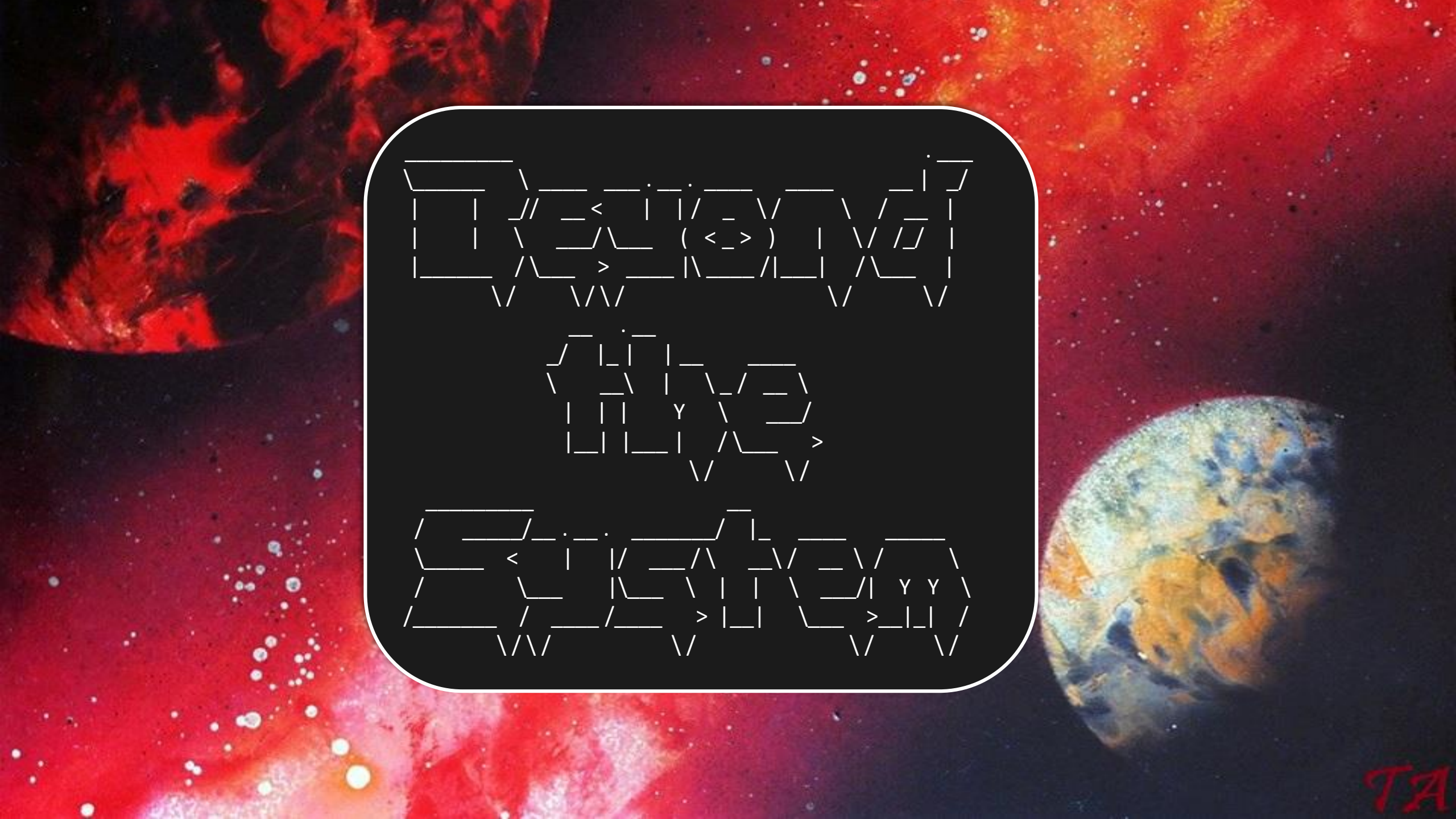
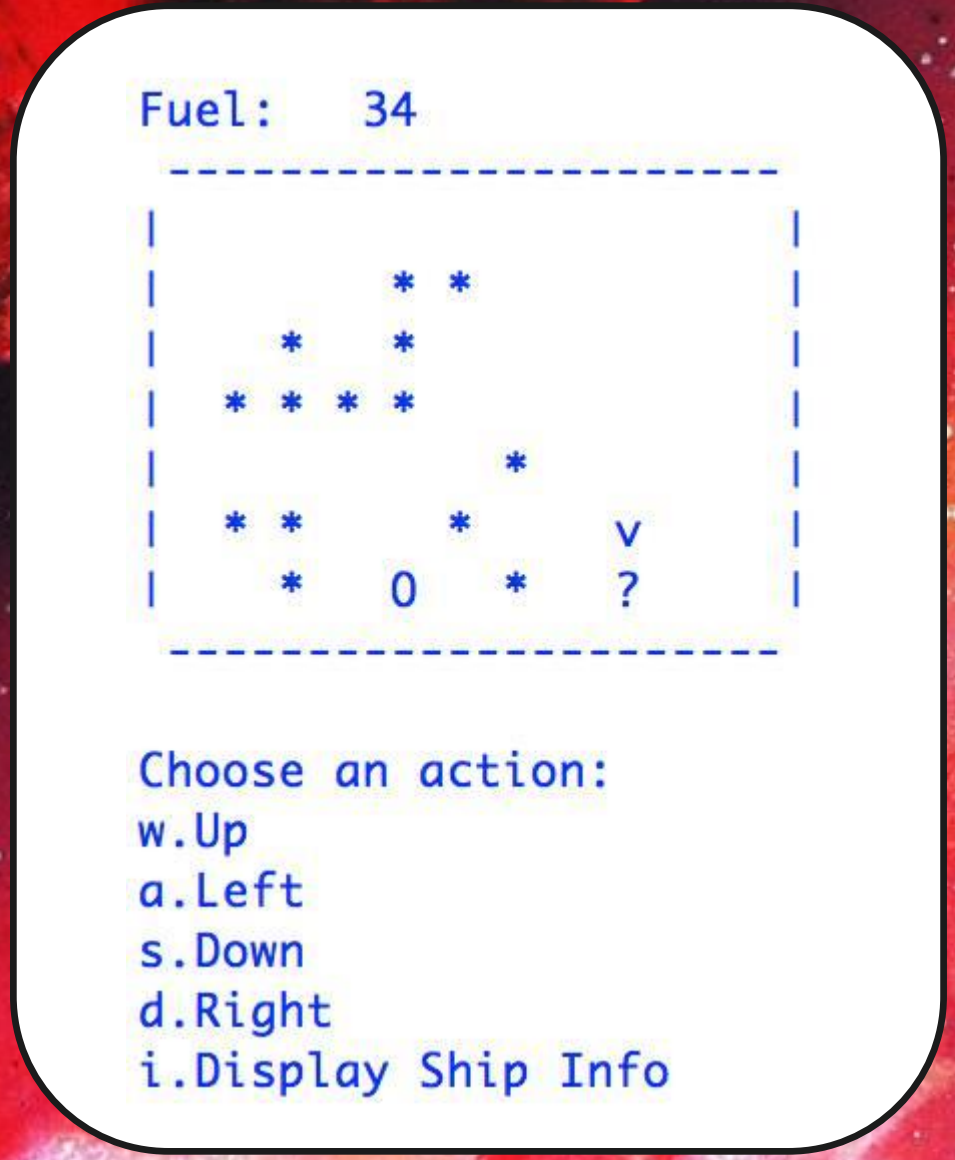

