# HTML

## HTML简介

**是用来描述网页的一种超文本标记语言 (Hyper Text Markup Language)**

不是编程语言

标记语言 就一套标记标签

HTML 使用标记标签来描述网页

## HTML作用

**Web浏览器的作用是读取html文档**

并以网页的形式显示它们 浏览器只会显示html标签包裹的内容

而不会显示html标签本身 简单的说

html就是用来展示信息的

## 书写规范

1. html标签由<>包围的关键字(一般**成对出现**，少数<br>)
2. 绝大多数的标签都具备属性(**用""引起来**)，可以嵌套
3. html标签对**大小写不敏感** 许多网站用的是大写
4. 多个空格 tab 回车 默认为一个空格（**&nbsp;**）
5. HTML框架
6. HTML中颜色的三种取值

颜色由 红色/绿色/蓝色 三原色

有三种取值方式 **RGB**

1.rgb(0,0,0) 值是在0-255之间

2.#000000 两位代表一个颜色

3.red green blue 直接写颜色对应的英文单词

## HTML标签

### 排版标签

### 注释

<!-- 注释 不显示在页面上的内容 -->

### p标签

段落标签 可以将html文档分割为若干段落 浏览器会自动识别 在段落前后添加空格 段落之间用空行

常用属性 **align 用于设置对齐方式** left right center 默认是left

### br（换行标签）

### hr标签

（一条水平线）

|  |  |  |
| --- | --- | --- |
| align | 设置水平线对其方式 | left right center |
| size | 设置水平线的厚度 | 以像素为单位 默认是2 |
| width | 设置水平线的长度 | 可以用绝对值或相对值 默认为100% |
| color | 设置水平线的颜色 | 默认为黑色 |

### 数值单位

（默认像素px、有时可以用百分比%）

### 块标签

* 1. **div标签**

用于在文档中设定一个块区域 默认情况下每个div 单独换行显示

常用属性 align

* 1. **span标签**

用于在行内设定一个块区域

### 字体标签

* 1. **font**

标签用于规定文本的字体 大小 颜色

常用属性

face:规定文本的字体

size:规定文本的大小

color:规定文本的颜色

* 1. **h1-h6**

<h1> <h6> 标签用于定义标题

黑体 标题之间有空行

<h1>是最大

<h6>是最小

### 列表标签

|  |  |  |
| --- | --- | --- |
| ul | 无序列表 | disc（实心圆） square（正方形） circle（空心圆）  默认是disc  type |
| ol | **有序列表** | **value start** |
| li | **列表项** | **type(无序) value(有序)** |
| dl、dt、dd | **自定义列表** |  |

* 1. **Ul（**无序列表**）**

常用属性

type:规定列表的项目符号类型 可以取值 disc square circle 默认是disc

* 1. **Li（列表项）**

标签标示的是一个列表项

常用属性

value 只适用于有序列表 用于设定列表的项目数字

type 只适用于无序列表 用于设定列表的项目符号 默认是disc

* 1. **ol（有序列表）**

常用属性

value 规定列表中使用的项目数字是什么 可取值1

start 这个属性规定列表的起始值

* 1. **自定义列表**

<dl>

<dt>

<dd>

### img（图形标签）

用于在页面上引入图片

|  |  |
| --- | --- |
| src | 用于设定要引入的图片的url 相对路径 |
| alt | 用于设定图像的替代文字 如果你的图片因为某种原因无法显示的时候 就会显示alt的内容 |
| width | 用于设定图片的宽度 |
| height | 用于设定图片的高度 |
| border | 图片边框厚度 |
| align | 用于设定图片的文字对齐方式 |
| title | 当你的鼠标悬停在图片的时候 显示的小提示 |

相对路径的写法

当当前文件与目标文件在同一目录下 直接书写目标文件名+后缀名

当当前文件与目标文件处在同一个文件夹下的时候 写法 文件夹名/目标文件+后缀名

../表示上一层

### a（链接标签）

用于定义超链接 用于从一个页面链接到另一个页面

|  |  |  |  |
| --- | --- | --- | --- |
| href | 用于定义链接指向的页面url |  |  |
| name | 设定锚的名字 |  |  |
| target | 用于设定如何打开链接 | \_blank | 在新页面打开 |
|  |  | \_self | 在当前页面打开 |
|  |  | \_parent | 在父窗口打开 |
|  |  | \_top | 在顶层窗口打开 |

