World of Zuul Assignment

Pawel Makles
(K21002534)

Created: 19th November 2021
Last Modified: 3rd December 2021

1 User Level Description

This game is largely open world but has a main storyline which the player can choose to follow to complete the game, as well as other side-quests which can be done alongside or after the story.

In this game, you are one of millions of citizens in an isolated city home to a Beastman[1] society, the city being loosely based off Animacity[2] from the series Brand New Animal.[4] The plot revolves around Sylvasta who is deeply intertwined with the city and owns the city's medical centre.

There have been reports that their research has been highly unethical and leaks have been coming out from former employees which have been riling up protests throughout the city. Your goal is to figure out what's really happening and to put a stop to it. The story is also loosely based off Utopia,[5] in which a group of people find out the truth behind a manuscript and find themselves in the middle of a conspiracy.[10]

The canonical name for the game is currently 'World of These', formerly 'World of Deez'.

2 Implementation

The project is split up into several packages, most of which work independently of each other, in general, the `Game` uses the `commands` package (providing actions the player can perform) along with the `world` (providing the things that the player can interact with in the game) package to run the game. They are laid out as follows:

2.1 Commands

This package provides the classes required to parse and run commands, it includes the `CommandManager` which registers objects of type `Command` . More about parsing commands is in the challenge task section.

This package also includes a subpackage called `core` containing all of the 'core' commands required to run any game world, this includes things such as picking items up, quiting the game, and so forth.

2.2 Entities

This package provides the basic tools required to create simple and complex entities within the world, more about entity detail is discussed later.

It also contains a subpackage `actions` which contains common interfaces which entities can implement to allow the ways that they can be interacted with to be quite modular.

2.3 Content / Campaign

The `content.campaign` package contains a lot of custom content derived from the base game used to construct the story and the story world.

2.4 World

This package has the basic building blocks for creating worlds including the `World` and `Room` classes which can be easily extended for more functionality. The only limitations are that traveling between rooms must be done in a direction specified in the `Direction` enum and that the `Location` of any entity must be either a room, inventory or neither.

2.5 Util

This package contains a bunch of small utility classes, including:

- **BlueJ.java**:[7] Contains improved methods on detecting whether the application is currently running in BlueJ. (taken from my previous coursework assignment)
- **Localisation.java**: A pretty straightforward implementation of a localisation engine. Simply maps (period-separated) keys to their respective (nested) keys in a language file.
- **Search.java**: Methods for searching through various data structures in the game.
- **Tree.java**: A very simple implementation of a tree with basic traversal methods.

2.6 Dialogue & IO & UI & Events

These are all discussed later on in the challenge task section.

3 Base Tasks

This section describes how I implemented the basic task requirements.

- *"The game has several locations the player can walk through."*

  I began by first designing the world map (see Figure 1), using Excalidraw,[6] then implemented each location as a `Room`.

  To build the world, I made a `World` class to house all the locations which exist in the game world and then extended this using the `CampaignWorld` class which builds the story world, creates rooms and registers events.

  I ended up adding 10 locations into my game:

  - **City Centre**: Centre of the city connecting major areas with the coast.
  - **Apartments**: This is the player's residence.
  - **Street**: This is the main city street connecting important buildings.
  - **Shop**: The local city shop where the player frequents to get necessary items.
  - **Back Alley**: This is where the player can start the main story mission.
  - **Coastline**: There are two coasts, one on the city side and one on the mainland.
  - **Forest**: The forest grants access to the Worm Hole and to other side-quests.
  - **Worm Hole**: For challenge task 3.

  The layout is heavily inspired by Animacity, although since there's no available map of the actual city, I made my own interpretation based off various pieces of art, (see Figure 2). I used an existing real life location to determine the size of the river.[8]

- *"There are items in some rooms that may or may not be picked up by players."*

  To achieve this, I considered all entities to be items which may or may not be picked up by other entities, each entity has its own `Inventory` which is in effect a list of other entities which it is holding.

- *"Each item has a weight and the player can only carry items up to a certain weight."*

  To do this, I added a new private field `weight` of type double to the `Entity` class, which is used to store the entity's weight. It is a double as I wanted access to fractional weights (say 0.01kg) and I wanted to have access to `Stream::mapToDouble` for summations.

  For the second part, I made it so each `Inventory` has a maximum weight it can store, which by default is set to 0 as each entity has an inventory but may not necessarily have the ability to store anything.

  When putting anything in an inventory, we check that the following is satisfied:

  $$\text{currentWeight} + \text{itemWeight} \leq \text{maxWeight}$$

  To determine the weight of common items, I referred to a document I found online published by the City of York Council.[9]

- *"Player can win."*

  The player may win by completing the main story mission (detailed in the walkthrough) which sets a flag that the game has been completed, the player may choose to keep playing in the open world or run *win* to end the game.

- *"There is a command back* which takes you back to the last room."

  I added two new private fields to the player entity, which were `previousRooms` and `retreatingDirection`, these are used to store the path back through the room and the direction which we need to go to get back there respectively. The direction is stored in order to run a check whether the player can actually go back in the direction they intend to, to verify this, the 'retreating direction' is used to call the method `canLeave` on the current room the player is in.

- *"Add at least four new commands."*

  I added several additional commands which are listed below:

  - *bag* : Allows the player to look at their or another entity's inventory.
  - *drop* : Drop any specified item from the player's inventory into current room.
  - *give* : Give a specified item to another entity, we ensure that the entity implements `IGiveable` and give the item to the entity using `IGiveable::give`.
  - *pet* : Pet a specified entity, has to implement `IPettable`, we use `IPettable::pet`.
  - *take* : Take any specified item from another entity's inventory.
  - *talk* : Initiate a conversation with an entity, must implement `ITalkwith`, we call `ITalkwith::talk` to start the conversation.
  - *use* : 'Use' an entity, we call `IUseable::use` to use the entity. (implemented by entity)
  - *where am i* : Tells the game to print out room information again, this is done by emitting `EventEntityEnteredRoom` again.

  See Figure 5 for an example of the help menu.

4 Challenge Tasks

This section describes how I implemented the challenge tasks and what I did in addition.

4.1 Required Tasks

- *"Add characters to your game."*

  I ended up adding 8 characters to my game, 3 of which can move on their own.

  - **Static NPCs**: There are several static NPCs which the player can interact with, talk with or complete missions for. Including the receptionist, city NPC, the old man, the security guard, the shopkeeper and Marie who is part of the main story.
  - **Dynamic NPCs**: There are 2 NPCs which appear or disappear depending on the state of the game, this includes the protestors and security guard.
  - **Moving NPCs**: There is a single NPC, the cat, which randomly follows a fixed path to move around the map. This is done by listening to `EventTick` from world events and then randomly deciding to move.

  Note on character creation: I decided to stick to mostly generic characters in the game, mainly since the people you encounter have not as much impact on the story, with the exception of Marie,[3] who I took straight from BNA,[4] see Figure 3, she is a sly and cunning character which I think perfectly fits the role I need.

- *"Extend the parser."*

  I entirely replaced how the parser worked, I began by implementing a basic model of `Command`, it took a few iterations but I settled on providing Regex `Pattern`s.

  This approach has several benefits:

  - Powerful Regex at low performance cost due to the low number of commands.
  - Ability to have named capture groups which are then interpreted as arguments.

  For each known `Command`, I would create a `Matcher` for each `Pattern`, execute it against the arbitrary command from the user and then pass it into a wrapper class `Arguments` which lets me safely pull out named groups, directions or any other argument type I need.

- *"Add a magic transporter room."*

  To do this, I added a `RoomWormHole` which I made implement `EventEntityEnteredRoom` to listen for when any entities entered the room, as soon as one is detected, a short animation is played and a room is selected at random to teleport the user to. Allowing the user to go to **any room** may interfere with my story so I chose to only spawn the user at any outside areas of the map.

4.2 World Event System

"Add a system for managing world events."

I began by creating a basic `Event` class which can be extended to hold any arbitrary data, it also provides basic functionality to prevent event propagation (see Figure 8).

I created an `EventSystem` class which would manage incoming and outgoing events, I decided that it should have the ability to handle any event and allow anything to create listeners for any object that extends Event.

To map these, I had a `HashMap` which mapped unique `Class<? extends Event>` to linked (as to keep order of registration) hash sets of `IEventListener<? extends Event>`. The interface simply provides a single method, `onEvent(Event e)`, which is called when the event is triggered.

Since Java has type erasure, it means my generic type annotations are gone once the program is compiled, so I can't do things such as object is 'instanceof' class. As such, I decided to ignore cast warnings and entirely rely on compiler type checking to verify that the objects provided are correct, I achieved this by adding a generic type argument to the add listener method and making the compiler check that the target class and event listener match, see Figure 15.

4.3 Dialogue

"Give NPCs interactive dynamic dialogue."

To implement dialogue, I made a simple data structure implemented as follows:

```
Dialogue
├─ DialogueNode
│  ├─ DialogueOption
│  └─ DialogueOption
└─ DialogueNode
   └─ DialogueOption
```

The `Dialogue` houses several nodes each of which have options which lead to other nodes. With this system, I can add dynamic conversations, such as with the shopkeeper, by extending the `DialogueNode` class and providing my own dynamically generated description and options.

For simpler tasks, such as changing game state through conversation, I can create a new `DialogueOption` with an `IDialogueHandler` which provides a method that the dialogue system calls to determine whether anything else needs to happen and what the next node is.

4.4 Terminal Emulator

"Implement a terminal emulator."

I began by creating a new `IOSystem` interface which provides common I/O methods, such as `print`, `println` and `readLine`. I then made all game output come through this interface and created two new classes that implemented it, `StandardIO` as a fallback to stdout and the `TerminalEmulator` which creates a new window.

To render the window I have a class extending `JFrame` which shows my `JTerminalView`. To render the text I override into `paintComponent` and draw my own content using the provided `Graphics` context.[19] (example: Figure 6)

The font used is VT323 Regular.[12]

4.4.1 Ansi Escape Codes

One of the main reasons I built this was for greater flexibility, so naturally I added support for Ansi escape codes. I matched the escape codes in the `TextBuffer` and adjusted the buffer accordingly. (Figure 17 for the implementation)

4.4.2 Emoji / Image Support

I also added a way to load and display small images like emojis (see Figure 4), although I had some difficulties with Unicode being mangled when working with BlueJ.[11]

4.4.3 EventDraw and the Map

Since I already had a way to load images, I also provided a way for anything to draw to the terminal. I added an `EventDraw` (Figure 14) which is emitted whenever the terminal view paints and used it to draw a map (see Figure 7).

5 Code Quality

5.1 Coupling

An example of loose coupling in my project is the `EventSystem`, it is completely independent from the rest of the project and can be used with anything (see Figure 12), as such I am using it with both the `World` and `TerminalEmulator` for different purposes to provide different sets of events. Another example of loose coupling is the `Dialogue<T>` system, it accepts any kind of key through the generic `T` so it can be used with practically anything and if needed, the type can be constricted (example: Figure 13). The only dependency it has is the `IOSystem` interface.

5.2 Cohesion

One example of high cohesion is the `IOSystem` interface (see Figure 9) and the classes which implement it. It is used throughtout the project to refer to an arbitrary input or output, as such, the first implementation I made was StandardIO (see Figure 10) which passes these methods through to `System.out/in`. One benefit is that I can easily slot in additional processing layers, such as `LocalisedIO` (see Figure 11) which can take in data, process it and pass it through to an arbitrary `IOSystem`.

5.3 Responsibility-driven design

I considered responsibility-driven design when first building the structure of my project, for example, the `World` class is responsible entirely for keeping track of rooms and entities and nothing else, to add functionality to it you must extend it as I have done with the `CampaignWorld`. As another example, each `Entity` just stores any information directly relating to the entity such as a name and its location, to implement additional actions such as 'using the entity', we must extend `Entity` and then implement `IUseable`.

5.4 Maintainability

One example of where I considered maintainability is the `Inventory` system, the logic is entirely self-contained and quite simple to understand. Fields such as `items`, `maxWeight` are both private and inaccessible outside of the inventory class, as such I can guarantee that the inventory always stays consistent and hence easily maintainable, these values may only be modified by methods with distinct tasks and appropriate validity checks (examples: Figure 16).

6 Walkthrough

This guide will skip over some parts of the plot, it serves as a way to just get to the end, I would recommend playing through first and then falling back on this if you are stuck.

You wake up in your home, *go down*, then *east* then *north*. Here, you must *talk to Marie*, go through the prompts ($1 \to 1 \to 2 \to 1$) to progress the story.

Now, *go south*, *north west*, *north* and *talk with shopkeeper*, buy the communicator device and leave. Now *go south*, *east*, *north* and *talk to Marie* again. Continue by going *south*, *north west*, *west*. Sit down by *use couch* then *use comms* and go through the prompts, afterwards *go down*. Here, you must pick up documents 2, 4 and 5 by using *take doc2*, ...

Now, *go up*, *east*, *south*, *up* to get back to your home and *use the laptop*. Go through the prompts ($1 \to 4 \to 2$). At this point, you can use the command *win*. If you want to do the side-quest, you'll have to figure it out yourself :-)

6.1 Access to teleporter room

To get to the teleporter room, you must go to the shopkeeper and buy the 'speed boat key', from there you can go to the south of the city and use the boat to get to the mainland. Going south to the forest then east to get to the teleporter room.

7 Known Issues

- Anything starting with 'b' will show inventory.
- NPC dialogue does not vary in conclusion.
- Can't include Unicode in Java source.

References

[1] Brand New Animal Wiki. Beastman
https://brand-new-animal.fandom.com/wiki/Beastman

[2] Brand New Animal Wiki. Animacity
https://brand-new-animal.fandom.com/wiki/Anima_City

[3] Brand New Animal Wiki. Marie Itami
https://brand-new-animal.fandom.com/wiki/Marie_Itami

[4] IMDb. BNA (TV Mini Series 2020)
https://www.imdb.com/title/tt12013558/

[5] IMDb. Utopia (TV Series 2013-2014)
https://www.imdb.com/title/tt2384811/

[6] Excalidraw.
https://excalidraw.com/

[7] GitHub. maven-bluej / BlueJ.java
https://github.com/KCLOSS/maven-bluej/blob/master/BlueJ.java

[8] Google Maps. Jezioro Świerklaniec, Poland
https://www.google.com/maps/@50.4293559,18.9742453,16.12z

[9] PDF. Set of average weights for furniture, appliances and other items https://democracy.york.gov.uk/documents/s2116/Annex%20C%20REcycling%20Report%20frnweights2005.pdf

[10] YouTube. The best (and worst) show you haven't seen
https://youtu.be/PFx2QM0Z8Qo

[11] GitHub. BlueJ Bug Reproductions (wip)
https://github.com/insertish/bluej-bug-demo

[12] Fontsource. VT323
https://fontsource.org/fonts/vt323

References in code.

[13] Ansi.java StackOverflow. How to print color in console using System.out.println?
https://stackoverflow.com/a/5762502

[14] LocalisedIO.java StackOverflow. What is the equivalent of Regex-replace-with-function-evaluation in Java 7? https://stackoverflow.com/a/27359491

[15] JTerminalFrame.java How to detect a key press in Java
https://stackoverflow.com/a/21970006

[16] JTerminalView.java How to "do something" on Swing component resizing?
https://stackoverflow.com/a/8917978

[17] JTerminalView.java Java Documentation. Font Concepts
https://docs.oracle.com/javase/tutorial/2d/text/fontconcepts.html

[18] JTerminalView.java Java Documentation. Java Thread Primitive Deprecation https://docs.oracle.com/javase/1.5.0/docs/guide/misc/threadPrimitiveDeprecation.html

[19] JTerminalView.java StackOverflow. Drawing Canvas on JFrame
https://stackoverflow.com/a/17922749

[20] TerminalEmulator.java StackOverflow. Passing values between 2 threads without intrrrupting each other https://stackoverflow.com/a/23413506

Libraries used:

[21] *com.moandjiezana.toml* https://github.com/mwanji/toml4j

[22] *commons-io* https://commons.apache.org/proper/commons-io

[23] *kuusisto.tinysound* (fork by DrogoniEntity) https://github.com/DrogoniEntity/TinySound

[24] *com.projectdarkstar.ext.jorbis*
https://search.maven.org/artifact/com.projectdarkstar.ext.jorbis/jorbis

[25] *com.googlecode.soundlibs.tritonus-share*
https://search.maven.org/artifact/com.googlecode.soundlibs/tritonus-share/0.3.7.4/bundle

[26] *com.googlecode.soundlibs.vorbisspi*
https://search.maven.org/artifact/com.googlecode.soundlibs/vorbisspi/1.0.3.3/bundle

Sounds used:

[27] Freesound. "Money Bag" by PhilSavlem (licensed under CC0)
https://freesound.org/people/PhilSavlem/sounds/338260/

[28] Freesound. "bay of fundy 01.flac" by tim.kahn (licensed CC BY 3.0)
https://freesound.org/people/tim.kahn/sounds/127569/

[29] Freesound. "bay of fundy 02.flac" by tim.kahn (licensed CC BY 3.0)
https://freesound.org/people/tim.kahn/sounds/127568/

[30] Freesound. "2020-03-17 Lofi Trip Hop" by Doctor_Dreamchip (licensed CC BY 3.0)
https://freesound.org/people/Doctor_Dreamchip/sounds/511279/

[31] Freesound. "Remix of GioMilko Freesound #417960.flac" by Timbre (licensed CC BY-NC 3.0)
https://freesound.org/people/Timbre/sounds/418692/

[32] Freesound. "loopable Lo-fied remix of Tenshi_Mixer freesound #585947.flac" by Timbre (licensed CC BY-NC 3.0) https://freesound.org/people/Timbre/sounds/586988/

[33] Freesound. "Birds In The Forest" by BurghRecords (licensed under CC0)
https://freesound.org/people/BurghRecords/sounds/456123/

[34] Freesound. "Bed of entanglement" by CosmicD (licensed CC BY 3.0)
https://freesound.org/people/CosmicD/sounds/133007/

[35] Freesound. "VOC_150325-0973-1_HK_citywalk.wav" by kevp888 (licensed CC BY 3.0)
https://freesound.org/people/kevp888/sounds/440973/

[36] Freesound. "VOC_150325-0972_HK_citywalk.wav" by kevp888 (licensed CC BY 3.0)
https://freesound.org/people/kevp888/sounds/440974/

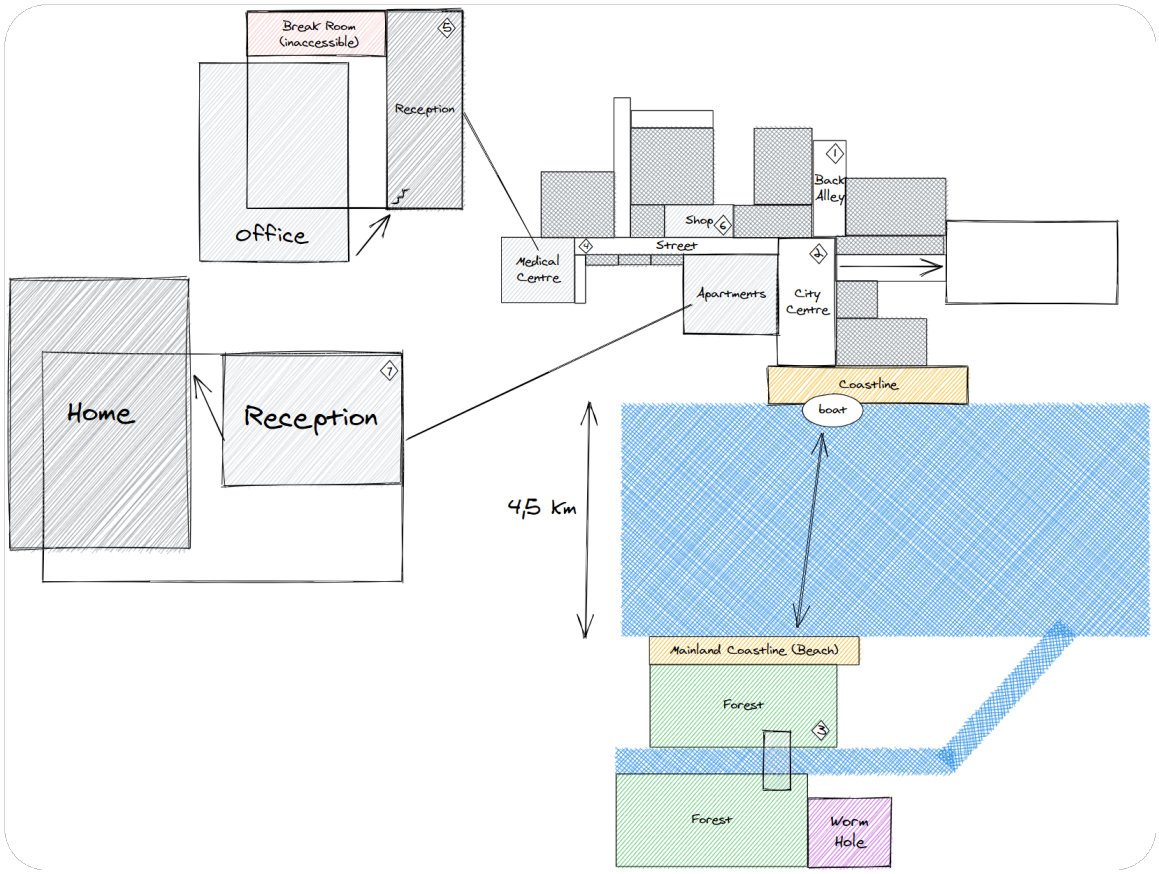

Figure 1: World Map

Figure 2: Shot of Animacity as seen in Episode 1 at 3:02 of Brand New Animal[4]

Figure 3: Michiru Kagemori and Marie Itami pictured left to right in Episode 1 at 12:51 of Brand New Animal[4]

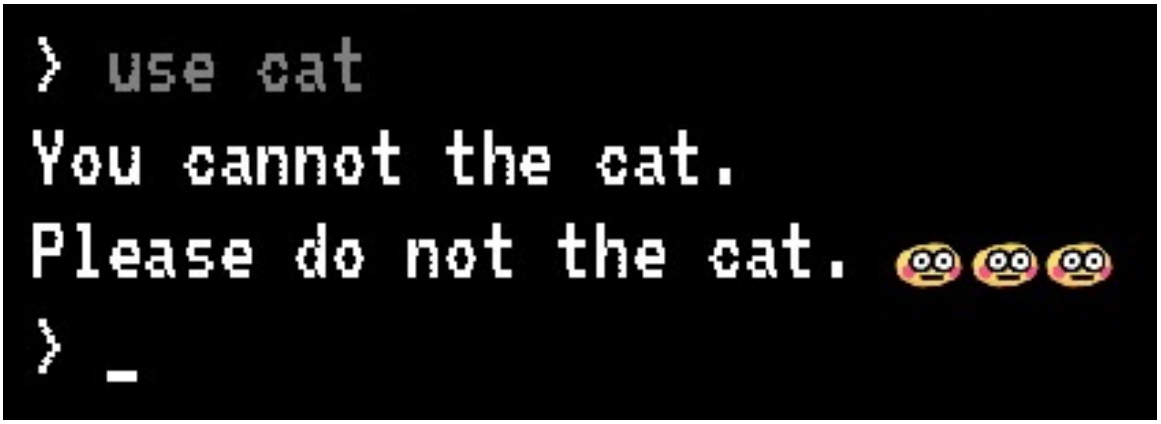

Figure 4: Sample emoji output

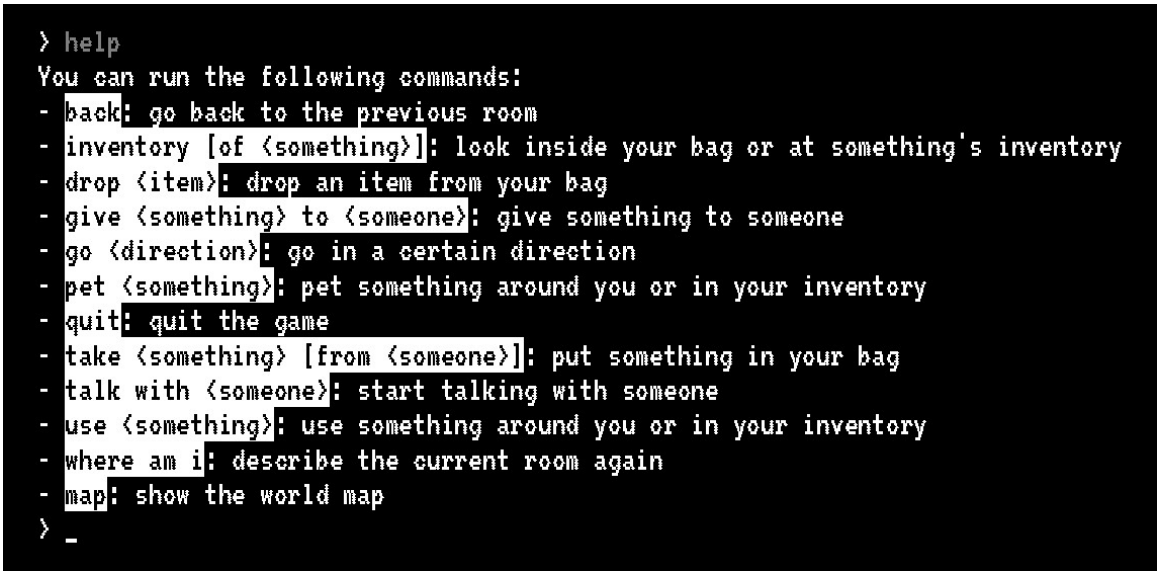

Figure 5: Output from the help command

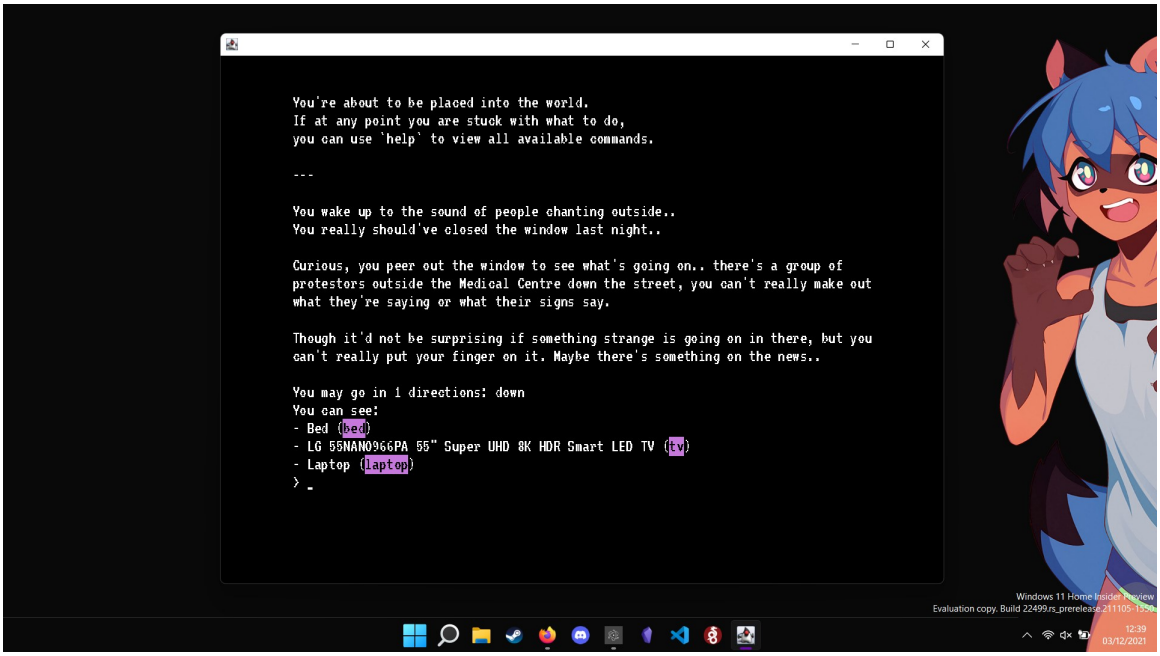

Figure 6: The terminal emulator

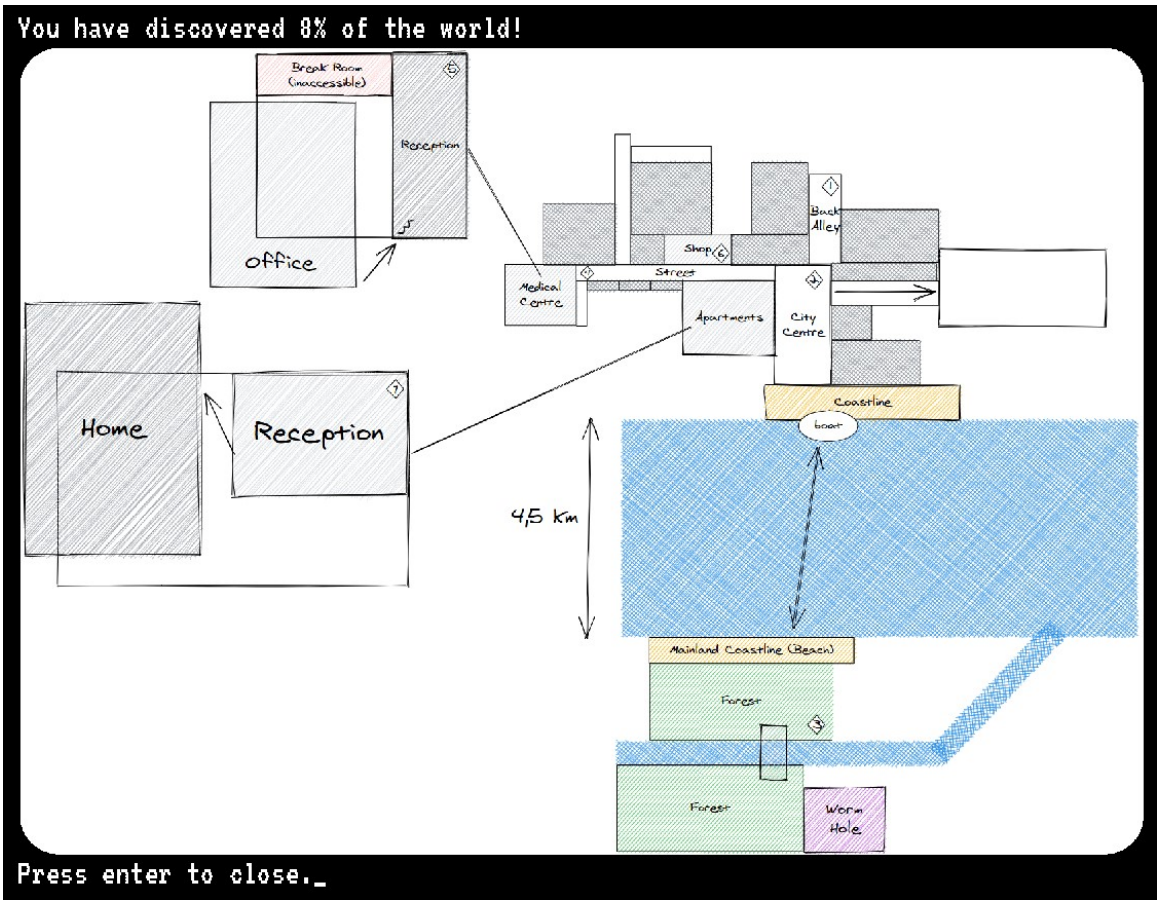

Figure 7: Output from the map command

```java
package uk.insrt.coursework.zuul.events;

/**
 * Represents a single event fired from
 * any source to be consumed by anything.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Event {
    private boolean propagating = true;

    /**
     * Whether this event can continue running.
     * @return Whether propogation of this event was stopped
     */
    public boolean canRun() {
        return this.propagating;
    }

    /**
     * Stop further propagation of this event.
     */
    public void stopPropagation() {
        this.propagating = false;
    }
}
```

Figure 8: Event.java
(uk.insrt.coursework.zuul.events.Event)

```java
package uk.insrt.coursework.zuul.io;

/**
 * Interface representing an arbitrary IO system.
 * This can be implemented to input or output from various interfaces.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public interface IOSystem {
    /**
     * Print a string out through an arbitrary output channel.
     * @param out String to print
     */
    public void print(String out);

    /**
     * Print a string out through an arbitrary output channel and append {@code \n}.
     * @param out String to print
     */
    public void println(String out);

    /**
     * Read a String up until the first encountered {@code \n} from an arbitrary input channel.
     * @return String of line read in
     */
    public String readLine();

    /**
     * Dispose of the arbitrary input and output channels.
     */
    public void dispose();

    /**
     * Clear the output.
     */
    public void clear();
}
```

Figure 9: IOSystem.java
(uk.insrt.coursework.zuul.io.IOSystem)

```java
package uk.insrt.coursework.zuul.io;

import java.util.Scanner;

/**
 * A simple IO system implementation which feeds
 * into System.out and takes data from System.in
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class StandardIO implements IOSystem {
    private Scanner reader;

    /**
     * Construct a new StandardIO.
     */
    public StandardIO() {
        this.reader = new Scanner(System.in);
    }

    @Override
    public void print(String out) {
        System.out.print(out);
    }

    @Override
    public void println(String out) {
        System.out.println(out);
    }

    @Override
    public String readLine() {
        return this.reader.nextLine();
    }

    @Override
    public void dispose() {}

    @Override
    public void clear() {
        this.print("\u000C");
    }
}
```

Figure 10: StandardIO.java
(uk.insrt.coursework.zuul.io.StandardIO)

```java
package uk.insrt.coursework.zuul.io;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.util.Localisation;

/**
 * Translate and localise any incoming output.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class LocalisedIO implements IOSystem {
    private final Pattern pattern = Pattern.compile("<([\\w\\.]+?)>");

    private IOSystem io;
    private Localisation locale;

    /**
     * Construct a new LocalisedIO.
     * @param io Provided IO system we should feed into
     * @param locale Locale to apply to any i18n strings
     */
    public LocalisedIO(IOSystem io, Localisation locale) {
        this.io = io;
        this.locale = locale;
    }

    /**
     * Replace i18n strings in any given String with their actual localised values.
     * Using replacement code from https://stackoverflow.com/a/27359491.
     * @param input String to process
     * @return Final processed string
     */
    private String replace(String input) {
        StringBuffer result = new StringBuffer();
        Matcher matcher = this.pattern.matcher(input);

        while (matcher.find()) {
            matcher.appendReplacement(result, this.locale.get(matcher.group(1)));
        }

        matcher.appendTail(result);
        return result.toString();
    }

    @Override
    public void print(String out) {
        this.io.print(this.replace(out));
    }

    @Override
    public void println(String out) {
        this.io.println(this.replace(out));
    }

    @Override
    public String readLine() {
        return this.io.readLine();
    }

    @Override
    public void dispose() {
        this.io.dispose();
    }

    @Override
    public void clear() {
        this.io.clear();
    }
}
```

Figure 11: LocalisedIO.java
(uk.insrt.coursework.zuul.io.LocalisedIO)

```java
package uk.insrt.coursework.zuul.events;

import java.util.HashMap;
import java.util.HashSet;
import java.util.LinkedHashSet;

/**
 * Event system which manages taking in events
 * from different sources and handles them
 * by firing callbacks on event listeners.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EventSystem {
    private HashMap<Class<? extends Event>, LinkedHashSet<IEventListener<? extends Event>>> listeners = new HashMap<>();

    /**
     * Get existing Event listener list or create a new one if not exists.
     * @param event Event
     * @return Set of event listeners
     */
    private HashSet<IEventListener<? extends Event>> getList(Class<? extends Event> event) {
        var list = this.listeners.get(event);
        if (list == null) {
            list = new LinkedHashSet<>();
            this.listeners.put(event, list);
        }

        return list;
    }

    /**
     * Add a new event listener to this system.
     * @param <E> Generic Event type
     * @param event Event to remove from
     * @param listener Event listener callback
     */
    public<E extends Event> void addListener(Class<E> event, IEventListener<E> listener) {
        this.getList(event).add(listener);
    }

    /**
     * Remove an new event listener from this system.
     * @param <E> Generic Event type
     * @param event Event to remove from
     * @param listener Event listener callback
     */
    public<E extends Event> void removeListener(Class<E> event, IEventListener<E> listener) {
        this.getList(event).remove(listener);
    }

    /**
     * Emit an Event.
     * @param <E> Generic Event type
     * @param event Event to emit
     */
    @SuppressWarnings("unchecked")
    public <E extends Event> void emit(E event) {
        var listeners = this.listeners.get(event.getClass());
        if (listeners == null) return;

        for (@SuppressWarnings("rawtypes") IEventListener listener : listeners) {
            listener.onEvent(event);
            // Previously, there was a try catch ClassCastException
            // but I've since constricted the types on `addListener`
            // and `removeListener` so this should never happen.

            if (!event.canRun())
                break;
        }
    }
}
```


Figure 12: EventSystem.java
(uk.insrt.coursework.zuul.events.EventSystem)

```java
package uk.insrt.coursework.zuul.dialogue;

import uk.insrt.coursework.zuul.io.IOSystem;

/**
 * An option which branches off a {@link DialogueNode} into another node.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class DialogueOption<T> {
    private IDialogueHandler<T> handler;

    private String description;
    private boolean shouldExit;
    private T target;

    /**
     * Construct a new simple DialogueOption with a description and destination.
     * @param description Description of this option
     * @param target Target node to jump to
     */
    public DialogueOption(String description, T target) {
        this.target = target;
        this.description = description;
    }

    /**
     * Construct a complex DialogueOption with a description and select handler.
     * @param description Description of this option
     * @param handler Method called when this option is selected
     */
    public DialogueOption(String description, IDialogueHandler<T> handler) {
        this.handler = handler;
        this.description = description;
    }

    /**
     * Tell the Dialogue system to exit if this option is selected.
     * @return This dialogue option so method calls can be chained
     */
    public DialogueOption<T> mustExit() {
        this.shouldExit = true;
        return this;
    }

    /**
     * Get the description of this option.
     * @return Description string
     */
    public String getDescription() {
        return this.description;
    }

    /**
     * Get the destination of this option.
     * @return Destination if it exists
     */
    public T getTarget() {
        return this.target;
    }

    /**
     * Handle the player selecting this dialogue option.
     * @param io Provided IO system
     * @return The new node or null if we should exit and stay put.
     */
    public T handle(IOSystem io) {
        if (this.handler != null) {
            return this.handler.onAction(io);
        } else if (!this.shouldExit) {
            return this.target;
        }

        return null;
    }
}
```


