Dynamically set disk quota

Paul Huygen <paul.huygen@huygen.nl>

1st May 2017
10:16 h.

Abstract

This document generates a script (Bash) that adapts quota settings to the amount of free space on the disk.

Contents

1 Introduction

When a group of users have a computer in common use, the users tend to take up all the available disk-space, thereby making the system useless. It is an example of the “tragedy of the commons”[1].

To avoid this problem, a “quota” system can be set up that limits the the amount of disk-space that a user may occupy. This document describes and sets up a script that adapts the user-quota to the amount of disk-space that is still free. Initially, when there is sufficient disk-space, the quota is set to a large value (“begin-quota”). When the free disk-space decreases below a given thresholt, the quota-system sets in and reduces the user-quota slowly until there is enough free space. When a large fraction of the disk is free, the quota are gradually increased, until “begin-quota” is reached.

The script can be implemented as a “cron” task.

In our computer, we discern two user groups, regular users and students/guest, who may use the computer temporarily for specific projects. The students are allowed a smaller amount of quota than the regular users.

1.1 Quota settings

When there is plenty of free-disk-space, there is no need for a tight limit. Usually only few users need a really large amount of disk space, so let us set the absolute maximum quota to one fifth of the capacity of the disk. Let us assign a max amount of one tenth of that to students.

⟨ *quota settings* 2a ⟩ ≡

```
max_quota_perc=20
max_studquota_perc=2
◇
```

Fragment defined by 2abc, 3a.
Fragment referenced in 4c.
Defines: `max_quota_perc` 8g, `max_studquota_perc` Never used.

It is difficult to manipulate floating-point numbers in this script, therefore we will use percentages. Set the percentage of free space below which this script will reduce the quota and the percentage above which the script will possibly increase the quota:

⟨ *quota settings* 2b ⟩ ≡

```
minfreespace_perc=5
maxfreespace_perc=30
◇
```

Fragment defined by 2abc, 3a.
Fragment referenced in 4c.
Defines: `maxfreespace_perc` 5c, `minfreespace_perc` 5c.

When free space is too little, the script reduces the quota to `reduction_perc` percent of the original value. When free space is abundant, the script may expand the quota with `expansion_perc` percent.

⟨ *quota settings* 2c ⟩ ≡

```
reduction_perc=95
expansion_perc=105
◇
```

Fragment defined by 2abc, 3a.
Fragment referenced in 4c.
Defines: `expansion_perc` 8g, `reduction_perc` 8g.

The Unix quota system knows, besides the hard limit of allowed disc-space usage, a soft limit. If a user exceeds the soft limit, she will get warnings.

⟨ *quota settings* 3a ⟩ ≡
```
soft_perc=80
◇
```
Fragment defined by 2abc, 3a.
Fragment referenced in 4c.
Defines: `soft_perc` 8l.

1.2 Regular users and students

The users of the computer are divided up in a group "user" (with group-id 100) and a group "studs" with group-id 1035. The group-id is the fourth item in the "passwd file" (`/etc/passwd`).

⟨ *variables* 3b ⟩ ≡
```
usergroup_id=100
studgroup_id=1035
◇
```
Fragment referenced in 4c.
Defines: `studgroup_id` 10b, `usergroup_id` 8b, 10b.

To find a username and a group-id in a line of `/etc/passwd`, the following macro's can be used. The first argument (`@1`) represents the line from `/etc/passwd` and the second argument represents the username resp. group-id:

⟨ *find username in line of password-file* 3c ⟩ ≡
```
@2=‘echo @1 | gawk '{print $1}' FS=':'‘
◇
```
Fragment referenced in 8b, 10b.
Uses: `print` 17c.

⟨ *find group-id in line of password-file* 3d ⟩ ≡
```
@2=‘echo @1 | gawk '{print $4}' FS=':'‘
◇
```
Fragment referenced in 8b, 10b.
Uses: `print` 17c.

1.3 The quota system

The command `repquota` obtains information about the quota of users. The following shows an example of use:

```
huygen@kyoto:~/projecten/kyoto/quota/kyotoquota/nuweb$ sudo repquota /
[sudo] password for huygen:
*** Report for user quotas on device /dev/vda3
Block grace time: 00:00; Inode grace time: 00:00
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
[..]
brasser    --   554012 81895040  102368800          3241     0     0
vossen     -- 29239788 81895040  102368800        107543     0     0
segers     --   189764 81895040  102368800            38     0     0
```

So, the command results in a table with the name of the user in the first column, the “soft block-limit” in the fourth column and the “hard block limit” in the fifth column.

Hence, to obtain the current hard block-limit of a user, find the fifth column in the line that starts with the name of the user:

⟨ *get hard quotum of user* 4a ⟩ ≡
```
@2=`repquota -vu / | grep "@1" | gawk '{print $5}'`
◇
```
Fragment referenced in 6i.
Uses: `print` 17c.

Modify the quota of a user with the following macro. Arguments

1. (`@1`) Name of the user.
2. soft block-quotum
3. hard block-quotum

⟨ *set quota of a user* 4b ⟩ ≡
```
setquota -u @1 @2 @3 0 0 /
◇
```
Fragment referenced in 10b.
Defines: `setquota` Never used.

