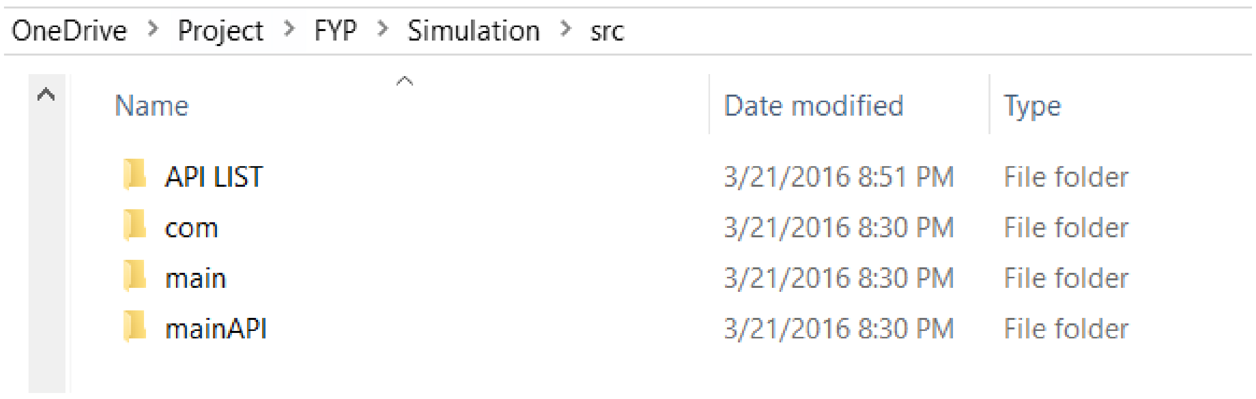
2016-3-21

Simulator Manual

Wang Bowen
Nanyang Technological University

How to do the simulation

1. Import all the Source Code into java workspace:



OneDrive > Project > FYP > Simulation > src

Name	Date modified	Type
API LIST	3/21/2016 8:51 PM	File folder
com	3/21/2016 8:30 PM	File folder
main	3/21/2016 8:30 PM	File folder
mainAPI	3/21/2016 8:30 PM	File folder

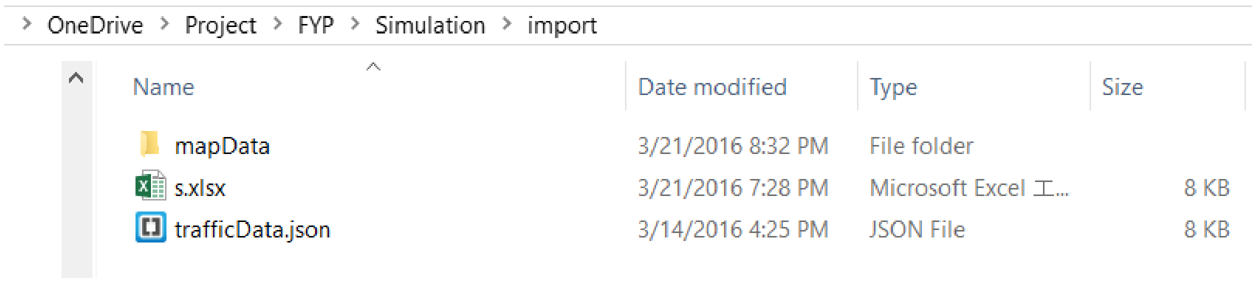
"com/graphhopper/api" Folder:	Overwrite the Graphhoper function so that the users are able to run the route function offline.
Main Folder:	Simulator.java and shortestDistanceStrategy is an example. Write own simulator class with setting of simulator Write own strategy class extending the strategy class in main API
Main API Folder	Main class for the simulator, user can call the function required from simulator.java class
Import external library:	xmlbeans-2.6.0.jar poi-ooxml-schemas-3.14-20160307.jar

	poi-ooxml-3.14-20160307 poi-excelant-3.14-20160307.jar poi-3.14-20160307.jar : handle the excel file writing and export function. graphhopper-web-0.5.0-with-dep.jar: Graphhopper API jxmapviewer2-2.0.jar: Jxmapview API okhttp-2.7.1.jar, okio-1.4.0.jar: Internet connection for the Jxmapview and Graphhopper e

