# Web开发（二）

--- 1-7 DOM模型（一）

河北师范大学软件学院
Software College of Hebei Normal University

# BOM模型

- BOM 是操作浏览器窗口等一些相关操作的接口
- Document 对象，可以获得一些特定的标签并且对其进行操作
  - 只能获得特定标签，不能获得 HTML 中的任意标签，如 标签
  - 通过数组索引方式获得特定标签，不方便维护
- 如何方便地操作 HTML 文档，动态修改文档内容？

# 内容提纲

- DOM 简介
- DOM 树和 DOM 节点
- 访问 DOM 节点

# DOM简介

- DOM（Document Object Model）：文档对象模型
  - 浏览器提供的操作 HTML 文档内容的应用程序接口
  - 用于对文档进行动态操作，如增加文档内容、删除文档内容、修改文档内容等
- DOM 的应用十分广泛，各种网页特效均有 DOM 的踪影

# 内容提纲

- DOM 简介
- DOM 树和 DOM 节点
- 访问 DOM 节点

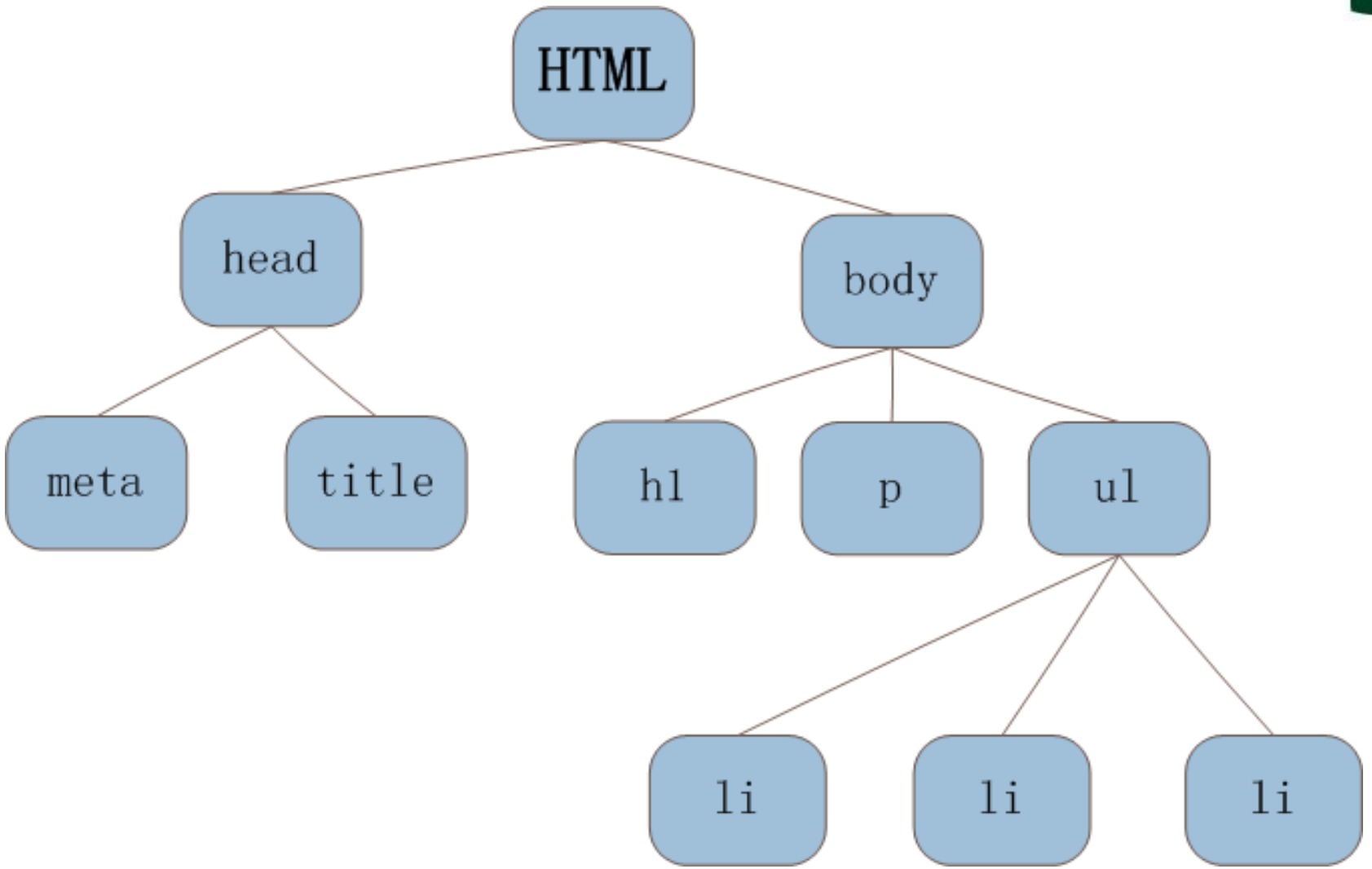
# DOM树

- DOM 将 HTML 文档抽象为树形结构，称这棵树为 DOM 树
- HTML 中的每一项内容（标签和内容）都可以在 DOM 树中找到
- DOM 的核心就是对 DOM 树的操作，即增加、删除、修改 DOM 树中的内容

# DOM树

```
<html>
    <head>
        <mate charset = ‘utf-8’ />
        <title>DOM树型结构</title>
    </head>
    <body>
        <h1>DOM树型结构演示</h1>
        <p>DOM树可执行的操作</p>
        <ul>
            <li>增加文档内容</li>
            <li>删除文档内容</li>
            <li>修改文档内容</li>
        </ul>
    </body>
</html>
```

