Linux - Friheden til at skrive dokumentation

Version 2.7.20040524 - 2020-12-31

Peter Toft

Henrik Grove

Kristian Sørensen

Michael Rasmussen

Linux - Friheden til at skrive dokumentationVersion 2.7.20040524 - 2020-12-31

af Peter Toft, Henrik Grove, Kristian Sørensen og Michael Rasmussen

Dette er en bog om at skrive bøger i Dokumentation i Linux. Installation af SGML programmel på Linux Red Hat 7.2 gennemgås, og derefter vises, hvordan man skriver bøger og oversætter disse til HTML, PDF, RTF eller Postscript.

Indholdsfortegnelse

Tabelliste

Forord

Når man skal skrive bøger så kan man vælge mange forskellige måder. I denne bog vil vi gennemgå nogle af dem. Hvis man er typen der vil arbejde med teksten præcist som den kommer til at se ud på tryk, så skal man have fat i Abiword, KWord eller OpenOffice.org, der ikke bliver beskrevet i denne bog. Vi vil beskrive systemene DocBook/XML, HTML og LaTeX, samt nogle værktøjer der lettere at bruge dem. Disse tre systemer har til fælles at de bruger almindelige tekstfiler som grundlæggende format. DocBook/XML er designet til at producere brugsanvisninger i mange forskellige slutformater (HTML, 'info', 'man', PDF og Postscript). HTML er *formatet* på WWW. Det er modsat de to andre formater vi omtaler her ikke beregnet til at blive konverteret til andre filformater. LaTeX er designet til at lave trykte bøger (via DVI-, PDF- og Postscript-formaterne), men kan også oversættes til bl.a. HTML og RTF. En anden ting der kendetegner LaTeX er at det nok er det bedste system der findes til at skrive formler med[1].

DocBook/XML har ca. 250 forskellige elementtyper og er primært beregnet på at skrive om EDB-relaterede emner. I DocBook/XML definerer man ikke hvordan teksten skal se ud, men kun hvad teksten er for en type. Med DocBook/XML kan man oversætte til HTML, Postscript, PDF og RTF. Man skriver (som med LaTeX og HTML) dokumentet i klar tekst med mærker der viser hvor elementer af forskellig type starter og slutter. Derefter oversætter man dokumentet til det format man ønsker. Når så DocBook-dokumentet oversættes til for eksempel HTML, bliver teksten formateret på en bestemt måde. En elementtype er for eksempel "`para`", der bruges til at markere et afsnit tekst. Skal man angive en forfatters navn placerer man det i et "`author`"-element. Ved at al tekst står et bestemt sted i et hierarki af elementer, kan man senere vælge, ikke bare hvordan teksten skal se ud, men også hvilke dele skal med.

Fordelen med DocBook/XML er at det er ret nemt at skrive i, idet de elementer man anvender i praksis er begrænset til måske 20, den anden store fordel er, at man ret nemt kan oversætte både til HTML og tilsvarende udskrift-formater såsom PDF og Postscript. Den tredje store fordel er at krydsreferencer og stikordsregister også er ret nemme at administrere. Til store bøger (uden formler) er DocBook/XML et særdeles interessant system.

1. Linux-bøgerne

Bogen er en del af en serie, som kan findes på http://www.linuxbog.dk/

- *Linux – Friheden til at vælge installation* – Om at installere Linux.
- *Linux – Friheden til at lære Unix* – Om hvordan man bruger Linux' (og Unix') kommandolinjeværktøjer.
- *Linux – Friheden til at vælge grafisk brugergrænseflade* – Om alle de grafiske brugergrænseflader, der findes til Linux.
- *Linux – Friheden til at vælge programmer* – Om de programmer du kan få til Linux.
- *Linux – Friheden til systemadministration* – Om at administre sit eget linuxsystem.

- *Linux – Friheden til at programmere* – Programmering på Linux
- *Linux – Friheden til at programmere i C* – Om at programmere i sproget "C".
- *Linux – Friheden til at programmere i Java* – Om at programmere i sproget "Java".
- *Linux – Friheden til sikkerhed på internettet* – Om at sikre dit Linuxsystem mod indbrud fra internettet.
- *Linux – Friheden til egen webserver* – Om at sætte en webserver med databaser, CGI-programmer og andet godt op.
- *Linux – Friheden til at skrive dokumentation* – Om at skrive dokumentation (og andet) i SGML/DocBook, LaTeX eller andre formater.
- *Linux – Friheden til at vælge kontorprogrammer* – Kontorfunktioner på et Linux/KDE/OpenOffice.org-system.
- *Linux – Friheden til at vælge IT-løsning* – Om muligheder, fordele og ulemper ved at bruge Linux i sin IT-løsning.
- *Linux – Friheden til at vælge OpenOffice.org* – Om at bruge OpenOffice.org, både på Linux og på andre styresystemer.
- *Linux – Friheden til at vælge digital signatur* – Digital signatur på Linux.

2. Ophavsret

Denne bog er skrevet af Linux-brugere til Linux-brugere. Store dele af bogen er skrevet eller redigeret af enkelte forfattere, hvilket er nævnt i revisions-historien til bogen.

Bogen kan findes i opdateret form på http://www.linuxbog.dk/, mens prøve-udgaver kan findes på http://cvs.linuxbog.dk/.

Figur 1. ÅDL

Kommentarer, ris og ros og specielt fejl og mangler bedes sendt til linuxbog@sslug.dk (mailto:linuxbog@sslug.dk), men er du medlem af SSLUG kan du i stedet for med fordel skrive til sslug-bog@sslug.dk (mailto:sslug-bog@sslug.dk).

3. Om forfatterne og bogens historie

Figur 2. Peter Toft

Peter Toft er civilingeniør og har en Ph.D.-grad fra DTU. Han har kørt Linux dagligt siden 1994. Aktiv Linuxforfatter og foredragsholder om Linux i Danmark og Sverige. Tidligere bestyrelsesformand for SSLUG. Medorganisator ved de danske Linux-konferencer (Linux98, Open Networks 99, LinuxForum 2000-2003 og Guadec2). Hjemmeside http://pto.linux.dk

Figur 3. Henrik Christian Grove

Henrik Christian Grove er uddannet som matematiker på Københavns Universitet. Har brugt Linux siden sommeren 1997 og har været aktiv deltager i de fleste af SSLUG's arrangementer siden Open Networks 99. Begyndte at bruge LaTeX i foråret 1996 hvor han blev medredaktør på de matematikstuderendes fagblad Famøs. Siden da har han skrevet stort set alt i LaTeX.

Kristian Sørensen har bidraget med kapitlet om LaTeX, som Henrik Christian Grove sidenhen har omskrevet dele af.

4. Vi siger tak for hjælpen

Vi har haft stor glæde af mange SSLUG-medlemmers støtte, rettelser og forslag til forbedringer - bliv ved med dette. Specielt vil vi nævne:

- Kristian Sørensen har skrevet det originale kapitel om LaTeX.
- Andre bidragydere er: Anna Jonna Armannsdottir, Erik Martin, Erik Sørensen, Erling Sjørlund, Harry Jensen, Henrik Skov Midtiby, Søren Ulrik, Mads Sejersen, Philip Heede, Jørgen Kristensen, Jesper Laisen, Stig Jensen, Gunner Poulsen, Simon-Shlomo Poil.

Du kan i Appendiks B finde en liste over alle de revisioner, som bogen har været igennem.

Hvis du har ord du ikke forstår, så kan http://www.whatis.com være interessant. Her kan du slå mange computerord op dog kun på engelsk.

5. Typografi

Vi vil afslutte indledningen med at nævne den anvendte typografi.

- Navne på filer og kataloger skrives som `foo.bar`
- Kommandoer, du udfører ved at taste, skrives som **help**
- Der er flere steder i bogen, hvor vi viser, hvad brugeren taster, og hvad Linux svarer. Det vil se ud som:

  ```
  [tyge@hven ~]$ Dette taster brugeren
  Dette svarer Linux.
  ```

- Der er tilsvarende flere steder i bogen hvor vi viser hvad systemadministratoren (root) taster, og hvad Linux svarer. Det vil se ud som:

  ```
  hven# Dette taster systemadministratoren
  Dette svarer Linux.
  ```

  Det vigtige her er at kommandofortolkeren bruger nummertegnet (#) til at markere at man har systemadministratorrettigheder.

Slutbemærkning:

1. Blandt fysikere og matematikere er det sågar udbredt at bruge LaTeX' notation, når man skriver formler i e-post.

Kapitel 1. HTML

Det er ofte smart at skrive dokumentation i HTML, fordi HTML forstås på alle platforme. Ideen med HTML er, at man skriver en fil i ren tekst og tilføjer nogle mærker der afgrænser elementer i teksten med bestemte typer oplysninger. Det kan for eksempel være overskrifter eller henvisninger til illustrationer. Det er vigtigt at bemærke at HTML principielt slet ikke beskriver *hvordan* teksten skal vises, men kun hvilken slags tekst der er tale om. Hvis du vil styre hvordan dine HTML-dokumenter skal vises, skal du bruge CSS (stilark).

Da de fleste programmer der viser HTML (primært netlæsere) på hver deres måde forsøger at kompensere for eventuelle fejl i et HTML-dokument, er det vigtigt at man — inden offentliggørelsen — kører sine HTML-filer gennem en egentlig HTML-syntakstjekker for at sikre sig at syntaksen er korrekt og at de forskellige programmer dermed ikke kommer med forskellige gæt på hvad du egentlig mente. "HTML Tidy" er en praktisk HTML-syntakstjekker og -kodepynter. Hvis du ikke allerede har "HTML Tidy" på dit system (det kan du undersøge med kommandoen **tidy -v**) bør du installere det inden du fortsætter.

Der findes to typer mærker; startmærker og slutmærker. Sammen med teksten mellem mærkerne udgør et start- og et slutmærke et element. Startmærker skrives med et "<" efterfulgt af elementets navn, eventuelle atributter til elementet og et ">". For eksempel:

```
<p>
```

eller

```
<html lang="da">
```

Slutmærker består blot af "</", elementets navn og ">". For eksempel:

```
</p>
```

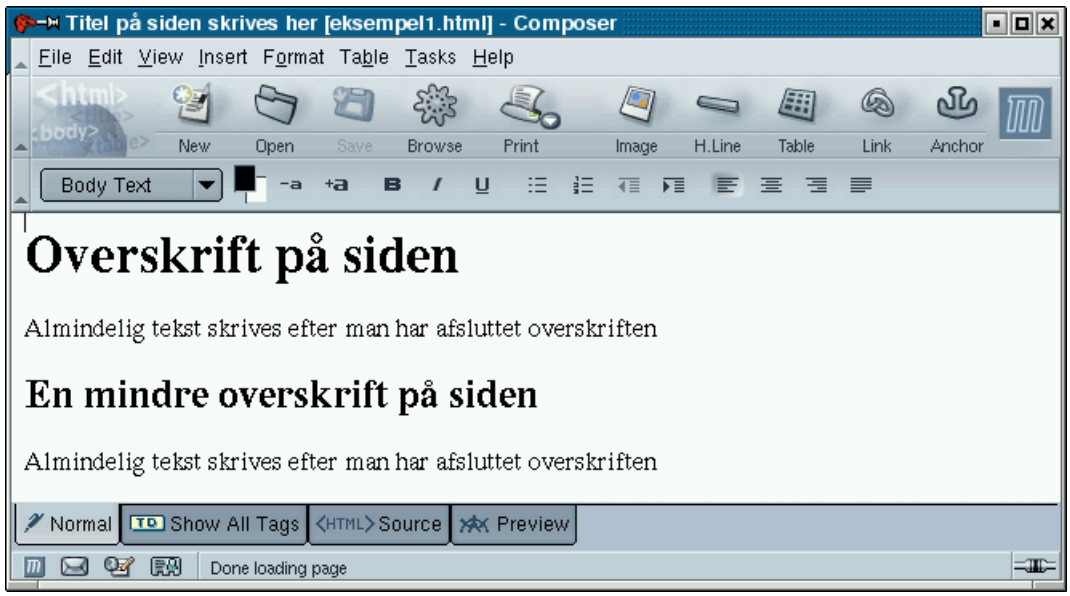
eller

```
</html>
```

Elementernes navne er ikke versalfølsomme, men de bør generelt skrives med småt.

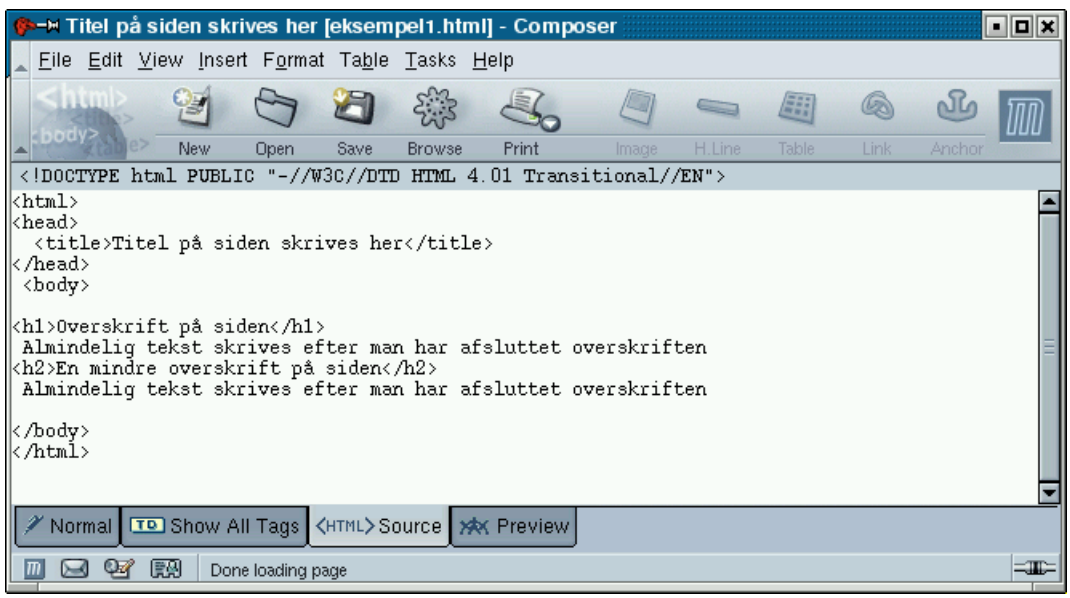
Der er mange muligheder for at lave HTML. Nogle personer skriver HTML i en almindelig tekst-editor og har selv fuld kontrol over alle detaljer. Andre skriver siderne i et grafisk værktøj såsom Netscape eller Mozilla, hvor man sætter mærkerne ind via menuerne.

Figur 1-1. Mozilla kan anvendes til at skrive HTML



Fordelen med at anvende et program som Mozilla er at man med knapperne <HTML>Source og <Normal> nemt kan hoppe frem og tilbage mellem HTML-koden og den grafiske visning af hjemmesiden.

Figur 1-2. Mozilla kan anvendes til at skrive HTML



1.1. Basal HTML

Det første basale HTML-dokument (resultatet kan ses på Figur 1-1) man kan skrive kan se således ud:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
 "http://www.w3.org/TR/REC-html40/loose.dtd">
<html>
<head>
  <title>Titel på siden skrives her</title>
</head>
<body>
<h1>Overskrift på siden</h1>
<p>Almindelig tekst skrives efter man har afsluttet overskriften</p>
<h2>En sekundær overskrift på siden</h2>
<p>Almindelig tekst skrives efter man har afsluttet overskriften</p>
</body>
</html>
```

Det starter med en markering af dokumenttypen:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
 "http://www.w3.org/TR/REC-html40/loose.dtd">
```

Dernæst kommer dokumenthovedet:

```
<head>
...
</head>
```

hvor man kan sætte information om f.eks. titel på dokumentet. Selve indholdet af siden sættes i "body"-elementet mellem

```
<body>
```

og

```
</body>
```

hvor man med

```
<h1>OVERSKRIFT</h1>
```

kan sætte en overskrift ind. Typisk vises denne overskrift med en stor skrifttype. Der findes yderligere fem niveauer overskrifter, "h2", "h3", "h4", "h5" og "h6", men i praksis bør man nok ikke gå længere end til "h3" eller "h4".

Som det vil blive forklaret i Afsnit 1.2 så er det vigtigt at hjemmesiderne skrives uden fejl. Det kræver blandt andet at dokumentet starter med en dokumenttypehenvisning. I dag bruger man typisk overgangsudgaven af HTML 4.0, som man kan henvise til med disse to linjer:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
 "http://www.w3.org/TR/REC-html40/loose.dtd">
```

Den præcise definition af HTML 4.0 kan læses på http://www.w3c.org/TR/html4/.

Mange programmer der skal forestille at gemme HTML-filer har desværre svært ved at sørge for at det er *korrekt* HTML der bliver gemt. Derfor gennemgår vi dele af selv at skrive HTML.

1.1.1. Henvisninger i HTML

Hvis man ønsker at lave en henvisning (eng. "a link") til et dokument, som ligger på en anden server, bruger man "a"-elementet ("a" står for "anchor"). Man kan for eksempel skrive:

```
Se mere på <a href="http://www.sslug.dk/">SSLUG's hjemmeside</a>
```

hvis man vil henvise til http://www.sslug.dk/. Teksten mellem start- og slutmærkerne (i eksemplet mellem `<a href="http://www.sslug.dk/">` og `</a>`) vil være den tekst netlæseren fremhæver som henvisningen. Det kan for eksempel ske ved understregning eller farveskift.

Henvisninger behøver ikke være absolutte. Hvis man har to HTML-filer liggende i samme katalog kan man for eksempel blot bruge filnavnet (nogle gange endda uden "`.html`") som adresse:

```
Jeg har også en <a href="bsd.html">BSD-maskine</a>.
```

(her gik vi ud fra at den anden fil hed "`bsd.html`")

Hvis dokumentet ligger i et andet bibliotek på serveren skal man huske at gøre opmærksom på det.

Hvis det dokument, som man vil henvise til, ligger i et underbibliotek i forhold til ens eget dokument, skriver man:

```
Se mere i <a href="underbibliotek/filnavn.html">dette
dokument</a>.
```

Hvis det dokument, som man vil henvise til, ligger i et overbibliotek i forhold til ens eget dokument, skriver man

```
Se mere i <a href="../filnavn.html">dette dokument</a>.
```

Hvis det dokument, som man vil henvise til, ligger i et andet bibliotek på samme niveau som til ens eget dokument, skriver man

```
Se mere i <a href="../bibliotek/filnavn.html">dette
dokument</a>.
```

Hvis man vil lave en henvisning til et andet sted i ens eget dokument, skal man både lave en henvisning samt navngive det sted, som man ønsker at henvise til. Henvisningen skrives som:

```
Læs mere om <a href="#cirkus">cirkusteltet</a> senere.
```

Og man laver et sted der kan henvises til med:

```
<a name="cirkus">Da vi endelig kom i cirkus var teltet væltet</a>
```

Det er strengt taget ikke nødvendigt at et "a"-element indeholder tekst, hvis det kun bruges til at henvise til, men det er nok mest anvendeligt sammen med en overskrift:

```
<h2><a name="bsd">Min BSD-maskine</a></h2>
```

så folk kommer til starten på et afsnit, når de følger en henvisning.

Man kan kombinere et andet dokuments adresse med et navngivet sted i dette dukument:

```
<a href="udflugt.html#cirkus">Cirkusforestillingen</a>
```

Hvis man henviser til et bestemt sted i en anden forfatters dokument, skal man huske være opmærksom på, at forfatteren kan finde på at ændre sin navngivning, så henvisningen ikke længere virker helt som forventet. Dette er på den anden side ikke anderledes end det problem man generelt vil støde ind i, hvis man henviser til materiale på internettet - det har det med pludselig at forsvinde.

1.1.2. Afsnit og tvungne linjeskift

Afsnit markeres med "p"-elementet (for "paragraph") der forhåbentlig er det mest brugte HTML-element:

```
<p>
 Skåne Sjælland Linux User Group (SSLUG) er en gruppe personer med
 basis i Øresundsregionen og med fælles interesse for styresystemet
```

```
 Linux.
</p>

<p>
 SSLUG er non-profit og så vidt muligt non-økonomi. Vi eksisterer via
 fælles arbejde samt hardware donationer fra tid til anden. Vi har
 ikke noget politisk eller religiøst tilhørsforhold, og skelner ikke
 folk på race eller hudfarve. SSLUG bringer ikke reklamer, spam eller
 lignende. SSLUG vil gerne hjælpe firmaer og private med Linux, men
 har som sagt ingen kommercielle eller økonomiske interesser.
</p>
```

Hvis man af en eller anden grund har behov for at vælge om et afsnit skal venstrestilles, centreres eller højrestilles bruger man atributten "align" med værdien "left", "center" eller "right":

```
<p align="right">
 Teksten her vil blive højrestillet.
</p>

<p>
 Mens denne vil fremstå som læseren foretrækker det (typisk
 venstrestillet).
</p>
```

HTML har et element specifikt til at angive tvungne linjeskift. Navnet er "br" (for "line break") og elementet skrives blot:

```
<br>
```

Da "br" per definition er et tomt element skal der ikke bruges et slutmærke.

Tilsvarende findes det også et tomt element, der indsætter en vandret linje. Navnet er "hr" og det har mulighed for at man bruger attributten "width" til at angive linjens længde. For eksempel vil:

```
<hr width="50%">
```

give en linje svarende til halvdelen af sidens bredde.

1.1.3. Billeder

Det er nemt at indsætte billeder i et HTML-dokument med "img"-elementet. Det er dog vigtigt at være opmærksom på at "img"-elementet er beregnet til såkaldt "in-line"-illustrationer, som for eksempel

specielle symboler der skal stå som en del af teksten:

```
Han skrev en <img src="/logoer/latex.png" alt="LaTeX">-brugsanvisning.
```

Hvis det er en egentlig illustration man har brug for, og den ikke skal stå indlejret i den løbende tekst, bliver man nødt til at snyde lidt. Der er to muligheder. Den ene er at benytte sig af at "img"-elementet tillader at man bruger attributten "align" til at højre- eller venstrestille billedet, og på den måde trække det ud af den løbende tekst:

```
<p>
 <img src="/billeder/tux-solskin.png"
      alt="[billede af Tux med solbriller]"
      align="right">
 Skåne Sjælland Linux User Group (SSLUG) er en gruppe personer med
 basis i Øresundsregionen og med fælles interesse for styresystemet
 Linux.
</p>

<p>
 SSLUG er non-profit og så vidt muligt non-økonomi. Vi eksisterer via
 fælles arbejde samt hardware donationer fra tid til anden. Vi har
 ikke noget politisk eller religiøst tilhørsforhold, og skelner ikke
 folk på race eller hudfarve. SSLUG bringer ikke reklamer, spam eller
 lignende. SSLUG vil gerne hjælpe firmaer og private med Linux, men
 har som sagt ingen kommercielle eller økonomiske interesser.
</p>
```

Den anden er at sætte "img"-elementet alene i et "p"-element:

```
<p align="center">
 <img src="/billeder/tux-solskin.png"
      alt="[billede af Tux med solbriller]">
</p>
```

Attributten "alt" er den tekst der skal vises i stedet for billedet, hvis det af den ene eller den anden grund ikke er indlæst. Af hensyn til blinde og søgemaskiner (og **lynx**-brugere) bør man altså sikre sig at teksten også giver mening, hvis man erstatter billederne med "alt"-teksterne. Hvis billedet ikke er væsentligt (fx fordi det kun anvendes kosmetisk), skal attributten "alt" stadig anvendes. Den skal blot være tom (alt=""), hvilket betyder, at den kan ignoreres.

Hvis man desuden ønsker at bruge billedet som en henvisning til SSLUG's hjemmeside, skal man blot kombinere de forskellige elementer:

```
<p align="center">
 <a href="http://www.sslug.dk/"><img src="/billeder/tux-solskin.png"
  alt="[billede af Tux med solbriller]"></a>
```

```
</p>
```

1.1.4. Lister

Man kan lave unummererede (uordnede) lister som denne

- Element 1
- Element 2

med

```
<ul>
 <li>Element 1</li>
 <li>Element 2</li>
</ul>
```

Man kan lave nummererede (ordnede) lister som denne:

1. Element 1
2. Element 2

med

```
<ol>
 <li>Element 1</li>
 <li>Element 2</li>
</ol>
```

Der er ikke noget i vejen for at lave lister inden i lister:

- Element 1
  - Element 1a
  - Element 1b

- Element 2

med

```
<ul>
 <li>Element 1</li>
  <ul>
   <li>Element 1a</li>
```

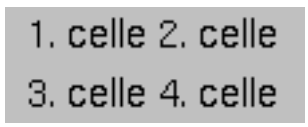
```
   <li>Element 1b</li>
  </ul>
 <li>Element 2</li>
</ul>
```

Man skal være meget påpasselig, når man indlejrer en liste i en anden. Det er nemt at lave fejl, og ens fejl bliver ikke altid afsløret med det samme i netlæseren (en god grund til at køre sine HTML-filer gennem en syntakstjekker). Man skal altid afslutte sine elementer i den modsatte rækkefølge af den, som man startede dem i.

1.1.5. Tabeller i HTML

Det er forholdsvist enkelt at lave enkle skemaer i HTML, men det bliver hurtigt kompliceret, hvis skemaerne bliver store. Her er et 2x2 skema:

Figur 1-3. Tabel lavet i HTML



som du laver med følgende HTML-kode:

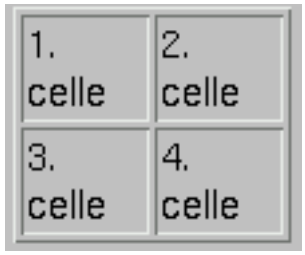
```
<table>
<tr>
 <td>1. celle</td>
 <td>2. celle</td>
</tr>
<tr>
 <td>3. celle</td>
 <td>4. celle</td>
</tr>
</table>
```

<table> og </table> omslutter hele skemaet. <tr> og </tr> omslutter den enkelte række. <td> og </td> omslutter den enkelte celle.

Der kan være et vilkårligt antal rækker og celler i et skema, men det er en god idé at sørge for, at der er lige mange celler i alle rækkerne.

Hvis du vil have en ramme om cellerne i skemaet, skal du bruge attributten border="bredde i pixels" i startmærket, <table>.

Figur 1-4. Tabel lavet i HTML

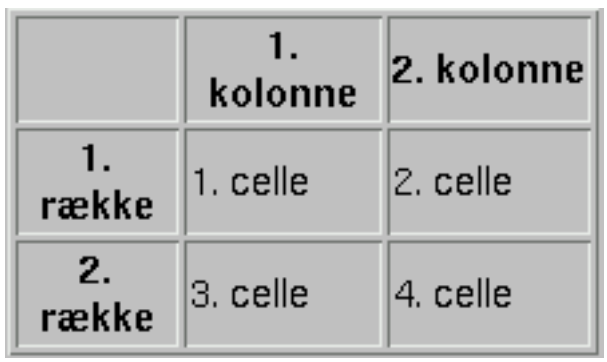


som du laver med følgende HTML-kode:

