I love deadlines.
I like the whooshing sound they make as they fly by.
(Douglas Adams)

Automatic binary building for TEX Live

(using buildbot)

Mojca Miklavec

TUG@BachoTEX, 1st May 2017

```
From: Karl Berry <karl@...>
To:   tlbuild <tlbuild@...>
```

Hopefully the synctex-related bugs have been ameliorated now, so please rebuild, those of you who haven't already.

--thanks, karl.

```
From: Mojca Miklavec <mojca.miklavec@...>
To:   Karl Berry <karl@...>
```

Dear Karl,

I know you are in the middle of finalizing the TEX Live 2017. I'm aware that I volunteered to build binaries for 8 platforms.

But I'm super busy at the moment and will be gone in April & May for two weeks, most likely without internet connectivity.

Sorry,
Mojca

Binaries in TeX Live

- ~20 platforms
- effort by a number of volunteers
- built “once” per year – reasonable compromise between:
  - demand for new binaries
  - burden on volunteer builders and packagers
  - stability & amount of testing (LuaTeX)

Disadvantages

- No access to the latest features for more than a year.
- Long and “painful” bugfixing period around March.

ConTEXt Distribution (I)

- ~13 + 2 platforms (5 more dropped for lack of interest)
- 100 % compatible with TEX Live
- Windows binaries from W32TEX
- other binaries built by volunteers
  with a single command (or by cronjobs)
- distribution checks for updates every 15 minuts,
  binaries for every MetaPost, LuaTEX, XƎTEX release
- (binaries found their way to TL users via TLContrib)

ConTEXt Distribution (II)

- many build problems discovered during the year
  (in particular on exotic platforms like Solaris),
  less debugging left for the TEX Live freeze period

- little work to do, but it would be even better
  if computers would do **all** the work for us

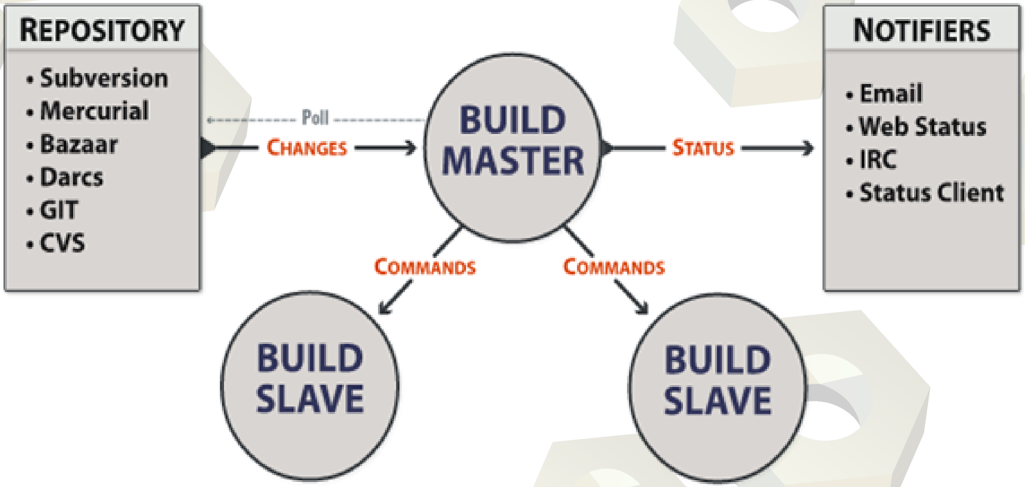
- number of builders decreased,
  Hans started setting up VMs for binary builds at Pragma

- it would be nice to have **nightly** LuaTEX builds

Automating the builds

What is Buildbot?

