



# Webbprogrammering grunder

2D1522 Datorteknik och -kommunikation  
2D2051 Databasteknik och  
datorkommunikation  
<http://www.nada.kth.se/kurser/kth/2D1522/>  
<http://www.nada.kth.se/kurser/kth/2D2051/>

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

1



## Dagens föreläsning

- Syfte
  - Ge grundläggande förståelse för hur enkla system för dynamisk webpublicering är uppbyggda, för att i senare skeden kunna göra mer avancerade, praktiska tillämpningar.
- Mål
  - Förstå hur olika typer av data skickas över internet med http-protokollet.
  - Förstå hur program, såsom webbläsare, vet hur de ska hantera mottagna filer med MIME.
  - Veta hur parametrar kan skickas mellan webbläsare och webserver
  - Kunna göra enkla, dynamiska websidor med cgi-script och senare med PHP.

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

2



## Dagens innehåll

- Kort repetition av html
- Hur datatyp identifieras med MIME
- Hur data överförs med HTTP
- Hur dynamiskt webbinnehåll kan skapas med cgiscript
- Hur variabelvärden överförs med formulär/cgi

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

3



## Repetition xhtml

- xhtml är ett märkspråk för att beskriva websidors struktur.
- Består av ”tags” som märker upp olika delar av ett dokument.  
Viktiga delar är:
  - Metainformation och innehåll (<head></head>, <body></body>)
  - stycken, rubriker etc. (<p>stycke</p>, <h1>rubrik</h1>)
  - Länkar (<a href=’http://www.kth.se’>Till KTH</a>)
  - Bilder (<img src=’bilden.gif’/>)
- Hur innehållet sedan tolkas är upp till webbläsaren

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

4



## Exempel - XHTML

```
<html>
  <head>
    <title>En websida</title>
  </head>
  <body>
    <p>
      <a href="http://www.kth.se">KTH</a>
      
    </p>
  </body>
</html>
```



## Inline-resurser

- En html-sida kan innehålla två typer av element.
  - ”Vanliga” element (<p>Ett stycke</p>)
  - Inlineelement ()
- Inline-element innehåller länkar till andra resurser (t.ex. filer), som ska hämtas utan att en användare behöver interagera, och på något sätt hanteras av webläsaren.
- Vanligast är bilder, men även ljud, filmer och animationer kan användas.



## MIME

- Kort repetition av html
- Hur datatyp identifieras med MIME
- Hur data överförs med HTTP
- Hur dynamiskt webbinnehåll kan skapas med cgiscript
- Hur variabelvärden överförs med formulär/cgi




## Hantering av ”resurser”

- Bilder kan vara kodade på olika sätt (gif, jpg, png ...)
- Samma gäller ljud, video etc.
- Några format kan hanteras av webläsaren själv, några andra kräver ”plug-ins”, ytterligare andra hanteras av externa applikationer.



BTH Research Center  
Tech Research

- I inställningarna för webbläsare kan man bestämma hur en viss filtyp ska hanteras.
- Hanteras av en plugin... 



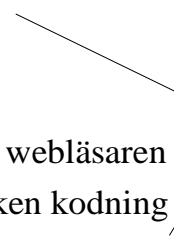
2006-04-20

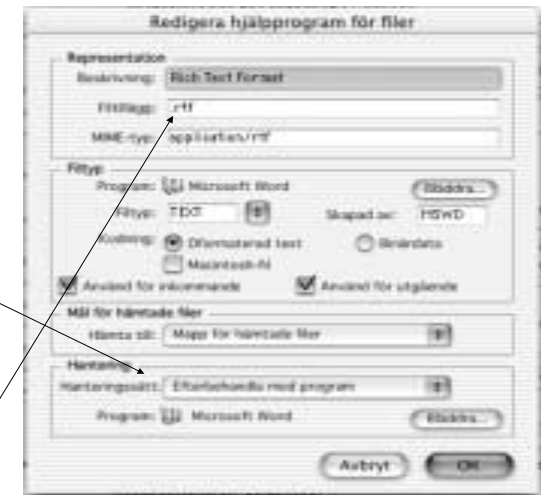
© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

9



BTH Research Center  
Tech Research

- ... eller av ett externt program 
- Ofta kan webbläsaren gissa vilken kodning en fil har baserat på filens ändelse.



