## 一.CSS部分

## 1.什么是盒模型？

盒模型是网页制作的基础，在网页文档中每个元素都将被描绘为一个矩形盒子来进行渲染。通常来说，一个完整的盒子由内容(content)、内边距(padding)、边框(border)、外边距 (margin) 组成。

## 2.Doctype的几种类型？

Doctype是Document Type（文档类型）的简写，在页面中，用来指定页面所使用的XHTML（或者HTML）版本。

在 XHTML1.0和HTML 4.01 中有三种 <!DOCTYPE> 声明。分别为过渡的（Transitional）、严格的（Strict）和框架的（Frameset）。

（1）过渡的

一种要求不太严格的方式，允许在页面中使用HTML4.01的标识（符合xhtml语法标准）。过渡的DTD的写法如下：

<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0 Transitional//EN”

“http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd”>

（2）严格的

一种要求严格的DTD，不允许使用任何表现层的标识和属性，例如<br/>等。严格的DTD的写法如下：

<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0 Strict//EN”

“http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd”>

（3）框架的

一种专门针对框架页面所使用的DTD，当页面中含有框架元素时，就要采用这种DTD。框架的DTD的写法如下：

<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0 Transitional//EN”

“http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd”>

在 HTML5 中只有一种：

<!DOCTYPE html>

## 3.如何布局左不动右边自适应的两列布局？

思路主要为以下两种：

1.左边定宽，浮动

2.左边绝对定位

<html>

<head>

<meta charset=”utf-8″>

<title>1</title>

<style type=”text/css”>

\*{

margin: 0;

padding: 0;

}

#left {

position:absolute;

/\*float: left;/\*区别\*/

background-color: red;

width: 250px;

}

#content {

background-color: orange;

margin-left: 250px;/\*==等于左边栏宽度==\*/

}

</style>

</head>

<body>

<div id=”left”>left</div>

<div id=”content”>content</div>

</body>

</html>

## 4.如何布局两列等高？

主要思路如下：

1.假等高列方法，在列的父元素上使用背景图片在y轴上平铺，造成等高列的假象

2.table-cell法

<!DOCTYPE html>

<html lang=”en”>

<head>

<meta charset=”UTF-8″>

<title>Document</title>

<style type=”text/css” media=”screen”>

#box{

width:800px;

background:red;

margin:0 auto;

overflow:hidden;

}

.col{

display:table-cell;

width:400px;

}

.col:first-child{

background:blue;

}

.col:last-child{

background:green;

}

</style>

</head>

<body>

<div id=”box”>

<div class=”col”>

<p>123</p>

<p>456</p>

<p>456</p>

<p>456</p>

</div>

<div class=”col”>8989</div>

</div>

</body>

</html>

3.补丁法

<!DOCTYPE html>

<html>

<head>

<meta charset=”utf-8″>

<title>两列等高布局（float+margin+position)</title>

<style type=”text/css”>

#fa {

width: 800px;

margin: 0 auto;

background-color: #1524e5;

/\*

\*overflow就决定了#fa元素生成了bfc或者触发了haslayout

\*并且还把超出的高度截掉了

\*/

overflow: hidden;

}

.col {

float: left;

width: 50%;

padding-bottom: 10000px;

margin-bottom: -10000px;

/\*

\*把下面两行注释去掉试试看

\*你会发现父元素的背景出现了

\*这说明父元素的高度是最高的子元素的高度

\*而子元素的高度就是根据盒模型算出来的

\*/

/\* padding-bottom: 100px; \*/

/\* margin-bottom: -100px; \*/

/\*

\*由于生成了bfc，所以如果给子元素加一个margin-top，也会被计算进高度

\*不信去掉下面的注释试试~

\*上边出现了父元素的背景吧~~~

\*/

/\* margin-top: 20px; \*/

}

.col:first-child {

background-color: #34ef34;

}

.col:last-child {

background-color: #ef34ef;

}

</style>

</head>

<body>

<div id=”fa”>

<div class=”col”>

<p>1231321321</p>

<p>1313213</p>

<p>1313213</p>

<p>1313213</p>

<p>1313213</p>

<p>1313213</p>

</div>

<div class=”col”>45645456456</div>

</div>

</body>

</html>

## 5.如何布局右侧定宽，左侧或中间自适应？

<!DOCTYPE html>

<html>

<head>

<meta charset=”utf-8″>

<title>右侧定宽，左侧自适应宽</title>

<style type=”text/css”>

\*{

margin: 0;

padding: 0;

