第二部分[a]

第 6 周：网络
- 1 个 HTTP
  - 1.1. 设置
  - 1.2. 探索 HTTP
  - 1.3. 在线研究
  - 1.4. Java 中的服务器
- 2 HTML5 格式
  - 视频
  - 2.1. 基本 HTML5
  - 2.2. 模板

第 7 周：CSS
- 3 CSS 系统
  - 3.1. 设置文本样式
  - 3.2. 框架
- 4 个 CSS 网格
  - 4.2. 课程练习
  - 4.3. 树木练习（响应式布局）

第 8 周：Javascript

第 9 周：网页抓取
- 7 网页抓取
  - 7.1. 爬行
  - 7.2. 美汤

第 10 周：实用加密
- 8 实用加密
  - 8.1. 开放 SSL
  - 8.2. PGP 的

流浪文件

第 6 周：网络

1 个 HTTP

1.1. 设置

1.2. 探索 HTTP

1.3. 在线研究

URL 的片段部分。

URI 片段是以哈希 （#） 字符开头的 URL 末尾的可选部分。它允许您引用已访问的文档的特定部分。

来源： 如何共享锚定到网页上任何文本的链接

URL：哈希属性 – Web API |MDN 的

HTTP 客户端发送的 Accept 标头。

接受请求 HTTP 标头指示客户端能够理解哪些内容类型（表示为 MIME 类型）

来源：接受 – HTTP |MDN 的

Accept 标头是 HTTP 请求头的一部分，用于告知服务器客户端所能接受的 MIME 类型。这样服务器就可以根据客户端的偏好来选择合适的内容类型进行响应。

Accept 标头的值是一个由逗号分隔的 MIME 类型列表，每个 MIME 类型都有一个可选的关联权重。权重越高表示客户端越偏好该内容类型。

例如，以下是一个示例 Accept 标头：

```
Accept: text/html, application/xhtml+xml, application/xml;q=0.9,
*/*;q=0.8
```

