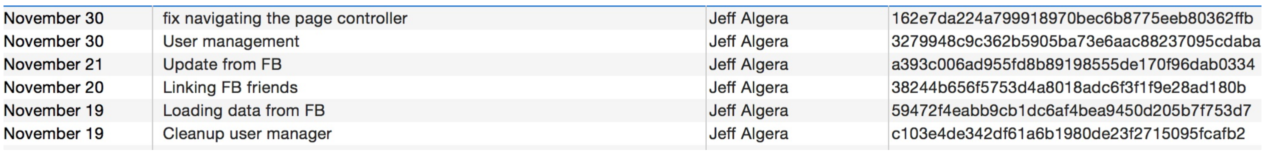
Git and GitHub

Overview

Our Mobile Development course will be using Git as our versioning control system. We will use GitHub as our interface to that system. This document will cover

---

Git is a tool software developers use to manage changes in code. Those changes may be your own, or may be made by members of your team.



November 30	fix navigating the page controller	Jeff Algera	162e7da224a799918970bec6b8775eeb80362ffb
November 30	User management	Jeff Algera	3279948c9c362b5905ba73e6aac88237095cdaba
November 21	Update from FB	Jeff Algera	a393c006ad955fd8b89198555de170f96dab0334
November 20	Linking FB friends	Jeff Algera	38244b656f5753d4a8018adc6f3f1f9e28ad180b
November 19	Loading data from FB	Jeff Algera	59472f4eabb9cb1dc6af4bea9450d205b7f753d7
November 19	Cleanup user manager	Jeff Algera	c103e4de342df61a6b1980de23f2715095fcafb2

A history of changes made in a Git repository

GitHub is a company that provides a really excellent interface for managing Git repositories.

Secret lair of GitHub in San Francisco.

[Video] Introduction to Git and Github

What Git and GitHub do, why you should use them, and how to get started:

References

Git quick reference for beginners - http://www.dataschool.io/git-quick-reference-for-beginners/

GitHub - http://github.com

Git according to Wikipedia - http://en.wikipedia.org/wiki/Git_(software)

GitHub instructional videos published on YouTube - https://www.youtube.com/user/GitHubGuides

Version Control

Version control provides a number of benefits to the developer. Here are some of those benefits:

1. History
2. Version Control
3. Branching
4. A Distributed System

Benefit 1: History

Imagine working on a piece of code. Let's call this version 1:

```
var eureka = "The meaning of life is the following number"
var meaning_of_life = 42
print("Guys, I've figured it out.  \(eureka) is \(meaning_of_life)!")
```

Then after a few days, you change your mind and update the code. Let's call this version 2:

```
var eureka = "The meaning of life is the following number"
var meaning_of_life = 99999
print("Guys, I've figured it out.  \(eureka) is \(meaning_of_life)!")
```

Some time passes, you close Xcode, shut down your computer and then you realize:

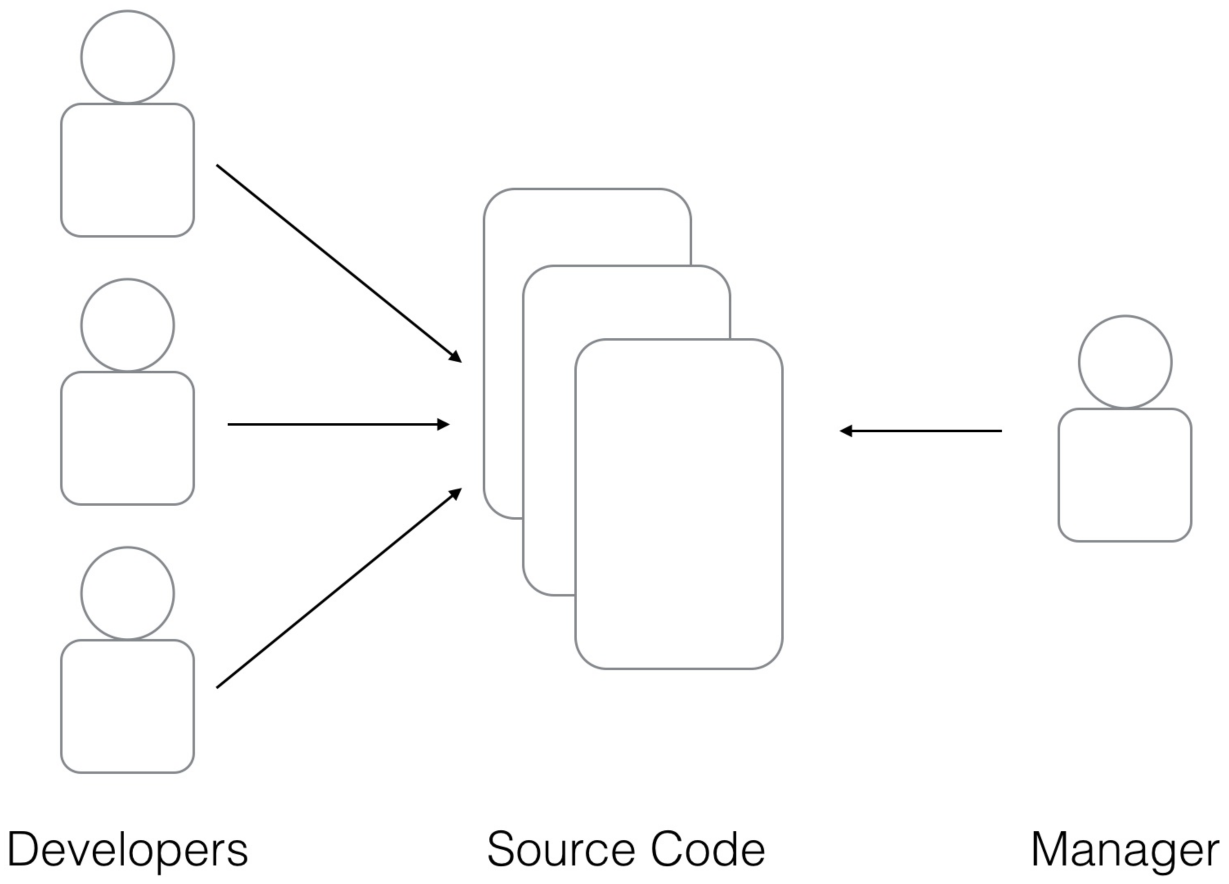
> The first version was right! What did I have?

With Git, you can have a historical record of all changes made in your code. You can then reference those changes at a later time.

```
var meaning_of_life = 42
```

Benefit 2: Working with a team

Now imagine you are working with a team of three programmers on an awesome new app. There is one Xcode project, three programmers and one manager.

