Computational Symbiosis
Methods That Meld Mind and Machine

Mike Gerwitz

LibrePlanet 2019
Choreographed Workflows
Practical Freedom
Example: Web Browser

GNU Operating System

Sponsored by the Free Software Foundation

What is GNU?

GNU is an operating system that is free software—that is, it respects users' freedom. The GNU operating system consists of GNU packages (programs specifically released by the GNU Project) as well as free software released by third parties. The development of GNU made it possible to use a computer without software that would trample your freedom.

We recommend installable versions of GNU (more precisely, GNU/Linux distributions) which are entirely free software. More about GNU below.

Try GNU/Linux
Finding Text (Mouse-Driven GUI Interaction)

LibrePlanet is an annual conference hosted by the Free Software Foundation for free software enthusiasts and anyone who cares about the intersection of technology and social justice. LibrePlanet brings together software developers, law and policy experts, activists, students, and computer users to learn skills, celebrate free software accomplishments, and face challenges to software freedom. Newcomers are always welcome, and LibrePlanet 2019 will feature programming for all ages and experience levels.

LibrePlanet 2019's theme is “Trailblazing Free Software.” In 1983, the free software movement was born with the announcement of the GNU Project. FSF founder Richard Stallman saw the dangers of proprietary code from the beginning: when code was kept secret from
GUIs Change Over Time

Ctrl+F

Mike Gerwitz
Computational Symbiosis
LibrePlanet 2019
GUIs Change Over Time

Ctrl+F
Muscle Memory

Visual $\Rightarrow$ Tactile
Research Task:

Given a list of webpage URLs find all that *do not* contain “free software”
Executing the Research Task

Mouse

1. Click ‘+’ for each new tab, enter URL
2. Menu → Find in This Page
3. Type “free software”
4. If found, go to #9
5. If not found, highlight URL, right-click, copy
6. Click on text editor
7. Right-click, paste URL, hit RET for newline
8. Click on web browser
9. Click ‘X’ on tab, go to #2
## Executing the Research Task

<table>
<thead>
<tr>
<th><strong>Mouse</strong></th>
<th><strong>Keyboard</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1</strong> Click ‘+’ for each new tab, enter URL</td>
<td><strong>1</strong> $\text{Ctrl}+\text{T}$ for each new tab, enter URL</td>
</tr>
<tr>
<td><strong>2</strong> Menu $\rightarrow$ Find in This Page</td>
<td><strong>2</strong> $\text{Ctrl}+\text{F}$ to find</td>
</tr>
<tr>
<td><strong>3</strong> Type “free software”</td>
<td><strong>3</strong> Type “free software”</td>
</tr>
<tr>
<td><strong>4</strong> If found, go to #9</td>
<td><strong>4</strong> If found, go to #9</td>
</tr>
<tr>
<td><strong>5</strong> If <em>not</em> found, highlight URL, right-click, copy</td>
<td><strong>5</strong> If <em>not</em> found, $\text{Ctrl}+\text{L}$ $\text{Ctrl}+\text{C}$ to copy URL</td>
</tr>
<tr>
<td><strong>6</strong> Click on text editor</td>
<td><strong>6</strong> $\text{Alt}+\text{Tab}$ to text editor</td>
</tr>
<tr>
<td><strong>7</strong> Right-click, paste URL, hit RET for newline</td>
<td><strong>7</strong> $\text{Ctrl}+\text{V}$ RET to paste URL and add newline</td>
</tr>
<tr>
<td><strong>8</strong> Click on web browser</td>
<td><strong>8</strong> $\text{Alt}+\text{Tab}$ to web browser</td>
</tr>
<tr>
<td><strong>9</strong> Click ‘X’ on tab, go to #2</td>
<td><strong>9</strong> $\text{Ctrl}+\text{W}$ to close tab, go to #2</td>
</tr>
</tbody>
</table>
Same Keybindings Across (Most) GUIs!

Browser, Editor, Window Manager, OS, ...
Macro-Like

Ctrl+T ‘‘https://...’’ <N times>

Ctrl+F ‘‘free sofware’’

[ Ctrl+L Ctrl+C Alt+Tab Ctrl+V RET Alt+Tab ]

Ctrl+W

<N times>
Macro-Like

Ctrl+T ‘‘https://...’’ <N times>

Ctrl+F ‘‘free sofware’’
[ Ctrl+L Ctrl+C Alt+Tab Ctrl+V RET Alt+Tab ]
Ctrl+W
<N times>

- Requires visual inspection for conditional
- Still manual and tedious—what if there were 1000 URLs?
Lifting the Curtain

Mike Gerwitz

Computational Symbiosis

LibrePlanet 2019
Text.
Text is a Universal Interface
The Shell Command Prompt

mikegerwitz@lp2019-laptop:~$
# ~ user ~ host ~ working directory (home)
The Shell Command Prompt

mikegerwitz@lp2019-laptop:~$
# ~ user ~ host ~ working directory (home)

