超文本标记语言 (英语：**H**yper**t**ext **M**arkup **L**anguage，简称：HTML ) 是一种用来结构化 Web 网页及其内容的标记语言。网页内容可以是：一组段落、一个重点信息列表、也可以含有图片和数据表。

**一,入门介绍**

开始标签（Opening tag）：包含元素的名称（本例为 p），被左、右角括号所包围。表示元素从这里开始或者开始起作用

结束标签（Closing tag）：与开始标签相似，只是其在元素名之前包含了一个斜杠。这表示着元素的结尾果。

内容（Content）：元素的内容，本例中就是所输入的文本本身。

元素（Element）：开始标签、结束标签与内容相结合，便是一个完整的元素。

嵌套元素：元素放到其它元素之中——这被称作嵌套。

块级元素和内联元素

块级元素：单独一行，有宽度和高度，block形式不会嵌套进行内元素中，但可以嵌套在块级元素 div,p,h1,ul,ol,li,dl,dt,dd，br，hr

行内元素：可以并排一行，没有宽度和高度，不会文本换行 em,span,strong,b，a,img,input

空元素

元素: html 根标签，head 头部标签，<meta charset="utf-8"> 设置文档使用utf-8字符集编码，title 网页标题，<body> 标签页面显示的内容

属性

标签自带属性-img的src属性

布尔属性-input表单标签的disabled属性，表示禁用与否，true和false

标签属性-使用单引号和双引号包裹住

字符引用 —— 表示字符的特殊编码 &lt;&gt;&quot;&amp;

html中的空白是没有必要的，HTML解释器会将连续出现的空白字符减少为一个单独的空格符。

注释是被浏览器忽略的，而且是对用户不可见的，它们的目的是允许你描述你的代码是如何工作的和不同部分的代码做了什么等等。

如果你在半年后重新返回你的代码库，而且不能记起你所做的事情 —— 或者当你处理别人的代码的时候， 那么注释是很有用的.

<head>标签里有什么

内容不会在浏览器中显示,保存页面的一些 元数据

meta 文档的字符编码,文档中被允许使用的字符集。

name:指定了meta 元素的类型； 说明该元素包含了什么类型的信息。

content:指定了实际的元数据内容。

网页字体标签：<link rel="shortcut icon" href="favicon.ico" type="image/x-icon">

 <link> rel="stylesheet"表明这是文档的样式表，而 href包含了样式表文件的路径：

<script> 放在文档的尾部（在 </body>标签之前）是一个更好的选择，这样可以确保在加载脚本之前浏览器已经解析了HTML内容

lang属性 lang="zh-CN"

文档的分段设置为不同的语言 <p>日语实例: <span lang="jp">ご飯が熱い。</span>.</p>

文字处理基础

内容结构化会使读者的阅读体验更轻松，更愉快。

标题级别钟每页不能使用超过3个

SEO (搜索引擎优化) 是一种让网站在搜索引擎结果中更加清晰, 也帮助我们将搜索结果更靠前：https://developer.mozilla.org/zh-CN/docs/Glossary/SEO

