# CS50's
# Introduction to Game Development

OpenCourseWare

Colton Ogden (https://www.linkedin.com/in/colton-ogden-0514029b/)
cogden@cs50.harvard.edu

David J. Malan (https://cs.harvard.edu/malan/)
malan@harvard.edu
(https://www.facebook.com/dmalan) (https://github.com/dmalan)
(https://www.instagram.com/davidjmalan/) (https://www.linkedin.com/in/malan/)
(https://orcid.org/0000-0001-5338-2522) (https://www.quora.com/profile/David-J-Malan) (https://www.reddit.com/user/davidjmalan)
(https://www.tiktok.com/@davidjmalan) (https://davidjmalan.t.me/)
(https://twitter.com/davidjmalan)

## Portal

### Objectives

- Create your own level in a new scene using ProBuilder and ProGrids!
- Ensure that the level has an `FPSController` to navigate with in the scene.
- Ensure that there is an object or region with a trigger at the very end that will trigger the end of the level (some zone with an invisible `BoxCollider` will work).
- When the level ends, display "You Won!" on the screen with a `Text` object.

### Demo

*by Edward Kang*

## Getting Started

Download the distro code for your game from cdn.cs50.net/games/2018/x/projects/10/portal.zip (https://cdn.cs50.net/games/2018/x/projects/10/portal.zip) and unzip `portal.zip`, which should yield a directory called `portal`.

Then, in a terminal window (located in `/Applications/Utilities` on Mac or by typing `cmd` in the Windows task bar), move to the directory where you extracted `portal` (recall that the `cd` command can change your current directory), and run

```
cd portal
```

Having some trouble with Unity? The staff has found that **Version 2018.4.28f1 (https://unity3d.com/unity/qa/lts-releases)** has worked well for them on a variety of different operating systems.

## Becoming a Pro

Welcome to your eleventh and final regular assignment! This assignment is going to be a fun conclusion to what's been a challenging but hopefully exciting term! Rather than build upon Portal in this example, and to afford you some extra time for your final project (and hopefully

save a little stress!), we're going to leverage some of Unity's brand-new tools to create a level! ProBuilder and ProGrids are a key feature that's changed the game for Unity, and having them makes creating game worlds (and more!) all the easier.

## Specification

- *Create your own level in a new scene using ProBuilder and ProGrids!* The distro should already have ProBuilder and ProGrids imported and ready for use, but just in case they aren't, you can easily find them by searching in the Asset Store (where they are now free, thanks to Unity having acquired them!). There are many resources for learning how to use ProGrids effectively, but two resources in particular that are worth checking out are here (https://www.youtube.com/watch?v=PUSOg5YEflM) and here (https://procore3d.github.io/probuilder2/), which should more than prepare you for creating a simple level.
- *Ensure that the level has an* `FPSController` *to navigate with in the scene.* This part's probably the easiest; just import an FPSController from the Standard Assets! It should already be imported into the project in the distro, where you can find the prefabs under `Assets > Standard Assets > Characters > FirstPersonCharacter > Prefabs`!
- *Ensure that there is an object or region with a trigger at the very end that will trigger the end of the level (some zone with an invisible* `BoxCollider` *will work).* This one should be easy as well, just relying on the creation of an empty GameObject and giving it a `BoxCollider` component, which you can then resize via its resize button in the component inspector!
- *When the level ends, display "You Won!" on the screen with a* `Text` *object.* Recall that `OnTriggerEnter` is the function you'll need to write in a script you also associate with the `BoxCollider` trigger, and ensure that the `BoxCollider` is set to a trigger in the inspector as well! Then simply program the appropriate logic to toggle on the display of a `Text` object that you also include in your scene (for an example on how to do this, just see the Helicopter Game 3D project, specifically the `GameOverText` script)!

## How to Submit

When you submit your project, the contents of your `games50/projects/2018/x/portal` branch must match the file structure of the unzipped distribution code exactly as originally received. That is to say, your files should not be nested inside of any other directories of your own creation or otherwise deviate from the file structure we gave you. Your branch should also not contain any code from any other projects, only this one. Failure to adhere to this file structure will likely result in your submission being rejected.

By way of a simple example, for this project that means that if the grading staff visits

`https://github.com/me50/USERNAME/tree/games50/projects/2018/x/portal/Assets/Scripts` (where `USERNAME` is your own GitHub username as provided in the form, below) we should be brought to the `Scripts` subdirectory for Portal. If that's not how your code is organized when you check (e.g., you get a 404 error or don't see your edits), reorganize your repository as needed to match this paradigm.

1. If you haven't done so already, visit this link (https://submit.cs50.io/invites/46e6f2ea29954ce9bb1bdc478a440055), log in with your GitHub account, and click **Authorize cs50**. Then, check the box indicating that you'd like to grant course staff access to your submissions, and click **Join course**.

The change to `/projects/2018` below is intentional, as CS50 courses have changed to a scheme that reflects when the project was initially released. So the 2018 here is correct, even though it's no longer 2018!

2. Using Git (https://git-scm.com/downloads), push your work to `https://github.com/me50/USERNAME.git`, where `USERNAME` is your GitHub username, on a branch called `games50/projects/2018/x/portal` or, if you've installed `submit50` (https://cs50.readthedocs.io/submit50/), execute

   ```
   submit50 games50/projects/2018/x/portal
   ```

   instead.
3. Record a screencast (https://www.howtogeek.com/205742/how-to-record-your-windows-mac-linux-android-or-ios-screen/), not to exceed 5 minutes in length (and not uploaded more than one month prior to your submission of this project) in which you demonstrate your app's functionality. Upload that video to YouTube (https://www.youtube.com/upload) (as unlisted or public, but not private) or somewhere else.
   - To aid in the staff's review, in your video's description on YouTube, you should timestamp where each of the following occurs *in your gameplay demonstration*. This is **not optional**; failure to do this will result in your submission being rejected:
     - "You win" (or similar) message shows
4. Submit this form (https://forms.cs50.io/7f172f11-a59f-4015-ae1d-fddd927b45fa).

You can then go to https://cs50.me/cs50g (https://cs50.me/cs50g) to view your current progress!