2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

10



BTH Research Center  
Tech Research

- Ibland är dock "filen" skapad av ett program, t.ex. gulasidornas karttjänst. Då ger inte filändelsen någon information om filtypen



``

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

11



BTH Research Center  
Tech Research

## Flera filtyper med samma ändelse

- Antalet treställiga förkortningar är dessutom begränsat vilket kan leda till att en ändelse inte entydigt kan användas för att avgöra filtypen

### Exempel

.rpm - Red Hat package manager  
eller  
.rpm - Real Player Media

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

12



## Överlåt ansvaret på servern

- Den som publicerar en resurs på internet bör dock (rimligen) veta vilket filformat resursen har.
- Överlåt därför ansvaret att avgöra filformatet till *webservern* istället för *webklienten*.
- Låt sedan varje överföring av en resurs (fil) inte enbart innehålla resursen i sig, utan även filtypen.

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

13



## MIME-typer

- Den entydiga bestämningen av filformat görs med s.k. "MIME-typer" (Multi purpose Internet Mail Extensions)
- Anges på formen huvudgrupp/filformat t.ex video/mpeg, text/html eller image/gif



2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

14



## Hur känner webservern till vilken MIME-typ en fil har?

- I de flesta fall gör webservern en enkel tabelluppslagning av filändelsen för att bestämma MIME-typ
  - Filnamnet xxx.gif ger MIME-typen image/gif
- Detta gör att en viss webserver inte kan avgöra om en fil med ändelsen .rpm är "Real Player Media" eller "Red Hat Package Manager"
- På en Apache-webserver lagras informationen i filen mime.types

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

15



## mime.types



2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

16



## Överföringar med HTTP

- Kort repetition av html
- Hur datatyp identifieras med MIME
- Hur data överförs med HTTP
- Hur dynamiskt webbinnehåll kan skapas med cgi-script
- Hur variabelvärden överförs med formulär/cgi



## http

- Själva överföringen av resurser på webben sker med http - HyperText Transfer Protocol.
- Ett protokoll bland flera, t.ex. smtp, pop etc.
- Ni har testat smtp och http på laboration 1.

```
nada11:->telnet www.nada.kth.se 80
Trying 130.237.225.40...
Will send login name and/or authentication information.
Automatic encryption of output is enabled
Automatic decryption of input is enabled
Encryption is verbose
rlogin character is '-'.
Connected to w2.nada.kth.se.
Escape character is '^]'.
GET /-inge/gurka.txt HTTP/1.0
```

```
HTTP/1.1 200 OK
Date: Tue, 08 Apr 2003 13:15:39 GMT
Server: Apache/1.3.26 (Unix) mod_jk/1.1.0PHP/4.2.2
mod_perl/1.26 mod_ssl/2.8.10 OpenSSL/0.9.6d
Last-Modified: Tue, 18 Mar 2003 14:22:23 GMT
ETag: "de941cc-24-3e772b9f"
Accept-Ranges: bytes
Content-Length: 36
Connection: close
Content-Type: text/plain
```

```
Detta är en testfil för kurs 2D1522
Connection closed by foreign host.
nada11:->
```

Mime-typen

Själva resursen



## Vad är http

- Ett nätverksprotokoll för att överföra filer och annan data, generellt kallat *resurser*.
- Använder i stort sett alltid TCP/IP-sockets.
- En http-transaktion kräver en http-klient (browser) och en http-server (webserver).
- Använder normalt port 80, men kan använda vilken port som helst.



## Resurser

- En resurs behöver inte nödvändigtvis vara en fil, utan kan vara vad som helst som kan pekats ut av en URL (Uniform Resource Locator)
- Exempel: Filer, dynamiskt genererade bilder, resultat genererade efter att ha fyllt i ett formulär.



