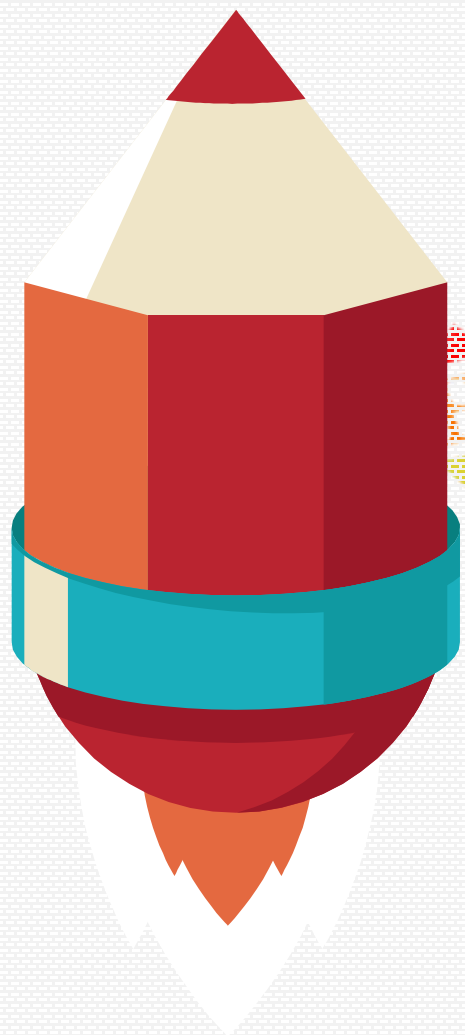




# 第四课：浏览器事件和函数应用

■ 主讲老师：万章



## 目录

函数的基本概念

浏览器事件基础讲解

元素的CSS操作

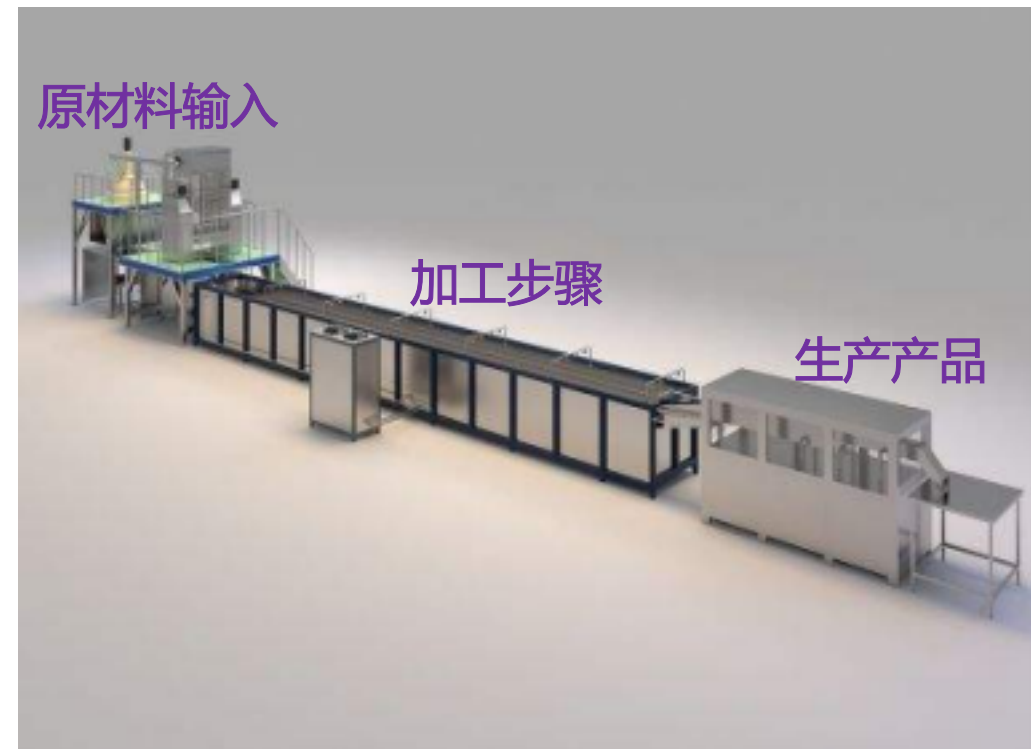


# 函数的基本概念

# 函数的基本概念



- 1: 输入
- 2: 步骤
- 3: 输出



辣条工厂  
(工厂名称)

