黑马程序员

传智播客
www.itcast.cn

面试宝典V4.0

就业部--前端小组

此次版本升级，是集合了上海校区前端学科开班以来就业学员提供的面试题，由就业部联合教研部整理出版。

2018.06.26

Web 前端与移动开发面试宝典

一、HTML+CSS 部分（）

1、怎么让一个不定宽高的 DIV，垂直水平居中?

答：1）**使用 CSS 方法：**

父盒子设置：display：table-cell； text-align：center；vertical-align：middle；

Div 设置： display：inline-block；vertical-align：middle；

2）**使用 CSS3 transform：**

父盒子设置：display：relative

Div 设置： transform： translate(-50%，-50%)；position： absolute；top： 50%；left：50%；

2、position 几个属性的作用?

答：position 的常见四个属性值： relative，absolute，fixed，static。一般都要配合"left"、"top"、"right"以及 "bottom" 属性使用。

1）Static：默认位置，设置为 static 的元素，它始终会处于页面流给予的位置（static 元素会忽略任何 top、bottom、left 或 right 声明）。一般不常用。

2）Relative：位置被设置为 relative 的元素，可将其移至相对于其正常位置的地方，意思就是如果设置了 relative 值，那么，它偏移的 top，right，bottom，left 的值都以它原来的位置为基准偏移，而不管其他元素会怎么样。注意 relative 移动后的元素在原来的位置仍占据空间。

3）Absolute：位置设置为 absolute 的元素，可定位于相对于包含它的元素的指定坐标。意思就是如果它的父容器设置了 position 属性，并且 position 的属性值为 absolute 或者 relative，那么就会依据父容器进行偏移。如果其父容器没有设置 position 属性，那么偏移是以 body 为依据。注意设置 absolute 属性的元素在标准流中不占位置。

4）Fixed：位置被设置为 fixed 的元素，可定位于相对于浏览器窗口的指定坐标。不论窗口滚动与否，元素都会留在那个位置。它始终是以 body 为依据的。 注意设置 fixed 属性的元素在标准流中不占位置。

3、px，em，rem 的区别?

答： 1）px 像素（Pixel）。绝对单位。像素 px 是相对于显示器屏幕分辨率而言的，是一个虚拟长度单位，是计算机系统的数字化图像长度单位，如果 px 要换算成物理长度，需要指定精度 DPI。

2）em 是相对长度单位，相对于当前对象内文本的字体尺寸。如当前对行内文本的字体尺寸未被人为设置，则相对于浏览器的默认字体尺寸。它会继承父级元素的字体大小，因此并不是一个固定的值。

3）rem 是 CSS3 新增的一个相对单位（root em，根 em），使用 rem 为元素设定字体大小时，仍然是相对大小，但相对的只是 HTML 根元素。

4）区别：IE 无法调整那些使用 px 作为单位的字体大小，而 em 和 rem 可以缩放，rem 相对的只是 HTML 根元素。这个单位可谓集相对大小和绝对大小的优点于一身，通过它既可以做到只修改根元素就成比例地调整所有字体大小，又可以避免字体大小逐层复合的连锁反应。目前，除了 IE8 及更早版本外，所有浏览器均已支持 rem。

4、什么是 BFC?

答： 1）定义：

BFC(Block formatting context)直译为"块级格式化上下文"。它是一个独立的渲染区域，只有 Block-level box 参与， 它规定了内部的 Block-level Box 如何布局，并且与这个区域外部毫不相干。

布局规则：

A. 内部的 Box 会在垂直方向，一个接一个地放置。

B. Box 垂直方向的距离由 margin 决定。属于同一个 BFC 的两个相邻 Box 的 margin 会发生重叠。

C. 每个元素的 margin box 的左边， 与包含块 border box 的左边相接触(对于从左往右的格式化，否则相反)。即使存在浮动也是如此。

D. BFC 的区域不会与 float box 重叠。

E. BFC 就是页面上的一个隔离的独立容器，容器里面的子元素不会影响到外面的元素。反之也如此。

F. 计算 BFC 的高度时，浮动元素也参与计算。

3）哪些元素会生成 BFC：

A. 根元素

B. float 属性不为 none

C. position 为 absolute 或 fixed

D. display 为 inline-block， table-cell， table-caption， flex， inline-flex

F. overflow 不为 visible

5、CSS 引入的方式有哪些? link 和@import 的区别是?

答：内联 内嵌 外链 导入

区别 ： 同时加载

前者无兼容性，后者 CSS2.1 以下浏览器不支持

Link 支持使用 javascript 改变样式，后者不可

6、描述 css reset 的作用和用途?

答：Reset 重置浏览器的 css 默认属性 ，览器的品种不同，样式不同，然后重置，让他们统一。

7、解释 css sprites，如何使用?

答：Css 精灵图，把一堆小的图片整合到一张大的图片（png）上，减轻服务器对图片的请求数量。

8、清除浮动的几种方式?

答：1，父级 div 定义 height

原理：父级 div 手动定义 height，就解决了父级 div 无法自动获取到高度的问题。 简单、代码少、容易掌握 ，但只适合高度固定的布局.

2，结尾处加空 div 标签 clear：both

原理：在浮动元素的后面添加一个空 div 兄弟元素，利用 css 提高的 clear：both 清除浮动，让父级 div 能自动获取到高度 ，如果页面浮动布局多，就要增加很多空 div，让人感觉很不好 .

3，父级 div 定义 伪类：after 和 zoom

```
/*清除浮动代码*/
.clearfix：after{
content：""；
display：block；
visibility：hidden；
height：0；
line-height：0；
```

clear：both；

}

.clearfix{zoom：1}

原理：IE8 以上和非 IE 浏览器才支持：after，原理和方法 2 有点类似，zoom(IE 转有属性)可解决 ie6，ie7 浮动问题，推荐使用，建议定义公共类，以减少 CSS 代码。

4，父级 div 定义 overflow：hidden

超出盒子部分会被隐藏，不推荐使用.

5. 双伪元素法：

```
.clearfix：before，.clearfix：after {
        content：  "";
        display：  block;
        clear：  both;
    }
    .clearfix {
        zoom：  1;
    }
```

9、能否简述一下如何使一套设计方案，适应不同的分辨率，有哪些方法可以实现?

答：流式布局：

使用非固定像素来定义网页内容，也就是百分比布局，通过盒子的宽度设置成百分比来根据屏幕的宽度来进行伸缩，不受固定像素的限制，内容向两侧填充。

这样的布局方式，就是移动 web 开发使用的常用布局方式。这样的布局可以适配移动端不同的分辨率设备。

响应式开发：

那么 Ethan Marcotte 在 2010 年 5 月份提出的一个概念，简而言之，就是一个网站能够兼容多个终端。越来越多的设计师也采用了这种设计。

CSS3 中的 Media Query（媒介查询），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）　　768px 以下
- ✧ 小屏设备　　768px-992px
- ✧ 中等屏幕　　992px-1200px

✧ 宽屏设备 1200px 以上

由于响应式开发显得繁琐些，一般使用第三方响应式框架来完成，比如 bootstrap 来完成一部分工作，当然也可以自己写响应式。

阐述下移动 web 和响应式的区别：

开发方式	移动web开发+PC开发	响应式开发
应用场景	一般在已经有PC端的网站，开发移动站的时候，只需单独开发移动端。	针对新建站的一些网站，现在要求适配移动端，所以就一套页面兼容各种终端，灵活。
开发	针对性强，开发效率高。	兼容各种终端，效率低，
适配	只适配 移动设备，pad上体验相对较差。	可以适配各种终端
效率	代码简洁，加载快。	代码相对复杂，加载慢。

10、你能描述一下渐进增强和优雅降级之间的不同吗?

答：渐进增强 progressive enhancement：针对低版本浏览器进行构建页面，保证最基本的功能，然后再针对高级浏览器进行效果、交互等改进和追加功能达到更好的用户体验。

优雅降级 graceful degradation：一开始就构建完整的功能，然后再针对低版本浏览器进行兼容。区别：优雅降级是从复杂的现状开始，并试图减少用户体验的供给，而渐进增强则是从一个非常基础的，能够起作用的版本开始，并不断扩充，以适应未来环境的需要。降级（功能衰减）意味着往回看；而渐进增强则意味着朝前看，同时保证其根基处于安全地带

举个例子：

```
a{
    display: block;
    width: 200px;
    height: 100px;
    background:aquamarine;
    /*我就是要用这个新 css 属性*/
     transition: all 1s ease 0s;
   /*可是发现了一些低版本浏览器不支持怎么吧*/
```

```
    /*往下兼容*/
    -webkit-transition: all 1s ease 0s;
    -moz-transition: all 1s ease 0s;
    -o-transition:  all 1s ease 0s;
   /*那么通常这样考虑的和这样的侧重点出发的 css 就是优雅降级*/
}
a: hover{
    height:  200px;
}

/  *那如果我们的产品要求我们要重低版本的浏览器兼容开始*/
  a{
 /*优先考虑低版本的*/
    -webkit-transition: all 1s ease 0s;
    -moz-transition: all 1s ease 0s;
    -o-transition:  all 1s ease 0s;
    /*高版本的就肯定是渐进渐强*/
   transition:  all 1s ease 0s;
}
```

“优雅降级”观点认为应该针对那些最高级、最完善的浏览器来设计网站。

“渐进增强”观点则认为应关注于内容本身。

11、合理的页面布局中常听过结构与表现分离，那么结构是什么？表现是什么？

答：结构是 html，表现是 css。

12、简述对 Web 语义化的理解？

答：就是让浏览器更好的读懂你写的代码，在进行 HTML 结构、表现、行为设计时，尽量使用语义化的标签，使程序代码简介明了，易于进行 Web 操作和网站 SEO，方便团队协作的一种标准，以图实现一种“无障碍”的 Web 开发。

13、CSS 都有哪些选择器？CSS 选择器的优先级是怎么样定义的？

答：一般而言，选择器越特殊，它的优先级越高。也就是选择器指向的越准确，它的优先级就越高。

！important >行内样式> ID > 类 > 标签 | 伪类 | 属性选择 > 伪对象 > 继承 > 通配符。

14、display： none；与 visibility： hidden 的区别是什么?

答：display：none； 使用该属性后，HTML 元素（对象）的宽度、高度等各种属性值都将“丢失”；

visibility：hidden； 使用该属性后，HTML 元素（对象）仅仅是在视觉上看不见（完全透明），而它所占据的空间位置仍然存在，也即是说它仍具有高度、宽度等属性值。

15、用 CSS 定义 p 标签，要求实现以下效果：字体颜色在 IE6 下为黑色(#000000)；IE7 下为红色(#ff0000)；而其他浏览器下为绿色(#00ff00)

```
答：p{
    color：green；
    *color：blue；
    _color：black；
}
```

16、HTML 与 XHTML——二者有什么区别

HTML 是一种基本的 WEB 网页设计语言，XHTML 是一个基于 XML 的置标语言

最主要的不同：

XHTML 元素必须被正确地嵌套。

XHTML 元素必须被关闭。

标签名必须用小写字母。

XHTML 文档必须拥有根元素。

17、Doctype 作用？严格模式与混杂模式-如何触发这两种模式，区分它们有何意义?

答：DOCTYPE 是一种标准通用标记语言的文档类型声明，它的目的是要告诉标准通用标记语言解析器，它应该使用什么样的文档类型定义来解析文档。只有确定了一个正确的文档类型，超文本标记语言或可扩展超文本标记语言中的标签和层叠样式表才能生效，甚至对 javascript 脚本都会有所影响。

标准模式是指，浏览器按 W3C 标准解析执行代码；怪异模式则是使用浏览器自己的方式解析执行代码，因为不同浏览器解析执行的方式不一样，所以我们称之为怪异模式。

浏览器解析时到底使用标准模式还是怪异模式，与你网页中的 DTD 声明直接相关，DTD 声明定义了标

准文档的类型（标准模式解析）文档类型，会使浏览器使用相应的方式加载网页并显示，忽略 DTD 声明,将使网页进入怪异模式(quirks mode)。

如何触发两种模式

DOCTYPE 不存在或形式不正确会导致 HTML 和 XHTML 文档以混杂模式呈现。

触发严格模式:

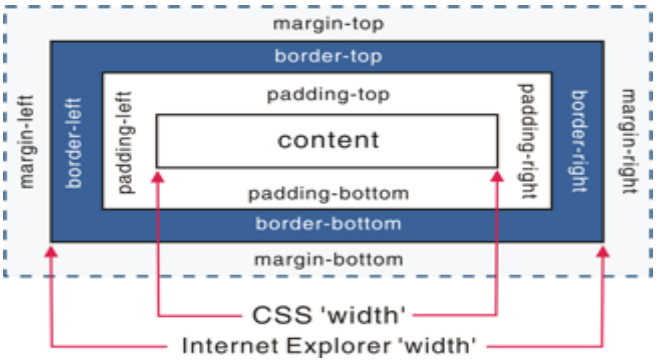
```
<!-- HTML 4.01 严格型 -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<!-- XHTML 1.0 严格型 -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

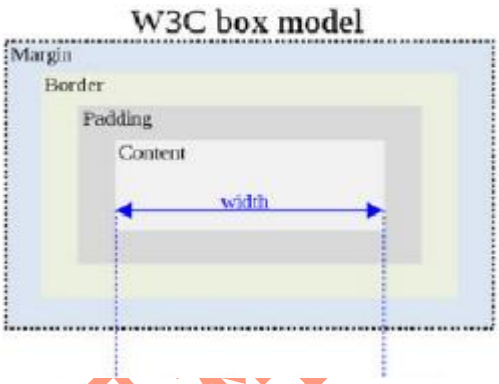
触发混杂模式

```
<!-- HTML 4.01 过渡型 -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<!-- HTML 4.01 框架集型 -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">

<!-- XHTML 1.0 过渡型 -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<!-- XHTML 1.0 框架集型 -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

区分它们的意义：doctype 声明指出阅读程序应该用什么规则集来解释文档中的标记。

18、盒模型：3C 标准中，如果设置一个元素的宽度和高度，指的是元素内容的宽度和高度，而在 Quirks 模式下，IE 的宽度和高度还包含了 padding 和 border。





设置行内元素的高宽：在 Standards 模式下，给等行内元素设置 wdith 和 height 都不会生效，而在 quirks 模式下，则会生效。

设置百分比的高度：在 standards 模式下，一个元素的高度是由其包含的内容来决定的，如果父元素没有设置百分比的高度，子元素设置一个百分比的高度是无效的用,quirk 模式下的解决办法，用 text-align 属性:

```
body{text-align:center};#content{text-align:left}
```

19、Css Hack？ie6,7,8 的 hack 分别是什么？

答案：针对不同的浏览器写不同的 CSS code 的过程，就是 CSS hack。

示例如下：

```
#test{
Width:300px;
Height:300px;
background-color:blue
}
   #test{
                width:300px;
                height:300px;
                background-color:blue;          /*firefox*/
```

```
            background-color:red\9;            /*all ie*/
            background-color:yellow;        /*ie8*/
            +background-color:pink;                /*ie7*/
            _background-color:orange;              /*ie6*/       }
            :root #test { background-color:purple\9; }   /*ie9*/
@media all and (min-width:0px)
     { #test {background-color:black;} }  /*opera*/
@media screen and (-webkit-min-device-pixel-ratio:0)
{ #test {background-color:gray;} }        /*chrome and safari*/
```

20、请用 div+css 写出左侧固定(width:200px;),右侧自适应页面宽度.

Html: 左侧 右侧

Css：
```
#content{width:100%; float:right;margin-left:240px}
    #side{width:240px;height:400px; float:left}
```

21、display：inline-block 什么时候会显示间隙?

真正意义上的 inline-block 水平呈现的元素间，换行显示或空格分隔的情况下会有间距。
解决：父元素{font-size：0；-webkit-text-size-adjust：none；}

22、overflow 有哪些属性值?

