0001-workshop.patch

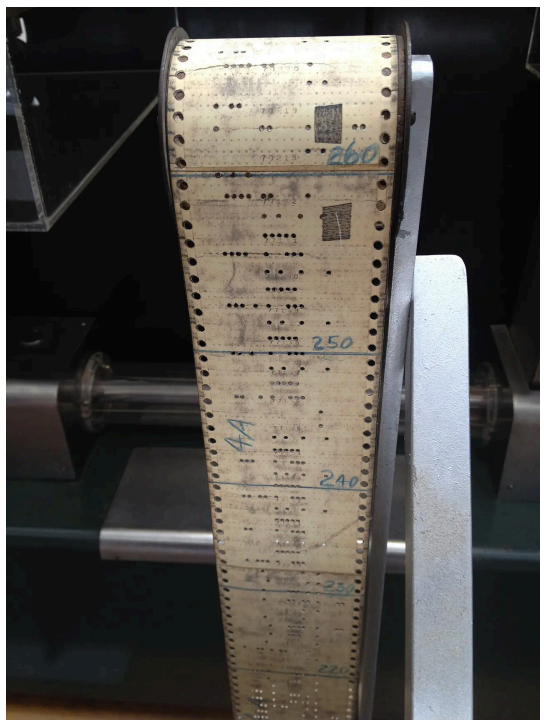
Using git the “good old way”!

Hugo ARNAL

Some history

Apparently, patches have existed since punch cards!

You would cut the right part and replace it with the patched one!



Some history

Patches always existed in two forms:

- Source code patching
- Binary patching

We’re going to be only talking about source code patching.

Some ~~git~~ history

How modifications to the Linux kernel work:

- Create modifications to the kernel
- Generate the patch using `diff(1)` and `patch(1)`
- Send the .patch files to maintainers via email

The Linux kernel was using BitKeeper before git which simplified this flow.

Some ~~git~~ history

Problems with BitKeeper:

- Slow (most CVS took >30s to apply patches)
- Proprietary license

This made Linus create git once BitKeeper revoked the Kernel’s license.

Example of a patch

Left: original source code, right: patch to apply

```c
#include <stdio.h>

int main(void)
{
  printf("Hello world!\n");
}
```

```
--- a.c 2026-05-10 16:47:17.250776740 +0200
+++ b.c 2026-05-10 16:47:31.502421173 +0200
@@ -2,5 +2,5 @@

 int main(void)
 {
-    printf("Hello world!\n");
+    puts("This is an example");
 }
```

Why learn patches today?

Most ~~Linux distros~~ Operating Systems use patches. If you're running an Arch Linux system, some packages you install are patched to work on your system.

clang requires 3 patches to work (as of 10th May 2026):

- 0001-Revert-clang-driver-When-fveclib-ArmPL-flag-is-in-us.patch
- 0002-Reapply-CUDA-HIP-Add-a-__device__-version-of-std-__g.patch
- enable-fstack-protector-strong-by-default.patch

rustc requires 8 patches to work (as of 10th May 2026):

- 0001-bootstrap-Change-libexec-dir.patch
- 0002-bootstrap-Change-bash-completion-dir.patch
- 0003-bootstrap-Workaround-for-system-stage0.patch
- 0004-compiler-Change-LLVM-targets.patch
- 0005-compiler-Use-ld.lld-by-default.patch
- 0006-compiler-Use-target-specific-GCC-linkers.patch
- 0007-compiler-Link-riscv64-musl-statically.patch
- 0008-compiler-Swap-primary-and-secondary-lib-dirs.patch

Why learn patches today?

You’re probably using patched software without even knowing!

Most of the newer web browsers (Zen, Helium, Arc…) are patching Firefox or Chromium directly! Using software to automate their patches such as GNU Quilt.

Why learn patches today?

A dependency you’re using isn’t working correctly? Patch it!

Lots of projects are patching dependencies to assure either a better state or a personalized version of it.

Why learn patches today?

Some git forges only work on patches.

This is the case of Sourcehut which doesn’t have the Pull Request system but works via Email Patches and a new interface.

Patches vs forking

Forking is taking a repository, making a copy of it then modifying the source code.

Why would I use patches instead of just forking?

Overall, forking is heavy and can make it harder to follow upstream commits.

