[**HTML5教程（第1篇）-前言**](http://html5cn.net/archives/92.html)

发表时间：2010-05-08评论数：[0](http://html5cn.net/archives/92.html#respond)

-

前言

超文本标记语言（HTML）的开发到 1999 年 HTML 4 就停止了。万维网联盟（W3C）把重点转向将 HTML 的底层语法从标准通用标记语言（SGML）改为可扩展标记语言（XML），以及可缩放向量图型（SVG）、XForms 和 MathML 这些全新的标记语言。浏览器厂商则把精力放到选项卡和富站点摘要（RSS）阅读器这类浏览器特性上。Web 设计人员开始学习使用异步 JavaScript XML（Ajax），在现有的框架下通过层叠样式表（CSS）和 JavaScript™ 语言建立自己的应用程序。但是在接下来的八年中，HTML 本身没有任何变化。

最近，它又复活了。三家重要的浏览器厂商 — Apple、Opera 和 Mozilla Foundation — 成立了 Web Hypertext Application Technology Working Group（WhatWG）来开发传统 HTML 的新版本。最近，W3C 也注意到了这些活动，也启动了自己的新一代 HTML 项目，双方的成员有很多是相同的。这两个项目最终很可能合并。虽然很多细节还在争论之中，但下一版本 HTML 的大体轮廓已经清楚了。

Web 开发人员从 1999 年就一直期待 HTML 的新版本（通常称为 HTML 5，但是也称为 Web Applications 1.0），现在它终于发布了。它保持了 HTML 原来的特色：没有名称空间或模式。元素不必结束。浏览器会宽容地对待错误。p 仍然是 p，table 仍然是 table。

HTML5第一次向 HTML 中引入新的元素。新的结构元素包括 aside、figure 和 section。新的内联元素包括 time、meter 和 progress。新的内嵌元素有 video 和 audio。新的交互元素有 details、datagrid 和 command。

如果在 1999 年将一位 Web 开发人员冷冻起来，现在再解冻，那么他会遇到一些新的让人迷惑的元素。是的，他熟悉的元素（比如 div）仍然保留了；但是，HTML 现在还包含 section、header、footer 和 nav 等新元素。em、code 和 strong 仍然存在，但是增加了 meter、time 和 m。img 和 embed 仍然可用，但是还增加了 video 和 audio。但是，他仔细观察一下就会发现，这些元素实际上没什么区别。其中许多元素可能在 1999 年是开发人员需要而没有得到的。通过与他已经掌握的元素进行简单的类比，这些新元素都很容易理解。实际上，与 Ajax 或 CSS 相比，它们非常容易掌握。

最后，当他打开 300MHz 的笔记本（运行的是 Windows 98，也是在 1999 年冷冻起来的）时，他可能对 Netscape 4 和 Windows® Internet Explorer® 5 中显示的新页面效果很吃惊。当然，这些老式浏览器不认识新元素，会完全忽略它们，但是页面仍然会显示，内容仍然是完整的。

这并不是什么虚构的故事。HTML 5 的设计原则就是在不支持它的浏览器中能够平稳地退化。原因很简单：我们都是这样的 “原始人”。浏览器现在有选项卡、CSS 和 XmlHttpRequest，但是它们的 HTML 显示引擎仍然停留在 1999 年的水平。除了用户量大大增加之外，Web 其实在本质上没什么进步。HTML 5 考虑到了这一点。它目前为 Web 开发人员一些真正的好处，随着浏览器的缓慢升级，页面浏览者会逐渐享受到这些好处。

转载自：http://www.niuc.net/post/317/

版权所有。转载时必须以链接形式注明作者和原始出处及本声明！

前言<!–more–>　　超文本标记语言（HTML）的开发到 1999 年 HTML 4 就停止了。万维网联盟（W3C）把重点转向将 HTML 的底层语法从标准通用标记语言（SGML）改为可扩展标记语言（XML），以及可缩放向量图型（SVG）、XForms 和 MathML 这些全新的标记语言。浏览器厂商则把精力放到选项卡和富站点摘要（RSS）阅读器这类浏览器特性上。Web 设计人员开始学习使用异步 JavaScript XML（Ajax），在现有的框架下通过层叠样式表（CSS）和 JavaScript™ 语言建立自己的应用程序。但是在接下来的八年中，HTML 本身没有任何变化。  
最近，它又复活了。三家重要的浏览器厂商 — Apple、Opera 和 Mozilla Foundation — 成立了 Web Hypertext Application Technology Working Group（WhatWG）来开发传统 HTML 的新版本。最近，W3C 也注意到了这些活动，也启动了自己的新一代 HTML 项目，双方的成员有很多是相同的。这两个项目最终很可能合并。虽然很多细节还在争论之中，但下一版本 HTML 的大体轮廓已经清楚了。  
Web 开发人员从 1999 年就一直期待 HTML 的新版本（通常称为 HTML 5，但是也称为 Web Applications 1.0），现在它终于发布了。它保持了 HTML 原来的特色：没有名称空间或模式。元素不必结束。浏览器会宽容地对待错误。p 仍然是 p，table 仍然是 table。  
HTML5第一次向 HTML 中引入新的元素。新的结构元素包括 aside、figure 和 section。新的内联元素包括 time、meter 和 progress。新的内嵌元素有 video 和 audio。新的交互元素有 details、datagrid 和 command。  
如果在 1999 年将一位 Web 开发人员冷冻起来，现在再解冻，那么他会遇到一些新的让人迷惑的元素。是的，他熟悉的元素（比如 div）仍然保留了；但是，HTML 现在还包含 section、header、footer 和 nav 等新元素。em、code 和 strong 仍然存在，但是增加了 meter、time 和 m。img 和 embed 仍然可用，但是还增加了 video 和 audio。但是，他仔细观察一下就会发现，这些元素实际上没什么区别。其中许多元素可能在 1999 年是开发人员需要而没有得到的。通过与他已经掌握的元素进行简单的类比，这些新元素都很容易理解。实际上，与 Ajax 或 CSS 相比，它们非常容易掌握。  
最后，当他打开 300MHz 的笔记本（运行的是 Windows 98，也是在 1999 年冷冻起来的）时，他可能对 Netscape 4 和 Windows® Internet Explorer® 5 中显示的新页面效果很吃惊。当然，这些老式浏览器不认识新元素，会完全忽略它们，但是页面仍然会显示，内容仍然是完整的。  
这并不是什么虚构的故事。HTML 5 的设计原则就是在不支持它的浏览器中能够平稳地退化。原因很简单：我们都是这样的 “原始人”。浏览器现在有选项卡、CSS 和 XmlHttpRequest，但是它们的 HTML 显示引擎仍然停留在 1999 年的水平。除了用户量大大增加之外，Web 其实在本质上没什么进步。HTML 5 考虑到了这一点。它目前为 Web 开发人员一些真正的好处，随着浏览器的缓慢升级，页面浏览者会逐渐享受到这些好处。