TA

The Idea

Beyond the System is a procedurally generated space exploration game meant to ask one simple question:

Is there life out in the universe besides us?

Overall Design



- Scalability
  - Capable of having 10,000+ maps
  - Capable of having hundreds of events
- Modularity
  - Most functionality is encapsulated in functions/methods
- Version Control
  - Using GitHub

Encounter Workflow

```
while stillGivingInput:


    curEventName = "event" + str(eventNum) + ".txt"
    curBlurbFile = open_file(curEventName, "w")

    curEventChoices = "event" + str(eventNum) + "_c.txt"
    curChoiceFile = open_file(curEventChoices, "w")


        ## get the description
    curBlurbFile.write(input("Give me a situation for the ship to encounte

    numOptions = int(input("How many choices would you like available?: \r

    for i in range(numOptions):
        ## get the first choice for a response
        curChoiceFile.write(input("Give me your action! Try to hit space c

        ## get 4 value mods
        for i in range(4):
            if i == 0:
                curChoiceFile.write(input("What's choice's affect on fuel?
            elif i == 1:
                curChoiceFile.write(input("What's choice's affect on oxyge
            elif i == 2:
                curChoiceFile.write(input("What's choice's affect on biomc
            elif i == 3:
                curChoiceFile.write(input("What's choice's affect on the h

        ## get the resolution to that choice
        curChoiceFile.write(input("What's the resolution to this choice?:
```

