TDP001 - Handhavande av datormiljö

Pontus Haglund Emma Enocksson

Institutionen för datavetenskap

LINKÖPING UNIVERSITY

1 Latex grunderna
2 Latex features
3 Bashprogrammering

LINKÖPING UNIVERSITY

1 Latex grunderna
2 Latex features
3 Bashprogrammering

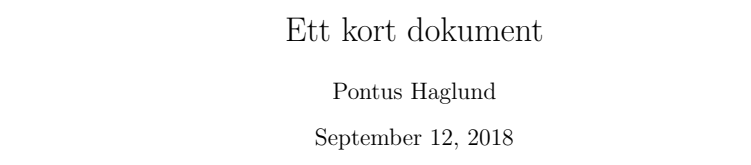
LINKÖPING UNIVERSITY

LaTeX

- LaTeX är ett ”document preperation system”
- Vanliga ordbehandlare (t.ex. MS Word och OpenOffice) fungerar enligt principen **W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et (WYSIWYG)
- LaTeX låter författaren fokusera på innehållet istället för utseendet - **W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **M**ean

LaTeX

- Antag att vi vill skapa ett dokument med följande innehåll:



- I en ordbehandlare krävs det ganska mycket jobb...

LaTeX

- I LaTeXskriver man ”enkel” kod i en texteditor:

```
\documentclass{article}

\author{Pontus Haglund}
\title{Ett kort dokument}

\begin{document}
\maketitle
Ett kort dokument utan vettig text
\end{document}
```

LaTeX

- Likt ett programmeringsspråk kompileras sedan din källkod till ett dokument
- Det finns stöd för flera olika filformat, men pdf är ju alltid trevligt
- På grund av referenser, inkluderingar och annat behöver man ofta kompilera i flera steg, därför finns det ett script `latexmk` som löser problemet åt en
  - Väljaren `-pdf` ger en pdf-fil som resultat
  - Väljaren `-pvc` uppdaterar resultatfilen när källkoden ändras

LaTeX

- I Ubuntu finns det flera paket för att installera LaTeX, men `texlive-full` installerar ”allt”:
  `sudo apt-get install texlive-full latexmk`

LaTeX

Syntax

- Ett dokument består av fyra grundläggande element:
  - Kommandon (command):
    `\namn[valfria parametrar]{parametrar}`
  - Block (group):
    `{ ... }`
  - Miljöer (environment):

```
\begin{miljönamn}
    ...
\end{miljönamn}
```

kommentarer

- Kommentarer:
  Endast enradskommentarer, inleds med %

1 Latex grunderna

2 Latex features

3 Bashprogrammering

LINKÖPING UNIVERSITY

LaTeX

En generell LaTeX-fil

```
\documentclass{...}
% Beskriver vilken typ av dokument vi skriver.
% Finns flera inbyggda som tex article, report och letter

% Inledning (preamble)

\begin{document}
% Här står allt som ska synas i dokumentet
\end{document}
```

LaTeX

Inledningen

- Området mellan `\documentclass` och `\begin{document}` kallas inledning (preamble)
- I inledningen kan man göra inställningar och ladda in extra paket för att lösa saker som språket inte har i grunden
- Ett paket laddas in med kommandot `usepackage`:
  `\usepackage[paketparametrar]{paketnamn}`

Inledningen

Några bra paket

- inputenc - används för att få ”specialtecken”(tex å,ä och ö) att fungera. Behövs endast när man använder pdflatex eller latex
  `\usepackage[utf8]{inputenc}`
- fontspec - Används istället för inputenc vid användning av xelatex
- babel - Språkinställningar såsom avstavning och formatering av tal och datum
  `\usepackage[swedish]{babel}`

Bra paket forts.

