

# Kapittel 1

## Kom i gang med PHP

### Læringsmål:

Dette kapittelet vil fungere som en enkel oppstartsguide for å komme i gang med PHP. Du vil lære å:

- Installere tjenerprogramvare og PHP.
- Lage et enkelt «kom-i-gang-script», som illustrerer hvor enkelt det er å skape dynamikk med PHP.
- Forstå samspillet mellom tjener og klient. Etter å ha jobbet med dette kapittelet skal du forstå at PHP og HTML ikke er motsetninger, men utfyllende.
- Du skal også kjenne til konfigurasjonsfilen, og hvordan dens innstillinger påvirker din programmeringsstil.

## 1.1 Hvorfor PHP?

Med PHP kan du programmere dynamiske, interaktive websider. Det betyr at det som vises i nettleseren genereres i samme øyeblikk som noen besøker sidene. PHP er et fullverdig programmeringsspråk og kan dermed brukes til å lage enkle så vel som de mest avanserte webløsninger. Du har kanskje allerede lagd vanlige Internett-sider som er skrevet i HTML? Slike er statiske, og passer for informasjon som ikke endres over tid. Endringer er mulige, men krever at en person aktivt går inn for å redigere sidene. Denne boka tar for seg hvordan PHP kan brukes til å lage dynamiske, fleksible, brukervennlige og ikke minst sikre løsninger.

PHP er et relativt nytt programmeringsspråk (lagd i 1994) og faller i kategorien *scriptspråk*. En forklaring på navnet PHP er *Personal Home Page (Tools)*. En annen vinkling er at PHP er et rekursivt definert akronym. Med dette menes at akronymet (forkortelsen) inngår i betydningen – det refererer til seg selv. Dermed kan vi si at *PHP* står for *PHP: Hypertext Preprocessor*.<sup>1</sup>

Mange frivillige har vært aktivt med i prosessen med å utvikle språket, blant annet kan Stig Sæther Bakken fra Trondheim nevnes. Hans navn finner du i den offisielle PHP-manualen på Internett, under adressen

<http://www.php.net/manual/>

Interessen for PHP blir stadig større, og det kommer jevnlig nye, forbedrede utgaver av språket. PHP modnet gjennom versjon 3. Versjon 4 satte søkelyset på sikkerhet. Versjon 5 kom sommeren 2004. PHP 5 er et naturlig valg, ikke bare fordi det er nytt, men også fordi det kan brukes av både nybegynneren og den avanserte bruker. PHP 5 har fullstendig støtte for objektorientering til glede for de som måtte ønske å programmere PHP objektorientert fremfor proseduralt. PHP 5 har også en rekke finesser. Disse avanserte mulighetene blir introdusert i denne boka, men merk deg at hovedvekten ligger på generell, basis PHP-programmering. Det er en rekke grunner til at språket er populært:

- *For alle*: PHP er så enkelt at nybegynnere kan komme i gang uten å ha erfaring i programmering fra før av. Muligheten for å blande HTML og PHP-kode gjør at du gradvis kan legge til dynamikk i dine eksisterende Internett-sider, uten å måtte starte på begynnelsen. Syntaksen ligner på den i Java, C og Perl, hvilket gjør at det er ekstra lett å lære PHP for de som kan programmering fra før av.
- *Utbredelse*: En hel rekke Internett-leverandører har støtte for PHP. I tillegg til mange hobbyløsninger laget med PHP, er mange kjente, store nettstedet lagd i PHP. Et nettsted som har lenker som slutter med adressen *.php*, bruker PHP som teknologi. Stadig flere tar i bruk PHP, og det er derfor ekstra verdifullt for deg å lære å mestre språket.
- *Fleksibel koding*: Du kan enten skrive tradisjonell prosedural kode eller velge en objektorientert tilnærming. I tillegg kan de to fremgangsmåtene blandes, noe som av og til er hendig. Med PHP versjon 5 er støtten for OOP fullstendig.

---

<sup>1</sup> Et annet kjent akronym er *GNU* som står for *GNU's Not UNIX*.

Denne boka ser i utgangspunktet på vanlig prosedural koding, men tar også opp den objektorienterte muligheten i kapittel 13. Skulle du allerede ha programmert løsninger i Java, kan disse gjenbrukes siden PHP tilbyr funksjonalitet som bruker Java-objekter som om de var PHP-objekter.

- *Hurtighet*: Løsninger bygget på PHP utføres raskt og stabilt, og krever lite ressurser på tjenermaskinen.
- *Sikkerhet*: PHP kan settes opp til å tilfredsstille ulike grader av sikkerhet. Dette er et ikke-trivielt tema. Sikkerhetsaspekter med programmeringen vil bli tatt hensyn til og diskutert i mange av eksemplene i boka. Kapittel 11 har en mer grundig gjennomgang av datasikkerhet, og tar opp problematikken rundt og teknikker for å unngå hacking, adgangsbegrensning, og mye mer.
- *Avansert, men enkelt*: Det kommer stadig nye tilleggsmoduler/biblioteker som gjør det mulig å lage for eksempel grafikk, PDF-filer og Flash-filmsnutter dynamisk. Du kan lage handlekurv-systemer, spill, fotoalbum der du presenterer bildene dine på flotte måter, bursdagssystemer, spørrekonkurranser, gjestebøker, distribusjonssystemer, ditt eget nyhetsforum, søkemotorer, oversettelsesprogrammer og så videre. Bare fantasien setter grenser. Fellesnevneren er at selv avansert funksjonalitet blir enkelt når du kan det grunnleggende, noe som skyldes fine eksempler på web og en rekke innebygde funksjoner i PHP.
- *Plattformuavhengighet*: De scriptene du lager må kjøres av PHP, som igjen må være installert på en webtjener. PHP fungerer med de fleste operativsystemer, både Windows, Linux, mange varianter av UNIX, Mac OS (nyere enn OSX) og RISC OS. Webtjenere som kan benyttes er Apache, Internet Information Server (IIS) fra Microsoft, Personal Web Server, Netscape & iPlanet servers, O'Reilly Website Pro server, Caudium, Xitami, OmniHTTPd, og mange flere. Du har altså mulighet til fritt å velge både operativsystem og webtjener. Det er godt å vite at du som utvikler kan lage løsninger som fungerer for alle og er portable mellom plattformer.
- *Åpen kildekode*: Mange tusen utviklere bidrar til å lage forbedrede versjoner av PHP og moduler for utvidet funksjonalitet. Det betyr at PHP er gratis å bruke for alle parter.
- *Informasjon*: PHP har støtte for alle store databasesystemer og god XML-støtte. Boka tar for seg bruk av databasesystemet MySQL, men forskjellene mellom ulike databasesystemer er relativt små i PHP.
- *Tjenester*: PHP støtter kjente protokoller så som IMAP, NNTP, HTTP, LDAP, COM og så videre. Dette er ansett for å være mer avansert programmering som vi ikke går igjennom i denne boka.
- *Hjelp*: Denne læreboka er et godt utgangspunkt for å lære PHP, men av og til vil du kunne trenge annen hjelp. Det fins per i dag utallige portaler på Internett, e-postlister, nyhetsgrupper/diskusjonsfora og samlinger av kodeeksempler hvor du kan søke hjelp.
- *Samme prinsipper som ASP og JSP*: Når du kan programmere i et av de tre store språkene ASP, JSP eller PHP, kan du enkelt overføre dine kunnskaper til et annet. Hvilket av de tre store du bør velge som førstespråk er litt avhengig av

din bakgrunn, dine interesser og dine fremtidsplaner. PHP er trolig enklest å komme i gang med, og dessuten veldig kraftig. De foregående punktene bør ikke levne tvil om at du har gjort et godt valg!

Til oppsummering: Det er mange fordeler med PHP, ikke bare at det er gratis i bruk. Med PHP kan du lage solide, sikre løsninger som fungerer raskt for tusenvis av samtidige brukere. Det er enkelt å komme i gang, og samtidig har språket mange avanserte muligheter du som webutvikler kan benytte til å lage det du ønsker av webløsninger. PHP er i kontinuerlig vekst og blir stadig mer utbredt. Fleksibiliteten er ivaretatt i og med at alle store operativsystemer og webtjenere støttes, i tillegg til at både prosedural og objektorientert tilnærming kan brukes av den som programmerer løsningene.

