Builder

Improving GNOME's Developer Story

Christian Hergert
<christian@hergert.me>

http://hergert.me/talks/scale13x-builder.pdf

Scale 13x
February 2015

GNOME

What's GNOME?

A Free Software Desktop
Focused on UX

GNOME

We tend to expose bugs in the stack

And then we go fix them

Graphics Drivers and Display Servers
Network Configuration
Audio Subsystems
Message Passing and RPCs
Settings
Race-free Application Launching
Init Systems
Application Sandboxing

Who am I?

Long time Free Software hacker

Glib/GObject/Gtk+ @ GNOME
Databases @ MongoDB
Virtual Machines @ VMware
JIT/GC/Language Runtime Geek @ Mono

I quit my job to focus on Free Software.

In particular, GNOME

And even more specifically, Builder

Builder is my attempt to improve the developer experience on GNU/Linux

So what is this GNOME thing anyway?

GNOME is an ecosystem

Compositor, graphics toolkits, build tools, settings, media pipelines, music players, calendars, file manager, VFS, email systems, maps, locations, input methods, network managers, battery monitors, display configuration, printers, profilers, touchpads, tablets, content sharing, privacy tools, desktop search, hardware integration, color management, key management, phones, tablets, kiosks, etc...

...there is a lot involved in making a modern usable desktop

But every year the workload grows.

In 2012, I made a **bold** statement.

Email me, and I'll teach you **C**.

Um, my inbox is still full.

We literally have thousands of people that want to contribute to Free Software but don't know how.

And that is for **C**!

Imagine if it was something like **Python** or **JavaScript** (or **Vala**).

Keep in mind, I don't think everyone should learn C. We just have a lot of it in GNOME and we are better off if more people can help maintain that code.

So I tried to teach people C.

…and failed pretty miserably.

We were lacking the proper tools.

And...

Learning how to get started is the number one hurdle in contributing to Free Software.

Let me repeat that for you

Learning how to get started is the number one hurdle in contributing to Free Software.

This is why I'm building Builder.

We need a developer story that is as simple as enabling a “Developer Mode” switch in GNOME's Control Center.

There exists other IDEs.
And some of them even work.

But none of them are focused on making it easy to contribute to GNOME.

We need better tools that will help guide newcomers through the process.

If learning autotools is a pre-requisite, we are not in particularly great shape.

P.S. I like autotools

So what is the plan for Builder?

We are pre-pre-alpha, but we have a code-base, and we make forward progress almost every day (except when I'm giving talks)

Around 50 contributors thus far

1 fulltime engineer

I paid for the first 4 months of development.
Community has crowdfunded the next half year.

We are building a modern,
fully functional IDE for GNOME.

That includes build systems, version control, project management, editors (vim/emacs too), file settings, modelines, auto completion, auto indentation, integrated help, debuggers, UI designer, D-Bus, external hardware devices (tablets, dev boards, etc), code search, refactory, IDE scripting, live diagnostics, tutorials, settings, embedded resources, simulators, profilers, etc...

With the same attention to detail
that goes into UX.

Developers are people too, and they deserve a great developer experience.

DX

GNOME 3.16 is about 5 weeks away.
So don't expect the magic by then.

GNOME 3.18 is a more realistic goal.