2 The script

The script works as follows:

- Find out the amount of free diskspace.
- Determine whether the quota must be reduced or increased.
- If so, perform the change.

When argument `-l` is provided, the script will print a summary of the numbers.

`"../adaptquota"` 4c≡
```
#!/bin/bash
# adaptquota -- adapt user-quota to amount of free disk space
# usage: adaptquota [-l]
#  -l: perform logging
⟨ set logging 14b ⟩
⟨ variables 3b ⟩
⟨ quota settings 2a, ... ⟩
⟨ find out free diskspace 5a, ... ⟩
⟨ determine whether quota should be reduced or possibly expanded 6g ⟩
⟨ expand or reduce quota 6h ⟩
◇
```
Uses: `logging` 14b.

2.1 Find out free disk-space

Use Unix command `df` to find out the capacity of the disk and the amount of disk-space that is still free. An example of the result of the `df` command:

```
huygen@kyoto:~$ df /dev/vda3
Filesystem     1K-blocks      Used Available Use% Mounted on
/dev/vda3      511844016 406344896  79475852  84% /
huygen@kyoto:~$
```

So, it seems that the second word of the second line of the output gives us the total disk capacity and the fourth word gives us the remaining capacity.

⟨*find out free diskspace* 5a⟩ ≡

```
    disk_capacity=`df /dev/vda3 2>/dev/null | gawk 'NR==2 {print $2}'`
    disk_free=`df /dev/vda3 2>/dev/null | gawk 'NR==2 {print $4}'`
    ◇
```

Fragment defined by 5abcd.
Fragment referenced in 4c.
Defines: `disk_capacity` 5b, 6b, `disk_free` 5b, 6g.
Uses: `print` 17c.

To perform integer arithmatic with the obtained data, let us create variables that represent one percent of the capacity. To do this, chop off the two rightmost digits:

⟨*find out free diskspace* 5b⟩ ≡

```
    disk_capacity_onep=${disk_capacity%??}
    disk_free_onep=${disk_free%??}
    ◇
```

Fragment defined by 5abcd.
Fragment referenced in 4c.
Defines: `disk_capacity_onep` 5c, 8g, `disk_free_onep` Never used.
Uses: `disk_capacity` 5a, `disk_free` 5a.

Define `min-diskfree`, the amount of free disk-space below which the quota will be restricted and `max-diskfree` above which the quota might be expanded.

⟨*find out free diskspace* 5c⟩ ≡

```
    min_diskfree=$((minfreespace_perc*$disk_capacity_onep))
    max_diskfree=$((maxfreespace_perc*$disk_capacity_onep))
    ◇
```

Fragment defined by 5abcd.
Fragment referenced in 4c.
Defines: `max_diskfree` 6fg, 8a, `min_diskfree` 6dg.
Uses: `disk_capacity_onep` 5b, `maxfreespace_perc` 2b, `minfreespace_perc` 2b.

If logging is on, print the numbers.

⟨*find out free diskspace* 5d⟩ ≡

```
    ⟨ log variable (6a disk capacity      ,6b disk_capacity ) 14c⟩
    ⟨ log variable (6c minimal required   ,6d min_diskfree ) 14c⟩
    ⟨ log variable (6e safe minimum       ,6f max_diskfree ) 14c⟩

    ◇
```

Fragment defined by 5abcd.
Fragment referenced in 4c.
Uses: `max_diskfree` 5c.

2.2 Determine whether quota should be expanded or reduced

Varianble `change` is going to indicate whether we have to increase or decrease quota.

⟨ *determine whether quota should be reduced or possibly expanded* 6g ⟩ ≡

```
change="No"
if
  [ $disk_free -lt $min_diskfree ]
then
  change="Dec"
elif
  [ $disk_free -gt $max_diskfree ]
then
  change="Inc"
fi
◇
```

Fragment referenced in 4c.
Defines: `change` 6h, 8g.
Uses: `disk_free` 5a, `max_diskfree` 5c, `min_diskfree` 5c.

2.3 Change the quota

If we have to change the quota, we must first find out what the quota currently are, then calculate what the quota should be, and finally set the new quota.

Note, that when variable `change` tells us to increase the quota, it is possible that we do not want to do that because the quota have already reached their maximum values. In that case, we set the new value for the quotum equal to the current value.

⟨ *expand or reduce quota* 6h ⟩ ≡

```
if
  [ ! "$change" == "No" ]
then
  ⟨ find out what the quota currently are 6i ⟩
  ⟨ calculate new quota 8g, ... ⟩
  if [ ! $new_hardquotum == $current_quotum ]
  then
    ⟨ activate new quota 10b, ... ⟩
  fi
fi
◇
```

Fragment referenced in 4c.
Uses: `change` 6g.

To find out what the quota currently are, find the quota of a random regular user:

- Find the name of a user of the “user” group in `/etc/passwd`.
- Find her quota in a “quota report”.