}

#content {

width: 100%;

color: red;

float: left;

background:#666;

height:400px;

margin-right: -230px;/\*==等于右侧栏宽度==\*/

}

#side{

width:230px;/\*==等于右侧栏宽度==\*/

height:500px;

color:#fff;

background:green;

float: left;

}

</style>

</head>

<body>

<div id=”content”>Main Content</div>

<div id=”side”></div>

</body>

</html>

## 6.如何布局三列自适应？

主要思路如下：

1.绝对定位法(左右两栏绝对定位，固定于页面两侧，主体栏用margin撑开距离）

<!DOCTYPE html>

<html>

<head>

<title></title>

</head>

<style type=”text/css” media=”screen”>

\*{

margin:0px;

height: 100%;

}

#left,#right{

position: absolute;

top:0px;

width: 200px;

height: 100%;

}

#left{

left:0px;

background: red;

}

#right{

right:0px;

background: green;

}

#main{

margin:0 210px;background: blue;height: 100%;

}

</style>

<body>

<div id=”left”></div>

<div id=”main”></div>

<div id=”right”></div>

</body>

</html>

2.margin负值法

主体采用双层标签，外层DIV宽度100%显示，且浮动。内层div为真正主体内容，左栏与右栏采用margin负值定位，左栏左浮动，margin-left:-100%,使其正好到达最左端，

<!DOCTYPE html>

<html>

<head>

<meta http-equiv=”Content-Type” content=”text/html; charset=utf-8″ />

<title>11</title>

<style type=”text/css”>

html,body{margin:0; height:100%;}

#main{width:100%; height:100%; float:left;}

#main #body{margin:0 210px; background:#ffe6b8; height:100%;}

#left,#right{width:200px; height:100%; float:left; background:#a0b3d6;}

#left{margin-left:-100%;}

#right{margin-left:-200px;}

</style>

</head>

<body>

<div id=”main”>

<div id=”body”></div>

</div>

<div id=”left”></div>

<div id=”right”></div>

</body>

</html>

3.自身浮动法:

应用标签浮动跟随的特性，左栏左浮动，右栏右浮动，主体直接放后面，实现自适应

<!DOCTYPE html>

<html>

<head>

<meta charset=”utf-8″>

<title>11</title>

<style type=”text/css”>

html,body{margin:0; height:100%;}

#main{height:100%; margin:0 210px; background:#ffe6b8;}

#left,#right{width:200px; height:100%; background:#a0b3d6;}

#left{float:left;}

#right{float:right;}

</style>

</head>

<body>

<div id=”left”></div>

<div id=”right”></div>

<div id=”main”></div>

</body>

</html>

## 7.gif,png,jpg的区别？

参考: [^图片特点比较](http://wenku.baidu.com/view/fef1587c6bd97f192379e95a.html?re=view)

| 图片类型 | 透明性 | 动画支持 | 损耗性 | 显示方式 |
| --- | --- | --- | --- | --- |
| GIF | Gif是一种布尔透明类型，它可以是全透明或全不透明，但是它并没有半透明（alpha 透明） | 支持 | Gif是一种无损耗的图像格式，这也意味着你可以对gif图片做任何操作也不会使图像质量产生损耗; | Gif使用LZW算法进行压缩，当压缩gif的过程中，像素是由上到下水平压缩的，这也意味着同等条件下，横向的gif图片比竖向的gif图片更小。例如500\*10的图片比10\*500的图片更小,Gif支持可选择性的间隔渐进显示；只有256种颜色的gif图片不适合作为照片，它适合做对颜色要求不高的图形。 |
| PNG | Png是完全支持alpha透明的(透明，半透明，不透明) | 不支持 | png是一种无损耗的图像格式，这意味着png图片做任何操作也不会使图像质量产生损耗。这也使得png可以作为jpeg编辑的过渡格式; | 像GIF一样，png也是水平扫描的，这意味着水平重复颜色比垂直重复颜色的图片更小,它支持间隔渐进显示，但是会使图片大小更大 |
| JPG | 不支持 | 不支持 | 除了一些如旋转（仅仅是90、180、270度旋转），裁切，从标准类型到先进类型，编辑图片的原数据之外，所有其它对jpeg图像的处理都会使它的质量产生损失。所以我们在编辑过程一般用png作为过渡格式 | 它支持隔行渐进显示（但是ie浏览器并不支持这个属性，但是ie会在整个图像信息完全到达的时候显示）。 由上可以看出Jpeg是最适web上面的摄影图片和数字照相机中。 |

