201 FINAL PROJECT

BOWSER

by: Rui Zeng, Xingchen Liao, Blake Carroll, Jonathan Esquibel,

CSCI 201 Final Project

Group Member: Blake Carroll, Jt Esquibel, Xingchen Liao, Rui Zeng
Meeting Time: Tuesday, Thursday 12:00pm

Project Proposal:

We want to make a 2D Multiplayer Action RPG game in which players can connect via internet to take adventure together in a dungeon and fight enemies.

The map would be grid-based and separated by rooms in which enemies and terrains procedurally generated by the program. Players can walk from room to room, to collect items, fight enemies and level up.

There would be very minimal art assets involved, most likely just moving sprites without actual animation. We expect to implement some basic physics system like simple collisions. We expects to implement networking for allowing different players to play the game at the same time. We also want to use multi-thread for handling enemies and each player's action at the same time.

Basic Requirement:
a program with which users can interact. The software may cover any functional category (e.g., game, tool, travel, communication, etc.).

- has user login functionality and a central server to authenticate
- can interact with software as a guest but limited functionality compared to authenticated users.
    - Users must be able to register so they can become an authenticated user. The method by which they register is up to you.
    - When a user registers, the data should be stored in a database on the server.

- The program must incorporate graphics to make it more aesthetic.

- The program must have multi-threaded and network functionality outside of chatting.
- You can include chat, but there must be multi-threaded and networking functionality other than that as well.
- The program must be written in Java. It does not *have* to be a stand-alone Java
- application, so if you would like to adapt it to an Android device, that would be acceptable.

High-level Requirement:

GUILayout: briefly describe our user interface, including login interface and game interface

The GUI layout is going to be a grid layout that replicates a dungeon like surface. It will have images in each grid position to make it seem like a dungeon floor. Also there will be some grid positions that have images that replicate a wall in order to form the rooms. Scattered throughout the grid there will also be items that are images as well.

Players will be able to navigate throughout the grid based on either key commands or the will be able to use arrow buttons that are also in the GUI layout. The grid layout will go in the center of a border layout. On the east side of the border layout there will the buttons with arrows going North, South, East, and West.

Each room of a dungeon will be a different level that either one or two players can work their way through. The program will have about 10 - 15 levels that the user can play, each one getting a little bit harder.

How the levels work:
Throughout the grid there will be the items scattered about in rooms and also out in the open. There will be some items that are keys that players must get in order to open doors. Also some items will be weapons to be used against the monsters or food to regenerate a player's health.

Game System:

In our game, the two playable characters are demons with different magic abilities, the goal of players will be to get a princess out of a strictly guarded castle.

Characters' description:

Main characters:

Demons: the only playable characters in our game, have certain amount of Health Points, which will reduce when attacked by enemies and and Magic Points, which will reduce when using abilities; these points will recover in certain rate by time. Demons' normal attack will also cost magic points, it will appear as some small light bullets in the game.

Guards: we will have 5-7 kinds different guards in different levels, from the easiest to defeat to the hardest to defeat; they will be distributed **randomly** to different levels, so that means you meet the strongest one in the first level and meet the hardest one in the last level.

Heroes: we will have two heros in total and they will be distributed to 5th level, 9th level and the last level(12th level or 15th level according to the total number of our levels). Heroes will have certain special attack mode and harder to defeat than guards.

Prince: same as hero but harder to defeat.

Win condition: arrive the last level and defeat prince and escape the dungeon in limited time
Defeat condition: lost all of your Health Points, if you're defeated, you need to start over from the first level

Items:
defeat enemies will randomly generate useful items, there are three different items in this game, armor, weapon and drug.

Armors: some armors can increase character's' defense.
Weapons: some weapons can increase attack damage, and some of them can provide certain special effect to deal damage.
Potion: drugs will be simply used as HP and MP reversion

Player Interaction:

2 players can connect through internet to play the game. The game would have a multiplayer mode, which when player selected, would open a public or private room on the server and waiting for other players to join. Once 2 players all join the room, they can start the game. They would be cooperate together to fight enemies and walk through levels.

Players can interact with each other in the ways of exchanging items and reviving each other if one of them is dead.

Finishing Level:
Once all the enemies in a level is defeated, players are ready to move to the next one. They would do so by both standing in front of the door (grid) connecting to the next level.

Exchange Item:
When players are standing right next to each other, they could press a key(P), to exchange items. A inventory window would pop out with a exchange button. Player can select any item he has to pass to another player.