- framework for automating builds, highly costumisable (Python)
- **Build Master**: *central server, delegates work, collects results*
- **Build Slaves**: *multiple "stupid" computers to compile binaries*



Build Slaves

- a number of computers on a variety of platforms
- may live behind a firewall on a private network
- easy to set up
  - Debian: `sudo apt-get install buildbot-slave`
    PIP: `pip install buildbot-slave`

    ```
    buildslave create-slave /some/path your.server:9989 name pass
    ```
  - install just compiler and any build dependencies,
    no need to worry **what** to build / **how** to build it

Build Master

- a single server, publicly accessible
- slaves connect to it with username & password
- contains all the “brains” to delegate work:
    - checks for software updates, schedules builds
    - sends build commands to build slaves
    - sends emails on build failures
    - collects results, may distributes binaries

Build Slaves

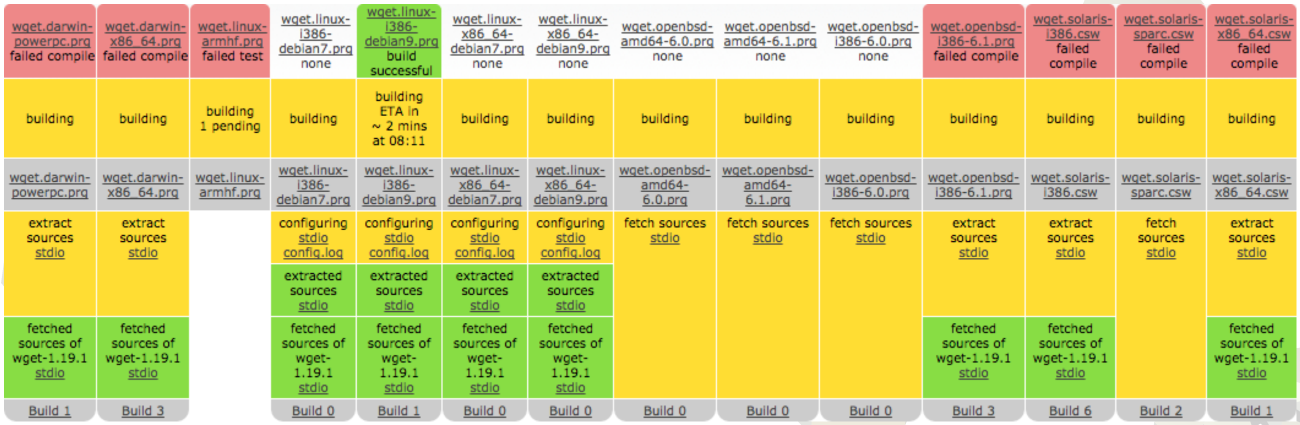
- (4) Solaris: sparc (sparc64), i386 (x86_64) @ opencsw.org
- (4) Mac OS X 10.6: (x86_64, i386, ppc) + macOS 10.12
- (5) Linux:
  - Raspbian (armhf) @ Raspberry PI
  - Debian 7 (oldstable) & 9 (testing), (i386, x86_64)
- (4) OpenBSD: 6.0, 6.1 (i386, amd64)

TODO:

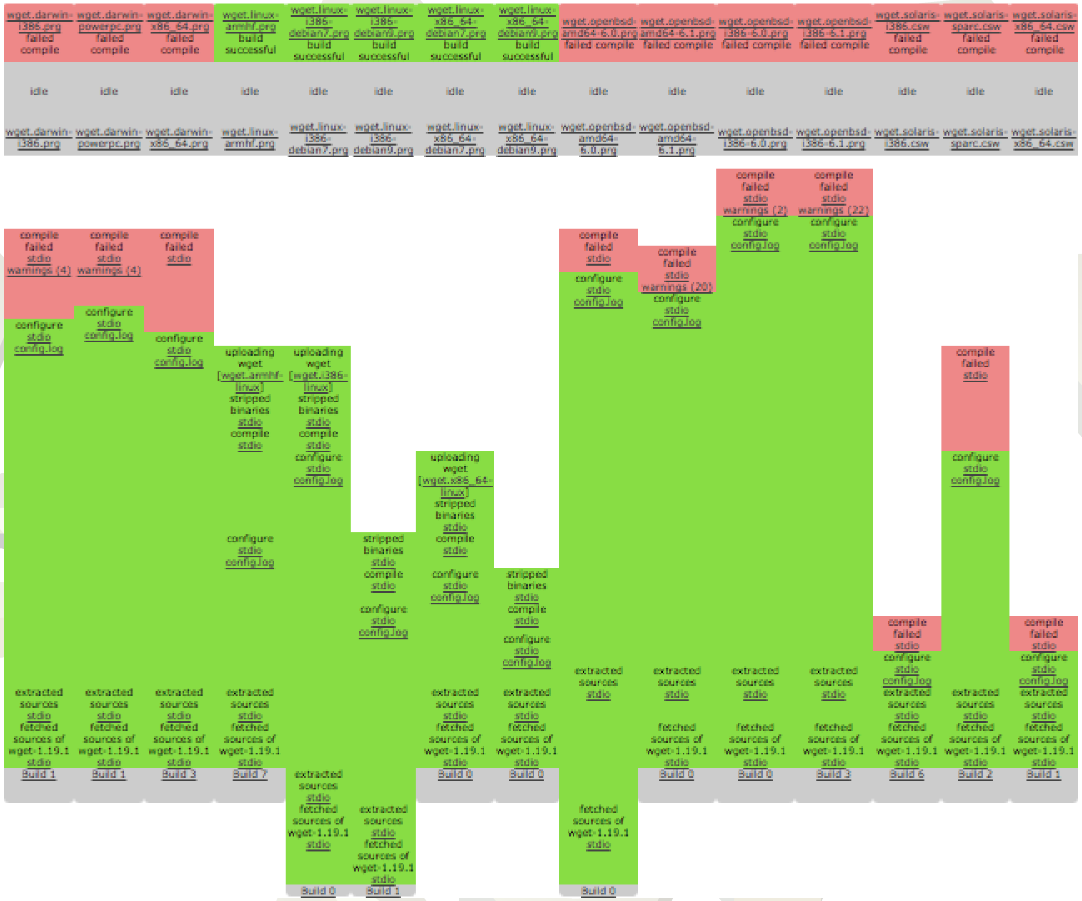
- (2) Linux: CentOS 5.11 (i386, x86_64)
- (4) FreeBSD: 9.3 & 11.0 (i386, amd64)
- (2?) NetBSD
- (2) mingw64 on Linux for cross-compilation (win32, win64)

Demo

Starting the build



wget.darwin-powerpc.prg failed compile	wget.darwin-x86_64.prg failed compile	wget.linux-armhf.prg failed test	wget.linux-i386-debian7.prg none	wget.linux-i386-debian9.prg build successful	wget.linux-x86_64-debian7.prg none	wget.linux-x86_64-debian9.prg none	wget.openbsd-amd64-6.0.prg none	wget.openbsd-amd64-6.1.prg none	wget.openbsd-i386-6.0.prg none	wget.openbsd-i386-6.1.prg failed compile	wget.solaris-i386.csw failed compile	wget.solaris-sparc.csw failed compile	wget.solaris-x86_64.csw failed compile
building	building	building 1 pending	building	building ETA in ~ 2 mins at 08:11	building	building	building	building	building	building	building	building	building
wget.darwin-powerpc.prg	wget.darwin-x86_64.prg	wget.linux-armhf.prg	wget.linux-i386-debian7.prg	wget.linux-i386-debian9.prg	wget.linux-x86_64-debian7.prg	wget.linux-x86_64-debian9.prg	wget.openbsd-amd64-6.0.prg	wget.openbsd-amd64-6.1.prg	wget.openbsd-i386-6.0.prg	wget.openbsd-i386-6.1.prg	wget.solaris-i386.csw	wget.solaris-sparc.csw	wget.solaris-x86_64.csw
extract sources stdio	extract sources stdio		configuring stdio config.log	configuring stdio config.log	configuring stdio config.log	configuring stdio config.log	fetch sources stdio	fetch sources stdio	fetch sources stdio	extract sources stdio	extract sources stdio	fetch sources stdio	extract sources stdio
			extracted sources stdio	extracted sources stdio	extracted sources stdio	extracted sources stdio							
fetched sources of wget-1.19.1 stdio	fetched sources of wget-1.19.1 stdio		fetched sources of wget-1.19.1 stdio	fetched sources of wget-1.19.1 stdio	fetched sources of wget-1.19.1 stdio	fetched sources of wget-1.19.1 stdio				fetched sources of wget-1.19.1 stdio	fetched sources of wget-1.19.1 stdio		fetched sources of wget-1.19.1 stdio
Build 1	Build 3		Build 0	Build 1	Build 0	Build 0	Build 0	Build 0	Build 0	Build 3	Build 6	Build 2	Build 1