### table（表格标签）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 常用属性 | align | 用于设定表格对齐方式 |  |  |
|  | bgcolor | 用于设定表格的背景颜色 |  |  |
|  | border | 用于设定表格的边框宽度 |  |  |
|  | width | 用于设定表格的宽度 |  |  |
|  | cellspacing | 单元格的间距 |  |  |
|  | cellpadding | 单元格内容与边框的距离 |  |  |

#### tr

行 里面包含多个th或者td元素

常用属性

align:用于设定表格中行的内容对齐方式

bgcolor:表格中行的背景颜色

#### td

定义单元格 td元素中的文本一般显示为正常字体且左对齐

常用属性

align:用于设定单元格内容的对齐方式

bgcolor:单元格背景颜色

height:单元格的高度

width:单元格的宽度

colspan:用于设定列合并

rowspan:用于设定行合并

#### caption

表格标题

这个标签必须紧跟<table>标签之后 一个表格只能有一个标题 文字会被居中于表格之上

#### th

标签用于表格的表头 th内部的文字 通常居中 加粗

html里面表格中两种类型的单元格

<tr><Td>包含数据</td></tr>

<tr><Th>包含表头信息</th></tr>

<thead>

表格的页眉

<tfoot>

表格的页脚

<tbody>

用于定义表格主体

### meta

写在head标签中的

提供关于页面的元信息(meta-information)

比如 针对搜索引擎和更新频率的描述和关键字 不包含任何内容

常用属性

content 定义与http-equiv或name属性相关的元信息

http-equiv 把content属性关联到http头部

name 把content属性关联到一个名称

### link

写在head标签中的

定义文档与外部资源的关系

最常见的用途 是 链接层叠样式表(css)

只能用在头标签中 但是可以出现多次

常用属性

type定义被链接的文档的MIME类型 text/css text/javascript

href定义被链接的文档URL

rel定义当前文档与被链接文档之间的关系 stylesheet

### 框架标签

所谓的框架 就是将页面分割成几个窗框 就可以让我们在同一个浏览器页面中显示不止一个页面

#### frameset

是框架结构标签 定义如何将窗口分割成多个页面组合

注意: frameset 不能和body标签一起使用

常用属性

cols 垂直切割

rows 横向切割

frameborder 定义框架的边框 0(no)/1(yes) 0 表示不要边框 1表示要边框

border 定义边框的厚度

bordercolor 定义边框的颜色

framespacing 定义框架与框架之间的距离

#### frame

框架标签 它定义放置在每个框架中的页面

常用属性

src:定义此框架要显示的页面url

name:定义此框架的名字

frameborder:边框

scrolling 定义滚动条(卷轴) yes表示显示 no表示不显示 auto视情况而定

#### iframe

frame和iframe区别

1.frame不能脱离frameset使用 iframe可以

2.frame不能放在body中 嵌套在frameset中的iframe必须放在body中

3.frame的高度只能通过frameset来控制 iframe可以自己控制

4.如果同一个页面上使用了两个以上的iframe ie中可以正常显示 firefox只能显示第一个

使用两个以上的frame在ie和firefox都可以正常使用

### 表单标签

#### form表单

表单 用于向服务器传输数据

#### 常用属性

name 用于定义表单的名称

action 用于规定表单想哪儿提交

method 用于规定提交的方式 get / post

get:方式只能提交少量数据 不安全 因为数据会显示在地址栏里

post:方式提交时 可以携带大量数据 安全 数据不会显示在地址栏里

#### <input>

标签用于搜集用户信息

根据不同的type属性值 来确定输入的内容是什么形式的

包括 文本框 复选框 单选框 上传 按钮 密码框等等

text 文本框

password 密码框

checkbox 复选框

常用属性:checked 默认被选中

radio 单选框

常用属性:checked 默认被选中

hidden 隐藏域

file 文件上传

button 按钮

submit 提交按钮

reset 重置按钮

image 图片按钮

#### <textarea>

文本区域 多行多列的文本区域

常用属性

name 定义多行文本框的名称

cols 可见宽度

rows 可见行数

wrap 文字如何换行

#### <select>与<option>标签

select 定义一个下拉列表

常用属性

name 定义下拉列表名称

size 定义可选的项有多少

option 用于定义一个下拉列表中若干的选项

option标签需要在select里面内部

常用属性

value 定义送往服务器的选项值

selected 定义选项的状态为被选中

# CSS

## CSS简介

w3school

层叠样式表（Cascading Style Sheets）

样式表用来将HTML页面变得更加美观，网站的整体样式通常都存放在**.CSS**文件中，作为该网站的通用样式来使用

大大的提高了工作效率

多个样式可以层叠为一个(class)

