

Open Source SW & Lab - Summer 2023

8. GitHub: Maintaining

Walid Abdullah Al

Computer and Electronic Systems Engineering
Hankuk University of Foreign Studies

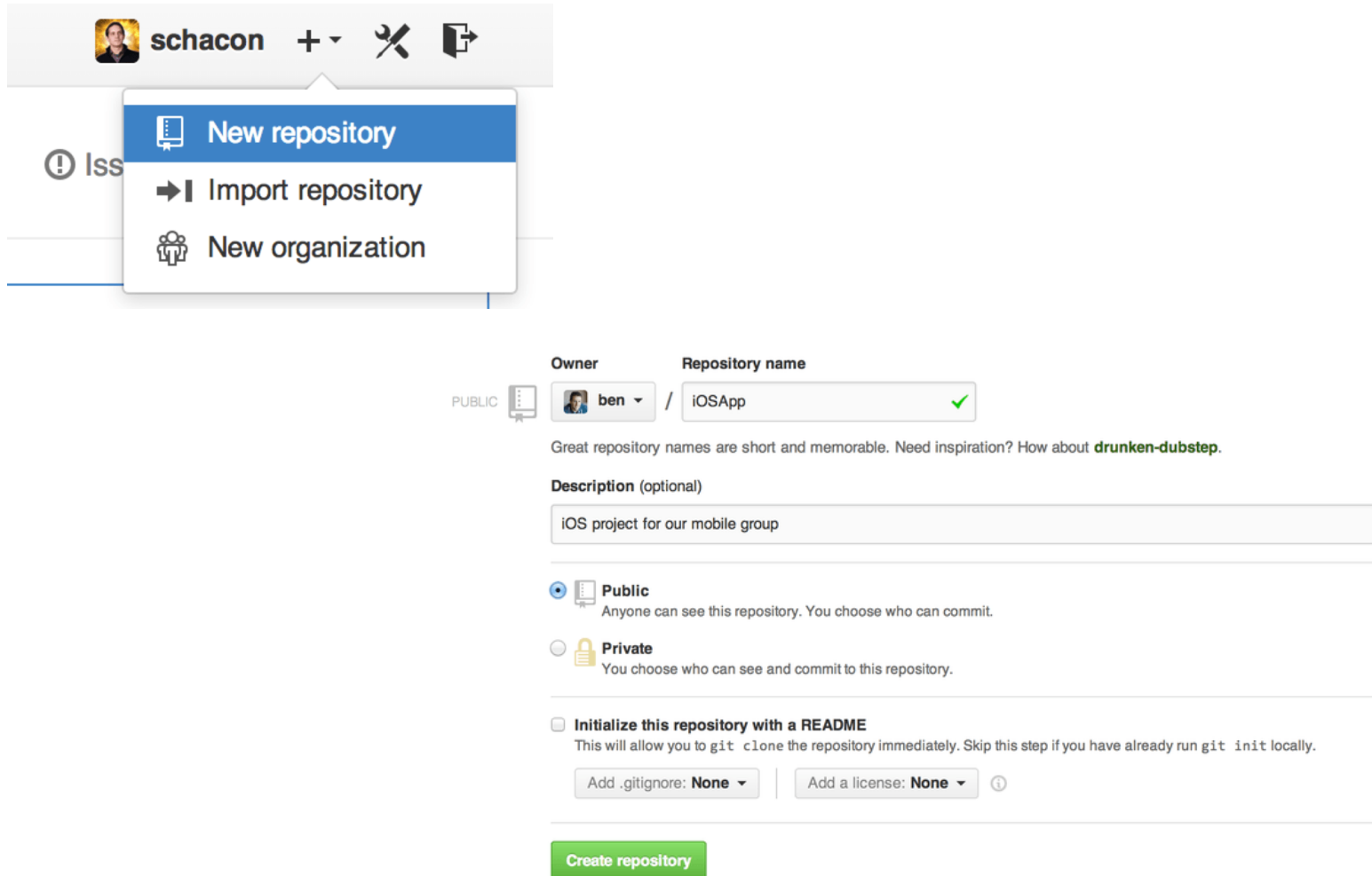


Computer Vision Lab
Hankuk University of Foreign Studies

Based on:

Pro Git (2022) by Scott Chacon, Ben Straub



GitHub: Create a new repository



schacon + ▾

- New repository
- ➔ Import repository
- 👤 New organization

Owner **Repository name**

PUBLIC   ben / iOSApp ✓

Great repository names are short and memorable. Need inspiration? How about **drunken-dubstep**.