Technical Specification:

Game Structure:
Server Program

- Player Database

Client Program

- Login Window
- Start Game Window
    - Setting
    - Multiplayer Connection

In-Game GUI

- HUD
    - Inventory
    - Player Info Display
    - Setting
- Input & Player Control
- Object Rendering
    - Player
    - Enemy
    - Terrain

Algorithm

- Procedural Map Generation
- Enemy AI
- Collision Detection

Networking

- Exchanging Players Info
- Real-Time Players Action Update
- Saving & Loading Game Status

1. Pre-Game GUI Design

People in charge: Blake Carroll
Description: Please design a group of neat and polished pre-game GUI windows, including Login Windows, Game Windows and Setting Windows.
Detailed Design:

Login Window: we want the login window inside the game layout, so you don't need to create a popup login dialog. Instead use a part of the window as the login area, you may use another background color for this are to distinguish with the window's background. There should be two text fields for users to input usernames and password, and two buttons, one for Login and another for Exit.
Reference: http://i.ytimg.com/vi/J8yEwOgFbhI/maxresdefault.jpg

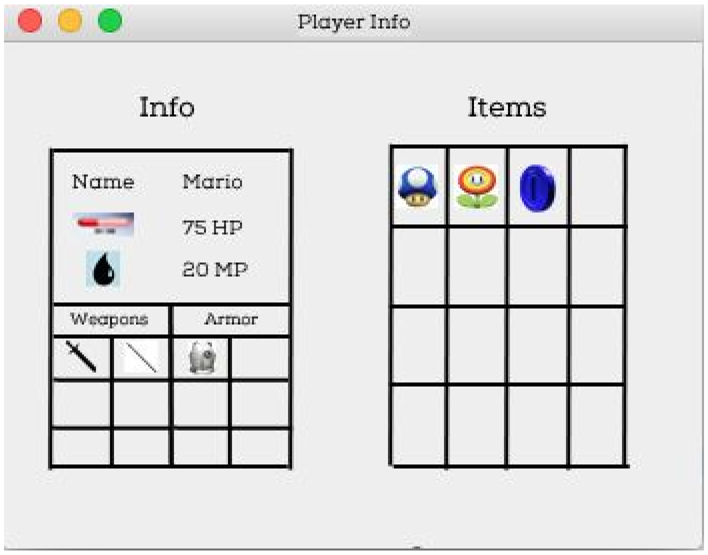
Game Window: after user logs in, the login window should disappears and there will be three buttons for play, setting and exit. The entire window should be well polished.

Setting Window: when player click on the setting button, there should popup a setting window in which player can set different settings the way they please

2. HUD

People in charge: Jt Esquibel
Description: Please design all the in-game GUI dialogs, including menu and player info-window.
Detailed Design:

Inventory and Player Info Window: an inventory will popup when player click on the "Info" in the menu or press the shortcut "I" on the keyboard. Inside the inventory window, on the right half part of the window, there will be a 4x4 grids which will be able to display different items. And when player move his/her mouse on the item, a small popup window will display to give detail information this item. On the left half part of the window, there will be two different areas. The top area will display player status, including player name, HP and MP. The bottom area will display player's weapon, armor and consumable items
Reference:http://media.indiedb.com/images/games/1/16/15535/Inventory_wip.png



3. Input & Player Control
Description: The player should be able to use keyboard to control his character. Such as arrow keys for movement and space key for attack. E for pick up items as well as interacting with other player, and I for open inventory and view character's information. As long as the arrow key is held down, the player should keep moving in that corresponding direction until hit an obstacle. Attack key should work in similar way.

4. Object Rendering
Description: When the game is playing, there should be sprites of different objects being real-time rendered on screen.
Terrain background will be rendered under other object, while each unit of the rest would take its own space and not overlapping with others. Depending on the team capacity, some moving objects could have simple 2D animation.
Objects are:

1. Player
2. Enemy
3. Terrain
4. Obstacle
5. Items

5. Procedural Map Generation
Description: Instead of premade map. We want the map to be procedurally generated each time the game runs. So we need an algorithm to generate the map in a random way while also maintain certain rules.

6. Enemy AI
Description: The enemy in the game should have simple AI. While in the room, if the player in within a certain distance with an enemy, it walk toward the player and try to