## http-transaktioner

- Klienten öppnar en förbindelse med servern.
  - Klienten skickar ett *request message* till servern
  - Servern skickar ett *response message*, oftast innehållande resursen
  - Servern stänger förbindelsen
- Request/response**
- En inledande rad
  - noll eller flera header-rader
  - en tom rad
  - eventuellt en meddelandekropp, som exempelvis själva resursen



## Inledande rad i en request

```
GET /~inge/gurka.txt HTTP/1.0
```

- GET är ”metodnamnet”. Andra vanliga metoder är POST och HEAD
- /~inge/gurka.txt är sökvägen till resursen
- HTTP/1.0 är versionen av HTTP som används



## Inledande rad i en response

```
HTTP/1.0 200 OK
```

- HTTP/1.0 är HTTP-versionen i svaret
- 200 är en statuskod
- OK är en klartextsträng som beskriver statuskoden.
- Vanliga varianter man ofta råkar ut för är

```
HTTP/1.0 404 Not Found
```

```
HTTP/1.0 500 Internal Server Error
```



## Statuskoder

- Statuskoder är tre-siffriga, där första siffran indikerar huvudklass på status
- 1xx är ett informativt meddelande
- 2xx visar att förfrågan lyckades
- 3xx omdirigerar förfrågan till en annan resurs
- 4xx betyder fel hos klienten
- 5xx betyder fel hos servern



## http-headers

- Både i http-requests och http-responses går det att lägga till diverse header-information.
- Exempel för request är
  - User-Agent: Mozilla/5.0
  - From: bjornh@kth.se
- Exempel för responses är
  - Server: Apache/1.3
  - Last-Modified: Thu, 02 May 2002 08:59:59 GMT



## MIME och HTTP

- Om en *response* innehåller en meddelandekropp, t.ex. själva innehållet i en fil, bör en header-rad som beskriver MIME-typen finnas med.
- Att rekommendera är även längden av meddelandekroppen ifall den är känd.
  - Content-Type: text/html
  - Content-Length: 36



## http via telnet

Öppna förbindelse till http-servern

Inledande request-rad

Statuskod

Replyheaders

Data

```
nada11:->telnet www.nada.kth.se 80
Trying 130.237.225.40...
Will send login name and/or authentication information.
Automatic encryption of output is enabled
Automatic decryption of input is enabled
Encryption is verbose
login character is '-'.
Connected to w2.nada.kth.se.
Escape character is '^]'.
GET /-inge/gurka.txt HTTP/1.0
HTTP/1.1 200 OK
Date: Tue, 08 Apr 2003 13:15:39 GMT
Server: Apache/1.3.26 (Unix) mod_jk/1.1.0PHP/4.2.2
mod_perl/1.26 mod_ssl/2.8.10 OpenSSL/0.9.6d
Last-Modified: Tue, 18 Mar 2003 14:22:23 GMT
ETag: "de941cc-24-3e772b9f"
Accept-Ranges: bytes
Content-Length: 36
Connection: close
Content-Type: text/plain
Detta är en testfil för kurs 2D1522
Connection closed by foreign host.
nada11:->
```

Request-header

Blankrad

Blankrad



## Spår av http-transaktioner

- Man väljer ofta att logga vissa delar av transaktionerna i loggfiler.
  - agent\_log
    - loggar vilken klient som använts (IE, Netscape etc)
  - access\_log
    - loggar vilka resurser som hämtats och statuskoder
  - error\_log
    - loggar eventuella fel
- Dessa kan bl.a. användas för statistik.



## Använda informationen för att påverka innehållet

- Informationen som skickas från klient till server kan användas för att påverka vilket innehåll som ska skickas tillbaka till klienten.
  - versiontracker.com och download.com väljer startsida beroende på om klienten är en mac/pc/linux etc.
  - Språkspecifika sidor kan väljas genom att se vilket språk webbläsaren är inställd på
    - Om filen banan.html efterfrågas kommer i första hand filen banan.html.se returneras om webbläsaren är inställd på Svenska och filen banan.html.en returneras om den är inställd på engelska.



