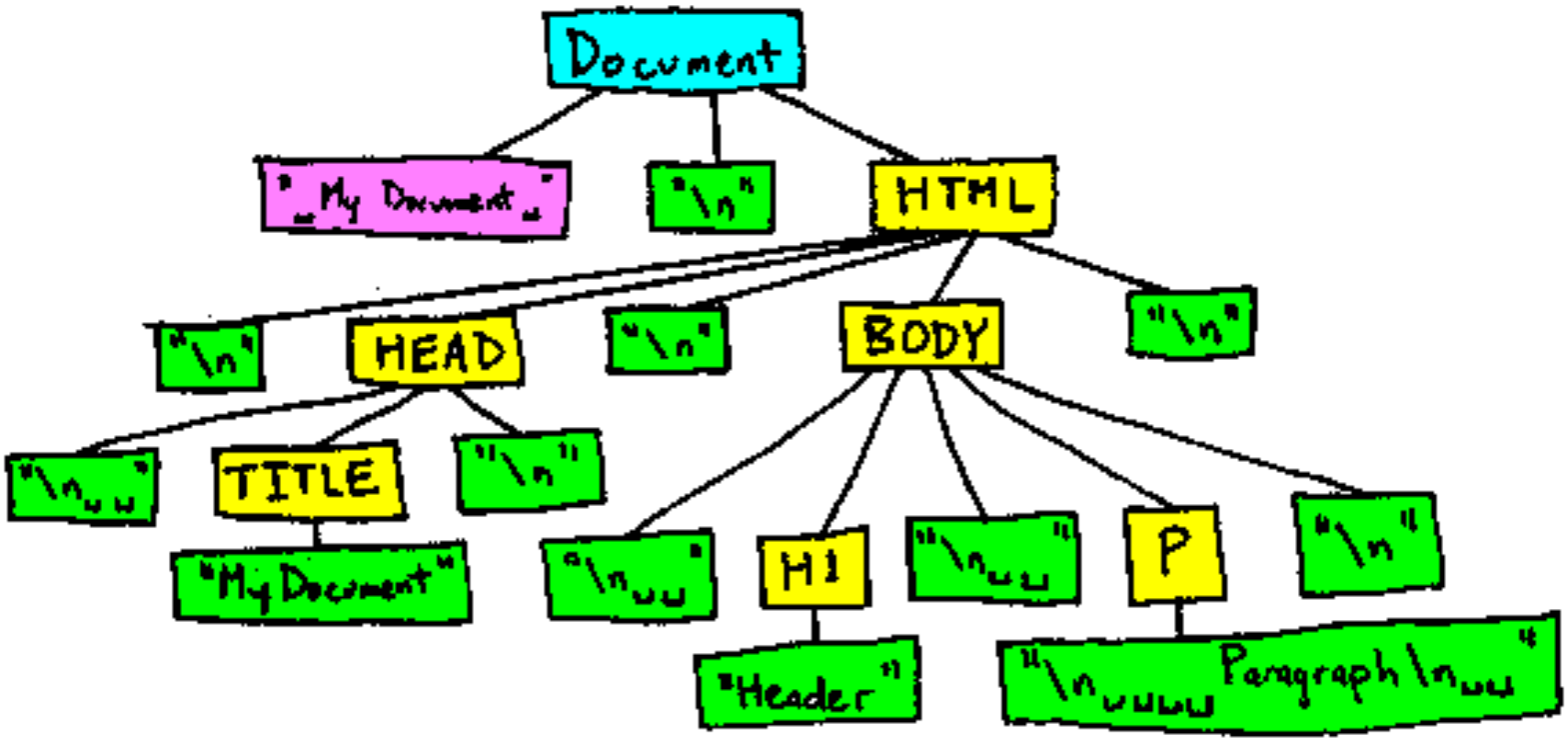
XML 原理与应用

Summer X

RUC


Document
"␣My␣Document␣"
"\n"
HTML
"\n"
HEAD
"\n"
BODY
"\n"
"\n␣␣"
TITLE
"\n"
"\n␣␣"
"My␣Document"
"\n␣␣"
H1
"\n␣␣"
P
"\n"
"Header"
"\n␣␣␣␣Paragraph␣1\n␣␣"

本章学习目标

- 了解 DOM 的特点及规范级别
- 掌握常见的 DOM 对象
- 熟悉浏览器操纵 HTML 文档和 XML 文档的基本方法

目录

9.1 DOM 概述

- DOM——Document Object Model
  - 文档对象模型的缩写，把整个 XML 文档表示成一棵由结点组成的层次树，方便人们通过编程语言进行操纵处理
  - 面向程序员实现对 XML 的操纵处理

DOM 的优点

- DOM 优点
  - DOM 以易于程序员理解的结点树表示 XML 文档，简化了文档操作
  - DOM 提供了语言中立的标准接口，降低了学习和交流成本
- 其他处理技术
  - SAX：Simple API for XML
  - StAX：Streaming API for XML

