BECOMING A PRO AT GIT

CHRIS KEATHLEY / @CHRISKEATHLEY / C@KEATHLEY.IO
I work with a distributed team.
I work with a distributed team.
“WE DON’T HAVE THE LUXURY OF A FOCUS ROOM”
EVERYTHING IS ASYNCHRONOUS
submitPR()
 .then(waitForApproval)
 .then(mergePR)
 .then(startDeployFromSlack)
This is a work in progress to add Selenium support and allows the existing test suite to be run against a local instance of GeckoDriver or ChromeDriver. This is definitely a work in progress, but will show us how the existing test suite runs against different webdriver backend.

I'd also be interested in your thoughts on how you'd like to see the driver modules designed. Should I be writing each one of the core interaction functions like visit, click, etc, in each module to make them easier to customize behavior per driver? Since the drivers differ, then per driver and element. Does this matter, should I have one driver per backend? If so, do you think each module should share the same functionality or should each module have its own implementations? If not, is it possible to share common functionality between driver? This is a major point for me.

I'm looking forward to hearing your ideas so I can build it right the first time.

I think for now we shouldn't try to share functionality between the different driver implementations. Once we have multiple implementations we can see exactly where the common code patterns are and we can use shared code at that point. It's easier to remove common code once you can see all of it.

For now it would be a good idea to share a common app between the drivers. We should create a behavior that each driver needs to conform to. If you want to work on that tonight then we can use that as a basis for the other drivers.
This looks great and I've marked it. Will you allow me to download and play with it tonight, then evaluate it in detail?

https://www.example.com/download.pdf

---

1. Open the PDF and review the content.
2. Identify any areas that need improvement or clarification.
3. Provide feedback on the overall quality and usefulness of the document.

---

Owner

I'm working on all these things and positions in our current document.

Reply...

---

Owner

I'll send the current version along with a draft version.

Reply...

---

Owner

I sent the revised document. Let me know if you have any comments.

Reply...
@keathley I looked into the webdriver page Strategist setting that you mentioned in your link, but I am still getting issues with tests to check the current page using the currentUrl function. I was thinking we could add a function like `IndexPage.loadAll()` that would find a unique element on the page to ensure the expected page has loaded. Maybe that's something that could be done as part of removing the deprecated `Browser.click_button` work.

Please let me know if there's anything else you'd like to see on this PR. I think it should be good to go after #209 is merged.

@aaronrenner added some commits 11 days ago
- Experimental Selenium driver
- Handle 204 response from webdriver api
- Fix integration tests for selenium
- Fix test failure on a small screen
- Fixes issues with click_button test
- Fixing integration test on Selenium 2
- Removed PageObjects to Pages
- Set default Session.server to none

@aaronrenner added a commit to aaronrenner/wallaby that referenced this pull request a day ago
- Set default Session.server to none

@aaronrenner added a commit to aaronrenner/wallaby that referenced this pull request a day ago
- Set default Session.server to none

@aaronrenner referenced this pull request 23 hours ago
Fix leaking webdriver sessions in QueryTest #209
- [Merged]
All checks have passed
1 successful check

This branch has no conflicts with the base branch
Merging can be performed automatically.

You can also open this in GitHub Desktop or view command line instructions.
HISTORY IS CRITICAL
$ git checkout c652b6b
Note: checking out 'c652b6b'.

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using -b with the checkout command again. Example:

    git checkout -b <new-branch-name>

HEAD is now at c652b6b... Add kubernetes setup to fish shell.
You are in 'detached HEAD' state.
This is fine.
$ hg merge --help --verbose | wc -w
# => 327
$ hg merge --help --verbose | wc -w
# => 327

$ git merge --help | wc -w
# => 3958
LET'S TALK ABOUT...