解释：

- `text/html`：表示客户端能够接受 HTML 文档。

- `application/xhtml+xml`：表示客户端能够接受 XHTML 文档。

-`application/xml;q=0.9`：表示客户端能够接受 XML 文档，但是给予的权重较低（0.9）。

- `*/*;q=0.8`：表示客户端能够接受任意类型的内容，但是给予的权重最低（0.8）。

这个 Accept 标头表示客户端更偏好 HTML 和 XHTML 文档，稍次于这两种类型的是 XML 文档，最后是任意类型的内容。

HTTP 客户端发送的 User-agent 标头。您的浏览器会发送什么？

User-Agent 请求标头是一个特征字符串，它允许服务器和网络对等方标识请求用户代理的应用程序、操作系统、供应商和/或版本。

来源：User-Agent - HTTP |MDN 的

如何对 URL 中包含空格的路径进行编码？还有哪些角色是"特殊"的，需要在路径中以这种方式对待？

URL 不能包含空格。URL 编码通常用加号 （+） 或 %20 替换空格。

这里列出了许多特殊字符：

HTML URL 编码参考。

根据标题，布里斯托大学使用哪个 Web 服务器进行 www.bristol.ac.uk？在线阅读此服务器和维护它的同名组织。

阿帕奇。

https://httpd.apache.org/

源：

1.4. Java 中的服务器

2 HTML5 格式

视频

无序列表 <ul> 和有序列表 <ol> 有什么区别？

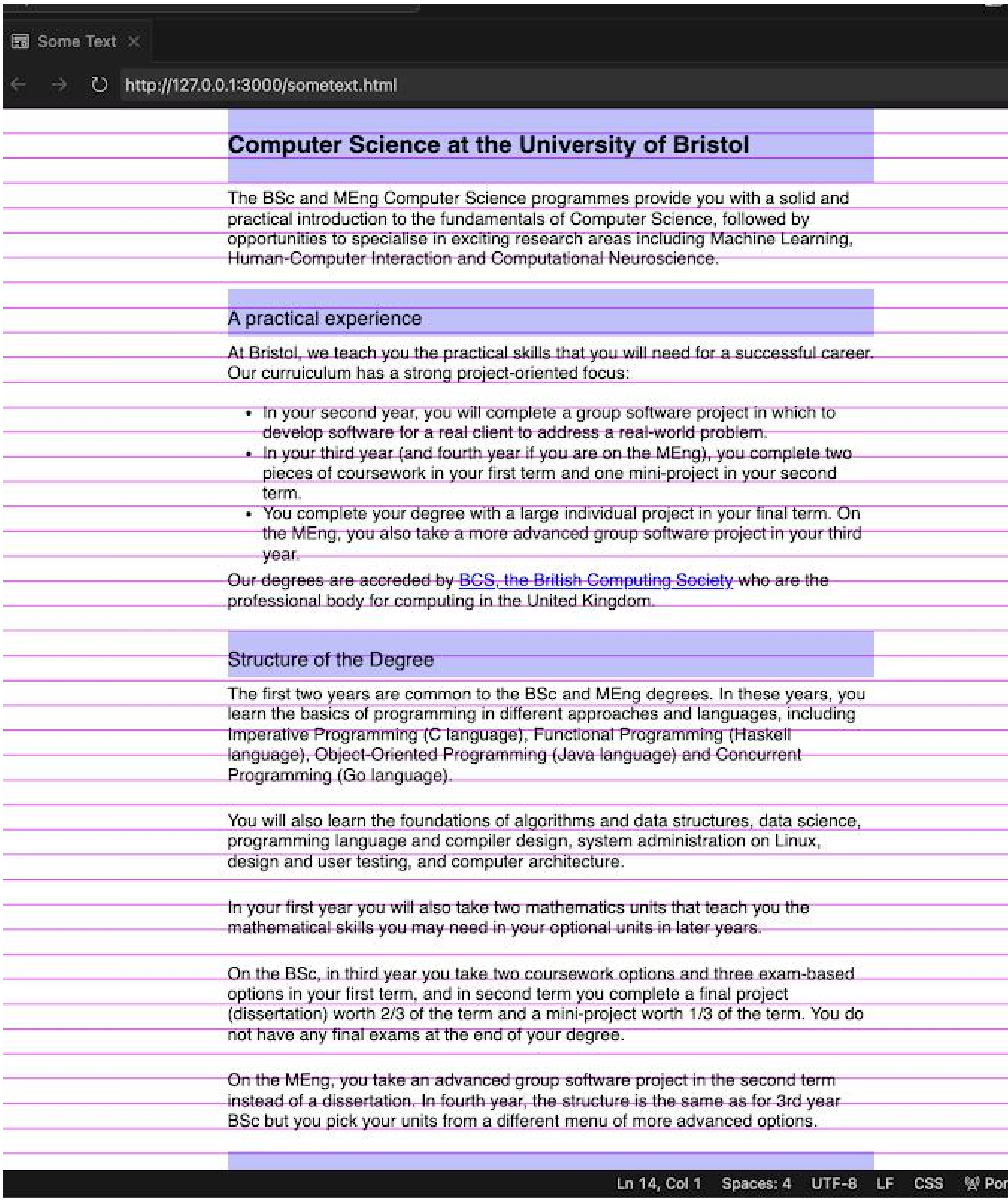
2.1. 基本 HTML5

2.2. 模板

第 7 周：CSS

3 CSS 系统

3.1. 设置文本样式


Some Text

http://127.0.0.1:3000/sometext.html

Computer Science at the University of Bristol

The BSc and MEng Computer Science programmes provide you with a solid and practical introduction to the fundamentals of Computer Science, followed by opportunities to specialise in exciting research areas including Machine Learning, Human-Computer Interaction and Computational Neuroscience.

A practical experience

At Bristol, we teach you the practical skills that you will need for a successful career. Our curruiculum has a strong project-oriented focus:

- In your second year, you will complete a group software project in which to develop software for a real client to address a real-world problem.
- In your third year (and fourth year if you are on the MEng), you complete two pieces of coursework in your first term and one mini-project in your second term.
- You complete your degree with a large individual project in your final term. On the MEng, you also take a more advanced group software project in your third year.

Our degrees are accreded by BCS, the British Computing Society who are the professional body for computing in the United Kingdom.

Structure of the Degree

The first two years are common to the BSc and MEng degrees. In these years, you learn the basics of programming in different approaches and languages, including Imperative Programming (C language), Functional Programming (Haskell language), Object-Oriented Programming (Java language) and Concurrent Programming (Go language).

You will also learn the foundations of algorithms and data structures, data science, programming language and compiler design, system administration on Linux, design and user testing, and computer architecture.

In your first year you will also take two mathematics units that teach you the mathematical skills you may need in your optional units in later years.

On the BSc, in third year you take two coursework options and three exam-based options in your first term, and in second term you complete a final project (dissertation) worth 2/3 of the term and a mini-project worth 1/3 of the term. You do not have any final exams at the end of your degree.

On the MEng, you take an advanced group software project in the second term instead of a dissertation. In fourth year, the structure is the same as for 3rd year BSc but you pick your units from a different menu of more advanced options.