wget.darwin-i386.prg failed compile
idle
wget.darwin-i386.prg
compile failed stdio warnings (4)
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 1

wget.darwin-powerpc.prg failed compile
idle
wget.darwin-powerpc.prg
compile failed stdio warnings (4)
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 1

wget.darwin-x86_64.prg failed compile
idle
wget.darwin-x86_64.prg
compile failed stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 3

wget.linux-armhf.prg build successful
idle
wget.linux-armhf.prg
uploading wget [wget.armhf-linux]
stripped binaries stdio
compile stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 7

wget.linux-i386-debian7.prg build successful
idle
wget.linux-i386-debian7.prg
uploading wget [wget.i386-linux]
stripped binaries stdio
compile stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 0

wget.linux-i386-debian9.prg build successful
idle
wget.linux-i386-debian9.prg
stripped binaries stdio
compile stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 1

wget.linux-x86_64-debian7.prg build successful
idle
wget.linux-x86_64-debian7.prg
uploading wget [wget.x86_64-linux]
stripped binaries stdio
compile stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 0

wget.linux-x86_64-debian9.prg build successful
idle
wget.linux-x86_64-debian9.prg
stripped binaries stdio
compile stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 0

wget.openbsd-amd64-6.0.prg failed compile
idle
wget.openbsd-amd64-6.0.prg
compile failed stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 0

wget.openbsd-amd64-6.1.prg failed compile
idle
wget.openbsd-amd64-6.1.prg
compile failed stdio warnings (20)
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 0

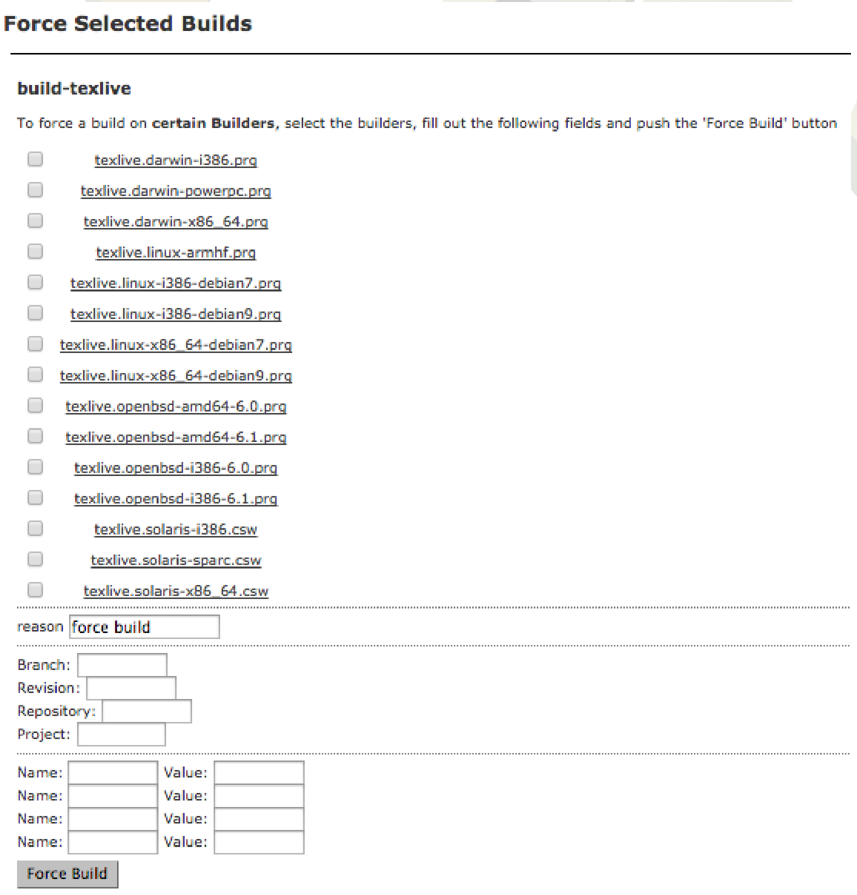
wget.openbsd-i386-6.0.prg failed compile
idle
wget.openbsd-i386-6.0.prg
compile failed stdio warnings (2)
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 0