[**HTML5 特性的检测：Web Workers**](http://html5cn.net/archives/85.html)

发表时间：2010-05-08评论数：[0](http://html5cn.net/archives/85.html#respond)

-

Web Workers 提供了一种标准的方式让浏览器能够在后台运行Javascript。有了它，你就能够或多或少的实现多线程了（就好像你的计算机能够同一时间运行多个程序一样）。这些“后台线程”可以在页面响应用户的滚屏、单击或者输入操作的时候做些诸如复杂的数学运算、发送请求或者操作本地数据库的事情。

检测你的浏览器是否支持web workers特性可以使用之前介绍过的检测方法中的第一种。如果你的浏览器支持Web Worker API的话，在全局window对象上会有一个Worker的属性，反之则这个属性值会是undefined。

view plaincopy to clipboardprint?  
function supports\_web\_workers(){  
return !!window.Worker;  
}

function supports\_web\_workers(){  
return !!window.Worker;  
}

同样的，如果你不想自己亲手去写这个检测方法的话，你可以使用Modernizr(1.1或者更近版本)来检测你的浏览器是否支持Web Worker  
view plaincopy to clipboardprint?  
if(Modernizr.webworkers){  
//window.Worker is available  
} else {  
//no native support for web workers  
//maybe try Gears or another third-party solution  
}

if(Modernizr.webworkers){  
//window.Worker is available  
} else {  
//no native support for web workers  
//maybe try Gears or another third-party solution  
}

[**HTML5教程(第2篇) – 网页结构**](http://html5cn.net/archives/93.html)

发表时间：2010-05-08评论数：[0](http://html5cn.net/archives/93.html#respond)

-

由于缺少结构，即使是形式良好的 HTML 页面也比较难以处理。必须分析标题的级别，才能看出各个部分的划分方式。边栏、页脚、页眉、导航条、主内容区和各篇文章都由通用的 div 元素来表示。HTML 5 添加了一些新元素，专门用来标识这些常见的结构：

· section：这可以是书中的一章或一节，实际上可以是在 HTML 4 中有自己的标题的任何东西

· header：页面上显示的页眉；与 head 元素不一样

· footer：页脚；可以显示电子邮件中的签名

· nav：指向其他页面的一组链接

· article：blog、杂志、文章汇编等中的一篇文章

我们来考虑一个典型的 blog 主页，它的顶部有页眉，底部有页脚，还有几篇文章、一个导航区和一个边栏，见代码1 典型的 blog 页面

<html>

<head>

<title>Mokka mit Schlag </title>

</head>

<body>

<div id=”page”>

<div id=”header”>

<h1><a href=”http://www.elharo.com/blog”>Mokka mit Schlag</a></h1>

</div>

<div id=”container”>

<div id=”center”>

<div id=”post-1000572″>

<h2><a href=

“/blog/birding/2007/04/23/spring-comes-and-goes-in-sussex-county/”>

Spring Comes (and Goes) in Sussex County</a></h2>

<div>

<p>Yesterday I joined the Brooklyn Bird Club for our

annual trip to Western New Jersey, specifically Hyper

Humus, a relatively recently discovered hot spot. It

started out as a nice winter morning when we arrived

at the site at 7:30 A.M., progressed to Spring around

10:00 A.M., and reached early summer by 10:15. </p>

</div>

</div>

<div id=”post-1000571″>

<h2><a href=

“/blog/birding/2007/04/23/but-does-it-count-for-your-life-list/”>

But does it count for your life list?</a></h2>

<div>

<p>Seems you can now go .</p>

</div>

</div>

</div>

<div>

<div>

<a href=”/blog/page/2/”>« \_fcksavedurl=”"/blog/page/2/”>«” Previous Entries</a>

</div>

<div></div>

</div>

</div>

<div id=”right”>

<ul id=”sidebar”>

<li><h2>Info</h2>

<ul>

<li><a href=”/blog/comment-policy/”>Comment Policy</a></li>

<li><a href=”/blog/todo-list/”>Todo List</a></li>

</ul></li>

<li><h2>Archives</h2>

<ul>

<li><a href=’/blog/2007/04/’>April 2007</a></li>

<li><a href=’/blog/2007/03/’>March 2007</a></li>

<li><a href=’/blog/2007/02/’>February 2007</a></li>

<li><a href=’/blog/2007/01/’>January 2007</a></li>

</ul>

</li>

</ul>

</div>

<div id=”footer”>

<p>Copyright 2007 Elliotte Rusty Harold</p>

</div>

</div>

</body>

</html>

即使有正确的缩进，这些嵌套的 div 仍然让人觉得非常混乱。在 HTML 5 中，可以将这些元素替换为语义性的元素，见代码2 用 HTML5编写的典型blog页面

<html>

<head>

<title>Mokka mit Schlag </title>

</head>

<body>

<header>

<h1><a href=”http://www.elharo.com/blog”>Mokka mit Schlag</a></h1>

</header>

<section>

<article>

<h2><a href=

“/blog/birding/2007/04/23/spring-comes-and-goes-in-sussex-county/”>

Spring Comes (and Goes) in Sussex County</a></h2>

<p>Yesterday I joined the Brooklyn Bird Club for our

annual trip to Western New Jersey, specifically Hyper

Humus, a relatively recently discovered hot spot. It

started out as a nice winter morning when we arrived at

the site at 7:30 A.M., progressed to Spring around 10:00

A.M., and reached early summer by 10:15. </p>

</article>

<article>

<h2><a href=

“/blog/birding/2007/04/23/but-does-it-count-for-your-life-list/”>

But does it count for your life list?</a></h2>

<p>Seems you can now go .</p>

</article>

<nav>

<a href=”/blog/page/2/”>« \_fcksavedurl=”"/blog/page/2/”>«” Previous Entries</a>

</nav>

</section>

<nav>

<ul>

<li><h2>Info</h2>

<ul>

<li><a href=”/blog/comment-policy/”>Comment Policy</a></li>

<li><a href=”/blog/todo-list/”>Todo List</a></li>

</ul></li>

<li><h2>Archives</h2>

<ul>

<li><a href=’/blog/2007/04/’>April 2007</a></li>

<li><a href=’/blog/2007/03/’>March 2007</a></li>

</ul>

</li>

</ul>

</nav>

<footer>

<p>Copyright 2007 Elliotte Rusty Harold</p>

</footer>

</body>

</html>

[**HTML5 特性的检测：离线Web应用**](http://html5cn.net/archives/83.html)