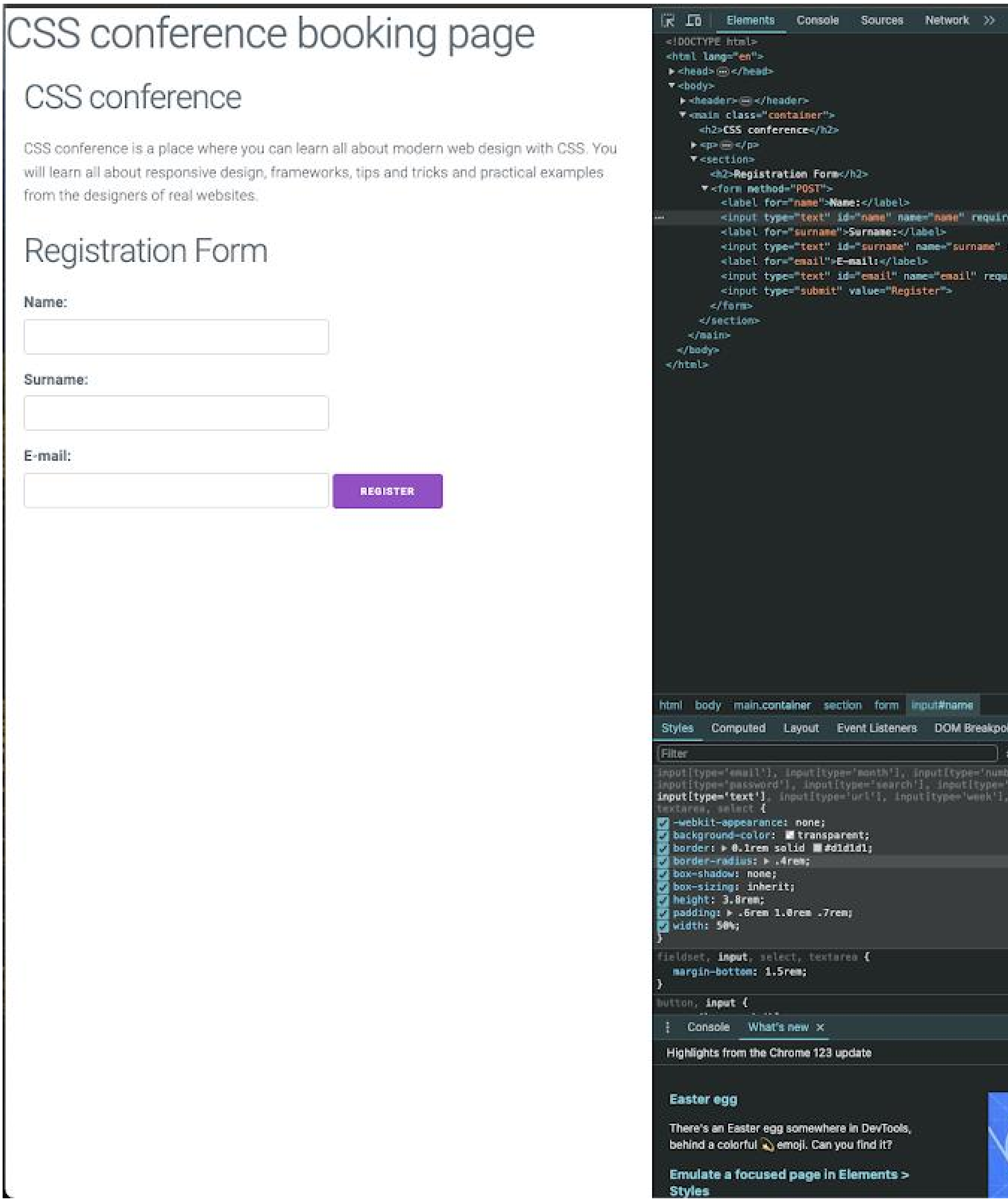
Ln 14, Col 1 Spaces: 4 UTF-8 LF CSS

```
body {

    margin: 0 auto;
    max-width: 40em;

    line-height: 125%;

    font-size: small;

    font-family: sans-serif;

    background-image: url("baseline.png");


}

h1{

    font-size: 150%;

    font-weight: bold;

    background-color: rgba(0, 0, 255, 0.25);

    padding-top: 4%;

    padding-bottom: 4.2%;

}

h2{

    font-size: medium;

    background-color: rgba(0, 0, 255, 0.25);

    padding-top: 3%;

    padding-bottom: 1.2%;

}

p{

    font-size: 14px;
    padding-top: 1%;

    padding-bottom: 3%;

}

ul{

    padding-left: 2em; /* the bullets appear in the padding */

    list-style-type: disc;

    list-style-position: outside;

    font-size: 14px;

}

a{

    color:blue;

}
```

3.2. 框架

Milligram 如何设置以下内容？

- 标题字体的大小
    - 字体大小：3.6rem;
- 表单字段占据了容器的整个宽度
    - 宽度： 100%;
- 表单标签和字段显示在彼此的下方[c]
- 标签比上面的字段更接近自己的字段
    - 边距底部：.5rem;
- 在足够宽的屏幕上将容器中的所有东西的大小和居中[d]



CSS conference booking page

CSS conference

CSS conference is a place where you can learn all about modern web design with CSS. You will learn all about responsive design, frameworks, tips and tricks and practical examples from the designers of real websites.

Registration Form

Name:

Surname:

E-mail:

```
<!DOCTYPE html>
<html lang="en">
 ▶<head> ⋯ </head>
 ▼<body>
   ▶<header> ⋯ </header>
   ▼<main class="container">
      <h2>CSS conference</h2>
     ▶<p> ⋯ </p>
     ▼<section>
        <h2>Registration Form</h2>
       ▼<form method="POST">
          <label for="name">Name:</label>
          <input type="text" id="name" name="name" requir
          <label for="surname">Surname:</label>
          <input type="text" id="surname" name="surname"
          <label for="email">E-mail:</label>
          <input type="text" id="email" name="email" requ
          <input type="submit" value="Register">
        </form>
      </section>
    </main>
  </body>
</html>
```

```
input[type='email'], input[type='month'], input[type='numb
input[type='password'], input[type='search'], input[type='
input[type='text'], input[type='url'], input[type='week'],
textarea, select {
  -webkit-appearance: none;
  background-color: transparent;
  border: ▶ 0.1rem solid #d1d1d1;
  border-radius: ▶ .4rem;
  box-shadow: none;
  box-sizing: inherit;
  height: 3.8rem;
  padding: ▶ .6rem 1.0rem .7rem;
  width: 50%;
}
fieldset, input, select, textarea {
  margin-bottom: 1.5rem;
}
button, input {
```