⟨ *find out what the quota currently are* 6i ⟩ ≡

```
⟨ find the name of a regular user (6j sixpack ) 8b ⟩
⟨ get hard quotum of user (7a $sixpack,7b current_quotum ) 4a ⟩
⟨ log variable (7c old quotum          ,8a max_diskfree ) 14c ⟩
◇
```

Fragment referenced in 6h.
Uses: `max_diskfree` 5c.

⟨ *find the name of a regular user* 8b ⟩ ≡
```
while
  read line
do
    ⟨ find username in line of password-file (8c $line,8d user ) 3c ⟩
    ⟨ find group-id in line of password-file (8e $line,8f group_id ) 3d ⟩
    if
      [ $group_id -eq $usergroup_id ]
    then
      @1=$user
      break
    fi
done < /etc/passwd
◇
```
Fragment referenced in 6i.

If the quota should be reduced, multiply the current hard-quotum with the decrease-fraction. If the quota might possibly be increased, first look whether the quotum has not yet attained its max.

⟨ *calculate new quota* 8g ⟩ ≡
```
current_quotum_onep=${current_quotum%??}
if
  [ "$change" == "Dec" ]
then
  new_hardquotum=$((reduction_perc*current_quotum_onep))
else
  new_hardquotum=$current_quotum
  max_hardquotum=$((max_quota_perc*$disk_capacity_onep))
  if
    [ $current_quotum -lt $max_hardquotum ]
  then
    new_hardquotum=$((expansion_perc*current_quotum_onep))
  fi
fi
⟨ log variable (8h max quotum          ,8i max_hardquotum ) 14c ⟩
⟨ log variable (8j new quotum          ,8k new_hardquotum ) 14c ⟩
◇
```
Fragment defined by 8gl, 10a.
Fragment referenced in 6h.

We have to set a soft-max and a quota for students. When a user occupies more diskspace than the the soft-max limit, she wil get warnings.

⟨ *calculate new quota* 8l ⟩ ≡
```
new_hardquotum_onep=${new_hardquotum%??}
new_softquotum=$((soft_perc*$new_hardquotum_onep))
◇
```
Fragment defined by 8gl, 10a.
Fragment referenced in 6h.
Uses: `soft_perc` 3a.

⟨ *calculate new quota* 10a ⟩ ≡
```
new_hardquotum_studs=$((10*$new_hardquotum_onep))
new_softquotum_studs=$((8*$new_hardquotum_onep))
◇
```
Fragment defined by 8gl, 10a.
Fragment referenced in 6h.

2.4 Activate new quota

Find the names of regular and student users and set the quota for each of them.

⟨ *activate new quota* 10b ⟩ ≡
```
while
  read line
do
  ⟨ find username in line of password-file (10c $line,10d user ) 3c ⟩
  ⟨ find group-id in line of password-file (10e $line,11a group_id ) 3d ⟩
  if
    [ $group_id == $usergroup_id ]
  then
    ⟨ set quota of a user (11b $user,12a $new_softquotum,12b $new_hardquotum ) 4b ⟩
  elif
    [ $group_id == $studgroup_id ]
  then
    ⟨ set quota of a user (12c $user,13a $new_softquotum_studs,13b $new_hardquotum_studs ) 4b ⟩
  fi
done < /etc/passwd
◇
```
Fragment defined by 10b, 14a.
Fragment referenced in 6h.

It seems that the quotacheck program has to be performed after modifying quota.

⟨ *activate new quota* 14a ⟩ ≡
```
quotaoff /
quotacheck -vgum /
quotaon /
◇
```
Fragment defined by 10b, 14a.
Fragment referenced in 6h.
Defines: `quotacheck` Never used, `quotaon` Never used.

2.5 Logging

When the user gives argument `-l`, write a summary of the numbers.

First, find out whether the user gave the argument

⟨ *set logging* 14b ⟩ ≡

```
key="$1"
if
  [ "$key" == "-l" ]
then
  logging="y"
else
  logging="n"
fi
```

◇

Fragment referenced in 4c.
Defines: `logging` 4c, 14c.

The following macro prints a variable if logging is on:

⟨ *log variable* 14c ⟩ ≡

```
if
  [ "logging" == "y" ]
then
  echo "@1:" $@2
fi
```

◇

Fragment referenced in 5d, 6i, 8g.
Uses: `logging` 14b.

A How to read and translate this document

This document is an example of *literate programming* [2]. It contains the code of all sorts of scripts and programs, combined with explaining texts. In this document the literate programming tool `nuweb` is used, that is currently available from Sourceforge (URL:`nuweb.sourceforge.net`). The advantages of Nuweb are, that it can be used for every programming language and scripting language, that it can contain multiple program sources and that it is very simple.

A.1 Read this document

The document contains *code scraps* that are collected into output files. An output file (e.g. `output.fil`) shows up in the text as follows:

`"output.fil"` 4a ≡

```
# output.fil
```

< a macro 4b >
< another macro 4c >
◇

The above construction contains text for the file. It is labelled with a code (in this case 4a) The constructions between the < and > brackets are macro's, placeholders for texts that can be found in other places of the document. The test for a macro is found in constructions that look like:

< a macro 4b > ≡

```
This is a scrap of code inside the macro.
It is concatenated with other scraps inside the
macro. The concatenated scraps replace
```

```
     the invocation of the macro.
```

```
Macro defined by 4b, 87e
Macro referenced in 4a
```

Macro's can be defined on different places. They can contain other macroÂ´s.