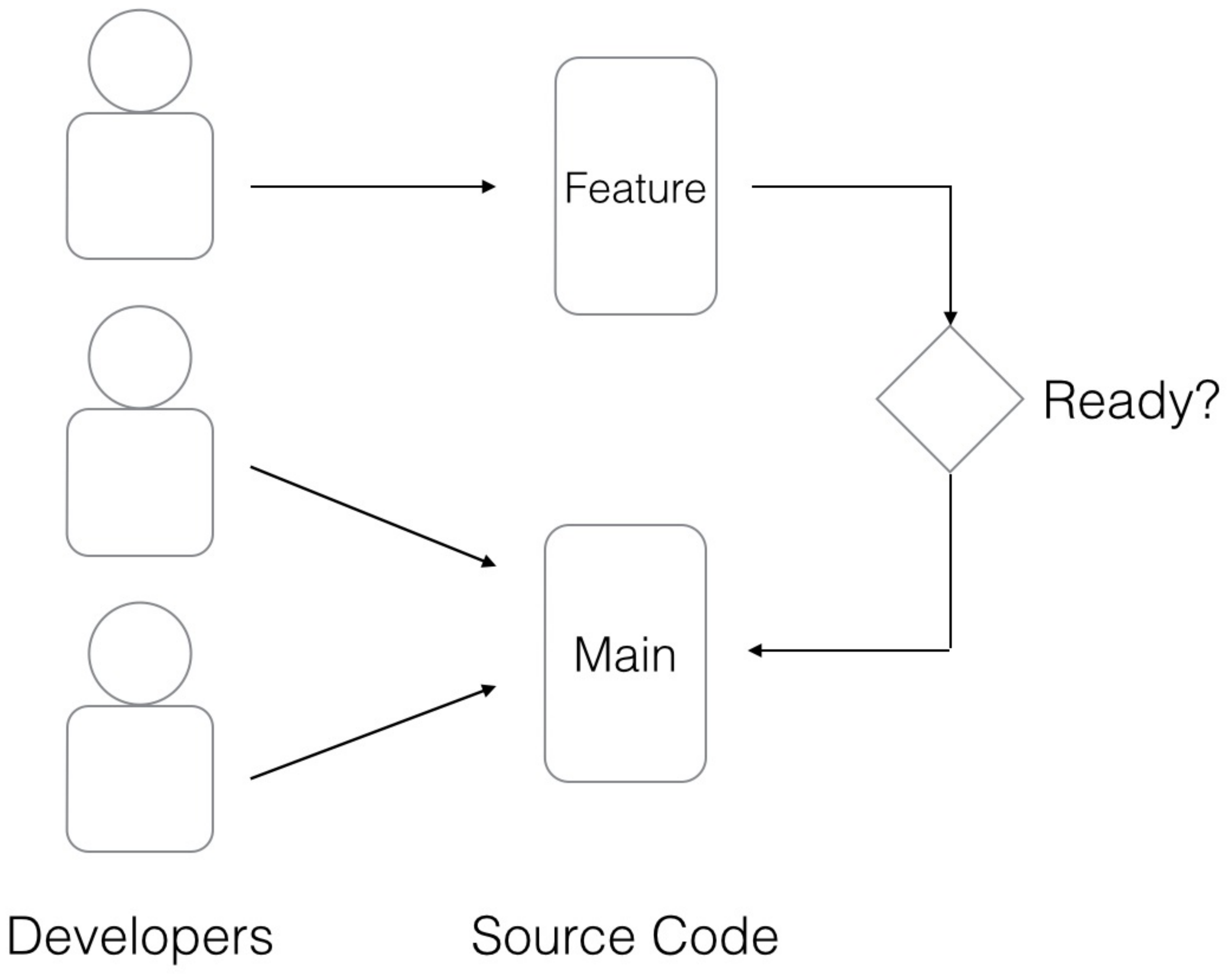


The benefits you get with a version control system, such as Git, in this scenario:

1. Historical record of what each programmer was working on and when
2. The ability to go back in time and view changes from a different day for any of the programmers on the team
3. Automatic merging of changes from each developer into one central project
4. One central project that a manager can download and send off to the client.

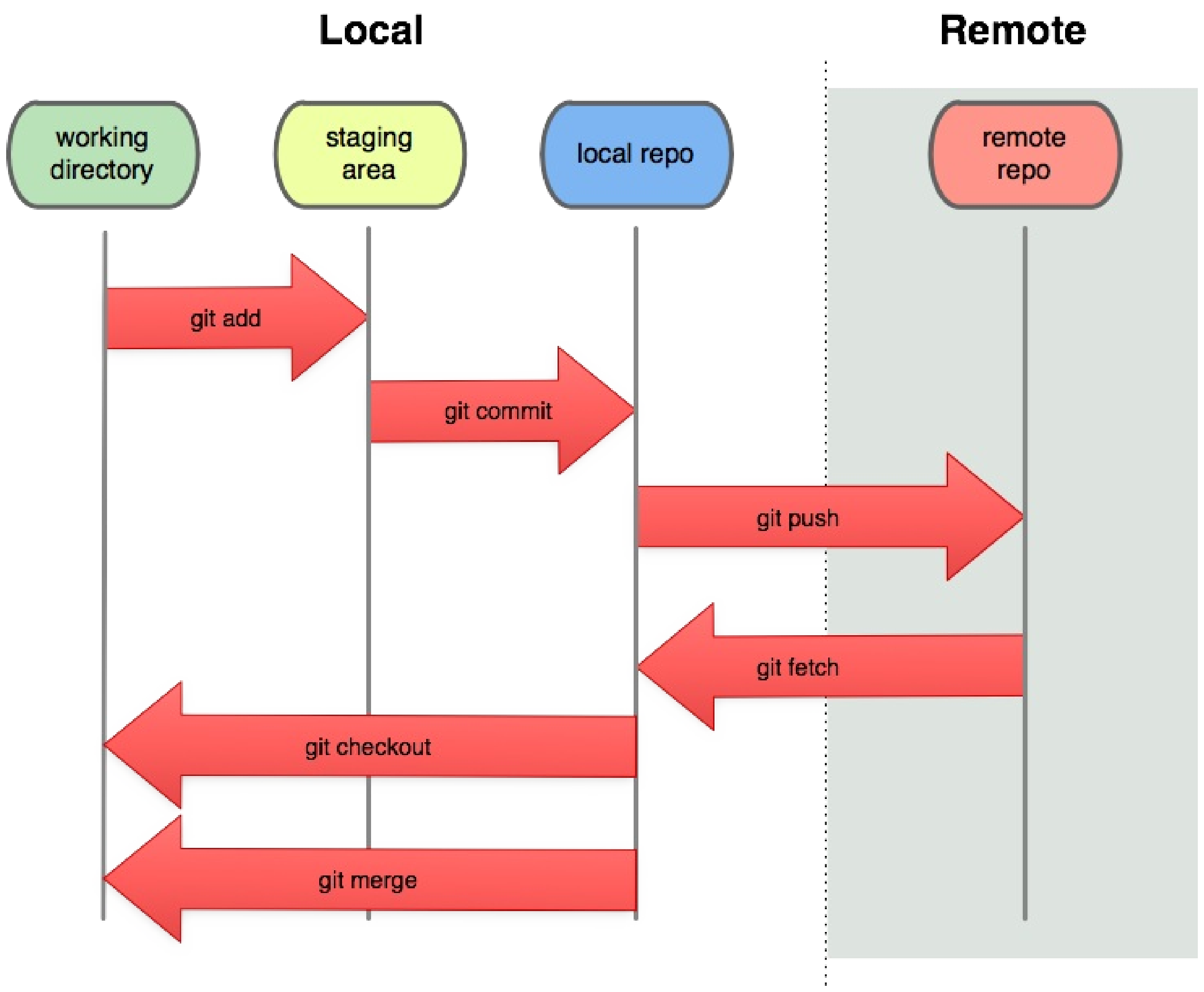
Benefit 3: Branching

Branches are commonly used to segregate new code from the existing project. Let's say your manager decides she wants to incorporate a new feature into your awesome app and only wants that feature to be available to the rest of the developers when it's complete and ready.



