Boston University Adventure

Issue & Solution Documentation [Opal]

**Client**

**Graphics**

Issue: How is the map rendered? [C.G.000]

Solution: The map is rendered by the GameMap’s render function [C.G.001] & the UIEngine’s render function. [C.G.002]

Issue: How often is the map rendered/what triggers a render? [C.G.010]

Solution: The map is rendered according to CoreGameLogic’s renderLoop function [C.G.011], which is called on each iteration of the infinite loop in Main. [C.G.012]

Issue: How does the client know where to render characters on the map? [C.G.020]

Solution: Characters, known as “Actors” are represented by animated sprites (AnimatedSprite.java). ActorEngine.java stores a list of all actors, and Actor.java stores data for the actor and contains its render function. In other words, each character is rendered individually on top of the map [C.G.021].

**Networking**

Issue: How does the client join a game? [C.N.000]

Solution: A client joins a game by entering their name as well as the server’s IP address at a screen generated by LoginScreen.java. Once this information is submitted [C.N.001], a NetworkEngine is created and it uses a NetworkStreamWriter to send a login request. [C.N.002].

Issue: How does the client receive the map? [C.N.010]

Solution: The client receives the map via the network in packets with an opcode of MI for map image and MD for map data. These are processed in NetworkEngine [C.N.011], and then depending on the opcode, the appropriate retrieve function is called from the NetworkStreamParser, getMapData() [C.N.012] or getMapImage() [C.N.013].

Issue: How does the client know when the game starts?

Solution: \*\*\*STILL NEED SOLUTION\*\*\*

Issue: How does the client tell the server it wants to move? [C.N.020]

Solution: As the client’s coreGameLogic’s renderLoop runs, it checks keyboard input and processes it with processInput [C.N.021], which in turn calls the NetworkEngine’s NetworkStreamWriter’s sendActorMove. [C.N.022]

Issue:

Solution:

**User Interface**

Issue: How does the client setup the user interface? [C.U.000]

Solution: The user interface is setup first by the CoreGameLogic class based upon the state of the game [C.U.001], resulting in handing UI setup to one of three classes:

LoginScreen.java (LOGIN\_STATE) creates a background [C.U.001], input fields with labels [C.U.002] and a login button [C.U.003], and waits for it to be pushed [

LoadingScreen.java (LOADING\_STATE)

UIEngine.java (INGAME\_STATE)

Issue: How are key press events detected and processed?

Solution: See solution to issue [C.N.020].

Issue:

Solution:

**Server**

**Networking**

Issue: How do clients join a game? [S.N.000]

Solution: Clients join a game by requesting a connection to the server and then sending a “login request” packet. Main recognizes this incoming connection and establishes a socket [S.N.001] and then establishes a ClientHandler thread for each client. [S.N.002]

Issue: How is the map transmitted? [S.N.010]

Solution: After the server receives a login request from a client [S.N.011] it uses a NetworkStreamWriter to send both the map image [S.N.012] and data [S.N.013] files.

Issue: How are collisions detected?

Solution:

Issue:

Solution:

Issue:

Solution:

**Game Logic**

Issue: How do non player characters move? [S.L.000]

Solution: Non player characters move based on the AIEngine and the NPCEngine \*\*\*NEED MORE\*\*\*

Issue: How are the contents of a manhole decided? [S.L.010]

Solution: The contents of a manhole are decided based on a probability calculation performed each time there is an interaction with a manhole. \*\*\*CODE CITATION NEEDED\*\*\*

Issue: How are collisions (interactions) between objects detected? [S.L.020]

Solution:

Issue: How does the server start a game?

Solution:

Issue: How does the server end a game?

Solution:

**Data Storage**

Issue: How are players represented? [S.D.000]

Solution: Players are represented by the PlayerCharacter class, which extends Actor. The ActorEngine maintains a list of all Actors.

Issue: How are professors/viruses represented? [S.D.010]

Solution: Professors and viruses are represented by the NonPlayerCharacter class, which extends Actor. The ActorEngine maintains a list of all actors.

Issue: How are eating establishments represented?

Solution: See solution to issue [S.D.010]

Issue:

Solution:

Issue:

Solution:

Issue:

Solution: