昨日知识点回顾：

01、京东页面轮播图的完善及完成

02、zepto实现京东轮播图+zepto的定制

03、分类页面全屏布局+结构创建

04、分类页面左侧实现滑动

05、移动端tap事件对tap事件封装及使用

今日知识点：

01、使用zepto实现左侧点击操作

02、移动端touch事件点透问题+fastclick插件的使用

03、iscroll插件的使用

04、swipe和swiper的插件的使用

05、响应式布局

# 1 使用zepto实现左侧点击操作

## 1.1 01-移动web-使用zepto来实现左侧单击操作

1. **如何使用zepto来代替封装好的tap事件**
   1. 把我们之前写的调用封装函数给注释掉

itcast.tap(ulBox,**function**(e){

* 1. 把里面的执行回调函数通过zepto的方法来执行

$(ulBox).on("tap",**function**(e){

* 1. 把之前引入的common.js注释掉，引入我们的zepto.min.js 就可实现zepto左侧单机操作

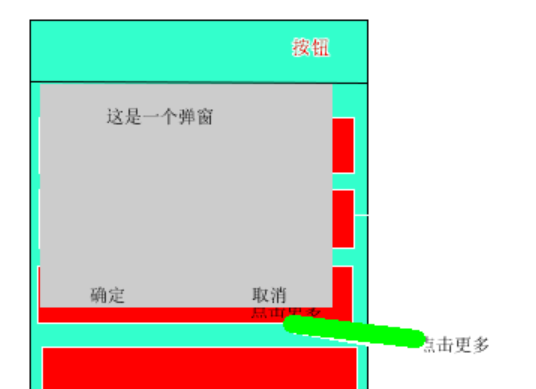
# 2 移动端touch事件点透问题+fastclick插件的使用

## 2.1 02-移动web-移动端touch事件所产生的点透问题

1. **什么是点透**

当我们触发当前的事件时会导致跟它相关的对象也会触发绑定在他们身上的的事件就是点透。下面用例子和图展示一下

点击按钮会出现一个弹窗，弹窗上面有一个确定按钮和取消按钮 ，当我们点击取消按钮的时候，同时也会触发点击更多这个按钮，这个效果我们就叫做点透 (详图如下)



## 2.2 03-移动web-fastclick的使用

**1、touch和click在解决点透上有什么弊端？**

* 1. touch 只能在移动端使用 pc端会无效
  2. touch 会产生点透
  3. click 不会产生点透,但是会有时间延迟

**2、怎么通过fastclick解决点透问题**

* 1. **引入fastclick.js文件**
  2. **通过js代码**

**if** ('addEventListener' **in** document) {  
 document.addEventListener('DOMContentLoaded', **function**() {  
 /\***参数可以是任意的dom元素，如果写document.body，说明会将document.body下面的所的元素都绑定fastclick**\*/  
 FastClick.attach(document.body);  
 }, **false**);  
}

**或者jquery**

$(**function**() {  
 FastClick.attach(document.body);  
});

* 1. **在对我们需要的对象绑定事件**

tap.addEventListener("click",**function**(){

**此处的click是fastclick.js封装好的click**  
 tap.style.visibility="hidden";  
})  
click.addEventListener("click",**function**(){  
 console.log(123);  
})

## 2.3 04-使用fastclick实现分类页面左侧栏的点击操作

1. 如何使用fastcick来替代之前tap实现的点击切换
   1. 引入fastclick.js文件
   2. 替换原始tap事件之前的代码

$(ulBox).on("tap",**function**(e){  
 /\*1.修改li元素的样式：将所有li元素的active样式清除，再为当前被点击的li元素添加active样式\*/  
 **for**(**var** i=0;i<lis.length;i++){  
 lis[i].classList.remove("active");  
 }  
 /\*为当前被单击的li元素添加样式\*/  
 **var** li=e.target.parentNode;  
 **var** liHeight=li.offsetHeight;  
 li.classList.add("active");  
 /\*2.移动当前的li元素到父容器的最顶部，但是不能超出之前设定了静止状态下的最小top值\*/  
 /\*获取当前li元素的索引值\*/  
 **var** index=li.index;  
 /\*开启过渡\*/  
 ulBox.style.transition="top .5s";  
 /\*设置偏移\*/  
 **if**(-index\*liHeight < minTop){  
 /\*只能偏移到minTop位置\*/  
 ulBox.style.top=minTop+"px";  
 currentY=minTop;  
 }  
 **else**{  
 ulBox.style.top=-index\*liHeight+"px";  
 currentY=-index\*liHeight;  
 }  
});