2. Download the OSM file for the offline routing:

https://www.openstreetmap.org/export#map=12/1.3406/103.8847

3. Set import file in the root file, here the s.xlsx is the station file

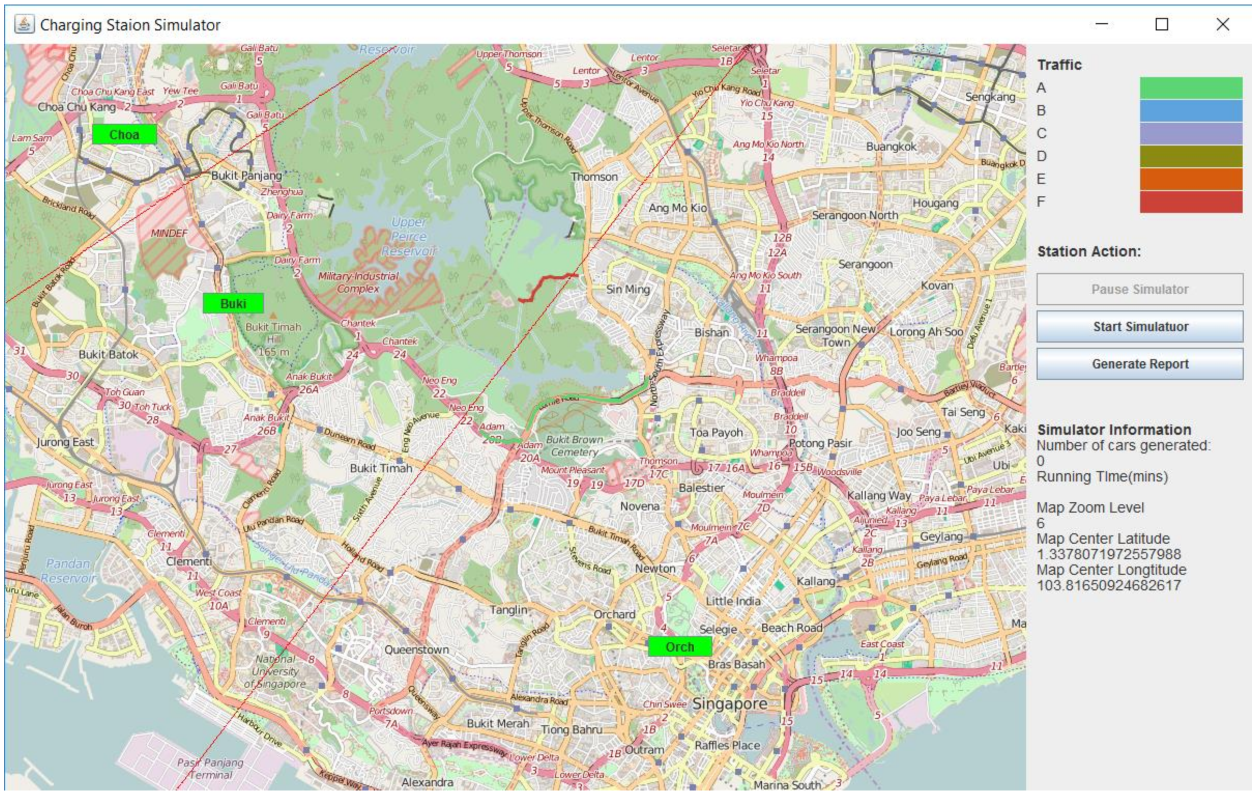


> OneDrive > Project > FYP > Simulation > import

Name	Date modified	Type	Size
mapData	3/21/2016 8:32 PM	File folder	
s.xlsx	3/21/2016 7:28 PM	Microsoft Excel 工...	8 KB
trafficData.json	3/14/2016 4:25 PM	JSON File	8 KB

