Testing Document

Samir Lavingia, Alexei Naumann, Jeff Kang, Ian Whelan, Aidan Blant

JSONs Parser

This class has two main methods, pullTheNewData and ParseTheData, which as their names suggest pull the data from the online json file and parse it into useful data.

parseTheData(){}

This method parses the currentData.JSON file and updates the car data

Three catches

Malformed Exception: JSON file is not appropriately formatted

Check JSON file for proper formatting

FileNotFoundException: File is not Present

1) File either does not exist or is in incorrect location

Fixes: Ensure file has not been deleted by another method

Correct the location and run solely this method again

Alter method to run with input string with hard coded location

```
pullTheNewData(){
        String toRead = "C:\User\Myself\Desktop\toRead.JSON"
        .....
        ....( toRead );
]
```

2) Could be related to pullTheNewData method not updating currentData.json

- Make sure pullTheNewData does not delete old method or replace it if Exception is caught, detailed below

IOException: Other misc exception with IO

Review JSON if variables are of incorrect type, use Java library methods to check if is of appropriate type or throw an exception so problem can be more accurately located.

private void pulltheNewData(){}

This method accesses JSON file located at specified URL, and replace the previous "currentData.json" file that parser accesses.

Three catches

MalformedURLException:

Check if URL is incorrectly formatted, search where URL might either be input incorrectly or edited by another method (should not be as it is instantiated and accessed within method).

```
            FileNotFoundException:
                    1) File either does not exist or is in incorrect location
                    Fixes:  Ensure file has not been deleted by another method
                                    Correct the location and run solely this method again
                                    Alter method to run with input string with hard coded location
                            parseTheData(){
                                    String toRead =
                                    .....
                                    ....( toRead );
                            ]

            IOException:
                    1)
```

Freeway
Freeway class holds Ramp and Waypoint vectors, with a method 'addFreewayPoints()

```
  public void addFreewayPoints(){};
        Parses waypoint and ramp file matching the freeway

        Three catches:
            NumberFormatException:
            FileNotFoundException:
                    1) File either does not exist or is in incorrect location
                    Fixes:  Ensure file has not been deleted by another method
                                    Correct the location and run solely this method again
            IOException:
                    1)
        testFreewayPoints(){
            for( each freeway ){
                    for( freeway.waypoins.length() ){
                            System.out.println(
                                    Freeway.waypoint[i].xLocation,
                                    Freeway.waypoint[i].yLocation,
                            );
                    }
                    for( freeway.ramps.length ){
                            System.out.println(
                                            Freeway.ramps[i].xLocation,
                                            Freeway.ramps[i].yLocation,
                                            Freeway.ramps[i].orientation,
                                            Freeway.ramps[i].name
                    }
            }
        }
```

This lets us know if information is being handed to other methods correctly, also if individual variables or whole sets of data are inaccurate. Some variables might be out of range of map (xLocation and yLocation), possible mismatch through parsing that testing parsing doesn't catch.

```
private void setTheCurrentXandYs(){}
```

This calls the draw Car method for every highway, issues should be traced back along the separate methods it calls upon, including but not limited to drawCarsOnFreeway(int freeway_num, Car car), Car.setWaypoints(Waypoint w1), Car.setRamps(Ramp r1)

```
public synchronized CarDot drawCarOnTheMap( double speed, double xLoc, double yLoc){}
                CarDot currentDot = new CarDot(speed, Color.BLACK, xLocation, yLocation);
                map().addMapMarker(currentDot);
                return currentDot;
```

Blah blah blah blah, this returns cardot so a test can test it...
code and stuff

```
    private void addFreewayPoints(){}
        I(Freeway).addFreeWayPoints();
        Ramp.setCurrentID(0);
            Repeats x5 for each freeway

        CarDot currentDot = new CarDot(speed, Color.BLACK, xLocation, yLocation);
        map().addMapMarker(currentDot);
        return currentDot;

        private void testTheFreewayPoints(){
            for(int i = 0; i < I101.waypoints.size(); i++){
                drawTheRampOnTheMap( I101.waypoints.get(i).getxLocation(),
I101.waypoints.get(i).getyLocation() );
            }
            for(int i = 0; i < I405.waypoints.size(); i++){
                drawTheRampOnTheMap( I405.waypoints.get(i).getxLocation(),
I405.waypoints.get(i).getyLocation() );
            }
            for(int i = 0; i < I10.waypoints.size(); i++){
                drawTheRampOnTheMap( I10.waypoints.get(i).getxLocation(),
I10.waypoints.get(i).getyLocation() );
            }
            for(int i = 0; i < I105.waypoints.size(); i++){
              drawTheRampOnTheMap( I105.waypoints.get(i).getxLocation(),
I105.waypoints.get(i).getyLocation() );
              }
        }

    private void drawTheRampOnTheMap(Double xLocation, Double yLocation){
        map().addMapMarker( new RampDot( xLocation, yLocation ) );
    }
```

This goes into either input of Rampdot (incorrect locations parsed), the addMapMarker Method within map (part of GUI interface and API), or testing of drawing Methods as outlined below.

The Various Drawing Methods

This is more as it calls upon different methods and different classes. In short method of Testing will initialize by simply using draw Methods at hard-coded locations to test that methods work the way we desire, if they do it is a short matter of constructing methods to call upon these drawing methods with our specified parameters. If drawing fails it is more likely to occur due to incorrect parameters, in which case it can be traced back all the way to the input file. The values the parameters call upon can be output at various points along the flow of the data to ensure the intended values are being passed forward.