Figure 13: DialogueOption.java
(uk.insrt.coursework.zuul.dialogue.DialogueOption)

```java
package uk.insrt.coursework.zuul.ui;

import java.awt.Graphics;

import uk.insrt.coursework.zuul.events.Event;

/**
 * Event fired when the terminal emulator draws a new frame.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EventDraw extends Event {
    private Graphics g;
    private float ox;
    private float oy;
    private float fw;
    private float fh;

    /**
     * Construct a new EventDraw Event.
     * @param g Graphics context
     * @param ox Origin X position
     * @param oy Origin Y position
     * @param fw Font character width
     * @param fh Font character height
     */
    public EventDraw(Graphics g, float ox, float oy, float fw, float fh) {
        this.g = g;
        this.ox = ox;
        this.oy = oy;
        this.fw = fw;
        this.fh = fh;
    }

    /**
     * Get the Graphics relating to this event.
     * @return Graphics
     */
    public Graphics getGraphics() {
        return this.g;
    }

    /**
     * Get the origin X position of the contents of the terminal.
     * @return X position
     */
    public float getOriginX() {
        return this.ox;
    }

    /**
     * Get the origin Y position of the contents of the terminal.
     * @return Y position
     */
    public float getOriginY() {
        return this.oy;
    }

    /**
     * Get the character width.
     * @return Character width
     */
    public float getCharWidth() {
        return this.fw;
    }

    /**
     * Get the character height.
     * @return Character height
     */
    public float getCharHeight() {
        return this.fh;
    }
}
```

Figure 14: EventDraw.java
(uk.insrt.coursework.zuul.ui.EventDraw)

```java
public<E extends Event> void addListener(Class<E> event,
    IEventListener<E> listener) {
    this.getList(event).add(listener);
}
```

Figure 15: Excerpt from EventSystem.java
(uk.insrt.coursework.zuul.events.EventSystem)

```java
/**
 * Get the current weight of this inventory.
 * @return Weight (in kg)
 */
public double getWeight() {
    return this
        .items
        .stream()
        .mapToDouble(Entity::getWeight)
        .sum();
}

/**
 * Check if the inventory is full.
 * @return True if the weight is greater than the max weight
 */
public boolean isFull() {
    return this.getWeight() >= this.getMaxWeight();
}

/**
 * Add an entity to this inventory.
 *
 * There must be sufficient space for the entity.
 * @param entity Target Entity
 * @return Whether we successfully added the new entity.
 */
public boolean add(Entity entity) {
    if (this.getWeight() + entity.getWeight() > this.maxWeight) {
        return false;
    }

    this.items.add(entity);
    return true;
}
```

Figure 16: Excerpt from Inventory.java
(uk.insrt.coursework.zuul.entities.Inventory)

```java
/**
 * Write a string value to the text buffer.
 * @param value String value to write
 */
public void write(String value) {
    // Write each character sequentially.
    for (int i=0;i<value.length();i++) {
        char c = value.charAt(i);

        // If we encounter an Ansi escape character, then take the
        // substring from this point on and determine if it is a valid
        // escape code. If it is, apply any changes before continuing.
        if (c == '\u001B') {
            Matcher matcher = Ansi.AnsiPattern.matcher(value.substring(i));
            if (matcher.find()) {
                int v = Integer.parseInt(matcher.group(1));
                i += 3 + (v > 9 ? 1 : 0);

                if (v == 0) {
                    this.bg = Color.BLACK;
                    this.fg = Color.WHITE;
                } else if (v >= 30 && v < 38) {
                    this.fg = Ansi.fromEscapeCode(v);
                } else if (v >= 40 && v < 48) {
                    this.bg = Ansi.fromEscapeCode(v);
                }

                continue;
            }
        }

        this.write(c);
    }
}
```

Figure 17: Excerpt from TextBuffer.java
(uk.insrt.coursework.zuul.ui.TextBuffer)

```java
package uk.insrt.coursework.zuul.commands;

import java.util.regex.Matcher;

import uk.insrt.coursework.zuul.world.Direction;

/**
 * Wrapper around regex Matcher for deriving the values of given arguments in commands.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Arguments {
    private Matcher matcher;

    /**
     * Construct a new Arguments wrapper.
     * @param matcher Regex Matcher
     */
    public Arguments(Matcher matcher) {
        this.matcher = matcher;
    }

    /**
     * Take a named group from the Matcher.
     * We assume that the provided Regex doesn't match the group if it's empty.
     * @param group Named group
     * @return String value of named group or null if it doesn't exist
     */
    public String group(String group) {
        // We ignore and return null on error as a bit of convenience.
        // This may not be best practice but it is justified by the fact
        // that it avoids an incredible amount of boilerplate further up
        // the chain, and in my opinion that's worth this design decision.
        try {
            return this.matcher.group(group);
        } catch (Exception e) {
            return null;
        }
    }

```

```java
    /**
     * Check whether this named group was matched.
     * @param group Named group
     * @return Whether this named group was matched
     */
    public boolean has(String group) {
        return this.group(group) != null;
    }

    /**
     * Get the provided Direction.
     * @return Parsed Direction value
     */
    public Direction direction() {
        return Direction.fromString(this.group("direction"));
    }
}
```

```java
package uk.insrt.coursework.zuul.commands;

import java.util.List;
import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.Inventory;
import uk.insrt.coursework.zuul.util.Search;
import uk.insrt.coursework.zuul.world.World;

/**
 * Representation of an action which can be performed by the user.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public abstract class Command {
    private Pattern[] patterns;
    private String syntax;
    private String usage;

    /**
     * Construct a new Command.
     * @param syntax Information about how to use the command
     * @param usage Information about what the command does
     * @param patterns Patterns to execute this command on
     */
    public Command(String syntax, String usage, Pattern[] patterns) {
        this.patterns = patterns;
        this.syntax = syntax;
        this.usage = usage;
    }

    /**
     * Get information about how to use the command.
     * @return String Information about how to use the command
     */
    public String getSyntax() {
        return this.syntax;
    }

```

```java
    /**
     * Get information about what the command does.
     * @return String Information about what the command does
     */
    public String getUsage() {
        return this.usage;
    }

    /**
     * Get all applicable patterns to match to execute this command.
     * @return Regex Pattern array
     */
    public Pattern[] getPatterns() {
        return this.patterns;
    }

    /**
     * Check whether this command is visible to the player in the help menu.
     * @return True if visible
     */
    public boolean isVisible() {
        return true;
    }

    /**
     * Run this command within the scope of a world and with any parsed arguments.
     * @param world Current World object
     * @param args Arguments passed into command
     * @return Boolean indicating whether the game loop should exit.
     */
    public abstract boolean run(World world, Arguments args);

    /**
     * Filter entities by those that are in the current room.
     */
    public static final int FILTER_ROOM = 1;

    /**
     * Filter entities by those that are in the player's inventory.
     */
    public static final int FILTER_INVENTORY = 2;
```

```
 83
 84      /**
 85       * Don't filter entities and instead search through both the current room and the player's inventory.
 86       */
 87      public static final int FILTER_ALL = FILTER_ROOM + FILTER_INVENTORY;
 88
 89      /**
 90       * Given a World and filter, use the provided Arguments and the relevant group to find an entity.
 91       * If an Entity is not found or not provided, the appropriate error is displayed to the player.
 92       * @param world World to look for the Entity within
 93       * @param filter Integer value which represents the filtering, specify one of: {@link #FILTER_ROOM}, {@link
#FILTER_INVENTORY}, {@link #FILTER_ALL}
 94       * @param args Arguments object to pull information out of
 95       * @param group Group we should pull the Entity query out of
 96       * @param failure Failure message if an Entity is not specified
 97       * @return An Entity if one is found, or null if one isn't.
 98       */
 99      public Entity findEntity(World world, int filter, Arguments args, String group, String failure) {
100          String name = args.group(group);
101          if (name == null) {
102              world.getIO().println(failure);
103              return null;
104          }
105
106          // Search the inventory first.
107          Entity player = world.getPlayer();
108          Entity entity = null;
109          if ((filter & FILTER_INVENTORY) == FILTER_INVENTORY) {
110              Inventory inventory = player.getInventory();
111              entity = Search.findEntity(inventory.getItems(), name, true);
112          }
113
114          // If we haven't found an entity yet, search the room.
115          if (entity == null
116           && (filter & FILTER_ROOM) == FILTER_ROOM) {
117              List<Entity> entities = world.getEntitiesInRoom(player.getRoom());
118              entity = Search.findEntity(entities, name, true);
119          }
120
121          if (entity == null) {
122              world.getIO().println("<selectors.cant_find.1> " + name + " <selectors.cant_find.2>.");
```

```
123             }
124
125             return entity;
126         }
127
128         /**
129          * Given a World and filter, use the provided Arguments and using the group "entity" to find an entity.
130          * If an Entity is not found or not provided, the appropriate error is displayed to the player.
131          * @param world World to look for the Entity within
132          * @param filter Integer value which represents the filtering, specify one of: {@link #FILTER_ROOM}, {@link
#FILTER_INVENTORY}, {@link #FILTER_ALL}
133          * @param args Arguments object to pull information out of
134          * @param failure Failure message if an Entity is not specified
135          * @return An Entity if one is found, or null if one isn't.
136          */
137         public Entity findEntity(World world, int filter, Arguments args, String failure) {
138             return this.findEntity(world, filter, args, "entity", failure);
139         }
140
141         /**
142          * Given a World and using the {@link #FILTER_ROOM} filter, use the provided Arguments and using the group "entity"
to find an entity.
143          * If an Entity is not found or not provided, the appropriate error is displayed to the player.
144          * @param world World to look for the Entity within
145          * @param args Arguments object to pull information out of
146          * @param failure Failure message if an Entity is not specified
147          * @return An Entity if one is found, or null if one isn't.
148          */
149         public Entity findEntity(World world, Arguments args, String failure) {
150             return this.findEntity(world, FILTER_ROOM, args, failure);
151         }
152
153         /**
154          * Given a World and using the {@link #FILTER_ROOM} filter, use the provided Arguments and the relevant group to
find an entity.
155          * If an Entity is not found or not provided, the appropriate error is displayed to the player.
156          * @param world World to look for the Entity within
157          * @param args Arguments object to pull information out of
158          * @param group Group we should pull the Entity query out of
159          * @param failure Failure message if an Entity is not specified
160          * @return An Entity if one is found, or null if one isn't.
```

```
161         */
162        public Entity findEntity(World world, Arguments args, String group, String failure) {
163            return this.findEntity(world, FILTER_ROOM, args, group, failure);
164        }
165    }
```

```java
package uk.insrt.coursework.zuul.commands;

import java.util.ArrayList;
import java.util.List;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.core.CommandBack;
import uk.insrt.coursework.zuul.commands.core.CommandBag;
import uk.insrt.coursework.zuul.commands.core.CommandDrop;
import uk.insrt.coursework.zuul.commands.core.CommandGive;
import uk.insrt.coursework.zuul.commands.core.CommandGo;
import uk.insrt.coursework.zuul.commands.core.CommandHelp;
import uk.insrt.coursework.zuul.commands.core.CommandPet;
import uk.insrt.coursework.zuul.commands.core.CommandQuit;
import uk.insrt.coursework.zuul.commands.core.CommandTake;
import uk.insrt.coursework.zuul.commands.core.CommandTalk;
import uk.insrt.coursework.zuul.commands.core.CommandUse;
import uk.insrt.coursework.zuul.commands.core.CommandWhereAmI;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command handler which constructs, then resolves
 * and executes commands from an arbitrary input.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandManager {
    private ArrayList<Command> commands = new ArrayList<>();

    /**
     * Construct a new CommandManager.
     *
     * You should only need one present at any given time.
     */
    public CommandManager() {
        this.initialiseCommands();
    }

    /**
```

```
42          * Register a new Command.
43          * @param command Command
44          */
45         public void registerCommand(Command command) {
46             this.commands.add(command);
47         }
48
49         /**
50          * Register multiple commands.
51          * @param commands Command array
52          */
53         public void registerCommands(Command[] commands) {
54             for (Command command : commands) {
55                 this.registerCommand(command);
56             }
57         }
58
59         /**
60          * Get Commands provided by this Command manager.
61          * @return List of Commands
62          */
63         public List<Command> getCommands() {
64             return this.commands;
65         }
66
67         /**
68          * Initialise all the commands a player can execute.
69          */
70         private void initialiseCommands() {
71             final Command[] DEFAULT_COMMANDS = {
72                 new CommandBack(),
73                 new CommandBag(),
74                 new CommandDrop(),
75                 new CommandGive(),
76                 new CommandGo(),
77                 new CommandHelp(this),
78                 new CommandPet(),
79                 new CommandQuit(),
80                 new CommandTake(),
81                 new CommandTalk(),
82                 new CommandUse(),
```

```
83                  new CommandWhereAmI(),
84              };
85
86              this.registerCommands(DEFAULT_COMMANDS);
87          }
88
89          /**
90           * Interpret a given command and execute it within the scope of a given world.
91           * @param world Current World object
92           * @param cmd Arbitrary input to match against
93           * @return Boolean indicating whether the game loop should exit.
94           */
95          public boolean runCommand(World world, String cmd) {
96              for (Command command : this.commands) {
97                  for (Pattern pattern : command.getPatterns()) {
98                      Matcher matcher = pattern.matcher(cmd);
99                      if (matcher.find()) {
100                         Arguments arguments = new Arguments(matcher);
101                         return command.run(world, arguments);
102                     }
103                 }
104             }
105
106             world.getIO().println("<commands.unknown>");
107             return false;
108         }
109 }
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the Player to walk back through the previous Rooms.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandBack extends Command {
    public CommandBack() {
        super("back", "<commands.back>",
            new Pattern[] {
                Pattern.compile("^(?:(?:go|walk)\\s+)*back(?!\\w)"),
                // back, go back, walk back
            });
    }

    @Override
    public boolean run(World world, Arguments arguments) {
        // We call a specialised method on the player as we keep
        // track of visited rooms within the Player class itself.
        world.getPlayer().back();
        return false;
    }
}
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.text.DecimalFormat;
import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.Inventory;
import uk.insrt.coursework.zuul.io.Ansi;
import uk.insrt.coursework.zuul.io.IOSystem;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the Player to look at their or another Entity's inventory.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandBag extends Command {
    private final DecimalFormat format = new DecimalFormat("0.00");

    public CommandBag() {
        super("inventory [of <selectors.something>]", "<commands.bag.usage>",
            new Pattern[] {
                Pattern.compile("^(?:b(?:ag)*|inv(?:entory)*)(?:\\s+(?<entity>[\\w\\s]+))*"),
                // b, bag, inv, inventory, bag of <entity>, inventory of <entity>, (+2)
            });
    }

    @Override
    public boolean run(World world, Arguments arguments) {
        IOSystem io = world.getIO();

        // Figure out if we're checking our own inventory or another entity's inventory.
        Entity entity;
        boolean ours;
        if (arguments.has("entity")) {
            ours = false;
            entity = this.findEntity(world, arguments, "<commands.bag.cant_find>");
            if (entity == null) return false;
```

```
42          } else {
43              ours = true;
44              entity = world.getPlayer();
45          }
46
47          // Get the selected entity's inventory and provide output if empty.
48          Inventory inv = entity.getInventory();
49          if (inv.getWeight() == 0) {
50              if (ours) {
51                  io.println("<commands.bag.empty> <commands.bag.can_carry_kg> "
52                      + inv.getMaxWeight() + " kg.");
53              } else {
54                  io.println(entity.getHighlightedName() + " <commands.bag.entity_empty>.");
55              }
56
57              return false;
58          }
59
60          // Otherwise describe some statistics about the inventory.
61          if (ours) {
62              io.println("<commands.bag.are_carrying_kg> " + this.format.format(inv.getWeight())
63                  + " / " + inv.getMaxWeight() + " kg.\n<commands.bag.look_in_bag>:");
64          } else {
65              io.println(entity.getHighlightedName() + " <commands.bag.entity_appears_to_have>:");
66          }
67
68          // Describe all the items in this inventory we are currently looking at.
69          for (Entity item : inv.getItems()) {
70              io.println("- " + Ansi.Yellow + item.getWeight() + " kg"
71                  + Ansi.Reset + " " + item.describe()
72                  + " (" + item.getHighlightedName() + ")");
73          }
74
75          return false;
76      }
77  }
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.io.Ansi;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the Player to drop any item in their inventory.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandDrop extends Command {
    public CommandDrop() {
        super("drop <selectors.item>", "<commands.drop.usage>",
            new Pattern[] {
                Pattern.compile("^(?:drop|place|put down)(?:\\s+(?<entity>[\\w\\s]+))*")
                // drop, place, put down, drop <item>, place <item>, put down <item>
            });
    }

    @Override
    public boolean run(World world, Arguments args) {
        // Find the given entity within our inventory and drop it if it's found.
        Entity entity = this.findEntity(world, Command.FILTER_INVENTORY, args, "<commands.drop.nothing_specified>");
        if (entity != null) {
            world.getIO().println("<commands.drop.dropped.1> " + Ansi.BackgroundWhite + Ansi.Black
                + entity.getName() + Ansi.Reset + " <commands.drop.dropped.2>!");
            entity.setLocation(world.getPlayer().getRoom());
        }

        return false;
    }
}
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.EntityPlayer;
import uk.insrt.coursework.zuul.entities.actions.IGiveable;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the player to give something to someone.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandGive extends Command {
    public CommandGive() {
        super("give <selectors.something> to <selectors.someone>", "<commands.give.usage>",
            new Pattern[] {
                Pattern.compile("^(?:give|put)(?:\\s+(?<item>[\\w\\s]+)\\s+(?:to|in)\\s+(?<entity>[\\w\\s]+))*")
                // give, put, give <something> to <someone>, put <something> in <something>, (+2)
            });
    }

    @Override
    public boolean run(World world, Arguments args) {
        // Find the entity we want to give in the room or our inventory.
        Entity item = this.findEntity(world, Command.FILTER_ALL, args, "item", "<commands.give.nothing_specified>");
        if (item == null) return false;

        // Explicitly deny the player being given to anything, otherwise we will end up in an inventory.
        var io = world.getIO();
        if (item instanceof EntityPlayer) {
            io.println("<commands.give.denied_player>");
            return false;
        }

        // Find the target to give to.
        Entity target = this.findEntity(world, args, "<commands.give.no_target>");
```

```java
        if (target == null) return false;

        // If the target entity is an IGiveable, check if they accept this item.
        if (target instanceof IGiveable) {
            ((IGiveable) target).give(item);
        } else {
            io.println("<commands.give.denied.1> " + item.getHighlightedName()
                + " <commands.give.denied.2> " + target.getHighlightedName() + "!");
        }

        return false;
    }
}
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the Player to walk in a particular Direction.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandGo extends Command {
    public CommandGo() {
        super("go <selectors.direction>", "<commands.go.usage>",
            new Pattern[] {
                Pattern.compile("^(?:go|walk)(?:\\s+(?<direction>[\\w\\s]+))*")
                // go, walk, go <direction>, walk <direction>
            });
    }

    @Override
    public boolean run(World world, Arguments arguments) {
        Direction direction = arguments.direction();
        if (direction == null) {
            world.getIO().println("<commands.go.nothing_specified>");
            return false;
        }

        world.getPlayer().go(direction);
        return false;
    }
}
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;
import java.util.stream.Collectors;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.commands.CommandManager;
import uk.insrt.coursework.zuul.io.Ansi;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the player to list all the available commands.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandHelp extends Command {
    private CommandManager commandManager;

    public CommandHelp(CommandManager commandManager) {
        super("help", "<commands.help.usage>",
            new Pattern[] {
                Pattern.compile("^(?:h(?:elp)*)(?!\\w)")
                // h, help
            });

        this.commandManager = commandManager;
    }

    @Override
    public boolean run(World world, Arguments arguments) {
        // Describe all the commands the player can run.
        world.getIO()
            .println(
            "<commands.help.can_run>\n" +
            this.commandManager
                .getCommands()
                .stream()
                .filter(Command::isVisible)
                .map(c -> "- " + Ansi.BackgroundWhite + Ansi.Black
```

```
                    + c.getSyntax() + Ansi.Reset + ": " + c.getUsage())
                .collect(Collectors.joining("\n"))
        );

        return false;
    }

    @Override
    public boolean isVisible() {
        return false;
    }
}
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.actions.IPettable;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the player to pet another entity.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandPet extends Command {
    public CommandPet() {
        super("pet <selectors.something>", "<commands.pet.usage>",
            new Pattern[] {
                Pattern.compile("^pet(?:\\s+(?<entity>[\\w\\s]+))*")
                // pet, pet <something>
            });
    }

    @Override
    public boolean run(World world, Arguments args) {
        // Scan the room for entities that have IPettable.
        Entity entity = this.findEntity(world, Command.FILTER_ALL, args, "<commands.pet.nothing_specified>");
        if (entity != null) {
            if (entity instanceof IPettable) {
                ((IPettable) entity).pet();
            } else {
                world.getIO().println("<commands.pet.denied> " + entity.getHighlightedName() + ".");
            }
        }

        return false;
    }
}
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the player to quit the game.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandQuit extends Command {
    public CommandQuit() {
        super("quit", "<commands.quit>",
            new Pattern[] {
                Pattern.compile("^quit|exit(?!\\w)"),
                // quit
            });
    }

    @Override
    public boolean run(World world, Arguments arguments) {
        // We return true from run() in order to tell the game loop to exit.
        return true;
    }
}
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.Inventory;
import uk.insrt.coursework.zuul.io.IOSystem;
import uk.insrt.coursework.zuul.util.Search;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the Player to take an Entity and put it in their Inventory.
 * They may also take these Entities from other Entity Inventories.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandTake extends Command {
    public CommandTake() {
        super("take <selectors.something> [from <selectors.someone>]", "<commands.take.usage>",
            new Pattern[] {
                Pattern.compile("^take\\s+(?<entity>[\\w\\s]+)\\s+from\\s+(?<other>[\\w\\s]+)"),
                Pattern.compile("^take(?:\\s+(?<entity>[\\w\\s]+))*")
                // take, take <item>, take <item> from <entity>
            });
    }

    @Override
    public boolean run(World world, Arguments args) {
        IOSystem io = world.getIO();
        Entity player = world.getPlayer();
        Inventory target = player.getInventory();

        // Detect if we are taking from another entity, in that case run different logic.
        if (args.has("other")) {
            String name = args.group("entity");
            Entity other = this.findEntity(world, args, "other", "<commands.take.nothing_specified>");
            if (other == null) return false;

```

```
42              Entity item = Search.findEntity(other.getInventory().getItems(), name, true);
43              if (item == null) {
44                  io.println(other.getHighlightedName() + " <commands.take.entity_does_not_have_entity> " + name + "!");
45                  return false;
46              }
47
48              if (item.setLocation(target)) {
49                  io.println("<commands.take.took.1> " + item.getHighlightedName()
50                      + " <commands.take.took.2> " + other.getHighlightedName()
51                      + " <commands.take.took.3>.");
52              } else {
53                  io.println("<commands.take.denied.1> " + item.getName()
54                      + ", <commands.take.denied.2>.");
55              }
56
57              return false;
58          }
59
60          // Otherwise, look around the room and find something we can take.
61          Entity entity = this.findEntity(world, args, "<commands.take.item_not_specified>");
62          if (entity != null) {
63              if (entity == player) {
64                  io.println("\u1F633");
65                  return false;
66              }
67
68              if (entity.setLocation(target)) {
69                  io.println("<commands.take.took.1> " + entity.getHighlightedName()
70                      + " <commands.take.took.3>.");
71              } else {
72                  io.println("<commands.take.denied.1> " + entity.getHighlightedName()
73                      + ", <commands.take.denied.2>.");
74              }
75          }
76
77          return false;
78      }
79  }
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.actions.ITalkwith;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the Player to talk with other Entities.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandTalk extends Command {
    public CommandTalk() {
        super("talk with <selectors.someone>", "<commands.talk.usage>",
            new Pattern[] {
                Pattern.compile("^talk(?:(?:\\s*with|to)*(?:\\s+(?<entity>[\\w\\s]+))*)*")
                // talk, talk with <entity>, talk to <entity>
            });
    }

    @Override
    public boolean run(World world, Arguments args) {
        Entity entity = this.findEntity(world, args, "<commands.talk.nothing_specified>");
        if (entity != null) {
            if (entity instanceof ITalkwith) {
                ((ITalkwith) entity).talk();
            } else {
                world.getIO().println("<commands.talk.denied> " + entity.getHighlightedName() + ".");
            }
        }

        return false;
    }
}
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.actions.IUseable;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the Player to use an Entity.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandUse extends Command {
    public CommandUse() {
        super("use <selectors.something>", "<commands.use.usage>",
            new Pattern[] {
                Pattern.compile("^use(?:\\s+(?<entity>[\\w\\s]+))*")
                // use, use <entity>
            });
    }

    @Override
    public boolean run(World world, Arguments args) {
        Entity entity = this.findEntity(world, Command.FILTER_ALL, args, "<commands.use.nothing_specified>");
        if (entity != null) {
            if (entity instanceof IUseable) {
                ((IUseable) entity).use(world.getPlayer());
            } else {
                world.getIO().println("<commands.use.denied> " + entity.getHighlightedName() + ".");
            }
        }

        return false;
    }
}
```

```java
package uk.insrt.coursework.zuul.commands.core;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.events.world.EventEntityEnteredRoom;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command which allows the player to reorient themselves in the world.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandWhereAmI extends Command {
    public CommandWhereAmI() {
        super("where am i", "<commands.where_am_i>",
            new Pattern[] {
                Pattern.compile("^where(\\s+am\\s+(i(?!\\w))*)*"),
                // where am i
            });
    }

    @Override
    public boolean run(World world, Arguments arguments) {
        // We can just re-emit the enter room event to
        // trigger the room description logic to run again.
        world.emit(new EventEntityEnteredRoom(world.getPlayer()));
        return false;
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign;

import java.io.IOException;
import java.util.HashSet;
import java.util.stream.Collectors;

import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Stage;
import uk.insrt.coursework.zuul.content.campaign.entities.EntityCat;
import uk.insrt.coursework.zuul.content.campaign.entities.EntityWithDialogue;
import uk.insrt.coursework.zuul.content.campaign.events.EventGameStageChanged;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomApartmentsHome;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomApartmentsReception;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomBackAlley;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomCityCentre;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomCoastline;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomForest;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomMainlandCoastline;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomMedicalCentreOffice;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomMedicalCentreReception;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomShop;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomStreet;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomWormHole;
import uk.insrt.coursework.zuul.dialogue.DialogueLoader;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.EntityPlayer;
import uk.insrt.coursework.zuul.events.IEventListener;
import uk.insrt.coursework.zuul.events.world.EventEntityEnteredRoom;
import uk.insrt.coursework.zuul.events.world.EventEntityLeftRoom;
import uk.insrt.coursework.zuul.io.Ansi;
import uk.insrt.coursework.zuul.io.IOSystem;
import uk.insrt.coursework.zuul.sound.EventMusic;
import uk.insrt.coursework.zuul.sound.MusicType;
import uk.insrt.coursework.zuul.world.Room;
import uk.insrt.coursework.zuul.world.World;

/**
 * The main campaign World.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.1-SNAPSHOT
 */
```

```java
public class CampaignWorld extends World {
    private StoryFlags flags;
    private HashSet<Room> visitedRooms;
    private DialogueLoader dialogueLoader;

    /**
     * Construct a new Campaign World
     * @param io Provided IO system
     */
    public CampaignWorld(IOSystem io) {
        super(io);

        this.visitedRooms = new HashSet<>();
        this.dialogueLoader = new DialogueLoader();
        this.flags = new StoryFlags(this.getEventSystem());

        try {
            this.dialogueLoader.load("/dialogue.toml");
        } catch (IOException e) {
            System.err.println("Failed to load resources for campaign world!");
            e.printStackTrace();
        }

        this.buildWorld();
        this.spawnEntities();
        this.registerEvents();
    }

    /**
     * Get this World's Dialogue Loader
     * @return Dialogue Loader
     */
    public DialogueLoader getDialogueLoader() {
        return this.dialogueLoader;
    }

    /**
     * Get the global story flags.
     * @return Story flags instance
     */
    public StoryFlags getStoryFlags() {
```

```
83          return this.flags;
84      }
85
86      /**
87       * Check whether the Player has visited a certain Room yet
88       * @param room Room to check
89       * @return True if the Player has visited the given Room
90       */
91      public boolean hasVisited(Room room) {
92          return this.visitedRooms.contains(room);
93      }
94
95      /**
96       * Get a rounded whole number percentage of how much the World has been explored.
97       * @return Integer representing percentage of World explored
98       */
99      public int percentVisited() {
100         return Math.round((float) this.visitedRooms.size() / this.rooms.size() * 100.0f);
101     }
102
103     /**
104      * Create all the Worlds and link adjacent Rooms together.
105      */
106     private void buildWorld() {
107         final Room[] rooms = {
108             new RoomCityCentre(this),
109             new RoomStreet(this),
110             new RoomShop(this),
111             new RoomBackAlley(this),
112             new RoomApartmentsReception(this),
113             new RoomApartmentsHome(this),
114             new RoomMedicalCentreReception(this),
115             new RoomMedicalCentreOffice(this),
116             new RoomCoastline(this),
117             new RoomMainlandCoastline(this),
118             new RoomForest(this),
119             new RoomWormHole(this)
120         };
121
122         for (Room room : rooms) {
123             this.addRoom(room);
```

```
124         }
125
126         this.linkRooms();
127     }
128
129     /**
130      * Spawn and setup any Entities within this World.
131      */
132     private void spawnEntities() {
133         for (Room room : this.rooms.values()) {
134             room.spawnEntities();
135         }
136
137         // Entangle boat inventories.
138         Entity boat1 = this.entities.get("boat1");
139         Entity boat2 = this.entities.get("boat2");
140         boat1.entangleInventory(boat2.getInventory());
141     }
142
143     /**
144      * Register all the game logic.
145      */
146     private void registerEvents() {
147         // Capture all Events for Entities entering Rooms.
148         this.eventSystem.addListener(EventEntityEnteredRoom.class,
149             event -> {
150                 Entity entity = event.getEntity();
151                 if (entity instanceof EntityPlayer) {
152                     Room room = entity.getRoom();
153
154                     // Whenever the Player enters a Room, we should print the
155                     // description of the Room and list things found in the Room.
156                     this.io.println(
157                         room.describe()
158                             + "\n<global.can_go_in_x_directions.1> "
159                             + room.getDirections().size()
160                             + " <global.can_go_in_x_directions.2>: "
161                             + room.getDirections()
162                                 .stream()
163                                 .map(x ->
164                                     x.toString()
```

```
165                                     .toLowerCase()
166                                     .replaceAll("_", " ")
167                                 )
168                                 .collect(Collectors.joining(", "))
169                     );
170
171                     // Mark current room as previously visited.
172                     this.visitedRooms.add(room);
173
174                     // When we enter a new room, list what we can see.
175                     String entities = this.getEntitiesInRoom(entity.getRoom())
176                         .stream()
177                         .filter(e -> !(e instanceof EntityPlayer))
178                         .map(e -> "- " + e.describe() + " ("
179                             + Ansi.BackgroundPurple + Ansi.Black
180                             + e.getName() + Ansi.Reset + ")")
181                         .collect(Collectors.joining("\n"));
182
183                     if (entities.length() > 0) {
184                         this.io.println("<global.sight>\n" + entities);
185                     }
186                 } else {
187                     // If another entity enters the room,
188                     // conditionally mention this to the player.
189                     EntityPlayer player = this.getPlayer();
190                     if (entity.getRoom() == player.getRoom()) {
191                         if (entity instanceof EntityCat) {
192                             this.io.println("\n<entities.cat.enter>");
193                         }
194                     }
195                 }
196             });
197
198         // Capture all Events for Entities leaving Rooms.
199         this.eventSystem.addListener(EventEntityLeftRoom.class,
200             event -> {
201                 Entity entity = event.getEntity();
202                 if (entity instanceof EntityPlayer) return;
203
204                 Room room = event.getRoom();
205                 if (room != this.player.getRoom()) return;
```

```
206
207                 // If another entity leaves the room,
208                 // conditionally mention this to the player.
209                 if (entity instanceof EntityCat) {
210                     this.io.println("\n<entities.cat.leave>");
211                 }
212             });
213
214         // Register event for game stage changing.
215         this.eventSystem.addListener(EventGameStageChanged.class,
216             event -> {
217                 Stage stage = event.getStage();
218                 switch (stage) {
219                     case Recon: {
220                         for (Entity entity : this.entities.values()) {
221                             if (entity instanceof EntityWithDialogue) {
222                                 ((EntityWithDialogue<?>) entity).setDialogueNodeIfPresent("recon");
223                             }
224                         }
225                         break;
226                     }
227                     case End: {
228                         io.println("<stage.reached_conclusion>");
229                         break;
230                     }
231                     default: break;
232                 }
233             });
234
235         // Register required Events for Worm Hole room to function.
236         @SuppressWarnings("unchecked")
237         var wh = (IEventListener<EventEntityEnteredRoom>) this.getRoom("Worm Hole");
238         this.eventSystem.addListener(EventEntityEnteredRoom.class, wh);
239
240         // Register required Events for the protestors to disappear.
241         @SuppressWarnings("unchecked")
242         var st = (IEventListener<EventGameStageChanged>) this.getRoom("Street");
243         this.eventSystem.addListener(EventGameStageChanged.class, st);
244
245         // Play BGM when player enters room.
246         this.eventSystem.addListener(EventEntityEnteredRoom.class,
```

```
247             event -> {
248                 Entity entity = event.getEntity();
249                 if (entity == this.getPlayer()) {
250                     Room room = entity.getRoom();
251                     MusicType type = null;
252                     if (room instanceof RoomCoastline) {
253                         type = MusicType.Bay1;
254                     } else if (room instanceof RoomMainlandCoastline) {
255                         type = MusicType.Bay2;
256                     } else if (room instanceof RoomForest) {
257                         type = MusicType.Nature;
258                     } else if (room instanceof RoomCityCentre) {
259                         type = MusicType.City1;
260                     } else if (room instanceof RoomStreet) {
261                         type = MusicType.City2;
262                     }
263 
264                     if (type != null) {
265                         this.getEventSystem()
266                             .emit(new EventMusic(type, true));
267                     }
268                 }
269             });
270 
271         // Stop prevous BGM when player leaves room.
272         this.eventSystem.addListener(EventEntityLeftRoom.class,
273             event -> {
274                 if (event.getEntity() == this.getPlayer()) {
275                     Room room = event.getRoom();
276                     MusicType type = null;
277                     if (room instanceof RoomCoastline) {
278                         type = MusicType.Bay1;
279                     } else if (room instanceof RoomMainlandCoastline) {
280                         type = MusicType.Bay2;
281                     } else if (room instanceof RoomForest) {
282                         type = MusicType.Nature;
283                     } else if (room instanceof RoomCityCentre) {
284                         type = MusicType.City1;
285                     } else if (room instanceof RoomStreet) {
286                         type = MusicType.City2;
287                     }
```

```
288
289                     if (type != null) {
290                         this.eventSystem
291                             .emit(new EventMusic(type, false));
292                     }
293                 }
294             });
295
296         // Adaptive BGM depending on the part of the story.
297         this.eventSystem.addListener(EventGameStageChanged.class,
298             event -> {
299                 switch (event.getStage()) {
300                     case Stealth: {
301                         this.eventSystem
302                             .emit(new EventMusic(MusicType.BgmExplore, false));
303                         this.eventSystem
304                             .emit(new EventMusic(MusicType.BgmMission, true));
305                         break;
306                     }
307                     case End: {
308                         this.eventSystem
309                             .emit(new EventMusic(MusicType.BgmMission, false));
310                         this.eventSystem
311                             .emit(new EventMusic(MusicType.BgmConclusion, true));
312                         break;
313                     }
314                     default: break;
315                 }
316             });
317     }
318
319     @Override
320     public void spawnPlayer() {
321         this.player.setLocation(this.rooms.get("Apartments: Home"));
322     }
323 }
```

```java
package uk.insrt.coursework.zuul.content.campaign.commands;

import java.awt.Image;
import java.io.InputStream;
import java.util.regex.Pattern;

import javax.imageio.ImageIO;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.events.IEventListener;
import uk.insrt.coursework.zuul.io.IOSystem;
import uk.insrt.coursework.zuul.ui.EventDraw;
import uk.insrt.coursework.zuul.world.World;

/**
 * Command available for the terminal emulator which displays a graphical map.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandMap extends Command implements IEventListener<EventDraw> {
    private boolean visible;
    private Image image;

    public CommandMap() {
        super("map", "<commands.map.usage>",
            new Pattern[] {
                Pattern.compile("^(?:m(?:ap)*)(?!\\w)")
                // m, map
            });

        this.visible = false;

        try {
            InputStream stream = this.getClass().getResourceAsStream("/map/base.png");
            this.image = ImageIO.read(stream);
        } catch (Exception e) {}
    }