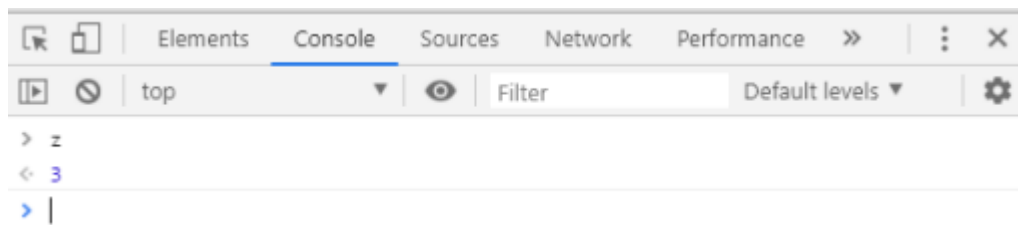
# 函数的调用

```
<script>
  var x=1;
  var y=2;

  function factory(arguments1,arguments2){
    var data;
    data=arguments1+arguments2;
    return data;//返回一个数据
  }

  var z;

  z=factory(x,y);
</script>
```



创建变量x,赋值为1

创建变量y,赋值为2

创建名称为**factory**的函数  
(函数名称也是一个变量,变量的值就是一个函数数据)  
这个函数要运行需要输入两个参数,这些参数也是变量,函数内部预设了一些处理步骤;  
如果函数收到了参数并且执行了,那么内部的处理步骤就会开始执行

运行函数**factory**  
运行函数的方式是:函数名+()  
运行函数的时候我们传了两个变量xy  
变量x的值复制了一份并赋值给arguments1  
变量y的值复制了一份并赋值给arguments2  
函数运行完成之后,arguments1和arguments2的值都会被清空,下一次运行还需要再次传入数据

# 函数的调用

```
<script>
  var x=1;
  var y=2;

  function factory(){
    var data;
    data=x+y;
    return data;//返回一个数据
  }

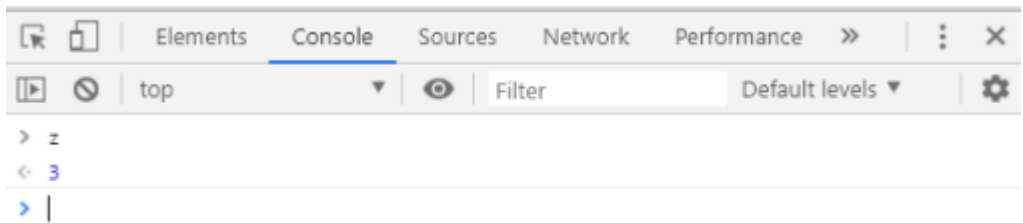
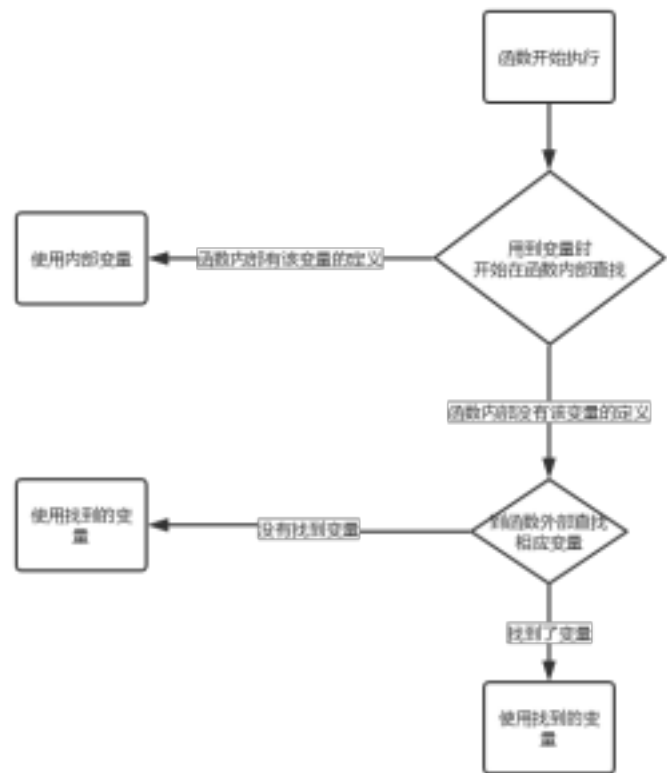
  var z;

  z=factory();
</script>
```

