）
    - Butt （默认值）
* Canvas画笔规范
* 因为存在样式栈，所以写canvas时save和restore成对出现来保持不混乱，如下面做法
  + Pen.save()
  + 中间写样式
  + Pen.beginpath()
  + 这边绘制内容
  + Pen.restore()
* 原因：每次调用样式api时，都会往样式容器里做登记，调用save来将空的样式容器压入样式栈，调用restore的时候，将样式栈里的栈顶状态（空样式）弹出到样式容器里，进行覆盖，来保持样式容器为空，方便下次样式容器储存样式
* Canvas绘制曲线
* Arc(x,y,radius,startAngle,endAngle,anticlockwise)
  + x,y为圆心坐标
  + startAngle ,endAngle为起始，结束位置的弧度，记住\*Math.pi/180
  + anticlockwise为true，表示逆时针
  + Arc(100,100,50,360\*Math.pi/180) 能画一个圆
* arcTo( x1,y1,x2,y2,radius) ： 根据给定的控制点和半径画一段圆弧
  + 配合moveTo画出一个角，然后再画出一个半径为radius的圆往角靠
* quadraticCurveTo(x1,y1,x2,y2) 根据给定的控制点画出一段圆弧，半径不固定
* 除了二次元贝赛尔曲线，还有三次元贝赛尔曲线，很难，pass。
* Canvas的变换
* 变换也属于样式，会进样式容器。
* Translate(x, y)： 改变画布的原点
  + Translate(50,50)会把画布的原点从0，0变成50，50
  + Canvas中的变换是累加的，两次调用translate就会把原点再次移位
* Rotate(angle) ：旋转画布的原始坐标轴的方向
  + 需要注意，先旋转再变换，和先变换再旋转的坐标轴原点是不一样的
  + 旋转也可以累加
* Scale(倍数，倍数)
  + 放大（缩小）的是css的像素面积，也就是说，css的像素变少，画布的可见区域会变小
* Canvas的综合实践：做一个会动的钟表
  + 每次画一个部件，就使用一次save,beginpath,restore
  + 把整个绘制内容的方法，设置为定时器内，每秒都要花一次
  + 每秒画一次后，都需要清楚之前的画布，不然会出现重叠
  + 清楚画布时，需要知道当前画布的原点是否在最左上角
* Canvas的注意点
* Canvas的渲染极快，不会出现代码覆盖后延迟渲染的问题，写canvas代码需要有同步思想
* 在获取上下文（画笔）时，一定要先判断是否存在
* 画布高宽问题
  + 画布具有默认高宽：300\*150
  + 切记一定要使用html的attr的形式来定义画布的宽高
  + 通过css形式定义会缩放画布内的图像
* 绘制矩形的问题
  + 边框宽度的问题，边框宽度是在偏移量（x,y）上下分别渲染一半，可能会出现小数边框，一旦出现小数边框，都会向上取整
  + Canvas的api只支持一种图像的直接渲染 ：矩形
* 画布api：
  + Getcontext(“2d”)
  + Width
  + Height
* 上下文api
  + fillRect(x,y,w,h)填充矩形
  + strokesRect(x,y,m,h)带边框的矩形
  + clearRect(清除整个画布)
  + fillstyle
  + stroketyle
  + linewidth
  + linecap
  + linejoin
  + moveTo(x,y) 将画笔抬起到x,y处
  + lineTo(x,y) 将画笔移到x,y
  + rect(x,y,w,h)
  + arc(x,y,r,degS,degE,direction)
  + arcTo(x1,y1,x2,y2,r)：2个坐标，一个半径
    - 接合moveTo（x,y）方法使用
    - x,y 起始点
    - x1,y1控制点
    - x2,y2结束点
  + quadraticCurveTo(x1,y1,x2,y2)：2个坐标
    - 接合moveTo（x,y）方法使用