With Git you can work on your own branch and incorporate your changes at a later time, while retaining the other benefits of version control, as previously described.

Benefit 4: Local versus Remote - A Distributed System



When you're working with Git, you are working with your own copy. This is what we refer to as your **local** copy. With your local copy, you can do whatever you want. Add some new buttons to your app, change todo en español, get dangerous and experiment and tinker throughout the app.

> This is a great way to learn and experiment

This is your local copy and you're not affecting others. You can commit changes to your local copy (even without Internet access) and decide to abandon it all and get a fresh copy from the remote.

When you're ready, you can publish your changes from your local copy into the remote repository.

References

Why Git? - http://www.markus-gattol.name/ws/scm.html

Git User Interfaces

We are going to describe three different ways to interface into Git. Those are:

1. The GitHub website
2. SourceTree from Altassain
3. (optional) GitHub for Mac

Wrap Up

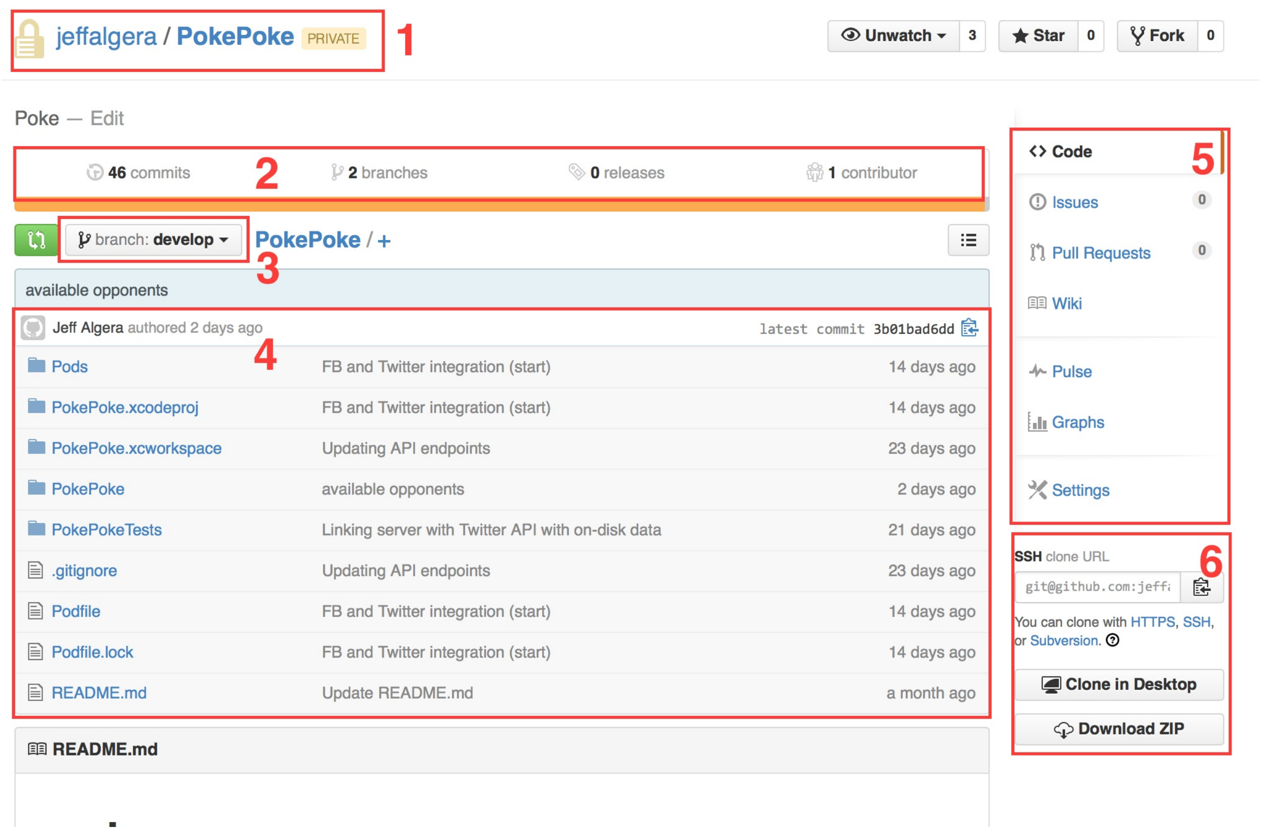
GitHub for web, SourceTree and GitHub for Mac are powerful tools that developers use each day. They enable the syncing of code across distributed systems and keep teams progressing a unified code base.

For further study, please consult this series of excellent YouTube videos published by Data School - https://www.youtube.com/playlist?list=PL5-da3qGB5IBLMp7LtN8Nc3Efd4hJq0kD

GitHub for Web

http://github.com

GitHub provides an excellent web interface for viewing and managing repositories.



jeffalgera / PokePoke PRIVATE 1

Unwatch 3 Star 0 Fork 0

Poke — Edit

46 commits 2 branches 0 releases 1 contributor 2

branch: develop 3 PokePoke / +

available opponents

Jeff Algera authored 2 days ago latest commit 3b01bad6dd 4

Pods	FB and Twitter integration (start)	14 days ago
PokePoke.xcodeproj	FB and Twitter integration (start)	14 days ago
PokePoke.xcworkspace	Updating API endpoints	23 days ago
PokePoke	available opponents	2 days ago
PokePokeTests	Linking server with Twitter API with on-disk data	21 days ago
.gitignore	Updating API endpoints	23 days ago
Podfile	FB and Twitter integration (start)	14 days ago
Podfile.lock	FB and Twitter integration (start)	14 days ago
README.md	Update README.md	a month ago