event3.txt

```
A random chunk of ice is floating around. It's rather beautiful but who knows how you
would get it on board.
```

event3_c.txt

```
Try opening up the cargo hold. That water might really come in handy.
0
-5
10
0
You open the cargo hold but forget to fasten one of the interior doors causing
precious oxygen to exit the ship. All is not lost though as you did manage to harvest
drinkable water from the comet.
Run away! It could be dangerous.
0
0
0
0
You successfully ran away.
Get close and inspect it for anything useful.
-2
0
5
-5
It scratches the ship and takes a bit of fuel to get up and close but you manage to
get a bit of drinking water by vacuuming some of the ice into your ship.
```

shelveMaker.py - /Users/bgspiral/Documents/School/HCC/CSI106/fina

```
import shelve

def importFile (fileName):
    ## changes file names to the correct ones for each file (T is text
    fileNameT = fileName + ".txt"
    fileNameC = fileName + "_c.txt"

    ## import the description file
    file = open(fileNameT, "r")
    blurbText = []
    blurbText = file.readline()
    file.close()

    ## imports the choice file
    file = open(fileNameC, "r")
    choiceText = []
    choiceText = file.readlines()
    file.close()

    ## meant for something later, if we later decide to add a link
    ## to another event, 7 will let us do so.
    if len(choiceText) % 6 == 0:
        numOptions = len(choiceText) / 6
    elif len(choiceText) % 7 == 0:
        numOptions = len(choiceText) / 7

    ## creates the encounter list
    encounter = []
    encounter.append(int(numOptions))
    encounter.append(blurbText)
    encounter.append(choiceText)

    return encounter
```

```
You have encountered an event!!


You're floating through the outskirts of an advanced solar system. You
spy a white, rocky planet and think you might be able to land there.


Captain, these are your options:

0) Attempt to land. There might be something useful.

1) Continue on. It could be dangerous.

0

You manage to land your ship. The planet's surface is rocky and icy.
Though it's too cold for you to traverse, you send out the droids to
collect water and materials. They bring back plenty of both and today,
luck must be on your side because they find materials for both repairing
the hull and converting to fuel.

The ship has gained 15 fuel.
Levels of oxygen remain stable.
The ship has gained 10 biomass.
The ship has gained 20 hull integrity.
```

Automation

```
def printProperly(text, printSpeed = 0.0,end = "\n"):
    """Prints 66 characters then starts looking to make a new line."""

    import time
    linePos = 0
    for letter in text:
        if letter == "$":
            print("\n")
            linePos = 0
        else:
            print(letter, end="")
            time.sleep(printSpeed)
            linePos += 1

        ## probably had a long, don't want weird spacing if we did!
        if linePos >= 77:
            linePos = 0
        ## creates a new line once we hit a space after char position 67
        elif linePos > 67 and letter == " ":
            print()
            linePos = 0

    print(str(end), end="")
```

^

Automatic Text Formatting

Encounter Creator

V

```
What's the first unused event number (event#.txt) That you want to start with:
22
Give me a situation for the ship to encounter:
You come across an ominous, abandoned ship.
How many choices would you like available?:
3
Give me your action! Try to hit space around 10-12 words:
Waltz right in. What's the worst that could happen?
What's choice's affect on fuel?: -5
What's choice's affect on oxygen?: 0
What's choice's affect on biomass?: 0
What's choice's affect on the hull's integrity?: -20
```

Map and Events Collide

```
def displayMap(self):
    """Displays the map."""
    ## alternative method for displaying the map
    mapDisplay = ""
    mapDisplay += " "+"-" * 2 * self.numCols + "\n"
    for row in range (self.numRows):
        mapDisplay += "|"
        for col in range (self.numCols):
            mapDisplay += self.grid[row][col] + " "
        mapDisplay += "|\n"
    mapDisplay += " "+"-" * 2 * self.numCols

    print(mapDisplay)
```

```
eventName = ""
if tile != 0:

    eventNum = None

    ## if unusedEncounter is empty, this fills it back up
    ## should avoid any index errors with random.choice but
    ## the except covers any possible issues
    try:
        if Map.unusedEncounters == []:
            Map.unusedEncounters = Map.usedEncounters[:]
            Map.usedEncounters = []

        eventNum = random.choice(Map.unusedEncounters)

    except IndexError:
        print("You encountered an Index Error\n")

    Map.usedEncounters.append(eventNum)
    Map.unusedEncounters.remove(eventNum)
    eventName = "event" + str(eventNum)

tileInfo = [tile, [randomRow, randomCol], eventName]
self.tileList.append(tileInfo)
```

