

Using GitHub for Rally Apps (Mac Version)

PILOT DOCUMENTATION

[SOURCE DOCUMENT](#) (must have a rallydev.com email address to load and edit)

Introduction

Rally is conducting a pilot that leverages public repositories in GitHub to enable customer collaboration in creating and evolving apps.

The purpose of this document is to help you get set up to participate in this pilot, as well as to understand the relevant workflows. Note that this document covers the setup and workflows for the Mac OSX user. If you will be using a Windows machine, please consult the **Using GitHub for Rally apps (Windows Version)** instead.

This document is divided into two sections: **Getting Started**, and **Understanding Developer GitHub Workflows with Rally Apps**:

- In the **Getting Started** section, you'll create a GitHub account, and install the necessary software on your machine.
- In the **Understanding Developer GitHub Workflows with Rally Apps** section, you'll learn to use GitHub and your installed software to support the development workflows in the pilot: **maintaining an existing app**, and **creating a new app**

So, let's get started!

Getting Started

To access the public repositories, you'll need to have a GitHub account. In addition, there are two software applications you'll need to install on your Mac in order to interact with the GitHub repositories. These applications are named Git, and GitHub for Mac.

Creating a GitHub Account

We recommend you use a single account for all of your interaction with GitHub. So if you already have a personal GitHub account, use it. Repositories will be distinctly associated with either your personal or employer(s) work. Also, your single GitHub account can be given "owner

team" status on a work account.

To create a GitHub account, visit <http://github.com> and choose the **Signup and Pricing** menu item, followed by clicking the **Create a free account** button. Fill out the form, and click the **Create an account** button.

Installing and Configuring Git

Now that you have a GitHub account, you'll need to install and configure an application named Git on your machine. Git is an open source version control system that runs on your local machine, and communicates with the GitHub repositories. To download and install Git, follow the instructions provided on the Set Up Git page on the GitHub site:

<http://help.github.com/mac-set-up-git/>

Installing GitHub for Mac

As you are aware from installing and configuring Git, it has a command-line interface. It does not, however, have a graphical user interface. To make the process of interacting with the repositories easier, you'll want to download and install the application named GitHub for Mac. To do so, follow the instructions on the GitHub for Mac site: <http://mac.github.com/>

Note: You won't need it to work with Rally repositories but if you are curious, additional information on using Git and its command-line interface can be found at: <http://progit.org/book/>

Congratulations, you are now ready to participate in the development workflows of this pilot. The following section covers these workflows, walking you through the steps required in each.

Understanding Developer GitHub Workflows with Rally Apps

There are two main workflows in the GitHub for Rally Apps pilot. They are:

1. maintaining an existing app
2. creating a new app

Let's take a look at the **maintaining an existing app** workflow first:

Maintaining an Existing App

This workflow is appropriate when you want to make an update to an existing app to which you don't have write access. The idea here is that you'll make your modifications in your copy of the main repository for the app, and then submit a request to the maintainer of the main repository for your modifications to be applied. Here are the steps that you'll follow:

1. Fork the repository that contains the app
2. Clone the new repository to your local machine
3. Modify the application
4. Commit and sync to your fork
5. Send a pull request for inclusion into the original repository

To make these steps easier to visualize, Figure 1 depicts them graphically:

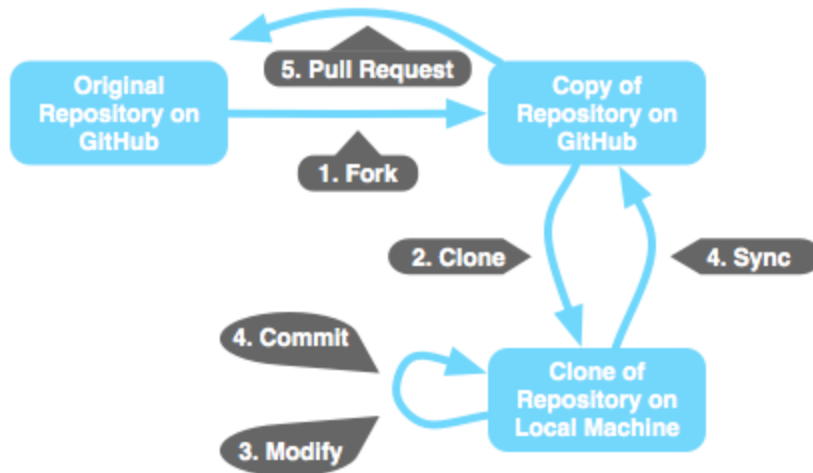


Figure 1. The *maintain an existing app* workflow

We'll elaborate on each of these steps, beginning with forking the original repository:

1. Forking the Original Repository

When you **fork** a GitHub repository, a copy of that repository is created on the GitHub server. Before performing the fork, you must first navigate to the repository that contains the app that you want to modify. You can do this in a number of ways, such as pasting a GitHub URL supplied by a Rally GitHub administrator into your browser. Note that all of the Rally approved apps will be located in the RallySoftware account on GitHub, which is located at the following URL:

<https://github.com/RallySoftware>

When you are on the desired repository page, click the **Fork** button as shown in Figure 2 below to perform the fork operation. [TODO: Change the screen shot after a Rally App has been added to the repository, remembering to highlight the Fork button]