Highlights from the Chrome 123 update

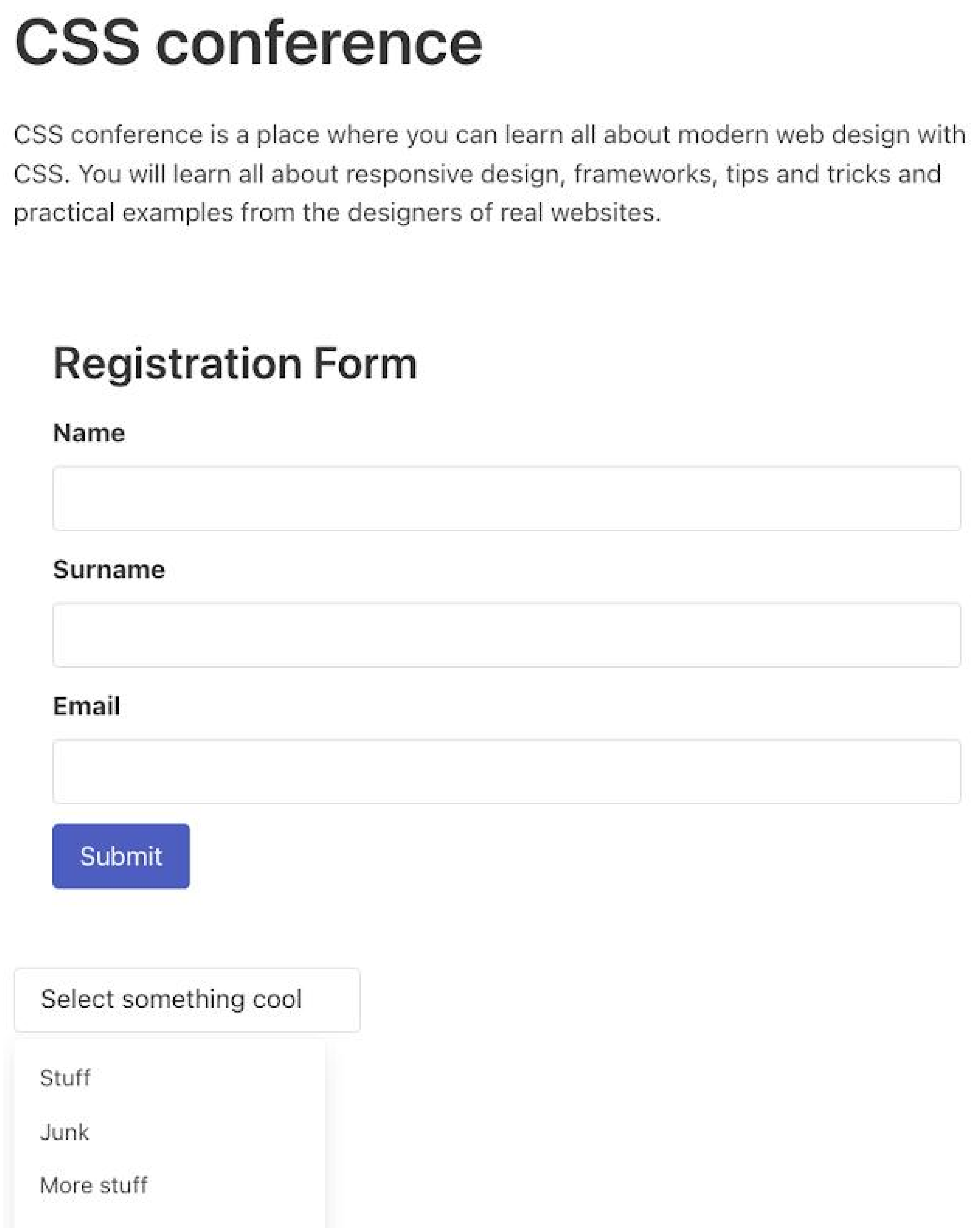
Easter egg

There's an Easter egg somewhere in DevTools, behind a colorful emoji. Can you find it?

Emulate a focused page in Elements > Styles

布尔玛

https://pastebin.com/QEGVjuR6 [e]



CSS conference

CSS conference is a place where you can learn all about modern web design with CSS. You will learn all about responsive design, frameworks, tips and tricks and practical examples from the designers of real websites.

Registration Form

Name

Surname

Email

Submit

Select something cool

Stuff

Junk

More stuff

4 个 CSS 网格

4.2. 课程练习

艾萨克的解决方案：

```
main {

        max-width: 1500px;

        margin: 0 auto;

        display: inline-grid;



        grid-template-columns: repeat(12, 1fr);

        gap: 15px;

        grid-template-rows: repeat(12, 1fr);

}


body {

        font-family: sans-serif;

        background-color: rgba(112, 145, 53, 0.1);

}


.unit {

        background-color: rgba(0, 67, 79, 0.2);

        width: auto;

}


.unit b {

        display: block;

        background-color: rgb(0, 67, 79);

        color: white;

        padding: 5px;

}


.cp10 {

        grid-auto-flow: column;

        grid-column: span 2;

}


.cp15 {

        grid-auto-flow: column;

        grid-column: span 3;

}


.cp20 {

        grid-auto-flow: column;

        grid-column: span 4;

}


.cp40 {

        grid-auto-flow: column;
```

```css
        grid-column: span 8;
}

.y1-tb1 {
        grid-row: 1;
}

.y1-tb2 {
        grid-row: 2;
        margin-bottom: 15px;
}

.y2-tb1 {
        grid-row: 3;
}

.y2-tb2 {
        grid-row: 4;
        margin-bottom: 15px;
}

.y3-tb1 {
        grid-row: 5;
}

.y3-tb2 {
        grid-row: 6;
        margin-bottom: 15px;
}

.y2-tb4 {
        grid-auto-flow: column;
        grid-column: span 2;
        grid-row: span 2;
        margin-bottom: 15px;
}

.unit b ~ p {
        padding-left: 12px;
        padding-right: 12px;
        align-content: center;
}
```

4.3. 树木练习（响应式布局）

艾萨克的解决方案：[f]

```
body {
    font-family: "Open Sans", sans-serif;
    margin: 0;
}

header {
    background-color: #04443c;
    color: white;
    margin: 0;
    padding-left: 10px;
    padding-right: 10px;
}

@media only screen and (max-width: 400px) {

.container {
        margin-left: 10px;
        margin-right: 10px;
        display: inline-grid;
        grid-template-columns: repeat(1, 1fr);
        grid-gap: 20px;
}

}

@media only screen and (min-width: 400px) {

.container {
        margin-left: 10px;
        margin-right: 10px;
        display: inline-grid;
        grid-template-columns: repeat(2, 1fr);
        grid-gap: 20px;
}

}

@media only screen and (min-width: 600px) {
        .container {
```

```css
                margin-left: 10px;
                margin-right: 10px;
                display: inline-grid;
                grid-template-columns: repeat(4, 1fr);
                grid-gap: 20px;
        }
}

.featured {
        grid-column: span 2;
        padding-right: 15;
}

.card {
        margin: 0 auto;
  background-color: #167f5f;
  color: white;
  padding: 10px, 10px, 10px, 10px;
}

.card-image {
  max-width: 100%;
  height: auto;
}

.card span.latin-name {
  display: block;
  font-style: italic;
  padding: 10px;
  padding-bottom: 0;
}

.card span.common-name {
  display: block;
  padding: 10px;
  padding-top: 5px;
  font-weight: bold;
}
```

第 8 周：Javascript

注意：此代码要求您将 YOUR_API_KEY 替换为 API 密钥。否则，它是练习第一部分的解决方案。

感谢 Sam 提供此解决方案。

```
const IMG_PATH = 'https://image.tmdb.org/t/p/w500'

const SEARCH_API = `https://api.themoviedb.org/3/search/movie?api_key=YOUR_API_KEY&query="`

const API_URL_HOME = 'https://api.themoviedb.org/3/discover/movie?api_key=YOUR_API_KEY& \

                   include_adult=false&include_video=false&language=en-US&page=1&sort_by=popularity.desc';


const main =  document.getElementById("main");

const form = document.getElementById("form");

const search = document.getElementById("search");


async function getMovies(url) {


   const apiRes = await fetch(url);

   const resJson = await apiRes.json();


   if (!apiRes.ok) {

       console.error("Error fetching movies: ", apiRes.statusText);

   }

   showMovies(resJson.results);

}



function showMovies(movies) {

   main.innerHTML = ''


   movies.forEach((movie) => {

       const { title, poster_path, vote_average, overview } = movie


       const movieEl = document.createElement('div')

       movieEl.classList.add('movie')


       movieEl.innerHTML = `

           <img src="${IMG_PATH + poster_path}" alt="${title}">

           <div class="movie-info">

         <h3>${title}</h3>

         <span class="${getClassByRate(vote_average)}">${vote_average}</span>

           </div>

           <div class="overview">

         <h3>Overview</h3>
```

```
        ${overview}

      </div>

      `

      main.appendChild(movieEl)

  })

}


function getClassByRate(vote) {

  if (vote >= 8) {

      return ".movie-info blue";

  } else if (vote >= 5) {

      return ".movie-info black"

  } else return ".movie-info orange";

}


// Get initial movies

getMovies(API_URL_HOME);


form.addEventListener('submit', (e) => {

  e.preventDefault()


  const searchTerm = search.value // we create a var with the search term


  if(searchTerm && searchTerm !== '') { // and if the term exists

      getMovies(SEARCH_API + searchTerm);


      search.value = ''

  } else {

      window.location.reload();

  }

})
```

第 9 周：网页抓取

7 网页抓取

7.1. 爬行

- 阅读 *man wget* 以了解 *-i --force-html* 和 *--spider* 选项的作用。下载此网页的副本（您当前正在阅读的网页）并使用 *wget* 测试页面上的所有链接。是否有任何断开的链接？

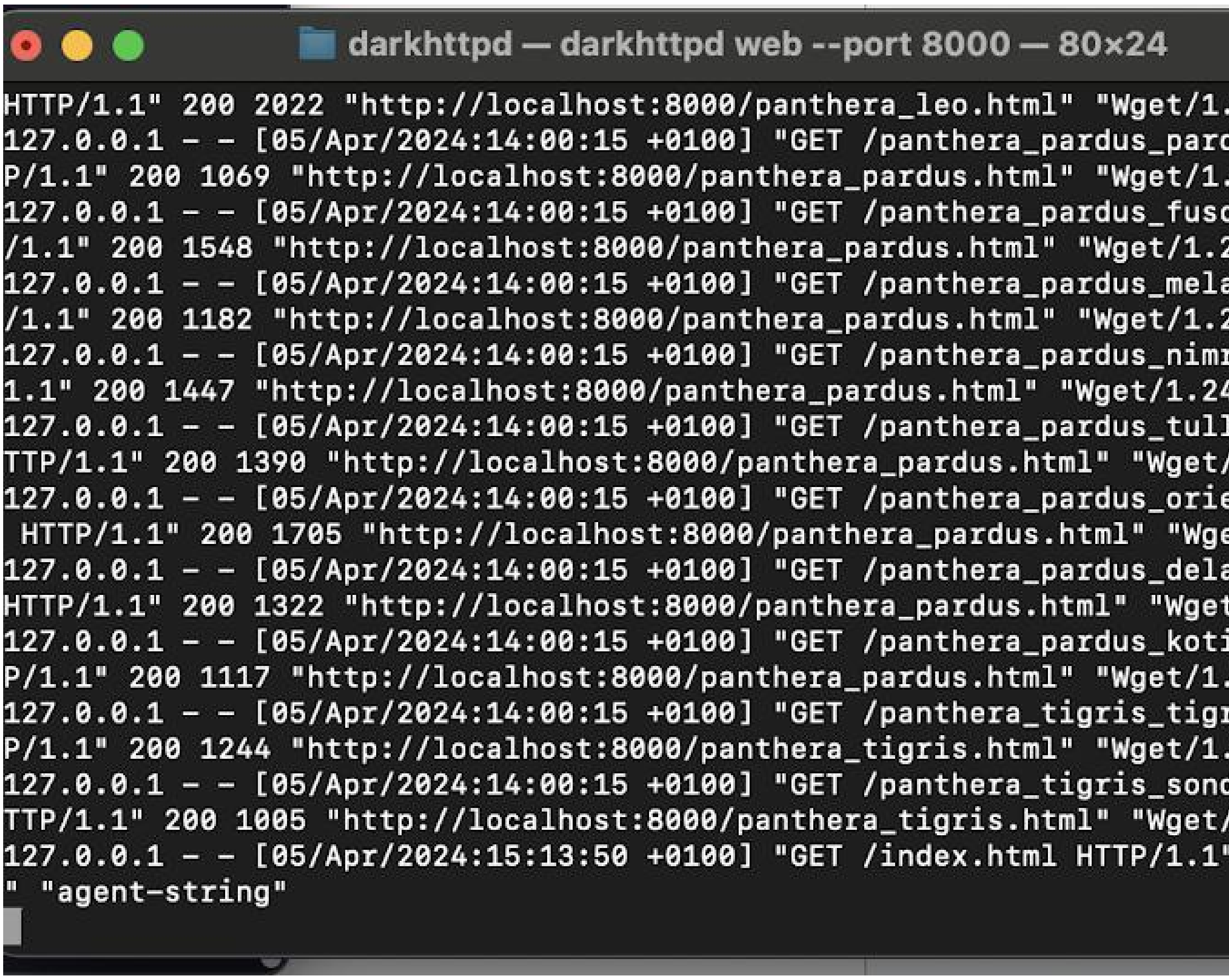
wget --spider -r -l 1

--force-html [https://cs-uob.github.io/COMSM0085/exercises/part2/scrape/crawl.html[g]](https://cs-uob.github.io/COMSM0085/exercises/part2/scrape/crawl.html)

```
-i file
     --input-file=file
         Read URLs from a local or external file.  If - is specified as
         file, URLs are read from the standard input.  (Use ./- to read
         from a file literally named -.)
         If this function is used, no URLs need be present on the command
         line.  If there are URLs both on the command line and in an input
         file, those on the command lines will be the first ones to be
         retrieved.  If --force-html is not specified, then file should
         consist of a series of URLs, one per line.
         However, if you specify --force-html, the document will be
         regarded as html.  In that case you may have problems with
         relative links, which you can solve either by adding "<base
         href="url">" to the documents or by specifying --base=url on the
         command line.
         If the file is an external one, the document will be
         automatically treated as html if the Content-Type matches
         text/html.  Furthermore, the file's location will be implicitly
         used as base href if none was specified.
     --force-html
         When input is read from a file, force it to be treated as an HTML
         file.  This enables you to retrieve relative links from existing
         HTML files on your local disk, by adding "<base href="url">" to
         HTML, or using the --base command-line option.
     --spider
         When invoked with this option, Wget will behave as a Web spider,
         which means that it will not download the pages, just check that
         they are there.  For example, you can use Wget to check your
         bookmarks:
                 wget --spider --force-html -i bookmarks.html
         This feature needs much more work for Wget to get close to the
         functionality of real web spiders
```

· Tell wget to use a different user agent string in a request to your server running on localhost. Check what the request looks like to your server.

```
wget  --user-agent=agent-string localhost:8000/index.html
```



```
darkhttpd — darkhttpd web --port 8000 — 80×24
HTTP/1.1" 200 2022 "http://localhost:8000/panthera_leo.html" "Wget/1.
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_pardus_parc
P/1.1" 200 1069 "http://localhost:8000/panthera_pardus.html" "Wget/1.
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_pardus_fusc
/1.1" 200 1548 "http://localhost:8000/panthera_pardus.html" "Wget/1.2
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_pardus_mela
/1.1" 200 1182 "http://localhost:8000/panthera_pardus.html" "Wget/1.2
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_pardus_nimr
1.1" 200 1447 "http://localhost:8000/panthera_pardus.html" "Wget/1.24
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_pardus_tull
TTP/1.1" 200 1390 "http://localhost:8000/panthera_pardus.html" "Wget/
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_pardus_orie
 HTTP/1.1" 200 1705 "http://localhost:8000/panthera_pardus.html" "Wge
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_pardus_dela
HTTP/1.1" 200 1322 "http://localhost:8000/panthera_pardus.html" "Wget
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_pardus_koti
P/1.1" 200 1117 "http://localhost:8000/panthera_pardus.html" "Wget/1.
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_tigris_tigr
P/1.1" 200 1244 "http://localhost:8000/panthera_tigris.html" "Wget/1.
127.0.0.1 - - [05/Apr/2024:14:00:15 +0100] "GET /panthera_tigris_sonc
TTP/1.1" 200 1005 "http://localhost:8000/panthera_tigris.html" "Wget/
127.0.0.1 - - [05/Apr/2024:15:13:50 +0100] "GET /index.html HTTP/1.1"
" "agent-string"
```

· wget -r -l 1 http://example.com 与 wget -p http://example.com 有何不同？（提示：考虑外部资源）。

wget -r -l 1 http://example.com 告诉 wget 递归下载网站，但深度为 1。这意味着 wget 将下载给定 URL 的页面以及从它直接链接的任何页面，但不会进一步下载。

wget -p http://example.com 指示 wget 在指定的 URL 下载网页，以及正确显示页面所需的所有文件，例如 CSS、JavaScript 和图像。此选项用于下载单个页面及其离线查看要求。

主要区别在于 -r -l 1 用于具有指定深度的递归下载，其中可以包含同一域上的页面，而 -p 专门用于下载正确呈现单个页面所需的所有组件，包括外部资源，如托管在不同域上的样式表和图像。

