Bonus round: putting an existing Unity project into git

First up! We need to make sure our Unity project is configured correctly. In the project, go to Edit > Project Settings > Editor and check that your settings look like this:

unityeditorsettings.png

Open up the command line. This will be Terminal on Mac OS, on Windows you should use Git Bash.

If you're not familiar with the command line, the way to use it is to type a command, and then press Enter to run that command.

Navigate to your Unity project folder using the 'cd' command - cd stands for "change directory".

The command looks like this: `cd "path/to/your/unity/project"` but you need to replace "path/to/your/unity/project" with the actual filepath to the top level folder of your project!

If you don't know the path, you can copy it from Finder or Windows Explorer. On a Mac, hold Option and Right-click the folder and select "Copy as Pathname".

On Windows hold Shift and Right-click and choose the command "Copy as Path".

You can paste the path directly into the command line after typing "cd". If you're on Windows, you can't use the normal Ctrl+C shortcut in Git Bash - use Shift+Insert instead.

Then we can set up our repository with this command:

`git init`

Now there should be a folder called .git in your project. You may need to show hidden files to see it (Shift+Option+. on Mac, in Windows Explorer click 'View' and select the "Hidden Items" checkbox).

Type `git status` and press Enter - you should get a list of all the folders git can see in your project. None of these have been added to the repository though.

Don't commit anything yet! Unity has a lot of weird crap in it which we don't want to keep track of. We need a gitignore file, we're going to create an empty one using this command.

`touch .gitignore`

Here's the .gitignore content we want to use - this is the list GitHub uses. Open your .gitignore file in a text editor (such as NotePad on Windows or TextEdit on Mac) and copy-paste this.

IMPORTANT: On Windows your .gitignore file will look like a text file with no name in the top level folder of your project.

```
[Ll]ibrary/
[Tt]emp/
[Oo]bj/
[Bb]uild/
[Bb]uilds/
Assets/AssetStoreTools*

# Visual Studio 2015 cache directory
/.vs/

# Autogenerated VS/MD/Consulo solution and project files
ExportedObj/
.consulo/
*.csproj
*.unityproj
*.sln
*.suo
*.tmp
*.user
*.userprefs
*.pidb
*.booproj
*.svd
*.pdb

# Unity3D generated meta files
*.pidb.meta
*.pdb.meta

# Unity3D Generated File On Crash Reports
sysinfo.txt

# Builds
*.apk
*.unitypackage
```

if you're working on Mac, you'll want to add this to it too:

```
# ============ #
# OS generated #
# ============ #
.DS_Store
.DS_Store?
._*
.Spotlight-V100
.Trashes
Icon?
ehthumbs.db
Thumbs.db
```

Don't forget to save your text file before closing! Now we're ready to start adding things to the repository.

`git add .gitignore`

This stages the gitignore file to be committed. We need to commit this before anything else, or git won't know what to ignore.

If you type `git status` it should show you that .gitignore is staged to commit, but your other files aren't.

`git commit -m "Adding gitignore"`

You can use a different commit message if you like but this one is pretty clear.

Try `git status` again. It shouldn't show you the Library folder in the list of unstaged changes now. So you're safe to commit everything else, like this:

`git add -A`

This command will 'stage' all of your files. This means it marks them as ready for commit.

`git commit -m "Initial project commit"`

This command will commit all of the files you staged in the previous step from your working copy into your local repository.

Revision #12
Created 15 February 2018 19:38:01
Updated 19 January 2020 22:19:18