Maps Within Maps

```
class Galaxy (object):
    """A Collection of Maps"""
    import ShipClass

    def __init__ (self, maps, ship):
        import random
        self.maps = []
        self.numMaps = maps - (int(len(ship.getLog()) / 2))
        self.minLength = 5
        self.maxLength = 14

        ## creates a series of maps and stores them in a list
        for i in range(self.numMaps):
            random1 = random.randint(self.minLength, self.maxLength)
            random2 = random.randint(self.minLength, self.maxLength)
            if i == self.numMaps - 1:
                mapNew = initializeFinalMap(ship,5, 5, 0)
            else:
                mapNew = initializeMap(random1, random2, 3)

            self.maps.append(mapNew)
```

Main Function

```
def main():
    GAME_LENGTH = 6

    text.displayIntro()

    text.displayStory()

    ship = load()
    if ship == None:
        ship = ShipClass.Ship(input("What would you like to name your ship:\n"))
    print(ship)

    cheatCode(ship)

    numMaps = GAME_LENGTH

    galaxy = Galaxy(numMaps, ship)
    galaxy.play(ship)
```

Inheriting Victory

```
class FinalMap (Map):
    endingFile = open("endings.txt","r")
    endings = endingFile.readlines()
    endingFile.close()

    def populateTiles(self, ship):
        import random
        usedTileLocations = []

        randomCol = random.randrange(self.numCols)
        randomRow = random.randrange(self.numRows)

        while [randomRow, randomCol] == self.position or [randomRow,randomCol]
        in usedTileLocations or [randomRow,randomCol] in self.blockedTiles:
                randomCol = random.randrange(self.numCols)
                randomRow = random.randrange(self.numRows)

        self.grid[randomRow][randomCol] = "?"

        tileInfo = [1, [randomRow, randomCol]]
        self.tileList.append(tileInfo)
```

```
def play(self, ship):
    """Standard play function for the game."""
    mapCounter = 0
    print("self.numMaps = " + str(self.numMaps))
    print("length of the log = " + str(len(ship.getLog())))
    ## allows you to iterate through the list of maps using foundDoor
    ##as a flag
    while mapCounter < self.numMaps and self.notDeadYet(ship)
    and not self.maps[mapCounter].foundEnding:
        runMap(self.maps[mapCounter], ship)
        if self.maps[mapCounter].foundDoor:
            mapCounter += 1
            events.printProperly("\nYou take some time to refine a little fuel
            ship.increaseFuel(5)
            print("\n" + str(ship))
            self.maps[mapCounter].save(ship)
            input("Hit enter when you're ready to move on.")

    if self.notDeadYet(ship):
        print("\nYou've made it through the game! Thanks for playing!")
    elif not self.notDeadYet(ship):
        if ship.getFuel() <= 0:
            text.displayDeathOutro("fuel")
        elif ship.getBio() <= 0:
            text.displayDeathOutro("biomass")
        elif ship.getOxygen() <= 0:
            text.displayDeathOutro("oxygen")
        elif ship.getHull() <= 0:
            text.displayDeathOutro("hull")
```

Saving + Loading

```
def save(self,ship):
    """Saves the current status of the log and ship."""
    import shelve

    while True:
        try:
            saveInput = input("Would you like to save? (y/n): \n")
            if saveInput.lower() == "y":
                saveFile = shelve.open("saveFile.dat")
                saveFile["ship"] = ship
                saveFile["logEvents"] = ship.getLog()
                saveFile.sync()
                saveFile.close()
                break
            else:
                break
        except:
            print("Sorry that's not a valid input.\n")
```

```
def load():
    """Loads in the information from the save file."""
    ## opens the load file and updates the game's ship/Map.
    import shelve
    import time
    ship = None

    while True:
        try:
            loadInput = input("Would you like to load a past save file? (y/n): \
            if "y" in loadInput:
                print("\nLoading in your save file now...\n")
                time.sleep(1)
                loadFile = shelve.open("saveFile.dat", "r")

                print("Preparing your ship for space travel...\n")
                time.sleep(1)
                ship = loadFile["ship"]

                print("Loading in your ship's star log...\n")
                time.sleep(1)
                Map.usedEncounters = loadFile["logEvents"]

                loadFile.sync()
                loadFile.close()
                print("Load complete.")
                break
            elif "n" in loadInput:
                break
        except:
            print("Looks like you don't have a save file yet or we can't find it
            ship = None
            Map.usedEncounters = []
    print()

    return ship
```

Beyond the Game

Where do we go from here?