# HTML5 第一天笔记

## 基础内容

### 1. HTML5 简介

HTML5 是 HTML 标准的下一个重要版本，用来替代 HTML 4.01，XHTML 1.0 以及 XHTML 1.1。HTML5 也是一种在万维网上构建和呈现内容的标准。

HTML5 是万维网联盟（W3C）和网页超文本技术工作小组（WHATWG）合作的产物。

HTML5 是近十年来 Web 开发标准最巨大的飞跃。HTML5 并非仅仅用来表示 Web 内容，它将 Web 带入一个成熟的应用平台，在 HTML5 平台上，视频、音频、图象、动画，以及同电脑的交互都被标准化。

#### 1）HTML5 引入了许多新元素和属性帮助我们构建现代化的网站。下面是 HTML5 引入的主要特性：

* 新的语义化元素： 比如 <header>，<footer> 和 <section>。
* 表单 2.0： 改进了 HTML Web 表单，为 <input> 标签引入了一些新的属性。
* 持久的本地存储： 为了不通过第三方插件实现。
* WebSocket： 用于 Web 应用程序的下一代双向通信技术。
* 服务器推送事件： HTML5 引入了从 Web 服务器到 Web 浏览器的事件，也被称作服务器推送事件（SSE）。
* Canvas： 支持用 JavaScript 以编程的方式进行二维绘图。
* 音频和视频： 在网页中嵌入音频或视频而无需借助第三方插件。
* 地理定位： 用户可以选择与我们的网页共享他们的地理位置。
* 拖放： 把同一网页上的条目从一个位置拖放到另一个位置。

#### 2）HTML5 浏览器支持：

最新版 Apple Safari，Mozilla FireFox 和 Opera 支持大部分 HTML5 新特性，IE9 也支持一些 HTML5 的新功能。

预装在 iPhones，iPads 和 Android 手机上的手机浏览器都对 HTML5 有良好的支持。

**值得注意的是：** IE 8及之前版本浏览器基本不支持 HTML5 新特性。

### 2. HTML5 参考网站

<https://developer.mozilla.org/zh-CN/>

## 表单 2.0

### 1. Input 新类型

| **类型** | **描述** | **说明** |
| --- | --- | --- |
| email | 邮箱类型 | 判断当前字符串中是否包含@符号 |
| search | 搜索类型 |  |
| url | 网址类型 | 判断当前字符串中是否包含http:// |
| number | 数字类型 |  |
| tel | 电话号码类型 | 只在移动端浏览器有效 |
| range | 范围类型 | 适用于应该包含某个范围内数值的输入字段 |
| color | 颜色类型 |  |
| date | 日期类型 | 按照 ISO 8601 编码的日期（包括年，月，日） |
| month | 月份类型 | 由 ISO 8601 编码的年和月组成的日期 |
| week | 星期类型 | 由 ISO 8601 编码的年和星期数组成的日期 |

