HTML Learning

<p>标签代表 paragraph，在写网页的的时候尽量使用小写，在使用VSC的时候，按下P+tab可自动生成

```
<p>Hello world</p>
```

<h1>为一级标题，作为标题最多可以有6级<h1>, <h2>, <h3>, <h4>….. <h6>.

```
<h1>Title1</h1>
<h2>Title2</h1>
...
<h6>Title6</h1>
```

对于任何一个物品，他都有自己的Attribute和method，在HTML中所有的tags都是Object，他们都有自己Attribute和method

作为HTML的核心，也就是HTML Skeleton（Boilerplate）是由两个部分组成。

- Head
- Body

在Head中放入网页背后的基本设定与编码，并且在HTML最开始要给网页定义document Type，以及在网页中所使用的语言，例如

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="utf-8" />
 <meta name="viewport" content="width=device-width, initial-scale=1" />
 <meta name="robots" content="index,follow" />
 <meta name="description" content="This is a demo," />
 <meta name="author" content="YouMIng Zheng" />
</head>
```

在Body中，则主要写入网页的内容，包括一些文字，图片，链接 etc

```
<body>
  <h1>Title for the demo</h1>
  <p> This is a demo</p>
</body>
```

<a> 代表anchor tag 适用于建立超链接（hypertext reference）

```
<a href="website link">click context</a>
```

<base> tag是放在<head>内，base must have either an herb of a target attribute present or both. 在target 有4种key可以选择

```
<base traget="_self"/>
<!--show the result in the current browsing context（不开启新页面）-->

<base traget="_blank"/>
<!--show the result in a new, unnamed browsing contex（开启新页面）-->
```

```
<base traget="_parent"/>
<!-- show the result in the parent browsing context of the current one,is the
current page is inside a frame. If there is no parent, acts the same as _self-->

<base traget="_top"/>
<!-- show the result in the topmost browsing context(the browsing context that is ancestor
of the current one and has no parent).If there is no parent, acts the same _self -->
```

因为<base>是在head内，所以当定义为target之后，所有的链接都会随之改动，但是当如果某特殊的链接想采用不同的target时，直接在anchor tag中自己定义就可以。

```
<head>
  <base target="_self"/>
</head>
<body>

  <a herf="https://www.google.ca/?gfe_rd=cr&ei=0KeoWcvDDs-fXo39ktgK&gws_rd=ssl"> Google</a>
  <!-- Google will follow the target define in the <base>, because in the anchor
  it does not have a traget, this line will follow the traget in the <base> -->

  <a target="_blank" herf="https://www.ualberta.ca/index.html">university of Alberta</a>
  <!-- The university of Alberta link will not follow the target define in the <base>
  Bacause the target is already define in the anchor. -->
</body>
```

对于添加图片，HTML中有两种方式：

- Absolute file path
- Relative file path

Absolute file path 是通过输入图片的URL并对其进行调用，优点是不用存放在本地文件里不占用空间，缺点是当所制定网站的图片URL失效，你的网站图片也会失效。

Relative file path 是通过内部文件的调用，并显示在网页中，优点是不受别的网站影响，因为存放在自己本地文件夹内，缺点则是占用一定量的资源。

对于Absolute file path 通常存放在本地，但是需要手动更新。但是对于Relative file path 它是通过调用别的网站或者外部资源来显示在自己的网页上，所以需要依赖于其他网页的更新来更新自己的网页调用的内容。通常比如天气预报，股票市场是使用Relative file path，因为这些在实时更新的内容如果自己没有数据则无法完成更新。通常图片视频之类的本地文件使用Absolute file path，因为这些内容大部分不需要更新，所以如果使用Relative file path来调用这些图片和视频，当你调用的网页地址发生了改变会在系统崩溃了，你的网页也没有办法显示出正常的内容。

对于两个方式，写法都是一样除了制定地址不一样

```
<img src="https://upload.wikimedia.org/wikipedia/commons/thumb/2/28/HelloWorld.svg/1024px-HelloWorld.svg.png" alt="Hello world"/>
<img src="../source/image/helloword.png" alt="Helloworld"/>
<!-- .代表当前html所在文件夹，..代表最顶层文件夹-->
```

List 列表，在HTML中间有两种标签

- UL （unordered list)
- OL （ ordered list)

```
<ol>
  <li>CSS</li>
  <li>HTML</li>
  <li>JAVASCRIPT</li>
</ol>
```

Output:

1. CSS
2. HTML
3. JAVASCRIPT

```
<ul>
  <li>CSS</li>
  <li>HTML</li>
  <li>JAVASCRIPT</li>
</ul>
```

Output:

- CSS
- HTML
- JAVASCRIPT

Table 在HTML中，有4个标签一定需要，同时也有三个不一定需要

- <table>
- <tr>. Table row
- <th>. Table heading
- <td>. Table data

Unnecessary tag in HTML table

- <thead>
- <tbody>
- <tfoot>

```
<head>
<style>
  table,
  th,
  td{
    border-collapse: collapse;
    border: 1px solid black;
  }
  th,
  td {
    padding: 0.25rem;
  }
</style>
</head>
<body>
<table>
 <thead>
  <tr>
```