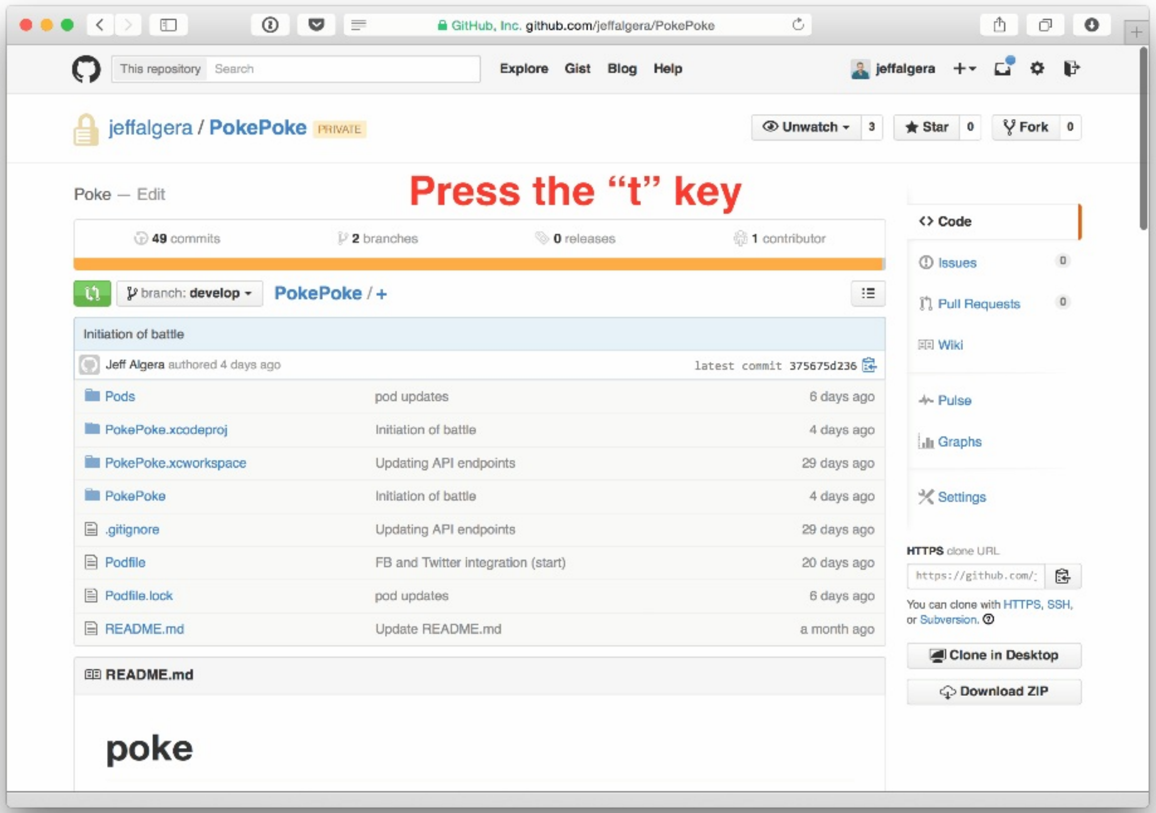
README.md

What you are seeing here is the remote representation of a repository. In detail:

1. **Repo name and type** - Describes who owns the repository, what the name of the repo is and whether the repo is public or private
2. **A selectable overview** of commits, branches, releases and contributors for a particular repository. Selecting any one of these options will bring a detail view of that selection.
3. A drop-down selection of the **currently viewed branch**. Changing the branch will update the source code to reflect that branch.
4. A selectable **source code list** organized within the committed file structure.
5. **Side bar** - use the side bar for responding to issues and creating pull requests. Also contained here are the settings per repository.
6. **Connection settings** - the lower right pane contains information on connecting to your repository. Typically, you will be wanting the SSH connection string to input into your Mac Github client. ZIP archive is good for downloading a backup of the repo.

Blame

One of the great features of the GitHub website, is the ability to drill down and view specific information about individual source code files including commit logs.



jeffalgera / PokePoke PRIVATE

Poke — Edit

49 commits · 2 branches · 0 releases · 1 contributor

branch: develop · PokePoke / +

Initiation of battle

Jeff Algera authored 4 days ago · latest commit 375675d236

Pods	pod updates	6 days ago
PokePoke.xcodeproj	Initiation of battle	4 days ago
PokePoke.xcworkspace	Updating API endpoints	29 days ago
PokePoke	Initiation of battle	4 days ago
.gitignore	Updating API endpoints	29 days ago
Podfile	FB and Twitter integration (start)	20 days ago
Podfile.lock	pod updates	6 days ago
README.md	Update README.md	a month ago

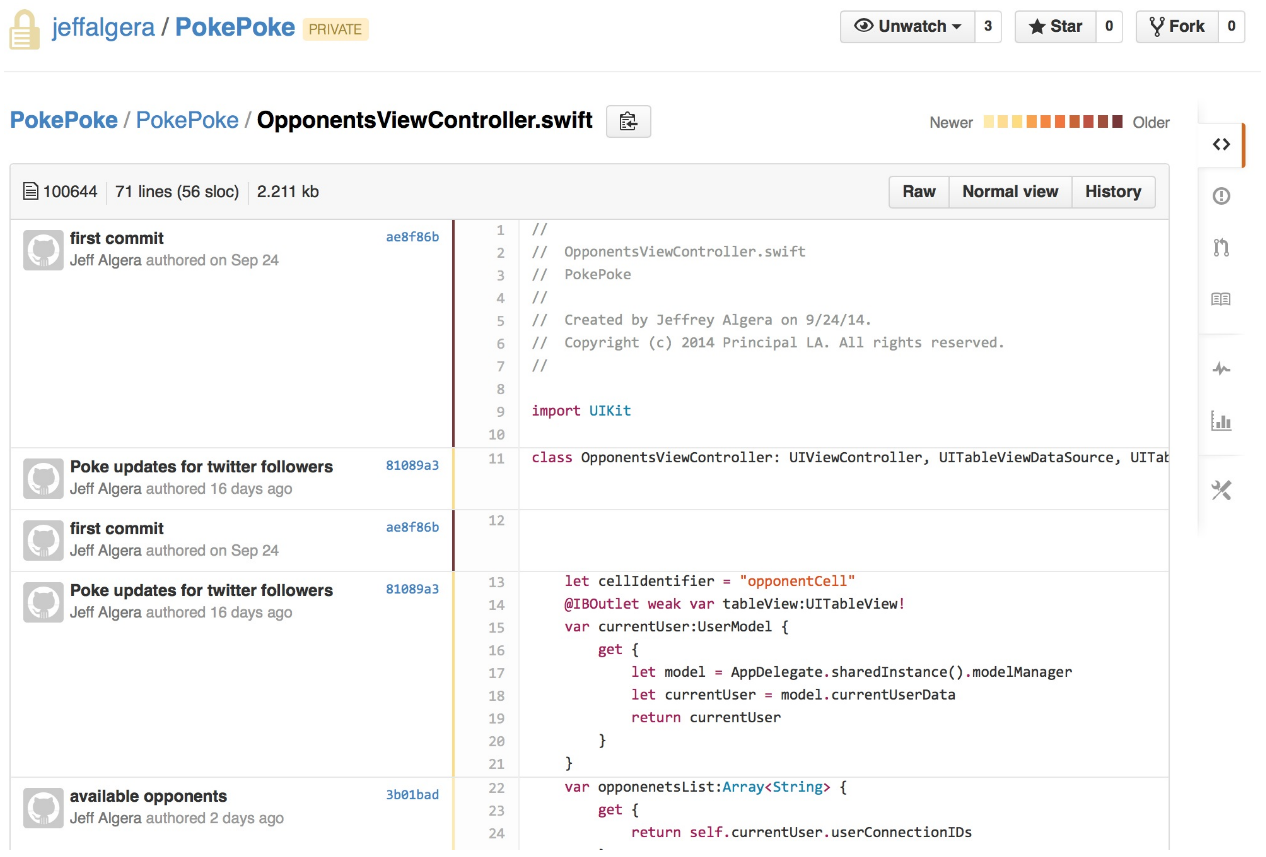
README.md

poke

You can clone with HTTPS, SSH, or Subversion.