Description (optional)

iOS project for our mobile group

☒ **Public**
Anyone can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

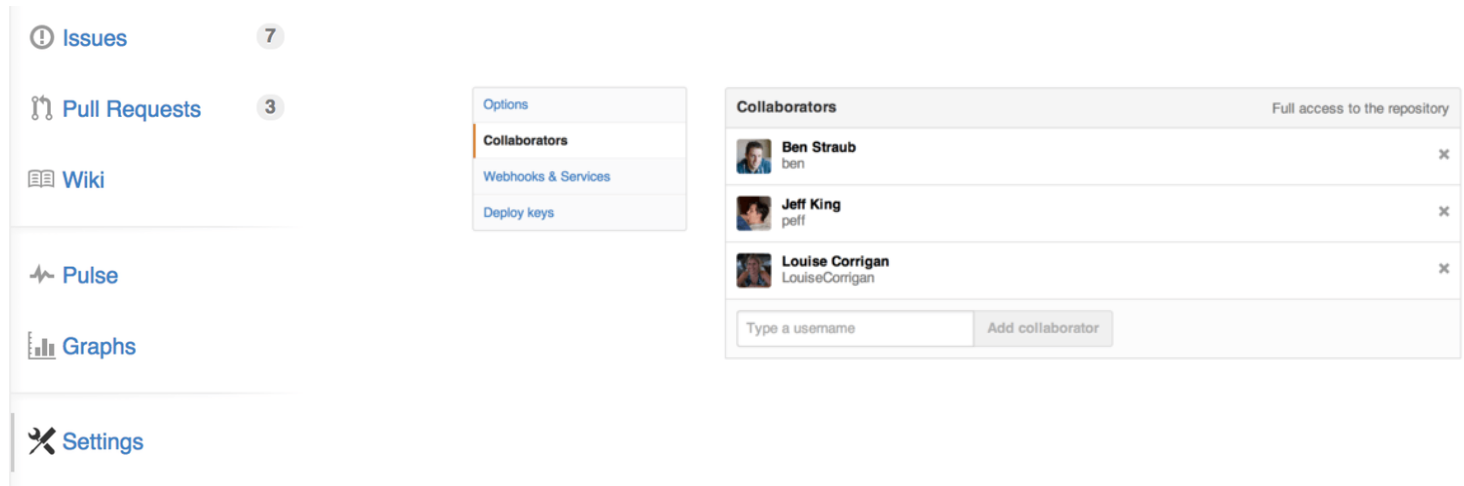
☐ **Initialize this repository with a README**
This will allow you to `git clone` the repository immediately. Skip this step if you have already run `git init` locally.

Add .gitignore: **None** ▾ | Add a license: **None** ▾ ⓘ




Create repository

Adding Collaborator

- **With read/write access**



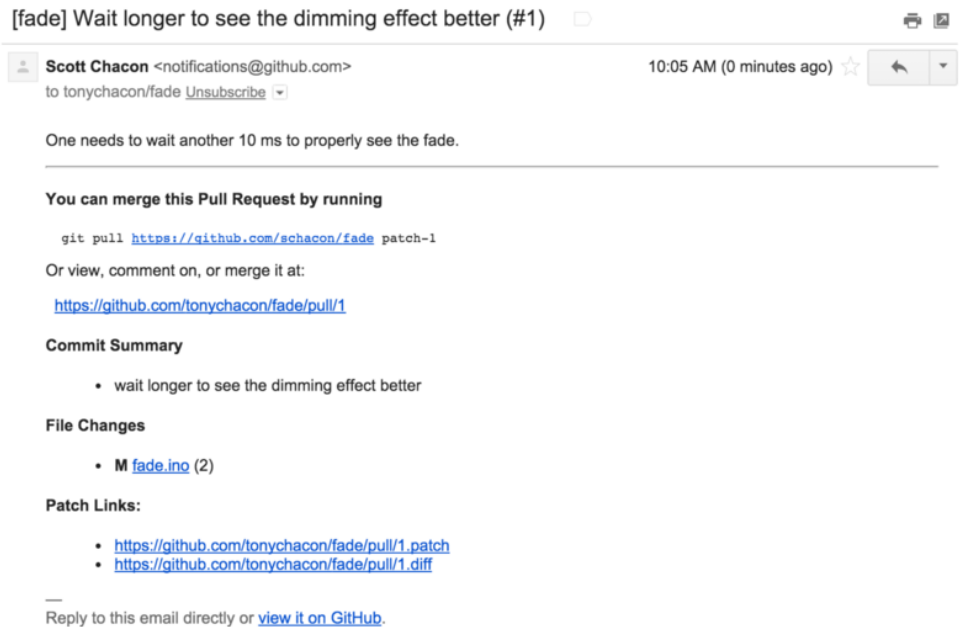
The screenshot displays the GitHub repository settings interface. On the left, a sidebar contains navigation links: Issues (7), Pull Requests (3), Wiki, Pulse, Graphs, and Settings. The main content area shows a dropdown menu with 'Options', 'Collaborators' (selected), 'Webhooks & Services', and 'Deploy keys'. The 'Collaborators' section is titled 'Collaborators' with a subtitle 'Full access to the repository'. It lists three collaborators: Ben Straub (username: ben), Jeff King (username: peff), and Louise Corrigan (username: LouiseCorrigan). Each entry includes a profile picture, name, username, and a removal icon (X). At the bottom, there is a text input field labeled 'Type a username' and an 'Add collaborator' button.