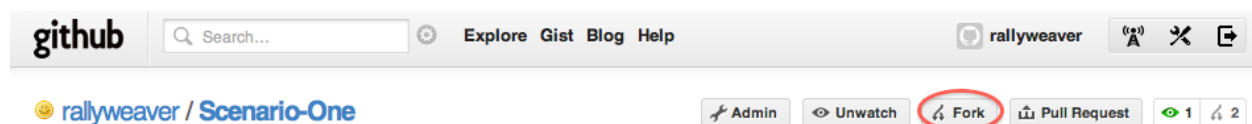


Figure 2. The **Fork** button on the repository page

If the dialog shown in Figure 3 appears, select your account as the one to which you want to fork the repository. [TODO: Change the screen shot after a Rally App has been added to the repository]

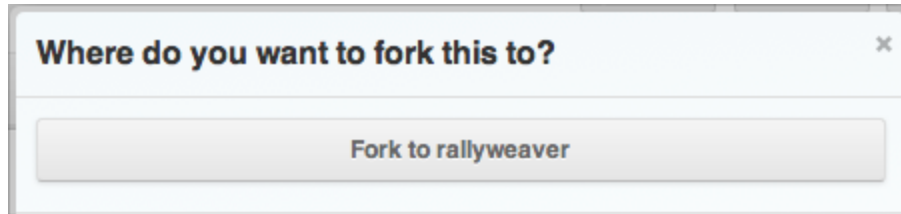


Figure 3. The **Fork** dialog on the repository page

Now that you've forked the repository, a copy of that repository exists on the GitHub server but is associated with your account. Note that you may navigate to this new repository in subsequent sessions by choosing it from the **Your Repositories** box on your GitHub home page as shown in Figure 4:

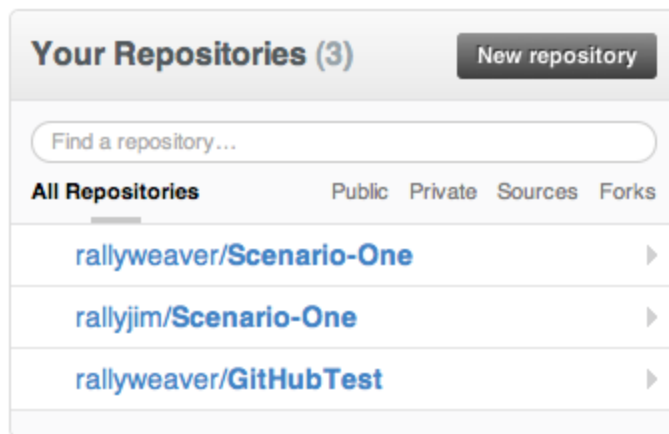


Figure 4. Selecting a repository from the **Your Repositories** box

For more information about forking a repository, consult the corresponding GitHub Help page: <http://help.github.com/fork-a-repo/>

The next step will be to clone the new repository on your local machine so that you can modify the application.

2. Cloning the New Repository

When you **clone** a repository, you are making a copy of the repository that will reside on your local machine. Note that there is a **Clone in Mac** button on the repository's page as shown in Figure 5.

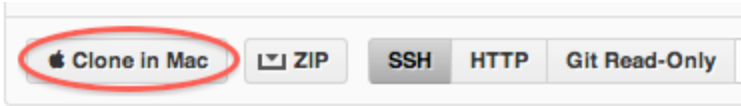


Figure 5. The **Clone in Mac** button on the repository page

After clicking the Clone in Mac button, GitHub for Mac should be invoked as shown in Figure 6, prompting you to specify the location in which you'd like the clone to be created. [TODO: Change the screen shot after a Rally App has been added to the repository]

