Day24

java课程-李彦伯

- Day24
  - JavaScript
    - JavaScript的组成
    - JavaScrpt的引入
    - ECMAScript
      - 基本语法
      - JS对象
        - Array对象
        - Date对象
        - Math对象
        - String对象
    - Bom
      - Window对象
        - window属性
        - window方法
      - History 对象
        - History 对象常用方法
      - Location对象
        - Location对象常用属性
    - Dom
      - Document对象
        - Document对象常用方法
      - Element对象
        - 常用方法
        - 节点操作

JavaScript

> JavaScript是web上的一种编程语言,用于开发交互式的web页面,不需要进行编译,嵌套在html文件中由浏览器执行,和Java没有任何关系

JavaScript的组成

1. 核心(ECMAScript),包括基本的语法,控制语句,基本运算等
2. 文档对象模型(Dom),主要是操作html的文档内容,设置标签的内容和属性
3. 浏览器对象模型(Bom),主要是操作当前浏览器窗口的内容,如页面的返回和跳转等

JavaScrpt的引入

- 内嵌式

> 在html文档中,将相关的js代码放在`<script>`标签中

```
<script type="text/javascript">
//js代码
</script>
```

- 外联式

> 在外部创建一个文件后缀名是.js,在html中通过`<script src="">`进行引入

```
<script src="js/index.js" type="text/java
script" charset="utf-8"></script>
```

ECMAScript

基本语法

> 基本语法和java大致相同,个别操作会不同

- 声明变量的时候不写数据类型要写`var` 如`var a = 3;`,js的变量是弱类型,同一变量可以表示多种类型,当赋值给不同的值的时候,就代表不同的类型

- 所有的数字类型都是Number类型,可以表示整数和小数
- Boolean类型使用true和false
- 字符串使用单引号和双引号都可以
- 比较运算符 `==` 会将两边的值转换成相同类型判断数值是否相同, `===` 不会转换,就是比较类型和数值必须全部相同才会返回true

```
<script type="text/javascript">
window.onload = function test(){
    var a = "1";
    var b = 1;
    alert(a === b);
}
</script>
```

- 方法的定义注意不写返回值类型,不写参数的类型,如果方法需要写返回值,就在方法体中写return

```
<script type="text/javascript">
function test(){
    //js代码
    return true;
}
</script>
```

- if if..else if..else switch for while和java的语法相同

JS对象

Array对象

- 创建Array对象的语法

```
var arr = new Array();
var arr = new Array(size);
var arr = new Array(element0, element1,
..., elementn);
```

- 给数组赋值,通过索引进行赋值

```
arr[0] = "123";
arr[1] = true;
arr[2] = 45;
```

- lenth属性:获取数组的长度

```
var lenth = arr.length;
```

- push()方法:向数组中添加元素,返回值是数组的长度

```
//参数可以是1个或者多个
var a = arr.push("好好学习","天天向上");
```

- toString()方法:将数组中的内容转换为字符串

```
var str  = arr.toString();
```

Array 对象属性

属性	描述
constructor	返回对创建此对象的数组函数的引用。
length	设置或返回数组中元素的数目。
prototype	使您有能力向对象添加属性和方法。

Array 对象方法

方法	描述
concat()	连接两个或更多的数组，并返回结果。
join()	把数组的所有元素放入一个字符串。元素通过指定的分隔符进行分隔。
pop()	删除并返回数组的最后一个元素
push()	向数组的末尾添加一个或更多元素，并返回新的长度。
reverse()	颠倒数组中元素的顺序。
shift()	删除并返回数组的第一个元素
slice()	从某个已有的数组返回选定的元素
sort()	对数组的元素进行排序
splice()	删除元素，并向数组添加新元素。
toSource()	返回该对象的源代码。
toString()	把数组转换为字符串，并返回结果。
toLocaleString()	把数组转换为本地数组，并返回结果。
unshift()	向数组的开头添加一个或更多元素，并返回新的长度。
valueOf()	返回数组对象的原始值

Date对象

- 创建Date对象的语法`var myDate=new Date()`
- Date的常用方法

```
var date = new Date();
var dayOfMonth = date.getDate();//1-31
var dayOfWeek = date.getDay();//0-6
var month = date.getMonth();//0-11
var year = date.getFullYear();
```

