**Welcome to Day #2 of CGCC!**

Every day we will have a GitHub repository page that outlines each day and the activities that we will complete. We will also provide all homework on these pages.

Feel free to browse the other days to see what is coming up!

As always, let us know if you need any help or have any questions.

*Link to Camp GitHub*: <https://github.com/paigerodeghero/ClemsonGameCodingCamp/tree/master/2021>

**Links for Camp Day #2**:

* GoDot Engine
  + <https://godotengine.org/>
* GitHub Overview Video
  + <https://www.youtube.com/watch?v=w3jLJU7DT5E&t=30s>
* GitHub Classroom
  + TODO
* How to use GitHub?
  + <https://docs.github.com/en/github/authenticating-to-github/creating-a-personal-access-token>
  + <https://github.com/git-guides/>
* Game engines
  + <https://youtu.be/DKrdLKetBZE>
* ProfessorPlatypus
  + <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.01>
  + <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.02>
  + <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.03>
  + <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.04>
  + <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.05>
  + <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.06>
  + <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.07>
  + <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.08>

**Optional Resources**:

* What is open source software (OSS)?
  + <https://www.youtube.com/watch?v=a8fHgx9mE5U>
* Tutorials:
  + FlappyBird
    - <https://youtu.be/8_ThGJG9Kqg>

**Day 2: GitHub, Godot, and creating first game**

**SCHEDULE**:

* Homework review
* Meeting your team
* GitHub Classroom setup
* GitHub introduction
* “ProfessorPlatypus” story
* Godot game engine
* Making “ProfessorPlatypus”
* Homework: Make one change to “ProfessorPlatypus” and commit to GitHub

**ACTIVITY**: Homework Review (20 minutes)

Homework Review:

* Each student presents their findings from the game they choose to play
* Discuss the following game elements from the game
  + Goal
  + Story
  + Rules
  + Players
  + Player interactions

**ACTIVITY**: Meeting your team (30 min)

* Students are introduced to their virtual workspace
* Activity: Meeting your team.
  + Introductory Interview with Project Partner
    - Name
    - Which grade are you in?
    - What is your background with computers? What skills do you have?
    - What do you find most interesting about computers and how does this impact your college plans?
    - How do you manage your time when you get busy with a lot of tasks?
    - Have you worked on a team project before? If yes,
    - How often did your team meet together?
    - Did your team have a leader? What did that leader do?
    - What was your role on the team?
    - How well did you get along with your teammates related to work, or related to non-work?
    - Who are the other members of your family? Do you live with them?
    - What are your hobbies/interests/passions that are not related to this class?
    - Do you have any unique skills/tricks that you can show me now?
    - Who is your favorite teacher in high school? What do they teach? Why are they your favorite?
    - What was the last non-software-related book you read and describe what it was about in 2 sentences.
    - What is the best movie you saw in 2020, and why was it your favorite?
    - Do you have a personal hero in your life? Who is it and why are they your hero?
    - What is your phone number? For backup purposes in case technology doesn't work.
    - When can you work on the project outside of class?
    - Negotiate mutually exclusive times to edit code
    - What kind of programming experiences have you had in the last 3 years?
    - Tell me about a trip you took that was far away from where you live?

**ACTIVITY**: GitHub Classroom setup (20 min)

* Invite students to GitHub classroom
* Students accept the invitation
* Make sure all student groups and projects are setup in GitHub

**INSTRUCTION**: GitHub Introduction (5 min) (Put it in context with game development)

* GitHub intro
  + What is GitHub?
    - <https://www.youtube.com/watch?v=w3jLJU7DT5E&t=30s> (check)
  + Create a token
    - <https://docs.github.com/en/github/authenticating-to-github/creating-a-personal-access-token>
  + How to use GitHub?
    - <https://github.com/git-guides/>
    - Commit
    - Pull
    - Push
    - Merge

**INSTRUCTION**: Creating a game: Story (10 min)

* Discuss the story of ProfessorPlatypus
* What is the goal?
  + To go through as many obstacles as possible
* How can we make the game more interesting?
  + Add enemies in the game