## 基本语法

### 选择器

**选择器主要是为了确定需要改变样式的HTML元素**

一条或者多条声明

每一条声明由属性和值构成 使用花括号来包围 属性之间用冒号隔开

h1{

color:red;

font-size:12px;

}

### 值

px:像素

%：百分比

cm:厘米

in：英寸

em:当前字体尺寸

pt:磅

### 颜色

rgb(0,0,0)

rgb(%,%,%)

#000000

饱和度

## 导入CSS

|  |  |  |
| --- | --- | --- |
| 内联样式表 | 在HTML标签中使用style属性 |  |
| 内部样式表 | 写在style标签中 | 通常写在head中，可以写在任何地方 |
| 外联样式表 | 单独写.CSS文件，通过link引用 |  |

## CSS&JavaScript外部引用

当css与JSP、HTML页面都处于文件夹中时，前面需要加../

<link rel="stylesheet" type="text/css" href="../css/my.css" />

<script type="text/javascript" src="../js/my.js" ></script>

优先级 内联>内部>外联

## ./ ../以及/的区别

./ 当前目录  
../ 父级目录  
/ 根目录

## CSS选择器

|  |  |  |
| --- | --- | --- |
| 类选择器 | .red{color:red} | class属性引用(可以含有多个样式，样式中用空格隔开) |
| ID选择器 | 前面加# | #green{color:red;} ID属性引用 |
| 标签选择器 | 以html标签为选择器 | 让所有的html中该标签的样式都符合这一个标准 |

优先级：ID>类>标签

**类选择器和ID选择器区别**

ID在一个HTML页面里只能使用一次

class可以使多个HTML页面标签共同使用

**HTML表单中**

name用于表单提交服务

ID用于JavaScript的操作 可以确定是哪一个具体的标签

## 常用属性

### 字体

字体类型 大小 加粗 倾斜 变形

|  |  |  |
| --- | --- | --- |
| font | 简写属性 | 作用是把所有的针对字体的属性设置在一个声明中 |
| font-family | 字体表 | 宋体 微软雅黑 |
| font-style | 字体样式 | normal正常 italic斜体 oblique倾斜 |
| font-weight | 浓淡 | normal正常 bold加粗 |
| font-size | 大小 |  |
| line-height | 行高 |  |
| text-align | 对齐方式 |  |
| letter-spacing | 字符间距 |  |
| text-decoration | 文本修饰 |  |

font简写形式

顺序font-style|font-variant|font-weight|font-size|line-height|font-family

注意事项

**简写的时候，font-size和line-heigh 12px/1.5em**

顺序不能更改

### 背景

CSS允许应用纯色和背景图像作为背景

|  |  |  |
| --- | --- | --- |
| background | 简写属性 |  |
| background-color | 背景颜色 |  |
| background-image | 背景图片 |  |
| background-position | 背景图片的起始位置 x/y |  |
| background-repeat | 背景图片是否以及如何重复 |  |

### 定位

|  |  |
| --- | --- |
| position | 元素定位 |
| position：absolute | 绝对定位 |
| position：relative | 相对定位 |

常用属性

position

top/right/left/ bottom间距

### 列表

列表属性允许你设置 改变列表标志 或者将图像作为列表项

常用属性

|  |  |
| --- | --- |
| list -style | 简写属性 |
| list –style- image | 定义列表项标志图像 |
| list -style -position | 定义图片位置 |