* 1. **用fastclick替换后**

/\*绑定fastclick\*/  
**if** ('addEventListener' **in** document) {  
 document.addEventListener('DOMContentLoaded', **function**() {  
 /\*参数可以是任意的dom元素，如果写document.body，说明会将document.body下面的所的元素都绑定fastclick\*/  
 FastClick.attach(document.body);  
 }, **false**);  
}  
/\*fastclick使用的时候就是来绑定添加click事件\*/  
ulBox.addEventListener("click",**function**(e){  
 /\*1.修改li元素的样式：将所有li元素的active样式清除，再为当前被点击的li元素添加active样式\*/  
 **for**(**var** i=0;i<lis.length;i++){  
 lis[i].classList.remove("active");  
 }  
 /\*为当前被单击的li元素添加样式\*/  
 **var** li=e.target.parentNode;  
 **var** liHeight=li.offsetHeight;  
 li.classList.add("active");  
 /\*2.移动当前的li元素到父容器的最顶部，但是不能超出之前设定了静止状态下的最小top值\*/  
 /\*获取当前li元素的索引值\*/  
 **var** index=li.index;  
 /\*开启过渡\*/  
 ulBox.style.transition="top .5s";  
 /\*设置偏移\*/  
 **if**(-index\*liHeight < minTop){  
 /\*只能偏移到minTop位置\*/  
 ulBox.style.top=minTop+"px";  
 currentY=minTop;  
 }  
 **else**{  
 ulBox.style.top=-index\*liHeight+"px";  
 currentY=-index\*liHeight;  
 }  
})

# 3、iscroll插件的使用

## 3.1 05-移动web-iscroll的使用说明(上)

1. **iscrolls 是什么？**

一个可以实现客户端原生滚动效果的类库。

## 3.2 06-移动web-iscroll的使用说明(下)

1. **Iscroll基本使用的结构：**

<div id="wrapper">  
 <ul>  
 <li>...</li>  
 <li>...</li>  
 ...  
 </ul>  
</div>

<script src="./js/iscroll.js"></script>  
<script type="text/javascript">  
 **var** myScroll = **new** IScroll('.ct\_hotCategory',{  
 mouseWheel: **true**, 鼠标滚动也可以实现滑动  
 scrollbars: **true 添加滚动条** });  
</script>

1. **滚动条位置如何调整**

**只需要给id为wrapper的标签元素添加一个position:relative属性就可以实现**

# 4 swipe和swiper的插件的使用

## 4.1 07-移动web-swipe插件的使用

1. **加visibility:hidden的作用**

不加就会加载很慢会让图片都显示出来(呈现先伸开后隐藏)影响用户体验

1. **在加入visibility:hidden后如何让图片滑动轮播**

window.mySwipe = Swipe(document.querySelector(".jd\_banner"));

1. **其他参数的设置和了解**



1. **设置上一张和下一张**

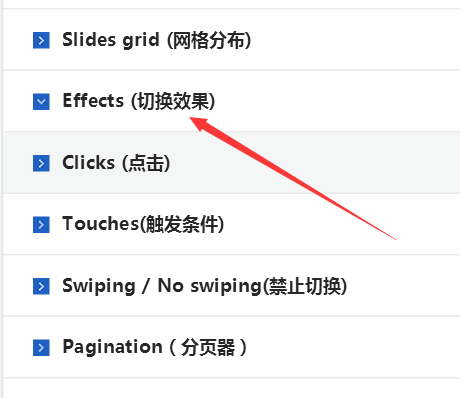
document.getElementById("prev").onclick=**function**(){  
 window.mySwipe.prev(); **上一张**}

