# Memòria pràctica 1

**DAT QT2019**

Alba Mendez

12 d'octubre de 2019

# Part I.
# Introducció

En aquesta primera pràctica es fa una introducció al protocol HTTP, els tipus de continguts que es poden servir, el funcionament bàsic d'un servidor web (en aquest cas Apache), i el funcionament d'un programa CGI.

**Exercici 1.** Es demana que fem una pàgina simple i la fem accessible a la arrel del nostre espai web personal, que en el meu cas és `http://soft0.upc.edu/~ldatusr10/`.

En primer lloc creem la carpeta `public_html` dins la nostra home, i el fitxer `index.html` a dins. Per afegir varietat, també he creat una fulla d'estils (`styles.css`) i una imatge de fons dins la mateixa carpeta.

L'HTML és el següent:

```
<!doctype html>
<html>
<head>
  <meta charset='UTF-8'>
  <meta name="robots" content="noindex">
  <title>Main</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="stylesheet" href="assets/style.css">
</head>
<body>
```

```
  <div class="content">
    <h1>Hola.</h1>
    <p>Em dic Alba.</p>
  </div>

</body>
</html>
```

Els estils no s'inclouen per brevetat. Accedim a la URL i comprovem que funciona correctament.

**Exercici 2.** Ara es demana que afegim un enllaç a la pàgina nova `http://soft0.upc.edu/~ldatusr10/practica1/`.

Per fer-ho, afegim l'HTML següent a la pàgina anterior, després del paràgraf:

```
<a class="practica" href="practica1/">Pràctica 1</a>
```

I creem una nova carpeta `practica1` a `public_html`, i dins un fitxer `index.html` amb un HTML similar:

```
<!doctype html>
<html>
<head>
  <meta charset='UTF-8'>
  <meta name="robots" content="noindex">
  <title>Pràctica 1</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="../assets/style.css">
</head>
<body>

  <div class="content">
    <h1>Pràctica 1</h1>
  </div>

</body>
</html>
```

**Exercici 3.** Ara es demana que afegim un enllaç a la pàgina nova `http://soft0.upc.edu/~ldatusr10/practica1/exemple.html`, i li canviem el sufix temporalment a `txt` per veure els efectes.

De forma similar a l'exercici anterior, afegim l'enllaç a la pàgina:

```
<a href="exemple.html">Exemple</a>
```

I llavors creem el fitxer `exemple.html` al costat de l'anterior, amb un HTML similar (no es reprodueix per brevetat).

Ara, si li canviem el sufix (extensió) com s'indica, veiem que el navegador mostra el text literalment (sense interpretar-lo com a HTML) i en una font monoespaiada.

Aquesta decisió la pren en realitat el *servidor*. Els servidors estàtics acostumen a fer servir l'extensió del fitxer per determinar el tipus MIME de la resposta que s'envia al navegador (header `Content-Type`). Abans el navegador rebia una resposta amb tipus `text/html` i ara és `text/plain`.

**Exercici 4.** Ara hem de reanomenar el directori `practica1` i observar els efectes.

En fer-ho, veiem que es trenca l'enllaç que hem afegit a l'exercici 2, i hem de corregir-lo amb el nou nom de la carpeta.

Això és perquè hem fet servir URLs *relatives*; en canvi, si haguessin estat *absolutes*, hauriem hagut de corregir també els enllaços entre fitxers de les carpetes, com el que hem creat a l'exercici 3.

**Exercici 5.** Ara hem de crear un CGI (el codi es dona ja fet) que presenta el dia i hora del servidor, i enllaçar-lo des de `practica1`.

Creem el fitxer a `hora.cgi` dins la carpeta `practica1`, hi posem el codi que es dona, i li donem permisos d'execució amb `chmod +x hora.cgi`. Llavors només queda afegir un enllaç des de `index.html`:

```
<a href="hora.cgi">CGI dia i hora</a>
```

Comprovem que el CGI i l'enllaç funcionen correctament.