The steps to view blame are as follows:

1. From the GitHub repository home page, hit the letter 't'
2. Being typing the filename. In our above example, we are typing AppDelegate
3. GitHub will filter in realtime file names, object and method names and display them in a clickable list
4. In our example, we were looking for AppDelegate.swfit, so we click on that
5. Then hit the **blame** button on the top-right toolbar
6. And behold, blame! A sequential list of your source code file with all changes made.


jeffalgera / PokePoke PRIVATE

Unwatch 3 | Star 0 | Fork 0

PokePoke / PokePoke / OpponentsViewController.swift

Newer Older

100644 | 71 lines (56 sloc) | 2.211 kb

Raw | Normal view | History

first commit ae8f86b
Jeff Algera authored on Sep 24

```
//
//  OpponentsViewController.swift
//  PokePoke
//
//  Created by Jeffrey Algera on 9/24/14.
//  Copyright (c) 2014 Principal LA. All rights reserved.
//

import UIKit

```

Poke updates for twitter followers 81089a3
Jeff Algera authored 16 days ago

```
class OpponentsViewController: UIViewController, UITableViewDataSource, UITab
```

first commit ae8f86b
Jeff Algera authored on Sep 24

Poke updates for twitter followers 81089a3
Jeff Algera authored 16 days ago

```
    let cellIdentifier = "opponentCell"
    @IBOutlet weak var tableView:UITableView!
    var currentUser:UserModel {
        get {
            let model = AppDelegate.sharedInstance().modelManager
            let currentUser = model.currentUserData
            return currentUser
        }
    }
```

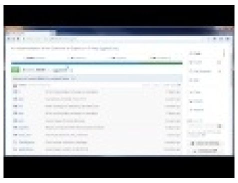
available opponents 3b01bad
Jeff Algera authored 2 days ago

```
    var opponenetsList:Array<String> {
        get {
            return self.currentUser.userConnectionIDs
```

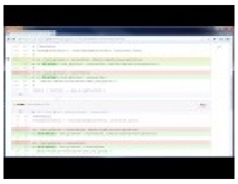
In the above graphic, we are looking at a blame of an individual source code file. Note the detail. You have a line-by-line listing of when that code was changed, by whom and a selectable commit entry to reference how that change was made. All items in blue are selectable.

[Videos] Navigating a GitHub Repository

Part 1:



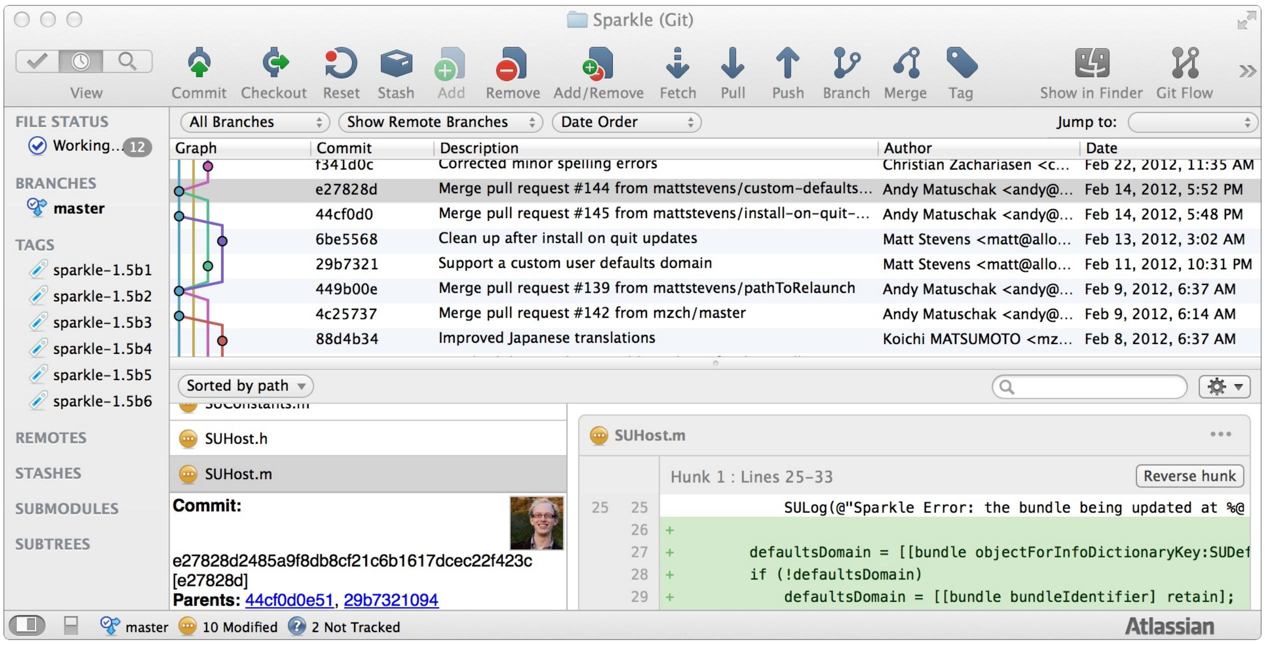
Part 2:



SourceTree for Mac

SourceTree is a Git client created by Altassian. They describe their product as, "(P)erfect for newcomers." Where "(c)reate, clone, commit, push, pull, merge, and more are all just a click away."

Download from http://www.sourcetreeapp.com



Graph	Commit	Description	Author	Date
	f341d0c	Corrected minor spelling errors	Christian Zachariasen <c...	Feb 22, 2012, 11:35 AM
	e27828d	Merge pull request #144 from mattstevens/custom-defaults...	Andy Matuschak <andy@...	Feb 14, 2012, 5:52 PM
	44cf0d0	Merge pull request #145 from mattstevens/install-on-quit-...	Andy Matuschak <andy@...	Feb 14, 2012, 5:48 PM
	6be5568	Clean up after install on quit updates	Matt Stevens <matt@allo...	Feb 13, 2012, 3:02 AM
	29b7321	Support a custom user defaults domain	Matt Stevens <matt@allo...	Feb 11, 2012, 10:31 PM
	449b00e	Merge pull request #139 from mattstevens/pathToRelaunch	Andy Matuschak <andy@...	Feb 9, 2012, 6:37 AM
	4c25737	Merge pull request #142 from mzch/master	Andy Matuschak <andy@...	Feb 9, 2012, 6:14 AM
	88d4b34	Improved Japanese translations	Koichi MATSUMOTO <mz...	Feb 8, 2012, 6:37 AM

Commit:

e27828d2485a9f8db8cf21c6b1617dcec22f423c [e27828d]

Parents: 44cf0d0e51, 29b7321094

SUHost.m

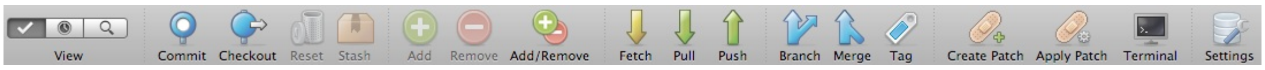
Hunk 1 : Lines 25-33

```
25 25          SULog(@"Sparkle Error: the bundle being updated at %@
   26  +
   27  +      defaultsDomain = [[bundle objectForInfoDictionaryKey:SUDef
   28  +      if (!defaultsDomain)
   29  +          defaultsDomain = [[bundle bundleIdentifier] retain];
```

Window Layout

Below is a brief overview of SourceTree, its various controls and layout. This material is found within SourceTree itself by selecting Help from the file menu.

The Toolbar



The View Button This button allows you to switch between the 3 main views of the repository: File Status, Log, and Search.

Commit

Opens the Commit dialog so you can commit your changes. In Git, if you're using staging, the default is to open the dialog with the commit of staged changes selected.

Checkout

Allows you to switch your local working copy to a different point in history.

Reset

Use this to undo changes in your working copy.

Stash

Moves any uncommitted changes you have in your working copy to a storage area, and returns your working copy to a clean state.

Add

Adds any selected untracked files to source control - the next commit will start to track these files.

Remove

Stops tracking selected files and removes them from the working copy. The next commit will remove these files from being tracked in source control.

Add/Remove

A shortcut to stop tracking all files which are missing from the working folder, and to add any untracked files to source control.

Fetch

Download new commits from your remotes, but don't bring them in to your own branch yet.