下述示例代码使用了 HTML5 提供的表单 2.0 中的 input 类型：

<form>

Email:<input type="email"><br>

搜索:<input type="search"><br>

URL地址:<input type="url"><br>

电话号码:<input type="tel"><br>

数字类型:<input type="number" min="18" max="50" step="2"><br>

范围类型:<input type="range" min="0" max="10" step="2" value="10"><br>

颜色类型:<input type="color"><br>

日期类型:<input type="date"><br>

<input type="submit">

</form>

#### 1）email类型

email 类型是专门用于输入 email 地址的文本框。

如果输入内容不是 email 地址，则不允许提交，并提示错误信息。效果如下图所示：



根据上图效果，总结 email 类型的规则是：提交表单时，检查 email 类型的文本框中是否包含 @ 符号。

**值得注意的是：** email 类型的文本框并不能检查 email 地址是否真实存在。

<input type="email" />

#### 2）url类型

url 类型是专门用于输入 url 地址的文本框。

如果输入内容不是 url 地址，则不允许提交，并提示错误信息。效果如下图所示：



根据上图效果，总结 url 类型的规则是：提交表单时，检查 url 类型的文本框中是否包含 http:// 符号。

<input type="url" />

#### 3）search类型

search 类型是专门用于输入搜索关键词的文本框。

在移动设备中，当 search 类型的元素获取焦点时，键盘的“前往”会改为“搜索”字样。效果如下图所示：

* iPhone设备：



* Android设备：



**值得注意的是：**在PC端浏览器 search 类型与 text 类型外观相同。

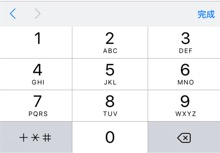
<input type="search" />

#### 4）tel类型

tel 类型是专门用于输入电话号码的文本框。

在移动设备中，当 tel 类型的元素获取焦点时，键盘使用数字面板。效果如下图所示：

* iPhone设备：



* Android设备：



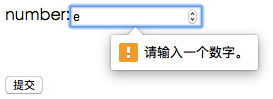
**值得注意的是：**该元素没有特殊的校验规则，不强制输入数字。

<input type="tel" />

#### 5）number类型

number 类型是专门用于输入数字的文本框。

如果输入内容不是数字，则不允许提交，并提示错误信息。效果如下图所示：



根据上图效果，总结 number 类型的规则是：提交表单时，检查 number 类型的文本框中是否为数字。

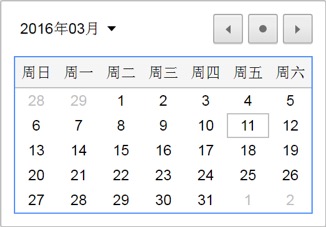
| **属性名称** | **类型** | **描述** |
| --- | --- | --- |
| min | Number | 允许输入的最小数值。 |
| max | Number | 允许输入的最大数值。 |
| step | Number | 设置增加或减少的间隔。（步长） |

<input type="number" min="0" max="100" step="5" />

#### 6）date类型

date 类型是专门用于输入日期，提供日期选择控件。

当用户单击向下箭头按钮时浏览器页面中显示日历。效果如下图所示：



<input type="date" />

#### 7）range类型

range 类型是专门用于输入一定范围内数字值。

range 类型在HTML页面显示为滑动条。效果如下图所示：

http://i2.piimg.com/567571/5db51979471825f8.png

| **属性名称** | **类型** | **描述** |
| --- | --- | --- |
| min | Number | 允许输入的最小数值。 |
| max | Number | 允许输入的最大数值。 |
| step | Number | 设置增加或减少的间隔。（步长） |
| value | Number | 设置滑动条的当前值。 |

<input type="range" min="0" max="100" step="5" value="50" />

#### 8）color类型

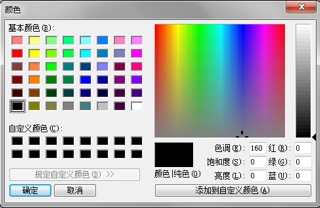
color 类型是专门用于选取颜色。

color 类型提供了一个颜色选取器。效果如下图所示：

* 显示效果：

http://i2.piimg.com/567571/ebbfc8a3555ab547.jpg

* 颜色选取器：



color 类型的显示效果，当用户单击 color 类型弹出显示颜色选取器。

<input type="color" />

#### 练习：自定义调色器

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>自定义调色器</title>

<style>

body {

padding-left: 100px;

padding-top: 100px;

}

.show {

width: 300px;

height: 300px;

border: 1px solid black;

}

</style>

</head>

<body>

<div class="show"></div>

<br><br>

红:<input onchange="mychange()" id="red" min="0" max="255" step="1" value="255" type="range"><br>

绿:<input onchange="mychange()" id="green" min="0" max="255" step="1" value="255" type="range"><br>

蓝:<input onchange="mychange()" id="blue" min="0" max="255" step="1" value="255" type="range">

<script>

function mychange(){

var red = document.getElementById("red").value;

var green = document.getElementById("green").value;

var blue = document.getElementById("blue").value;

var show = document.getElementsByTagName("div")[0];

show.style.backgroundColor = "rgb("+red+","+green+","+blue+")";

}

</script>

</body>

</html>

### 2. 表单新元素

| **元素名称** | **描述** |
| --- | --- |
| <datalist> | <input>标签定义选项列表。请与 input 元素配合使用该元素，来定义 input 可能的值 |
| <progress> | <progress> 标签定义运行中的任务进度（进程） |
| <meter> | <meter> 标签定义度量衡。仅用于已知最大和最小值的度量 |
| <output> | <output> 标签定义不同类型的输出，比如脚本的输出 |