## CSS框模型/盒子模型

box model 规定了元素的内容 边框 内边距 外边距 浮动

常用属性

|  |  |
| --- | --- |
| border | 边框 围绕元素内容和内边距的一或多条线 |
| border-width | 宽度 |
| border-style | 样式 |
| border-color | 颜色 |

border-top/right/bottom/left

### margin外边距

常用属性

margin：10px 15px;(上下10，左右15)

margin-top、margin-right、margin-bottom、margin-left

### padding内边距

常用属性

padding：10px 15px;(上下10，左右15)

padding-top、padding-right、padding-bottom、padding-left

float 浮动

控制元素如何显示 设置图像显示位置 用绝对值来定位元素 以及元素的可见度

### a 超链接

text-decoration下划线

CSS伪类 用于向某些选择器添加特殊效果

选定、鼠标悬浮、已访问、未访问

active hover visited link

# JavaScript

## 概念&组成

JavaScript是一种脚本语言

不需要编译

源代码在客户端浏览器上直接运行

由浏览器解析

直译语言

Netscape（网景） 公司研发的

Jscript（微软公司） 发明的脚本语言

ECMAScript 脚本语言开发标准（Netscape公司提供）

**一个完整的JavaScript由三部分组成**

**核心（ECMAScript）** JS核心规范，描述了基本语法和基本对象

**文档对象模型（DOM）**document object model 整合JS、HTML、CSS

**浏览器对象模型（BOM）**broswer object model 整合浏览器和JS

## 基础语法

### JS对浏览器操作部分

window 整个窗口

location 地址栏

history 历史

document 文档区域

### JavaScript书写规范

可以写在网页HTML页面的任何地方 但是位置越靠前 执行越早

JavaScript的标签

**内嵌** <script>JavaScript代码</script>

**外链** <script src=”document.js”></script> （中间不能使用JavaScript代码）

### 语法

* 1. 行尾结束 用； 也可以不写 用回车代替 但是不推荐
  2. 变量
     1. 变量的声明 var **如果不加var 认为是全局变量**
     2. 变量区分大小写
     3. 字符串 可以用单引号 和双引号
     4. 变量类型可以随时改变
  3. 注释

//fs /\* fsf\*/

* 1. 封装代码块 与Java一样{}
  2. JavaScript的数据类型（原始数据类型）
     1. **String** 字符串””
     2. **Number** 任何数字都是Number类型 32位 64位 八进制 十进制
     3. **Boolean** 两个值 true和false
     4. **Undefined** 未初始化的变量
     5. **Null** 尚未存在的对象

### 方法

* 1. isNaN(boolean) is not a number
  2. eval(“1+2\*3-4/5”) 计算字符串表达式

## 常用对象

浏览器对象 window location history document  
脚本对象String RegExp Math Date

HTML对象 各种HTML标签 段落 图片 超链接

### String 字符串

创建字符串有两种方式

使用var 语句

1.var str = "这是我的字符串";

2.创建String 对象 var str = new String("字符串");

### RegExp 正则对象

创建正则有两种写法

1. 用/正则代码/ 包括

var reg1 = new RegExp("^[a-zA-Z][a-zA-Z\_0-9]{5,9}$","g");

var reg2 = /^[a-zA-Z][a-zA-Z\_0-9]{5,9}$/g;

2.var reg = new RegExp("匹配项","规则");

**reg.test(“fsfsfs”) 测试字符串**

^ 只与整个被搜索字符串的开始位置相匹配

$ 只与被索索字符串的结尾位置相匹配

var str=”hello world”

str.split(/o/g);

str.replace(/o/g,”hahha”)

str.search(/l/g)

str.match(/l/g)

### Math

|  |  |
| --- | --- |
| Math方法 | 描述 |
| random() | 0-1随机小数 |
| round() | 四舍五入取整 |
| ceil() | 向上取整 只要有小数 就将整数部分+1 |
| floor() | 向下取整 |
| Math.floor(3.4567\*100)/100 | 保留小数的后两位 |

### Date

|  |  |  |
| --- | --- | --- |
| Date | var d = new Date(); | 1900年1月1日 00:00:00 |
| d.getFullYear(); | 年 |  |
| d.getMonth(); | 月 | 0-11 |
| d.getDate(); | 日 | 月份中的天数 |
| d.getHours(); | 小时 | 0-23 |
| d.getMinutes(); | 分钟 | 0-59 |
| d.getSeconds(); | 秒 | 0-59 |
| d.getDay(); | 星期 | 0-6 |