< a scrap 87e > ≡

```
     This is another scrap in the macro. It is
     concatenated to the text of scrap 4b.
     This scrap contains another macro:
     < another macro 45b >
```

```
Macro defined by 4b, 87e
Macro referenced in 4a
```

A.2 Process the document

The raw document is named `a_kyotoquota.w`. Figure 1 shows pathways to translate it into print-

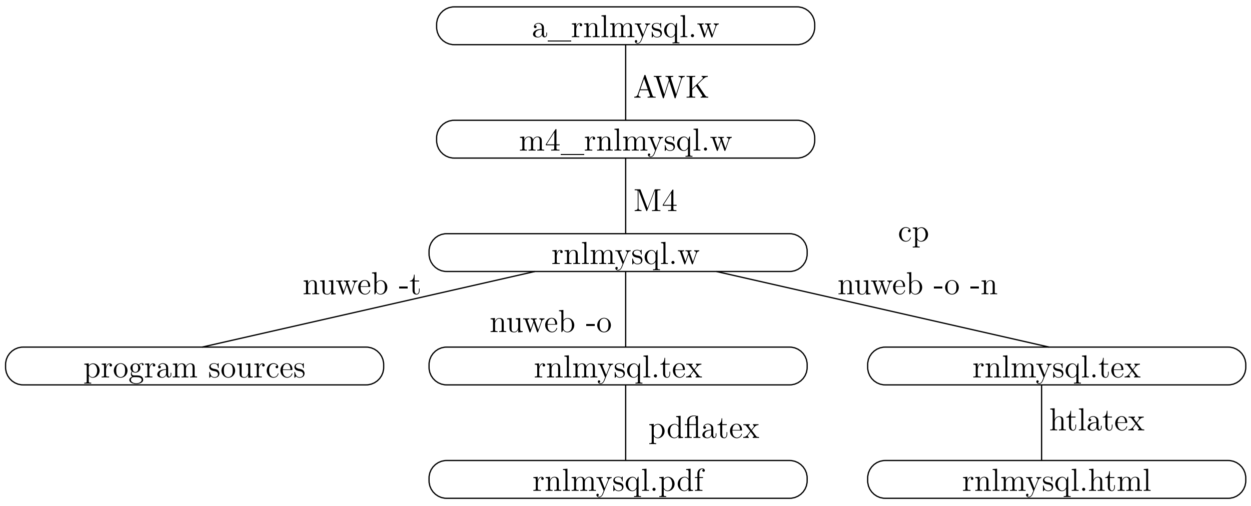

Figure 1: Translation of the raw code of this document into printable/viewable documents and into program sources. The figure shows the pathways and the main files involved.

able/viewable documents and to extract the program sources. Table 1 lists the tools that are

Tool	Source	Description
gawk	`www.gnu.org/software/gawk/`	text-processing scripting language
M4	`www.gnu.org/software/m4/`	Gnu macro processor
nuweb	`nuweb.sourceforge.net`	Literate programming tool
tex	`www.ctan.org`	Typesetting system
tex4ht	`www.ctan.org`	Convert TEX documents into `xml/html`

Table 1: Tools to translate this document into readable code and to extract the program sources

needed for a translation. Most of the tools (except Nuweb) are available on a well-equipped Linux system.

⟨ *parameters in Makefile* 14d ⟩ ≡
```
NUWEB=/usr/local/bin/nuweb
◇
```
Fragment defined by 14d, 15c, 16bc, 17e, ?, ?.
Fragment referenced in 14e.
Uses: `nuweb` 19d.

A.3 Translate and run

This chapter assembles the Makefile for this project.

`"Makefile"` 14e≡
> ⟨ *default target* 15a ⟩
>
> ⟨ *parameters in Makefile* 14d, . . . ⟩
>
> ⟨ *impliciete make regels* 17a, . . . ⟩
> ⟨ *expliciete make regels* 15d, . . . ⟩
> ⟨ *make targets* 17c, . . . ⟩
> ◇

The default target of make is `all`.

⟨ *default target* 15a ⟩ ≡
```
all : ⟨ all targets 15b ⟩
.PHONY : all

◇
```
Fragment referenced in 14e.
Defines: `all` Never used, `PHONY` 17b.

One of the targets is certainly the PDF version of this document.

⟨ *all targets* 15b ⟩ ≡
```
kyotoquota.pdf◇
```
Fragment referenced in 15a.
Uses: `pdf` 17c.

We use many suffixes that were not known by the C-programmers who constructed the `make` utility. Add these suffixes to the list.

⟨ *parameters in Makefile* 15c ⟩ ≡
```
.SUFFIXES: .pdf .w .tex .html .aux .log .php

◇
```
Fragment defined by 14d, 15c, 16bc, 17e, ?, ?.
Fragment referenced in 14e.
Defines: `SUFFIXES` Never used.
Uses: `pdf` 17c.