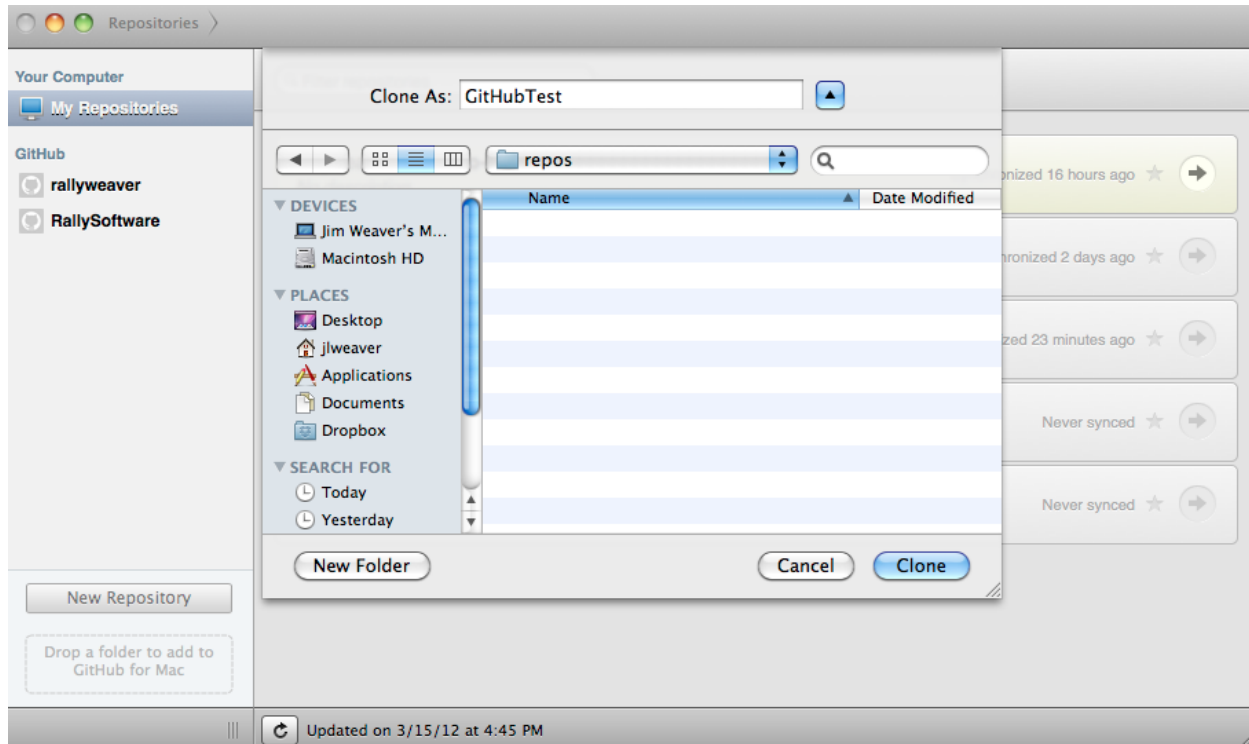


Figure 6. GitHub for Mac prompting for location to save the clone

A full copy of the repository will be created on your local machine at the location you specified.

3. Modify New Repository

Now that you have cloned the repository you can make modifications to the app, and continue on to the **commit and sync to the fork** step.

Note: As with any version control system, branching is supported by Git. For information on Git's very powerful branching capabilities, and ideas for implementing them in your project workflow, look here: <http://progit.org/book/ch3-0.html>

4. Committing and Syncing to the Fork

The modifications you made to the app exist on your local machine, but in order for the changes to exist on the GitHub server you must do two things: **Commit** your changes to the local repository, and **sync** the local repository to the remote repository on the GitHub server. As you'll see shortly, both of these operations may be performed in one step using GitHub for Mac.

We'll begin performing these operations by invoking GitHub for Mac. Figure 7 shows the GitHub for Mac startup screen that contains a list of local repositories. [TODO: Change the screen shot after a Rally App has been added to the repository]

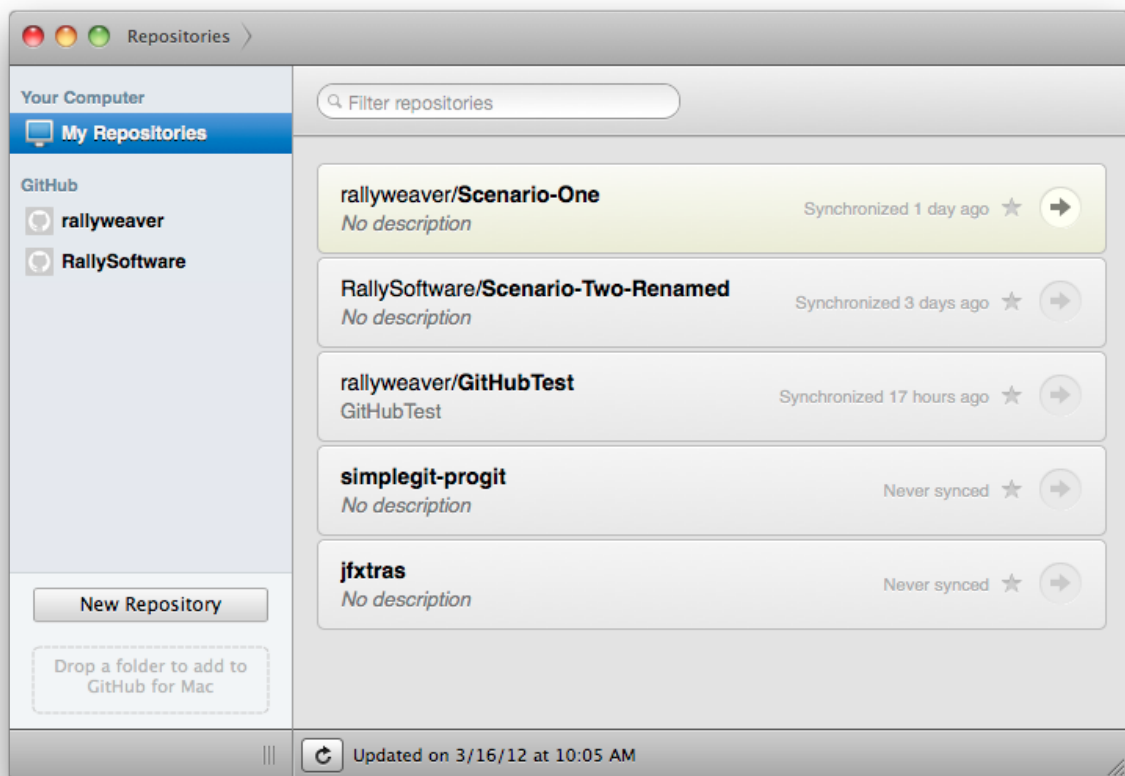


Figure 7. GitHub for Mac local repository list

We'll then double-click on the repository we'd like to commit and sync, which as shown in Figure 8 navigates to the **Changes** tab of the repository detail view. [TODO: Change the screen shot after a Rally App has been added to the repository]