### 定时器

setTimeOut(“方法名”,毫秒) 每隔毫秒 执行该方法

clearTimeOut() 关闭定时器

### 数组

#### 创建方式

* 1. var arr1 = [2,6,3,4,5];
  2. var arr2 = new Array(2,3,4,5);
  3. var arr3 = new Array(4);

arr3[0] = 1;arr3[1] = "abc";arr3[2] = true;

#### 长度 length

#### 遍历

for(var i=0;i<arr3.length;i++){

document.write(arr3[i]+"<br />");

}

#### join 分割

将数组中的元素 组成字符串 默认以,分割

alert(arr3.join("-"));//StringBuffer

#### reverse 反转

arr3.reverse();

alert(arr3.join("-"));

#### sort 排序

arr1.sort(function(a,b){

return b-a;

});

alert(arr1.join("-"));

## DOM和BOM的区别

1. BOM包含DOM

（因为document对象可以使用window.document来调用）

1. 浏览器提供出给予访问的BOM对象

从BOM对象再访问到DOM对象 从而js可以操作浏览器以及浏览器读到的文档 (先要有窗口 然后才会有窗口内的内容)

document 对象是DOM模型的根节点

window 对象是BOM模型的根节点

## 浏览器对象

### 事件

|  |  |  |
| --- | --- | --- |
| 事件 | 事件名称 | 作用 |
| 提交事件 | onSubmit | 表单提交事件 |
|  | onChange | 文本内容或者下拉列表中的选项发生改变 |
|  | onLoad | 网页文件加载事件 |
| 焦点事件 | onFocus | 获得焦点 表示 文本框等 获得鼠标、光标 |
|  | onBlur | 失去焦点 |
| 鼠标事件 | onMouseOver | 鼠标悬停 |
|  | onMouseOut | 鼠标移出 |
|  | onMouseMove | 鼠标移动 div层等上方移动 |
|  | onMouseDown | 鼠标按下 |
|  | onMouseUp | 鼠标弹起 |

### 常用浏览器对象

#### window

|  |  |  |
| --- | --- | --- |
| Window |  |  |
| 属性 | document | html部分 |
|  | history | 客户访问过的url历史信息 |
|  | location | 当前url的信息 |
|  | name | 窗口或框架的名字 |
|  | status | 底部状态栏 |
|  | screen | 有关客户端的屏幕和显示性能 |
| ----------------------- | -------------------------------------- | ------------------------------------------------------------- |
| 方法 | alert("提示信息") | 显示包含消息的提示框 |
|  | prompt("提示信息",0) | 弹出一个带输入框的消息提示框 |
|  | confirm("提示信息") | 显示一个带有确认 和 取消按钮的窗口 true false |
|  | open("url","name") | 打开具有指定名称的新窗口 并加载给定url的页面 |
|  | close() | 关闭open打开的窗口 |
|  | setTimeout("函数",毫秒数) | 设置定时器 经过指定毫秒数 调用某个函数 |
|  | clearTimeout(定时器对象) | 清除定时 |
|  | window.方法名调用 | window 在调用方法的时候可省略 |

#### document

|  |  |  |
| --- | --- | --- |
| document | 方法 | 方法说明 |
| 属性 | alinkColor | 设置或检索文档中所有活动链接的颜色 |
|  | bgColor | 设置Document对象的背景颜色 |
|  | linkColor | 设置或检索文档链接的颜色 |
|  | location | 关于当前url的信息 |
|  | vlinkColor | 设置或检索用户访问过的链接颜色 |
|  | fgColor | 设置文档中文本的颜色 |
| -------------------- | ------------------------- | ----------------------------------------------------------------------------- |
| history | back() | 相当于后退按钮 |
|  | forward() | 相当于前进按钮 |
|  | go(url or number) | 加载history列表中的一个url 或者跳到要求的页面数 |
| *Tips* | go(1)<---> foward() |  |
|  | go(-1) <---> back() |  |
| -------------------- | ------------------------ | --------------------------------------------------------------------------- |
| location | 属性 | host url 的主机名及端口号 |
| 包含有关当前URL的信息 |  | hostname 主机名部分  href 完整的url字符串 |
| Location对象是Window对象的一个部分 | assign("url") reload() | 加载url指定的新的html文档 不会覆盖  重新加载当前页 刷新 |
| 可以通过window.location属性来访问到 | replace("url") | 通过加载url指定的文档来替换当前文档 覆盖掉原来的历史记录 |