## 1.2 Installasjon av nødvendig programvare

Alle PHP-script kjøres på en webtjener. Resultatet av kjøringen sendes til den besøkendes nettleser for visning. For å kunne lage dine egne script, er det derfor nødvendig at du har tilgang på en *webtjener* som har installert *PHP-programvare*. Derksom du bruker en ISP (Internet Service Provider, eller Internett-leverandør på godt norsk), vil trolig denne kunne gi deg muligheten til å laste opp og kjøre PHP-filer på sine tjenermaskiner.

### Hva må til for å programmere i PHP?

Det må være installert PHP-programvare på en webtjener som skal kunne kjøre PHP-script. Når denne forutsetningen er på plass kan du utvikle webløsninger i PHP. Legg merke til at du kan installere en webtjener på din vanlige arbeidsmaskin, både om du har Windows, Linux eller Mac. Samspillet mellom tjener og klient blir for øvrig forklart nærmere mot slutten av dette kapittelet.

I praksis innebærer det å programmere i PHP å gjenta følgende prosess inntil løsningen er ferdig:

1. Skrive/endre/videreutvikle kode i en editor.
2. Lagre filen(e) med filendelse **.php** på riktig sted.
3. Teste de siste endringene ved å skrive inn riktig URL i adressefeltet i nettleseren, på formen <http://www.dinISP/enSide.php>. Dersom du bruker din egen maskin under utvikling, brukes <http://localhost/enSide.php>.
4. Søke hjelp i lærebøker (for eksempel denne), den offisielle PHP-manualen eller andre steder for å løse konkrete problemstillinger.

Dersom du legger ut og tester sidene du lager hos en ISP, må du overføre endrede filer, for eksempel med FTP, for hver gang du skal teste på nytt. Dette kan være

dyrt og er tungvint. Et godt alternativ er derfor å installere både en webtjener og den nødvendige PHP-programvaren lokalt på din egen maskin. Du vil da bruke <http://localhost/enSide.php> som adresse (URL) for å teste scriptene dine som om de lå hos en ordentlig ISP. Når løsningen er fullstendig testet, kan den publiseres på Internett ved å overføre alle filene til riktig ISP. Slik trenger du ikke være koblet på nettet hver gang du skal jobbe med PHP. Samtidig sparer du mye tid på filoverføring og slipper å bekymre deg for om riktig versjon av filene er lastet opp.



En annen fordel er at du da viser «alt-eller-ingenting» til dine besøkende. Det er uheldig fra et sikkerhetsmessig perspektiv å la andre se at sidene dine blir gradvis utviklet – en hacker som får overvåke utviklingsprosessen vil nemlig kunne avsløre mange særtrekk ved oppbyggingen av nettstedet, som senere kan brukes mot deg.

Du kan velge mellom flere webtjenere. *Apache* er mest utbredt og enkel i bruk. Legg merke til at PHP også fungerer med IIS fra Microsoft og mange andre tjenerløsninger. Dersom du allerede har installert PHP og en webtjener, kan du gå til delkapittel 1.3. Det kan uansett være greit for forståelsen av webprogrammering å gjennomføre (eller lese om) en installering av nødvendig programvare på egen maskin.

Du må bestemme deg for hvilken versjon av PHP du vil bruke. PHP 5 er et godt valg. Jo høyere versjonstallet er, desto nyere er utgaven. Vær derimot forsiktig med å laste ned Beta-utgaver. Disse kan være ustabile og gi deg mer problemer enn glede. Har du en eldre maskin med for eksempel en gammel utgave av Windows, bør du lese på de offisielle PHP-sidene for å finne ut hvilken av de eldre versjonene du bør velge og eventuelle oppdateringer som må kjøres.

Nå skal vi i korte trekk gå gjennom det du trenger for å installere PHP og Apache under både Windows (neste delkapittel), Linux/UNIX (delkapittel 1.2.7 på side 30) og Mac (delkapittel 1.2.8 på side 32). Gjennomgangen er basert på veiledningen som er å finne på de offisielle PHP-sidene, dokumentasjonen fra Apache-sidene, og flere engelske lærebøker. Vi starter med Windows. Målet er å gi en så kortfattet gjennomgang som mulig, og samtidig gi forståelse for hva som skjer og hvorfor.

Installasjon av databasesystemet MySQL gjøres bare for Linux i dette kapittelet, siden det er nødvendig å kompilere PHP med MySQL-støtte fra starten av. Brukere av Windows vil først få forklart hvordan MySQL kan installeres i kapittel 9, fordi det ikke er nødvendig å installere MySQL med en gang under Windows. Legg også merke til at PHP 5 kommer med et utmerket, innebygd alternativ som heter SQLite. Dette gjennomgås kort i kapittel 13.4.

### 1.2.1 Installasjon av Apache og PHP for Windows på 1-2-3

Neste ramme har en veldig kort oppsummering av hovedpunktene i installasjonsprosessen av Apache og PHP for Windows. Etterpå følger flere delkapitler med en mer detaljert beskrivelse som utdypet nærmere hva som skjer i installasjonsprosessen, viktige sikkerhetsmessige forhold å tenke på, og noen andre muligheter.

Har du installert PHP før, men glemt hvordan, er punktene under trolig tilstrekkelige til at du kan installere det du trenger på 1-2-3. Er du nybegynner og vil forstå detaljene, kan det være lurt å først lese forklaringen, for så å vende tilbake til punktlisten når du skal foreta installasjonen. Ditt valg.

## 10 enkle steg for å installere Apache og PHP for Windows

Merk: I denne boka installeres først Apache, så PHP, og til slutt MySQL (kapittel 9). Dette er også for å få en bedre forståelse av hvordan samspillet mellom de tre programmene fungerer – noe som kan gjøre det lettere å forstå webprogrammeringen du snart skal ta fatt på. Det eksisterer pakkeløsninger som består av Apache, PHP og MySQL. I følge PHP-manualen er det anbefalt å installere de ulike delene hver for seg, men dersom du ønsker, kan du søke etter en pakkeløsning på web og få en noe enklere og raskere installasjonsprosess. Tips til søkeord: «install php apache mysql windows». Her er en oppsummering av den manuelle fremgangsmåten:

1. Last ned ønsket versjon av Apache og start installasjonsprogrammet.
2. Sett `localhost` som domene og installer som en tjeneste. Sørg for at ikke brannmuren i Windows blokkerer Apache.
3. Endre `DocumentRoot` i konfigurasjonsfilen ***httpd.conf*** for å kunne lagre scriptene dine adskilt fra programinstallasjonen (det er alltid lurt å skille dokumenter og programvare).
4. Legg følgende setninger på riktig sted i ***httpd.conf*** slik at Apache bruker PHP til å kjøre alle PHP-script (her antas at PHP blir installert i katalogen ***C:\php\***).

```
LoadModule php5_module "C:/php/php5apache2.dll"
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
PHPIniDir "C:/php"
```

5. Last ned zip-utgaven av PHP-versjonen du ønsker.
6. Pakk ut zip-filen i katalogen ***C:\php\***.
7. Legg til `C:\php` i miljøvariabelen `PATH` i Windows.
8. Endre navnet på konfigurasjonsfilen ***php.ini.recommended*** til ***php.ini***.
9. Sjekk at innstillingen `display_errors = On` i ***php.ini*** slik at feilmeldinger vises når du tester scriptene dine.
10. Legg et enkelt php-script som du kaller ***test.php*** i den katalogen du satte til å være `DocumentRoot` i ***httpd.conf***, og test at alt fungerer ved å skrive adressen ***http://localhost/test.php*** i nettleserens adressefelt.

## 1.2.2 Nærmere forklaring til installasjon av Apache under Windows

Apache er fri, gratis og meget solid. Versjoner i 1.3 serien er mye brukt, men etter at 2.0 serien har vist seg å være stabil gjennom flere år, er det mest naturlig å installere siste stabile versjon av Apache 2. Du finner Apache til nedlasting fra adressen <http://httpd.apache.org/>.