#### 1）datalist元素

<datalist> 元素规定输入域的选项列表。

<datalist> 属性规定 form 或 input 域应该拥有自动完成功能。当用户在自动完成域中开始输入时，浏览器应该在该域中显示填写的选项：

使用 <input> 元素的列表属性与 <datalist> 元素绑定：

<input list="browsers">

<datalist id="browsers">

<option value="Internet Explorer">

<option value="Firefox">

<option value="Chrome">

<option value="Opera">

<option value="Safari">

</datalist>

#### 2）progress元素

<progress> 标签表示进度条，用于显示一个任务的完成进度。

**属性：**

| **属性** | **值** | **描述** |
| --- | --- | --- |
| max | number | 规定需要完成的值 |
| value | number | 规定进程的当前值 |

**提示：**请将 <progress> 标签与 JavaScript 一起使用来显示任务的进度。

**值得注意的是：**<progress> 标签不适合用来表示度量衡（例如，磁盘空间使用情况或相关的查询结果）。表示度量衡，请使用<meter> 标签代替。

<progress id="progress" max="100" value="0"></progress>

<script>

var progress = document.getElementById("progress");

var max = progress.max;

var t = setInterval(function(){

var v = progress.value;

if(v == max){

clearInterval(t);

}else{

v++;

progress.value = v;

}

},100);

</script>

#### 3）meter元素

<meter> 标签定义度量衡。仅用于已知最大和最小值的度量。

比如：磁盘使用情况，查询结果的相关性等。

**属性：**

| **属性** | **值** | **描述** |
| --- | --- | --- |
| high | number | 规定被界定为高的值的范围 |
| low | number | 规定被界定为低的值的范围 |
| max | number | 规定范围的最大值 |
| min | number | 规定范围的最小值 |
| value | number | 必需。规定度量的当前值 |

**值得注意的是：**<meter> 不能作为一个进度条来使用， 进度条 <progress> 标签。

<meter min="0" max="100" value="95" low="10" high="90"></meter>

#### 4）output元素

<output> 元素用于不同类型的输出，比如计算或脚本输出：

<form oninput="x.value=parseInt(a.value)+parseInt(b.value)">0

<input type="range" id="a" value="50">100 +

<input type="number" id="b" value="50">=

<output name="x" for="a b"></output>

</form>

### 3. 表单新属性

#### 1）placeholder属性

placeholder 属性提供一种提示（hint），描述输入域所期待的值。

简短的提示在用户输入值前会显示在输入域上。

**值得注意的是：**placeholder 属性适用于以下类型的 <input> 标签：text, search, url, telephone, email 以及password。

<input type="text" name="fname" placeholder="First name">

#### 2）autofocus属性

autofocus 属性是一个 boolean 属性。

autofocus 属性规定在页面加载时，域自动地获得焦点。

First name:<input type="text" name="fname" autofocus>

#### 3）multiple属性

multiple 属性是一个 boolean 属性。

multiple 属性规定<input> 元素中可选择多个值。

**值得注意的是：**multiple 属性适用于以下类型的 <input> 标签：email 和 file。

Email: <input type="email" multiple>

#### 4）form属性

form 属性规定输入域所属的一个或多个表单。

**值得注意的是：**如需引用一个以上的表单，请使用空格分隔的列表。

位于form表单外的input 字段引用了 HTML form (该 input 表单仍然属于form表单的一部分)：

<form action="demo-form.php" id="form1">

First name: <input type="text" name="fname"><br>

<input type="submit" value="Submit">

</form>

Last name: <input type="text" name="lname" form="form1">

## 表单验证

### 1. 验证属性

| **属性名称** | **描述** |
| --- | --- |
| required | 验证不为空 |
| pattern | 匹配正则表达式 |
| min和max | 验证最小值和最大值 |
| minlength和maxlength | 验证最小长度和最大长度 |
| validity | 获取validityState对象 |