发表时间：2010-05-07评论数：[0](http://html5cn.net/archives/83.html#respond)

-

查看一个离线的静态web页面是很容易的: 连到internet,载入这个web页面,随后断开网络,躲到一个角落里在你空的时候就可以看这个页面了(你甚至可以为了减少时间,省去”躲到一个角落”这一步).  
这对于一个静态的web页面是比较容易实现,可是对于像Gmail 和 Google Docs这样的web应用又是如何做到的呢? 这里就要感谢HTML5了,因为有了它任何人(不仅仅是Google)都可以构建一个离线的web应用了.

离线的web应用从在线的web应用开始.在第一次访问有离线访问功能的web站点的时候, web服务器会告诉你的浏览器哪些文件是为了保证能够让你离线访问该站点所必须的,这些文件可以是任何文件,可以是HTML,也可以是Javascript或者图片,甚至可以是视频.一旦你的浏览器下载了这些必须的文件之后,下次哪怕你没有连接到网络你也可以正常访问该站点.你的浏览器能够识别到你当前是离线状态,然后就会使用之前下载下来的那些文件,而一旦你再次连接到网络,任何你对这些文件做的改动都会同步到远程服务器上.

检测你的浏览器是否支持离线功能可以使用之前介绍过的检测方法中的第一种.如果你的浏览器支持离线web应用的话,在全局的window对象上会有一个叫”applicationCache”的属性,反之该属性值就为undefined.

function supports\_offline(){

    return !!window.applicationCache;

}

function supports\_offline(){

    return !!window.applicationCache;

}

同样的，如果你不想自己亲手去写这个检测方法的话，你可以使用Modernizr(1.1或者更近版本)来检测你的浏览器是否支持离线web应用

if(Modernizr.applicationcache){

    //window.applicationCache is available

} else {

    //no native support for offline

    //maybe try Gears or another third-party solution

}

if(Modernizr.applicationcache){

    //window.applicationCache is available

} else {

    //no native support for offline

    //maybe try Gears or another third-party solution

}

[**各主流浏览器对HTML5的支持情况对比**](http://html5cn.net/archives/64.html)

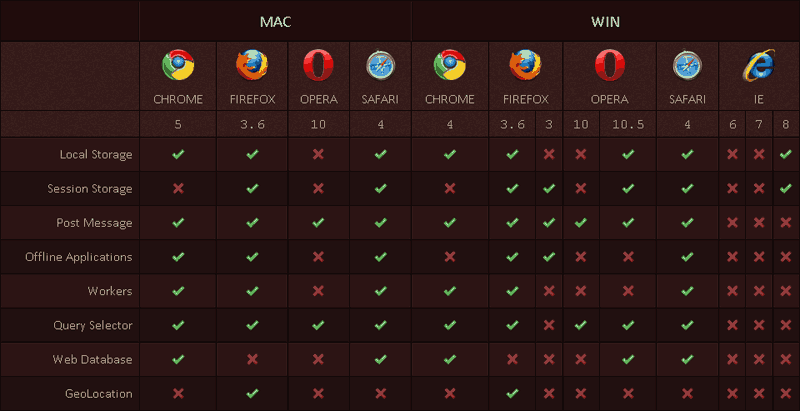
发表时间：2010-05-02评论数：[1](http://html5cn.net/archives/64.html#respond)

-

1.HTML5的web应用

safari 对 HTML5的Web 应用的支持最好，只有地理定位还不支持。

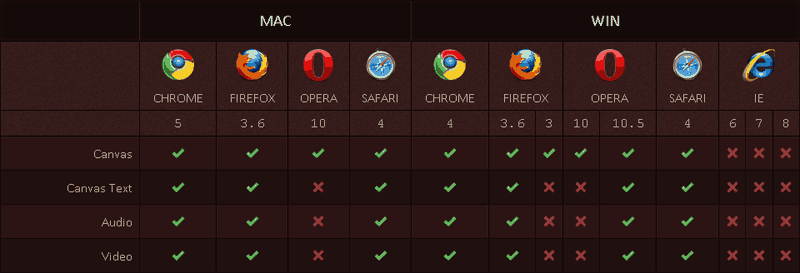
其次是chrome和firefox 3.6，除地理应用外各不支持session和web DB。

如图：[](http://html5cn.net/wp-content/uploads/2010/05/liulanqi_1.gif)

2.HTML5网页内嵌对象

HTML5 最令人期待的地方，包括内置画布canvas，视频，音频等对象。

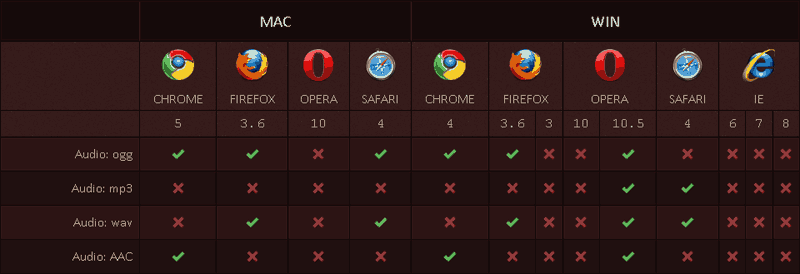
这方面结果还算乐观，Chrome，Safari，Firefox 3.6，Opera 10.5都能全部支持。IE家族则全军覆没。

[](http://html5cn.net/wp-content/uploads/2010/05/liulanqi_2.gif)