Collaborators		Full access to the repository
	Ben Straub ben	X
	Jeff King peff	X
	Louise Corrigan LouiseCorrigan	X


Type a username Add collaborator

Managing Pull Request

- **Pull request**
 - From a branch in a fork repository
 - Or, from another branch in the same repository
- **Direct merge**
 - `git pull <url> patch-1`
- **Merge locally**
 - Fetch, merge, push
- **Merge on GitHub**





Safer merging



This pull request can be automatically merged.


You can also merge branches on the [command line](#).



Merging via command line


If you do not want to use the merge button or an automatic merge cannot be performed, you can perform a manual merge on the command line.

HTTP	Git	Patch
		<code>https://github.com/schacon/fade.git</code>




Step 1: From your project repository, check out a new branch and test the changes.

```
git checkout -b schacon-patch-1 master
git pull https://github.com/schacon/fade.git patch-1
```



Step 2: Merge the changes and update on GitHub.

```
git checkout master
git merge --no-ff schacon-patch-1
git push origin master
```



Pull request refs

- Useful if you have a lot of pull requests

```
$ git ls-remote https://github.com/schacon/blink
10d539600d86723087810ec636870a504f4fee4d      HEAD
10d539600d86723087810ec636870a504f4fee4d      refs/heads/master
6a83107c62950be9453aac297bb0193fd743cd6e       refs/pull/1/head
afe83c2d1a70674c9505cc1d8b7d380d5e076ed3       refs/pull/1/merge
3c8d735ee16296c242be7a9742ebfbc2665adec1       refs/pull/2/head
15c9f4f80973a2758462ab2066b6ad9fe8dcf03d       refs/pull/2/merge
a5a7751a33b7e86c5e9bb07b26001bb17d775d1a       refs/pull/4/head
31a45fc257e8433c8d8804e3e848cf61c9d3166c       refs/pull/4/merge
```

```
$ git fetch origin refs/pull/958/head
From https://github.com/libgit2/libgit2
* branch                refs/pull/958/head -> FETCH_HEAD
```

More at: <https://git-scm.com/book/en/v2/GitHub-Maintaining-a-Project>

Git-Tools

- **Now, you know all the basics you need for day-to-day collaboration**
- **Let's learn something of rare use but you will definitely need them at some point**
- **Starting with:**
 - Revision selection

Revision selection

- **You can refer to any commit using SHA-1 hash**

- Full hash
- Initial part of the hash

```
$ git show 1c002dd4b536e7479fe34593e72e6c6c1819e53b
$ git show 1c002dd4b536e7479f
$ git show 1c002d
```

```
$ git log
commit 734713bc047d87bf7eac9674765ae793478c50d3
Author: Scott Chacon <schacon@gmail.com>
Date:   Fri Jan 2 18:32:33 2009 -0800

    Fix refs handling, add gc auto, update tests

commit d921970aadf03b3cf0e71becdaab3147ba71cdef
Merge: 1c002dd... 35cfb2b...
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Dec 11 15:08:43 2008 -0800

    Merge commit 'phedders/rdocs'
```

- **Git can figure out a short unique abbreviation**

```
$ git log --abbrev-commit --pretty=oneline
ca82a6d Change the version number
085bb3b Remove unnecessary test code
a11bef0 Initial commit
```

- **Or, you can simply use branch name (if possible)**

How to checkout a certain commit

- **Just to checkout**

```
# This will detach your HEAD, that is, leave you with no branch checked out:  
git checkout 0d1d7fc32
```