```

```java
    @Override
    public boolean run(World world, Arguments arguments) {
        CampaignWorld campaignWorld = (CampaignWorld) world;
        IOSystem io = world.getIO();
        io.print("<commands.map.discovered.1> " + campaignWorld.percentVisited()
            + "% <commands.map.discovered.2>!" + "\n".repeat(24) + "<commands.map.close>");

        // We make the map visible and block on user input,
        // Once the user interacts, the map is hidden again.
        this.visible = true;
        io.readLine();
        this.visible = false;
        return false;
    }

    @Override
    public void onEvent(EventDraw event) {
        if (!this.visible) return;

        // We are drawing the map from [0,1] to [80,24].
        float fw = event.getCharWidth();
        float fh = event.getCharHeight();

        var g = event.getGraphics();
        g.drawImage(
            this.image,
            Math.round(event.getOriginX()),
            Math.round(event.getOriginY() + fh),
            Math.round(fw * 80),
            Math.round(fh * 23),
            null
        );
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.commands;

import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.commands.Arguments;
import uk.insrt.coursework.zuul.commands.Command;
import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Stage;
import uk.insrt.coursework.zuul.world.World;

/**
 * This command is unlocked after the player completes the final mission.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class CommandWin extends Command {
    public CommandWin() {
        super("win", "<commands.win.usage>",
            new Pattern[] {
                Pattern.compile("^win(?!\\w)"),
                // win
            });
    }

    @Override
    public boolean run(World world, Arguments args) {
        var io = world.getIO();
        var w = (CampaignWorld) world;
        var flags = w.getStoryFlags();
        if (flags.getStage() != Stage.End) return false;

        io.println("<commands.win.conclusion>\n<commands.win.stats>\n"
            + "<commands.win.total_ticks>" + flags.getTicks() + "\n"
            + "<commands.win.total_time>" + flags.prettyPrintTimeElapsed() + "\n"
            + "<commands.win.sidequests_complete>"
            + flags.getCompletedQuests() + " / " + flags.getTotalQuests()
            + "\n\n<commands.win.press_enter_key>");

        io.readLine();
        return true;
```

```
42      }
43
44      @Override
45      public boolean isVisible() {
46          return false;
47      }
48  }
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.EntityObject;
import uk.insrt.coursework.zuul.entities.actions.IUseable;
import uk.insrt.coursework.zuul.events.world.EventTick;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * Bed entity which lets the player tick the World forwards.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityBed extends EntityObject implements IUseable {
    public EntityBed(World world, Location location) {
        super(world, location, 80, new String[] { "bed" }, "<entities.bed.description>");
    }

    public void use(Entity target) {
        // We emit EventTick an arbitrary amount of times to
        // in-effect push the time forwards. This will trigger
        // all random events which listen to this event.
        for (int i=0;i<20;i++) {
            world.emit(new EventTick());
        }

        this.world.getIO().println("<entities.bed.use>");
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Stage;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.Inventory;
import uk.insrt.coursework.zuul.entities.actions.IGiveable;
import uk.insrt.coursework.zuul.entities.actions.IUseable;
import uk.insrt.coursework.zuul.io.IOSystem;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.Room;
import uk.insrt.coursework.zuul.world.World;

/**
 * Boat entity which ferries the Player to an arbitrary location.
 * There is no restriction on location but they should be place as
 * appropriate and where it would be realistic to put a boat.
 *
 * Boats may not be operated by the player while they are carrying
 * anything so instead they must use the boat's storage.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityBoat extends Entity implements IUseable, IGiveable {
    private Room destination;

    public EntityBoat(World world, Location location, Room destination) {
        super(world, location, 200);
        this.destination = destination;
        this.inventory.setMaxWeight(100);
    }

    @Override
    public String[] getAliases() {
        return new String[] { "boat" };
    }

    @Override
    public String describe() {
        return "<entities.boat.description>";
```

```
42      }
43
44      @Override
45      public void use(Entity target) {
46          CampaignWorld world = (CampaignWorld) this.getWorld();
47          IOSystem io = world.getIO();
48          Inventory inventory = target.getInventory();
49
50          // Check if the player has the key to this boat.
51          boolean hasKey = false;
52          for (Entity item : inventory.getItems()) {
53              if (item instanceof EntityBoatKey) {
54                  hasKey = true;
55              }
56          }
57
58          if (!hasKey) {
59              if (world.getStoryFlags().getStage() == Stage.Exposition) {
60                  io.println("<entities.boat.locked>");
61              } else {
62                  io.println("<entities.boat.locked_for_sale>");
63              }
64
65              return;
66          }
67
68          // Check whether the player is carrying too much.
69          if (inventory.getWeight() > 1) {
70              io.println("<entities.boat.denied>");
71              return;
72          }
73
74          // If we're good to go, travel to the other side.
75          io.println("<entities.boat.travel>\n");
76          target.setLocation(this.destination);
77      }
78
79      @Override
80      public void give(Entity item) {
81          var io = this.getWorld().getIO();
82          if (item.setLocation(this.getInventory())) {
```

```
83              io.println("<entities.boat.give.1> " + item.getHighlightedName() + " <entities.boat.give.2>.");
84          } else {
85              io.println("<entities.boat.too_heavy>");
86          }
87      }
88  }
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.entities.EntityObject;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * Boat key object which is used to unlock and start the
 * speed boat present on the coast.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityBoatKey extends EntityObject {
    public EntityBoatKey(World world, Location location) {
        super(world, location, 0.01d,
            new String[] { "key" },
            "<entities.boat_key>");
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.actions.IPettable;
import uk.insrt.coursework.zuul.entities.actions.IUseable;
import uk.insrt.coursework.zuul.events.world.EventTick;
import uk.insrt.coursework.zuul.events.world.behaviours.SimpleWanderAI;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.Room;
import uk.insrt.coursework.zuul.world.World;

/**
 * Cat entity which wanders around the map.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityCat extends Entity implements IPettable, IUseable {
    public static final int WEIGHT = 5;

    public EntityCat(World world, Location startingLocation) {
        super(world, startingLocation, WEIGHT);
    }

    @Override
    public String[] getAliases() {
        return new String[] {
            "cat",
            "the cat"
        };
    }

    @Override
    public String describe() {
        return "<entities.cat.description>";
    }

    @Override
    public void pet() {
        this.getWorld().getIO().println("<entities.cat.pet>");
    }
```

```java

    @Override
    public void use(Entity target) {
        this.getWorld().getIO().println("<entities.cat.use>");
    }

    /**
     * Enable a simple wander behaviour for entity within given bounds.
     * @param rooms Path that this Entity should follow
     * @param chance The chance x that this entity moves, where x gives a 1/x fractional chance of moving.
     */
    public void useWanderAI(Room[] rooms, int chance) {
        this.getWorld()
            .getEventSystem()
            .addListener(EventTick.class, new SimpleWanderAI(this, rooms, chance));
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Stage;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomMedicalCentreOffice;
import uk.insrt.coursework.zuul.content.campaign.rooms.RoomMedicalCentreReception;
import uk.insrt.coursework.zuul.dialogue.Dialogue;
import uk.insrt.coursework.zuul.dialogue.DialogueOption;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.actions.IUseable;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.Room;
import uk.insrt.coursework.zuul.world.World;

/**
 * Comms entity which is used to communicate between
 * Marie and the player during the mission.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityComms extends EntityWithDialogue<String> implements IUseable {
    public EntityComms(World world, Location location) {
        super(world, location, 0.3d);
        this.setupDialogue();
    }

    @Override
    public void use(Entity target) {
        var w = (CampaignWorld) this.world;
        var io = w.getIO();

        if (w.getStoryFlags().getStage() == Stage.Stealth) {
            Room room = w.getPlayer().getRoom();
            if (room instanceof RoomMedicalCentreOffice) {
                this.dialogue.setNodeIfPresent("office");
            } else if (room instanceof RoomMedicalCentreReception) {
                if (((RoomMedicalCentreReception) room).getCouch().isSitting()) {
                    this.dialogue.setNodeIfPresent("in_position");
                } else {
                    this.dialogue.setNodeIfPresent("complaint");
```

```
                }
            } else {
                this.dialogue.setNodeIfPresent("orientation");
            }

            this.dialogue.run(io);
        } else {
            io.println("<entities.comms.off>");
        }
    }

    @Override
    public void setupDialogue(Dialogue<String> dialogue) {
        this.setupDialogueFromId(dialogue, "comms_marie");

        // Make the guards disappear at this point.
        dialogue.getPart("distraction")
            .addOption(new DialogueOption<>(
                "<marie.comms.distraction.option_1>",
                io -> {
                    ((RoomMedicalCentreReception) this.world.getRoom("Medical Centre: Reception"))
                        .getGuard()
                        .consume(false);

                    return "coast_is_clear";
                }
            ));
    }

    @Override
    public String[] getAliases() {
        return new String[] { "comms" };
    }

    @Override
    public String describe() {
        return "<entities.comms.description>";
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.EntityObject;
import uk.insrt.coursework.zuul.entities.actions.IUseable;
import uk.insrt.coursework.zuul.events.IEventListener;
import uk.insrt.coursework.zuul.events.world.EventEntityLeftRoom;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * Couch present in the Medical Centre reception area.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityCouch extends EntityObject implements IUseable, IEventListener<EventEntityLeftRoom> {
    private boolean isSitting;

    /**
     * Construct a new EntityCouch.
     * @param world World
     * @param location Location
     */
    public EntityCouch(World world, Location location) {
        super(world, location, Double.MAX_VALUE, "couch", "<entities.couch.description>");
    }

    @Override
    public void use(Entity target) {
        var io = this.getWorld().getIO();
        if (this.isSitting) {
            io.println("<entities.couch.sitting>");
        } else {
            io.println("<entities.couch.sit>");
            this.isSitting = true;
        }
    }

    @Override
    public void onEvent(EventEntityLeftRoom event) {
```

```
42          this.isSitting = false;
43      }
44
45      /**
46       * Whether the player is sitting on the couch.
47       * @return True if the player is sat down
48       */
49      public boolean isSitting() {
50          return this.isSitting;
51      }
52  }
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.EntityObject;
import uk.insrt.coursework.zuul.entities.actions.IUseable;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * Documents which the player needs to find and take.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityDocument extends EntityObject implements IUseable {
    private int count;

    /**
     * Construct a new EntityDocument
     * @param world World
     * @param location Location
     * @param count Document Id
     */
    public EntityDocument(World world, Location location, int count) {
        super(world, location, 10, "doc" + count, "<medical_centre_office.books." + count + ".title>");
        this.count = count;
    }

    @Override
    public void use(Entity target) {
        this.getWorld()
            .getIO()
            .println("<medical_centre_office.books." + count + ".contents>");
    }

    /**
     * Check whether these are the documents we want.
     * @return Whether we want this document
     */
    public boolean getIsValid() {
        switch (this.count) {
```

```
42              case 2:
43              case 4:
44              case 5: return true;
45          }
46
47          return false;
48      }
49  }
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import java.awt.Desktop;
import java.net.URI;

import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Stage;
import uk.insrt.coursework.zuul.dialogue.Dialogue;
import uk.insrt.coursework.zuul.dialogue.DialogueNode;
import uk.insrt.coursework.zuul.dialogue.DialogueOption;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.Inventory;
import uk.insrt.coursework.zuul.entities.actions.IUseable;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * This is the player's laptop which resides in their home.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityLaptop extends EntityWithDialogue<String> implements IUseable {
    /**
     * Construct a new EntityLaptop.
     * @param world World
     * @param location Location
     */
    public EntityLaptop(World world, Location location) {
        super(world, location, 2);
        this.setupDialogue();
    }

    @Override
    public void use(Entity target) {
        this.dialogue.run(this.getWorld().getIO());
    }

    @Override
    public void setupDialogue(Dialogue<String> dialogue) {
        this.setupDialogueFromId(dialogue, "entity_laptop");
```

```java

        // Add funny cat videos.
        this.dialogue.addPart("funny_cat_videos",
            new DialogueNode<String>("<entities.laptop.cat_videos.dialog>")
                .addOption(new DialogueOption<String>("<entities.laptop.cat_videos.option_q>",
                    io -> {
                        try {
                            Desktop.getDesktop().browse(new URI("https://youtu.be/k35aiQPgNI4"));
                        } catch (Exception e) { /* If we fail, just ignore this ever happen. */ }

                        return "home";
                    })));

        // Story handler for sending documents to Marie.
        this.dialogue.getPart("document")
            .addOption(new DialogueOption<String>("<entities.laptop.document.option_1>",
                io -> {
                    var w = (CampaignWorld) this.getWorld();
                    var flags = w.getStoryFlags();
                    if (flags.getStage() != Stage.Stealth) {
                        io.println("<marie.comms.no_access>");
                        return "document";
                    }

                    Inventory inv = w.getPlayer().getInventory();
                    int validPieces = 0;
                    for (Entity item : inv.getItems()) {
                        if (item instanceof EntityDocument) {
                            if (((EntityDocument) item).getIsValid()) {
                                validPieces++;
                            }
                        }
                    }

                    if (validPieces == 3) {
                        io.println("<marie.comms.received>");
                        flags.setStage(Stage.End);
                        return null;
                    }

                    io.println("<marie.comms.bad_documents>");
```

```
                    return "document";
                }));
    }

    @Override
    public String[] getAliases() {
        return new String[] { "laptop" };
    }

    @Override
    public String describe() {
        return "<entities.laptop.description>";
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Stage;
import uk.insrt.coursework.zuul.dialogue.Dialogue;
import uk.insrt.coursework.zuul.dialogue.DialogueOption;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.Inventory;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * Marie Itami
 * https://brand-new-animal.fandom.com/wiki/Marie_Itami
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityMarie extends EntityNpc {
    /**
     * Construct a new EntityMarie.
     * @param world World
     * @param location Location
     */
    public EntityMarie(World world, Location location) {
        super(world, location,
            "npc_marie",
            "<marie.description>",
            new String[] { "marie", "itami", "mink" });
    }

    @Override
    public void setupDialogue(Dialogue<String> dialogue) {
        super.setupDialogue(dialogue);
        var w = (CampaignWorld) this.world;

        // Progress story if player accepts mission.
        dialogue.getPart("confirm")
            .addOption(new DialogueOption<>("<marie.alley.confirm.option_1>",
                io -> {
                    w.getStoryFlags()
```

```
                        .setStage(Stage.Recon);

                    return "recon";
                }));
    }

    @Override
    public void talk() {
        if (this.dialogue.getCurrentNode().equals("waiting")) {
            var w = ((CampaignWorld) this.getWorld());
            Inventory inv = w.getPlayer().getInventory();
            for (Entity item : inv.getItems()) {
                if (item instanceof EntityComms) {
                    this.dialogue.setNodeIfPresent("mission_brief");
                    w.getStoryFlags().setStage(Stage.Stealth);
                }
            }
        }

        super.talk();
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.dialogue.Dialogue;
import uk.insrt.coursework.zuul.entities.actions.ITalkwith;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * NPC entity which provides dialog and can be talked with by the Player.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityNpc extends EntityWithDialogue<String> implements ITalkwith {
    private String description;
    private String alias[];
    private String id;

    public EntityNpc(World world, Location location, String id, String description, String alias[]) {
        super(world, location, 75, null);

        this.description = description;
        this.alias = alias;
        this.id = id;

        this.setupDialogue();
    }

    public void talk() {
        this.dialogue.run(this.getWorld().getIO());
    }

    @Override
    public String[] getAliases() {
        return this.alias;
    }

    @Override
    public String describe() {
        return this.description;
    }
```

```
42
43      @Override
44      public void setupDialogue(Dialogue<String> dialogue) {
45          this.setupDialogueFromId(dialogue, id);
46      }
47  }
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Quest;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.actions.IGiveable;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * The old man who is in the forest.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityOldMan extends EntityNpc implements IGiveable {
    /**
     * Construct a new EntityOldMan.
     * @param world World
     * @param location Location
     */
    public EntityOldMan(World world, Location location) {
        super(world, location, "npc_old_man",
            "<forest.old_man.description>",
            new String[] { "oldman", "man" });

        this.inventory.setMaxWeight(EntityCat.WEIGHT);
    }

    @Override
    public void give(Entity item) {
        var io = this.getWorld().getIO();
        if (this.inventory.isFull()) {
            io.println("<forest.old_man.full>");
            return;
        }

        if (item instanceof EntityCat) {
            item.setLocation(this.inventory);
            this.dialogue.setNodeIfPresent("praise");
            ((CampaignWorld) this.getWorld())
```

```
                    .getStoryFlags()
                    .completeSideQuest(Quest.Cat);

                io.println("<forest.old_man.accept>");
            } else {
                io.println("<forest.old_man.deny> " + item.getHighlightedName());
            }
        }
    }
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Stage;
import uk.insrt.coursework.zuul.dialogue.Dialogue;
import uk.insrt.coursework.zuul.dialogue.DialogueNode;
import uk.insrt.coursework.zuul.dialogue.DialogueOption;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.EntityObject;
import uk.insrt.coursework.zuul.entities.Inventory;
import uk.insrt.coursework.zuul.io.Ansi;
import uk.insrt.coursework.zuul.sound.EventSound;
import uk.insrt.coursework.zuul.sound.SoundType;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * Shop keeper which the player can buy items from in the town.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityShopkeeper extends EntityNpc {
    private HashMap<Stage, Entity[]> items;
    private HashMap<Entity, Integer> stock;
    private HashMap<Entity, Integer> price;
    private HashMap<Entity, IEntityFactory> entityFactory;

    /**
     * Construct a new EntityShopkeeper.
     * @param world World
     * @param location Location
     */
    public EntityShopkeeper(World world, Location location) {
        super(world, location,
            "npc_shopkeeper",
            "<shop.npc.description>",
```

```
42              new String[] { "shopkeeper", "shop", "keeper" });
43
44          this.items = new HashMap<>();
45          this.stock = new HashMap<>();
46          this.price = new HashMap<>();
47          this.entityFactory = new HashMap<>();
48
49          this.createItems(world);
50      }
51
52      /**
53       * Interface implemented by entity factories for producing entities.
54       */
55      private interface IEntityFactory {
56          /**
57           * Produce a new Entity of a certain type.
58           * @return New entity
59           */
60          public Entity produce();
61      }
62
63      /**
64       * Generate all the items and populate the data.
65       * @param world World to place items in
66       */
67      private void createItems(World world) {
68          EntityObject itemBoatKey = new EntityBoatKey(world, new Location());
69          this.stock.put(itemBoatKey, 1);
70          this.price.put(itemBoatKey, 39_260);
71          this.entityFactory.put(itemBoatKey, () -> new EntityBoatKey(world, new Location()));
72
73          EntityComms itemComms = new EntityComms(world, new Location());
74          this.stock.put(itemComms, 3);
75          this.price.put(itemComms, 3_100);
76          this.entityFactory.put(itemComms, () -> new EntityComms(world, new Location()));
77
78          EntityObject itemCat = new EntityObject(world, new Location(), 5, "", "<shop.npc.fake_item.cat>");
79          this.stock.put(itemCat, 0);
80          this.price.put(itemCat, 21_300);
81
82          this.items.put(Stage.Exposition, new Entity[] {  });
```

```java
        this.items.put(Stage.Recon, new Entity[] { itemBoatKey, itemComms });
        this.items.put(Stage.Stealth, new Entity[] { itemBoatKey, itemComms, itemCat });
        this.items.put(Stage.End, new Entity[] { itemBoatKey, itemCat });
    }

    /**
     * Index dialogue node, displays things that the player can buy.
     * Implicitly takes the context of the EntityShopkeeper.
     */
    private class IndexNode extends DialogueNode<String> {
        /**
         * Construct a new IndexNode.
         * We may ignore the description since we override {@code #getDescription}.
         */
        public IndexNode() {
            super(null);
        }

        @Override
        public String getDescription() {
            var w = (CampaignWorld) world;
            return "<shop.npc.greeting."
                + w.getStoryFlags().getStage().toString()
                + ">\n"
                + "<shop.npc.currently_have_amount_of_money> ¥ "
                + w.getStoryFlags().getBalance()
                + "\n";
        }

        @Override
        protected List<DialogueOption<String>> getOptions() {
            ArrayList<DialogueOption<String>> options = new ArrayList<>();

            var w = (CampaignWorld) world;
            var flags = w.getStoryFlags();
            var player = w.getPlayer();

            // Get all the items we can access at this story stage.
            Entity[] list = items.get(flags.getStage());
            for (Entity item : list) {
                int count = stock.get(item);
```

```
124                 int cost = price.get(item);
125                 IEntityFactory factory = entityFactory.get(item);
126
127                 // Add the option for this item.
128                 options.add(new DialogueOption<String>(
129                     item.describe()
130                         + " ["
131                         + Ansi.Yellow
132                         + item.getWeight()
133                         + " kg"
134                         + Ansi.Reset
135                         + "] (¥ "
136                         + Ansi.Green
137                         + cost
138                         + Ansi.Reset
139                         + ") - "
140                         + (count == 0 ?
141                             "<shop.npc.out_of_stock>!"
142                           : count + " <shop.npc.x_left>"),
143                     io -> {
144                         // Check that this item is in stock.
145                         if (count == 0) {
146                             io.println("\n\n<shop.npc.item_out_of_stock.1> "
147                                 + Ansi.Red
148                                 + "<shop.npc.out_of_stock>"
149                                 + Ansi.Reset
150                                 + ", <shop.npc.item_out_of_stock.2>!");
151                         } else {
152                             // Make sure the player can hold this item without
153                             // going over their inventory weight limit.
154                             Inventory inv = player.getInventory();
155                             if (inv.getWeight() + item.getWeight() > inv.getMaxWeight()) {
156                                 io.println("\n\n<shop.npc.too_heavy>");
157                             } else {
158                                 // Try to deduct money from the player's money.
159                                 if (flags.deductFromBalance(cost)) {
160                                     io.println("\n\n<shop.npc.bought.1> "
161                                         + item.getHighlightedName()
162                                         + " <shop.npc.bought.2>!");
163
164                                     stock.put(item, count - 1);
```

```
165
166                                     Entity entity = factory.produce();
167                                     entity.setLocation(inv);
168
169                                     world.emit(new EventSound(SoundType.MoneyBag));
170                                 } else {
171                                     io.println("\n\n<shop.npc.not_enough> "
172                                         + item.getHighlightedName() + "!");
173                                 }
174                             }
175                         }
176
177                         return "index";
178                     }
179                 ));
180             }
181
182             options.add(new DialogueOption<String>("<shop.npc.leave>", "index").mustExit());
183             return options;
184         }
185     }
186
187     @Override
188     public void setupDialogue(Dialogue<String> dialogue) {
189         dialogue.addPart("index", new IndexNode());
190         dialogue.setNodeIfPresent("index");
191     }
192 }
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.dialogue.Dialogue;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.actions.IUseable;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * TV entity present in the player's room which they interact
 * with at the start of the game to learn more about the world.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityTV extends EntityWithDialogue<String> implements IUseable {
    public EntityTV(World world, Location location) {
        super(world, location, 40);
        this.setupDialogue();
    }

    @Override
    public void use(Entity target) {
        this.dialogue.run(this.getWorld().getIO());
    }

    @Override
    public void setupDialogue(Dialogue<String> dialogue) {
        this.setupDialogueFromId(dialogue, "home_tv");
    }

    @Override
    public String[] getAliases() {
        return new String[] { "tv", "television" };
    }

    @Override
    public String describe() {
        return "<home.tv.description>";
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.entities;

import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.dialogue.Dialogue;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * Abstract implementation of an entity which provides some sort of dialogue.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public abstract class EntityWithDialogue<T> extends Entity {
    protected Dialogue<T> dialogue;

    /**
     * Construct a new EntityWithDialogue with a starting dialogue node
     * @param world Current World object
     * @param location Initial Location of this Entity
     * @param weight The weight (in kg) of this Entity
     * @param startNode The starting dialogue node
     */
    public EntityWithDialogue(World world, Location location, double weight, T startNode) {
        super(world, location, weight);

        Dialogue<T> dialogue = new Dialogue<T>(startNode);
        this.dialogue = dialogue;
    }

    /**
     * Construct a new EntityWithDialogue without a starting dialogue node
     * @param world Current World object
     * @param location Initial Location of this Entity
     * @param weight The weight (in kg) of this Entity
     */
    public EntityWithDialogue(World world, Location location, double weight) {
        this(world, location, weight, null);
    }

```

```java
    /**
     * Configure this Entity's dialogue,
     * create nodes and options to add to this Entity.
     * @param dialogue Entity Dialogue
     */
    public abstract void setupDialogue(Dialogue<T> dialogue);

    /**
     * Configure dialogue.
     */
    public void setupDialogue() {
        this.setupDialogue(this.dialogue);
    }

    /**
     * Use the CampaignWorld's DialogueLoader to populate this Entity's Dialogue
     * @param dialogue Entity Dialogue
     * @param id Target dialogue ID in file
     */
    public void setupDialogueFromId(Dialogue<String> dialogue, String id) {
        var world = (CampaignWorld) this.getWorld();
        world.getDialogueLoader().populate(dialogue, id);
    }

    /**
     * Set the current dialogue node if the given node is present.
     * @param node Target node
     */
    public void setDialogueNodeIfPresent(Object node) {
        try {
            @SuppressWarnings("unchecked")
            T n = (T) node;

            this.dialogue.setNodeIfPresent(n);
        } catch (ClassCastException ex) {
            // Ignore the error since if we can't cast it
            // to whatever type this is, then obviously this
            // node is not present within this Dialogue.
        }
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.events;

import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Stage;
import uk.insrt.coursework.zuul.events.Event;

/**
 * Event fired when the story stage (chapter) changes.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EventGameStageChanged extends Event {
    private Stage stage;

    /**
     * Construct a new GameStageChanged event.
     * @param stage New stage
     */
    public EventGameStageChanged(Stage stage) {
        this.stage = stage;
    }

    /**
     * Get the new game stage.
     * @return Stage
     */
    public Stage getStage() {
        return this.stage;
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.world.Room;
import uk.insrt.coursework.zuul.world.World;

/**
 * Class which overrides getWorld to instead provide the CampaignWorld.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public abstract class CampaignRoom extends Room {
    /**
     * Construct a new CampaignRoom.
     * @param world World
     * @param name Internal name used to refer to this Room
     */
    public CampaignRoom(World world, String name) {
        super(world, name);
    }

    @Override
    public CampaignWorld getWorld() {
        return (CampaignWorld) super.getWorld();
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityBed;
import uk.insrt.coursework.zuul.content.campaign.entities.EntityLaptop;
import uk.insrt.coursework.zuul.content.campaign.entities.EntityTV;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * The player's home in the apartments complex.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomApartmentsHome extends CampaignRoom {
    public RoomApartmentsHome(World world) {
        super(world, "Apartments: Home");
    }

    @Override
    public String describe() {
        var world = this.getWorld();
        if (!world.hasVisited(this)) {
            return "<home.first_load>";
        }

        return "<home.enter>";
    }

    @Override
    protected void setupDirections() {
        this.setAdjacent(Direction.DOWN, this.getWorld().getRoom("Apartments: Reception"));
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();

        world.spawnEntity("tv", new EntityTV(world, this.toLocation()));
        world.spawnEntity("bed", new EntityBed(world, this.toLocation()));
        world.spawnEntity("laptop", new EntityLaptop(world, this.toLocation()));
```

```
42        }
43    }
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityNpc;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * The reception of the apartments complex.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomApartmentsReception extends CampaignRoom {
    public RoomApartmentsReception(World world) {
        super(world, "Apartments: Reception");
    }

    @Override
    public String describe() {
        return "<apartments.enter>";
    }

    @Override
    protected void setupDirections() {
        World world = this.getWorld();
        this.setAdjacent(Direction.NORTH, world.getRoom("Street"));
        this.setAdjacent(Direction.EAST, world.getRoom("City Centre"));
        this.setAdjacent(Direction.UP, world.getRoom("Apartments: Home"));
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();
        world.spawnEntity("receptionist",
            new EntityNpc(
                world,
                this.toLocation(),
                "npc_receptionist",
                "<apartments.receptionist.description>",
                new String[] { "receptionist" }
            ));
```

```
42        }
43    }
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityMarie;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * The back alley in the North East side of the map.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomBackAlley extends CampaignRoom {
    public RoomBackAlley(World world) {
        super(world, "Back Alley");
    }

    @Override
    public String describe() {
        var world = this.getWorld();
        if (!world.hasVisited(this)) {
            return "<back_alley.first_load>";
        }

        return "<back_alley.enter>";
    }

    @Override
    protected void setupDirections() {
        this.setAdjacent(Direction.SOUTH, this.getWorld().getRoom("City Centre"));
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();
        world.spawnEntity("npc_marie", new EntityMarie(world, this.toLocation()));
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityCat;
import uk.insrt.coursework.zuul.content.campaign.entities.EntityNpc;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.Room;
import uk.insrt.coursework.zuul.world.World;

/**
 * The city centre connecting most major locations.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomCityCentre extends CampaignRoom {
    public RoomCityCentre(World world) {
        super(world, "City Centre");
    }

    @Override
    public String describe() {
        var world = this.getWorld();
        if (!world.hasVisited(this)) {
            return "<city_centre.first_load>";
        }

        return "<city_centre.enter>";
    }

    @Override
    protected void setupDirections() {
        World world = this.getWorld();
        this.setAdjacent(Direction.NORTH, world.getRoom("Back Alley"));
        this.setAdjacent(Direction.NORTH_WEST, world.getRoom("Street"));
        this.setAdjacent(Direction.WEST, world.getRoom("Apartments: Reception"));
        this.setAdjacent(Direction.SOUTH, world.getRoom("Coastline"));
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();
```

```java

        EntityCat cat = new EntityCat(world, this.toLocation());
        world.spawnEntity("cat", cat);
        cat.useWanderAI(
            new Room[] {
                world.getRoom("City Centre"),
                world.getRoom("Street"),
                world.getRoom("Shop"),
                world.getRoom("Street"),
                world.getRoom("City Centre"),
                world.getRoom("Back Alley"),
                world.getRoom("City Centre")
            },
            8
        );

        world.spawnEntity("city_npc",
            new EntityNpc(
                world,
                this.toLocation(),
                "npc_city_centre",
                "<city_centre.npc.description>",
                new String[] { "stranger", "person", "people" }
            ));
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityBoat;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * The coast which connects the main city to the mainland.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomCoastline extends CampaignRoom {
    public RoomCoastline(World world) {
        super(world, "Coastline");
    }

    @Override
    public String describe() {
        return "<coastline.enter>";
    }

    @Override
    protected void setupDirections() {
        this.setAdjacent(Direction.NORTH, this.getWorld().getRoom("City Centre"));
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();
        world.spawnEntity("boat1",
            new EntityBoat(world, this.toLocation(),
                world.getRoom("Mainland: Coastline")));
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityOldMan;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * A forest on the mainland side.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomForest extends CampaignRoom {
    public RoomForest(World world) {
        super(world, "Forest");
    }

    @Override
    public String describe() {
        return "<forest.enter>";
    }

    @Override
    protected void setupDirections() {
        World world = this.getWorld();
        this.setAdjacent(Direction.NORTH, world.getRoom("Mainland: Coastline"));
        this.setAdjacent(Direction.EAST, world.getRoom("Worm Hole"));
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();
        world.spawnEntity("npc_old_man", new EntityOldMan(world, this.toLocation()));
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityBoat;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * The coast which connects the mainland to the main city.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomMainlandCoastline extends CampaignRoom {
    public RoomMainlandCoastline(World world) {
        super(world, "Mainland: Coastline");
    }

    @Override
    public String describe() {
        return "<mainland_coastline.enter>";
    }

    @Override
    protected void setupDirections() {
        this.setAdjacent(Direction.SOUTH, this.getWorld().getRoom("Forest"));
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();
        world.spawnEntity("boat2",
            new EntityBoat(world, this.toLocation(),
                world.getRoom("Coastline")));
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityDocument;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * Private, usually inaccessible room within the Medical Centre complex.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomMedicalCentreOffice extends CampaignRoom {
    public RoomMedicalCentreOffice(World world) {
        super(world, "Medical Centre: Office");
    }

    @Override
    public String describe() {
        return "<medical_centre_office.enter>";
    }

    @Override
    protected void setupDirections() {
        this.setAdjacent(Direction.UP, this.getWorld().getRoom("Medical Centre: Reception"));
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();
        for (int i=1;i<=6;i++) {
            world.spawnEntity("doc" + i, new EntityDocument(world, this.toLocation(), i));
        }
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityCouch;
import uk.insrt.coursework.zuul.content.campaign.entities.EntityNpc;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.events.world.EventEntityLeftRoom;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * Reception of the Medical Centre complex.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomMedicalCentreReception extends CampaignRoom {
    private Entity guardEntity;
    private EntityCouch couchEntity;

    public RoomMedicalCentreReception(World world) {
        super(world, "Medical Centre: Reception");
    }

    @Override
    public String describe() {
        return "<medical_centre.enter>";
    }

    @Override
    protected void setupDirections() {
        World world = this.getWorld();
        this.setAdjacent(Direction.EAST, world.getRoom("Street"));
        this.setAdjacent(Direction.DOWN, world.getRoom("Medical Centre: Office"));
    }

    @Override
    public boolean canLeave(Direction direction) {
        if (direction == Direction.DOWN) {
            if (this.guardEntity.getRoom() == this) {
                this.getWorld()
                    .getIO()
```

```
                    .println("<medical_centre.guard.blocking>");

                return false;
            }
        }

        return true;
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();

        this.guardEntity = new EntityNpc(
            world,
            this.toLocation(),
            "npc_security_guard",
            "<medical_centre.guard.description>",
            new String[] { "guard", "security" }
        );
        world.spawnEntity("npc_guard", this.guardEntity);

        this.couchEntity = new EntityCouch(world, this.toLocation());
        world.spawnEntity("couch", this.couchEntity);
        world.getEventSystem().addListener(EventEntityLeftRoom.class, this.couchEntity);
    }

    public EntityCouch getCouch() {
        return this.couchEntity;
    }

    public Entity getGuard() {
        return this.guardEntity;
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.entities.EntityShopkeeper;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * A shop within the city, the only one the player can interact with.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomShop extends CampaignRoom {
    public RoomShop(World world) {
        super(world, "Shop");
    }

    @Override
    public String describe() {
        return "<shop.enter>";
    }

    @Override
    protected void setupDirections() {
        this.setAdjacent(Direction.SOUTH, this.getWorld().getRoom("Street"));
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();
        world.spawnEntity("npc_shopkeeper",
            new EntityShopkeeper(world, this.toLocation()));
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Stage;
import uk.insrt.coursework.zuul.content.campaign.entities.EntityNpc;
import uk.insrt.coursework.zuul.content.campaign.events.EventGameStageChanged;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.events.IEventListener;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.World;

/**
 * One of the major connecting points between locations in the city.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomStreet extends CampaignRoom implements IEventListener<EventGameStageChanged> {
    private Entity protestorsEntity;

    public RoomStreet(World world) {
        super(world, "Street");
    }

    @Override
    public String describe() {
        var world = this.getWorld();
        if (!world.hasVisited(this)) {
            return "<street.first_load>";
        }

        return "<street.enter>";
    }

    @Override
    protected void setupDirections() {
        World world = this.getWorld();
        this.setAdjacent(Direction.SOUTH, world.getRoom("Apartments: Reception"));
        this.setAdjacent(Direction.EAST, world.getRoom("City Centre"));
        this.setAdjacent(Direction.NORTH, world.getRoom("Shop"));
        this.setAdjacent(Direction.WEST, world.getRoom("Medical Centre: Reception"));
    }
```

```java

    @Override
    public boolean canLeave(Direction direction) {
        if (direction == Direction.WEST) {
            if (this.protestorsEntity.getRoom() == this) {
                this.getWorld()
                    .getIO()
                    .println("<street.protestors.blocking>");

                return false;
            }
        }

        return true;
    }

    @Override
    public void spawnEntities() {
        World world = this.getWorld();
        this.protestorsEntity = new EntityNpc(
            world,
            this.toLocation(),
            "npc_protestors",
            "<street.protestors.description>",
            new String[] { "protestors", "protestor" }
        );
        world.spawnEntity("npc_protestors", this.protestorsEntity);
    }

    @Override
    public void onEvent(EventGameStageChanged event) {
        // Remove the protestors if we are in Stealth chapter.
        if (event.getStage() == Stage.Stealth) {
            this.protestorsEntity.consume(false);
        } else {
            this.protestorsEntity.setLocation(this);
        }
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign.rooms;

import java.util.Random;

import uk.insrt.coursework.zuul.content.campaign.StoryFlags.Quest;
import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.EntityPlayer;
import uk.insrt.coursework.zuul.events.IEventListener;
import uk.insrt.coursework.zuul.events.world.EventEntityEnteredRoom;
import uk.insrt.coursework.zuul.sound.EventSound;
import uk.insrt.coursework.zuul.sound.SoundType;
import uk.insrt.coursework.zuul.world.Room;
import uk.insrt.coursework.zuul.world.World;

/**
 * Teleporter room implemented as required by the challenge tasks.
 * Any Entity that walks into the worm hole is transported into a random public location.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class RoomWormHole extends CampaignRoom implements IEventListener<EventEntityEnteredRoom> {
    public RoomWormHole(World world) {
        super(world, "Worm Hole");
    }

    @Override
    public String describe() {
        return "";
    }

    @Override
    protected void setupDirections() {}

    @Override
    public void onEvent(EventEntityEnteredRoom event) {
        Entity entity = event.getEntity();
        Room room = entity.getRoom();
        if (room != this) return;
        event.stopPropagation();