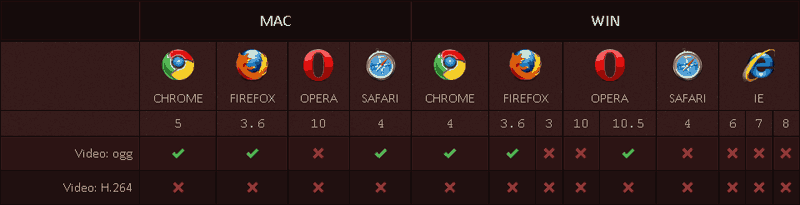
3.html5音频视频编码

Opera 10.5 支持的最全，IE 家族又是全军覆没。其他浏览器也不容乐观。

音频：

[](http://html5cn.net/wp-content/uploads/2010/05/liulanqi_3.gif)

视频：

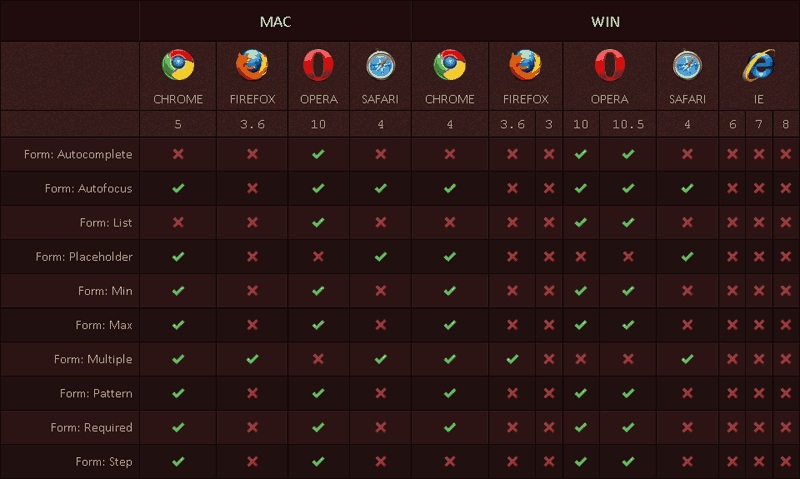
[](http://html5cn.net/wp-content/uploads/2010/05/liulanqi_4.gif)

4.HTML5的表单对象及属性行为：

这些表单对象让人想起了桌面程序。

[](http://html5cn.net/wp-content/uploads/2010/05/liulanqi_5.gif)

HTML5表单的属性行为：

[](http://html5cn.net/wp-content/uploads/2010/05/liulanqi_6.gif)

总结：

无论是MAC平台还是windows平台，chrome都以较大的优势胜出，这可能是因为google向来都是新技术的热烈追逐者把（[google甚至为了HTML5放弃了gears的研发](http://html5cn.net/archives/13.html)）。其次是firefox 3.6和safari，而firefox的用户算是比较多的。而IE对HTML5的支持，还是让我们等待IE9吧。

[**HTML5 特性检测：本地存储（Local Storage）**](http://html5cn.net/archives/112.html)

发表时间：2010-05-10评论数：[0](http://html5cn.net/archives/112.html#respond)

-

**Local Storage(本地存储)**

HTML5 storage提供了一种方式让网站能够把信息存储到你本地的计算机上，并再以后需要的时候进行获取。这个概念和cookie相似，区别是它是为了更大容量存储设计的。Cookie的大小是受限的，并且每次你请求一个新的页面的时候cookie都会被发送过去。HTML5的storage是存储在你的计算机上，网站在页面加载完毕后可以通过Javascript来获取这些数据。

本地存储真的是HTML5的一部分吗？那为什么它出现在另外单独的标准中？

简单的说本地存储是HTML5的一部分。更为详细准确的说是本地存储过去是HTML5的标准中的一部分，而后来由于有些工作组的人表示HTML5太庞大了，于是就剥离出来作为一个单独的标准。这听起来好像是把一块馅饼分成很多块目的是为了减少总的卡路里……好吧，世界的标准就是这么的古怪！

想要检测你的浏览器是否支持HTML5的本地存储特性可以使用之前介绍过的检测方法中的第一种方法。如果你的浏览器支持该特性的话，那么全局对象：window上会有一个localStorage的属性，反之，你的浏览器不支持的话，那么该属性值为undefined

function supports\_local\_storage(){

return !!window.localStorage;

}

同样的，如果你不想自己亲手去写这个检测方法的话，你可以使用Modernizr来检测你的浏览器是否支持本地存储。

if(Modernizr.localstorage){

//window.localStorage is available!

}else{

//no native support for local storage

//maybe try Gears or another third-party solution

}

HTML5的本地存储安全吗？是不是所有人都可以读取到存储的数据？

任何可以物理上能访问你的计算机的人都有可能看看到你的HTML5的本地数据库，在你的浏览器中，任何网站都可以读取和修改他们自己存储的数据，但是不能不同站点的存储数据不能相互访问。这叫域限制

[**HTML5 特性检测：Video Format(视频格式)**](http://html5cn.net/archives/110.html)

发表时间：2010-05-10评论数：[0](http://html5cn.net/archives/110.html#respond)

-

**Video Formats(视频格式)**

视频格式就好比书面语言。一份用英文描述和一份用西班牙文描述的具有相同内容的报纸，其实它们本质都是在描述同样的信息，但是对于只会英文的你而言，只有英文的那份对你才是真正有帮助的。同样的，对于一个视频而言，你的浏览器需要“读得懂”描述你视频的语言。

描述视频的语言被称为：”codec”——一种用来将视频编码成为字节流的算法。目前世界上有许许多多的编码算法，那么你应该使用哪一个？一个不幸的事实是：不是所有的浏览器都遵循一种视频编码算法，它们主要遵循两种编码算法，一种是Safari和iphone遵循的需要收费的，另外一种是Chromium和Mozilla Firefox支持的开源免费的。

要检测你的浏览器支持那种视频格式，可以使用之前介绍的检测方法中的第三种。如果你的浏览器支持HTML5的Video特性，那么你用来检测的时候创建的<video>元素会有一个叫canPlayType()的方法，这个方法可以告诉你你的浏览器支持哪种视频格式。

如果你不想自己亲手去写这个检测方法的话，你可以使用Modernizr来检测你的浏览器支持哪种HTML5的视频格式。

if(Modernizr.video){

//let's play some video! but what kind?