4. Implement the Strategy Class following the API menu
5. Modify the source code of Simulator to do simulator setting

6. Compile and run the simulation



Traffic

A	🟩
B	🟦
C	🟪
D	🟫
E	🟧
F	🟥

Station Action:

Pause Simulator

Start Simulatuor

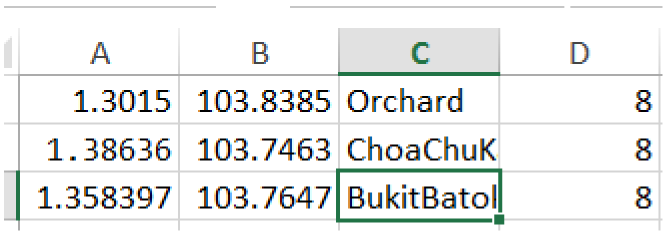
Generate Report

Simulator Information
Number of cars generated:
0
Running TIme(mins)

Map Zoom Level
6
Map Center Latitude
1.3378071972557988
Map Center Longtitude
103.81650924682617

7. Collect result

Station Excel Format:



A	B	C	D
1.3015	103.8385	Orchard	8
1.38636	103.7463	ChoaChuK	8
1.358397	103.7647	BukitBatol	8

Sample Code

```
package main;
import java.awt.Point;
import com.graphhopper.GHResponse;
import mainAPI.Simulator;
import mainAPI.Strategy;
public class ShortestDistanceStrategy extends Strategy {

	@Override
	public int setChargingTime(Point generationPoint) {
		// TODO Auto-generated method stub
		return 30;
	}

	public String chooseDestination(Point generationPoint){
		String zoneID=super.getZoneID(generationPoint);
		int carNo=super.getCarNo(zoneID);//real time car number within the zone
		System.out.println("Zone ID "+ super.getZoneID(generationPoint)+" Car No 
"+carNo);
		GHResponse shortestDistanceRes=null;
		String selectedStation=null;
		for (String myVal : Simulator.getStations().keySet()) {
			GHResponse res=super.getInfomration(generationPoint, myVal);
			if(res==null)
			{

			}
			else if(shortestDistanceRes==null)
			{
				double averageCapacity=super.getRouteCapacity(res);
				shortestDistanceRes=res;
				selectedStation=myVal;
				continue;
			}
			else
			{
				double averageCapacity=super.getRouteCapacity(res);
				System.out.println(averageCapacity);

				if(res.getDistance()<shortestDistanceRes.getDistance())
				{
					shortestDistanceRes=res;
					selectedStation=myVal;
				}
			}
		}
		return selectedStation;
	}
}
```