## Exempel på tillgänglig information

```

CGI Process Environment
-----
Variable      Value
-----
QUERY_STRING  /banan.html
HTTP_HOST     192.168.1.100
REMOTE_ADDR   192.168.1.100
HTTP_ACCEPT   */*
SERVER_PROTOC /HTTP/1.1
HTTP_ACCEPT_ENCODING   gzip;q=1.0,*/*;q=0.5
SERVER_PROTOCOL   HTTP/1.1
GATEWAY_INTERFACE   CGI/1.1
HTTP_HOST       192.168.1.100
SERVER_SOFTWARE Apache/1.3.37 (Ubuntu DDE-4.1.2)
SERVER_SIGNATURE   Apache/1.3.37 (Ubuntu DDE-4.1.2)
REMOTE_ADDR     192.168.1.100
QUERY_STRING    /banan.html
SERVER_NAME      192.168.1.100
DOCUMENT_ROOT   /usr/www/htdocs
SCRIPT_FILENAME /usr/www/htdocs/banan.html
SCRIPT_URL       /banan.html
SCRIPT_NAME      /banan.html
  
```



## Skillnader http/1.0 http/1.1

http version 1.1 används mest idag. Det finns framför allt två viktiga skillnader:

- En klient-header med namn "Host" krävs nu som anger hostnamnet på resursen man hämtar. Behövs för att kunna köra webhotell, med flera domäner på samma IP-adress.
- Keep-alive-connections tillåts, dvs förbindelsen behöver inte kopplas ner mellan varje hämtning av resurser. Detta används för att ladda ner alla delar (bilder etc.) av en webbsida med en uppkoppling.



## Sammanfattning av "vanlig" http-kommunikation

- Klienten öppnar en TCP/IP-socket mot webservern.
- Klienten skickar en begäran att få en resurs + olika frivilliga eller obligatoriska headers.
- Webservern hittar resursen, och bestämmer bl.a. dess längd och dess MIME-typ.
- Webservern skickar tillbaka en statuskod, olika reply-headers såsom MIME-typ samt själva resursen.
- Webservern stänger förbindelsen.





## Dynamiskt innehåll med cgi-script

- Kort repetition av html
- Hur datatyp identifieras med MIME
- Hur data överförs med HTTP
- Hur dynamiskt webbinnehåll kan skapas med cgiscript
- Hur variabelvärden överförs med formulär/cgi

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

33



## Dynamiskt innehåll

- Hittills har vi endast arbetat med ”statiska”, icke-föränderliga resurser såsom filer.
- Ofta måste dock resurserna skapas ”dynamiskt”, vid anropstillfället
  - Sökningar i sökmotorer
  - Shopping på webben

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

34



## Innehåll skapat av program

- html-filer är mycket enkelt uppbyggda
- Enkelt skriva ett program som genererar en html-fil.
- Sådana program kallas ofta för ”cgi-script”
- Kan skrivas i godtyckligt programmeringsspråk. Själva html-koden genereras med ”print-satser” eller motsvarande.

2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

35



## Enkelt exempel (shellscript)

```
media:~$ cat date.cgi
#!/bin/sh
echo "Content-type: text/html"
echo
echo "<html>"
echo "<head>"
echo "<title>Datatest</title>"
echo "</head>"
echo "<body>"
echo "<h1> date </h1>"
echo "</body>"
echo "</html>"
media:~$
```



2006-04-20

© Björn Hedin, Inge Frick, NADA/KTH 2002 - 2006

36



## MIME-typ och cgi

- Eftersom ett program kan generera alla tänkbara filformat (excel, html, gif-bilder), måste programmet självt ange vilken MIME-typ innehållet som genereras har.
- Programmet har ansvar för att generera hela http-headern, exklusive statussignalen men inklusive blankraden

```
media>cat date.cgi
#!/bin/sh
echo "Content-type: text/html"
echo
echo "<html>"
echo "<head>"
echo "<title>Datumst</title>"
echo "</head>"
echo "<body>"
echo "<h1>date</h1>"
echo "</body>"
echo "</html>"
media>
```

MIME-typ

Blankrad

Innehåll

Dynamiskt genererat



## Speciella krav för cgi

- Skillnaden mellan statiska resurser och cgi-script är alltså att scripten *exekveras* och *resultatet* skickas tillbaka, inte själva scriptet i sig.
- Oftast måste cgi-script ligga i speciella kataloger, pga säkerhetsaspekter. Oftas heter katalogerna /cgi-bin/
- Det går dock att koppla vissa ändelser så att t.ex. alla filer med ändelsen .cgi eller .pl exekveras, oavsett var de ligger.
- Viktigt att, på en unix-dator, aktivera exekveringsflaggan för "others" (>chmod 755 filnamn, eller motsvarande för AFS)