if(Modernizr.video.ogg){

//try Ogg Theora+Vorbis in an Ogg container

}else if(Modernizr.video.h264){

//try H.264 video + AAC audio in an MP4 container

}

}

用来检测Macs和iphones支持的视频格式的方法是:

function supports\_h264\_baseline\_video(){

if(!supports\_video()){ return false; }

var v = document.createElement("video");

return v.canPlayType('video/mp4; codecs="avc1.42E01E, mp4a.40.2"');

}

这个方法从使用之前介绍过的”supports\_video()”方法来浏览器是否支持Video特性开始。如果你的浏览器不支持HTML5 的video特性，那么它自然也肯定不支持什么所谓的视频格式了。

if(!supports\_video()){ return false; }

“视频格式”：简简单单一句话，其实包含了很多的东西。从技术的形式来讲，你可以问：“浏览器是否可以播放基于“H.264”的视频和在MPEG-4容器上的AAC LC音频（我会在后面详细介绍视频的章节告诉你这是什么意思）。如果你想了解更多，你可以阅读视频[编码概述](http://diveintomark.org/tag/give)这篇文章

return v.canPlayType('video/mp4; codecs="avc1.42E01E, mp4a.40.2"');

canPlayType()方法不会返回boolean值（True或者False）。因为视频格式非常的复杂，所以这个方法的返回值有以下几种值：

1. “probably“如果你的浏览器确认可以支持你传入的视频格式

2. “maybe”如果你的浏览器或许可以支持你传入的视频格式

3. “”（空的字符串）如果你的浏览器确认不能支持你传入的视频格式

第二种检测Mozilla Firefox和其他一些开源浏览器支持的开源视频编码格式的方式其实和第一种类似，唯一的不同点是你传入的参数不同，从技术的角度来讲，你可以问你的浏览器是否支持在Ogg容器内的“Theora“视频格式和”Vorbis”音频格式。

function supports\_ogg\_theora\_video(){

if(!supports\_video()){ return false; }

var v = document.createElement('video');

return v.canPlayType('video/ogg; codecs="theora,vorbis"');

}

[**HTML5 特性检测：Video（视频）**](http://html5cn.net/archives/109.html)

发表时间：2010-05-10评论数：[0](http://html5cn.net/archives/109.html#respond)

-

你或许会问：“如果早期的浏览器不支持HTML5我该如何开始使用它呢？”。但是这个问题本身就会让人产生误解。HTML5只是一些特性的集合，因此你不能说检测浏览器“是否支持HTML5”，但是你可以检测浏览器是否支持诸如“canvas“，“video”，“geolocation”等等这些特性。

**Video**

HTML5定义了一个新的元素：<video>，用来将视频内嵌到你的页面中。在页面中内嵌视频在过去如果没有诸如Apple的Quick Time或者是Adobe的Flash播放器的话是根本不可能的。

<video>元素的可用性是不需要任何脚本来检测的。你可以指定多个视频文件，随后支持HTML5的video特性的浏览器会选择一个它支持的视频格式来进行播放。（要想了解更多关于视频格式的内容，可以去查看“视频编码概述”的第一部分：Container formats和第二部分：lossy video codecs）。

不支持HTML5 video的浏览器会完全忽略<video>元素，但是你可以告诉浏览器用第三方插件来播放。Kroc Camen设计了一个能让HTML5的video元素在哪里都可以被支持的叫Video for Everybody!的解决方案，这个解决方案对于在早期的浏览器中会自动降级到用第三方的插件来播放视频文件。这个解决方案完全不用javascript，并且在所有的浏览器包括移动终端的浏览器上都工作的很好。

如果你要做的不仅仅是把视频“丢”到页面上，然后让它播放的话，你需要用javascript。这里你可以用之前介绍的检测方法的第二种方法来检测浏览器是否支持HTML5 的video特性。如果你的浏览器支持的话，那么在检测时候用来创建的<video>元素会有一个叫canPlayType()的方法，反之，该元素就不会有这个方法。

function supports\_video(){

return !!document.createElement('video').canPlayType;

}

同样的，如果你不想自己亲手去写这个检测方法的话，你可以使用Modernizr来检测你的浏览器是否支持HTML5的video特性。

if(Modernizr.video){

//let's play some video!

}else{

//no native video support available

//maybe check for QuickTime or Flash instead

}

P.S.

**检测技术**

当你的浏览器渲染一个web页面的时候，它会构造一个对象的集合DOM(Document Object Model)来表示页面上的HTML元素。每一个元素：<p>,<div>,<span>等都会被表示成DOM中不同的对象。

所有的DOM 对象共享一个公共属性的集合，但是有些对象会拥有更多的属性。在支持HTML5特性的浏览器中，部分特定的对象会有独一无二的属性。通过对DOM的快速了解能够告诉你你哪些HTML5的特性是被支持的。

检测了浏览器是否在支持一个特定的HTML5特性有四种基本的技术，由简入繁依次是：

1、检测全局对象（诸如window或者nagivator）是否支持题特定的属性

2、创建一个元素，然后检测该元素是否存在特定的一些属性

3、创建一个元素，然后检测这个元素上是否存在特定的方法，同时调用这个方法了检查它的返回值

4、创建一个元素，给这个元素设定HTML5特有的属性，然后检查浏览器是否保留了该属性值

[**HTML5 特性检测：Canvas Text**](http://html5cn.net/archives/108.html)

发表时间：2010-05-10评论数：[0](http://html5cn.net/archives/108.html#respond)

-

你或许会问：“如果早期的浏览器不支持HTML5我该如何开始使用它呢？”。但是这个问题本身就会让人产生误解。HTML5只是一些特性的集合，因此你不能说检测浏览器“是否支持HTML5”，但是你可以检测浏览器是否支持诸如“canvas“，“video”，“geolocation”等等这些特性。

**Canvas Text**

即使你的浏览器支持The Canvas API，也有可能不支持The Canvas Text API,原因是The Canvas API一直在不断健壮不断的加入新的内容，the text 方法也是后来才被加入进去的，有些浏览器在完成实现The Canvas API的时候，The Text API还没有被加入进来，在这种情况下，就会发生你的浏览器支持The Canvas API而不支持The Canvas Text API。