. 在 wget 手册页中查找"递归接受/拒绝选项"。您将如何获得 wget 来抓取来自多个不同域的页面？

-H

--span-hosts

在执行递归检索时启用跨主机。

. 查找 -nc 的作用。什么是喧嚣，你为什么要或不想这样做？

-nc 或 --no-clobber。

默认情况下，wget 使用新名称（example.html.1 等）重复下载。如果没有 clobber，wget 将拒绝下载该文件的新版本。

Clobbering 是在重复下载时被覆盖的东西。

-nc

--no-clobber

If a file is downloaded more than once in the same directory, Wget's behavior depends on a few options, including **-nc**. *In certain cases, the local file will be clobbered, or overwritten, upon repeated download. In other cases it will be preserved. When running Wget without* **-N**, **-nc**, **-r**, *or* **-p**, *downloading the same file in the same directory will result in the original copy of file being preserved and the second copy being named file*.**1**. *If that file is downloaded yet again, the third copy will be named file*.**2**, *and so on. (This is also the behavior with* **-nd**, *even if* **-r** *or* **-p** *are in effect.) When* **-nc** *is specified, this behavior is suppressed, and Wget will refuse to download newer copies of file. Therefore, ""no-clobber"" is actually a misnomer in this mode---it's not clobbering that's prevented (as the numeric suffixes were already preventing clobbering), but rather the multiple version saving that's prevented.*

When running Wget with ***-r*** *or* ***-p****, but without* ***-N****,* ***-nd****, or* ***-nc****, re-downloading a file will result in the new copy simply overwriting the old. Adding* ***-nc*** *will prevent this behavior, instead causing the original version to be preserved and any newer copies on the server to be ignored.*
When running Wget with ***-N****, with or without* ***-r*** *or* ***-p****, the decision as to whether or not to download a newer copy of a file depends on the local and remote timestamp and size of the file.* ***-nc*** *may not be specified at the same time as* ***-N****.*
A combination with ***-O/--output-document*** *is only accepted if the given output file does not exist.*
Note that when ***-nc*** *is specified, files with the suffixes* ***.html*** *or* ***.htm*** *will be loaded from the local disk and parsed as if they had been retrieved from the Web.*

7.2. 美汤

看看 BeautifulSoup 文档中的其他一些示例，特别是关于使用 .find_all（） 方法的示例。

使用您的解释器访问网页中所有<强>元素的列表，并弄清楚如何打印出其中包含的文本。

>>> 表示 soup.find_all（'strong'） 中的行：
...打印（line.get_text（））

如何使用 .find_all（） 查找具有特定类值（例如"container"）的所有 <div> 元素？

如果 div 有多个类，您的方法会起作用吗？

soup.find_all（'div'， {'class'： '容器'}）
不，对于多个类，我们应该使用类似
soup.select（'div.container'）

```
from bs4 import BeautifulSoup
import os


for file in os.listdir('cattax'):
  if file[-4:] == 'html':
    soup = BeautifulSoup(open('cattax/'+file,'r'), features='html.parser')
    print(soup.title.text + " : " + soup.h1.text)
```

修改 *scrape.py*，使其也打印出每页中“*info*”段落的内容（这可以是第二个打印语句）。再次运行脚本以测试它是否正常工作。

```
from bs4 import BeautifulSoup
import os


[h]for file in os.listdir('cattax'):
  if file[-4:] == 'html':
    soup = BeautifulSoup(open('cattax/'+file,'r'), features='html.parser')
    print(soup.title.text + " : " + soup.h1.text)
    info_paragraph = soup.find('p', class_='info')
    if info_paragraph:
        print(info_paragraph.text)

```

目前，该脚本为每一页打印一些内容。修改它，使其只为叶节点打印一些东西 - 那些没有自己的“容器”元素的页面。

```
from bs4 import BeautifulSoup
import os


for file in os.listdir（'cattax'）:
if file[-4: ] == 'html':
soup = BeautifulSoup（open（'cattax/'+file， 'r'）， features='html.parser'）
if（soup.find（'div'， class_='container'））:
continue[i]
print（soup.title.text + “ :  ” + soup.h1.text）
info_paragraph = soup.find（'p'， class_='info'）
如果 info_paragraph:
print（info_paragraph.text）

```

打印出来可能很有用，但通常我们希望存储我们抓取的值，以备以后的编程工作使用。不要打印出信息，而是为所有叶节点创建和更新 *Python* 字典，其中字典键是页面的标题，值是“*info*”框的相应内容。使用 *python3 -i scrape.py* 运行您的脚本，它将执行您的脚本，然后在脚本执行后立即将您置于交互式会话中。然后，您可以通过与解释器中的 *dict* 对象进行交互来检查字典的内容是否符合您的期望。

~

```
from bs4 import BeautifulSoup
import os

d = dict()
for file in os.listdir('cattax'):
  if file[-4:] == 'html':
    soup = BeautifulSoup(open('cattax/'+file,'r'), features='html.parser')
    if(soup.find('div', class_='container')):
            continue
    info_paragraph = soup.find('p', class_='info')
    if info_paragraph:
        d.update({soup.title.text:info_paragraph})