DOM 历史及其规范级别

- DOM Level 0
- DOM Level 1
- DOM Level 2
- DOM Level 3

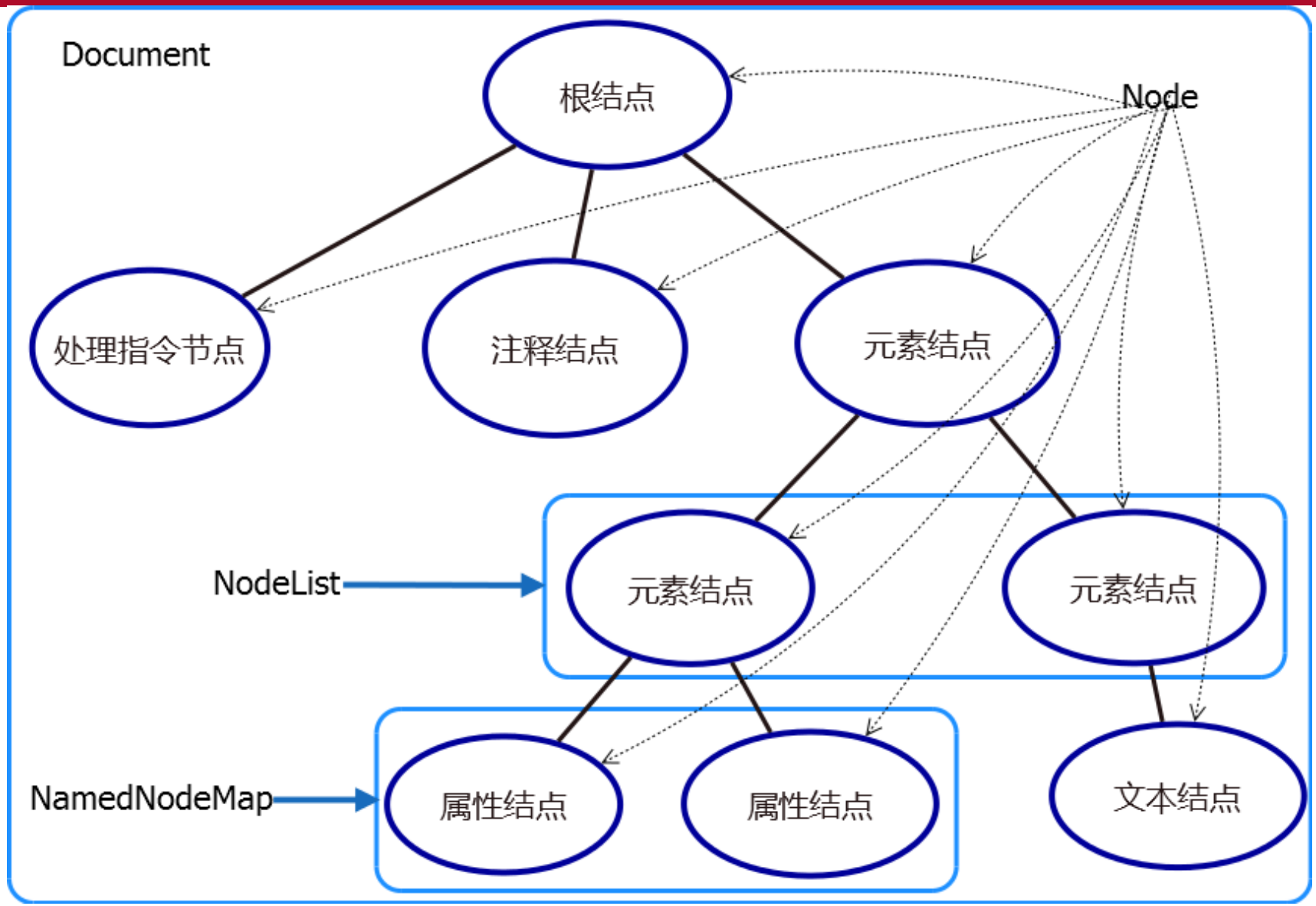
9.2 DOM 基本对象

对象名称	描述
Attr	表示一个属性结点
Document	表示 XML 文档整体
DocumentFragment	表示一个 XML 文档片段结点
DocumentType	表示 <!DOCTYPE> 元素结点
Entity	表示 XML 文档中的一个实体结点
EntityReference	表示 XML 文档中的一个实体引用结点
Element	表示 XML 文档中的一个元素结点
NamedNodeMap	由若干个属性名称和属性值构成的无序集合
Node	表示文档树中的任一个结点
NodeList	表示一组结点构成的有序列表
Notation	表示一个 Notation 结点
ProcessingInstruction	表示一个处理指令结点
Text	表示元素或属性的文本内容结点