THE GOLDEN RULE OF GIT
TOOLS
SCENARIOS
Git’s Golden Rule
DON'T PANIC
If you don’t panic you can fix 100% of problems.
TOOLING
Git

No, Seriously
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:  config/fish/config.fish

no changes added to commit (use "git add" and/or "git commit -a")
YOU GIT CLI
ALIASSES
alias g='git'
alias gaa='git add -A'
alias gb='git branch'
alias gc='git commit'
alias gco='git checkout'
alias gd='git diff'
alias gglr='git pull --rebase'
alias ggp='git push'
alias gst='git status'
alias wip='git add -A && git commit -m "WIP"'
alias gc!='git commit -v --amend'
alias ggp!='git push --force'
alias **ibase**='git rebase -i $(git merge-base HEAD master)'
alias **gpr**='git pull-request -o'
alias **gcm**='git remote show origin \\n  | awk '/HEAD branch:/ {print \$3}' \\n  | xargs git checkout'
DIFFS
diff --git a/integration_test/cases/browser/local_storage_test.exs b/integration_test/cases/browser/local_storage_test.exs
index a684799..d901ee8 100644
--- a/integration_test/cases/browser/local_storage_test.exs
+++ b/integration_test/cases/browser/local_storage_test.exs
@@ -7,7 +7,6 @@
defmodule Wallaby.Integration.Browser.LocalStorageTest do
  @get_value_script "return localStorage.getItem('test')"
  @set_value_script "localStorage.setItem('test', 'foo')"
  
-  @tag :skip_test_session
  test "local storage is not shared between sessions" do
    # Checkout all sessions
    {:ok, session} = start_test_session()

diff --git a/integration_test/cases/query_test.exs b/integration_test/cases/query_test.exs
index 9a7f3ca..f0595a9 100644
--- a/integration_test/cases/query_test.exs
+++ b/integration_test/cases/query_test.exs
@@ -1,8 +1,15 @@
defmodule Wallaby.Integration.QueryTest do
-  use Wallaby.Integration.SessionCase, async: true
+  use ExUnit.QCase,    async: true
+  use Quixir

-  test "the driver can execute queries", %{session: session} do
+  alias Wallaby.Browser
+  alias Wallaby.Query
+  import Wallaby.Integration.SessionCase, only: [start_test_session: 0]
+  import Query
+  
+  :
DIFF-SO-FANCY
modified: integration_test/cases/browser/local_storage_test.exs

```elixir
defmodule Wallaby.Integration.BrowserLocalStorageTest do
  @get_value_script "return localStorage.getItem('test')"
  @set_value_script "localStorage.setItem('test', 'foo')"

  @tag :skip_test_session
  test "local storage is not shared between sessions" do
    # Checkout all sessions
    {:ok, session} = start_test_session()
  end
end```

modified: integration_test/cases/query_test.exs

```elixir
defmodule Wallaby.Integration.QueryTest do
  use Wallaby.Integration.SessionCase, async: true
  use ExUnit.Case, async: true
  use Quixir

  test "the driver can execute queries", %{session: session} do
    alias Wallaby.Browser
    alias Wallaby.Query
    import Wallaby.Integration.SessionCase, only: [start_test_session: 0]
    import Query

    test "the driver can execute queries" do
      {:ok, session} = start_test_session()
    end
  end
end```
**hub** is a command-line wrapper for git that makes you better at GitHub.

# OS X: install the latest release
$ brew install hub

# Other platforms: fetch a precompiled binary release, or build your own from source.
# You will need Go development environment
$ git clone https://github.com/github/hub.git && cd hub
# assuming that `~/bin` is in your PATH:
$ script/build -o ~/bin/hub

# alias it as git
$ alias git=hub

$ git version
git version 2.1.4
hub version 2.2.0 ← it works!
$ git checkout https://github.com/github/hub/pull/134
APPLY PATCH

$ git am https://github.com/xoebus/hub/commit/177eeb8
SUBMIT PR FROM CLI

$ git pull-request
GIT SCRIPTS
Git Scripts

λ cat ~/.bin/git-guilt
#!/bin/sh

git commit --amend --author "$1 <$2>" -C HEAD
You need a passphrase to unlock the secret key for user: "Chris Keathley <c@keathley.io>"
4096-bit RSA key, ID EA3A16BD, created 2017-04-23

[test-branch 0beb550] New file for demo
Author: Grace Hopper <grace@hopper.com>
Date: Wed May 3 21:54:10 2017 -0400
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 new_file
SCENARIOS
SCENARIO:
YOUR FIRST BUG
Here is a bug
DON’T PANIC
GIT BISECT
λ git bisect start HEAD HEAD~10
START A NEW BISECT

