The User in the Loop

Extensibility, Guile, and GNU

GNU Hackers Meeting 2011

Andy Wingo

Greetings!

Andy Wingo

Guile co-maintainer, along with Ludovic Courtès

Goal

Understand the problem

Guile is part of the solution

Agenda

Extensibility: Macro perspectives

E11y for users

E11y for GNU

Getting there

Why Are We Here?

To advance software freedom?

Making the Right Thing

The MIT approach vs. ‘worse is better’

Emacs Makes A Computational Space?

A common thread: Richard Gabriel

* Patterns of Software

The Quality Without a Name

Thesis: Some places just feel right

Christoper Alexander: This feeling is ‘living structure’

Architectural patterns help produce that feeling

What about software?

Architect vs World

Some patterns from Alexander's ‘A Pattern Language’:

* Your own home
* Site repair
* Self-governing workshops

Is possible for our programs to make our users whole, to satisfy their needs, without the possibility of of modification, of change, of extension?

More Than Nice-to-Have

E11y is fundamental to human agency and happiness

Moglen: ‘Software is the steel of the 21st century’

Incremental change, qualitative change

{Counter,}example: GNOME 3.0

Building Materials

Le Corbusier's concrete; GNU's C

Gabriel: ‘The good news is that in 1995 we will have a good operating system and programming language; the bad news is that they will be Unix and C++.’

Guile

‘This is a song about Alice. Remember Alice?’

Guile: Practical e11y for GNU

3 models:

* Embedding (within a program)
* Extending (outside a program)
* Scripting (the space between programs)

Embedding

App links to lib$lang

App provides extension points: functions and data types, mostly

Guile Facilitates Embedding

Libguile: A lovely C interface

Widely deployed

Forgiving of mistakes

* Garbage collection
* Functional programming

Some tool support to help create correct extensions

No static typing though

Embedding Tips

Think about user experience

Debugging: usually printf-based

Provide a console if you can

Documentation is essential

I started this way

Embedding Is Inferior

Embedded extensions compose poorly

Embedding privileges C over $lang

The developer experience is poor

Tools support is poor

Beyond Embedding: Extension

Turn your app into a library

Bind that library in $lang

The GNU Extension Language

Guile is a fantastic extension language.

Practical Extensibility

Guile over C(oncrete):

* Language safety
* Source code availability
* Online compiler
* Online debugger
* Online apropos / meta-dot
* Multi-paradigm: Functional, OOP, message-passing, ...

Easy e11y == more extensions

RDD

REPL-driven development

```
guile> (import (my-app))
```

Put the user in the [read-eval-print] loop

REPL provides discoverability and hackability

Guile: A Practical Scheme

Draw with Cairo, make UIs with GTK+ or Clutter, fetch and serve HTTP with the built-in web modules, parse and generate XML with SXML, access RPC services with Guile-RPC, do so securely with GnuTLS, access databases with Guile-DBI, process mail with mailutils, build distributed job servers and workers with ZeroMQ, make new bindings with the dynamic FFI, extend your way to a new language with modules and macros, write powerful servers with the POSIX bindings and native pthreads, load your extensions quickly with the bytecode VM, write your own control constructs with delimited continuations, do all this from Emacs without

Distributed Extensibility

Where are the programs we are extending?

*   on traditional computers
*   on devices
*   on the web

Practical extensibility must reach these all platforms

Remote development and debugging in Emacs: Geiser

Web hacking: Guile's web modules

Extensions and GNU

(Exhale.)

(Inhale.)

Extensions and GNU

E11y: for whom? Just for the the user-programmer, or to also share?

E11y begins with the user-programmer

GNU can facilitate sharing; sharing can build GNU

Building the Guild Hall

A CPAN for Guile

Port of Debian's APT

Use it as an extensions archive for your program

Your extensions are guild packages

Extensions Are Gateway Drugs

Users are already sold on ideology, now hook them in with practical programming environments

The GPL is all we need; no copyright assignment!

But What About Scripting?

Scripting: The space between applications

All about protocols

Canonical example: SH and POSIX: Pipes between processes

Challenges & Opportunities

Scripting languages tend to not be good general-purpose languages (Tcl)

Platforms change, scripting languages (often) don't

Decline of Tcl, Perl

What is the current ‘space between programs’? What will it be in the future?

Guile: an adaptable general-purpose language

* Modules, macros, syntax, semantics

Sites of the Future

In the future things will be distributed

We have a chance to make it

Guile can play an important part

Lots of tinkering, engineering, and science to do

Hacker vs User?

‘This monument must be preserved against the passing of time.’

--Ricardo Bofill, architect of Barcelona's new airport terminal

The User in the Loop

E11y builds the living GNU project -- not the monument.

Strengthening programs (centers) while organically knitting cohesive, living common spaces.

Final words

Christopher Alexander speaking to computer scientists at OOPSLA 1996

fin.

* http://gnu.org/s/guile/
* http://wingolog.org/