XML 文档与 DOM 树 I

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!-- DOM Demo -->
3 <book>
4   <info author="乔治•伽莫夫" isbn="978-7-03-010759-6"/>
5   <title>从一到无穷大</title>
6 </book>
```

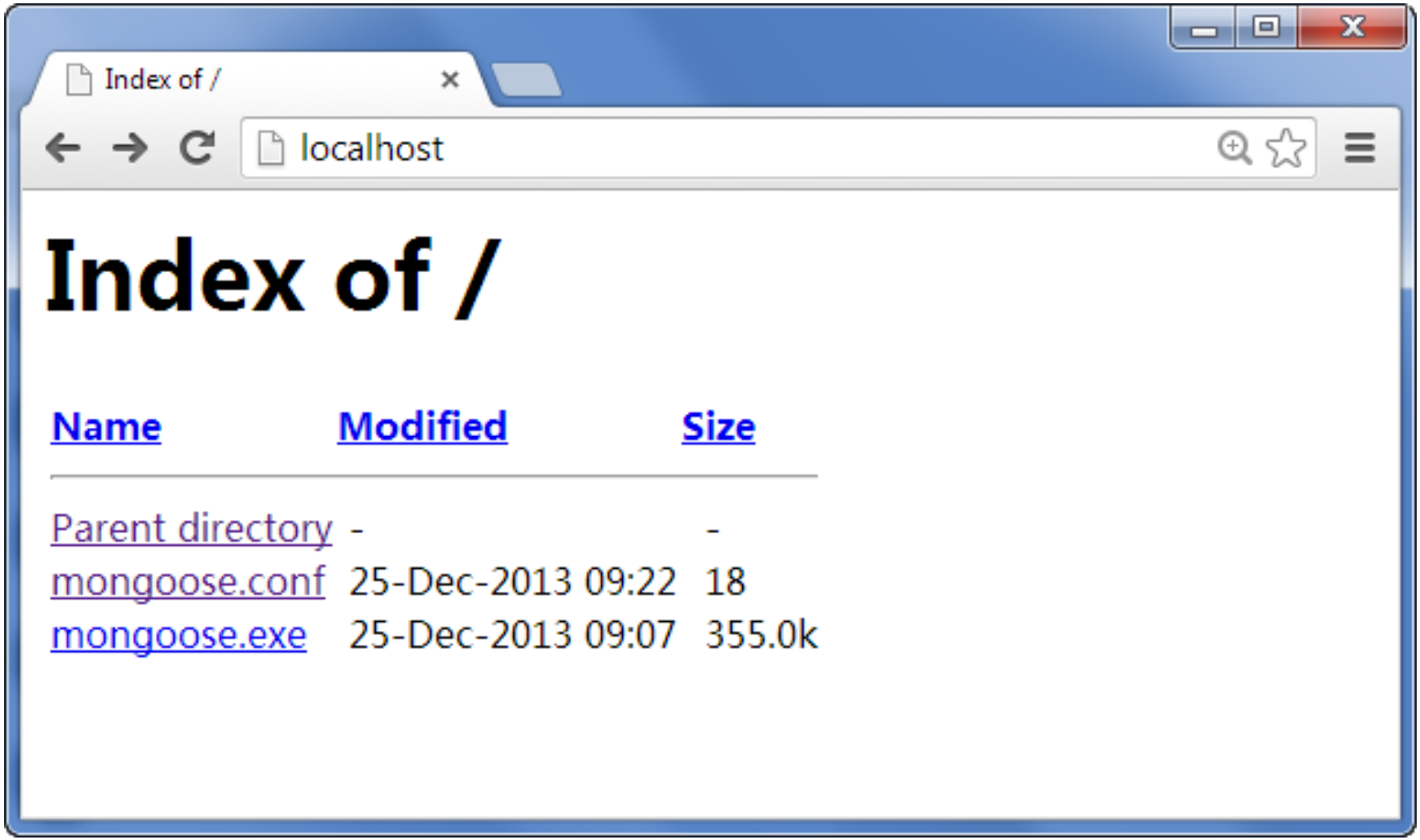
XML 文档与 DOM 树 II



9.3 利用 Mongoose 搭建 DOM 测试环境

- Mongoose
  - 跨平台的 Web 服务器
- 为什么需要 Web 服务器
  - 访问拒绝错误
- 搭建步骤
  - 参考教材

利用浏览器访问本地搭建的 Mongoose 服务器



9.4 利用 DOM 操纵 HTML

- HTML DOM 及元素定位方法
- 改变元素结点内容
- 改变属性结点内容
- 综合示例

9.4.1 HTML DOM 及元素定位方法 I

- HTML 示例文档：

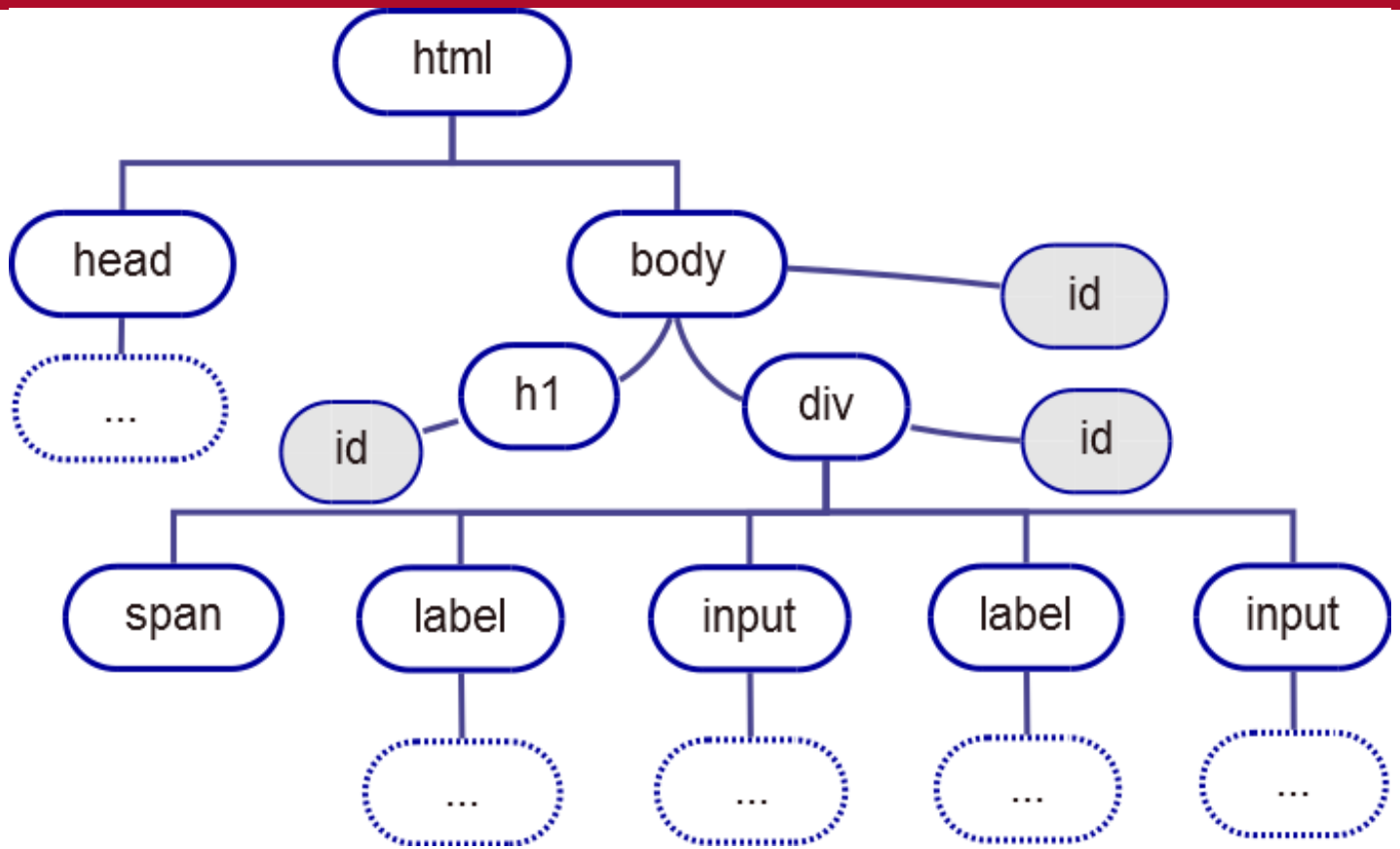
```
1  <html>
2    <head>
3      <title>HTML DOM Example</title>
4      <meta http-equiv="Content-Type"
        content="text/html;charset=utf-8" />
5      <script type="text/javascript" src="jquery.js"></script>
6    </head>
7    <body id="main">
8      <h1 id="headline">心情选择</h1>
9      <div id="content">
10       <span>今天的心情是：</span>
11       <label for="happy_mood">开心</label>
12       <input id="happy_mood" type="radio" name="mood"
        value="happy"/>
```

9.4.1 HTML DOM 及元素定位方法 II