Two methods to generate station

```
package main;
import java.util.Random;

import javax.swing.JFrame;
public class Simulator {
	public static void main(String[] args) {
		double latitude=1.3378071972557988;
		double longitude=103.81650924682617;
		int zoomLevel=6;
		mainAPI.Simulator s= new
mainAPI.Simulator(latitude,longitude ,zoomLevel);
		s.setSize(1024 + 200, 768);
		s.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		s.setVisible(true);
		s.setTimeMapping(1,900);//1min=20miliseconds
		//https://mapzen.com/data/metro-extracts can get OSM data here,
noted: first time using the OSM file it needs more time for extracting the map
data
		s.setMapOSM("mapData"+"\\"+"singapore.osm");
		s.integrateTraffic("trafficData.json");
		s.addStation(1.3015 ,103.8385 ,"Orchard",8);
		s.addStation(1.38636,103.74632,"ChoaChuKang",8);
		s.addStation(1.358397, 103.764725 ,"BukitBatok",8);
		s.addStation(1.355357, 103.867871 ,"Seragoon",8);
		s.addStation(1.37521888903652, 103.84401798248291,"AngMokio",8);
		s.addStation(1.3151968719906737,
103.76599788665771,"Clementi",8);
		s.addStation(1.333682, 103.853703,"ToaPayoh",8);
		s.addStation(1.320857, 103.842432,"Novena",8);
		s.addStation(1.306932, 103.818884,"Tanglin",8);

		s.addStation(1.283896, 103.843464,"ChinaTown",8);

		s.addStation(1.294166, 103.786127,"CityHall",8);
		String
zoneArrayCenter[]={"Tanglin","Orchard","Novena","CityHall"};
		String
zoneArray[]={"ChoaChuKang","BukitBatok","Seragoon","AngMokio","Clementi","ToaP
ayoh","ChinaTown"};
		ShortestDistanceStrategy sDs=new ShortestDistanceStrategy();
		s.addRandomCarGenerater(sDs,5,4,1280, zoneArrayCenter);
		s.addRandomCarGenerater(sDs,5,4,1120, zoneArray);
		s.draw();
	}
}
```

```
package main;
import java.io.IOException;
import java.util.Random;

import javax.swing.JFrame;

import mainAPI.ReadExcelFile;
public class Simulator {
	public static void main(String[] args) {
		double latitude=1.3378071972557988;
		double longitude=103.81650924682617;
		int zoomLevel=6;
		mainAPI.Simulator s= new
mainAPI.Simulator(latitude,longitude ,zoomLevel);
		s.setSize(1024 + 200, 768);
		s.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		s.setVisible(true);
		s.setTimeMapping(1,900);//1min=20miliseconds
		//https://mapzen.com/data/metro-extracts can get OSM data here, noted:
first time using the OSM file it needs more time for extracting the map data

		s.setMapOSM("import\\mapData"+"\\"+"singapore.osm");
		s.integrateTraffic("import\\trafficData.json");

		s.setStations("import\\s.xlsx");
		String zoneArrayCenter[]={"Tanglin","Orchard","Novena","CityHall"};
		String
zoneArray[]={"ChoaChuKang","BukitBatok","Seragoon","AngMokio","Clementi","ToaPayoh","
ChinaTown"};
		ShortestDistanceStrategy sDs=new ShortestDistanceStrategy();
		s.addRandomCarGenerater(sDs,5,4,1280, zoneArrayCenter);
		s.addRandomCarGenerater(sDs,5,4,1120, zoneArray);
		s.draw();
	}
}
```

Strategy Class:

- ***Method Detail***

- **setChargingTime**

```
public abstract int setChargingTime(java.awt.Point generationPoint)
```

Set charging time for the car

Parameters:
`generationPoint - the point car generated`

Returns:
`charging time`

- **chooseDestination**

```
public
abstract java.lang.String chooseDestination(java.awt.Point generationPoint
)
```

Choose the destination station to charge the car

Parameters:
    `generationPoint` - the point car generated

Returns:
    the ID of the destination(a Charging station ID)

- **getPointGelocation**

```
public org.jxmapviewer.viewer.GeoPosition getPointGelocation(java.awt.Poin
t generationPoint)
```

Parameters:
    `generationPoint` - a point

Returns:
    GeoPosition

- **getZoneID**

```
public java.lang.String getZoneID(java.awt.Point generationPoint)
```

Get the zone id for the point on the map

Parameters:
    `generationPoint` - point on the map(x and y coordinate, left bottom is 0,0)

Returns:
    Zone ID(a station ID string)

- **getCarNo**

```
public int getCarNo(java.lang.String zoneID)
```

get number of cars within the zone(at the time function calls)

Parameters:
    `zoneID` - a stationID

Returns:
    number of cars

- **getInfomration**

-
  ```
  public com.graphhopper.GHResponse getInfomration(java.awt.Point generation
  Point,

  java.lang.String chargingStation)
  ```

get route information such as distance, time from generation point to the charging station

Parameters:

    generationPoint - point on the map that car generate

    chargingStation - ID of charging station

Returns:

    GHResponse Graphhoper API

- **getRouteCapacity**

```
public double getRouteCapacity(com.graphhopper.GHResponse res)
```