This presentation will show:

$ command
output line 1
output line 2
...
output line N
Eliminating the Web Browser

$ wget https://libreplanet.org/2019/speakers/

$ wget -O speakers.html https://libreplanet.org/2019/speakers/

Mike Gerwitz  
Computational Symbiosis  
LibrePlanet 2019
Eliminating the Web Browser

$ wget https://libreplanet.org/2019/speakers/

--2019-03-24 00:00:00-- https://libreplanet.org/2019/speakers/
Resolving libreplanet.org (libreplanet.org)... 209.51.188.248
Connecting to libreplanet.org (libreplanet.org)|209.51.188.248
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: ‘index.html’

2019-03-24 00:00:00 (1.78 MB/s) - ‘index.html’ saved [67789]
Eliminating the Web Browser

$ wget https://libreplanet.org/2019/speakers/

--2019-03-24 00:00:00-- https://libreplanet.org/2019/speakers/
Resolving libreplanet.org (libreplanet.org)... 209.51.188.248
Connecting to libreplanet.org (libreplanet.org)|209.51.188.248|
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: ‘index.html’
...
2019-03-24 00:00:00 (1.78 MB/s) - ‘index.html’ saved [67789]

$ wget -O speakers.html \ https://libreplanet.org/2019/speakers/
Browser vs. wget Comparison

Ctrl+L ‘‘https://libreplanet.org/2019/speakers/’’

$ wget https://libreplanet.org/2019/speakers/
Finding Text on the Command Line

$ grep 'free software' speakers.html
$ grep 'free software' speakers.html

<p>Mike Gerwitz is a free software hacker and activist with a focus on exclusively free software. Mike spends most of his free time working on free software projects and advocating for the importance of free software in the open source community.

</p>
A More Gentle Reply

$ grep --quiet 'free software' speakers.html && echo yes

yes
$ grep --quiet 'free software' speakers.html && echo yes

yes

$ echo 'Hello, world!'

Hello, world!
$ grep --quiet 'free software' speakers.html && echo yes

yes

$ grep --quiet 'open source' speakers.html || echo no

no
**Writing to Files (Redirection)**

- Commands write to standard out (stdout) by default
- *Output redirection* writes somewhere else

```
# overwrites each time
$ echo 'Hello, world!' > hello.txt
$ echo 'Hello again, world!' > hello.txt

# appends (echo adds a newline)
$ echo 'First line' >> results.txt
$ echo 'Second line' >> results.txt
```
$ wget --quiet -O speakers.html \
https://libreplanet.org/2019/speakers/ \
&& grep --quiet 'free software' speakers.html \
|| echo https://libreplanet.org/2019/speakers/ \
>> results.txt
Mike Gerwitz
Computational Symbiosis
LibrePlanet 2019
$ URL=https://libreplanet.org/2019/speakers/
$ wget -q0 speakers.html \ 
  "$URL" \ 
  && grep -q 'free software' speakers.html \ 
  || echo "$URL" \ 
  >> results.txt
$ URL=https://libreplanet.org/2019/speakers/
$ wget -q0 - \\n  "$URL" \\n  | grep -q 'free software' \\n  || echo "$URL" \\n  >> results.txt
$ URL=https://libreplanet.org/2019/speakers/
$ wget -q0 - "$URL" \ 
   | grep -q 'free software' || echo "$URL" >> results.txt
$ alias fetch-url='wget -qO-'

$ URL=https://libreplanet.org/2019/speakers/
$ fetch-url "$URL" \
  | grep -q 'free software' || echo "$URL" >> results.txt
Text is a Universal Interface
“Expect the output of every program to become the input to another”

—Doug McIlroy (1978)
The Unix Philosophy

This is the Unix philosophy: Write programs that do one thing and do it well. Write programs to work together. Write programs to handle text streams, because that is a universal interface.

—Doug McIlroy
Bdale Garbee
Closing keynote

Tarek Loubani
Opening keynote (Day 1)
<p><em>Closing keynote</em></p>
<p><em>Opening keynote (Day 1)</em></p>
<p><em>Opening keynote (Day 2)</em></p>

[...]

<p><em>The Tor Project: State of the Onion</em> and <em>Library Freedom Institute: A new hope</em></p>

[...]