wget.openbsd-i386-6.1.prg failed compile
idle
wget.openbsd-i386-6.1.prg
compile failed stdio warnings (22)
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 3

wget.solaris-i386.csw failed compile
idle
wget.solaris-i386.csw
compile failed stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 6

wget.solaris-sparc.csw failed compile
idle
wget.solaris-sparc.csw
compile failed stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 2

wget.solaris-x86_64.csw failed compile
idle
wget.solaris-x86_64.csw
compile failed stdio
configure stdio config.log
extracted sources stdio
fetched sources of wget-1.19.1 stdio
Build 1



Force Selected Builds

build-texlive

To force a build on **certain Builders**, select the builders, fill out the following fields and push the 'Force Build' button

- ☐ texlive.darwin-i386.prg
- ☐ texlive.darwin-powerpc.prg
- ☐ texlive.darwin-x86_64.prg
- ☐ texlive.linux-armhf.prg
- ☐ texlive.linux-i386-debian7.prg
- ☐ texlive.linux-i386-debian9.prg
- ☐ texlive.linux-x86_64-debian7.prg
- ☐ texlive.linux-x86_64-debian9.prg
- ☐ texlive.openbsd-amd64-6.0.prg
- ☐ texlive.openbsd-amd64-6.1.prg
- ☐ texlive.openbsd-i386-6.0.prg
- ☐ texlive.openbsd-i386-6.1.prg
- ☐ texlive.solaris-i386.csw
- ☐ texlive.solaris-sparc.csw
- ☐ texlive.solaris-x86_64.csw

reason: force build

Branch:
Revision:
Repository:
Project:

Name: Value:
Name: Value:
Name: Value:
Name: Value:

Advantages & Opportunities (I)

- build binaries for all platforms:
  after each commit / daily / manually triggered
- automatically send emails when something breaks,
  less problems left for TEX Live pretest period
- get the latest binaries to (adventurous) TEX Live users
- no need for Karl to send "*please rebuild now*" emails
  and wait until people have time

Advantages & Opportunities (II)

- also build wget, xz, asymptote, xindy, …
- compile different programs with different compilers (C++11, bug in ICU & upmendex, …)
- one could build/test development versions of dependencies (icu, poppler, libpng, luajit, …), detect problems & get fixes before release
- adding some of the 150 VMs from Utah to the list to extend the OS coverage
- testing for reproducibility of builds

More work to do

- complete setups for different software & components
- add encryption, binary signing, signature checking of sources
- write more test cases to detect problems / consistency
- set up proper private TEX Live repositories
  with updated binaries

… After Karl fixed a problem in TL sources,

while no internet connectivity was available at BT …

```
From: Karl Berry <karl@...>
To:   Mojca Miklavec <mojca.miklavec@...>
```