```

```java
        // This is a restricted set of locations as to not break
        // the game's plot, say if we were transported to the medical
        // centre complex office when we're not meant to go there yet.
        final Random random = new Random();
        final String[] locations = {
            "City Centre",
            "Coastline",
            "Mainland: Coastline",
            "Forest",
            "Street",
            "Back Alley"
        };

        var world = this.getWorld();
        var io = world.getIO();
        io.println("\n<worm_hole.enter>");

        try {
            Thread.sleep(1000);

            final int WIDTH = 79;

            for (int i=0;i<5;i++) {
                io.println("*".repeat(i*3) + "\\"
                    + " ".repeat(WIDTH - i * 6 - 2) + "/" + "*".repeat(i*3));

                Thread.sleep(60);
            }

            // Play worm hole sound while we are falling through time and space.
            world.emit(new EventSound(SoundType.WormHole));
            world.getStoryFlags().completeSideQuest(Quest.WormHole);

            for (int i=0;i<25*16;i++) {
                var out = "";
                for (int j=0;j<WIDTH;j++) {
                    out += random.nextInt(8) == 0 ? "*" : " ";
                }

                io.println(out);
                Thread.sleep(40);
```

```
            }

            for (int i=5;i>0;i--) {
                io.println("*".repeat(i*3) + "/"
                    + " ".repeat(WIDTH - i * 6 - 2) + "\\" + "*".repeat(i*3));

                Thread.sleep(60);
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
            io.println("There was a disruption when travelling.");
        }

        io.print("\n");

        // Pick a random location and put the entering entity in it.
        String location = locations[random.nextInt(locations.length)];
        Room target = this.getWorld().getRoom(location);
        entity.setLocation(target);

        // If it was the player, clear their walk history.
        if (entity instanceof EntityPlayer) {
            ((EntityPlayer) entity).clearHistory();
        }
    }
}
```

```java
package uk.insrt.coursework.zuul.content.campaign;

import java.util.HashSet;

import uk.insrt.coursework.zuul.content.campaign.events.EventGameStageChanged;
import uk.insrt.coursework.zuul.events.EventSystem;
import uk.insrt.coursework.zuul.events.world.EventTick;

/**
 * Class which controls story progression within the Campaign World.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class StoryFlags {
    /**
     * The current story chapter.
     */
    public enum Stage {
        Exposition, // Ch 1.
        Recon, // Ch 2.
        Stealth, // Ch 3.
        End, // Current Ending

        Twist, // Ch 4. Skipped
        Conclusion, // Ch 5. Skipped
    }

    /**
     * Side-quests available in the game.
     */
    public enum Quest {
        Cat,
        WormHole
    }

    private EventSystem eventSystem;
    private HashSet<Quest> quests;

    private long startTime;
    private int balance;
```

```java
    private Stage stage;
    private int ticks;

    /**
     * Construct a new instance of StoryFlags
     * @param eventSystem World event system
     */
    public StoryFlags(EventSystem eventSystem) {
        this.eventSystem = eventSystem;
        this.stage = Stage.Exposition;
        this.quests = new HashSet<>();

        this.startTime = System.currentTimeMillis();
        this.balance = 100_000;
        this.ticks = 0;

        this.eventSystem.addListener(EventTick.class, e -> this.ticks++);
    }

    /**
     * Get the current stage (chapter) of the story.
     * @return Current stage
     */
    public Stage getStage() {
        return this.stage;
    }

    /**
     * Set the current stage (chapter) of the story.
     * @param stage New stage
     */
    public void setStage(Stage stage) {
        this.stage = stage;
        this.eventSystem.emit(new EventGameStageChanged(stage));
    }

    /**
     * Get the player's balance
     * @return Player's balance
     */
    public int getBalance() {
```

```
83          return this.balance;
84      }
85
86      /**
87       * Set player's new balance.
88       * @param balance New balance
89       */
90      public void setBalance(int balance) {
91          this.balance = balance;
92      }
93
94      /**
95       * Deduct money from the player's balance.
96       * @param value Amount to deduct
97       * @return Whether we could deduct the balance without going below zero
98       */
99      public boolean deductFromBalance(int value) {
100         if (value > this.balance) {
101             return false;
102         }
103
104         this.balance -= value;
105         return true;
106     }
107
108     /**
109      * Get ticks since start of the World.
110      * @return Number of ticks since start
111      */
112     public int getTicks() {
113         return this.ticks;
114     }
115
116     /**
117      * Mark a side-quest as complete
118      */
119     public void completeSideQuest(Quest quest) {
120         this.quests.add(quest);
121     }
122
123     /**
```

```java
     * Get completed side-quests.
     * @return Number of completed side-quests
     */
    public int getCompletedQuests() {
        return this.quests.size();
    }

    /**
     * Get total number of side-quests.
     * @return Total number of side-quests
     */
    public int getTotalQuests() {
        return Quest.values().length;
    }

    /**
     * Get time elapsed since the start of the game.
     * @return Time elapsed
     */
    public long timeElapsed() {
        return System.currentTimeMillis() - this.startTime;
    }

    /**
     * Take the time elapsed and pretty print it.
     * @return Pretty printed time elapsed.
     */
    public String prettyPrintTimeElapsed() {
        long time = this.timeElapsed() / 1000;
        return (time / 60) + " <commands.win.minutes> "
            + (time % 60) + " <commands.win.seconds>";
    }
}
```

```java
package uk.insrt.coursework.zuul.dialogue;

import java.util.HashMap;

import uk.insrt.coursework.zuul.io.IOSystem;

/**
 * Simple dialogue engine which navigates between {@link DialogueNode}(s).
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Dialogue<T> {
    private HashMap<T, DialogueNode<T>> parts;
    private T currentNode;

    /**
     * Contruct a new Dialogue engine.
     */
    public Dialogue() {
        this.parts = new HashMap<>();
    }

    /**
     * Construct a new Dialogue engine and initialise us at a starting node.
     * @param start Starting node
     */
    public Dialogue(T start) {
        this();
        this.currentNode = start;
    }

    /**
     * Get the current node
     */
    public T getCurrentNode() {
        return this.currentNode;
    }

    /**
     * Set the current node
```

```java
     * @param node New node
     */
    public void getCurrentNode(T node) {
        this.currentNode = node;
    }

    /**
     * Change the current node to a different one if it exists
     * @param node New node
     */
    public void setNodeIfPresent(T node) {
        if (this.parts.containsKey(node)) {
            this.currentNode = node;
        }
    }

    /**
     * Add a new part to the dialogue
     * @param part What this node is identified by
     * @param node The new node
     */
    public void addPart(T part, DialogueNode<T> node) {
        this.parts.put(part, node);
    }

    /**
     * Get an existing part from the dialogue
     * @param part What the node is identified by
     * @return The node if it exists, otherwise null
     */
    public DialogueNode<T> getPart(T part) {
        return this.parts.get(part);
    }

    /**
     * Run the Dialogue engine until one of the options exits us out
     * @param io Provided IO system
     */
    public void run(IOSystem io) {
        var part = this.parts.get(this.currentNode);
        io.println("\n" + part.getDescription());
```

```
83          DialogueOption<T> option = part.pickOption(io);
84
85          T target = option.handle(io);
86          if (target == null) {
87              T newTarget = option.getTarget();
88              if (newTarget != null) {
89                  this.currentNode = newTarget;
90              }
91
92
93              return;
94          }
95
96          this.currentNode = target;
97          this.run(io);
98      }
99  }
```

```java
package uk.insrt.coursework.zuul.dialogue;

import java.io.IOException;
import java.io.InputStream;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Map.Entry;

import com.moandjiezana.toml.Toml;

/**
 * This is a helper class for loading and populating {@link Dialogue}.
 * This DialogueLoader assumes that i18n is being used in the dialogue data.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class DialogueLoader {
    private Map<String, Object> data;

    /**
     * Construct a new DialogueLoader
     */
    public DialogueLoader() {
        this.data = new HashMap<>();
    }

    /**
     * Load all necessary data for populating Dialogue.
     * @param path Path to the dialogue resource file
     * @throws IOException if we can't read the dialogue file
     */
    public void load(String path) throws IOException {
        InputStream stream = DialogueLoader.class.getResourceAsStream(path);
        this.data = new Toml().read(stream).toMap();
    }

    /**
     * Populate a Dialogue system using a specific dialog definition represented by a given key.
     *
```

```java
     * This method is unchecked as we expect a valid data structure to have
     * been loaded from the resource, this should be verified by the developer.
     * @param dialogue Dialogue system
     * @param key Key to lookup
     */
    @SuppressWarnings("unchecked")
    public void populate(Dialogue<String> dialogue, String key) {
        Map<String, Object> nodes = (Map<String, Object>) this.data.get(key);

        // Process any special keys first before we continue.
        String prefix = "";
        for (String nodeKey : nodes.keySet()) {
            if (nodeKey.equals("_prefix")) {
                prefix = (String) nodes.get(nodeKey);
            } else if (nodeKey.equals("_start")) {
                dialogue.getCurrentNode((String) nodes.get(nodeKey));
            }
        }

        for (Entry<String, Object> node : nodes.entrySet()) {
            // Ignore any keys starting with _, as they are used above.
            String nodeKey = node.getKey();
            if (nodeKey.startsWith("_")) continue;

            // Read each node's values and find the description and options.
            Map<String, Object> values = (Map<String, Object>) node.getValue();

            // Description strings are assumed to be i18n paths.
            String description = "<" + prefix + (String) values.get("description") + ">";
            List<Map<String, Object>> options = (List<Map<String, Object>>) values.get("options");

            // Construct a new Dialogue Node with the given data.
            DialogueNode<String> dialogueNode = new DialogueNode<>(description);
            for (Map<String, Object> object : options) {
                String desc = "<" + prefix + (String) object.get("description") + ">";
                String to = (String) object.get("to");
                Boolean mustExit = (Boolean) object.get("mustExit");

                if (mustExit == null) {
                    dialogueNode.addOption(desc, to);
                } else {
```

```
                    dialogueNode.addOption(desc, to, mustExit);
                }
            }

            dialogue.addPart(nodeKey, dialogueNode);
        }
    }
}
```

```java
package uk.insrt.coursework.zuul.dialogue;

import java.util.ArrayList;
import java.util.List;

import uk.insrt.coursework.zuul.io.IOSystem;

/**
 * A node in a {@link Dialogue} system.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class DialogueNode<T> {
    private String description;
    private ArrayList<DialogueOption<T>> options;

    /**
     * Construct a new node.
     * @param description Description of this node
     */
    public DialogueNode(String description) {
        this.description = description;
        this.options = new ArrayList<>();
    }

    /**
     * Get this node's description
     * @return Description string
     */
    public String getDescription() {
        return this.description;
    }

    /**
     * Add a new option which branches off this node.
     * @param option Dialogue Option
     * @return This Dialogue Node so other method calls can be chained
     */
    public DialogueNode<T> addOption(DialogueOption<T> option) {
        this.options.add(option);
```

```java
        return this;
    }

    /**
     * Create a new option which branches off this node.
     * @param description Description of this option
     * @param stage The next stage of the dialogue this should jump to
     * @param mustExit Whether we must exit from the dialogue after selecting this option
     * @return This Dialogue Node so other method calls can be chained
     */
    public DialogueNode<T> addOption(String description, T stage, boolean mustExit) {
        var option = new DialogueOption<T>(description, stage);
        if (mustExit) option.mustExit();
        this.options.add(option);
        return this;
    }

    /**
     * Create a new option which branches off this node.
     * @param description Description of this option
     * @param stage The next stage of the dialogue this should jump to
     * @return This Dialogue Node so other method calls can be chained
     */
    public DialogueNode<T> addOption(String description, T stage) {
        return this.addOption(description, stage, false);
    }

    /**
     * Get the options available to this node.
     * @return List of options
     */
    protected List<DialogueOption<T>> getOptions() {
        return this.options;
    }

    /**
     * Ask the player to pick one of the valid options branching off this node.
     * @param io Provided IO system
     * @return The selected option
     */
    public DialogueOption<T> pickOption(IOSystem io) {
```

```
83          List<DialogueOption<T>> options = this.getOptions();
84          for (int i=0;i<options.size();i++) {
85              io.println((i + 1) + ". " + options.get(i).getDescription());
86          }
87
88          while (true) {
89              io.print("Choice: ");
90              String value = io.readLine();
91              try {
92                  int v = Integer.parseInt(value);
93                  if (v < 1 || v > options.size()) {
94                      io.println("Provide a valid option!");
95                      continue;
96                  }
97
98                  return options.get(v - 1);
99              } catch (Exception e) {
100                 io.println("Provide a valid number!");
101             }
102         }
103     }
104 }
```

```java
package uk.insrt.coursework.zuul.dialogue;

import uk.insrt.coursework.zuul.io.IOSystem;

/**
 * An option which branches off a {@link DialogueNode} into another node.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class DialogueOption<T> {
    private IDialogueHandler<T> handler;

    private String description;
    private boolean shouldExit;
    private T target;

    /**
     * Construct a new simple DialogueOption with a description and destination.
     * @param description Description of this option
     * @param target Target node to jump to
     */
    public DialogueOption(String description, T target) {
        this.target = target;
        this.description = description;
    }

    /**
     * Construct a complex DialogueOption with a description and select handler.
     * @param description Description of this option
     * @param handler Method called when this option is selected
     */
    public DialogueOption(String description, IDialogueHandler<T> handler) {
        this.handler = handler;
        this.description = description;
    }

    /**
     * Tell the Dialogue system to exit if this option is selected.
     * @return This dialogue option so method calls can be chained
     */
```

```java
    public DialogueOption<T> mustExit() {
        this.shouldExit = true;
        return this;
    }

    /**
     * Get the description of this option.
     * @return Description string
     */
    public String getDescription() {
        return this.description;
    }

    /**
     * Get the destination of this option.
     * @return Destination if it exists
     */
    public T getTarget() {
        return this.target;
    }

    /**
     * Handle the player selecting this dialogue option.
     * @param io Provided IO system
     * @return The new node or null if we should exit and stay put.
     */
    public T handle(IOSystem io) {
        if (this.handler != null) {
            return this.handler.onAction(io);
        } else if (!this.shouldExit) {
            return this.target;
        }

        return null;
    }
}
```

```java
package uk.insrt.coursework.zuul.dialogue;

import uk.insrt.coursework.zuul.io.IOSystem;

/**
 * Interface implemented to provide an onAction method.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public interface IDialogueHandler<T> {
    /**
     * Handle the selection of a dialogue option.
     * @param io Provided IO system
     * @return Destination node, may be null
     */
    public T onAction(IOSystem io);
}
```

```java
package uk.insrt.coursework.zuul.entities.actions;

import uk.insrt.coursework.zuul.entities.Entity;

/**
 * Interface implemented to provide the ability for
 * an Entity to have other Entities given to them.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public interface IGiveable {
    /**
     * Give this entity another entity.
     * @param item Item being given
     */
    public void give(Entity item);
}
```

```java
package uk.insrt.coursework.zuul.entities.actions;

/**
 * Interface implemented to provide the
 * ability for an Entity to be pet.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public interface IPettable {
    /**
     * Pet this entity.
     */
    public void pet();
}
```

```java
package uk.insrt.coursework.zuul.entities.actions;

/**
 * Interface implemented to provide the
 * ability for an Entity to talk with the player.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public interface ITalkwith {
    /**
     * Talk with this entity.
     */
    public void talk();
}
```

```java
package uk.insrt.coursework.zuul.entities.actions;

import uk.insrt.coursework.zuul.entities.Entity;

/**
 * Interface implemented to provide the
 * ability for an Entity to be used by the player.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public interface IUseable {
    /**
     * Use this entity.
     * @param target The Entity taking this entity.
     */
    public void use(Entity target);
}
```

```java
package uk.insrt.coursework.zuul.entities;

import uk.insrt.coursework.zuul.events.world.EventEntityEnteredRoom;
import uk.insrt.coursework.zuul.events.world.EventEntityLeftRoom;
import uk.insrt.coursework.zuul.io.Ansi;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.Room;
import uk.insrt.coursework.zuul.world.World;

/**
 * Representation of any Entity in the World.
 *
 * Any living beings, items, or otherwise things that
 * exist in the world are considered an Entity. Each
 * Entity also has an Inventory so things may be stored
 * inside of it.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public abstract class Entity {
    protected World world;
    protected Inventory inventory;

    private Location location;
    private double weight;

    /**
     * Construct a new Entity.
     * @param world Current World object
     * @param location Initial Location of this Entity
     * @param weight The weight (in kg) of this Entity
     */
    public Entity(World world, Location location, double weight) {
        this.world = world;
        this.location = location;
        this.inventory = new Inventory();
        this.weight = weight;
    }

    /**
```

```
42          * Construct a new Entity.
43          *
44          * Weight value is set to {@link Integer#MAX_VALUE}.
45          * @param world Current World object
46          * @param location Initial Location of this Entity
47          */
48         public Entity(World world, Location location) {
49             this(world, location, Integer.MAX_VALUE);
50         }
51 
52         /**
53          * Get this Entity's weight.
54          * @return Weight (in kg)
55          */
56         public double getWeight() {
57             return this.weight;
58         }
59 
60         /**
61          * Get the name of this Entity.
62          * Shorthand for {@link #getAliases()}[0].
63          * @return First matched alias
64          */
65         public String getName() {
66             return this.getAliases()[0];
67         }
68 
69         /**
70          * Get a highlighted representation of this Entity's name.
71          * @return Ansi hightlighted name
72          */
73         public String getHighlightedName() {
74             return Ansi.BackgroundWhite + Ansi.Black + this.getName() + Ansi.Reset;
75         }
76 
77         /**
78          * Get the Inventory that this Entity holds.
79          * @return Inventory
80          */
81         public Inventory getInventory() {
82             return this.inventory;
```

```
83      }
84
85      /**
86       * Get the World this Entity resides in.
87       * @return World
88       */
89      public World getWorld() {
90          return this.world;
91      }
92
93      /**
94       * Get the Room that this Entity is currently in.
95       * @return Room
96       */
97      public Room getRoom() {
98          return this.location.getRoom();
99      }
100
101     /**
102      * Get the Inventory that this Entity is currently in.
103      * @return Inventory
104      */
105     public Inventory getInventoryWithin() {
106         return this.location.getInventory();
107     }
108
109     /**
110      * Remove this Entity from any existing place.
111      * Provides a consistent way to clean up the Entity before placing it anywhere.
112      * @param suppressEvents Whether to suppress the "Entity Left Room" Event
113      * @return Whether this Entity was removed from an Inventory
114      */
115     public boolean consume(boolean suppressEvents) {
116         boolean consumed = false;
117         Inventory inventory = this.location.getInventory();
118         if (inventory != null) consumed = inventory.remove(this);
119
120         Room previousRoom = this.getRoom();
121         if (previousRoom != null && !suppressEvents) this.world.emit(new EventEntityLeftRoom(this, previousRoom));
122
123         this.location.clear();
```

```
124         return consumed;
125     }
126 
127     /**
128      * Move the Entity into a Room.
129      * @param room Destination Room
130      */
131     public void setLocation(Room room) {
132         boolean consumed = this.consume(false);
133         this.location.setLocation(room);
134         if (!consumed) this.world.emit(new EventEntityEnteredRoom(this));
135     }
136 
137     /**
138      * Move the Entity into an Inventory.
139      * @param inventory Destination Inventory
140      * @return Whether we successfully moved the entity into the inventory.
141      */
142     public boolean setLocation(Inventory inventory) {
143         if (inventory.add(this)) {
144             this.consume(true);
145             this.location.setLocation(inventory);
146             return true;
147         }
148 
149         return false;
150     }
151 
152     /**
153      * Link this Entity's inventory with an existing inventory.
154      * @param inventory Target inventory
155      */
156     public void entangleInventory(Inventory inventory) {
157         this.inventory = inventory;
158     }
159 
160     /**
161      * Get names that this Entity can be called by.
162      * @return String array of names for this Entity
163      */
164     public abstract String[] getAliases();
```

```
165
166     /**
167      * Get a description of this Entity.
168      * @return String describing the Entity
169      */
170     public abstract String describe();
171 }
```

```java
package uk.insrt.coursework.zuul.entities;

import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.World;

/**
 * Generic object class which avoids some boilerplate.
 * Use this for entities which are guaranteed to never change.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityObject extends Entity {
    private String description;
    private String[] aliases;

    /**
     * Construct a new EntityObject
     * @param world Current World object
     * @param location Initial Location of this Entity
     * @param weight The weight (in kg) of this Entity
     * @param aliases Aliases which this object can be referred to by
     * @param description A description of this object
     */
    public EntityObject(World world, Location location, double weight, String[] aliases, String description) {
        super(world, location, weight);
        this.description = description;
        this.aliases = aliases;
    }

    /**
     * Construct a new EntityObject
     * @param world Current World object
     * @param location Initial Location of this Entity
     * @param weight The weight (in kg) of this Entity
     * @param name Name of this object
     * @param description A description of this object
     */
    public EntityObject(World world, Location location, double weight, String alias, String description) {
        this(world, location, weight, new String[] { alias }, description);
    }
```

```
42
43      @Override
44      public String describe() {
45          return this.description;
46      }
47
48      @Override
49      public String[] getAliases() {
50          return this.aliases;
51      }
52  }
```

```java
package uk.insrt.coursework.zuul.entities;

import java.util.ArrayList;

import uk.insrt.coursework.zuul.io.IOSystem;
import uk.insrt.coursework.zuul.world.Direction;
import uk.insrt.coursework.zuul.world.Location;
import uk.insrt.coursework.zuul.world.Room;
import uk.insrt.coursework.zuul.world.World;

/**
 * Player entity which we can control and move around.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EntityPlayer extends Entity {
    private ArrayList<Room> previousRooms;
    private ArrayList<Direction> retreatingDirection;

    /**
     * Construct a new Player Entity.
     * @param world World to place Player in
     */
    public EntityPlayer(World world) {
        super(world, new Location(), 70);
        this.previousRooms = new ArrayList<>();
        this.retreatingDirection = new ArrayList<>();
        this.inventory.setMaxWeight(35);
    }

    @Override
    public String[] getAliases() {
        return new String[] {
            "player", "me", "myself", "self", "yourself"
        };
    }

    @Override
    public String describe() {
        // We may skip defining how the Player looks,
```

```
42          // this is because EntityPlayer is ignored
43          // when looking around the room.
44          return "";
45      }
46
47      /**
48       * Move in a direction as instructed by command.
49       * @param direction Target Direction
50       */
51      public void go(Direction direction) {
52          var io = this.getWorld().getIO();
53
54          Room room = this.getRoom();
55          if (room == null) {
56              io.println("You appear to be trapped.");
57              return;
58          }
59
60          if (!room.canLeave(direction)) return;
61
62          Room destination = room.getAdjacent(direction);
63          if (destination == null) {
64              io.println("You cannot go this way.");
65              return;
66          }
67
68          this.retreatingDirection.add(direction.flip());
69          this.previousRooms.add(this.getRoom());
70          this.setLocation(destination);
71      }
72
73      /**
74       * Move to the previous room the player was in.
75       */
76      public void back() {
77          IOSystem io = this.getWorld().getIO();
78          int index = this.retreatingDirection.size() - 1;
79
80          if (index < 0) {
81              io.println("Nowhere to go back to!");
82              return;
```

```
83          }
84
85          Direction lastDirection = this.retreatingDirection.get(index);
86          if (this.getRoom().hasExit(lastDirection)) {
87              this.retreatingDirection.remove(index);
88              this.setLocation(this.previousRooms.remove(index));
89          } else {
90              io.println("Cannot leave the room this way.");
91          }
92      }
93
94      /**
95       * Clear walk history.
96       */
97      public void clearHistory() {
98          this.retreatingDirection.clear();
99          this.previousRooms.clear();
100     }
101 }
```

```java
package uk.insrt.coursework.zuul.entities;

import java.util.ArrayList;

/**
 * Representation of an Entity's inventory
 * and what they are holding.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Inventory {
    private ArrayList<Entity> items = new ArrayList<>();
    private double maxWeight;

    /**
     * Construct a new Inventory.
     */
    public Inventory() {
        super();
        this.maxWeight = 0;
    }

    /**
     * Set the max weight that can be carried in this inventory.
     * @param maxWeight Max weight (in kg)
     */
    public void setMaxWeight(double maxWeight) {
        this.maxWeight = maxWeight;
    }

    /**
     * Get the maximum weight that can be carried in this inventory.
     * @return Maximum weight that can be carried
     */
    public double getMaxWeight() {
        return this.maxWeight;
    }

    /**
     * Get the current weight of this inventory.
```

```java
     * @return Weight (in kg)
     */
    public double getWeight() {
        return this
            .items
            .stream()
            .mapToDouble(Entity::getWeight)
            .sum();
    }

    /**
     * Check if the inventory is full.
     * @return True if the weight is greater than the max weight
     */
    public boolean isFull() {
        return this.getWeight() >= this.getMaxWeight();
    }

    /**
     * Add an entity to this inventory.
     *
     * There must be sufficient space for the entity.
     * @param entity Target Entity
     * @return Whether we successfully added the new entity.
     */
    public boolean add(Entity entity) {
        if (this.getWeight() + entity.getWeight() > this.maxWeight) {
            return false;
        }

        this.items.add(entity);
        return true;
    }

    /**
     * Remove an entity from this inventory.
     * @param entity Target Entity
     * @return Whether there was any change to the inventory.
     */
    public boolean remove(Entity entity) {
        return this.items.remove(entity);
```

```
83      }
84
85      /**
86       * Get an Iterable over the Entities within this inventory.
87       * @return Iterable over Entities
88       */
89      public Iterable<Entity> getItems() {
90          return this.items;
91      }
92  }
```

```java
package uk.insrt.coursework.zuul.events;

/**
 * Represents a single event fired from
 * any source to be consumed by anything.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Event {
    private boolean propagating = true;

    /**
     * Whether this event can continue running.
     * @return Whether propogation of this event was stopped
     */
    public boolean canRun() {
        return this.propagating;
    }

    /**
     * Stop further propagation of this event.
     */
    public void stopPropagation() {
        this.propagating = false;
    }
}
```

```java
package uk.insrt.coursework.zuul.events;

import java.util.HashMap;
import java.util.HashSet;
import java.util.LinkedHashSet;

/**
 * Event system which manages taking in events
 * from different sources and handles them
 * by firing callbacks on event listeners.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EventSystem {
    private HashMap<Class<? extends Event>, LinkedHashSet<IEventListener<? extends Event>>> listeners = new HashMap<>();

    /**
     * Get existing Event listener list or create a new one if not exists.
     * @param event Event
     * @return Set of event listeners
     */
    private HashSet<IEventListener<? extends Event>> getList(Class<? extends Event> event) {
        var list = this.listeners.get(event);
        if (list == null) {
            list = new LinkedHashSet<>();
            this.listeners.put(event, list);
        }

        return list;
    }

    /**
     * Add a new event listener to this system.
     * @param <E> Generic Event type
     * @param event Event to remove from
     * @param listener Event listener callback
     */
    public<E extends Event> void addListener(Class<E> event, IEventListener<E> listener) {
        this.getList(event).add(listener);
    }
```

```java

    /**
     * Remove an new event listener from this system.
     * @param <E> Generic Event type
     * @param event Event to remove from
     * @param listener Event listener callback
     */
    public<E extends Event> void removeListener(Class<E> event, IEventListener<E> listener) {
        this.getList(event).remove(listener);
    }

    /**
     * Emit an Event.
     * @param <E> Generic Event type
     * @param event Event to emit
     */
    @SuppressWarnings("unchecked")
    public <E extends Event> void emit(E event) {
        var listeners = this.listeners.get(event.getClass());
        if (listeners == null) return;

        for (@SuppressWarnings("rawtypes") IEventListener listener : listeners) {
            listener.onEvent(event);
            // Previously, there was a try catch ClassCastException
            // but I've since constricted the types on `addListener`
            // and `removeListener` so this should never happen.

            if (!event.canRun())
                break;
        }
    }
}
```

```java
package uk.insrt.coursework.zuul.events;

/**
 * Interface implementing an listener for an arbitrary {@link Event}.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public interface IEventListener<E extends Event> {
    /**
     * Method called when this specific Event is emitted.
     * @param event Event to handle
     */
    public void onEvent(E event);
}
```

```java
package uk.insrt.coursework.zuul.events.world.behaviours;

import java.util.Random;

import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.events.IEventListener;
import uk.insrt.coursework.zuul.events.world.EventTick;
import uk.insrt.coursework.zuul.world.Room;

/**
 * This is a simple behaviour which just randomly decides to move an Entity
 * through a set path whenever the game ticks forwards.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class SimpleWanderAI implements IEventListener<EventTick> {
    private Entity entity;
    private Room[] path;
    private int chance;

    private int index;
    private Random random;

    /**
     * Construct a new wandering behaviour for an Entity with a given path.
     * @param entity Entity which should be moved
     * @param path Path that this Entity should follow
     * @param chance The chance x that this entity moves, where x gives a 1/x fractional chance of moving.
     */
    public SimpleWanderAI(Entity entity, Room[] path, int chance) {
        this.entity = entity;
        this.path = path;
        this.chance = chance;

        this.index = 0;
        this.random = new Random();
    }

    @Override
    public void onEvent(EventTick event) {
```

```
42          if (this.entity.getRoom() != this.path[this.index]) return;
43          if (random.nextInt(this.chance) > 0) return;
44
45          this.index = (this.index + 1) % this.path.length;
46          this.entity.setLocation(this.path[this.index]);
47      }
48  }
```

```java
package uk.insrt.coursework.zuul.events.world;

import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.events.Event;

/**
 * Event fired when an Entity enters a room.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EventEntityEnteredRoom extends Event {
    private Entity entity;

    /**
     * Construct a new EntityEnteredRoom Event.
     * @param entity Target Entity
     */
    public EventEntityEnteredRoom(Entity entity) {
        this.entity = entity;
    }

    /**
     * Get the Entity relating to this event.
     * @return Entity
     */
    public Entity getEntity() {
        return this.entity;
    }
}
```

```java
package uk.insrt.coursework.zuul.events.world;

import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.events.Event;
import uk.insrt.coursework.zuul.world.Room;

/**
 * Event fired when an Entity enters a room.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EventEntityLeftRoom extends Event {
    private Entity entity;
    private Room room;

    /**
     * Construct a new EntityLeftRoom Event.
     * @param entity Target Entity
     * @param room Room the entity left
     */
    public EventEntityLeftRoom(Entity entity, Room room) {
        this.entity = entity;
        this.room = room;
    }

    /**
     * Get the Entity relating to this event.
     * @return Entity
     */
    public Entity getEntity() {
        return this.entity;
    }

    /**
     * Get the Room relating to this event.
     * @return Room
     */
    public Room getRoom() {
        return this.room;
```

```
41        }
42    }
```

```java
package uk.insrt.coursework.zuul.events.world;

import uk.insrt.coursework.zuul.events.Event;

/**
 * Event fired when an arbitrary command is about to be run.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EventProcessCommand extends Event {
    private String cmd;

    /**
     * Construct a new EventProcessCommand Event.
     * @param cmd Target command
     */
    public EventProcessCommand(String cmd) {
        this.cmd = cmd;
    }

    /**
     * Set command for this event.
     * @param cmd Overwrite current command
     */
    public void setCommand(String cmd) {
        this.cmd = cmd;
    }

    /**
     * Get the command relating to this event.
     * @return Arbitrary command
     */
    public String getCommand() {
        return this.cmd;
    }
}
```

```java
package uk.insrt.coursework.zuul.events.world;

import uk.insrt.coursework.zuul.events.Event;

/**
 * Event fired when the game ticks forward.
 * Such as when the player performs an action or goes to sleep.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EventTick extends Event {}
```

```java
package uk.insrt.coursework.zuul;

import java.io.IOException;

import javax.swing.JOptionPane;

import uk.insrt.coursework.zuul.commands.CommandManager;
import uk.insrt.coursework.zuul.content.campaign.CampaignWorld;
import uk.insrt.coursework.zuul.content.campaign.commands.CommandMap;
import uk.insrt.coursework.zuul.content.campaign.commands.CommandWin;
import uk.insrt.coursework.zuul.events.world.EventProcessCommand;
import uk.insrt.coursework.zuul.events.world.EventTick;
import uk.insrt.coursework.zuul.io.IOSystem;
import uk.insrt.coursework.zuul.io.LocalisedIO;
import uk.insrt.coursework.zuul.io.SanitiseIO;
import uk.insrt.coursework.zuul.io.StandardIO;
import uk.insrt.coursework.zuul.sound.EventMusic;
import uk.insrt.coursework.zuul.sound.MusicType;
import uk.insrt.coursework.zuul.sound.SoundManager;
import uk.insrt.coursework.zuul.ui.EventDraw;
import uk.insrt.coursework.zuul.ui.TerminalEmulator;
import uk.insrt.coursework.zuul.util.BlueJ;
import uk.insrt.coursework.zuul.util.Localisation;
import uk.insrt.coursework.zuul.world.World;

/**
 * Class for managing the game loop and initialising the world.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.1-SNAPSHOT
 */
public class Game {
    public static final String GAME_NAME = "World of These";

    private World world;
    private IOSystem io;
    private CommandManager commands;
    private SoundManager soundManager;

    /**
     * Entrypoint to our application.
```

```java
     * @param args Arguments provided to the application
     */
    public static void main(String[] args) {
        new Game().play();
    }

    /**
     * Initialise and start the game.
     */
    public void play() {
        this.init();
        this.start();
    }

    /**
     * Initialise all required resources for the game to run.
     */
    private void init() {
        // Load audio resources.
        this.soundManager = new SoundManager();

        // Determine how the game should run.
        boolean inBlueJ = BlueJ.isRunningInBlueJ();
        int selection = JOptionPane.showConfirmDialog(null, "Play full experience?\nUses custom terminal emulator.
\n(recommended option)", GAME_NAME, JOptionPane.YES_NO_OPTION);
        if (selection != 1) {
            if (inBlueJ) {
                // Fullscreen minimises itself immediately
                // when running from BlueJ, not sure what's
                // going on exactly, just disabling it in general.
                selection = 1;
            } else {
                selection = JOptionPane.showConfirmDialog(null, "Immersive mode?\nRuns emulator in fullscreen.
\n(recommended option)", GAME_NAME, JOptionPane.YES_NO_OPTION);
            }

            this.io = new TerminalEmulator(selection == 0);
        } else {
            this.io = new StandardIO();

            if (inBlueJ) {
```

```
81                  this.io = new SanitiseIO(this.io);
82              }
83          }
84
85          // Notify player that resources are loading.
86          this.io.println("Loading resources...\nThis may take a second.");
87
88          // Setup the command manager.
89          this.commands = new CommandManager();
90          this.commands.registerCommand(new CommandWin());
91
92          // Register the Map command if we're in term emu mode.
93          // We draw images here so it's not available generally.
94          if (this.io instanceof TerminalEmulator) {
95              CommandMap map = new CommandMap();
96              this.commands.registerCommand(map);
97              ((TerminalEmulator) this.io).getEventSystem().addListener(EventDraw.class, map);
98          }
99
100         // Load sounds
101         try {
102             this.soundManager.init();
103         } catch (Exception e) {
104             e.printStackTrace();
105             // If we can't load the sound system,
106             // just play without sound.
107         }
108
109         // Load all the data we need and initialise world.
110         Localisation locale = new Localisation();
111         this.io = new LocalisedIO(this.io, locale);
112
113         // Prompt for language
114         this.io.clear();
115         this.io.print("Welcome! \u1F604\nBefore we start...\n\n"
116             + "\u1F508 Note: \u001B[35mthis game uses sound.\u001B[0m \n\n"
117             + "What language would you like to use?\n"
118             + "1. :uk:Traditional English (recommended)\n"
119             + "2. :us:Simplified English\n"
120             + "3. :de:German\n"
121             + "4. :cz:Czech\n"
```

```
122             + "Selection: ");
123
124         String input = this.io.readLine();
125         this.io.clear();
126
127         String selectedLanguage;
128         if (input.equals("3")) {
129             selectedLanguage = "de_DE";
130         } else if (input.equals("4")) {
131             selectedLanguage = "cs_CZ";
132         } else {
133             selectedLanguage = "en_GB";
134         }
135
136         try {
137             locale.loadLocale(selectedLanguage);
138         } catch (IOException e) {
139             System.err.println("Failed to load translations!");
140             e.printStackTrace();
141         }
142
143         // Construct the world.
144         this.world = new CampaignWorld(this.io);
145
146         // Register sound events.
147         this.soundManager.register(this.world.getEventSystem());
148         this.world.getEventSystem().emit(new EventMusic(MusicType.BgmExplore, true));
149     }
150
151     /**
152      * Start the game loop.
153      */
154     private void start() {
155         this.world.spawnPlayer();
156
157         while (true) {
158             this.io.print("> ");
159             String input = this.io.readLine().toLowerCase();
160
161             EventProcessCommand event = new EventProcessCommand(input);
162             this.world.emit(event);
```

```
163
164             if (this.commands.runCommand(this.world, event.getCommand())) {
165                 break;
166             }
167
168             this.world.emit(new EventTick());
169         }
170
171         this.io.println("Goodbye.");
172
173         try {
174             Thread.sleep(1000);
175             this.io.dispose();
176         } catch (Exception e) {}
177
178         this.soundManager.dispose();
179     }
180 }
```

```java
package uk.insrt.coursework.zuul.io;

import java.awt.Color;
import java.util.regex.Pattern;

/**
 * ANSI escape codes
 * Used https://stackoverflow.com/a/5762502 as a reference.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Ansi {
    public static final String Reset = "\u001B[0m";
    public static final String Black = "\u001B[30m";
    public static final String Red = "\u001B[31m";
    public static final String Green = "\u001B[32m";
    public static final String Yellow = "\u001B[33m";
    public static final String Blue = "\u001B[34m";
    public static final String Purple = "\u001B[35m";
    public static final String Cyan = "\u001B[36m";
    public static final String White = "\u001B[37m";

    public static final String BackgroundBlack = "\u001B[40m";
    public static final String BackgroundRed = "\u001B[41m";
    public static final String BackgroundGreen = "\u001B[42m";
    public static final String BackgroundYellow = "\u001B[43m";
    public static final String BackgroundBlue = "\u001B[44m";
    public static final String BackgroundPurple = "\u001B[45m";
    public static final String BackgroundCyan = "\u001B[46m";
    public static final String BackgroundWhite = "\u001B[47m";

    /**
     * Regex Pattern used to match Ansi codes forwards.
     */
    public static final Pattern AnsiPattern = Pattern.compile("^\\u001B\\[(\\d{1,3})m");

    private static final Color ColorBlack = new Color(0, 0, 0);
    private static final Color ColorRed = new Color(224, 108, 117);
    private static final Color ColorGreen = new Color(152, 195, 121);
    private static final Color ColorYellow = new Color(229, 192, 123);
```

```
42      private static final Color ColorBlue = new Color(97, 175, 239);
43      private static final Color ColorMagenta = new Color(198, 120, 221);
44      private static final Color ColorCyan = new Color(86, 182, 194);
45      private static final Color ColorWhite = new Color(255, 255, 255);
46
47      /**
48       * Convert a given escape code value, {@code (\d+?)} in {@link #AnsiPattern}, to a Color.
49       * @param value Escape code value
50       * @return Resolved Java awt Color
51       */
52      public static Color fromEscapeCode(int value) {
53          switch (value % 10) {
54              case 0: return ColorBlack;
55              case 1: return ColorRed;
56              case 2: return ColorGreen;
57              case 3: return ColorYellow;
58              case 4: return ColorBlue;
59              case 5: return ColorMagenta;
60              case 6: return ColorCyan;
61              case 7:
62              default: return ColorWhite;
63          }
64      }
65  }
```

```java
package uk.insrt.coursework.zuul.io;

/**
 * Interface representing an arbitrary IO system.
 * This can be implemented to input or output from various interfaces.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public interface IOSystem {
    /**
     * Print a string out through an arbitrary output channel.
     * @param out String to print
     */
    public void print(String out);

    /**
     * Print a string out through an arbitrary output channel and append {@code \n}.
     * @param out String to print
     */
    public void println(String out);

    /**
     * Read a String up until the first encountered {@code \n} from an arbitrary input channel.
     * @return String of line read in
     */
    public String readLine();

    /**
     * Dispose of the arbitrary input and output channels.
     */
    public void dispose();

    /**
     * Clear the output.
     */
    public void clear();
}
```

```java
package uk.insrt.coursework.zuul.io;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

import uk.insrt.coursework.zuul.util.Localisation;

/**
 * Translate and localise any incoming output.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class LocalisedIO implements IOSystem {
    private final Pattern pattern = Pattern.compile("<([\\w\\.]+?)>");

    private IOSystem io;
    private Localisation locale;

    /**
     * Construct a new LocalisedIO.
     * @param io Provided IO system we should feed into
     * @param locale Locale to apply to any i18n strings
     */
    public LocalisedIO(IOSystem io, Localisation locale) {
        this.io = io;
        this.locale = locale;
    }

    /**
     * Replace i18n strings in any given String with their actual localised values.
     * Using replacement code from https://stackoverflow.com/a/27359491.
     * @param input String to process
     * @return Final processed string
     */
    private String replace(String input) {
        StringBuffer result = new StringBuffer();
        Matcher matcher = this.pattern.matcher(input);

        while (matcher.find()) {
            matcher.appendReplacement(result, this.locale.get(matcher.group(1)));
```

```
42          }
43
44          matcher.appendTail(result);
45          return result.toString();
46      }
47
48      @Override
49      public void print(String out) {
50          this.io.print(this.replace(out));
51      }
52
53      @Override
54      public void println(String out) {
55          this.io.println(this.replace(out));
56      }
57
58      @Override
59      public String readLine() {
60          return this.io.readLine();
61      }
62
63      @Override
64      public void dispose() {
65          this.io.dispose();
66      }
67
68      @Override
69      public void clear() {
70          this.io.clear();
71      }
72  }
```

```java
package uk.insrt.coursework.zuul.io;

/**
 * Sanitise incoming output and remove any Ansi escape sequences.
 * This is required to print out into the BlueJ console without additional characters.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class SanitiseIO implements IOSystem {
    private final String ansiPattern = "\\u001B\\[(\\d{1,3})m";
    private IOSystem io;

    /**
     * Construct a new SanitiseIO.
     * @param io Provided IO system we should feed into
     */
    public SanitiseIO(IOSystem io) {
        this.io = io;
    }

    @Override
    public void print(String out) {
        this.io.print(out.replaceAll(this.ansiPattern, " "));
    }

    @Override
    public void println(String out) {
        this.io.println(out.replaceAll(this.ansiPattern, " "));
    }

    @Override
    public String readLine() {
        return this.io.readLine();
    }

    @Override
    public void dispose() {
        this.io.dispose();
    }