**Exercici 6.** Es demana que canviem el codi del CGI perquè retorni tipus MIME de la resposta `text/plain`.

Si ho fem, passa el mateix que en l'exercici 3. En aquest cas però, no hi ha gaire diferència ja que la resposta no conté elements HTML o caràcters especials.

# Part II.
# Experimentant amb HTTP

En aquesta part es demana que obtenim un document qualsevol mitjançant una petició GET amb telnet. Com a exemple, hem triat l'URL `http://upc.edu/foo`. Executem:

```
telnet upc.edu http
```

I escrivim la petició GET:

```
GET /foo HTTP/1.1          L'URL acostuma a ser només un path (origin-form)
Host: upc.edu              La capçalera Host és obligatòria en HTTP 1.1.
                           Línia en blanc, ja que la petició no té cos.
```

Un cop l'hem acabat d'escriure, obtenim la resposta següent del servidor:

```
HTTP/1.1 302 Found                El codi d'estat 302 és una redirecció temporal.
Location: https://upc.edu/foo     Indica l'URL cap al que se'ns redirigeix.
Connection: Keep-Alive            Indica com es gestiona la connexió.
Content-Length: 0                 Indica la longitud del cos en bytes (buit).
                                  Línia en blanc, ja que la resposta no té cos.
```

Ara se'ns demana fer una petició POST a la URL `http://soft0.upc.edu/web/cgi/exemples/test-http.cgi`:

```
telnet soft0.upc.edu http

POST /web/cgi/exemples/test-http.cgi HTTP/1.1
Host: soft0.upc.edu

HTTP/1.1 200 OK
Date: Wed, 23 Oct 2019 10:31:29 GMT
Server: Apache
Vary: Accept-encoding
Transfer-Encoding: chunked
Content-Type: text/html

[HTML de la resposta]
```

Les diferències entre els dos mètodes són *semàntiques*: Les peticions GET no haurien de tenir cos (i si el tenen, s'hauria d'ignorar), les peticions POST acostumen a tenir-ne. Una petició GET no hauria de tenir efectes secundaris, i és *idempotent*; mentre que una POST normalment sí tindrà efectes secundaris i no necessàriament és idempotent.

# Part III.
# Experimentant amb els CGIs

En aquesta part farem un CGI simple que actuarà de comptador de visites. En aquest cas hem triat escriure el CGI fent servir Python. Es crea el fitxer `visits.cgi` dins de `practica1` amb el següent contingut:

```
#!/usr/bin/env python3
import fcntl

with open('../../visits.txt', 'r+') as f:
  fcntl.flock(f, fcntl.LOCK_EX)
  visits = int(f.read()) + 1    # Read visit count, increment
  f.seek(0)
  f.write(str(visits))          # Save visit count

# Write response!
print('Content-Type: text/plain')
print('Cache-Control: no-cache')
print()
print(visits)
```

**Nota:** La crida a `flock` garanteix que la operació és atòmica; sinó podria haver-hi corrupció de dades o visites no comptades, en situacions d'alt tràfic.

S'inicialitza el fitxer on es desa el número de visites (`echo 0 > ~/visits.txt`), se li donen permisos d'execució com hem fet abans, i comprovem que funciona correctament.

Ara es demana que incrustem el contingut d'aquest CGI a `practica1/index.html`; per a fer-ho inclourem aquest script abans del `</body>`:

```
<script>
  fetch('visits.cgi').then(x => x.text()).then(x =>
    document.querySelector('.visits .counter').innerText = x)
</script>
```

Ja només queda afegir en algun lloc de la pàgina, el paràgraf on es mostrarà el nombre de visites:

```
<p class="visits">Visitant número: <span class="counter"></span></p>
```

Es pot veure el resultat final a `http://soft0.upc.edu/~ldatusr10/practica1/`.