为什么需要语义化 标签的语义值

身边的任何地方都要依赖语义学 — 我们依靠以前的经验就知道日常事物都代表什么；当我们看到什么，我们就会知道它代表什么。举个例子,

我们知道红色交通灯表示“停止”，绿色交通灯表示”通行“。如果运用了错误的语义，事情会迅速地变得非常棘手 (难道有某个国家使用红色代表通行？我不希望如此

同样的道理，我们需要确保使用了正确的元素来给予内容正确的意思、作用以及外形。

元素也是一个语义元素，它给出了包裹在您的页面上用来表示顶级标题的角色（或意义）的文本

重要的是，它的语义值将以多种方式被使用，比如通过搜索引擎和屏幕阅读器

<span style="font-size: 32px; margin: 21px 0;">这是顶级标题吗？</span>

<span> 元素，它没有语义。当您想要对它用CSS（或者JS）时，您可以用它包裹内容，且不需要附加任何额外的意义（在未来的课程中你会发现更多这类元素）。

我们已经对它使用了CSS来让它看起来像一个顶级标题。然而，由于它没有语义值，所以它不会有任何上文提到的帮助。

超连接

通过将文本（或其他内容，见块级链接)转换为<a>元素内的链接来创建基本链接， 给它一个href属性（也称为目标），它将包含您希望链接指向的网址。

指向当前目录 contacts.html

指向子目录 projects/index.html

指向上级目录 ../pdfs/project-brief.pdf

超链接文档片段-

<h2 id="Mailing\_address">邮寄地址</h2>

<a href="contacts.html#Mailing\_address">

<p>本页面底部可以找到 <a href="#Mailing\_address">公司邮寄地址</a>。</p>

绝对URL和相对URL

绝对URL：指向由其在Web上的绝对位置定义的位置，包括 protocol（协议） 和 domain name（域名）。

相对URL：指向与您链接的文件相关的位置如果PDF文件能够在projects的子目录pdfs中访问到，相对路径就是pdfs/project-brief.pdf（对应的绝对URL是

http://www.example.com/projects/pdfs/project-brief.pdf）

1.尽可能使用相对路径,检查代码要容易得多——相对URL通常比绝对URL短得多，这使得阅读代码更容易,

2.在可能的情况下使用相对URL更有效。当使用绝对URL时，浏览器首先通过DNS（见万维网是如何工作的）查找服务器的真实位置，然后再转到该服务器并查找所请求的文件

3.相对URL，浏览器只在同一服务器上查找被请求的文件。因此，如果你使用绝对URL而不是相对URL，你就会不断地让你的浏览器做额外的工作，这意味着它的效率会降低。

链接到非HTML资源 ——留下清晰的指示

当链接到一个需要下载的资源（如PDF或Word文档）或流媒体（如视频或音频）或有另一个潜在的意想不到的效果（打开一个弹出窗口，或加载Flash电影），添加明确的措辞，以减少任何混乱

如果你是在低带宽连接，点击一个链接，然后就开始下载大文件。

如果你没有安装Flash播放器，点击一个链接，然后突然被带到一个需要Flash的页面。

在下载链接时使用 download 属性

当您链接到要下载的资源而不是在浏览器中打开时，您可以使用 download 属性来提供一个默认的保存文件名；

当点击一个链接或按钮时，打开一个新的电子邮件发送信息而不是连接到一个资源或页面，这样做是使用<a>元素mailto：URL的方案。

块引用

块级内容 div

行内引用

行内元素用同样的方式工作，浏览器默认将其作为普通文本放入引号内表示引用

缩略语 <abbr>

下标<sub>8</sub>

上标<sup>2</sup>

//较少使用

<code>: 用于标记计算机通用代码。

<pre>: 用于保留空白字符（通常用于代码块）——如果您在文本中使用缩进或多余的空白，浏览器将忽略它，您将不会在呈现的页面上看到它。但是，如果您将文本包含在

<pre></pre>标签中，那么空白将会以与你在文本编辑器中看到的相同的方式渲染出来。

<var>: 用于标记具体变量名。

<kbd>: 用于标记输入电脑的键盘（或其他类型）输入。

<samp>: 用于标记计算机程序的输出

**要敬畏语义，做到正确选用元素。**

这是因为视觉效果并不是一切。 我们可以修改最重要内容（例如导航菜单和相关链接）的颜色、字体大小来吸引用户的注意，**但是这对视障人士是无效的**，“粉红色”和“大字体”对他们并不奏效。

文档的基本组成部分

页眉：通常横跨于整个页面顶部有一个大标题 和/或 一个标志。 这是网站的主要一般信息，通常存在于所有网页。

导航栏：指向网站各个主要区段的超链接。通常用菜单按钮、链接或标签页表示。

主内容:中心的大部分区域是当前网页大多数的独有内容

侧边栏:一些外围信息、链接、引用、广告等

页脚:横跨页面底部的狭长区域。

<header>：页眉。

<nav>：导航栏。

<main>：主内容。主内容中还可以有各种子内容区段，可用<article><section>和<div>等元素表示。

<aside>：侧边栏，经常嵌套在<main>中。

<footer>：页脚。

**无语义标签**

HTML提供了 [<div>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/div) 和 [<span>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/span) 元素。应配合使用 [class](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Global_attributes#attr-class) 属性提供一些标签，使这些元素能易于查询。