```
<table border="1">
<tr>
 <td>1. celle</td>
 <td>2. celle</td>
</tr>
<tr>
 <td>3. celle</td>
 <td>4. celle</td>
</tr>
</table>
```

Hvis du vil have overskrifter på dine rækker og kolonner, kan du bruge <th> og </th> i stedet for <td> og </td>. Du kan dog opnå stort set funktionalitet med <td> og </td>.

Figur 1-5. Tabel lavet i HTML

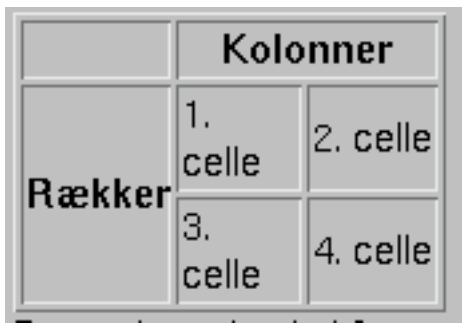


Dette laves med følgende HTML-kode:

```
<table border="1">
<tr>
 <th></th>
 <th>1. kolonne</th>
 <th>2. kolonne</th>
</tr>
<tr>
 <th>1. række</th>
 <td>1. celle</td>
 <td>2. celle</td>
</tr>
<tr>
 <th>2. række</th>
 <td>3. celle</td>
 <td>4. celle</td>
</tr>
</table>
```

Du kan også lave overskrifter, der spænder over flere rækker eller kolonner med attributterne colspan="antal kolonner" og rowspan="antal rækker".

Figur 1-6. Tabel lavet i HTML



Dette kan laves med følgende HTML-kode:

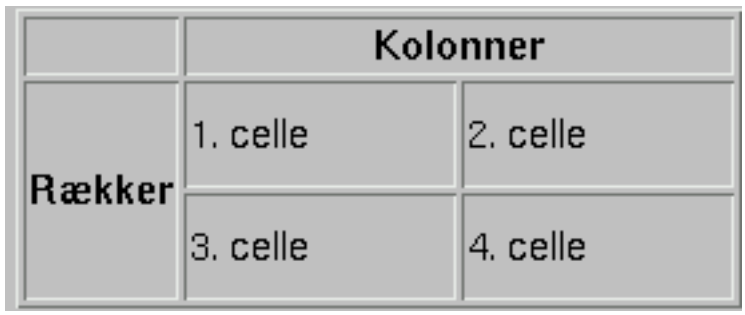
```
<table border="1">
<tr>
 <th></th>
 <th colspan="2">Kolonner</th>
</tr>

<tr>
 <th rowspan="2">Rækker</th>
 <td>1. celle</td>
 <td>2. celle</td>
```

```
</tr>

<tr>
 <td>3. celle</td>
 <td>4. celle</td>
</tr>
</table>
```

Bemærk, at du skal fjerne den efterfølgende kolonne og den efterfølgende række. Det er en god idé at lade rækken stå tom i din HTML-kode, så du nemmere kan overskue koden. Husk at din HTML-kode hurtig bliver meget kompliceret, efterhånden som skemaet bliver større og større. Du kan godt lade dine kolonner og rækker spænde over mere end 2 celler.

Hvis du ikke selv sørger for at bestemme bredden og højden på de enkelte celler, vil browseren selv sørge for, at skemaet bliver stort nok. Du kan selv bestemme størrelsen med attributterne width="bredde i pixel" og height="højde i pixel".

Figur 1-7. Tabel lavet i HTML



som du laver med følgende HTML-kode:

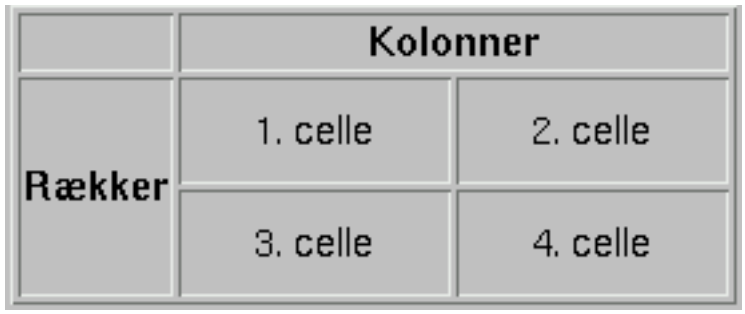
```
<table border="1">
<tr>
 <th></th>
 <th colspan="2">Kolonner</th>
</tr>

<tr>
 <th rowspan="2">Rækker</th>
 <td width="100" height="40">1. celle</td>
 <td width="100" height="40">2. celle</td>
</tr>

<tr>
 <td width="100" height="40">3. celle</td>
 <td width="100" height="40">4. celle</td>
```

```
</tr>
</table>
```

Attributterne width og height er godt nok på vej ud, men du kan ind til videre bruge dem uden problemer.

Du kan også bestemme, om teksten skal være centreret, højre- venstrestillet med attributten align="center, left eller right".

Figur 1-8. Tabel lavet i HTML



som du laver med følgende HTML-kode:

```
<table border="1">
<tr>
 <th></th>
 <th colspan="2">Kolonner</th>
</tr>

<tr>
 <th rowspan="2">Rækker</th>
 <td width="100" height="40" align="center">1. celle</td>
 <td width="100" height="40" align="center">2. celle</td>
</tr>

<tr>
 <td width="100" height="40" align="center">3. celle</td>
 <td width="100" height="40" align="center">4. celle</td>
</tr>
</table>
```

1.2. Validering af hjemmesider

Det er altid irriterende hvis en hjemmeside vises dårligt eller slet ikke kan vises. Et godt trick er at bruge http://validator.w3.org/ til at tjekke syntaksen af sine HTML-dokumenter.

For dem der vil til at lære at lave hjemmesider, er en syntaksvalidator et fortrinligt værktøj, især fordi den udpeger enhver fejl og henviser til dokumentation hvor man kan læse mere om hvordan fejlen kan undgåes. Denne dokumentation er selve standarden i dens nyeste version. Der findes næppe bedre muligheder for selvstudium end at bruge netop denne validator og ovenikøbet får man hjemmesider af højest mulig kvalitet. På denne måde kan man lave hjemmesider der følger de samme standarder som browserne laves efter.

Et smart trick er at have en henvisning på hjemmesiden der peger direkte på validator.w3.org (http://validator.w3.org/) syntakstjek af siden:

```
<p>
 <a href="http://validator.w3.org/check/referer"><img
    src="http://www.w3.org/Icons/valid-html401"
    alt="Valid HTML 4.01!" height="31" width="88"></a>
</p>
```

Fidusen med ovenstående stump HTML er at validator.w3.org (http://validator.w3.org/) selv finder ud af hvilken side du kom fra. Der er altså ikke brug for at rette det til for hver enkel webside.

Når hjemmesiden er blevet fejlfri, får man et tilbud om en skræddersyet henvisning til at lægge ind på hjemmesiden. Det tilbud bør man tage imod.

Alternativt kan man installere WDG HTML validator fra http://www.htmlhelp.com/tools/validator/packages/ - dette kræver dog typisk opdatering af flere Perl-pakker.

1.3. HTML-reference

I dette afsnit gives en HTML-reference.

1.3.1. Strukturelle elementer

- Kommentar:

  ```
  <!--...-->
  ```

  Eksempel:

  ```
  <!-- Dette er en kommentar -->
  ```

- "HTML"-elementet:

  `<html>...</html>`
  omkredser hele HTML-dokumentet (bortset fra dokumenttypehenvisningen).
- "Head"-elementet:

  `<head>...</head>`
  omkredser dokumenthovedet, hvor dokumentets titel og andre oplysninger *om* dokumentet findes.
- "Title"-elementet:

  `<title>...</title>`
  indeholder dokumentets titel.
- "Meta"-elementer er elementer med blandede oplysninger *om* dokumentet. Det kan for eksempel være stikord fra dokumentet:

  ```
  <meta name="keywords" content="SSLUG, Skåne, Sjælland, Linux, brugergruppe">
  ```
- "Body"-elementet indeholder selve dokumentet.

Et samlet eksempel:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
 "http://www.w3.org/TR/REC-html40/loose.dtd">

<html lang="da">
 <head>
  <title>Dokumentets titel</title>
  <meta name="keywords" content="Stikord fra dokumentet">
  <meta name="description" content="Beskrivelse af dokumentet">
 </head>
 <body>
  Her indsættes den egentlige HTML-kode til websiden.
 </body>
</html>
```

1.3.2. Overskrifter

```
<h1>...</h1>    Overskift (hovedniveauet)
<h2>...</h2>    Overskift (niveau 2)
<h3>...</h3>    Overskift (niveau 3)
<h4>...</h4>    Overskift (niveau 4)
```

For eksempel:

```
<h2>Dette er en overskrift i niveau 2</h2>
```

1.3.3. Afsnit

```
<p>... Tekstafsnit ...</p>
```

For eksempel:

```
<p>
 Dette er et afsnit.
</p>
<p>
 Her kommer det næste.
</p>
```

1.3.4. Henvisninger

```
<a>...</a>    Henvisning
```

Eks.:

```
<a href="andet dokument" target="navn på vindue">Titel på andet dokument</a>
<a name="ankernavn"></a>
```

1.3.5. Lister

```
<ol>...</ol>   Ordnet liste
<ul>...</ul>   Uordnet liste
<li>                Punkt på liste
```

Eksempel:

```
<ol>
 <li>Punkt 1
 <li>Punkt 2
</ol>
```

1.3.6. Formattering

```
<em>...</em>          Fremhævet (sædvanligvis i kursiv)
<strong>...</strong>  Yderligere fremhævet (sædvanligvis i fed)
<code>...</code>      Kodeeksempel (sædvanligvis courier)
<kbd>...</kbd>        Maskinskrevet (sædvanligvis courier)
<cite>...</cite>      Citat
```

1.3.7. Andre HTML-elementer

```
<hr>                                Vandret linje
<br>                                Ombryd linje
<blockqoute>...</blockqoute>  Længere citat
```

1.3.8. Billeder i HTML

```
<img>      Indsætter et billede i den løbende tekst
```

Eks.:

```
<img src="billedets adresse" alt="tekstalternativ til billedet" align="left" height="100" w
```

1.3.9. Tabeller i HTML

```
<table></table>    Tabel
<tr>                     Række i tabellen
<td>                     Felt i celle
```

Eks.:

```
<table>
<tr>
    <td>1. celle</td>
    <td>2. celle</td>
</tr>
```

Eksempel:

```
<td rowspan="2">  Felt, der fylder to rækker
<td colspan="2">  Felt, der fylder to kolonner
<tr>
    <td>3. celle</td>
    <td>4. celle</td>
</tr>
</table>
```

Kapitel 2. DocBook

DocBook er et sæt af stylesheets, som beskriver hvordan man laver en bog, artikel eller lignende ud af tekstfiler. DocBook er ligesom LaTeX således *ikke* et grafisk miljø hvor man skriver og kan se resultatet grafisk som det er tilfældet med Word, WordPerfect eller StarOffice. Den store fordel med DocBook er at der er gode værktøjer til at oversætte DocBook til HTML, PostScript, RTF og PDF og få et pænt og ensartet resultat.

Før vi tager fat på installation og en teknisk gennemgang af det "sprog" man skal skrive i - med en vilkårlig teksteditor - så er der i Eksempel 2-1 et eksempel på lidt tekst skrevet i DocBook. Som det kan ses så er det kildeteksten til teksten ovenfor som er vist. DocBook minder meget om at skrive HTML-kode - dvs. koderne afgrænses som <KODEN>.

Grunden til at denne bogserie er skrevet i DocBook er at det var sjovt at prøve. Det har vist sig at det fungerer godt med automatiske krydshenvisninger; det er nemt at håndtere store bogværker på denne måde. Da alle filerne der skrives i "ren ASCII" (tekst-filer), kan flere personer nemmere redigere i de samme filer samtidigt, og disse ændringer kan så blandes sammen med **patch** kommandoen.

Eksempel 2-1. Eksempel på DocBook kode

```
<chapter id="docbook">
<title>DocBook</title>

<para>
DocBook er et sæt af stylesheets, som beskriver hvordan man laver en
bog, artikel eller lignende ud af tekstfiler. DocBook er ligesom
LaTeX således <emphasis>ikke</emphasis> et grafisk miljø hvor man
skriver og kan se resultatet grafisk som det er tilfældet med Word,
WordPerfect eller StarOffice. Den store fordel med DocBook er at der
er gode værktøjer til at oversætte DocBook til HTML, PostScript, RTF
og PDF og få et pænt resultat.
</para>
```

2.1. Installation af Docbook værktøjer

En engelsk installationsvejledning kan findes på følgende URL;

http://www.tldp.org/HOWTO/mini/DocBook-Install/index.html. Nedenfor er en kort vejledning på dansk.

Man kan som regel installere docbook systemet ved under system-installation (eller system-opgradering) at medtage "document processing".

På Mandrake 9.1 er det ret enkelt at få DocBook værktøjerne til at virke. Installer følgende pakker med **urpmi** (efterfulgt af følgende navne). Dette gøres som root.

```
docbook-dtd412-xml
docbook-style-dsssl
docbook-dtd41-sgml
docbook-style-xsl
docbook-dtd30-sgml
docbook-dtd42-xml
docbook-utils-pdf
docbook-dtd31-sgml
docbook-utils
openjade
jadetex
libopenjade0
```

Hvis du bruger en af de distributioner, som har det hele med, kan du afprøve installationen ved for eksempel at bruge docbook-eksemplet, der kan hentes fra bogens hjemmeside under eksempler, eller med Eksempel 2-2 hallo-bog eksemplet her.

Eksempel 2-2. Hallo-hallo-bog

```
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook V4.1//EN">

<book id="index" lang="da">

<chapter id="hellobog-minikapitel"><title>Sgml-afprøvning</title>
<sect1 id="hellobog-minisect">
<title>Hallo, hallo!</title>

<para>
Hallo, Verden! Hallo, hallo! Her kommer indholdet i den nye bog.
Disse linjer ender med at blive en paragraf i den færdige bog.
For at kunne se, hvordan linjeombrydning fungerer, er denne
paragraf lidt længere end de berømte ord, Hello, World!
</para>

<para>
Hvis du vil se et lidt længere eksempel med flere elementer og et
billede, kan du hente og udpakke docbook-eksemplet fra bogens
hjemmeside.
</para>

</sect1>
</chapter>
</book>
```

Du kan selvfølgelig også afprøve systemet ved at bruge sgml-filerne til andre bøger i denne serie. Så skal du huske at flytte billederne ned fra *_images/ directory'et til directory'et med .sgml filerne, husk at rette magic.sgml, så der står *png* i stedet for eps.

Når man står i directory'et med .sgml filerne og skriver følgende kommando, skal systemet generere en .pdf fil.

```
[root@MinMaskine RPMS]# db2pdf bog.sgml
```

Og hvis det fungerer er alt jo godt [1]

Hvorfor ikke installere det hele selv fra source? Det kan man selvfølgelig godt, men der er så mange pakker, der skal spille sammen, så det er faktisk ret svært og måske grænser det til spild af tid, når nu andre har gjort det for en. Men - hvis du prøver, vil vi gerne høre resultatet.

Hvis du kører Red Hat-8.0, Mandrake-9.0, eller SuSE kan du regne med, at tingene fungerer med det samme, du har installeret.

Man kan med for eksempel Red Hat bruge en fil-manager til at kigge på CD-Rom RPM-filerne, og så installere med point-and-click.

Du kan også hente docbook pakker ved at følge links fra linuxdoc (se ovenfor) eller hente RPM-pakker fra Red Hat. Hvis du gør det, så skal du ikke bruge installations-CD'erne og får ofte en nyere version ud af det. Med undtagelse af openjade*rpm skulle der ikke være noget i vejen for at bruge nyere pakker på en ældre installation. Se i øvrigt afsnittet i Sysadm-bogen om pakke-administration. www.linuxbog.dk/admin/bog/rpm.html (http://www.linuxbog.dk/admin/bog/rpm.html)

Med installations - CD'er er fremgangsmåden som følger:

Man "mounter" den første CD. Hvis ikke desktoppen selv gør det (hvis man ikke har slået det fra) så skal man på en kommandolinje skrive:

```
[root@MinMaskine RPMS]# mount /dev/cdrom
[root@MinMaskine RPMS]# cd /mnt/cdrom/Red Hat/RPMS
```

Kommandoen "cd /mnt/cdrom/Red Hat/RPMS" gør, at man "står" i det katalog (directory) hvor pakkerne ligger. Nu kan vi installere, kør:

- Red Hat 7.3 - cd-rom 1:

  ```
  [root@MinMaskine RPMS]# rpm -Uvh --nodeps docbook*.rpm
  [root@MinMaskine RPMS]# rpm -Uvh --nodeps sgml-common-*.rpm
  ```

  Kommandolinjetilvalget "`--nodeps`" er brugt her for at undgå et stop, hvis du kommer til at installere i forkert rækkefølge. Det skyldes RPM-kontrol af, om forudsætninger for anvendelse af et stykke software er til stede. Det kan du læse mere om i administrations-bogen.

  Skift til CD nummer to:

```
[root@MinMaskine RPMS]# cd /
[root@MinMaskine RPMS]# umount /dev/cdrom
[root@MinMaskine RPMS]# mount /dev/cdrom
[root@MinMaskine RPMS]# cd -
```

- Red Hat 7.3 - cd-rom 2:

```
[root@MinMaskine RPMS]# rpm -Uvh --nodeps openjade*.rpm
[root@MinMaskine RPMS]# rpm -Uvh --nodeps jadetex*.rpm
[root@MinMaskine RPMS]# rpm -Uvh --nodeps docbook*.rpm
[root@MinMaskine RPMS]# rpm -Uvh --nodeps tetex*.rpm
```

- Red Hat 7.3 - cd-rom 3, hvis du vil have tetex dokumentationen med:

```
[root@MinMaskine RPMS]# rpm -Uvh tetex*.rpm
```

Princippet er det samme for andre RPM-baserede systemer (fx. Mandrake). SuSE Yast2 og DEBIAN apt-get følger samme principper som RPM, men har helt andre kommandolinjetilvalg og kræver andre opsætningsfiler. Til gengæld kan man med en hurtig internetforbindelse her gratis drage fordel af, at apt-get kan have en liste af servere, hvorfra friske pakker kan hentes.

Se afsnit om administration af softwarepakker i sysadm-bogen eller på http://www.rpm.org.

Dette burde være nok til at installere pakkerne. Hvis du har problemer efter installationen, kan du følge vejledningen fra Norman Walsh, som du kan finde her. http://docbook.sourceforge.net/projects/dsssl/ (http://docbook.sourceforge.net/projects/dsssl/)

Norman Walsh har skrevet de "modulære docbook-style sheets". Hans anvisninger er jo de korrekte for fejlfinding af en installation, men på den anden side, hvis der er problemer, og man har mulighed for at installere pakkerne fra en nyere version af sit system, så er det som regel en hurtigere løsning.

Grunden til, at vi skriver så meget om, hvad man skal gøre i tilfælde af fejl, er selvfølgelig, at vi er nogle stykker, som har kæmpet mod underlige fejl og har haft svært ved at finde en udvej; vi vil gerne hjælpe, hvis du sender en mail.

Se evt. også http://hvdkooij.xs4all.nl/docs-docbook.cms.

Der er en enkelt pakke man nok skal opdatere - læs mere på http://www.redhat.com/support/errata/RHEA-2001-167.html.

2.2. Igang med Docbook

Når man skal igang med docBook er det nemmest at have en model, en tom ramme, som man så kan starte med at skrive i med det samme. Du kan for eksempel fjerne alle paragraffer i eksemplet, som følger med denne bog, og skrive i de rensede filer.

DocBook-dokumentationen ligger på http://www.oasis-open.org/docbook/. Her kan du finde information om alle de elementer, som kan anvendes. Desværre må man sige, at det er en ringe hjælp for dem som skal igang med at skrive for første gang. Det skal denne bog råde bod på. Først ser vi på et par af de vigtigste SGML-elementer og senere hvordan man bruger dem til at lave et helt dokument.

Det første vi skal vide er at alle elementer åbnes og lukkes med et mærke. Åbningsmærket starter med **<** fulgt af elementets navn, eventuelle atributter og til sidst **>**. Afslutningsmærket er blot **</** fulgt af elementets navn og **>**. Et eksempel er `<command>`ls -l`</command>` som markerer at strengen "`ls -l`" er en kommando.

Ud over at **<** har speciel betydning, så har **&** også speciel betydning. Det betyder, at du ikke kan skrive `<` og `&` *direkte* i din tekst, men i stedet må man skrive `&lt;` og `&amp;` (Du kan så vælge også at skrive **>** som > - men det er ikke tvungent). Tegn, som ikke opfører sig almindeligt, kaldes *metategn*.

2.2.1. Tekstblokke - para

Alle tekstblokke skal indkapsles med **<para>** og **</para>**. Linjedeling overlades til docbook-formateringsprogrammerne. Hvis man skriver et linjeskift midt i en <para> blok, ignorerer docbook-programmerne det.

Hvis man vil have linjeskift, må man slutte paragraffen og enten starte en ny paragraf, en liste, en SCREEN, eller, hvis man foretrækker det, skrive al sin tekst som programlisting, så får man docbook til at acceptere ren asciitekst, "typewriter-style".

Der findes ikke en
-konstruktion som man kender det fra HTML.

Eksempel 2-3. para

```
<para>
Dette er en tekstblok.
</para>

<para>
Dette er en anden tekstblok, som starter på ny linje.
</para>
```

Dette kommer til at se ud som:

Dette er en tekstblok.

Dette er en anden tekstblok, som starter på ny linje.

2.2.1.1. Kommandoer - command

I en **<para>** block skal man måske indikere at nu kommer der en kommando, som kan køres.

Eksempel 2-4. command

```
I Linux kan man skrive <command>find / -name
"*.htm?"</command> for at finde HTML-filer.
```

Dette kommer til at se ud som:

I Linux kan man skrive **find / -name "*.htm?"** for at finde HTML-filer.

2.2.1.2. Filer - filename

For at indikere et filnavn i en **<para>** kan man bruge **<filename>**

Eksempel 2-5. filename

```
Linux har altid filen <filename>/etc/password</filename>.
```

Dette kommer til at se ud som:

Linux har altid filen `/etc/password`.

2.2.1.3. Skråskrift - emphasis

For at markere en del af sætningen med kursiv skrift i en **<para>** kan man bruge **<emphasis>**

Eksempel 2-6. emphasis

```
Linux giver <emphasis>mange</emphasis> muligheder.
```

Dette kommer til at se ud som:

Linux giver *mange* muligheder.

2.2.1.4. URL - ulink

For at markere en URL i en **<para>** kan man bruge **<ulink>**

Eksempel 2-7. ulink

```
Læs om SSLUG på
<ulink url="http://www.sslug.dk">SSLUG's hjemmeside http://www.sslug.dk</ulink>. Der er
mange aktiviteter.
```

Dette kommer til at se ud som:

Læs om SSLUG på SSLUG's hjemmeside http://www.sslug.dk (http://www.sslug.dk). Der er mange aktiviteter.

2.2.2. Vise kildetekst og skærmbilleder - programlisting og screen

Har man brug for at vise noget kildetekst bør man sætte det ind i et "programlisting"-element. Hvis man derimod vil vise noget der står på skærmen bør man sætte det ind i et "screen"-element. Både i "programlisting"- og "screen"-elementer følger linjeskiftene i uddata (selvfølgelig) linjeskiftene i SGML/XML-koden.

Eksempel 2-8. programlisting

Koden:

```
<para>
Nu vil jeg vise lidt Perl-kode
</para>

<programlisting>
#!/usr/bin/perl -i.bak -p
s/Microsoft/Linux/g;
</programlisting>
```

vil resultere i disse uddata:

Nu vil jeg vise lidt Perl-kode

```
#!/usr/bin/perl -i.bak -p
s/Microsoft/Linux/g;
```

2.2.2.1. Elementer i "screen"-elementet

For at kunne skelne mellem hvad et program udskriver og hvad brugeren taster, så anvender jeg et par Docbook elementer inden i "screen"-elementet. "prompt" bruges til spørgsmål til brugeren og "userinput" bruges til det brugeren indtaster.

