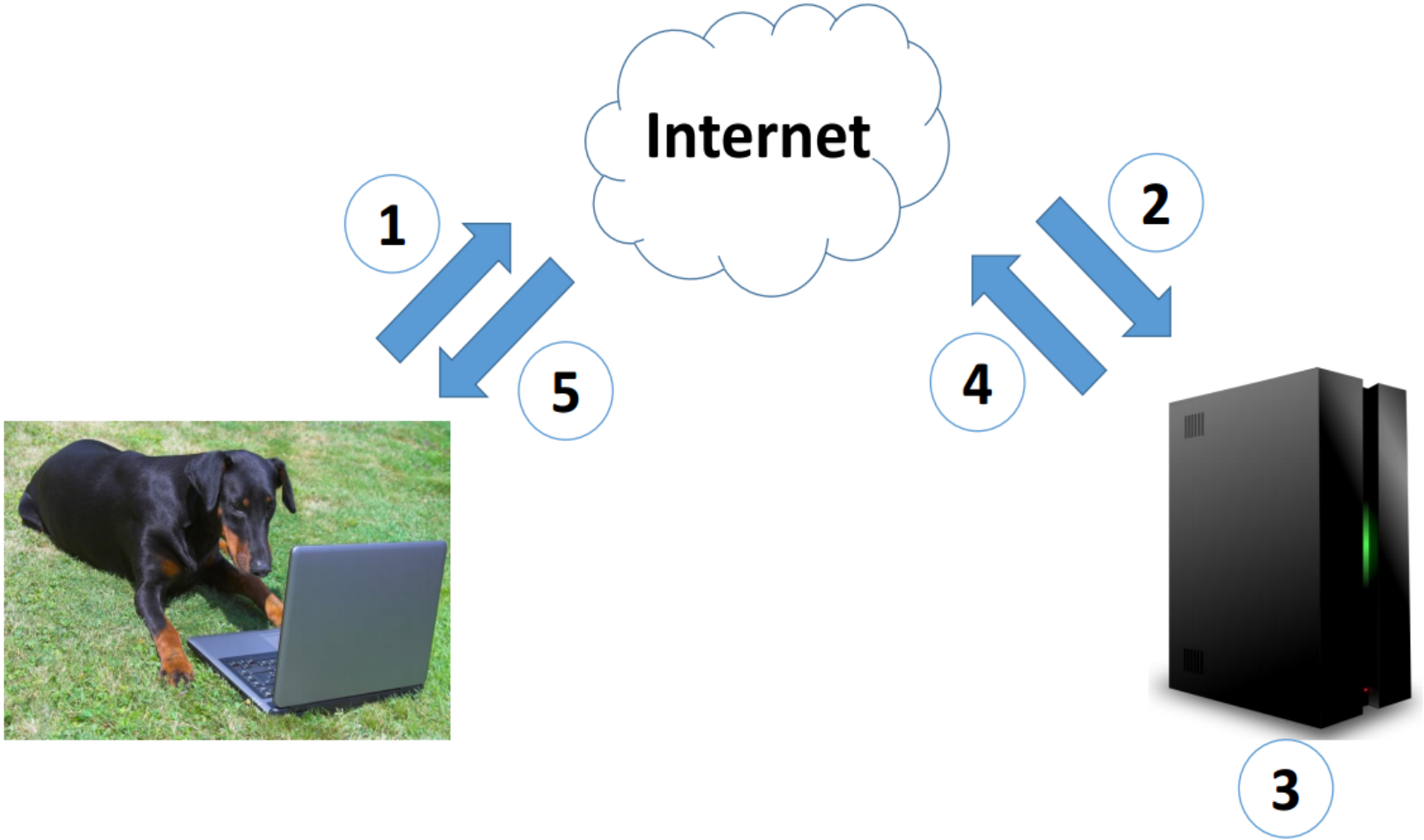
Scrapping & Crawling

PYMTY Febrero 2015

¿Cómo funciona la web?


Internet
1
2
3
4
5

¿Cómo funciona mi navegador?