```

```
42      @Override
43      public void clear() {
44          this.io.clear();
45      }
46  }
```

```java
package uk.insrt.coursework.zuul.io;

import java.util.Scanner;

/**
 * A simple IO system implementation which feeds
 * into System.out and takes data from System.in
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class StandardIO implements IOSystem {
    private Scanner reader;

    /**
     * Construct a new StandardIO.
     */
    public StandardIO() {
        this.reader = new Scanner(System.in);
    }

    @Override
    public void print(String out) {
        System.out.print(out);
    }

    @Override
    public void println(String out) {
        System.out.println(out);
    }

    @Override
    public String readLine() {
        return this.reader.nextLine();
    }

    @Override
    public void dispose() {}

    @Override
    public void clear() {
```

```
42            this.print("\u000C");
43        }
44    }
```

```java
package uk.insrt.coursework.zuul.sound;

import uk.insrt.coursework.zuul.events.Event;

/**
 * This event is used to tell the sound manager to start or stop playing music.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.1-SNAPSHOT
 */
public class EventMusic extends Event {
    private MusicType target;
    private boolean play;

    /**
     * Construct new EventMusic
     * @param target Target music type
     * @param play Whether to play or stop
     */
    public EventMusic(MusicType target, boolean play) {
        this.target = target;
        this.play = play;
    }

    /**
     * Get the music type
     * @return music type
     */
    public MusicType getTarget() {
        return this.target;
    }

    /**
     * Get whether it should play
     * @return True if it should play or false if it should stop
     */
    public boolean shouldPlay() {
        return this.play;
    }
}
```

```java
package uk.insrt.coursework.zuul.sound;

import uk.insrt.coursework.zuul.events.Event;

/**
 * Event used to tell the sound manager to play a sound.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.1-SNAPSHOT
 */
public class EventSound extends Event {
    private SoundType target;

    /**
     * Construct new EventSound
     * @param target Target sound
     */
    public EventSound(SoundType target) {
        this.target = target;
    }

    /**
     * Get sound type
     * @return Sound type
     */
    public SoundType getTarget() {
        return this.target;
    }
}
```

```java
package uk.insrt.coursework.zuul.sound;

/**
 * Music available in the game.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.1-SNAPSHOT
 */
public enum MusicType {
    Bay1,
    Bay2,
    Nature,
    City1,
    City2,

    BgmExplore,
    BgmMission,
    BgmConclusion
}
```

```java
package uk.insrt.coursework.zuul.sound;

import java.util.HashMap;

import kuusisto.tinysound.Sound;
import kuusisto.tinysound.TinySound;
import uk.insrt.coursework.zuul.events.EventSystem;

/**
 * Class used to manage background music and foreground sounds.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.1-SNAPSHOT
 */
public class SoundManager {
    private TinySound lib;
    private boolean loaded;
    private boolean initialised;

    private HashMap<SoundType, Sound> sounds;
    private HashMap<MusicType, Sound> music;

    /**
     * Construct a new SoundManager
     */
    public SoundManager() {
        this.loaded = false;
        this.initialised = false;
        this.sounds = new HashMap<>();
        this.music = new HashMap<>();
    }

    /**
     * Initialise the sound manager and load all music and sound files.
     * @throws Exception if something goes wrong.
     */
    public void init() throws Exception {
        this.lib = TinySound.init();
        this.initialised = true;

        this.sounds.put(SoundType.MoneyBag, lib.loadSound("/sounds/money-bag.ogg"));
```

```
42          this.sounds.put(SoundType.WormHole, lib.loadSound("/sounds/worm-hole-enc.ogg"));
43
44          this.music.put(MusicType.Bay1, lib.loadSound("/sounds/bay1-enc.ogg"));
45          this.music.put(MusicType.Bay2, lib.loadSound("/sounds/bay2-enc.ogg"));
46          this.music.put(MusicType.Nature, lib.loadSound("/sounds/nature-enc.ogg"));
47          this.music.put(MusicType.City1, lib.loadSound("/sounds/city-1-quieter.ogg"));
48          this.music.put(MusicType.City2, lib.loadSound("/sounds/city-2-quieter.ogg"));
49
50          this.music.put(MusicType.BgmExplore, lib.loadSound("/sounds/bgm-explore-quieter.ogg"));
51          this.music.put(MusicType.BgmMission, lib.loadSound("/sounds/bgm-mission-quieter.ogg"));
52          this.music.put(MusicType.BgmConclusion, lib.loadSound("/sounds/bgm-conclude-quieter.ogg"));
53
54          this.loaded = true;
55      }
56
57      /**
58       * Clean up the sound library.
59       */
60      public void dispose() {
61          if (!this.initialised) return;
62          this.lib.shutdown();
63      }
64
65      /**
66       * Register sound and music events with provided event system.
67       * @param eventSystem Event system
68       */
69      public void register(EventSystem eventSystem) {
70          eventSystem.addListener(EventSound.class, event -> {
71              if (!this.loaded) return;
72              try {
73                  this.sounds.get(event.getTarget()).play();
74              } catch (Exception e) { /* ignore sound if we fail here */ }
75          });
76
77          eventSystem.addListener(EventMusic.class, event -> {
78              if (!this.loaded) return;
79              try {
80                  Sound song = this.music.get(event.getTarget());
81                  if (event.shouldPlay()) {
82                      song.play();
```

```
                } else {
                    song.stop();
                }
            } catch (Exception e) { /* ignore sound if we fail here */ }
        });
    }
}
```

```java
package uk.insrt.coursework.zuul.sound;

/**
 * Sounds available in the game.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.1-SNAPSHOT
 */
public enum SoundType {
    MoneyBag,
    WormHole,
}
```

```java
package uk.insrt.coursework.zuul.ui;

import java.awt.Image;

/**
 * Representation of a single Emoji which can be rendered in the terminal emulator.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Emoji {
    private Image image;
    private int width;
    private int height;

    /**
     * Construct a new Emoji given the image and unicode representation.
     * @param image Image to render when this Emoji is used
     * @param unicode Unicode representation of this Emoji, used to determine width
     */
    public Emoji(Image image, String unicode) {
        this.image = image;
        this.width = (int) unicode.chars().count();
        this.height = 1;
    }

    /**
     * Construct a new Emoji given the image and size constraints.
     * @param image Image to render when this Emoji is used
     * @param width Width of this Emoji
     * @param height Height of this Emoji
     */
    public Emoji(Image image, int width, int height) {
        this.image = image;
        this.width = width;
        this.height = height;
    }

    /**
     * Get the Image for this Emoji
     * @return Image
```

```
42          */
43         public Image getImage() {
44             return this.image;
45         }
46 
47         /**
48          * Get the calculated width of this Emoji
49          * @return Width
50          */
51         public int getWidth() {
52             return this.width;
53         }
54 
55         /**
56          * Get the calculated height of this Emoji
57          * @return Height
58          */
59         public int getHeight() {
60             return this.height;
61         }
62     }
```

```java
package uk.insrt.coursework.zuul.ui;

import java.awt.Image;
import java.io.IOException;
import java.io.InputStream;
import java.nio.charset.StandardCharsets;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;

import javax.imageio.ImageIO;

import com.moandjiezana.toml.Toml;

import org.apache.commons.io.IOUtils;

import uk.insrt.coursework.zuul.util.Tree;

/**
 * Class which helps manage loading and resolving Emojis.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EmojiManager {
    private HashMap<String, Emoji> emojis;
    private Tree<Character, String> emojiTree;
    private Tree<Character, String> currentNode;

    /**
     * Construct a new EmojiManager.
     */
    public EmojiManager() {
        this.emojis = new HashMap<>();
        this.emojiTree = new Tree<>();
        this.currentNode = this.emojiTree;
    }

    /**
     * Check whether a given Emoji is present in this manager.
```

```java
     * @param emoji Unicode representation of Emoji
     * @return True if the Emoji is available
     */
    public boolean hasEmoji(String emoji) {
        return this.emojis.containsKey(emoji);
    }

    /**
     * Get an Emoji by its Unicode representation.
     * @param emoji Unicode representation of Emoji
     * @return The Emoji or null if it doesn't exist
     */
    public Emoji getEmoji(String emoji) {
        return this.emojis.get(emoji);
    }

    /**
     * Get currently matched emoji and resets position.
     * @return Emoji if it was found, or null if not
     */
    public Emoji getEmoji() {
        String value = this.currentNode.getValue();
        Emoji emoji = this.emojis.get(value);
        this.resetState();
        return emoji;
    }

    /**
     * Reset the state of the matching mechanism.
     */
    public void resetState() {
        this.currentNode = this.emojiTree;
    }

    /**
     * The matching mechanism has not matched any characters to potential emojis.
     */
    public static final int MATCH_NONE = 0;

    /**
     * The matching mechanism has matched some characters to potential emojis.
```

```
83          */
84         public static final int MATCH_SOME = 1;
85
86         /**
87          * The matching mechanism has matched an emoji.
88          */
89         public static final int MATCH_FOUND = 2;
90
91         /**
92          * Match the next character.
93          * @param c Character to match against
94          * @return One of {@link #MATCH_NONE}, {@link #MATCH_SOME} or {@link #MATCH_FOUND}
95          */
96         public int match(char c) {
97             var child = this.currentNode.getChild(c);
98             if (child != null) {
99                 this.currentNode = child;
100                if (child.getValue() == null) return MATCH_SOME;
101                return MATCH_FOUND;
102            }
103
104            if (this.currentNode != this.emojiTree) {
105                this.currentNode = this.emojiTree;
106                child = this.emojiTree.getChild(c);
107                if (child != null) {
108                    if (child.getValue() != null) return MATCH_SOME;
109                    return MATCH_FOUND;
110                }
111            }
112
113            return MATCH_NONE;
114        }
115
116        /**
117         * Load emoji definitions and resources from a given resource directory.
118         * @param rootDir Root directory at which we expect a valid {@code definitions.toml} to exist
119         * @throws IOException if the definition file is missing or defined emojis are invalid
120         */
121        public void loadResources(String rootDir) throws IOException {
122            InputStream defnStream = this.getClass().getResourceAsStream(rootDir + "/definitions.toml");
123            // We need to force UTF-8 encoding or else unicode emojis may get mangled.
```

```
124             String defnString = IOUtils.toString(defnStream, StandardCharsets.UTF_8);
125             Toml defn = new Toml().read(defnString);
126             List<HashMap<String, Object>> emojis = defn.getList("emojis");
127 
128             // Load each emoji in sequence
129             for (var emoji : emojis) {
130                 String path = (String) emoji.get("path");
131                 String unicode = (String) emoji.get("unicode");
132 
133                 InputStream stream = this.getClass().getResourceAsStream(rootDir + "/" + path);
134                 Image image = ImageIO.read(stream);
135 
136                 Emoji newEmoji;
137                 if (emoji.containsKey("width") && emoji.containsKey("height")) {
138                     int width = ((Long) emoji.get("width")).intValue();
139                     int height = ((Long) emoji.get("height")).intValue();
140                     newEmoji = new Emoji(image, width, height);
141                 } else {
142                     newEmoji = new Emoji(image, unicode);
143                 }
144                 this.emojis.put(unicode, newEmoji);
145                 this.emojiTree.addChildWithPath(
146                     new ArrayList<>(Arrays.asList(
147                         unicode.chars()
148                             .mapToObj(c -> (char)c)
149                             .toArray(Character[]::new))),
150                     unicode
151                 );
152             }
153         }
154     }
```

```java
package uk.insrt.coursework.zuul.ui;

import java.awt.Graphics;

import uk.insrt.coursework.zuul.events.Event;

/**
 * Event fired when the terminal emulator draws a new frame.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class EventDraw extends Event {
    private Graphics g;
    private float ox;
    private float oy;
    private float fw;
    private float fh;

    /**
     * Construct a new EventDraw Event.
     * @param g Graphics context
     * @param ox Origin X position
     * @param oy Origin Y position
     * @param fw Font character width
     * @param fh Font character height
     */
    public EventDraw(Graphics g, float ox, float oy, float fw, float fh) {
        this.g = g;
        this.ox = ox;
        this.oy = oy;
        this.fw = fw;
        this.fh = fh;
    }

    /**
     * Get the Graphics relating to this event.
     * @return Graphics
     */
    public Graphics getGraphics() {
        return this.g;
```

```
42      }
43
44      /**
45       * Get the origin X position of the contents of the terminal.
46       * @return X position
47       */
48      public float getOriginX() {
49          return this.ox;
50      }
51
52      /**
53       * Get the origin Y position of the contents of the terminal.
54       * @return Y position
55       */
56      public float getOriginY() {
57          return this.oy;
58      }
59
60      /**
61       * Get the character width.
62       * @return Character width
63       */
64      public float getCharWidth() {
65          return this.fw;
66      }
67
68      /**
69       * Get the character height.
70       * @return Character height
71       */
72      public float getCharHeight() {
73          return this.fh;
74      }
75  }
```

```java
package uk.insrt.coursework.zuul.ui;

import java.awt.BorderLayout;
import java.awt.Dimension;
import java.awt.GraphicsEnvironment;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;

import javax.swing.JFrame;

/**
 * Window frame for {@link TerminalEmulator}
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class JTerminalFrame extends JFrame {
    private JTerminalView view;
    private boolean fullscreen;

    /**
     * Construct a new JTerminalFrame
     * @param emulator Terminal emulator this frame belongs to
     */
    public JTerminalFrame(TerminalEmulator emulator) {
        this.view = new JTerminalView(emulator);
        this.fullscreen = emulator.isFullscreen();
        this.makeFrame(emulator);
    }

    /**
     * Make and display all of the elements within this frame.
     * @param emulator Terminal emulator this frame belongs to
     */
    public void makeFrame(TerminalEmulator emulator) {
        this.setLayout(new BorderLayout());
        this.add(this.view);
        this.pack();
        this.setLocationRelativeTo(null);
        this.setMinimumSize(new Dimension(640, 640));
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
42
43          // Add a listener for any user input.
44          // https://stackoverflow.com/a/21970006
45          this.addKeyListener(new KeyListener() {
46              public void keyPressed(KeyEvent event) {
47                  int code = event.getKeyCode();
48                  switch (code) {
49                      case 8: {
50                          emulator.pop();
51                          return;
52                      }
53                      case 10: {
54                          emulator.flush();
55                          break;
56                      }
57                      default: {
58                          if ((code >= 65 && code <= 90) || (code >= 48 && code <= 57) || code == 32) {
59                              emulator.push(event.getKeyChar());
60                          }
61                      }
62                  }
63              }
64
65              public void keyTyped(KeyEvent e) {}
66              public void keyReleased(KeyEvent e) {}
67          });
68
69          // We are ready to display, show everything.
70          // Prerequisite to making the frame fullscreen too.
71          this.setVisible(true);
72
73          // If we are allowed to launch in fullscreen, switch to that mode now.
74          if (this.fullscreen) {
75              GraphicsEnvironment
76                  .getLocalGraphicsEnvironment()
77                  .getDefaultScreenDevice()
78                  .setFullScreenWindow(this);
79          }
80      }
81
82      @Override
```

```
83      public void dispose() {
84          super.dispose();
85          this.view.dispose();
86      }
87  }
```

```java
package uk.insrt.coursework.zuul.ui;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.FontFormatException;
import java.awt.Graphics;
import java.awt.event.ComponentAdapter;
import java.awt.event.ComponentEvent;
import java.awt.font.FontRenderContext;
import java.awt.geom.AffineTransform;
import java.io.IOException;
import java.io.InputStream;

import javax.swing.JPanel;

import com.moandjiezana.toml.Toml;

/**
 * Rendering component of {@link TerminalEmulator}
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class JTerminalView extends JPanel {
    private TerminalEmulator emulator;

    private EmojiManager emojiManager;
    private Font derivedFont;
    private Font font;

    private Thread blinkThread;
    private boolean blinkState;

    private float fw, fh, foffset, fratio;

    /**
     * Construct a TerminalView
     * @param emulator Terminal emulator this view belongs to
     */
    public JTerminalView(TerminalEmulator emulator) {
```

```
42          this.emulator = emulator;
43          this.emojiManager = new EmojiManager();
44          this.blinkState = false;
45          this.loadResources();
46          this.makeFrame();
47      }
48
49      /**
50       * Prepare the terminal view for rendering.
51       */
52      public void makeFrame() {
53          var view = this;
54          this.setBackground(Color.BLACK);
55
56          // Register a listener to repaint and adjust measurements when resizing window.
57          // https://stackoverflow.com/a/8917978
58          this.addComponentListener(new ComponentAdapter() {
59              public void componentResized(ComponentEvent e) {
60                  view.setBackground(Color.BLACK);
61                  view.deriveFont();
62                  view.repaint();
63              }
64          });
65
66          // Start a new thread for blinking the cursor.
67          this.blinkThread = new Thread("Blink Thread") {
68              public void run() {
69                  try {
70                      while (true) {
71                          view.blinkState = !view.blinkState;
72                          view.repaint();
73
74                          Thread.sleep(500);
75                      }
76                  } catch (InterruptedException e) {}
77              }
78          };
79
80          this.blinkThread.start();
81      }
82
```

```java
    /**
     * Load any resources required by the terminal view to render itself properly.
     */
    public void loadResources() {
        try {
            InputStream stream = this.getClass().getResourceAsStream("/emulator.toml");
            Toml defn = new Toml().read(stream);

            // If a font is defined, load it.
            String font = defn.getString("font");
            if (font != null) {
                this.loadFont(font, 32.0f / 12.8f);
            }

            // If an emoji root directory is defined, load it.
            String rootDir = defn.getString("emojis");
            if (rootDir != null) {
                this.emojiManager.loadResources(rootDir);
            }
        } catch (Exception e) {
            System.err.println("Failed to load any resources for terminal view!");
            e.printStackTrace();
        }
    }

    /**
     * Load a specific font with a known ratio.
     * We expect this font to be monoscape.
     * @param source Path to the font to be loaded
     * @param ratio Ratio of width to height for this font
     * @throws IOException if the font cannot be loaded from a given path
     * @throws FontFormatException if the font is of an incorrect format, we expect a TTF
     */
    public void loadFont(String source, float ratio) throws IOException, FontFormatException {
        InputStream stream = this.getClass().getResourceAsStream(source);
        this.fratio = ratio;
        this.font = Font.createFont(Font.TRUETYPE_FONT, stream);
    }

    /**
     * Derive the font measurements before we continue rendering.
```

```
124         */
125        public void deriveFont() {
126            final int padding = 100;
127
128            // To find the height of the font, we take the smallest
129            // side of the window height or the proportional height
130            // found from the window width, and then we divide it
131            // by our fixed terminal height.
132            float height = Math.min(
133                this.fratio *
134                      (float) (this.getWidth() - padding)
135                    / (float) TerminalEmulator.TERMINAL_WIDTH,
136                (this.getHeight() - padding)
137                    / (float) TerminalEmulator.TERMINAL_HEIGHT
138            );
139
140            this.derivedFont = this.font.deriveFont(height);
141
142            // The FontRenderContext is used to determine the font dimensions
143            var frc = new FontRenderContext(new AffineTransform(), true, true);
144            var bounds = this.derivedFont.getStringBounds(" ", frc);
145
146            this.fw = (float) bounds.getWidth();
147            this.fh = (float) bounds.getHeight();
148
149            // We need to find the distance between the baseline
150            // and the ascender so we can properly align everything.
151            // https://docs.oracle.com/javase/tutorial/2d/text/fontconcepts.html
152            this.foffset = this.derivedFont.getLineMetrics(" ", frc).getAscent();
153        }
154
155        /**
156         * Dispose of this terminal view.
157         */
158        public void dispose() {
159            // We need to kill the blind thread when disposing
160            // of the UI, but that's not possible so instead I
161            // am interrupting the thread, catching that and
162            // exiting out peacefully.
163            // https://docs.oracle.com/javase/1.5.0/docs/guide/misc/threadPrimitiveDeprecation.html
164            this.blinkThread.interrupt();
```

```
165     }
166
167     @Override
168     public Dimension getPreferredSize() {
169         return new Dimension(1280, 960);
170     }
171
172     @Override
173     protected void paintComponent(Graphics g) {
174         // https://stackoverflow.com/a/17922749
175         super.paintComponent(g);
176
177         // Setup our canvas for rendering.
178         g.setColor(Color.BLACK);
179         g.setFont(this.derivedFont);
180         g.fillRect(0, 0, this.getWidth(), this.getHeight());
181
182         // Find our topleft-most (x,y) to start rendering from.
183         TextBuffer buffer = this.emulator.getBuffer();
184         float ox = (this.getWidth() - this.fw * buffer.getWidth()) / 2;
185         float oy = (this.getHeight() - this.fh * buffer.getHeight()) / 2;
186
187         // Render each cell individually.
188         for (int y=0;y<buffer.getHeight();y++) {
189             int skipChars = 0;
190             for (int x=0;x<buffer.getWidth();x++) {
191                 // If needs be, skip chars in this line.
192                 if (skipChars > 0) {
193                     skipChars--;
194                     continue;
195                 }
196
197                 // Get the character to render.
198                 char c = buffer.getChar(x, y);
199
200                 // Match each char for emoji codepoints.
201                 // If we start to match an emoji, peek ahead.
202                 int emojiMatch = this.emojiManager.match(c);
203                 int offset = 0;
204                 while (emojiMatch == EmojiManager.MATCH_SOME) {
205                     emojiMatch = this.emojiManager.match(buffer.getChar(x + ++offset, y));
```

```
206                 }
207
208                 // Get this cell's background and foreground colours.
209                 Color bg = buffer.getBg(x, y);
210                 Color fg = buffer.getFg(x, y);
211
212                 // Find this char's offset.
213                 int drawX = Math.round(ox + this.fw * x);
214                 int drawY = Math.round(oy + this.fh * y);
215
216                 // Draw rect if there's a background present.
217                 if (bg != null && bg != Color.BLACK) {
218                     g.setColor(bg);
219                     g.fillRect(drawX, drawY, (int) Math.ceil(this.fw), (int) Math.ceil(this.fh));
220                 }
221
222                 // If we're drawing an emoji, get the image and skip text.
223                 if (emojiMatch == EmojiManager.MATCH_FOUND) {
224                     Emoji emoji = this.emojiManager.getEmoji();
225                     g.drawImage(
226                         emoji.getImage(),
227                         drawX, drawY,
228                         Math.round(this.fw * emoji.getWidth()),
229                         Math.round(this.fh * emoji.getHeight()),
230                         this
231                     );
232
233                     skipChars = offset;
234                     continue;
235                 }
236
237                 // Drawing the char if it's not a space, we have to
238                 // take care to add the offset we previously found or
239                 // otherwise the distance between the baseline and
240                 // the ascender. This is because Graphics.drawString
241                 // draws text from the leftmost baseline equal to (x,y).
242                 if (c != 0 && c != ' ') {
243                     g.setColor(fg);
244                     g.drawString(
245                         String.valueOf(c),
246                         drawX,
```

```
247                             Math.round(drawY + this.foffset)
248                         );
249                     }
250                 }
251 
252                 this.emojiManager.resetState();
253             }
254 
255             // Draw a blinker to indicate the user can type here.
256             // Offset it a bit to not interfere with any text on-screen.
257             if (this.blinkState) {
258                 g.setColor(Color.WHITE);
259                 g.fillRect(
260                     (int) (ox + this.fw * buffer.getPosX() + 1),
261                     (int) (oy + this.fh * buffer.getPosY() + this.foffset),
262                     (int) this.fw - 2,
263                     (int) (this.fh / 8)
264                 );
265             }
266 
267             // Fire draw event for custom rendering.
268             this.emulator.getEventSystem().emit(new EventDraw(g, ox, oy, this.fw, this.fh));
269         }
270     }
```

```java
package uk.insrt.coursework.zuul.ui;

import java.awt.Color;
import java.awt.EventQueue;
import java.util.concurrent.BlockingQueue;
import java.util.concurrent.LinkedBlockingQueue;

import uk.insrt.coursework.zuul.events.EventSystem;
import uk.insrt.coursework.zuul.io.IOSystem;

/**
 * A terminal emulator which implements an IO system to
 * be arbitrarily plugged into any existing components.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class TerminalEmulator implements IOSystem {
    public static final int TERMINAL_WIDTH = 80;
    public static final int TERMINAL_HEIGHT = 25;

    private BlockingQueue<String> queue;
    private EventSystem eventSystem;
    private JTerminalFrame frame;
    private boolean fullscreen;
    private TextBuffer buffer;
    private String input;

    /**
     * Construct and build a new TerminalEmulator
     * @param fullscreen Whether to launch the emulator in fullscreen
     */
    public TerminalEmulator(boolean fullscreen) {
        this.queue = new LinkedBlockingQueue<>();
        this.eventSystem = new EventSystem();
        this.buffer = new TextBuffer(TERMINAL_WIDTH, TERMINAL_HEIGHT);
        this.fullscreen = fullscreen;
        this.input = new String();

        this.buildFrame();
    }
```

```java

    /**
     * Construct a new TerminalEmulator and force windowed mode.
     */
    public TerminalEmulator() {
        this(false);
    }

    /**
     * Build and show the terminal emulator.
     */
    public void buildFrame() {
        var emulator = this;
        EventQueue.invokeLater(new Runnable() {
            @Override
            public void run() {
                emulator.frame = new JTerminalFrame(emulator);
            }
        });
    }

    /**
     * Get the local event system for this emulator.
     * @return The event system
     */
    public EventSystem getEventSystem() {
        return this.eventSystem;
    }

    /**
     * Get the emulator's text buffer.
     * @return The text buffer
     */
    public TextBuffer getBuffer() {
        return this.buffer;
    }

    /**
     * Tell the terminal frame to repaint contents.
     */
    private void repaint() {
```

```
83          if (this.frame != null) {
84              this.frame.repaint();
85          }
86      }
87
88      /**
89       * Check whether we are in fullscreen mode.
90       * @return True if we are fullscreen
91       */
92      public boolean isFullscreen() {
93          return this.fullscreen;
94      }
95
96      /**
97       * Push a new character to the input buffer.
98       * @param c Character to push
99       */
100     public void push(char c) {
101         if (this.buffer.getPosX() + 1 == TERMINAL_WIDTH) return;
102
103         this.input += c;
104         this.buffer.write(new String(new char[] { c }));
105         this.buffer.setLastFg(Color.GRAY);
106         this.repaint();
107     }
108
109     /**
110      * Pop last character from the input buffer.
111      */
112     public void pop() {
113         if (this.input.length() > 0) {
114             this.input = this.input.substring(0, this.input.length() - 1);
115             this.buffer.backspace();
116             this.repaint();
117         }
118     }
119
120     /**
121      * Flush input from terminal emulator thread and send it to whatever is
122      * waiting for it on another thread. Uses a blocking queue to send data
123      * between threads, as seen here: https://stackoverflow.com/a/23413506
```

```
124         */
125        public void flush() {
126            this.queue.add(this.input);
127            this.input = "";
128        }
129
130        @Override
131        public void print(String out) {
132            buffer.write(out);
133            this.repaint();
134        }
135
136        @Override
137        public void println(String out) {
138            buffer.write(out + "\n");
139            this.repaint();
140        }
141
142        @Override
143        public String readLine() {
144            try {
145                String line = this.queue.take();
146                this.print("\n");
147                return line;
148            } catch (Exception err) {
149                err.printStackTrace();
150                System.exit(1);
151                return " ";
152            }
153        }
154
155        @Override
156        public void dispose() {
157            this.frame.dispose();
158        }
159
160        @Override
161        public void clear() {
162            this.buffer.clear();
163            this.input = "";
```

```
164         }
165     }
```

```java
package uk.insrt.coursework.zuul.ui;

import java.awt.Color;
import java.util.regex.Matcher;

import uk.insrt.coursework.zuul.io.Ansi;

/**
 * Representation of a text and colour buffer.
 * Provides various utilities for manipulating text on screen.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class TextBuffer {
    private char[][] buffer;
    private Color[][] bufferBg;
    private Color[][] bufferFg;

    private int width;
    private int height;

    private int posX;
    private int posY;

    private Color bg;
    private Color fg;

    private boolean overflow;

    /**
     * Construct a new TextBuffer with given constraints.
     * @param width Buffer width
     * @param height Buffer height
     */
    public TextBuffer(int width, int height) {
        this.width = width;
        this.height = height;
        this.clear();
    }

```

```java
    /**
     * Remove top-most row and move all the other rows up.
     */
    public void shift() {
        for (int i=0;i<this.height-1;i++) {
            this.buffer[i] = this.buffer[i + 1];
            this.bufferBg[i] = this.bufferBg[i + 1];
            this.bufferFg[i] = this.bufferFg[i + 1];
        }

        this.bufferBg[this.height - 1] = new Color[this.width];
        this.bufferFg[this.height - 1] = new Color[this.width];
        this.buffer[this.height - 1] = new char[this.width];
    }

    /**
     * Remove the last previous character written to the buffer.
     */
    public void backspace() {
        if (this.posX == 0) return;
        this.buffer[this.posY][--this.posX] = ' ';
    }

    /**
     * Retroactively set the foreground for the previous character written.
     * @param color Foreground colour
     */
    public void setLastFg(Color color) {
        this.bufferFg[this.posY][this.posX - 1] = color;
    }

    /**
     * Retroactively set the background for the previous character written.
     * @param color Background colour
     */
    public void setLastBg(Color color) {
        this.bufferBg[this.posY][this.posX - 1] = color;
    }

    /**
     * Write a new character to the text buffer.
```

```
 83         * This will move the cursor forwards.
 84         * @param c Character to write
 85         */
 86        public void write(char c) {
 87            // If we encounter a newline, shift downwards or go on to a new line.
 88            if (c == '\n') {
 89                if (this.overflow) {
 90                    this.overflow = false;
 91                    return;
 92                }
 93
 94                this.posX = 0;
 95
 96                if (this.posY == this.height - 1) {
 97                    this.shift();
 98                } else {
 99                    this.posY++;
100                }
101
102                return;
103            }
104
105            // Clear any overflow value.
106            this.overflow = false;
107
108            // Commit new character to buffer.
109            this.bufferBg[this.posY][this.posX] = this.bg;
110            this.bufferFg[this.posY][this.posX] = this.fg;
111            this.buffer[this.posY][this.posX++] = c;
112
113            // If we're at the end of the line, shift downwards or move to new line.
114            if (this.posX == this.width) {
115                this.posX = 0;
116
117                if (this.posY == this.height - 1) {
118                    this.shift();
119                } else {
120                    this.posY++;
121                }
122
123                // Set a flag to say we just naturally overflowed and to ignore
```

```
124             // the next newline character that may appear.
125             this.overflow = true;
126         }
127     }
128 
129     /**
130      * Write a string value to the text buffer.
131      * @param value String value to write
132      */
133     public void write(String value) {
134         // Write each character sequentially.
135         for (int i=0;i<value.length();i++) {
136             char c = value.charAt(i);
137 
138             // If we encounter an Ansi escape character, then take the
139             // substring from this point on and determine if it is a valid
140             // escape code. If it is, apply any changes before continuing.
141             if (c == '\u001B') {
142                 Matcher matcher = Ansi.AnsiPattern.matcher(value.substring(i));
143                 if (matcher.find()) {
144                     int v = Integer.parseInt(matcher.group(1));
145                     i += 3 + (v > 9 ? 1 : 0);
146 
147                     if (v == 0) {
148                         this.bg = Color.BLACK;
149                         this.fg = Color.WHITE;
150                     } else if (v >= 30 && v < 38) {
151                         this.fg = Ansi.fromEscapeCode(v);
152                     } else if (v >= 40 && v < 48) {
153                         this.bg = Ansi.fromEscapeCode(v);
154                     }
155 
156                     continue;
157                 }
158             }
159 
160             this.write(c);
161         }
162     }
163 
164     /**
```

```java
     * Get a character at a certain position
     * @param x X position
     * @param y Y position
     * @return Character at given position
     */
    public char getChar(int x, int y) {
        return this.buffer[y][x];
    }

    /**
     * Get the background colour at a certain position
     * @param x X position
     * @param y Y position
     * @return Background colour at given position
     */
    public Color getBg(int x, int y) {
        return this.bufferBg[y][x];
    }

    /**
     * Get the foreground colour at a certain position
     * @param x X position
     * @param y Y position
     * @return Foreground colour at given position
     */
    public Color getFg(int x, int y) {
        return this.bufferFg[y][x];
    }

    /**
     * Get the width of this buffer.
     * @return Width
     */
    public int getWidth() {
        return this.width;
    }

    /**
     * Get the height of this buffer.
     * @return Height
     */
```

```java
    public int getHeight() {
        return this.height;
    }

    /**
     * Get the current X position of the cursor.
     * @return X position of cursor
     */
    public int getPosX() {
        return this.posX;
    }

    /**
     * Get the current Y position of the cursor.
     * @return Y position of cursor
     */
    public int getPosY() {
        return this.posY;
    }

    /**
     * Clear the buffer
     */
    public void clear() {
        this.buffer = new char[height][width];
        this.bufferBg = new Color[height][width];
        this.bufferFg = new Color[height][width];

        this.posX = 0;
        this.posY = 0;

        this.bg = Color.BLACK;
        this.fg = Color.WHITE;

        this.overflow = false;
    }
}
```

```java
package uk.insrt.coursework.zuul.util;

import java.lang.reflect.Field;
import java.net.URLClassLoader;
import java.util.Vector;

/**
 * Utilities for detecting we are running in BlueJ.
 *
 * @author Paul Makles <https://insrt.uk>
 * @version 2.0
 */
public class BlueJ {
    /**
     * Whether to ignore deprecation warnings.
     * Enable to allow isRunningInBlueJ() to confidently determine status.
     */
    private static boolean liveOnTheEdge = false;

    /**
     * Check whether this is being exported as BlueJ using maven-bluej
     * https://github.com/KCLOSS/maven-bluej
     * @return Whether this was exported as a BlueJ project.
     */
    public static boolean isExportedAsBlueJ() {
        return BlueJ.class.getResource("/ThisIsABlueJProject") != null;
    }

    /**
     * Detect whether we are currently running under BlueJ.
     * @return Whether we are running from BlueJ.
     */
    public static boolean isRunningInBlueJ() {
        ClassLoader classLoader = Thread.currentThread().getContextClassLoader();

        // When we load the project typically, i.e. from a JAR file, it is instead
        // loaded by jdk.internal.loader.ClassLoaders$PlatformClassLoader and then
        // $AppClassLoader, which we should also see further up the chain from the
        // java.net.URLClassLoader loader.
        if (classLoader instanceof URLClassLoader) {
            if (getJavaVersion() > 8 && !liveOnTheEdge) {
```

```
42                  // Using setAccessible() as below is deprecated in Java 9 onwards,
43                  // so to avoid any errors in stderr, we can take a safe bet and
44                  // assume that we are in BlueJ given the way we are being loaded.
45                  return true;
46              }
47
48              // We can verify we are running under BlueJ by looping through all
49              // classes which exist on the parent class loader and to check if
50              // a BlueJ class is present.
51              try {
52                  // Finding classes loaded by ClassLoader.
53                  // https://stackoverflow.com/a/10261850
54                  Field f = ClassLoader.class.getDeclaredField("classes");
55                  f.setAccessible(true);
56
57                  @SuppressWarnings("unchecked")
58                  Vector<Class<?>> classes = (Vector<Class<?>>) f.get(classLoader.getParent());
59
60                  for (Class<?> cls : classes) {
61                      if (cls.getName().startsWith("bluej.runtime")) {
62                          return true;
63                      }
64                  }
65              } catch(NoSuchFieldException | IllegalAccessException | ClassCastException e) {}
66          }
67
68          return false;
69      }
70
71      /**
72       * Gets the current Java version as a single integer.
73       * Taken from https://stackoverflow.com/a/2591122
74       * @return Current Java major version number.
75       */
76      private static int getJavaVersion() {
77          String version = System.getProperty("java.version");
78          return Integer.parseInt(
79              version.startsWith("1.")
80                  ? version.substring(2, 3)
81                  : (
82                      version.indexOf(".") != -1
```

```
83                          ? version.substring(0, version.indexOf("."))
84                          : version
85                  )
86          );
87      }
88  }
```

```java
package uk.insrt.coursework.zuul.util;

import java.io.IOException;
import java.io.InputStream;
import java.util.Arrays;
import java.util.HashMap;
import java.util.Map;
import java.util.stream.Collectors;

import com.moandjiezana.toml.Toml;

/**
 * This class provides localisation capabilities for the game.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Localisation {
    private Map<String, Object> map;

    /**
     * Construct a new instance of Localisation.
     */
    public Localisation() {
        this.map = new HashMap<>();
    }

    /**
     * Load a certain locale by name.
     * @param locale The target locale to load
     * @throws IOException if the locale does not exist in resources
     */
    public void loadLocale(String locale) throws IOException {
        InputStream stream = this
            .getClass()
            .getResourceAsStream("/locale/" + locale + ".toml");

        this.map = new Toml().read(stream).toMap();
    }

    /**
```

```java
     * Given a path of keys, find the value at the end of the path.
     *
     * Unchecked errors are supressed as they would only occur if the
     * developer provides an incorrect data structure, in that case the
     * error will be emitted from within this method. It is not a critical
     * error but it should be handled immediately.
     * @param path Path to value we want
     * @return The value at the given path
     */
    @SuppressWarnings("unchecked")
    public String from(String... path) {
        if (path.length == 0) return "<empty string>";

        try {
            var index = 1;
            var node = this.map.get(path[0]);
            while (index != path.length) {
                Map<String, Object> map = (Map<String, Object>) node;
                node = map.get(path[index++]);
            }

            if (node != null) {
                return (String) node;
            }
        } catch (Exception e) {
            // We don't want this to be a fatal error,
            // we instead return the original template.
        }

        return "<" + Arrays.asList(path).stream().collect(Collectors.joining(".")) + ">";
    }

    /**
     * Given a path, find the value at the end of the path.
     * @param path Path to value we want, keys separated by period
     * @return The value at the given path
     */
    public String get(String path) {
        return this.from(path.split("\\."));
    }
}
```

```java
package uk.insrt.coursework.zuul.util;

import java.util.Arrays;

import uk.insrt.coursework.zuul.entities.Entity;

/**
 * Utilities for searching through data structures related to the game
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Search {
    /**
     * Find an Entity within an Iterable of entities given certain parameters.
     * @param entities Iterable of Entities which we search through
     * @param name Query which we are matching for
     * @param fuzzy Whether to match whether the alias contains this name in contrast to just doing exact matching
     * @return The Entity if it is found or null
     */
    public static Entity findEntity(Iterable<Entity> entities, String name, boolean fuzzy) {
        String normalised = name.toLowerCase();
        for (Entity entity : entities) {
            String[] aliases = entity.getAliases();
            for (String alias : aliases) {
                if (fuzzy) {
                    if (Arrays.asList(normalised.split("\\s"))
                            .contains(alias)) {
                        return entity;
                    }
                } else if (normalised.equals(alias)) {
                    return entity;
                }
            }
        }

        return null;
    }
}
```

```java
package uk.insrt.coursework.zuul.util;

import java.util.HashMap;
import java.util.List;

/**
 * This is an implementation of a Tree-like data structure.
 * Each node has a one or more children identified by a key
 * and each node can have more children or have a value.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Tree<K, V> {
    private HashMap<K, Tree<K, V>> children = new HashMap<>();
    private Tree<K, V> parent;
    private V value;

    /**
     * Construct a new Empty Tree node.
     */
    public Tree() {}

    /**
     * Construct a new Tree node with a parent only.
     * @param parent Tree node which owns this node
     */
    public Tree(Tree<K, V> parent) {
        this.parent = parent;
    }

    /**
     * Construct a new Tree node with parent and value.
     * @param parent Tree node which owns this node
     * @param value The value this node should hold
     */
    public Tree(Tree<K, V> parent, V value) {
        this.parent = parent;
        this.value = value;
    }