Eksempel 2-9. "prompt" og "userinput" i et "screen"-element

```
<programlisting>
<prompt>[root@tyge /root]# </prompt> <userinput>ls -al /etc/passwd</userinput>
-rw-r--r-- 1 root root 649 jun 16 21:54 /etc/passwd
</programlisting>
```

Dette kommer til at se ud som:

```
[root@tyge /root]#  ls -al /etc/passwd
-rw-r--r-- 1 root root 649 jun 16 21:54 /etc/passwd
```

2.2.3. Tabeller, figurer, kommentarer og lister

2.2.3.1. Tabeller - tabel

Tabeller laves med **<table>**. Bemærk at denne ikke må være inden i en **<para>**. I eksemplet skal med **cols=** angives antallet af søjler. Indrykningen har ingen betydning for resultatet, og anvendes alene til at man selv har overblik over koderne. Igen bemærkes, at det er ligegyldigt om man anvender store eller små bogstaver.

Eksempel 2-10. table

```
<TABLE id="db-firmaet">
<TITLE>Firma-tabel</TITLE>
<TGROUP cols=3 align="char">
<THEAD>
  <ROW>
   <ENTRY>FirmaNavn</ENTRY>
   <ENTRY>Vej</ENTRY>
   <ENTRY>PostNr</ENTRY>
  </ROW>
</THEAD>
<TBODY>
<ROW>
 <ENTRY>Vagabondos</ENTRY>
 <ENTRY>Tagensvej 100</ENTRY>
 <ENTRY>2200</ENTRY>
</ROW>
<ROW>
```

```
 <ENTRY>DKUUG</ENTRY>
 <ENTRY>Fruebjergvej 3</ENTRY>
 <ENTRY>2100</ENTRY>
</ROW>
<ROW>
 <ENTRY>Niels Bohr Institutet</ENTRY>
 <ENTRY>Blegdamsvej 17-21</ENTRY>
 <ENTRY>2100</ENTRY>
</ROW>
</TBODY>
</TGROUP>
</TABLE>
```

Dette kommer til at se ud som:

Tabel 2-1. Firma-tabel

FirmaNavn	Vej	PostNr
Vagabondos	Tagensvej 100	2200
DKUUG	Fruebjergvej 3	2100
Niels Bohr Institutet	Blegdamsvej 17-21	2100

Bemærk at vi har anført `id="label"` sammen med **<table>**. Det er for at kunne krydsreferere til tabellen fra et sted i teksten.

Det skal bemærkes, at man med **<colspec>** kan lave en del bedre formattering af tabeller, såsom farvelægning af celler og andre fikse ting som bredde af tabeller, kolonner etc.

2.2.3.2. Figurer - figure

Figurer kan inkluderes i Docbook-dokumenter. Selvom det anføres at en lang række grafik formater er understøttet, så gælder det i praksis ikke, hvis man skal kunne oversætte både til HTML, PDF og Postscript. Vi anbefaler PNG-format til billeder, da dette er generelt understøttet.

Eksempel 2-11. figure

```
<FIGURE id="linustorvalds-fig" float="1">
<TITLE>Linus Torvalds</TITLE>
<GRAPHIC fileref="linus.&magic;"  scale="60"></GRAPHIC>
</FIGURE>
```

Dette kommer til at se ud som:

Figur 2-1. Linus Torvalds

Bemærk at vi igen her har sat **id="label"** ind under **<figure id="linustorvalds-fig" float="1">** for at kunne krydsreferere til figuren. Du skal selv skalere billedet (i PDF og Postscript udskrift via **scale**-parameteren.

Det kan være lidt vanskeligt at gennemskue, hvor stort ens billede bliver, når man har taget et screenshot og det skal med i ens bog. Det kan også være lidt svært at regne ud hvor store billeder man skal lave.Her følger en gennemgang af hvordan man kan forudsige lidt om skaleringen, så man kan forstå hvad der sker og hvordan **scale**-paremeteren skal bruges.

Det er formentlig velkendt at alle billeder har en opløsning i pixels, det vi som regel forstår som størrelsen af et billede. Et skærmskud kan f.eks. være 900x600 pixels. Hvis man ser det billede på to forskellige computere, hvor den ene har en 20" skærm med en opløsning på 1024x768 og den anden har en 15" skærm med en opløsning på 1024x768, vil det samme billede have to meget forskellige størrelser, når de ses på skærmen. Det er vi vant til og tager ikke så højtideligt – typisk kan vi forstørre eller formindske billedet for at se det bedre alligevel.

Det er meget værre hvis vi taler om udskrivning. Hvis vi fulgte samme metode som for skærme – hvor hver af billedets pixels optager een pixel på skærmen – ville et billede på 900 pixels bredde være 3 tommer bredt på en 300 dpi printer (en printers dpi bestemmer hvor mange (evt. farvede) prikker den kan lave pr. tomme/cm) og 1.5 tomme bredt på en 600 dpi printer. Det er ikke særligt ønskeligt. For at imødekomme det, taler man også om et billedes opløsning pr. enhed, typisk pixels pr. centimeter, eller pixels pr. tomme (det sidste er det der kaldes dpi).

Afhængigt af hvordan du har lavet dit billede (og hvilket format det er) har det allerede en dpi størrelse. Nogen programmer tillader dig at se og ændre denne størrelse (f.eks. **gimp** med jpg billeder). Men, de fleste billeder har ikke et tal indbygget, og derfor antager Docbook en standard værdi, når den skal lave dit billede om til noget der kan skrives ud på en printer (f.eks. postscript eller pdf). Denne standardværdi er, så vidt jeg ved, 72 dpi.

Lad os tage et eksempel. Antag at du har et 900 pixel bredt billede og du vil skrive det ud på en 300 dpi printer (eller en fil med indbygget antagelse om opløsningen, som f.eks. postscript eller pdf). Docbook antager at billedet er 72 dpi, derfor skaleres billedet først op med printerdpi/billededpi = 300/72 ~= 4.2, dvs. at billedet bliver ca. 3750 pixels bredt eller 12.5 tommer, eller ca. 31 cm. Det er jo alt for stort til en A4 side, så her kommer scale parameteren ind. Vi bruger scale parameteren til at sætte det til 40%, så bliver det 1500 pix, 5 tommer eller ca. 13 cm, som passer fint på docbook siden. Det endelige billedes opløsning bliver 1500/5 = 300 dpi – lige som det skal være. Når Docbook laver f.eks. HTML, sker der dog ingen skalering. Her vil billedet stadig være 900 pixels bredt.

Så, hvor stort skal ens billede være? Det vigtigeste er at undgå nedskalering – det er bedre at skulle skalere op på papir, end ned, fordi der uundgåeligt går detaljer tabt ved nedskalering. Så, hvornår bliver der skaleret ned? Hvis man har 15 cm plads, har man typisk plads til mindst 15/2.54*300 pixels = 1771 pixels. Billeder mindre end det, bliver skaleret op, mens billeder større bliver skaleret ned. Hvis man udskriver på 600 dpi, har man selvklart plads til det dobbelte. Men, det er nok ikke smart at lave billeder meget større end 1000 pixels, da de kan blive meget store på skærme med en lav opløsning.

Hvis man vil lave små billeder, så lav dem små til at starte med, lad være med at brug gimp eller lignende til at resample dem med: de bliver alligevel resamplet når de skal skrives ud/vises og på webben betyder det ingenting.

Hvis du gerne vil regne ud hvad din **scale**-parameter skal være, for at optage en bestemt bredde, kan du bruge den her formel: "scale = ønsket bredde i tommer * billeddpi / billedbredde i pixels". Her kan du antage at billeddpi er 72 hvis du ikke ved andet. Det er måske lidt nemmere i centimeter for os danskere, og så bliver det 72 dpi (ca) = 28.3 punkter pr. centimeter. Så bliver formlen: "scale = ønsket bredde i cm * 28.3 / billedebredde i pixels". Et eksempel; vi ønsker at et 900 pixels billede skal være 15 cm bredt ved udskrift, så vi får "scale = 15 * 28.3 / 900 = 0.47". **scale**-parameteren skal altså sættes til 47%. På webben bliver scale parameteren ignoreret, og brugeren får et 900 pix bredt billede. Det kan han så se det hele af, og skalere ned eller bruge scrollbars.

Hvad nu med &magic;? Der er et problem med grafik - hvis man alene laver PDF og HTML-filer, så kan man anføre at det er en `.png` fil eller lignende. Skal man også lave PostScript, så skal man anvende `.eps`-filer. Der skal en konvertering af billederne til. En nem måde at styre magic-variablen er at tilføje denne i den hoved-sgml-fil man laver

```
<!ENTITY % magic-entities SYSTEM "magic.sgml">
  %magic-entities;  <!-- This is where we include it -->
```

2.2.3.3. Fodnoter

Skal du lige lave en fodnote i din SGML-fil skal du prøve dette

```
<para>
Denne tekst er til almindelig brug
<footnote id="fodid">
<para>
```

```
Denne tekst ender som fodnote.
</para>
</footnote>
</para>
```

Som eksempler bliver dette til

Denne tekst er til almindelig brug [2]

2.2.3.4. Info-boks

Skal du lave en info-boks findes tagget "sidebar".

```
<sidebar>
<para>
Det er da ikke særlig svært at se hvordan man laver en
"sidebar".
</para>
</sidebar>
```

Det er da ikke særlig svært at se hvordan man laver en "sidebar".

2.2.3.5. Kommentarer

Skal du lige lave en kommentar i din SGML-fil som ikke kommer med ud i HTML, PDF og Postscript-filerne, så kan du anvende.

```
<!-- Dette er en kommentar -->
```

Bemærk at der er to minus-tegn efter hinanden. I udprintning vil de se ud som ét langt minustegn.

2.2.3.6. Referencer - id og xref

Vi har allerede set at vi kan give en tabel og en figur en label. Dette gjorde vi med **id="label"**. Tilsvarende kan vi referere til den label i en tekstblok et andet sted ved at bruge **<xref>**. Bemærk at denne kommando *ikke* har en tilsvarende **</xref>**. Lad i øvrigt være med at bruge danske bogstaver, mellemrum og understregning (eng. underscore). Grunden er at værdien af "ID" bruges til filnavne - og her kan man ikke tillade de nævnte tegn.

Eksempel 2-12. id og xref

```
Vi kan referere til den forrige figur, altså
<xref linkend="linustorvalds-fig"/> eller tabellen
<xref linkend="db-firmaet"/> uden at tænke på hvilket
nummer de har. Vi anvender blot deres label.
```

Dette kommer til at se ud som:

Vi kan referere til den forrige figur, altså Figur 2-1 eller tabellen Tabel 2-1 uden at tænke på hvilket nummer de har. Vi anvender blot deres label.

2.2.3.7. Lister - itemizedlist og orderedlist

Ofte har man brug for at lave en liste. I Docbook er der flere muligheder, jeg bruger dog kun en type med enkle punkter

Eksempel 2-13. itemizedlist

```
<itemizedlist mark="bullet">
<listitem>
 <para>
  Første punkt i listen skriver jeg her.
 </para>
</listitem>
<listitem>
 <para>
  Andet punkt i listen skriver jeg derefter her.
 </para>
</listitem>
</itemizedlist>
```

Dette kommer til at se ud som:

- Første punkt i listen skriver jeg her.
- Andet punkt i listen skriver jeg derefter her.

Tilsvarende til **<itemizedlist>** kan man lave lister, som er nummererede.

Eksempel 2-14. orderedlist

```
<orderedlist>
<listitem>
 <para>
  Første punkt i listen skriver jeg her.
 </para>
</listitem>
<listitem>
 <para>
  Andet punkt i listen skriver jeg derefter her.
 </para>
</listitem>
</orderedlist>
```

Dette kommer til at se således ud

1. Første punkt i listen skriver jeg her.
2. Andet punkt i listen skriver jeg derefter her.

2.2.4. Opdeling af dokument

En bog inddeles i kapitelniveauer med **<chapter>**, og tilsvarende lavere niveauer **<sect1>**, **<sect2>**, og **<sect3>**. Opdelingen afspejler ikke nødvendigvis de filer man skriver. For hver ny inddeling *skal* man have et **<title></title>** til at anføre titel.

En artikel har samme inddelinger, pånær **<chapter>**, og første inddelingsniveau er **<sect1>**.

Hver af de elementer kan tage en **id="label"**, som man kan bruge til at referere til.

Eksempel 2-15. Opdeling

```
<sect2 id="introduktion">
<title>Dette er en introduktion</title>
 <para>
  Tekst kommer her i en para-blok og bemærkede du at afsnittet har
  label "introduktion"?
 </para>
 <para>
  Havde det været en bog, skulle vi starte med chapter og sect1, men
  men det er jo et eksempel :-)
 </para>
 <para>
  Læs mit næste afsnit i <xref linkend="naeste-inddeling"/>.
 </para>
```

```
 <sect3 id="naeste-inddeling">
 <title>Dette er næste inddeling</title>
   <para>
    Her er et underafsnit. Du skal bemærke, at jeg ikke bruger æ, ø og
    å i labels. Det giver ellers problemer med HTML-filnavne.
   </para>
   <para>
   Bemærk at <command>&lt;!-- Dette er en kommentar --&gt;</command> kan
   bruges til at angive en kommentar som ikke vises
   </para>
 </sect3>
</sect2> <!-- Denne afslutning skal man huske -->
```

Dette kommer til at se ud som:

2.2.4.1. Dette er en introduktion

Tekst kommer her i en para-blok og bemærkede du at afsnittet har label "introduktion"?

Havde det været en bog, skulle vi starte med chapter og sect1, men det er jo et eksempel :-)

Læs mit næste afsnit i Afsnit 2.2.4.1.1.

2.2.4.1.1. Dette er næste inddeling

Her er et underafsnit. Du skal bemærke, at jeg ikke bruger æ, ø og å i labels. Det giver ellers problemer med HTML-filnavne.

Bemærk at **<!-- Dette er en kommentar -->** kan bruges til at angive en kommentar som ikke vises

2.2.4.2. Start filen

Nu har vi set på inddelinger af dokumentet og hvordan man laver tabeller, figurer, viser kildetekst og URL'er.

Lad os antage at man har lavet tre kapitler med hver deres **<chapter id="label-for-kapitel">**, som er gemt i tre filer `introduktion.sgml`, `linux.sgml` og `konklusion.sgml`. Her er så en opskrift på at lave hoved-filen som binder det hele sammen.

Forfattere angives i **<AUTHORGROUP>** og titler som tidligere vist i **<title>**.

```
<!doctype book PUBLIC "-//OASIS//DTD DocBook V3.1//EN" [
 <!entity intro SYSTEM "introduktion.sgml"> <!-- Filnavn ej lig id -->
 <!entity Linux SYSTEM "linux.sgml"> <!-- Filnavn ej lig id -->
 <!entity konklusion SYSTEM "konklusion.sgml"> <!-- Filnavn her lig id -->
]>

<book id="index" lang="da"> <!-- med lang="da" vælges
den danske docbook-localization fil, altså man får danske ord
for chapter, section, content, appendix, index m.v. -->
<bookinfo>
 <title>Bog titel kommer her</title>
 <subtitle>Her er en undertitel</subtitle>

 <AUTHORGROUP>

 <AUTHOR>
   <FIRSTNAME>Peter</FIRSTNAME>
   <SURNAME>Toft</SURNAME>
 </AUTHOR>

 <AUTHOR>
   <FIRSTNAME>Hans</FIRSTNAME>
   <SURNAME>Schou</SURNAME>
 </AUTHOR>

 </AUTHORGROUP>
 <COPYRIGHT>
   <YEAR>2002</YEAR>
   <HOLDER>Peter Toft og mange andre under
   "ÅDL" licensen</HOLDER>
 </COPYRIGHT>

 <ABSTRACT>
  <PARA>
   Her skriver vi et kort abstract.
  </PARA>
 </ABSTRACT>
</bookinfo>

<!-- Følgende linje skaber automatisk en indholdsfortegnelse -->
<toc id="toc"></toc>
&intro;
&Linux;
&konklusion;
</book>
```

I hovedet af filen erklæres hvilke eksterne filer, vil skal læse ind. Hver af disse får en label (intro, Linux og konklusion) som vi anvender til sidst i filen med et & foran. Der er her til sidst vi angiver den rækkefølge filerne indsættes i.

Bemærk at kommandoerne **<toc></toc>** bruges til at få genereret en indholdsfortegnelse. Dette kræver intet andet end den linje.

2.2.4.3. Entities

Vi har i Afsnit 2.2.4.2 allerede set ordet **entity** anvendt. Denne bruges til at definere enten en forkortelse eller at inkludere en anden fil. Vi så tidligere at tre filer blev defineret og her er vist samme definitioner samt at en forkortelse "pto" er sat til "Peter Toft".

```
<!entity intro SYSTEM "introduktion.sgml"> <!-- Filnavn ej lig id -->
<!entity Linux SYSTEM "linux.sgml"> <!-- Filnavn ej lig id -->
<!entity konklusion SYSTEM "konklusion.sgml"> <!-- Filnavn her lig id -->
<!entity pto "Peter Toft">
```

Når man ønsker indholdet af `introduktion.sgml`, så skriver man blot **&intro;**, tilsvarende **&Linux;** og **&konklusion;**. Skal man bruge teksten "Peter Toft", så er det nemmere at bruge **&pto;**.

2.2.5. Specialtegn

På dansk skrives tankestreger sådan – altså en mellemlang streg med mellemrum omkring. Dette skal ikke forveksles med minus, som er en kort streg, som her -. Eller den amerikanske udgave, som er en ret lang streg, uden mellemrum—som ikke anvendes på dansk. (NB: Hvis du ser dette i en HTML udgave, er det ikke sikkert at tegnene ser forskellige ud!)

Docbook koderne for disse tegn er "-", "&ndash" og "—".

2.3. Oversætte SGML til andre formater

Du kan oversætte SGML filerne til HTML, PDF, Postscript og RTF. Der er også mulighed for at vælge at oversætte SGML til én HTML-fil og ikke til mange, opdelt for hver **<sect1>**. Skal din HTML-fil hentes lokalt på læserens harddisk, kan dette være en fordel, men skal den sendes via nettet, bliver det for langsommeligt.

Du oversætter din hoved SGML fil (f.eks. `bog.sgml`) til HTML ved at skrive **docbook2html bog.sgml**. Dine HTML filer gemmes i `bog` under dit nuværende katalog, fordi SGML-filen hedder `bog.sgml`.

Du skal selv sørge for at kopiere dine billed-filer (f.eks. PNG) til kataloget `bog`. Typisk kan dette gøres med en `Makefile`.

Tilsvarende kan man oversætte `bog.sgml` til PDF ved at skrive **docbook2pdf bog.sgml**. Igen skal alle billed-filer (typisk PNG) være placeret der hvor SGML-filerne oversættes. Jeg kopierer normalt alle de filer til et midlertidigt katalog og oversætter der.

Skal man oversætte til Postscript hedder kommandoen **docbook2ps** og RTF laves med **docbook2rtf**.

2.3.1. Fejl i SGML-teksten

Har du lavet fejl i din SGML-tekst, så er den oversætter man har i **docbook2html** faktisk til stor hjælp. Vi skal se på et par eksempler.

Eksempel 2-16. Manglende /para

Lad os prøve at oversætte følgende tekst

```
<!doctype book PUBLIC "-//OASIS//DTD DocBook V3.1//EN" [
]>

<book id="index" lang="da">
  <bookinfo>
    <title>Min første Docbook-bog</title>
  </bookinfo>

<chapter id="forord">
<title>Mit første kapitel</title>

<para>
Vil du skrive i Docbook ?
</para>

<para>
Ja - men jeg kan ikke huske at slutte mine para-blokke.

</chapter>
</book>
```

Filen `eks1.sgml` oversættes nu

```
[root@MinMaskine /root]#   docbook2html eks1.sgml
TMPDIR is DBTOHTML_OUTPUT_DIR3088

input file was called eks1.sgml -- output will be in eks1

working on ../eks1.sgml
jade:../eks1.sgml:24:9:E: end tag for "PARA" omitted, but OMITTAG NO was specified
jade:../eks1.sgml:19:0: start tag was here
about to copy cascading stylesheet and admon graphics to temp dir
about to rename temporary directory to eks1
```

For det første laves et temporært katalog `DBTOHTML_OUTPUT_DIR3088`, og resultatet vil ende i `eks1/` idet filen hedder `eks1.sgml`. Man får to ret klare fejlbeskeder. I linje 24 (næstsidste linje) må der ikke komme en **</chapter>**, når der bør komme en slut PARA, dvs. **</para>**. Den følgende linje fortæller dernæst at i linje 19 var den **<para>**, som ikke kunne matches.

Tilføj en **</para>** før næstsidste linje og teksten oversætter fejlfrit.

Eksempel 2-17. To ens ID

Lad os prøve at oversætte følgende tekst

```
<!doctype book PUBLIC "-//OASIS//DTD DocBook V3.1//EN" [
]>

<book id="index" lang="da">
  <bookinfo>
    <title>Eksempel</title>
  </bookinfo>

<toc id="toc"></toc>


<chapter id="id1">
<title>Mit første kapitel</title>

<para>
Vil du skrive i Docbook ?
</para>

<sect1 id="id1">
<title>Mit første underkapitel</title>
<para>
tekst1
</para>

</sect1>
</chapter>
</book>
```

Filen `eks2.sgml` oversættes nu

```
[root@MinMaskine /root]# docbook2html eks2.sgml
TMPDIR is DBTOHTML_OUTPUT_DIR3110

input file was called eks2.sgml -- output will be in eks2

working on ../eks2.sgml
jade:../eks2.sgml:19:11:E: ID "ID1" already defined
jade:../eks2.sgml:12:13: ID "ID1" first defined here
about to copy cascading stylesheet and admon graphics to temp dir
```

```
about to rename temporary directory to eks2
```

Her ses at ID1 er erklæret to gange - og der oplyses hvilke steder det sker. Omdefiner den ene og oversæt igen.

2.3.2. "Alle" de grimme detaljer

2.3.2.1. Stikordsregister

At få lavet et stikordsregister er ikke så integreret i Docbook som man kunne ønske sig, men modsat så kan det nemt indføres, når bare man har en vejledning. Det første man skal gøre er at hente et Perl-program, **collateindex.pl**, som f.eks. kan hentes fra www.linuxbog.dk/misc/collateindex.pl (http://www.linuxbog.dk/misc/collateindex.pl). Filen skal gøres kørbar og f.eks. placeres i `/usr/local/bin`.

2.3.2.1.1. Markere til stikordsregisteret - indexterm

Før man får en stikordsliste skal man igennem hele ens dokument og markere de steder, man ønsker et link til. Det er manuelt arbejde. Hvert sted man ønsker et link til skal man tilføje

```
<indexterm><primary>Søgeord</primary></indexterm>
```

I stikordsregisteret vil man så kunne se at "Søgeord" kan man læse om det pågælende afsnit. Man kan godt lave henvisning til flere forskellige steder med samme søgeord. Indføj blot flere linjer som den ovenstående og man får et tilsvarende antal referencer i stikordsregisteret.

Samme markeringsteknik kan laves med underinddelinger, så man under "Søgeord" kan have f.eks. "At lave søgeord" og et andet sted "At finde søgeord". For at lave det første eksempel skrives følgende, og det kan bemærkes, at linjedelingen og indrykningen nedenfor alene tjener til at gøre SGML-koden nemmere at læse.

```
<indexterm>
 <primary>Søgeord</primary>
 <secondary>At lave søgeord</secondary>
</indexterm>
```

2.3.2.1.2. Generere indholdsfortegnelsen - collateindex.pl

Når man har sat alle sine stikordshenvisninger ind i dokumentet kommer turen til at få genereret dokumentet med **collateindex.pl**. Tricket er at man skal oversætte sit dokument en gang før man kører

docbook2html (eller hvad man nu har som mål-format).

Jeg plejer typisk at have en `Makefile`, hvor jeg oversætter min hoved SGML-fil `bog.sgml` på følgende måde. Jeg har kommenteret Makefilen med kommentarer før kommandoen

```
# Slet mit midlertidige katalog "tempdir" og "stikord.sgml"
    rm -rf tempdir stikord.sgml
# Skab et  midlertidige katalog "tempdir"
    mkdir tempdir
# Initialiser "stikord.sgml" til at være tom
    perl /usr/local/bin/collateindex.pl \
        -s Symboler  -t Stikordsregister -g \
        -i stikord -N -o stikord.sgml
# Kopier alle SGML filer til "tempdir"
    cp *.sgml tempdir
# Gå ned i "tempdir" og oversæt SGML-filerne hvor HTML.index dannes
    (cd tempdir; jade -t sgml -ihtml \
        -d /usr/lib/sgml/stylesheets/cygnus-both.dsl\#html \
        -V html-index bog.sgml)
# Kør nu "collateindex.pl" som oversætter "HTML.index" til indholdet
# af SGML-filen "stikord.sgml". -s er at symboler optræder under
# navnet "symboler". -t angiver overskrift for "stikord.sgml"
    perl /usr/local/bin/collateindex.pl -s Symboler \
        -t Stikordsregister  -g -i stikord \
        -o stikord.sgml tempdir/HTML.index
# Nu er "stikord.sgml" fuld af referencer som kan oversættes
    docbook2html bog.sgml
# Kopier alle PNG-filer fra "images/" til "bog/"
    cp images/*.png bog
```

Skal du tilsvarende oversætte til PDF kan du have en del af en `Makefile` med et lidt andet indhold. Ideen er den samme, men **jade**, som anvendes til at lave stikordsregisteret får "print" overført ikke "html". Dette gør, at man få sidenumre og ikke afsnitsnavne i stikordsregisteret. Bemærk også at **-V nochunks** gør at filen oversættes til en stor fil (svarende til den endelige PDF-fil).

```
# Slet "stikord.sgml" min midlertidige kataloger
#  "tempdir" og "bogpdf"
    rm -rf tempdir stikord.sgml bogpdf
# Skab et  midlertidige katalog "tempdir"
    mkdir tempdir
# Initialiser "stikord.sgml" til at være tom
    perl /usr/local/bin/collateindex.pl \
        -s Symboler  -t Stikordsregister -g \
        -i stikord -N -o stikord.sgml
# Kopier alle SGML filer til "tempdir"
    cp *.sgml tempdir
# Gå ned i "tempdir" og oversæt SGML-filerne hvor HTML.index dannes
    (cd tempdir; jade -t sgml -ihtml \
        -d /usr/lib/sgml/stylesheets/cygnus-both.dsl\#print
        -V html-index -V nochunks bog.sgml > slet_fil.html)
# Kør nu "collateindex.pl" som oversætter "HTML.index" til indholdet
```

```
# af SGML-filen "stikord.sgml". -s er at symboler optræder under
# navnet "symboler". -t angiver overskrift for "stikord.sgml"
    perl /usr/local/bin/collateindex.pl -s Symboler \
        -t Stikordsregister  -g -i stikord \
        -o stikord.sgml tempdir/HTML.index
# Nu er "stikord.sgml" fuld af referencer som kan oversættes
    mkdir bogpdf
# Kopier alle SGML-filer til "bogpdf/"
    cp *.sgml bogpdf
# Kopier alle PNG-filer fra "images/" til "bog/"
    cp images/*.png bogpdf
# Kør "docbook2pdf" i "bogpdf/"
    (cd bogpdf; docbook2pdf bog.sgml)
# Flytter PDF-filen til nuværende katalog
    mv bogpdf/bog.pdf .
```

2.3.2.1.3. Postscript er et problem

Som vist i forrige afsnit kopieres PNG-filer blot til det katalog hvor man oversætter. Skal man oversætte Docbook dokumentet til Postscript opstår dog problemet; Man kan ikke oversætte dokumentet med PNG-billeder, men godt med EPS-billeder. Det problem kan også løses, men det er bøvlet og kræver at alle billeder oversættes til EPS, f.eks. ved at bruge **convert**fra ImageMagick-programmet.