**ACTIVITY**:How to make ProfessorPlatypus more interesting? (5 min)

* Students share their ideas to make ProfessorPlatypus more interesting

**INSTRUCTION**: INTRO GODOT: (game engines in general) (5 min)

* What is a game engine?
  + A game engine is a framework that allows us to create games without worry about how our code interacts with the computer
  + <https://youtu.be/DKrdLKetBZE>
* What is godot?
  + Is an open-source game engine
* Games made in godot
  + Go over a few and/or show
    - The Adventures of Dog Mendonça & Pizzaboy
    - Commander Keen in Keen Dreams

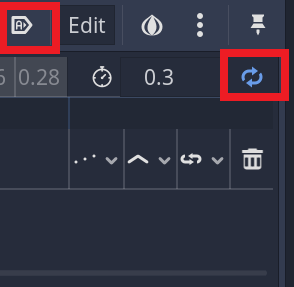
**INSTRUCTION**: Setup Visual studio code (20 min)

* Download visual studio code
  + Open visual studio code
    - Open extensions
    - Install the following
      * Godot-tools
      * Live Share Extension Pack

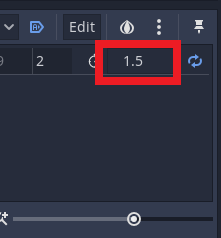
15 min break

**ACTIVITY**: Create “ProfessorPlatypus” in godot [see one – do one – show one]

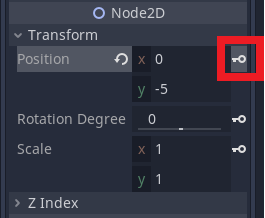
* **INSTRUCTION**: Download project starter from GitHub (10 min)
  + Use this link to download the starter version of ProfessorPlatypus
    - <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.01>
  + Unzip the file
* **INSTRUCTION**: Open the project in visual studio code, press F1 and select the below option (5 min)
  + - Godot Tools: Open workspace with Godot editor
    - If requested, point it towards you Godot installation
* **INSTRUCTION**: Create player object (10 min)
  + Open world scene by double clicking **World.tscn** in the **FileSystem**
  + Create a new scene by using the **+** next to **World**
  + Press **+** to add a **RigidBody2D** node. Rename the node to **player**
  + Press **+** to add a **Sprite**
  + Select **player** and press **+** to add a **CollisionShape2D**
  + Select the **Sprite**. Drag the **platypus.png** file into **Texture** in the **Inspector**
  + In **Inspector** -> **Animation** -> **Hframes**, set the value to 3
  + Select **CollisionShape2D**. In **Inspector** -> **Shape**. Select **New RectangleShape 2D**
  + Increase the size of the collider as required
  + Save scene as player.tscn
  + GitHub: <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.02>
* **INSTRUCTION**: Create player animation (10 min)
  + Select **player** and press **+** to add an **AnimationPlayer**
  + Create idle animation
    - Use **Animation** -> **New** to create a new animation
    - Name it as **idle** and press **OK**
    - Enable auto play and looping using the buttons



* + - Set animation duration to 1.5 seconds



* + - Select **Sprite** from **Scene** tab and navigate to **Animation** in **Inspector**
    - Select time **0** on animation
    - Set Hframes to 0
    - Create a keyframe by pressing the key button

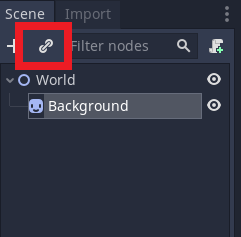


* + - Select time **0.2** on animation
    - Set Hframes to 1
    - Create another keyframe
    - Select time **0.4** on animation
    - Set Hframes to 2
    - Create another keyframe
  + Save the game
  + GitHub: <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.03>
* **INSTRUCTION** Create a second animation (10 min)
  + Creating swim animation
    - Use **Animation** -> **New** to create a new animation
    - Name it as **swim** and press **OK**
    - Enable looping using the button
    - Set duration of animation to **0.6**

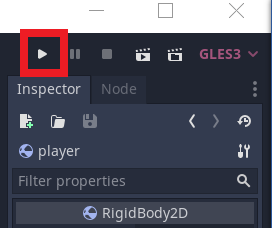
Graphical user interface, application

Description automatically generated

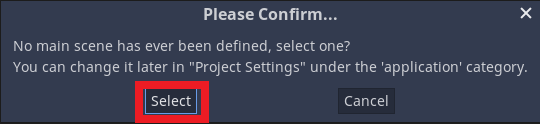
* + - Select **Sprite** from **Scene** tab and navigate to **Animation** in **Inspector**
    - Select time **0** on animation
    - Set Hframes to 0
    - Create a keyframe by pressing the key button
    - Select time **0.2** on animation
    - Set Hframes to 1
    - Create a keyframe by pressing the key button
    - Select time **0.4** on animation
    - Set Hframes to 2
    - Create a keyframe by pressing the key button
  + Save the game
  + GitHub: <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.04>
* **INSTRUCTION**: Add player instance to World scene (5 min)
  + Open the World scene
  + Create an instance of the Player