```
13          <label for="bad_mood">心情不好</label>
14          <input id="bad_mood" type="radio" name="mood"
          value="bad"/>
15        </div>
16      </body>
17  </html>
```

- HTML 文档对于的 HTML DOM 树：

9.4.1 HTML DOM 及元素定位方法 III



定位方法

- getElementById 方法

```
document.getElementById("content");
```

- getElementsByName 方法

```
document.getElementsByTagName(“mood”);
```

- getElementsByTagName 方法

```
document.getElementsByTagName("h1");
```

- jQuery 选择方法

```
1 $('#content'); //选择id为content的元素
2 $('input[name="mood"]'); //选择名称为mood的所有input元素
3 $('h1'); //选择标签名称为h1的所有元素
```

9.4.2 改变元素结点内容

- innerText

```
1 var headline = document.getElementById("headline");
2 headline.innerText = '天天好心情';
```

- innerHTML

```
1 var headline = document.getElementById("headline");
2 headline.innerHTML = '<font color="red">天天好心情</font>';
```

- jQuery 方式

```
1 $('#headline').text('天天好心情');
2 $('#headline').html('<font color="red">天天好心情</font>');
```

9.4.3 改变属性结点内容

- 设置属性
  - getAttribute()
- 获取属性
  - getAttribute()

```
1 var headline = document.getElementById("headline");
2 headline.setAttribute('style', 'color:red;');
3 headline.getAttribute('style');
```

- jQuery 方式

```
1 $('#headline').attr('style','color:red;');
2 $('#headline').attr('style');
```

9.4.4 结点的创建与删除 I

- 创建元素结点
  - createElement( )
- 把新创建的结点附加到特定结点内
  - appendChild( )

```
1 var button = document.createElement("input");
2 button.setAttribute('id', 'guess_button');
3 button.setAttribute('type', 'button');
4 button.setAttribute('value', 'Guess');
5 var container = document.getElementById('content');
6 container.appendChild(button);
```

- 创建一个文本结点
  - createTextNode( )

9.4.4 结点的创建与删除 II

```
1 var textNode = document.createTextNode('静以修身');
2 container.appendChild(textNode);
```

- 删除结点
  - removeChild( )

```
1 container.removeChild(textNode);
```

9.4.5 示例 I

- 代码清单

```
<html>
  <head>
    <title>HTML DOM Example</title>
    <meta http-equiv="Content-Type"
     content="text/html;charset=utf-8" />
    <script type="text/javascript" src="jquery.js"></script>
    <script type="text/javascript">
      $(document).ready(function(){
        var headline = document.getElementById("headline");

        headline.innerText = '天天好心情';
        headline.innerHTML = '<font color="red">天天好心情</font>';

        $('#headline').text('天天好心情');
```

9.4.5 示例 II

```
14          $('#headline').html('<font color="red">天天好心情</font>');
15
16          headline.setAttribute('style', 'color:red;');
17          console.info(headline.getAttribute('style'));
        //获取h1的style属性值
18
19          $('#headline').attr('style','color:red;');
20          console.info($('#headline').attr('style')); //获取h1的style属性值
21
22          var button = document.createElement("input");
23          button.setAttribute('id', 'guess_button');
24          button.setAttribute('type', 'button');
25          button.setAttribute('value', 'Guess');
26          var container = document.getElementById('content');
27          container.appendChild(button);
        //把创建的button结点附加到id为content的元素下
```

9.4.5 示例 III

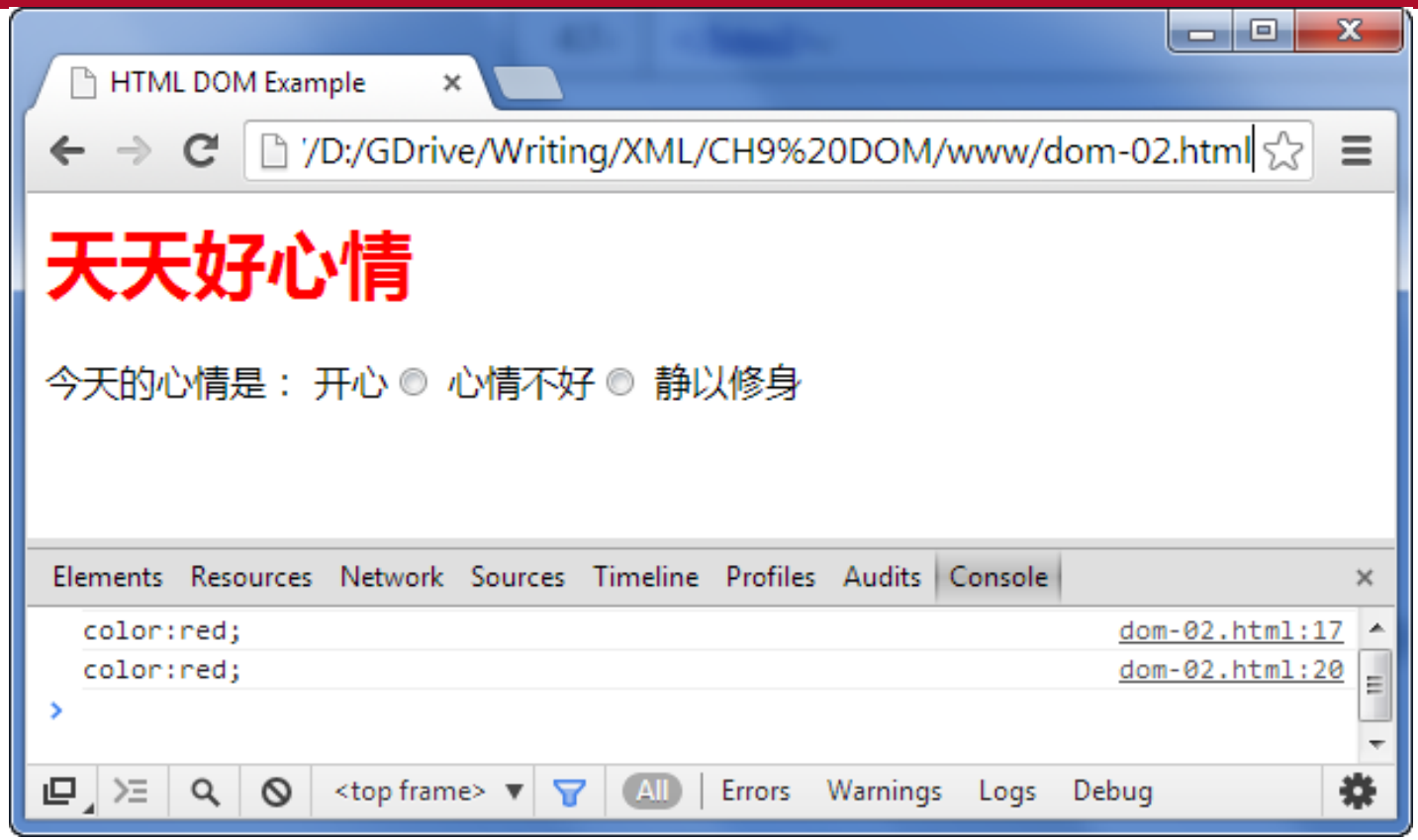
```
28
29          var textNode = document.createTextNode('静以修身');
30          container.appendChild(textNode);
31
32          container.removeChild(button); //删除结点
33        });
34    </script>
35  </head>
36  <body id="main">
37    <h1 id="headline">心情选择</h1>
38    <div id="content">
39      <span>今天的心情是：</span>
40      <label for="happy_mood">开心</label>
41      <input id="happy_mood" type="radio" name="mood"
        value="happy"/>
42      <label for="bad_mood">心情不好</label>
```

9.4.5 示例 IV