### 1. Nedlasting:

Du bør laste ned den versjonen som er oppført som siste stabile versjon på nettstedet. I skrivende stund er dette 2.0.53. Pass på å laste ned en såkalt «Win32 binary», og ikke kildekoden med mindre du ønsker å kompilere Apache selv. Vi har lastet ned filen

```
apache_2.0.53-win32-x86-no_ssl.msi
```

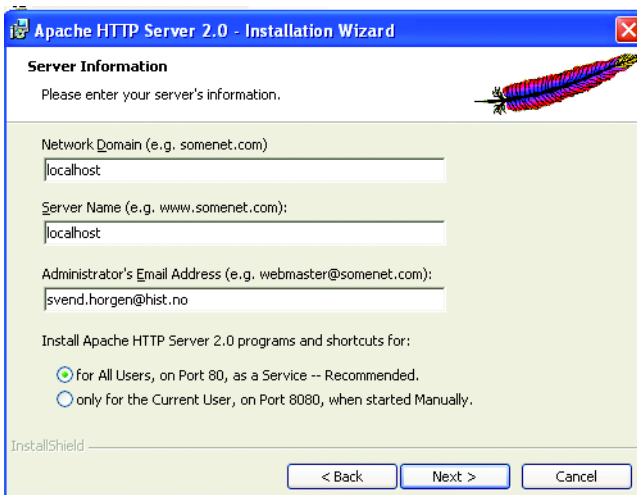
som forutsetter at Microsoft System Installer (MSI) er installert på maskinen. Hvis du mangler MSI-støtte (dette får du eventuelt beskjed om når du prøver å installere MSI-utgaven), kan du laste ned en noe større exe-variant.

### 2. Installasjon:

Start installasjonen av Apache ved å dobbeltklikke på filen du har lastet ned, og følg instruksjonene. Du blir bedt om å oppgi noen opplysninger, og kan sette domene til `localhost` som vist i figur 1.1 eller et annet navn om du ønsker det.

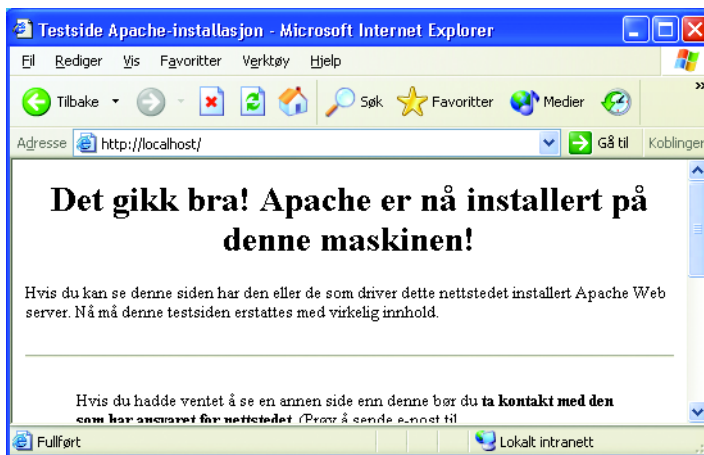
Velg «for All users, on Port 80, as a Service -- Recommended». Deretter må du velge hva som skal installeres, enten «Typical» eller «Custom».

Under installasjonsprosessen kan du få spørsmål fra Windows-brannmuren om du vil blokkere Apache. Dersom du ønsker å kontakte webtjeneren fra en annen maskin må du ikke blokkere Apache. Etter at installasjonen er fullført skal du ha fått en ny katalog på harddisken. Hver gang Windows starter opp vil Apache starte samtidig. Et ikon i statuslinjen gjør at du lett kan starte, stoppe eller restarte Apache.



Figur 1.1 En veiviser hjelper deg gjennom installasjonen av Apache

Åpne en nettleser og skriv inn adressen `http://localhost`. Merk at det ikke er noen sammenheng mellom det du skrev som nettverksdomene og tjenernavn under installasjonsprosessen, og localhost-delen av URL-en. Apache må kjøre før du sender forespørsler til localhost, fordi det er Apache som behandler forespørslene og sender svar tilbake til nettleseren. Dersom alt er gjort riktig skal en webside komme frem, for eksempel følgende:



En velkomsthilsen vises hvis installasjonen var vellykket og Apache fungerer

### 1.2.3 Konfigurasjonsfilen `httpd.conf`

Filen `httpd.conf` har en rekke innstillinger som sier hvordan Apache skal oppføre seg. Denne kalles for *konfigurasjonsfilen* og ligger i katalogen `conf` der du installerte Apache (`C:\Programfiler\Apache Group\Apache2\` som standard). I hovedsak er den fylt med tekst som gir enkle forklaringer på hva de ulike innstillingene gjør. Alle steder som starter med tegnet `#` (heter skigard) angir at resten av linjen er en kommentar og ikke skal leses av Apache. Noen innstillinger er også kommentert ut og vil dermed ikke ha innvirkning. Du kan fjerne kommentarene bak disse. I så fall blir innstillingen tatt hensyn til (anbefales bare hvis du vet hva du gjør).

Når du skal konfigurere Apache må du redigere `httpd.conf` ved å åpne den i en editor, for eksempel Notisblokk, endre på innholdet på riktig sted, og lagre filen på nytt igjen. Neste gang Apache starter vil endringene tre i kraft.

Åpne `httpd.conf` og søk deg frem til innstillingen

```
Listen 80
```

Dette betyr at Apache skal lytte på port 80.



For å øke sikkerheten kan du angi at Apache bare skal lytte etter forespørsler på IP-adressen 127.0.0.1. Dette medfører at bare prosesser som kjører på samme maskin får kontakte webtjeneren. Det betyr at ingen utenfra kan få kontakt med din webtjener selv om du er tilkoblet Internett. Endre `Listen 80` til `Listen 127.0.0.1:80`.

Du kan gjøre endringer så ofte du vil, men innstillingene trår først i verk når du restarter Apache (eventuelt start og stopp igjen).



Kort fortalt brukes IP-adresser til å identifisere ressurser tilkoblet Internett. Informasjonen som sendes mellom to maskiner deles inn i pakker, hvor hver pakke blir merket med IP-adressen til mottakeren. Når du skriver inn adressen <http://www.aitel.hist.no/fag/php/index.php>, vil klientmaskinen spørre navnesystemet DNS om IP-adressen som skjuler seg bak navnet. Når IP-adressen som tilsvare <http://www.aitel.hist.no/> er funnet, vil nettleseren sende en forespørsel til denne IP-adressen og be om å få tilsendt filen `index.php` som ligger i katalogen `fag/php`. Alt dette skjer i bakgrunnen, så du trenger ikke å huske IP-adresser, og trenger heller ikke gjøre mer enn å skrive inn riktig adresse i nettleseren. Når du bruker `localhost`, sender du forespørsel lokalt.

Dersom du åpner katalogen `htdocs/` hvor Apache er installert, vil du se en rekke index-filer, blant annet filen `index.html.no`. Det er språkoppsettet i din nettleser som bestemmer hvilken fil som skal brukes. Har du svensk oppsett vises `index.html.sv`

For å teste lager vi nå vår egen, enkle HTML-fil med dette innholdet:

```
<html><body>
<h1>Dette er en test</h1>
</body></html>
```

Filen lagres som `enTest.html` i en ny underkatalog av `htdocs/` som heter `test/`. Dens innhold skal komme frem når denne adressen skrives inn i nettleseren:

<http://localhost/test/enTest.html>

Det er innstillingen `DocumentRoot` i filen `httpd.conf` som angir i hvilken katalog Apache skal lete etter ressurser som forespørres, for eksempel filen `enTest.html`. `DocumentRoot` kalles også for webrot, og det er her du må legge alle dine PHP-script. Vi skal nå endre dette slik at alle de løsningene vi lager kan legges et annet sted på harddisken, gjerne på en annen partisjon:

- Åpne `httpd.conf`.
- Bla et stykke nedover (eller søk) til du finner direktivet `DocumentRoot`. Det som står her er gjeldende webrot, for eksempel

```
DocumentRoot "C:/Programfiler/Apache Group/Apache2/htdocs"
```

- Legg merke til at skråstrekene går motsatt vei av det du ser i Windows Utforsker. Endre til ønsket katalog, for eksempel

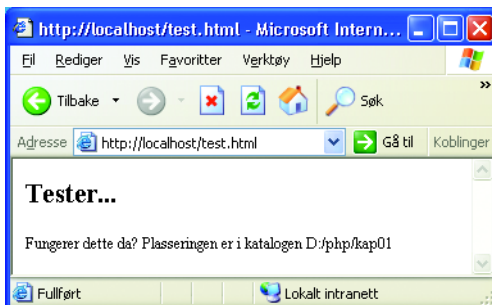
- DocumentRoot "D:/PHP/kap01"
- Bla ned til du kommer til
 

```
<Directory "C:/Programfiler/ApacheGroup/Apache2/htdocs">
```
- Endre også den til samme sti som over
 

```
<Directory "D:/PHP/kap01">
```
- Lagre **httpd.conf**.
- Restart Apache slik at endringene i konfigurasjonsfilen trer i kraft.
- Lag en ny HTML-fil som heter **test.html** og plasser den i katalogen **D:\PHP\kap01\**, som nå er den nye roten til webtreet.
- Skriv inn adressen <http://localhost/test.html> i nettleseren.

