Texty Adventure – Part 1

An open world of interesting things, including quests and crafts.

# Submission Guidelines:

In GitHub, create a branch for your project named “Part1\_Completed”. Treat the branch as an archive.

# Project Administration (15)

* Create a new private repo named Sp21-EGAM102-TextyAdventure-YourNameHere
* Include the Unity .gitignore file
* Put the project starter files in the repo, at the level of the .gitignore file
* Share the repo with me: TimHandleyAC
* Put an annotated rubric in the root directory of your project. Use Word .docx format.

# Base Requirements (68)

* Constraints:
  + You may use the images in the starter project
  + You may use SpriteShapes
  + You may not use any other graphical assets
* Player Inventory:
  + Use a List of strings for the PlayerInventory
  + When the player types “i” or “inventory”, print a string with all items in the Player’s inventory
* Get and Wear: Create \*two\* items that the player can *get* and *wear*
  + Build reasonable logic and print informative errors:
    - When the player *get*s an item, it is made invisible, deleted from SceneInventory, and added to PlayerInventory.
    - If the player tries to *get* something that isn’t in the SceneInventory, print an error.
    - If the player tries to *wear* something that isn’t in the PlayerInventory, print an error.
  + Commands to get and wear item #1:
  + Effect of item #1:
  + Commands to get and wear item #2:
  + Effect of item #2:
* Look & Scene Inventory:
  + Use a List of strings for the SceneInventory
  + When the player types “look”, print a string with all items in the SceneInventory
* NPC Conversation with *talk* command
  + The NPC can be whatever you like. The NPC should be easily visible in the scene.
  + If the player inputs “talk [NPCname]” should produce a few sentences of interesting info.
  + AND the first time the player says “talk [NPCname]”, it should add 2 *topics* to your inventory.
  + The topic words should show up in the NPC’s text, and be somehow emphasized.
  + Note that you can add markup to UI text: [documentation](https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/StyledText.html)
  + If the player inputs “talk [NPCname] [topic]” AND they have the matching topic in their inventory, the NPC should say something about that topic.
  + Write your NPC’s name here: w

# Stretch Goals:

* (+3) Prepare for, and participate in next Wednesday’s Show ‘n Tell.
  + Before 5PM PST next Wednesday, send me an email with an image or a short video (< 60 seconds) from your project. You pick what you want to share.
  + Be prepared to say 2-3 sentences about ONE THING from your project.
  + Say 2-3 sentences about your image in our next class meeting.
* (+3 to +10) Better useability and accessibility:
  + Upgrade logic for *talk* to give different messages for different mistakes:
    - If the player tries to talk to something that doesn’t talk, print an error.
    - If the player talks to a talking NPC about an unknown topic, print an informative error.
    - If the player tries to talk to something that isn’t even in the scene, print an error.
  + Upgrade logic for *wear*:
    - If the player tries to *wear* something that they are already wearing, print an error.
    - Hint: Create a new List of strings called EquippedItems ...
  + Add a *help* command:
    - When the player types ‘help’, you print a list of all the basic commands
    - Extra respect if the help messages are somehow context sensitive
  + If you did any of these useability tasks, explain below:
    - What have you done to make the game more accessible?
* (+5 to +20) Create a varied quest of 5 or more steps.
  + Quest must use all three basic commands: *get*, *wear*, and *talk*.
  + In one sentence, describe is the premise of your quest:
  + Write a walkthrough. What commands do you use to solve the quest:
    1. Command 1:
    2. Command 2:
    3. etc.
* (+5 to +20) Build a cooking system:
  + Example: “cook hat flower” will
    - Check that the player has a *hat* and a *flower* in their inventory
    - If they do not have both ingredients, print an error message
    - If they have both ingredients, remove hat and flower from PlayerInventory, and add “fancyhat” to PlayerInventory
  + Outline your cooking system:
    - Command to create recipe #1:
    - etc.
  + Extra respect if the things that you craft are \*useful\*. If they are useful, outline the usage:
    - Command to use crafted item #1:
    - Effect of crafted item #1:
    - etc.
* (+10) Tab completion:
  + If the user types a piece of a command, and presses tab, the system autocompletes the word
  + Ex: If the user types ‘w’ and presses tab, the ‘w’ is expanded into “wear”
* (+1 to +20) Other. Something related to this week’s topics: if, else, logic, inventory, and adventure.
  + Explain: What is your nifty thing, and where have you used this thing in your project?