Visible：默认值。内容不会被修剪，会呈现在元素框之外。
Hidden：内容会被修剪，并且其余内容是不可见的。
Scroll：内容会被修剪，但是浏览器会显示滚动条以便查看其余的内容。
Auto：如果内容被修剪，则浏览器会显示滚动条以便查看其余的内容。
Inherit：规定应该从父元素继承 overflow 属性的值

23、css 去掉 iPhone、iPad 默认按钮样式

```
input[type="button"], input[type="submit"], input[type="reset"] {
-webkit-appearance: none;
}
textarea {  -webkit-appearance: none;}
```

24、CSS 样式初始化的目的

是为了考虑到浏览器的兼容问题，其实不同浏览器对有些标签的默认值是不同的，如果没对 CSS 初始化往往会出现浏览器之间的页面差异。当然，初始化样式会对 SEO 有一定的影响，但鱼和熊掌不可兼得，但力求影响最小的情况下初始化。

25、用 div+css 网站布局的好处

1：表现和内容相分离

将设计部分剥离出来放在一个独立样式文件中，HTML 文件中只存放文本信息。

2：提高搜索引擎对网页的索引效率

用只包含结构化内容的 HTML 代替嵌套的标签，搜索引擎将更有效地搜索到你的网页内容，并可能给你一个较高的评价。

3：提高页面浏览速度

对于同一个页面视觉效果，采用 CSS+DIV 重构的页面容量要比 TABLE 编码的页面文件容量小得多，前者一般只有后者的1/2大小。

4：易于维护和改版

你只要简单的修改几个 CSS 文件就可以重新设计整个网站的页面。从以上的描述来看，采用 CSS+DIV 对网站重构可以大大提升网站用户与搜索引擎的友好度,CSS+DIV 所以成为目前网页布局主流.

26、用表格 table 对 seo 的影响

1、简介：为何使用表格排版是不明智的?

表格之所以存在于 HTML 中，只是为了一个目的：显示表格状的数据。然而此后的 border="0" 使得设计师可以将图片和文本放在这无形的网格中。迄今为止，表格仍然主导着视觉丰富的网站的设计方式，但它却阻碍了一种更好的、更有亲和力的、更灵活的，而且功能更强大的网站设计方法。

2、表格带来的问题

把格式数据混入你的内容中,这使得文件的大小无谓地变大，而用户访问每个 页面时都必须下载一次这样的格式信息。

27、console 有哪些常用方法?(答出三种并说出它的用法即可，想考察你有没有工作经验)

console.log/console.info,console.error/console.warning, console.time/console.timeEnd, console.trace,

console.table

28、为什么利用多个域名来存储网站资源会更有效?

① CDN 缓存更方便 ②突破浏览器并发限制③节约 cookie 带宽④节约主域名的连接数，优化页面响应速度
⑤防止不必要的安全问题

29、网页的重绘与重排以及重构（2016 年腾讯校招面试题）

重绘是一个元素外观的改变所触发的浏览器行为，例如改变 visibility、outline、背景色等属性。浏览器会根据元素的新属性重新绘制，使元素呈现新的外观。重绘不会带来重新布局，并不一定伴随重排。

重排是更明显的一种改变，可以理解为渲染树需要重新计算。下面是常见的触发重排的操作：

1.DOM 元素的几何属性变化 2. DOM 树的结构变化 3. 获取某些属性 4. 此外，改变元素的一些样式，调整浏览器窗口大小等等也都将触发重排。

注：重排对性能有很大的影响

页面重构：编写 CSS、让页面结构更合理化，提升用户体验，实现良好的页面效果和提升性能。

网站重构：在不改变外部行为的前提下，简化结构、添加可读性，而在网站前端保持一致的行为。也就是说是在不改变 UI 的情况下，对网站进行优化，在扩展的同时保持一致的 UI。

对于传统的网站来说重构通常是：

表格(table)布局改为 DIV+CSS

使网站前端兼容于现代浏览器(针对于不合规范的 CSS、如对 IE6 有效的)

对于移动平台的优化

针对于 SEO 进行优化

深层次的网站重构应该考虑的方面：

减少代码间的耦合

让代码保持弹性

严格按规范编写代码

设计可扩展的 API

代替旧有的框架、语言(如 VB)

增强用户体验

优化响应速度

速度的优化重构：

压缩 JS、CSS、image 等前端资源(通常是由服务器来解决)

程序的性能优化(如数据读写)

采用 CDN 来加速资源加载

对于 JS DOM 的优化

HTTP 服务器的文件缓存

30、谈谈以前端角度出发做好 SEO 需要考虑什么?

了解搜索引擎如何抓取网页和如何索引网页

你需要知道一些搜索引擎的基本工作原理，各个搜索引擎之间的区别，搜索机器人（SE robot 或叫 web crawler）如何进行工作，搜索引擎如何对搜索结果进行排序等等。

Meta 标签优化

主要包括主题（Title），网站描述(Description)，和关键词（Keywords）。还有一些其它的隐藏文字比如 Author（作者），Category（目录），Language（编码语种）等。

如何选取关键词并在网页中放置关键词

搜索就得用关键词。关键词分析和选择是 SEO 最重要的工作之一。首先要给网站确定主关键词(一般在 5 个上下)，然后针对这些关键词进行优化，包括关键词密度(Density)，相关度(Relavancy)，突出性（Prominency）等等。

了解主要的搜索引擎

虽然搜索引擎有很多，但是对网站流量起决定作用的就那么几个。比如英文的主要有 Google，Yahoo，Bing 等；中文的有百度，搜狗，有道等。不同的搜索引擎对页面的抓取和索引、排序的规则都不一样。还要了解各搜索门户和搜索引擎之间的关系，比如 AOL 网页搜索用的是 Google 的搜索技术，MSN 用的是 Bing 的技术。

主要的互联网目录

Open Directory 自身不是搜索引擎，而是一个大型的网站目录，他和搜索引擎的主要区别是网站内容的收集方式不同。目录是人工编辑的，主要收录网站主页；搜索引擎是自动收集的，除了主页外还抓取大量的内容页面。

按点击付费的搜索引擎

搜索引擎也需要生存，随着互联网商务的越来越成熟，收费的搜索引擎也开始大行其道。最典型的有 Overture 和百度，当然也包括 Google 的广告项目 Google Adwords。越来越多的人通过搜索引擎的点击广告来定位商业网站，这里面也大有优化和排名的学问，你得学会用最少的广告投入获得最多的点击。

搜索引擎登录

网站做完了以后，别躺在那里等着客人从天而降。要让别人找到你，最简单的办法就是将网站提交（submit）到搜索引擎。如果你的是商业网站，主要的搜索引擎和目录都会要求你付费来获得收录（比如 Yahoo 要 299 美元），但是好消息是（至少到目前为止）最大的搜索引擎 Google 目前还是免费，而且它主宰着 60%以上的搜索市场。

链接交换和链接广泛度（Link Popularity）

网页内容都是以超文本（Hypertext）的方式来互相链接的，网站之间也是如此。除了搜索引擎以外，人们也每天通过不同网站之间的链接来 Surfing（“冲浪”）。其它网站到你的网站的链接越多，你也就会获得更多的访问量。更重要的是，你的网站的外部链接数越多，会被搜索引擎认为它的重要性越大，从而给你更高的排名。

合理的标签使用

31、rgba()和 opacity 的透明效果有什么不同?

rgba()和 opacity 都能实现透明效果，但最大的不同是 opacity 作用于元素，以及元素内的所有内容的透明度，
而 rgba()只作用于元素的颜色或其背景色。（设置 rgba 透明的元素的子元素不会继承透明效果！）

32、Sass、LESS 是什么？大家为什么要使用他们?

他们是 CSS 预处理器。他是 CSS 上的一种抽象层。他们是一种特殊的语法/语言编译成 CSS。
例如 Less 是一种动态样式语言. 将 CSS 赋予了动态语言的特性，如变量，继承，运算， 函数. LESS 既可以在客户端上运行 (支持 IE 6+, Webkit, Firefox)，也可一在服务端运行 (借助 Node.js)。
为什么要使用它们?
结构清晰，便于扩展。
可以方便地屏蔽浏览器私有语法差异。这个不用多说，封装对浏览器语法差异的重复处理，减少无意义的机械劳动。
可以轻松实现多重继承。
完全兼容 CSS 代码，可以方便地应用到老项目中。LESS 只是在 CSS 语法上做了扩展，所以老的 CSS 代码也可以与 LESS 代码一同编译。

33、行内元素有哪些？块级元素有哪些？空(void)元素有那些？样式之间的转换

（1）CSS 规范规定，每个元素都有 display 属性，确定该元素的类型，每个元素都有默认的 display 值，比如 div 默认 display 属性值为“block”，成为“块级”元素；span 默认 display 属性值为“inline”，是“行内”元素。
（2）行内元素有：a b span img input select strong（强调的语气）
块级元素有：div ul ol li dl dt dd h1 h2 h3 h4…p
（3）知名的空元素：

<hr><img><input><link><meta>鲜为人知的是： <area><base><col><command>
<embed><keygen><param><source><track><wbr>
（4）样式转换：
display:block 行内元素转换为块级元素
display:inline 块级元素转换为行内元素
display:inline-block 转为内联元素

二、HTML5+CSS3 部分（）

1、CSS3 新特性有哪些?

答：1.颜色：新增 RGBA，HSLA 模式
2. 文字阴影（text-shadow、）
3.边框： 圆角（border-radius）边框阴影： box-shadow
4. 盒子模型：box-sizing
5.背景：background-size 设置背景图片的尺寸 background-origin 设置背景图片的原点 background-clip 设置背景图片的裁切区域，以”，”分隔可以设置多背景，用于自适应布局
6.渐变：linear-gradient、radial-gradient
7. 过渡：transition，可实现动画
8. 自定义动画 animate @keyfrom
9. 在 CSS3 中唯一引入的伪元素是 ：：selection.
10. 媒体查询，多栏布局 @media screen and (width:800px){ ... }
11. border-image
12.2D 转换：transform：translate(x，y) rotate(x，y) skew(x，y) scale(x，y)
13. 3D 转换
14 字体图标 font-face
15 弹性布局 flex

2、CSS3 选择器有哪些?

答：属性选择器、伪类选择器、伪元素选择器。

属性选择器 例如:[href="a.mp4"] {color: green;}

伪类选择器 例如:li:nth-child(3) {color: red;} span:empty h2:target li:not(.special) li:nth-last-child(3) {color: blue;}

伪元素选择器 例如:p::selection p::first-line li::first-letter ：：after ：：bofer

3、移动端优先使用弹性布局（flex）来解决布局问题，请列出弹性布局的相关属性，并说明属性用途

flex 的 3 个属性　flex-grow　| flex-shrink| flex-basis
flex-grow　一个数字，规定项目将相对于其他灵活的项目进行扩展的量。
flex-shrink　一个数字，规定项目将相对于其他灵活的项目进行收缩的量。
flex-basis　项目的长度。合法值："auto"、"inherit" 或一个后跟 "%"、"px"、"em" 或任何其他长度单位的数字。
用 js 设置 flex
object.style.flex="1"

4、CSS3 的兼容问题怎么处理?

一般加私有前缀否则不做特殊处理，再有就是遵循 2 大原则，渐进增强和优雅降级

5、 CSS3 新增伪类有那些?

p:first-child 选择属于其父元素的首个 <p> 元素的每个 <p> 元素。
p:last-child　选择属于其父元素的最后 <p> 元素的每个 <p> 元素。
p:nth-child(2n)　选择属于其父元素的第二个子元素的每个 <p> 元素。
:enabled、:disabled 控制表单控件的禁用状态。
:checked，单选框或复选框被选中。

6、 CSS3 动画和 JS 动画主要的不同点是什么?

1. 功能涵盖面，JS 比 CSS3 大

①定义动画过程的@keyframes 不支持递归定义，如果有多种类似的动画过程，需要调节多个参数来生成的话，将会有很大的冗余（比如 jQuery Mobile 的动画方案），而 JS 则天然可以以一套函数实现多个不同的动画过程

② 时间尺度上，@keyframes 的动画粒度粗，而 JS 的动画粒度控制可以很细
③ CSS3 动画里被支持的时间函数非常少，不够灵活
④ 以现有的接口，CSS3 动画无法做到支持两个以上的状态转化

2. 实现/重构难度不一，CSS3 比 JS 更简单，性能调优方向固定
3. 对于帧速表现不好的低版本浏览器，CSS3 可以做到自然降级，而 JS 则需要撰写额外代码
4. CSS 动画有天然事件支持（TransitionEnd、AnimationEnd，但是它们都需要针对浏览器加前缀），JS 则需要自己写事件
5. CSS3 有兼容性问题，而 JS 大多时候没有兼容性问题

7、请用 Css 写一个简单的幻灯片效果页面

答：答：
知道是要用css3。使用animation 动画实现一个简单的幻灯片效果。

```
/**HTML**/
div.ani

/**css**/
.ani{
width:480px;
height:320px;
margin:50px auto;
overflow: hidden;
box-shadow:0 0 5px rgba(0,0,0,1);
background-size: cover;
background-position: center;
-webkit-animation-name: "loops";
-webkit-animation-duration: 20s;
-webkit-animation-iteration-count: infinite;
}
@-webkit-keyframes "loops" {
0% {
background:url(http://d.hiphotos.baidu.com/image/w%3D400/sign=c01e6adca964034f0fcdc3069fc27980/e824b899a9014c08e5e38ca4087b02087af4f4d3.jpg) no-repeat;
}
25% {
background:url(http://b.hiphotos.baidu.com/image/w%3D400/sign=edee1572e9f81a4c2632edc9e72b6029/30adcbef76094b364d72bceba1cc7cd98c109dd0.jpg) no-repeat;
}
50% {
```

```
background:url(http://b.hiphotos.baidu.com/image/w%3D400/sign=937dace2552c
11dfded1be2353266255/d8f9d72a6059252d258e7605369b033b5bb5b912.jpg)
no-repeat;
26 }
27 75% {
28
background:url(http://g.hiphotos.baidu.com/image/w%3D400/sign=7d37500b854
4ebf86d71653fe9f9d736/0df431adcbef76095d61f0972cdda3cc7cd99e4b.jpg)
no-repeat;
29 }
30 100% {
31
background:url(http://c.hiphotos.baidu.com/image/w%3D400/sign=cfb239ceb0fb
43161a1f7b7a10a54642/3b87e950352ac65ce2e73f76f9f2b21192138ad1.jpg)
no-repeat;
32 }
```

8、Rem 的使用方法

目前，IE9+，Firefox、Chrome、Safari、Opera 的主流版本都支持了 rem。

就算对不支持的浏览器，应对方法也很简单，就是多写一个绝对单位的声明。这些浏览器会忽略用 rem 设定的字体大小。

- 使用%单位方便使用

css 中的 body 中先全局声明 font-size=62.5%，这里的%的算法和 rem 一样。

因为 100%=16px，1px=6.25%，所以 10px=62.5%，

这是的 1rem=10px，所以 12px=1.2rem。px 与 rem 的转换通过 10 就可以得来，很方便了吧！

- 使用方法

注意，rem 是只相对于根元素 htm 的 font-size，即只需要设置根元素的 font-size，其它元素使用 rem 单位设置成相应的百分比即可；

例子：

```
1 /*16px * 312.5% = 50px;*/
2 html{font-size: 312.5%;}
1 /*50px * 0.5 = 25px;*/
2 body{
3       font-size: 0.5rem;
4       font-size\0:25px;
5 }
```

一般情况下，是这样子使用的

```
1 html{font-size:62.5%;}
```

```
2 body{font-size:12px;font-size:1.2rem ;}
3 p{font-size:14px;font-size:1.4rem;}
```

- 优点

用一个东西肯定要知道它的好处啦，由于其他字体大小都是基于 html 的，所以在移动端做适配的时候，可以使用这样的方法

```
1 @media only screen and (min-width: 320px){
2     html {
3         font-size: 62.5% !important;
4     }
5 }
6 @media only screen and (min-width: 640px){
7     html {
8         font-size: 125% !important;
9     }
10 }
11 @media only screen and (min-width: 750px){
12     html {
13         font-size: 150% !important;
14     }
15 }
16 @media only screen and (min-width: 1242px){
17     html {
18         font-size: 187.5% !important;
19     }
20 }
```

9、用 css 画一个三角形，圆，椭圆?