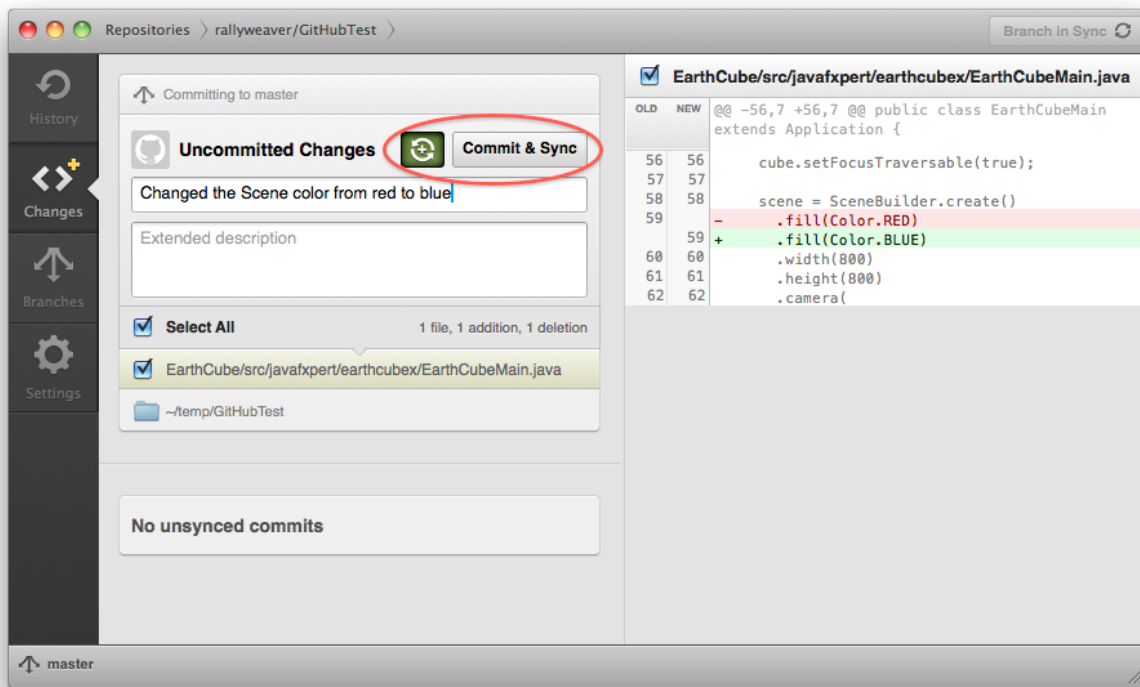


Figure 8. GitHub for Mac repository detail view

Notice in Figure 8 there is a button labeled **Commit & Sync**, and that clicking the button to its left causes this button's label to toggle between **Commit & Sync** and **Commit**. We're using the **Commit & Sync** setting so both operations are performed with one button click.

After typing in the **Commit Summary** message "Change the Scene color from red to blue", we'll click the **Commit & Sync** button which causes the changes to be committed to the local repository, and for the remote repository to be synchronized with the local repository. Now that the remote repository has the modifications, we'll need to make a request for the original repository to be updated as well by creating a **pull request**.

5. Sending a pull request for inclusion into the original repository

To send a pull request, we'll click the **Pull Request** button on the GitHub repository page shown in Figure 9. [TODO: Change the screen shot after a Rally App has been added to the repository]

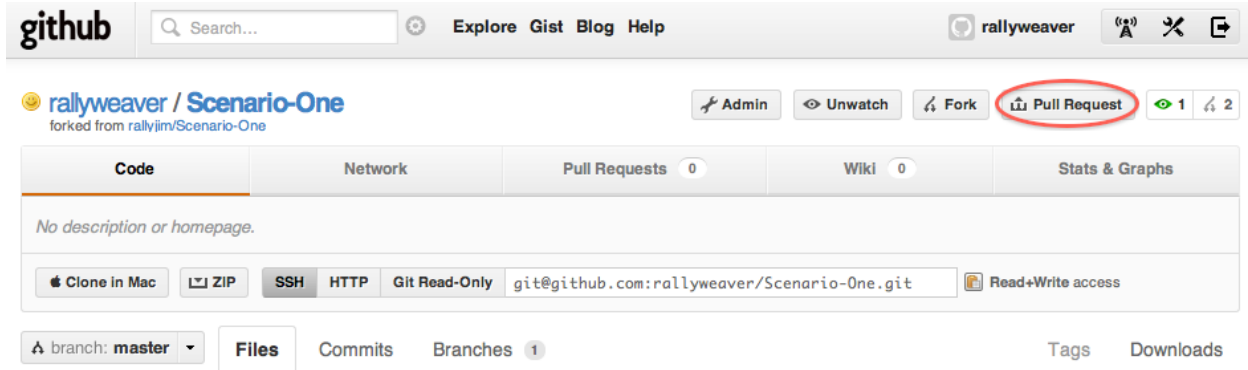


Figure 9. The **Pull Request** button on the GitHub repository page

As a result of clicking the **Pull Request** button, the page in Figure 10 appears in which you can view and modify information about the pull request. For example, we entered the message “This modification changes the Scene color to blue”. When ready to send the request, click the **Send pull request** button. [TODO: Change the screen shot after a Rally App has been added to the repository]