方法	描述
Date()	返回当日的日期和时间。
getDate()	从 Date 对象返回一个月中的某一天（1 ~ 31）。
getDay()	从 Date 对象返回一周中的某一天（0 ~ 6）。
getMonth()	从 Date 对象返回月份（0 ~ 11）。
getFullYear()	从 Date 对象以四位数字返回年份。
getYear()	请使用 getFullYear() 方法代替。
getHours()	返回 Date 对象的小时（0 ~ 23）。
getMinutes()	返回 Date 对象的分钟（0 ~ 59）。
getSeconds()	返回 Date 对象的秒数（0 ~ 59）。
getMilliseconds()	返回 Date 对象的毫秒(0 ~ 999)。
getTime()	返回 1970 年 1 月 1 日至今的毫秒数。
getTimezoneOffset()	返回本地时间与格林威治标准时间（GMT）的分钟差。
getUTCDate()	根据世界时从 Date 对象返回月中的一天（1 ~ 31）。
getUTCDay()	根据世界时从 Date 对象返回周中的一天（0 ~ 6）。
getUTCMonth()	根据世界时从 Date 对象返回月份（0 ~ 11）。
getUTCFullYear()	根据世界时从 Date 对象返回四位数的年份。
getUTCHours()	根据世界时返回 Date 对象的小时（0 ~ 23）。
getUTCMinutes()	根据世界时返回 Date 对象的分钟（0 ~ 59）。
getUTCSeconds()	根据世界时返回 Date 对象的秒钟（0 ~ 59）。
getUTCMilliseconds()	根据世界时返回 Date 对象的毫秒(0 ~ 999)。
parse()	返回1970年1月1日午夜到指定日期（字符串）的毫秒数。
setDate()	设置 Date 对象中月的某一天（1 ~ 31）。
setMonth()	设置 Date 对象中月份（0 ~ 11）。
setFullYear()	设置 Date 对象中的年份（四位数字）。
setYear()	请使用 setFullYear() 方法代替。
setHours()	设置 Date 对象中的小时（0 ~ 23）。
setMinutes()	设置 Date 对象中的分钟（0 ~ 59）。
setSeconds()	设置 Date 对象中的秒钟（0 ~ 59）。
setMilliseconds()	设置 Date 对象中的毫秒（0 ~ 999）。
setTime()	以毫秒设置 Date 对象。

Math对象

> Math的属性和方法都是使用类来直接调用

Math 对象属性

属性	描述
E	返回算术常量 e，即自然对数的底数（约等于2.718）。
LN2	返回 2 的自然对数（约等于0.693）。
LN10	返回 10 的自然对数（约等于2.302）。
LOG2E	返回以 2 为底的 e 的对数（约等于 1.414）。
LOG10E	返回以 10 为底的 e 的对数（约等于0.434）。
PI	返回圆周率（约等于3.14159）。
SQRT1_2	返回返回 2 的平方根的倒数（约等于 0.707）。
SQRT2	返回 2 的平方根（约等于 1.414）。

Math 对象方法

方法	描述
abs(x)	返回数的绝对值。
acos(x)	返回数的反余弦值。
asin(x)	返回数的反正弦值。
atan(x)	以介于 -PI/2 与 PI/2 弧度之间的数值来返回 x 的反正切值。
atan2(y,x)	返回从 x 轴到点 (x,y) 的角度（介于 -PI/2 与 PI/2 弧度之间）。
ceil(x)	对数进行上舍入。
cos(x)	返回数的余弦。
exp(x)	返回 e 的指数。
floor(x)	对数进行下舍入。
log(x)	返回数的自然对数（底为e）。
max(x,y)	返回 x 和 y 中的最高值。
min(x,y)	返回 x 和 y 中的最低值。
pow(x,y)	返回 x 的 y 次幂。
random()	返回 0 ~ 1 之间的随机数。
round(x)	把数四舍五入为最接近的整数。
sin(x)	返回数的正弦。
sqrt(x)	返回数的平方根。
tan(x)	返回角的正切。
toSource()	返回该对象的源代码。
valueOf()	返回 Math 对象的原始值。

```
var pi = Math.PI;
var test1 = Math.abs(-1);
```