Pull

Download new commits from your remotes and bring them in to your current branch, either by merging or rebasing.

Push

Upload new commits to a remote. This icon will have a number superimposed on it if you have commits which you haven't pushed yet.

Branch

Create a new branch (also includes a tab for removing branches)

Merge

Merge changes into your current branch.

Tag

Create and manage tags.

Create Patch

Create a patch file either from your current uncommitted changes, or from one or more commits that you've already made.

Apply Patch

Apply a patch file either to your working copy or directly to the commit log.

Terminal

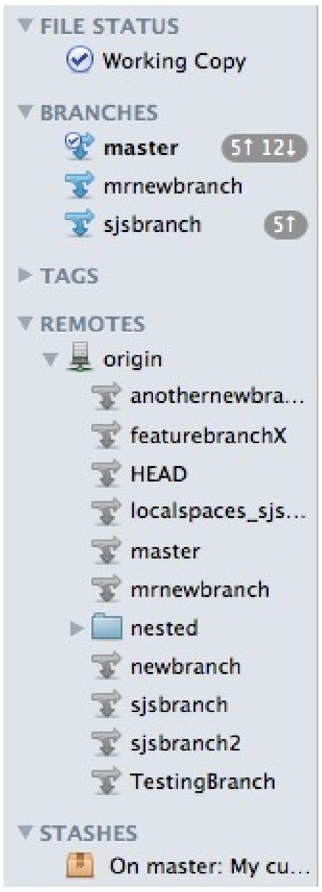
Open a Terminal at the selected location.

Settings

Access per-repository settings such as remotes.

The Sidebar

The sidebar is found on the left-hand side of the SourceTree application.



Git

The File Status Section

Selecting the "Working Copy" item behaves the same way as if you click on the File Status button on the View control on the toolbar.

The Branches section

This displays all your local branches. You can also right-click on the branch to get a number of context-sensitive options such as updating to/checking out the branch, merging it into your current branch, diffing that branch against your current branch. Double-clicking a branch will switch your current branch (after a confirmation dialog).

Your current branch is indicated with bold text and a 'tick badge' on the branch icon. The number of commits that your branch is ahead or behind the remote branch that it's tracking is displayed alongside the name, with an up arrow for ahead, and a down arrow for behind.

Note: if you create a branch with a name which includes a '/' character, SourceTree will display the branch as nested within a folder.

The Tags section

This section is not expanded by default but if you expand it you'll see one entry for each tag. Selecting one will jump to that tag in the Log view, and there are options available in the context menu such as updating to the tag, deleting the tag or diffing your current working copy against it.

The Remotes section This is where you find a list of remotes configured for this repository. You can pull from or push to a specific remote from the context menu on each item.

On Git, you will also find a list of remote branches, and you can perform actions on these via the right-click context menu or double-clicking to checkout.

The Stashes

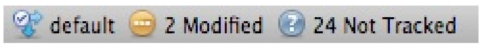
A new item is created in this section, each one representing a set of changes that have been stored away for future use. You can click on these items to see a diff view of the changes they represent, and right-click an item to either apply it back to your working copy, or to delete it if you don't want it anymore.

The Submodules

If your project has nested repositories within it, called submodules. They will appear here. See the submodules section for more details.

The Footer

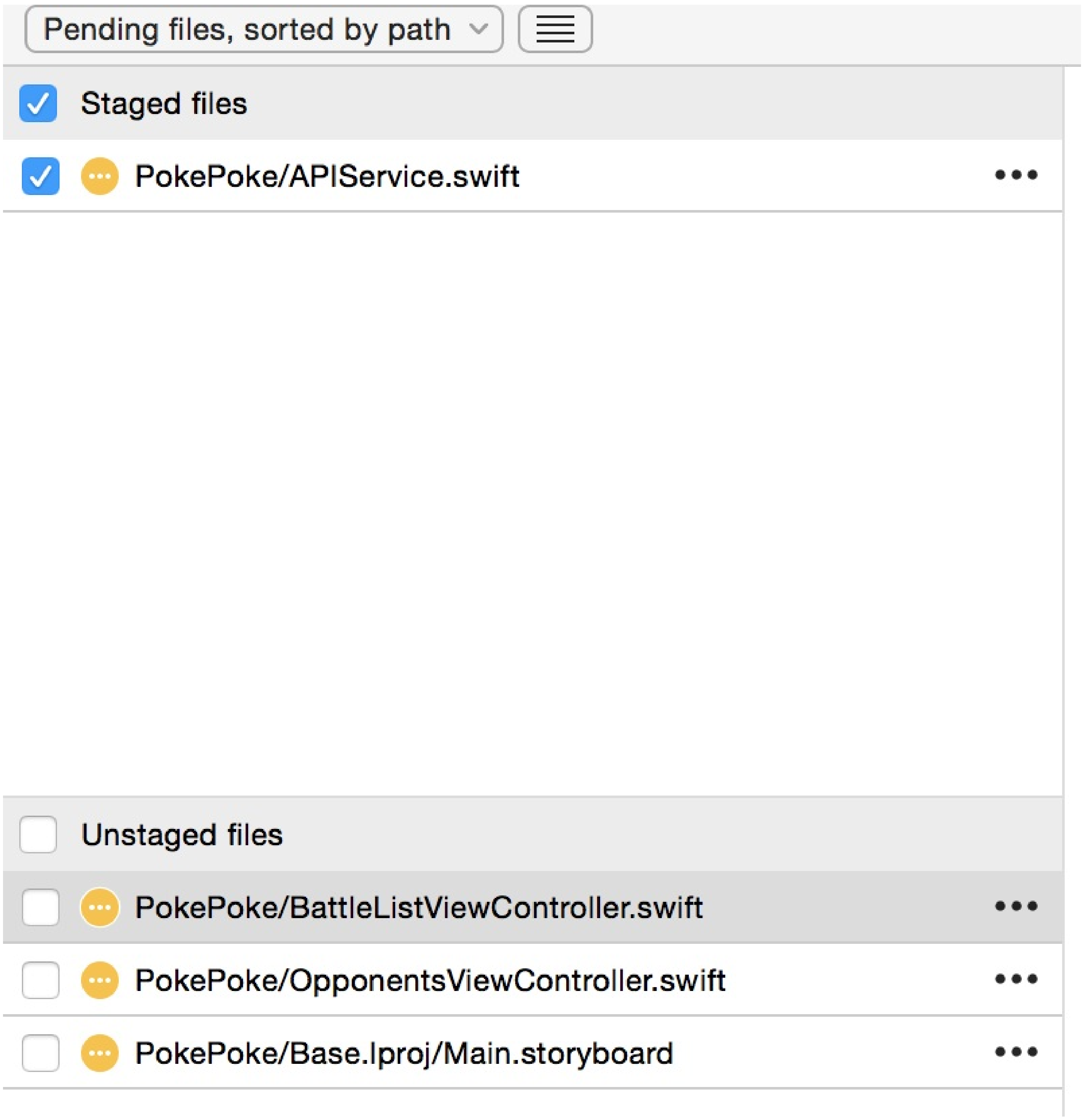
Found at the bottom of SourceTree window is the footer.



You'll see here the name of the current branch, followed by either a 'Clean' marker if there are no local changes, or a summary of the number of files outstanding in each state if there are local changes. You will also see information here relating to any outstanding merges and conflicts.