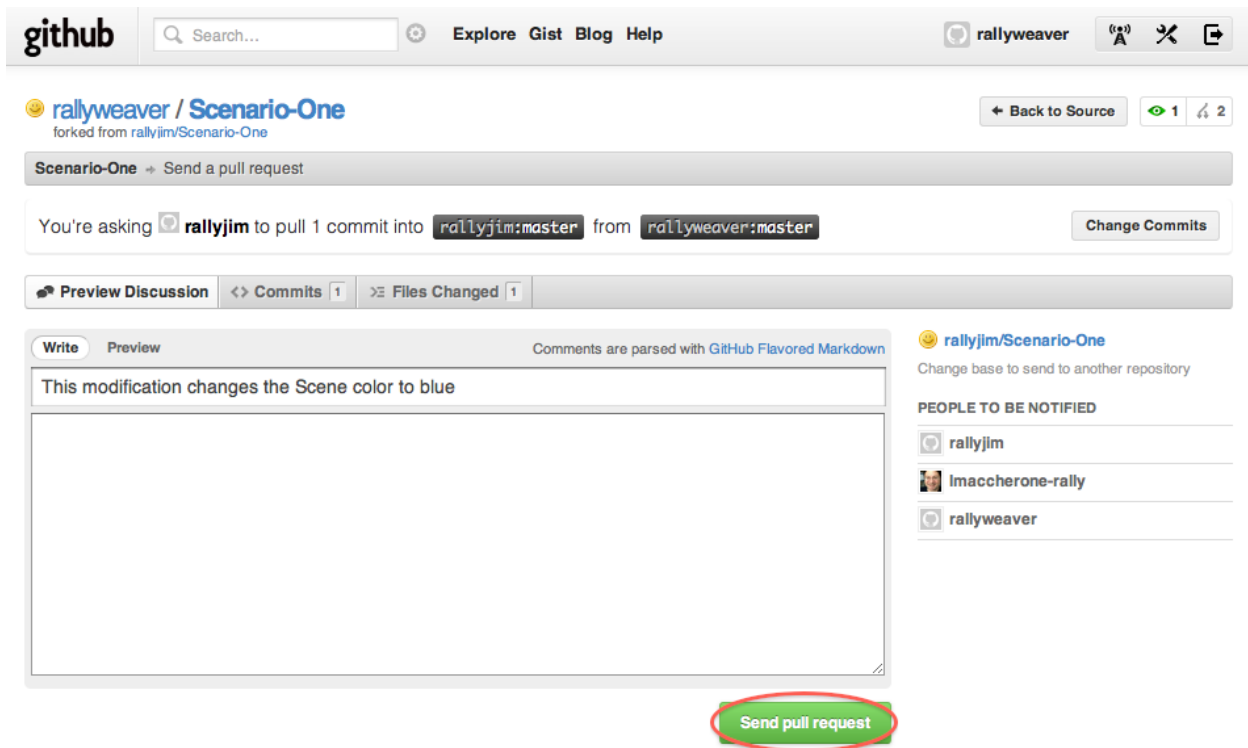


Figure 10. The GitHub pull request information page

As a result of sending the pull request, the repository maintainer will be prompted to review and merge the modifications into the original repository. For more information about pull requests, consult the corresponding GitHub Help page: <http://help.github.com/send-pull-requests/>

Now that we've examined the **maintaining an existing app** workflow, let's take a look at the **creating a new app** workflow:

Creating a New App

This workflow is appropriate when you want to create a new app that uses an existing app or **app template** as a starting point. The idea here is that you'll fork a copy of an existing app or app template that becomes the original repository for your new app. Modifications made to the application may be committed and synchronized directly to this original repository, eliminating the need for the pull request discussed in the **maintaining an existing app** workflow discussed earlier. Here are the steps that you'll follow:

1. Fork the repository that contains the app or app template
2. Rename the project
3. Clone the new repository to your local machine
4. Modify the application
5. Commit and sync to the fork
6. (optional) Submit a request for the new app to be added to the Rally Apps library

To make these steps easier to visualize, Figure 11 depicts them graphically:

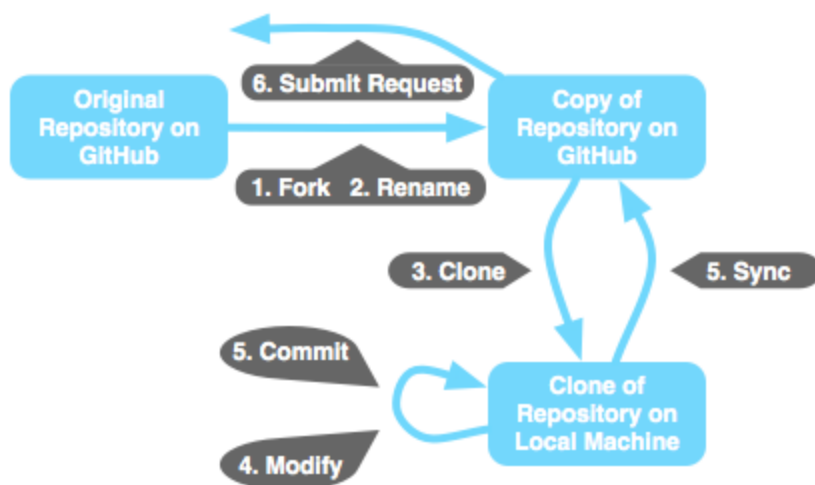


Figure 11. The **create a new app** workflow

Many of these steps are identical to the steps in the **maintaining an existing app** workflow, so some information previously covered will be repeated.

1. Forking the repository that contains the app or app template

When you **fork** a GitHub repository, a copy of that repository is created on the GitHub server. Before performing the fork, you must first navigate to the repository which contains the app you

want to modify. You can do this in a number of ways, such as pasting a GitHub URL supplied by a Rally GitHub administrator into your browser. Note that all of the Rally approved apps will be located in the RallySoftware account on GitHub, which is located at:

<https://github.com/organizations/RallySoftware>

When you are on the desired repository page, click the **Fork** button as shown in Figure 12 to perform the fork operation. [TODO: Change the screen shot after a Rally App has been added to the repository, remembering to highlight the Fork button]



Figure 12. The **Fork** button on the repository page

If the dialog shown in Figure 13 appears, select your account as the one to which you want to fork the repository. [TODO: Change the screen shot after a Rally App has been added to the repository]