2.3.2.2. Ændring af opsætning

Er du utilfreds med hvordan marginer i PDF-filen er eller hvordan dine HTML style-filer anvendes, så har Docbook et hav af muligheder for at skrue på format.

I `/usr/lib/sgml` ligger alle de opsætningsfiler Docbook bruger. Oftest vil man ændre i ganske få filer, såsom `/usr/lib/sgml/stylesheets/nwalsh-modular/common/dbl1da.ent`. I denne fil står alle de oversættelser man har til dansk for referencer og afsnit. Mener du at en "Section" (på engelsk) hedder "Sektion" og ikke "Afsnit" så kan du ændre linjen

```
<!ENTITY Section        "Afsnit">
```

til

```
<!ENTITY Section        "Sektion">
```

Tilsvarende finder du mange opsætningsparametre fælles for HTML og PDF/Postscript i `/usr/lib/sgml/stylesheets/nwalsh-modular/common/dbcommon.dsl`. Filen er skrevet i LISP (ligesom Emacs opsætningsfiler) og har mange kommentarer. Oftest er det ikke denne fil man vil ændre i.

For HTML-generering vil du nok ændre i `/usr/lib/sgml/stylesheets/nwalsh-modular/html/dbparam.dsl`. Du vil se at filen har mange kommentarer og vil du ændre en parameter fra sand til falsk er det **#t** som ændres til **#f**.

I forhold til standardopsætningen kan det være interessant at kigge på:

```
(define %section-autolabel%
```

Sættes den til **#t** bliver afsnit ("sections") automatisk nummereret. Første afsnit ("sect1") i kapitel 7 vil så få nummer 7.1.

Hvis man vil undgå at få en indholdsfortegnelse over tabeller og figurer nedenunder den almindelige indholdsfortegnelse, skal $generate-book-lot-list$ sættes til

```
(define ($generate-book-lot-list$)
  ;; REFENTRY generate-book-lot-list
  ;; PURP Which Lists of Titles should be produced for Books?
  ;; DESC
  ;; This parameter should be a list (possibly empty) of the elements
  ;; for which Lists of Titles should be produced for each 'Book'.
  ;;
  ;; It is meaningless to put elements that do not have titles in this
  ;; list.  If elements with optional titles are placed in this list, only
  ;; the instances of those elements that do have titles will appear in
  ;; the LOT.
  ;;
  ;; /DESC
  ;; AUTHOR N/A
  ;; /REFENTRY
  (list ))
```

Hvis du ønsker at styre udseendet af bogen (i HTML-udgaven), kan det gøres ved at vælge at bruge et eksternt Cascading Style Sheet. Filnavnet på dette angives i

```
 (define %stylesheet%
  ;; REFENTRY stylesheet
  ;; PURP Name of the stylesheet to use
  ;; DESC
  ;; The name of the stylesheet to place in the HTML LINK TAG, or '#f' to
  ;; suppress the stylesheet LINK.
  ;; /DESC
  ;; AUTHOR N/A
  ;; /REFENTRY
  "../style/style.css")
```

Du skal selvfølgelig sørge for, at der er et stylesheet på den angivne placering...

For PDF og Postscript-generering (kaldes "print" i Docbook) vil du nok ændre i `/usr/share/sgml/docbook/dsssl-stylesheets/print/dbparam.dsl` (Red Hat 6.2 `/usr/lib/sgml/stylesheets/nwalsh-modular/print/dbparam.dsl`). En ting du nok skal se på er "%paper-type%" som sættes til

```
(define %paper-type%
  ;; REFENTRY paper-type
  ;; PURP Name of paper type
  ;; DESC
  ;; The paper type value identifies the sort of paper in use, for example,
  ;; 'A4' or 'USletter'. Setting the paper type is an
  ;; easy shortcut for setting the correct paper height and width.
  ;;
  ;; As distributed, only 'A4' and 'USletter' are supported.  You can add
  ;; additional paper types by updating 'page-width' and 'page-height'.
  ;; If you do, please pass along your updates.
  ;; /DESC
  ;; AUTHOR N/A
  ;; /REFENTRY
  ;; "USletter"
  "A4")
```

Vil du have to kolonner tekst i PS og PDF-udskrift rettes tilsvarende

```
(define %page-n-columns%
  ;; REFENTRY page-n-columns
  ;; PURP Sets the number of columns on each page
  ;; DESC
  ;; Sets the number of columns on each page
  ;; /DESC
  ;; AUTHOR N/A
  ;; /REFENTRY
  1)
```

til

```
(define %page-n-columns%
  ;; REFENTRY page-n-columns
  ;; PURP Sets the number of columns on each page
  ;; DESC
  ;; Sets the number of columns on each page
  ;; /DESC
  ;; AUTHOR N/A
  ;; /REFENTRY
  2)
```

Tilsvarende kan man lede efter margin og finde de marginer, som anvendes i PDF-dokumentet, ligesom "%section-autolabel%" og "$generate-book-lot-list$" kan ændres på samme vis som i `/usr/share/sgml/docbook/dsssl-stylesheets/html/dbparam.dsl`.

2.4. De forskellige dele af docbook-systemet

Her er en kort beskrivelse af de dele, som indgår i docbook systemet.

Et dokument system består af tre dele:

1. En specifikation af formatet, altså en syntax og/eller dokumentation af de elementer, man bruger, og hvad meningen er med dem.
2. Filer med formateringskommandoer, som omformer det abstrakte dokument til et output, som kan printes eller på anden måde publiceres.
3. Et program (eller flere programmer), som læser og udfører formateringskommandoerne.
4. En print-mekanisme, som kan omforme fra et almindeligt printerformat til det, som printeren nu måtte forlange.

Det var jo 4 ting!? Jo, men den sidste del er jo oftest en generel mekanisme, som for eksempel Linux/GNU print systemet, der kan tage en postscript fil og konvertere den til de fleste af de printerformater, som man kan købe i dag.

2.4.1. Hvad docbook består af

DocBook systemet er sammensat af mange elementer. De kommer forskellige steder fra, og programmørerne (eller forfatterne) har ikke ønsket at sætte dem sammen i én pakke.

Her er en liste over alle de ting, som skal fungere sammen for at DocBook kildetekster kan "oversættes" til html, pdf eller postscript filer.

- Openjade, programmet, som foretager konverteringer af sgml dokumenter til andre formater. **jade** læser en document type definition (DTD) og agerer efter denne instruktion på dokument "source".
- DocBook SGML - DTD. Det er de filer, som beskriver reglerne for de elementer, som OpenJade skal acceptere, altså *regler* eller *syntax* for, hvornår man skal bruge for eksempel <para>.
- ISO entity filer. De definerer de tegn, som bruges i output fra openjade. Der er mange typer tegn, græske bogstaver, matematiske tegn, specielle valutasymboler etc. Det er faktisk filer fra International Standards Organization (ISO).
- DocBook DSSSL, Document Style Semantics and Specification Language (DSSSL) filer med endelsen .dsl specificerer hvordan OpenJade skal konvertere til html, rtf, tex med videre. DSSSL er en ISO standard.
- Docbook-utils. Nogle konverteringer foregår i flere trin. Disse tools er blot scripts, som samler flere kommandoer til én kommando.
- collateindex.pl, et perl-script, som kan generere index.sgml til pdfjadetex - systemet.

- TeX, for eksempel tetex implementationen, som følger med Red Hat og de fleste andre Linux/GNU distributioner. Man skal også have hyperref og latex, for at det hele spiller sammen, men dem har distributøren pakket sammen med tetex.
- jadetex og pdfjadetex, formaterings-specifikationer til TeX, som gør det muligt at konvertere det output, som jade laver, til postscript filer og pdf filer.
- HTMLdoc, er en af måderne til generering af bookmarks eller index til PDF filer.
- SGMLSpm and docbook2X, for konvertering til man-page format.

2.4.2. Pakke-forklaringer

Her er korte forklaringer af pakkerne, som vi skal arbejde med:

2.4.2.1. Openjade

Openjade: OpenJade er et program, som behandler "Standard Generalized Markup Language" (SGML) og "Document Style Semantics and Specification Language" (DSSSL). Den transformerer DocBook sgml kildefiler til html, tex, rtf, txt og andet. OpenJade er den essentielle maskine til konvertering af en DocBook fil til andre formater. TeX output formatet bruges mest som et mellemformat for at opnå dvi, pdf og postscript filer via TeX macro'er og dvi konverterings programmer.

2.4.2.2. Dokument-type-definitioner

DocBook SGML DTD, dokument-type-definitionen for den eller de docbook - versioner, som du ønsker at benytte. Der er ikke den store forskel mellem for eksempel version 4.0 og version 4.1. Ved at rette i dokument-type-versions-specifikationen øverst i kildeteksten kan man skifte fra den ene til den anden version, og ofte vil oversættelse, d.v.s. generering af html og/eller pdf-filer, stadig kunne foretages uden at der opstår fejl.

DTD filerne for en DocBook version ligger i et directory, som på Red Hat er `/usr/share/sgml/docbook/<dir-navn>` Disse filer definerer sammen med entity filerne et sæt regler som anvendes af OpenJade, når den læser kildeteksten af en sgml-pakke.

OpenJade skal have tilgang til et katalog for hver type dokument, som man forventer, at den skal kunne oversætte. Strengt taget kan man specificere dcl-filer, d.v.s. docbook-erklæringer, på kommandolinjen, men det er altså ikke praktisk, så derfor har man catalog-mekanismen.

2.4.2.3. ISO-entitets-SGML filer

ISO8879 ENTITY SGML. Entiteter definerer repræsentation af specielle tegn, som enten ikke har nogen keyboard-knap eller som har speciel funktion i SGML-kildeteksten. Eksempel: "&" = '&' og

"≈" = '≈' (approximation, tilnærmelsesvis-lig-med).

2.4.2.4. Style-sheets eller DSSSL-filer

DocBook DSSSL, Document Style Semantics and Specification Language (DSSSL) filerne med .dsl endelse, for hver af de forskellige typer output format, som man ønsker, html, rtf og så videre. DSSSL er en ISO-standard, og man kan finde dokumentationen på OASIS-hjemmesiden, http://www.oasis-open.org/ (http://www.oasis-open.org/docbook/).

2.4.2.5. Docbook-utils

Det er små scripts, som benytter et lidt større script, jw, jade-wrapper, til opsætning, så oversættelsen sker uden for mange overflødige krav til kommandolinjen.

Konvertering af en docbook fil til postscript eller pdf er en to-trins proces. Openjade skriver en tex-fil, som så er input for jadetex, som producerer en dvi, og pdfjadetex, som producerer en pdf-fil. Postscript opnås så ved at sende dvi filen igennem dvips. Docbook-utils sparer os for alle de kommandoer, så vi med en enkelt kommando kan udføre disse operationer.

2.4.2.6. HTMLdoc - et hjælpeprogram

Mini-Howto'en for installering af DocBook pakker nævner dette som et specielt system, men jeg har p.t. ikke nogen idé om, hvilke Red Hat 7.1++ pakker dette svarer til. HTMLdoc er et frit program, som konverterer html filer til pdf eller ps filer.

2.4.2.7. SGMLSPM og docbook2X

SGMLspm er et perl modul, som sammen med andre programmer (docbook2X) kan generere man-pages. SGMLSpm er et perl-library modul, som behandler parsed output fra onsgmls (open new sgml - parser). SGMLSpm inkluderer en applikation, som hedder sgmlspl, for at kunne bruge SGMLSpm modul-library'et. Sgmlspl har brug for spec filer, som kan fås fra forskellige andre sites på internettet, for hver type konvertering, som den skal foretage. DocBook2X er en pakke, som leverer spec-filer, der kan transformere docbook-filer til man-pages.

2.5. Epilog

Har du brug for at se flere SGML eksempler, så se f.eks. www.linuxbog.dk/linuxbog/friheden (http://www.linuxbog.dk/linuxbog/friheden). Tilsvarende er der en del eksempler i

`/usr/lib/sgml/stylesheets/nwalsh-modular/doc`.

Har du brug for hjælp til Docbook, så er der en god diskussionsgruppe at finde på docbook-tools-discuss@sourceware.cygnus.com (mailto:docbook-tools-discuss@sourceware.cygnus.com). Man tilmelder sig via at skrive til docbook-tools-discuss-subscribe@sourceware.cygnus.com (mailto:docbook-tools-discuss-subscribe@sourceware.cygnus.com).

- Mark Galassi "Kom i gang" http://nis-www.lanl.gov/~rosalia/mydocs/docbook-intro.html
- Du kan også have glæde af den fulde Docbook dokumentation, der kan hentes fra http://www.oasis-open.org/docbook/.
- Bogen "DocBook - The Definitive Guide" af Norman Walsh & Leonard Muellner - O'Reilly. Den kan læses på http://docbook.org/tdg/. Den er måske ikke "The Definitive Guide" i praksis...
- A Practical Introduction to DocBook http://www.oswg.org/oswg-nightly/oswg/en_US.ISO_8859-1/articles/DocBook-Intro/docbook-intro/index.html
- DocBook HOWTO http://metalab.unc.edu/godoy/using-docbook/using-docbook.html
- Practical information about the use of SGML/XML and DocBook on Debian http://www.debian.org/doc/manuals/sgml-howto/howto.html
- Praktisk guide i brug af SGML/XML og Docbook på Windows NT http://ourworld.compuserve.com/homepages/hoenicka_markus/ntsgml.html
-

  Skal du konvertere fra HTML til SGML/DocBook kan du have glæde af Carsten Svaneborgs oversætter, der findes på www.linuxbog.dk/misc/html2sgml (http://www.linuxbog.dk/misc/html2sgml)

- Ret komplet guide om O'Reilly's DocBook baserede produktions systemer http://www.oreilly.com/people/staff/crism/dsssl/orastyle/index.html
- FreeBSD tutorial http://www.freebsd.org/tutorials/docproj-primer
- OASIS links http://www.oasis-open.org/docbook/documentation/other.html (http://www.oasis-open.org/docbook/documentation/other.html)
- Integration mellem Emacs og Docbook http://www.lst.de/~eric/crash-course/HTML/emacs-psgml-mode-tips.html
- Helt ny "Crash-Course to DocBook" http://public.lst.de/~eric

2.5.1. Editorer med support for SGML syntax highlighting

2.5.1.1. Emacs med nxml

Der er nogen, som har haft stor gavn af en xml syntax support pakke til GNU emacs. Pakken hedder nxml, er skrevet af James Clark og alene dette navn kan få Linux Groff og Jade brugere til at spidse øren,

idet James Clark har skrevet de to programmer. Pakken kan hentes på James Clark's site (http://www.thaiopensource.com/download/) og er under stadig udvikling.

Det er lettest at gå til nxml hvis man kender Emacs i forvejen - Emacs ligner ikke rigtigt nogen anden editor. Så er nxml til gengæld ret enkelt at bruge da kernefunktionaliteten ligger i blot tre kommandoer som til gengæld gør et stort arbejde for en. On-the-fly validering er bestemt også en rar ting.

Valideringsskemaer ligger i mappen `nxml-mode-yyyymmdd/schema` som RELAX NG skemaer i kompakt notation, .rnc filer. Som default kommer nxml med skemaer for docbook, xhtml, xsl og RELAX NG(xml notation). nxml bestemmer hvilket skema der skal anvendes på baggrund af rodelementet i ens dokument. Hvis rodelementet er 'article' eller 'book' anvendes docbook. Filen schema.xml i schema-mappen bestemmer hvordan skemaer og rodelementer hænger sammen.

Hvis man starter på et nyt docbook-dokument, må man give kommandoen C-c C-a (svarer til Ctrl-C Ctrl-A i normal notation) efter at have skrevet rodelementet - ellers har man blot den generiske xml-funktionalitet (Using vacuous schema).

Hjælpen findes ved at give kommandoen C-h m som giver en oversigt over nxml's kommandoer (hvis man er i nxml-mode) eller C-h i som giver adgang til Emacs' info system. Her finder man også dokumentation på nxml-mode (hvis info-oversigten er lang, brug C-s nxml). For selve installationen som ikke er ret kompliceret, kig i Readme.

2.5.1.2. gvim

Gvim har de samme faciliteter som Emacs, og flere til endda. Der er mange typer completion af XML syntaxen. Syntax for SGML er installeret fra starten (VIM havde SGML syntax for 6 år siden allerede, men det anbefales at bruge gvim 6.1 eller nyere.)

Brug kommandoen **:help completion** for at få mere at vide om faciliteter for hjælp under editering, og brug kommandoen **:help syntax** for at få en forklaring af selve syntax systemet.

Slutbemærkning:

1. Mandrake 9.x og SuSE 8.1 samt Red Hat 8.0 kører med det samme, hvis du har valgt de relevante pakker. Vi kan ikke anbefale Red Hat 7.0, 7.1 og 7.2, da der har været mange problemer med at få deres docbook output til at fungere. Fejlene er rettede i senere versioner, så Red Hat 7.3 og 8.x fungerer uden problemer - og hvis man hellere vil baglæns, så kan man stadig få Red Hat 6.2 - docbook pakker til at fungere på en intel-baseret Linux.
2. Denne tekst ender som fodnote.

Kapitel 3. DocBook i XML : Ett konkret exempel

Autogenerering av databasbeskrivning

Här skall vi prova att sätta DocBook i arbete, och samtidigt prova köra DockBook i XML istället för SGML. Vi tänker oss att vi jobbar med en relativt stor databas, som dessutom genomgår förändringar ibland. Vi vill kunna ta ut, automatiskt, ett dokument som beskriver tabellerna i databasen. Dokumentformaten skall vara pdf, en ensam HTML fil och en HTML struktur, uppdelad per tabell. Dessutom vill vi gärna kunna byta plattform, om det skulle behövas.

Ett externt verktyg kommer att gå igenom databasen, i detta fall en PostgreSQL 7.3.1, och skapa en fil med XML-syntax för att beskriva databasen. Vi använder oss av tcl för att gå igenom databasen, pgtcl som har stöd för PostgreSQl inbyggt. Utdata från pgtcl passar dock inte riktig in i DocBooks struktur, så vi måste stuva om datat. Det gör vi med XSL. Vi beskriver hur transformeringen skall gå till i ett XSL-dokument. Vi har dessutom ett ramdokument för att beskriva metadata om dokumentet, och vi gör 'include' på den XSL-processade databasbeskrivningen. Vi är dessutom lata, och stoppar allt i en Makefile.

Databasen kan givetvis bytas mot någon annan, men då måste XSL-dokumentet anpassas därefter, likaså det externa verktyget, dvs tcl-skriptet.

3.1. Verktyg

För att klara detta kommer vi att använda oss av:

- PostgreSQL med pgtclsh
- Java
- Fop (Formatting Objects Processor) från Apache
- Saxon (XSL motor)
- DocBook Stylesheets
- Make

För att bringa ordning i detta kommer vi att hantera dessa filer:

- `datamodel.xml` - statiskt huvuddokument
- `make_tables.tcl` - skriptet som skaffar rådata
- `tables.xml` - utdata från `make_tables.tcl`
- `to_docbook_table.xsl` - stylesheet för att dockbook-anpassa rådata
- `docbook_tables.xml` - utdata från stylsheetet, i docbook-format
- `Makefile` - hur vi utför kommandona

3.1.1. Installation av verktyg

Här går vi bara genom de docbookrelaterade verktygen, databas och make utgår jag från att de redan finns.

3.1.1.1. Java

Java kan hämtas på Suns javasidor (http://java.sun.com/). Detta exempel har använt Java SDK 1.4.1 rpm-versionen, men de flesta nyare versioner duger. Rpm-versionen installerar det mesta, men jag brukar lägga till följande i `~/.basrc`:

```
export JAVA_HOME=/usr/java/j2sdk1.4.1_01/jre
```

Dessutom brukar jag skapa en symbolisk länk till java

```
ln -s $JAVA_HOME/bin/java /usr/bin/java
```

3.1.1.2. FOP

Fop kan hämtas på Apaches XML sidor (http://xml.apache.org/fop/). Detta exempel har använt Fop 0.20.5. Tara upp Fop, och lägg det t.ex. under `/usr/java/fop`. Det är bra att lägga till följande i `~/.basrc`:

```
export FOP_HOME=/usr/java/fop/fop-0.20.5rc3a
```

För att hantera bilder behövs ett bildbibliotek. Hämta Jimi hos Sun (http://java.sun.com/products/jimi/) och packa upp det med **`tar xvzf jimi1_0.tar.Z`**. Leta upp `JimiProClasses.zip` och kopiera det till `$FOP_HOME/lib/jim-1.0.jar`. Det skall alltså till ett namnbyte på filen.

3.1.1.3. Saxon

Saxon kan hämtas hos SourceForge (https://sourceforge.net/project/showfiles.php?group_id=29872). Detta exempel har använt Saxon 6.5.2, eftersom det är den rekomenderade versionen, även om det finns nyare. Packa upp Saxon, och lägg det t.ex. under `/usr/java/saxon`.

3.1.1.4. DocBook Stylesheets

Stylesheets kan hämtas hos SourceForge (http://sourceforge.net/projects/docbook). Det är 'Docbook-xsl' som skall hämtas, jag kör version 1.61.2. Mandrake 9.1 har 1.60.1 installerat och det duger bra. Finns som `/usr/share/sgml/docbook/xsl-stylesheets-1.60.1`. Det är ganska vettigt att skapa katalogen `/usr/share/sgml/docbook/xsl-stylesheets` som en symbolisk länk till den docbook-version som används i systemet. Då blir det lätt att byta mellan versioner, om man vill. Mandrake 9.1 har satt upp det så.

Dessutom behövs en DTD, Mandrake har xml-dtd-4.1.2, som finns som `/usr/share/sgml/docbook/xml-dtd-4.1.2` men vill man den nyaste är det 4.2, som finns hos Oasis (http://www.oasis-open.org/docbook/).

3.2. Skaffa fram rådatat

Vi utgår från att vi har PostgreSQL körande, med möjlighet att koppla upp via TCP/IP. Vi har endast 2 tabeller tills vidare, skapade så här:

```
create table master (
  m_id integer,
  some_data char(20),
  primary key (m_id));

create table slave (
  m_id integer,
  s_id integer,
  some_other_data char(30),
  primary key (m_id, s_id),
  foreign key (m_id) references master (m_id));
```

Vi ska nu få till en fil med detta useende:

```
<Tables>
  <Table>
    <Name>
    <Keys>
      <Key>
        <Name>
        <IsPrimary>
        <IsUnique>
        <Definition>
      </Key>
    <Keys>
    .
    .
```

```
    <Relations>
    </Relations>
    .
    .
    <Fields>
      <Fielddesc>
        <Name>
        <Type>
        <NotNull>
        <HasDef>
        <FieldNum>
      </Fielddesc>
    </Fields>
    .
    .
  </Table>
  .
  .
</Tables>
```

Vi bryr oss inte om relationerna i detta exempel.