String对象

- length属性,获取字符串的长度

```
var theLength = "abc".length;
```

- 字符串方法

String 对象方法

方法	描述
anchor()	创建 HTML 锚。
big()	用大号字体显示字符串。
blink()	显示闪动字符串。
bold()	使用粗体显示字符串。
charAt()	返回在指定位置的字符。
charCodeAt()	返回在指定的位置的字符的 Unicode 编码。
concat()	连接字符串。
fixed()	以打字机文本显示字符串。
fontcolor()	使用指定的颜色来显示字符串。
fontsize()	使用指定的尺寸来显示字符串。
fromCharCode()	从字符编码创建一个字符串。
indexOf()	检索字符串。
italics()	使用斜体显示字符串。
lastIndexOf()	从后向前搜索字符串。
link()	将字符串显示为链接。
localeCompare()	用本地特定的顺序来比较两个字符串。
match()	找到一个或多个正则表达式的匹配。
replace()	替换与正则表达式匹配的子串。
search()	检索与正则表达式相匹配的值。
slice()	提取字符串的片断，并在新的字符串中返回被提取的部分。
small()	使用小字号来显示字符串。

split()	把字符串分割为字符串数组。
strike()	使用删除线来显示字符串。
sub()	把字符串显示为下标。
substr()	从起始索引号提取字符串中指定数目的字符。
substring()	提取字符串中两个指定的索引号之间的字符。
sup()	把字符串显示为上标。
toLocaleLowerCase()	把字符串转换为小写。
toLocaleUpperCase()	把字符串转换为大写。
toLowerCase()	把字符串转换为小写。
toUpperCase()	把字符串转换为大写。
toSource()	代表对象的源代码。
toString()	返回字符串。
valueOf()	返回某个字符串对象的原始值。

Bom

Window对象

> Window 对象表示一个浏览器窗口或一个框架。在客户端 JavaScript 中，Window 对象是全局对象，所有的表达式都在当前的环境中计算。也就是说，要引用当前窗口根本不需要特殊的语法，可以把那个窗口的属性作为全局变量来使用。例如，可以只写 document，而不必写 window.document。
> 同样，可以把当前窗口对象的方法当作函数来使用，如只写 alert()，而不必写 Window.alert()。
> Window 对象的 window 属性和 self 属性引用的都是它自己。

window属性

> window的属性有很多,在今后的学习和开发过程中我们最为常用的三个属性分别为document,history,location属性

Window 对象属性

属性	描述
closed	返回窗口是否已被关闭。
defaultStatus	设置或返回窗口状态栏中的默认文本。
document	对 Document 对象的只读引用。请参阅 Document 对象。
history	对 History 对象的只读引用。请参数 History 对象。
innerheight	返回窗口的文档显示区的高度。
innerwidth	返回窗口的文档显示区的宽度。
length	设置或返回窗口中的框架数量。
location	用于窗口或框架的 Location 对象。请参阅 Location 对象。
name	设置或返回窗口的名称。
Navigator	对 Navigator 对象的只读引用。请参数 Navigator 对象。
opener	返回对创建此窗口的窗口的引用。
outerheight	返回窗口的外部高度。
outerwidth	返回窗口的外部宽度。
pageXOffset	设置或返回当前页面相对于窗口显示区左上角的 X 位置。
pageYOffset	设置或返回当前页面相对于窗口显示区左上角的 Y 位置。
parent	返回父窗口。
Screen	对 Screen 对象的只读引用。请参数 Screen 对象。
self	返回对当前窗口的引用。等价于 Window 属性。
status	设置窗口状态栏的文本。
top	返回最顶层的先辈窗口。
window	window 属性等价于 self 属性，它包含了对窗口自身的引用。
screenLeft screenTop screenX screenY	只读整数。声明了窗口的左上角在屏幕上的的 x 坐标和 y 坐标。IE、Safari 和 Opera 支持 screenLeft 和 screenTop，而 Firefox 和 Safari 支持 screenX 和 screenY。

- document属性,是Document对象会在DOM中介绍,使用的时候直接写document
- history属性,是History对象,会在介绍History对象中介绍,使用的时候直接写history
- location属性,是Location对象,会在介绍Location对象中