1、圆形

最终效果：

CSS 代码如下：

```
#css3-circle{
  width: 150px;
  height: 150px;
  border-radius: 50%;
```

```
  background-color: #232323;}
```

2、椭圆
最终效果：

CSS 代码如下：

```
#css3-elipse{
  width: 200px;
  height: 100px;
  border-radius: 50%;
  background-color: #232323;}
```

3、上三角
最终效果：

CSS 代码如下：

```
#css3-triangle{
  width: 0;
  height: 0;
  border-left: 100px solid transparent;
  border-right: 100px solid transparent;
  border-bottom: 150px solid #232323;}
```

10、html5 有哪些新特性、移除了那些元素？如何处理 HTML5 新标签的浏览器兼容问题？

新特性：

1. 拖拽释放(Drag and drop) API　ondrop
   自定义属性　data-id　获取 li.getAttribute('data-id')或者 li.dataset.type = 'guoji'
2. 语义化更好的内容标签（header,nav,footer,aside,article,section）
3. 音频、视频 API(audio,video)　如果浏览器不支持自动播放怎么办？
4. 画布(Canvas) API 热　canvas 和 image 的区别？
5. 地理(Geolocation) API
6. 本地离线存储 localStorage 长期存储数据，浏览器关闭后数据不丢失；
7. sessionStorage 的数据在浏览器关闭后自动删除
8. 表单控件，calendar、date、time、email、url、search 、tel、file、number
9. 新的技术 webworker, websocket, Geolocation

10．文件读取
移除的元素-纯表现的元素：basefont，big，center，font, s，strike，tt，u；
支持 HTML5 新标签：
* IE8/IE7/IE6 支持通过 document.createElement 方法产生的标签，
可以利用这一特性让这些浏览器支持 HTML5 新标签，
浏览器支持新标签后，还需要添加标签默认的样式：
* 当然最好的方式是直接使用成熟的框架、使用最多的是 html5shim 框架
<!--[if lt IE 9]>
<script> src="http://html5shim.googlecode.com/svn/trunk/html5.js"</script>
<![endif]-->

11、HTML5 中新增的操作 DOM 的方法?

document.querySelector
document.querySelectorAll
document.getElementByClassName();

12、image 和 canvas 在处理图片的时候有什么区别?

答：image 是通过对象的形式描述图片的；canvas 通过专门的 API 将图片绘制在画布上。

13、HTML5 中的本地存储概念是什么？生命周期有多长?

标签之间的通信和购物车这两块都会用到本地存储
答：随着互联网的快速发展，基于网页的应用越来越普遍，同时也变的越来越复杂，为了满足各种各样的需求，会经常性在本地存储大量的数据，传统方式我们以 document.cookie 来进行存储的，但是由于其存储大小只有 4k 左右，并且解析也相当的复杂，给开发带来诸多不便，HTML5 规范则提出解决方案。HTML5 storage 提供了一种方式让网站能够把信息存储到你本地的计算机上，并再以后需要的时候进行获取。这个概念和 cookie 相似，区别是它是为了更大容量存储设计的。
答：cookies 兼容所有的浏览器，Html5 提供的 storage 存储方式。
① Document.cookie
② Window.localstorage
③ Window.sessionstorage
cookie 数据始终在同源的 http 请求中携带（即使不需要），即 cookie 在浏览器和服务器间来回传递。而 sessionStorage 和 localStorage 不会自动把数据发给服务器，仅在本地保存。
存储大小限制也不同，cookie 数据不能超过 4k，同时因为每次 http 请求都会携带 cookie，所以 cookie 只适合保存很小的数据，如会话标识。sessionStorage 和 localStorage 虽然也有存储大小的限制，但比 cookie 大得多，sessionStorage 约 5M、localStorage 约 20M

。

数据有效期不同，sessionStorage：仅在当前浏览器窗口关闭前有效，自然也就不可能持久保持；localStorage：始终有效，窗口或浏览器关闭也一直保存，因此用作持久数据；cookie 只在设置的 cookie 过期时间之前一直有效，即使窗口或浏览器关闭。

作用域不同，sessionStorage 不在不同的浏览器窗口中共享，即使是同一个页面；localStorage 在所有同源窗口中都是共享的；cookie 也是在所有同源窗口中都是共享的。

相关的一些方法详解

setItem(key， value) 设置存储内容

getItem(key) 读取存储内容

removeItem(key) 删除键值为 key 的存储内容

clear() 清空所有存储内容

key(n) 以索引值来获取存储内容

```
<script type="text/javascript">
        //添加 key-value 数据到 sessionStorage
        localStorage.setItem("demokey", "http://blog.itjeek.com");
        //通过 key 来获取 value
        var dt = localStorage.getItem("demokey");
        alert(dt);
        //清空所有的 key-value 数据。
        //localStorage.clear();
        alert(localStorage.length);
    </script>
```

14、移动端页面开发调试

BrowserSync，调试利器--自动刷新 对这个不懂的自己百度下，官网很全面。

15、什么是 Web Worker？为什么我们需要它？

参考答案：

查看如下代码（模拟会执行上百万次的繁重代码）：

```
function test(){
for(i=0;i< 10000000000 ; i ++){
x = x + i;
}
}
```

如果上述代码在 HTML 按钮点击以后执行，这种执行是同步的，即，浏览器必须等到此执行完毕之后才能进行其他操作。因为此操作耗时较长，那么这个操作会导致浏览器冻结并且没有响应，而且屏幕还会出现异常信息。

如果可以将这些繁重的代码移动到 Javascript 文件中，并采用异步的方式运行，就可以解决这个问题。这就是 web worker 的作用。Web Worker 用于异步执行 JavaScript 文件，提高浏览器的敏捷度。

16、HTML5 应用程序缓存和浏览器缓存有什么区别?

参考答案：
应用程序缓存是 HTML5 的重要特性之一，提供了离线使用功能，让应用程序可以获取本地的网站内容，例如 HTML、CSS、图片以及 Javascript。这个特性可以提高网站性能，它的实现借助于 manifest 文件，代码如下：

```
<!doctype html>
<html manifest="example.appcache">
....</html>
```

与传统浏览器缓存相比，它不强制用户访问的网站内容被缓存。

三、Javascript

1、JavaScript 是一门什么样的语言，它有哪些特点?

javaScript 一种直译式脚本语言，是一种动态类型、弱类型、基于原型的语言，内置支持类型。它的解释器被称为 JavaScript 引擎，为浏览器的一部分，广泛用于客户端的脚本语言，最早是在 HTML 网页上使用，用来给 HTML 网页增加动态功能。JavaScript 兼容于 ECMA 标准，因此也称为 ECMAScript。
基本特点
1．是一种解释性脚本语言（代码不进行预编译）。
2．主要用来向 HTML（标准通用标记语言下的一个应用）页面添加交互行为。
3．可以直接嵌入 HTML 页面，但写成单独的 js 文件有利于结构和行为的分离。
4．跨平台特性，在绝大多数浏览器的支持下，可以在多种平台下运行（如 Windows、Linux、Mac、Android、iOS 等）。

2、javascript 的本地对象，内置对象和宿主对象

本地对象为 array obj regexp 等可以 new 实例化
内置对象为 gload Math 等不可以实例化的
宿主为浏览器自带的 document,window 等

3、js 的内置对象有哪些？列举一下 arry 和 string 的常用方法？

JavaScript 常见的内置对象有 Object,Math,String,Array,Number,Function,Boolean,JSON 等,其中 Object 是所有对象的基类,采用了原型继承方式.

String:charAt(); charCodeAt(); indexOf(); match(); replace(); search(); slice(); toUpperCase(); toLowerCase(); 等方法

Array:shift(); unshift(); pop(); push(); concat(); reverse(); splice(); slice(); 等方法

4、例举 3 种强制类型转换和 2 种隐式类型转换?

强制：parseInt()、parseFloat()、Number()
隐式：== 、console.log()、alert()

5、操作 DOM 的常用 API ?

1.节点查找 document.getElementById.....
2.节点创建 createElement cloneNode.....
3.节点修改 appendChild insertBefore RemoveChild replaceChild.....
4.元素属性型 setAttribute getAttribute.....

注意：详细内容请查看 https://blog.csdn.net/hj7jay/article/details/53389522

6、BOM 之常用 API ？

navigator：window 中封装浏览器属性和配置信息的对象
cookieEnabled：识别浏览器是否启用 cookie，返回值 true/false
userAgent：保存了浏览器名称和版本的字符串
plugins：保存浏览器中所有插件信息的集合，每个 plugin 对象的 name 属性保存了插件的名称
screen：保存显示屏信息的对象
history：保存窗口的历史记录栈
location：指代当前窗口正在访问的 url 地址对象
location.href：保存了当前窗口正在访问的 url 地址，设置 href 属性为新 url，会在当前窗口打开新 url
location.search()：获取 url 上面？后面的参数
location.reload()：刷新当前页面
location.assign(url)：设置当前窗口的新 url

location.reload(true/false):true —— 无论是否更改，都获取更新；false —— 被修改的页面，重新获取，未被修改的页面，从缓冲获取

定时器：让程序按指定时间间隔，自动执行任务，任务是所有定时器的核心

7、prop 和 attr 的区别：

对于 HTML 元素本身就带有的固有属性，在处理时，使用 prop 方法。

对于 HTML 元素我们自己自定义的 DOM 属性，在处理时，使用 attr 方法。

栗子 1：

```
<a href="http://www.baidu.com" target="_self" class="btn">百度</a>
```

这个例子里<a>元素的 DOM 属性有“href、target 和 class”，这些属性就是<a>元素本身就带有的属性，也是 W3C 标准里就包含有这几个属性，或者说在 IDE 里能够智能提示出的属性，这些就叫做固有属性。处理这些属性时，建议使用 prop 方法。

```
<a href="#" id="link1" action="delete">删除</a>
```

这个例子里<a>元素的 DOM 属性有“href、id 和 action”，很明显，前两个是固有属性，而后面一个“action”属性是我们自己自定义上去的，<a>元素本身是没有这个属性的。这种就是自定义的 DOM 属性。处理这些属性时，建议使用 attr 方法。

8、用原生 js 实现一个倒计时项目？

左侧输入为大于 0 的整数，单位为毫秒，右侧显示倒计时的时间，间隔是 1000ms,注意小于 1000ms 的时间的处理。中间是开始的按钮可以重复点击，重复点击后则从新计算

```
<input type="text" id="leftTime" placeholder="请输入 0 以上的整数">
<button id="btn">开始</button>
<input type="text" id="rightTime" placeholder="开始倒计时">
<script>
    var btn = document.getElementById("btn");
    var leftTime = document.getElementById("leftTime");
    var rightTime = document.getElementById("rightTime");
    btn.onclick=function(){
        if(leftTime.value > 0){
            rightTime.value=leftTime.value
        }else{
            rightTime.value = 1
        }
        var cle = setInterval(function(){
            rightTime.value--;
            if(rightTime.value == 0){
```

```
                window.clearInterval(cle)
            }
        },1000)
    }
```

9、JS 的 typeof 返回哪种数据类型?

基本数据类型：String,boolean,Number,Undefined, Null, ；引用数据类型：Object(Array,Date,RegExp,Function)

如何判断某变量是否为数组数据类型？

```
    if(typeof Array.isArray==="undefined")
{
  Array.isArray = function(arg){
        return Object.prototype.toString.call(arg)==="[object Array]"
    };
}
```

补充：null 是一个表示"无"的对象，转为数值时为 0；undefined 是一个表示"无"的原始值，转为数值时为 NaN。

当声明的变量还未被初始化时，变量的默认值为 undefined。null 用来表示尚未存在的对象，常用来表示函数企图返回一个不存在的对象。

undefined 表示"缺少值"，就是此处应该有一个值，但是还没有定义。典型用法是：

（1）变量被声明了，但没有赋值时，就等于 undefined。

（2) 调用函数时，应该提供的参数没有提供，该参数等于 undefined。

（3）对象没有赋值的属性，该属性的值为 undefined。

（4）函数没有返回值时，默认返回 undefined。

null 表示"没有对象"，即该处不应该有值。典型用法是：

（1） 作为函数的参数，表示该函数的参数不是对象。

（2） 作为对象原型链的终点。

10、null 和 undefined 的区别?

null 是一个表示”无”的对象，转为数值时为 0；undefined 是一个表示”无”的原始值，转为数值时为 NaN。

当声明的变量还未被初始化时，变量的默认值为 undefined。

null 用来表示尚未存在的对象，常用来表示函数企图返回一个不存在的对象。

undefined 表示”缺少值”，就是此处应该有一个值，但是还没有定义。

11、new 操作符具体干了什么呢?

1、创建一个空对象，并且 this 变量引用该对象，同时还继承了该函数的原型。
2、属性和方法被加入到 this 引用的对象中。
3、新创建的对象由 this 所引用，并且最后隐式的返回 this 。

11、js 中深拷贝和浅拷贝的区别?

浅拷贝只复制指向某个对象的指针，而不复制对象本身，新旧对象还是共享同一块内存。
深拷贝会另外创造一个一模一样的对象，新对象跟原对象不共享内存，修改新对象不会改到原对象。

深拷贝的方法：1、递归拷贝
2、使用 Object.create()方法
3、jquery 有提供一个$.extend 也可以实现
4、函数库 lodash，也有提供 cloneDeep 用来实现

12、什么是回调地狱，怎么解决回调地狱?

由于回调函数是异步的，每一层的回调函数都需要依赖上一层的回调执行完，所以形成了层层嵌套的关系最终形成了回调地狱。例如：定时器中再写定时器再写定时器，这种就形成了通常所说的回调地狱。

解决 1、避免函数的嵌套
2、模块化开发
3、使用 Promise 解决

13、对 ES6 中 Promise 的理解?

Promise 是异步编程的一种解决方案，比传统的解决方案——回调函数和事件——更合理和更强大。所谓 Promise，简单说就是一个容器，里面保存着某个未来才会结束的事件（通常是一个异步操作）的结果。从语法上说，Promise 是一个对象，从它可以获取异步操作的消息。Promise 提供统一的 API，各种异步操作都可以用同样的方法进行处理。

特点：
1、自己身上有 all、reject、resolve 这几个方法
2、原型上有 then、catch 等方法
3、一旦建立，就无法取消，这是它的缺点

注意具体方法使用请参看：https://www.cnblogs.com/whybxy/p/7645578.html

14、谈谈你对 this 理解?

普通函数中：this->window　　定时器中：this->window　　构造函数中：this->当前实例化的对象
事件处理函数中：this->事件触发对象
在 js 中一般理解就是谁调用这个 this 就指向谁

15、Js 的原型和原型链？原型链的应用?

每个对象都会在其内部初始化一个属性，就是 prototype(原型)，当我们访问一个对象的属性时，如果这个对象内部不存在这个属性，那么他就会去 prototype 里找这个属性，这个 prototype 又会有自己的 prototype，于是就这样一直找下去，也就是我们平时所说的原型链的概念。
应用：
原型链是实现继承的主要方法。

16、几种常用的继承方式?

1、扩展原型对象实现继承

```
function Animal(name,color,say){
    this.name = name;
    this.color = color;
    this.say = function(){
        console.log('喵喵喵');
    }
}
var cat = new Animal('cat','white');// 如果给 Animal 的 prototype 属性上添加个 cry 方法 ，那么实例对象 cat 将也会有 cry 方法
Animal.prototype.cry = function(){
    console.log('呜呜呜');
}
```

2、利用 apply(),和 call 实现继承

```
function person(name,age)
{
    this.name = name;
    this.age = age;
}
function man(name,age){
```

```
        person.apply(this,[name,age]);   //这里就是伪造成 person 的一个事例
    }
    var Man = new man('wozien',12);

    console.log(Man.name);
    console.log(Man.age);
```

3、**组合 call+prototype，最常用的一种方式**

```
        function person(name)
    {
        this.name = name;
    }
    person.prototype.showName = function(){
        return this.name;
    }
    function man(name,age){
        person.call(this,name);
        this.age = age;   //这里就是伪造成 person 的一个事例
    }
    man.prototype = new person();
    man.prototype.showAge = function(){
        return this.age;
    }
    var Man = new man('wozien',12);
    Man.showName(); //wozien
    Man.showAge(); //12
```

17、对 json 的了解?