Ett sätt att få fram detta utseende är att köra detta tcl-skript. Vi kör det genom att skicka stdout till `tables.xml`. Databasen vi skall dokumentera heter bnl. Se till att skriptet är exekverbart.

```
[bnl@della source]$make_tables.tcl bnl > tables.xml
```

```
# ExtractDefinition :
# Get the description of tables in database provided
# as first argument to the script, in XML format, to stdout
# Björn Lundin 2003-06-21

#!/bin/sh
# \
exec pgtclsh $0 $*

proc getFields {Connection The_Table} {
        set Fields {}
        set res [pg_exec $Connection \
          "SELECT a.attname, \
             pg_catalog.format_type(a.atttypid, a.atttypmod), \
             a.attnotnull, a.atthasdef, a.attnum \
           FROM pg_catalog.pg_attribute a \
           WHERE a.attrelid = (select relfilenode from pg_catalog.pg_class \
                               where relname = '$The_Table') \
           AND a.attnum > 0 AND NOT a.attisdropped \
           ORDER BY a.attnum"]
        set ntups [pg_result $res -numTuples]
        for {set i 0} {$i < $ntups} {incr i} {
```

```
           lappend Fielddata [pg_result $res -getTuple $i]
        }
        puts "    <Fields>"
        foreach Field $Fielddata {
          puts "      <Fielddesc>"
          puts "        <Name>[lindex $Field 0]</Name>"
          puts "        <Type>[lindex $Field 1]</Type>"
          puts "        <NotNull>[lindex $Field 2]</NotNull>"
          puts "        <HasDef>[lindex $Field 3]</HasDef>"
          puts "        <FieldNum>[lindex $Field 4]</FieldNum>"
          puts "      </Fielddesc>"
        }
        puts "    <Fields>"
        pg_result $res -clear
}

proc getKeys {Connection The_Table} {
        set Fields {}
        set res [pg_exec $Connection \
          "SELECT c2.relname, i.indisprimary, i.indisunique, \
                 pg_catalog.pg_get_indexdef(i.indexrelid) \
           FROM pg_catalog.pg_class c, pg_catalog.pg_class c2, pg_catalog.pg_index i \
           WHERE c.oid = (select relfilenode from pg_catalog.pg_class \
                          where relname = '$The_Table') \
           AND c.oid = i.indrelid AND i.indexrelid = c2.oid \
           ORDER BY i.indisprimary DESC, i.indisunique DESC, c2.relname"]
        set ntups [pg_result $res -numTuples]
        for {set i 0} {$i < $ntups} {incr i} {
           lappend Keydata [pg_result $res -getTuple $i]
        }
        puts "    <Keys>"
        foreach Key $Keydata {
          puts "      <Key>"
          puts "        <Name>[lindex $Key 0]</Name>"
          puts "        <IsPrimary>[lindex $Key 1]</IsPrimary>"
          puts "        <IsUnique>[lindex $Key 2]</IsUnique>"
          puts "        <Definition>[lindex $Key 3]</Definition>"
          puts "      </Key>"
      }
      puts "    </Keys>"
      pg_result $res -clear
}

proc getRelations {Connection The_Table} {
        puts "    <Relations>"
        puts "    </Relations>"
}


proc ExtractDefinition { db {host "localhost"} {port "5432"} } {
    set conn [pg_connect $db -host $host -port $port]
    #Put all tables in list
    set res [pg_exec $conn \
```

```
    "SELECT  c.relname as Name \
     FROM pg_catalog.pg_class c \
       LEFT JOIN pg_catalog.pg_user u ON u.usesysid = c.relowner \
       LEFT JOIN pg_catalog.pg_namespace n ON n.oid = c.relnamespace \
     WHERE c.relkind ='r' \
       AND n.nspname NOT IN ('pg_catalog', 'pg_toast') \
       AND pg_catalog.pg_table_is_visible(c.oid) \
     ORDER BY c.relname"]
    set ntups [pg_result $res -numTuples]
    for {set i 0} {$i < $ntups} {incr i} {
        lappend Tables [pg_result $res -getTuple $i]
    }
    pg_result $res -clear

    puts "<Tables>"
    foreach Table $Tables {
        puts "  <Table>"
        puts "    <Name>$Table</Name>"
    #For each table put keys
        getKeys $conn $Table
    #For each table put relations (implemented as dummy)
        getRelations $conn $Table
    #For each table put fields
        getFields $conn $Table
        puts "  </Table>"
    }
    puts "</Tables>"

    pg_disconnect $conn
    return ""
}

#Main
ExtractDefinition [lindex $argv 0]
```

3.3. Transformera till DocBook

I `tables.xml` har vi nu ett 'well-formed' XML dokument. Detta skall nu mappas mot docbook. XSL-transformering är precis vad docbook i XML går ut på. En XML-parser validerar dokumentet som skall översättas, mot en DTD, docbooks DTD. Om det är OK kan processen med att skapa måldokumentet fortsätta. Skapandet av måldokumentet sker genom att köra det genom ett antal XSL-stylesheet, som byter docbook-taggarna mot t.ex. HTML-taggar.

Problemet här, är att `tables.xml` inte alls stämmer med docbooks DTD. Vi löser det med att skicka in det i ett eget XSL-stylesheet, som byter ut XML-taggarna från `make_tables.tcl` till docbook-XML-taggar. Detta stylesheet kallar vi `to_docbook_table.xsl`. Om det verkar helt obegripligt, så gör det inte så mycket, XSL är inte helt enkelt.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml"/>

  <!-- Main -->
  <xsl:template match="/">
    <chapter> <!-- document root -->
      <title>
        Database Table Description
      </title>
      <xsl:apply-templates/>
    </chapter>
  </xsl:template>

  <!-- The table -->
  <xsl:template match="Tables">
    <para>  <!-- found node Tables, print something and continue -->
       This chapter was autogenerated to document the
       structure of database tables used in project XXX.
    </para>
    <xsl:apply-templates/>
  </xsl:template>

  <!-- The tables -->
  <xsl:template match="Table">
  <!-- Inside a table node, start a new section-->
    <xsl:variable name="Table_Name" select="Name"/>
    <sect1 id="{$Table_Name}">
      <title>
        <xsl:value-of select="$Table_Name"/>
      </title>
      <xsl:apply-templates/>
    </sect1>
  </xsl:template>

  <xsl:template match="Name"/>         <!-- surpress output again-->
  <xsl:template match="Description"/>  <!-- surpress output again -->

  <!-- Keys -->
  <xsl:template match="Keys">
  <!-- If there are any keys, process them, else say they are missing-->
    <xsl:choose>
      <xsl:when test="count(Key) > 0">
        <table>
          <title>Keys in <xsl:value-of select="../Name"/> </title>
          <tgroup cols="4">
            <thead>
              <row>
                <entry> Name </entry>
                <entry> Primary? </entry>
                <entry> Unique? </entry>
```

```
              <entry> Definition </entry>
            </row>
          </thead>
          <tbody>
            <xsl:apply-templates/>
          </tbody>
        </tgroup>
      </table>
    </xsl:when>
    <xsl:otherwise>
      <para>
        There are no keys.
      </para>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>

<xsl:template match="Key">
<!-- Fill the table -->
  <row>
    <entry>
      <xsl:value-of select="Name"/>
    </entry>
    <entry>
      <xsl:value-of select="IsPrimary"/>
    </entry>
    <entry>
      <xsl:value-of select="IsUnique"/>
    </entry>
    <entry>
      <xsl:value-of select="Definition"/>
    </entry>
  </row>
</xsl:template>

<!-- Relations -->
<xsl:template match="Relations">
  <xsl:choose>
    <xsl:when test="count(Relation) > 0">
      <table>
        <title>Relations for <xsl:value-of select="../Name"/> </title>
        <tgroup cols="5">
          <thead>
            <row>
              <entry> Description </entry>
              <entry> Constraints </entry>
              <entry> # of keys </entry>
              <entry> Index </entry>
              <entry> Foreign keys </entry>
            </row>
          </thead>
          <tbody>
            <xsl:apply-templates/>
```

```
          </tbody>
        </tgroup>
      </table>
    </xsl:when>
    <xsl:otherwise>
      <para>
        There are no relations.
      </para>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>

<xsl:template match="Relation">
  <row>
    <entry>
      <xsl:value-of select="Relatedtable"/>
    </entry>
    <entry>
      <xsl:value-of select="Delconstr"/>
    </entry>
    <entry>
      <xsl:value-of select="Keyno"/>
    </entry>
    <entry>
      <xsl:value-of select="Index"/>
    </entry>
    <entry>
      <xsl:value-of select="Foreignkeyfields"/>
    </entry>
  </row>
</xsl:template>

<!-- Fields -->
<xsl:template match="Fields">
  <xsl:choose>
    <xsl:when test="count(Fielddesc) > 0">
      <table>
        <title>Fields in <xsl:value-of select="../Name"/></title>
        <tgroup cols="5">
          <thead>
            <row>
              <entry> Fieldname </entry>
              <entry> Type </entry>
              <entry> Not null? </entry>
              <entry> Has defult? </entry>
              <entry> Field number </entry>
            </row>
          </thead>
          <tbody>
            <xsl:apply-templates/>
          </tbody>
        </tgroup>
      </table>
```

```
      </xsl:when>
      <xsl:otherwise>
        <para>
          There are no fields.
        </para>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <xsl:template match="Fielddesc">
    <row>
      <entry>
        <xsl:value-of select="Name"/>
      </entry>
      <entry>
        <xsl:value-of select="Type"/>
      </entry>
      <entry>
        <xsl:value-of select="NotNull"/>
      </entry>
      <entry>
        <xsl:value-of select="HasDef"/>
      </entry>
      <entry>
        <xsl:value-of select="FieldNum"/>
      </entry>
    </row>
  </xsl:template>
</xsl:stylesheet>
```

Om du nu har orkat hänga med, så är resten ganska enkelt. Vi vill skapa `docbook_tables.xml` som är datat från `tables.xml`, fast med docbook-syntax. Dags att utnyttja Saxon. Saxon behöver en del parametrar, så vi skapar en `Makefile`, som vi bygger på efter hand. Justera sökvägarna efter ditt system.

Vi skapar en katalogstrukur enligt nedan

```
[bnl@della docbook_xml]$tree
|-- source
|   |-- Makefile
|   |-- datamodel.xml
|   |-- make_tables.tcl
|   |-- tables.xml
|   `-- to_docbook_table.xsl
`-- target
    |-- html
    |-- html_one
    `-- pdf
```

där `Makefile` ser ut så här:

```
#A list over source files
SRC=datamodel.xml docbook_tables.xml tables.xml to_docbook_table.xsl
DB=/usr/share/sgml/docbook/xsl-stylesheets
SAXON=/home/bnl/distributions/java/saxon.jar
TARGET=/home/bnl/public_html/sslug/docbook_xml/target

OPTIONS=paper.type=A4 \
        admon.graphics=1 \
        admon.graphics.path=$(DB)/images/ \
        use.extensions=1 \
        fop.extensions=1 \
        section.autolabel=1 \
        callout.graphics.path=$(DB)/images/callouts/ \
        tablecolumns.extension=0

#What to do when just 'make' is run
all: docbook_tables.xml

docbook_tables.xml: tables.xml
 java -jar $(SAXON) tables.xml  to_docbook_table.xsl \
        $(OPTIONS) > docbook_tables.xml
```

Skapa docbook_tables.xml genom att skriva:

```
[bnl@della source]$make docbook_tables.xml
```

Vi har nu allt vi behöer för att komma i mål

3.4. Skapa ett HTML-dokument

För att skapa ett HTML-dokument behöver vi bara fylla på make-filen. Vi lägger till följande:

```
all: docbook_tables.xml html_one
html_one: $(SRC)
 java -jar $(SAXON) datamodel.xml  $(DB)/html/docbook.xsl \
        $(OPTIONS) > $(TARGET)/html_one/datamodel.html
 touch html_one
```

Som du ser är skillnaden bara argumenten till Saxon. Vilken källdatafil som skall användas, och vilket stylesheet som skall anvädas. `$(DB)/html/docbook.xsl` är startpunkten för HTML.

3.5. Skapa ett uppdelat HTML-dokument

För att skapa en HTML-struktur uppdelad per section behöver vi bara fylla på make-filen igen. Vi lägger till följande:

```
all: docbook_tables.xml html_one html
html: $(SRC)
 java -jar $(SAXON) datamodel.xml  $(DB)/html/chunk.xsl \
        $(OPTIONS)
 cp *.html $(TARGET)/html
 rm -f *.html
 touch html
```

Genom att använda `$(DB)/html/chunk.xsl` som XSL-startpunkt får vi ett annat resultat.

3.6. Skapa ett PDF-dokument

Båda HTML-varianterna klarar Saxon av i ett svep. Med PDF är det annorlunda. Saxon kan skapa utdata i FO-format, som är *Formatting Objects*. Det är ett sidbeskrivningsspråk/format i XML/XSL. FOP kan sedan konvertera detta till PDF. Det är alltså en tvåstegsraket. Vi fortsätter med make-filen.

```
all: docbook_tables.xml html_one html pdf
pdf: $(SRC)
 java -jar $(SAXON) datamodel.xml  $(DB)/fo/docbook.xsl \
        $(OPTIONS) > datamodel.fo
 $$FOP_HOME/fop.sh -fo datamodel.fo -pdf $(TARGET)/pdf/datamodel.pdf
 rm -f datamodel.fo
 touch pdf
```

Vi byter stylesheetet mot `$(DB)/fo/docbook.xsl` fö att få vårt fo-dokument, `datamodel.fo`. Sedan ber vi Fop att skapa en pdf-fil av fo-filen.

3.7. Parametrar

Vi har skickat in lite parametrar till Saxon, utan att kommentera dem.

- paper.type=A4 : Utan denna blir det US-letter
- admon.graphics=1 : Tjusiga figurer till t.ex <note> taggen
- admon.graphics.path=$(DB)/images/ : Sökväg till figurerna
- use.extensions=1 : Använd XSL/Fop utökningar
- fop.extensions=1 : Använd Fop utökningar

- section.autolabel=1 : Använd sektions-texter
- callout.graphics.path=$(DB)/images/callouts/ : Sökväg till bilderna i bildnumrerade listor
- tablecolumns.extension=0 : Förbättra kolumnstorlek i tabeller. Kräver att $CLASSPATH är satt till extensions/saxon(version).jar

Det finns många fler parametrar, dokumenterade i `$(DB)/doc/reference.html`

3.8. XML - SGML

Huvuddokumentet är väldigt likt dess SGML-kusin. Skillanden ligger i att ?xml? taggen skall finnas med, och att en URL till docbooks DTD skall finnas med. Jag brukar lägga in en lokal referens när jag jobbar med dokumentet, annars hämtas den in via nätverket varje gång, och det tar längre tid.

```
<?xml version="1.0"?>
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML V4.2//EN"
                          "file:///usr/share/sgml/docbook/xml-dtd-4.2/docbookx.dtd" [
]>
<!--
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML V4.2//EN"
                          "http://www.oasis-open.org/docbook/xml/4.2/docbookx.dtd" [
]>
-->
<book id="SSLUG" lang="en">
  <bookinfo>
    <title>En bra titel</title>
  </bookinfo>
  <toc id="toc"></toc>
  Inkluderat autogenererat dokument skapat enligt ovan
</book>
```

För att lyckas med DocBook i XML bör man tänka på följande:

- Alla taggar skall avslutas. Taggar utan avslutande tagg skall skrivas som tomma taggar. Ex <xref linkend=...../>. Notera den sista ’/’.
- Alla taggar skall skrivas med små bokstäver.
- Inga genvägar är tillåtna, som i SGML där man kan skriva <tag> ....</>

I övrigt är det väldigt likt SGML.

3.9. Komplett Makefile

Vi lägger till ett clean kommando:

```
#A list over source files
SRC=datamodel.xml docbook_tables.xml tables.xml to_docbook_table.xsl
DB=/usr/share/sgml/docbook/xsl-stylesheets
SAXON=/home/bnl/distributions/java/saxon.jar
TARGET=/home/bnl/public_html/sslug/docbook_xml/target

OPTIONS=paper.type=A4 \
        admon.graphics=1 \
        admon.graphics.path=$(DB)/images/ \
        use.extensions=1 \
        fop.extensions=1 \
        section.autolabel=1 \
        callout.graphics.path=$(DB)/images/callouts/ \
        tablecolumns.extension=0

#What to do when just 'make' is run
all: docbook_tables.xml html html_one pdf

html_one: $(SRC)
 java -jar $(SAXON) datamodel.xml  $(DB)/html/docbook.xsl \
        $(OPTIONS) > $(TARGET)/html_one/datamodel.html
 touch html_one

html: $(SRC)
 java -jar $(SAXON) datamodel.xml  $(DB)/html/chunk.xsl \
        $(OPTIONS)
 cp *.html $(TARGET)/html
 rm -f *.html
 touch html

pdf: $(SRC)
 java -jar $(SAXON) datamodel.xml  $(DB)/fo/docbook.xsl \
        $(OPTIONS) > datamodel.fo
 $$FOP_HOME/fop.sh -fo datamodel.fo -pdf $(TARGET)/pdf/datamodel.pdf
 rm -f datamodel.fo
 touch pdf

docbook_tables.xml: tables.xml
 java -jar $(SAXON) tables.xml  to_docbook_table.xsl \
        $(OPTIONS) > docbook_tables.xml

.PHONY: clean
clean:
 rm -f docbook_tables.xml pdf html_one html
 rm -f $(TARGET)/pdf/*
 rm -f $(TARGET)/html/*
 rm -f $(TARGET)/html_one/*
```

3.10. Problem/Fördelar/Nackdelar

När dokumenten blir stora, kan man råka ut för att Fop eller Saxon krashar på ’out-of-memory’. Jag upplever detta vid 100+ tabeller. Lösningen är att skicka med -Xmx128m till java, dvs tillåt att java får använda 128 mb minne.

Stora dokument tar tid, framför allt om PDF skall genereras.

Java/XML/XSL är mycket flexibelt.

Enkelt att sätta upp, få verktyg.

Saxon och Fop kan ersättas med andra XSLT verktyg.

Kapitel 4. LaTeX

Dette afsnit er ment som en introduktion til LaTeX. Der kræves ingen forudsætninger ud over at du skal kunne bruge en computer. LaTeX minder i de grundlæggende *principper* om HTML eller SGML (DocBook), så kender du lidt til dét, får du ingen problemer (ellers kan det sagtens læres!).

Målgruppen for dette kapitel er absolutte nybegyndere udi LaTeX, der bare gerne vil igang. Alle kan være med og du behøver ikke være en ørn til Linux for at kunne lave noget *professionelt* i LaTeX. Faktisk har LaTeX som sådan slet ikke noget med Linux/Unix at gøre, man kan også få LaTeX til Windows, MacOS, Amiga og andre systemer, faktisk er LaTeX nok det mest portable format for formateret tekst man kan finde!

Dette kapitel beskæftiger sig overhovedet ikke med det der ellers er en af de største styrker ved LaTeX, nemlig mulighederne for at skrive matematiske formler. Årsagen er ganske enkelt at det ville fordoble omfanget af kapitlet, og det har der ganske enkelt ikke været tid til. Derudover er det ikke det sværeste at lære når man først har forstået de grundlæggende ting.

Rigtig god fornøjelse!

4.1. Indledning om LaTeX

I (slet ikke så) gamle dage talte man om bogtrykker*kunsten* og overlod fremstilling af trykt materiale til dem (bogtrykkerne) der havde forstand på det. Efter at computere er blevet udbredt forventer de fleste uden videre at kunne producere professionelt udseende tekster.

Det er ikke så naivt som det kunne lyde, men det kræver enten at man lærer en hel del om emnet, eller at man anvender et egnet værktøj. Sådanne findes der kun ét af: LaTeX.

Udover at være det bedste værktøj til produktion af professionelt udseende tekster, er LaTeX det eneste værktøj der er bare nogenlunde velegnet til produktion af tekniske tekster, det er det til gengæld også suverænt til.

Figur 4-1. LaTeX-logoet

Enhver tekst om LaTeX skal indeholde et afsnit om navnet, og denne er naturligvis ingen undtagelse. LaTeX er en udvidelse til TeX, og navnet er følgelig afledt deraf, de sidste tre bogstaver er i virkeligheden de tre store græske bogstaver Tau, Epsilon og Chi, og de skal derfor udtales som sådan. Derfor udtales LaTeX [latæk]. Med mindre man har mulighed for at gengive logoet præcist (se figur Figur 4-1) bør man skrive LaTeX, med et lille a og et lille e, som det er gjort her. Det siger sig selv at man har mulighed for at producere det rigtige logo når man skriver i LaTeX, det gøres med kommandoen **\LaTeX** (det var så din første LaTeX-kommando).

Klassiske tekstbehandlingsprogrammer er bygget op, så det du har på skærmen svarer til det du får ud på printeren. Disse programmer kaldes tit "What You See Is What You Get"-programmer (WYSIWYG). Dette kan *både* være en fordel, og en ulempe. Fordelen er klar, hvis man ikke er så tryg ved at bruge en computer, eller hvis det altid er meget små ting man skal have skrevet ned. Men lige så snart teksterne bliver lidt længere eller en lille smule mere avancerede, viser begrænsningerne i det koncept sig.

I modsætning hertil kunne LaTeX kaldes et "What You Need Is What You Get"-program. Det skal forstås sådan at medmindre man selv beder om det, får man noget der ser pænt ud! LaTeX kan virke skræmmende første gang man støder på det, og det kan virke omstændigt hvis det er meget en kort tekst man skal have skrevet, men hvis man blot har en standardpræambel (det vender vi tilbage til) er det kun selve teksten der skal skrives.

Er man flere, der arbejder på et dokument/rapport, sparer man tid ved at sætte sig ind i LaTeX. Her kan man nemlig dele et dokument op i lige så mange filer man vil. Det gør, at man kan skrive mange på én gang, uden at skulle tænke på om de andre gemmer oven i, og man mister sin tekst. Endvidere skal man ikke sidde og fedte med indholdsfortegnelsen, for at få den til at passe, rette alle henvisninger til sidst, lave hele litteraturlisten fra starten - og mange mange andre ting!

Der er dog en grund til ikke at lære LaTeX: Erfaringen viser at man kommer til at ærgre sig over at man ikke lærte LaTeX for længe siden, fordi det ville have sparet en for en masse arbejde gennem årene.

LaTeX har to ulemper: Den ene er at det man bliver nødt til at sætte sig ned og læse en times tid (f.eks. denne tekst) for at lære det. Den anden er at man for at se hvad man egentlig har lavet skal have et program som kan vise .dvi-filerne - sidstnævnte er mest et problem, hvis man skriver noget i LaTeX til folk der kun kan bruge Windows. Der findes dog DVI-fremvisere til Windows, efter sigende skulle Yap (Yet Another Previewer) være god. Yap er en del af MikTeX 2.x hvis hjemmeside hedder http://www.miktex.org/.

Et LaTeX-dokument skrives som en tekstfil i ens favoriteditor (hvad enten den så hedder Emacs, vi, pico eller noget helt fjerde). Det betyder at du føler dig hjemme når du skriver din tekst, og ikke hele tiden skal huske hvordan man gør dit og dat.

LaTeX-dokumentet er som sagt ren tekst. Denne kan betragtes som kildeteksten til f.eks. et program. LaTeX-dokumentet skal oversættes, før man får et resultat, det er her man bruger programmet **latex** (bemærk at programmet som de fleste andre Unix-programmer har et navn med rent små bogstaver – Dette er det eneste tilfælde hvor det kan tillades ikke at skrive LaTeX).

Uddata fra **latex** er en dvi-fil (DVI står for DeVice Independent), som man oftest vil vælge at bruge **xdvi** til at se. Hvis man skal se på dvi-filer, der inkluderer postscript-figurer, skal man dog være opmærksom på at det kan ske at **xdvi** ikke viser figurerne korrekt. I det tilfælde bør man bruge **dvips** til at oversætte dvi-filen til Postscript, og få vist postscript-filen med Ghostview (**gv**). Hvis LaTeX-dokumentet hedder `fil.tex` vil det for eksempel kunne gøres således:

```
[tyge@hven tyge]$ latex fil
...
[tyge@hven tyge]$ dvips fil -o
...
[tyge@hven tyge]$ gv fil.ps &
```

I www.linuxbog.dk/dokumentation/eksempler/latex_eks/ (http://www.linuxbog.dk/dokumentation/eksempler/latex_eks/) ligger der et lille LaTeX-dokument kaldet `eks1.tex`, som du passende kan prøve at hente, og udsætte for følgende:

```
[tyge@hven tyge]$ latex eks1.tex
...
[tyge@hven tyge]$ xdvi eks1.dvi &
```

Denne bog tager kun det mest basale med, for bare at komme igang! Ting der ikke er uddybet, kan undersøges nærmere i "The Not So Short Introduction to LaTeX (http://www.sunsite.auc.dk/ftp/pub/tex/ctan/info/lshort/english/lshort.ps)", som kan hentes gratis på nettet.

Inden vi går i gang, har vi lige et citat fra en underviser på Aalborg Universitet:

Microsoft Word er et rigtig godt program... specielt hvis man vil have plads på sin harddisk – så kan man jo slette det!

4.2. Opbygning af et LaTeX-dokument

Som sagt skriver man sit dokument i en editor efter eget valg, og oversætter det herefter. Før vi går i detaljer med indholdet af den fil man kan oversætte, skal vi lige se på den overordnede opbygning af filen.

Den grundlæggende struktur i et LaTeX-dokument fremgår af følgende

```
Præambel
\begin{document}
Indhold
\end{document}
```

Hvad *indhold* dækker over er ikke så svært at regne ud, derimod siger ordet *præambel* måske ikke umiddelbart så meget.

Præamblen er det sted man fortæller LaTeX hvilken type tekst det er man skriver (om det er en artikel, en rapport eller måske en hel bog), og andre ting om teksten som helhed (f.eks. hvilken skriftstørrelse man ønsker). Det er også her man fortæller hvilke ekstra pakker man ønsker at gøre brug af (der findes pakker til de mærkeligste ting). Hvis man er avanceret er det også her man definerer sin egne kommandoer.

I langt de fleste tilfælde vil man skrive det samme i sin præambel som sidst, og derfor anvender har de fleste LaTeX-brugere en standardpræambel, som de har opbygget over tid, oftest på grundlag af en de har fået af en anden. Den mulighed skal du naturligvis også have, så derfor er der til denne tekst udarbejdet et par (faktisk 4) præambler til diverse formål. Præamblerne kan hentes gratis på www.linuxbog.dk/dokumentation/eksempler/latex_eks/ (http://www.linuxbog.dk/dokumentation/eksempler/latex_eks/), til gengæld vil vi i denne tekst ikke gøre ret meget ud af indholdet af præamblen (men måske kommer der et afsnit til senere). Disse præambler er tilpasset denne tekst, eftersom vi ikke kommer til at beskæftige os med produktion af stikordsregister giver præamblerne heller ikke adgang til de nødvendige kommandoer til dette. Bemærk at LaTeX ikke er særlig glad for danske bogstaver i filnavne og derfor er præambel stavet på engelsk i filnavnene.

Der er forskellige måder at bruge sin standardpræambel på. En løsning er at kopiere den ind i hver eneste LaTeX-fil man skriver (det kan man få nogle editorer bl.a. emacs til at gøre automatisk. En anden løsning (som vi vil gøre brug af) er at bede LaTeX inkludere den, det kan man gøre med kommandoen **\input{preamble}** (det var din anden LaTeX-kommando).

Nu kan vi faktisk lave et minimalt LaTeX-dokument:

```
\input{preamble}
\begin{document}
```

```
Hej SSLUG!
\end{document}
```

Hvis du prøver at køre det her gennem LaTeX er resultatet en side med teksten "Hej SSLUG!", ikke så ophidsende men nu er vi da i gang.

4.3. Basale kommandoer

I ethvert LaTeX dokument har du brug for en række kommandoer. Alt efter hvad dit dokument omhandler, er disse mere eller mindre avancerede. Det lyder måske skræmmende at skulle lære en række kommandoer, men du får hurtigt lært de fleste af dem du nogensinde vil få brug for. Generelt er alle kommandoer man bruger i LaTeX logiske, det betyder at du skal koncentrere dig om hvad teksten er i stedet for hvordan den skal se ud.

Først en række kommandoer du kan bruge til at dele din tekst op i afsnit. Husk at en fornuftig opdeling gør teksten mere overskuelig. De kommer her i den rækkefølge der svarer til deres naturlige orden.

- **\chapter{Kapitlets navn}** (Kun i rapporter og bøger) Begynder et nyt kapitel, bemærk at LaTeX selv holder styr på hvad nummer du er kommet til, så det er kun navnet du skal tænke på! Dette gælder også de kommende kommandoer.
- **\section{Afsnitsoverskrift}** Denne kommando laver et nyt afsnit med en selvstændig overskrift
- **\subsection{Underafsnitsoverskrift}** Laver en undersektion med en selvstændig overskrift.
- **\subsubsection{Underunderafsnitsoverskrift}** Laver en underundersektion med sin egen overskrift.

Bemærk at LaTeX selv holder styr på hvad nummer du er kommet til, det betyder at det *kun* er sektionens navn du selv skal angive. LaTeX sørger også for at nummerere ordentligt i forhold til de overordnede inddelinger. Det betyder også at man skal sørge for at et afsnit på et givet niveau er indeholdt i et afsnit på et højere niveau. Hvis du f.eks. kommer til at begynde en **\subsection** før den første **\section** vil den få nummer 0.1 (måske med et kapitelnummer foran), ikke videre elegant.

En kommando der er beslægtet med de nævnte er **\tableofcontents**, som indsætter en *indholdsfortegnelse* for hele dit dokument. Indholdsfortegnelsen bliver dannet udfra oplysninger der blev samlet sidste gang du kørte dit dokument gennem LaTeX, derfor skal du første gang du oversætter dokumentet huske at *oversætte to gange*, hvis du vil have en indholdsfortegnelse med indhold. Ved efterfølgende kørsler plejer LaTeX at opdage hvis der er noget der har flyttet sig og opfordrer så til at man kører LaTeX endnu engang. Hvis man har et meget stort dokument kan man endog komme ud for at skulle køre det igennem LaTeX tre gange før indholdsfortegnelsen bliver rigtig.

Hvis man har en sektion man ikke vil have med i sin indholdsfortegnelse har alle kommandoerne en *-variant (**\chapter*, \section*,** osv.) som får LaTeX til at undlade at give sektionen et nummer og udelade den fra indholdsfortegnelsen.

4.4. Formatering

Nu er tiden vist kommet til at se lidt på hvordan vi får skrevet noget reelt tekst, og hvordan vi kan formatere det.

I udgangspunktet drejer det sig naturligvis om bare at skrive den tekst man vil have. Det er selvfølgelig nødvendigt at man sætter mellemrum mellem ordene, men det er ligegyldigt om man sætter et eller flere. Man kan også skrive hvert ord på sin egen linje, LaTeX er ligeglad, afstanden mellem ordene bliver regnet ud så man får en lige højre margen. Det eneste man ikke må er at lave blanke linjer, det fortolker LaTeX som at man ønsker et nyt afsnit. Bemærk at kommandoer der ikke tager et argument som f.eks. **\LaTeX** "sluger" de efterfølgende mellemrum, så hvis man vil have mellemrum bagefter skal man sætter **{}** efter disse kommandoer. Dette betyder at udsagnet "LaTeX, LaTeX og mere LaTeX", skal skrives som

```
\LaTeX, \LaTeX{} og mere \LaTeX.
```

Med undtagelse af det første afsnit efter en af sektioneringskommandoerne nævnt ovenfor, indrykker LaTeX selv den første linje i et afsnit. Formålet er gøre det nemmere at se hvor der begynder et afsnit, hvilket f.eks. er rart at kunne når man skimmer en tekst.

Der er 10 tegn der har en speciel betydning i LaTeX, og som følgelig kan være lidt besværlige at lave, det drejer sig om $, &, %,#, _, {, }, ~, ^ og \. De syv første kan indsættes ved simpelthen at sætte en bagstreg (\) foran, dvs. med **\$**, **\amp;**, **\%**, **\#**, _, **\{** og **\}**. De sidste tre er lidt sværere at lave, og der findes flere løsninger (det gør der også for nogle af de 7 første), som har hver deres fordele. En løsning er kommandoerne **\textasciitilde**, **\textasciicircum** og **\textbackslash**.

Man fremhæver tekst med kommandoen **\emph{tekst der skal fremhæves}**. Så vil LaTeX sædvanligvis skrive teksten med kursiv, og hvis man af en eller anden grund allerede skriver med kursiv, med ikke-kursiv. Bemærk at man beder om at få *fremhævet* teksten, ikke om at få den skrevet med kursiv/fed/... Det er der flere fordele ved, en er at det gør det meget nemt at ændre hvordan fremhævet tekst skal se ud, en anden er at man bruger den samme kommando lige meget om det er i almindelig tekst, eller f.eks. i en matematisk sætning som man i forvejen bruger kursiv til.

Bemærk at nogen bruger formen **{\em tekst}**, det er en gammeldags måde at gøre det samme på, og kræver at man har overblik over virkefeltet. Det kan ikke anbefales at bruge denne form.

Da der nærmest ikke er nogen grænse for hvor mange årsager der kan være til at markere et stykke tekst, og det (med undtagelse af fremhævning) i hvert fald ikke er de samme fra tekst til tekst, har LaTeX ikke logiske kommandoer til markering af andre årsager. Derimod findes et udvalg af kommandoer til at bestemme hvordan teksten fysisk skal se ud, som man så kan benytte til at definere sine egne logiske kommandoer.

Hvis man f.eks. skriver en tekst med en masse filnavne, kan man bruge en definition i stil med

```
\newcommand{\filnavn}[1]{\textbf{#1}}
```

der fortæller at filnavne skal sættes med fed (**\textbf** bruges til at lave fed skrift). Denne definition bør placeres i præamblen. Hvis du bruger **\input** for at benytte en standardpræambel, er det smartest at placere definitionen mellem **\input{preambel}** og **\begin{document}**. Kommandoen **\newcommand** er noget mere avanceret end antydet her, men det vil vi ikke gå ind i nu.

Der findes naturligvis andre muligheder end fed skrift, her er en oversigt over hvad man kan bruge i stedet for **\textbf** i definitionen ovenfor (uden at lave andre ændringer):

Tabel 4-1. Oversigt over de kommandoer man kan bruge til at påvirke teksts udseende

Kommando	Effekt
\textbf	Fed skrift
\texttt	Skrivemaskineskrift
\textsl	Hældende skrift (ikke det samme som kursiv)
\textsf	Sans serif-skrift
\textsc	Kapitæler (Formindskede store bogstaver)

4.5. Omgivelser

Omgivelser (på engelsk: environments) er nogle kommandoer, som man bruger til at omgive større mængder tekst, som af en eller andre grund udgår en enhed. Et eksempel herpå er f.eks. punktopstilling. Denne omgivelse hedder "itemize".

Man begynder en punktopstilling med **\begin{itemize}**, og afslutter den med **\end{itemize}**. Imellem disse to kommandoer begynder man et nyt punkt med **\item**. En hel punktopstilling kommer altså til at se sådan her ud:

```
\begin{itemize}
\item Første punkt
\item Andet punkt
  \begin{itemize}
  \item Første underpunkt til andet punkt
  \item Andet underpunkt til andet punkt
  \end{itemize}
\item Tredje punkt
\end{itemize}
```

hvor man også kan se at man sagtens kan have punktopstillingen i flere niveauer. LaTeX sørger selv for at bruge forskellige symboler foran punkterne på forskellige niveauer.

Vil du gerne have nummereret punktopstilling hedder omgivelsen "enumerate". LaTeX holder selvfølgelig selv styr på hvilke numre de forskellige punkter har, du skal bare sætte dem i den rigtige rækkefølge. Ligesom før kan du sagtens have punktopstillinger (både nummererede og unummererede) inden i et punkt, for nummererede punktopstillinger bruger LaTeX så a,b,c...

En tredje form for punktopstilling kan laves med omgivelsen **description**, der fungerer stort set som **itemize** og **enumerate**. Der er dog den forskel at **\item** tager et argument og giver hængende indrykning.

En sidste omgivelse det kan være nyttigt at stifte bekendskab med på nuværende tidspunkt er **center**, som centrerer indholdet på linjen. Eftersom LaTeX normalt sørger for en lige højre margen, er **center** sjældent anvendelig på længere tekststykker. En praktisk anvendelse kunne være

```
\begin{center}
Med venlig hilsen
\end{center}
```

i slutningen af et brev.

4.6. Henvisninger m.m.

Fodnoter laves med **\footnote{Fodnotetekst}** på det sted i teksten hvor fodnoten skal være. Behøver man at skrive at du blot skal skrive fodnoten, og så holder LaTeX styr på dem.

Hvis man har brug for at have krydshenvisninger til teksten, sætter man et mærke på det sted man har brug for at henvise til med **\label{Mærkenavn}**. Det kan man så henvise til med **\ref{Mærkenavn}**. Som udgangspunkt kommer henvisningen til at bestå af nummeret på den nærmest foregående sektioneringskommando (se afsnit Afsnit 4.3), men i forbindelse med figurer (se afsnit Afsnit 4.7), tabeller, ligninger og andre ting der har særskilte numre benyttes disse. Hvis man hellere vil have en henvisning til den side mærket er på, bruges **\pageref{Mærkenavn}**. Hvis man både vil have en henvisning til afsnit og sidetal, er kommandoen **\vref**[l] et godt bud, den kan bl.a. også finde ud af at skrive foregående/næste side i stedet for en sidetal, hvis det er relevant.

Du kan lave en litteraturliste i dit dokument med omgivelsen **thebibliography**, hvori **\bibitem{navn}** bruges til at angive hver kilde. Så kan du indsætte en reference til den pågældende kilde med **\cite{navn}**. Et eksempel:

```
\begin{thebibliography}{9}
\bibitem{tnss} Tobias Oetiker et al.: \emph{The not so short...}
\bibitem{lamport} Leslie Lamport: \emph{\LaTeX{} A Document
  Preparation System}
\end{thebibliography}
```

vil lave en litteraturliste med de to bøger i, og gøre det muligt at henvise til dem med henholdsvis **\cite{tnss}** og **\cite{lamport}**. Nitallet efter **\begin{thebibliography}** fortæller LaTeX, hvor langt det

største mærke er, LaTeX tager længden af den tekst der står der og bruger den til at skabe et pænt udseende, som oftest vil man skrive enten 9 eller 99 der.

4.7. Billeder

Der findes mange måder at sætte billeder ind i dokumenterne på, min anbefaling er at konvertere alt til indkapslet Postscript (EPS/Encapsulated PostScript), og sætte dem ind med følgende kode:

```
\begin{figure}[htbp]
\begin{center}
\includegraphics{billede.eps}
\end{center}
\caption{Billedtekst.}
\label{billedmærke}
\end{figure}
```

Hvis man bruger pdfLaTeX (kommandonavn: **pdflatex**) er det muligt at inkludere jpg-, png- og gif-filer direkte, men til gengæld ikke eps.

Omgivelsen **figure** fortæller LaTeX at indholdet er en figur, som den skal placere et sted hvor det vil se pænt ud, efter de retningslinjer der er angivet i kantede parenteser bagefter. Eksemplet viser de fire muligheder der er, **h** beder om at få figuren placeret her, **t** beder om at få det øverst på en side, **b** om at få det nederst på en side og **p** siger at LaTeX må placere det på en side der kun indeholder figurer. Rækkefølgen er helt ligegyldig. Hvis man ikke angiver nogen retningslinjer er standardindstillingen **tbp**.

At LaTeX som standard placerer figurer der hvor det bliver pænest bør ikke være et problem, hvis man sørger for en fornuftig billedtekst, og det relevante sted i teksten indsætter en reference med **\ref{billedmærke}**.

4.7.1. Xfig

Skal du tegne dine egne billeder, kan du med fordel anvende programmet Xfig. Det kan gemme i LaTeX-format og postscript (m.fl.), hvilket resulterer i nogle *meget* skarpe og flotte figurer. Programmet kræver lige som LaTeX lidt tid at sætte sig ind i – men det er tiden værd!!

4.7.2. MetaPost

En anden mulighed er at lave sin grafik i MetaPost. Det er sublimt til præcisionsarbejde, det har en glimrende pakke til at tegne grafer og så kan man inkludere LaTeX-formateret tekst! Læs mere om MetaPost på http://cm.bell-labs.com/who/hobby/MetaPost.html

4.8. Tabeller

Tabeller kan laves ved hjælp af omgivelsen **tabular**. Efter **\begin{tabular}** skal man i krøllede parenteser angive hvordan indholdet i de forskellige søjler skal justeres, og om der skal være en streg mellem to søjler. Man angiver at en søjle skal være venstrejusteret ved med et **l**, at den skal være centreret med et **c**, at den skal være højrejusteret med et **r** og at der skal være en streg mellem to søjler med en |. Hvis man vil have en tabel med en venstrestillet, en centreret og to højrestillede søjler og en streg mellem de to højrestillede, bliver det alt i alt til:

```
\begin{tabular}{lcr|r}
```

Indholdet angives række for række, med de enkelte felter adskilt af **&** og rækker adskilles af ****. Vandrette linjer mellem to rækker laves med kommandoen **\hline**.

Ofte har man ikke behov for at få tabellen placeret et bestemt sted, men snarere et sted hvor det ser pænt ud. Her kan man så benytte en mekanisme fuldstændig analog til den for billeder, den hedder blot **table** i stedet for **figure**.

Et fuldstændigt eksempel på en tabel som LaTeX får lov at bestemme placeringen af:

```
\begin{table}[htpb]
\begin{tabular}{|l|l|r|}
\hline
Firmanavn & Adresse & Postnummer \\
\hline
Vagabondos & Tagensvej 100 & 2200 \\
\hline
DKUUG & Fruebjergvej 3 & 2100 \\
\hline
Niels Bohr Instituttet & Blegdamsvej 17-21 & 2100 \\
\hline
\end{tabular}
\caption{Steder af betydning for SSLUG}
\label{tabelmærke}
\end{table}
```

4.9. Diverse

Man kan indsætte kommentarer (som ikke kommer med i det oversatte dokument) i sin LaTeX-fil, det gøres med et %-tegn, som gør resten af linjen til en kommentar. Det kan f.eks. bruges til at huske sig selv

på at et afsnit ikke er færdigskrevet, eller til at placere tydelige identifikationer af at man f.eks. begynder på et nyt afsnit, det kan f.eks. gøres som:

```
...

%---------- Her begynder et nyt afsnit kaldet: Næste afsnit ----------
\section{Næste afsnit}
```

Hvis man så senere har behov for at rette noget i det afsnit, er det nemmere at finde på grund af denne linje.

Der er forskel på vandrette streger! En bindestreg er kortere og lidt bredere end en tankestreg, og de laves derfor på forskellig vis. En bindestreg laver man med en enkelt **-**, mens en tankestreg laves med to -'er efter hinanden (dvs. **--**). To -'er bruges også til at angive intervaller som f.eks. 1991–2002. På dansk laves en tankestreg med mellemrum omkring, altså som "tekst – tekst", mens det på amerikansk laves uden mellemrum og med tre streger (**---**) "tekst—tekst".

Det rette udseende og placering af gåseøjne kan diskuteres længe. Dansk tradition er at åbne med "nitaller forneden" og lukke med "sekstaller foroven", sådanne gåseøjne laves med **"‘** og **"’**. Engelsk tradition er at åbne med "sekstaller foroven" og lukke med "nitaller foroven", sådanne laves med **“** og **”**.

Hvis du har brug for dags dato, så fås den med kommandoen **\today**.

En enkel overskrift laves med kommandoen **\maketitle**, som forudsætter at du har defineret en titel med **\title{Dokumentets titel}**, forfatter med **\author{Dit navn}**, og evt. en dato med **\date{Dato}**. Hvis datoen blot skal være dags dato, er det ikke nødvendigt at definere denne, det er nemlig standard. Det kan måske være en idé at sætte **\author{Dit navn}** ind i din præambel, så behøver du ikke skrive det hver gang.

4.9.1. Præambler

Som skrevet tidligere, vil vi ikke gøre noget ud af indholdet af præamblen. Her følger blot nogle korte beskrivelse af de standardpræambler der er lavet til denne tekst, og som kan findes på www.linuxbog.dk/dokumentation/eksempler/latex_eks/ (http://www.linuxbog.dk/dokumentation/eksempler/latex_eks/).

Den første præambel (`preamble1.tex`) er tænkt til korte tekster (uden teknisk indhold). Det betyder at den siger til LaTeX at dokumentet er en artikel, hvilket igen betyder at du ikke kan lave kapitler. Derudover kan du ikke indsætte billeder eller bruge kommandoen **\vref**.

Den anden præambel (`preamble2.tex`) er tænkt til længere tekster uden teknisk indhold. Det betyder at LaTeX får at vide at den skal opfatte teksten som en rapport, hvorfor du kan lave kapitler. Faktisk skulle du have mulighed for at bruge alle de ting du har læst om her.

Derudover indeholder kataloget to præambler mere (`preamble3.tex` og `preamble4.tex`) som er tænkt til henholdsvis kortere og længere tekster med et mere teknisk indhold. Eftersom vi her ikke har beskæftiget os med de muligheder er de nok ikke interessante i første omgang.

4.10. Afsluttende bemærkninger

Om jeg så havde haft alverdens papir til rådighed bliver jeg aldrig færdig med at fortælle om LaTeX. Det skyldes (som jeg håber du har fornemmet indtil nu) at kombinationsmulighederne er ganske ufattelige. Selv efter at have brugt LaTeX til stort set alt i over 6 år, kan jeg stadig lære nye ting (en del af det er dog nærmere ren TeX), og ved der er masser af ting jeg ikke forstår. Selvom dette kun er en kort introduktion håber jeg dog at du har fået noget ud af det. Har du det har du gjort mig glad! :-)

Der er skrevet meget litteratur om LaTeX, men ikke ret meget på dansk, og slet ikke noget der ikke har fået mig til at krumme tæer over alle de uvaner der blev videregivet til de uvidende stakler der kom for skade at læse det. Derfor denne tekst. Hvis du nu har fået lyst til at lære mere om LaTeX, så er her en række bøger der er ret populære:

- *The Not So Short Introduction To LaTeX2e*, af Tobias Oetiker m.fl. Dette er en gratis bog, som du kan finde på internettet. Jeg vil anbefale dig at hente den hjem og skrive den ud, for der er et godt stikordsregister bag i. Du kan finde den på adressen http://www.sunsite.auc.dk/ftp/pub/tex/ctan/info/lshort/english/lshort.ps.
- *The LaTeX Companion*, Denne bog indeholder alt hvad der har med "normale" dokumenter at gøre. Der bliver henvist til den, fra Oetiker og fra selve oversætteren af LaTeX, hvis du har lavet en fejl. Koster lige omkring 500,- kr.
- *Guide to LaTeX*, af Helmut Kopka og Patrick W. Daly. En bog der ser ud til at give en grundig dækning af et rimelig bredt spektrum af emner. Denne bog koster omkring 650,- kr.
- *The LaTeX Graphics Companion*, af Michel Goossens m.fl. Denne bog indeholder alt hvad der har med grafik at gøre, hvad enten det er billeder, grafer, noder, streger, kurver eller figurer - simpelthen alt. Koster lige omkring 500,- kr.
- *The LaTeX Web Companion*, Denne bog handler om hvordan man integrerer sine kundskaber i LaTeX på internettet. Den er ret ny og rimelig spændende. Koster ligeledes lige omkring 500,- kr.

Alle bøgerne udgives af Addison-Wesley, og en del af overskuddet går til LaTeX3-projektet, så ved at købe dem støtter du også udviklingen af nye og bedre versioner af LaTeX.

Slutbemærkning:

1. **\vref** er ikke en standardkommando i LaTeX, men stammer fra pakken varioref, som inkluderes hvis du benytter enten preambel2.tex eller preambel4.tex

Kapitel 5. DocBook med WYSIWYG

Hvis du ikke er så stærk i at anvende DocBook, eller blot vil have en mulighed for at fremstille en hurtig artikel, er dette kapitel sikkert interessant for dig.

5.1. OpenOffice.org1.1

Dette afsnit vil omhandle en af de nye spændende faciliteter i OpenOffice.org 1.1, som er muligheden for, at kunne fremstille dokumenter i OpenOffice.org og bagefter få dem gemt i DocBook format. At denne facilitet er interessant, har sin begrundelse i, at man bagefter kan konvertere sit dokumentet til et utal af forskellige formater. I øjeblikket er følgende formater understøttet:

- HTML
- PostScript (PS)
- Portable Document Format (PDF)
- Rich Text Format (RTF)
- DVI

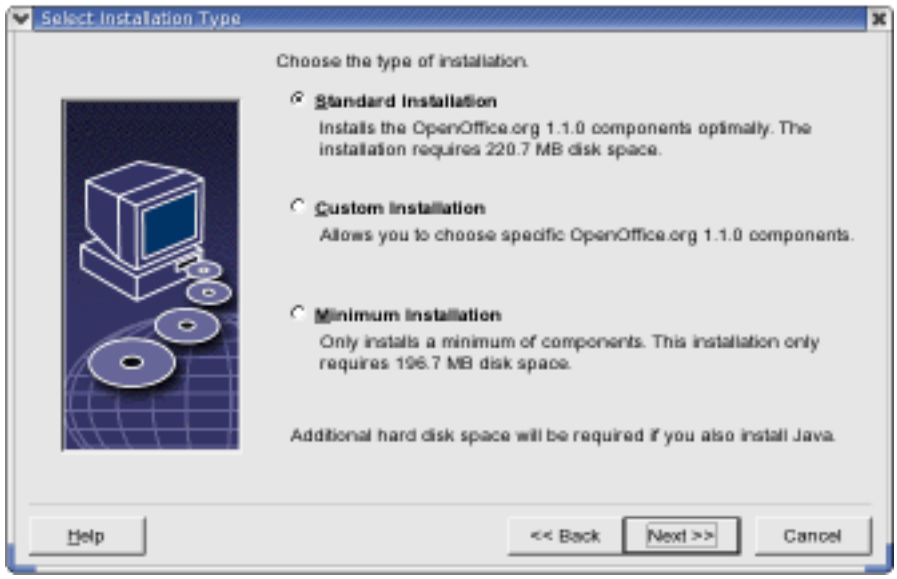
Efter gennemlæsning af dette afsnit vil du være i stand til, at fremstille dit første dokument i DocBook format.

5.1.1. Installation af openOffice 1.1

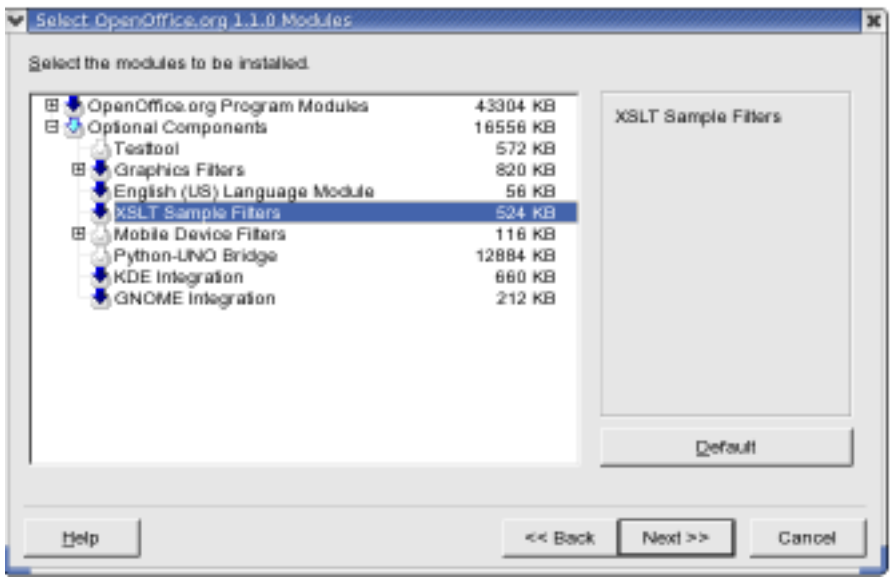
Før du kan anvende DocBook fra OpenOffice.org, skal det selvfølgeligt først installeres, og da formålet er at skrive DocBook-filer, kan du ikke anvende standardinstallationen; du må i stedet vælge interaktiv installation. Se nedenstående skærmdump viser, skal du køre install med kommandolinjetilvalget `-interactive`. Det gøres på følgende måde: **./install -interactive** .