```

```java
    /**
     * Get a child of this Tree node with a given key K.
     * @param key Given key
     * @return The child represented by this key if it exists, otherwise returns null
     */
    public Tree<K, V> getChild(K key) {
        return this.children.get(key);
    }

    /**
     * Private method used to accumulate the edges travelled up to the root node.
     * @param acc Accumulator value
     * @return The current accumulator value
     */
    private int getHeight(int acc) {
        if (this.parent == null) return acc;
        return this.parent.getHeight(++acc);
    }

    /**
     * The height of the Tree from this point.
     * This is the number of edges to get from this node to the root node.
     * @return The height of the tree
     */
    public int getHeight() {
        return this.getHeight(0);
    }

    /**
     * Whether this Tree node has a value.
     * @return True if this node has a value
     */
    public boolean hasValue() {
        return this.value != null;
    }

    /**
     * Get the value of this Tree node.
     * @return Value stored if there is one, otherwise null.
     */
    public V getValue() {
```

```java
        return this.value;
    }

    /**
     * Add a child to this Tree node represented by a key K.
     * @param key Key to represent this new child
     * @param node Child node to add
     */
    public void addChild(K key, Tree<K, V> node) {
        this.children.put(key, node);
    }

    /**
     * Recurse through a given key path and add value as a node at the bottom of the path.
     * @param keys Keys to iterate through
     * @param value Value to add at the end of the path
     */
    public void addChildWithPath(List<K> keys, V value) {
        Tree<K, V> node = this;
        while (keys.size() > 0) {
            K key = keys.remove(0);
            Tree<K, V> child = node.getChild(key);
            if (child == null) {
                child = new Tree<>(node, keys.size() == 0 ? value : null);
                node.addChild(key, child);
            }

            node = child;
        }
    }
}
```

```java
package uk.insrt.coursework.zuul.world;

import java.util.Arrays;
import java.util.List;

/**
 * Enum which represents a Cardinal direction.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public enum Direction {
    NORTH(new String[] { "N" }),
    NORTH_EAST(new String[] { "NE", "NORTH EAST" }),
    EAST(new String[] { "E" }),
    SOUTH_EAST(new String[] { "SE", "SOUTH EAST" }),
    SOUTH(new String[] { "S" }),
    SOUTH_WEST(new String[] { "SW", "SOUTH WEST" }),
    WEST(new String[] { "W" }),
    NORTH_WEST(new String[] { "NW", "NORTH WEST" }),

    UP(new String[] {}),
    DOWN(new String[] {});

    private List<String> aliases;

    /**
     * Consturct a new Direction
     * @param aliases Alternative ways to refer to this Direction
     */
    private Direction(String[] aliases) {
        this.aliases = Arrays.asList(aliases);
    }

    /**
     * Check whether this Direction matches the given aliases.
     * @param direction Direction in String format
     * @return Whether it matches.
     */
    private boolean matches(String direction) {
        return this.aliases.contains(direction);
```

```
42      }
43
44      /**
45       * Flip a given Direction in the opposite direction.
46       * @return Direction in the opposite direction.
47       */
48      public Direction flip() {
49          switch (this) {
50              default:
51              case NORTH: return Direction.SOUTH;
52              case NORTH_EAST: return Direction.SOUTH_WEST;
53              case EAST: return Direction.WEST;
54              case SOUTH_EAST: return Direction.NORTH_WEST;
55              case SOUTH: return Direction.NORTH;
56              case SOUTH_WEST: return Direction.NORTH_EAST;
57              case WEST: return Direction.EAST;
58              case NORTH_WEST: return Direction.SOUTH_EAST;
59              case UP: return Direction.DOWN;
60              case DOWN: return Direction.UP;
61          }
62      }
63
64      /**
65       * Convert an arbitrary String to a Direction.
66       * @param direction Raw string representing a Direction
67       * @return Direction or null from given string
68       */
69      public static Direction fromString(String direction) {
70          if (direction == null) return null;
71
72          String directionFormatted = direction.toUpperCase();
73          try {
74              return Direction.valueOf(directionFormatted);
75          } catch (Exception ex) {
76              for (Direction dir : Direction.values()) {
77                  if (dir.matches(directionFormatted)) {
78                      return dir;
79                  }
80              }
81
82              return null;
```

```
83              }
84          }
85      }
```

```java
package uk.insrt.coursework.zuul.world;

import uk.insrt.coursework.zuul.entities.Inventory;

/**
 * Representation of a physical location in the world,
 * whether it is a room or inventory or neither but not both.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class Location {
    private Room room;
    private Inventory inventory;

    /**
     * Construct a new Location outside of the World.
     */
    public Location() {}

    /**
     * Construct a new Location pointing to a Room.
     * @param room Room
     */
    public Location(Room room) {
        this.room = room;
    }

    /**
     * Construct a new Location pointing to an Inventory.
     * @param inventory Inventory
     */
    public Location(Inventory inventory) {
        this.inventory = inventory;
    }

    /**
     * Change this Location to point to a Room.
     * @param room Room
     */
    public void setLocation(Room room) {
```

```
42          this.room = room;
43          this.inventory = null;
44      }
45
46      /**
47       * Change this Location to point to an Inventory.
48       * @param inventory Inventory
49       */
50      public void setLocation(Inventory inventory) {
51          this.room = null;
52          this.inventory = inventory;
53      }
54
55      /**
56       * Reset the Location and put us outside of the World.
57       */
58      public void clear() {
59          this.room = null;
60          this.inventory = null;
61      }
62
63      /**
64       * Get the current Room this Location represents.
65       * @return Room or null if in an inventory or out of the World.
66       */
67      public Room getRoom() {
68          return this.room;
69      }
70
71      /**
72       * Get the current Inventory this Location represents.
73       * @return Inventory or null if in a room or out of this World.
74       */
75      public Inventory getInventory() {
76          return this.inventory;
77      }
78  }
```

```java
package uk.insrt.coursework.zuul.world;

import java.util.HashMap;
import java.util.Set;

/**
 * Representation of a Room within the World.
 * Handles how entities can move from this to other Rooms.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public abstract class Room {
    private World world;
    private String name;
    private HashMap<Direction, Room> adjacentRooms;

    /**
     * Construct a new Room in a given World with a given name.
     * @param world World
     * @param name Internal name used to refer to this Room
     */
    public Room(World world, String name) {
        this.world = world;
        this.name = name;
        this.adjacentRooms = new HashMap<>();
    }

    /**
     * Get the World that this Room is in.
     * @return World
     */
    public World getWorld() {
        return this.world;
    }

    /**
     * Get the internal name of this Room.
     * @return Internal name
     */
    public String getName() {
```

```
42          return this.name;
43      }
44
45      /**
46       * Make another Room adjacent to this Room in a particular Direction.
47       * @param direction Direction the other Room is in
48       * @param room Room which we are making adjacent
49       */
50      public void setAdjacent(Direction direction, Room room) {
51          if (room == null) System.err.println("Warning: assigned null Room to direction " + direction + " for the Room " +
this.name);
52          this.adjacentRooms.put(direction, room);
53      }
54
55      /**
56       * Get an adjacent Room in a particular Direction.
57       * @param direction Direction to look at
58       * @return The Room if one is present in that Direction, otherwise null
59       */
60      public Room getAdjacent(Direction direction) {
61          return this.adjacentRooms.get(direction);
62      }
63
64      /**
65       * Whether the player can leave in any particular direction.
66       * Should print reason if not.
67       * @param direction Direction which we are checking
68       * @return Whether the player can leave
69       */
70      public boolean canLeave(Direction direction) {
71          return true;
72      }
73
74      /**
75       * Get Directions that you can leave this Room in.
76       * @return Set of Directions we can leave in
77       */
78      public Set<Direction> getDirections() {
79          return this.adjacentRooms.keySet();
80      }
81
```

```java
    /**
     * Check whether there is an exit in a particular Direction.
     * @param direction Direction to check
     * @return True if there is an exit in a given Direction
     */
    public boolean hasExit(Direction direction) {
        return this.adjacentRooms.containsKey(direction);
    }

    /**
     * Reset adjacent Rooms and reconfigure adjacent Rooms.
     * This should be called after all Rooms have been spawned into the World.
     */
    public void linkRooms() {
        this.adjacentRooms.clear();
        this.setupDirections();
    }

    /**
     * Spawn Entities in this World.
     * By default, nothing is done but this should be used further up to spawn
     * the Entities for this particular Room.
     */
    public void spawnEntities() {}

    /**
     * Convert this Room into a Location.
     * @return Location representation of Room
     */
    public Location toLocation() {
        return new Location(this);
    }

    /**
     * Describe what this Room looks like.
     * @return Description of this Room
     */
    public abstract String describe();

    /**
     * Setup adjacent Rooms.
```

```
123         */
124        protected abstract void setupDirections();
125    }
```

```java
package uk.insrt.coursework.zuul.world;

import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;

import uk.insrt.coursework.zuul.entities.Entity;
import uk.insrt.coursework.zuul.entities.EntityPlayer;
import uk.insrt.coursework.zuul.events.Event;
import uk.insrt.coursework.zuul.events.EventSystem;
import uk.insrt.coursework.zuul.io.IOSystem;

/**
 * Representation of the game World.
 * Contains all the Rooms and Entities as well as the Player.
 * Has its own Event system for signaling when things should happen.
 * Also has access to the IO system which is provided to all Rooms and Entities.
 *
 * @author Pawel Makles (K21002534)
 * @version 1.0-SNAPSHOT
 */
public class World {
    protected Map<String, Room> rooms = new HashMap<>();
    protected Map<String, Entity> entities = new HashMap<>();
    protected EntityPlayer player;

    protected IOSystem io;
    protected EventSystem eventSystem;

    /**
     * Consturct a new game World with a given IO system.
     * @param io IO system to provide to everything
     */
    public World(IOSystem io) {
        this.io = io;
        this.eventSystem = new EventSystem();
        this.player = new EntityPlayer(this);
        this.entities.put("player", this.player);
    }

```

```java
    /**
     * Find an Entity by its ID.
     * @param id Entity ID
     * @return Entity if it exists, otherwise null.
     */
    public Entity getEntity(String id) {
        return this.entities.get(id);
    }

    /**
     * Find an Room by its ID.
     * @param room Room ID
     * @return Room if it exists, otherwise null.
     */
    public Room getRoom(String room) {
        return this.rooms.get(room);
    }

    /**
     * Get the Player entity.
     * @return The player entity
     */
    public EntityPlayer getPlayer() {
        return this.player;
    }

    /**
     * Get the IO system provided to this World.
     * @return IO system
     */
    public IOSystem getIO() {
        return this.io;
    }

    /**
     * Get this World's event system.
     * @return World event system
     */
    public EventSystem getEventSystem() {
        return this.eventSystem;
    }
```

```java

    /**
     * Add a Room to this World.
     * @param room Room to add
     */
    protected void addRoom(Room room) {
        this.rooms.put(room.getName(), room);
    }

    /**
     * Spawn a new Entity in the World.
     * @param id Unique Entity ID
     * @param entity Entity to spawn
     */
    public void spawnEntity(String id, Entity entity) {
        this.entities.put(id, entity);
    }

    /**
     * Get all the Entities found in a given Room.
     * @param room Room to search for
     * @return List of Entities in the World in a given Room
     */
    public List<Entity> getEntitiesInRoom(Room room) {
        return this
            .entities
            .values()
            .stream()
            .filter(e -> e.getRoom() == room)
            .collect(Collectors.toList());
    }

    protected void linkRooms() {
        for (Room room : this.rooms.values()) {
            room.linkRooms();
        }
    }

    public void emit(Event event) {
        this.eventSystem.emit(event);
    }
```

```
124
125     /**
126      * Try to spawn the player in the first available room.
127      */
128     public void spawnPlayer() {
129         this.player.setLocation(this.rooms.values().iterator().next());
130     }
131 }
```

```toml
emojis = [
    { unicode = "\u1F633", path = "flosh.png" }, # 😳
    { unicode = "\u1F601", path = "trol.png" }, # 😁
    { unicode = "\u1F438", path = "monkaStare.png" }, # 🐸
    { unicode = "\u1F642", path = "pauseChamp.png" }, # 🙂
    { unicode = "\u1F610", path = "weirdChamp.png" }, # 😐
    { unicode = "\u1F604", path = "peepoHappy.png" }, # 😄
    { unicode = "\u1F508", path = "sound.png" }, # 🔈
    { unicode = "\u1F99D", path = "they is stuck.jpg", width = 24, height = 8 }, # 🦝
    { unicode = ":uk:", path = "uk.png", width = 3, height = 1 }, # 🇬🇧
    { unicode = ":us:", path = "us.png", width = 3, height = 1 }, # 🇺🇸
    { unicode = ":de:", path = "de.png", width = 3, height = 1 }, # 🇩🇪
    { unicode = ":cz:", path = "cz.png", width = 3, height = 1 }, # 🇨🇿

    # Example definitions:
    #{ unicode = "[wideChamp]", path = "weirdChamp.png" },
    #{ unicode = "[widePeepo]", path = "peepoHappy.png" },
    #{ unicode = "😄", path = "peepoHappy.png", width = 8, height = 4 },
]
```

```toml
# Translations for World of Deez
# Provided by FatalErrorCoded

[global]
sight = "Vidíš:"

    [global.can_go_in_x_directions]
    1 = "Můžeš jít do"
    2 = "směrů"

[selectors]
direction = "<směr>"
something = "<něco>"
someone = "<někdo>"
item = "<věc>"

    [selectors.cant_find]
    1 = "Podíváš se kolem po"
    2 = "ale nemůžeš nic najít"

[commands]
unknown = "Nejsem si jistý o co se snažíš."

back = "vrátit se do předchozí místnosti"
quit = "opustit hru"
where_am_i = "znova popsat tuto místnost"

    [commands.bag]
    usage = "podíváš se do svého batohu nebo na inventář nečeho"
    cant_find = "Nemůžu najít na co se snažíš podívat."
    empty = "Tvoje taška je prázdná!"
    entity_empty = "nevypadá, že něco má"
    can_carry_kg = "Můžeš unést"
    are_carrying_kg = "Nosíš"
    look_in_bag = "Podíváš se do batohu a vidíš"
    entity_appears_to_have = "vypadá, že má"

    [commands.drop]
    usage = "vyhodit předmět z batohu"
    nothing_specified = "Co chceš vyhodit?"

```

```toml
        [commands.drop.dropped]
        1 = "Vyhodíš"
        2 = "z batohu"

    [commands.give]
    usage = "dát něco někomu"
    nothing_specified = "Co si přeješ dát?"
    no_target = "Komu / do čeho tohle dáváš?"
    denied_player = "Sebe nemůžeš dát nikomu ani ničemu. \u1F438"

        [commands.give.denied]
        1 = "Nemůžeš dát"
        2 = ""

    [commands.go]
    usage = "jít určitým směrem"
    nothing_specified = "Kam jdeš?"

    [commands.help]
    usage = "zobrazit help menu"
    can_run = "Můžeš spustit následující příkazy:"

    [commands.pet]
    usage = "pohlaď něco okolo sebe nebo ve svém inventáři"
    nothing_specified = "Co se snažíš pohladit?"
    denied = "Nemůžeš pohladit"

    [commands.take]
    usage = "dát něco do tvé tašky"
    nothing_specified = "Od koho?"
    entity_does_not_have_entity = "nemá"
    item_not_specified = "Co si chceš vzít?"

        [commands.take.took]
        1 = "Bereš si"
        2 = "od"
        3 = "a dáváš si ho do tašky"

        [commands.take.denied]
        1 = "Nemůžeš si vzít"
        2 = "je moc těžký pro tvůj batoh"
```

```
    [commands.talk]
    usage = "začít s něčím mluvit"
    nothing_specified = "S čím chceš mluvit?"
    denied = "Nemůžeš mluvit s"

    [commands.use]
    usage = "použít něco kolem tebe nebo v tvé tašce"
    nothing_specified = "Co chceš použít?"
    denied = "Nemůžeš použít"

    [commands.map]
    usage = "ukázat světovou mapu"
    close = "Stiskni Enter pro zavření."

        [commands.map.discovered]
        1 = "Objevil si"
        2 = "světa"

    [commands.win]
    usage = "vyhrát hru"
    conclusion = """...

Kapitola 4.:
Soubory jsou vypuštěny do světa pro každého ke stažení, lidi rychle rozebírají
každý malý detail, pár bezprostředních detailů vyjde na světlo:
- Sylvasta zkoumala eticky sporné vědecké oblasti, zejména denně prováděla
  stovky testů na několik různých beastmanů jako testovacích subjektů.
  Nikdo ale nedokázal dokázat, že tam někdo byl proti jejich vůli.
- Nicméně, na tento výzkum se zároveň okamžitě podívali i cizí moci, které
  velmi rychle zjistili, že beastmanní socializace žijící vě městě je mnohem
  větší hrozba než původně očekávali.
- Město společně s veřejnou radou Sylvasta okamžitě nařídila evakuaci všech
  občanů do jakékoliv oblasti, kterou mohli najít, pod obavama potenciálně
  smrtelného konfliktu. Město se v přístích dnech stalo terčem palby.
"""
    stats = "\u001B[47m\u001B[30mVaše konečné statistiky\u001B[0m"
    total_ticks = "Herních ticků celkem: "
    total_time = "Celkový čas: "
    minutes = "minut"
    seconds = "sekund"
```

```toml
    sidequests_complete = "Postranních misí dokončeno: "
    press_enter_key = "Stiskněte Enter pro uzavření hry."

[entities]
boat_key = "Klíč k rychlostnímu člunu"

    [entities.bed]
    description = "Postel"
    use = "Dáš si šlofíka."

    [entities.boat]
    description = "Rychlostní člun je zakotvená na pobřeží"
    locked = "Člun je zamčen."
    locked_for_sale = "Člun je zamčen.\nJe na něm poznámka, která říká, že máte kontaktovat obchodníka ohledně prodeje."
    denied = "Nesmíš u sebe nic mít, aby si mohl člun použít.\nMůžeš ale dávat věci do člunu."
    travel = "Skočíš do člunu a přecestuješ na druhou stranu..."
    too_heavy = "Člun už toho v sobě má až moc!"

        [entities.boat.give]
        1 = "Dát"
        2 = "do člunu"

    [entities.cat]
    description = "Toulavá černá kočka"
    pet = "Pohladíš kočku."
    use = "Nemůžes kočku.\nProsím ne kočku. \u1F633\u1F633\u1F633"
    enter = "Kočka přirazila mezi nás."
    leave = "Vidíš kočku jak odejde."

    [entities.comms]
    description = "Komunikační zařízení"
    off = "Zařízení je vypnuté."

    [entities.couch]
    description = "Hnědý kožený gauč"
    sitting = "Už sedíš na gauči."
    sit = "Sedl sis na gauč."

    [entities.laptop]
    description = "Laptop"

```

```
        [entities.laptop.boot]
        dialog = "Zapneš počítač..."
        option_1 = "[počkat]"

        [entities.laptop.home]
        dialog = "Select an option:"
        option_q = "Vypnout"
        option_1 = "/Moje Obrázky"
        option_2 = "/Vtipné kočičí videa"
        option_3 = "/Mariin skener dokumentů"

        [entities.laptop.pictures]
        dialog = """Je jenom jedna fotka v tvých obrázkách:

\u1F99D









"""
        option_q = "Vrátit se."

        [entities.laptop.cat_videos]
        dialog = "Podíváš se na vtipné kočičí videa..."
        option_q = "Super."

        [entities.laptop.document]
        dialog = """\u001B[35mMariin\u001B[0m skener dokumentů"""
        option_q = "Vrátit se"
        option_1 = "Poslat dokumenty"

[home]
first_load = '''
Zrovna vcházíš do světa.
Pokud se v jakémkoliv bodě zasekneš,
můžeš použít `help` pro zobrazení všech příkazů.

---
```

```
206
207 Probudíš se ke zvuku lidí protestujících venku..
208 Měl si si přes noc zavřít okno..
209
210 Zaujatý, podíváš se z okna, aby si viděl, co se děje.. venku je skupina
211 protestantů před Zdravotním střediskem. Nemůžeš úplně přijít na to,
212 co říkají, nebo co je napsáno na jejich cedulích.
213
214 Ikdyž by nebylo překvapující kdyby se tam dělo něco divného, nemůžeš na to
215 nějak přiložit prst. Možná je něco ve zprávách..
216 '''
217
218 enter = '''
219 Vstoupíš do svého bytu.
220 '''
221
222     [home.tv]
223     description = "LG 55NANO966PA 55\" Super UHD 8K HDR Smart LED TV"
224     off = "Vypnout televizi."
225     keep_watching = "Dále sledovat..."
226
227         # Red: \u001B[31m
228         # Green: \u001B[32m
229         # Yellow: \u001B[33m
230         # Cyan: \u001B[36m
231         [home.tv.first_on]
232         dialog = "Zapneš televizi.\n\nNaskočí zpravodajský kanál..."
233         dialog_a="""\u001B[31mZprávy\u001B[0m: Občanské nepokoje stoupají, Sylvasta čelí konfliktům ze všech
234 stran, a mnoho lidí se cítí neklidní o jejich budoucnosti mezitím co stoupající
235 napětí mezi lidským a beastman socializacemi způsobuje spory okolo
236 městké hranice.
237
238 \u001B[31mZprávy\u001B[0m: Přinášíme vám scény před Zdravotním střediskem kde
239 se ukázala skupina protestantů v opozici proti výzkumu prováděnému v
240 Sylvasta.
241
242 \u001B[36mKorespondent\u001B[0m: Jsem tady, stojím venku se skupinou protestantů..
243
244 \u001B[36mKorespondent\u001B[0m na \u001B[33mProtestanta\u001B[0m: Co vás sem dnes přináší?
245 """
246
```

```
247         dialog_b="""\u001B[33mProtestant\u001B[0m:
248 Protestant: Berou peníze města a používají je pro svoje vlastní dobro,
249 neměli by dostávat žádné financování, natož mít dovoleno zde operovat.
250 
251 \u001B[31mZprávy\u001B[0m: Odvážné tvrzení přicházející přímo před Sylvasta, jestli jsou tyto
252 tvrzení na nečem založené, to ještě nikdo neví, viděli jsme měsíce a měsíce
253 úniků od dřívějších zaměstnanců a různých vnitřních nehod ale ještě nevíme
254 skutečné úmysly lidí pracující pro Sylvasta.
255 
256 \u001B[31mZprávy\u001B[0m: Sylvasta oznámila, že odmíta komunikovat
257 nebo dále oznámit jejich vnitřní výzkum, citující veřejnou bezpečnost.
258 Co to znamená, nikdo kromě jejich vnitřního personálu neví.
259 """
260 
261 dialog_c="""\u001B[31mZprávy\u001B[0m: Také to pokládá otázku jestli jsou protesty neopodstatněné,
262 pouze se snažíci vyvolat nesnáz ve městě. Dnes dříve jsme mluvili s několika místními
263 obyvateli žijící ve středu města..
264 
265 \u001B[32mObchodník\u001B[0m: Myslím si, že se nás jenom snaží vyprovokovat. Sylvasta měla
266 důležitou roli v založení tohoto města a umožnila nám žít v míru bez obav toho,
267 že by na nás něco zaútočilo, nemyslím si, že je dostatečný důvod protestovat.
268 
269 \u001B[32mObchodník\u001B[0m: Už vidíme, jak se svět otáči proti městu, a tyto typy
270 vnitřních sporů jim jenom dává důvod vstoupit a ujmout se řízení.
271 
272 \u001B[31mZprávy\u001B[0m: Toto zakončuje náš ranní pořad, budeme zpátky
273 pro zpravodajskou hodinu v jednu odpoledne.
274 """
275 
276 [apartments]
277 enter = '''
278 Vstoupíš do recepce bytového komplexu.
279 '''
280 
281     # Cyan: \u001B[36m
282     [apartments.receptionist]
283     description = "Recepční sedící za stolem"
284 
285         [apartments.receptionist.first_encounter]
286         dialog = "\u001B[36mRecepční\u001B[0m: Dobré ráno, jak se dneska máte?"
287         option_1 = "Co se to venku děje?"
```

```
288         option_q = "Ale nic."
289
290         [apartments.receptionist.protestors]
291         dialog = """\u001B[36mRecepční\u001B[0m: Vůbec nevím ale vypadá to jako skupina lidí řvající
292 před Zdravodním střediskem..."""
293         option_1 = "Nevíte něco víc?"
294         option_q = "Dobře."
295
296         [apartments.receptionist.protestors2]
297         dialog = """\u001B[36mRecepční\u001B[0m: Rozdávali letáky když přišli, možná někdo
298 poblíž bude vědět..."""
299         option_q = "Díky."
300
301         [apartments.receptionist.repeated]
302         dialog = "\u001B[36mRecepční\u001B[0m: Dobré ráno, s čím vám můžu pomoci?"
303         option_1 = "Co se to venku s těmi protestanty děje?"
304         option_q = "To je vše, díky."
305
306 [city_centre]
307 first_load = '''
308 Přijdeš na náměstí. Ráno bývá docela rušno. Kolem pobýhá hodně lidí,
309 docela nepravděpodobné, že by si si s někým mohl popovídat.
310
311 V oblasti je všeobecná nesnáz, někteří jsou docela nervózní, ostatní vypadají,
312 jako kdyby byl poslední den na zemi...
313 '''
314
315 enter = '''
316 Přijdeš na náměstí.
317 '''
318
319     # Yellow: \u001B[33m
320     [city_centre.npc]
321     description = "Člověk sedící na lavičce"
322
323         [city_centre.npc.small_talk]
324         dialog = "\u001B[33mCizinec\u001B[0m: Dobrý den, mohl bych vám pomoci?"
325         option_1 = "Tuším, že si tady neviděl projít protestanty?"
326         option_2 = "Co to čteš?"
327         option_q = "Ale nic."
328
```

```
329         [city_centre.npc.protestors]
330         dialog = """\u001B[33mCizinec\u001B[0m: Ale jo, byli docela hlasití..
331  Byli pěkně otravní, pěkně mi vyrušili ráno.
332  ...
333  Nechali mi tady ale tenhle leták"""
334         option_1 = "Můžu se podívat?"
335
336         [city_centre.npc.enquire]
337         dialog = "\u001B[33mCizinec\u001B[0m: Támhleti kolem mě prošli a nechali mi tady leták."
338
339         [city_centre.npc.leaflet]
340         # dialog = "\u001B[33mStranger\u001B[0m: Sure, here you go, keep it.\n[\u001B[33mStranger\u001B[0m gave you an
item.]"
341         dialog = """\u001B[33mCizinec\u001B[0m: Jistě, tady ho máš...
342
343  Podíváš se na leták:
344
345  \u001B[41mNEETICKÝ VÝZKUM
346  \u001B[40m\u001B[31mNaše město je v nebezpečí, Sylvasta
347  dále financuje jejich výzkum ale nic nedělá,
348  aby pomohla městu přežít.
349
350  Sylvasta zneužívá jejich pozici aby
351  pokračovali v práci kompletně NELEGÁLNÍHO
352  bioinženýrství beastmenů.\u001B[0m
353
354  Podpořte nás: \u001B[36mhttps://sylvasta.vercel.app\u001B[0m
355  """
356         option_1 = "Díky."
357
358  [street]
359  first_load = '''
360  Přijdeš do hlavní městské ulice propojující hlavní části města.
361
362  Je tu hodně hluku a chaosu, na západ vidíš protestanty před
363  komplexem zdravotního střediska, drží cedule, "NEETICKÝ VÝZKUM",
364  "NAŠE VZTAHY V KRIZI", "NEPŘEDVÍDATELNÉ NÁSLEDKY".
365  '''
366
367  enter = '''
368  Přijdeš do městské ulice.
```

```
'''

    # Red: \u001B[31m
    [street.protestors]
    description = "Skupina protestantů držící cedule"
    blocking = "Je tu skupina protestantů blokující hlavní dveře."

        [street.protestors.small_talk]
        dialog = "\u001B[31mProtestant\u001B[0m: Slyšeli jste o tom, co Sylvasta dělá, a proč jsme tady venku?"
        option_1 = "Ne, prosím vysvětlete mi."
        option_2 = "Ale nic."

        [street.protestors.enquire]
        dialog = """\u001B[31mProtestant\u001B[0m: Používají městské finance, aby pokračovali v jejich upřímně řečeno
nelegálním výzkumu, viděli jste ty úniky, ne?"""
        option_1 = "Ne."
        option_2 = "Dobře.. Nechám vás v tom."

        [street.protestors.leaks]
        dialog = """\u001B[31mProtestant\u001B[0m: No někteři bývalí zaměstnanci vypustili informace o jejich
aktuálních výzkumech a co z toho můžeme říct je že dělají bioinženýrství na
beastmanech, toto by nemělo být tolerováno natož finanocáno městem."""
        option_1 = "Ok."

        [street.protestors.confrontation]
        dialog = "\u001B[31mProtestant\u001B[0m: Ze mě děláte nějakého konspiračního teoretika nebo co?"
        option_1 = "Ano."
        option_2 = "Odcházím."

[shop]
enter = '''
Vstoupíš do obchodu, za zavírajícíma dveřma zazvoní vstupní zvonek.
'''

    # Green: \u001B[32m
    [shop.npc]
    description = "Obchodník"
    leave = "Ale nic."

    currently_have_amount_of_money = "Zrovna máš:"
    not_enough = "Nemáš dost peněz aby si koupil"
```

```
410      too_heavy = "Je to moc těžké!"
411
412      out_of_stock = "není skladem"
413      x_left = "zbývá"
414
415          [shop.npc.fake_item]
416          cat = "Jedna jediná kočka"
417
418          [shop.npc.item_out_of_stock]
419          1 = "Tato položka"
420          2 = "nemůžeš jí koupit"
421
422          [shop.npc.bought]
423          1 = "Koupil si"
424          2 = "a dal si ji do tašky"
425
426          [shop.npc.greeting]
427          Exposition = "\u001B[32mObchodník\u001B[0m: Dobré ráno, co byste si chtěli koupit?"
428          Recon = "\u001B[32mObchodník\u001B[0m: Dobrý den, něco dneska na mysli?"
429          Stealth = "\u001B[32mObchodník\u001B[0m: Hledáte něco ke koupi?"
430          End = "\u001B[32mObchodník\u001B[0m: Je třeba něco nového?"
431
432  [back_alley]
433  first_load = '''
434  Vstoupíš do temné uličky, je tady docela ticho, velký rozdíl oproti náměstí
435  přímo umístěna skoro přímo uprostřed města. Strany uličky jsou obklopeny
436  zastaralými budovami a temným světlem.
437
438  Rozeznáš povědomou postavu ahead před tebou.
439  '''
440
441  enter = '''
442  Vstoupíš do temné uličky.
443  '''
444
445  [coastline]
446  enter = '''
447  Dorazíš na městskou pobřežní čáru, voda je převážně klidná.
448  '''
449
450  [mainland_coastline]
```

```
enter = '''
Dorazíš na pobřežní čáru na pevnině,
moře boural do kamenů postavených v cestě,
je tady mnohem chladněji, pryč od betonové džungle.
'''

[forest]
enter = '''
Bloudíš lesem, slyšíš ptáky zpívat z vršků stromů...
Přes prostředek proudí řeka, se starým dřevěným mostem nad ní.
Bylo by pěkně jednoduché se tady na pár hodin ztratit.
'''

    # Yellow: \u001B[33m
    [forest.old_man]
    description = "Starý muž procházející se lesem"
    full = "Není nic jiného, co mu můžeš dát."
    accept = "\u001B[33mStarý Muž\u001B[0m: Děkuji, že jste mi ji přinesli."
    deny = "\u001B[33mStarý Muž\u001B[0m: Nechci tvůj"

        [forest.old_man.small_talk]
        dialog = "\u001B[33mStarý Muž\u001B[0m: Neviděl si nikde moji kočku, co?"
        option_1 = "Viděl jsem tuhle černou kočku okolo města..."
        option_q = "Ne."

        [forest.old_man.request]
        dialog = "\u001B[33mStarý Muž\u001B[0m: Mohl by si mi ji přinést?"
        option_1 = "Jistě."
        option_q = "Omlouvám se, teď ne."

        [forest.old_man.praise]
        dialog = "\u001B[33mStarý Muž\u001B[0m: Děkuji vám za pomoc!"
        option_q = "[odejít]"

[worm_hole]
enter = '''
Vstoupíš do červí díry...
'''

[medical_centre]
enter = '''
```

```
Nyní jsi u recepce Zdravotního střediska.
'''

    # Red: \u001B[31m
    [medical_centre.guard]
    description = "Ochranka umístěná u dveří"
    blocking = "U dveří je ochranka hlídající schody, žádný způsob jak se přes ni dostat."

        [medical_centre.guard.small_talk]
        dialog = "\u001B[31mOchranka\u001B[0m: Co chceš?"
        option_q = "Nic..."

[medical_centre_office]
enter = '''
Nacházíš se v kanceláři Zdravotního střediska.
Rozhodně by si tady neměl být...
'''

    [medical_centre_office.books]
        [medical_centre_office.books.1]
        title = "Alan Stern. Hon za Novými Obzory"
        contents = """
14. července , 2015 se stalo něco úžasného. Víc než 3 miliardy mil od
Země, malá NASA loď pojmenovaná Nové Obzory přeletěla kolem Pluta rychlostí
více než 32 000 mil za hodinu, mířící její instrumenty na tajemné ledové
světy Pluto systému, a pak, stejně rychle, pokračovala na své cestě do dále.
"""

        [medical_centre_office.books.2] # yes
        title = "Štos výzkumných papíru"
        contents = """
Po našich nejnovějších testech jsme zjistili že \u001B[47m\u001B[37m[redigováno]\u001B[0m má
značné záporné účinky. Tyto testy byly provedeny napříč 8 dnů a způsobily
\u001B[47m\u001B[37m[redigováno]\u001B[0m a \u001B[47m\u001B[37m[redigováno]\u001B[0m uvnitř těla.

Tento dokument je důvěrný a neměl by být sdílen s třetími stranami.
"""

        [medical_centre_office.books.3]
        title = "Barry A. Burd. Java pro Blbce"
        contents = """
```

```
  Java je platformově–nezávislý, objektově–orientovaný programovací jazyk
  používaný pro vývoj webových a mobilních aplikací. Revizovaná verze obsahuje
  funkce, které mají programátory nadšené, a tento oblíbený průvodce
  se jimi všemi zabývá.

  Naučíte se programovat v Javě jako takto:

  \u001B[36mimport \u001B[33mjava\u001B[37m.\u001B[33mio\u001B[37m.\u001B[33mBufferedReader\u001B[37m;
  \u001B[36mimport \u001B[33mjava\u001B[37m.\u001B[33mio\u001B[37m.\u001B[33mInputStreamReader\u001B[37m;
  \u001B[33mpublic class \u001B[31mMain  \u001B[34m{
      \u001B[33mpublic static  void \u001B[31mmain  \u001B[34m(\u001B[33mString[] args\u001B[34m) \u001B[34m{
          \u001B[33mtry{ String a=new \u001B[31mBufferedReader \u001B[34m(
              \u001B[33mnew \u001B[31mInputStreamReader\u001B[34m( System\u001B[37m.\u001B[33mout\u001B[37m \u001B[34m))
\u001B[37m.
                  \u001B[31mreadLine() \u001B[37m;
          \u001B[33mif
              \u001B[34m(a == \u001B[32m"ahoj"\u001B[34m) {
              System\u001B[37m.\u001B[33mout\u001B[37m.\u001B[31mprintln\u001B[34m(\u001B[32m"Ahoj!!"\u001B[34m)
\u001B[37m;
          \u001B[34m} \u001B[33melse
         \u001B[34m{
              System\u001B[37m.\u001B[33mout\u001B[37m.\u001B[31mprintln\u001B[34m(\u001B[32m"nepozdravil si mě!"\u001B[34
m) \u001B[37m;
          \u001B[34m}
      } \u001B[33mcatch \u001B[34m(\u001B[33mException e\u001B[34m) {\u001B[35m/*něco špatného se stalo! */ \u001B[34m} } }
  \u001B[0m"""

          [medical_centre_office.books.4] # yes
          title = "Dr. Mike Ox. Introducing Impostors in Society"
          contents = """
  [Orazítkováno pouze pro vnitřní využití.]

  Tato kniha podrobně popisuje metody, které můžeme použít k manipulaci obecného
  lidu, budu se podrobně věnovat způsobům, jak můžeme nenápadně odstranit
  nebezpečné nápady a udržet vaše tajemství pod pokličkou.
  """

          [medical_centre_office.books.5] # yes
          title = "RNA sekvence"
          contents = """
  Důvěrný dokument
```

```
571
572 Tvar 1.
573 GAGAATAAACTAGTATTCTTCTGGTCCCCACAGACTCAGAGAGAACCCGCCACCATGTTCGTGTTCCTGGTGCTGCTGCC
574 TCTGGTGTCCAGCCAGTGTGTGAACCTGACCACCAGAACACAGCTGCCTCCAGCCTACACCAACAGCTTTACCAGAGGCG
575 TGTACTACCCCGACAAGGTGTTCAGATCCAGCGTGCTGCACTCTACCCAGGACCTGTTCCTGCCTTTCTTCAGCAACGTG
576 ACCTGGTTCCACGCCATCCACGTGTCCGGCACCAATGGCACCAAGAGATTCGACAACCCCGTGCTGCCCTTCAACGACGG
577 GGTGTACTTTGCCAGCACCGAGAAGTCCAACATCATCAGAGGCTGGATCTTCGGCACCACACTGGACAGCAAGACCCAGA
578 GCCTGCTGATCGTGAACAACGCCACCAACGTGGTCATCAAAGTGTGCGAGTTCCAGTTCTGCAACGACCCCTTCCTGGGC
579 GTCTACTACCACAAGAACAACAAGAGCTGGATGGAAAGCGAGTTCCGGGTGTACAGCAGCGCCAACAACTGCACCTTCGA
580 GTACGTGTCCCAGCCTTTCCTGATGGACCTGGAAGGCAAGCAGGGCAACTTCAAGAACCTGCGCGAGTTCGTGTTTAAGA
581 ACATCGACGGCTACTTCAAGATCTACAGCAAGCACACCCCTATCAACCTCGTGCGGGATCTGCCTCAGGGCTTCTCTGCT
582 CTGGAACCCCTGGTGGATCTGCCCATCGGCATCAACATCACCCGGTTTCAGACACTGCTGGCCCTGCACAGAAGCTACCT
583 GACACCTGGCGATAGCAGCAGCGGATGGACAGCTGGTGCCGCCGCTTACTATGTGGGCTACCTGCAGCCTAGAACCTTCC
584 TGCTGAAGTACAACGAGAACGGCACCATCACCGACGCCGTGGATTGTGCTCTGGATCCTCTGAGCGAGACAAAGTGCACC
585 """
586
587         [medical_centre_office.books.6]
588         title = "Jules Verne. Cesta kolem světa za osmdesát dní"
589         contents = """
590 Jednoho nešťastného večera v Reformním klubu, Phileas Fogg se unáhleně vsadí
591 se svými společníky o £20,000 že dokáže objet celou zemi za pouhých osmdesát
592 dnů - a je rozhodnutý neprohrát. Porušení zaběhnuté rutiny svého každodenního
593 života, rezervovaný Angličan se okamžitě vydává do Doveru, doprovázený svým
594 horkokrevným francouzským sluhou Passepartoutem. Cestování vlakem, parníkem,
595 plachetnicí, saněmi a dokonce i slonem, musí překonat, bouře, únos,
596 přírodní katastrofy, útoky Siouxů and zarputilého inspektora Fixe ze Scotland
597 Yardu, který věří, že Fogg vyloupil Banku Anglie. - aby vyhrál mimořádnou sázku
598 """
599
600 [marie]
601 description = "\u001B[35mMarie, Mink\u001B[0m"
602
603     # Purple: \u001B[35m
604     [marie.alley]
605
606         [marie.alley.small_talk]
607         dialog = "\u001B[35mMarie\u001B[0m: Co tě jsem přináší?"
608         option_1 = "Slyšela si o protestech?"
609         option_2 = "Sylvasta něco chystá..."
610         option_q = "Ale nic, měj hezký den."
611
```

```
612         [marie.alley.protests]
613         dialog = """\u001B[35mMarie\u001B[0m: To jo, ty týpci od Sylvasta rozhodně něco dělají,
614 ale nevím, jestli věřím těm davům, co se prý objevili."""
615         option_1 = "Chceš mi pomoct na to přijít?"
616         option_q = "Ale nic."
617 
618         [marie.alley.sylvasta]
619         dialog = "\u001B[35mMarie\u001B[0m: Myslíš si, že nevím? Chceš mi pomoct na to přijít?"
620         option_1 = "Klidně."
621         option_q = "Ne, díky."
622 
623         [marie.alley.confirm]
624         dialog = "\u001B[35mMarie\u001B[0m: Předpokládám že bychom se mohli vloupat a přijít na pravdu."
625         option_1 = "Pojďme na to! [pokračovat v příběhu]"
626         option_q = "V žádném případě!"
627 
628         [marie.alley.recon]
629         dialog = """\u001B[35mMarie\u001B[0m: Ok, mezitím co přijdu na to jak se tam dostat, potřebuji, aby si našel
630 něco co můžem použít pro komunikaci, Jsem si jistá, že ten obchod něco má,
631 setkej se se mnou tady zase až něco najdeš."""
632         option_q = "Chápu."
633 
634         [marie.alley.waiting]
635         dialog = "\u001B[35mMarie\u001B[0m: No? Běž pro něj."
636         option_q = "Ok."
637 
638         [marie.alley.mission_brief]
639         dialog = """\u001B[35mMarie\u001B[0m: Tak plán je, ty půjdeš do Zdravotního střediska, přijdeš k recepci,
640 zkus ale splynout s ostatními... Třeba najdi židli na kterou si sedneš
641 nebo něco, tak či tak, až tam budeš a ochranka nebude koukat,
642 použij svůj komunikátor a dej mi vědět.
643 
644 \u001B[35mMarie\u001B[0m: Odtamtud zkusím způsobit rozrušení a poslat je do jejich odpočívárny,
645 mezitím co ty zkusíš vniknout do kanceláře a najít cokoliv co můžeš...
646 ale až tam budeš, dej mi vědět.
647 
648 \u001B[35mMarie\u001B[0m: Pokud chceš, můžeš vyzkoušet svůj komunikátor teďkon."""
649         option_q = "Ok."
650 
651     [marie.comms]
652     received = """\u001B[35mMarie\u001B[0m: Dobře, mám ty dokumenty, všude je rozesílám,
```

```
Dobrá práce."""
    bad_documents = "\u001B[35mMarie\u001B[0m: Tyhle nevypadaj jako ty správný dokumenty..."
    no_access = "\u001B[31mPřístup zamítnut!\u001B[0m"

        [marie.comms.orientation]
        dialog = "\u001B[35mMarie\u001B[0m: Co chceš?"
        option_1 = "Kam mám zase jít?"
        option_2 = "Slyšíš mě?"
        option_q = "Ale nic."

        [marie.comms.directions]
        dialog = """\u001B[35mMarie\u001B[0m: Musíš jít do Zdravotního střediska, až tam budeš, zkus splynout,
a dej mi vědět až budeš připraven."""
        option_q = "Mám to."

        [marie.comms.hear]
        # translation(jan): is jasně, čistě or zřetelně better here?
        dialog = "\u001B[35mMarie\u001B[0m: Hlasitě a jasně."
        option_q = "Super."

        [marie.comms.complaint]
        dialog = "\u001B[35mMarie\u001B[0m: Zkus zapadnout..."
        option_q = "Ok."

        [marie.comms.in_position]
        dialog = "\u001B[35mMarie\u001B[0m: Ok, udělám rozrušení, dej mi chvilku..."
        option_1 = "[počkat]"

        [marie.comms.distraction]
        dialog = """\u001B[31mOchranka\u001B[0m: Kdo tam je?
Měl by si vyjít nebo budeme mít velký problem.

(vidíš ochranku vejít do odpočívárny)
"""
        option_1 = "[počkat]"

        [marie.comms.coast_is_clear]
        dialog = "\u001B[35mMarie\u001B[0m: Ok, nyní je tvoje šance."
        option_q = "Jdu na to."

        [marie.comms.gogogo]
```

```
694         dialog = "\u001B[35mMarie\u001B[0m: Na co čekáš??"
695         option_q = "ÁÁÁÁÁÁÁÁÁÁÁÁÁÁÁÁÁÁÁ"
696
697         [marie.comms.office]
698         dialog = "\u001B[35mMarie\u001B[0m: Co vidíš?"
699         option_1 = "Hromadu všude rozprostřených dokumentů."
700
701         [marie.comms.documents]
702         dialog = """\u001B[35mMarie\u001B[0m: Ok, projdi je a vyber cokoliv, co vypadá relevantní, vyhledej
703 cokoliv, co zmiňuje to, o čem protestovali. Jakmile je najdeš, budeš mi je
704 muset poslat, tuším běž zpátky do svého bytu a uděláme to odtamtud,
705 dej mi vědět až tam budeš.
706
707 \u001B[35mMarie\u001B[0m: Budu jistit, aby si tě nevšimli.
708
709 \u001B[33mNápověda: můžeš použít dokumenty aby si je přečetl.\u001B[0m"""
710         option_q = "Ok."
711
712         [marie.comms.home]
713         dialog = """\u001B[35mMarie\u001B[0m: Pravděpodobně máš ty dokumenty, zapni si počítač, otevři si skener
714 dokumentů, a nahraj je."""
715
716 [stage]
717 # replace "režimu otevřeného světa" with "režimu open world" if you think that suits more
718 reached_conclusion = """\n
719 Dosáhl si konec hry, jsi nyní v režimu otevřeného světa.
720 Pokračuj v prozkoumávání všeho, co si neprozkoumal, pro maximální skóre.
721
722 Můžeš skončit tím, že napíšeš \u001B[47m\u001B[30mwin\u001B[0m
723 """
```

```toml
# Translations for World of These
# Provided by Infi

[global]
sight = "Du kannst Folgendes sehen:"

    [global.can_go_in_x_directions]
    1 = "Du kannst in"
    2 = "Richtungen gehen"

[selectors]
direction = "<Richtung>"
something = "<Etwas>"
someone = "<Jemand>"
item = "<Item>"

    [selectors.cant_find]
    1 = "Du schaust nach"
    2 = ", aber du findest nichts"

[commands]
unknown = "Keine Ahnung, was du gerade versuchst."

back = "Zurück zum vorherigen Raum"
quit = "Spiel beenden"
where_am_i = "Das Zimmer nochmal beschreiben"

    [commands.bag]
    usage = "Schau in deinen Beutel oder nach dem Inventar von etwas"
    cant_find = "Was du suchst kann nicht gefunden werden"
    empty = "Dein Beutel ist leer!"
    entity_empty = "hat nichts im Inventar"
    can_carry_kg = "Deine Tragekraft beträgt"
    are_carrying_kg = "Du trägst"
    look_in_bag = "Du schaust in deinen Beutel und siehst folgendes:"
    entity_appears_to_have = "hat anscheinend"

    [commands.drop]
    usage = "Lege ein Item ab"
    nothing_specified = "Was willst du ablegen?"