## 综述：

GIF适合图形，JPEG适合照片，PNG系列两种都适合。

PNG相比GIF：

对于相同色值，gif格式图片会比png32格式图片小。但png8才是最优的选择。

PNG 8除了不支持动画外，PNG8有GIF所有的特点，但是比GIF更加具有优势的是它支持alpha透明和更优的压缩。所以，大多数情况下，你都应该用PNG8不是GIF（除了非常小的图片GIF会有更好的压缩外）。

PNG相比JPEG：

JPEG比全色PNG具有更加好的压缩，因此也使得JPEG适合照片，但是编辑JPEG过程中容易造成质量的损失，所以全色PNG适合作为编辑JPEG的过渡格式.

## 8.什么是css sprite?优缺点？

CSS Sprites(雪碧图）就是把网页中用到的一些背景图片整合到一张图片文件中，利用CSS的“background-image”，“background-repeat”，“background-position”来组合进行背景定位。

优点： 减少网页的http请求，大大提高页面性能。

缺点：在图片合并的时候，你要把多张图片有序的合理的合并成一张图片，还要留好足够的空间，防止板块内出现不必要的背景。修改编辑图片时也相对较为繁琐。

## 9.制作细线表格？

定义CSS:border-collapse属性，将值设为collapse,合并边框

## 10.position:relative,absolute,fixed区别与联系？

postion一共有四个可选属性：static|relative|absolute|fixed，默认是static。

### 一：relative

relative会导致元素自身位置的相对变化，但不会影响其它元素的位置、大小。

### 二：absolute

①absolute元素脱离文档结构（和float元素类似，float元素也会脱离文档结构）

②absolute元素具有“包裹性”。会使其之前的块级元素的宽度变为内容的宽度。

③absolute元素具有“跟随性”。没有设置top、left的值时，与其前一个元素的top,left值为0。

如果为absolute设置了top、left，absolute元素会根据最近的定位上下文确定位置，浏览器会递归查找该元素的所有父元素，如果找到一个设置了position:relative/absolute/fixed的元素，就以该元素为基准定位，如果没找到，就以浏览器边界定位。

### 三：fixed

其实fixed和absolute是一样的，唯一的区别在于：absolute元素是根据最近的定位上下文确定位置，而fixed永远根据浏览器确定位置.

## 11.如何居中一个float:left的元素

主要思路如下：

1.定义父级元素为行级元素，在父级元素之上设定text-align:center

<!DOCTYPE html>

<html lang=”en”>

<head>

<meta charset=”UTF-8″>

<title>Document</title>

</head>

<body >

<div style=”text-align: center;”>

<div style=”display: inline-block;”>

<div style=”float:left;background: green;”>

子元素1

</div>

<div style=”float:left;background: blue;”>

子元素2

</div>

</div>

</div>

</body>

</html>

2.父元素浮动法

在外层包裹一个父元素，使他浮动，相对定位,left:50%；内部元素同样相对定位，left:-50%；

<!DOCTYPE html>

<html lang=”en”>

<head>

<meta charset=”UTF-8″>

<title>Document</title>

</head>

<body >

<div style=”float:left;position:relative;left:50%;”>

<div style=”float:left;left: -50%;background: green;”>

子元素1

</div>

<div style=”float:left;background: blue;”>

子元素2

</div>

</div>

</body>

</html>

## 12.Css在各浏览器下的兼容问题你通常是怎么来解决的，请分享你的经验；

解决办法一般都是平时为记笔记记录，搜索时熟悉内容找笔记内容+不熟悉的找谷歌、百度。

## 二.JS部分

## 1.parseInt 与 parseFloat 的区别？

parseInt 与 parseFloat 是把字符串转换成整数和浮点数的方法，

二者语法如下

parseInt(string, radix)

parseFloat(string)

parseInt多一个可选的radix数字基数。

两者的一些参数执行返回结果如下

| str | parseInt(str) | parseFloat(str) |
| --- | --- | --- |
| 零长度字符串 | NaN | NaN |  |
| null | NaN | NaN |  |
| undefined | NaN | NaN |  |
| .3 | 0 | 0.3 |  |
| -.3 | NaN | -0.3 |  |
| -3 | -3 | -3 |  |
| 3e7 | 3 | 30000000 |  |
| 0xFF | 255 | 0 |  |
| 0x3e7 | 999 | 0 |  |
| 08 | 8 | 8 |  |
| 3 | 3 | 3 |  |
| 123abc | 123 | 123 |  |