要检测浏览器是否支持The Canvas Text API的话可以用之前提到的检测方法中的第二种方法。如果你的浏览器支持The Canvas API,那么用来检测的时候创建的<canvas>有元素会有一个getContext()方法，而相反，如果你的浏览器不支持The Canvas API的话，那么该<canvas>元素不会拥有canvas特有的属性。

function supports\_canvas\_text(){

if(supports\_canvas()){ return false; }

var dummy\_canvas = document.createElement(‘canvas’);

var canvas = dummy\_canvas.getContext(‘2d’);

return typeof context.fillText == ‘function’;

}

该方法从从使用上一节中介绍的supports\_canvas（）方法来检测浏览器是否支持Canvas开始。如果你的了浏览器不支持The Canvas API，那么它肯定不支持The Canvas Text API.

if( !supports\_canvas() ){ return false; }

接下来，你创建了一个对与页面无关的<canvas>元素进而获取它的drawing context. 这个获取操作一定会成功，因为在supports\_canvas()方法中就已经检测过canvas对象上存在getContext()方法。

var dummy\_canvas  = document.createElement(‘canvas’);

var context = dummy\_canvas.getContext(‘2d’);

最后，你检测之前获取的drawing context是否具有fillText()方法，如果有，那恭喜你：当前浏览器支持The Canvas Text API。

return typeof context.fillText == ‘function’

同样的，如果你不想自己亲手去写这个检测方法的话，你可以使用Modernizr来检测你的浏览器是否支持The Canvas Text API

if(Modernizr.canvastext){

//let’s draw some text

}else{

//no native canvas text support available

}

P.S.

**检测技术**

当你的浏览器渲染一个web页面的时候，它会构造一个对象的集合DOM(Document Object Model)来表示页面上的HTML元素。每一个元素：<p>,<div>,<span>等都会被表示成DOM中不同的对象。

所有的DOM 对象共享一个公共属性的集合，但是有些对象会拥有更多的属性。在支持HTML5特性的浏览器中，部分特定的对象会有独一无二的属性。通过对DOM的快速了解能够告诉你你哪些HTML5的特性是被支持的。

检测了浏览器是否在支持一个特定的HTML5特性有四种基本的技术，由简入繁依次是：

1、检测全局对象（诸如window或者nagivator）是否支持题特定的属性

2、创建一个元素，然后检测该元素是否存在特定的一些属性

3、创建一个元素，然后检测这个元素上是否存在特定的方法，同时调用这个方法了检查它的返回值

4、创建一个元素，给这个元素设定HTML5特有的属性，然后检查浏览器是否保留了该属性值

[**HTML5 特性检测：Canvas（画布）**](http://html5cn.net/archives/107.html)

发表时间：2010-05-10评论数：[1](http://html5cn.net/archives/107.html#respond)

-

你或许会问：“如果早期的浏览器不支持HTML5我该如何开始使用它呢？”。但是这个问题本身就会让人产生误解。HTML5只是一些特性的集合，因此你不能说检测浏览器“是否支持HTML5”，但是你可以检测浏览器是否支持诸如“canvas“，“video”，“geolocation”等等这些特性。

**Canvas**

HTML5定义了<canvs>元素作为位图画布在浏览器中的相关解决方案，它用于渲染图形，游戏图像或者其他飞行中的视觉图形。一个Canvas代表了页面中的一个矩形区域，在这个区域里面，你可以用Javascript来画你想画的任何东西。HTML5定义了一系列的方法(“The Canvas API”)用来画图形，定位路径，创建渐变以及将图形变形。

检测你的浏览器是否支持The Canvas API可以使用之前介绍过的第二种方法，如果你的浏览器支持The Canvas API，那么检测时候被创建的<canvas>DOM对象会有getContext()这个方法，如果你的浏览器不支持的话，那被创建的<canvas>DOM对象就不会拥有Canvas特定的属性。

function supports\_canvas(){

return !!document.createElement(‘canvas’).getContext;

}

这个方法执行后会创建一个和页面无关的<canvas>元素。这个元素有永远不会被看到。它只是内存中的一个片段，并且游离在内存中，就好像是水流缓慢的小河上一叶扁舟。

return !!document.createElement(‘canvas’).getContext;

在你创建完<canvas>元素后，紧接着你开始测试它是否存在getContext()方法。这个方法只会在你浏览器支持The Canvas API才会存在。

return !!document.createElement(‘canvas’).getContext;

最后，你用两个否定的一元操作符来强制这个检测方法返回一个Boolean值。  
return !!document.createElement(‘canvas’).getContext;

这个方法可以检测你的浏览器是否支持绝大部分的the canvas API,包括 shapes,paths,gradients 和 patterns。它不会检测实现让微软的IE浏览器支持HTML5特性的第三方explorercanvas库。

如果你不想自己亲手去写检测方法的话，你可以使用Modernizr来检测你的浏览器是否支持The Canvas API

if(Modernizr.canvas){

//let’s draw some shapes!

}else{

//no native canvas support available

}

P.S.:

**检测技术**

当你的浏览器渲染一个web页面的时候，它会构造一个对象的集合DOM(Document Object Model)来表示页面上的HTML元素。每一个元素：<p>,<div>,<span>等都会被表示成DOM中不同的对象。

所有的DOM 对象共享一个公共属性的集合，但是有些对象会拥有更多的属性。在支持HTML5特性的浏览器中，部分特定的对象会有独一无二的属性。通过对DOM的快速了解能够告诉你你哪些HTML5的特性是被支持的。

检测了浏览器是否在支持一个特定的HTML5特性有四种基本的技术，由简入繁依次是：

1、检测全局对象（诸如window或者nagivator）是否支持题特定的属性

2、创建一个元素，然后检测该元素是否存在特定的一些属性

3、创建一个元素，然后检测这个元素上是否存在特定的方法，同时调用这个方法了检查它的返回值

4、创建一个元素，给这个元素设定HTML5特有的属性，然后检查浏览器是否保留了该属性值

# [现在开始，教你用HTML5做网站](http://html5cn.net/archives/136.html)

发表时间：2010-05-23评论数：[1](http://html5cn.net/archives/136.html#respond)

-

如果你现在询问web开发的资深人士对于**HTML5**的看法，你得到的答案可能是两个极端。”HTML 5是让网络更语义化和开放的重要步伐”，或者“HTML5是在用一系列不完整的标记让web陷入困境”。