A.4 Pre-processing

To make usable things from the raw input `a_kyotoquota.w`, do the following:

1. Process `$` characters.
2. Run the m4 pre-processor.
3. Run nuweb.

This results in a LaTeX file, that can be converted into a PDF or a HTML document, and in the program sources and scripts.

A.4.1 Process 'dollar' characters

Many "intelligent" TeX editors (e.g. the auctex utility of Emacs) handle `$` characters as special, to switch into mathematics mode. This is irritating in program texts, that often contain `$` characters as well. Therefore, we make a stub, that translates the two-character sequence `\$` into the single `$` character.

⟨ *expliciete make regels* 15d ⟩ ≡

```
m4_kyotoquota.w : a_kyotoquota.w
        gawk '{if(match($$0, "@%")) {printf("%s", substr($$0,1,RSTART-1))} else print}' a_kyotoquota.
          | gawk '{gsub(/[\\][\$$]/, "$$");print}'  > m4_kyotoquota.w

◇
```

Fragment defined by 15d, 16a, 17b, 18b, ?, ?, ?, ?.
Fragment referenced in 14e.
Uses: `print` 17c.

A.4.2 Run the M4 pre-processor

⟨ *expliciete make regels* 16a ⟩ ≡

```
kyotoquota.w : m4_kyotoquota.w
        m4 -P m4_kyotoquota.w > kyotoquota.w

◇
```

Fragment defined by 15d, 16a, 17b, 18b, ?, ?, ?, ?.
Fragment referenced in 14e.

A.5 Typeset this document

Enable the following:

1. Create a PDF document.
2. Print the typeset document.
3. View the typeset document with a viewer.
4. Create a HTMLdocument.

In the three items, a typeset PDF document is required or it is the requirement itself.

A.5.1 Figures

This document contains figures that have been made by `xfig`. Post-process the figures to enable inclusion in this document.

The list of figures to be included:

⟨ *parameters in Makefile* 16b ⟩ ≡
```
FIGFILES=fileschema
```

◇

Fragment defined by 14d, 15c, 16bc, 17e, ?, ?.
Fragment referenced in 14e.
Defines: `FIGFILES` 16c, ?.

We use the package `figlatex` to include the pictures. This package expects two files with extensions `.pdftex` and `.pdftex_t` for `pdflatex` and two files with extensions `.pstex` and `.pstex_t` for the `latex/dvips` combination. Probably tex4ht uses the latter two formats too.

Make lists of the graphical files that have to be present for latex/pdflatex:

⟨ *parameters in Makefile* 16c ⟩ ≡
```
FIGFILENAMES=$(foreach fil,$(FIGFILES), $(fil).fig)
PDFT_NAMES=$(foreach fil,$(FIGFILES), $(fil).pdftex_t)
PDF_FIG_NAMES=$(foreach fil,$(FIGFILES), $(fil).pdftex)
PST_NAMES=$(foreach fil,$(FIGFILES), $(fil).pstex_t)
PS_FIG_NAMES=$(foreach fil,$(FIGFILES), $(fil).pstex)
```

◇

Fragment defined by 14d, 15c, 16bc, 17e, ?, ?.
Fragment referenced in 14e.
Defines: `FIGFILENAMES` Never used, `PDFT_NAMES` 17d, `PDF_FIG_NAMES` 17d, `PST_NAMES` Never used,
`PS_FIG_NAMES` Never used.
Uses: `FIGFILES` 16b.

Create the graph files with program `fig2dev`:

⟨ *impliciete make regels* 17a ⟩ ≡
```
%.eps: %.fig
        fig2dev -L eps $< > $@

%.pstex: %.fig
        fig2dev -L pstex $< > $@

.PRECIOUS : %.pstex
%.pstex_t: %.fig %.pstex
        fig2dev -L pstex_t -p $*.pstex $< > $@

%.pdftex: %.fig
        fig2dev -L pdftex $< > $@

.PRECIOUS : %.pdftex
%.pdftex_t: %.fig %.pstex
        fig2dev -L pdftex_t -p $*.pdftex $< > $@
```

◇

Fragment defined by 17ad, ?.
Fragment referenced in 14e.
Defines: `fig2dev` Never used.

A.5.2 Bibliography

To keep this document portable, create a portable bibliography file. It works as follows: This document refers in the |bibliography| statement to the local `bib`-file `kyotoquota.bib`. To create this file, copy the auxiliary file to another file `auxfil.aux`, but replace the argument of the command `\bibdata{kyotoquota}` to the names of the bibliography files that contain the actual references (they should exist on the computer on which you try this). This procedure should only be performed on the computer of the author. Therefore, it is dependent of a binary file on his computer.

⟨ *expliciete make regels* 17b ⟩ ≡

```
bibfile : kyotoquota.aux /home/paul/bin/mkportbib
        /home/paul/bin/mkportbib kyotoquota litprog

.PHONY : bibfile
◇
```

Fragment defined by 15d, 16a, 17b, 18b, ?, ?, ?, ?.
Fragment referenced in 14e.
Uses: `PHONY` 15a.

A.5.3 Create a printable/viewable document

Make a PDF document for printing and viewing.