## 2.valueof 与 toString 的区别？

基本上，所有的JS数据类型都拥有这两种方法(NULL除外)，他们俩解决了JS值的运算和显示问题。

在《javaScript　高级程序设计》一书中描述如下：

toString()——返回对象的原始字符串表示

valueOf()——返回最适合该对象的原始值

一个例子：

<script>

function Money(gold, silver, coin)

{

this.gold = gold;

this.silver = silver;

this.coin = coin;

}

Money.parse = function(value)

{

var coin = parseInt(value % 25);

var silver = parseInt(value / 25 % 20);

var gold = parseInt(value / 500);

return new Money(gold, silver, coin);

}

Money.prototype.valueOf = function( )

{

return ((this.gold \* 20) + this.silver) \* 25 + this.coin;

}

Money.prototype.toString = function( )

{

return this.gold + “金，” + this.silver + “银，” + this.coin + “铜”;

}

var money1 = new Money(5, 6, 1);

var money2 = new Money(2, 4, 6);

var money3 = Money.parse(money1 + money2);

alert(money3);//7金，10银，7铜

</script>

这里隐式调用了valueOf跟toString，二者并存的情况下，在数值运算中，优先调用了valueOf，字符串运算中，优先调用了toString。比如money1 + money2，调用的就是两者valueOf之后的值相加，而alert的时候，把money3先toString了一下。

| str | valueOf(str) | toString(str) |
| --- | --- | --- |
| Array | Array数组的元素被转换为字符串，这些字符串由逗号分隔，连接在一起。其操作与 Array.toString 和 Array.join 方法相同。 | Array将 Array 的元素转换为字符串。结果字符串由逗号分隔，且连接起来。 |  |
| Boolean | Boolean的原始值 | Boolean如果 Boolean 值是 true，则返回 “true”。否则，返回 “false”。 |  |
| Date | Date存储的时间是从 1970 年 1 月 1 日午夜开始计的毫秒数 UTC。 | Date返回日期的文字表示法。Error返回一个包含相关错误消息的字符串。 |  |
| Function | Function函数本身。 | Function返回如下格式的字符串，其中 functionname 是被调用 toString 方法函数的名称： function functionname( ) { [native code] } |  |
| Number | Number数字值。 | Number返回数字的文字表示。 |  |
| Object | Object对象本身。这是默认情况。 | Object默认返回 “[object objectname]”，其中 objectname 是对象类型的名称。 |  |
| String | String字符串值。 | String返回 String 对象的值。 |  |

## 3.全等与相等？

相等(==),当且仅当两个运算符相等时，返回true。在确定两个运算符时会进行类型转换。

执行类型转换的规则如下：

如果一个运算数是 Boolean 值，在检查相等性之前，把它转换成数字值。false 转换成 0，true 为 1。

如果一个运算数是字符串，另一个是数字，在检查相等性之前，要尝试把字符串转换成数字。

如果一个运算数是对象，另一个是字符串，在检查相等性之前，要尝试把对象转换成字符串。

如果一个运算数是对象，另一个是数字，在检查相等性之前，要尝试把对象转换成数字。

全等(===)

只有在无需类型转换运算符就相等的情况下，才返回 true。

## 4.break 与 continue 的区别？

break表示跳出循环，continue表示结束本次循环

## 5.js的作用域？

js的作用域以函数function来划分，根据作用域，使得变量存在作用域的概念分为全局变量和局部变量。

## 6.Array 的 slice 与 splice 的区别？

arrayObject.slice(start,end)

slice() 方法从已有的数组中返回选定的元素,其接收1或2个参数，返回一个新的数组，包含数组下标从 start 到 end（不包括该元素）的 arrayObject 中的元素。（如果没有指定end参数，那么切分的数组包含从 start 到数组结束的所有元素。)