Json 指的是 js 对象表示法。
1.轻量级的数据交互格式
2.可以形成复杂的嵌套格式
3.解析非常方便
4.易于读写，占用带宽小

18、总结常用的 es6 语法

1、**Let**：声明的变量，只作用于使用了 let 命令的代码块
2、**const :**声明一个常量，声明过后，就不可修改其值(会报错)
3、解构性赋值

3、**箭头函数**：用法的两个明显的优点： * 函数的写法更加简洁 *
箭头函数内部没有自己的 this 对象，而是全部继承外面的，所以内部的 this 就是外层代码块的 this。
4、**字符串模板**：ES6 中允许使用反引号 ` 来创建字符串，此种方法创建的字符串里面可以包含由美元符号加花括号包裹的变量${vraible}
5、**Proxy**：Proxy 可以监听对象身上发生了什么事情，并在这些事情发生后执行一些相应的操作
6、**循环**：for of 实现数组的遍历（以及 keys(),value(),entries()等方法）

注意：详细内容请参考：http://es6.ruanyifeng.com/

19、apply，call，和 bind 有什么区别

三者都可以把一个函数应用到其他对象上，注意不是自身对象．apply,call 是直接执行函数调用，bind 是绑定，执行需要再次调用．apply 和 call 的区别是 apply 接受数组作为参数，而 call 是接受逗号分隔的无限多个参数列表

20、什么是内存泄漏，哪些常见操作会造成内存泄漏

内存泄露是指一块被分配的内存既不能使用，又不能回收，直到浏览器进程结束。在 C++中，因为是手动管理内存，内存泄露是经常出现的事情。而现在流行的 C#和 Java 等语言采用了自动垃圾回收方法管理内存，正常使用的情况下几乎不会发生内存泄露。浏览器中也是采用自动垃圾回收方法管理内存，但由于浏览器垃圾回收方法有 bug，会产生内存泄露。
1 全局变量引起的内存泄漏
2 闭包引起的内存泄漏
3dom 清空或删除时，事件未清除导致的内存泄漏

21、简述 readyonly 与 disabled 的区别

ReadOnly 和 Disabled 的作用是使用户不能够更改表单域中的内容.
但是二者还是有着一些区别的：
1、Readonly 只针对 input(text/password)和 textarea 有效，而 disabled 对于所有的表单元素有效，包括 select,radio,checkbox,button 等。
2、在表单元素使用了 disabled 后，我们将表单以 POST 或者 GET 的方式提交的话，这个元素的值不会被传递出去，而 readonly 会将该值传递出去

22、jq 的在项目中的常用的方法?

使用 JQuery 获取元素
JQ 获取元素的内容：$("xxxx").html()
JQ 中的事件操作
JQ 操作控件属性
JQ 操作 css
ajax 操作
jq 函数级插件--$.fn.extend-添加成员函数/$.extend-添加静态方法

23、$(this)和 this 的区别是什么?

$(this) 返回一个 jQuery 对象，你可以对它调用多个 jQuery 方法，比如用 text() 获取文本，用 val() 获取值等等。而 this 代表当前元素，它是 JavaScript 关键词中的一个，表示上下文中的当前 DOM 元素。

24、jquery 对象和 dom 对象是怎样转换的?

jquery 转 DOM 对象:jQuery 对象是一个数组对象，可以通过[index]的得到相应的 DOM 对象还可以通过 get[index]去得到相应的 DOM 对象。DOM 对象转 jQuery 对象:$(DOM 对象)

25、jq 和 zepto 区别是什么?

相同点：
Zepto 最初是为移动端开发的库，是 jQuery 的轻量级替代品，因为它的 API 和 jQuery 相似，而文件更小。Zepto 最大的优势是它的文件大小，只有 8k 多，是目前功能完备的库中最小的一个，尽管不大，Zepto 所提供的工具足以满足开发程序的需要。大多数在 jQuery 中·常用的 API 和方法 Zepto 都有，Zepto 中还有一些 jQuery 中没有的。另外，因为 Zepto 的 API 大部分都能和 jQuery 兼容，所以用起来极其容易，如果熟悉 jQuery，就能很容易掌握 Zepto。你可用同样的方式重用 jQuery 中的很多方法，也可以方面地把方法串在一起得到更简洁的代码，甚至不用看它的文档。
不同点
针对移动端程序，Zepto 有一些基本的触摸事件可以用来做触摸屏交互（tap 事件、swipe 事件），Zepto 是不支持 IE 浏览器的，这不是 Zepto 的开发者 Thomas Fucks 在跨浏览器问题上犯了迷糊，而是经过了认真考虑后为了降低文件尺寸而做出的决定，就像 jQuery 的团队在 2.0 版中不再支持旧版的 IE（6 7 8）一样。因为 Zepto 使用 jQuery 句法，所以它在文档中建议把 jQuery 作为 IE 上的后备库。那样程序仍能在 IE 中，而其他浏览器则能享受到 Zepto 在文件大小上的优势，然而它们两个的 API 不是完全兼容的，所以使用这种方法时一定要小心，并要做充分的测试。

1、width()和 height()的区别：Zepto 由盒模型(box-sizing)决定，用.width()返回赋值的 width，用.css('width')返回加 border 等的结果；jQuery 会忽略盒模型，始终返回内容区域的宽/高(不包含 padding、border)。

2、offset()的区别：Zepto 返回{top,left,width,height}；jQuery 返回{width,height}。

3、Zepto 无法获取隐藏元素宽高，jQuery 可以。

4、Zepto 中没有为原型定义 extend 方法而 jQuery 有。

26、jQuery.fn 的 init 方法返回的 this 指的是什么对象？为什么要返回 this?

this 执行 init 构造函数自身，其实就是 jQuery 实例对象，返回 this 是为了实现 jQuery 的链式操作

27、jquery.extend 与 jquery.fn.extend 的区别?

jQuery 为开发插件提拱了两个方法，分别是：

1. jQuery.fn.extend();用来扩展 jQuery 实例
2. jQuery.extend();用来扩展 jQuery 对象本身

28、Zepto 的点透问题如何解决?

点透主要是由于两个 div 重合，例如：一个 div 调用 show()，一个 div 调用 hide()；这个时候当点击上面的 div 的时候就会影响到下面的那个 div；
解决办法主要有 2 种：
1.github 上有一个叫做 fastclick 的库，它也能规避移动设备上 click 事件的延迟响应，https://github.com/ftlabs/fastclick
将它用 script 标签引入页面（该库支持 AMD，于是你也可以按照 AMD 规范，用诸如 require.js 的模块加载器引入），并且在 dom ready 时初始化在 body 上，
2.根据分析，如果不引入其它类库，也不想自己按照上述 fastclcik 的思路再开发一套东西，需要 1.一个优先于下面的“divClickUnder”捕获的事件；2.并且通过这个事件阻止掉默认行为（下面的“divClickUnder”对 click 事件的捕获，在 ios 的 safari，click 的捕获被认为和滚屏、点击输入框弹起键盘等一样，是一种浏览器默认行为，即可以被 event.preventDefault()阻止的行为）。

29、jQuery 中.bind() .live() .delegate() .on()的区别

1、bind(type,[data],fn) 为每个匹配元素的特定事件绑定事件处理函数

```
$("a").bind("click",function(){alert("ok");});
```

2、live(type,[data],fn) 给所有匹配的元素附加一个事件处理函数，即使这个元素是以后再添加进来的

```
$("a").live("click",function(){alert("ok");});
```

3、delegate(selector,[type],[data],fn) 指定的元素（属于被选元素的子元素）添加一个或多个事件处理程序，并规定当这些事件发生时运行的函数

```
$("#container").delegate("a","click",function(){alert("ok");})
```

4、on(events,[selector],[data],fn) 在选择元素上绑定一个或多个事件的事件处理函数

差别：
.bind()是直接绑定在元素上
.live()则是通过冒泡的方式来绑定到元素上的。更适合列表类型的，绑定到 document DOM 节点上。和.bind()的优势是支持动态数据。
.delegate()则是更精确的小范围使用事件代理，性能优于.live()
.on()则是最新的 1.9 版本整合了之前的三种方式的新事件绑定机制

30、json 字符串和 json 对象怎么相互转化?

JSON.parse() 从一个字符串中解析出 json 对象
JSON.stringify() 从一个对象中解析出字符串
在前后数据交互中经常使用，必须要记住。

31、http 和 https 的区别是什么?

http 和 https 使用的是完全不同的连接方式,用的端口也不一样,前者是 80,后者是 443。
HTTPS 是以安全为目标的 HTTP 通道，简单讲是 HTTP 的安全版。
由于 https 要还密钥和确认加密算法的过程，所以更安全

32、get 和 post 的区别?

客户端对服务器的请求常用类型主要有四种：
GET（从服务器获取）
POST（向服务器发送请求数据）
PUT（更新）

DELETE（删除）

POST 和 GET 区别
GET 的所有参数全部包装在 URL 中，明文显示，且服务器的访问日志会记录，非常不安全
POST 的 URL 中只有资源路径，不包含参数，参数封装在二进制的数据体中，服务器也不会记录参数，相对安全。所有涉及用户隐私的数据都要用 POST 传输。

GET：不同的浏览器和服务器不同，一般限制在 2~8K 之间，更加常见的是 1k 以内
POST 方法提交的数据比较大，大小靠服务器的设定值限制，PHP 默认是 2M

33、jsonp 的原理，有什么优缺点?

ajax 请求受同源策略影响，不允许进行跨域请求，而 script 标签 src 属性中的链接却可以访问跨域的 js 脚本，利用这个特性，服务端不再返回 JSON 格式的数据，而是返回一段调用某个函数的 js 代码，在 src 中进行了调用，这样实现了跨域。

Jsonp 优点:

完美解决在测试或者开发中获取不同域下的数据,用户传递一个 callback 参数给服务端，然后服务端返回数据时会将这个 callback 参数作为函数名来包裹住 JSON 数据，这样客户端就可以随意定制自己的函数来自动处理返回数据了。

Jsonp 缺点:

Jsonp 只支持 get 请求而不支持 post 请求,也即是说如果想传给后台一个 json 格式的数据,此时问题就来了,浏览器会报一个 http 状态码 415 错误,告诉你请求格式不正确.

34、跨域的几种解决方式

① 通过 jsonp 跨域
② CORS
核心思想：在服务器端通过检查请求头部的 origin，从而决定请求应该成功还是失败。具体的方法是在服务端设置 Response Header 响应头中的 Access-Control-Allow-Origin 为对应的域名，实现了 CORS（跨域资源共享），这里出于在安全性方面的考虑就是尽量不要用 *，但对于一些不重要的数据则随意。

35、同源策略

1. 同源，就是指两个页面具有相同的协议，主机（域名），端口，浏览器会对不同源的脚本或者

文本的访问方式进行的限制

2. 页面中的链接，重定向以及表单提交不会受到同源策略的限制，允许跨域资源嵌入

36、jq 中 ajax 请求的步骤？怎么解决跨域的？

```
$.ajax({
    Type:”get/post”,
    Url:””,
    dataType:”jsonp”,  利用 jsonp 进行跨域
    Data:{},
     async: false，--默认为 true 异步   false 同步
     cache: false,--默认为 true 缓存   false 不缓存
     beforeSend:fucntion(){},
    Success:function(){},
    Errorr:function(){}

  })
```

37、简述 ajax 的交互原理，以及同步和异步的区别

Ajax 的原理简单来说通过 XmlHttpRequest 对象来向服务器发异步请求，从服务器获得数据，然后用 javascript 来操作 DOM 而更新页面。这其中最关键的一步就是从服务器获得请求数据

1、创建 XMLHttpRequest 对象

2、建立连接，设置为 GET 发送：xmlHttp.open("GET", "/ajax/getHint.jsp?q=" + txtValue, true);

3、发送数据：xmlHttp.send();

4、注册回调方法：xmlHttp.onreadystatechange = ajaxCallback;

5、执行回调：

```
function ajaxCallback(){
  //响应就绪时
  if(xmlHttp.readyState == 4 && xmlHttp.status == 200){
     searchInput.value = xmlHttp.responseText;
  }
}
```

同步：同步任务指的是，在主线程上排队执行的任务，只有前一个任务执行完毕，才能执行后一个任务。

异步：不进入主线程、而进入"任务队列"（task queue）的任务，只有等主线程任务执行完毕，"任务队列"开始通知主线程，请求执行任务，该任务才会进入主线程执行。

38、ajax 出现错误怎么调试?

JQuery 使我们在开发 Ajax 应用程序的时候提高了效率，减少了许多兼容性问题，我们在 Ajax 项目中，遇到 ajax 异步获取数据出错怎么办，我们可以通过捕捉 error 事件来获取出错的信息。
发送 error 可能有下面两张引起的，或者其他程序问题，需要我们认真仔细。
1、data:"{}", data 为空也一定要传"{}"；不然返回的是 xml 格式的。并提示 parsererror.
2、parsererror 的异常和 Header 类型也有关系。及编码 header('Content-type: text/html; charset=utf8');

39、ajax 的缺点（答出三点即可）

1、ajax 不支持浏览器 back 按钮。
2、安全问题 AJAX 暴露了与服务器交互的细节。
3、对搜索引擎的支持比较弱。
4、破坏了程序的异常机制。
5、不容易调试
注意详细内容请查看：https://blog.csdn.net/javazilu/article/details/70240288

40、WEB 应用从服务器主动推送 Data 到客户端有那些方式?

html5 websoket
WebSocket 通过 Flash
XHR 长时间连接
XHR Multipart Streaming
不可见的 Iframe
<script>标签的长时间连接(可跨域)

41、请写出至少 5 种常见的 http 状态码以及代表的意义

5 种常见的 http 状态码以及代表的意义如下：
200（OK）：请求已成功，请求所希望的响应头或数据体将随此响应返回。
303（See Other）：告知客户端使用另一个 URL 来获取资源。
400（Bad Request）：请求格式错误。1)语义有误，当前请求无法被服务器理解。除非进行修改，否则客户端不应该重复提交这个请求；2)请求参数有误。
404（Not Found）：请求失败，请求所希望得到的资源未被在服务器上发现。

500（Internal Server Error）：服务器遇到了一个未曾预料的状况，导致了它无法完成对请求的处理。

42、window.onload 和 document ready 的区别?

window.onload 是在 dom 文档树加载完和所有文件加载完之后执行一个函数 Document.ready 原生种没有这个方法，jquery 中有 $().ready(function),在 dom 文档树加载完之后执行一个函数（注意，这里面的文档树加载完不代表全部文件加载完）。
$(document).ready 要比 window.onload 先执行
window.onload 只能出来一次，$(document).ready 可以出现多次

43、什么是事件代理？请写出一个事件代理的示例?

1、当网页中需要触发事件的对象比较多的时候，为了避免内存泄漏，我们把事件委托到其父对象上，借助事件冒泡机制，可以将事件委托到 body，document 等元素上，这样等于一个页面就只有一个事件触发，避免直接把事件添加到多个对象上
2、动态加载的数据是不能直接进行事件的绑定，这时就需要使用事件委托。
例如：$(body).on('click','#div',function(){})

44、IE 和 DOM 事件流的区别

1.执行顺序不一样、2.参数不一样 3.事件加不加 on 4.this 指向问题
注：（1）IE9 以前：attachEvent(“onclick”)、detachEvent(“onclick”)
（2）IE9 开始跟 DOM 事件流是一样的，都是 addEventListener
比较 attachEvent 和 addEventListener：
1、attachEvent 只支持事件冒泡 addEventListener 既支持事件冒泡，也支持事件捕获
2、参数：attachEvent 事件类型需要 on 前缀 addEventListener 事件类型不需要 on 前缀
3、如果使用 attachEvent 对一个元素的目标阶段绑定了多次事件，那么会按照绑定顺序的相反顺序进行触发
如果使用 addEventListener 对一个元素的目标阶段绑定了多次事件，那么会按照绑定顺序进行触发

```
obj.onclick=method1
obj.onclick=method2
obj.onclick=method3
```

如果这样写,那么只有最后绑定的事件,这里是method3会被执行,这个时候我们就不能用onclick这样的写法了,主角改登场了,在IE中我们可以使用attachEvent方法

```
btn1Obj.attachEvent("onclick",method1);
btn1Obj.attachEvent("onclick",method2);
btn1Obj.attachEvent("onclick",method3);
```

使用格式是前面是事件类型,注意的是需要加on,比如onclick,onsubmit,onchange,执行顺序是

method3->method2->method1

可惜这个微软的私人方法,火狐和其他浏览器都不支持,幸运的是他们都支持W3C标准的addEventListener方法

```
btn1Obj.addEventListener("click",method1,false);
btn1Obj.addEventListener("click",method2,false);
btn1Obj.addEventListener("click",method3,false);
```

执行顺序为method1->method2->method3

45、如何阻止事件冒泡和默认事件?