Innholdet i Internett-siden du nettopp lagde, skal komme frem hvis alt er gjort riktig.

Apache er nå satt opp til å håndtere alle forespørsler etter vanlige HTML-filer. Filene er lagret i katalogen **D:\php\kap01\** på harddisken, men angis med URL-adresser på formen <http://localhost/filnavn.html>. Før Apache kan konfigureres til å håndtere PHP-script, må PHP-programvaren installeres.



Figur 1.2 Test på om filen **test.html** i katalogen **D:\php\kap01\** behandles av Apache

### 1.2.4 Installasjon av PHP under Windows

Du har to muligheter for å installere nødvendig programvare:

- Bruke installasjonsveiviser – last ned exe-fil.
- Foreta manuell installasjon – last ned zip-fil.

Ønsker du å komme fortest mulig i gang, kan du bruke installasjonsveiviseren. Vær oppmerksom på at denne har flere ulemper knyttet til sikkerhet og fleksibilitet. En manuell installasjon anbefales derfor. Manuell installasjon er nokså lett dersom du følger denne oppskriften nøye (testet under Windows XP med Apache 2):

1. Nedlasting:

På <http://www.php.net/downloads.php> finner du siste stabile versjon av PHP,

nye beta-versjoner som ikke er ferdige enda, og sikkerhetsoppdateringer. Avsnittet «Complete Source Code» er først og fremst myntet på Unix-brukere. Gå til avsnittet «Windows Binaries». Last ned ønsket versjon av PHP. Dette skal være en zip-fil med et navn på formen

***php-X.Y.Z-Win32.zip***

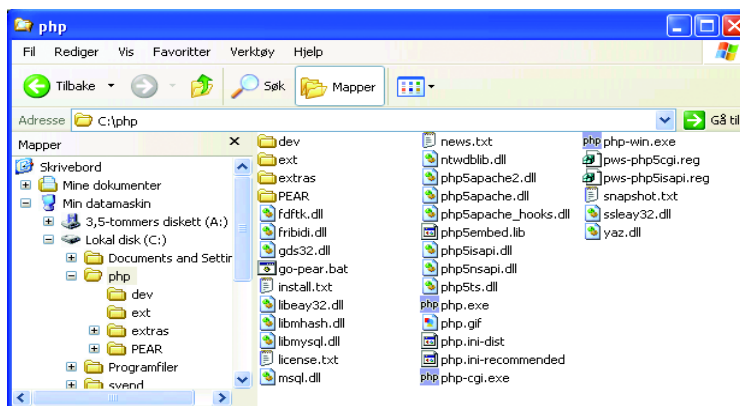
for eksempel en av PHP 5 versjonene:

***php-5.0.3-Win32.zip***

Her er PHP 5.0.3 brukt, men det er lurt å bruke den nyeste versjonen som gjelder når du leser dette.

## 2. Riktig katalog:

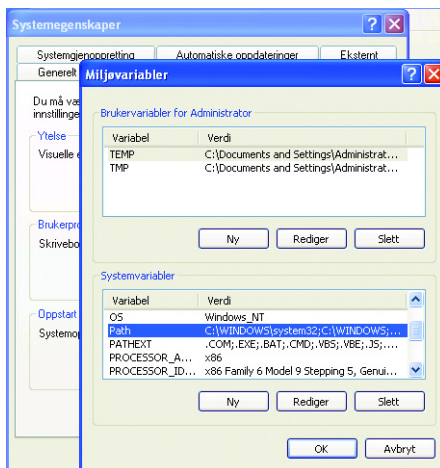
Pakk ut zip-filen til for eksempel **C:\php** og du vil se en rekke filer og underkataloger, som illustrert i figur 1.3.



Figur 1.3 Katalogstrukturen etter utpakking av PHP

## 3. Sett opp miljøvariabelen PATH i Windows:

Du må gjøre det mulig for PHP-programvaren å bruke biblioteksfiler som ligger lagret i PHP-katalogen. Det enkleste er å gjøre den katalogen der PHP ligger (her **C:\php**) tilgjengelig for det som er PATH i Windows. Katalogen legges til PATH ved å høyreklikke på «Min Datamaskin», velge «Egenskaper», «Avansert», og knappen «Miljøvariabler». Du ser nå en liste med «Systemvariabler». Marker elementet «Path» og trykk på knappen «Rediger». Se figur 1.4. Helt bakerst skriver du inn `;C:\php` og klikker OK. Dermed er **C:\php** lagt til i Windows PATH. Det er viktig med semikolon mellom det du skriver inn og katalognavnene foran, siden det er semikolonet som skiller de ulike katalogene i PATH fra hverandre.



Figur 1.4 Du må legge til katalogen hvor PHP ligger i Windows PATH

#### 4. Konfigurasjon av PHP:

Filen *php.ini-recommended* inneholder standardinnstillinger for hvordan PHP skal oppføre seg. Det er *php.ini* som er konfigurasjonsfilen til PHP. Denne fins i utgangspunktet ikke, så lag en kopi av *php.ini-recommended* og gi denne navnet *php.ini*. Du kan endre innstillingene på samme måte som i Apaches *httpd.conf*. Legg merke til at kommentarer i *php.ini* starter med semikolon. Dersom du har spesielt høye krav til sikkerhet, bør du gå gjennom hver linje i filen og dobbeltsjekke at innstillingene passer dine behov.

Det kan være lurt å åpne *php.ini* og sørge for at `display_errors` er slått på. Det gjøres ved å fjerne semikolonet slik at innstillingen ikke er kommentert ut:

```
display_errors = On
```

Da vil det komme frem feilmeldinger i nettleseren hvis det er feil i et PHP-script som gjør at det ikke kjører som forventet. Av sikkerhetsmessige årsaker bør denne innstillingen være avslått på den webtjeneren hvor siden er offentliggjort. Dermed hindres at informasjon om filnavn, databasetabeller og liknende vises til brukeren i feilsituasjoner. Under utvikling er det derimot en viktig hjelp å få se nettopp slike feilmeldinger. Ha derfor innstillingen på (On) så lenge du utvikler i PHP på egen maskin.

### 1.2.5 Nødvendig konfigurasjon av *php.ini* og *httpd.conf*

Du har nå lastet ned og installert Apache og PHP. Webtjeneren kan håndtere HTML, men må konfigureres til å gi PHP-programvaren i oppgave å håndtere PHP-script. Let deg frem til følgende avsnitt i konfigurasjonsfilen *httpd.conf* (en del linjer er kuttet bort for å spare plass på siden):

```
# Example:
# LoadModule foo_module modules/mod_foo.so
#
LoadModule access_module modules/mod_access.so
LoadModule actions_module modules/mod_actions.so
LoadModule alias_module modules/mod_alias.so
LoadModule asis_module modules/mod_asis.so
LoadModule auth_module modules/mod_auth.so
#LoadModule auth_anon_module modules/mod_auth_anon.so
#LoadModule auth_dbm_module modules/mod_auth_dbm.so
#LoadModule usertrack_module modules/mod_usertrack.so
#LoadModule vhost_alias_module modules/mod_vhost_alias.so
#LoadModule ssl_module modules/mod_ssl.so
```

Avsnittet finnes et stykke ned i *httpd.conf* og har flere setninger kommentert ut, det vil si at de ikke tas hensyn til av Apache. Skriv inn følgende setninger nederst i avsnittet:

```
LoadModule php5_module "C:/php/php5apache2.dll"
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
PHPIniDir "C:/php"
```

Den første setningen gjør at PHP blir tilgjengelig i Apache. Neste setning sikrer at PHP vil få i oppdrag å behandle alle filer som slutter med filnavnet *.php*. Den andre setningen gjør at koden i alle filer som har endelsen *.phps* vises som HTML med passende farger. Det kan være lurt å bruke denne muligheten for å vise selve koden til scriptene dine i nettleseren, for eksempel om du skal lage et lite opplæringssted på web hvor du både demonstrerer det koden gjør og viser hvordan den ser ut. Strengt tatt kan denne linjen utelates.