为什么会有这样的结果。其实归根结底是因为，很少有站点在自然情况下使用HTML 5，所以目前所认识到的理论方案仍然在很大程度上来自于理想，并且未经考验。

现在，我们就开始自己用HTML5动手制作一个网站，让你亲身试验一下，HTML5到底是“福音”还是“灾难”。

## HTML 5的不同在哪里？

第一，我们想通过HTML 5表达什么？理论上，我们表达所有的事——新的语义结构标签，例如canvas或者离线储存等API规范，以及新的内联语义标签。但是，我们把实际的原因仅仅局限于结构标签。canvas，离线储存，本地视频或者地理定位API都很绝妙，然而他们还不能被所有浏览器一致的支持。

人们常说“大多数浏览器也都不支持新的结构元素！”这是真的，但他们中的绝大多数将会很乐意去接受你想要创建的任何标签，并且即将默认支持。甚至IE6也可以处理新标签，尽管如果你想要使用CSS设置样式并且需要一点JavaScript的帮助。

当你对HTML5新标签设置CSS时，你需要注意，那就是未知标签在大部分浏览器中没有默认样式，或者默认为是简单的行级元素。虽然如此，由于大部分HTML 5的新标签可以构造，我们将让他们拥有块级元素的行为。解决方法是在CSS样式中包含display:block;。

为了帮助你了解HTML 5的一些新东西，我们马上开始使用一些新的结构元素。

## 一个任何人都可以记住的文档类型

我们创建HTML 5文档一件很重要的事情就是使用新的文档类型（DOCTYPE）。目前，如果你还清楚的记得HTML 4或者XHTML 1.x的文档类型，你真是一个聪明人，因为他们是如此复杂。每当我们新建一个页面，我们必须打开一个旧的文件，剪切并粘贴文档类型定义。

这真是痛苦，也是为什么我们喜欢新的HTML 5文档类型。因为在HTML5中，你只需要：

<!DOCTYPE html>

不难记,简单并且容易理解,不区分大小写。

这个构想是取消HTML版本化，使其向后兼容变得更容易。从长远看是否成功不得而知，但这至少节省了你输入的时间。

## 最基本语义结构

我们已经将我们的页面定义为HTML 5文档。到目前为止，一切都不错。那么，这些我们一直听说的新标签到底是什么？

在我们研究新标签前，想想你一般网页的结构，这点不需要我再举例了：

这对于展示用途很好，但如果我们想要知道一些关于页面元素包含什么的问题，这又怎么办呢？

上面的例子中，我们为我们所有的结构div添加了ID。这在有见识的设计师中是很平常的事。目 的有两个方面，首先，ID提供了可以能用于给页面的特殊段落应用样式的锚，其次，ID充当基本的伪语义结构。高明的解析器将会查看标签的ID属性，并尝试 去猜测他们的含义，但当每个站点的ID名称不同的时候很难。

这就是新结构标签到来的理由。

当认识到这些ID成为了惯例，HTML 5的缔造者们更进一步，使这些元素中的一部分变成他们独立的标签。这儿有一个HTML 5中生效的新标签的快速概要：

## <header>

头部标签被设计作为关于一个章节或者一整张网页介绍信息的容器。<header> 标签可以包含从你位于大多数页面顶部的典型标志或者标语，到介绍一个章节的标语和开场白的任何东西。如果你还在你的页面里使用<div id=”header”>，那可以使用<header>替换

## <nav>

nav元素非常明显，这是你的导航元素。当然什么被算为导航是有一些争议的，有一个基本的站点 导航，但一些情况下还可能有页面导航元素。HTML5的缔造者WHATWG最近在修改<nav>的解释，来表现怎样在同一个页面使用两次。

如果你还在使用<div id=”nav”>标签来包含你的页面导航，你可以使用简洁的<nav>标签来替换。

## <section>

Section可能是新标签中最模糊的。根据HTML 5定义，一个章节是一个内容的主题集合，通常在header标签后，在footer标签前。但是如果 需要，section也可以相互嵌套。

在我们上面的例子里，被“content”标记的div就是一个变为section的很好的选 择。另外在那个section内，根据内容，我们可以增加section。

## <article>

根绝WHATWG的注释，article元素可以包含“组成文档或站点独立部分的一段内容；例 如，杂志或者新闻的文章，或者博客条目。”

记住一个页面里可以有多个article标签；例如一个博客首页可能有最新的十篇文章，每一篇 包含在一个article标签内。Article也可以通过使用section标签分为多个段落，然而当你计划你的结构时需要稍微仔细一些，否则你容易引 起以一些难看的标签大杂烩结尾的情况。

## <aside>

另一个相当模糊的标签，aside元素用于“与组成文档主要的正文流内容无关的”内容。那表示 一条附加的评论，内联的脚注，引用，注解或者像你看到的在这篇文章右边的更多典型的边栏内容。

根据WHATWG的注释，看起来<aside>可以用于所有的这些情况，尽管你边 栏里的引用和标签云有着很大的不同。

没人说HTML 5是完美的！

## <footer>

Footer的用处也应该是很明显的，除了可能你不清楚可以拥有多个脚标。换句话说，除了通常 在大多数页面底部看到的主脚标，段落也可以含有脚标。

## 把他们放在一起

让我们使用新标签重新编写我们原来的例子：

<!DOCTYPE html>

<html>

<head>…stuff..</head>

<body><header><h1>My Site</h1></header>

<nav>

<ul>

<li>Home</li><li>About</li><li>Contact</li>

</ul>

</nav>

<section><h1>My Article</h1>

<article>

</section>

<footer><p>…</p></footer>

<body>

</html>

非常清楚，并且容易理解，不是么？一些注释：我们可以在header标签中包含我们 的<h1>My Article</h1>标题。我没有这样做，因为h1元素已经表达了标题的含义，但如果你还有发布日期，署名或者其他数据在你文章的顶部， 为标签集添加一个header容器标签是一个很好的选择。

同时注意我们可以在article元素下添加第二个footer元素来包含诸如翻页导航，相关 文章或者其他内容。

## 为新标签编写样式

在大多数浏览器中，所有你需要做的就是像你通常做的那样，为在新标签上应用样式表，简单的定义 你的样式。但请确认为每一个元素添加了display:block;规则，无论如何，从现在开始。经过一段时间后，当浏览器开始标准化，并支持新元素后， 那就不必要了。