> Implicando que no usas IE

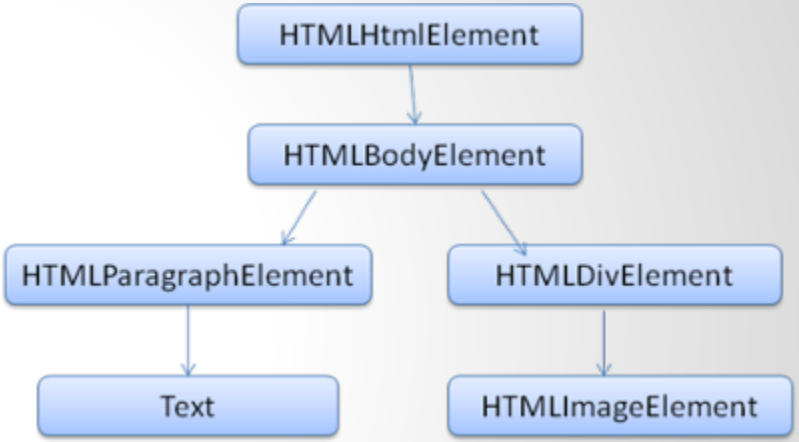
¿Cómo funciona el navegador?

- HTML
- CSS
- JS

HTML como árbol

HTML Como árbol

```
<html>
  <body>
    <p>
      Hello World
    </p>
    <div> <img src="example.png"/></div>
  </body>
</html>
```



Selectores

```
<html>
  <body>
    <div class="err" id="div1">
      <p>
        this is a <span class="big"> big error </span>
        this is also a
        <span class="big"> very  big  error</span> error
      </p>
    </div>
    <div class="err" id="div2">another error</div>
  </body>
</html>
```

```
1.  div {margin:5px;color:black}
2.  .err {color:red}
3.  .big {margin-top:3px}
4.  div span {margin-bottom:4px}
5.  #div1 {color:blue}
6.  #div2 {color:green}
```

XPath

```
<html>
<body>
  <div class="product-list">
    <div class="product">
      <img class="image" src="usb-wireless.jpg"></img>
      <span class="name"></span>
      <span class="sku">1</span>
      <span class="price">300</span>
    </div>
    <div class="product">
      <img class="image" src="gpu.jpg"></img>
      <span class="name">GPU</span>
      <span class="sku">2</span>
      <span class="price">999</span>
    </div>
  </div>
</body>
</html>
```

```
/html/body/div
//div
//div[@class="text"]
//div[@id="1"]
//div/img
//div/span[2]
```

Peticiones HTTP

- La manera síncrona
- La manera asíncrona (threads)
- La manera asíncrona (no IO blocking)

La manera síncrona

```
import requests

urls = [
        'http://example.com/1/',
        'http://example.com/2/',
        'http://example.com/3/',
]

responses = []
for url in urls:
    responses.append(requests.get(url))
```

La manera asíncrona (threads)

```
import requests
import threading

urls = [
        'http://example.com/1/',
        'http://example.com/2/',
        'http://example.com/3/',
]
responses = []

def worker(url):
    responses.append(requests.get(url))

threads = []
for url in urls:
    t = threading.Thread(target=worker, args=(url,))
    threads.append(t)
    t.start()

for t in threads:
    t.join()
```

La manera asíncrona (sin IO block)

```
from requests import async

urls = [
        'http://example.com/1/',
        'http://example.com/2/',
        'http://example.com/3/',
]

responses = []
def do_something(response):
    responses.append(response)

async_list = []

for url in urls:
    action_item = async.get(url)
    async_list.append(action_item, hooks = {'response' : do_something})

async.map(async_list)
```

Comparativa

Bloqueante

Threads

No bloqueante

Antes de hacer crawling o scrapping

- Asegúrate que la información que necesites no esté en un API
- Asegúrate que sea legal
- Asegúrate de obtener algún tipo de id para tu información

Crawling

1. Obtener html
2. Obtener urls de interés
3. Crawling a esas urls

Scrapping

1. Obtener html
2. Extraer datos
3. Almacenarlos en algún lugar

¿Porqué usar un framework?

- Hay muchos problemas entre cada uno de esos pasos:
  - Problemas de encoding
  - Problemas con las urls
  - Problemas con el html
  - Mantener un record de urls visitadas
  - Hacer peticiones asíncronas de la manera correcta
  - Hooks en todos lados

Referencias

- http://www.html5rocks.com/es/tutorials/internals/howbrowserswork/