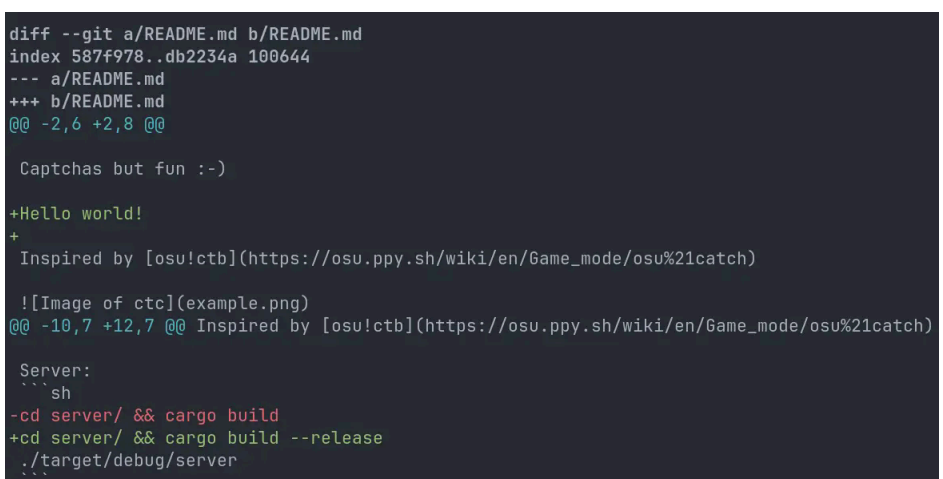
I recommand this blog post:

https://nesbitt.io/2026/05/01/patching-and-forking-in-package-managers.html

git diff

Simplest version of a git commit patch

- Make some changes to your repo, don’t push them



```
diff --git a/README.md b/README.md
index 587f978..db2234a 100644
--- a/README.md
+++ b/README.md
@@ -2,6 +2,8 @@

 Captchas but fun :-)

+Hello world!
+
 Inspired by [osu!ctb](https://osu.ppy.sh/wiki/en/Game_mode/osu%21catch)

 ![Image of ctc](example.png)
@@ -10,7 +12,7 @@ Inspired by [osu!ctb](https://osu.ppy.sh/wiki/en/Game_mode/osu%21catch)

 Server:
 ```sh
-cd server/ && cargo build
+cd server/ && cargo build --release
 ./target/debug/server
 ```
```

git format-patch

How do I create patches of a git project?

- Create a new branch
- Create your commits changing what you need
- Use `git format-patch <original branch>`

This will create `000?-*.patch` files

git apply

Applies the patches of a given `.patch` file (but does not commit it)

```
git apply 0001-interesting.patch
```

git am

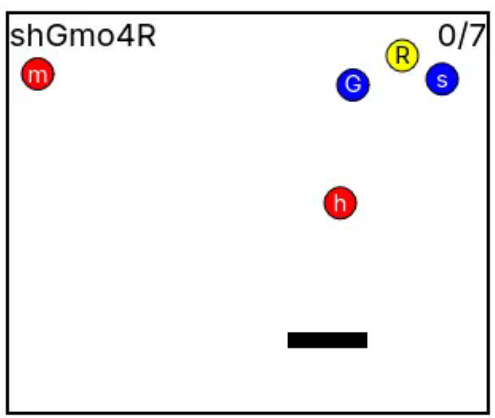
- Description: Apply a series of patches from a mailbox

Applies the patches of a given `.patch` file **AND** applies it.

```
git am 0001-interesting.patch
```

Workshop

ctc: complete a captcha by getting all the letters dropping from the top of the screen.



Composed of:

- a very simple frontend in HTML/JS
- Rust backend

- git clone https://github.com/hugoarnal/ctc.git

Workshop

Goals:

- Create a new branch and checkout to it
- Create a commit hard coding the WIDTH and HEIGHT values to 300 and 250. (JS)
- Remove all colors **except** red (JS)
- Make Captcha length 4 instead of 7 (Rust & JS)
- Unhard code the WIDTH & HEIGHT variables **(must use a rebase method)**
- Apply the given patch `(0001-ref-separate-Fruit-Pad-from-script.js.patch)` with the original commit

Sources

- https://en.wikipedia.org/wiki/Git
- https://en.wikipedia.org/wiki/Patch_(computing)
- https://fr.wikipedia.org/wiki/Fichier:Bitkeeper-logo.png
- https://docs.kernel.org/process/submitting-patches.html
- https://github.com/imputnet/helium
- https://github.com/zen-browser/desktop
- https://sourcehut.org

Slides

https://github.com/hugoarnal/talks