例如，让我们在我们的header里应用一些样式：

header{

display:block;

font-size:36px;

font-weight:blod;

}

记住，你仍然可以给这些标签添加类和ID属性。所以，如果你想要单独为一个导航设置样式，你可 以轻易的给这个标签添加一个类或者样式，就象这样：=

<nav class = “main-menu”>

然后你可以应用一个样式：

nav.main-menu{

font-size:18px;

}

## 兼容老的浏览器

但等一下，IE怎么办？这些样式完全不能在IE6下工作。如果你仍然需要支持像IE6一类遗产 般的浏览器，这儿有一个解决方法。IE6解析和显示这些标签还好，但你不能对他们设置任何CSS。解决方法是使用一点JavaScript。

我们只需要让IE去给我们使用createElement方法创造的的HTML 5标签设置样式。在HTML 5文件的head标签内添加这点东西。或者，你可以把他保存在一个特定的文件里，并用这种方法包含。

<script>

document.createElement('header');

document.createElement('nav');

document.createElement('section');

document.createElement('article');

document.createElement('aside');

document.createElement('footer');

</script>

我知道你在想什么：“你根本没有为那个脚本标签定义一个MIME类型。”

你根本不需要在HTML 5做这些事情。在HTML 5中，所有的脚本都被假定为type=”text/javascript”，所以没有必要让属性把你的脚本标签搞得乱七八糟（除非你的脚本并不是 JavaScript）。

这解决了IE的问题，但我们并没有摆脱困境。目前被证明Gecko渲染引擎有一个bug，导致 了Firefox2和Camino的一些版本在这些标签上卡住。

这儿有两个方法来处理这个bug，没有一个是理想的。

记住，尽管Firefox 2的使用率很快在所有网站流量中降到了10%以下，但单纯忽略这个bug可能还是需要根据你网站的访问者来定。

## 你可以使用HTML 5了，但你会用么？

简短的回答是：我们会。

复杂一点的是：那要看站点了。如果你指责重新制作CNN主页，好吧，你可能会有一点抗拒，直到 浏览器的支持变好些。但如果你要给你的博客改版，我们支持你。

但是，如果IE的缺点阻止你了，这样考虑吧：就连Google也在他们的主要搜索页面上使 用了HTML 5的文档类型。就算如果你不使用所有新的结构标签，你可以至少利用一下简洁的脚本声明。

[**开发人员需要注意的HTML5安全问题**](http://html5cn.net/archives/138.html)

发表时间：2010-05-23评论数：[1](http://html5cn.net/archives/138.html#respond)

-

HTML5给开发人员带来了新的安全挑战。苹果公司与Adobe 公司之间的口水战带来对HTML 5命运的诸多猜测，尽管HTML 5的实现还有很长的路要走，但可以肯定的一点是，运用HTML 5的开发人员将需要为应用程序安全开发生命周期部署新的安全功能以应对HTML5带来的安全挑战。

那么HTML5将会对我们需要覆盖的攻击面带来怎样的影响?

**客户端存储**

早期版本的HTML仅允许网站将cookies作为本地信息存储，而这些空间相对较小，仅适用于存储简单的档案信息或者作为存储在其他位置的数 据(例如会话ID)的标识符，Denim集团应用程序安全研究部门的主管Dan Cornell表示。然而，HTML5 LocalStorage则允许浏览器本地存储大量据库，允许使用新类型应用程序。

“随之而来的风险就是，敏感数据可能被存储在本地用户工作站，而物理访问或者破坏该工作站的攻击者，就能够轻松获得敏感数据，”Cornell 表示，“这对于使用共享计算机的用户更加危险。”

“从定义上来说，它真的只是能够在客户端系统存储信息，”Rapid7公司的安全研究人员Josh Abraham表示，“那么你就具备基于客户端SQL注入攻击的潜在能力，或者可能你的某个客户端的数据库是恶意的，当与生产系统同步时，则可能出现同步 问题，或者客户端的潜在恶意数据将被插入到生产系统。”

为了解决这个问题，开发人员需要能够验证数据是否为恶意的，这其实是个很复杂的问题。

对于这个问题的重要性并不是所有人都赞同。Veracode公司首席技术官Chris Wysopal表示，例如web应用程序通过使用插件或者浏览器扩展存储数据客户端就一直存在很多方法。

“有很多已知的方法可以操控目前部署的HTML5 SessionStorage属性，但是标准最终确定时，这个问题才会解决，”Wysopal表示。

**跨域通信**

而其他版本的HTML可能直允许JavaScript发出XML HTTP请求调用回原来的服务器，而HTML5放宽了这个限制，XML HTTP请求可以发送给任何允许这种请求的服务器。当然，如果服务器不可信任的话，这也会带来严重安全问题。

“例如，我可以建立一个mashup(糅合，将两种以上使用公共或者私有数据库的web应用合并形成一个整合应用)通过 JSON(Javascript Object Notation)将第三方网站的比赛比分拉过来，”Cornell表示，“这个网站可能会发送恶意数据到我的用户浏览器正在运行的应用程序上。虽说 HTML5允许新类型的应用程序的建立，但如果开发人员在开始使用这些功能时，并不理解他们所建立的应用程序的安全意义，那么将会给用户带来很大安全风 险。”

对于依赖于PostMessage()来编写应用程序的开发人员而言，必须仔细检查以确保信息是来源于他们自己的网站，否则来自其他网站的恶意 代码可能会制造恶意信息，Wysopal补充说。这个功能本身并不是安全的，开发人员已经开始使用不同的DOM(文档对象模型)/浏览器功能来效仿跨域通 讯。

另一个相关问题是，万维网联盟目前为跨源资源共享设计提供了一种使用类似与跨域机制绕过同源政策的方法。

“IE部署的安全功能与Firefox、Chrome以及Safari都不相同，”他指出，“开发人员需要确保他们创建过于宽松访问控制列表的 危害，特别是因为某些参考代码目前非常不安全。

**Iframe安全**

从安全角度来看，HTML5也有不错的功能，例如计划支持iframe的沙盒属性。

“这个属性将允许开发者选择数据如何解译的方式，”Wysopal表示，“不幸的是，与大部分HTML一样，这个设计很可能被开发人员误解，很 可能因为不便于使用而被开发人员禁用。如果处理得当，这个功能将能够帮助抵御恶意第三方广告或者防止不可信任内容重放。”