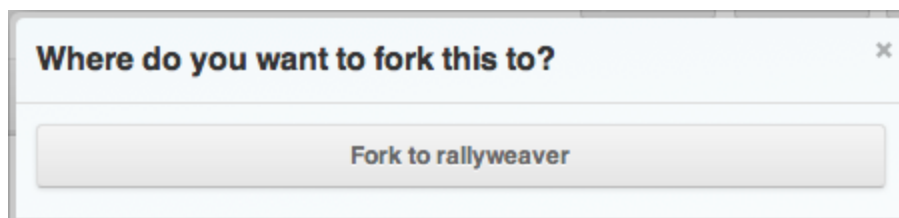


Figure 13. The **Fork** dialog on the repository page

Now that you've forked the repository, a copy of the repository exists on the GitHub server and it is associated with your account. Note that you may navigate to this new repository in subsequent sessions by choosing it from the **Your Repositories** box on your GitHub home page as shown in Figure 14:

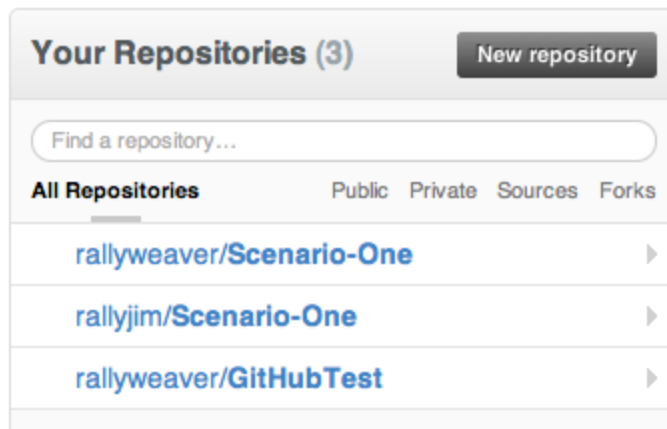


Figure 14. Selecting a repository from the **Your Repositories** box

For more information about forking a repository, consult the corresponding GitHub Help page: <http://help.github.com/fork-a-repo/>

2. Renaming the Project

Because you'll be creating a new app, you need to rename the new repository you just created (via the fork) in the previous step. To do this, click the **Admin** button on the GitHub repository page shown in Figure 15. [TODO: Change the screen shot after a Rally App has been added to the repository]

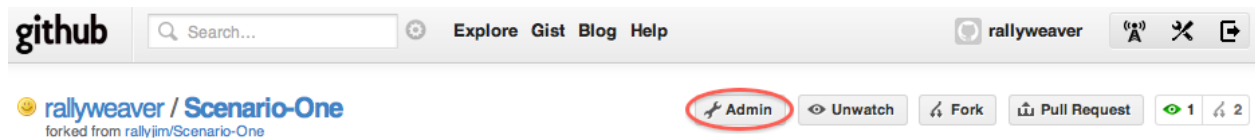


Figure 15. The **Admin** button on the GitHub repository page

Then, from the GitHub repository admin page shown in Figure 16, enter a new **Repository Name** and click the **Rename** button. [TODO: Change the screen shot after a Rally App has been added to the repository]

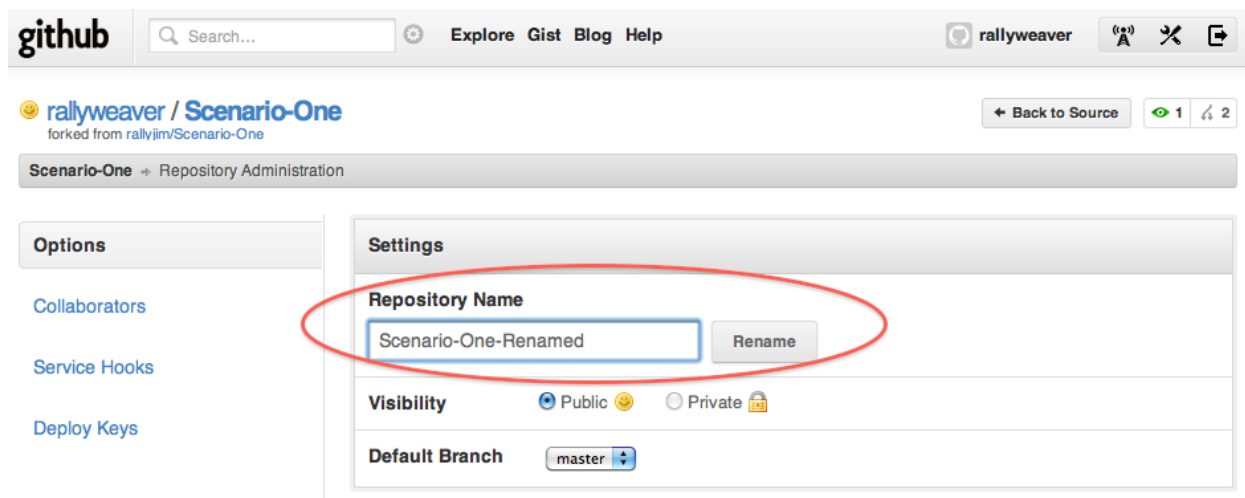


Figure 16. The GitHub repository admin page

3. Cloning the New Repository

When you **clone** a repository, you are making a copy of the repository which will reside on your local machine. Note that there is a **Clone in Mac** button on the repository's page as shown in Figure 17.