File Status List

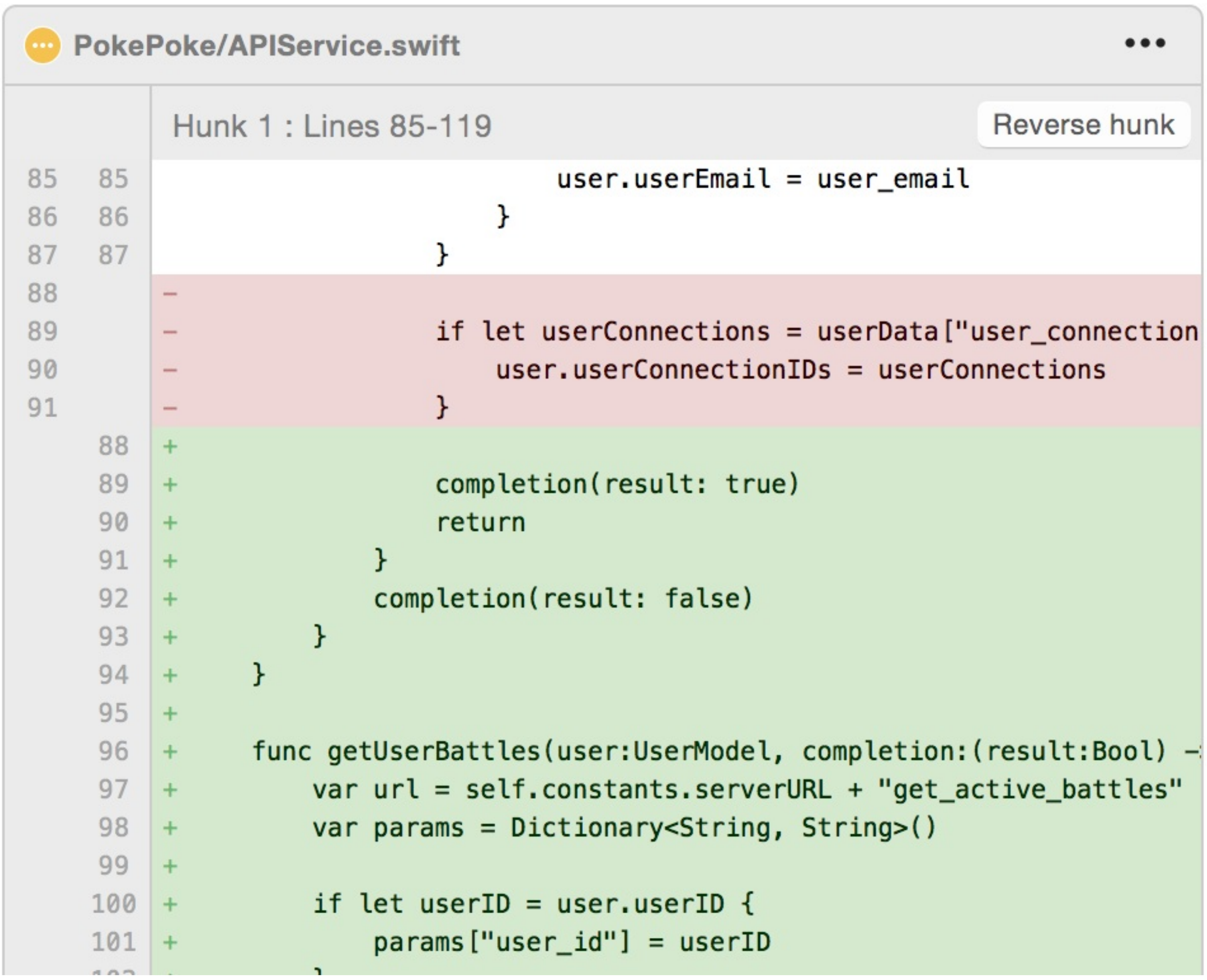
On the left-side of the SourceTree window, after the sidebar, is the File Status list.


Pending files, sorted by path
Staged files
PokePoke/APIService.swift
Unstaged files
PokePoke/BattleListViewController.swift
PokePoke/OpponentsViewController.swift
PokePoke/Base.lproj/Main.storyboard

This area shows files in your working copy which have been modified in some way, although you can use the checkboxes at the bottom to filter the statuses that are displayed. Selecting one of these files will display those changes in the differences pane.

Differences Pane

Colorful green/pink view on the right side of the SourceTree application.



PokePoke/APIService.swift

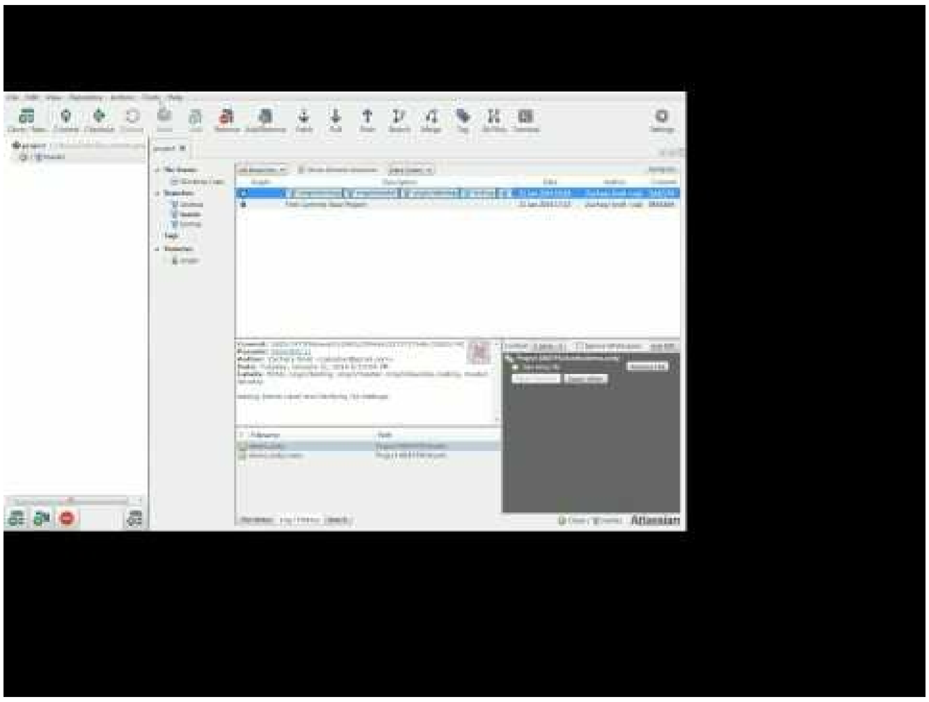
Hunk 1 : Lines 85-119

```
                            user.userEmail = user_email
                        }
                    }
-
-                   if let userConnections = userData["user_connection
-                       user.userConnectionIDs = userConnections
-                   }
+
+                   completion(result: true)
+                   return
+               }
+               completion(result: false)
+           }
+       }
+
+       func getUserBattles(user:UserModel, completion:(result:Bool) -
+           var url = self.constants.serverURL + "get_active_battles"
+           var params = Dictionary<String, String>()
+
+           if let userID = user.userID {
+               params["user_id"] = userID
```

This is a color-coded view with green describing additions, pink describing deletions and white describing no changes.