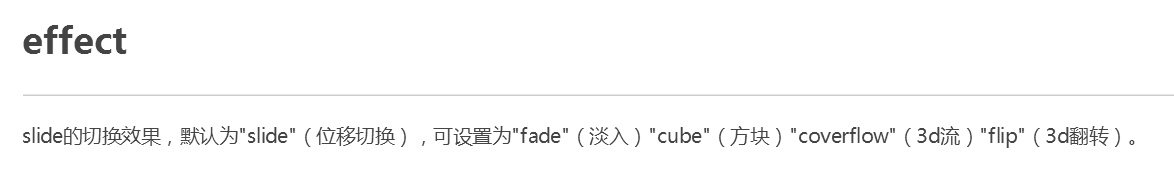
document.getElementById("next").onclick=**function**(){  
 window.mySwipe.next(); **下一张**}

## 4.2 08-移动web-swiper插件的使用介绍

**var** mySwiper = **new** Swiper('.swiper-container',{  
 effect : 'cube',  
 cube: {  
 slideShadows: **true**,  
 shadow: **true**,  
 shadowOffset: 100,  
 shadowScale: 0.6  
 }  
})

参数的了解(也可通过文档详细查阅)





slideShadows：开启slide阴影。默认 true。  
shadow： 开启投影。默认 true。  
shadowOffset：投影距离。默认 20，单位px。  
shadowScale： 投影缩放比例。默认0.94。

## 4.3 09-使用iscroll插件完成分类页面右侧内容块的滑动操作

**步骤分析：**

1. 引入iscroll.js文件
2. 引入iscroll.js代码

<script src="./js/iscroll.js"></script>  
<script type="text/javascript">  
 **var** myScroll = **new** IScroll('.ct\_hotCategory',{  
 mouseWheel: **true**,  
 scrollbars: **true** });  
</script>

1. 设置当前对象有滚动条，给当前对象加一个 position:relative

# 5 响应式布局

## 5.1 10-移动web-网页布局方式

1. **网页布局有几种方式？（红色加粗为即将要学的重点布局）**

a、固定宽度布局：为网页设置一个固定的宽度，通常以px做为长度单位，常见于PC端网页。

b、流式布局：为网页设置一个相对的宽度，通常以百分比做为长度单位。

c、**栅格化布局**：将网页宽度人为的划分成均等的长度，然后排版布局时则以这些均等的长度做为度量单位，通常利用百分比做为长度单位来划分成均等的长度。

d、**响应式布局**：通过检测设备信息，决定网页布局方式，即用户如果采用不同的设备访问同一个网页，有可能会看到不一样的内容，一般情况下是检测设备屏幕的宽度来实现。

注：以上几种布局方式并不是独立存在的，实际开发过程中往往是相互结合使用的。

## 5.2 11-移动web-响应式布局的简单说明

1. **什么是响应式布局？**

屏幕尺寸不一样展示给用户的网页内容也不一样

1. **响应式布局是如何实现的？**

利用媒体查询可以检测到屏幕的尺寸（主要检测宽度），并设置不同的CSS样式，就可以实现响应式的布局。

1. **响应式开发的优缺点**
   1. 一个页面可适配不同的终端
   2. 产生代码体积过大各种终端的资源都会存在加载
2. **常见的设备尺寸分类 (做专门的布局设计区分)**