```
Usage: install [options]
Options: [defaults in brackets after descriptions]
Configuration:
--help print help message (this message)
--version print the version of OpenOffice.org to be installed
--prefix=PREFIX install OpenOffice.org into PREFIX [/usr/local]
--single install single user version of OpenOffice.org
--interactive install OpenOffice.org using interactive mode
```

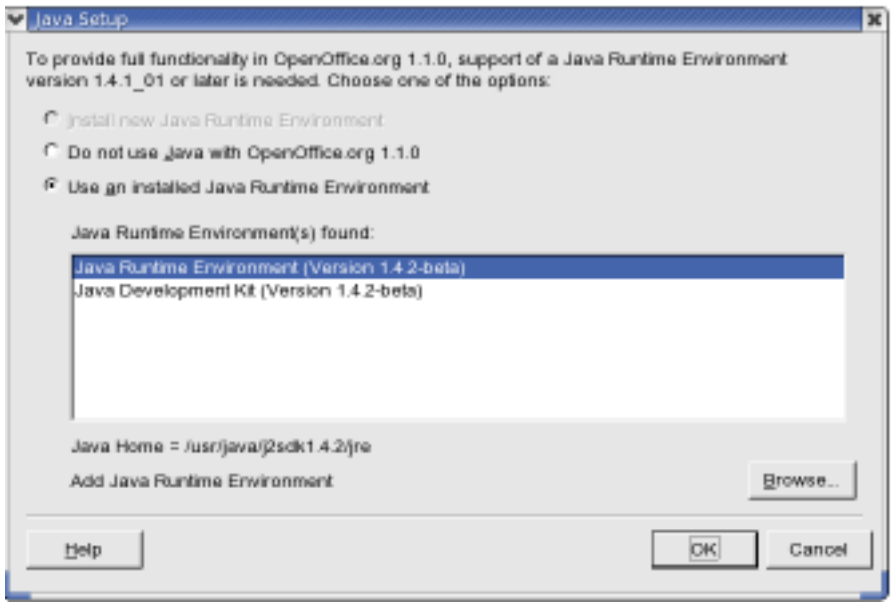
Når du når frem til nedenstående skærmbillede, skal du vælge custom install , da du her skal tilvælge filtre, der gør import/eksport af DocBook-filer muligt.



Herefter vil nedenstående skærmbillede fremkomme, hvor du skal klikke på symbolet ud for XSLT Sample Filters. Efter du har klikket, skulle du gerne se det markeret med en blå pil. Markeringen sikrer, at du installerer de nødvendige DocBook filtre.



Når du har foretaget ovenstående valg, vil du få at vide, at du har givet alle nødvendige informationer til, at OpenOffice.org kan installeres. Du skal så sikre dig, at det næste skærmbillede ser ud på følgende måde:

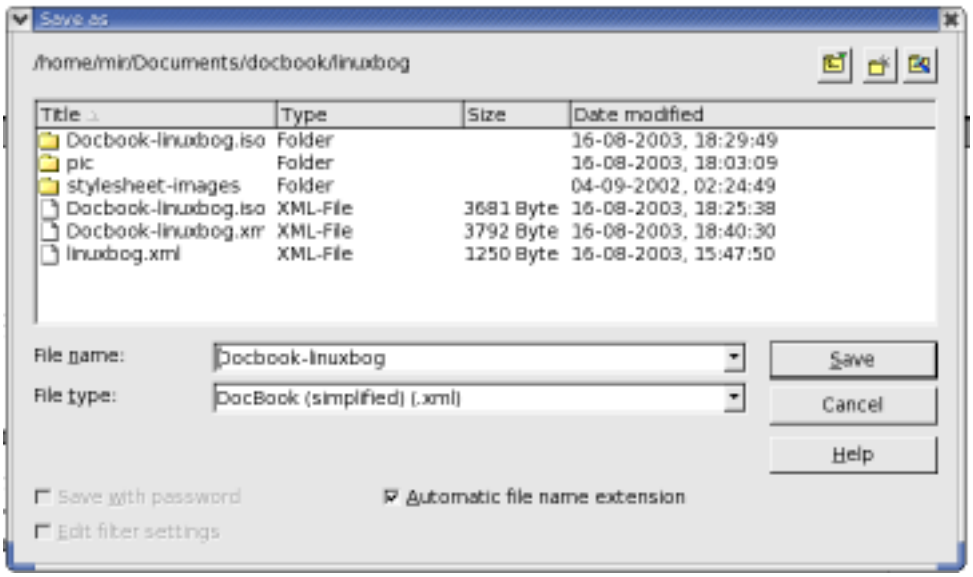


Det er af vital betydning, at du har installeret Java på din computer, da import/eksport af DocBook filer skal parses gennem et Java-program. Er der ikke installeret Java, bør du afslutte installationen og installere Java først - bemærk, at kun Sun's Java j2se, se http://java.sun.com/j2se/, kan anvendes. Har du derimod Java installeret, men installationsprogrammet blot ikke kan finde det, skal du vælge Browse. Du vil her have mulighed for at vise installationsprogrammet, hvor du har installeret Java. Hvis du viser den det rigtige sted, skal det fremgå af vinduet, hvor du kan se mine Java-installationer.

Når du har installeret OpenOffice.org, kan du begynde at lave DocBook filer.

5.1.2. Fremstilling af DocBook filer

Når du selv skal fremstille DocBook filer, er det såre enkelt: Åben et nyt dokument - File->New->Text Document, og gem det i DocBook format - File->Save As. Se nedenstående billede.



De faciliteter du har til rådighed, kan du se i under paragraph styles - fremkommer når du trykker <F11 > på tastaturet. Bemærk endvidere her, at du har 5 mulige styles, da iconerne øverst i vinduet underindeler custom styles i 5 kategorier: Paragraph styles, character styles, frame styles, page styles og numbering styles. Dem jeg har prøvet indtil videre, har virket efter hensigten, men en væsentlig undtagelse: Alle elementer under paragraph styles der starter med Section, virker ikke! For at kunne inddele dokumentet i afsnit, skal du i stedet for vælge de normale heading-elementer, og husk lige på i den forbindelse, at HTML-standarden kun understøtter heading1 (<h1></h1>) - heading6 (<h6></h6>). Indsættelse af grafik virker også; anvendes på normal vis, blot skal grafikken efterfølgende kopieres ned i samme katalog, hvor du placerer html-filerne.

Små fejl, jeg har fundet, udover den nævnte med sections er, at når man laver en tankestreg, udskifter OpenOffice.org den med en anden type tankestreg, der desværre ikke findes i Latin-1 tegnsættet, hvorfor oversættelse vil give fejl, du må derfor enten undlade at benytte tankestreg eller manuelt udskifte alle OpenOffice.org tankestreger med de generiske fra Latin-1. En anden væsentlig fejl er, at den attribut der bestemmer hvilket sprog dokumentet generes i, sættes forkert. Herudover er der problemet med, at OpenOffice.org arbejder i UTF-8 tegnsættet, hvorimod at DocBook, på trods af at der er specificeret UTF-8 som tegnsæt, oversætter filerne til Latin-1, hvilket betyder, at alle danske tegn forsvinder fra dokumentet. Det er sikkert blot et spørgsmål om at sætte et korrekt kommandolinjetilvalg til docbook2html, men det har jeg indtil videre ladet ligge. Alle disse fejl kan du selvfølgeligt rette manuelt, mens hvis du ikke har mod på det, kan du i stedet for, benytte dette lille perl-script til at få dine dokumenter konverteret fra UTF-8 til Latin-1. Hent perl-scriptet (http://www.linuxbog.dk/dokumentation/eksempler/oo2db) . Scriptet ser ud på følgende måde:

```
#!/usr/bin/env perl
use strict;
use warnings qw(syntax);
use File::Find;
my ($lang, $encoding, $name) = (undef,undef,"");
my $copyright = "This script is copyrigt 2003, by Michael Rasmussen, mir\@datanom.net. The
if (!@ARGV || $ARGV[0] eq "-u" || $ARGV[0] eq "--usage") {
```

```
 print $copyright;
 print "Usage:\n";
 if (substr($0,0,1) eq ".") {
  $name = substr($0,2);
 }
 else {
  $name = "$0";
 }
 print "\t",$name;
 print " [ [-h | --help] | [-u | --usage] ]\n";
 print "or\n";
 print "\t",$name," files...\n\n";
 print "\t-h [ --help ]\n\t\tShow extended help.\n";
 print "\t-u [ --usage ]\n\t\tShow briefe help (this text).\n";
 exit(0);
}
if ($ARGV[0] eq "-h" || $ARGV[0] eq "--help") {
 print $copyright;
 print "This script is intended to help people who wants to use OpenOffice.org\n";
 print "to write DocBook-XML source files. The purpose for this script is to\n";
 print "make it easy for people to correct the Openoffice files for known errors,\n";
 print "of which the most important ones is the wrong attribut value for the language\n";
 print "attribut <lang>, and that the encoding option for XML does not reflect the chosen\n
 print "language format for the output.\n";
 print "\nFor this transition to happen smoothly, you need to suply input to the script for
 print "the desired language and encoding style. The default is:\n";
 print "lang = da\n";
 print "encoding = iso-8859-1\n";
 print "\nIf you find any bugs or have any comments, you are most welcome to send me\n";
 print "an email. My email address is: mir\@datanom.net.\n";
 exit(0);
}
my @files = @ARGV;
my ($filename,@temp,$name,$ext);
my $first = 1;
while (<>) {
 if ($first) {
 $filename = shift @files;
 ($name,my $dummy,$ext) = File::Basename::fileparse($filename,qr{\.xml});
 if (!defined $lang) {
  print "Input desired language for DocBook output[da]: ";
  chomp ($lang = <STDIN>);
  if ($lang =~ /^\s*$/) {
   $lang = "da";
  }
 }
 if (!defined $encoding) {
  print "Input desired encoding style for DocBook output[iso-8859-1]: ";
  chomp ($encoding = <STDIN>);
  if ($encoding =~ /^\s*$/) {
   $encoding = "iso-8859-1";
  }
  $first = 0;
```

```
 }
 if (/^\<\?xml/) {
  $_ =~ s/encoding=['|"](.*)['|"]/encoding="$encoding"/g;
 }
 if (/(^<book|^<article)/) {
  $_ =~ s/lang=['|"](.*)['|"]/lang="$lang"/g;
 }
 push (@temp,$_);
 if (eof(ARGV)) {
  close ARGV;
  $filename = "temp".$$;
  open (OUTPUT,">$filename") or die "Error: $!";
  print OUTPUT @temp;
  close OUTPUT;
  system "iconv -f utf-8 -t $encoding -o $name.iso$ext $filename";
  $first = 1;
  @temp = ();
  system "rm -f $filename";
 }
}
exit(0);
```

Til sidst kan jeg i parantes bemærke, at hele dette dokument er skrevet udelukkende i OpenOffice.org med en enkelt udtagelse. OpenOffice.org har ikke implementeret elementet screen endnu, så derfor har jeg selv manuelt måtte indsætte det. Man kunne i stedet for have benyttet elementet ComputerOutput, men da vi som standard benytter screen i Friheden til at vælge serien, har jeg undladt dette.

Kapitel 6. Fremstilling af "Friheden til at vælge" bøgerne

Dette kapitel har oprindeligt været linuxbog-gruppens interne retningslinjer og hjælp for skrivning og fremstilling af bøgerne "Linux - Friheden til at vælge".

Målgruppen til dette kapitel er således dem der gerne vil skrive noget til FTAV, eller dem der gerne vil vide noget om hvordan vi fremstiller bøgerne.

I linuxbog-gruppen er forskellige personer med forskellige roller/arbejdsområder.

- *DocBook-skriverkarl* En person der skriver i bøgerne og selv skriver SGML-tags, således at bøgerne med det samme er klar til videre automatisk behandling. Personen uploader også selv nye tekster, så disse bliver automatisk vist på cvs.linuxbog.dk (http:/cvs.linuxbog.dk/)

  De fleste der er med i linuxbog-gruppen har denne rolle. Det svinger lidt hvor mange personer det drejer sig om, men et sted imellem 10 og 20 personer.

- *Korrekturlæser* En person der læser korrektur på bøgerne. Det er ikke noget der sådan foregår i faste intervaller, men mere sådan "on and off". Det har nogle gange foregået ved at alle bøgerne blev printet ud, og så satte personen røde mærker hvor det forkert. Efterfølgende har en *DocBook-skriverkarl* så rettet bøgerne igennem efter disse printudskrifter.

  Der er typisk kun én person der har denne rolle.

- *Book build programmer* En person der vedligeholder det automatisk byggesystem. Arbejdet indebære vedligeholdelse af **autoconf**, **automake**, `Makefile`, **perl**, **shell** og **php** -scripts.

  Der er omkring en 5-6 personer der har denne rolle.

- *Release manager* En person der klargøre bøgerne til at blive udgivet. En del af dette arbejde er at flytte filerne fra cvs.linuxbog.dk (http:/cvs.linuxbog.dk/) til www.linuxbog.dk (http:/www.linuxbog.dk/) . Personen skal også skrive en annoncering af de nye bøger (eller bede en anden om det).

  Der er typisk kun én person der har denne rolle.

6.1. Hvad er Linux -- Friheden til at vælge (FTAV)

"Linux -- Friheden til at vælge" er en række bøger om Linux, Unix og andre relaterede emner, som er skrevet af en gruppe frivillige, fortrinsvis medlemmer af Linux brugergruppen "Skåne-Sjælland Linux User Group" (SSLUG; http://www.sslug.dk/).

FTAV udgives med jævne mellemrum på http://www.linuxbog.dk/ . Her kan du downloade færdige udgaver af bøgerne, i flere forskellige formater. Nogen formater er velegnet til online brug, f.eks. HTML, andre egner sig bedre til tryk, f.eks. PDF, ligesom der er formater til palmpilot, mv. Endeligt er kildeteksten til bøgerne, svarende til de færdige udgaver også tilgængeligt her.

Udviklingen af bøgerne -- rettelser, opdateringer, mv -- foregår på http://cvs.linuxbog.dk/ . Her finder du i princippet det samme som på http://www.linuxbog.dk/, men bøgerne indeholder de sidste rettelser (og fejl, unøjagtigheder, mv) som forfatterne har tilføjet. Her findes også information om hvordan du kan komme i kontakt med forfatterne/udviklerne på bøgerne.

6.2. Bygning af bøgerne fra kildekode

Der er relativt få grunde til at bygge bøgerne selv fra kildekode. Her opsummerer vi et par af de årsager der kan være.

- Du ønsker at skrive et afsnit til bøgerne, eller at rette noget der allerede er skrevet og vil derfor gerne kunne bygge dem selv.
- Du ønsker at bygge bøgerne på en anden måde (f.eks. anden sidebredde til fast tryk) end de bliver bygget på linuxbog.dk.

Hvis ikke du har et af ovennævnte behov, burde du kunne "nøjes" med de færdigbyggede bøger på cvs.linuxbog.dk, eller www.linuxbog.dk. Hvis du vil bygge bøgerne selv, bliver det i det følgende beskrevet hvordan dette gøres.

For at kunne bygge bøgerne, skal du have en del forskellige programmer installeret. Det drejer sig om værktøjer der kan oversætte sgml koden (som bøgerne er skrevet i) til de forskellige formater (html, pdf, ps, palm pilot, mv) som bøgerne kan oversættes til.

Fremgangsmåden for at bygge bøgerne er lidt forskellig, afhængigt af hvordan du har fået fat i kildeteksterne. Hvis du har fået CVS adgang til kildekoden, skal du starte med at læse afsnittet nedenfor kaldet "Adgang til kildekoden fra CVS". I det følgende beskrives hvordan en enkelt bog, downloadet som en tar.gz fil, oversættes.

Først skal du udpakke bogen, f.eks. med kommandoen

```
[tyge@hven ~]$ $tar zxvf linuxbog-applikationer-dist-*.tar.gz
```

Derefter skal du skifte til kataloget er der blevet oprettet, og konfigurere bogen:

```
[tyge@hven ~]$ cd applikationer
[tyge@hven ~]$ ./configure --help
```

Ved at skrive --help vil du få en oversigt over de argumenter configure kan tage. Som udgangspunkt burde det ikke være nødvendigt at anvende nogen argumenter, men der kan være situationer hvor det er ønskværdigt.

Du kan nu konfigurere bygning af bogen, med f.eks.

```
[tyge@hven ~]$ ./configure --enable-softlink
```

Herefter vil configure undersøge dit system for at finde ud af om du har de nødvendige værktøjer til at bygge bogen med. Hvis configure finder ud af at du mangler centrale værktøjer, vil den afbryde med en fejl. Det kan f.eks. være at du mangler programmet "jade" som kan læse sgml filer, eller programmet "jw" som er en frontend til jade. Disse programmer er altid krævet for overhovedet at kunne oversætte bøgerne. For de forskellige formater bøgerne skal oversættes til, kræves desuden nogle forskellige værktøjer. F.eks. kræves programmet "db2html" for at kunne lave en udgave af bøgerne i html format. Configure programmet vil ikke afbryde hvis disse værktøjer mangler, men blot konfigurere oversættelsen således at disse ikke kan bygges. Til slut vil configure udskrive en oversigt over hvilke formater der kan oversættes til. Det kan f.eks. se sådan her ud:

```
configure: Oversigt over hvilke moduler der kan laves
configure:  Kan SGML bygges    : ja
configure:  Kan PALM bygges    : nej
configure:  Kan PDF bygges     : ja
configure:  Kan HTML bygges    : ja
configure:  Kan PAKHTML bygges : ja
```

Hvis det format du gerne vil bygge til, ikke understøttes, kan du kigge i det configure har skrevet for at finde årsagen. Eksempelvis finder jeg følgende linie:

Hvis du mangler nogen værktøjer må du installere dem, og køre configure igen. Hvis du har vanskeligt ved at finde ud af hvorfor du ikke kan bygge et bestemt format, kan du kigge i filen "config.log" hvor configure skriver detaljeret information om hvad det foretager sig.

Når du har fået support for de formater du gerne vil have, kan du skrive f.eks.

```
[tyge@hven ~]$ make html
```

for at lave html udgaven. Eller, make pdf for at lave pdf udgaven, osv. Bemærk at palm formatet hedder palmpilot.

6.3. Adgang til kildekoden fra CVS

For at få adgang til bøgerne via CVS, skal du vise at du vil være aktiv og kan arbejde ansvarligt med bøgerne :) Få login på tyge.sslug.dk ved at skrive til Peter Toft <pto@linuxbog.dk> og gør dernæst følgende.

Installer OpenSSH på din maskine og følg vejledningen på
http://cvs.linuxbog.dk/sikkerhed/bog/sikker-net-trafik.html
http://cvs.linuxbog.dk/sikkerhed/bog/opsaetning-af-openssh.html

Gør følgende (Fremhævede linier udføres kun første gang)

```
[tyge@hven ~]$  ssh DITLOGIN@tyge.sslug.dk
[tyge@hven ~]$  mkdir public_html
[tyge@hven ~]$  chmod a+rx ~ ~/public_html
[tyge@hven ~]$  cd public_html
[tyge@hven ~]$  cvs -d /usr/local/CVSROOT checkout linuxbog
[tyge@hven ~]$  cd linuxbog
[tyge@hven ~]$  ./configure      <<-- NB, se afsnit længere nede om byggesystemet.
[tyge@hven ~]$  cd BOGNAVN
[tyge@hven ~]$  make html
```

Så kan du køre Netscape http://cvs.sslug.dk/~DITLOGIN/linuxbog/java/bog

Og teksten kommer på web hver morgen http://cvs.linuxbog.dk

configure --help fortæller om nogle options, blandt andet --med-alle, som får make til at generere et arkiv og en samlet html-version af alle bøgerne.

Du kan også køre ./configure i de enkelt bog-directories. cd BOGNAVN && ./configure -- men det er kun nødvendigt, hvis du selv har ændret i bootstrap.subdir.

Hvis du gerne vil have filerne hjem, så kan du sætte følgende miljø-variabel (environment-variable)

```
[tyge@hven ~]$ export CVS_RSH=ssh
```

og skrive

```
[tyge@hven ~]$ cvs -d DITLOGIN@tyge.sslug.dk:/usr/local/CVSROOT checkout linuxbog
```

hvorefter du får bogen checket ud.

Hvis du kun vil have en enkelt bog checket ud, bruger du top-navnet: cvs -d DITLOGIN@tyge.sslug.dk:/usr/local/CVSROOT checkout linuxbog/friheden

Den samlede størrelse af CVS-checkout af alle bøger er ca. 30MB

De enkelte bøger varierer fra < 1.0 MB (dokumentation) til 8.8 MB (applikationer)

Læs http://cvs.linuxbog.dk/program/bog/vaerktoej.html#VAERKTOEJ-CVS

6.4. E-mail notifikation

Hvis du vil have en email hver gang der ryger en fil ind i CVS-arkivet (cvs commit), så kan I tilmelde jer <linuxbog-commit@tyge.sslug.dk> ved at skrive til <linuxbog-commit-subscribe@tyge.sslug.dk>

I som har konto på tyge kan bruge

```
[tyge@hven ~]$ cd katalog
[tyge@hven ~]$ cvs watch add
```

til at følge de filer I er interesserede i.

Hvordan ser de mails så ud man kan modtage på den nye liste? De vil indeholde info om hvem som lavede ændringen (pto), hvilket projekt der er ændret (linuxbog) og hvilke filer som er ændret (her kun hjaelpe.html) og man få den log-besked vi skriver hver gang (kan ses i http://cvs.linuxbog.dk/cvs2html/ -> http://cvs.linuxbog.dk/cvs2html/cvs_crono.html) Man får således ikke selve ændringen at se kun log-beskeden og hvad der er ændret - og af hvem.

6.5. Vejledning i at skrive

Læs DocBook

Hver bog er organiseret gennem bog.sgml og indhold.sgml pga. integration med "alle-upgaven" (alle bøgerne samlet) så er det smart at jeg i bog.sgml anfører filernes kobling til et SGML-tag - eksempler

```
<!entity java-indledning SYSTEM "indledning.sgml">
<!entity java-indhold SYSTEM "indhold.sgml">
<!entity java-appendixRevHist SYSTEM "apprevhist.sgml">
<!entity java-forord SYSTEM "forord.sgml">
```

og at jeg sidst i filen bog.sgml siger at NU kommer java-indhold - i praksis skriver du &java-indhold; og dermed kopieres filen indhold.sgml ind (jfr. ovenstående).

Hvis du så kigger i indhold.sgml så vil jeg i bunden bruge din indhold.sgml ved at skrive

```
&java-forord;      <---- defineret i bog.sgml
&java-indledning;  <---- do.
&java-appendixRevHist; <---- do. versionshistorien.
```

Dermed kommer de tre filer ind i følgende rækkefølge

1. `forord.sgml`
2. `indledning.sgml`
3. `apprevhist.sgml`

Når I har indført en ny fil VM.sgml og den skal med i f.eks. java bogen, så skal I indføje en linie i bog.sgml

```
<!entity java-VM SYSTEM "VM.sgml">
```

og i indhold.sgml skal I på passende sted referere

```
&java-VM;
```

Dernæst skal I tilføje en linie under den rette bog i linuxbog/alle/bog.sgml - samme som i bog.sgml for den enkelte bog - men med ../../../ sat foran stien. På den måde vil alle-bogen også være opdateret.