# DOM树



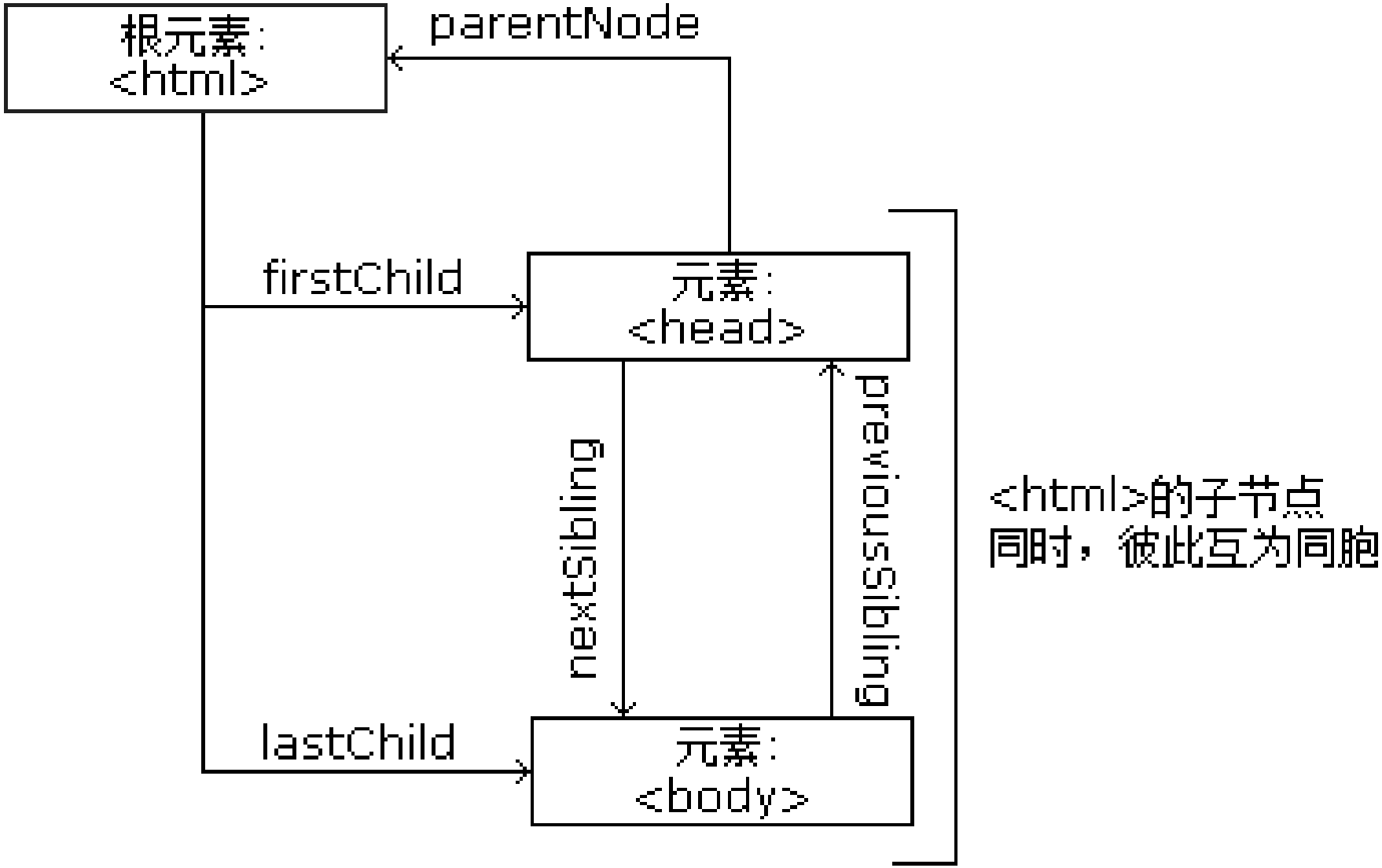
# DOM节点需要讨论的问题

- DOM节点是一个对象（属性和方法）
- DOM节点有三类：元素节点、属性节点、文本节点
- DOM节点之间有特定关系（父子兄弟关系）
- DOM节点核心问题：
  - 如何获取一个节点（节点对象）
  - 如何访问节点对象之间的依赖关系
  - 如何动态添加、删除、更新一个节点

# 节点的层次关系

- 在节点树中，顶端节点被称为根（root）
- 父节点
  - 每个节点都有父节点、除非该元素是文档的根节点。
- 子节点
  - 每个元素节点可以有0个、1个或多个子节点。
- 同胞节点
  - 指拥有相同父节点的节点。

# 节点父、子关系


根元素:
<html>
parentNode
firstChild
元素:
<head>
nextsibling
previoussibling
lastChild
元素:
<body>
<html>的子节点
同时，彼此互为同胞

# DOM节点

DOM节点是一个对象，拥有属性和方法

**属性**是节点（HTML 元素）的值，能够获取或设置。

**方法**是可以在节点（HTML 元素）上执行的动作。

# 内容提纲

- DOM简介
- DOM树和DOM节点
- 访问DOM节点

# 访问DOM节点


DOM
节点
直接
获取
节点
通过id属性获得节点
通过标签名获得所有同名标签
通过类名获得所有类名相同的标签
通过
节点
关系
获取
通过父节点获得子节点
通过子节点获得父节点
获得前后兄弟节点

# 1. 直接获取节点

① 通过id属性获得节点

document.getElementById( )

② 通过标签名获得所有同名标签

document.getElementsByTagName( )

③ 通过类名获得所有类名相同的标签

document.getElementsByClassName( )

# ① getElementById( )方法

- 通过id属性获得节点

语法：

node.getElementById("id");

例子：

document.getElementById("intro");

demo1-7-1

# ② getElementsByTagName( )方法

- 通过标签名查找 HTML 元素。
- 返回带有指定标签名的所有元素（数组或列表）

语法：

node.getElementsByTagName（“tagname”）;

例子：

document.getElementsByTagName("p");

## ② getElementsByTagName( )方法

例1： **x=document.getElementsByTagName(" p");**

```
for (i=0;i<x.length;i++) {
    document.write(x[i].innerHTML);
    document.write("<br/>");
}
```

例2： demo1-7-2

# ③ getElementsByClassName( )方法

- 通过类名获得所有类名相同的标签

语法：

node.getElementsByClassName（“class”）;