### 表单处理(DOM) 文档对象模型

|  |  |  |
| --- | --- | --- |
| 表单处理（DOM） | 可以控制页面中html标签和css标签 | 所有内容都封装成了对象 |
| 1.下拉列表 | <select> | 下拉菜单 |
| eg:[省市二级联动](Code/web/province.html) | <option value ="volvo">Volvo</option> | 下拉菜单中的值 |
| ------------------------- | ----------------------------------------------------- | -------------------------------------------- |
| 2.文本框 | onFocus() | 获得焦点 |
|  | onBlur() | 失去焦点 火狐有问题 |
|  | focus() | 获得焦点 火狐有问题 |
|  | select() | 选中文本内容 突出显示输入区域 |
|  | readonly | 只读属性 |
| ------------------------- | ----------------------------------------------------- | -------------------------------------------- |
| 3.按钮 | onclick() | 单击 |
|  | onsubmit() | 表单提交事件（true/false） |
| ------------------------- | ----------------------------------------------------- | -------------------------------------------- |
| 4.复选框/单选按钮 | checkbox | 复选框 |
|  | radio | 单选按钮 |
|  | .checked方法 | true false |
| 6.文本区域 | textarea |  |

# XML

**extensible Markup Language 可扩展标记语言**

可扩展:所有的标签都是自定义的

功能:

数据存储(如果数据量不是很大 可以放在xml里)

配置文件 properties web.xml

html与xml的区别:

1.html 语法松散 (不区分大小写<div> <DIV>) <hr/><br/>

xml 语法严谨 严格区分大小写 并且标签都是成对出现

2.html 作用:页面展示

xml 作用:存储数据用的

3.html标签都是预定义的

xml 所有标签都是自定义的

语法

文档声明 必须写在xml文档的第一行

<?xml version="1.0" encoding="UTF-8" ?>

version="1.0" 版本号 不可改

encoding="UTF-8" 指定文件的编码 默认 iso-8859-1

元素:xml文档中的标签

文档必须有且只能有一个根元素

元素需要正确闭合 (有开始标签 有结束标签)

元素需要正确嵌套

元素名称要遵守命名规则:

1.元素名称严格区分大小写

2.不能以数字开头

属性:属性值必须用引号引起来 单引号 / 双引号 都可以

注释:<!--你好-->

处理指令:现在已经基本不使用

<?xml-stylesheet type="text/css" href="1.css">

XML约束

两种 dtd 和 schema

什么是XML约束呢

在XML技术里,可以编写一个文档来约束XML文档的书写规范,这称之为 XML约束

就是XML的书写规范

dtd是先出现的 现在基本用 schema

DTD (Document Type Definition) 全称为 文档类型定义

内部dtd :在XML内部定义

外部dtd :在外部文件中定义dtd

然后引入到XML文件中

<!DOCTYPE 文档根节点 SYSTEM "DTD文件的url">本地dtd/网络dtd

ELEMENT:定义元素

ATTLIST:定义属性

<!ELEMENT students (student\*) >

定义元素 元素名为students

在students 这个标签下 可以有 0个或者多个student标签

+ 一次或多次

? 0次或一次

\* 0次或多次

<!ELEMENT student (name,age,sex)>

在student元素下 有 name age sex 三个元素

<!ELEMENT name (#PCDATA)>

<!ELEMENT age (#PCDATA)>

<!ELEMENT sex (#PCDATA)>

设置元素可以存放什么类型的数据

PCDATA 随便

CDATA 是字符串

<!ATTLIST student number ID #REQUIRED>

属性

定义student元素的属性

属性名叫 number

ID --> 主键 唯一性 不能重复 表示属性的设置值为一个唯一的值

#REQUIRED 必须设置该属性

#IMPLIED 可以设置也可以不设置

#FIXED 该属性的取值为一个固定的值 在XML文件中不能为该属性设置其他值

但是需要为该属性提供一个值

Schema

xmlns="http://www.qianfeng.cn/xml"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://www.qianfeng.cn/xml student.xsd"

XML解析

DOM web开发

SAX web开发

PULL -->手机用

Dom4j web开发 dom for java

Xpath web开发

解析XML可以做

1.如果XML做为配置文件: 读取

2.如果XML作为传输文件: 写 / 读

DOM(Document Object Model) -- 文档对象模型

DOM思想:

将整个XML加载内存中 形成文档对象 所有对XML的操作都是在额内存中文档对象进行操作

DOM 是官方的XML解析标准

所以DOM是所有开发语言都支持的 ---java javascript 都支持DOM

SAX(Simple API for XML) -- XML 简单的 API

SAX思想:

一边解析 一边处理 一边释放内存资源 不允许在内存中保留大规模的XML数据

区别:

1.