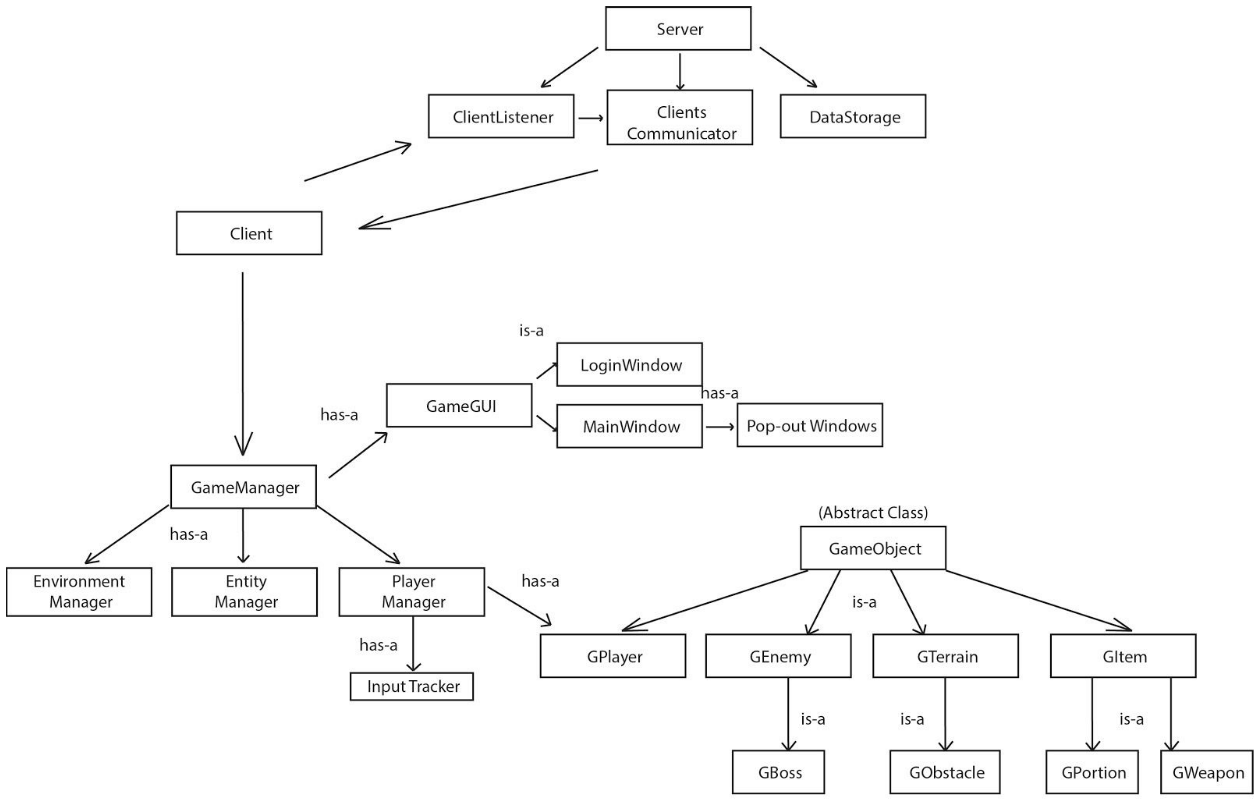
attack. If the player run farther away with a larger distance, the enemy would just walk back to its previous position.

7.Collision Detection
Description: We want to game to have simple 2D physics. The game should be able to detect if two object is colliding with each other at a given time. So it knows when to stop the player or enemy when they are hitting a wall. Also, when the player or enemy shoot out projectiles, the game would check if they hit the target.

8. Unit Class
Description: The player and the enemy share a lot similar properties, and should inherit from the same unit class. A unit object should have health, mana and position variables. It also has the function to perform attack, and die when the health run out.

9. Networking - Exchanging Information
Description: The game would have networking function to support multiple-player mode. When two players are both connected to the server and choose to play together. The client program on each player's computer would communicate information through a server. Through exchanging the information, the game should be able to update 2 players' position and actions they performed.

10. Database
Description: Players of our game can save their account the game progress in the database in the server. They will first register their account username and password. Then when they exit the game, they can save the progress they have made on the server, and load it next time they play. The saved file should contain the player's current HP and MP, the item they possess, as well as how many enemies he has defeated. The file would be saved under the user's account and would only be accessible by this user.



Detailed Design

Environment Manager:

- int level;
- int mapWidth, mapHeight;
- double timeSinceLevelLoad;
- void mapGeneration();
- void enemySpawn();
- void loadLevel();

Entity Manager:

- int numEnemy;
- void registerEnemy();
- void removeEnemy();
- void enemyStateSwitch();

Player Manager:

- int lifes;
- Point position;
- Point destination;
- bool isDead;
- enum direction;
- int health, mana;
- Item currentWeapon;
- Item currentArmor;

- Vector<Item> inventory;
- Vector<Item> getInventory();
- void addItem(Item it);
- void removeItem(Item it);
- void useItem(Item it);
- void respawn(Point position);

Input Tracker:
- enum inputType;
- inputType lastInput;
- void getLeftKey();
- void getRightKey();
- void getUpKey();
- void getDownKey()

GameObject
- protected instances:
  - int HP, MP
  - String gLabel
  - Image gImage
  - Rectangle renderBounds
- private instances: primitive location variables gameObject Class hold itselves
  - double x, y, gXScale, gYScale
  - int width, height
- void draw(Graphics, Point)
- void resize(double, double)
- void center(int)
- void changeX(double), changeY(double), changeHP(int), changeMP(int)
- int getX(), getY(), getHP(), getMP()
- void moveTowards(GameObject, double time)
- void animate()
- void onCollide(); //work as a listener
- void attack()

