

- ▶ Generel kursusinformation
 - ▶ Kursusindhold og motivation
 - ▶ Eksamen og ugentlige øvelser
- ▶ Web-publicering og Web-baserede services
- ▶ Udvikling af statiske sider
- ▶ Versionskontrol
- ▶ Detaljeret kursusindhold
 - ▶ Basal HTML på 15 minutter
 - ▶ Nogle web-design regler
 - ▶ HTML validering
 - ▶ Eksempel på SQL

- ▶ **Kursusmål:** Lær at anvende avancerede web-teknikker til at udvikle realistiske, brugbare web-sites, der involverer udveksling af information med en database.
- ▶ **Undervisere:** Martin Elsman (mael@itu.dk, kontor 4C02) og Jonas Holbech (holbech@itu.dk)
- ▶ **Hjælpelærere:** Se kursushjemmesiden
- ▶ **Kursets hjemmeside** er <http://www.itu.dk/courses/DSDS/E2007>
- ▶ **I skal holde øje med hjemmesiden hele semestret igennem**
- ▶ **Hvornår:** forelæsning torsdag 17.00-19.00; øvelser torsdag eftermiddag (15.00-17.00) og aften (19.00-21.00)

- ▶ **Lærebog:**

- ▶ "Introduktion til PHP, MySQL og Apache". Julie C. Meloni. September 2003.

- ▶ **Noter**, der udskrives fra hjemmesiden:

- ▶ *HTML overview.*, Peter Sestoft.

- ▶ **Relaterede Noter**

- ▶ *Start på PHP*, Thomas G. Kristensen. IDG Forlag 2001.
- ▶ *Start på SQL*, F. D. Rolland. IDG Forlag 1998.

- ▶ Konto på IT-Universitetets netværk.

- ▶ Man lærer kun ved for alvor at prøve selv.
- ▶ Læsevejledning: Læs lærebog og noter forud for hver forelæsning. **Løs opgaverne** i god tid.
- ▶ Øvelser:
 - ▶ Dagtimer: torsdag 15.00-17.00
 - ▶ Aftentimer: torsdag 19.00-21.00
- ▶ Øvelsesbesvarelser afleveres senest ugen efter øvelsen.
- ▶ Besvarelserne rettes den efterfølgende uge.

- ▶ Eksamen er 4-timers skriftlig eksamen; alle ikke-elektroniske hjælpemidler er tilladte.
- ▶ Øvelserne løses og afleveres individuelt.
- ▶ Øvelserne er **ikke** obligatoriske, men lav dem alligevel.
- ▶ Øvelserne dækker:
 - ▶ Statisk **HTML**
 - ▶ Web-programmering med **PHP**
 - ▶ Databaseprogrammering med **SQL**
 - ▶ **Konstruktion** af websider med HTML, PHP og SQL
- ▶ **Pensum** er lærebogen (angivne sider i forelæsningsplanen), angivne noter og artikler i forelæsningsplanen, øvelsessæt samt forelæsningsplancerne. Justering kan forekomme.

Web-publicering kræver generalister:

tekstbehandling, fotografering, publicering, system administration, databasekendskab, forståelse for brugergrænseflader, programmeringserfaring, kendskab til designmetodikker,

Udgangspunkt: Statiske sites (HTML)

Hvordan kan man konstruere mere interessante sites:

- ▶ Sites som er programmer (dynamisk HTML)
 - ▶ Beregning af skat
 - ▶ Bill Gates personal wealth clock
(<http://philip.greenspun.com/WealthClockIntl>)
- ▶ Sites som er databaser (dynamisk HTML med databasetilgang)
 - ▶ ITU's projektbase
 - ▶ CourseGrader (<http://hug.it.edu:8002/vu/index.tcl>)
 - ▶ Amazon (<http://www.amazon.com>)

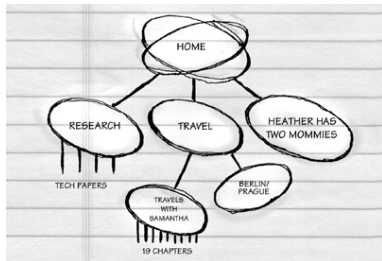
Populære og brugbare sites kan bygges med få midler!

- ▶ Web-publicering er ofte kendetegnet ved et magasinagtigt indhold, f.eks. produktkataloger, avistjenester m.m.
- ▶ Web-services er kendetegnet ved at løse et problem for en bruger, f.eks. huskeliste, kommentarservice, bogkøb, billetbestilling m.m.
- ▶ Start med at spørge hvad en bruger kan være interesseret i!
- ▶ Hvad kan du gøre for at berige en bruger?
- ▶ Undgå:
 - ▶ user-modeling, entry tunnel, exit tunnel
 - ▶ flashy langsom grafik
- ▶ Understøt flere syn på indholdet (eng. *Multiple Views*):
 - ▶ Tillad f.eks. brugerkommentarer...

