块级元素和行内元素有哪些：

行内元素列表：

<a>标签可定义锚

<abbr>表示一个缩写形式

<acronym>定义只取首字母缩写

<b>字体加粗

<bdo>可覆盖默认的文本方向

<big>大号字体加粗

换行

<cite>引用进行定义

<code>定义计算机代码文本

<dfn>定义一个定义项目

定义为强调的内容

<i>斜体文本效果

<img>向网页中嵌入一幅图像

<input>输入框

<kbd>定义键盘文本

<label>标签为

<input> 元素定义标注（标记）

<q>定义短的引用

<samp>定义样本文本

<select>创建单选或多选菜单

<small>呈现小号字体效果

组合文档中的行内元素

语气更强的强调的内容

<sub>定义下标文本

<sup>定义上标文本

<textarea>多行的文本输入控件

<tt>打字机或者等宽的文本效果

<var>定义变量

块级元素列表:

<address>定义地址

<caption>定义表格标题

<dd>定义列表中定义条目

定义文档中的分区或节

<dl>定义列表

<dt>定义列表中的项目

<fieldset>定义一个框架集

<form>创建 HTML 表单

<h1>定义最大的标题

<h2>定义副标题

<h3>定义标题

<h4>定义标题

<h5>定义标题

<h6>定义最小的标题

<hr>创建一条水平线

<legend>元素为

<fieldset>元素定义标题

标签定义列表项目

<noframes>为那些不支持框架的浏览器显示文本，于 frameset 元素内部

<noscript>定义在脚本未被执行时的替代内容

定义有序列表

定义无序列表

<p>标签定义段落

<pre>定义预格式化的文本

<table>标签定义 HTML 表格

<tbody>标签表格主体（正文）

<td>表格中的标准单元格

<tfoot>定义表格的页脚（脚注或表注）

<th>定义表头单元格

<thead>标签定义表格的表头

<tr>定义表格中的行
空元素：

<hr/>
<input>
<img>
<link>
<meta>

常见的行内元素、块级元素、行内块元素：

行内元素：

a,span，i（斜体）,em（强调）,sub(下标)，sup（上标），label等

块级元素：

div,h1-h6,p,pre,ul,ol,li,**form**,table,等

行内块元素：

(button,input，textarea,select), img等

html5新元素和被移除的元素：

html5新元素：

<canvas>元素：

<canvas> 标签定义图形，比如图表和其他图像。该标签基于 JavaScript 的绘图 API

新多媒体元素：

<audio> 定义音频内容

<video> 定义视频（video 或者 movie）

<source> 定义多媒体资源 <video> 和 <audio>

<embed> 定义嵌入的内容，比如插件。

<track> 为诸如 <video> 和 <audio> 元素之类的媒介规定外部文本轨道。

新表单元素：

<datalist> 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。

<keygen> 规定用于表单的密钥对生成器字段。当提交表单时，私钥存储在本地，公钥发送到服务器。

<output> 定义不同类型的输出，比如脚本的输出。输出，例如两个input的和。

新的语义和结构元素：

<article> 定义页面独立的内容区域。

<aside> 定义页面的侧边栏内容。

<bdi> 允许您设置一段文本，使其脱离其父元素的文本方向设置。
<command> 定义命令按钮，比如单选按钮、复选框或按钮
<details> 用于描述文档或文档某个部分的细节
<dialog> 定义对话框，比如提示框
<summary> 标签包含 details 元素的标题。details里面会包含summary。

```
<details>
<summary>Copyright 1999-2011.</summary>
<p> - by Refsnes Data. All Rights Reserved.</p>
<p>All content and graphics on this web site are the property of the company
Refsnes Data.</p>
</details>
```

<figure> 规定独立的流内容（图像、图表、照片、代码等等）
<figcaption> 定义 <figure> 元素的标题

```
<figure>
  <img src="img_pulpit.jpg" alt="The Pulpit Rock" width="304"
height="228">
  <figcaption>Fig1. - A view of the pulpit rock in Norway.</figcaption>
</figure>
```

<footer> 定义 section 或 document 的页脚。
<header> 定义了文档的头部区域
<mark> 定义带有记号的文本。
<meter> 定义度量衡。仅用于已知最大和最小值的度量。
<nav> 定义导航链接的部分。
<progress> 定义任何类型的任务的进度。

```
<progress value="22" max="100"></progress>
```