介绍.使用的时候直接写location

window方法

Window 对象方法

方法	描述
alert()	显示带有一段消息和一个确认按钮的警告框。
blur()	把键盘焦点从顶层窗口移开。
clearInterval()	取消由 setInterval() 设置的 timeout。
clearTimeout()	取消由 setTimeout() 方法设置的 timeout。
close()	关闭浏览器窗口。
confirm()	显示带有一段消息以及确认按钮和取消按钮的对话框。
createPopup()	创建一个 pop-up 窗口。
focus()	把键盘焦点给予一个窗口。
moveBy()	可相对窗口的当前坐标把它移动指定的像素。
moveTo()	把窗口的左上角移动到一个指定的坐标。
open()	打开一个新的浏览器窗口或查找一个已命名的窗口。
print()	打印当前窗口的内容。
prompt()	显示可提示用户输入的对话框。
resizeBy()	按照指定的像素调整窗口的大小。
resizeTo()	把窗口的大小调整到指定的宽度和高度。
scrollBy()	按照指定的像素值来滚动内容。
scrollTo()	把内容滚动到指定的坐标。
setInterval()	按照指定的周期（以毫秒计）来调用函数或计算表达式。
setTimeout()	在指定的毫秒数后调用函数或计算表达式。

- alert() 方法用于显示带有一条指定消息和一个 OK 按钮的警告框.

```
alert("你好");
alert(123);
var a = new Array();
alert(a)
```

- confirm() 方法用于显示一个带有指定消息和 OK 及取消按钮的对话框.点击ok的时候返回值为true,点击取消

的时候返回值为false.

```
var orConfirm = confirm("你确认么?");
alert(orConfirm);
```

- prompt() 方法用于显示可提示用户进行输入的对话框。

```
/*
text 表示提示框的标题,可以省略
defaultText 表示文本框中默认内容,可以省略
返回值表示文本框中输入的内容,只有点击确定才会有内容
*/
var text = prompt(text,defaultText)
```

- 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
- clearInterval(定时操作的对象)方法,可以取消setInterval()的定时操作的对象

```
 var interval = setInterval("show()",100
0);
 function show(){
     document.write("aaa");
 }
 function stop(){
    clearInterval(interval);
}
```

- setTimeout() 方法用于在指定的毫秒数后调用函数或计算表达式.(只调用一次)
- clearTimeout() 方法可取消由 setTimeout() 方法设置的 timeout

```
 var timeout = setTimeout("show()",1000);
 function show(){
     document.write("aaa");
 }
 function stop(){
    clearInterval(timeout);
}
```

History 对象

> 在js中直接写history,实质上window的属性,也就是History的对象,所以我们一般使用其对象也是直接写history

History 对象常用方法

- back(),加载 history 列表中的前一个 URL。
- forward(),加载 history 列表中的下一个 URL。
- go(),加载 history 列表中的某个具体页面。 -1表示前一个页面,1表示后一个页面,以此类推,所以我们一般使用的时候都会使用go()方法

Location对象

> 在js中直接写location,实质上window的属性,也就是Location的对象,所以我们一般使用其对象也是直接写location

Location对象常用属性

- href 属性是一个可读可写的字符串，可设置或返回当前显示的文档的完整 URL。

```
location.href = "http://www.baidu.com";
```

Dom

Document对象

> 在js中直接写document,实质上window的属性,也就是Document的对象,所以我们一般使用其对象也是直接写document

Document对象常用方法

- getElementById() 方法可返回对拥有指定 ID 的第一个对象的引用。
- getElementsByName() 方法可返回带有指定名称的对象的集合.(必须设置name属性)

```
<html>
<head>
<script type="text/javascript">
function getElements()
  {
  var x=document.getElementsByName("myInp
ut");
  alert(x.length);
  }
</script>
</head>
<body>

<input name="myInput" type="text" size="2
0" /><br />
<input name="myInput" type="text" size="2
0" /><br />
<input name="myInput" type="text" size="2
0" /><br />
<br />
<input type="button" onclick="getElement
s()"
value="How many elements named 'myInpu
t'?" />
</body>
</html>
```

- getElementsByTagName() 方法可返回带有指定标签名的对象的集合。

```
<html>
<head>
<script type="text/javascript">
function getElements()
  {
  var x=document.getElementsByTagName("in
put");
  alert(x.length);
  }
</script>
</head>
<body>
<input name="myInput" type="text" size="2
0" /><br />
<input name="myInput" type="text" size="2
0" /><br />
<input name="myInput" type="text" size="2
0" /><br />
<br />
<input type="button" onclick="getElement
s()"
value="How many input elements?" />
</body>
</html>
```

- write() 方法可向文档写入 HTML,注意文档流一旦关闭了,在调用此方法会建立新的文档流进行书写操作,就会覆盖掉之前所有的内容

```
document.write("<h1 style='background :yellow;'>你好</h1>");
```

