

Scripting Git

CB Bailey

 @hashpling

Bloomberg

Saturday, 14th April 2018

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

CB Bailey

 @hashpling

they/them or he/him

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Where would you extend Git?

- Helper scripts
- Hooks
- Bisecting
- Automated updates to a repository

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Why script?

- No “first-tier” API for Git
- A relatively stable interface
- Low barrier to working results

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Know your plumbing

- The most stable Git commands
- Listed in `git help git`
- But sometimes it's not that simple

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

porcelain:

```
git reset --soft <commit id>
```

plumbing:

```
git update-ref HEAD <commit id>
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

porcelain:

```
git reset <commit id> .
```

plumbing:

```
git read-tree -m <commit id>
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

porcelain:

```
git branch
```

plumbing:

```
git symbolic-ref -q HEAD
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Check your exit codes

Most Git commands:

- Exit with status 0 on success
- Exit with non-zero status (usually 128) on failure

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Delimiting with NUL

- Many commands use `-z` (sometimes `--nul`)
- Uses ASCII NUL instead of ASCII LF as delimiter
- May also turn off escaping and quoting

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Consider concurrency and sharing

- Do you need the working tree?
- Or just a working tree? (`git worktree`)
- Can you use a temporary index?
(`GIT_INDEX_FILE`)
- What parallel operations do you support?

Use “atomic” operations where possible

- `git push --force-with-lease`
- `git update-ref <ref> <newval>
<oldvalue>`

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

A quick sample: `git ff`

- Fast-forward a branch that isn't checked out

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

The shell function I always write

```
die () {  
    echo >&2 "$*"   
    exit 1  
}
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Do we have both parameters?

```
if [[ $# -ne 2 ]]
then
    die "Syntax: $0 <branch-name> <target-commit-id>"
fi
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Is it a branch?

```
current=$(git rev-parse --verify --quiet "refs/heads/$1") ||  
die "$1 is not a valid branch name"
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Is it the current branch?

```
if [[ "$(git symbolic-ref --quiet HEAD)" = "refs/heads/$1" ]]
then
    die "$1 is currently checked out, use\
'git merge --ff-only' to fast-forward"
fi
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Can this be resolved as a commit?

```
target=$(git rev-parse --verify --quiet "$2^{ }") ||  
die "Could not resolve $2 as a commit"
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Is this really a fast-forward?

```
git merge-base --is-ancestor "$current" "$target" ||  
die "Cannot fast forward, $1 is not an ancestor of $2"
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Lets do it!

```
git update-ref -m "Fast-forwarded by script to $2" \  
"refs/heads/$1" "$target" "$current"
```

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

`git bisect`

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

A brief primer on `git bisect`

- You identify one “bad” commit
- ... and one or more “good” commmits
- Test commmits as prompted
- Use `git bisect run`

Considerations for automation

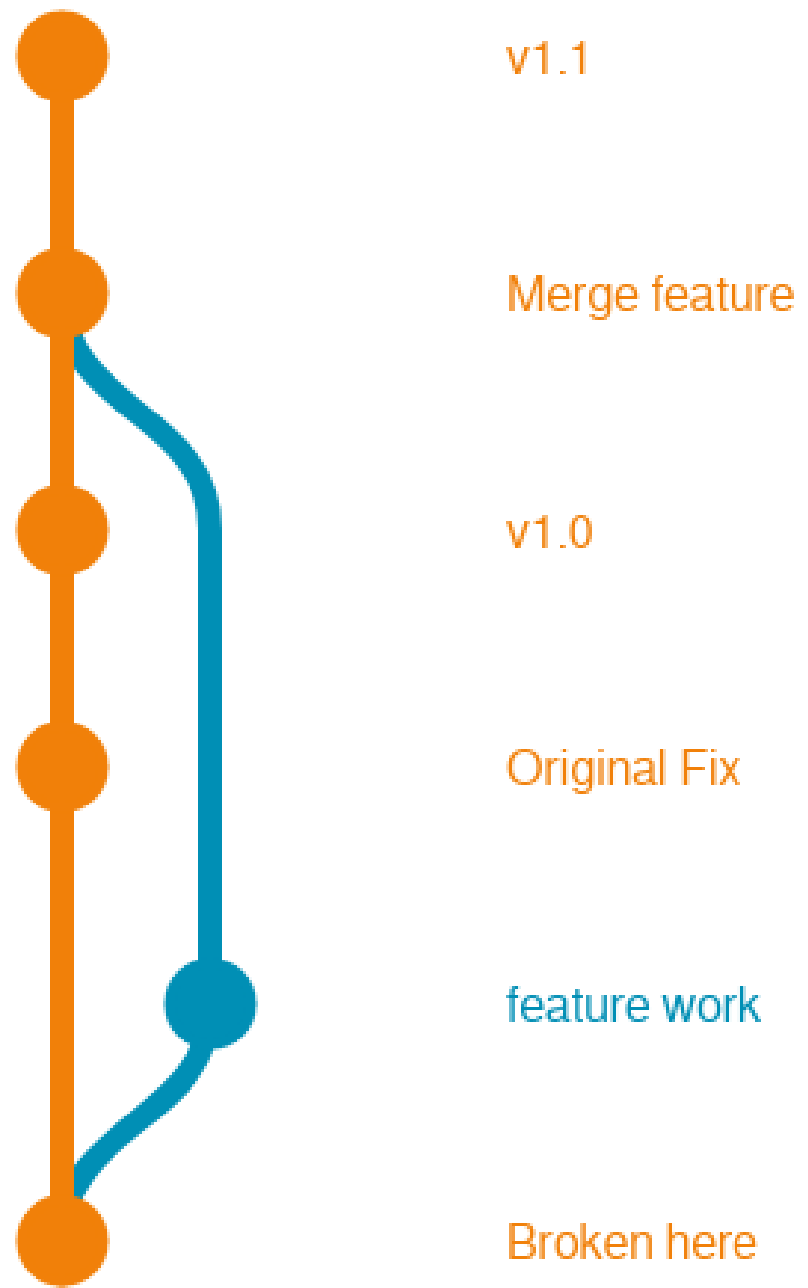
- You need to be sure about your test
- Your test should be as fast as possible

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Testing for a regression

- You fixed a bug...
- ...but now it's back. What went wrong?



Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Things to consider scripting

- Patching around known issues
- Use exit code 125 to signal 'skip'
- Marking good or bad based on commit position

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Scripting merges

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Never script `git merge`

- ...for automated updates
- ...or be careful if you do

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

git merge deconstructed

1. Populate the index with three trees
2. Collapse "easy" cases into stage 0
3. Perform file level merges on the remaining entries

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Merging into the index

```
git read-tree -m
```

Other useful options:

- `--trivial / --aggressive`
- `-u` update the working tree
- `-i` don't check the working tree

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

File level merges

- `git merge-index`
- `git merge-one-file`

`git merge -s resolve`

`recursive` is more sophisticated.

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.

Key points

- Follow best practices for your language
- Choose the most suitable Git commands for your task
- Understand the whole context for your scripts

Tech at Bloomberg

Explore our repo: github.com/bloomberg

Check out our blog: TechAtBloomberg.com

We're hiring: bloomberg.com/careers/technology

Bloomberg

© 2018 Bloomberg Finance L.P.
All rights reserved.