⟨ *make targets* 17c ⟩ ≡

```
pdf : kyotoquota.pdf

print : kyotoquota.pdf
        lpr kyotoquota.pdf

view : kyotoquota.pdf
        evince kyotoquota.pdf

◇
```

Fragment defined by 17c, 20b, ?, ?.
Fragment referenced in 14e.
Defines: `pdf` 15bc, 17d, `print` 3cd, 4a, 5a, 15d, `view` Never used.

Create the PDF document. This may involve multiple runs of nuweb, the LaTeX processor and the bibTeX processor, and depends on the state of the `aux` file that the LaTeX processor creates as a by-product. Therefore, this is performed in a separate script, `w2pdf`.

The w2pdf script The three processors nuweb, LaTeX and bibTeX are intertwined. LaTeX and bibTeX create parameters or change the value of parameters, and write them in an auxiliary file. The other processors may need those values to produce the correct output. The LaTeX processor may even need the parameters in a second run. Therefore, consider the creation of the (PDF) document finished when none of the processors causes the auxiliary file to change. This is performed by a shell script `w2pdf`.

Note, that in the following `make` construct, the implicit rule `.w.pdf` is not used. It turned out, that make did not calculate the dependencies correctly when I did use this rule.

⟨ *impliciete make regels* 17d ⟩ ≡
```
%.pdf : %.w $(W2PDF)  $(PDF_FIG_NAMES) $(PDFT_NAMES) %.bib
        chmod 775 $(W2PDF)
        $(W2PDF) $*

◇
```
Fragment defined by 17ad, ?.
Fragment referenced in 14e.
Uses: `pdf` 17c, `PDFT_NAMES` 16c, `PDF_FIG_NAMES` 16c.

The following is an ugly fix of an unsolved problem. Currently I develop this thing, while it resides on a remote computer that is connected via the `sshfs` filesystem. On my home computer I cannot run executables on this system, but on my work-computer I can. Therefore, place the following script on a local directory.

⟨ *parameters in Makefile* 17e ⟩ ≡
```
W2PDF=../nuweb/bin/w2pdf
◇
```
Fragment defined by 14d, 15c, 16bc, 17e, ?, ?.
Fragment referenced in 14e.
Uses: `nuweb` 19d.

⟨ *directories to create* 18a ⟩ ≡
```
../nuweb/bin ◇
```
Fragment referenced in ?.
Uses: `nuweb` 19d.

⟨ *expliciete make regels* 18b ⟩ ≡
```
$(W2PDF) : kyotoquota.w
        $(NUWEB) kyotoquota.w
◇
```
Fragment defined by 15d, 16a, 17b, 18b, ?, ?, ?, ?.
Fragment referenced in 14e.

`"../nuweb/bin/w2pdf"` 18c≡
```
#!/bin/bash
# w2pdf -- compile a nuweb file
# usage: w2pdf [filename]
# 20170501 at 1016h: Generated by nuweb from a_kyotoquota.w
NUWEB=/usr/local/bin/nuweb
LATEXCOMPILER=pdflatex
⟨ filenames in nuweb compile script 19b ⟩
⟨ compile nuweb 19a ⟩

◇
```
Uses: `nuweb` 19d.

The script retains a copy of the latest version of the auxiliary file. Then it runs the four processors nuweb, LaTeX, MakeIndex and bibTeX, until they do not change the auxiliary file or the index.

⟨ *compile nuweb* 19a ⟩ ≡
```
    NUWEB=m4_nuweb
    ⟨ run the processors until the aux file remains unchanged 20a ⟩
    ⟨ remove the copy of the aux file 19c ⟩
    ◇
```
Fragment referenced in 18c.

The user provides the name of the nuweb file as argument. Strip the extension (e.g. `.w`) from the filename and create the names of the LaTeX file (ends with `.tex`), the auxiliary file (ends with `.aux`) and the copy of the auxiliary file (add `old.` as a prefix to the auxiliary filename).

⟨ *filenames in nuweb compile script* 19b ⟩ ≡
```
    nufil=$1
    trunk=${1%%.*}
    texfil=${trunk}.tex
    auxfil=${trunk}.aux
    oldaux=old.${trunk}.aux
    indexfil=${trunk}.idx
    oldindexfil=old.${trunk}.idx
    ◇
```
Fragment referenced in 18c.
Defines: `auxfil` 20a, ?, ?, `indexfil` 20a, ?, `nufil` 19d, ?, ?, `oldaux` 19c, 20a, ?, ?, `oldindexfil` 20a, ?, `texfil` 19d, ?, ?, `trunk` 19d, ?, ?, ?.

Remove the old copy if it is no longer needed.

⟨ *remove the copy of the aux file* 19c ⟩ ≡
```
    rm $oldaux
    ◇
```
Fragment referenced in 19a, ?.
Uses: `oldaux` 19b, ?.

Run the three processors. Do not use the option `-o` (to suppres generation of program sources) for nuweb, because `w2pdf` must be kept up to date as well.

