

Lab 6

[Obligatorisk] HTTP Request

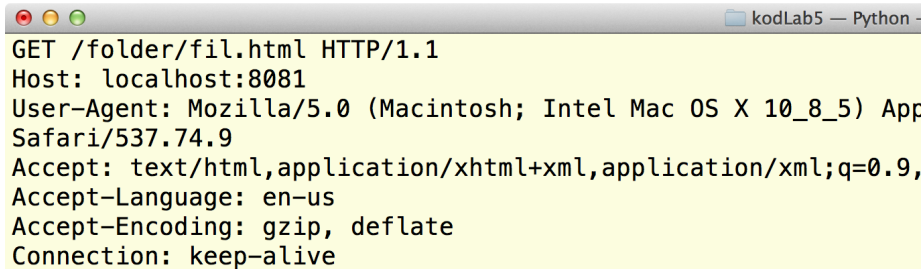
Syftet med denna laboration är dels att öva på socketprogrammering och dessutom att undersöka http-protokollet.

I laborationen skall man skriva en liten http-server som returnerar en hemsida som visar hur klientens *request* såg ut. Vi delar upp uppgiften i två delar, den andra delen är fortsättning på den första.

6.1 Deluppgift 1

I denna deluppgift skall du skriva en ‘webbserver’ som skriver ut klientens s.k. *request* på skärmen.

1. Skriv en TCP-server med följande uppgift: Så fort en klient kopplar upp en förbindelse och skickar text till servern skall servern skriva ut denna text på skärmen och sedan koppla ned. Välj gärna port 80 om det går, och annars ett annat portnummer. (Tips: din server tar emot en s.k. *byte-array*. För att göra om den till en sträng kan du anropa metoden `decode("ASCII")`.)
2. Öppna sedan en webbläsare och surfa till *localhost*, och det aktuella portnumret. Webbläsaren skickar nu en request till ditt serverprogram, och om det fungerar korrekt skriver det ut denna request ut på skärmen. Nedanstående skrämdump visar hur utskriften kan se ut om servern lyssnar på port 8081 och om man surfat till *http://localhost:8081/folder/fil.html*. Det är viktigt att utskriften blir *snygg* med en request-rubrik per rad!



```
GET /folder/fil.html HTTP/1.1
Host: localhost:8081
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_5) AppleWebKit/537.74.9
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

6.2 Deluppgift 2

Istället för att skriva ut klientens request på skärmen skall servern nu skicka tillbaka en html-sidan som beskriver denna request. Nedanstående Bild visar hur html-sidan skall se ut i webbläsaren.



Tips:

1. Serverns s.k. response ska inledas med raden `HTTP/1.1 200 ok`. Därefter borde det egentligen komma flera response-headers, och sedan en *tom* rad, och sedan *html-dokumentet*. Nedanstående kodfragment visar hur jag själv har genererat de tre första raderna i serverns response.

```
sock.sendall(bytearray("HTTP/1.1 200 ok\n", "ASCII"))
sock.sendall(bytearray("\n", "ASCII"))
sock.sendall(bytearray("<html>\n", "ASCII"))
```

2. Använd html taggen `<pre>` för att bevara request-rubrikernas formatering.

6.3 Redovisning

Ladda upp din pythonkod för deluppgift 2 till pingpong (och redovisa sedan muntligt).