- graphicx - Stöd för att inkludera bilder och grafik
- listings - Inkludera och formatera källkod
- xcolor - Massor av färger

Att skriva LaTeX

En vanlig arbetsordning

1. Öppna ett tidigare dokument
2. Modifiera inledningen och ta bort resten
3. Spara som ett nytt dokument
4. Starta `latexmk` (med `-pvc`)
5. Låt texteditorn ta halva skärmen och din dokumentvisare andra halvan
6. Direkt du sparar uppdateras dokumentet i läsaren

Inbyggt stöd

Rubriker

- Rubriker skapas enkelt med kommandon:
  - Huvudrubriker med `\section{rubriknamn}`
  - Underrubrik med `\subsection{namn}`
  - Lägre nivå med `\subsubsection{namn}`
- Vid användning av dokumentklasserna report eller book finns även `chapter` i toppen av hierarkin

Rubriker forts.

- Alla rubriker blir automatiskt numrerade
  - Går att stänga av med *-varianten:
    `\section*{namn}`
- Innehållsförteckning skapas automatiskt med kommandot
  `tableofcontents`

Inbyggt stöd

Listor

- Precis som i HTML finns två typer av listor;
  - `itemize` - punktlista (ul)
  - `enumerate` - numrerad lista (ol)
- Exempel:

```
\begin{enumerate}
\item Punkt ett
  \begin{enumerate}
  \item hej
  \end{enumerate}
\item Punkt två
\end{enumerate}
```

1. Punkt ett
    1. hej
2. Punkt två

Inbyggt stöd

Tabeller

```
\begin{table}[!h]
 \begin{tabular}{|r|l|}
  \hline
  7C0         & hexadecimalt \\
  3700  & oktalt \\ \cline{2-2}
  11111000000 & binärt \\
  \hline \hline
  1984        & decimalt \\
  \hline
 \end{tabular}
 \caption{Olika talbaser}
 \label{tab:talbaser}
\end{table}
```

7C0	hexadecimalt
3700	oktalt
11111000000	binärt
1984	decimalt

Tabell: Olika talbaser

Inbyggt stöd

Referenser

- För att referera till andra delar i sitt dokument använder man sig av `\label` och `\ref`
  - `\label{namn}` används för att markera området (t.ex. en rubrik eller bildtext) man vill referera till
  - `\ref{namn}` används i löpande text för att sätta in en referens till markeringen `namn`
- För att referera till andra källor används `\cite`
  - Paketet `natbib` ger även tillgång till kommanona citet och citep som formaterar referenserna snyggare
  - Paketet `biblatex` är nyare och rekommenderas generellt

Inbyggt stöd

Referenser

```
\documentclass{article}
\usepackage[swedish]{babel}
\usepackage{fontspec}

\begin{document}
\section{Historia}
\label{sec:history}
Här berättar vi lite historia

\section{Sammanfattning}
I del \ref{sec:history} ...
\end{document}
```

1 Historia

Hr berttar vi lite historia

2 Sammanfattning

I del 1 ...

Referenser

BibTeX

- Om man har många externa källor kan man flytta ut dem på en separat fil
- Man strukturerar upp dem i en fil med filändelse `.bib`
- Där man sedan vill inkludera källorna kör man kommandot `\bibliography{filnamn}` (där filnamn är utan filändelse)

bibtex forts

- Exempel på BibTeX-källa:

```
@article{berland13,
author = {Berland, Matthew and Martin, Taylor and Benton, Tom and Petrick Smith, Carmen
      and Davis, Don},
title = {Using Learning Analytics to Understand the Learning Pathways of Novice
      Programmers},
journal = {Journal of the Learning Sciences},
volume = {22},
number = {4},
pages = {564—599},
year = {2013},
}
```

LaTeX- fördelar

- Det är ganska lätt att skriva ett snyggt dokument
- Källkoden kan läsas i alla texteditorer - en viss ordbehandlares text är ofta svår att läsa i en annan (eller en annan version)
- Du kan fokusera helt på texten
- Indexering, fotnötter och citeringar sköts automatiskt
- Eftersom allt är ren text går det att generera texten i vilket program (eller programspråk...) som helst!
- Kompilatorn tvingar dig att göra rätt - din text byggs upp på ett vettigt sätt
- Inbyggt stöd för att dela upp texten - lätt att samarbeta med andra
- Källkod går bra att versionshantera...