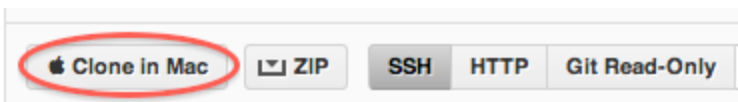


Figure 17. The **Clone in Mac** button on the repository page

After clicking the **Clone in Mac** button, GitHub for Mac should be invoked as shown in Figure 18, prompting you to specify the location (on the local machine) in which you'd like the clone to be created. [TODO: Change the screen shot after a Rally App has been added to the repository]

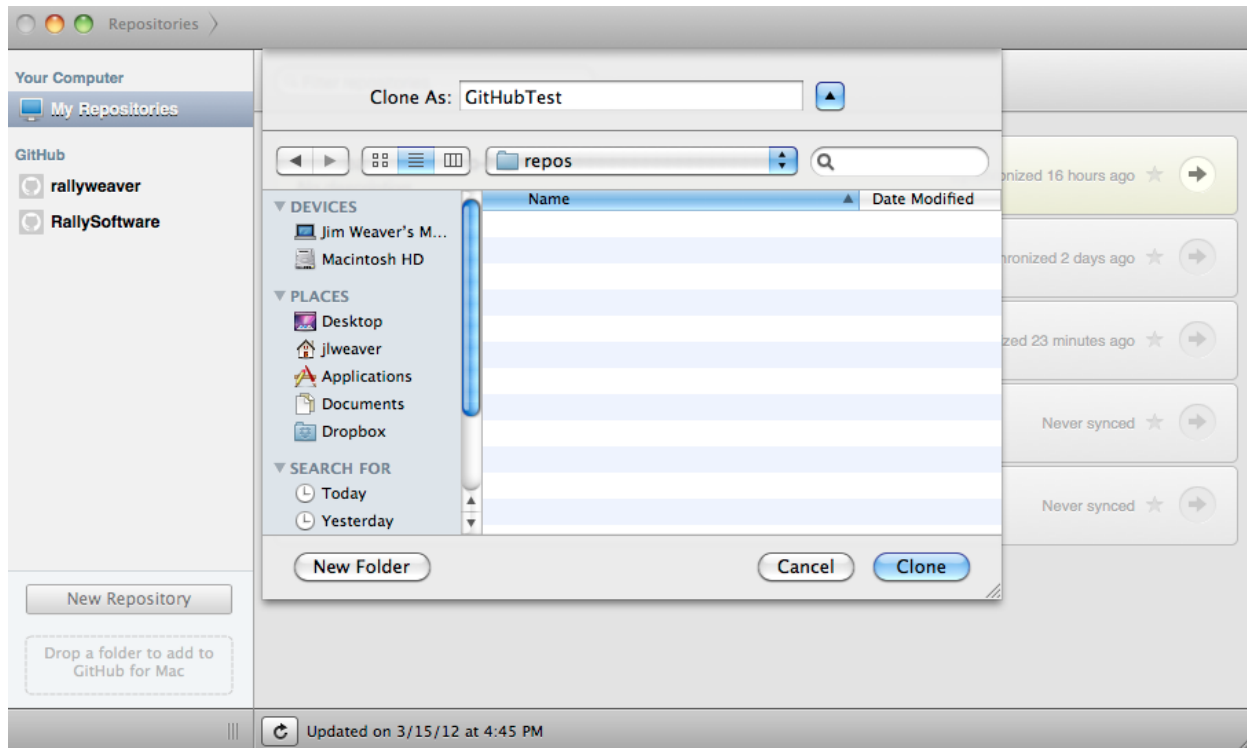


Figure 18. GitHub for Mac prompting for location to save the clone

A full copy of the repository will be created on your local machine at the location you specified. Now that you have cloned the repository you can make modifications to the app, and continue on to the **committing and syncing to the fork** step.

5. Committing and Syncing to the Fork

The modifications you made to the app exist only on your local machine, and in order for the changes to exist on the GitHub server you need to do two things: **Commit** your changes to the local repository, and **sync** the local repository to the remote repository on the GitHub server. As you'll see shortly, both of these operations may be performed in one step using GitHub for Mac.

We'll begin performing these operations by invoking GitHub for Mac. Figure 19 shows the GitHub for Mac startup screen which contains a list of local repositories. [TODO: Change the screen shot after a Rally App has been added to the repository]