- 创建元素节点`createElement()`,创建文本节点`createTextNode`,创建属性节点`createAttribute`

```
var node=document.createElement("li");
var textnode=document.createTextNode("Water");
var typ=document.createAttribute("class");
```

Element对象

> 在 HTML DOM （文档对象模型）中，每个部分都是节点：
>
> 1. 文档本身是文档节点
> 2. 所有 HTML 元素是元素节点
> 3. 所有 HTML 属性是属性节点
> 4. HTML 元素内的文本是文本节点

常用方法

```
window.onload = function test(){
    //获取元素节点
    var element = document.getElementsByTagName("a")[0];
    //设置style属性
    element.style.color = "yellow";
    //如果属性中有-字符需要设置此样式
    element.style["text-decoration"] = "none";
    //设置href属性
    element.href = "http://www.sina.com";
    //设置标签内容
    element.innerHTML = "新浪";
}
```

节点操作

- 创建li节点

```
var node=document.createElement("li");
```

- 创建文本节点,内容为北京

```
var textnode=document.createTextNode("北京");
```

- 将文本节点添加到li节点中

```
 node.appendChild(textnode);
```

- 将li节点添加到ul节点中

```
 var element = document.getElementsByTagN
ame("ul")[0];
 element.appendChild(node);
```

- 创建属性节点

```
var att = document.createAttribute("styl
e");
```

- 设置属性值为blue

```
att.nodeValue = "background: blue";
```

- 给li添加属性节点

```
node.setAttributeNode(att);
```

属性和方法

下面的属性和方法可用于所有 HTML 元素上：

属性 / 方法	描述
element.accessKey	设置或返回元素的快捷键。
element.appendChild()	向元素添加新的子节点，作为最后一个子节点。
element.attributes	返回元素属性的 NamedNodeMap。
element.childNodes	返回元素子节点的 NodeList。
element.className	设置或返回元素的 class 属性。
element.clientHeight	返回元素的可见高度。
element.clientWidth	返回元素的可见宽度。
element.cloneNode()	克隆元素。
element.compareDocumentPosition()	比较两个元素的文档位置。
element.contentEditable	设置或返回元素的文本方向。
element.dir	设置或返回元素的文本方向。
element.firstChild	返回元素的首个子。
element.getAttribute()	返回元素节点的指定属性值。
element.getAttributeNode()	返回指定的属性节点。
element.getElementsByTagName()	返回拥有指定标签名的所有子元素的集合。
element.getFeature()	返回实现了指定特性的 API 的某个对象。
element.getUserData()	返回关联元素上键的对象。
element.hasAttribute()	如果元素拥有指定属性，则返回true，否则返回 false。
element.hasAttributes()	如果元素拥有属性，则返回 true，否则返回 false。
element.hasChildNodes()	如果元素拥有子节点，则返回 true，否则 false。
element.id	设置或返回元素的 id。
element.innerHTML	设置或返回元素的内容。
element.insertBefore()	在指定的已有的子节点之前插入新节点。
element.isContentEditable	设置或返回元素的内容。
element.isDefaultNamespace()	如果指定的 namespaceURI 是默认的，则返回 true，否则返回 false。
element.isEqualNode()	检查两个元素是否相等。
element.isSameNode()	检查两个元素是否是相同的节点。
element.isSupported()	如果元素支持指定特性，则返回 true。
element.lang	设置或返回元素的语言代码。
element.lastChild	返回元素的最后一个子元素。