Den siste linjen er viktig. Den sier hvor konfigurasjonsfilen til PHP, nemlig *php.ini*, ligger lagret.

Hvis du har programmert tidligere i PHP versjon 3, vil filnavnene være av type *.php3*. I stedet for å endre filtypen på alle de gamle scriptene du har, kan du legge til denne endelsen slik at PHP behandler både gamle og nye PHP-filer. Av sikkerhetsmessige årsaker bør du også gjøre dette med filer som ender på *.inc*. Denne navngivingen brukes ofte i forbindelse med inkludering (men som du vil se i kapittel 5.1, bruker vi endelsen *.inc.php* på include-filer i denne boka ).

For å få filendelsene *.php3* og *.inc* til å tolkes som PHP-script, må du bytte ut den andre setningen slik at den ser slik ut:

```
AddType application/x-httpd-php .php .php3 .inc
```

Dersom brukeren skriver inn <http://adresse.no/katalog/> og ikke angir noe filnavn, vil Apache i henhold til standardkonfigurasjonen se etter en fil som heter *index.html* i denne katalogen. Du kan åpne for at også filer som heter *index.php* skal behandles på samme måte ved å endre direktivet *DirectoryIndex*. Det som står først får høyest prioritet.

```
DirectoryIndex index.php index.html index.html.var
```

Restart Apache, og du er nå ferdig med installasjonen! Gå nå til kapittel 1.2.9 på side 33 og se hvordan du kan teste om alt fungerer som det skal.

### 1.2.6 Oppgrader PHP til en ny versjon – lett som en lek (Windows)

Det er veldig enkelt å oppgradere PHP dersom du har gjort den manuelle installasjonen som gjennomgått til nå. Gjør slik:

- Stopp Apache.
- Endre navnet på katalogen der PHP er installert, til for eksempel *C:\php-gammel*.
- Last ned en zip-fil (med den nye versjonen) og pakk den ut til der forrige versjon av PHP lå, for eksempel *C:\php*
- Kopier *php.ini* fra den gamle katalogen til den nye dersom du vil gjenbruke innstillingene.
- Start Apache på nytt.

### 1.2.7 Installasjon av MySQL, Apache og PHP under Linux/UNIX

Det fins mange måter å installere MySQL, PHP og Apache under Linux. I denne boka blir databaser gjennomgått i kapittel 9, med utgangspunkt i MySQL, men du kan også bruke andre systemer. Det er ikke mulig å dekke alle her, men en generell forklaring er på sin plass. Det er alltid lurt å lese medfølgende installasjonsveiledning for å få et riktig oppsett. Du har i hovedsak to alternativer: Gjøre bruk av programpakker som følger med din distribusjon, eller installere fra kildekode.

I dokumentasjonen til distribusjonen du bruker, vil du finne informasjon om hvordan pakker skal installeres. Dette vil variere mellom de ulike distribusjonene. Ved installasjon av programpakker er det viktig å benytte pakker fra din versjonen av din distribusjon. Hvis du har Mandrake 10.0, må du altså bruke pakker som tilhører Mandrake 10.0.

Hvis du velger å installere fra kildekode, er det nødvendig å installere databasesystemet MySQL før PHP blir installert.

Sett også opp konfigurasjonsfilene til Apache (*httpd.conf*) og PHP (*php.ini*) som beskrevet i de foregående kapitler.

**Installasjon av MySQL** fra kildekode består av følgende steg:

- Last ned kildekoden i form av en pakket fil, for eksempel *mysql-4.0.20.tar.gz*
- Pakk ut arkivet, `tar zxf mysql-4.0.20.tar.gz`
- Gå til katalogen som ble opprettet: `cd mysql-4.0.20`
- Kjør konfigureringsskriptet (*./configure*) og foreta eventuelle innstillinger. På dette steget må en ta stilling til hvilken funksjonalitet som skal være med, hvor

ting skal installeres, hvor tredje-parts bibliotek befinner seg, og så videre.

Eksempel: `./configure --prefix=/usr/local/mysql`

- Kompiler programmet ved å skrive `make`
- Installer programmet ved å skrive `make install`
- Kjør et script som oppretter databasefilene: `scripts/mysql_install_db`
- Opprett en bruker og brukergruppe for MySQL:
 

```
groupadd mysql
useradd -g mysql mysql
```
- Gi denne brukeren rettigheter til å aksessere databasefilene:
 

```
chown -R mysql:mysql /usr/local/mysql/var
```
- Start MySQL-tjeneren: `/usr/local/mysql/bin/myslqd_safe &`
- Sett passord på administratorbrukeren av databasesystemet:
 

```
/usr/local/mysql/bin/mysqladmin -u root password 'pass'
```

MySQL brukes først i kapittel 9, men det er best å installere allerede nå. Du kan også lese i dokumentasjonen til din distribusjon om hvordan MySQL kan settes opp til å starte automatisk sammen med systemet.

**Installasjon av Apache** fra kildekode består av følgende steg:

- Last ned kildekode i form av en pakket fil, for eksempel ***httpd-2.0.49.tar.gz***
- Pakk ut arkivet: `tar zxf httpd-2.0.49.tar.gz`
- Gå til katalogen som ble opprettet: `cd httpd-2.0.49`
- Kjør konfigureringsscriptet (`./configure`) og foreta eventuelle innstillinger. På dette steget må en ta stilling til hvilken funksjonalitet som skal være med, hvor ting skal installeres, hvor tredje parts bibliotek befinner seg, og så videre. Eksempel: `./configure --enable-so`
- Kompiler programmet ved å skrive `make`
- Installer programmet ved å skrive `make install`
- Konfigurer Apache ved å redigere ***httpd.conf***. Kapittel 1.2.3 på side 24 beskriver noen muligheter. For eksempel: `DocumentRoot "/www/"`
- Start Apache: `/usr/local/apache2/bin/apachectl start`

Du kan også lese i dokumentasjonen til din distribusjon om hvordan Apache kan settes opp til å starte automatisk sammen med systemet.

**Installasjon av PHP** fra kildekode består av følgende steg:

- Last ned kildekode i form av en pakket fil, for eksempel ***php-5.0.0.tar.gz***
- Pakk ut arkivet: `tar zxf php-5.0.0.tar.gz`
- Gå til katalogen som ble opprettet: `cd php-5.0.0`
- Kjør konfigureringsscriptet (`./configure`) og foreta eventuelle innstillinger. På dette steget må en ta stilling til hvilken funksjonalitet som skal være med, hvor ting skal installeres, hvor tredje parts bibliotek befinner seg, og så videre.
 

```
./configure \
```

```
--with-apxs2=/usr/local/apache2/bin/apxs \  
--with-mysql=/usr/local/mysql \  
--with-gd
```

- Kompiler programmet ved å skrive `make`
- Installer programmet ved å skrive `make install`
- Konfigurasjonsfilen for PHP skal hete ***php.ini***, og opprettes med utgangspunkt i ***php.ini.dist*** (følger med kildekoden), slik:  

```
cp php.ini.dist /usr/local/lib/php.ini
```
- Apache kan håndtere HTML, men må også konfigureres til å gi PHP-programvaren i oppgave å håndtere PHP-script. Legg til følgende tre linjer i ***httpd.conf***:  

```
LoadModule php5_module /sti/til/libphp5.so  
AddType application/x-httpd-php .php  
AddType application/x-httpd-php-source .phps
```
- Restart Apache: `/usr/local/apache2/bin/apachectl restart`

### 1.2.8 Installasjon av PHP og Apache for Mac

Det er enkelt å ta i bruk PHP på Mac siden både Apache og PHP følger med Mac OS X. Alt du trenger å gjøre, er å slå på *webdeling* og *aktivere* PHP. Denne beskrivelsen gjelder for Mac OS X 10.3, som har PHP 4 installert. Det påpekes at bruk av PHP 4 fungerer bra til opplæringsformål og eksemplene i denne boka (foruten kapittel 13). Nyere utgaver av OS X vil trolig komme med nyere versjoner av PHP.