<ruby> 定义 ruby 注释（中文注音或字符）
<rt> 定义字符（中文注音或字符）的解释或发音
<rp> 在 ruby 注释中使用，定义不支持 ruby 元素的浏览器所显示的内容。
<section> 定义文档中的节（section、区段）。
<time> 定义日期或时间。
<wbr> 规定在文本中的何处适合添加换行符。如果单词太长，或者您担心浏览器会在错误的位置换行，那么您可以使用 <wbr> 元素来添加 Word Break

Opportunity（单词换行时机）。
<hgroup> 标签用于对网页或区段（section）的标题进行组合。
<time> 标签定义日期或时间，或者两者。
已移除的元素：
<acronym>、<applet>、<basefont>、<big>、 、<dir>、<font>、<frame>、<frameset>、<noframes>、<strike>、<tt>

label标签只有两个属性：for（规定绑定到哪个表单元素，元素的ID）、form（规定label字段所属的一个或多个表单，表单的ID）

head标签中必不可少的是title

HTML文档中的图像格式可以是：
gif、bmp、jpg、png
tif不行

html显示层级
在html中，帧元素（frameset）的优先级最高，表单元素比非表单元素的优先级要高。
表单元素包括：文本输入框，密码输入框，单选框，复选框，文本输入域，列表框等等；
非表单元素包括：连接（a），div,table,span等。
所有的html元素又可以根据其显示分成两类：有窗口元素以及无窗口元素。有

窗口元素总是显示在无窗口元素的前面。

有窗口元素包括：select元素，object元素，以及frames元素（flash）等等。
无窗口元素：大部分html元素都是无窗口元素。

置换元素和不可替换元素：

a) 置换元素：浏览器根据元素的标签和属性，来决定元素的具体显示内容。
例如：浏览器会根据<img>标签的src属性的 值来读取图片信息并显示出来，而如果查看(x)html代码，则看不到图片的实际内容；<input>标签的type属性来决定是显示输入 框，还是单选按钮等。(x)html中 的<img>、<input>、<textarea>、<select>、<object> 都是置换元素。这些元素往往没有实际的内容，即是一个空元素。
置换元素在其显示中生成了框，这也就是有的内联元素能够设置宽高的原因。

b) 不可替换元素：(x)html 的大多数元素是不可替换元素，即其内容直接表现给用户端（如浏览器）。
例如： <label>label中的内容</label> 标签<label>是一个非置换元素，文字label中的内容”将全被显示。

iframe标签：

iframe可用在以下几个场景中：

1：典型系统结构，左侧是功能树，右侧就是一些常见的table或者表单之类的。为了每一个功能，单独分离出来，采用iframe。

2：ajax上传文件。

3：加载别的网站内容，例如google广告，网站流量分析。

4： 在上传图片时，不用flash实现无刷新。

5： 跨域访问的时候可以用到iframe，使用iframe请求不同域名下的资源

上传文件的表单：

form里面加上：enctype=“multipart/form-data"

input里面加上：type="file"

meta标签中的viewport：

layout viewport：浏览器默认的viewport，也就是大于浏览器可视区的宽度，比如980px。可以用document.documentElement.clientWidth来获取

visual viewport：表示浏览器可视区域的大小。可以用window.innerWidth来获取。

ideal viewport：没有固定的尺寸。所有的iPhone的ideal viewport都是320px，也就是说所有的iPhone中css中的320px就是屏幕的宽度。

利用meta标签对viewport进行控制：

例如：<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=0”>，该meta标签的作用是让当前viewport的宽度等于设备的宽度，同时不允许用户手动缩放。

在苹果的规范中，meta viewport 有6个属性(暂且把content中的那些东西称为一个个属性和值)，如下：

width 设置layout viewport 的宽度，为一个正整数，或字符串"width-device"

initial-scale 设置页面的初始缩放值，为一个数字，可以带小数

minimum-scale 允许用户的最小缩放值，为一个数字，可以带小数

maximum-scale 允许用户的最大缩放值，为一个数字，可以带小数

height 设置layout viewport 的高度，这个属性对我们并不重要，很少使用

user-scalable 是否允许用户进行缩放，值为"no"或"yes", no 代表不允许，yes 代表允许

此外，在安卓中还支持 **target-densitydpi** 这个私有属性，它表示目标设备的密度等级，作用是决定css中的1px代表多少物理像素

target-densitydpi 值可以为一个数值或 high-dpi 、 medium-dpi、 low-dpi、 device-dpi 这几个字符串中的一个

