# Package: saperlipopette (via r-universe)

March 14, 2025

**Title** Create Example Git Messes

**Version** 0.0.0.9000

**Description** Holds functions creating Git messes, that users would then solve, to follow <https://ohshitgit.com/>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2.9000

**Imports** brio, cli, fs, gert, parsedate, purrr, rlang, usethis, withr

**URL** https://docs.ropensci.org/saperlipopette/, https://github.com/ropensci-training/saperlipopette

**BugReports** https://github.com/ropensci-training/saperlipopette/issues

**Suggests** testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Config/pak/sysreqs** git make libssl-dev libx11-dev

**Repository** https://ropensci.r-universe.dev

**RemoteUrl** https://github.com/ropensci-training/saperlipopette

**RemoteRef** main

**RemoteSha** b23ac8d2093e12387b196f738f3242c18c706dc6

## Contents

---

`create_all_exercises` *Create all exercises folder at once*

---

## Description

But do not open them!

## Usage

```
create_all_exercises(parent_path)
```

## Arguments

| | |
|---|---|
| `parent_path` | Path where to create the exercise repo |

## Value

The parent path

---

`exo_bisect` *"Hey I'd like to find which commit introduced a bug!"*

---

## Description

I notice a bug in my codebase. I can see the bug was not there a bunch of commits ago. Beside doing regular debugging, I can find out which commit introduced the bug by using `git bisect`. See https://git-scm.com/docs/git-bisect and https://www.jimhester.com/post/2019-04-24-git-bisect/.

## Usage

```
exo_bisect(parent_path)
```

## Arguments

| | |
|---|---|
| `parent_path` | Path where to create the exercise repo |

### Value

The path to the new project

### Git commands

`git bisect`.

### Examples

```
parent_path <- withr::local_tempdir()
path <- exo_bisect(parent_path = parent_path)
```

---

## exo_clean_dir *"Hey I'd like to remove these untracked files I created to test stuff!"*

---

### Description

If debugging for instance created now useless untracked files and directories, there's no need to remove them "manually". The tool for that is `git clean`:

- `git clean -n` for a dry run;
- `git clean -f` to run it; Add `-d` to also remove directories. See https://git-scm.com/docs/git-clean.

### Usage

```
exo_clean_dir(parent_path)
```

### Arguments

| | |
|---|---|
| `parent_path` | Path where to create the exercise repo |

### Value

The path to the new project

### Git commands

`git clean`.

### Examples

```
parent_path <- withr::local_tempdir()
path <- exo_clean_dir(parent_path = parent_path)
```

---

`exo_committed_to_main` *"Oh shit, I accidentally committed something to main that should have been on a brand new branch!"*

---

## Description

To go with https://ohshitgit.com/#accidental-commit-master

## Usage

```
exo_committed_to_main(parent_path)
```

## Arguments

| | |
|---|---|
| `parent_path` | Path where to create the exercise repo |

## Value

The path to the new project

## Git commands

`git reset --hard`, `git branch`, `git checkout`

## Examples

```
parent_path <- withr::local_tempdir()
path <- exo_committed_to_main(parent_path = parent_path)
```

---

`exo_committed_to_wrong` *"Oh shit, I accidentally committed to the wrong branch!"*

---

## Description

To go with https://ohshitgit.com/#accidental-commit-wrong-branch

## Usage

```
exo_committed_to_wrong(parent_path)
```

## Arguments

| | |
|---|---|
| `parent_path` | Path where to create the exercise repo |

### Value

The path to the new project

### Git commands

`git cherry-pick`, `git reset`, `git checkout`

### Examples

```
parent_path <- withr::local_tempdir()
path <- exo_committed_to_wrong(parent_path = parent_path)
```

---

## exo_conflict *"Hey I'd like to see what merge conflicts look like!"*

---

### Description

I made some work in a feature branch and want to merge it. Meanwhile, the main branch advanced. Unfortunately someone touched the same file as I did. Now I need to fix a merge conflict!

See also https://happygitwithr.com/git-branches.html#dealing-with-conflicts.

### Usage

```
exo_conflict(parent_path)
```

### Arguments

| | |
|---|---|
| `parent_path` | Path where to create the exercise repo |

### Value

The path to the new project

### Git commands

`git merge`.