λ git bisect start HEAD HEAD~10
START A NEW BISECT

λ git bisect start HEAD HEAD~10

THE “BAD” REVISION
START A NEW BISECT

λ git bisect start HEAD HEAD~10

THE “GOOD” REVISION

THE “BAD” REVISION
\[ \lambda \text{ git bisect start HEAD HEAD}^{\sim 10} \]
λ git bisect start HEAD HEAD~10
\( \lambda \) git bisect good
\lambda \text{ git bisect good}
\( \lambda \text{ git bisect good} \)
\( \lambda \) git bisect bad
λ git bisect bad
\lambda \text{git bisect bad}
λ git bisect bad
λ git bisect bad

6fb10e5ec916063e4b4bd744678559afbf45d6791 is the first bad commit
commit 6fb10e5ec916063e4b4bd744678559afbf45d6791
Author: Aaron Renner <aaron.renner@gmail.com>
Date:   Tue May  2 07:55:01 2017 -0600

Fix leaking webdriver sessions in QueryTest (#209)

Wallaby.Integration.QueryTest was starting a ton of sessions and not
cleaning them up. This switches it over to use
Wallaby.Integration.SessionCase and uses the session that's
automatically created and cleaned up. I also extracted the unit tests
out of this integration test so they can be run separately.
λ git show -p 6fb10e5ec916063e4b4bd744678559af84d6791
YOU JUST FOUND THE CODE THAT INTRODUCED THE BUG!
$ cat test_script.sh
#!/bin/bash

if tests_pass; then exit 1; else exit 0; fi
$ git bisect run test_script.sh
A BRIEF INTERLUDE TO DISCUSS THE VALUE OF COMMIT HISTORY
BISECT ONLY WORKS IF THE REPO HISTORY IS CLEAN
If you history looks like this it's almost impossible to use bisects effectively.
MAKE COMMITS SMALL ATOMIC UNITS OF AWESOME!
SCENARIO:
START A FEATURE
Here's a feature story
LET'S TALK ABOUT BRANCHING
Some of y'all have some crazy-ass branching strats
HOW WE MERGE OUR CODE
$ git checkout master
$ git pull --rebase
$ git checkout -b feature/my-fancy-form-123456
feature/my-fancy-form-123456
feature/
bug/
chores/
DON'T PANIC
WIP && PUSH

$ git add -A && git commit -m "WIP"
$ git push -u origin HEAD
NOW YOUR WORK IS SAVED AND SHARED
$ git checkout master
$ git checkout -b bugs/put-out-server-fire
All checks have failed
1 failing check

This branch has conflicts that must be resolved
Use the web editor or the command line to resolve conflicts.

Conflicting files
apps/dc_interface/web/static/js/components/customer_returns/checkin.jsx

You can also open this in GitHub Desktop or view command line instructions.
WHAT HAPPENED?
$ git checkout master
$ git checkout -b bugs/put-out-server-fire
$ git checkout master
$ git pull --rebase
$ git checkout -b bugs/put-out-server-fire
MASTER
MASTER
MASTER
Conflict between files
DON’T PANIC
$ git rebase master
$ git rebase master
$ git rebase master
SCENARIO:
FIXING HISTORY
$ git checkout master
$ git pull --rebase
$ git checkout -
$ git rebase master
$ git checkout master
$ git pull --rebase
$ git checkout -
$ git rebase master
$ git checkout master
$ git pull --rebase
$ git checkout -
$ git rebase master
λ git rebase master
First, rewinding head to replay your work on top of it...
Applying: My Fancy Fix
Using index info to reconstruct a base tree...
M README.md
Falling back to patching base and 3-way merge...
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
error: Failed to merge in the changes.
Patch failed at 0001 WIP
The copy of the patch that failed is found in: .git/rebase-apply/patch

When you have resolved this problem, run "git rebase --continue".
If you prefer to skip this patch, run "git rebase --skip" instead.
To check out the original branch and stop rebasing, run "git rebase --abort".
# Chris Keathley's Dotfiles

<<<< HEAD
I :heart: dotfiles a whole lot.

## About

======

>>>>>> WIP

Dotfiles are what power your system. These are mine. They’re mostly for OS X since that’s what I use. My default shell these days is fish so everything is designed to work around fish functions and completions. All of the dotfile management is done with [rcm](https://github.com/thoughtbot/rcm) and packages are managed with homebrew.

## What’s in it?
RESOLVE CONFLICTS

$ git add -A
$ git rebase --continue
RESOLVE CONFLICTS

$ git push --force
All checks have passed
1 successful check

This branch has no conflicts with the base branch
Merging can be performed automatically.

Squash and merge
You can also open this in GitHub Desktop or view command line instructions.
SCENARIO: FINISHING OUR FEATURE
CHECKOUT THE FEATURE BRANCH

$ git checkout feature/my-fancy-feature
WIP commit

New Stuff
WIP commit

New Stuff
WIP commit

New Stuff

other stuff
keathley in ~/D/wallaby on chores/remove-select

λ
keathley in ~/D/wallaby on chores/remove-select

λ git add -i

<table>
<thead>
<tr>
<th>staged</th>
<th>unstaged path</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:</td>
<td>unchanged +4/-0 README.md</td>
</tr>
<tr>
<td>2:</td>
<td>unchanged +12/-0 integration_test/cases/browser/click_test.exs</td>
</tr>
<tr>
<td>3:</td>
<td>unchanged +0/-8 integration_test/cases/browser/select_test.exs</td>
</tr>
<tr>
<td>4:</td>
<td>unchanged +2/-32 lib/wallaby/browser.ex</td>
</tr>
</tbody>
</table>

*** Commands ***

1: status  2: update  3: revert  4: add untracked
5: patch   6: diff    7: quit    8: help

What now>
keathley in ~/D/wallaby on chores/remove-select

λ git add -i

  staged   unstaged   path
  1:  unchanged   +4/-0  README.md
  2:  unchanged   +12/-0 integration_test/cases/browser/click_test.exs
  3:  unchanged   +0/-8  integration_test/cases/browser/select_test.exs
  4:  unchanged   +2/-32  lib/wallaby/browser.exs

*** Commands ***
1: status  2: update  3: revert  4: add untracked
5: patch  6: diff    7: quit    8: help

What now> 5

  staged   unstaged   path
  1:  unchanged   +4/-0  README.md
  2:  unchanged   +12/-0 integration_test/cases/browser/click_test.exs
  3:  unchanged   +0/-8  integration_test/cases/browser/select_test.exs
  4:  unchanged   +2/-32  lib/wallaby/browser.exs

Patch update>> *
Wallaby helps you test your web applications by simulating user interactions. By
* Manages multiple browser processes.
* Works with Ecto's test Sandbox.

## This is some new documentation

Did you know that the average life-span of a Wallaby is 9 years.

## Setup

Add Wallaby to your list of dependencies in `mix.exs`:

Use of uninitialized value $_ in print at `/usr/local/Cellar/git/2.10.0/libexec/git-core/git-add--interactive line 1368.```
@ click_test.exs:97 @ defmodule Wallaby.Integration.Browser.ClickTest do
  assert click(page, Query.checkbox("Hidden Checkbox"))
end

describe "click/2 with options" do
  test "works with option queries", %{page: page} do
    assert page
    |> find(Query.select("My Select"))
    |> click(Query.option("Option 2"))
  end
  
  test "escapes quotes", %{page: page} do
    assert click(page, Query.option("I'm an option"))
  end
end

Use of uninitialized value $ in print at /usr/local/Cellar/git/2.10.0/libexec/git-core/git-add--interactive line 1368.
Stage this hunk [y,n,q,a,d,,e,?]? y
Stage this hunk [y,n,q,a,d,/,e,?] y

modified: integration_test/cases/browser/select_test.exs

```elixir
@select_test.exs:66 @defmodule Wallaby.Integration.Browser.SelectTest do
  assert select(page, "I'm a select", option: "I'm an option")
end

describe "select/2" do
  test "works with option queries", %{page: page} do
    assert page |
    find(Query.select("My Select")) |
    select(Query.option("Option 2")) |
  end
end

describe "selected?/2" do
  test "returns a boolean if the option is selected", %{page: page} do
    assert page
Use of uninitialized value $ in print at /usr/local/Cellar/git/2.10.0/libexec/git-core/git-add--interactive line 1368.
Stage this hunk [y,n,q,a,d,/,e,?] y
```
parent
 | find(query, &Element.click/1)
end

def select(parent, locator, [option: option_text]=opts) do
  IO.warn ""
  select/3 has been deprecated. Please use:
  