```
43          <input id="bad_mood" type="radio" name="mood"
          value="bad"/>
44        </div>
45      </body>
46    </html>
```

- 显示效果

9.4.5 示例 V


HTML DOM Example
/D:/GDrive/Writing/XML/CH9%20DOM/www/dom-02.html
天天好心情
今天的心情是： 开心 心情不好 静以修身
Elements Resources Network Sources Timeline Profiles Audits Console
color:red; dom-02.html:17
color:red; dom-02.html:20
<top frame>
All
Errors Warnings Logs Debug

9.5 利用 DOM 操纵 XML

- 加载 XML 文档
- 结点访问方法
- 结点定位属性
- 结点常用属性
- 结点常用方法
- 示例

示例 XML 文档：books.xml I

```
1  <?xml version="1.0"?>
2  <books>
3    <book isbn="978-1449319793" id="b1">
4      <title lang="EN">Python for Data Analysis</title>
5      <price currency="dollar">25.40</price>
6      <authors>
7        <author>Wes McKinney</author>
8      </authors>
9      <press> O'Reilly Media</press>
10     <pages>470</pages>
11     <description>Python for Data Analysis is concerned with the
         nuts...</description>
12     <cover>book-python.jpg</cover>
13   </book>
14   <book isbn="978-7-115-28282-8" id="b2">
15     <title lang="CHN">数学之美</title>
```

示例 XML 文档：books.xml II

```
16      <price>45.00</price>
17      <authors>
18        <author>吴军</author>
19      </authors>
20      <press>人民邮电出版社</press>
21      <pages>304</pages>
22
        <description>读了“数学之美”，才发现大学时学的数学知识...</descripti
23      <cover>book-math.jpg</cover>
24    </book>
25  </books>
```

9.5.1 加载 XML 文档 I

- XMLHttpRequest
  - IE7+、Firefox、Chrome、Safari 以及 Opera
  - 老版本的浏览器可能不支持
- 自定义的加载 XML 文档函数

```
1 function loadXml(xmlFile) {
2     var xhttp = new XMLHttpRequest();
3     xhttp.open("GET", xmlFile, false);
4     xhttp.send();
5     return xhttp.responseXML;
6 }
```

- XMLHttpRequest 的 open()

9.5.1 加载 XML 文档 II

```
void open(string method, string URL, boolean asynch, string username,
    string password);
```

- method
  - 向服务器发送请求的方式
  - 可以为 GET 或 POST
- URL
  - 所调用的服务器资源的 URL
- asynch
  - 布尔值，说明调用时是异步方式还是同步方式
  - 默认为 true，即异步调用方式
- username 和 password

9.5.1 加载 XML 文档 III

- 在必要时指定访问资源所需要的用户名和口令

- XMLHttpRequest 的 send()

```
void send(content);
```

  - 如 open() 方法中的 asynch 参数为 true，send() 方法就会立即返回
  - 否则它会等待，直到接收到响应为止
  - content 是可选参数

- XMLHttpRequest 的返回结果
  - responseXML
  - responseText

利用 DOMParser 对象从字符串中加载 XML

```
1 function loadXmlFromString(str) {
2     var parser = new DOMParser();
3     var xmlDoc = parser.parseFromString(str, "text/xml");
4     return xmlDoc
5 }
```

9.5.2 结点访问方法

- 获取指定结点下指定标记名称的结点集合