阻止浏览器的默认行为
IE9 之前：window.event.returnValue=false;
IE9+ FF Chrome： e.preventDefault();
停止事件冒泡
IE9+ FF Chrome：e. stopPropagation();
event.canceBubble=true;//ie9 之前
原生 JavaScript 中，return false;只阻止默认行为，不阻止冒泡，jQuery 中的 return false;既阻止默认行为，又阻止冒泡

46、js 延迟加载的方式有哪些?

defer 属性
async 属性
动态创建 DOM 方式
使用 jQuery 的 getScript 方法
使用 setTimeout 延迟方法
让 JS 最后加载

47、你如何优化自己的代码?

A、代码重用
B、避免使用过多的全局变量（命名空间，封闭空间，模块化 mvc..）

C、拆分函数避免函数过于臃肿：单一职责原则
D、将面向过程的编程方式改为使用面向对象编程
E、适当的注释，尤其是一些复杂的业务逻辑或者是计算逻辑，都应该写出这个业务逻辑的具体过程
F、内存管理，尤其是闭包中的变量释放

48、前端开发的优化问题（看雅虎 14 条性能优化原则）

（1） 减少 http 请求次数：CSS Sprites, JS、CSS 源码压缩、图片大小控制合适；网页 Gzip，CDN 托管，data 缓存 ，图片服务器。
（2） 前端模板 JS+数据，减少由于 HTML 标签导致的带宽浪费，前端用变量保存 AJAX 请求结果，每次操作本地变量，不用请求，减少请求次数
（3） 用 innerHTML 代替 DOM 操作，减少 DOM 操作次数，优化 javascript 性能。
（4） 当需要设置的样式很多时设置 className 而不是直接操作 style。
（5） 少用全局变量、缓存 DOM 节点查找的结果。减少 IO 读取操作。
（6） 避免使用 CSS Expression（css 表达式)又称 Dynamic properties(动态属性)。
（7） 图片预加载，将样式表放在顶部，将脚本放在底部 加上时间戳。
（8） 避免在页面的主体布局中使用 table，table 要等其中的内容完全下载之后才会显示出来，显示比 div+css 布局慢。

四、框架问题

1、 Vue 的双向数据绑定原理是什么？

vue.js 是采用数据劫持结合发布者-订阅者模式的方式，通过 Object.defineProperty()来劫持各个属性的 setter，getter，在数据变动时发布消息给订阅者，触发相应的监听回调。
具体步骤：
第一步：需要 observe 的数据对象进行递归遍历，包括子属性对象的属性，都加上 setter 和 getter，这样的话，给这个对象的某个值赋值，就会触发 setter，那么就能监听到了数据变化。
第二步：compile 解析模板指令，将模板中的变量替换成数据，然后初始化渲染页面视图，并将每个指令对应的节点绑定更新函数，添加监听数据的订阅者，一旦数据有变动，收到通知，更新视图。
第三步：Watcher 订阅者是 Observer 和 Compile 之间通信的桥梁，主要做的事情是:1、在自身实例化时往属性订阅器(dep)里面添加自己 2、自身必须有一个 update()方法 3、待属性变动 dep.notice()通知时，能调用自身的 update()方法，并触发 Compile 中绑定的回调，则功成身退。

第四步：MVVM 作为数据绑定的入口，整合 Observer、Compile 和 Watcher 三者，通过 Observer 来监听自己的 model 数据变化，通过 Compile 来解析编译模板指令，最终利用 Watcher 搭起 Observer 和 Compile 之间的通信桥梁，达到数据变化 -> 视图更新；视图交互变化(input) -> 数据 model 变更的双向绑定效果。

2、请详细说下你对 vue 生命周期的理解?

总共分为 8 个阶段创建前/后，载入前/后，更新前/后，销毁前/后。创建前/后：

1、在 beforeCreated 阶段，vue 实例的挂载元素$el 和数据对象 data 都为 undefined，还未初始化。

2、在 created 阶段，vue 实例的数据对象 data 有了，$el 还没有。

3、载入前/后：在 beforeMount 阶段，vue 实例的$el 和 data 都初始化了，但还是挂载之前为虚拟的 dom 节点，data.message 还未替换。

4、在 mounted 阶段，vue 实例挂载完成，data.message 成功渲染。

5、更新前/后：当 data 变化时，会触发 beforeUpdate 和 updated 方法。

6、销毁前/后：在执行 destroy 方法后，对 data 的改变不会再触发周期函数，说明此时 vue 实例已经解除了事件监听以及和 dom 的绑定，但是 dom 结构依然存在

3、请说出 vue.cli 项目中 src 目录每个文件夹和文件的用法?

assets 文件夹是放静态资源；
components 是放组件；
router 是定义路由相关的配置;
view 视图；
app.vue 是一个应用主组件；
main.js 是入口文件

4、怎么定义 vue-router 的动态路由？怎么获取传过来的动态参数?

在 router 目录下的 index.js 文件中，对 path 属性加上/:id。　使用 router 对象的 params.id

5、vue-router 有哪几种导航钩子?

1、全局导航钩子：router.beforeEach(to,from,next)，作用：跳转前进行判断拦截。
2、组件内的钩子；
3、单独路由独享组件

6、scss 是什么？在 vue.cli 中的安装使用步骤是？有哪几大特性?

css 的预编译。
使用步骤：
第一步：用 npm 下三个 loader（sass-loader、css-loader、node-sass）
第二步：在 build 目录找到 webpack.base.config.js，在那个 extends 属性中加一个拓展.scss
第三步：还是在同一个文件，配置一个 module 属性
第四步：然后在组件的 style 标签加上 lang 属性 ，例如：lang="scss"
有哪几大特性:1、可以用变量，例如（$变量名称=值）；2、可以用混合器；3、可以嵌套

7、mint-ui 是什么？怎么使用？说出至少三个组件使用方法?

基于 vue 的前端组件库。npm 安装，然后 import 样式和 js，
vue.use（mintUi）全局引入。
在单个组件局部引入：import {Toast} from 'mint-ui'。
组件一：Toast('登录成功');
组件二：mint-header;
组件三：mint-swiper

8、请说下封装 vue 组件的过程?

首先，组件可以提升整个项目的开发效率。能够把页面抽象成多个相对独立的模块，解决了我们传统项目开发：效率低、难维护、复用性等问题。

然后，使用 Vue.extend 方法创建一个组件，然后使用 Vue.component 方法注册组件。子组件需要数据，可以在 props 中接受定义。而子组件修改好数据后，想把数据传递给父组件。可以采用 emit 方法

9、vue-loader 是什么？使用它的用途有哪些？

解析.vue 文件的一个加载器，跟 template/js/style 转换成 js 模块。
用途：js 可以写 es6、style 样式可以 scss 或 less、template 可以加 jade 等

10、vue.cli 中怎样使用自定义的组件？有遇到过哪些问题吗？

第一步：在 components 目录新建你的组件文件（smithButton.vue），script 一定要 export default
第二步：在需要用的页面（组件）中导入：import smithButton from '../components/smithButton.vue'
第三步：注入到 vue 的子组件的 components 属性上面,components:{smithButton}
第四步：在 template 视图 view 中使用，<smith-button> </smith-button>
问题有：smithButton 命名，使用的时候则 smith-button。

11、说下你对 mvvm 的理解？双向绑定的理解？

mvvm 就是 vm 框架视图、m 模型就是用来定义驱动的数据、v 经过数据改变后的 html、vm 就是用来实现双向绑定

双向绑定:一个变了另外一个跟着变了，例如：视图一个绑定了模型的节点有变化，模型对应的值会跟着变

12、请说下具体使用 vue 的理解？

1、使用 vue 不必担心布局更改和类名重复导致的 js 重写，因为它是靠数据驱动双向绑定，底层是通过 Object.defineProperty() 定义的数据 set、get 函数原理实现。
2、组件化开发，让项目的可拓展性、移植性更好，代码重用性更高，就好像农民工建房子，拿起自己的工具包就可以开工。项目经理坐等收楼就好。
3、单页应用的体验零距离接触安卓原生应用，局部组件更新界面，让用户体验更快速省时。
4、js 的代码无形的规范，团队合作开发代码可阅读性更高。

13、你是怎么认识 vuex 的?

1、vuex 可以理解为一种开发模式或框架。比如 PHP 有 thinkphp，java 有 spring 等。
2、通过状态（数据源）集中管理驱动组件的变化（好比 spring 的 IOC 容器对 bean 进行集中管理）。
3、应用级的状态集中放在 store 中； 改变状态的方式是提交 mutations，这是个同步的事物； 异步逻辑应该封装在 action 中。

14、vuex 有哪几种属性?

有五种，分别是 State、 Getter、Mutation 、Action、 Module

15、vuex 的 State 特性是?

1、Vuex 就是一个仓库，仓库里面放了很多对象。其中 state 就是数据源存放地，对应于与一般 Vue 对象里面的 data。
2、state 里面存放的数据是响应式的，Vue 组件从 store 中读取数据，若是 store 中的数据发生改变，依赖这个数据的组件也会发生更新。
3、它通过 mapState 把全局的 state 和 getters 映射到当前组件的 computed 计算属性中。

16、vuex 的 Getter 特性是?

1、getters 可以对 State 进行计算操作，它就是 Store 的计算属性
2、 虽然在组件内也可以做计算属性，但是 getters 可以在多组件之间复用
3、 如果一个状态只在一个组件内使用，是可以不用 getters

17、vuex 的 Mutation 特性是?

1、Action 类似于 mutation，不同在于：
2、Action 提交的是 mutation，而不是直接变更状态。
3、Action 可以包含任意异步操作

18、<keep-alive>的作用是什么?

<keep-alive></keep-alive> 包裹动态组件时，会缓存不活动的组件实例，主要用于保留组件状态或避免重新渲染。

19、vue 中 ref 的作用是什么?

ref 被用来给 DOM 元素或子组件注册引用信息。引用信息会根据父组件的 $refs 对象进行注册。如果在普通的 DOM 元素上使用，引用信息就是元素; 如果用在子组件上，引用信息就是组件实例
注意：只要想要在 Vue 中直接操作 DOM 元素，就必须用 ref 属性进行注册

20、vue 中组件直接的通信是如何实现的?

组件关系可分为父子组件通信、兄弟组件通信。
1、父组件向子组件：
通过 props 属性来实现
2、子组件向父组件：
子组件用$emit （）来触发事件，父组件用$on（）来监听子组件的事件。
父组件可以直接在子组件的自定义标签上使用 v-on 来监听子组件触发的自定义事件。
3、兄弟之间的通信：

通过实例一个 vue 实例 Bus 作为媒介，要相互通信的兄弟组件之中都引入 Bus，之后通过分别调用 Bus 事件触发和监听来实现组件之间的通信和参数传递。

21、你对 Webpack 的认识?

webpack 是收把项目当作一个整体，通过一个给定的的主文件，webpack 将从这个文件开始找到你的项目的所有依赖文件，使用 loaders 处理它们，最后打包成一个或多个浏览器可识别的 js 文件

22、webpack 中的 entry 是做什么的?

entry: 用来写入口文件，它将是整个依赖关系的根。当我们需要多个入口文件的时候，可以把 entry 写成一个对象。

```
var baseConfig = {
        entry: {
            main: './src/index.js'
```

```
        }
    }
```

23、webpack 中的 output 是做什么的?

output: 即使入口文件有多个，但是只有一个输出配置，如果你定义的入口文件有多个，那么我们需要使用占位符来确保输出文件的唯一性。

```
var baseConfig = {
        entry: {
            main: './src/index.js'
        },
        output: {
            filename: '[name].js',
            path: path.resolve('./build')
        }
    }
module.exports = baseConfig
```

24、webpack 中的 Loader 的作用是什么?

1、实现对不同格式的文件的处理，比如说将 scss 转换为 css，或者 typescript 转化为 js
2、转换这些文件，从而使其能够被添加到依赖图中

这里介绍几个常用的 loader:

babel-loader： 让下一代的 js 文件转换成现代浏览器能够支持的 JS 文件。
babel 有些复杂，所以大多数都会新建一个.babelrc 进行配置
css-loader,style-loader:两个建议配合使用，用来解析 css 文件，能够解释@import,url()如果需要解析 less 就在后面加一个 less-loader
file-loader: 生成的文件名就是文件内容的 MD5 哈希值并会保留所引用资源的原始扩展名
url-loader: 功能类似 file-loader,但是文件大小低于指定的限制时，可以返回一个 DataURL 事实上，在使用 less,scss,stylus 这些的时候，npm 会提示你差什么插件，差什么，你就安上就行了

25、webpack 中的 Plugins 的作用是什么?

loaders 负责的是处理源文件的如 css、jsx，一次处理一个文件。而 plugins 并不是直接操作单个文件，它直接对整个构建过程起作用

1、ExtractTextWebpackPlugin：它会将入口中引用 css 文件，都打包都独立的 css 文件中，而不是内嵌在 js 打包文件中。

2、HtmlWebpackPlugin:依据一个简单的 index.html 模版，生成一个自动引用你打包后的 js 文件的新 index.html。

3、HotModuleReplacementPlugin：它允许你在修改组件代码时，自动刷新实时预览修改后的结果注意永远不要在生产环境中使用 HMR。

26、webpack 中什么是 bundle,什么是 chunk,什么是 module?

1、bundle 是由 webpack 打包出来的文件，
2、chunk 是指 webpack 在进行模块的依赖分析的时候，代码分割出来的代码块。
3、module 是开发中的单个模块。
4、chunk 打包后变成 bundle.

27、webpack-dev-server 和 http 服务器如 nginx 有什么区别?

webpack-dev-server 使用内存来存储 webpack 开发环境下的打包文件，并且可以使用模块热更新，他比传统的 http 服务对开发更加有效。

28、什么是模块热更新?

模块热更新，是 webpack 的一个功能，他可以使得代码通过修改过后，不用刷新浏览器就可以更新。是高级版的自动刷新浏览器。

29、什么是长缓存？在 webpack 中如何做到长缓存优化?

浏览器在用户访问页面的时候，为了加快加载速度，会对用户访问的静态资源进行存储，但是每一次代码升级或者更新，都需要浏览器去下载新的代码，最方便和最简单的更新方式就是引入新的文件名称。在 webpack 中，可以在 output 给出输出的文件制定 chunkhash,并且分离经常更新的代码和框架代码，通过 NamedModulesPlugin 或者 HashedModulesIdsPlugin 使再次打包文件名不变。

30、简单描述下微信小程序的相关文件类型?

微信小程序项目结构主要有一下几个文件类型,如下

1、WXML （WeiXin Markup Language）是框架设计的一套标签语言，结合基础组件、事件系统，可以构建出页面的结构。内部主要是微信自己定义的一套组件。

2、WXSS (WeiXin Style Sheets)是一套样式语言，用于描述 WXML 的组件样式，

3、js 逻辑处理，网络请求

4、json 小程序设置，如页面注册，页面标题及 tabBar。

5、app.json 必须要有这个文件，如果没有这个文件，项目无法运行，因为微信框架把这个作为配置文件入口，整个小程序的全局配置。包括页面注册，网络设置，以及小程序的 window 背景色，配置导航条样式，配置默认标题。

6、app.js 必须要有这个文件，没有也是会报错！但是这个文件创建一下就行 什么都不需要写以后我们可以在这个文件中监听并处理小程序的生命周期函数、声明全局变量。

31、你是怎么封装微信小程序的数据请求的?

1、将所有的接口放在统一的 js 文件中并导出

2、在 app.js 中创建封装请求数据的方法

3、在子页面中调用封装的方法请求数据

32、小程序有哪些参数传值的方法?

1、给 HTML 元素添加 data-*属性来传递我们需要的值，然后通过 e.currentTarget.dataset 或 onload 的 param 参数获取。但 data-名称不能有大写字母和不可以存放对象

2、设置 id 的方法标识来传值通过 e.currentTarget.id 获取设置的 id 的值,然后通过设置全局对象的方式来传递数值

3、在 navigator 中添加参数传值

33、简述微信小程序原理?