例子：

```
document.getElementsByClassName("intro");
```

# 2. 通过节点关系访问节点

通过父节点获得子节点：

node.children[ ]　　node.childNodes[ ]

node.firstChild　　node.lastChild

通过子节点获得父节点：

node.parentNode

获得前后兄弟节点：

node.previous(next)Sibling

# ①通过父节点获得子节点

① children属性——返回节点的所有元素子节点集合。之后可以通过循环或者索引找到需要的元素节点。

例：document.getElementById("myList").children;

document.getElementById("myList").children[0];

② childNodes 属性——返回节点的子节点集合。之后可以通过循环或者索引找到需要的节点。

例：document.getElementById("myList").childNodes;

document.getElementById("myList").childNodes[0];

# ①通过父节点获得子节点

③ firstChild 属性——返回指定节点的首个子节点。可递归使用，即支持 parentObj.firstChild.firstChild... 的形式，如此可获得更深层次的节点。

例：document.getElementById("myList").firstChild;

④ lastChild 属性——返回指定节点的最后一个子节点。可递归使用

例：document.getElementById("myList").lastChild;

demo1-7-3

# 实例代码

- 重新做 demo1-7-2
  - 使用更合理的方式实现图片更换

# ②通过子节点获得父节点

node.parentNode——返回指定节点的父节点

**html代码:**

```
<h1>新闻动态<span id="time">2016-10-7</span></h1>
```

**js代码：**

```
var noder=document.getElementById('time').parentNode;
alert(noder.innerHTML);
```

**输出结果：**

```
新闻动态<span id="time">2016-10-7</span>
```

# ③获得前后兄弟节点

- previousSibling属性

——返回同一层级中指定节点的前一个节点。

**例：document.getElementById("item2").previousSibling;**

- nextSibling属性

——返回同一层级中指定节点之后紧跟的一个节点

**例： document.getElementById("item1").nextSibling;**

demo1-7-4

# 访问DOM节点属性

- 获得某一元素节点的属性节点
  - 标准方式获得属性：node.getAttribute( *name* )
  - 简单方式获得属性：node.*attrName*
- 修改某一元素节点的属性节点
  - 直接赋值给属性

# 获得某一元素节点的属性节点

❶ 标准方式获得节点属性：getAttribute( ) 方法

- 返回指定属性名的属性值。

例：<a id=“aa” href=“#" target="_blank"></a>

```
var result=document.getElementById("aa").getAttribute("target");
alert(result)；//结果：_blank
```

❷ 简单方式获得节点属性：node.attrName

例：<img id="image" src="smiley.gif">

```
document.getElementById("image").src="landscape.jpg";
```