特别说明的是，当 target-densitydpi=device-dpi 时， css中的1px会等于物理像素中的1px。因为这个属性只有安卓支持，并且安卓已经决定要废弃target-densitydpi 这个属性了，所以这个属性我们要避免进行使用。

把当前的viewport宽度设置为ideal viewport：

要得到ideal viewport就必须把默认的layout viewport的宽度设为移动设备的屏幕宽度。因为meta viewport中的width能控制layout viewport的宽度，所以我们只需要把width设为width-device这个特殊的值就行了。

<meta name="viewport" content="width=device-width">

可以通过width=device-width，所有浏览器都能把当前的viewport宽度变成ideal viewport的宽度，但要注意的是，**在iphone和ipad上，无论是竖屏还是横屏，宽度都是竖屏时ideal viewport的宽度。**

<meta name="viewport" content="initial-scale=1">

这句代码也能达到和前一句代码一样的效果，也可以把当前的的viewport变为ideal viewport。

缩放是相对于 ideal viewport来进行缩放的，当对ideal viewport进行100%的缩放，也就是缩放值为1的时候，不就得到了 ideal viewport。

测试结果表明 initial-scale=1 也能把当前的viewport宽度变成 ideal viewport 的宽度，但这次轮到了windows phone 上的**IE 无论是竖屏还是横屏都把宽度设为竖屏时ideal viewport的宽度**。

如果width 和 initial-scale=1同时出现，并且还出现了冲突呢？比如：

<meta name="viewport" content="width=400, initial-scale=1">

当遇到这种情况时，浏览器会取它们两个中较大的那个值。例如，当width=400，ideal viewport的宽度为320时，取的是400；当width=400，ideal viewport的宽度为480时，取的是ideal viewport的宽度。（ps:在uc9浏览器中，当initial-scale=1时，无论width属性的值为多少，此时viewport的宽度永远都是ideal viewport的宽度）

总结一下，要把当前的viewport宽度设为ideal viewport的宽度，既可以设置width=device-width，也可以设置 initial-scale=1，但这两者各有一个小缺陷，就是iphone、ipad以及IE 会横竖屏不分，通通以竖屏的ideal viewport宽度为准。所以，最完美的写法应该是，两者都写上去，这样就 initial-scale=1 解决了iphone、ipad的毛病，width=device-width则解决了IE的毛病：

<meta name="viewport" content="width=device-width, initial-scale=1”>

visual viewport宽度 = ideal viewport宽度 / 当前缩放值

当前缩放值 = ideal viewport宽度 / visual viewport宽度

动态改变meta viewport标签

第一种方法：

可以使用document.write来动态输出meta viewport标签，例如：

document.write('<meta name="viewport" content="width=device-width,initial-scale=1">')

第二种方法：

通过setAttribute来改变

```
<meta id="testViewport" name="viewport" content="width = 380">
<script>
var mvp = document.getElementById('testViewport');
mvp.setAttribute('content','width=480');
</script>
```

<marquee></marquee>

HTML技术中使文字滚动的方法是使用双标签<marquee></marquee>。在HTML代码中可使其作用区文字滚动，默认为从右到左，循环滚动。 <marquee direction="left">default scroll direction</marquee>

title属性和alt属性：
title：用于<a>、<p>等标签
alt：专门用于<image>标签

form标签的enctype属性：

值	描述
application/x-www-form-urlencoded	在发送前编码所有字符（默认）
multipart/form-data	不对字符编码。 在使用包含文件上传控件的表单时，必须使用该值。
text/plain	空格转换为 "+" 加号，但不对特殊字符编码。

回流与重绘

1. 当render tree中的一部分(或全部)因为元素的规模尺寸，布局，隐藏等改变而需要重新构建。这就称为回流(reflow)。每个页面至少需要一次回流，就是在页面第一次加载的时候。在回流的时候，浏览器会使渲染树中受到影响的部分失效，并重新构造这部分渲染树，完成回流后，浏览器会重新绘制受影响的部分到屏幕中，该过程成为重绘。

2. 当render tree中的一些元素需要更新属性，而这些属性只是影响元素的外观，风格，而不会影响布局的，比如background-color。则就叫称为重绘。

注意：回流必将引起重绘，而重绘不一定会引起回流。

<marquee direction="up">hello</marquee>文字滚动