## Överföra variabelvärden

- Kort repetition av html
- Hur datatyp identifieras med MIME
- Hur data överförs med HTTP
- Hur dynamiskt webbinnehåll kan skapas med cgiscript
- Hur variabelvärden överförs med formulär/cgi



## Överföra variabler

- I exemplet ändras endast datumet, det finns inga andra möjligheter att påverka resultatet.
- Det vore önskvärt att kunna skicka med ett antal parametrar, som sedan kan användas för att styra resultatet, t.ex. söksträngar för sökmotorer.



## Formulär

- I html kan interaktion med användaren framför allt skapas med hjälp av formulär.
- Tillåter enkla parametrar att skickas
  - Av typen variabelnamn=variabelvärde
- Variablernas namn och värden kan sedan överföras till scriptet på serversidan, som kan anpassa sitt beteende därefter.

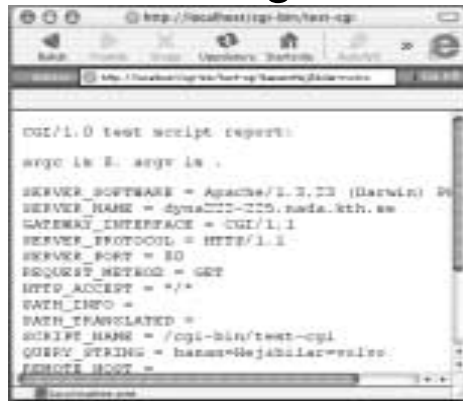


## Ett enkelt formulär

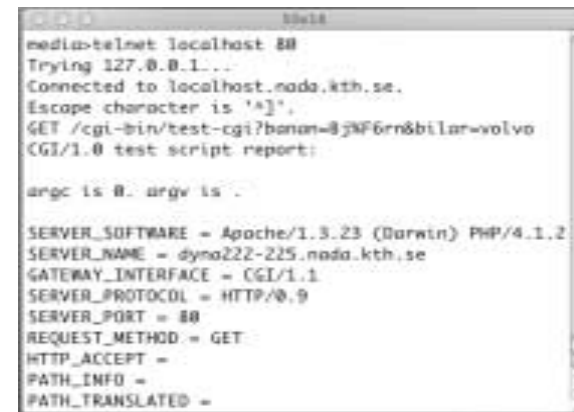


## GET - URL-encoding

- När metoden GET används kodas variablerna in i själva URLen.
- Värdena lagras i variabeln QUERY\_STRING



## Samma exempel via telnet





## URL-encoding

- Vissa steg måste dock vidtagas för att koda variabler-värden så de kan skickas via en URL.
  - Konvertera ”konstiga” , icke-alfanumeriska och icke-amerikanska tecken till %xx där xx är tecknets kod-värde skrivet hexadecimalt.
  - Ändra mellanslag till ”+”
  - bind ihop variabelnamn-variabelvärde med ”=”
  - Separera variabler med ”&”
  - Inled med ett ”?” direkt efter scriptnamnet
  - Exempel:
    - `http://localhost/cgi-bin/test-cgi?banan=Bj%F6rn&bilar=volvo`



## Metoden POST

- Det finns även en metod vid namn POST som liknar GET
- Variablerna skickas i ”body-delen” av förfrågan.
- Filer kan skickas med (t.ex. bifoga attachments i hotmail).
- En effekt blir att variabelnamn och variabelvärden inte syns i URLen, samt att variablerna inte längre lagras i environmentvariabeln QUERY\_STRING



## Samma exempel fast med POST

```

media:more form.html
<html>
<head>
<title>Ett formul&uml;r;</title>
</head>
<body>
<form action="/cgi-bin/test-cgi" method="POST">
<input name="banan" />
<select name="bilar">
<option value="volvo">Volvo</option>
<option value="saab">Saab</option>
</select>
<input type="submit" value="skicka"/>
</form>
</body>
</html>
media:
  