```
<!entity java-VM SYSTEM "../../../VM.sgml">
```

6.5.1. Tag ID

XML versionen kræver, at alle tags skrives med små bogstaver og alle tags har formen `<tag [attributer]>tagged text</tag>`

Afsnit tags, fx. `<sect1 id="xx-beskrivende-navn">` bliver brugt til at give html-filerne navne, så det beskrivende navn vælges med omhu. Det er vistnok en god ide at bruge en bogstavkode for selve bogens navn, altså: erstat xx med den identifikation, som benyttes for den bog, du skriver på.

6.5.2. Notation for skrivning

For at bøgerne ligner hinanden mest muligt i de viste eksempler, skal der anvendes samme brugernavn og maskinnavn i alle bøger.

Den primære bruger er 'tyge' og hans fulde navn er 'Tyge Brahe'. Denne bruger symbolisere brugeren selv. Skal der illustreres andre brugere der kommunikeres med, eller der skal oprettes på et system, bruges:

```
otto 'Otto Brahe'
axel 'Axel Brahe'
```

Valget af 'tyge', 'otto' og 'axel' udemærker sig ved at de alle har 4 tegn i navnet, og giver overskuelig liste når de bliver brugt i eksempler.

Den primære maskine som brugeren fysisk sidder ved hedder 'hven.sslug.dk', hvor normalt brug så blot er 'hven'. Skal der kommunikeres med andre maskiner, hedder de:

```
saltholm.sslug.dk
peberholm.sslug.dk
```

Tabel 6-1. E-mail eksempler

Navn	E-mail
Tyge Brahe	<tyge@sslug.dk>
Tyge Brahe	<tyge@hven.sslug.dk>
Otto Brahe	<otto@saltholm.sslug.dk>
Axel Brahe	<axel@peberholm.sslug.dk>

En normal-prompt (bash) i homedir bliver så:

```
[tyge@hven ~]$
```

Det er ikke på forhånd givet at 'otto' og 'axel' har login på 'hven.sslug.dk', det skal fremgå af eksemplet.

Baggrund: Tyge Brahe[1] (latinsk: Tycho) er en kendt dansk astronom der blev født på Sjælland og voksede op i Skåne. Sidst bosat på Hven. Otto er hans far og Axel er hans bror.

6.6. Mere om bygning af bøgerne fra CVS

Dette afsnit er til dig der har fået CVS adgang, eller dig der har haft det igennem et stykke tid, men gerne vil vide hvordan byggesystemet fungerer (eller forsøger at fungere).

Først beskrives hensigten med det nuværende system, derefter beskrives hvordan det virker, og hvilke muligheder man har med det.

Når man checker bøgerne ud fra CVS, vil jeg referere til det katalog der hedder "linuxbog" som "toplevel" og de enkelte bøgers kataloger (f.eks. applikationer, sikkerhed, mv) som subdir.

- Hensigten med byggesystemet.

  Hensigten med det byggesystem vi bruger i CVS og i de enkelte subdirs, er at

  1. Det skal være nemt for folk at bidrage til de enkelte bøger, _uden_ at have skriveadgang til CVS.
  2. Alle bøger skal kunne bygges til html, palmpilot, mv, afhængigheder skal fungere korrekt
  3. Alle bogen skal kunne bygges
  4. Der skal være install, uninstall, mv, targets
  5. På sigt skal man kunne bygge bøgerne under andre platforme end Linux (f.eks. FreeBSD)

  Motivation og uddybning af disse punkter:

  a. Tidligere har det været vanskeligt at bidrage til bøgerne af to hovedårsager: Den eneste måde at få adgang til _alle_ de nødvendige filer, var ved at få adgang til CVS udgaven af bøgerne & tests for de nødvendige værktøjer var uigennemskuelig og flettet ind i makefilerne.

     At få adgang via CVS var problematisk af to årsager. For det første skulle der skriveadgang til, samt login på CVS serveren. For det andet er det ikke alle der er fortrolige med CVS. De tests der har været for de forskellige værktøjer var tidligere dels inkomplette, dels lagt ind i selve makefilerne. Det betød at det var vanskeligt for brugeren at finde ud af hvorfor oversættelsen gav problemer. I praksis var det en langvarig "trial-and-error" proces, indtil man opnåede det ønskede resultat.

     Med autotools udgaven af bøgerne (hvor autotools bruges til at lave de nødvendige makefiler, mv) forsøges disse problemer løst. Problemet med tests er løst ved at lade autotools (autoconf) håndtere nødvendige tests. Det har den fordel at brugeren kan køre ./configure og få en oversigt over hvilke værktøjer der mangler. Når disse er installeret burde brugeren have en høj grad af sikkerhed for at testene virke. Test gennemføres kun i subdirs, da det er disse der skal bygges (som de enkelte bøger). Problemet med distribution af kildekoden er klaret ved at lave et nyt toplevel target "dist", der laver distributioner af de enkelte bøger. De pakker man får ud af dette, er i princippet "stand-alone" og har ikke nogen afhængigheder til resten af kildekoden.

  b. Ikke noget overraskende i dette. Imidlertid vil det nye system (når det er færdigt) forhåbentligt have mere orthogonale afhængigheder, og de problemer der pt. er med at f.eks. html og pdf rører ved "hinandens" filer, vil være fjernet. Der mangler en del her.

c. c) Det nye i forhold til det eksisterende system vil være at man kan lave en "personlig" alleudgave af bøgerne, der indeholder præcist de bøger man er interesseret i. Dette fungerer stort set - dog er der problemer med ps udgaven i øjeblikket.

d. d) På sigt skal der implementeres install og uninstall targets, så en systemadministrator f.eks. kan vælge at installere bøgerne for en større gruppe af brugere på en struktureret måde.

e. e) Med udgangspunkt i autotools, skal der findes alternativer til de platforme der ikke har f.eks. jw i en tilstrækkelig ny udgave. (pto?)

6.6.1. Sådan virker byggesystemet

Byggesystemet er (pt) en blanding af autotools til subdirs, og håndskrevne scripts til toplevel.

Den ultrakorte udgave af "hvordan man gør", efter udcheck fra CVS for at bygge bogen "BOGNAVN" som html er som følger:

```
[tyge@hven ~]$ ./configure <options>
[tyge@hven BOGNAVN]$ cd BOGNAVN
[tyge@hven BOGNAVN]$ make html
```

Men, det er langt fra hele historien.

Ideen med configure scriptet i toplevel er at det skal anvendes til at opsætte hvilke (og eventuelt hvordan) bøger man ønsker bygget på sin maskine. Dette bestemmes ved at give en række argumenter til configure. Hvis du kører ./configure --help, vil du se følgende:

6.6.2. ./configure

‘configure’ opsætter hvilke af "Linux - friheden til at vælge bøgerne" der skal laves på dit system.

Brug: ./configure [OPTIONS] [-- SUBDIROPTIONS]

Hvor OPTIONS kan være een eller flere af

```
  -h, --help              Viser denne hjælp og afslutter
  -m, --med "BØGER"       Bestemmer hvilke bøger der medtages, hvor BØGER
                          er en liste af bøger. Kombiner:
                          "itplatform friheden unix kontorbruger applikationer wm signatur a
  -a, --med-alle          Medtag alle (een stor bog) bogen
  -u, --bogurl URL        Hvilken url bøgerne skal bruge. [cvs.linuxbog.dk]
```

```
Hvor SUBDIROPTIONS kan være
      --enable-softlink  Bruger softlinks for HTML targets
Eksempel: ./configure --med "friheden applikationer" -- --enable-softlink
```

I det følgende vil disse argumenter blive uddybet.

- --med "BØGER" argumentet, bruges til at fortælle _hvilke_ bøger du gerne vil have bygget på dit system. ./configure --help vil udskrive en liste af dem der mulige (se ovenfor). Så, hvis du f.eks. gerne vil have bygget de bøger der anbefales når du vil lære hvordan man bruger en Linux maskine til at programmere med, kan du give configure argumentet --med "friheden unix program c java". Så vil configure sørge for at opsætte systemet, således at disse bøger vil blive oversat. Hvis du ikke angiver noget --med argument, vil alle bøger blive medtaget.
- --med-alle argumentet bruges til at angive at man vil have bygget den særlige "alle" bog. Det er en udgave hvor de forskellige bøger flettes ind i hinanden. --med argumentet bruges af --med-alle; hvis du har angivet et --med argument, bruges kun de bøger i dette argument til --med-alle. Rækkefølgen er også som i --med, hvis det er angivet.
- --bogurl bruges fortrinsvist når vi udgiver en bog på nettet - så bygges bøgerne så interne (explicitte) referencer til &linuxbogurl; refererer til www.linuxbog.dk - normalt er det cvs.linuxbog.dk.
- SUBDIROPTIONS er argumenter der ikke bruges af configure programmet selv, men gives videre til configure i de forskellige subdirs. Se afsnittet om at bygge fra kildekode tidligere i denne fil.

Her er en liste af filer der anvendes direkte af byggeprocessen i toplevel.

Tabel 6-2. configure filer

Filnavn	Beskrivelse
`configure`	dette script kopierer en række filer til de forskellige subdirs, og kører "bootstrap", samt "configure" i disse subdirs.
`Makefile.in`	denne fil laves af configure om til Makefile i toplevel.
`faelles-filer/*`	disse filer kopieres af configure til de forskellige subdirs, som subdir/*
`bootstrap.subdir`	denne fil kopieres til de forskellige subdirs, som subdir/bootstrap
`Makefile.subdir`	denne fil kopieres af configure til de forskellige subdirs som Makefile.am, som af subdir/bootstrap laves om til Makefile
`configure.ac.subdir`	kopieres af configure til de forskellige subdirs som subdir/configure.ac

Endvidere kopieres scriptet "misc/insertimagesize" til subdir/misc/, samt de forskellige palm relaterede filer fra misc til subdir/palm-faelles/

Tabel 6-3. configure filer

Filnavn	Beskrivelse
`Makefile.alle`	Denne fil kopieres af configure til alle/Makefile.am, hvis --med-alle argumentet blev angivet.

Når configure er færdig med at kopiere, kører configure subdir/bootstrap og subdir/configure SUBDIROPTIONS for alle de bøger der skal opsættes. Bootstrap kører de forskellige autotools værktøjer der skal til for at opsætte de forskellige subdirs. Bemærk at det således er nødvendigt at have autotools værktøjerne for at bygge bøgerne fra CVS, men det er _ikke_ nødvendigt for at bygge individuelle bøger hentet i *dist* pakker.

Den centrale del af det videre byggeforløb ligger i filerne "Makefile.subdir" (der via toplevel/configure og subdir/configure bliver til subdir/Makefile). Denne fil er den centrale at modificere for at rette de tilbageværende problemer i byggeprocessen.

6.7. Specialudviklede scripts

Til brug for den automatiske bygning af bøgerne, har det været nødvendigt at udvikle specielle scripts. Nogle af scriptne er som sådan ikke nødvendige for at bygge bøgerne, men de hjælper læseren til bedre at læse bogen.

6.7.1. splitstikord.pl

Scriptet 'splitstikord.pl' der sørger for at blande alle stikord.html filerne sammen fra alle bøgerne, og dernæst lave en fil for hvert bogstav, kræver at perl-pakken HTML::TreeBuilder er installeret. Det er ikke alle der har den default.

På tyge.sslug.dk der kører Red Hat 9.0 (for tiden), er pakken http://dag.wieers.com/packages/perl-HTML-Tree/perl-HTML-Tree-3.17-0.dag.rh90.noarch.rpm blevet installeret. Original-pakken findes på http://search.cpan.org/dist/HTML-Tree/

6.7.2. usedtags.pl

I bogen om dokumentation er liste over hvilke DocBook-tags der er mest anvendt i bogen om dokumentation. Se eksempel på resultatet. Dertil er et lille script der skriver dem en gang ud på skærmen, med angivelse for hver gang de er brugt.

Scriptet kan også bruges på html-filer, og er dermed praktisk anvendeligt i andre sammenhænge, hvorfor det er listet her under:

```
#!/usr/bin/perl
# Count used SGML tags in files.
# by Hans Schou and Ole Tange
#
# Usage:
#   cat *.sgml | ./usedtags.pl | sort -nr > usedtags_now.sgml

while (<>) {
  if (/<([a-zA-Z][^>\s]+)[^>]*>/) {
    $s = lc $1;
    $all{$s}++;
  }
}

@keys = sort { $all{$a} <=> $all{$b} } keys %all;

@keys = keys %all;
for (@keys) {
 if ($all{$_} > 0) {
         print "$all{$_} $_\n";
 }
}
```

Slutbemærkning:

1. Valget af Tyge Brahe som eksempel betyder ikke at alle medlemmer af LinuxBog-redaktionen anerkender det Tychoniske-verdensbillede (jorden i midten). Jvf. afsnittet om tidszoner, tror heller ikke alle at jorden er flad, det ville dog have gjort det hele lidt nemmere.

Appendiks A. Elementer anvendt i denne bog

For at skrive denne bog har vi anvendt følgende elementer. Der er til venstre angivet hvor mange gang det enkelte element blev anvendt. DocBook har mange elementer, men du kan jo starte med at lære de nævnte elementer i denne rækkefølge.

Tabel A-1. Anvendte elementer i denne bog

Antal	Navn	DocBook.org	O'reilly
497	para	docbook.org para (http://www.docbook.org/tdg/en/html/para.html)	O'reilly para (http://www.oreilly.com/catalog/docbook/c
158	command	docbook.org command (http://www.docbook.org/tdg/en/html/command.html)	O'reilly command (http://www.oreilly.com/catalog/docbook/c
152	ulink	docbook.org ulink (http://www.docbook.org/tdg/en/html/ulink.html)	O'reilly ulink (http://www.oreilly.com/catalog/docbook/c
137	title	docbook.org title (http://www.docbook.org/tdg/en/html/title.html)	O'reilly title (http://www.oreilly.com/catalog/docbook/c
123	listitem	docbook.org listitem (http://www.docbook.org/tdg/en/html/listitem.html)	O'reilly listitem (http://www.oreilly.com/catalog/docbook/c
118	indexterm	docbook.org indexterm (http://www.docbook.org/tdg/en/html/indexterm.html)	O'reilly indexterm (http://www.oreilly.com/catalog/docbook/c
109	programlisting	docbook.org programlisting (http://www.docbook.org/tdg/en/html/programlisting.html)	O'reilly programlisting (http://www.oreilly.com/catalog/docbook/c
70	primaryie	docbook.org primaryie (http://www.docbook.org/tdg/en/html/primaryie.html)	O'reilly primaryie (http://www.oreilly.com/catalog/docbook/c
70	indexentry	docbook.org indexentry (http://www.docbook.org/tdg/en/html/indexentry.html)	O'reilly indexentry (http://www.oreilly.com/catalog/docbook/c
61	secondaryie	docbook.org secondaryie (http://www.docbook.org/tdg/en/html/secondaryie.html)	O'reilly secondaryie (http://www.oreilly.com/catalog/docbook/c
54	emphasis	docbook.org emphasis (http://www.docbook.org/tdg/en/html/emphasis.html)	O'reilly emphasis (http://www.oreilly.com/catalog/docbook/c

Antal	Navn	DocBook.org	O'reilly
52	entry	docbook.org entry (http://www.docbook.org/tdg/en/html/entry.html)	O'reilly entry (http://www.oreilly.com/catalog/docbook/c
46	prompt	docbook.org prompt (http://www.docbook.org/tdg/en/html/prompt.html)	O'reilly prompt (http://www.oreilly.com/catalog/docbook/c
43	filename	docbook.org filename (http://www.docbook.org/tdg/en/html/filename.html)	O'reilly filename (http://www.oreilly.com/catalog/docbook/c
39	primary	docbook.org primary (http://www.docbook.org/tdg/en/html/primary.html)	O'reilly primary (http://www.oreilly.com/catalog/docbook/c
33	sect2	docbook.org sect2 (http://www.docbook.org/tdg/en/html/sect2.html)	O'reilly sect2 (http://www.oreilly.com/catalog/docbook/c
32	screen	docbook.org screen (http://www.docbook.org/tdg/en/html/screen.html)	O'reilly screen (http://www.oreilly.com/catalog/docbook/c
30	sect1	docbook.org sect1 (http://www.docbook.org/tdg/en/html/sect1.html)	O'reilly sect1 (http://www.oreilly.com/catalog/docbook/c
26	sect3	docbook.org sect3 (http://www.docbook.org/tdg/en/html/sect3.html)	O'reilly sect3 (http://www.oreilly.com/catalog/docbook/c
26	secondary	docbook.org secondary (http://www.docbook.org/tdg/en/html/secondary.html)	O'reilly secondary (http://www.oreilly.com/catalog/docbook/c
24	row	docbook.org row (http://www.docbook.org/tdg/en/html/row.html)	O'reilly row (http://www.oreilly.com/catalog/docbook/c
18	itemizedlist	docbook.org itemizedlist (http://www.docbook.org/tdg/en/html/itemizedlist.html)	O'reilly itemizedlist (http://www.oreilly.com/catalog/docbook/c
17	example	docbook.org example (http://www.docbook.org/tdg/en/html/example.html)	O'reilly example (http://www.oreilly.com/catalog/docbook/c
16	indexdiv	docbook.org indexdiv (http://www.docbook.org/tdg/en/html/indexdiv.html)	O'reilly indexdiv (http://www.oreilly.com/catalog/docbook/c
13	graphic	docbook.org graphic (http://www.docbook.org/tdg/en/html/graphic.html)	O'reilly graphic (http://www.oreilly.com/catalog/docbook/c

Antal	Navn	DocBook.org	O'reilly
13	figure	docbook.org figure (http://www.docbook.org/tdg/en/html/figure.html)	O'reilly figure (http://www.oreilly.com/catalog/docbook/c
12	xref	docbook.org xref (http://www.docbook.org/tdg/en/html/xref.html)	O'reilly xref (http://www.oreilly.com/catalog/docbook/c
8	literal	docbook.org literal (http://www.docbook.org/tdg/en/html/literal.html)	O'reilly literal (http://www.oreilly.com/catalog/docbook/c
6	orderedlist	docbook.org orderedlist (http://www.docbook.org/tdg/en/html/orderedlist.html)	O'reilly orderedlist (http://www.oreilly.com/catalog/docbook/c
5	thead	docbook.org thead (http://www.docbook.org/tdg/en/html/thead.html)	O'reilly thead (http://www.oreilly.com/catalog/docbook/c
5	tgroup	docbook.org tgroup (http://www.docbook.org/tdg/en/html/tgroup.html)	O'reilly tgroup (http://www.oreilly.com/catalog/docbook/c
5	tbody	docbook.org tbody (http://www.docbook.org/tdg/en/html/tbody.html)	O'reilly tbody (http://www.oreilly.com/catalog/docbook/c
5	table	docbook.org table (http://www.docbook.org/tdg/en/html/table.html)	O'reilly table (http://www.oreilly.com/catalog/docbook/c
4	surname	docbook.org surname (http://www.docbook.org/tdg/en/html/surname.html)	O'reilly surname (http://www.oreilly.com/catalog/docbook/c
4	sect4	docbook.org sect4 (http://www.docbook.org/tdg/en/html/sect4.html)	O'reilly sect4 (http://www.oreilly.com/catalog/docbook/c
4	footnote	docbook.org footnote (http://www.docbook.org/tdg/en/html/footnote.html)	O'reilly footnote (http://www.oreilly.com/catalog/docbook/c
4	firstname	docbook.org firstname (http://www.docbook.org/tdg/en/html/firstname.html)	O'reilly firstname (http://www.oreilly.com/catalog/docbook/c
4	chapter	docbook.org chapter (http://www.docbook.org/tdg/en/html/chapter.html)	O'reilly chapter (http://www.oreilly.com/catalog/docbook/c
4	author	docbook.org author (http://www.docbook.org/tdg/en/html/author.html)	O'reilly author (http://www.oreilly.com/catalog/docbook/c

Antal	Navn	DocBook.org	O'reilly
3	email	docbook.org email (http://www.docbook.org/tdg/en/html/email.html)	O'reilly email (http://www.oreilly.com/catalog/docbook/c
2	sgmltag	docbook.org sgmltag (http://www.docbook.org/tdg/en/html/sgmltag.html)	O'reilly sgmltag (http://www.oreilly.com/catalog/docbook/c
2	link	docbook.org link (http://www.docbook.org/tdg/en/html/link.html)	O'reilly link (http://www.oreilly.com/catalog/docbook/c
2	appendix	docbook.org appendix (http://www.docbook.org/tdg/en/html/appendix.html)	O'reilly appendix (http://www.oreilly.com/catalog/docbook/c
1	year	docbook.org year (http://www.docbook.org/tdg/en/html/year.html)	O'reilly year (http://www.oreilly.com/catalog/docbook/c
1	toc	docbook.org toc (http://www.docbook.org/tdg/en/html/toc.html)	O'reilly toc (http://www.oreilly.com/catalog/docbook/c
1	subtitle	docbook.org subtitle (http://www.docbook.org/tdg/en/html/subtitle.html)	O'reilly subtitle (http://www.oreilly.com/catalog/docbook/c
1	sidebar	docbook.org sidebar (http://www.docbook.org/tdg/en/html/sidebar.html)	O'reilly sidebar (http://www.oreilly.com/catalog/docbook/c
1	preface	docbook.org preface (http://www.docbook.org/tdg/en/html/preface.html)	O'reilly preface (http://www.oreilly.com/catalog/docbook/c
1	index	docbook.org index (http://www.docbook.org/tdg/en/html/index.html)	O'reilly index (http://www.oreilly.com/catalog/docbook/c
1	holder	docbook.org holder (http://www.docbook.org/tdg/en/html/holder.html)	O'reilly holder (http://www.oreilly.com/catalog/docbook/c
1	copyright	docbook.org copyright (http://www.docbook.org/tdg/en/html/copyright.html)	O'reilly copyright (http://www.oreilly.com/catalog/docbook/c
1	bookinfo	docbook.org bookinfo (http://www.docbook.org/tdg/en/html/bookinfo.html)	O'reilly bookinfo (http://www.oreilly.com/catalog/docbook/c
1	book	docbook.org book (http://www.docbook.org/tdg/en/html/book.html)	O'reilly book (http://www.oreilly.com/catalog/docbook/c

Antal	Navn	DocBook.org	O'reilly
1	authorgroup	docbook.org authorgroup (http://www.docbook.org/tdg/en/html/authorgroup.html)	O'reilly authorgroup (http://www.oreilly.com/catalog/docbook/c
1	abstract	docbook.org abstract (http://www.docbook.org/tdg/en/html/abstract.html)	O'reilly abstract (http://www.oreilly.com/catalog/docbook/c
1	abbrev	docbook.org abbrev (http://www.docbook.org/tdg/en/html/abbrev.html)	O'reilly abbrev (http://www.oreilly.com/catalog/docbook/c

Appendiks B. Versionshistorie for bogen

Igennem tiden har bogen udviklet sig meget. Vi frigiver ofte nye versioner, når der er kommer en del rettelser ind, eller nye afsnit er blevet skrevet. Kommentarer, ris og ros, og specielt fejl og mangler bedes sendt til linuxbog@sslug.dk (mailto:linuxbog@sslug.dk), men er du medlem af SSLUG så skriv til sslug-bog@sslug.dk (mailto:sslug-bog@sslug.dk). Her er en liste over, hvad der er ændret i bogen.

- Version 2.7.20040524 - 24. maj 2004: Hans Schou har flyttet afsnittet `/README` ind i denne bog som et selvstændigt kapitel. Se Kapitel 6.
- Version 2.7.20040524 - 24. maj 2004: Henrik Zederkof bidrager med oplysninger om nxml mode til Emacs, sekunderet af Donald Axel, som skriver om Gvim completion modes og syntax. Jesper Laisen: Beskrivelse af <img alt=""> til mindre betydningsfulde billeder (ikoner/hjørner m.v.) Jacob Sparre Andersen: Forskellige sproglige fejl rettet. Strammet forordet op. Rettet HTML-eksempler til.
- Version 2.7 - 25. januar 2004: Peter Toft: Slåfejl rettet.
- Version 2.6 - 7. oktober 2003: Mads Bondo Dydensborg har skrevet et afsnit om billedstørrelser, dpi og scale parameteren til Docbook. Michael Rasmussen påbegynder kapitel om WYSIWYG fremstilling af DocBook filer. I denne initiale udgave er tilføjet afsnittet med OpenOffice.org 1.1. Björn Lundin har skrevet et kapitel med et praktisk eksempel på brug af Docbook/XML. Donald Axel skriver afsnittet om DocBook om, så det er ajour med nyeste Linux-distributioner. Harry Jensen fanger en defekt URL. Jens Stavnstrup retter flere fejl op.
- Version 2.5 - 1. december 2002: Erik Martin fanger et par defekte URL'er.
- Version 2.4 - 1. september 2002: Mads Bondo Dydensborg har tilføjet et par sætninger om tankestreger.
- Version 2.4B - 4. juli 2002: Jesper Laisen tilføjer HTML-reference-afsnit - se Afsnit 1.3. Jacob Sparre Andersen: Skriver videre (og om) på HTML-kapitlet.
- Version 2.3 - 14. juni 2002: Tue Abrahamsen finder en defekt URL. Henrik Christian Grove tilføjede afsnit om at lave tabeller i LaTeX. Jesper Laisen tilføjer afsnit om tabeller i HTML. Anna Jonna Armannsdottir tilføjer mere om validering af HTML. Jacob Sparre Andersen: Rettet sproglige småfejl. Henrik Skov Midtiby rettede en stribe fejl. Jesper Laisen finder et par fejl.
- Version 2.2 - 10. marts 2002: Ny licens for bogen - Åben dokumentlicens. Peter Toft: Nye stikord, nye links - mere om opsætning. Simon-Shlomo Poil fandt en URL-fejl. Stig Jensen laver et SGML-eksempel og finder fejl i afsnittet om Red Hat 7.2 DocBook. Peter Toft reviderer DocBook teksten så fokus er på Red Hat 7.2. Peter Toft omskriver introduktion til DocBook efter input fra Erling Sjørlund. Henrik Christian Grove har omskrevet hele kapitlet om Latex. Peter Toft og Jesper Laisen har påbegyndt et kapitel om at skrive HTML.
- Version 2.1 - 29. december 2001: Donald Axel og Philip Heede retter LaTeX-afsnittet igennem. Donald Axel tilføjer en masse i afsnittet om DocBook - specielt omkring Red Hat 7.1 og 7.2. Mads Sejersen har rettet en stribe fejl i DocBook-afsnittet. Peter Toft rettet URL til "The Not So Short Introduction To LaTeXe".
- Version 2.0 - 21. oktober 2001: Michael Rasmussen fandt ud af at det er jade og ikke openjade man skal installere fra Red Hat 6.2. Peter Toft: Bogen har fået nyt navn, så det afspejles at det er dokumentation som er fokus - Nu er der kommet et Latex-kapitel ind. Det er skrevet af Kristian Sørensen. Link til engelsk DocBook installationsvejledning indsat.

- Version 1.9 - 11. august 2001: Søren Ulrik retter videre i installation.
- Version 1.8 - 9. juli 2001: Peter Toft: Tilføjet en henvisning til ny bog om Office. Rettet screen til programlisting. Gunner Poulsen. Rettelser til installationen. Søren Ulrik retter videre i installation - vigtige ændringer!!!
- Version 1.7 - 24. maj 2001: Søren Ulrik tilføjer en del om hvordan man retter DocBook-opsætningen til.
- Version 1.6 - 12. marts 2001: Henrik Christian Grove fandt en dum trykfejl i oversigt over bøger. Erik Sørensen fandt nogle trykfejl.
- Version 1.5 - 4. februar 2001: Peter Toft: Flyttet fra artikel til linuxbog og ændret en opsætningen noget og titel. Tilføjet mere om grafik-filer. Tilføjet mere om footnote og sidebar. Tommy Mogensen fandt en sprogbøf. Link til Carsten Svaneborgs HTML->SGML/DocBook oversætter.
- Version 1.4 - 11. august 2000: Peter Toft: Eksempler på at oversætte dokumenter tilføjet. Eksempel tekst med. Link til OPL og mere dokumentation.
- Version 1.3 - 10. august 2000: Peter Toft: Nyt om lister. En bunke referencer til Docbook-dokumenter tilføjet. Lidt om begrænsninger i ID-navngivning.
- Version 1.2 - 8. august 2000: Peter Toft: Rettede fejl i kommentar. Tilføjet link til hjælp på nettet.
- Version 1.1 - 6. august 2000: Peter Toft: Har skrevet lidt videre om entities og kommentarer. Har tilføjet links til artikel kildetekst.
- Version 1.0 - 5. august 2000: Første offentlige version.

Stikordsregister

Symboler

C

D

E

F

H

I

L

M

O

P

S

T

U

V

X