**Spike:** 12

**Title:** Command Pattern

**Author:** Ben Holmes, 103024841

**Goals / deliverables:**

* Adventure file format needs to be extended to support game entities.
  + Commands will look at but not changes these entities
* Command processor with commands:
  + GO: change location
  + HELP: list commands and syntax
  + INVENTORY: lists entities in player inventory
  + LOOK, LOOK AT: shows details of location or entities
  + ALIAS: remap command names
  + DEBUG: shows location tree and all relevant data
* UML diagram for commands

**Technologies, Tools, and Resources used:**

* Visual Studio 2022
* Draw.io
* Spike 11 (as base)
* Spike 10 (player)

**Tasks undertaken:**

* Copied across spike 11
* Created entity class
* Get entities from the json file
* Copied across player from spike 10 and improve it
* Get player from json file
* Get basic command processing up (go and quit working considering they were done in spike 11)
* Get the other commands working one at a time
* Draw UML diagram

**What we found out:**

**A screenshot of a computer screen

Description automatically generated**

The first deliverable was getting the adventurer file format to generate entities:

First was the creation of the entity struct and the corresponding Json deserialization function:

header

A screen shot of a computer code

Description automatically generated

cpp

A screen shot of a computer code

Description automatically generated

Next the Json was updated to have an array of entities at each location called contents:

A screen shot of a computer

Description automatically generated

(this is the testing values which were identical for all the locations as there was no need for differing ones at this point)

And finally, I needed to update the Locations Json deserialization to create the contents

header



cpp

A screen shot of a computer code

Description automatically generated

(this loops through each object in the contents and assigns its id as the key to the contents map<string, Entity>)

This was very simple to do given the experience in spike 11 being nearly the same, so I also transferred over the player to the Json and have its Json deserialization.

header

A computer code on a black background

Description automatically generated

cpp

A computer screen with colorful code

Description automatically generated

Json

A screen shot of a computer program

Description automatically generated

A computer screen shot of a black background

Description automatically generated

The next deliverable was the creation of the commands:

First was setting up the various needed files:

* CommandManager.h/cpp
* Command.h/cpp

Then was the creation of the command manager

Header

A screenshot of a computer

Description automatically generated

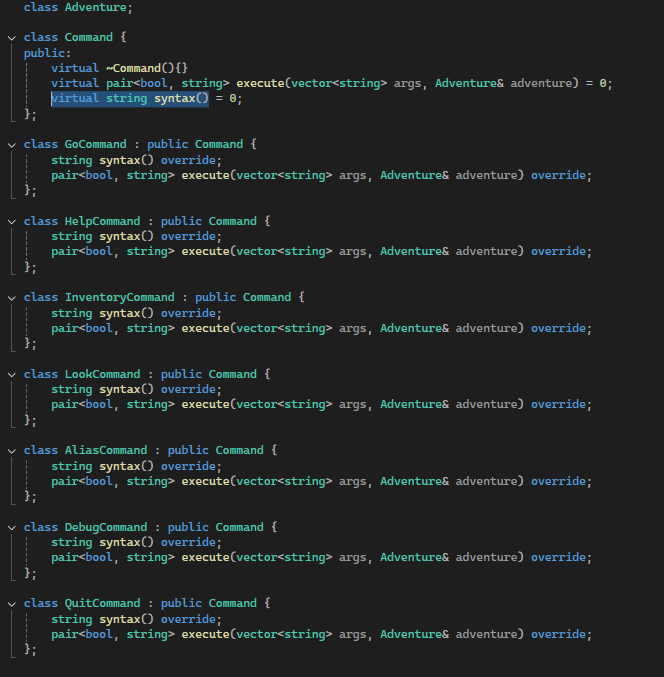
Cpp

A screen shot of a computer program

Description automatically generated

And the commands:

Header

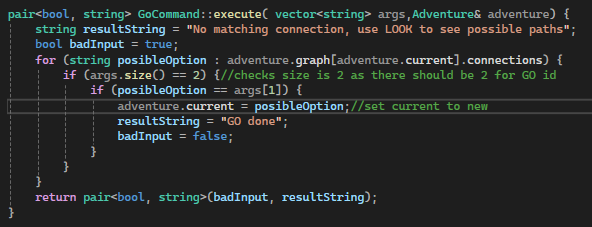


Go was the first to develop, as I already knew the rough process given spike 11. Setting up the commands with inheritance wasn’t too hard but I needed to pass the Adventure object down the chain to the commands which caused several issues, mainly with the header files and having looping includes as I didn’t realise that was a possibility so that took a long time to figure out. I ended up using forward declaration a lot.

Once I fixed all those issues though, it was simple to get every one of the commands to work.

Each command returns a pair<bool, string>, the bool keeps the input loop for bad inputs going (true is looping, false is end) and the string is for an error message if the input is wrong (and possible debug).

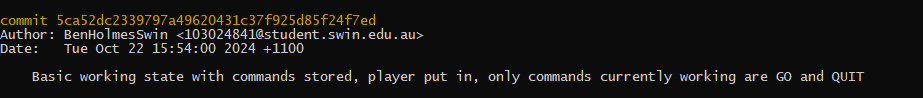
GO:



Go was simple once the issues were fixed, copied from 11 and edited to work

A black screen with white text

Description automatically generated



HELP

A screenshot of a computer program

Description automatically generated

Help itself was only slightly difficult, mainly because of ALIAS changing the commands I had to make the syntax(string cmdName) method of command of which each different command has one given their different syntax’s.

A black background with text

Description automatically generated

A black screen with numbers and letters

Description automatically generated

INVENTORY

A screen shot of a computer code

Description automatically generated

Very simple, basically just copying some debug things I had in spike 11

LOOK

A computer screen with many colorful text

Description automatically generated

Look was only tricky because I needed to handled both look\_at and look in the same command, and have look\_at be able to do both the inventory and the locations contents.

A computer screen with numbers and letters

Description automatically generated

ALIAS

A computer code on a black background

Description automatically generated

ALIAS was simple as I feel like I understand pointers and temp variables quite well now (not perfect obviously). My only issue was originally I tried to deconstruct temp at the end of the if(Ars[2]) (as I thought it was needed) but that broke the command.

A computer screen shot of text

Description automatically generated

A screen shot of a computer code

Description automatically generated

DEBUG

A computer screen shot of code

Description automatically generated

Debug was very simple, just nesting a couple of loops for the contents and connections and making it a good layout

A screenshot of a computer program

Description automatically generated

(as stated the Json contains the same contents for each location as no need for different at the moment)

A black screen with white text

Description automatically generated

I also added the QUIT to the command manager for ease of use

A computer screen shot of text

Description automatically generated

Here is the creation of the command manager in the Adventure class (along with the grabbing of player and locations from the Json)

A computer screen shot of code

Description automatically generated

The third and final deliverable was the UML diagram:

A screenshot of a computer

Description automatically generated