  click(parent, Query.option("#{option_text}"))
  ""

  find(parent, Query.select(locator, opts), fn(select_field) ->
    find(select_field, Query.option(option_text, []), fn(option) ->
      Element.click(option)
    end)
  end)
end

def wallaby_lifespan() do
  :9_years
end

@doc ""
Use of uninitialized value $._ in print at /usr/local/Cellar/git/2.10.0/libexec/git-core/git-add--interactive line 1368.
Stage this hunk [y,n,q,a,d,/,e,?]?
parent
  |> find(query, &Element.click/1)
end

def select(parent, locator, [option: option_text=opts) do
  IO.warn ""
  select/3 has been deprecated. Please use:

  click(parent, Query.option("#{option_text}"))
  ""

  find(parent, Query.select(locator, opts), fn(select_field) ->
    find(select_field, Query.option(option_text, []), fn(option) ->
      Element.click(option)
    end)
  end)
end)

def wallaby_lifespan() do
  :9_years
end

@doc ""
Use of uninitialized value $_[ in print at /usr/local/Cellar/git/2.10.0/libexec/git-core/git-add--interactive line 1368.
Stage this hunk [y,n,q,a,d,,e,?] ? e""
addp-hunk-edit.diff

- find(parent, Query.select(locator, opts), fn(select_field) ->
-   find(select_field, Query.option(option_text, []), fn(option) ->
-     Element.click(option)
-   end)
- end)
+ def wallaby_lifespan() do
+   :9_years
+ end

@doc ""

# ---
# To remove `-` lines, make them `' ' lines (context).
# To remove `+` lines, delete them.
# Lines starting with `#` will be removed.
#
# If the patch applies cleanly, the edited hunk will immediately be
# marked for staging. If it does not apply cleanly, you will be given
# an opportunity to edit again. If all lines of the hunk are removed,
# then the edit is aborted and the hunk is left unchanged.

~/Development/wallaby/.git/addp-hunk-edit.diff" 51L, 1660C
select/3 has been deprecated. Please use:

click(parent, Query.option("#{option_text}"))

find(parent, Query.select(locator, opts), fn(select_field) ->
    find(select_field, Query.option(option_text, []), fn(option) ->
        Element.click(option)
    end)
end)
def wallaby_lifespan() do
    :9_years
end

@doc ""
Use of uninitialized value $_ in print at /usr/local/Cellar/git/2.10.0/libexec/git-core/git-add--interactive line 1368.
Stage this hunk [y,n,q,a,d,/,e,?]? e

*** Commands ***
1: status   2: update     3: revert     4: add untracked
5: patch    6: diff       7: quit       8: help
What now>
```bash
find(parent, Query.select(locator, opts), fn(select_field) ->
    find(select_field, Query.option(option_text, []), fn(option) ->
        Element.click(option)
    end)
end)

def wallaby_lifespan() do
    :9_years
end

@doc ""
Use of uninitialized value $_ in print at /usr/local/Cellar/git/2.10.0/libexec/git-core/git-add--interactive line 1368.
Stage this hunk [y,n,q,a,d,,/e,?]? e

*** Commands ***
  1: status     2: update    3: revert    4: add untracked
  5: patch      6: diff      7: quit      8: help
What now> q
Bye.
keathley in ~/D/wallaby on chores/remove-select
```
```ruby
def select(parent, query) do
  IO.warn "select/2 has been deprecated. Please use click/2"

  parent
  |> find(query, &Element.click/1)
end

def select(parent, locator, [option: option_text]=opts) do
  IO.warn ""
  select/3 has been deprecated. Please use:

  click(parent, Query.option("#{option_text}"))
  ""

  find(parent, Query.select(locator, opts), fn(select_field) ->
    find(select_field, Query.option(option_text, [], fn(option) ->
      Element.click(option)
    end)
  end)
end

@doc ""
keathley in ~/D/wallaby on chores/remove-select
```
COMMIT THE CURRENT CHANGES

$ git commit
Commit everything else

$ git add -A
$ git commit
$ git add -A
$ git commit
INTERACTIVE

REBASE
INTERACTIVE REBASE

$ git rebase -i HEAD~3
keathley in ~/D/wallaby on chores/remove-select

g rebase -i HEAD~3
git-rebase-todo

pick c4686a0 WIP
pick edb868e Remove select function
pick 804ec33 Add fun information about wallabies

# Rebase f4b7d8b..804ec33 onto f4b7d8b (3 commands)
#
# Commands:
# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using shell
# d, drop = remove commit
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.