如：

var aCloros = [“red”,”green”,”blue”,”yellow”,”purple”];

var aCloros2 = aCloros.slice(1);

var aCloros3 = aCloros.slice(1,4);

alert(aCloros2);//output “green,blue,yellow,purple”

alert(aCloros3);//output “green,blue,yellow”

arrayObject.splice(index,howmany,item1,…..,itemX)

splice() 方法可从数组中添加或删除项目，然后返回被删除的项目。

splice() 方法可删除从 index 处开始的零个或多个元素，howmany表示删除数量，后面的参数表示向数组中添加的新项目。

var arr = new Array(6)

arr[0] = “red”

arr[1] = “green”

arr[2] = “blue”

arr[3] = “yellow”

arr[4] = “purple”

arr.splice(2,0,”white”) //output red,green,white,blue,yellow,purple

arr.splice(2,1,”white”) //output red,green,white,yellow,purple

## 7.正则中test 与 match 的区别？

RegExpObject.test(str)

test()方法用于检测一个字符串是否匹配某个模式.如果字符串str中含有与RegExpObject 匹配的文本，则返回 true，否则返回 false。

//匹配正整数

var patten = /^[1-9]\d\*$/;

patten.test(“1498924”) //true

patten.test(“-1498924”) //false

stringObject.match(regexp)

match() 方法可在字符串内检索指定的值，或找到一个或多个正则表达式的匹配。返回指定的值。

var str=”1 + 2 \* 3 / 4″

str.match(/\d+/g) //[“1”, “2”, “3”, “4”]

## 8.如何在js中创建一个对象并继承他？

function Person(name) {

this.name = name;

}

Person.prototype = {

constructor: Person, //强制声明构造函数，默认是Object

sayName : function( ) {

alert(this.name);

}

}

var Person1 = new Person(“xiaoming”);

## 9.什么是prototype ? 什么是原型链？

在JavaScript中每个对象都有一个内部的链接指向另一个对象，这个对象就是原对象的原型（prototype）。这个原型对象也有自己的原型，直到对象的原型为 null 为止。这种一级一级的链结构就是原型链。

如通过new运算符创建的对象，构造函数.prototype的值就是该对象的原型对象。

var now = new Date();

now对象的原型对象就是Date.prototype。

通过new 一个对象就可以调用里面的公开的方法，属性。

## 10.setTimeout 与 setInterval 的区别？

setTimeout()方法用于在指定的毫秒数后调用函数或计算表达式。

setInterval()方法可按照指定的周期（以毫秒计）来调用函数或计算表达式。它会不停地调用函数，直到 clearInterval() 被调用或窗口被关闭。由 setInterval() 返回的 ID 值可用作 clearInterval() 方法的参数。

## 11.cssText?

cssText的本质就是设置HTML元素的style属性值。

document.getElementById(“d1”).style.cssText = “color:red; font-size:13px;”;

## 12.offsetWidth, scrollLeft, scrollHeight?

offsetWidth:元素的宽度（width + padding + border）

scrollLeft:返回和设置当前横向滚动条的坐标值

scrollHeight:获取对象的滚动高度。

## 13.IE的事件与w3c事件的区别？

w3c事件流:

从根文档(html)开始遍历所有子节点，如果目标事件的父节点设置为捕获时触发，则执行该事件，直到目标被执行，然后再事件冒泡(设置为捕获时触发的事件不再被执行)。

ie事件流:

从目标事件被执行，然后再冒泡父节点的事件，直到根文档。

## 三.高级部分

## 1.除了jQuery以外，请写出你所知道的js框架，试说出不同框架的特点；

Zeptojs:轻量级的适于移动端的针对高级浏览器的JavaScript库（类jQuery）

Underscore.js:一个JavaScript实用库，提供了一整套函数式编程的实用功能，没有扩展任何JavaScript内置对象，弥补了部分jQuery没有实现的功能。

Bootstrap:拥有丰富的Web组件。

Node.js: 跨平台、可用于服务器端和网络应用的运行环境；具有事件驱动、单线程运行、非阻塞I/O调用、V8引擎执行代码的特性。Node.js含有一系列内置模块，使得程序可以作为独立服务器运行,适合在分布式设备上运行的数据密集型的实时应用.