创建变量x,赋值为1

创建变量y,赋值为2

创建名称为factory的函数  
(函数名称也是一个变量,变量的值就是一个函数数据)  
如果创建函数时并没有要求传入什么参数,那么如果函数内部需要用到某个变量,那么会按照如下顺序查找:





# 浏览器事件基础讲解

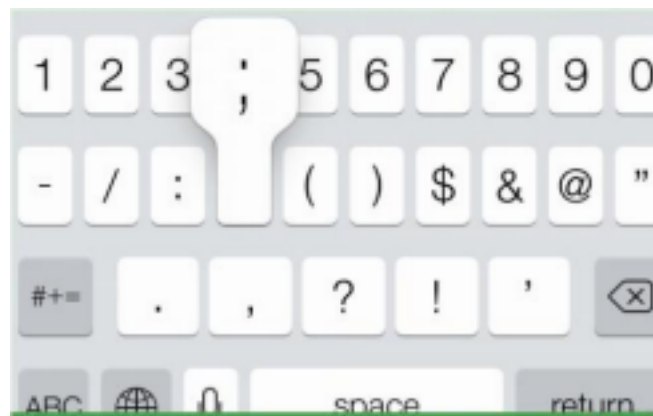
## 浏览器事件的基本概念



鼠标点击



鼠标拖拽



键盘输入

所谓的事件就是用户在一个网页上所做出的某些操作动作，  
操作动作就是事件



# 事件的作用



劳动委员

谁发生

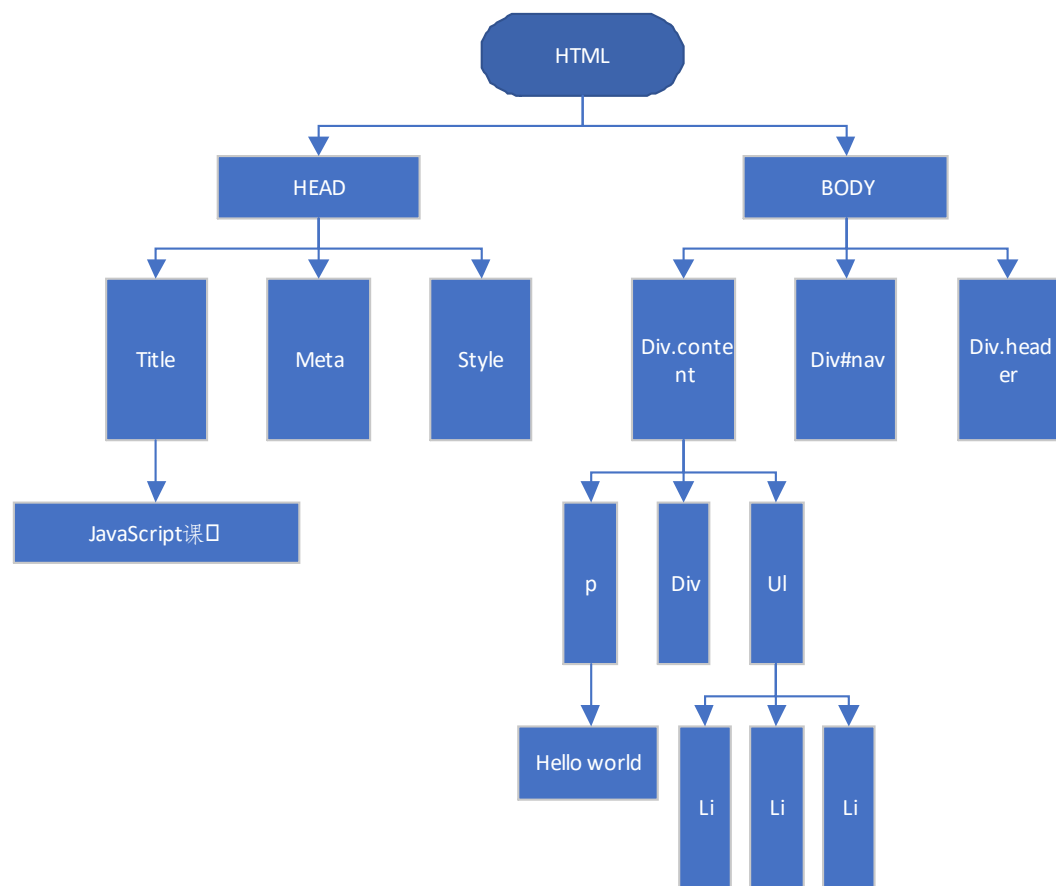
每天下午4点

什么事件

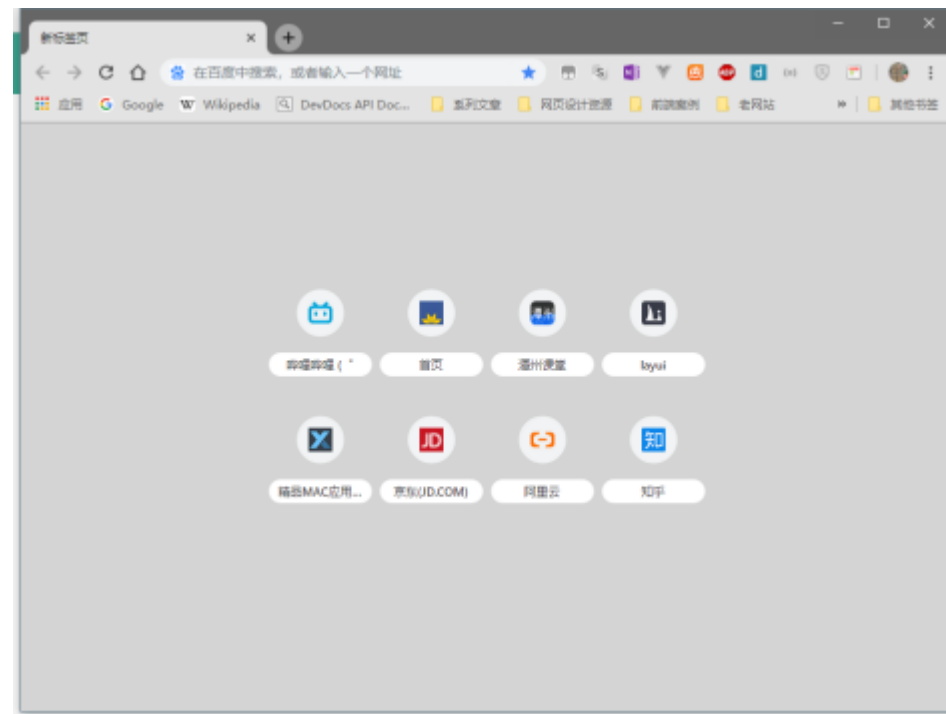
扫地

做什么

# 前端中会发生事件的对象



DOM (文档对象模型 document object model)  
比如document, 某个元素



BOM (浏览器对象模型 Browser object model)  
window  
指用户操作浏览器做哪些动作, 如滚动, 加载, 窗口变化

# DOM的常见事件

| 事件                 | 事件名  |
|--------------------|------|
| onclick            | 左键单击 |
| ondblclick         | 左键双击 |
| onmouseover        | 鼠标移入 |
| onmouseenter(推荐使用) |      |
| onmouseout         | 鼠标移出 |
| onmouseleave(推荐使用) |      |
| onmousedown        | 鼠标按下 |
| onmousemove        | 鼠标移动 |
| oncontextmenu      | 右键单击 |
| onmouseup          | 鼠标抬起 |

鼠标事件

| 事件       | 事件名    |
|----------|--------|
| onfocus  | 获取焦点后  |
| onblur   | 失去焦点   |
| onchange | 内容发生变动 |
| onreset  | 重置后    |
| onselect | 选择后    |
| onsubmit | 提交后    |

表单事件

| 事件         | 事件名  |
|------------|------|
| onkeydown  | 按键按下 |
| onkeypress |      |
| onkeyup    | 按键抬起 |

键盘事件

# BOM的常见事件

| 事件       | 事件名   |
|----------|-------|
| onload   | 加载完之后 |
| onerror  | 加载出错后 |
| onresize | 窗口改变时 |
| onscroll | 滚动时   |

系统事件

# DOM事件的使用方式

```
<div class="wrap"></div>
<script>
  function factory() {
    //步骤1
    //... ..
  }
  document.getElementsByClassName("wrap")[0].onclick = factory //点击
  document.getElementsByClassName("wrap")[0].onmouseover = factory //鼠标移入
  document.getElementsByClassName("wrap")[0].onmouseenter = factory //鼠标移入
</script>
```

谁发生

什么事件

做什么

## BOM事件的使用方式

```
<script>
    function factor(){
        //步骤1
        //步骤2
        //...
    }
    window.onresize=factor;
</script>
```