#### 1）required属性

required 属性规定必须在提交之前填写输入域（不能为空）。

**值得注意的是：**required 属性适用于以下类型的 <input> 标签：text, search, url, telephone, email,password, date pickers, number, checkbox, radio 以及 file。

Username: <input type="text" name="usrname" required>

#### 2）pattern属性

pattern 属性的值与指定的正则表达式进行匹配（一般用于验证特定格式）。

**值得注意的是：**pattern 属性适用于以下类型的 <input> 标签: text, search, url, tel, email, 和password。

Country code: <input type="text" name="country\_code" pattern="[A-Za-z]{3}" title="Three letter country code">

#### 3）min和max属性

min、max 和 step 属性用于为包含数字或日期的 input 类型规定限定（约束）。

**值得注意的是：**min、max 和 step 属性适用于以下类型的 <input> 标签：date pickers、number 以及range。

Enter a date before 1980-01-01:

<input type="date" name="bday" max="1979-12-31">

Enter a date after 2000-01-01:

<input type="date" name="bday" min="2000-01-02">

Quantity (between 1 and 5):

<input type="number" name="quantity" min="1" max="5">

#### 4）maxlength属性

maxlength 属性用于设定允许输入的最大字符个数。

**值得注意的是：**maxlength 属性应该叫做限制属性。

### 2. 有效状态

ValidityState 代表了一个元素可以处于的与约束验证相关的有效性状态。

| **名称** | **类型** | **描述** |
| --- | --- | --- |
| valid | Boolean | 其他的约束验证条件都不为 true |
| valueMissing | Boolean | 该元素有 required 属性,但却没有值 |
| typeMismatch | Boolean | 该元素的值不符合元素类型所要求的格式(当 type 是 email 或者 url 时) |
| patternMismatch | Boolean | 该元素的值与指定的 pattern 属性不匹配 |
| stepMismatch | Boolean | 该元素的值不符合由 step 属性指定的规则 |
| rangeUnderflow | Boolean | 该元素的值小于指定的 min 属性 |
| tooLong | Boolean | 该元素的值的长度超过了 HTMLInputElement 或者 HTMLTextAreaElement 对象指定的maxlength 属性中的值 |
| rangeOverflow | Boolean | 该元素的值大于指定的 max 属性 |
| customError | Boolean | 该元素的自定义有效性消息已经通过调用元素的setCustomValidity() 方法设置成为一个非空字符串 |

<form>

用户名:<input id="user" type="text" required><br>

密码:<input id="pwd" type="text" pattern="^[0-9]{6,8}$"><br>

Email:<input id="mail" type="email"><br>

年龄:<input id="age" step="5" type="number" min="20"><br>

<input type="submit">

</form>

<script>

var user = document.getElementById("user");

user.onblur = function(){

if(user.validity.valid){

alert("验证通过.");

}else if(user.validity.valueMissing){

alert("用户名为空.");

}

}

var pwd = document.getElementById("pwd");

pwd.onblur = function(){

if(pwd.validity.valid){

alert("验证通过.");

}else if(pwd.validity.patternMismatch){

alert("密码输入有误.");

}

}

var mail = document.getElementById("mail");

mail.onblur = function(){

if(mail.validity.valid){

alert("验证通过.");

}else if(mail.validity.typeMismatch){

alert("Email输入有误.");

}

}

var age = document.getElementById("age");

age.onblur = function(){

if(age.validity.valid){

alert("验证通过.");

}else if(age.validity.rangeUnderflow){

alert("年龄过小.");

}else if(age.validity.stepMismatch){

alert("年龄有误.");

}

}

</script>

### 3. setCustomValidity() 方法

setCustomValidity() 方法设置自定义验证信息，用于即将实施与验证的约束来覆盖预定义的信息。

<form>

<input id="data" type="text" required>

<input type="submit">

</form>

<script>

var data = document.getElementById("data");

data.onblur = function(){

if(data.validity.valueMissing){

data.setCustomValidity("不能为空.");

}else if(data.validity.customError){

data.setCustomValidity("");

}

}

</script>

**原创声明：**如发布或转载请注明以下作者信息，谢谢！

本文作者 - 金云龙

联系方式 - QQ：2080954535；微信：imlonge；微信公众号：longestory

个人网站：<http://www.longestory.com>