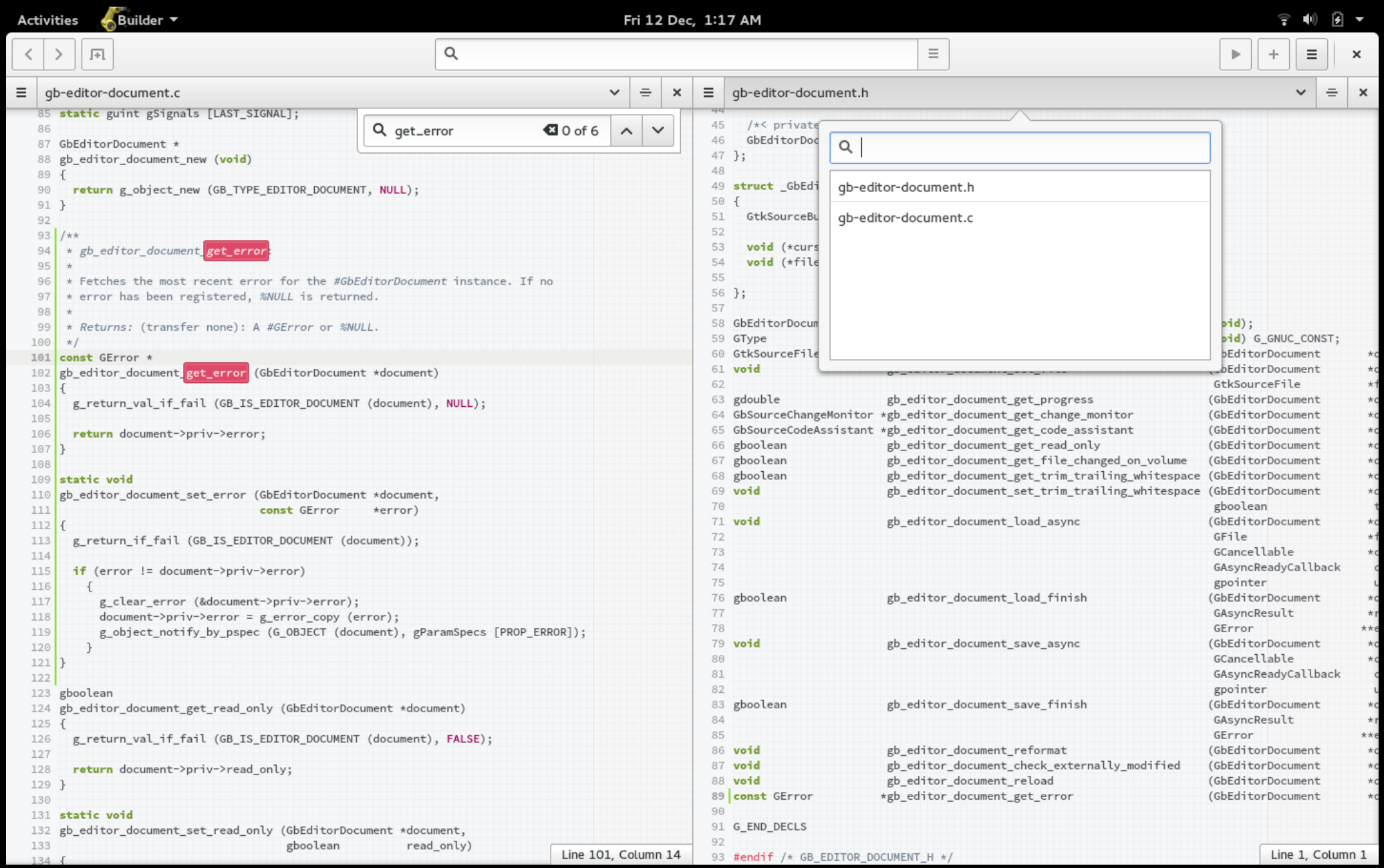
So what exists today?

Builder is broken into two pieces.

Builder the UI, and LibIDE the shared library.

The UI chrome is practice in minimalism

Code is important
Toolbar buttons are not


Activities
Builder
Fri 12 Dec, 1:17 AM
gb-editor-document.c
gb-editor-document.h
get_error
0 of 6
gb-editor-document.h
gb-editor-document.c
Line 101, Column 14
Line 1, Column 1

Lots to do.

We need to integrate nemiver, gitg, devhelp, glade, project management, etc.

And “LibIDE” is under heavy development.

Okay then, what does LibIDE do?

Semantic Syntax Highlighting
Auto Indentation
Auto Completion
Symbol Resolution
Code Navigation
Code Search
Debugger
Device Management (tablets, odroids, etc)
D&D (Deploy and Debug)
Live Diagnostics and Error reporting
Project Management
Build Systems
Version Control
Buffer Management
Editor Settings (modelines, editorconfig, etc)
Cross-Compiling

And IDE Scripting.

Even though LibIDE is written in C, you can inject scripts using JavaScript or Python.

(Way better than .vimrc)

For example, to translate a buffer on save...

```
/* “Context” is a special variable provided to you */
let BufferManager = Context.get_buffer_manager();

/* upon save request, translate newlines */
BufferManager.connect('save-buffer', function(buffer){
  let text = buffer.get_text().replace('\r\n', '\n');
  buffer.set_text(text);
});
```

```
let DeviceManager = Context.get_device_manager();
let BuildSystem = Context.get_build_system();

/* auto-build current project when a new device is connected */
DeviceManager.connect('device-added', function(provider, device){
  if (device.get_id() != 'local') {
    let builder = BuildSystem.get_builder(null, device);
    builder.build_async(null, null);
  }
});
```

You can implement editor hooks, search providers, build extensions, device bring-up, refactory tools, etc.

So much of what goes into an IDE isn't visible.
If we do our jobs right, you won't even notice.

Questions?

Fin

https://wiki.gnome.org/Apps/Builder