* + Place player in the middle of the scene
    - You can use the mouse to click and drag the player or you could use the transform property
  + Use the play button on top right to play the game



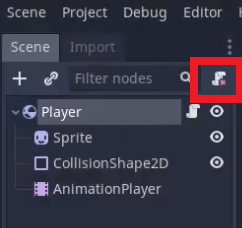
* + Click **Select** from the below shown popup. This will help us define the main scene



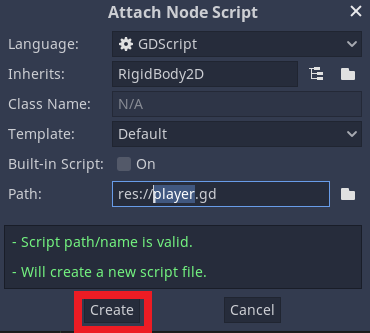
* + Select **World.tscn** and click **open**
  + Save the game
  + GitHub: <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.05>

15 min break

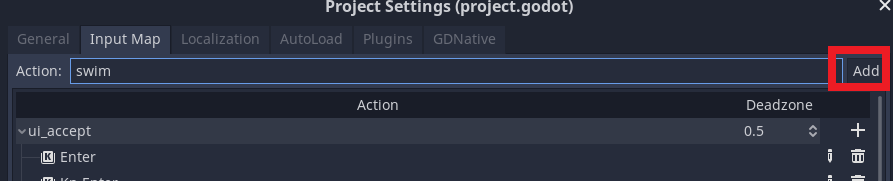
* **INSTRUCTION**: Make player move with keyboard input (10 min) (VS code liveshare)
  + Open **Player** scene
  + Select **Player**
  + Set **Inspector** -> **RigidBody2D** -> **Gravity Scale** to 0
  + Add a new empty script to the player by using the script button



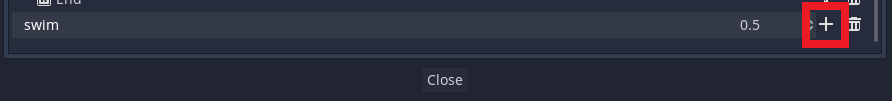
* + Click Create



* + Define an action
    - Navigate to **Project** -> **Project Settings** -> **Input Map**
    - Add an action **swim**



* + - Use the **+** next to swim to link action with space bar press
      * Key -> Press **Space** -> OK



* + Click close
  + Save the game
  + GitHub: <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.06>
* **INSTRUCTION**: Add code to move player up and down (15 min)
  + Open player.gd in Microsoft Visual Studio Code
  + Replace the contents with below code
    - This will move the player up and down as well as play the swim animation

extends RigidBody2D

var started = false

func \_physics\_process(delta):

    if Input.is\_action\_just\_pressed("swim"):

        if !started:

            start\_swim()

        swim\_speed()

func start\_swim():

    if started: return

    started = true

    gravity\_scale = 5.0

    $AnimationPlayer.play("swim")

func swim\_speed():

    linear\_velocity.y = -200

* + Save the game
  + GitHub: <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.07>
* **INSTRUCTION**: Add limits to where the player can go (10 min)
  + Modify Player.gd script to sop the player from falling beyond a limit

extends RigidBody2D

var started = false

func \_physics\_process(delta):

    if Input.is\_action\_just\_pressed("swim"):

        if !started:

            start\_swim()

        swim\_speed()

    if position.y >= 500:

        gravity\_scale = 0

        linear\_velocity.y = 0

        $AnimationPlayer.stop()

func start\_swim():

    if started: return

    started = true

    gravity\_scale = 5.0

    $AnimationPlayer.play("swim")

func swim\_speed():

    linear\_velocity.y = -200

* + Save the game
  + GitHub: <https://github.com/domini4/ProfessorPlatypus/releases/tag/1.08>

**HOMEWORK**:

* Complete all steps shown today
* Make one change to “ProfessorPlatypus”
* Commit homework to GitHub