```
node.getElementsByTagName("tagname");
```

  - 返回结果为 NodeList
    - 可通过下标访问每一个结点
    - 可通过 length 属性获取结点集合的大小
    - 示例：

```
1 var titleList = xmlDoc.getElementsByTagName("title");
2 for(var i=0; i<titleList.length; i++) {
3    var titleNode = titleList[i];
4    //其他处理
5 }
```

9.5.3 结点定位属性

- 从当前结点根据结点树关系进行访问
  - parentNode：返回当前结点的父节点。
  - childNodes：该属性用于获取当前结点的直接子结点集合，返回结果为 NodeList 对象，与 getElementsByTagName() 方法相同。
  - firstChild：返回当前结点的第一个孩子结点。
  - lastChild：返回当前结点的最后一个孩子结点。
  - nextSibling：返回当前结点的下一个兄弟结点。
  - previousSibling：返回当前结点的上一个兄弟结点。

示例

```
1 var firstBookNode = xmlDoc.getElementsByTagName('book')[0];
2 for(var node = firstBookNode.firstChild;
3         node!=null;
4         node = node.nextSibling) {
5     console.info(node);
6     if(node==firstBookNode.lastChild) break;
7 }
```

9.5.4 结点常用属性

- nodeName
- nodeValue
- nodeType
- attributes

nodeName

- 元素结点的 nodeName 与标签名相同
- 属性结点的 nodeName 是属性的名称
- 文本结点的 nodeName 永远是“#text”
- 文档结点的 nodeName 永远是“#document”

nodeValue

- 元素结点的 nodeValue 是 undefined
- 文本结点的 nodeValue 是文本自身
- 属性结点的 nodeValue 是属性的值

nodeType

结点类型	常量字符串	数值表示结果
元素结点	ELEMENT_NODE	1
属性结点	ATTRIBUTE_NODE	2
文本结点	TEXT_NODE	3
CDATA 块结点	CDATA_SECTION_NODE	4
实体引用结点	ENTITY_REFERENCE_NODE	5
实体结点	ENTITY_NODE	6
处理指令结点	PROCESSING_INSTRUCTION_NODE	7
注释结点	COMMENT_NODE	8
文档结点	DOCUMENT_NODE	9
文档类型结点	DOCUMENT_TYPE_NODE	10
文档片段结点	DOCUMENT_FRAGMENT_NODE	11
NOTATION 结点	NOTATION_NODE	12

attributes

- 如果当前结点是元素类型的结点，attributes 包含了当前元素的所有属性信息，否则为 null
- attributes 类型：NamedNodeMap
  - 无序结点集合

9.5.5 结点常用方法

- appendChild
- cloneNode
- hasChildNodes
- createElement
- insertBefore
- removeChild
- replaceChild

9.5.6 完整示例 I

- 在浏览器中显示”books.xml” 的主要文档结构
- dom-tree.js

```
1  function loadXmlFromFile(xmlFile) {
2      var xhttp = new XMLHttpRequest();
3      xhttp.open("GET", xmlFile, false);
4      xhttp.send();
5      return xhttp.responseXML;
6  }
7
8  var buffer = '';
9  $(document).ready(function () {
10     var xmlDoc = loadXmlFromFile('books.xml'); //加载xml文档
11     showTree(xmlDoc, 0);    //调用showTree()函数，输出结点树
12     $('#main').html(buffer); //替换网页内容
13 });
```

9.5.6 完整示例 II

```
14
15 function showTree(node, depth) {
16     //输出当前node结点的信息
17     writeNode(node, depth);
18
19     //循环输出当前结点的子结点信息
20     var child = node.firstChild;
21     for(var child=node.firstChild;child!=null; child = child.nextSibling){
22         showTree(child, depth + 4);
23         if(child==node.lastChild) break;
24     }
25 }
26
27 function writeNode(node, depth){
28     if(node.nodeType==3){
29         //忽略文本结点
```

9.5.6 完整示例 III

```
30        return;
31    } else if(node.nodeType==9){ //文档结点
32        //处理文档结点
33        buffer = node.nodeName;
34    } else if(node.nodeType == 1) {
35        //处理元素结点
36        buffer = buffer + '<br/>';
37        for(var d=0; d<depth; d++)
38            buffer = buffer + '+';
39
40        buffer = buffer + node.nodeName;
41
42        //处理元素的属性
43        var attributes = node.attributes;
44        if(attributes.length > 0) {
45            buffer = buffer + '[';
```

9.5.6 完整示例 IV