```

```

CGI/1.0 test script report:
-----
script is S. array is
SERVER_SOFTWARE = Apache/1.3.33 (Ubuntu) PHP/4.1.1
SERVER_NAME = localhost:80
SERVER_PORT = 80
REQUEST_METHOD = POST
HTTP_ACCEPT = */*
HTTP_HOST = localhost
HTTP_REFERER = /cgi-bin/test-cgi
QUERY_STRING =
REQUEST_URI = /cgi-bin/test-cgi
SCRIPT_NAME = /cgi-bin/test-cgi
CONTENT_LENGTH = 25
CONTENT_TYPE = application/x-www-form-urlencoded
  
```



## Samma exempel via telnet

```

media:telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.nada.kth.se.
Escape character is '^]'.
POST /cgi-bin/test-cgi HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 25
banan=Bj&F6rn&bilar=volvo
HTTP/1.1 200 OK
Date: Thu, 02 May 2002 07:36:14 GMT
Server: Apache/1.3.23 (Darwin) PHP/4.1.2
Connection: close
Content-Type: text/plain
CGI/1.0 test script report:
  
```

Observera Content-Length som måste vara med och som innehåller längden på body-delen



## Metoden HEAD

- Den sista ”vanligt” använda metoden är HEAD.
- Den används om man ENDAST är intresserad av headerdelen av svaret, dvs är ointresserad av själva datadelen.
- Kan vara användbart t.ex. för att se ifall en resurs har ändrats sedan en tidigare tidpunkt. Om så inte är fallet behöver man inte ladda ner resursen ifråga.



## Exempel på HEAD via telnet

```
media-telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.nada.kth.se.
Escape character is '^]'.
HEAD /a.txt HTTP/1.0

HTTP/1.1 200 OK
Date: Thu, 02 May 2002 07:43:47 GMT
Server: Apache/1.3.23 (Darwin) PHP/4.1.2
Last-Modified: Wed, 01 May 2002 17:15:48 GMT
ETag: "93195-1e-3cd822c4"
Accept-Ranges: bytes
Content-Length: 30
Connection: close
Content-Type: text/plain
X-Pad: avoid browser bug

Connection closed by foreign host.
```



## GET, POST och HEAD

- GET används vid all hämtning av statiska resurser, och ibland för dynamiska resurser.
  - Förfrågningar av dynamiska resurser kan lagras som bokmärken, eftersom indata kodas i URLen
- POST används ibland för dynamiska resurser och alltid när filer ska bifogas.
- HEAD används när klienten är ointresserad av innehållet i resursen, men däremot är intresserad av metadata gällande resursen.



## Programspråkspecifika bindningar för cgi

- För många programspråk finns idag ”paket” som kan användas för att göra det enklare att komma åt cgi-variabler.
- Speciellt arbetet att dela upp en QUERY\_STRING i dess variabelnamn/variabelvärde-par är en trevlig funktion.
- Vanligtvis kan man dock, om inte annat, komma åt själva råvariablerna, t.ex. i ett shell-script
- echo "\$QUERY\_STRING"



## Vanliga cgi-scriptspråk

- Några språk som ofta används för cgi-script är:
  - Perl
  - Python
  - C/C++
- Ingen av dessa ska vi dock gå igenom i denna kurs



## Nästa gång

- Hur skapa dynamiskt innehåll med PHP
- Web och databaser
- Sessionshantering
  - Cookies, URL-rewriting, hidden fields
- Skicka/ta emot mail från websidor



## Sammanfattning

- Datatyper på webben hanteras med MIME-typer
- Filöverföring sker oftast med http
- Parametrar kan skickas mellan webläsare och webserver via GET (i URLen) eller POST (i body-delen). I båda fallen ska parametrarna URL-kodas
- En metod att generera dynamiskt innehåll är att använda vanliga program med printsatser.



## Referenser

### Referenser

#### http

- <http://www.jmarshall.com/easy/http>

#### headers

- RFC 822

#### MIME

- RFC 1521

#### URI

- RFC 2396