1、微信小程序采用 JavaScript、WXML、WXSS 三种技术进行开发，从技术讲和现有的前端开发差不多，但深入挖掘的话却又有所不同。

2、JavaScript：首先 JavaScript 的代码是运行在微信 App 中的，并不是运行在浏览器中，因此一些 H5 技术的应用，需要微信 App 提供对应的 API 支持，而这限制住了 H5 技术的应用，且其不能称为严格的 H5，可以称其为伪 H5，同理，微信提供的独有的某些 API，H5 也不支持或支持的不是特别好。

3、WXML：WXML 微信自己基于 XML 语法开发的，因此开发时，只能使用微信提供的现有标签，HTML 的标签是无法使用的。

4、WXSS：WXSS 具有 CSS 的大部分特性，但并不是所有的都支持，而且支持哪些，不支持哪些并没有详细的文档。

5、微信的架构，是数据驱动的架构模式，它的 UI 和数据是分离的，所有的页面更新，都需要通过对数据的更改来实现。

6、小程序分为两个部分 webview 和 appService。其中 webview 主要用来展现 UI，appService 有来处理业务逻辑、数据及接口调用。它们在两个进程中运行，通过系统层 JSBridge 实现通信，实现 UI 的渲染、事件的处理。

34、小程序的双向绑定和 vue 哪里不一样?

小程序直接 this.data 的属性是不可以同步到视图的，必须调用：

小程序：

```
Page({
  data: {
    items: []
  },
  onLoad: function(options) {
    this.setData({
      items: [1,2,3]
    })
  }})
```

Vue：

```
new Vue({
  data: {
    items: []
  },
  mounted () {
    this.items = [1, 2, 3]
  }})
```

35、webview 中的页面怎么跳回小程序中?

<web-view/>网页中可使用 JSSDK 1.3.2 提供的接口返回小程序页面。

例如：wx.miniProgram.navigateTo({url: '/path/to/page'})

36、小程序关联微信公众号如何确定用户的唯一性?

使用 wx.getUserInfo 方法 withCredentials 为 true 时可获取 encryptedData，里面有 union_id。后端需要进行对称解密。

37、小程序如何实现下拉刷新?

用 view 代替 scroll-view,,设置 onPullDownRefresh 函数实现。

1、在 json 文件中配置 enablePullDownRefresh 为 true(app.json 中在 window 中设置 enablePullDownRefresh,此效果作用于全局)。

2、在 js 文件中实现 onPullDownRefresh 方法,在网络请求完成后调用 wx.stopPullDownRefresh()来结束下拉刷新。

38、小程序调用后台接口遇到哪些问题?

1、数据的大小有限制，超过范围会直接导致整个小程序崩溃，除非重启小程序；

2、小程序不可以直接渲染文章内容页这类型的 html 文本内容，若需显示要借住插件，但插件渲染会导致页面加载变慢，所以最好在后台对文章内容的 html 进行过滤，后台直接处理批量替换 p 标签 div 标签为 view 标签，然后其它的标签让插件来做，减轻前端的时间。

39、小程序的 wxss 和 css 有哪些不一样的地方?

1、wxss 的图片引入需使用外链地址。

2、没有 Body，样式可直接使用 import 导入。

40、分析下微信小程序的优劣势

优势：

1、无需下载，通过搜索和扫一扫就可以打开。

2、良好的用户体验：打开速度快。

3、开发成本要比 App 要低。

4、安卓上可以添加到桌面，与原生 App 差不多。
5、为用户提供良好的安全保障。小程序的发布，微信拥有一套严格的审查流程， 不能通过审查的小程序是无法发布到线上的。

劣势：
1、限制较多。页面大小不能超过 1M。不能打开超过 5 个层级的页面。
2、样式单一。小程序的部分组件已经是成型的了，样式不可以修改。例如：幻灯片、导航。
3、推广面窄，不能分享朋友圈，只能通过分享给朋友，附近小程序推广。其中附近小程序也受到微信的限制。
4、依托于微信，无法开发后台管理功能。

41、Angular 中是怎么实现数据双向绑定的?

Angular 实现了双向绑定机制。所谓的双向绑定，无非是从界面的操作能实时反映到数据，数据的变更能实时展现到界面。即数据模型（Module）和视图（View）之间的双向绑定。

脏检查机制。双向数据绑定是 AngularJS 的核心机制之一。当 view 中有任何数据变化时，会更新到 model ，当 model 中数据有变化时，view 也会同步更新，显然，这需要一个监控。
原理就是，Angular 在 scope 模型上设置了一个 监听队列，用来监听数据变化并更新 view 。每次绑定一个东西到 view 上时 AngularJS 就会往 $watch 队列里插入一条 $watch，用来检测它监视的 model 里是否有变化的东西。当浏览器接收到可以被 angular context 处理的事件时，$digest 循环就会触发，遍历所有的 $watch，最后更新 dom。

42、Angular 中 ng-show/ng-hide 与 ng-if 可以用来作什么？有什么区别?

angularJS 中的 ng-show、ng-hide、ng-if 指令都可以用来控制 dom 元素的显示或隐藏。ng-show 和 ng-hide 根据所给表达式的值来显示或隐藏 HTML 元素。当赋值给 ng-show 指令的值为 false 时元素会被隐藏，值为 true 时元素会显示。ng-hide 功能类似，使用方式相反。元素的显示或隐藏是通过改变 CSS 的 display 属性值来实现的。
ng-if 指令可以根据表达式的值在 DOM 中生成或移除一个元素。如果赋值给 ng-if 的表达式的值是 false，那对应的元素将会从 DOM 中移除，否则生成一个新的元素插入 DOM 中。ng-if 同 no-show 和 ng-hide 指令最本质的区别是，它不是通过 CSS 显示或隐藏 DOM 节点，而是删除或者新增结点。

43、angular 中$rootScope 和 scope 有什么区别?

scope 是 html 和单个 controller 之间的桥梁，数据绑定就靠他了。rootscope 是各个 controller 中 scope 的桥梁。用 rootscope 定义的值，可以在各个 controller 中使用。

44、angular 中应用是怎么启动的，有哪几种方式，且怎么实现的，请写出实现代码？

① angular 启动的主要步骤：

（1）匿名自执行函数，保证 angular.js 加载完后，立即执行其中的代码。

（2）通过 window.angular.bootstrap 检测是否 angular 被多次启动/angular.js 被多次加载。多次加载，耗时耗力，不值得提倡。在 window 对象上绑定一个属性，这就是个全局属性，全局的嘛，就能用来判断是否多次加 载了（自己写 lib 也可以好好利用 window 属性哦）。

（3）绑定 jQuery，即 bindJQuery()：如果用户导入了 jQuery，就用这个导入的外部 jQuery。否则用 angular 内 置的 jQLite。看来 jQuery 已经成为不可或缺的神物。

（4）发布 ng 的 API，即 publishExternalAPI()。这样我们才能用 angular.module()之类的方法。

（5）查找 ng-app，即 angularInit()。ng 的边界就是 ng-app。

② 启动方式有自启动和手启动

自启：angular.moudle(“myApp”).run(function($rootScope){ console.log($rootScope)});

手启：angular.moudle(“myApp”). run(function($rootScope){ console.log($rootScope)});

angular.bootstrap().bootstrap(DOM 对象，[‘myApp’]);

注：手动启动的时候，不需要在页面中指定 ng-app 指令；一般使用自启动。对于一个页面中存在多个 ng-app 的情况，第一个 ng-app 会自动启动，其余的需要手动启动才可以，否则，不生效而且会报错，可以忽略这种方法。

45、angular 自定义指令中 restrict 可以怎么样设置，分别是什么意思？Scope 中@，=，& 有什么区别？怎么实现与父级别作用进行交互？

restrict 属性，来决定这个指令是作为标签（E）、属性（A）、属性值（C）、还是注释（M）。

Scope

1 false（默认值）：直接使用父 scope。比较“危险”。

2 true：继承父 scope

3 {}可以理解成指令内部并没有一个新的 scope，它和指令以外的代码共享同一个 scope。

@：单向绑定，外部 scope 能够影响内部 scope，但反过来不成立

=：双向绑定，外部 scope 和内部 scope 的 model 能够相互改变

&：把内部 scope 的函数的返回值和外部 scope 的任何属性绑定起来

46、不同模块之间可以使用哪些方式实现交互？可以怎么实现优化 angular 性能？以及你对在 angular 中使用的 jquery 的个人看法

不同模块间的通信

1 采用 SharedService，利用共享的公共服务来实现数据交换。 AngularJS 中的 Service 被设计成单例的，这为这一方案，提供来底层的实现可行性，这一方案也是被广泛采用的。

2 利用 AngularJS 提供的事件机制，$rootScope.$broadcast/ $scope.$emit 配合 $on 方法实现。该方案的实现具备一定的局限性，比如：$emit 方法只能向上传递事件，而不能实现向下的事件传播。但是进行合理的搭配组合已经基本够用了。

3 利用浏览器的 SessionStorage 或者 LocalStorage 进行数据交换。 由于浏览器提供的这两类本地存储方案都提供了相应的 storage 事件，所以我们也可以使用该方案进行数据交换。使用该方案是应该注意及时清理无关数据。

优化性能：

1 官方提倡的，关闭 debug,$compileProvider

```
myApp.config(function ($compileProvider) {
        $compileProvider.debugInfoEnabled(false);
 });
```

2 使用一次绑定表达式即{{::yourModel}}

3 减少 watcher 数量

4 在无限滚动加载中避免使用 ng-repeat,关于解决方法可以参考这篇文章

5 使用性能测试的小工具去挖掘你的 angular 性能问题，我们可以使用简单的 console.time()也可以借助开发者工具以及 Batarang

```
console.time("TimerName");
codeconsole.timeEnd("TimerName");
```

angularJS 的用法更像是面向对象的编程模式。它会要求你定义一个 view model，然后所有的页面变化，是通过改变这个 view model 来实现的。一旦搭建好了这个框框之后，页面的操作会非常爽，完全不用考虑具体页面怎么变化，怎么去操作 dom 的问题，浏览器兼容性的问题 angularJS 也一并帮你解决了。

而 jquery 的用法，更像是面向过程的编程模式。你在用 jquery 写具体代码的时候，你需要先考虑到你要实现的场景是怎么样的，然后把具体的实现过程用代码表示出来，而且这种实现往往是单向的。也就是说下次你要再进行页面改变的时候，又得用代码实现一边（而 angularJS 则只要吧 view model 的数据还原下就可以了）。

47、关于 angular.js 中，ng-repeat 迭代数组的时候，如果数组中有相同值，会有什么问题，如何解决；

会提示 Duplicates in a repeater are not allowed. 加 track by $index 可解决。当然，也可以 trace by 任何一个普通的值，只要能唯一性标识数组中的每一项即可（建立 dom 和数据之间的关联）。

48、开发 one page 页面，需要有哪些注意事项；

1 事件执行效率

大量实现给未来 dom 节点绑定事件，这种绑定的很多封装实现，是利用往 document 绑定事件，然后冒泡到未来节点来的原理。当页面切换后，这些事件并未清理，页面会给新页面再执行类似的事件绑定。当 document 上点击后，执行的冒泡判断会越来越多，不仅影响到冒泡到未来 dom 节点模式的事件响应速度（因为冒泡判断），还会影响到精确绑定到 dom 节点事件的响应速度（因为执行完事件后，如果没有返回 false，会继续冒泡到 document），导致页面的点击响应越来越慢。

解决这类问题，可以往未来 dom 节点中已存在的父节点绑定事件实现冒泡，或将未来 dom 节点的精确绑定事件触发延迟执行。

2 第三方插件重复渲染报错

页面切换，将原理的 uploadify flash 从 dom 中清除，后面再次进入需要显示 uploadify 的页面时，因为没有刷新页面，uploadify 再次渲染会报 id 冲突错误。

类似问题可以在页面切换前或新页面渲染前销毁插件对象。

3 会话状态变化

单页应用不刷新页面，很多交互操作只是发出 ajax 请求。如果同一浏览器打开两个窗口，在一个窗口进行了帐号退出操作或切换了帐号，另一窗口继续进行操作，因为不会刷新页面，而且基本都是发 ajax 请求获取数据，这样就会出现页面显示的是切换前的帐号数据，而 ajax 获取到的是切换后的帐号的数据。

可以在每次 ajax 请求，都缓存会话 id，然后 ajax 请求前，比较缓存的会话 id 和 cookie 中的会话 id 是否一致，如果不一致，则表示已经切换帐号，可以刷新当前页面。

49、Angular 和 vue 的优缺点，你是怎么看待的

① Vue：

优点：
1. 简单：官方文档很清晰，比 Angular 简单易学。
2. 快速：异步批处理方式更新 DOM。
3. 组合：用解耦的、可复用的组件组合你的应用程序。
4. 紧凑：~18kb min+gzip，且无依赖。
5. 强大：表达式 & 无需声明依赖的可推导属性 (computed properties)。
6. 对模块友好：可以通过 NPM、Bower 或 Duo 安装，不强迫你所有的代码都遵循 Angular

的各种 规定，使用场景更加灵活。

缺点： 1. 新生儿：Vue.js 是一个新的项目，没有 angular 那么成熟。

2. 影响度不是很大：google 了一下，有关于 Vue.js 多样性或者说丰富性少于其他一些有名库。

3. 不支持 IE8：

②angularJS：

优点：1. 模板功能强大丰富，自带了极其丰富的 angular 指令。

2. 是一个比较完善的前端框架，包含服务，模板，数据双向绑定，模块化，路由，过滤器， 依赖注入等所有功能；

3. 自定义指令，自定义指令后可以在项目中多次使用。

4. ng 模块化比较大胆的引入了 Java 的一些东西（依赖注入），能够很容易的写出可复用的 代码，对于敏捷开发的团队来说非常有帮助。

5. angularjs 是互联网巨人谷歌开发，这也意味着他有一个坚实的基础和社区支持。

缺点：1. angular 入门很容易 但深入后概念很多，学习中较难理解.

2. 文档例子非常少，官方的文档基本只写了 api，一个例子都没有，很多时候具体怎么用都是 google 来的，或直接问 misko,angular 的作者.

3. 对 IE6/7 兼容不算特别好，就是可以用 jQuery 自己手写代码解决一些.

4. 指令的应用的最佳实践教程少，angular 其实很灵活，如果不看一些作者的使用原则，很容易 写出 四不像的代码，例如 js 中还是像 jQuery 的思想有很多 dom 操作.