element.namespaceURI	返回元素的 namespace URI。
element.nextSibling	返回位于相同节点树层级的下一个节点。
element.nodeName	返回元素的名称。
element.nodeType	返回元素的节点类型。
element.nodeValue	设置或返回元素值。
element.normalize()	合并元素中相邻的文本节点，并移除空的文本节点。
element.offsetHeight	返回元素的高度。
element.offsetWidth	返回元素的宽度。
element.offsetLeft	返回元素的水平偏移位置。
element.offsetParent	返回元素的偏移容器。
element.offsetTop	返回元素的垂直偏移位置。
element.ownerDocument	返回元素的根元素（文档对象）。
element.parentNode	返回元素的父节点。
element.previousSibling	返回位于相同节点树层级的前一个元素。
element.removeAttribute()	从元素中移除指定属性。
element.removeAttributeNode()	移除指定的属性节点，并返回被移除的节点。
element.removeChild()	从元素中移除子节点。
element.replaceChild()	替换元素中的子节点。
element.scrollHeight	返回元素的整体高度。
element.scrollLeft	返回元素左边缘与视图之间的距离。
element.scrollTop	返回元素上边缘与视图之间的距离。
element.scrollWidth	返回元素的整体宽度。
element.setAttribute()	把指定属性设置或更改为指定值。
element.setAttributeNode()	设置或更改指定属性节点。
element.setIdAttribute()	
element.setIdAttributeNode()	
element.setUserData()	把对象关联到元素上的键。
element.style	设置或返回元素的 style 属性。
element.tabIndex	设置或返回元素的 tab 键控制次序。
element.tagName	返回元素的标签名。
element.textContent	设置或返回节点及其后代的文本内容。
element.title	设置或返回元素的 title 属性。
element.toString()	把元素转换为字符串。
nodelist.item()	返回 NodeList 中位于指定下标的节点。
nodelist.length	返回 NodeList 中的节点数。

Event 对象

事件句柄 (Event Handlers)

HTML 4.0 的新特性之一是能够使 HTML 事件触发浏览器中的行为，比如当用户点击某个 HTML 元素时启动一段 JavaScript。下面是一个属性列表，可将之插入 HTML 标签以定义事件的行为。

属性	此事件发生在何时...
onabort	图像的加载被中断。
onblur	元素失去焦点。
onchange	域的内容被改变。
onclick	当用户点击某个对象时调用的事件句柄。
ondblclick	当用户双击某个对象时调用的事件句柄。
onerror	在加载文档或图像时发生错误。
onfocus	元素获得焦点。
onkeydown	某个键盘按键被按下。
onkeypress	某个键盘按键被按下并松开。
onkeyup	某个键盘按键被松开。
onload	一张页面或一幅图像完成加载。
onmousedown	鼠标按钮被按下。
onmousemove	鼠标被移动。
onmouseout	鼠标从某元素移开。
onmouseover	鼠标移到某元素之上。
onmouseup	鼠标按键被松开。
onreset	重置按钮被点击。
onresize	窗口或框架被重新调整大小。
onselect	文本被选中。
onsubmit	确认按钮被点击。
onunload	用户退出页面。

- 文本框获取焦点后使其背景颜色变成红色
- 文本框失去光标的时候,在其后显示文本框中的内容
- 点击按钮的时候先判断文本框中内容是否为空,如果不为空做提交操作
- 制作表格,鼠标移上去的时候当前行背景颜色为黄色,移出后背景颜色为默认的白色

对于元素的操作

- 提交表单

```
<html>
<head>
<script type="text/javascript">
function formSubmit()
  {
  document.getElementById("myForm").submi
t()
  }
</script>
</head>
<body>
<form id="myForm" action="js_form_actio
n.asp" method="get">
Firstname: <input type="text" name="first
name" size="20"><br />
Lastname: <input type="text" name="lastna
me" size="20"><br />
<br />
<input type="button" onclick="formSubmi
t()" value="Submit">
</form>
</body>
</html>
```

作业

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<script type="text/javascript">
 var cities = new Array(3);
 cities[0] = new Array("郑州","开封","洛
阳","南阳","商丘");
 cities[1] = new Array("保定","石家庄","邢
台");
 cities[2] = new Array("西安","渭南","铜
川","宝鸡","咸阳");
 function changeProvince(index) {
    var theCity = document.getElementByI
d("city");
    city.innerHTML = "<option value=''>请
选择城市</option>";
    for(var i = 0; i < cities[index].leng
th; i++){
        var node=document.createElemen
t("option");
        var textnode=document.createText
Node(cities[index][i]);
        node.appendChild(textnode);
        city.appendChild(node);
    }
 }
 function show(){
     var theProvince = document.getElemen
```

```
tById("province");
      var theCity = document.getElementById("city");
      alert(theCity.value);
 }
 </script>
<title>标题</title>
</head>
<body style="background: red;">
<select onchange="changeProvince(this.value)" id="province">
    <option value="">请选择省份</option>
    <option value="0">河南省</option>
    <option value="1">河北省</option>
    <option value="2">陕西省</option>
</select>
<select id="city" onchange="show()">
    <option value="">请选择城市</option>
</select>
</body>
</html>
```