⟨ *run the three processors* 19d ⟩ ≡
```
    $NUWEB $nufil
    $LATEXCOMPILER $texfil
    makeindex $trunk
    bibtex $trunk
    ◇
```
Fragment referenced in 20a.
Defines: `bibtex` ?, ?, `makeindex` ?, ?, `nuweb` 14d, 17e, 18ac, ?.
Uses: `nufil` 19b, ?, `texfil` 19b, ?, `trunk` 19b, ?.

Repeat to copy the auxiliary file and the index file and run the processors until the auxiliary file and the index file are equal to their copies. However, since I have not yet been able to test the `aux` file and the `idx` in the same test statement, currently only the `aux` file is tested.

It turns out, that sometimes a strange loop occurs in which the `aux` file will keep to change. Therefore, with a counter we prevent the loop to occur more than 10 times.

⟨ *run the processors until the aux file remains unchanged* 20a ⟩ ≡
```
LOOPCOUNTER=0
while
  ! cmp -s $auxfil $oldaux
do
  if [ -e $auxfil ]
  then
   cp $auxfil $oldaux
  fi
  if [ -e $indexfil ]
  then
   cp $indexfil $oldindexfil
  fi
  ⟨ run the three processors 19d ⟩
  if [ $LOOPCOUNTER -ge 10 ]
  then
    cp $auxfil $oldaux
  fi;
done
◇
```
Fragment referenced in 19a.
Uses: `auxfil` 19b, ?, `indexfil` 19b, `oldaux` 19b, ?, `oldindexfil` 19b.

A.5.4 Create HTML files

HTML is easier to read on-line than a PDF document that was made for printing. We use `tex4ht` to generate HTML code. An advantage of this system is, that we can include figures in the same way as we do for `pdflatex`.

Nuweb creates a LaTeX file that is suitable for `latex2html` if the source file has `.hw` as suffix instead of `.w`. However, this feature is not compatible with tex4ht.

Make html file:

⟨ *make targets* 20b ⟩ ≡
```
html : m4_htmltarget

◇
```
Fragment defined by 17c, 20b, ?, ?.
Fragment referenced in 14e.

The HTML file depends on its source file and the graphics files.

Make lists of the graphics files and copy them.

⟨ *parameters in Makefile* ? ⟩ ≡
```
HTML_PS_FIG_NAMES=$(foreach fil,$(FIGFILES), m4_htmldocdir/$(fil).pstex)
HTML_PST_NAMES=$(foreach fil,$(FIGFILES), m4_htmldocdir/$(fil).pstex_t)
◇
```
Fragment defined by 14d, 15c, 16bc, 17e, ?, ?.
Fragment referenced in 14e.
Uses: `FIGFILES` 16b.

⟨ *impliciete make regels* ? ⟩ ≡
```
m4_htmldocdir/%.pstex : %.pstex
        cp  $< $@

m4_htmldocdir/%.pstex_t : %.pstex_t
        cp  $< $@

```
◇

Fragment defined by 17ad, ?.
Fragment referenced in 14e.

Copy the nuweb file into the html directory.

⟨ *expliciete make regels* ? ⟩ ≡
```
m4_htmlsource : kyotoquota.w
        cp  kyotoquota.w m4_htmlsource

```
◇

Fragment defined by 15d, 16a, 17b, 18b, ?, ?, ?, ?.
Fragment referenced in 14e.

We also need a file with the same name as the documentstyle and suffix `.4ht`. Just copy the file `report.4ht` from the tex4ht distribution. Currently this seems to work.

⟨ *expliciete make regels* ? ⟩ ≡
```
m4_4htfildest : m4_4htfilsource
        cp m4_4htfilsource m4_4htfildest

```
◇

Fragment defined by 15d, 16a, 17b, 18b, ?, ?, ?, ?.
Fragment referenced in 14e.

Copy the bibliography.

⟨ *expliciete make regels* ? ⟩ ≡
```
m4_htmlbibfil : m4_anuwebdir/kyotoquota.bib
        cp m4_anuwebdir/kyotoquota.bib m4_htmlbibfil

```
◇

Fragment defined by 15d, 16a, 17b, 18b, ?, ?, ?, ?.
Fragment referenced in 14e.

Make a dvi file with `w2html` and then run `htlatex`.

⟨ *expliciete make regels* ? ⟩ ≡
```

m4_htmltarget : m4_htmlsource m4_4htfildest $(HTML_PS_FIG_NAMES) $(HTML_PST_NAMES) m4_htmlbibfil
        cp w2html /home/huygen/projecten/kyoto/quota/kyotoquota/bin
        cd /home/huygen/projecten/kyoto/quota/kyotoquota/bin && chmod 775 w2html
        cd m4_htmldocdir && /home/huygen/projecten/kyoto/quota/kyotoquota/bin/w2html kyotoquota.w

```
◇

Fragment defined by 15d, 16a, 17b, 18b, ?, ?, ?, ?.
Fragment referenced in 14e.

Create a script that performs the translation.

"w2html" ?≡
```
#!/bin/bash
w2html -- make a html file from a nuweb file
usage: w2html [filename]
[filename]: Name of the nuweb source file.
`#' m4_header
echo "translate " $1 >w2html.log
NUWEB=/usr/local/bin/nuweb
⟨ filenames in w2html ? ⟩

⟨ perform the task of w2html ? ⟩

◇
```
Uses: `nuweb` 19d.