- For å slå på webdeling må du gå til *Systemvalg*, velge *Deling*, *Personlig webdeling* og *Start*. Dermed starter Apache. Alternativt kan du bruke et terminalvindu og skrive inn `apachectl start`.
- For å aktivere PHP må du være innlogget som administrator og redigere konfigurasjonsfilen til Apache. Åpne `/etc/httpd/httpd.conf`, og fjern #-kommentaren der det står  

```
#LoadModule php4_module libexec/httpd/libphp4.so  
og  
#AddModule mod_php4.c
```

I ***httpd.conf*** står det `php4` fordi det er PHP 4 som følger med Mac OS X 10.3.
- Restart Apache. Dette kan du enten gjøre ved å slå av og på webdelingen som vist i første punkt, eller ved å skrive `apachectl graceful` i terminalvinduet.

Versjonen av PHP kan oppgraderes om ønskelig. Se for øvrig <http://www.php.net/manual/en/install.macosx.php> for ytterligere hjelp og problemløsning.

Du må laste ned og installere MySQL selv. MySQL brukes først i kapittel 9. Installasjon av MySQL for Mac er ikke gjennomgått i denne boka, men prinsippene for tilpasning med PHP er den samme som for Windows, og du kan finne en Mac-versjon på samme nedlastingsside som beskrevet under Windows-installasjonen i kapittel 9.2.

## 1.2.9 Test om alt fungerer

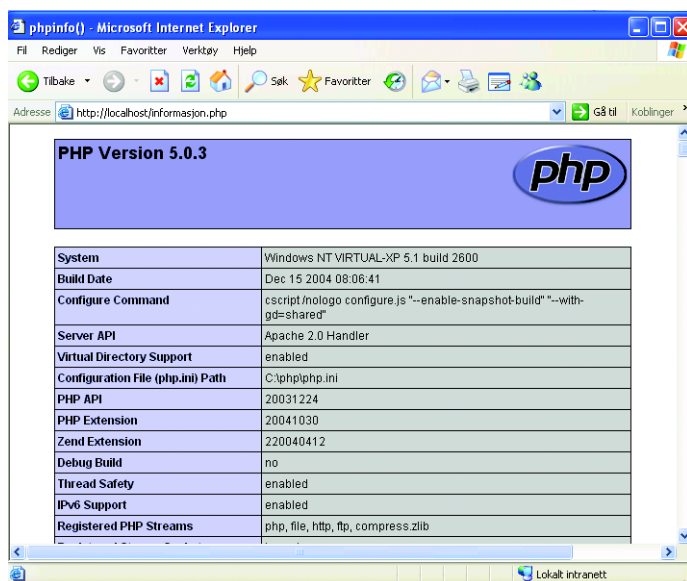
Du er nå klar til å teste om alt fungerer som det skal. Det er viktig at du lagrer **httpd.conf** før du starter Apache på nytt. Skriv følgende testscript:

```
<?php
    phpinfo();
?>
```

og lagre filen som **informasjon.php** i katalogen du har angitt som rot i **httpd.conf**. Når du skriver inn adressen

```
http://localhost/informasjon.php
```

i nettleseren, vil Apache behandle forespørselen. Apache ser at dette er en php-fil, og siden endelsen **.php** skal behandles av PHP (ifølge **httpd.conf**), gis kommandoen over til PHP. Resultatet som vises i nettleseren er som vist i figur 1.5. Det vises nå en veldig lang oversikt over innstillingene på tabellform som gjelder for PHP-versjonen som er installert. Du vil kunne kjenne igjen mange innstillinger fra filen **php.ini**, trolig er `display_errors` påslått og `register_globals` avslått.<sup>2</sup> Du vil også finne informasjon om Apache, både port-informasjon, hvor rota til webtjeneren befinner seg, og hvilken versjon av Apache som kjører. Vi kommer tilbake til **php.ini** jevnlig utover i boka.



Figur 1.5 Scriptet kjøres av PHP siden både PHP og Apache er satt riktig opp

<sup>2</sup> Endringen til `register_globals=off` i PHP-versjoner nyere enn 4.2.0, er gjort for å øke sikkerheten og unngå «forgiftning av variable». Du vil lære mer om dette i kapittel 4.3.5)

## 1.3 Samspillet mellom klient og tjener

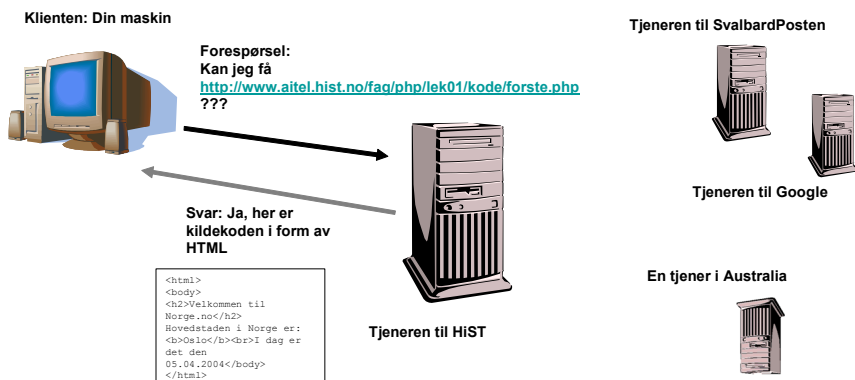
HTML står for Hyper Text Markup Language og har hatt en enorm betydning for veksten av Internett. Dette skyldes at HTML er enkelt å lære og logisk oppbygd. «Markup» betyr at formateringen av elementer angis ved å markere disse med spesielle *tagger* som nettleseren forstår. Med HTML kan Internett-sider med grafikk, lyd, tekst og andre elementer lenkes sammen. Når slike sider lagres på en tjenermaskin blir de tilgjengelige for hele verden. Bruk av HTML passer bra til statistisk informasjon som ikke endres ofte.

HTML spesifiserer altså *hvordan* informasjonen skal presenteres. Utvidet funksjonalitet, dynamikk og interaksjon med brukeren, krever derimot programmering. PHP bygger på to ting: At du forstår HTML og at du forstår programmering. Denne boka skal øve programmeringsforståelsen og vise hvordan PHP kan brukes til å lage fine webløsninger. Dersom du ikke allerede kan grunnleggende HTML er det lurt å gjennomgå det først. PHP og HTML er ikke konkurrenter. PHP bygger på HTML.

### 1.3.1 Klienten ber en tjener om å sende informasjon

*Klienten* er som regel den datamaskinen du (eller andre) benytter ved surfing på Internett. *Tjeneren* (ofte brukes det engelske navnet *server*) er en datamaskin med spesiell programvare som er plassert hos for eksempel Høgskolen i Sør Trøndelag (HiST). Tjeneren har den egenskapen at den har lagret informasjon som kan deles med andre. På nettet er det mange flere klienter enn tjenere.

Har du noen gang tenkt over hva som skjer når du besøker et nettsted? Du skriver enten inn en såkalt URL (Uniform Resource Locator) i nettleserens adressefelt eller klikker på en lenke (hyperreferanse). Det er nå opp til nettleseren å ta kontakt med den tjeneren hvor riktig nettsted ligger lagret, og be om å få tilsendt ønsket side. Vi sier at nettleseren på klienten sender en *forespørsel* til tjeneren, som så vil behandle forespørselen og returnere riktig informasjon. Til slutt vil nettleseren kunne vise innholdet for deg.



Figur 1.6 Basert på innskrevet URL vil klienten ta kontakt med riktig tjener på Internett, som forhåpentligvis kan returnere ønsket side

### 1.3.2 Et enkelt PHP-script

Et PHP-script kan skrives i en teksteditor så som «Notisblokk», «Textpad», «Front-Page», «Dreamweaver», «vi», «pico» og liknende. Microsoft Word skal aldri brukes til å programmere websider. Du vil finne lenker til flere gratis editorer på bokas hjemmeside.

Egentlig er det et skille mellom et script og et program, men i denne boka (og stort sett ellers i websammenheng) er dette skillet uvesentlig, og begrepene vil derfor brukes om hverandre. Et script vil tolkes linje for linje fra start til slutt. Hver gang scriptet skal kjøres må det tolkes på nytt. PHP er et scriptspråk hvor koden blir tolket hver gang den kjøres. Det betyr at koden kan produsere et resultat som kan vises selv om det skulle være feil i koden, i motsetning til kompilert kode.

Koden skal lagres i en vanlig fil med etternavnet *.php*. I Windows velger du «Lagre som...» i din editor. Dersom du ikke kan velge php som filendelse, (dette er for eksempel ikke blant valgene i programmet Notisblokk), velger du «Alle filer» fra nedtrekkmenyen og skriver inn ønsket filnavn og etternavnet *.php*. (Statiske HTML-sider har vanligvis *.html* eller *.htm* som etternavn.)