NORMAL > chores/remove-select > <t-rebase-todo gitrebase << 4% : 1/22 : 1 <<
"~/Development/wallaby/.git/rebase-merge/git-rebase-todo" 22L, 741C
git-rebase-todo

pick c4686a0 WIP
s edb868e Remove select function
pick 804ec33 Add fun information about wallabies

# Rebase f4b7d8b..804ec33 onto f4b7d8b (3 commands)
#
# Commands:
# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using shell
# d, drop = remove commit
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
Rebasing (2/3)
You need a passphrase to unlock the secret key for user: "Chris Keathley <c@keathley.io>"
4096-bit RSA key, ID EA3A16BD, created 2017-04-23

[detached HEAD f55b0d8] Remove select function
  Date: Thu May 4 22:57:12 2017 -0400
  4 files changed, 12 insertions(+), 40 deletions(-)
  create mode 100644 new_file
Rebasing (3/3)
You need a passphrase to unlock the secret key for user: "Chris Keathley <c@keathley.io>"
4096-bit RSA key, ID EA3A16BD, created 2017-04-23

Successfully rebased and updated refs/heads/chores/remove-select.
New Stuff

other stuff
SCENARIO:
LOOSING A COMMIT
$ git rebase -i HEAD~10
λ git rebase master
First, rewinding head to replay your work on top of it...
Applying: My Fancy Fix
Using index info to reconstruct a base tree...
M README.md
Falling back to patching base and 3-way merge...
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
error: Failed to merge in the changes.
Patch failed at 0001 WIP
The copy of the patch that failed is found in: .git/rebase-apply/patch

When you have resolved this problem, run "git rebase --continue". If you prefer to skip this patch, run "git rebase --skip" instead. To check out the original branch and stop rebasing, run "git rebase --abort".
$ git rebase --continue
λ mix test

1) test it works (Metaform.Web.PageViewTest)
   test/web/views/page_view_test.exs:4
   This is broken now
   stacktrace:
     test/web/views/page_view_test.exs:5: (test)

....

Finished in 0.1 seconds
5 tests, 1 failure
YOUR WORK WAS LOST IN A CONFLICT
DON’T PANIC
$ git reflog
THE REFLOG IS ALL OF YOUR REPOS HISTORY
$ git reflog
f1b1 HEAD@{0}: rebase -i (finish): returning to refs/heads/chores/remove-select
677f1b1 HEAD@{1}: rebase -i (pick): Update new file
f55b0d8 HEAD@{2}: cherry-pick: fast-forward
f4b7d8b HEAD@{3}: rebase -i (start): checkout HEAD~3
c47093 HEAD@{4}: commit: Update new file
039d19a HEAD@{5}: commit (amend): My super important commit don't lose this.
62fc606 HEAD@{6}: rebase -i (finish): returning to refs/heads/chores/remove-select
f1b1 HEAD@{0}: rebase -i (finish): returning to refs/heads/chores/remove-select
677f1b1 HEAD@{1}: rebase -i (pick): Update new file
f55b0d8 HEAD@{2}: cherry-pick: fast-forward
f4b7d8b HEAD@{3}: rebase -i (start): checkout HEAD~3
ca47093 HEAD@{4}: commit: Update new file
039d19a HEAD@{5}: commit (amend): My super important commit don't loose this.
62fc606 HEAD@{6}: rebase -i (finish): returning to refs/heads/chores/remove-select
039d19a
$ git show -p 039d19a

$ git reset --hard 039d19a
CONCLUSION
RESOURCES

DIFF-SO-FANCY - HTTPS://GITHUB.COM/SO-FANCY/DIFF-SO-FANCY
HUB - HTTPS://GITHUB.COM/GITHUB/HUB
SLIDES - HTTPS://SPEAKERDECK.COM/KEATHLEY
TOOLS
BRANCHING
REFLOG
DON’T PANIC
THANKS!

CHRIS KEATHLEY
@CHRISKEATHLEY
C@KEATHLEY.IO