Game Design Document (GDD)

Text101

CONTENTS

What's a GDD?

Solo developers often rush-in and just start coding (code, compile, edit, repeat). Big games studios need much more planning, and run much longer loops (design, specify, implement). There are benefits to both, and as-such we are going to strike a balance by writing a simple Game Design Document which outlines our game and its screens.

This is a living document that we will update as necessary during the development, most commonly by drawing ourselves a diagram before we dive into a complex implementation.

A little bit of planning goes a long way, and can save us hours of re-working our code.

Description Of Game

This game is a very simple text adventure. The user will read the story and press a single key from a list of one or more. The computer then will display a new screen of text in response to this input. Early examples of such games include Zork[1] and Colossal Cave Adventure[2].

The story doesn't matter in a way. We will be using the example of a very simple prison escape scene. The important thing is that the user knows at every stage what keys they can press, and that the computer responds appropriately to those key presses until the game is over. At the end of the game the text will simply state this, and invite the player to play again.

This simple game will be implemented as a "finite state machine"[3]. This implementation is complete, and fine for a simple game. It has the advantage of making sure you are explicit

[1] http://en.wikipedia.org/wiki/Zork
[2] http://en.wikipedia.org/wiki/Colossal_Cave_Adventure
[3] http://en.wikipedia.org/wiki/Finite-state_machine

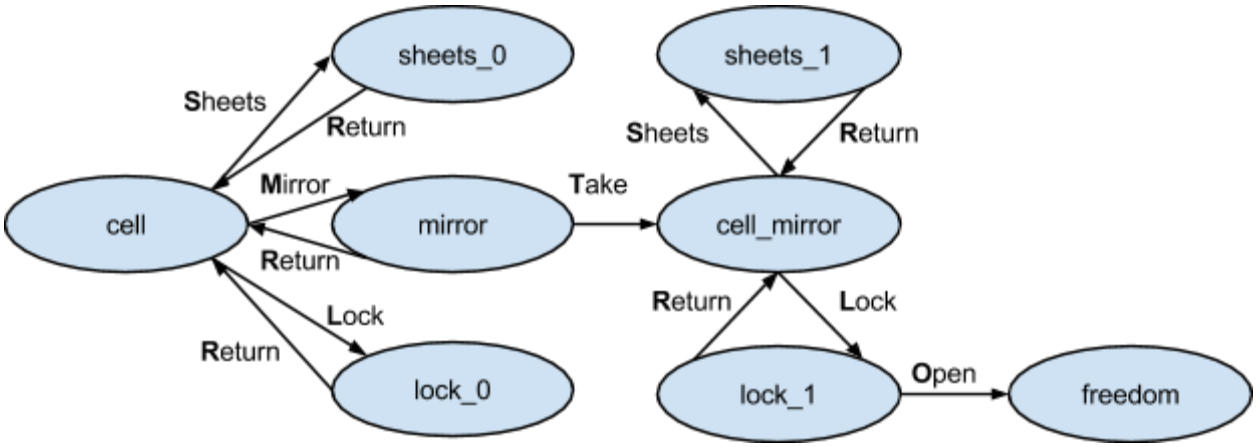
about all the possible pickles the player can get into, leading to a rich set of interactions. The disadvantage of this approach is that it quickly becomes impossible to keep track.

Our Story

In our simple story a prisoner starts in a cell. The only notable items are sheets on a bed (a red herring at this stage), a mirror on the wall (used to open the lock), and a locked door which can only be opened with the aid of the mirror.

Cell Scene

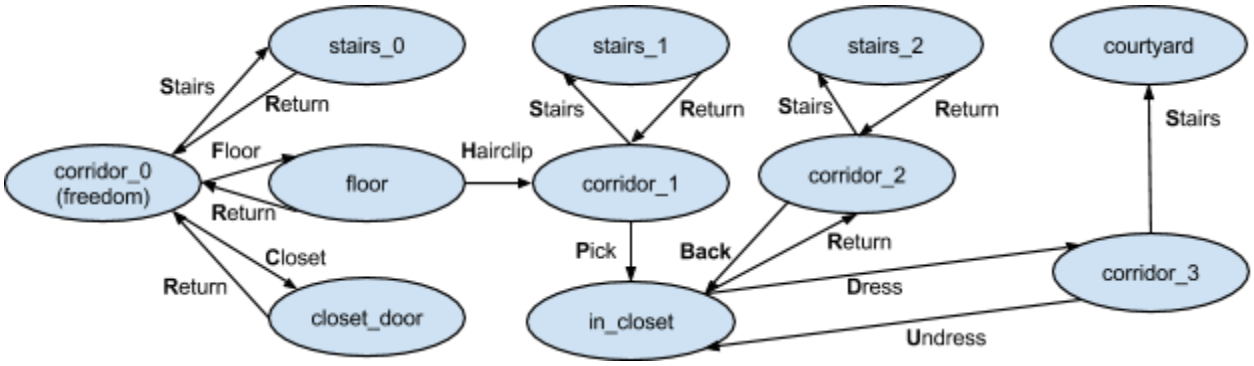
Here's the possible states the player can get into, and how they can move between them. **The first letter of each transition represents the key they would press.**



The way out of the cell is to press keys in the following order: M, T, L, O.