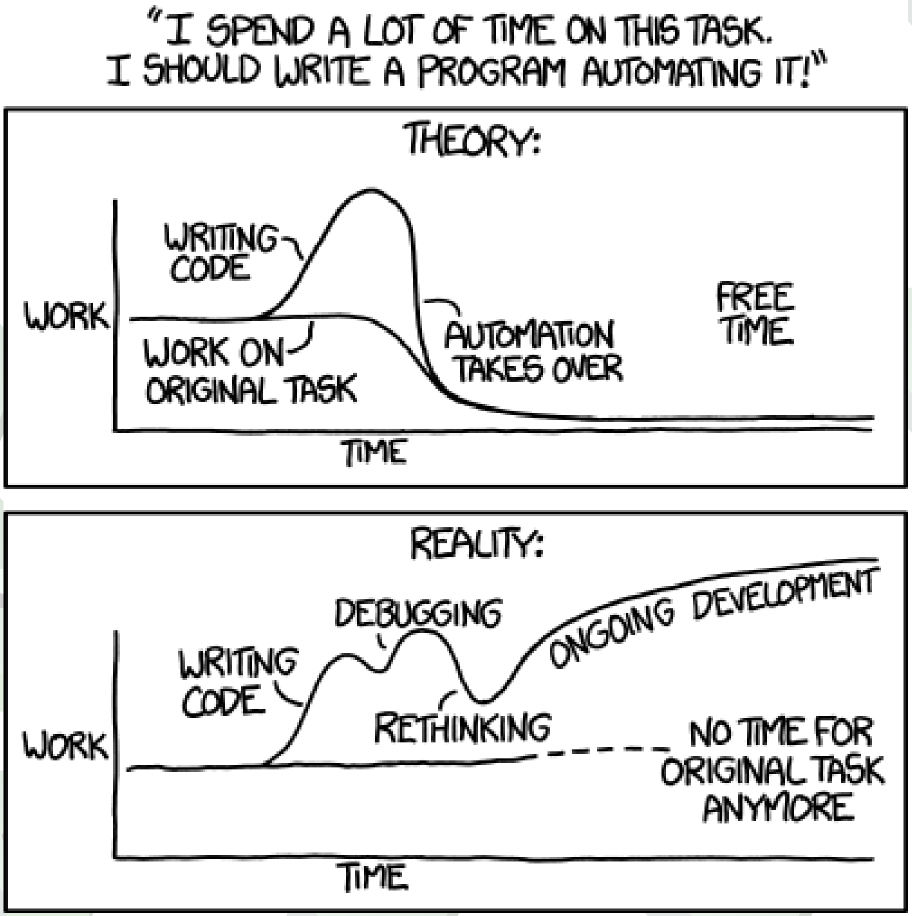
I imported all the binaries.

Oh, and I just found the OpenBSD binaries on the server. Do you mind if I include also those in TL?

Karl

Do we still need manpower?

- **Definitely.** We need skilled people to:
    - figure out which OS version / compiler / flags to use
    - setup working machines and keep them “up to date”
    - **debug** problems, report problems and **fix bugs**
    - **IMPROVE THE BUILD SYSTEM!**
- But there should be no need for **repetitive** task of re-running the same script each night, wait for the build to finish, check for consistency, sync, commit to repository.
- It would help to have a backup person, familiar with the sources and the full process.


"I SPEND A LOT OF TIME ON THIS TASK. I SHOULD WRITE A PROGRAM AUTOMATING IT!"
THEORY:
WRITING CODE
WORK
WORK ON ORIGINAL TASK
AUTOMATION TAKES OVER
FREE TIME
TIME
REALITY:
DEBUGGING
WRITING CODE
RETHINKING
ONGOING DEVELOPMENT
WORK
NO TIME FOR ORIGINAL TASK ANYMORE
TIME

Special thanks

- **Alan Braslau**
  - help setting up, debugging & maintaining lots of VMs
- **Hans Hagen** @ PRAGMA ADE
  - dedicated server with lots of VMs
- **Dagobert Michelsen** @ OpenCSW project
  - Solaris build farm, help with buildbot setup
- **Norbert Preining**
  - additional functionality in TL
- **Taco Hoekwater**
  - TLContrib
- **Johnny** @ Jožef Stefan Institute
  - administration and bandwidth for the main server