```
    <th colspan="3">GPU Table</th>
  </tr>
  <!-- colspan 设定为3会占用三个，并起到像下方开头标题的那种感觉，同理还有rolspan -->
  <tr>
    <th>Lithography</th>
    <th>Year</th>
    <th>Model</th>
  </tr>
 </thead>
 <tbody>
  <tr>
    <th>16nm</th>
    <th>GTX 1070</th>
    <th>2016</th>
  </tr>

  <tr>
    <th>12nm</th>
    <th>RTX 2070</th>
    <th>2018</th>
  </tr>
  <tr>
    <th>8nm</th>
    <th>RTX 3080</th>
    <th>2020</th>
  </tr>
 </tbody>
</table>
</body>
```

Output:

GPU Table

Lithography	Year	Model
16nm	2016	GTX1070
12nm	2018	RTX2070
8nm	2020	RTX3080

form 在HTML中需要连接到网页的后端数据库(database)

- action (资料所要传去的目的地）
- method (有get和post区分）
  - method get 和 post最基本的区别则是，get会在用户在input输入后显示在URL上，然而post则不会

  在form里有label标签(不强制使用)，label标签中有for=“”在input label中则有id=“”. 当for中所定的setting对对应id中的setting时，点击label则可锁定input内容。

  在input label中，如果需要后段数据库收到输入内容，需要在input lable中定义name属性，只有在定义了name属性后，后段才能接受的数据，并且可以通过网站的网址来判断是否后段收到了数据.

```
<form action="" method="GET">
  <label for="myname">姓名：</label>
  <input id="myname" type="text" name="InputNmae" value="" required/>
  <button type="submit">提交</button>
</form>
```

Input

- checkbox

checkbox是input这个label里的一种type，点击后可以打勾,value是传入后段的数据

```
<input id="check" type="checkbox" name="news" value="newspapaer"/>
<label for="check">订阅新闻</label>
```

- email

email type是要求用户输入一个email address, 如果用户输入的不是email,则会弹出错误窗口。和input比较类

```
<label for="email">邮箱</label>
<input id="email" type="email" name="emailadd" required/>
```

- file

file type 是要求用户从自己电脑上存放进文件进入系统的数据库

```
<input type="file" name="file" />
```

- number

number type 中有特殊的min和max的设定和step设定，min和max是设定用户input数的最大值和最小值，step则是设定每次增加的数值，默认是+1所以当需要用到一位小数点时, step 设定为 “0.01”

```
<label for="Height">身高</label>
<input id="Height" type="number" name="user_Height" value="" min="0" max="300" required step="0.01"/>
```

- password

password type 是要求用户输入密码，密码输入后不会显示出来。并使用maxlength和minlength来限定密码必须达到的长度的区间，并可以使用placeholder来提示用户改输入什么

```
<input type="password" name="" minlength="8" maxlength="12" value="" placeholder="Please enter password"/>
```

- range

range type 和number type比较类似

```
0<input type="range" min="0" max="120" step="1" name="" value=""/>120
```

- radio

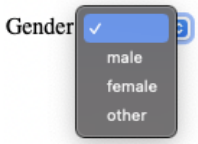
radio type 和checkbox非常类似，但是radio是在一个集合里选择一个值，并且radio会通过name去判断哪些内容是一个集合,因此required只要在一个集合中的任意一个input tag中就可以达到效果。

```
<input id="male" type="radio" name="gender" value="male" required/>
<label for="male">男性</label>
<input id="female" type="radio" name="gender" value="female"/>
<label for="female">女性</label>
```

select tag 和 option tag 在HTML中是一个非常常见的tag，注意的是如果需要用户必须选择，可以在select tag中写入 required. 如果想要在选项默认为空，则设定一个空的option tag。

```
<label for="gender">Gender:</label>
<select name="gender" id="gender" required>
  <option value=""></option>
  <option value="male">male</option>
  <option value="female">female</option>
  <option value="other">other</option>
</select>
```

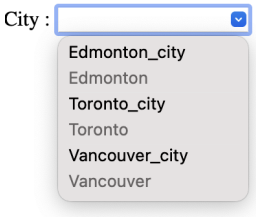
Output:



Datalist 可以让用户选择并输入制定的内容，在datalist中一定要确定 id对应input里的list。并且在datalist中传入数据库的是option value里的值，它不一定与显示的一样。

```
<label for="city">City : </label>
<input list="city_list" type="text" name="city" id="city" required/>
<datalist id="city_list">
  <option value="Edmonton_city">Edmonton</option>
  <option value="Toronto_city">Toronto</option>
  <option value="Vancouver_city">Vancouver</option>
</datalist>
```

Output:



Block and inline的排版区别，Block是block elemet，它等于device width（width：100%）相反Inline element的排版等于里面内容的宽度。

https://www.w3schools.com/html/html_blocks.asp

Icon 在HTML中是网页在上方的一个类似标志的图像，想要实现这个功能需要在<head>中添加

```
<link rel="shortcut icon" hred="link for the picture"/>
<link rel="bookmark" hred="link for the picture"/>
```

self closing tag : The important distinction between self-closing tags and all other tags is that self-closing tags represent void elements. Void elements like imp and br cannot contain any content. All other tags may (not required) contain content.

```
<!-- self closing tag example -->
<img src="" alt="" />
<br />
<!-- Non-self closing tag example -->
<h1> </h1>
```