Corridor Scene

Now outside your cell, you will need to inspect the Floor, find a Hairclip, Pick the closet door lock, Dress up as a cleaner then climb the Stairs to freedom…



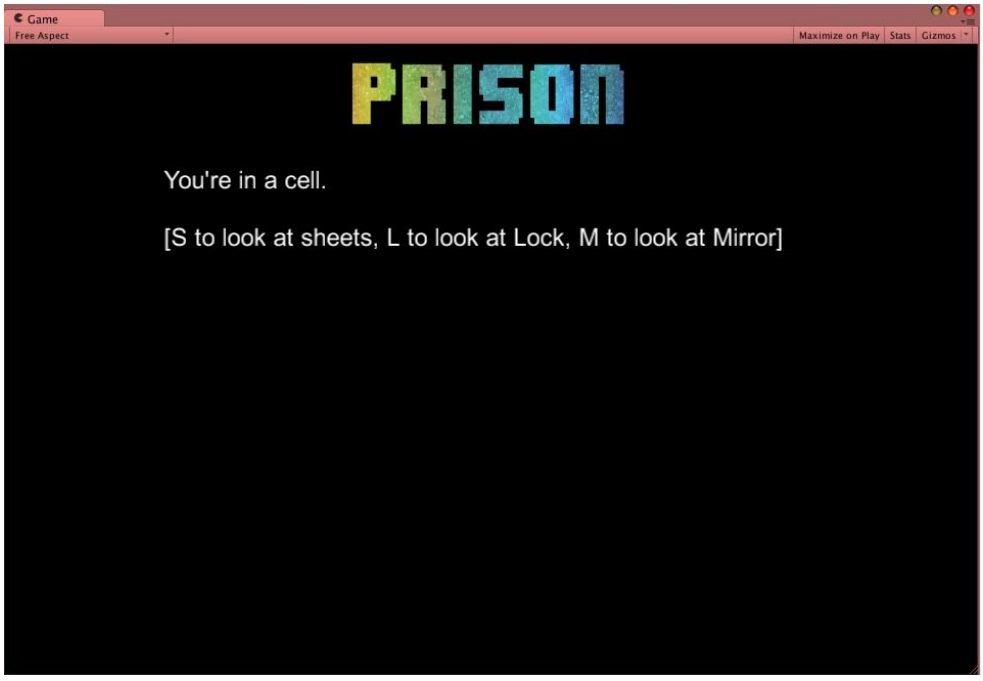
Each state is unique because of...

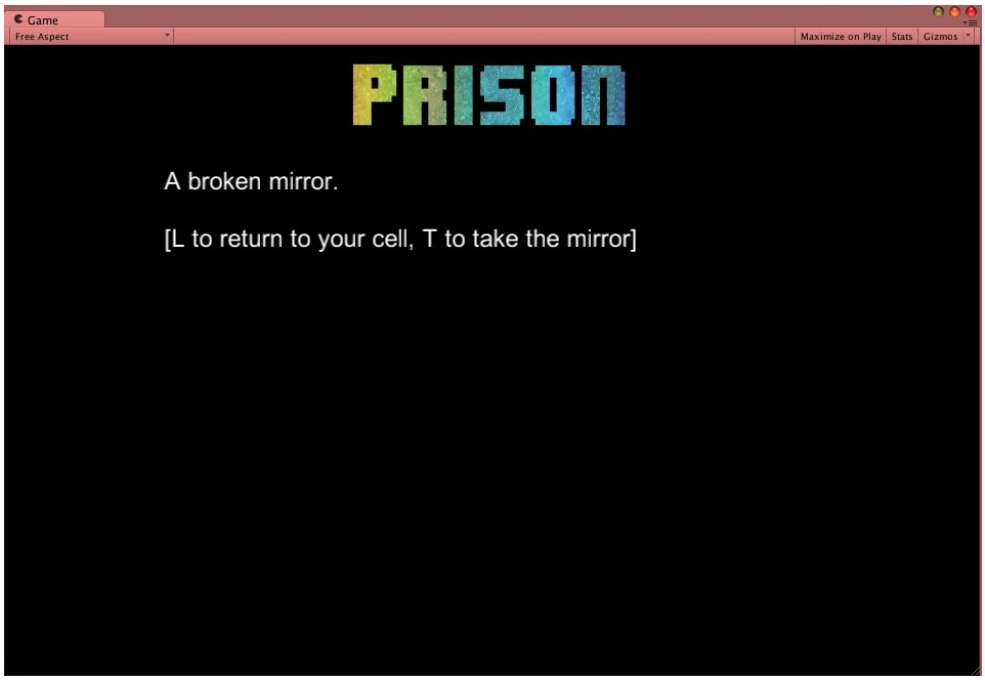
- **Description text:** even though you may choose to duplicate it, e.g. *sheets_0* and sheets_1.
- **Items being held:** for example *lock_0* where you have no mirror, vs *lock_1* where you do.
- **Path to this state:** *sheets_0* and *sheets_1* may have the same description, but they are still two different states. If we allowed *cell_mirror* to transition to *sheets_0* then we could get back to *cell*, then we would be offered the mirror again which makes no sense.
- **Key press options**

You can imagine how this could quickly get out-of-hand as the number of combinations increases, but it's great for such a trivial game.

Screen Mockups

Here is an example of what your game screen may look like. It consists of a simple image on a black background, and a single UI > Text element.





Now Create Your Own Story

There are several ways to approach this section…

1. Follow-us through, using our story at first. Then re-write with your own story.
2. Follow us through, creating your own story as you go.
3. Take on the challenge without us.