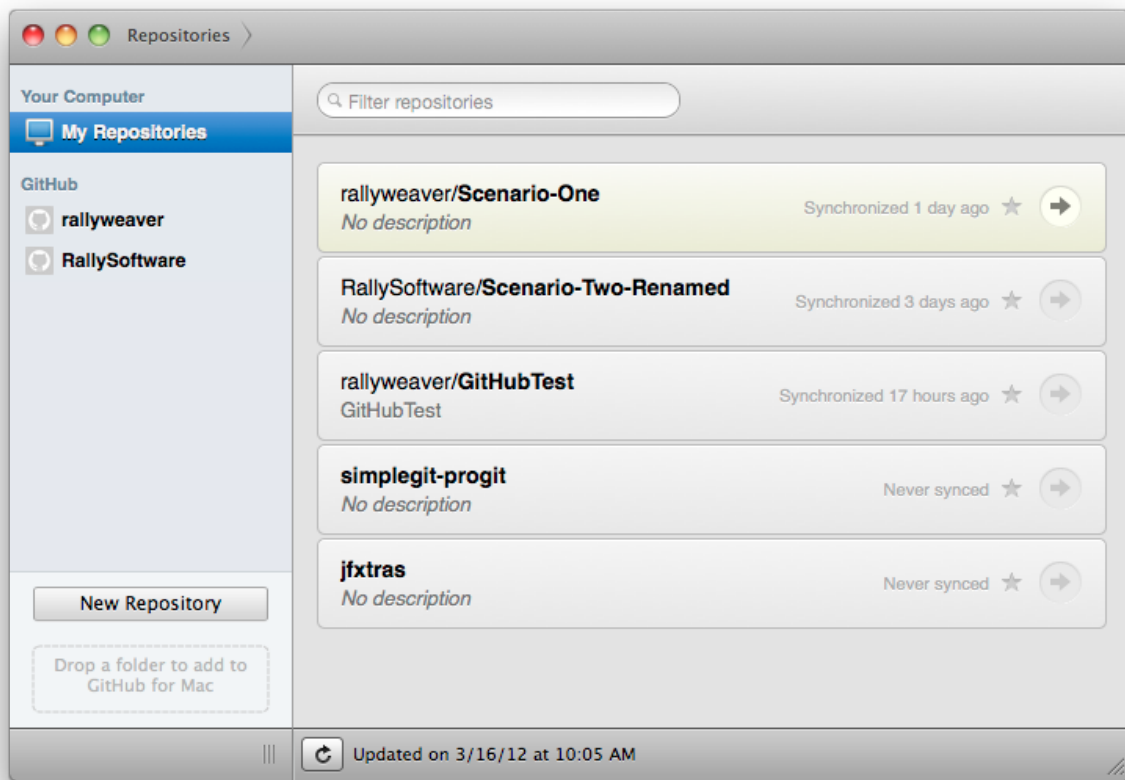


Figure 19. GitHub for Mac local repository list

We'll then double-click on the repository we'd like to commit and sync which, as shown in Figure 20, navigates to the **Changes** tab of the repository detail view. [TODO: Change the screen shot after a Rally App has been added to the repository]

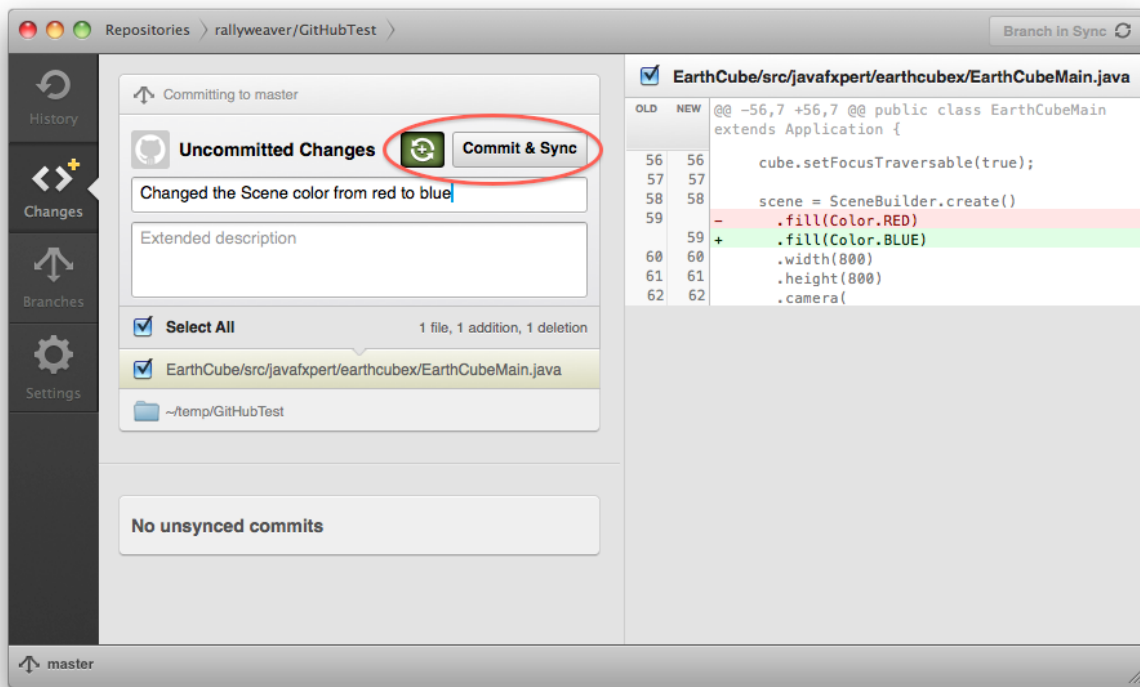


Figure 20. GitHub for Mac repository detail view

Notice in Figure 20 the **Commit & Sync** button, and that clicking the button to its left causes this button's label to toggle between **Commit & Sync** and **Commit**. We're using the **Commit & Sync** setting so both operations are performed with one button click.

After entering the **Commit Summary** message "Change the Scene color from red to blue", we'll click the **Commit & Sync** button which causes the changes to be committed to the local repository, and for the remote repository to be synchronized with the local repository. Now that the remote repository has our modifications, we'll submit a request for the new app to be added to the Rally Apps library.

6. (optional) Submitting a Request for your App to be Added to the Rally Apps Catalog

To submit your App for possible inclusion into the Rally Apps catalog, send an email to app-submission@rallydev.com. Be sure to include your contact information and a link to the repository for your App.

Summary

In this document, we've shown how to participate in the GitHub Rally for Apps from a developer's perspective. Specifically, we've shown how to:

- Create a **GitHub** account
- Install and configure **Git**
- Install **GitHub for Mac**
- Use GitHub and the installed software to implement the **two development workflows** in the pilot, which are: **maintaining an existing app**, and **creating a new app**.

We hope you've found this document helpful, and encourage you to contact us at github-for-apps@rallydev.com if you have any questions or feedback.