[Video] Git + SourceTree Tutorial

The following video provides an introduction to Git using SourceTree:



https://www.youtube.com/watch?v=gdJ01t3KYH0

SourceTree Resources

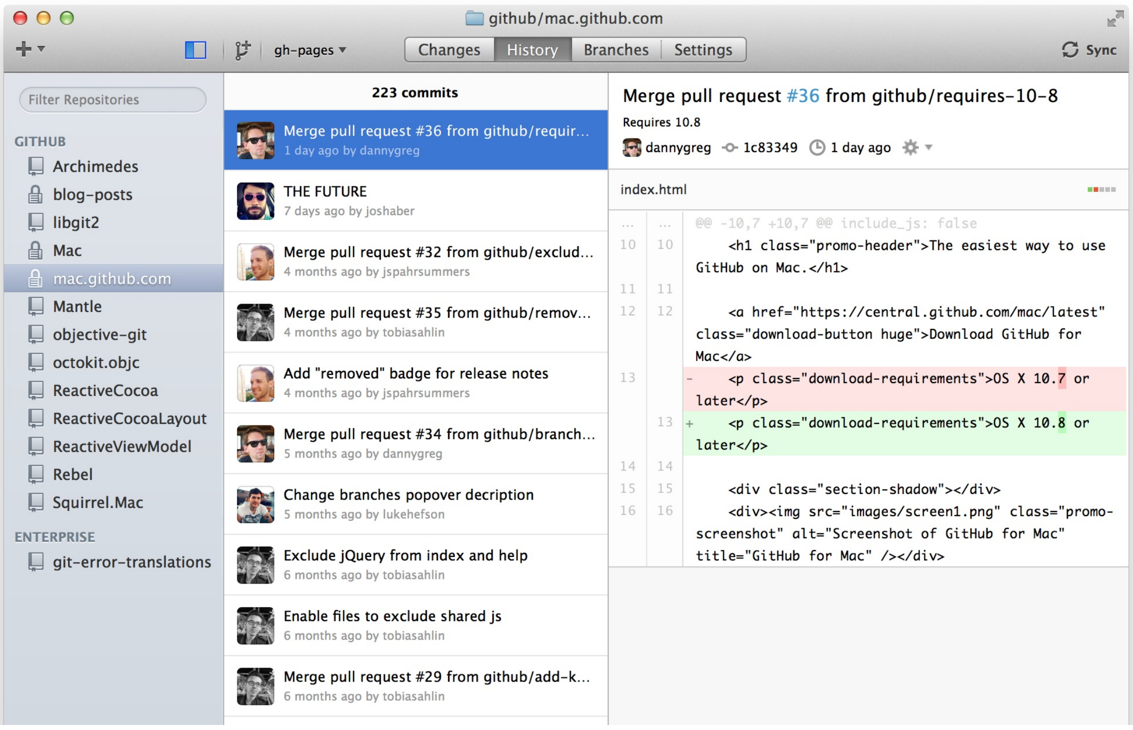
FAQ from SourceTree - http://www.sourcetreeapp.com/faq/

Support for SourceTree - http://www.sourcetreeapp.com/support/

BitBucket? It is important to note that there are competitors to GitHub. Altassian is one of those competitors. In addition to this Mac app, they also make a web interface called BitBucket, which is Altassian's answer to GitHub.

GitHub Mac App

Download GitHub for Mac here: https://mac.github.com



Merge pull request #36 from github/requir…
1 day ago by dannygreg

THE FUTURE
7 days ago by joshaber

Merge pull request #32 from github/exclud…
4 months ago by jspahrsummers

Merge pull request #35 from github/remov…
4 months ago by tobiasahlin

Add "removed" badge for release notes
4 months ago by jspahrsummers

Merge pull request #34 from github/branch…
5 months ago by dannygreg

Change branches popover decription
5 months ago by lukehefson

Exclude jQuery from index and help
6 months ago by tobiasahlin

Enable files to exclude shared js
6 months ago by tobiasahlin

Merge pull request #29 from github/add-k…
6 months ago by tobiasahlin

Merge pull request #36 from github/requires-10-8
Requires 10.8
dannygreg 1c83349 1 day ago

```
@@ -10,7 +10,7 @@ include_js: false
    <h1 class="promo-header">The easiest way to use
GitHub on Mac.</h1>

    <a href="https://central.github.com/mac/latest"
class="download-button huge">Download GitHub for
Mac</a>
-    <p class="download-requirements">OS X 10.7 or
later</p>
+    <p class="download-requirements">OS X 10.8 or
later</p>

    <div class="section-shadow"></div>
    <div><img src="images/screen1.png" class="promo-
screenshot" alt="Screenshot of GitHub for Mac"
title="GitHub for Mac" /></div>
```

The GitHub for Mac client provides a easy-to-use interface for Git and GitHub. GitHub has a fantastic overview of their product which we will outline below:

1. **First Launch** - Just getting started with a fresh download of GitHub for Mac? https://help.github.com/articles/first-launch/
2. **Working with Repositories** - Want to fork your first repository? Use this resource: https://help.github.com/articles/working-with-repositories/
3. **Making Changes** - View your source, commit locally and sync your changes with the server - https://help.github.com/articles/making-changes/
4. **Viewing History** - Want to see what's changed and when? https://help.github.com/articles/viewing-previous-commits/
5. **Branches** - Create a branch after a successful clone - https://help.github.com/articles/branching-out/
6. **Merging branches** - Once you're ready to merge your branch into another branch, check out - https://help.github.com/articles/merging-branches/

Mac App Resources

Keyboard shortcuts and further informative resources can be found here: https://mac.github.com/help.html

Common Terms

After using Git for a number of years, developers have created a language around the tool and their process. This list can be used as a quick reference to decipher those messages and assimilate this language into your own.

Terms

- **Add** - Creating a new file within the repository
- **Blame** - A term used to find who changed what line of code and when.
- **Branch** - describes a separate line of development from another branch
- **Checkout** - retrieve the latest copy of a repository into your local branch
- **Clone** - to initially copy a remote repository onto your computer.
- **Commit** - to take your changes and submit them to the repository
- **Conflict** - when two commits affect the same code and Git has trouble automatically merging the two
- **Git** - pronounced with a hard "G", rhymes with spit
- **GitHub** - A company in San Francisco that made an excellent interface to Git repositories
- **Local** or **Local changes** - a cloned copy of a repository that may have changes not yet pushed to the remote repository.
- **Master** - Typically, the base branch of which all other brances are derived
- **Merge** - Take a series of commits from one branch and put them into another branch
- **Merge conflict** - Same as conflict
- **Publish** - To submit your changes to the remote repository. May also refer to the creation of a branch on the remote repository.
- **Push** - Same as publish without the creation connotation.
- **Pull** - To retrieve changes on a remote repository into your own local repo.
- **Repo** or **Repository** - Where the source code lives. For our purposes, this is GitHub
- **Stage** - The precursor to commit. Observe changes about to be commited into your branch before those changes go into the local copy of your repo.
- **Stash** - To quickly push aside your changes into a temporary state which can be retrieved later or abandoned.
- **Version Control** - A system that handles keeping track of changes and provides automation around merging changes from multiple streams of input, such as people.

References

GitHub Glossary - https://help.github.com/articles/github-glossary/

Git Manual (*Warning: Heavy material, approach with coffee*) http://gitref.org

Simple guide to forks in GitHub and Git http://www.dataschool.io/simple-guide-to-forks-in-github-and-git/

Stack Overflow Q&A - http://stackoverflow.com/questions/7076164/terminology-used-by-git