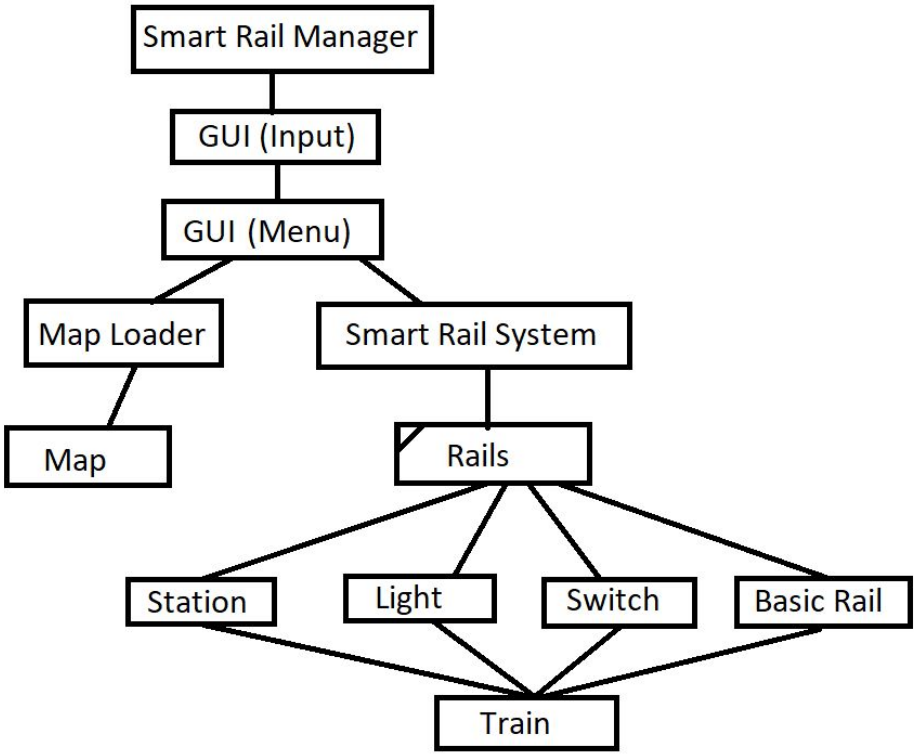
Design Document: Smart Rails



Rail Class: Rail class is the parent to all other types of rail. Rail class is abstract and has 6 abstract methods which are implemented in the children. The methods in Rail are:

Run: run method is called when thread starts. Loops until thread is stopped, Checks if the thread has a message that it needs to process and processes it.

Getter/setters: Rail class has getters and setters for the left/right rails, alternate path(if rail is a switch), and the train.

securePath: methods sets a secure variable to the id of the train it is securing path for.

giveTrain: giveTrain returns the current train that is on the rail. It also calls a helped method to change the Image view of the train, sets the secured train back to empty and sets the train on the rail to empty.

Light Class: Light class is a child to Rail. Light class implements 6 abstract methods from Rail which are:

PassTrain: PassTrain method gets the train from either the left or right rail and then sends a message to the next rail that notifies it that it has the train.

CheckPoint: CheckPoint method check if the rail has the correct train on it. If it does it gives the train the checkpoint message. If not it passes the method to the next rail.

FoundPath: FoundPath method adds a direction to the direction list then just passes the message to the next rail since it is not a station.

RequestPath: RequestPath method just passes the message to the next rail since it is not a station.

InvalidPath: InvalidPath does the same as requestPath and just passes the message to the next rail since it is not a station.

SecurePath: SecurePath method waits until the rail can secure a path for a train. Then once the path is secured it sends a checkpoint message to the rail is received the message from and sends the secure path message to the next rail.

BasicRail Class: BasicRail class is a child to Rail. This class implements 6 abstract methods from Rail which are:

PassTrain: Does the same thing as the Light class does.

CheckPoint: Does the Same thing as the Light class does.

SecurePath: This method waits until it can secure the path then passes the message to the next rail.

InvalidPath:Does the same thing as RequestPath.