<p><em>Large-scale collaboration with free software</em></p>
$ fetch-url https://libreplanet.org/2019/speakers/ \ 
  | grep -A5 speaker-header \ 
  | grep -o 'em>[^<]+</em>'

<em>Closing keynote</em>
<em>Opening keynote (Day 1)</em>
<em>Opening keynote (Day 2)</em>

[...]
<em>The Tor Project: State of the Onion</em>
<em>Library Freedom Institute: A new hope</em>
<em>The Tor Project: State of the Onion</em>

[...]
<em>Large-scale collaboration with free software</em>
<em>Large-scale collaboration with free software</em>
$ fetch-url https://libreplanet.org/2019/speakers/ \\ 
| grep -A5 speaker-header \\ 
| grep -o ’<em>[~<]\+</em>’ \\ 
| sort \\ 
| uniq -cd

2 <em>Hackerspace Rancho Electrónico</em>
4 <em>Large-scale collaboration with free software</em>
2 <em>Library Freedom Institute: A new hope</em>
2 <em>Right to Repair and the DMCA</em>
2 <em>Teaching privacy and security via free software</em>
2 <em>The joy of bug reporting</em>
5 <em>The Tor Project: State of the Onion</em>
$ fetch-url https://libreplanet.org/2019/speakers/ \
   | grep -A5 speaker-header \
   | grep -o '<em>[~]<\\]+</em>' \
   | sort \
   | uniq -cd \
   | sort -nr \
   | head -n5

5 <em>The Tor Project: State of the Onion</em>
4 <em>Large-scale collaboration with free software</em>
2 <em>The joy of bug reporting</em>
2 <em>Teaching privacy and security via free software</em>
2 <em>Right to Repair and the DMCA</em>
$ fetch-url https://libreplanet.org/2019/speakers/ \ 
  | grep -A5 speaker-header \ 
  | grep -o ’<em>[^<]\\+</em>’ \ 
  | sort \ 
  | uniq -cd \ 
  | sort -nr \ 
  | head -n5 \ 
  | sed ’s#^ *\(.*\) <em>\(.*\)</em>#\2 has \1 speakers#’

The Tor Project: State of the Onion has 5 speakers
Large-scale collaboration with free software has 4 speakers
The joy of bug reporting has 2 speakers
Teaching privacy and security via free software has 2 speakers
Right to Repair and the DMCA has 2 speakers
$ fetch-url https://libreplanet.org/2019/speakers/ \
  | grep -A5 speaker-header \
  | grep -o ’<em>[~<]\+</em>’ \
  | sort \
  | uniq -cd \
  | sort -nr \
  | head -n5 \
  | sed ’s/^ *\(.*\) <em>\(.*\)</em>#\2 has \1 speakers#’ \
  | espeak
Incremental Development

Interactive REPL, Iterative Decomposition
$ grep -o 'https\?://[^\s]+' email-of-links.txt
https://en.wikipedia.org/wiki/Free_software
https://en.wikipedia.org/wiki/Microsoft
https://opensource.org/about
$ grep -o 'https\?://[^ ]\+/' email-of-links.txt \
| while read URL; do
    echo "URL is $URL"
  done
URL is https://en.wikipedia.org/wiki/Free_software
URL is https://en.wikipedia.org/wiki/Open_source
URL is https://en.wikipedia.org/wiki/Microsoft
URL is https://opensource.org/about
$ grep -o 'https\?://[^ ]\+/' email-of-links.txt \
   | while read URL; do \
       fetch-url "$URL" | grep -q 'free software' \
       || echo "$URL \
   done \
> results.txt
$ grep -o 'https\?://[^ ]\+/' email-of-links.txt \
  | while read URL; do \
    fetch-url "$URL" | grep -q 'free software' \
    || echo "$URL" \
  done \
  | tee results.txt

https://en.wikipedia.org/wiki/Microsoft
https://opensource.org/about
$ grep -o 'https\?://[^ ]\+/' email-of-links.txt \
  | while read URL; do \
    fetch-url "$URL" | grep -q 'free software' \
    || echo "$URL" \
    done \
  | tee results.txt \
  | xclip -i -selection clipboard
$ grep -o 'https\?://[^\s]+' email-of-links.txt \
  | while read URL; do \
    fetch-url "$URL" | grep -q 'free software' \
    || echo "$URL" \
  done \
  | tee >( xclip -i -selection clipboard )

https://en.wikipedia.org/wiki/Microsoft
https://opensource.org/about
$ xclip -o -selection clipboard \
  | grep -o 'https\?://[^ ]+\+' \
  | while read URL; do \
    fetch-url "$URL" | grep -q 'free software' \n    || echo "$URL" \n  done \
  | tee >( xclip -i -selection clipboard )

https://en.wikipedia.org/wiki/Microsoft
https://opensource.org/about
$ xclip -o -selection clipboard \
  | grep -o 'https\?://[~ ]\+\)' \
  | while read URL; do \
    fetch-url "$URL" | grep -q 'free software' \
    || echo "$URL" \
  done \
  | tee results.txt
https://en.wikipedia.org/wiki/Microsoft
https://opensource.org/about