- ▶ **Fra desktop** applikationer til web-baserede programmer
- ▶ **Eksempler:**
 - ▶ Groupcare (<http://www.groupcare.com>)
 - ▶ Kalendersystem
 - ▶ Tekstbehandling/versionskontrol
 - ▶ CourseGrader
 - ▶ Kursusevaluering
 - ▶ Intranets
 - ▶ ...
- ▶ **Fejl** i web-baserede programmer **kræver** og **tillader** rettelser hurtigt!

Udvikling af Statiske Web-sites

- ▶ Tegn en site-oversigt (*site map*).
- ▶ Struktur indhold; navgiv filer.
 - ▶ *overvej hvorledes billeder m.m. skal organiseres*
- ▶ Konstruer et “kun-tekst” site.
- ▶ Ansæt en grafisk designer!?
- ▶ Konstruer vedligeholdelsesplan. Hvordan opdateres indhold?
 - ▶ *kan ansvaret uddelegeres til dem, der skriver indholdet, eller skal man have en central hjemmesideafdeling?*
- ▶ Lav periodiske brugertests.



Problem:

- ▶ **Hans** henter **Version A** af et dokument kl. 09.00 fra web-sitet og bruger en dag på at editere det.
- ▶ **Grete** henter **Version A** kl. 12.00 og retter en fejl.
- ▶ **Grete** skriver dokumentet (**Version B**) til serveren kl. 12.10.
- ▶ **Hans** skriver sit dokument (**Version C**) til serveren kl. 17.00.

Gretes rettelse er glemt! **Version C** indeholder ikke rettelsen i **Version B**!

Løsninger:

- ▶ “låsning” af filer med emails, CVS, Subversion (SVN),
- ▶ CVS: <http://www.cvshome.org/>

Tre teknologier

- ▶ HTML (Hyper Text Markup Language)
- ▶ PHP (PHP Hypertext Preprocessor)
- ▶ SQL (Structured Query Language)

Hvorfor lære programmering?

- ▶ Bedre forståelse for **begrænsninger** og **muligheder** indenfor udvikling af dynamiske websider.
- ▶ Nødvendig betingelse for udvikling af avancerede websider.

Hvorfor lære PHP?

- ▶ PHP er velegnet til udvikling af dynamiske web-sites (ligesom Perl, ASP, Java, Standard ML, ...).
- ▶ PHP er velegnet til behandling af strenge.
- ▶ PHP er frit tilgængelig!
- ▶ PHP supporteres af et flertal af web-hosting firmaer.

Specielle (domain-specifikke) sprog: HTML og SQL

HTML: Beskriver hyperteksters struktur og (til dels) layout.

Eksempel (<html/hello.html>):

```
<html>
  <head><title>Hello</title></head>
  <body>Hello You</body>
</html>
```

Browsere er normalt ikke pedantiske overfor korrekt HTML — men I skal skrive korrekt HTML alligevel!

SQL: Sender data frem og tilbage mellem et program og en database.

Ex 1: `select navn, adr from navne order by navn;`

Ex 2: `insert into navne (navn, adr) values ('Hans', 'Byvej 43');`

Der findes et utal af SQL-dialekter — men det er normalt trivielt at forstå forskellene.

Hvorfor interessere sig for HTML-koder?

- ▶ For at være sikker på at ens websider virker i alle browsere
- ▶ Fordi der er ting man ikke kan lave med sin WYSIWYG HTML-editor
- ▶ For at genbruge andres gode ideer
- ▶ WYSIAYG = What You See Is All You Get
- ▶ For web-programmering: I web-programmering skriver man programmer der *genererer* HTML; det kræver at man forstår HTML

- ▶ En HTML-editor vil altid generere korrekt HTML (eller *burde* altid)
- ▶ Man kan lære HTML af den (men pas på unødvendige eller browserspecifikke koder)
- ▶ Det er meget nemmere at komme igang
- ▶ Eksempler på HTML-editorer: Microsoft Frontpage, Word, Netscape Composer, DreamWeaver. . .

Basal HTML på 15 minutter

Et *validerende* HTML-dokument (html/legalt.html):

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head><meta http-equiv="Content-Type"
    content="text/html; charset=ISO-8859-1">
    <title>Mit Site</title></head>
  <body>
    <h2>Mit Site</h2>
    Se siden for <a href="http://www.itu.dk/courses/DSDS/E2007"
      >Scripting, Databaser og Systemarkitektur</a>
  </body>
</html>
```

- ▶ Det kan være svært at skrive pæn HTML — men forsøg alligevel.
- ▶ Skriv HTML-koden selv!

Du bør kende følgende tags i HTML:

Overskrifter	<code><h1>, </h1>, ..., <h4>, </h4></code>
Skillelinier	<code><hr></code>
Afsnit og ny linie	<code><p>, </p></code> og <code>
</code>
Quotes	<code><blockquote></code> og <code></blockquote></code>
Centrering	<code><center></code> og <code></center></code>
Fed	<code></code> og <code></code>
Kursiv	<code><i></code> og <code></i></code>
Understregning	<code><u></code> og <code></u></code>
Ordrede lister	<code>, </code>
Uordnede lister	<code> </code>
Listeelementer	<code></code> og <code></code>

- ▶ Links `navn`
- ▶ Mailto-links `nh@itu.dk`
- ▶ Tabeller med `<table>`, `</table>`, `<tr>`, `</tr>`, `<th>`, `</th>`, `<td>` og `</td>`
- ▶ billeder ``
- ▶ de mest almindelige attributter, såsom `bgcolor`

Find fem fejl

I et HTML-dokument uden konsistent indryk:

```
<html><head><title>Hello</title></head>Hello You<center>great  
page<p> this is in <b>bold<i> and italics</html>
```

I et HTML-dokument med indrykning:

```
<html>  
  <head><title>Hello</title></head>  
  Hello You  
  <center>  
    great page  
  <p>  
    this is in <b>bold<i> and italics  
</html>
```

Firefox viser dette dokument uden fejl! Eller gør den?

Problemer med fancy-tags, såsom FONT:

- ▶ Gamle browsere ignorerer dem
- ▶ Nye browsere ignorerer dem
- ▶ Når du udskifter den grafiske designer skal du ændre 10.000 HTML-sider!
- ▶ Istedet: brug Cascading Style Sheets (CSS).

Mobil-brugere kan ikke lide for megen grafik!

Flere HTML Regler

- ▶ Undgå store tabeller, til f.eks. side-layout (giver problemer med hastighed).
- ▶ Undgå redefinition af linksfarver (folk forbinder blå med et link).
- ▶ Sørg for at din side kan udskrives på printer (f.eks. udskriver Netscape hvid tekst på blå baggrund som hvid tekst på hvid baggrund!)
- ▶ Tænk på din brugers skærm som en begrænset ressource.
- ▶ Undgå frames (virker dårligt sammen med email, printer og bogmærker).
- ▶ Husk at give dine HTML sider en titel (til bl.a. bogmærke).

En målsætning kunne være, at både Firefox samt Internet Explorer kan vise din side som du forventer.

Hvad med PDAere og mobiltelefoner, . . .

- ▶ Brug `http://validator.w3.org` til at validere din HTML
- ▶ Almindelig fejl nummer 1: **Ingen DOCTYPE**

Tilføj følgende, som den første linie i dit dokument:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">
```

- ▶ Almindelig fejl nummer 2: **Ingen "Character Encoding"**

Tilføj følgende, som det første element i dit `<head>` element:

```
<meta http-equiv="Content-Type"  
content="text/html; charset=ISO-8859-1">
```

Bemærk at det kan være nødvendigt at benytte `utf-8` som karaktersæt.

Flere Web-design Regler

- ▶ Informationen skal være brugergrænsefladen: ved klik på informativ ord, bringes man til uddybende information (undgå “her”-links)
- ▶ Giv brugere et bredt, flat overblik af information, istedet for et sekventielt informationsforløb
- ▶ Organiser dit site efter forventede brugerinteresser, istedet for efter den interne organisation.
- ▶ Hvorfor bruge ikoner til at navigere, når ord er mere sigende og fylder mindre?
- ▶ Længere dokumenter fremfor brede.
- ▶ Husk at linke fra dine sider med indhold tilbage til en hovedside (indeks) — bl.a. p.g.a. søgemaskiner.
- ▶ Husk kontaktinformation på dine sider

Eksempel på SQL

Givet en tabel student:

efternavn	fornavn	studienummer
Olesen	Peter	L2143
Hansen	Erika	J0007
Funder	Ulrik	Hg0014

Vi kan forespørge på fornavn og efternavn:

```
select fornavn, efternavn from student order by efternavn;
```

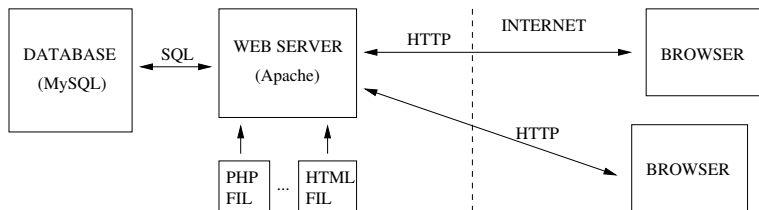
Vi kan indsætte en ny række:

```
insert into student (efternavn, fornavn, studienummer)
values ('Fong', 'Joe', 'JF0032');
```

Vi kan slette en række:

```
delete from student
where efternavn = 'Fong';
```

Brug af server-side web-scripting (programmer der afvikles på web-serveren)



1. En klient henter en HTML side med en HTML formular fra en web-server.
2. Klienten indtaster data i HTML formularen, og sender data tilbage til web-serveren.
3. Web-serveren afvikler et program, som behandler data fra klienten, bl.a. ved at opdatere en database.
4. Web-serveren sender HTML tilbage til klienten.
5. Klienten fortolker HTML-koden og viser resultatet i browseren.

Opgaverne

- ▶ HTML-hjemmeside
- ▶ HTML-kursusoversigt med tabel og links
- ▶ Validering af dine sider med W3C validator

Udskriv følgende

- ▶ Øvelsen:
`http://www.itu.dk/courses/DSDS/E2007/ps/ps1.html`
- ▶ HTML-oversigt:
`http://www.dina.kvl.dk/~sestoft/databehandling/html1.html`