Get Cpacity of the route(veh/h)

Parameters:

    res - response

Returns:

    average capacity

Json File Example

```
[
  {
    "id": "1",
    "geometry":  {
      "paths": [
        [


          [
            1.360856399555275,
            103.82404838854183
          ],
                [1.360793814676862,103.82107542055269]
            ]
        ]
    },
    "value": 40,
    "value_type": "speed",
    "mode": "REPLACE"
},
{
    "id": "1",
    "geometry": {
        "paths": [
            [
                [
                    1.336434280186183,
                    103.809814453125
                ],
                [
                    1.3363152709846813,
                    103.80970593050874
                ],
[
  1.3354270086160378,
  103.81068055653603
],
[
  1.3347901702252813,
  103.81142989869632
],
[
  1.3345552906667133,
  103.81178827163104
],
[
  1.334332518301886,
  103.81217067268872
],
[
  1.3341501653377004,
  103.81256052432717
],
[
    1.3337470889183973,
    103.81466922494775
],
[
    1.3336774259882591,
    103.81510228995462
],
[
    1.3336774259882591,
    103.81523248885347
],
[
    1.333709649750061,
    103.81555509900052
],
[
    1.3337599411702143,
    103.81572571730001
],
[
    1.33383016289391,
    103.81584120130184
],
[
    1.333880268049544,
    103.81599170303326
],
[
    1.3339203149211478,
    103.81623254305644
],
[
    1.3338953554755901,
    103.81651864535776
],
[
    1.3338050171838332,
    103.81687999852478
],
[
    1.3337899297577873,
    103.8170606751083
```

```
],
[
  1.33383016289391,
  103.81720111855569
],
[
  1.333961665644385,
  103.8173886869264
],
[
  1.3340669050976688,
  103.81748684832797
],
[
  1.3343382925019778,
  103.81773979554488
],
[
  1.3352094516577449,
  103.8182717670114
],
```

[
    1.335776254589325,
    103.81850590151188
],
[
    1.3370538429257388,
    103.8190417845333
],
[
    1.3379173652362235,
    103.81936849249978
],
[
    1.3387862892177618,
    103.81976635351255
],
[
    1.339114114771354,
    103.81995261803164
],
[

    1.3393571899687615,
    103.82014130398947
  ],
  [
    1.3395743743980164,
    103.82037264452218
  ],
  [
    1.3397539333944157,
    103.82060230867421
  ],
  [
    1.3398777992996083,
    103.8208412860522
  ],
  [
    1.3402385936730785,
    103.82176925588628
  ],
  [
    1.3405654879040754,

```
    103.82275273254706
],
[
    1.3407334985002912,
    103.82333164267239
],
[
    1.3410803230348303,
    103.82438031191484
],
[
    1.3411652596555337,
    103.82466827686135
],
[
    1.3412162961337633,
    103.82495307531103
],
[
    1.3412498237471988,
    103.82529487070356
```

],
[
    1.341256529269886,
    103.82564579305752
],
[
    1.341161534365152,
    103.82797298195898
],
[
    1.3410888912027081,
    103.82842839870814
],
[
    1.340851776469911,
    103.82960354155907
],
[
    1.3407161759000161,
    103.83048848428923
],

```
[
    1.3406450228537252,
    103.83104355255611
],
[
    1.3404816688704864,
    103.83222838116203
],
[
    1.34044832752157,
    103.83285124971385
],
[
    1.3404760809349139,
    103.83327835425612
],
[
    1.3405315877616015,
    103.83363654092632
],
[
          1.3406569437829468,
          103.83404297010696
        ],
        [
          1.3409426735552252,
          103.83462653684526
        ],
        [
          1.3415143193643015,
          103.83557182927963
        ],
        [
          1.3418739961506574,
          103.83621332428336
        ],
        [
          1.3420233065842553,
          103.83650832550367
        ]
      ]
    ]
```

```
        },
        "value": 20,
        "value_type": "speed",
        "mode": "REPLACE"
    }
]
```