```

```toml
        [commands.drop.dropped]
        1 = "Du legst"
        2 = "ab"

    [commands.give]
    usage = "Etwas an jemanden geben"
    nothing_specified = "Was willst du geben?"
    no_target = "Wem oder in was willst du das geben?"
    denied_player = "Du kannst dich selbst nicht selber geben \u1F438"

        [commands.give.denied]
        1 = "Du kannst"
        2 = "nicht geben an"

    [commands.go]
    usage = "Gehe in eine bestimmte Richtung"
    nothing_specified = "Wohin gehst du?"

    [commands.help]
    usage = "Hilfemenü anzeigen"
    can_run = "Du kannst die folgenden Befehle ausführen:"

    [commands.pet]
    usage = "Streichel etwas in deiner Umgebung oder deinem Inventar"
    nothing_specified = "Was willst du streicheln?"
    denied = "Du kannst nicht folgendes streicheln:"

    [commands.take]
    usage = "Nimm ein Item auf"
    nothing_specified = "Von wem willst du etwas nehmen?"
    entity_does_not_have_entity = "hat nicht"
    item_not_specified = "Was willst du nehmen?"

        [commands.take.took]
        1 = "Du nimmst"
        2 = "von"
        3 = "und machst es in deinen Beutel"

        [commands.take.denied]
        1 = "Du kannst"
        2 = "nicht nehmen, es ist zu schwer für dich und deinen Beutel"
```

```
83
84      [commands.talk]
85      usage = "Fang an, mit jemandem zu reden"
86      nothing_specified = "Mit was willst du reden?"
87      denied = "Du kannst nicht reden mit:"
88
89      [commands.use]
90      usage = "Benutze ein Item in deiner Umgebung oder deinem Inventar"
91      nothing_specified = "Was willst du benutzen?"
92      denied = "Du kannst nicht folgendes benutzen:"
93
94      [commands.map]
95      usage = "Zeige die Weltkarte"
96      close = "Drück Enter, um die Weltkarte zu schließen"
97
98          [commands.map.discovered]
99          1 = "Du hast"
100         2 = "der Weltkarte erkundet"
101
102     [commands.win]
103     usage = "Gewinnen"
104     conclusion = """...
105
106  Kapitel 4.:
107 Die Dateien werden ins Internet freigegeben, wo sie jedermann lesen kann, die
108 Leute sind schnell um jedes kleine Detail zu erkennen, einige Details kommen
109 schnell ans Licht:
110 - Slyvasta hat über ethisch fragwürdige Wissenschaftsgebiete geforscht, genau-
111   genommen, wurden hunderte Tests täglich an verschiedenen Tiermensch-
112   testobjekten durchgeführt. Aber niemand konnte beweisen, dass jemand gegen
113   ihren eigenen Willen dort war.
114 - Jedoch wurde die Forschung auch sofort von ausländischen Mächten aufgenommen,
115   die schnell entdeckt haben, dass die Tiermenschgesellschaft in der Stadt eine
116   viel größere Bedrohung ist, als sie zuerst erwartet haben.
117 - Die Staft, mit dem öffentlichem Rat von Sylvasta, hat sofort die Evakuierung
118   von allen Bürgern angeordnetm, zu jedem Ort an dem sie einen potenziell töd-
119   lichen Konflikt fürchteten. Die Stadt ging in den nächsten Tagen relativ
120   schnell unters Feuer.
121 """
122     stats = "\u001B[47m\u001B[30mDeine Spielstatistiken\u001B[0m"
123     total_ticks = "Insgesamt Spielticks: "
```

```toml
    total_time = "Spielzeit insgesamt: "
    minutes = "minuten"
    seconds = "sekunden"
    sidequests_complete = "Sidequests abgeschlossen: "
    press_enter_key = "Drücke Enter, um das Spiel zu schließen."

[entities]
boat_key = "Ein Schlüssel für ein Schnellboot"

    [entities.bed]
    description = "Bett"
    use = "Du schläfst im Bett."

    [entities.boat]
    description = "Schnellboot an der Küste"
    locked = "Das Boot ist abgeschlossen."
    locked_for_sale = "Das Boot ist abgeschlossen.\nEs klebt eine Notiz darüber, dass du den\nLadenbesitzer kontaktieren kannst, um es zu kaufen."
    denied = "Du darfst nichts tragen, sonst darfst du nicht aufs Boot.\nDu kannst aber Dinge ins Boot legen."
    travel = "Du springst ins Boot und reist zu der anderen Seite..."
    too_heavy = "Im Boot ist bereits viel zu viel Zeug drin!"

        [entities.boat.give]
        1 = "" # empty
        2 = "ins Boot legen"

    [entities.cat]
    description = "Eine verfallene schwarze Katze"
    pet = "Du streichelst die Katze."
    use = "Du kannst nicht die Katze.\nBitte nicht die Katze. \u1F633\u1F633\u1F633"
    enter = "Eine Katze läuft her."
    leave = "Du siehst, wie eine Katze wegläuft."

    [entities.comms]
    description = "Kommunikationsgerät"
    off = "Das Gerät ist ausgeschaltet."

    [entities.couch]
    description = "Eine braune Ledercouch"
    sitting = "Du sitzt bereits in der Couch."
    sit = "Du setzt dich auf die Couch."
```

```
164
165      [entities.laptop]
166      description = "Laptop"
167
168          [entities.laptop.boot]
169          dialog = "Du schaltest den Laptop an..."
170          option_1 = "[wait]"
171
172          [entities.laptop.home]
173          dialog = "Wähle eine Option:"
174          option_q = "Herunterfahren"
175          option_1 = "/Meine Bilder"
176          option_2 = "/Lustige Katzenvideos"
177          option_3 = "/Marie's Dokumentenscanner"
178
179          [entities.laptop.pictures]
180          dialog = """Es ist ein Bild in deinem Bilderordner.
181
182  \u1F99D
183
184
185
186
187
188
189
190  """
191          option_q = "Zurück."
192
193          [entities.laptop.cat_videos]
194          dialog = "Du schaust dir lustige Katzenvideos an..."
195          option_q = "Nett hier."
196
197          [entities.laptop.document]
198          dialog = """\u001B[35mMarie\u001B[0m's Dokumentenscanner"""
199          option_q = "Beenden"
200          option_1 = "Dokumente senden"
201
202  [home]
203  first_load = '''
204  Du wirst gleich in die Welt platziert.
```

```
Falls du an einem Punkt ratlos feststeckst,
kannst du `help` nutzen um all die verfügbaren
Befehle zu sehen.

---

Du wachst auf und hörst, wie Leute draußen miteinander quatschen..
Du solltest gestern wirklich das Fenster geschlossen haben..

Neugierig, schaust du aus dem Fenster um zu gucken, was los ist.. Es ist eine
Gruppe an Demonstranten außerhalb des Medizinzentrums weiter runter auf der
Straße, du kannst nicht wirklich erkennen, was sie sagen oder was auf den
Schildern steht.

Aber es wäre nicht überraschend, falls etwas komisches da drin passieren würde,
aber du kannst nicht wirklich sagen, was. Vielleicht ist ja etwas in den
Nachrichten..
'''

enter = '''
Du gehst in deine Wohnung.
'''

    [home.tv]
    description = "LG 55NANO966PA 55\" Super UHD 8K HDR Smart LED TV"
    off = "Schalte den Fernseher aus."
    keep_watching = "Schau weiter..."

        # Red: \u001B[31m
        # Green: \u001B[32m
        # Yellow: \u001B[33m
        # Cyan: \u001B[36m
        [home.tv.first_on]
        dialog = "Du schaltest den Fernseher ein.\n\nDer Nachrichtensender kommt..."
        dialog_a="""\u001B[31mNachrichtensprecher\u001B[0m: Bürgerunruhen nehmen zu, Sylvasta steht von allen
Seiten in der Kritik, und viele sind über ihre Zukunft unsicher, während die
Spannung zwischen den Menschlichen und Tiermenschlichen Gesellschaften
eskalierende Konflikte an den Grenzen der Stadt verursacht.

\u001B[31mNachrichtensprecher\u001B[0m: Wir bringen ihnen jetzt Szenen aus der Umgebung des Medizinzentrums,
in denen eine Gruppe Demonstranten in Opposition gegenüber der Forschung bei
```

```
Sylvasta erschienen ist.

\u001B[36mKorrespondent\u001B[0m: Ich bin hier, stehe dreußen mit einer Gruppe Demonstranten..

\u001B[36mKorrespondent\u001B[0m zum \u001B[33mDemonstranten\u001B[0m: Was bringt sie heute hierher?
"""

        dialog_b="""\u001B[33mDemonstrant\u001B[0m: Sie nehmen das Geld der Stadt und nutzen es für ihren eigenen
Vorteil, sie sollten keine Fördergelder bekommen, geschweige denn hier ihre
Forschung betreiben dürfen.

\u001B[31mNachrichtensprecher\u001B[0m: Mutige Behauptungen direkt von außerhalb Sylvasta, aber ob
diese begründet sind ist noch nicht bekannt. Wir haben Monate und
Monate an Leaks von früheren Mitarbeitern und internen Fehlern, aber
die echte Absicht hinter den Menschen bei Sylvasta bleibt noch abzuwarten.

\u001B[31mNachrichtensprecher\u001B[0m: Sylvasta hat persönlich angekündigt, dass sie sich weigern,
öffentlich jegliche Informationen über ihre Forschung preiszugeben,
begründet mit der öffentlichen Sicherheit. Was genau das bedeutet,
weiß niemand außer ihre eigenen Leute.
"""

dialog_c="""\u001B[31mNachrichtensprecher\u001B[0m: So stellt sich auch die Frage, ob die Demonstrationen
unbegründet sind und lediglich Probleme in der Stadt verursachen wollen.
Vorher haben wir auch mit lokalen Einwohnern der Stadtmitte Earlier
gesprochen..

\u001B[32mLadenbesitzer\u001B[0m: Ich glaube, diese Leute wollen nur Probleme, Sylvasta war
unglaublich wichtig darin, die Stadt aufzubauen und uns in Frieden leben zu
lassen, ohne dass wir uns darüber Sorgen machen müssen, attackiert zu
werden. Ich glaube es gibt nicht genug Gründe, zu demonstrieren.

\u001B[32mLadenbesitzer\u001B[0m: Wie sehen bereits, wie die Welt sich gegen unsere Stadt
wendet, und solche internen Konflifte werden ihnen einfach einen Grund geben,
die Stadt zu übernehmen.

\u001B[31mNachrichtensprecher\u001B[0m: Damit ist unsere Sendung für heute Morgen beendet,
und um eins werden wir wieder zurück sein mit der Nachrichtenstunde.
"""

[apartments]
```

```
enter = '''
Du gehst in die Rezeption des Wohnungsbaus.
'''

    # Cyan: \u001B[36m
    [apartments.receptionist]
    description = "Der Rezeptionist sitzt hinter einem Tisch"

        [apartments.receptionist.first_encounter]
        dialog = "\u001B[36mRezeptionist\u001B[0m: Guten morgen, wie geht es Ihnen heute?"
        option_1 = "Was ist da draußen los?"
        option_q = "Egal."

        [apartments.receptionist.protestors]
        dialog = """\u001B[36mRezeptionist\u001B[0m: Ich weiß nicht viel, aber anscheinend schreien
Leute außerhalb des Medizinzentrums rum..."""
        option_1 = "Wissen Sie noch mehr?"
        option_q = "Achso."

        [apartments.receptionist.protestors2]
        dialog = """\u001B[36mRezeptionist\u001B[0m: Sie haben vorhin Flyer ausgeteilt, vielleicht würde
jemand in der Nähe es wissen..."""
        option_q = "Danke."

        [apartments.receptionist.repeated]
        dialog = "\u001B[36mRezeptionist\u001B[0m: Hallo, wie kann ich behilflich sein?"
        option_1 = "Was ist draußen mit den Demonstranten los?"
        option_q = "Das war's, danke."

[city_centre]
first_load = '''
Du kommst in die Stadtmitte, es ist morgens relativ voll. Viele Menschen
sind hier, und es ist unwahrscheinlich dass du die meisten für ein Gespräch
anhalten kannst.

Es ist generell sehr unruhig im Gebiet, einige Menschen sehen angespannt aus,
und andere, als würde die Welt heute enden...
'''

enter = '''
Du kommst in die Stadtmitte.
```

```
'''

    # Yellow: \u001B[33m
    [city_centre.npc]
    description = "Eine Person auf einer Bank"

        [city_centre.npc.small_talk]
        dialog = "\u001B[33mFremder\u001B[0m: Hallo, kann ich ihnen helfen?"
        option_1 = "Haben Sie zufällig Demonstranten hier gesehen?"
        option_2 = "Was lesen Sie?"
        option_q = "Egal."

        [city_centre.npc.protestors]
        dialog = """\u001B[33mFremder\u001B[0m: Oh ja, sie waren sehr laut..
Eigentlich ganz nervig, mein Morgen wurde gestört.
...
Sie haben mir aber einen Flyer dagelassen."""
        option_1 = "Kann ich ihn sehen?"

        [city_centre.npc.enquire]
        dialog = "\u001B[33mFremder\u001B[0m: Die Demonstranten waren hier und haben mir ein Flyer dagelassen."

        [city_centre.npc.leaflet]
        # dialog = "\u001B[33mStranger\u001B[0m: Sure, here you go, keep it.\n[\u001B[33mStranger\u001B[0m gave you an
item.]"
        dialog = """\u001B[33mStranger\u001B[0m: Klar, hier hab ich es...

Du schaust auf den Flyer:

\u001B[41mUNETHISCHE FORSCHUNG
\u001B[40m\u001B[31mUnsere Stadt ist in Gefahr, Slyvasta finanziert
weiter ihre Forschung, aber machen nichts dafür,
dass die Stadt am Leben bleibt.

Sylvasta nutzt ihre Stellung aus, um
weiter an komplett ILLEGALER Biotech-
nischer Herstellung von Tiermenschen
zu forschen.\u001B[0m

Unterstützen Sie uns: \u001B[36mhttps://sylvasta.vercel.app\u001B[0m
"""
```

```
368         option_1 = "Dankeschön."
369
370 [street]
371 first_load = '''
372 Du kommst in die Hauptstraße, die die hauptsächlichen Teile der Stadt vereint.
373
374 Es gibt viel Chaos und Lärm, zu deinem Westen siehst du ein paar
375 Demonstranten vor dem Medizinzentrum, sie halten Schilder hoch,
376 "UNETHISCHE FORSCHUNG", "UNSERE BEZIEHUNGEN AUF DEM SPIEL",
377 "UNVORHERGESEHENE KONSEQUENZEN".
378 '''
379
380 enter = '''
381 Du kommst in die Hauptstraße.
382 '''
383
384     # Red: \u001B[31m
385     [street.protestors]
386     description = "Eine Gruppe Demonstranten hält Schilder hoch"
387     blocking = "Eine Gruppe an Demonstranten blockiert den Eingang."
388
389         [street.protestors.small_talk]
390         dialog = "\u001B[31mDemonstrant\u001B[0m: Hast du darüber Gehört, was Sylvasta macht und warum wir hier sind?"
391         option_1 = "Nein, aber erzähl doch mal darüber."
392         option_2 = "Egal."
393
394         [street.protestors.enquire]
395         dialog = """\u001B[31mDemonstrant\u001B[0m: Sie nutzen öffentliches Geld um ihre illegale Forschungen
396 weiterzuführen, sie haben doch die Leaks gesehen, oder?"""
397         option_1 = "Nein."
398         option_2 = "Alles klar.. Ich lass Sie dann mal in Ruhe."
399
400         [street.protestors.leaks]
401         dialog = """\u001B[31mDemonstrant\u001B[0m: Naja, einige frühere Mitarbeiter haben Infos über ihre aktuellen
402 Forschungsprojekte geleakt und anscheinend experimentieren sie mit Biotechnik
403 an Tiermenschen, das sollte nicht toleriert oder gar subventioniert werden."""
404         option_1 = "Ok."
405
406         [street.protestors.confrontation]
407         dialog = "\u001B[31mProtestor\u001B[0m: You taking me for some sort of conspiracy theorist?"
408         option_1 = "Ja."
```

```toml
        option_2 = "Ich gehe jetzt einfach."

[shop]
enter = '''
Du kommst in den Laden, die Glocke klingelt als du die Tür hinter dir schließt.
'''

    # Green: \u001B[32m
    [shop.npc]
    description = "Der Ladenbesitzer"
    leave = "Egal."

    currently_have_amount_of_money = "Du hast aktuell:"
    not_enough = "Du hast nicht genug Geld und leistest dir kein"
    too_heavy = "Zu schwer für dich!"

    out_of_stock = "ausverkauft"
    x_left = "übrig"

        [shop.npc.fake_item]
        cat = "Eine einzige Katze"

        [shop.npc.item_out_of_stock]
        1 = "Dieses Item ist"
        2 = ", du kannst es nicht kaufen"

        [shop.npc.bought]
        1 = "Du kaufst dir"
        2 = "und machst es in deinen Beutel."

        [shop.npc.greeting]
        Exposition = "\u001B[32mLadenbesitzer\u001B[0m: Guten Tag, was würden Sie gerne kaufen?"
        Recon = "\u001B[32mLadenbesitzer\u001B[0m: Hallo, was neues kaufen?"
        Stealth = "\u001B[32mLadenbesitzer\u001B[0m: Wollen Sie etwas kaufen?"
        End = "\u001B[32mLadenbesitzer\u001B[0m: Brauchen Sie etwas?"

[back_alley]
first_load = '''
Du kommst in eine dunkle Gasse, es ist hier ziemlich leise, ein starker
Kontrast zur Stadtmitte, obwohl sie direkt dazwischen liegt. Die Seiten der
Gasse sind voll mit Bauten und dunkler Beleuchtung.
```

```
450
451 Du siehst vor dir eine dir bekannte Figur.
452 '''
453
454 enter = '''
455 Du kommst in eine dunkle Gasse.
456 '''
457
458 [coastline]
459 enter = '''
460 Du kommst an die Stadtküste, das Wasser ist eher ruhig und still.
461 '''
462
463 [mainland_coastline]
464 enter = '''
465 Du kommst an die Festlandküste,
466 der See konfrontiert direkt die Steine, die gegen ihn ausgelegt sind,
467 es ist außerhalb des Beton-Dschungels außerdem viel leiser.
468 '''
469
470 [forest]
471 enter = '''
472 Du wanderst durch den Wald und hörst wie die Vögel an den Baumkronen pfeifen...
473 Durch die Mitte fließt ein Fluss und es ist eine alte Holzbrücke darüber.
474 Es würde relativ einfach sein, dich hier für eine Weile zu verlaufen.
475 '''
476
477     # Yellow: \u001B[33m
478     [forest.old_man]
479     description = "Ein alter Mann, der durch den Wald wandert"
480     full = "Du kannst ihm nichts mehr geben."
481     accept = "\u001B[33mAlter Mann\u001B[0m: Danke, dass du ihn mir zurückgebracht hast."
482     deny = "\u001B[33mAlter Mann\u001B[0m: Ich möchte aber nicht den"
483
484         [forest.old_man.small_talk]
485         dialog = "\u001B[33mAlter Mann\u001B[0m: Du hast meine schwarze Katze doch nirgendwo gesehen?"
486         option_1 = "In der Stadt habe ich so eine gesehen..."
487         option_q = "Nein."
488
489         [forest.old_man.request]
490         dialog = "\u001B[33mAlter Mann\u001B[0m: Könntest du sie mir zurückbringen?"
```

```
491         option_1 = "Klar doch."
492         option_q = "Gerade geht das leider nicht."
493
494         [forest.old_man.praise]
495         dialog = "\u001B[33mAlter Mann\u001B[0m: Danke für deine Hilfe!"
496         option_q = "[verlassen]"
497
498 [worm_hole]
499 enter = '''
500 Du gehst ins Wurmloch...
501 '''
502
503 [medical_centre]
504 enter = '''
505 Du bist jetzt an der Rezeption des Medizinzentrums.
506 '''
507
508     # Red: \u001B[31m
509     [medical_centre.guard]
510     description = "Sicherheitsmann an der Tür"
511     blocking = "Die Security schaut auf die Treppen, du kannst nicht an ihnen vorbei."
512
513         [medical_centre.guard.small_talk]
514         dialog = "\u001B[31mWachmann\u001B[0m: Was willst du?"
515         option_q = "Egal."
516
517 [medical_centre_office]
518 enter = '''
519 Du bist im Büro des Medizinzentrums.
520 Du solltest hier eigentlich auf jeden Fall nicht sein...
521 '''
522
523     [medical_centre_office.books]
524         [medical_centre_office.books.1]
525         title = "Alan Stern. Auf der Suche nach New Horizons"
526         contents = """
527 Am 14. Juli 2015 geschah etwas Erstaunliches. Mehr als 3 Milliarden Meilen von
528 der Erde, eine kleine NASA-Raumsonde namens New Horizons, flog an Pluto
529 mit 32.000 Meilen pro Stunde vorbei und konzentrierte seine Instrumente auf
530 die lang mysteriösen Eiswelten des Pluto-Systems und ging dann genauso schnell
531 weiter hinaus ins Jenseits.
```

```
"""

        [medical_centre_office.books.2] # yes
        title = "Stapel von Forschungspapieren"
        contents = """
Bei unseren jüngsten Testreihen haben wir festgestellt, dass die Wirkung von
\u001B[47m\u001B[37m[redacted]\u001B[0m erhebliche negative Auswirkungen hat.
Diese Tests wurden über eine Reihe von 8 Tagen durchgeführt und verursachten
\u001B[47m\u001B[37m[redacted]\u001B[0m und \u001B[47m\u001B[37m[redacted]\u001B[0m im Körper.

Dieses Dokument ist vertraulich und darf nicht an Dritte weitergegeben werden.
"""

        [medical_centre_office.books.3]
        title = "Barry A. Burd. Java For Dummies"
        contents = """
Java ist die Plattformunabhängige, objektorientierte Programmiersprache, die für
die Entwicklung von Web und Mobile-Anwendungen genutzt wird. Die neue Version
hat erweiterte Funkionalität, auf die Programmierer ganz aufgeregt warteten, und
dieses Handbuch wird alle davon vorstellen.

Sie werden lernen, wie man gut in Java programmiert, etwa so:

\u001B[36mimport \u001B[33mjava\u001B[37m.\u001B[33mio\u001B[37m.\u001B[33mBufferedReader\u001B[37m;
\u001B[36mimport \u001B[33mjava\u001B[37m.\u001B[33mio\u001B[37m.\u001B[33mInputStreamReader\u001B[37m;
\u001B[33mpublic class \u001B[31mMain  \u001B[34m{
    \u001B[33mpublic static  void \u001B[31mmain  \u001B[34m(\u001B[33mString[] args\u001B[34m) \u001B[34m{
        \u001B[33mtry{ String a=new \u001B[31mBufferedReader \u001B[34m(
            \u001B[33mnew \u001B[31mInputStreamReader\u001B[34m( System\u001B[37m.\u001B[33mout\u001B[37m \u001B[34m))
\u001B[37m.
                \u001B[31mreadLine() \u001B[37m;
        \u001B[33mif
            \u001B[34m(a == \u001B[32m"hi"\u001B[34m) {
            System\u001B[37m.\u001B[33mout\u001B[37m.\u001B[31mprintln\u001B[34m(\u001B[32m"Hello!!"\u001B[34m)
\u001B[37m;
        \u001B[34m} \u001B[33melse
       \u001B[34m{
            System\u001B[37m.\u001B[33mout\u001B[37m.\u001B[31mprintln\u001B[34m(\u001B[32m"you did not greet me!"\u001B[
34m) \u001B[37m;
        \u001B[34m}
    } \u001B[33mcatch \u001B[34m(\u001B[33mException e\u001B[34m) {\u001B[35m/*something bad happen! */ \u001B[34m} } }
```

```
\u001B[0m"""

        [medical_centre_office.books.4] # yes
        title = "Dr. Mike Ox. Hochstapler in die Gesellschaft integrieren"
        contents = """
[Markiert als nur für den internen Gebrauch.]

Dieses Buch beschreibt detalliert Methoden, welche genutzt werden können, um das
allgemeine Volk zu manipulieren. Ich erkläre ganz genau, wie man ge-
fährliche Personen ganz subtil entfernt und Geheimnisse geheim haltet.
"""

        [medical_centre_office.books.5] # yes
        title = "RNA-Sequenzen"
        contents = """
Vertrauliches Dokument

Figur 1.
GAGAATAAACTAGTATTCTTCTGGTCCCCACAGACTCAGAGAGAACCCGCCACCATGTTCGTGTTCCTGGTGCTGCTGCC
TCTGGTGTCCAGCCAGTGTGTGAACCTGACCACCAGAACACAGCTGCCTCCAGCCTACACCAACAGCTTTACCAGAGGCG
TGTACTACCCCGACAAGGTGTTCAGATCCAGCGTGCTGCACTCTACCCAGGACCTGTTCCTGCCTTTCTTCAGCAACGTG
ACCTGGTTCCACGCCATCCACGTGTCCGGCACCAATGGCACCAAGAGATTCGACAACCCCGTGCTGCCCTTCAACGACGG
GGTGTACTTTGCCAGCACCGAGAAGTCCAACATCATCAGAGGCTGGATCTTCGGCACCACACTGGACAGCAAGACCCAGA
GCCTGCTGATCGTGAACAACGCCACCAACGTGGTCATCAAAGTGTGCGAGTTCCAGTTCTGCAACGACCCCTTCCTGGGC
GTCTACTACCACAAGAACAACAAGAGCTGGATGGAAAGCGAGTTCCGGGTGTACAGCAGCGCCAACAACTGCACCTTCGA
GTACGTGTCCCAGCCTTTCCTGATGGACCTGGAAGGCAAGCAGGGCAACTTCAAGAACCTGCGCGAGTTCGTGTTTAAGA
ACATCGACGGCTACTTCAAGATCTACAGCAAGCACACCCCTATCAACCTCGTGCGGGATCTGCCTCAGGGCTTCTCTGCT
CTGGAACCCCTGGTGGATCTGCCCATCGGCATCAACATCACCCGGTTTCAGACACTGCTGGCCCTGCACAGAAGCTACCT
GACACCTGGCGATAGCAGCAGCGGATGGACAGCTGGTGCCGCCGCTTACTATGTGGGCTACCTGCAGCCTAGAACCTTCC
TGCTGAAGTACAACGAGAACGGCACCATCACCGACGCCGTGGATTGTGCTCTGGATCCTCTGAGCGAGACAAAGTGCACC
"""

        [medical_centre_office.books.6]
        title = "Jules Verne. Um die Welt in 80 Tagen"
        contents = """
An einem unglückseligen Abend im Reform Club wettet Phileas Fogg unvorsichtiger-
weise mit seinen um 20.000 Pfund, dass er in nur achtzig Tagen um den gesamten
Globus reisen kann. um die ganze Welt reisen kann - und er ist fest entschlossen,
nicht zu verlieren. Der zurückhaltende Engländer durchbricht die gewohnte Routine
sofort und macht sich auf den Weg nach Dover, in Begleitung seines heißblütigen
französischen Dieners Passepartout. Unterwegs mit Zug, Dampfschiff, Segelboot,
```

```
Schlitten und sogar einem Elefanten müssen sie Stürme, Entführungen, Natur-
katastrophen, Sioux-Angriffe und den verbissenen Inspektor Fix von Scotland Yard
- der glaubt, dass Fogg die Bank von England ausgeraubt hat - überwinden, um die
außergewöhnliche Wette zu gewinnen.
"""

[marie]
description = "\u001B[35mMarie der Nerz\u001B[0m"

    # Purple: \u001B[35m
    [marie.alley]

        [marie.alley.small_talk]
        dialog = "\u001B[35mMarie\u001B[0m: Was bringt dich hierher?"
        option_1 = "Hast du von den Protesten gehört?"
        option_2 = "Sylvasta hat etwas böses vor..."
        option_q = "Egal, hab einen schönen Tag."

        [marie.alley.protests]
        dialog = """\u001B[35mMarie\u001B[0m: Ja, die von Sylvasta haben auf jeden Fall
nichts Gutes auf dem Schirm, aber ich weiß auch nicht ob ich den Demonstranten
trauen sollte."""
        option_1 = "Willst du mir dabei helfen, das herauszufinden?"
        option_q = "Egal."

        [marie.alley.sylvasta]
        dialog = "\u001B[35mMarie\u001B[0m: Glaubst du, ich weiß das nicht? Willst du mir helfen, es herauszufinden?"
        option_1 = "Klar."
        option_q = "Nein danke."

        [marie.alley.confirm]
        dialog = "\u001B[35mMarie\u001B[0m: Wir könnten einbrechen und die Wahrheit herausfinden."
        option_1 = "Lass machen! [bringt die Story weiter]"
        option_q = "Auf keinen Fall!"

        [marie.alley.recon]
        dialog = """\u001B[35mMarie\u001B[0m: Okay, während ich nachdenke, wie wir reinkommen, musst du mir
eetwas bringen, worüber wir reden müssen, ich bin mir sicher, dass es im Laden etwas gibt, komm wieder hierher
wenn du zurückkommst."""
        option_q = "Verstanden."