The script is very much like the `w2pdf` script, but at this moment I have still difficulties to compile the source smoothly into HTML and that is why I make a separate file and do not recycle parts from the other file. However, the file works similar.

⟨ *perform the task of w2html* ? ⟩ ≡
```
⟨ run the html processors until the aux file remains unchanged ? ⟩
⟨ remove the copy of the aux file 19c ⟩
◇
```
Fragment referenced in ?.

The user provides the name of the nuweb file as argument. Strip the extension (e.g. `.w`) from the filename and create the names of the LaTeX file (ends with `.tex`), the auxiliary file (ends with `.aux`) and the copy of the auxiliary file (add `old.` as a prefix to the auxiliary filename).

⟨ *filenames in w2html* ? ⟩ ≡
```
nufil=$1
trunk=${1%%.*}
texfil=${trunk}.tex
auxfil=${trunk}.aux
oldaux=old.${trunk}.aux
indexfil=${trunk}.idx
oldindexfil=old.${trunk}.idx
◇
```
Fragment referenced in ?.
Defines: `auxfil` 19b, 20a, ?, `nufil` 19bd, ?, `oldaux` 19bc, 20a, ?, `texfil` 19bd, ?, `trunk` 19bd, ?, ?.
Uses: `indexfil` 19b, `oldindexfil` 19b.

⟨ *run the html processors until the aux file remains unchanged* ? ⟩ ≡

```
while
  ! cmp -s $auxfil $oldaux
do
  if [ -e $auxfil ]
  then
   cp $auxfil $oldaux
  fi
  ⟨ run the html processors ? ⟩
done
⟨ run tex4ht ? ⟩

◇
```

Fragment referenced in ?.
Uses: `auxfil` 19b, ?, `oldaux` 19b, ?.

To work for HTML, nuweb *must* be run with the `-n` option, because there are no page numbers.

⟨ *run the html processors* ? ⟩ ≡

```
$NUWEB -o -n $nufil
latex $texfil
makeindex $trunk
bibtex $trunk
htlatex $trunk
◇
```

Fragment referenced in ?.
Uses: `bibtex` 19d, `makeindex` 19d, `nufil` 19b, ?, `texfil` 19b, ?, `trunk` 19b, ?.

When the compilation has been satisfied, run makeindex in a special way, run bibtex again (I don't know why this is necessary) and then run htlatex another time.

⟨ *run tex4ht* ? ⟩ ≡

```
tex '\def\filename{{kyotoquota}{idx}{4dx}{ind}} \input idxmake.4ht'
makeindex -o $trunk.ind $trunk.4dx
bibtex $trunk
htlatex $trunk
◇
```

Fragment referenced in ?.
Uses: `bibtex` 19d, `makeindex` 19d, `trunk` 19b, ?.

create the program sources Run nuweb, but suppress the creation of the LaTeX documentation. Nuweb creates only sources that do not yet exist or that have been modified. Therefore make does not have to check this. However, "make" has to create the directories for the sources if they do not yet exist. So, let's create the directories first.

⟨ *parameters in Makefile* ? ⟩ ≡

```
MKDIR=mkdir -p

◇
```

Fragment defined by 14d, 15c, 16bc, 17e, ?, ?.
Fragment referenced in 14e.
Defines: `MKDIR` ?.

⟨ *make targets ?* ⟩ ≡
```
      DIRS=⟨ directories to create 18a ⟩

      $(DIRS) :
              $(MKDIR) $@

      ◇
```
Fragment defined by 17c, 20b, ?, ?.
Fragment referenced in 14e.
Defines: `DIRS` ?.
Uses: `MKDIR` ?.

⟨ *make targets ?* ⟩ ≡
```
      sources : kyotoquota.w $(DIRS)
              $(NUWEB) kyotoquota.w
              cd .. && chmod 775 adaptquota

      jetty : sources
              cd .. && mvn jetty:run

      ◇
```
Fragment defined by 17c, 20b, ?, ?.
Fragment referenced in 14e.
Uses: `DIRS` ?.

B References

B.1 Literature

References

[1] Garrett Hardin. The tragedy of the commons. *Science*, 162(3859):1243:1248, 1968.

[2] Donald E. Knuth. Literate programming. Technical report STAN-CS-83-981, Stanford University, Department of Computer Science, 1983.

B.2 URL's

Nuweb: `nuweb.sourceforge.net`
Apache Velocity: `m4_velocityURL`
Velocitytools: `m4_velocitytoolsURL`
Parameterparser tool: `m4_parameterparserdocURL`
Cookietool: `m4_cookietooldocURL`
VelocityView: `m4_velocityviewURL`
VelocityLayoutServlet: `m4_velocitylayoutservletURL`
Jetty: `m4_jettycodehausURL`
UserBase javadoc: `m4_userbasejavadocURL`
VU corpus Management development site: `http://code.google.com/p/vucom`

C Indexes

C.1 Filenames

C.2 Macro's

C.3 Variables