5. DI 依赖注入 如果代码压缩需要显示声明.

。

50、RequireJS 和 seaJs 的区别

相同之处

RequireJS 和 SeaJS 都是模块加载器，倡导的是一种模块化开发理念，核心价值是让 JavaScript 的模块化开发变得更简单自然。

不同之处

两者的区别如下：

② 定位有差异。RequireJS 想成为浏览器端的模块加载器，同时也想成为 Rhino / Node 等环境的模块加载器。SeaJS 则专注于 Web 浏览器端，同时通过 Node 扩展的方式可以很方便跑在 Node 服务器端。

③ 遵循的规范不同。RequireJS 遵循的是 AMD（异步模块定义）规范，SeaJS 遵循的是 CMD （通用模块定义）规范。规范的不同，导致了两者 API 的不同。SeaJS 更简洁优雅，更贴近 CommonJS Modules/1.1 和 Node Modules 规范。

④ 社区理念有差异。RequireJS 在尝试让第三方类库修改自身来支持 RequireJS，目前只有少数社区采纳。SeaJS 不强推，采用自主封装的方式来“海纳百川”，目前已有较成熟的封装策略。

⑤ 代码质量有差异。RequireJS 是没有明显的 bug，SeaJS 是明显没有 bug。

⑥ 对调试等的支持有差异。SeaJS 通过插件，可以实现 Fiddler 中自动映射的功能，还可以实现自动 combo 等功能，非常方便。RequireJS 无这方面的支持。

⑦ 插件机制不同。RequireJS 采取的是在源码中预留接口的形式，源码中留有为插件而写的代码。

SeaJS 采取的插件机制则与 javascript 语言以及 Node 的方式一致：开放自身，让插件开发者可直接访问或修改，从而非常灵活，可以实现各种类型的插件。

51、Angular 的架构思想

1、MVVM
指 Model View Controller
2、模块化和依赖注入
模块用于单独的逻辑表示服务，控制器，应用程序等，并保持代码的整洁。我们在单独的 js 文件中定义的模块，并将其命名为按照 module.js 文件形式
模块化的好处
1 增加了模块的可重用性
2 通过定义模块，实现加载顺序的自定义
3 在单元测试中，不必加载所有的内容
依赖注入是一种软件设计模式,用于处理如何让程序获得其依赖(对象的)引用
3、双向数据绑定
一但建立双向绑定，使用者输入，会由 Angular 自动传到一个变量中，再自动读到所有绑到它的内容，更新它，效果上就是立即的资料同步， 在程式码中修改变量，也会直接反应到呈现的外观上。
4、指令
指令是 DOM 元素上的标记，使元素拥有特定的行为。举例来说，静态的 HTML 不知道如何来创建和展现一个日期选择器控件。让 HTML 能识别这个语法，我们需要使用指令。指令通过某种方法来创建一个能够支持日期选择的元素。我们会循序渐进地介绍这是如何实现的。 如果你写过 AngularJS 的应用，那么你一定已经使用过指令，不管你有没有意识到。你肯定已经用过简单的指令，比如 ng-model, ng-repeat, ng-show 等。这些指令都赋予 DOM 元素特定的行为。例如，ng-repeat 重复特定的元素，ng-show 有条件地显示一个元素。如果你想让一个元素支持拖拽，你也需要创建一个指令来实现它。指令背后基本的想法很简单。它通过对元素绑定事件监听或者改变 DOM 而使 HTML 拥有真实的交互性。

52、谈谈模块化的理解

所谓的模块化开发就是封装细节，提供使用接口，彼此之间互不影响，每个模块都是实现某一特定的功能。模块化开发的基础就是函数

53、angular 中 service 服务三种方式是什么？区别是什么？

Factory ：把 service 的方法和数据放在一个对象里，并返回这个对象。
Service：通过构造函数方式创建 service，返回一个实例化对象。
Provider：创建一个可通过 config 配置的 service，$get 中返回的，就是用 factory 创建 service 的

内容

从底层实现上来看，service 调用了 factory，返回其实例；factory 调用了 provider，返回其 $get 中定义的内容。factory 和 service 功能类似，只不过 factory 是普通 function，可以返回任何东西（return 的都可以被访问，所以那些私有变量怎么写，你懂的）；service 是构造器，可以不返回（绑定到 this 的都可以被访问）；provider 是加强版 factory，返回一个可配置的 factory。

54、列举 React 的生命周期

1. 实例化：
getDefaultProps
getInitialState
componentWillMount
render
componentDidMount
2) 更新
componentWillReceiveProps
shouldComponentUpdate
componentWillUpdate
render
componentDidUpdate
3) 销毁&清理期
componentWillUnmount

55、react 中 state 和 prop 的区别，改变 state 将对页面有什么影响

props 放初始化数据，是一个父组件传递给子组件的数据流，这个数据流可以一直传递到子孙组件，组件本身不能修改自己的 props。而 state 代表的是一个组件内部自身的状态，改变一个组件自身状态，从语义上来说，就是这个组件内部已经发生变化，有可能需要对此组件以及组件所包含的子孙组件进行重渲染

56、在 react 中如何获取真实 dom 节点

React.findDOMNode(component)，在已经装载的组件中调用获取真实节点，在 render 中调用会报错。

57、props.children 的作用是什么，如何使用?

this.props.children 属性，它表示组件的所有子节点

可以使用以下方法遍历：
使用方法：

```
React.Children.map(this.props.children, function (child) {
    return <li>{child}</li>;
})
```

其他方法

```
this.props.children.forEach(function (child) {
    return <li>{child}</li>
})
```

58、如何为 react 组件设置样式

可以使用 style 或 className

```
style={{border:"1px solid #333333", background:"red"}}
className="input"
```

59、实现简单的 react 组件

```
var ReactCmponent = React.createClass({
  handleClick: function() {
    alert(this.refs.input.value);
  },
  render: function() {
    return (
      <div>
        <input type="text" ref="input" />
        <input type="button" value="Get Input Value" onClick={this.handleClick} style={{border:"1px solid #333333", background:"red"}} className="input"
/>
      </div>
    );
  }
});
```

```
ReactDOM.render(
  <ReactCmponent />,
  document.getElementById('container')
);
```

60、react 中 value 和 defaultValue 属性的区别是什么

defaultValue 在 react 为 form 组件里设置的初始值，组件成为非受控组件，设置 defaultValue，可以在页面上自动改变控件的值。
如果设置了 value 值，组件变成受控组件，当在页面上改变控件的值的时候，并不会生效，要响应用户的输入值，需为其绑定 onChange 事件，改变组件对应的 state，让页面重新渲染。

```
getInitialState: function() {
        return {value: 'Hello!'};
},
handleChange: function(event) {
        this.setState({value: event.target.value});
},
render: function() {
        return (<input   type="text" value={this.state.value} onChange={this.handleChange}
    />
  );
}
```

61、Bootstrap 有什么好处，为什么要用 bootstrap，什么情况用比较合适

Bootstrap 是一个用于快速开发 Web 应用程序和网站的前端框架。Bootstrap 是基于 HTML、CSS、JAVASCRIPT 的。Bootstrap 是由 Twitter 的 Mark Otto 和 Jacob Thornton 开发的。Bootstrap 是 2011 年八月在 GitHub 上发布的开源产品。

为什么使用 Bootstrap？
移动设备优先：自 Bootstrap 3 起，框架包含了贯穿于整个库的移动设备优先的样式。
浏览器支持：所有的主流浏览器都支持 Bootstrap。
容易上手：只要您具备 HTML 和 CSS 的基础知识，您就可以开始学习 Bootstrap。
响应式设计：Bootstrap 的响应式 CSS 能够自适应于台式机、平板电脑和手机。它为开发人员创建接口提供了一个简洁统一的解决方案。
它包含了功能强大的内置组件，易于定制。
它还提供了基于 Web 的定制。
它是开源的。

61、什么是 Bootstrap 网格系统（Grid System）？

Bootstrap 包含了一个响应式的、移动设备优先的、不固定的网格系统，可以随着设备或视口大小的增加而适当地扩展到 12 列。它包含了用于简单的布局选项的预定义类，也包含了用于生成更多语义布局的功能强大的混合类。
响应式网格系统随着屏幕或视口（viewport）尺寸的增加，系统会自动分为最多 12 列。

63、Bootstrap 网格系统（Grid System）的工作原理?

1、行必须放置在 .container class 内，以便获得适当的对齐（alignment）和内边距（padding）。
2、使用行来创建列的水平组。
3、内容应该放置在列内，且唯有列可以是行的直接子元素。
4、预定义的网格类，比如 .row 和 .col-xs-4，可用于快速创建网格布局。LESS 混合类可用于更多语义布局。
5、列通过内边距（padding）来创建列内容之间的间隙。该内边距是通过 .rows 上的外边距（margin）取负，表示第一列和最后一列的行偏移。
6、网格系统是通过指定您想要横跨的十二个可用的列来创建的。例如，要创建三个相等的列，则使用三个 .col-xs-4。

64、Bootstrap 网格系统列与列之间的间隙宽度是多少?

间隙宽度为 30px（一个列的每边分别是 15px）。

65、使用 Bootstrap 创建垂直表单的基本步骤?

1、向父<form>元素添加 role="form"；
2、把标签和控件放在一个带有 class="form-group"的 中，这是获取最佳间距所必需的；
3、向所有的文本元素<input>、<textarea>、<select>添加 class="form-control"。

66、使用 Bootstrap 创建水平表单的基本步骤?

1、向父<form>元素添加 class="form-horizontal"；

2、把标签和控件放在一个带有 class="form-group"的 中；
3、向标签添加 class="control-label"。

67、对于各类尺寸的设备，Bootstrap 设置的 class 前缀分别是什么?

超小设备手机（<768px）：.col-xs-
小型设备平板电脑（>=768px）：.col-sm-
中型设备台式电脑（>=992px）：.col-md-
大型设备台式电脑（>=1200px）：.col-lg-

68、为什么使用 node?

总结起来 node 有以下几个特点:简单强大，轻量可扩展．简单体现在 node 使用的是 javascript,json 来进行编码，人人都会；强大体现在非阻塞 IO,可以适应分块传输数据，较慢的网络环境，尤其擅长高并发访问；轻量体现在 node 本身既是代码，又是服务器，前后端使用统一语言;可扩展体现在可以轻松应对多实例，多服务器架构，同时有海量的第三方应用组件．

69、Node 有哪些全局对象?

答：process, console, Buffer 和 exports

70、Node.js 的适用场景?

1)、实时应用：如在线聊天，实时通知推送等等（如 socket.io）
2)、分布式应用：通过高效的并行 I/O 使用已有的数据
3)、工具类应用：海量的工具，小到前端压缩部署（如 grunt），大到桌面图形界面应用程序
4)、游戏类应用：游戏领域对实时和并发有很高的要求（如网易的 pomelo 框架）
5)、利用稳定接口提升 Web 渲染能力
6)、前后端编程语言环境统一：前端开发人员可以非常快速地切入到服务器端的开发（如著名的纯 Javascript 全栈式 MEAN 架构）

五、常见的兼容性问题

PC 端常见的兼容性问题

1、IE8 下面的 png 图片无法正常显示?

原因：打开调试面板，你会发现 IE8 浏览器把 PNG 格式的 img 解析成了 span 标签，导致图无法显示。

解决方案：在样式里面对 span 设置宽高和 display:inline-block;即可。

2、rgba 不支持 IE8?

解决方案:可以用 opacity

opacity:0.7;/*FF chrome safari opera*/

filter:alpha(opacity:70);/*用了 ie 滤镜,可以兼容 ie*/

但是,需要注意的是,opacity 会影响里面元素的透明度

3、C3 的新属性?

当一些 CSS3 样式语法还存在波动时，它们提供针对浏览器的前缀。现在主要流行的浏览器内核主要有：

Trident 内核：主要代表为 IE 浏览器

Gecko 内核：主要代表为 Firefox

Presto 内核：主要代表为 Opera

Webkit 内核：产要代表为 Chrome 和 Safari

而这些不同内核的浏览器，CSS3 属性（部分需要添加前缀的属性）对应需要添加不同的前缀，也将其称之为浏览器的私有前缀，添加上私有前缀之后的 CSS3 属性可以说是对应浏览器的私有属性：

Trident 内核：前缀为-ms

Gecko 内核：前缀为-moz

Presto 内核：前缀为-o

Webkit 内核：前缀为-webkit

4、document.form.item 问题

问题：
代码中存在 document.formName.item("itemName") 这样的语句，不能在 FF 下运行
解决方法：
改用 document.formName.elements["elementName"]

5、集合类对象问题

问题：
代码中许多集合类对象取用时使用()，IE 能接受，FF 不能
解决方法：
改用 [] 作为下标运算，例：
document.getElementsByName("inputName")(1) 改为 document.getElementsByName("inputName")[1]

6. window.event

问题：
使用 window.event 无法在 FF 上运行
解决方法：
FF 的 event 只能在事件发生的现场使用，此问题暂无法解决。可以把 event 传到函数里变通解决：

```
onMouseMove = "functionName(event)"
function functionName (e) {
    e = e || window.event;
    .....
}
```

7. HTML 对象的 id 作为对象名的问题

问题：
在 IE 中，HTML 对象的 ID 可以作为 document 的下属对象变量名直接使用，在 FF 中不能
解决方法：
使用对象变量时全部用标准的 getElementById("idName")

8. 用 idName 字符串取得对象的问题

问题：
在 IE 中，利用 eval_r("idName") 可以取得 id 为 idName 的 HTML 对象，在 FF 中不能
解决方法：
用 getElementById("idName") 代替 eval_r("idName")

9. 变量名与某 HTML 对象 id 相同的问题

问题：
在 FF 中，因为对象 id 不作为 HTML 对象的名称，所以可以使用与 HTML 对象 id 相同的变量名，IE 中不能
解决方法：
在声明变量时，一律加上 var ，以避免歧义，这样在 IE 中亦可正常运行
最好不要取与 HTML 对象 id 相同的变量名，以减少错误

10. event.x 与 event.y 问题

问题：
在 IE 中，event 对象有 x,y 属性，FF 中没有
解决方法：
在 FF 中，与 event.x 等效的是 event.pageX ，但 event.pageX IE 中没有
故采用 event.clientX 代替 event.x ，在 IE 中也有这个变量
event.clientX 与 event.pageX 有微妙的差别，就是滚动条
要完全一样，可以这样：
mX = event.x ? event.x : event.pageX;
然后用 mX 代替 event.x

11. 取得元素的属性

在 FF 中，自己定义的属性必须 getAttribute() 取得

10. 在 FF 中没有 parentElement，parement.children 而用 parentNode，parentNode.childNodes

问题：
childNodes 的下标的含义在 IE 和 FF 中不同，FF 的 childNodes 中会插入空白文本节点
解决方法：
可以通过 node.getElementsByTagName_r() 来回避这个问题
问题：
当 html 中节点缺失时，IE 和 FF 对 parentNode 的解释不同，例如：
<form>
<table>
<input/>
</table>
</form>
FF 中 input.parentNode 的值为 form，而 IE 中 input.parentNode 的值为空节点
问题：
FF 中节点自己没有 removeNode 方法
解决方法：
必须使用如下方法 node.parentNode.removeChild(node)

12. const 问题

问题：
在 IE 中不能使用 const 关键字
解决方法：
以 var 代替

13. body 对象

FF 的 body 在 body 标签没有被浏览器完全读入之前就存在，而 IE 则必须在 body 完全被读入之后才存在
这会产生在 IE 下，文档没有载入完时，在 body 上 appendChild 会出现空白页面的问题
解决方法：
一切在 body 上插入节点的动作，全部在 onload 后进行

14. url encoding

问题：
一般 FF 无法识别 js 中的&

解决方法：
在 js 中如果书写 url 就直接写&不要写&

15. nodeName 和 tagName 问题

问题：
在 FF 中，所有节点均有 nodeName 值，但 textNode 没有 tagName 值，在 IE 中，nodeName 的使用有问题
解决方法：
使用 tagName，但应检测其是否为空

16. 元素属性

IE 下 input.type 属性为只读，但是 FF 下可以修改

16. document.getElementsByName() 和 document.all[name] 的问题
问题：
在 IE 中，getElementsByName()、document.all[name] 均不能用来取得 div 元素
是否还有其它不能取的元素还不知道（这个问题还有争议，还在研究中）

17. 调用子框架或者其它框架中的元素的问题

在 IE 中，可以用如下方法来取得子元素中的值
document.getElementByIdx_x("frameName").(document.)elementName
window.frames["frameName"].elementName
在 FF 中则需要改成如下形式来执行，与 IE 兼容：
window.frames["frameName"].contentWindow.document.elementName
window.frames["frameName"].document.elementName

18. 对象宽高赋值问题

问题：
FireFox 中类似 obj.style.height = imgObj.height 的语句无效

解决方法：
统一使用 obj.style.height = imgObj.height + "px";

19. innerText 的问题

问题：
innerText 在 IE 中能正常工作，但是 innerText 在 FireFox 中却不行
解决方法：
在非 IE 浏览器中使用 textContent 代替 innerText

20. event.srcElement 和 event.toElement 问题

问题：
IE 下，even 对象有 srcElement 属性，但是没有 target 属性；Firefox 下，even 对象有 target 属性，但是没有 srcElement 属性
解决方法：

```
var source = e.target || e.srcElement;
var target = e.relatedTarget || e.toElement;
```

21. 禁止选取网页内容

问题：
FF 需要用 CSS 禁止，IE 用 JS 禁止
解决方法：
IE: obj.onselectstart = function() {return false;}
FF: -moz-user-select:none;

22. 捕获事件

问题：
FF 没有 setCapture()、releaseCapture()方法
解决方法：
IE:

```
obj.setCapture();
obj.releaseCapture();
```

FF:

```
window.captureEvents(Event.MOUSEMOVE|Event.MOUSEUP);
```

```
window.releaseEvents(Event.MOUSEMOVE|Event.MOUSEUP);
if (!window.captureEvents) {
        o.setCapture();
}else {
        window.captureEvents(Event.MOUSEMOVE|Event.MOUSEUP);
}
if (!window.captureEvents) {
        o.releaseCapture();
}else {
        window.releaseEvents(Event.MOUSEMOVE|Event.MOUSEUP);
}
```

移动端常见的兼容性问题

1、html5 调用安卓或者 ios 的拨号功能

html5 提供了自动调用拨号的标签，只要在 a 标签的 href 中添加 tel:就可以了。
如下：
<ahref="tel:4008106999,1034">400-810-6999 转 1034
拨打手机直接如下
点击拨打 15677776767