Neste script skal skrive ut to hovedsteder i en nettleser. Hvis du vil prøve selv, kan du skrive inn den etterfølgende koden i en ny fil i den teksteditoren du bruker, og lagre denne filen som *forste.php*. Om du vil plassere disse i en passende katalog, så sørg for at den er en underkatalog av det som er satt som `DocumentRoot`. Vi forklarer først samspillet mellom tjener og klient, og deretter syntaksen i koden. Dersom du ikke klarer å kjøre dette scriptet, kan det skyldes at PHP ikke er installert på din maskin. Du må i så fall installere nødvendig programvare (se kapittel 1.2).

## Kodesnutt 1.1 I scriptet første.php er HTML og PHP blandet sammen

```

<html>
<body>
<h2>Velkommen til Norge.no</h2>
<?php
    echo "Hovedstaden i Norge er: ";
    echo "<b>Oslo</b>";
    echo "<br>";
?>
I dag er det den
<?php
    echo date("d.m.Y");
?>
</body>
</html>

```

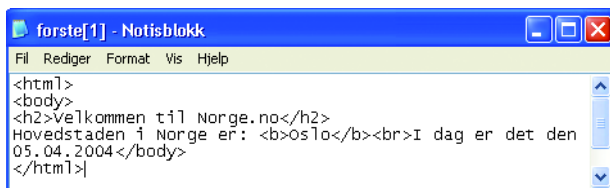
### 1.3.3 Resultatet vises i nettleseren

Legg merke til at HTML-taggene `<b>` og `</b>` forekommer inne i PHP-koden. Dersom du lagrer filen som beskrevet og skriver inn riktig adresse i nettleseren din, vil du få frem følgende resultat:



Figur 1.7 Resultatet av scriptet i kodesnutt 1.1

En titt på kildekoden bak denne Internett-siden (Velg «View Source» eller liknende funksjonalitet fra nettleseren, gjerne fra en kontekstsensitiv meny) viser at nettleseren bare har behandlet vanlig HTML:



Figur 1.8 Kildekoden avslører at bare HTML ble sendt tilbake fra tjeneren

Selv om en PHP-side etterspørres i nettleseren, vises altså bare vanlig HTML-kode. Hvor er det blitt av PHP-koden? Dessuten brytes ikke linjene i kildekoden fra nettleseren selv om vi lagde linjeskift i PHP-scriptet vårt. Hva har skjedd?

### 1.3.4 Tjeneren sender HTML-kode til klienten

Dette scriptet befinner seg på HiST under fagsidene til PHP, og kan nås på adressen:

<http://www.aitel.hist.no/fag/php/lek01/kode/forste.php>

Klienten ber i så fall tjenermaskinen som ligger bak adressen [www.aitel.hist.no](http://www.aitel.hist.no) om å utføre koden til filen **forste.php**, en fil som ligger i katalogen **fag/php/lek01/kode/** på AITeL sin tjener hos HiST i Norge. Merk at du gjerne kan legge filen lokalt på din egen maskin om du har riktig programvare installert. Da vil adressen bli for eksempel <http://localhost/forste.php>.

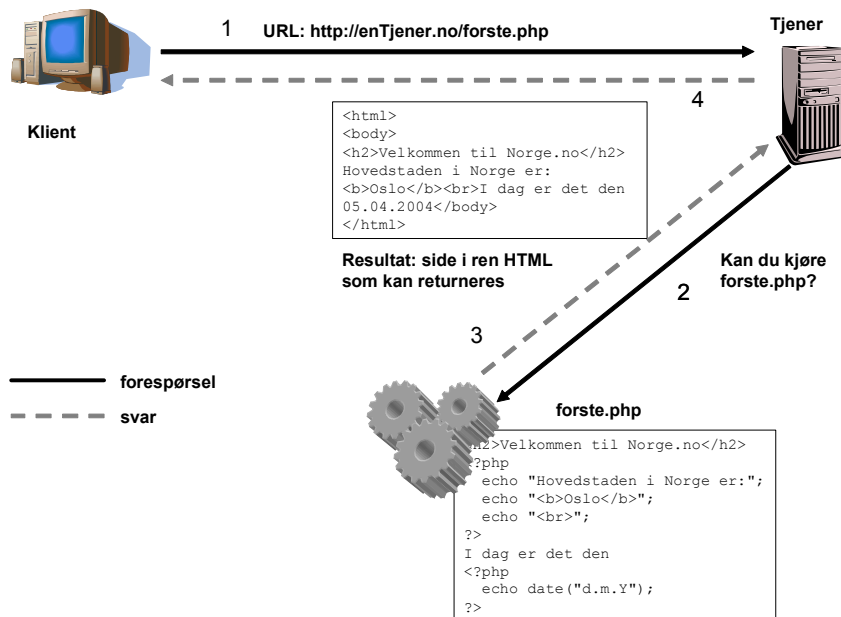
Hemmeligheten til at koden utføres på tjeneren som et script, ligger i etternavnet **.php**. Filer med etternavn **.html** vil returneres direkte slik de er, mens alle PHP-script vil prosesseres før resultatet returneres. Etter at koden er utført på HiST sin tjener vil altså resultatet sendes tilbake til klienten i form av vanlig HTML, og det er dette vi får frem på skjermen i nettleseren. Det er viktig å forstå at det er *resultatet*, og ikke selve scriptet, som returneres.

Den observante leser aner kanskje allerede nå at nettopp her ligger nøkkelen til dynamikk: Dersom tjeneren kjører programmet rett etter at en person har besøkt siden, og resultatet vises umiddelbart deretter, er det jo mulig å programmere Internett-sider slik at hver enkelt bruker får forskjellig side presentert på skjermen basert på visse kriterier som angitt i programmet! Ja, dette er riktig, og PHP har i tillegg mange andre interessante bruksområder som vi skal se utover i boka.

Det kan være nyttig å bevisst tenke på samspillet mellom klient og tjener når en programmerer. I gjennomgangen av installasjonen ble det påpekt at en form for PHP-programvare må være installert på tjeneren, og at tjeneren, for eksempel Apache, vil videresende forespørsler etter PHP-script til PHP-programvaren (egentlig PHP-tolkeren eller bare *tolkeren* som vi ofte vil si utover i boka).

Først sendes en forespørsel etter siden **forste.php** på riktig tjener (steg 1). Tjeneren mottar forespørselen og ser at dette er en oppgave for PHP-tolkeren (basert på for eksempel innstillingene i `httpd.conf` gjennomgått i kapittel 1.2.5, hvis Apache brukes). Tolkeren kjører scriptet (steg 2). Resultatet av kjøringen samles i en strøm av HTML (steg 3), som tjeneren sender tilbake til klienten (steg 4). På grunn av at all prosessering skjer på tjeneren, trenger ikke nettleseren å forstå noe annet enn vanlig HTML, og kan uten problem vise siden fra figur 1.7. Sett med nettleserens øyne er dette statisk informasjon, men brukeren vil oppleve siden som høyst dynamisk i og med at dagens dato vises.

Teorien vi har gått igjennom til nå er summert opp i figur 1.9.



Figur 1.9 Samspillet mellom klient og tjener

Hvis du bruker localhost, blir prinsippene for kommunikasjonen likedan. Forskjellen er at både klienten og tjeneren ligger på en og samme maskin. Det er derfor det er nødvendig å starte opp riktig programvare (for eksempel Apache og PHP) før du tester dine PHP-script.

### 1.3.5 PHP og HTML kan blandes

En forklaring til scriptet *forste.php* er på sin plass. Koden i kodesnutt 1.1 så slik ut:

```
<html>
<body>
<h2>Velkommen til Norge.no</h2>
<?php
    echo "Hovedstaden i Norge er: ";
    echo "<b>Oslo</b>";
    echo "<br>";
?>
I dag er det den
<?php
    echo date("d.m.Y");
?>
</body>
</html>
```

Først i scriptet kommer helt vanlig HTML-kode, og dette gir akkurat samme resultat tilbake. Når taggen `<?php` støtes på av tolkeren, vet den at alt som forekommer inntil `?>` er PHP-kode. Merk at det er vanlig å rykke inn koden med tabulator eller mellomrom for å få bedre oversikt, men at dette ikke er noe krav.