    - x,y 起始点
    - x1,y1控制点
    - x2,y2结束点
  + bezierCurveTo(x1,y1,x2,y2,x3,y3)
    - x,y 起始点
    - x1,y1 控制点
    - x2,y2 控制点
    - x3,y3 结束点
  + fill() 填充 自动闭合
  + stroke() 手动闭合
  + beginpath()
  + closepath
  + save() 将画布的当前状态（样式相关，变换相关）压入样式栈中
  + restore() 将样式栈中栈顶的元素弹到样式容器中
  + translate(x,y) 将原点按照当前坐标中位移x,y个单位
  + rotate(弧度) 将坐标轴按顺时针方向进行旋转
  + scale(因子)
    - 放大：放大画布，画布中的一个css像素所占据的物理面积变大，画布中包含的css像素的个数变少
    - 缩小：相反吗
* Canvas使用图片
* Canvas操作图片时，需要等图片加载完了才能操作
  + DrawImage(image,x,y,width,height)
    - 其中image是image或者canvas对象，
    - x,y是image在canvas中的起始位置
    - width和height控制image插入canvas时的大小，可以不写
* canvas操作背景
  + createPattern(image,repetition)
    - 可以将此方法的返回值作为fillstyle的值来设置画布背景
* Canvas的线性渐变
  + Var gradien =createLinerGradien(x,y,width,height)
  + Gradien.addColorStop(0,”red”)
  + Gradien.addColorStop(1,”pink”)
* Canvas的径向渐变
  + Var gradien =createRadialGradien(x1,y1,r1,x2,y2,r2)
  + Gradien.addColorStop(0,”red”)
  + Gradien.addColorStop(1,”pink”)
* Canvas绘制文本
* ctx.fillText(“” , x, y)
* ctx.font =“40px sans-serif”
* ctx.strokeText(“” , x,y)
* ctx.textalign =”right” 文字最右边是坐标，跟css的不一样，
* Ctx.textBaseLine = “middle” 基线在坐标中间
* Ctx.measureText(“文本”) 返回textmetrics对象，包含文本在canvas中的宽度
* Canvas中的文字水平垂直居中：
  + 
* 像素操作
* Var imageData = ctx.getImageData(0,0,100,100);
  + imageData有如下属性：
    - Width :横向上的像素的个数
    - Height :纵向上的像素的个数
    - Data :每一个像素上的rgba的信息
* Ctx.createImageData(width,height) 创建一个空的imagedata来进行修改
* Ctx.putImageData(imageData,x,y) 在(x,y)的位置放入编辑完的imageData
* 给定x,y，获取位置所在像素的颜色：
  + R :Data[y\*w+x]
  + G:Data[y\*w+x+1]
  + B :Data[y\*w+x+2]
  + A :Data[y\*w+x+3]
* 给定x,y,rgba可以操作画布中某个像素像素的颜色：
  + data[y\*w+x]=color[0]
  + data[y\*w+x+1]=color[1]
  + data[y\*w+x+2]=color[2]
  + data[y\*w+x+3]=color[3]
* 实战练习：马赛克
  + 操作单像素进行操作
* 实战练习 ：刮刮卡
* Canvas导为图片
* Canvas.toDataURL();