谁发生

什么事件

做什么



# 元素的CSS操作

## 元素的CSS操作

改变元素样式方法 外部样式 内部样式 行内样式

- 外部样式:单纯的JS不能操作外部样式
- 内部样式:JS可以操作HTML页面任何元素,可以操作style标签
- 行内样式:JS操作元素样式最常用方法,标签的自带属性style 通过点操作 元素.style.css属性 = "属性值"

```
<div id="wrap">123</div>
<script>
    var oWrap = document.getElementById("wrap");
    oWrap.style.color = "red"//单个属性
    oWrap.css.backgroundColor = "blue"//单个属性 分样式用驼峰写法
    oWrap.style.cssText = "font-size:25px;font-weight:bold;"//多个属性
    //个别样式有兼容以及保留字css属性
</script>
```



# 函数的调用方式1

```
<div id="wrap">万章大帅比</div>
```

```
<script>
```

```
var oWrap = document.getElementById("wrap");
```

```
function factor(){
```

```
    oWrap.style.color = "yellow" //单个属性
```

```
    oWrap.style.backgroundColor= "blue" //单个属性 分样式用驼峰写法
```

```
}
```

```
oWrap.onclick=factor;
```

```
</script>
```

创造变量oWrap,赋值为网页上一个id为wrap元素

创造名称为factor的函数

(函数名称也是一个变量,变量的值就是一个函数数据)

如果函数没有返回值,那么说明这个函数是一个纯操作类的函数,此类函数的目的就是为了操作外部的某些变量

当wrap元素被鼠标点击的时候就执行函数factor

## 函数的调用方式2

```
<div id="wrap">万章大帅比</div>
<script>
  var oWrap = document.getElementById("wrap");

  oWrap.onclick=function(){
    oWrap.style.color = "yellow" //单个属性
    oWrap.style.backgroundColor= "blue" //单个属性 分样式用驼峰写法
  };
</script>
```

创建变量oWrap,赋值为网页上一个id为wrap元素

如果某个函数只会在某个触发条件下,或是某个事件触发下才会执行,我们可以不用给函数命名,我们直接在触发动作后新建一个函数  
这种没有名字的函数叫做匿名函数

# 效果



```
<!doctype html>
<html lang="en">
  <head>_</head>
  <body style>
    ...
    <div id="wrap">万章大师比</div> == $0
    <script>_</script>
    <!-- Code injected by live-server -->
    <script type="text/javascript">_</script>
  </body>
</html>
```

html body div#wrap

Styles Event Listeners DOM Breakpoints Properties Accessibility

Filter :hov .cls +

element.style {

#wrap {

width: 400px;

height: 400px;

margin: 100px auto;

background-color: red;

}

demo.html:10

margin 100

border -

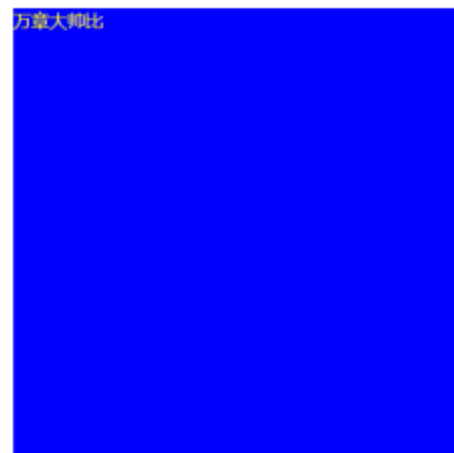
padding -

400 x 400

180

100

点击前



```
<!doctype html>
<html lang="en">
  <head>_</head>
  <body style>
    ...
    <div id="wrap" style="color: yellow; background-color: blue;">万章大师比
    </div> == $0
    <script>_</script>
    <!-- Code injected by live-server -->
    <script type="text/javascript">_</script>
  </body>
</html>
```

html body div#wrap

Styles Event Listeners DOM Breakpoints Properties Accessibility

Filter :hov .cls +

element.style {

color: yellow;

background-color: blue;

#wrap {

width: 400px;

height: 400px;

margin: 100px auto;

background-color: red;

}

demo.html:10

margin 100

border -

padding -

400 x 400

180

100

点击后



作业

## 作业要求

按钮1

按钮2

按钮3

元素

要求: 我们对不同的按钮做一些不同的动作时, 会给下方的元素设置一些特定的样式, 按钮的数量不限

例: 点击按钮1, 下面的元素颜色变成绿色