Den første setningen etter `<?php` starter med kodeordet `echo` og utgjør scriptets første *instruksjon*, eller *linje* som vi også sier. Hver instruksjon i PHP avsluttes av et semikolon, og dermed ser vi at scriptet totalt sett består av fire instruksjoner mellom HTML-koden.

Det som står omsluttet av anførselstegn er tekst. Tekst som står etter `echo` blir skrevet ut, og det betyr at PHP i første instruksjon skriver ut teksten "Hovedstaden i Norge er: ", der både kolon og mellomrommet etter kolonet er en del av tekststrengen.

Neste instruksjon skriver ut teksten "`<b>Oslo</b>`", hvilket vi gjenkjenner som HTML-taggen for å lage fet skrift. Her er essensen i samspillet mellom PHP og HTML: Når tolkeren er ferdig med å prosessere scriptet den har mottatt, vil siden som er klar for retur bestå av ren HTML (tekst og tagger), og elementet innrammet i taggen `<br>` vil deretter føre til at klienten korrekt viser fet skrift i nettleseren.

Det er enda en detalj som er verdt å merke seg. Selv om det er linjeskift i koden, blir det ikke linjeskift mellom "Hovedstaden i Norge er: " og "Oslo" når resultatet skal vises i nettleseren. Grunnen til dette er at filen ikke tolkes som en tekstfil med skjulte linjeskift, men som en HTML-fil hvor linjeskift må markeres eksplisitt med for eksempel `<br>` taggen. I eksempelet vårt har vi forutsatt at dette ville bli et problem mellom Oslo og teksten "I dag...".

Når tolkeren møter på taggen `?>` er den ferdig med å utføre PHP, og hopper tilbake til HTML-modus. Linjen "I dag er det den" vil derfor vises som vanlig HTML direkte, men så støter tolkeren på `<?php` og må hoppe tilbake til PHP-modus enda en gang. Linjen

```
echo date("d.m.Y");
```

betyr at det som står bak `echo` skal skrives ut til skjerm. Hva er det? Funksjonen `date()` returnerer informasjon om dagens dato. Denne er innebygd i PHP, og når "d.m.y" står inne i parentesene vil resultatet bli at dagens dato på formen dag.måned.år, skrives ut. Se tilbake på figur 1.7 om du er i tvil. Du vil lære mer om datofunksjoner i kapittel 6.

### 1.3.6 Noen detaljer

Følgende tre setninger gir akkurat samme resultat:

```
echo "tekst her";
echo("tekst her");
print("tekst her");
```

Hvilken av de tre du velger å bruke er en smaksak, og du må gjerne bruke alle i samme kodesnutt.

En fin ting med PHP er at det er så visuelt, noe vi har illustrert med echo-funksjonaliteten. Resultatet vises i nettleseren. Det er derimot også mulig å programmere funksjonalitet som ikke gir noe synlig resultat tilbake i nettleseren. Dette er nyttig hvis trafikk om antall besøkende og deres handlinger skal logges. Slik informasjon kan lagres i databaser eller filer, og bli nyttig for å legge strategi for videreutvikling av et nettsted. Databaser og filer brukes også for skape interaktivitet og bruker-vennlighet. Detaljene kommer senere i boka.

Det nevnes også at det er mulig å bruke følgende syntaks for å markere hva som er PHP-kode:

```
<script language="php"> PHP-koden kommer her </script>
```

Dersom instillingene `short_open_tag` og/eller `asp_tags` er slått på i `php.ini`, er det lov å bruke henholdsvis kortnotasjon og ASP-notasjon. Det betyr at du i PHP-script på Internett kan komme over følgende fire måter å gjøre ting på:

```
<?php kode her, standard måte å gjøre det på ?>
<? kode her, dette er kortnotasjonen ?>
<% kode her, dette er slik det gjøres i ASP %>
<script language="php"> PHP-koden kommer her </script>
```

PHP-tolkeren aksepterer at det veksles mellom HTML- og PHP-modus, og det gir mulighet til å skrive for eksempel:

```
<h2>Velkommen hit, <?php echo $navn ?></h2>
```

Følgende hurtigsyntaks er mulig å bruke dersom `<?>` er satt opp som gyldig starttag i *php.ini*:

```
<h2>Velkommen hit, <?= $navn ?></h2>
```

Med andre ord betyr `<?=` og `<?php echo` akkurat det samme. Det er fristende å bruke kortnotasjonen, men du kan risikere at en fremtidig ISP som skal ha Internett-sidene dine, ikke har slått på dette valget i *php.ini*, og dermed vil ikke scriptene kjøre som de skal.

Vi bruker notasjonen

```
<?php
kode kommer her...
?>
```

gjennomgående i denne boka, med noen få unntak.

### 1.3.7 Kommentarer i PHP

Alt som står på en linje bak `//` oppfattes som en kommentar av tolkeren, og vil dermed ikke bli utført. Det betyr at verdifulle kommentarer kan skrives for å doku-

mentere et script. Kommentarer over flere linjer følger for øvrig samme standard som C, C++ og Java, slik:

```
/* kommentar over
mange linjer eller midt i en linje,
tolkes ikke av PHP
*****!!#**** alle tegn kan brukes, helt til
sluttsekvensen kommer, nemlig */

//Her er en kommentar på bare en linje
echo "Velkommen"; //kommentar etter semikolon er også lov
#Dette er også en kommentar som det er lov å bruke
```

### 1.3.8 PHP vs. JavaScript

HTML spesifiserer *hvordan* informasjonen skal presenteres. Vi har nevnt at løsninger som bygger på HTML er statiske. Utvidet funksjonalitet, dynamikk og interaksjon med brukeren, krever programmering.

*JavaScript* er et scriptspråk som tilbyr en del interaktivitet og dynamikk. Alle script som er skrevet i JavaScript *utføres på klienten*, i motsetning til script skrevet i PHP som *utføres på tjeneren*. Det vil si at koden i scriptfilen blir kjørt av et program på tjenermaskinen hver gang siden blir forespurt. Tjeneren vil returnere *resultatet* til klienten for presentasjon, og dette resultatet vil typisk være i form av HTML.



Hva er best, å utføre et script på tjeneren eller klienten? Begge metodene har fordeler og ulemper, og helt klart egnede bruksområder. Du vil i løpet av boka få se mange eksempler hvor JavaScript inngår i PHP-scriptene for å gi ytterligere dynamikk og utnytte de sterke sidene til kjøring på klientsiden. Det er ikke bare mulig, men ofte veldig nyttig å kjøre begge variantene sammen.

## 1.4 Oppgaver og kontrollspørsmål

### Oppgave 1-1: Innstillinger

Utforsk `php.ini` dersom du har tilgang til denne.

- Sjekk hva innstillingene `display_errors` og `register_globals` er satt til.
- Kan du finne innstillingene på en annen måte enn å åpne `php.ini` i en teksteditor? Kan du i så fall se for deg noen situasjoner hvor dette kan være nyttig?

### Oppgave 1-2: Forståelse av hvordan PHP fungerer

Sørg for at setningen

```
AddType application/x-httpd-php-source .phps
```

er aktivert/lagt til i konfigurasjonsfilen til Apache (`httpd.conf`). Kopier koden fra eksempelet i kodesnutt 1.1 og lagre to versjoner med samme filnavn, der den ene for eksempel heter `norge.php` og den andre `norge.php.s`. Filene skal ha nøyaktig likt innhold.

- Skriv inn adressen til de to scriptene i to nettleservinduer og sammenlikn både resultatet og kildekode (velg funksjonalitet for å vise HTML-kildekode i nettleseren).
- Forklar forskjellen, gjerne med basis i samspillet mellom klient og tjener, og si noe om når denne teknikken med visning av scriptkoden kan være nyttig.
- Legg til passende kommentarer i koden du nettopp har lagd.

### Oppgave 1-3

Gå til et hvilket som helst nettsted. Se deg ut en side som du vil kopiere, vis kildekode i nettleseren, og lagre din egen variant av siden på din maskin/ditt område.

- Legg til dagens dato og ditt navn på et passende sted og observer resultatet.
- Lokaliser en side på Internett som slutter på `.php`. Hvorfor er det ingen PHP-kommandoer i kildekode? Forklar.
- Hva er fordelene med PHP som programmeringsspråk?
- Kan JavaScript og PHP brukes sammen? Hvorfor (ikke)?