# 音视频

* 目前还没有一种编解码器和容器的组合能适用所有的容器
* Source标签有type=video/mp4,video/ogg,video/webm三种
  + 浏览器兼容哪种就用哪种，可以同时写在一起
  + 如果上面三种都不兼容的话，还可以用a标签，让用户直接去下载
* 只有标签，不加属性时，不会占空间
* Duration属性只有在第一帧加载完成时才能出值，否则会NAN
* 音视频常用attribute属性：
* Controls 布尔值
* Src 要嵌到页面的url值
* ’autoplay: 是否自动播放
* Poster 媒体未播放时，显示的图片
* Muted : 是否静音
  + Preload :
  + None ： 默认
  + Metadata ： 认为用户不需要查看视频
  + Auto ： 用户需要这个视频优先加载
  + 空字符串： 相当于auto
* 音视频的property属性
* Duration ： 音视频的持续时间
* currentTime : 运行开始到当前的时间 （可读写）
* muted ： 是否静音 （可读写）
* volume : 音量控制 （可读写）
  + muted 和volume之间不会同步，
  + muted的优先级比,volume高
  + muted设置成true的时候，调用volume，如果volume非0，那么也不会拿到0
* paused : 是否暂停 (只读)
* ended ：是否结束 (只读)
* error ： 是否出错 (只读)
* cuurentSrc ： 当前src (只读)
* 只有video才有的属性：
  + Poster (可读写)
  + Width,heght （设置视频的宽高，可读写）
  + videoWidth,videoHeight （视频的实际尺寸，只读）
* 音视频的js函数
* Video.play()
* Video.pause()
* Video.load()
  + 在修改音视频的source的时候，进行重新加载
* 音视频的相关事件
* 视频:
  + abort 在播放被终止时触发,例如, 当播放中的视频重新开始播放时会触发这个事件。
  + canplay 在媒体数据已经有足够的数据（至少播放数帧）可供播放时触发。这个事件对应CAN\_PLAY的readyState。
  + canplaythrough 在媒体的readyState变为CAN\_PLAY\_THROUGH时触发，表明媒体可以在保持当前的下载速度的情况下不被中断地播放完毕。注意：手动设置currentTime会使得firefox触发一次canplaythrough事件，其他浏览器或许不会如此。
  + durationchange 元信息已载入或已改变，表明媒体的长度发生了改变。例如，在媒体已被加载足够的长度从而得知总长度时会触发这个事件。
  + emptied 媒体被清空（初始化）时触发。
  + ended 播放结束时触发。
  + error 在发生错误时触发。元素的error属性会包含更多信息。参阅Error handling获得详细信息。
  + loadeddata 媒体的第一帧已经加载完毕。
  + loadedmetadata 媒体的元数据已经加载完毕，现在所有的属性包含了它们应有的有效信息。
  + loadstart 在媒体开始加载时触发。
  + mozaudioavailable 当音频数据缓存并交给音频层处理时
  + pause 播放暂停时触发。
  + play 在媒体回放被暂停后再次开始时触发。即，在一次暂停事件后恢复媒体回放。
  + playing 在媒体开始播放时触发（不论是初次播放、在暂停后恢复、或是在结束后重新开始）。
  + progress 告知媒体相关部分的下载进度时周期性地触发。有关媒体当前已下载总计的信息可以在元素的buffered属性中获取到。
  + ratechange 在回放速率变化时触发。
  + seeked 在跳跃操作完成时触发。
  + seeking 在跳跃操作开始时触发。
  + stalled 在尝试获取媒体数据，但数据不可用时触发。
  + suspend 在媒体资源加载终止时触发，这可能是因为下载已完成或因为其他原因暂停。
  + timeupdate 元素的currentTime属性表示的时间已经改变。
  + volumechange 在音频音量改变时触发（既可以是volume属性改变，也可以是muted属性改变）.。
  + waiting 在一个待执行的操作（如回放）因等待另一个操作（如跳跃或下载）被延迟时触发
* 音频:
  + abort 在播放被终止时触发,例如, 当播放中的视频重新开始播放时会触发这个事件。
  + canplay 在媒体数据已经有足够的数据（至少播放数帧）可供播放时触发。这个事件对应CAN\_PLAY的readyState。
  + canplaythrough 在媒体的readyState变为CAN\_PLAY\_THROUGH时触发，表明媒体可以在保持当前的下载速度的情况下不被中断地播放完毕。注意：手动设置currentTime会使得firefox触发一次canplaythrough事件，其他浏览器或许不会如此。
  + durationchange 元信息已载入或已改变，表明媒体的长度发生了改变。例如，在媒体已被加载足够的长度从而得知总长度时会触发这个事件。
  + emptied 媒体被清空（初始化）时触发。
  + ended 播放结束时触发。
  + error 在发生错误时触发。元素的error属性会包含更多信息。参阅Error handling获得详细信息。
  + loadeddata 媒体的第一帧已经加载完毕
  + loadedmetadata 媒体的元数据已经加载完毕，现在所有的属性包含了它们应有的有效信息。
  + loadstart 在媒体开始加载时触发。
  + mozaudioavailable 当音频数据缓存并交给音频层处理时
  + pause 播放暂停时触发。
  + play 在媒体回放被暂停后再次开始时触发。即，在一次暂停事件后恢复媒体回放。
  + playing 在媒体开始播放时触发（不论是初次播放、在暂停后恢复、或是在结束后重新开始）。
  + progress 告知媒体相关部分的下载进度时周期性地触发。有关媒体当前已下载总计的信息可以在元素的buffered属性中获取到。
  + ratechange 在回放速率变化时触发。
  + seeked 在跳跃操作完成时触发。
  + seeking 在跳跃操作开始时触发。
  + stalled 在尝试获取媒体数据，但数据不可用时触发。
  + suspend 在媒体资源加载终止时触发，这可能是因为下载已完成或因为其他原因暂停。
  + timeupdate 元素的currentTime属性表示的时间已经改变。
  + volumechange 在音频音量改变时触发（既可以是volume属性改变，也可以是muted属性改变）.。
  + waiting 在一个待执行的操作（如回放）因等待另一个操作（如跳跃或下载）被延迟时触发