AngularJS: MVVM、模块化、自动化双向数据绑定、语义化标签、依赖注入等等，适用于单页面、CRUD应用。

ReactJs:基于组件化开发，而且所有组件有其状态。将DOM封装为可以相互组合的 Component, 并且将DOM操作抽象为状态的改变。

## 2.对于Javascript中OOP的理念，你认为在项目开发中有没有必要，什么样的项目适合前端OOP，请谈谈你的看法；

我个人认为不太重要，在前端页面不需要刻意的去追求面向对象。

因为对于面向对象我们需要注意“继承、多态、封装、组合”等特性，它们的出发点都是“分离减少关注点”。使程序以最小的代价来适应“关注点”的变化。

但对于前端代码需要关注的东西与后端存在许多不同，后端程序大多只需关注项目的“数据+行为”——关注点不多且容易预测。而前端则是“数据+行为+展现+交互”，其中多出来的“展现+交互”决定了前端的关注点更多且更加无从预测，除非人为限制减少相关的关注点，让UI和交互套在一个相对死的范围内进行，而这样难免会影响到UI的灵活性，不可避免的面对：“丑，慢，大”等特性。

结合以上，前端层面上做好代码的分层、解耦、结构化则更优，但做这些事和追求面向对象没有必然关系。

## 3.在Javascript开发中，关于性能优化，分享一下你相关的经验？

对于JS开发个人总结大概如下：

基本原则：

尽可能的减少请求。

代码开发遵守规范

具体细节：

基本的代码规范，如JS脚本放在底部加载，尽量写为外部文件，基本的上线压缩等

其余一些细节包括

资源加载

资源加载：首页加载性能提升

按需加载：静态资源依赖关系表，lazyload

开发规范

AJAX:缓存AJAX,请求：GET

减少DOM操作

避免全局变量，尽量使用语言本身的构造和内建函数等。

等等。。

具体可参考[前端性能优化指南](https://github.com/yangzj1992/web_performance_optimization)

## 4.对于模块开发你是怎么看的？

前端采用模块化开发，使得开发体验大大增强，也使代码管理更加清晰、规范。主要表现为以下几点：

减少命名冲突，消除全局变量

一个模块一个文件，组织更清晰

依赖自动加载，按需加载

其中文件按需加载，依赖自动管理，使得更多精力去关注模块代码本身，开发时不需要在页面上写一大堆script引用，一个require初始化模块就搞定。

前端模块化规范标准：

CommonJs (Node.js)

AMD (RequireJS)

CMD (SeaJS)

也可参考这里的前部分[前端模块化开发的价值](https://github.com/seajs/seajs/issues/547)

## 5.对于Javascript MVC开发你是怎么看的？分享一下你了解的相关信息？

简单了解的采用MVC开发模式的库包括：

Backbone.js——优点：社区较好，缺点：抽象较弱。

Ember.js——优点：丰富的模板系统，拥有复合视图和UI绑定；缺点：相对较新，还不够成熟

Angular.js——优点：对模板范围和控制器设计有很好的考虑，拥有依赖注入系统，支持丰富的UI绑定语法。缺点：代码的模块性不强，视图的模块化也不够。

其余了解的不多

但平常做项目的时候，特别是大项目，一定要把MVC的概念放在业务场景里多想想。MVC并不是终极思想。它有它适合的地方，但也有其局限的一面。还有前端MVC的话，对应的后台最好是REST风格的接口。还有并不是越复杂的前端业务，越紧迫地需要MVC。一定要多想多看。

MVC的一大问题在于:MVC开发模式会将本身简单的系统大大增加结构的复杂性，并且降低效率。

但整套前端开发其实本质上可以看做一个大大的MVC架构:

Model: HTML/XHTML中的信息

View: Style sheet

Controller: EMAScripts等等脚本

## 6.AJAX是什么? AJAX跨域的解决办法?

AJAX即“Asynchronous JavaScript and XML”（异步的JavaScript与XML技术），是一种创建交互式网页应用的网页开发技术。通过在后台与服务器进行少量数据交换，AJAX 可以使网页实现异步更新。意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

AJAX跨域解决：

使用中间层过渡的方式：

就是在AJAX与不同域的服务器进行通讯的中间加一层过渡，这一层过渡可以是PHP、JSP、c++等任何具备网络通讯功能的语言，由中间层向不同域的服务器进行读取数据的操作。

使用<script>标签

利用<script>标签中src来请求，因为<script>标签的src属性不存在跨域的问题。

jsonp

JavaScript 变量可以用来保存两种类型的值：基本类型值和引用类型值。基本类型的值源自以下 5  
种基本数据类型： Undefined、 Null、 Boolean、 Number 和 String。