```
46          for(var i=0; i<attributes.length; i++) {
47              if(i>0) buffer = buffer + ', ';
48              var att = attributes[i];
49              buffer = buffer + att.nodeName + '=' + att.nodeValue;
50          }
51          buffer = buffer + '] ';
52      }
53  }
54 }
```

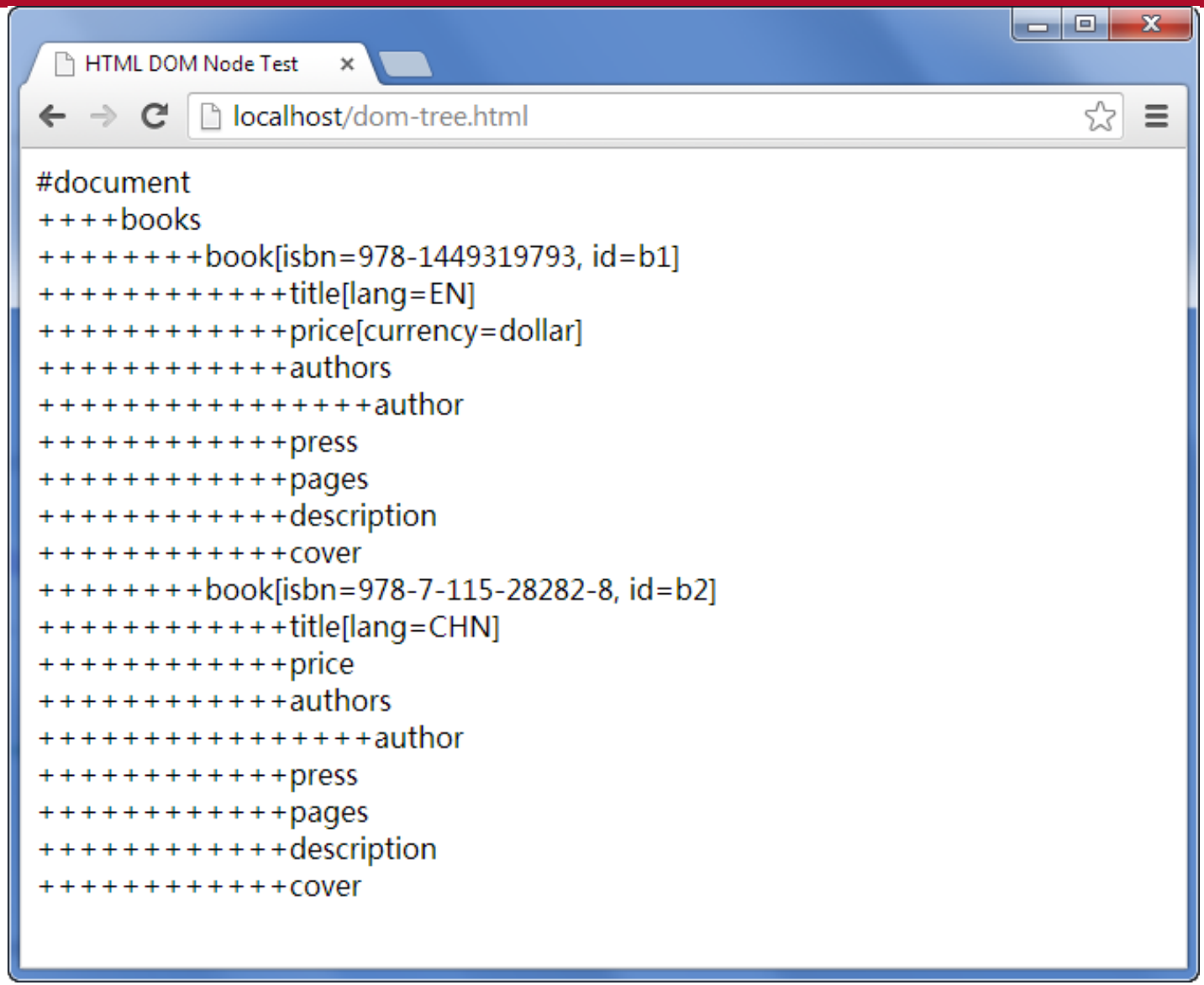
- dom-tree.html

9.5.6 完整示例 V

```
<html>
  <head>
    <title>HTML DOM Node Test</title>
    <meta http-equiv="Content-Type"
     content="text/html;charset=utf-8" />
    <script type="text/javascript" src="jquery.js"></script>
    <!-- 引用dom-tree.js, 调用编写的脚本文件 -->
    <script type="text/javascript" src="dom-tree.js"></script>
  </head>
  <body id="main"></body>
</html>
```

- 在浏览器中的运行结果

9.5.6 完整示例 VI


HTML DOM Node Test
localhost/dom-tree.html
#document
++++books
++++++++book[isbn=978-1449319793, id=b1]
++++++++++++title[lang=EN]
++++++++++++price[currency=dollar]
++++++++++++authors
++++++++++++++++author
++++++++++++press
++++++++++++pages
++++++++++++description
++++++++++++cover
++++++++book[isbn=978-7-115-28282-8, id=b2]
++++++++++++title[lang=CHN]
++++++++++++price
++++++++++++authors
++++++++++++++++author
++++++++++++press
++++++++++++pages
++++++++++++description
++++++++++++cover

END