**html调试**

浏览器并不会将 HTML 编译成其它形式，而是直接解析并显示结果（称之为解释，而非编译），浏览器解析 HTML 的过程比编程语言的编译运行的过程要宽松得多，但这是一把双刃剑。

语法错误

：由于拼写错误导致程序无法运行，就像上面的 Rust 示例。通常熟悉语法并理解错误信息后很容易修复。

逻辑错误：

不存在语法错误，但代码无法按预期运行。通常逻辑错误比语法错误更难修复，因为无法得到指向错误源头的信息。

1在浏览器中打开Markup Validation Service ：https://validator.w3.org/#validate\_by\_input

2.选择 Validate by Direct Input标签。

3.将整个示例文档的代码（而不仅仅是body部分）复制粘贴到正中的文本框内。

4.点击 Check按钮。将返回一个包含错误和其它信

**二,多媒体与嵌入**

img图片版权问题。

1.像<img>和<video>这样的元素有时被称之为替换元素，因为这样的元素的内容和尺寸由外部资源（像是一个图片或视频文件）所定义，而不是元素自身。

2.属性alt ，它的值应该是对图片的文字描述，用于在图片无法显示或不能被看到的情况，图片显示不出来就会显示alt文本

3.title属性来提供需要更进一步的支持信息

4.<video> 元素 ：https://developer.mozilla.org/zh-CN/docs/Learn/HTML/Multimedia\_and\_embedding/Video\_and\_audio\_content

5. Javascript 中调用 load() 方法来重置媒体 更多视频属性请观看上方文档

6.<iframe>元素旨在允许您将其他Web文档嵌入到当前文档中，嵌入第三方内容。

嵌入标签，的详解以及安全隐患了解更多请阅读：https://developer.mozilla.org/zh-CN/docs/Learn/HTML/Multimedia\_and\_embedding/Other\_embedding\_technologies

7.<embed>和<object>元素的功能不同于<iframe>—— 这些元素是用来嵌入多种类型的外部内容的通用嵌入工具，其中包括像Java小程序和Flash，PDF（可在浏览器中显示为一个PDF插件）插件技术，

甚至像视频，SVG和图像的内容！了解更多请阅读：https://developer.mozilla.org/zh-CN/docs/Learn/HTML/Multimedia\_and\_embedding/Other\_embedding\_technologies#%3Cembed%3E%E5%92%8C%3Cobject%3E%E5%85%83%E7%B4%A0

8.矢量图片

位图使用像素网格来定义 — 一个位图文件精确得包含了每个像素的位置和它的色彩信息。流行的位图格式包括 Bitmap (.bmp), PNG (.png), JPEG (.jpg), and GIF (.gif.)

矢量图使用算法来定义 — 一个矢量图文件包含了图形和路径的定义，电脑可以根据这些定义计算出当它们在屏幕上渲染时应该呈现的样子。SVG格式可以让我们创造用于 Web 的精彩的矢量图形。

当你放大网页的时候，区别就会变得明显起来 — 随着你的放大，PNG 图片变得像素化了，因为它存储是每个像素的颜色和位置信息 — 当它被放大时，每个像素就被放大以填满屏幕上更多的像素，所以图像就会开始变得马赛克感觉。

矢量图像看起来仍然效果很好且清晰，因为无论它的尺寸如何，都使用算法来计算出图像的形状，仅仅是根据放大的倍数来调整算法中的值。

要通过 <img>元素嵌入SVG，你只需要按照预期的方式在 src 属性中引用它。

如何在HTML中引入SVG代码

如何使用 <iframe> 嵌入SVG

9.自适应图片问题

辨率切换问题（resolution switching problem） 如果是在小屏手机屏幕上显示网页，那么没有必要在网页上嵌入这样大的图片

美术设计问题（art direction problem）。当网站在狭窄的屏幕上观看时，显示一幅图片的包含了重要细节的裁剪版本，第二个被裁剪的图片会在像平板电脑这样的中等宽度的屏幕设备上显示

分辨率切换：不同的尺寸

使用两个新的属性——srcset 和 sizes——来提供更多额外的资源图像和提示，帮助浏览器选择正确的一个资源。