### Examples

```
parent_path <- withr::local_tempdir()
path <- exo_conflict(parent_path = parent_path)
```

## exo_latest_message *"Oh shit, I need to change the message on my last commit!"*

### Description

To go with https://ohshitgit.com/#change-last-commit-message

### Usage

```
exo_latest_message(parent_path)
```

### Arguments

| | |
|---|---|
| parent_path | Path where to create the exercise repo |

### Value

The path

### Git commands

```
git commit --amend
```

### Examples

```
parent_path <- withr::local_tempdir()
path <- exo_latest_message(parent_path = parent_path)
```

## exo_one_small_change *"Oh shit, I committed and immediately realized I need to make one small change!"*

### Description

To go with https://ohshitgit.com/#change-last-commit

### Usage

```
exo_one_small_change(parent_path)
```

### Arguments

| | |
|---|---|
| parent_path | Path where to create the exercise repo |

### Value

The path

### Git commands

```
git commit --amend --no-edit
```

### Examples

```
parent_path <- withr::local_tempdir()
path <- exo_one_small_change(parent_path = parent_path)
# Now add "thing 3" to the "bla" file
# And amend the latest commit
```

---

## exo_rebase_i *"Hey I'd like to make my commits in a branch look informative and smart!"*

---

### Description

I am working in a feature branch that's all my own. I made many small commits as I was figuring things out. Now I want the commits to tell a story for the PR reviewers, and not a story of how many stupid mistakes I made! The tool for that is `git base --interactive` also available as `git rebase -i`. Useful links:

- https://jvns.ca/blog/2023/11/06/rebasing-what-can-go-wrong-/
- https://wizardzines.com/comics/rules-for-rebasing/
- https://github.com/MikeMcQuaid/GitInPractice/blob/main/06-RewritingHistoryAndDisasterRecovery.adoc#rebase-commits-interactively-git-rebase-interactive
- https://github.blog/2022-06-30-write-better-commits-build-better-projects/

### Usage

```
exo_rebase_i(parent_path)
```

### Arguments

| | |
|---|---|
| parent_path | Path where to create the exercise repo |

### Value

The path to the new project

### Git commands

```
git rebase -i
```

### Examples

```
parent_path <- withr::local_tempdir()
path <- exo_rebase_i(parent_path = parent_path)
```

## exo_reset *"Hey I'd like to restart from scratch"*

### Description

I am working in a feature branch that's all my own. I made many small commits as I was figuring things out. Now I want the commits to tell a story for the PR reviewers, and not a story of how many stupid mistakes I made! Instead of `git base --interactive` also available as `git rebase -i`¡, I can also use `git reset --mixed` and then build the commits. Useful links:

- https://github.blog/2022-06-30-write-better-commits-build-better-projects/
- https://masalmon.eu/2024/06/11/rewrite-git-history/

### Usage

```
exo_reset(parent_path)
```

### Arguments

| | |
|---|---|
| parent_path | Path where to create the exercise repo |

### Value

The path to the new project

### Git commands

```
git reset --mixed
```

### Examples

```
parent_path <- withr::local_tempdir()
path <- exo_reset(parent_path = parent_path)
```

## exo_revert_file *"Oh shit, I need to undo my changes to a file!"*

### Description

To go with https://ohshitgit.com/#undo-a-file

### Usage

```
exo_revert_file(parent_path)
```

### Arguments

| | |
|---|---|
| parent_path | Path where to create the exercise repo |

### Value

The path to the new project

### Git commands

`git log`, `git checkout`, `git commit`

### Examples

```
parent_path <- withr::local_tempdir()
path <- exo_revert_file(parent_path = parent_path)
```

---

## exo_split_changes *"Hey I'd like to split these changes to the same file into several commits!"*

---

### Description

I made many edits to a file, in different places. This is too much for a commit, since small commits are best practice. I need to add the changes to Git bit by bit. The tool for that is `git add --patch`, also available as `git add -p`. If all your changes are presented to you as one chunk by `git add --patch`, choose the "s" option for splitting. See https://git-scm.com/book/en/v2/Git-Tools-Interactive-Staging#_staging_patches.

Note that patch is also an option for `git commit`, if you prefer so.

### Usage

```
exo_split_changes(parent_path)
```

### Arguments

| | |
|---|---|
| parent_path | Path where to create the exercise repo |

### Value

The path to the new project

### Git commands

`git add --patch`, `git add -p`.

## Examples

```
parent_path <- withr::local_tempdir()
path <- exo_split_changes(parent_path = parent_path)
```

---

`exo_time_machine` *"Oh shit, I did something terribly wrong, please tell me git has a magic time machine!?!"*

---

## Description

To go with https://ohshitgit.com/#magic-time-machine

## Usage

```
exo_time_machine(parent_path)
```

## Arguments

| | |
|---|---|
| `parent_path` | Path where to create the exercise repo |

## Value

The path to the new project

## Git commands

`git reset`, `git reflog`

## Examples

```
parent_path <- withr::local_tempdir()
path <- exo_time_machine(parent_path = parent_path)
```

---

`exo_undo_commit` *"Oh shit, I need to undo a commit from like 5 commits ago!"*

---

## Description

To go with https://ohshitgit.com/#undo-a-commit

## Usage

```
exo_undo_commit(parent_path)
```

## Arguments

| | |
|---|---|
| parent_path | Path where to create the exercise repo |

## Value

The path to the new project

## Git commands

`git log`, `git revert`

## Examples

```
parent_path <- withr::local_tempdir()
path <- exo_undo_commit(parent_path = parent_path)
```

# Index