2、上下拉动滚动条时卡顿、慢

body {-webkit-overflow-scrolling: touch; overflow-scrolling: touch;}
Android3+和 iOS5+支持 CSS3 的新属性为 overflow-scrolling

3、圆角 bug

某些 Android 手机圆角失效
background-clip: padding-box;

4、ios 设置 input 按钮样式会被默认样式覆盖

解决方式如下：

```
input,
textarea {
  border: 0;
  -webkit-appearance: none;
}
```

设置默认样式为 none

5、IOS 键盘字母输入，默认首字母大写

解决方案，设置如下属性

```
<input type="text"autocapitalize="off"/>
```

6、h5 底部输入框被键盘遮挡问题

h5 页面有个很蛋疼的问题就是，当输入框在最底部，点击软键盘后输入框会被遮挡。可采用如下方式解决

```
var oHeight = $(document).height(); //浏览器当前的高度
    $(window).resize(function(){

        if($(document).height() < oHeight){

        $("#footer").css("position","static");
     }else{

        $("#footer").css("position","absolute");
     }

    });
```

7、IOS 移动端 click 事件 300ms 的延迟响应

移动设备上的 web 网页是有 300ms 延迟的，玩玩会造成按钮点击延迟甚至是点击失效。这是由于区分单击事件和双击屏幕缩放的历史原因造成的,
2007 年苹果发布首款 iphone 上 IOS 系统搭载的 safari 为了将适用于 PC 端上大屏幕的网页能比较好的展示在手机端上，使用了双击缩放(double tap to zoom)的方案，比如你在手机上用浏览器打开一个 PC 上的网页，你可能在看到页面内容虽然可以撑满整个屏幕，但是字体、图片都很小看不清，此时可以快速双击屏幕上的某一部分，你就能看清该部分放大后的内容，再次双击后能回到原始状态。

双击缩放是指用手指在屏幕上快速点击两次，iOS 自带的 Safari 浏览器会将网页缩放至原始比例。原因就出在浏览器需要如何判断快速点击上，当用户在屏幕上单击某一个元素时候，例如跳转链接，此处浏览器会先捕获该次单击，但浏览器不能决定用户是单纯要点击链接还是要双击该部分区域进行缩放操作，所以，捕获第一次单击后，浏览器会先 Hold 一段时间 t，如果在 t 时间区间里用户未进行下一次点击，则浏览器会做单击跳转链接的处理，如果 t 时间里用户进行了第二次单击操作，则浏览器会禁止跳转，转而进行对该部分区域页面的缩放操作。那么这个时间区间 t 有多少呢？在 IOS safari 下，大概为 300 毫秒。这就是延迟的由来。造成的后果用户纯粹单击页面，页面需要过一段时间才响应，给用户慢体验感觉，对于 web 开发者来说是，页面 js 捕获 click 事件的回调函数处理，需要 300ms 后才生效，也就间接导致影响其他业务逻辑的处理。
解决方案：
1、fastclick 可以解决在手机上点击事件的 300ms 延迟
2、zepto 的 touch 模块，tap 事件也是为了解决在 click 的延迟问题
3、触摸事件的响应顺序为 touchstart --> touchmove --> touchend --> click,也可以通过绑定 ontouchstart 事件，加快对事件的响应，解决 300ms 延迟问题

8、在 ios 和 andriod 中,audio 元素和 video 元素在无法自动播放

应对方案：触屏即播

```
$('html').one('touchstart',function(){
    audio.play() })
```

9、CSS 动画页面闪白,动画卡顿

解决方法:
1.尽可能地使用合成属性 transform 和 opacity 来设计 CSS3 动画，不使用 position 的 left 和 top 来定位
2.开启硬件加速

```
-webkit-transform: translate3d(0, 0, 0);
-moz-transform: translate3d(0, 0, 0);
-ms-transform: translate3d(0, 0, 0);
transform: translate3d(0, 0, 0);
```

10、fixed 定位缺陷

1、ios 下 fixed 元素容易定位出错，软键盘弹出时，影响 fixed 元素定位
2、android 下 fixed 表现要比 iOS 更好，软键盘弹出时，不会影响 fixed 元素定位
3、ios4 下不支持 position:fixed
解决方案： 可用 iScroll 插件解决这个问题

六、代码题

1、变量问题，写出下题结果

```
var test = function(x){
    var y = new Date();
    y = y.getMonth()+x;
    return y.toString();
};
console.log(test(3));//"8"
console.log(x);//报错
```

2、作用域问题，思考此题结果

```
var arr = ["第一次输出","第二次输出","第三次输出"];
for(var i = 0;i < arr.length;i++){
    setTimeout(function(){
        console.log(arr[i]);//undefined
    },i*10)
}
```

3、作用域问题，看题作答

```
var fullname = "John Doe";
var obj = {
    fullname:"Colin Ihrig",
    pop:{
        fullname:"Aurello",
        getFullname:function(){
            return this.fullname;
        }
    }
};
console.log(obj.pop.getFullname());//Aurello
var test = obj.pop.getFullname;
console.log(test());//John Doe
```

4、作用域问题，看题作答

```
var name = "World";
(function(){
    if(typeof name==="undefined"){
        var name = "Jack";
        console.log("Goodbye"+name);//GoodbyeJack
    }else{
        console.log("hello"+name)
    }
})()
```

5、看题作答

```
var a= 1;
Function test(){
Console.log(a);//函数体
    Try{
    a(3)
    A()
    }catch(e){
```

```
        a = 2
    }
    Function a(b){
        a = b
    }
    Console.log(++);//3
}
test()
console.log(a);//1
```

6、看题作答

```
function a(){alert(1)};
(function f(b){
    a();
    a = function(){alert(2)};
    Function a(){alert(3)};
    a();
    var a;
    a&&a();
    b();
})(a)
//3 2 2 1
```

7、用 JavaScript 实现冒泡排序。数据为 23、45、18、37、92、13、24

```
//升序算法
function sort(arr){
    for (var i = 0; i <arr.length; i++) {
        for (var j = 0; j <arr.length-i; j++) {
            if(arr[j]>arr[j+1]){
                var c=arr[j];//交换两个变量的位置
                arr[j]=arr[j+1];
                arr[j+1]=c;
            }
        };
    };
    return arr.toString();
}
```

```
console.log(sort([23,45,18,37,92,13,24]));
```

8、使用 jquery.extend 实现扩展，并解释其与 jquery.fn.extend 的区别

```
var object1 = {
   apple: 0,
   banana: { weight: 52, price: 100 },
   cherry: 97
};
var object2 = {
   banana: { price: 200 },
   durian: 100
};

$.extend( true, object1, object2 );
```

jquery.extends 与 jquery.fn.extend 都可以用来扩展 jquery 方法，jquery 扩展的是 Jquery 类本身，jQuery.fn=jQuery.prototype, jQuery.fn.extend 拓展的是 jQuery 对象（原型的）的方法，用在 jQuery 对象上面

```
$.extend({
      eat: function(){
            console.log("eat");
}
});

$.eat(); // eat

$.fn.extend({
      eat: function(){
            console.log("eat eat");
}
});
$("#eat").eat();    // eat eat
```

9、请说明要输出正确的 myName 的值要如何修改程序?并解释原因

```
foo = function(){
       this.myName = "Foo function.";
```

```
}
foo.prototype.sayHello = function(){
    alert(this.myName);
}
foo.prototype.bar = function(){
    setTimeout(this.sayHello, 1000);
}
var f = new foo;
f.bar();
```

this.sayHello 改成
this.sayHello.call(this)
this.sayHello.apply(this)
this.sayHello.bind(this)
给 this.sayHello 方法绑定作用域，使其指向 foo。

10、将字符串”abc1234efg9088dsd “ 中的数字转换成 * 号。

```
"abc1234efg9088dsd".replace(/\d/g, "*")
"abc1234efg9088dsd".replace(/\d/g, function(matchString, index, input){return "*"})
```

11、window.onload 与 document.ready 有什么区别?

document.ready 就是 jquery 中的$(function(){}) 和$(document).ready(function(){})
是在 dom 文档树加载完之后执行一个函数， window.onload 是在 dom 文档树加载完和所有文件加载完之后执行一个函数。window.onload 在 document.ready 之后执行。

12、创建一个列表，并将其插入 id 为 containter 的 DIV 元素中

1) 原生 javascript 方法：

```
function appendLi(){
    var ul = document.createElement("ul")
    for (var i=0;i<10;i++){
       var li =document.createElement("li");
       var text =document.createTextNode(i);
       li.appendChild(text);
       ul.appendChild(li);
    }
```

```
            console.log(ul);
            document.getElementById("containter").appendChild(ul);
        }
```

2) jquery 方法

```
        function appendLi(){
            var list = "<ul>";
            for(var i = 0; i<10; i++){
                list = list + "<li>" + i + "</li>";
            }
            list + "</ul>";
            $("#containter").append(list);
        }
```

13、希望获取到页面中所有的 checkbox 怎么做？(不使用第三方框架)

```
    var domList = document.getElementsByTagName('input')
    var checkBoxList = [];
    var len = domList.length;      //缓存到局部变量
    while (len--) {      //使用 while 的效率会比 for 循环更高
        if (domList[len].type ==  'checkbox' ) {
            checkBoxList.push(domList[len]);
        }
}
```

14、已知有字符串 foo="get-element-by-id",写一个 function 将其转化成驼峰表示法"getElementById"。

```
    function combo(msg){
    var arr=msg.split("-");
    for(var i=1;i<arr.length;i++){
        arr[i]=arr[i].charAt(0).toUpperCase()+arr[i].substr(1,arr[i].length-1);
    }
    msg=arr.join("");
    return msg;
}
```

(考察基础 API)

15、var ary = [3,6,2,4,1,5]; （考察基础 API）

1) 实现对该数组的倒排，输出[5,1,4,2,6,3]

（1）ary.reverse()

（2）function berarray（ary）{

```
var result=[];
for(var i=0;i<ary.length;i++){
result.unshift(arr[i]);
}
return result;
}
```

2) 实现对该数组的降序排列，输出[6,5,4,3,2,1]

```
(1)ary.sort(function(a,b){return b-a})
(2)function sortDesc(ary){
for (var i=0;i<ary.length;i++){
for (var j=i+1;j<ary.length;j++){
if (ary[j]>ary[i]){// >就是降序  <就是升序
var temp=ary[j];
ary[j]=arr[i];
ary[i]=temp;
} }}
  return ary;
}
```

16、<input type="radio" checked id="aRadio" customedAttr="1" data-param="param">

下面属性分别返回什么结果?

```
$("#aRadio").attr("checked")        //checked
$("#aRadio").prop("checked")          //true
$("#aRadio").attr("customedAttr")    //1
$("#aRadio").prop("data-param")      //undefined
$("#aRadio").attr("data-param") //param
$("#aRadio").data("param")//param
```

17、为了保证页面输出安全，我们经常需要对一些特殊的字符进行转义，请写一个函数 escapeHtml，将<, >, &, “进行转义

```
function escapeHtml(str) {
return str.replace(/[<>”&]/g, function(match) {
        switch (match) {
                                case “<”:
                                    return “&lt;”;
                                case “>”:
                                    return “&gt;”;
                                case “&”:
                                    return “&amp;”;
                                case “\””:
                                    return “&quot;”;
            }
    });
}
```

18、用 js 实现随机选取 10–100 之间的 10 个数字，存入一个数组，并排序。

```
var iArray = [];
funtion getRandom(istart, iend){
            var iChoice = istart - iend +1;
            return Math.floor(Math.random() * iChoice + istart;
}
Math.random()就是获取 0-1 之间的随机数（永远获取不到 1）
for(var i=0; i<10; i++){
var result= getRandom(10,100);
            iArray.push(result);
}
iArray.sort();
```

19、Javascript 中 callee 和 caller 的作用?

caller 是返回一个对函数的引用，该函数调用了当前函数；
callee 是返回正在被执行的 function 函数，也就是所指定的 function 对象的正文。
那么问题来了？如果一对兔子每月生一对兔子；一对新生兔，从第二个月起就开始生兔子；假定每

对兔子都是一雌一雄，试问一对兔子，第 n 个月能繁殖成多少对兔子？（使用 callee 完成）

```
 var result=[];
function fn(n){   //典型的斐波那契数列
    if(n==1){
        return 1;
    }else if(n==2){
            return 1;
    }else{
        if(result[n]){
                return result[n];
        }else{
                //argument.callee()表示 fn()
                result[n]=arguments.callee(n-1)+arguments.callee(n-2);
                return result[n];
        }}}
```

20、完成函数 showImg()，要求能够动态根据下拉列表的选项变化，更新图片的显示

考点：1、下拉框切换：onchange 事件 2、通过 value 获取下拉框的值

```
<body>
<script type=”text/javascript” >
function showImg (oSel) {
//在此处添加代码
var str = oSel.value;
document.getElementById(“pic”).src= str+”.jpg”;
}
</script>
<img id=” pic” src=” img1.jpg” width=” 200″  height=” 200″  />
<br />
<select id=”sel”>
<option value=” img1 “>城市生活</option>
<option value=” img2 “>都市早报</option>
<option value=” img3 “>青山绿水</option>
</select></body>
```

21、判断一个字符串中出现次数最多的字符，统计这个次数

```
var str = 'asdfssaaasasasasaa';
var json = {};
```

```
for (var i = 0; i < str.length; i++) {
        if(!json[str.charAt(i)]){
                json[str.charAt(i)] = 1;
        }else{
                json[str.charAt(i)]++;
        }
};
var iMax = 0;
var iIndex = '';
for(var i in json){
        if(json[i]>iMax){
                iMax = json[i];
                iIndex = i;
        }
}
alert('出现次数最多的是:'+iIndex+'出现'+iMax+'次');
```

22、去掉数组中重复的数字

```
//思路：每遍历一次就和之前的所有做比较，不相等则放入新的数组中！
//这里用的原型 个人做法；
Array.prototype.unique = function(){
    var len = this.length,
        newArr = [],
        flag = 1;
    for(var i = 0; i < len; i++, flag = 1){
        for(var j = 0; j < i; j++){
            if(this[i] == this[j]){
                flag = 0;           //找到相同的数字后，不执行添加数据
            }
        }
        flag ? newArr.push(this[i]) : '';
    }
    return newArr;
}
    方法二：
    var arr=[1,2,3,3,4,4,5,5,6,1,9,3,25,4];
Array.prototype.unique2 = function()
{
    var n = []; //一个新的临时数组
    for(var i = 0; i < this.length; i++) //遍历当前数组
```

```
    {
        //如果当前数组的第 i 已经保存进了临时数组，那么跳过，
        //否则把当前项 push 到临时数组里面
        if (n.indexOf(this[i]) == -1) n.push(this[i]);
    }
    return n;
}

var newArr2=arr.unique2(arr);
alert(newArr2); //输出 1,2,3,4,5,6,9,25
```

23、下面这个 ul，如何点击每一列的时候 alert 其 index？（闭包）

```
<ul id=”test”>
<li>这是第一条</li>
<li>这是第二条</li>
<li>这是第三条</li>
</ul>
    // 方法一：
var lis=document.getElementById('2223').getElementsByTagName('li');
for(var i=0;i<3;i++)
{
    lis[i].index=i;
    lis[i].onclick=function(){
        alert(this.index);
    };
}
//方法二：
var lis=document.getElementById('2223').getElementsByTagName('li');
for(var i=0;i<3;i++){
    lis[i].index=i;
    lis[i].onclick=(function(a){
        return function() {
            alert(a);
        }
    })(i);
}
```

24、给 js 内置对象 Array 添加一个原生方法 filter（args）。功能：将数组参数 args 中包含的元素从原数组中删除，并返回修改后的数组。

```
var arr=[];
Array.prototype={
  push:filter=function(args){
            var as=arr.indexof(args)
            return as；
       }
};
```

25、按如下要求，写一个简单的 jquery 插件。插件调用方法：$(“.light”).light({color:”red”});

功能：根据传入的参数 color，可以改变选择元素的前景色。（color 默认颜色为#000000）

```
$.fn.light = function(option) {
     defalut = {'color': '#000000'};
     var newObj = $.extend({},defalut,option);
       $(this).css(newObj);
   }
   $('.light').light({
      'color': 'blue'
   });
```