GPlayer
- overrides needed methods from GameObject
- void attack(); //by shooting particle towards enemies

GEnemy
- overrides needed methods from GameObject
- void attack(); //by shooting particle towards enemies
- void roaming();

Terrain

- void show();

Item
- void show();
- void removeItem();

GameGUI
- openMainWindow();
- openLoginWindow();

MainWindow
- void openLoginWindow();
- void openGameWindow();
- void openSettingWindow();
- void openInventoryWindow();
- void showGame();
- void createMenu();

LoginWindow
- JTextArea usernameArea;
- JTextArea passwordArea;
- JButton loginButton;
- JButton signupButton;

GameWindow
- JButton playButton;
- JButton settingButton;
- JButton exitButton;

SettingWindow
- JComboBox settingsBox;

InventoryWindow
- JLabel infoLabel;
- JLabel itemsLabel;
- JLabel nameLabel;
- ImageIcon hpImage;
- ImageIcon mpImage
- vector<ImageIcon> weaponsVector;
- vector<ImageIcon> armorVector;
- vector<ImageIcon> itemVector

Server
- ServerSocket ss;
- ServerListener sL;
- Server();
- sendServer() : void;

- listenForConnections() : void;

ServerGUI : JFrame
- long, serialVersionUID;
- JTextField : userNameTextField;
- JLabel : descriptionLabel;
- JLabel : usernameError;
- JButton : submitUsernameButton;
- initializeVariables() : void;
- createGUI() : void;
- getServerSocket() : ServerSocket;
- addActionAdapters(): void;

ServerListener : thread
- ServerSocket ss;
- Server server;
- ServerListener(Server) : void;
- sendServer() : void;
- run() : void;

ServerClientCommunicator : thread
- ServerSocket ss;
- ServerListener sL;
- sendServer() : void;
- run() : void;

Test document

Category

GUI Window	- There is a Login area works properly. - There is a menu containing different essential menu items which can work properly. - A game start window will appear after an user logs in. - An inventory and player status dialog will pop up and work properly when player clicks the corresponding menu item or "I" in the keyboard.
Graphics	- There will be different terrains in different levels. - Obstacles and normal terrains will be distinguished by different images - Enemies and player will be display eared by corresponding images on the game board.
Basic Animation	- A Enemies will move and chase player through a frequent animation.

	- The attack of player will be displayed through an animation
Gameplay	- Player can interact with another player - Player can pick up potions and armors by clicking ‘P’ on keyboard - Player can reduce enemies’ HP by shooting them and get rid of the enemies when their HP reduces to zero - Player will be able to move by uparrow, downarrow, leftarrow and rightarrow. - Player will be able to shoot by clicking ‘S’ on keyboard

Test1:
Run the program and you will see a 1000x800’s window
1. Window has a background image.
2. There is a login area distinguished with background image.
3. When you input correct username and password combination in the login dialog, you will log into the game.

Test2:
Get into the game window
1. There should be three buttons, START, EXIT, and SETTING.
2. The three buttons can work properly.

Test3:
Click Start and start the game
1. Window will display a well-polished dungeon and different terrains represented by different images.
2. Player will be able to move the character by up, down, left and right arrows in the keyboard.
3. Player can shoot by clicking ‘S’, and the bullet animation will play properly.
4. The inventory will pop up when player click ‘I’ or corresponding menu item in the menu.

Test4:
When player shoots an enemy
1. The bullet will disappear if it hits the enemy.
2. The bullet will keep flying until it reach the end of the map or an obstacle.
3. The enemy’s HP will disappear but will not display on the screen.

4. The enemy will die when its HP runs to zero.

Test5:
When player walks close to an enemy
1. The enemy will be activated and run towards to the player.
2. The path that the enemy chooses to chase player will be shortest path on the map.
3. The enemy will disappear when it hits the player and the player will lose certain amount of HP.

Test6:
When player clears one level and goes to the door to the next dungeon
1. The map will renew properly.
2. The enemy and item will renew properly.

Test7:
When two players play together
1. They won't collide with each other.
2. Animation and move should be the same as that of one player.

Deployment Document

Make sure you have Java installed on your computer. If you do not have it already, download a Java Development Kit (8 is the latest version) from the Java website, this will install Java on your computer.

Next, install Eclipse on your computer. Go to the Eclipse website if you do not already have i and download the latest version for your respective computer, and install it. You are finally ready to run our program on your computer.

To deploy our game within Eclipse, import the CSCI201GroupProject.zip file into Eclipse. This should generate a project called 201FinalProjectGame with metadata, bin, resource, and src directories.
To execute the project server, ProjectServer.java
To execute the client, run ProjectClient.java

The client will guide you to the initial Login Window, and the game will follow from there

Our database will most likely be a third party file that will store usernames and passwords, such as a .txt file