```

```
652         [marie.alley.waiting]
653         dialog = "\u001B[35mMarie\u001B[0m: Na dann? Hol es dir."
654         option_q = "Okay."
655 
656         [marie.alley.mission_brief]
657         dialog = """\u001B[35mMarie\u001B[0m: Also, der Plan ist, du gehst zum Medizinzentrum,
658 bummelst lässig zur Rezeption, du darfst aber nicht auffallen... Finde dir zum Beispiel einen
659 Stuhl um darauf zu sitzen oder so was. Jedenfalls, wenn du dort bist und denkst dass die
660 Security nicht zuschaut, nutz dein Kommunikationsgerät um mir bescheid zu sagen.
661 
662 \u001B[35mMarie\u001B[0m: Alsdann werde ich eine Ablenkung senden und sie in ihr
663 Pausenzimmer führen während du ins Büro schleichst und versuchst, alles was du willst, zu finden...
664 Aber wenn du dort bist, sag mir bescheid.
665 
666 \u001B[35mMarie\u001B[0m: Wenn du willst, kannst du das Kommunikationsgerät jetzt testen."""
667         option_q = "Okay."
668 
669     [marie.comms]
670     received = """\u001B[35mMarie\u001B[0m: Alles klar, ich hab die Dokumente gefunden,
671 Ich sende sie an die breiten Massen. Gut gemacht mit dem Einbruch."""
672     bad_documents = "\u001B[35mMarie\u001B[0m: Das sieht nicht wie die richtigen Dokumente aus..."
673     no_access = "\u001B[31mZugriff verweigert!\u001B[0m"
674 
675         [marie.comms.orientation]
676         dialog = "\u001B[35mMarie\u001B[0m: Was willst du?"
677         option_1 = "Wohin muss ich nochmal?"
678         option_2 = "Kannst du mich hören?"
679         option_q = "Egal."
680 
681         [marie.comms.directions]
682         dialog = """\u001B[35mMarie\u001B[0m: Du musst ins Medizinzentrum, wenn du dort bist, versuch
683 nicht aufzufallen und sage mir, wenn du in Position bist."""
684         option_q = "Verstanden."
685 
686         [marie.comms.hear]
687         dialog = "\u001B[35mMarie\u001B[0m: Laut und deutlich."
688         option_q = "Nett."
689 
690         [marie.comms.complaint]
691         dialog = "\u001B[35mMarie\u001B[0m: Versuche, nicht aufzufallen..."
692         option_q = "Okay."
```

```
693
694         [marie.comms.in_position]
695         dialog = "\u001B[35mMarie\u001B[0m: Okay, Ich werde eine Ablenkung schaffen, gib mir einen Moment..."
696         option_1 = "[warten]"
697
698         [marie.comms.distraction]
699         dialog = """\u001B[31mWachmann\u001B[0m: Wer ist da?
700 Komm raus, oder wir haben ein Problem.
701
702 (Du siehst, wie der Wache in den Pausenraum geht)
703 """
704         option_1 = "[warten]"
705
706         [marie.comms.coast_is_clear]
707         dialog = "\u001B[35mMarie\u001B[0m: Okay, jetzt ist deine Chance."
708         option_q = "Verstanden."
709
710         [marie.comms.gogogo]
711         dialog = "\u001B[35mMarie\u001B[0m: Auf was wartest du??"
712         option_q = "AAAAAAA"
713
714         [marie.comms.office]
715         dialog = "\u001B[35mMarie\u001B[0m: Was siehst du?"
716         option_1 = "Rumgestreute Dokumente."
717
718         [marie.comms.documents]
719         dialog = """\u001B[35mMarie\u001B[0m: Ok, schau sie dir durch und finde alles, was relevant
720 ist, achte besonders auf alles, was die Sachen erwähnt, über die demonstriert wird.
721 Wenn du das gemachst hast, solltest du die Dokumente mir hochladen,
722 geh zurück zu deiner Wohnung und wir machen es von dort, sag mir wenn du dort bist.
723
724 \u001B[35mMarie\u001B[0m: Ich schau genau darauf, dass deine Deckung nicht platzt.
725
726 \u001B[33mTipp: Du kannst die Dokumente benutzen (use), um sie zu lesen.\u001B[0m"""
727         option_q = "Okay."
728
729         [marie.comms.home]
730         dialog = """\u001B[35mMarie\u001B[0m: Warscheinlich hast du die Dokumente, schalte deinen PC an,
731 geh auf den Link den ich dir gesendet hab, er sollte dort erscheinen, und lade dann hoch."""
732
733 [stage]
```

```
reached_conclusion = """\n
Du hast das Spielende erreicht, du bist jetzt in einem Open-World-Modus,
erkunde weiter um deine Ergebnisse zu maximieren.

Du kannst endgültig aufhören mit dem Befehl \u001B[47m\u001B[30mwin\u001B[0m
"""
```

```toml
# Translations for World of These

[global]
sight = "You can see:"

    [global.can_go_in_x_directions]
    1 = "You may go in"
    2 = "directions"

[selectors]
direction = "<direction>"
something = "<something>"
someone = "<someone>"
item = "<item>"

    [selectors.cant_find]
    1 = "You look around for"
    2 = "but can't find anything"

[commands]
unknown = "Not sure what you're trying to do."

back = "go back to the previous room"
quit = "quit the game"
where_am_i = "describe the current room again"

    [commands.bag]
    usage = "look inside your bag or at something's inventory"
    cant_find = "Can't find what you want to look at."
    empty = "Your bag is empty!"
    entity_empty = "doesn't appear to have anything"
    can_carry_kg = "You can carry"
    are_carrying_kg = "You are carrying"
    look_in_bag = "You look in your bag to see"
    entity_appears_to_have = "appears to have"

    [commands.drop]
    usage = "drop an item from your bag"
    nothing_specified = "What do you want to drop?"

        [commands.drop.dropped]
```

```toml
        1 = "You drop"
        2 = "out of your bag"

    [commands.give]
    usage = "give something to someone"
    nothing_specified = "What do you want to give?"
    no_target = "What / who are you putting this in?"
    denied_player = "You cannot give yourself to anyone or anything. \u1F438"

        [commands.give.denied]
        1 = "Cannot give"
        2 = "to"

    [commands.go]
    usage = "go in a certain direction"
    nothing_specified = "Where are you going?"

    [commands.help]
    usage = "show help menu"
    can_run = "You can run the following commands:"

    [commands.pet]
    usage = "pet something around you or in your inventory"
    nothing_specified = "What are you trying to pet?"
    denied = "You cannot pet"

    [commands.take]
    usage = "put something in your bag"
    nothing_specified = "From who?"
    entity_does_not_have_entity = "does not have"
    item_not_specified = "What do you want to take?"

        [commands.take.took]
        1 = "You take"
        2 = "from"
        3 = "and put it in your bag"

        [commands.take.denied]
        1 = "You cannot take"
        2 = "it's too heavy to put in your bag"

```

```
    [commands.talk]
    usage = "start talking with someone"
    nothing_specified = "What do you want to talk with?"
    denied = "You cannot talk with"

    [commands.use]
    usage = "use something around you or in your inventory"
    nothing_specified = "What do you want to use?"
    denied = "You cannot use"

    [commands.map]
    usage = "show the world map"
    close = "Press enter to close."

        [commands.map.discovered]
        1 = "You have discovered"
        2 = "of the world"

    [commands.win]
    usage = "win the game"
    conclusion = """...

Chapter 4.:
The files are released into the internet for anyone to read, people are quick to
analyse through every single tiny detail, some immediate details come to light:
- Sylvasta was researching ethically questionable areas of science, in
  particular, they were running hundreds of tests daily on a variety of beastman
  test subjects. But nobody could prove anyone was there against their will.
- However, the research did also get immediately picked up by foreign powers who
  quickly discovered that the beastman society living in the city is a far
  greater threat than they initially anticipated.
- The city, with Sylvasta's public advice, immediately ordered evacuation of all
  citizens to any area they could find fearing a potentially deadly conflict on
  the horizon. The city quickly came under fire over the coming days.
"""
    stats = "\u001B[47m\u001B[30mYour final game stats\u001B[0m"
    total_ticks = "Total game ticks: "
    total_time = "Total time played: "
    minutes = "minutes"
    seconds = "seconds"
    sidequests_complete = "Side-quests completed: "
```

```toml
    press_enter_key = "Press enter to close the game."

[entities]
boat_key = "A key to the speed boat"

    [entities.bed]
    description = "Bed"
    use = "You take a nap."

    [entities.boat]
    description = "Speedboat docked on the coast"
    locked = "The boat is locked."
    locked_for_sale = "The boat is locked.\nThere is a note which says to contact the shopkeeper to buy this boat."
    denied = "You must not be carrying anything to use the boat.\nYou can however put things in the boat."
    travel = "You hop in the boat and travel to the other side..."
    too_heavy = "The boat is carrying too much stuff already!"

        [entities.boat.give]
        1 = "Put"
        2 = "in the boat"

    [entities.cat]
    description = "A stray black cat"
    pet = "You pet the cat."
    use = "You cannot the cat.\nPlease do not the cat. \u1F633\u1F633\u1F633"
    enter = "A cat has wandered in among us."
    leave = "You see a cat leave."

    [entities.comms]
    description = "Communicator device"
    off = "The device is off."

    [entities.couch]
    description = "A brown leather couch"
    sitting = "You are already sitting in the couch."
    sit = "You sit down on the couch."

    [entities.laptop]
    description = "Laptop"

        [entities.laptop.boot]
```

```
165         dialog = "You turn the computer on..."
166         option_1 = "[wait]"
167
168         [entities.laptop.home]
169         dialog = "Select an option:"
170         option_q = "Power off"
171         option_1 = "/My Pictures"
172         option_2 = "/Funny cat videos"
173         option_3 = "/Marie's document scanner"
174
175         [entities.laptop.pictures]
176         dialog = """There is only one picture in your pictures folder:
177
178 \u1F99D
179
180
181
182
183
184
185
186 """
187         option_q = "Go back."
188
189         [entities.laptop.cat_videos]
190         dialog = "You look at funny cat videos..."
191         option_q = "Neat."
192
193         [entities.laptop.document]
194         dialog = """\u001B[35mMarie\u001B[0m's document scanner"""
195         option_q = "Quit"
196         option_1 = "Send documents"
197
198 [home]
199 first_load = '''
200 You're about to be placed into the world.
201 If at any point you are stuck with what to do,
202 you can use `help` to view all available commands.
203
204 ---
205
```

```
You wake up to the sound of people chanting outside..
You really should've closed the window last night..

Curious, you peer out the window to see what's going on.. there's a group of
protestors outside the Medical Centre down the street, you can't really make out
what they're saying or what their signs say.

Though it'd not be surprising if something strange is going on in there, but you
can't really put your finger on it. Maybe there's something on the news..
'''

enter = '''
You enter your apartment.
'''

    [home.tv]
    description = "LG 55NANO966PA 55\" Super UHD 8K HDR Smart LED TV"
    off = "Turn the TV off."
    keep_watching = "Keep watching..."

        # Red: \u001B[31m
        # Green: \u001B[32m
        # Yellow: \u001B[33m
        # Cyan: \u001B[36m
        [home.tv.first_on]
        dialog = "You turn the TV on.\n\nThe news channel comes up..."
        dialog_a="""\u001B[31mNews Anchor\u001B[0m: Civil unrest is rising, Sylvasta is facing criticism from all
sides, and many people are uneasy about their future as rising tension between
human and beastman societies is causing escalated conflict around the city
borders.

\u001B[31mNews Anchor\u001B[0m: We bring you now to scenes outside of the Medical Centre where a
group of protestors have shown up in opposition to the research being led at
Sylvasta.

\u001B[36mCorrespondent\u001B[0m: I am here, standing outside with the group of protestors..

\u001B[36mCorrespondent\u001B[0m to \u001B[33mProtestor\u001B[0m: What brings you here today?
"""

        dialog_b="""\u001B[33mProtestor\u001B[0m: They're taking the city's money and using it for their own gain, they
```

```
shouldn't receive any funding let alone be allowed to operate here.

\u001B[31mNews Anchor\u001B[0m: Bold claims coming straight from outside Sylvasta, whether these
claims are grounded in anything is yet to be discovered, we've seen months and
months of leaks come out from former employees and internal mishaps but are yet
to truly find out the intention behind the people at Sylvasta.

\u001B[31mNews Anchor\u001B[0m: Sylvasta has personally announced that they refuse to communicate
or elaborate any further on their internal research citing public safety,
whatever that means nobody outside of their internal staff knows.
"""

dialog_c="""\u001B[31mNews Anchor\u001B[0m: It also begs the question whether the protests are unfounded and
just there to stir up trouble in the city. Earlier today we also spoke to local
residents living in the centre of the city..

\u001B[32mShopkeeper\u001B[0m: I think these guys just want to cause trouble, Sylvasta was vital in
establishing this city and letting us live in peace without having to worry
about being attacked, I just don't think there's enough reason to protest.

\u001B[32mShopkeeper\u001B[0m: We're already seeing the world turn against the city and these sorts
of internal conflicts will just give them reason to step in and take control.

\u001B[31mNews Anchor\u001B[0m: That concludes our broadcast for this morning, and we'll be back
for the 1pm News Hour.
"""

[apartments]
enter = '''
You enter the apartment complex reception.
'''

    # Cyan: \u001B[36m
    [apartments.receptionist]
    description = "The receptionist sitting behind a desk"

        [apartments.receptionist.first_encounter]
        dialog = "\u001B[36mReceptionist\u001B[0m: Good morning, how are you doing today?"
        option_1 = "What is that racket outside?"
        option_q = "Nevermind."

```

```
        [apartments.receptionist.protestors]
        dialog = """\u001B[36mReceptionist\u001B[0m: I don't know much but it looks like a group of people shouting
outside the Medical Centre..."""
        option_1 = "Do you know anything more?"
        option_q = "Alright."

        [apartments.receptionist.protestors2]
        dialog = """\u001B[36mReceptionist\u001B[0m: They were handing out flyers as they came up, maybe someone
nearby would know..."""
        option_q = "Thanks."

        [apartments.receptionist.repeated]
        dialog = "\u001B[36mReceptionist\u001B[0m: Hello, what can I help you with?"
        option_1 = "What's going on with those protestors outside?"
        option_q = "That's all, thanks."

[city_centre]
first_load = '''
You enter the city square, it's quite busy in the mornings. There's a lot of
people running around, unlikely you could stop most of them for a chat.

There's a general unease in the area, some people look quite tense and others
look like they're spending their last day on Earth...
'''

enter = '''
You enter the city square.
'''

    # Yellow: \u001B[33m
    [city_centre.npc]
    description = "A person sitting at a bench"

        [city_centre.npc.small_talk]
        dialog = "\u001B[33mStranger\u001B[0m: Hello, could I help you?"
        option_1 = "You don't happen to have seen the protestors go by here?"
        option_2 = "What are you reading?"
        option_q = "Nevermind."

        [city_centre.npc.protestors]
        dialog = """\u001B[33mStranger\u001B[0m: Oh yes, they were quite loud..
```

```
329  They were rather annoying, disturbed my morning.
330  ...
331  They did leave me this leaflet though"""
332          option_1 = "Can I see it?"
333
334          [city_centre.npc.enquire]
335          dialog = "\u001B[33mStranger\u001B[0m: These guys went past and left me a leaflet."
336
337          [city_centre.npc.leaflet]
338          # dialog = "\u001B[33mStranger\u001B[0m: Sure, here you go, keep it.\n[\u001B[33mStranger\u001B[0m gave you an
item.]"
339          dialog = """\u001B[33mStranger\u001B[0m: Sure, I've got it here...
340
341  You look at the leaflet:
342
343  \u001B[41mUNETHICAL RESEARCH
344  \u001B[40m\u001B[31mOur city is in danger, Sylvasta further
345  funding into their research yet do nothing
346  to help the city survive.
347
348  Sylvasta is abusing their position to
349  continue work into completely ILLEGAL
350  bioengineering of beastmen.\u001B[0m
351
352  Support us: \u001B[36mhttps://sylvasta.vercel.app\u001B[0m
353  """
354          option_1 = "Thanks."
355
356  [street]
357  first_load = '''
358  You enter the main city street connecting the major parts of the city.
359
360  There's a lot of chaos and noise here, to your west you can see protestors
361  outside of the medical centre complex, they're holding signs up, "UNETHICAL
362  RESEARCH", "OUR RELATIONS AT STAKE", "UNFORESEEN CONSEQUENCES".
363  '''
364
365  enter = '''
366  You enter the city street.
367  '''
368
```

```
    # Red: \u001B[31m
    [street.protestors]
    description = "A group of protestors holding signs"
    blocking = "There is a group of protestors blocking the way in."

        [street.protestors.small_talk]
        dialog = "\u001B[31mProtestor\u001B[0m: Have you heard about what Sylvasta is doing and why we're out here?"
        option_1 = "No, enlighten me."
        option_2 = "Nevermind."

        [street.protestors.enquire]
        dialog = """\u001B[31mProtestor\u001B[0m: They are using the city's money to further their frankly illegal
research, you've seen the leaks right?"""
        option_1 = "No."
        option_2 = "Right.. I'll leave you to it."

        [street.protestors.leaks]
        dialog = """\u001B[31mProtestor\u001B[0m: Well some former employees leaked information about their current
research projects and from what we can tell is that they're doing bioengineering
on beastmen, this shouldn't be tolerated let alone funded by the city."""
        option_1 = "Ok."

        [street.protestors.confrontation]
        dialog = "\u001B[31mProtestor\u001B[0m: You taking me for some sort of conspiracy theorist?"
        option_1 = "Yes."
        option_2 = "I'm leaving now."

[shop]
enter = '''
You enter the shop, the bell rings as you close the door behind you.
'''

    # Green: \u001B[32m
    [shop.npc]
    description = "The shop keeper"
    leave = "Nevermind."

    currently_have_amount_of_money = "You currently have:"
    not_enough = "You don't have enough money to buy"
    too_heavy = "Too heavy for you to carry!"

```

```
410      out_of_stock = "out of stock"
411      x_left = "left"
412
413          [shop.npc.fake_item]
414          cat = "A singular cat"
415
416          [shop.npc.item_out_of_stock]
417          1 = "This item is"
418          2 = "you may not buy it"
419
420          [shop.npc.bought]
421          1 = "You buy"
422          2 = "and put it in your bag"
423
424          [shop.npc.greeting]
425          Exposition = "\u001B[32mShopkeeper\u001B[0m: Good day, what would you like to buy?"
426          Recon = "\u001B[32mShopkeeper\u001B[0m: Hello, anything in mind today?"
427          Stealth = "\u001B[32mShopkeeper\u001B[0m: Looking to buy something?"
428          End = "\u001B[32mShopkeeper\u001B[0m: Need anything new?"
429
430  [back_alley]
431  first_load = '''
432  You enter a dark alley, it is quite quiet here, a stark contrast to the city
433  centre yet located nearly right in the middle. The sides of the alley are lined
434  with derelict buildings and dim lighting.
435
436  You make out a familiar figure further ahead.
437  '''
438
439  enter = '''
440  You enter the back alley.
441  '''
442
443  [coastline]
444  enter = '''
445  You arrive at the city's coastline, the water is rather calm and still.
446  '''
447
448  [mainland_coastline]
449  enter = '''
450  You arrive at the coastline on the mainland,
```

```
the sea is crashing against rocks lined against it,
it is significantly colder out here away from the concrete jungle.
'''

[forest]
enter = '''
You wander through the forest, you hear birds whistling at the peaks of trees...
There's a river flowing through the middle with an old wooden bridge above it.
It would be quite easy to lose yourself here for a few hours.
'''

    # Yellow: \u001B[33m
    [forest.old_man]
    description = "An old man wandering through the forest"
    full = "There's nothing else you can give them."
    accept = "\u001B[33mOld Man\u001B[0m: Thank you for bringing him back to me."
    deny = "\u001B[33mOld Man\u001B[0m: I don't want your"

        [forest.old_man.small_talk]
        dialog = "\u001B[33mOld Man\u001B[0m: You haven't seen my cat anywhere have you?"
        option_1 = "I've seen this black stray around town..."
        option_q = "No."

        [forest.old_man.request]
        dialog = "\u001B[33mOld Man\u001B[0m: Could you bring him back to me?"
        option_1 = "Sure."
        option_q = "Sorry, not right now."

        [forest.old_man.praise]
        dialog = "\u001B[33mOld Man\u001B[0m: Thank you for helping me!"
        option_q = "[leave]"

[worm_hole]
enter = '''
You step into the worm hole...
'''

[medical_centre]
enter = '''
You're now at the Medical Centre's reception.
'''
```

```
492
493     # Red: \u001B[31m
494     [medical_centre.guard]
495     description = "Security guard stationed at the door"
496     blocking = "There is security watching the stairs, there's no way to get past them."
497
498         [medical_centre.guard.small_talk]
499         dialog = "\u001B[31mGuard\u001B[0m: What do you want?"
500         option_q = "Nevermind."
501
502 [medical_centre_office]
503 enter = '''
504 You find yourself at the Medical Centre's office.
505 You definitely shouldn't be here...
506 '''
507
508     [medical_centre_office.books]
509         [medical_centre_office.books.1]
510         title = "Alan Stern. Chasing New Horizons"
511         contents = """
512 On July 14, 2015, something amazing happened. More than 3 billion miles from
513 Earth, a small NASA spacecraft called New Horizons screamed past Pluto at more
514 than 32,000 miles per hour, focusing its instruments on the long mysterious icy
515 worlds of the Pluto system, and then, just as quickly, continued on its journey
516 out into the beyond.
517 """
518
519         [medical_centre_office.books.2] # yes
520         title = "Stack of research papers"
521         contents = """
522 Upon our latest series of tests, we have found the effect of \u001B[47m\u001B[37m[redacted]\u001B[0m has
523 considerable negative effects. These tests were performed over a series of 8
524 days and caused \u001B[47m\u001B[37m[rdctd]\u001B[0m and \u001B[47m\u001B[37m[redact]\u001B[0m in the body.
525
526 This document is confidential and should not be shared with third parties.
527 """
528
529         [medical_centre_office.books.3]
530         title = "Barry A. Burd. Java For Dummies"
531         contents = """
532 Java is the platform–independent, object–oriented programming language used for
```

```
  developing web and mobile applications. The revised version offers new
  functionality and features that have programmers excited, and this popular guide
  covers them all.

  You will learn how to program good in Java, like this:

  \u001B[36mimport \u001B[33mjava\u001B[37m.\u001B[33mio\u001B[37m.\u001B[33mBufferedReader\u001B[37m;
  \u001B[36mimport \u001B[33mjava\u001B[37m.\u001B[33mio\u001B[37m.\u001B[33mInputStreamReader\u001B[37m;
  \u001B[33mpublic class \u001B[31mMain  \u001B[34m{
      \u001B[33mpublic static  void \u001B[31mmain  \u001B[34m(\u001B[33mString[] args\u001B[34m) \u001B[34m{
          \u001B[33mtry{ String a=new \u001B[31mBufferedReader \u001B[34m(
              \u001B[33mnew \u001B[31mInputStreamReader\u001B[34m( System\u001B[37m.\u001B[33mout\u001B[37m \u001B[34m))
\u001B[37m.
                  \u001B[31mreadLine() \u001B[37m;
          \u001B[33mif
              \u001B[34m(a == \u001B[32m"hi"\u001B[34m) {
              System\u001B[37m.\u001B[33mout\u001B[37m.\u001B[31mprintln\u001B[34m(\u001B[32m"Hello!!"\u001B[34m)
\u001B[37m;
          \u001B[34m} \u001B[33melse
         \u001B[34m{
              System\u001B[37m.\u001B[33mout\u001B[37m.\u001B[31mprintln\u001B[34m(\u001B[32m"you did not greet
me!"\u001B[34m) \u001B[37m;
          \u001B[34m}
      } \u001B[33mcatch \u001B[34m(\u001B[33mException e\u001B[34m) {\u001B[35m/*something bad happen! */ \u001B[34m} } }
  \u001B[0m"""

          [medical_centre_office.books.4] # yes
          title = "Dr. Mike Ox. Introducing Impostors in Society"
          contents = """
  [Stamped for internal use only.]

  This book details methods which we can use to manipulate and control the general
  populous, I will be going into detail about ways we can subtly remove dangerous
  ideas and to keep your secrets under covers.
  """

          [medical_centre_office.books.5] # yes
          title = "RNA sequences"
          contents = """
  Confidential document

```

```
Figure 1.
GAGAATAAACTAGTATTCTTCTGGTCCCCACAGACTCAGAGAGAACCCGCCACCATGTTCGTGTTCCTGGTGCTGCTGCC
TCTGGTGTCCAGCCAGTGTGTGAACCTGACCACCAGAACACAGCTGCCTCCAGCCTACACCAACAGCTTTACCAGAGGCG
TGTACTACCCCGACAAGGTGTTCAGATCCAGCGTGCTGCACTCTACCCAGGACCTGTTCCTGCCTTTCTTCAGCAACGTG
ACCTGGTTCCACGCCATCCACGTGTCCGGCACCAATGGCACCAAGAGATTCGACAACCCCGTGCTGCCCTTCAACGACGG
GGTGTACTTTGCCAGCACCGAGAAGTCCAACATCATCAGAGGCTGGATCTTCGGCACCACACTGGACAGCAAGACCCAGA
GCCTGCTGATCGTGAACAACGCCACCAACGTGGTCATCAAAGTGTGCGAGTTCCAGTTCTGCAACGACCCCTTCCTGGGC
GTCTACTACCACAAGAACAACAAGAGCTGGATGGAAAGCGAGTTCCGGGTGTACAGCAGCGCCAACAACTGCACCTTCGA
GTACGTGTCCCAGCCTTTCCTGATGGACCTGGAAGGCAAGCAGGGCAACTTCAAGAACCTGCGCGAGTTCGTGTTTAAGA
ACATCGACGGCTACTTCAAGATCTACAGCAAGCACACCCCTATCAACCTCGTGCGGGATCTGCCTCAGGGCTTCTCTGCT
CTGGAACCCCTGGTGGATCTGCCCATCGGCATCAACATCACCCGGTTTCAGACACTGCTGGCCCTGCACAGAAGCTACCT
GACACCTGGCGATAGCAGCAGCGGATGGACAGCTGGTGCCGCCGCTTACTATGTGGGCTACCTGCAGCCTAGAACCTTCC
TGCTGAAGTACAACGAGAACGGCACCATCACCGACGCCGTGGATTGTGCTCTGGATCCTCTGAGCGAGACAAAGTGCACC
"""

        [medical_centre_office.books.6]
        title = "Jules Verne. Around the World in Eighty Days"
        contents = """
One ill-fated evening at the Reform Club, Phileas Fogg rashly bets his
companions £20,000 that he can travel around the entire globe in just eighty
days - and he is determined not to lose. Breaking the well-established routine
of his daily life, the reserved Englishman immediately sets off for Dover,
accompanied by his hot-blooded French manservant Passepartout. Travelling by
train, steamship, sailing boat, sledge and even elephant, they must overcome
storms, kidnapping, natural disaster, Sioux attacks and the dogged Inspector Fix
of Scotland Yard - who believes that Fogg has robbed the Bank of England - to
win the extraordinary wager.
"""

[marie]
description = "\u001B[35mMarie the Mink\u001B[0m"

    # Purple: \u001B[35m
    [marie.alley]

        [marie.alley.small_talk]
        dialog = "\u001B[35mMarie\u001B[0m: What brings you here?"
        option_1 = "Did you hear about the protests?"
        option_2 = "Sylvasta is up to something..."
        option_q = "Nevermind, have a good day."

```

```
612         [marie.alley.protests]
613         dialog = """\u001B[35mMarie\u001B[0m: Yeah, those Sylvasta guys are definitely up to
614 something but I don't know whether I believe the crowds showing up."""
615         option_1 = "Do you want to help me find out?"
616         option_q = "Nevermind."
617 
618         [marie.alley.sylvasta]
619         dialog = "\u001B[35mMarie\u001B[0m: You think I don't know? You want to help me find out?"
620         option_1 = "Sure."
621         option_q = "No thanks."
622 
623         [marie.alley.confirm]
624         dialog = "\u001B[35mMarie\u001B[0m: I suppose we could break in and find out the truth."
625         option_1 = "Let's do it! [progress story]"
626         option_q = "No way!"
627 
628         [marie.alley.recon]
629         dialog = """\u001B[35mMarie\u001B[0m: Ok, while I go figure out how we get in, I need you to fetch something we
630 can talk over, I'm sure the shop will have something, meet me back here when you
631 get something."""
632         option_q = "Got it."
633 
634         [marie.alley.waiting]
635         dialog = "\u001B[35mMarie\u001B[0m: Well? Go get it."
636         option_q = "Ok."
637 
638         [marie.alley.mission_brief]
639         dialog = """\u001B[35mMarie\u001B[0m: Ok so the plan is, you go to the Medical Centre, casually stroll in to
640 the reception, try to blend in though... Maybe find a chair to sit on or
641 something, either way, once you're there and you think security isn't looking,
642 use your comms device to let me know.
643 
644 \u001B[35mMarie\u001B[0m: From there, I will send a distraction and lead them into their breakroom
645 while you sneak into the office and try to find anything you can... although
646 once you're down there, let me know.
647 
648 \u001B[35mMarie\u001B[0m: If you want to, you can test the comms device now."""
649         option_q = "Ok."
650 
651     [marie.comms]
652     received = """\u001B[35mMarie\u001B[0m: Alright, I've received the documents, I'm sending them out far and wide,
```

```
good job on getting in and getting these."""
    bad_documents = "\u001B[35mMarie\u001B[0m: These don't look like the right documents..."
    no_access = "\u001B[31mAccess denied!\u001B[0m"

        [marie.comms.orientation]
        dialog = "\u001B[35mMarie\u001B[0m: What do you want?"
        option_1 = "Where do I go again?"
        option_2 = "Can you hear me?"
        option_q = "Nevermind."

        [marie.comms.directions]
        dialog = """\u001B[35mMarie\u001B[0m: You need to go to the Medical Centre, once you're there, try to blend in
and let me know once you're in position."""
        option_q = "Got it."

        [marie.comms.hear]
        dialog = "\u001B[35mMarie\u001B[0m: Loud and clear."
        option_q = "Neat."

        [marie.comms.complaint]
        dialog = "\u001B[35mMarie\u001B[0m: Try to blend in..."
        option_q = "Ok."

        [marie.comms.in_position]
        dialog = "\u001B[35mMarie\u001B[0m: Ok, I'll create a distraction, give me a moment..."
        option_1 = "[wait]"

        [marie.comms.distraction]
        dialog = """\u001B[31mGuard\u001B[0m: Who's there?
You better come out or we're going to have a problem.

(you see the guard go into the breakroom)
"""
        option_1 = "[wait]"

        [marie.comms.coast_is_clear]
        dialog = "\u001B[35mMarie\u001B[0m: Ok, now is your chance."
        option_q = "Got it."

        [marie.comms.gogogo]
        dialog = "\u001B[35mMarie\u001B[0m: What are you waiting for??"
```

```
694         option_q = "AAAAAAA"
695
696         [marie.comms.office]
697         dialog = "\u001B[35mMarie\u001B[0m: What do you see?"
698         option_1 = "A bunch of documents scattered around."
699
700         [marie.comms.documents]
701         dialog = """\u001B[35mMarie\u001B[0m: Ok, look through them and find anything that looks relevant, look out for
702 anything that mentions the stuff they were protesting about. Once you do, you'll
703 need to upload them to me, I guess go back to your apartments and we'll do it
704 from there, let me know once you're there.
705
706 \u001B[35mMarie\u001B[0m: I'll be making sure your cover is not blown.
707
708 \u001B[33mHint: you can use the documents to read them.\u001B[0m"""
709         option_q = "Ok."
710
711         [marie.comms.home]
712         dialog = """\u001B[35mMarie\u001B[0m: Presumably you have the documents, turn your computer on, open the
713 document scanner and upload them."""
714
715 [stage]
716 reached_conclusion = """\n
717 You've reached the end of the game, you're now in an open world mode, keep
718 exploring anything you haven't to maximise your score.
719
720 You can conclude by typing \u001B[47m\u001B[30mwin\u001B[0m
721 """
```

```toml
# 1. Marie located in Back Alley
[npc_marie]
    _prefix = "marie.alley."
    _start = "small_talk"
    small_talk = { description = "small_talk.dialog", options = [
        { description = "small_talk.option_1", to = "protests" },
        { description = "small_talk.option_2", to = "sylvasta" },
        { description = "small_talk.option_q", to = "small_talk", mustExit = true }
    ] }
    protests = { description = "protests.dialog", options = [
        { description = "protests.option_1", to = "confirm" },
        { description = "protests.option_q", to = "small_talk", mustExit = true }
    ] }
    sylvasta = { description = "sylvasta.dialog", options = [
        { description = "sylvasta.option_1", to = "confirm" },
        { description = "sylvasta.option_q", to = "small_talk", mustExit = true }
    ] }
    confirm = { description = "confirm.dialog", options = [
        { description = "confirm.option_q", to = "small_talk", mustExit = true }
    ] }

    recon = { description = "recon.dialog", options = [
        { description = "recon.option_q", to = "waiting", mustExit = true }
    ] }
    waiting = { description = "waiting.dialog", options = [
        { description = "waiting.option_q", to = "waiting", mustExit = true }
    ] }

    mission_brief = { description = "mission_brief.dialog", options = [
        { description = "mission_brief.option_q", to = "mission_brief", mustExit = true }
    ] }

# 2. Random City NPC located in the City Centre
[npc_city_centre]
    _prefix = "city_centre.npc."
    _start = "small_talk"
    small_talk = { description = "small_talk.dialog", options = [
        { description = "small_talk.option_1", to = "protestors" },
        { description = "small_talk.option_2", to = "enquire" },
        { description = "small_talk.option_q", to = "small_talk", mustExit = true }
    ] }
```

```
    protestors = { description = "protestors.dialog", options = [
        { description = "protestors.option_1", to = "leaflet" },
        { description = "small_talk.option_q", to = "small_talk", mustExit = true }
    ] }
    enquire = { description = "enquire.dialog", options = [
        { description = "protestors.option_1", to = "leaflet", },
        { description = "small_talk.option_q", to = "small_talk", mustExit = true }
    ] }
    leaflet = { description = "leaflet.dialog", options = [
        { description = "leaflet.option_1", to= "small_talk", mustExit = true }
    ] }
    #recon = { description = "testing", options = [
    #    { description = "susususus!", to = "recon", mustExit = true }
    #] }

# 3. Old Man located in Forest
[npc_old_man]
    _prefix = "forest.old_man."
    _start = "small_talk"
    small_talk = { description = "small_talk.dialog", options = [
        { description = "small_talk.option_1", to = "request" },
        { description = "small_talk.option_q", to = "small_talk", mustExit = true }
    ] }
    request = { description = "request.dialog", options = [
        { description = "request.option_1", to = "small_talk", mustExit = true },
        { description = "request.option_q", to = "small_talk", mustExit = true }
    ] }
    praise = { description = "praise.dialog", options = [
        { description = "praise.option_q", to = "praise", mustExit = true }
    ] }

# 4. Protestors located on Street
[npc_protestors]
    _prefix = "street.protestors."
    _start = "small_talk"
    small_talk = { description = "small_talk.dialog", options = [
        { description = "small_talk.option_1", to = "enquire" },
        { description = "small_talk.option_2", to = "small_talk", mustExit = true }
    ] }
    enquire = { description = "enquire.dialog", options = [
        { description = "enquire.option_1", to = "leaks" },
```

```
        { description = "enquire.option_2", to = "confrontation" }
    ] }
    leaks = { description = "leaks.dialog", options = [
        { description = "leaks.option_1", to = "small_talk", mustExit = true }
    ] }
    confrontation = { description = "confrontation.dialog", options = [
        { description = "confrontation.option_1", to = "small_talk", mustExit = true },
        { description = "confrontation.option_2", to = "small_talk", mustExit = true }
    ] }

# 5. Security guard stationed at Medical Centre
[npc_security_guard]
    _prefix = "medical_centre.guard."
    _start = "small_talk"
    small_talk = { description = "small_talk.dialog", options = [
        { description = "small_talk.option_q", to = "small_talk", mustExit = true }
    ] }

# 6. Shop shopkeeper in the Shop
[npc_shopkeeper]
    _prefix = "shop.npc."
    _start = "index"
    index = { description = "index", options = [
        { description = "leave", to = "index", mustExit = true }
    ] }
    recon = { description = "recon", options = [
        { description = "leave", to = "recon", mustExit = true }
    ] }
    stealth = { description = "stealth", options = [
        { description = "leave", to = "stealth", mustExit = true }
    ] }

# 7. Receptionist located in Apartments
[npc_receptionist]
    _prefix = "apartments.receptionist."
    _start = "first_encounter"
    first_encounter = { description = "first_encounter.dialog", options = [
        { description = "first_encounter.option_1", to = "protestors" },
        { description = "first_encounter.option_q", to = "repeated", mustExit = true }
    ] }
    protestors = { description = "protestors.dialog", options = [
```

```
        { description = "protestors.option_1", to = "protestors2" },
        { description = "protestors.option_q", to = "repeated" }
    ] }
    protestors2 = { description = "protestors2.dialog", options = [
        { description = "protestors2.option_q", to = "repeated" }
    ] }
    repeated = { description = "repeated.dialog", options = [
        { description = "repeated.option_1", to = "protestors" },
        { description = "repeated.option_q", to = "repeated", mustExit = true }
    ] }

# 8. TV located in Apartments
[home_tv]
    _prefix = "home.tv."
    _start = "first_on"
    first_on = { description = "first_on.dialog", options = [
        { description = "keep_watching", to = "dialog_a" },
        { description = "off", to = "first_on", mustExit = true }
    ] }
    dialog_a = { description = "first_on.dialog_a", options = [
        { description = "keep_watching", to = "dialog_b" },
        { description = "off", to = "first_on", mustExit = true }
    ] }
    dialog_b = { description = "first_on.dialog_b", options = [
        { description = "keep_watching", to = "dialog_c" },
        { description = "off", to = "first_on", mustExit = true }
    ] }
    dialog_c = { description = "first_on.dialog_c", options = [
        { description = "off", to = "first_on", mustExit = true }
    ] }

# 9. Communicator device with Marie
[comms_marie]
    _prefix = "marie.comms."
    _start = "orientation"

    orientation = { description = "orientation.dialog", options = [
        { description = "orientation.option_1", to = "directions" },
        { description = "orientation.option_2", to = "hear" },
        { description = "orientation.option_q", to = "orientation", mustExit = true }
    ] }
```

```toml
    directions = { description = "directions.dialog", options = [
        { description = "directions.option_q", to = "orientation" }
    ] }
    hear = { description = "hear.dialog", options = [
        { description = "hear.option_q", to = "orientation" }
    ] }

    complaint = { description = "complaint.dialog", options = [
        { description = "complaint.option_q", to = "complaint", mustExit = true }
    ] }

    in_position = { description = "in_position.dialog", options = [
        { description = "in_position.option_1", to = "distraction" }
    ] }
    distraction = { description = "distraction.dialog", options = [
        #{ description = "in_position.option_1", to = "coast_is_clear" }
    ] }
    coast_is_clear = { description = "coast_is_clear.dialog", options = [
        { description = "coast_is_clear.option_q", to = "gogogo", mustExit = true }
    ] }
    gogogo = { description = "gogogo.dialog", options = [
        { description = "gogogo.option_q", to = "gogogo", mustExit = true }
    ] }

    office = { description = "office.dialog", options = [
        { description = "office.option_1", to = "documents" }
    ] }
    documents = { description = "documents.dialog", options = [
        { description = "documents.option_q", to = "office", mustExit = true }
    ] }

    home = { description = "home.dialog", options = [
        { description = "home.option_q", to = "home", mustExit = true }
    ] }

# 10. Laptop
[entity_laptop]
    _prefix = "entities.laptop."
    _start = "boot"

    boot = { description = "boot.dialog", options = [
```

```
        { description = "boot.option_1", to = "home" }
    ] }
    home = { description = "home.dialog", options = [
        { description = "home.option_q", to = "boot", mustExit = true },
        { description = "home.option_1", to = "pictures" },
        { description = "home.option_2", to = "funny_cat_videos" },
        { description = "home.option_3", to = "document" }
    ] }
    pictures = { description = "pictures.dialog", options = [
        { description = "pictures.option_q", to = "home" }
    ] }
    document = { description = "document.dialog", options = [
        { description = "document.option_q", to = "home" }
    ] }
```

```toml
# Path to emoji root directory
emojis = "/emojis"

# Path to font file
# https://fonts.google.com/specimen/VT323?category=Monospace#standard-styles
font = "/VT323-Regular.ttf"
```