Coder

Description:

A coder glues the pieces together in a game engine, but this process generally stays within the boundaries provided by game designers and/or storytellers. Coders can expect to stare at strings of text and try to puzzle out functionality.

Tasks:

- Decide on a game engine
- Set up version control
- If multiple people will use version control, establish a workflow for doing so
- Map out code structure - classes, functions, how information will be passed around
- Write code to implement the design within a game engine
- Deploy Minimum viable product (MVP)

Resources:

- Unity Hub - Hub for downloading and managing your unity projects/installations
- Unity Learn - Free tutorials, courses, and guides to learn coding
- Unity API - Docs and guides to work with the Unity ecosystem
- Visual Studio Code - Source-code editor supporting several languages
- Stack Overflow - A community space for finding and contributing answers to users' code problems
- Twine - An open-source tool for telling interactive, nonlinear stories. You don't have to write code to create a simple story, but you can use variables, conditional logic, etc. to make more complex ones. You can also use Twine to create/edit dialog trees and import them into Unity.
- Godot Documentation - Godot is beginner friendly and has good documentation
- Software carpentry - Version control with Git
- GitHub - A collaborative developer platform to build, scale, and deliver secure software.
- GitHub Collaborative Workflow

Weekly Goals:

1. Select game engine and set up project
2. Set up version control
3. Create something with game engine
4. Change the behavior of an object by attaching a script
5. Make something interactable

6. Make a roadmap to MVP. What scripts and functions need to be created for the game to work as envisioned?

Bit of Fun:

Six Stages of Debugging

- That can't happen
- That doesn't happen on my machine
- That shouldn't happen
- Why does that happen?
- Oh, I see
- How did that ever work?