```

第 10 周：实用加密

8 实用加密

8.1. 开放 SSL

未加密的文件和加密的文件有什么区别？尝试读取加密文件。
文件大小较大，并且由于字符奇怪，文本无法阅读

Salted_ G 9Cw CO 3 >? ： &

-rw-r--r-- 1 流浪者 流浪者 32 Apr 28 10：16 mytext.enc

-rw-r--r-- 1 流浪者 流浪者 6 Apr 28 10：16 mytext.txt

或者，您可以使用 base64 对加密文件进行编码。

openssl base64 -in mytext.txt -out mytext.encxt.enc -a

（-a 标志代表 ASCII）。

-rw-r--r-- 1 流浪者 流浪者 32 Apr 28 10：16 mytext.enc

-rw-r--r-- 1 流浪者流浪者 9 Apr 28 10：20 mytextbase64.enc

-rw-r--r-- 1 流浪者 流浪者 6 Apr 28 10：16 mytext.txt

这个新文件只有 9 个字节，如下所示：

aGVsbG8K

文件大小是否比以前大？如果是这样，为什么？

因为它已经加密 - 并且包含比以前更多的字母。（用盐散列。

利用 -d 参数解密文件，无论是二进制流还是仅由 ASCII 字符组成

```
openssl enc -base64 -d -in mytext.encxt.enc -out mytext.decoded
```

使用 openssl 手册页，了解如何使用 genrsa 生成 1024 位公钥-私钥对。

```
man openssl-genrsa

openssl genrsa -out private_key.pem  -numbits 1024
```

这将生成一个私钥 – 必须提取公钥。

公钥旨在与他人共享。使用 OpenSSL 工具使用 -pubout 参数将公钥提取到 .pem 文件中。

```
openssl rsa -in private_key.pem -pubout -out public_key.pem
```

接下来，使用 OpenSSL -encrypt 命令创建加密消息。

```
openssl rsautl -encrypt -inkey bob_public.pem -pubin -in top_secret.txt
-out top_secret.enc
```

[k]

关键是您可以使用其他人的公钥对消息进行加密。

然后，我们生成 RSA 私钥（记下密码）。将 OpenSSL 与 genrsa 命令和 -aes256 参数一起用于大小为 2048 的私钥。

```
openssl genrsa -aes256 -out private_key.pem
```

接下来，使用 req 命令和 -x509 参数生成具有 OpenSSL 的根证书（用于数字签名），-sha256 用于哈希函数。最后，使其有效期为 30 天。

```
man openssl-req
openssl req -x509 -sha256 -days 30 -out rootCert.pem
```

从技术上讲，我们不需要 30d 参数，因为它是默认参数。

OpenSSL 工具包含许多实用程序。我们可以使用其中之一来验证与服务器的安全连接。

为此，我们将使用 s_client 实用程序来建立客户端到服务器的通信。因此，我们可以确定端口是否打开，服务器是否配置为支持 SSL，以及证书何时过期。

```
Man open SSL-s_client
```

在终端中，对具有 HTTPS 的 Web 服务器使用 s_client。

```
openssl s_client -connect google.com:443
```

443 是默认的 SSL 端口

https://docs.pingidentity.com/r/en-us/solution-guides/htg_use_openssl_to_test_ssl_connectivity

与其他工具（如 netcat 或 telnet）相比有什么区别？

其他工具不是为 ssl 设计的

尝试使用 -crlf 和 -brief 标志再次运行该命令。有什么区别？

-短

仅提供连接参数的简要摘要，而不是正常的详细输出。

-CRLF

此选项根据某些服务器的要求将来自终端的换行转换为 CR+LF。

8.2. PGP 的

gpg --encrypt --sign --armor -r other_person@mail.com file_name

此命令对消息进行加密，并使用您的私钥对其进行签名，以确保消息来自您。请记住，此消息将使用收件人的公钥进行加密，使您无法读取（除非您以某种方式获得收件人的私钥！
提示：您可以修改上述命令，使用私钥生成只有您才能阅读的消息。想想怎么做。

```
gpg --encrypt --sign --armor -r your_own_email@mail.com file_name
```

- 使用带有 --search 参数的 gpg 命令，找出每个键的关联电子邮件地址或用户 ID（用户名）。

```
gpg --keyserver keyserver.ubuntu.com --search-key 413109AF27CBFBF9
```

- 如果您决定将密钥导入密钥环，请使用带有 --recv-keys 参数的 gpg 命令。

```
gpg --keyserver keyserver.ubuntu.com --recv-keys ABCDEF0123456789
```

- 使用带有 --fingerprint 选项的 gpg 命令来验证密钥的真实性。

```
gpg --keyserver keyserver.ubuntu.com --fingerprint ABCDEF0123456789
```

- 现在，您可以尝试根据电子邮件地址进行搜索吗？

gpg --搜索 bogwonch@bogwonch.net

- 你能找到约瑟夫的旧过期钥匙吗？

```
gpg --keyserver keyserver.ubuntu.com --search bogwonch@bogwonch.net
```

Vagrantfile

updated with the stuff for the last lecture, otherwise the same.

```
Vagrant.configure("2") do |config|
  config.vm.box = "generic/debian12"
  config.vm.synced_folder ".", "/vagrant"
  #config.vm.network "forwarded_port", guest: 8080, host: 8080
  config.vm.provision "shell", inline: <<-SHELL
    apt-get update -y
    apt-get install -y git git-man apt-file
    apt install -y gdb
    apt install -y maven
    apt install -y openjdk-17-jdk
    apt install -y unzip
    apt install -y maven
    apt install -y openjdk-17-jdk
    apt install -y mariadb-{server,client}
    apt-get install -y libreadline-dev
    apt install -y shellcheck
    apt install -y openssl
    apt install -y nginx
    systemctl start mariadb
    apt-get install gnupg
    systemctl enable mariadb
    mysqladmin -u root -p create mydatabase
    mysql -u root -p -e 'source /vagrant/sample-data.sql'
SHELL
config.vm.provision :shell, privileged: false, inline: <<-SHELL
    git config --global user.name "ali"
    git config --global user.email "jardine64@gmail.com"
    SHELL
  config.vm.provision "file", source: "~/.vimrc", destination: "~/.vimrc"
end
```