LaTeX- nackdelar

- Hög inlärningströskel
- Finns massor av kommandon och miljöer (såklart också en fördel)
- Kan vara svårt att få din text att se ut exakt som du vill
- Normalt sett ser man inte resultatet medan man skriver utan måste vänta på kompilering

Läs mer

- http://en.wikibooks.org/wiki/LaTeX/
  En väldigt bra wiki med både grunder och avancerade ämnen
- Comprehensive TeX Archive Network (CTAN) http://www.ctan.org
  Samling av latexmoduler

1 Latex grunderna
2 Latex features
3 Bashprogrammering

LINKÖPING UNIVERSITY

Ett enkelt ’skript’

```
#!/bin/bash
# Det här skriptet visar datum och tid,
# vem du är samt i vilken katalog du står.
echo "Datum och tid:"
date
echo "Du är inloggad som:"
echo "$USER"
echo "Du befinner dig i katalogen:"
pwd
```

Vad är ett skript

- En rad kommandon som tillsammans löser ett problem
- Mer avancerade skript nyttjar också kontrollstrukturer

```
#!/bin/sh
# Ta reda på om en användare varit inloggad eller inte.
resultat=`last | grep -s $1 | cut -f1 -d' ' | uniq`
if [ "$resultat" = "$1" ]
then
  echo $1 inloggad.
else
  echo $1 inte inloggad.
fi
```

Variabler och argument

- $N - argumenten som skickas till skriptet (0-9)
- $0 - namnet på skriptet
- $@ - alla variabler som skickades till scriptet
    - Sätt i princip alltid inom citattecken

if-satser

```
#!/bin/bash
if [ "$1" = "hej" ]; then
    echo expression evaluated as true
else
    echo expression evaluated as false
fi
```

while-loops

```
#!/bin/bash
COUNTER=0
while [  $COUNTER -lt 10 ]; do
    echo The counter is $COUNTER
    let COUNTER=COUNTER+1
done
```

```
#!/bin/bash
COUNTER=20
until [  $COUNTER -lt 10 ]; do
    echo COUNTER $COUNTER
    let COUNTER-=1
done
```

for-loop

```
#!/bin/bash
for i in 1, 2, 3; do
    echo item: $i
done
```

for-loop

- Dollar parentes låter oss köra kommandon och använda resultaten, se ls nedan

```
#!/bin/bash
for i in $( ls ); do
    echo item: $i
done
```

Funktion grund

```
#!/bin/bash
function quit {
    exit
}
function hello {
    echo Hello!
}
hello
quit
echo foo
```

Funktion parameter

```
#!/bin/bash
function quit {
   exit
}
function e {
    echo $1
}
e Hello
e World
quit
echo foo
```

Returvärden

- Finns inte, funktionen returnerar 0 eller annat
  - 0 - allt gick bra
  - något annat - något gick fel
- Man kan använda globala variabler

```
#!/bin/bash
function myfunc()
{
    myresult='some value'
}

myfunc
echo $myresult
```

Command substitution

- Det var inte bra, globala variabler är dåligt
- Men det finns command substitution i bash

```
#!/bin/bash
function myfunc()
{
    local  myresult='some value'
    echo "$myresult"
}

result=$(myfunc)
echo $result
```

- Finns fler sätt som du kan undersöka själv

Tillbaka till skriptet i början!

```
#!/bin/sh
resultat=`last | grep -s $1 | cut -f1 -d' ' | uniq`
if [ "$resultat" = "$1" ]
then
  echo $1 inloggad.
else
  echo $1 inte inloggad.
fi
```

- Kan vi förstå detta nu?

Välj rätt verktyg

- En viktig sak är att välja rätt verktyg för rätt jobb
- Behöver man skriva ett kort script som städar lite skräpfiler?
  - Bash är förmodligen helt rätt
- Behöver du skapa spelet sokoban med massor av logik och funktioner?
  - Python eller dylikt är förmodligen ett bättre val

www.liu.se

LINKÖPING UNIVERSITY