- **Checkout and commit**

- Create a branch then commit

```
git checkout -b old-state 0d1d7fc32
```

Interactive Staging

- Have done a lot of edits
- Now, want to stage partially and sequentially
- Interactive staging can help

```
$ git add -i

      staged      unstaged path
1:    unchanged    +0/-1 TODO
2:    unchanged    +1/-1 index.html
3:    unchanged    +5/-1 lib/simplegit.rb

*** Commands ***
1: [s]tatus      2: [u]pdate      3: [r]evert      4: [a]dd untracked
5: [p]atch       6: [d]iff        7: [q]uit        8: [h]elp
What now>
```

Staging and unstaging files

```
What now> u
```

	staged	unstaged path
1:	unchanged	+0/-1 TODO
2:	unchanged	+1/-1 index.html
3:	unchanged	+5/-1 lib/simplegit.rb

```
Update>>
```

```
Update>> 1,2
```

	staged	unstaged path
* 1:	unchanged	+0/-1 TODO
* 2:	unchanged	+1/-1 index.html
3:	unchanged	+5/-1 lib/simplegit.rb

```
Update>>
```

```
Update>>
```

```
updated 2 paths
```

```
*** Commands ***
```

1: [s]tatus	2: [u]pdate	3: [r]evert	4: [a]dd untracked
5: [p]atch	6: [d]iff	7: [q]uit	8: [h]elp

```
What now> s
```

	staged	unstaged path
1:	+0/-1	nothing TODO
2:	+1/-1	nothing index.html
3:	unchanged	+5/-1 lib/simplegit.rb

Unstage using revert

```
*** Commands ***
 1: [s]tatus      2: [u]pdate      3: [r]evert      4: [a]dd untracked
 5: [p]atch       6: [d]iff      7: [q]uit        8: [h]elp
What now> r
      staged      unstaged path
 1:      +0/-1      nothing TODO
 2:      +1/-1      nothing index.html
 3:    unchanged      +5/-1 lib/simplegit.rb
Revert>> 1
      staged      unstaged path
* 1:      +0/-1      nothing TODO
 2:      +1/-1      nothing index.html
 3:    unchanged      +5/-1 lib/simplegit.rb
Revert>> [enter]
reverted one path
```

```
*** Commands ***
 1: [s]tatus      2: [u]pdate      3: [r]evert      4: [a]dd untracked
 5: [p]atch       6: [d]iff      7: [q]uit        8: [h]elp
What now> s
      staged      unstaged path
 1:    unchanged      +0/-1 TODO
 2:      +1/-1      nothing index.html
 3:    unchanged      +5/-1 lib/simplegit.rb
```

Review the staged changes

```
*** Commands ***
 1: [s]tatus      2: [u]pdate      3: [r]evert      4: [a]dd untracked
 5: [p]atch       6: [d]iff       7: [q]uit         8: [h]elp
What now> d
      staged      unstaged path
 1:      +1/-1      nothing index.html
Review diff>> 1
diff --git a/index.html b/index.html
index 4d07108..4335f49 100644
--- a/index.html
+++ b/index.html
@@ -16,7 +16,7 @@ Date Finder

<p id="out">...</p>

-<div id="footer">contact : support@github.com</div>
+<div id="footer">contact : email.support@github.com</div>

<script type="text/javascript">
```

Stage patches using [p] option

- To stage part of file-changes

```
diff --git a/lib/simplegit.rb b/lib/simplegit.rb
index dd5ecc4..57399e0 100644
--- a/lib/simplegit.rb
+++ b/lib/simplegit.rb
@@ -22,7 +22,7 @@ class SimpleGit
   end

   def log(treeish = 'master')
-    command("git log -n 25 #{treeish}")
+    command("git log -n 30 #{treeish}")
   end

   def blame(path)
     Stage this hunk [y,n,a,d/,j,J,g,e,?]?

```

Today's team assignment

- **Decide your project topic.**
- **Elect your maintainer/leader.**
- **Maintainer's task:**
 - Creates a Github repository for the team project
 - with a readme.md file
 - Insert your project-title in the readme.md file and commit.
 - Configure the repository settings (based on your preferred workflow)
- **All member's task:**
 - Create topic branch(/branches) to:
 - add list/short-description of features/components/specs/requirements of your software.
 - Push (and ...)
- **Maintainer:**
 - Finalizes/merges all the specs
 - Distributes the tasks after discussing with all team members