$ xclip -i -selection clipboard < results.txt
Go Grab a Coffee
$ sleep 1 && echo done & echo start
start
done
```bash
$ sleep 1 && echo done & echo start
start
done

(Don’t do this for large numbers of URLs!)

$ while read URL; do
defetch-url "$URL" | grep -q ’free software’ \n   || echo "$URL" &
done | tee results.txt
```
url-grep
#!/bin/bash

search=$1
url=$2

wget -qO- "$url" \
  | grep -q "$search" || echo "$url"

$ chmod +x url-grep

$ while read URL; do
  ./url-grep 'free software' "$URL" >> results.txt
done
url-grep
#!/bin/bash

search=$1
url=$2

wget -qO- 
    "$url" \
    | grep -q "$search" || echo "$url"

$ chmod +x url-grep

$ xargs -n1 ./url-grep 'free software' > results.txt
url-grep
#!/bin/bash

search=$1
url=$2

wget -qO- "$url" \
  | grep -q "$search" || echo "$url"

$ chmod +x url-grep

$ xargs -n1 -P5 ./url-grep 'free software' > results.txt

# ~ 5 concurrent processes
$ wc -l url-list
1000

$ time xargs -n1 -P10 ./url-grep 'free software' < url-list
real    0m17.548s
user    0m8.283s
sys     0m4.877s
$ for img in *.png; do
    convert "$img" -resize 50% "sm-$img"
done

# nested directories
$ find . -name '*.png' -exec convert {} -resize 50% sm-{} \;
Password Generation

# generate a random 32-character password
$ tr -cd \[:graph:\] \< /dev/urandom | head -c32
‘TB~cmJQ1%S8&tJ,%FoD54}"Fm4}\Iwi

# generate passphrase from EFF’s large dice wordlist
# (https://www.eff.org/dice)
$ cut -f2 eff_large_wordlist.txt \  
   | sort -R --random-source=/dev/urandom \  
   | head -n6 \  
   | tr ’\n’ ’’
oppressor roman jigsaw unhappy thinning grievance
$ gpg --decrypt password-db.gpg | head -n3
https://foo.com
user mikegerwitz
pass !({:pT6DcJG.cr&OAc0_EC63r_*xg|uD
Password Manager

$ gpg --decrypt password-db.gpg | head -n3
https://foo.com
user mikegerwitz
pass !({:pT6DcJG.cr&OAco_EC63r_*xg|uD

$ gpg --decrypt password-db.gpg \ 
  | grep -A2 ^https://foo.com \ 
  | tail -n2 \ 
  | while read key value; do 
    echo "paste $key..."
    printf %s "$value" \ 
      | xclip -o -selection clipboard -l 1 -quiet
  done
Remote Password Manager With 2FA

- Add `extra-socket` to `.gnupg/gpg-agent.conf`
- Add `RemoteForward` in `.ssh/config` for host
- Save script on previous slide as `get-passwd`

```
$ ssh -Y mikegerwitz-pc get-passwd https://foo.com
```
Taking Screenshots

# draw region on screen, output to ss.png
$ import ss.png

# screenshot of entire screen after 5 seconds
$ import -pause 5 -window root ss.png

# screenshot to clipboard
import png:- | xclip -i -selection clipboard -t image/png
$ import png:- | tesseract -psm 3 - - | cowsay

/ Keynote Speakers \
| i | i |
| Bdale Garbee Micky Metts Richard | Stallman Tarek Loubani |

\ ^__^ \\
\ (oo)\_______
(____)
\
|\-----w |
# OCR at URL
$ wget -qO- https://.../keynote_banner.png \  
  | tesseract -psm 3 - -

# perform OCR on selected region and speak it
$ import png:- | tesseract -psm 3 - - | espeak

# perform OCR on clipboard image and show in dialog
$ xclip -o -selection clipboard -t image/png \  
  | tesseract -psm 3 - - \  
  | xargs -0 zenity --info --text
$ xdotool search --name ' GNU IceCat$' windowactivate --sync \n    windowsize 1024 600 \n    key ctrl+t \n    && while read url; do 
        xdotool getactivewindow \n            key ctrl+l type "$url" \n        && xdotool getactivewindow \n            key Return sleep 5 \n        && xdotool getactivewindow \n            key ctrl+f sleep 0.5 \n        && import -window "$x( xdotool getactivewindow )" \n            -crop 125x20+570+570 png:- \n            | tesseract -psm 3 - - \n            | grep -q ^Phrase && echo "$url" \n    done < url-list | tee results.txt
All GNU packages have Info manuals and --help.

Most programs (including GNU) have manpages.

$ grep --help  # usage information for grep
$ man grep     # manpage for grep
$ info grep    # full manual for grep

# bash help
$ help
$ man bash
Mike Gerwitz
mtg@gnu.org

Slides and Source Code Available Online
<https://mikegerwitz.com/talks/cs4m.pdf>

Licensed under the Creative Commons Attribution ShareAlike 4.0 International License