# 修改某一元素节点的属性节点

- 修改某一元素节点的属性节点——直接赋值给属性

例：

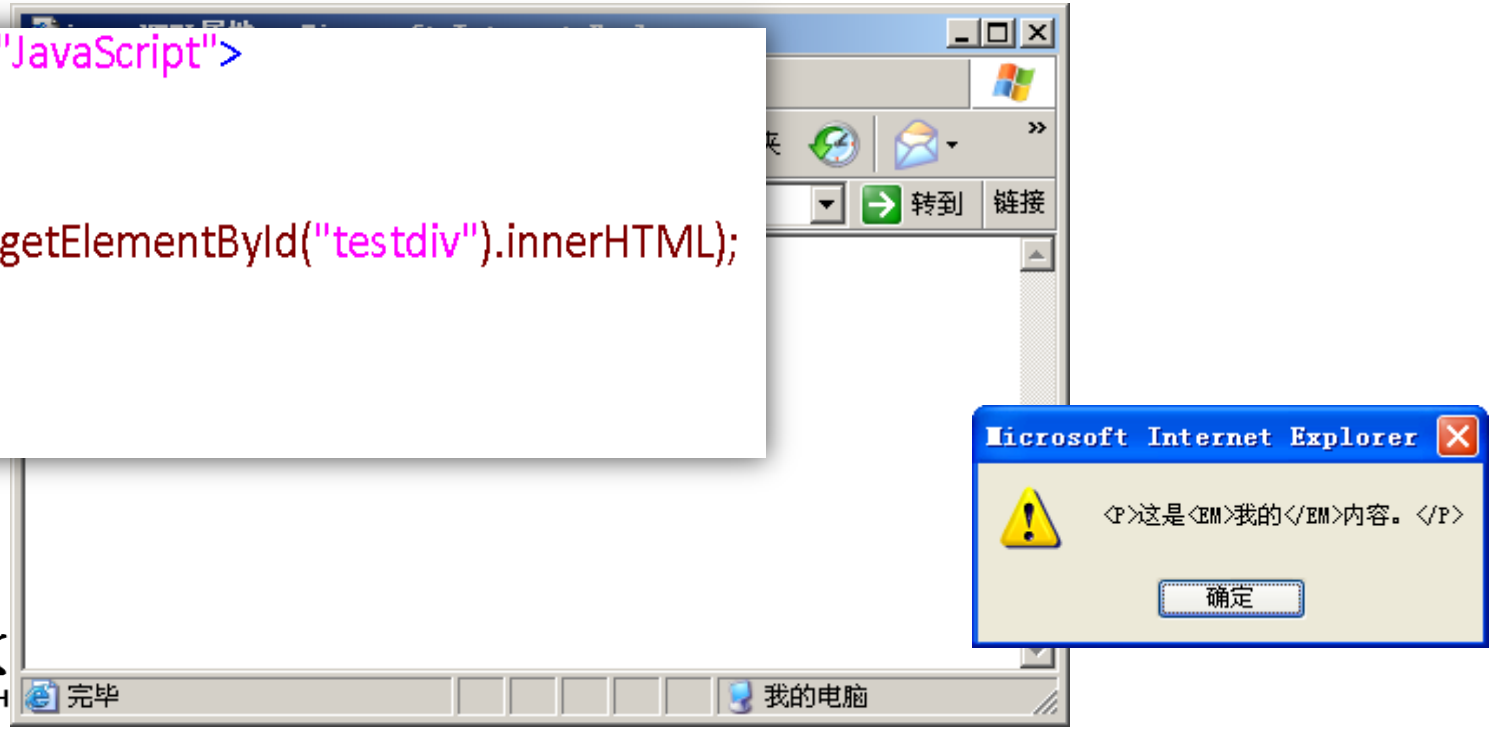
```
<img id="image" src="smiley.gif">
document.getElementById("image").src="landscape.jpg";
```

# innerHTML 属性

- 改变 HTML 内容—innerHTML 属性

**innerHTML是DOM中元素节点的属性，相当于一个容器，**可用于获取或改变任意 HTML 元素，包括 <html> 和 <body>

```
<script language="JavaScript">
<!--
function outPut(){
  alert(document.getElementById("testdiv").innerHTML);
  }
//-->
</script>
```



# innerHTML 属性

- innerHTML属性，可读可写
  - 读取节点内容：*node*.innerHTML
  - 修改节点内容：*node*.innerHTML = “”；
  - 为该节点添加一个 元素：*node*.innerHTML += “ … ”；
- 操作简单，几乎所有浏览器均支持

# innerHTML 属性

例子1：

var txt=document.getElementById("intro").innerHTML;

document.write(txt);

例子2：

document.getElementById(“intro”).innerHTML= “hello” ;

demo1-7-5

# DOM操作小结

- 获取带有指定 id 的节点（元素）- getElementById(id)
- 通过标签名获得所有同名标签-getElementsByTagName( )
- 通过类名获得节点-document.getElementClassName( )
- 通过父节点获得子节点：
  - node.children[ ]
  - node.childNodes[ ]
  - node.firstChild
  - node.lastChild

# DOM操作小结

- 通过子节点获得父节点：
  - node.parentNode
- 获得前后兄弟节点：
  - node.previous(next)Sibling
- 获得某一元素节点的属性节点
  - 标准方式获得属性：node.getAttribute( *name* )
  - 简单方式获得属性：node.*attrName*
- 修改某一元素节点的属性节点
- 改变 HTML 内容—innerHTML 属性
  - 直接赋值给属性

# DOM操作小结

- DOM操作注意事项：
  - 获取DOM节点的操作要在标记被浏览器加载之后进行
- 通用性强，几乎所有浏览器均支持
- 不仅可以操作HTML文档，也可以操作XML文档
- 操作稍嫌复杂，书写的代码量过大

# Thank You !