1. **常见的设备尺寸对应的媒体查询条件**

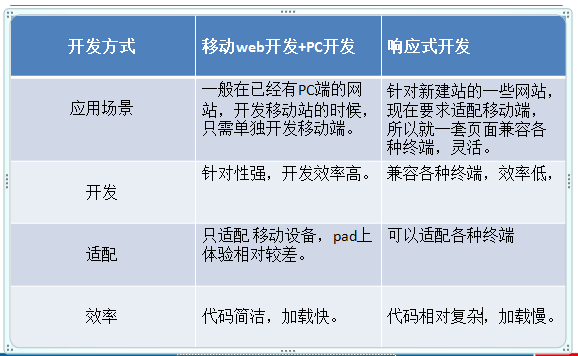


## 5.3 12-移动web-什么是响应式开发

**1、什么是响应式开发**

页面在不同的终端做不同的屏幕适配显示不同的页面效果

**2、移动开发和响应式开发的区别**



## 5.4 13-移动web-初体验媒体查询的语法

**1、响应式开发的原理：媒体查询：（**重点 **）**

* 1. 查询媒介，查询到当前屏幕(媒介媒体)的宽度，针对不同的屏幕宽度设置不同的样式来适应不同屏幕。当你重置浏览器大小的过程中，页面也会根据浏览器的宽度和高度重新渲染页面。简单说，你可以设置 不同屏幕下面的不同的样式，达到适配不同的屏幕的目的。
  2. 实现方式：通过查询screen的宽度来指定某个宽度区间的网页布局。

超小屏幕 （移动设备） w<768px

小屏设备 768px-992px 768 <= w <992

中等屏幕 992px-1200px 992 =< w <1200

宽屏设备 1200px以上 w>=1200

1. 媒体查询语法 (**注意：空格不能省略，否则无法生效**)

@media screen and (max-width: 768px){  
 body{  
 background-color: green;  
 }  
}  
@media screen and (min-width: 768px) and (max-width: 992px){  
 body{  
 background-color: blue;  
 }  
}

## 5.5 14-移动web-媒体查询的条件判断的顺序说明(上)

**1、向上兼容和向下覆盖的原则？**

1.向上兼容：如果设置了宽度更小时的样式，默认这些样式也会传递到宽度更大的条件范围内 (宽度向上找，直到找到宽度最大的那个)  
2.向下覆盖：宽度更大的样式会将前面宽度更小的样式覆盖\*

## 5.6 15-移动web-媒体查询的条件判断的顺序说明(下)

1、如果判断max-width该怎么书写规则？

1.如果是判断最小值 (min-width)，那么就应该从小到大写--bootstrap就是判断最小值，从小到大写  
2.如果是判断最大值 (max-width)，那么就应该从大到小写

## 5.7 16-移动web-媒体查询中的媒体功能的说明

**1、min-width和min-device-width的区别**

1. min-width:在pc端和移动端能正常的响应，效果一致 。它是指当前可视区域的宽度

2、min-device-width:在移动端和期望值一样 device：设备。它是指当前设备的宽度。当拖动来改变浏览器大小的时候，当前终端设备的宽度并不会变化，意味着媒体查询条件不会响应

## 5.8 17-移动web-媒体查询补充说明

**1、如何在不同屏幕调用不同样式？**

**当屏幕尺寸宽度在992-1200就调用b.css ,不在这个范围就调用a.css**

<link rel="stylesheet" href="a.css">  
<link rel="stylesheet" media="screen and (min-width:992px) and (max-width:1200px)" href="b.css">

**2、not的使用**

**屏幕的尺寸不在768-992，body就显示背景颜色blue**

**屏幕在这个尺寸范围就显示红色**

<style>  
 body{background-color: red;}  
 @media not screen and (min-width: 768px) and (max-width: 992px){  
 body{  
 background-color: blue;  
 }  
 }  
</style>

## 5.9 18-移动web-使用媒体查询设置页面布局

通过媒体查询重置在不同尺寸下重置页面结构的样式

## 5.10 19-移动web-常见的响应式框架简单介绍





对于这类框架我们都是按照他所提供好的文档来学习，这时就需要同学们有一定的自学能力

## 5.11 20-移动web-bootstrap简单介绍

**1、bootstrap的特点**

简洁、直观、强悍的前端开发框架，让web开发更迅速、简单。

**提示：**本地官方页面：1-教学资料\bootcss\v3.bootcss.com\index.html

## 5.12 21-移动web-bootstrap简单介绍（2）

1. **要兼容media query需引入 respond.js**

为了让 IE 浏览器运行最新的渲染模式下  
<meta http-equiv="X-UA-Compatible" content="IE=edge">

## 5.13 22-移动web-bootstrap的模板文件

**使用框架前需要引入的一些文件，掌握需要引入哪些文件即可**