Critical Thieving Writeup

Dylan Bohlender, Joseph Lewis, Ali AlMarhoon

Components and Assets

Code was mostly written by team-members, with the exception of the Cocos2d-x engine. Some code is automatically generated from CSV files via Python scripts that were also written by the team. Additionally, small snippets of code were taken from StackExchange examples, all of that is commented in the source. Other portions were taken from examples online and translated to C++.

- Art Assets – all sprites made by Ali AlMarhoon; tiles obtained from http://www4.atpages.jp/kururumilk/
- Sound Assets – obtained from http://www.freesound.org and http://www.mediacollege.com/downloads/
- Python Scripts to translate CSV files to C++ code – written by Joseph Lewis
- Action CSV file – written by Dylan Bohlender & Joseph Lewis
- Sprite CSV file – written by Dylan Bohlender & Joseph Lewis
- Faker utility – adapted from Perl version by Joseph Lewis
- Cocos2d-x – engine utilized to run the game and instantiate game objects, perform transitions, and draw game objects like sprites
- Tiled – tile-based map generation engine used to hold the tile backgrounds and collision-relevant data
- Blackboard mechanism for game data passing – written by Joseph Lewis

Walkthrough of "Critical Thieving"

You are placed in the role of the Thief – a man in need of $100. Your objective? Get ahold of $100 by all means necessary before time runs out.

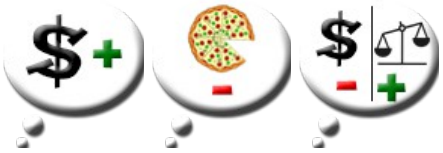
You have three major things to worry about: your money, your hunger, and your morality level, all shown in the HUD in the upper left corner. Based on your current levels of each of these, different actions will be available to you in the world. For instance, cultivating a higher level of morality may allow you to make off with more money from that bank than if you robbed it outright...

Critical Thieving Writeup

Dylan Bohlender, Joseph Lewis, Ali AlMarhoon

To take actions in the world, click something that interests you and the thief will pathfind to the location – once there, a menu will pop up listing all of the possible actions that can be taken, and who they would be applied to. Click to select an action, but be careful – some actions might not benefit your end goal!

When an action is taken, a sound will play and a bubble will come up over the actor's head that describes what just happened to the character (so players are well aware that they are taking actions and they are affecting the world). If an action affected one or two characteristics, the bubble will show + and – characters to describe what happened, as in the following examples:



Additionally, some actions are context-specific. Stealing that Barber's scissors may allow you to do many more things in the world, through our system of integrated Booleans. Part of the compelling nature of our project stems from allowing players to explore the game's state space and find optimal ways to win the game in as little time as possible. With over 30 actions built into the game, there are plenty of ways to interact with the environment.

Description of AI Behavior

AIs behave just as players do – except their actions are generated based on a combined utility function to make their behavior more interesting and less money-centric. AIs can take actions against other AIs, as well as against several sedentary objects that we have given actionable properties.

Three principal AI elements roam the game world, and they are as follows:

Policeman Bob

Bob is a rather unhappy fellow. He carries around a Tazer and has a knack for abusing his authority and lighting things on fire. If you haven't stolen $100 in 300 seconds, he will bust you and it will be game over!

Critical Thieving Writeup

Dylan Bohlender, Joseph Lewis, Ali AlMarhoon

Barber Joe

Barber Joe is known for his signature item: his scissors. He fancies giving people haircuts – but perhaps you can steal his scissors and benefit from taking some of his business!

Orange Guy

Orange guy is precisely that – orange. His actions are a bit mundane, but normal – he's known for giving hugs and punches in equal measure.

In addition to the AIs roaming the world, there is a ColaMatic Cola Machine, a Tree, a Bank, a Pizzeria, and a Mini Mart scattered about the world. Each of these entities may be interacted with, so stop by each of them and try to discover their secrets!

Description of AI Techniques

Three major pieces of AI were implemented in this system: an A* pathfinding system, a utility engine, and a semi-scripting environment.

The A* engine works by querying an invisible layer that is on top of the world to see if the tiles beneath are collidable or not and pathfinds around them. The data for whether or not a tile is collidable was implemented in the Tiled tile-map program, which allowed us to layer metadata onto simple sprites.

The utility-engine is in the style of the Sims, but limits itself to three characteristics: hunger, morality, and wealth. Each NPC searches for the action most beneficial to their net utility near them, pathfinds to it, and performs it. In the case of the player, the utility engine displays a list of potential actions, and allows the player to choose one of them.

The scripting environment happens through .csv files and Python scripts that process them and convert the descriptions of sprites and actions into C++ source code that is then compiled. This enabled us to write actions and sprite locations in a convenient spreadsheet format, and readily convert those actions to methods in the source code that can be called by the AI entities. While this isn't exactly run-time AI, it was incredibly helpful AI for coding productivity.

Critical Thieving Writeup

Dylan Bohlender, Joseph Lewis, Ali AlMarhoon

Two Things that Went Right

1. The team dynamics worked out well; everyone contributed where they fit best skill-wise, and the end result was a high level of synergy with much of the work for the project being done in parallel.
2. The major frameworks we used (Cocos2d & Tiled) enabled us to complete pretty fantastic things very quickly – we had maps and sprites generated and moving in the first week of the project.

Two Things that Went Wrong

1. We bit off far more than we could chew with our initial plan – we had to immensely reduce the scope of our game and our AI simulation, both for time reasons and for end-user complexity reasons. Our initial plan would have been really hard for a player to interact with meaningfully; we feel we've narrowed it down to the point that it's intelligible and intuitive.
2. Cross platform development between Mac OSX and Linux caused headaches – a change to the shared C++ class files would often fix the game on one platform and break it on the other. We certainly learned our fair share about code organization in this environment; it was truly necessary because we had to find out where things were going wrong!