FoundPath:Does the same thing as the Light class does.

RequestPath: Does the same thing as the Light class does.

Switch Class: Switch class is a child to Rail. Switch class has an alternate rail. This class implements 6 abstract methods from Rail which are:

PassTrain: Checks the direction of the message and gets the train from the correct rail. Then passes the message to the next correct rail.

CheckPoint: Does the Same thing as the Light class does.

SecurePath: This method does the same as BasicRail class.

InvalidPath:Does the same thing as RequestPath.

FoundPath: Creates a new message to send to both paths if there are two possible path to send the message, adds a direction to list to each message for both possible paths.

RequestPath: Does the same thing as the Light class does.

Station Class: Station class is a child to Rail. Switch class has an alternate rail. This class implements 6 abstract methods from Rail which are:

PassTrain: Checks it station is the starting station or a end station. If it is a starting station it tells the next rail it has a train that is ready to move. If it is a end station then it gets the train from the correct rail. It then deletes the train from the rail system to make room for the next train that may want to go to it.

CheckPoint: Does the same as Light class. Just added a check to see if the message goes left or right.

SecurePath: Does the same as the Light Class.

InvalidPath: Checks if it has the train the message is meant for. If it does it give the train the message.

FoundPath: Does the same as InvalidPath.

RequestPath: Checks if station is the starting station or potentially the end station. If it is the starting station then it just passes the message to the next rail over. If it is the correct end station it creates a FoundPath message to send back and also adds the first direction to the direction list

Train Class: Train Class is the train object that is passed from rail to rail.

Train constructor takes an id and ImageView for the train.

Train has a Getter/Setter for the current rail it is on.

NoPath is called from current rail when it receives a invalid path message. The method check is all the attempts are invalid paths if so tells current rail to clear the train.

requestFindPath creates a RequestPath message and sends it to the current rail.

givePath method is called when train receives a valid path that works. Train then checks if it has received all possible paths. If it has then it calls requestToSecurePath.

requestToSecurePath method creates a Secure Path message with a randomly chosen valid path and sends to current rail.

The last methods give checkpoints to the train and also remove checkpoints from the train.

InboxMessage Class: This Class is the message object passed from rail to rail. This class has message types in the form of Enums which are RequestPath, FoundPath, InvalidPath, SecurePath, CheckPoint, and PassTrain. This class has getters and setters for end station, train, path list, direction of message, start Station, type of message, number of paths, train ID, and a checkpoint. Not all getters and setters are used for each type of message.

TextFileReader Class: Takes in the file name and reads the text file into a list that represents the map being read.

FileToList(String fileName): reads in the text file and converts it a list of strings.

RailsInializer Class: This class takes a read in a text file of a rail map and converts the specified map into the appropriate rail pieces in the appropriate places. Besides the logic of initializing the map in the method initializeMap, it also has getters for specific info about the map such as its dimensions, if the map is a valid map and separate lists for all rails and only stations.

initializeMap(String mapFile): calls TextFileReader to read a specific map, converts that text file to rails sets any switches with their partner and gets metadata on the map.

MapLoader Class: Reads in all the potential maps in a Maps file. Returns them in an ArrayList of strings. Used in the SmartRailsGUI to read all the available map files in a folder and puts them in a list.

listOfMaps(final File folder): reads all the file names in the input path and returns a list of file titles.

SmartRailsGUI Class: Is the GUI for SmartRails to allow a user visualization of the SmartRails state and allow them to easily put input into the SmartRails instance. Has a toolbarSelector, which is initialized on start, and below the toolbar with inputs and outputs.

sendTrainCombos(): Initializes the toolbar for when the toolbar selector has send Train selected. Loads the combo box for departure and destination stations as well as a send button.

loadMapComobos(ArrayList<String> list): loads the toolbar to allow the user to select a map to load onto the screen.

loadImages(): Loads in the textures for the GUI display to SmartRails rails.

loadGridBack(): Loads up the appropriate textures in the appropriate order to represent the SmartRails rail states.

findStation(): Finds the appropriate station to the input coordinate