音视频实战

* 音阶导航
* 自定义视频布局，
  + 静止滚动条可以保证最小宽度时，浏览器窗口不能缩小
  + Overflow可以有overflow-x overflow-y
* 拖拽思路
  + 拿到鼠标点击元素时，元素一开始的位置
    - offsetLeft,offsetTop拿到元素相对父元素的x,y坐标
    - ev.clientX ev.clientY 拿到鼠标点击处的x,y坐标
  + 拿到鼠标移动的距离
    - 用mousemove事件，ev.clientX,clientY获取到移动的距离
  + 确定鼠标移动后，元素的位置
    - Move的鼠标距离加上down时的最初距离，等于元素的新距离

# Html5新增标签：

1. Meter : 显示已知范围的标量值
2. Progress ： 显示一项任务的进度条
   1. Value :
      1. 0为没有
      2. 1为全部

列表标签

1. Datalist: 包含一组option元素，这些元素表示可选值
   1. 通过Id和input绑在一块，配合input使用，可以做成下拉框
2. Details :
   1. 配合summary标签和若干个p标签一起使用，可以做成无框下拉框

标记标签

1. Mark : 用法类似span，马克文字

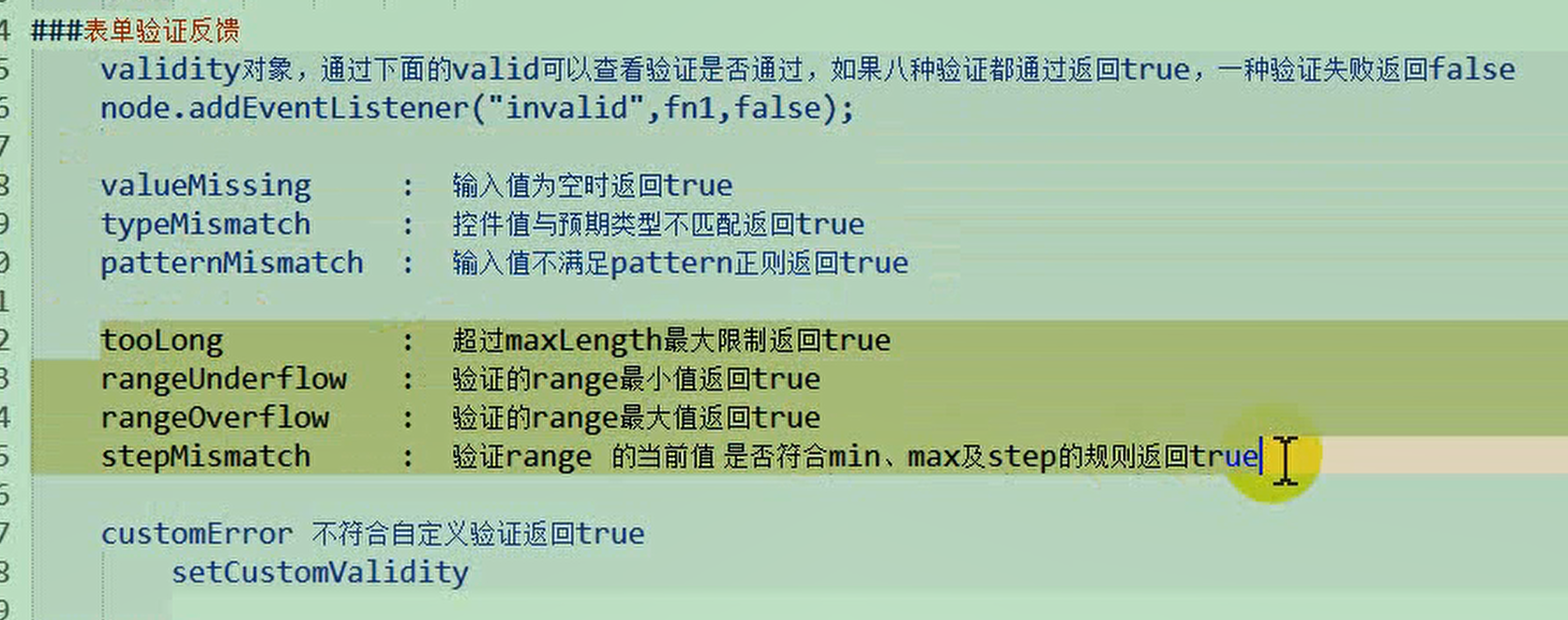
Html5表单

1. Text : 普通表单
2. Email : 邮件地址表单
3. Tel : 电话表单
4. url : 网址表单
5. search ： 有清空文本的功能
6. range : 有滑块可以调节
   1. min
   2. max ： 最大到
   3. step : 可以指定一次走几步
7. number : 只能输入数字
8. color : 可以打开颜色面板
9. datetime ：显示完整日期（移动端浏览器支持）
10. datetime-local : 显示时间，不含时区
11. time
12. date
13. week
14. month

新增表单的属性

1. placehoder :
2. required
3. autofocus
4. pattern : 可以指定表单输入正则验证
5. formation : 可以指定提交按钮到指定url

表单的验证反馈

1. invalid 事件 ： 表单验证失败时，发生的事件
2. validity对象 ：
   1. 
   2. CutomError : 用户自定义的验证
      1. 表单元素.setCutomValidity(“页面显示的警告信息”)

# Html5实战技术点

OffsetParent

* 1. 在body,html的margin和padding为0的情况下：
  2. 本身定位为fixed
     1. offsetParent 为null （浏览器不为火狐）
     2. offsetParent 为body （浏览器为火狐）
  3. 本身定位不为fixed（包括无定位static）
     1. 父级没有定位
        1. offsetParent : body
     2. 父级有定位
        1. offsetParent : 定位父级

offsetLeft 和offsetTop

* 1. 这两个是相对于父级元素的内边距的边界位置的偏移量

元素到页面的位置

1. 绝对位置 ： 到body的距离
2. 相对位置 ： 到视口的距离
3. 实战： 获许元素的绝对位置（到body的距离）
4. 实战：获取元素的相对位置
   1. 绝对位置减掉滚动条滚动的距离
5. getBoundingClentRect() : 获取一个带有元素的左上角和右下角的点相对位置和元素长宽的对象。
   1. 如果需要元素的绝对位置就加上滚动条的滚动过的距离scrollTop()
6. 元素的尺寸
   1. clientWidth,clientHeight
      1. 可视区域（padding）
   2. offsetWidth,offsetHeight
      1. 可视区域（padding） + border
   3. 所以元素拖拽的时候可拖拽的范围的写法是：父级容器的clientWidth –被拖拽元素的 offsetWidth的大小