DOM适用于解析比较简单的XML文件 如果文件过大 会造成内存溢出

DOM是基于内存的 不管文件有多大 都会将所有内容预先装载到内存中

消耗大量的内存空间

SAX适用于解析比较复杂的XML文件

SAX是基于事件的 当某个事件被触发时,才会获取相应的XML部分数据

不管文件有多大 都只占用了少量的内存空间

2.

DOM可以读取XML文件 也可以像XML文件中插入数据

SAX只能读 而不能在文件中插入数据

3.

DOM可以对指定的元素进行随机访问

SAX不可以 SAX是从文档开始执行遍历的 并且只能遍历一次 也就是说不能随机访问XML文件

只能从头读到尾 当然可以中间截断

SAX 逐行读取 基于事件驱动

优点: 不占内存 速度快

缺点: 只能读取 不能回写 只能解析 不能创建

javax.xml.parsers

类 SAXParserFactory

定义工厂 API，使应用程序能够配置和获取基于 SAX 的解析器以解析 XML 文档。

DOM 将文档加载进内存 形成一棵dom树(document对象)

将文档的各个组成部分封装为一些独立的对象 解析

优点:因为在内存中会形成dom树 所以可以对dom树的节点进行增删改查

缺点:dom树非常占内存 解析速度慢 容易内存溢出

DocumentBuilderFactory 创建解析工厂

DocumentBuilder 创建解析器

Document parse(String uri) 指定需要解析的xml文件

Document 接口表示整个 HTML 或 XML 文档。从概念上讲，它是文档树的根，

区别

Node 对象 是整个DOM的主要数据类型 节点对象 代表文档树中的一个单独节点

节点可以是元素节点 属性节点 文本节点

Element 对象 表示XML文档中元素 元素可以包括属性 其他元素 或者是文本

XML树结构中只有 Node 和 Element

元素是一个小范围的定义 ,必须包含有完整的信息 这样的节点才是一个元素

一个节点不一定是一个元素 而一个元素一定是一个节点

<student></student>

NodeList getElementsByTagName(String name)

基本的DOM对象

Document

Node

NodeList

Element

Attr 属性

createAttribute(String) 创建一个 Attr对象 属性

createElement(String) 创建一个标签 一个Element

createTextNode(String) 创建一个Text对象

getElementByTagName(String) 返回NodeList

getDocumentElement() 返回一个代表这个DOM树的根节点的Element对象

appendChild(Node) 为这个节点 添加一个子节点 并放在所有节点的最后 如果这个子节点已存在 先删掉 在添加

getFirstChild() 返回DOM树中这个节点下的第一个子节点

getNodeName() 获取节点名称

getNodeValue() 节点的值

removeChild(Node) 删除给定的子节点对象

replaceChild(Node new,Node old) 用一个新的Node对象代替给定的子节点对象

item(int) 返回指定位置的Node对象

getLength()返回列表的长度

abstract void transform(Source xmlSource, Result outputTarget)

XML常用的解析器

DOM4J 民间方式 非常好 支持DOM

Xpath 专门用来查询

JAXP sun公司提供的解析 支持 dom和sax

JDOME 是一个开源的项目 基于树形结构 利用纯java技术对xml文档实现 解析 生成 序列化以及多种操作

DOM4J 民间方式 非常好 支持DOM

1.导入jar包

2.创建解析器

3.解析xml获得document对象

Xpath 专门用来查询

定义了一种规则

使用方法

selectSingleNode()

selectNodes()

使用步骤

1.导包 dom4j-1.6.1\lib\jaxen-1.1-beta-6.jar

2.创建解析器

3.解析xml 获得document对象

List list = document.selectNodes( "//foo/bar" );

Node node = document.selectSingleNode( "//foo/bar/author" );