

# Documentation

Andrew id: yiq

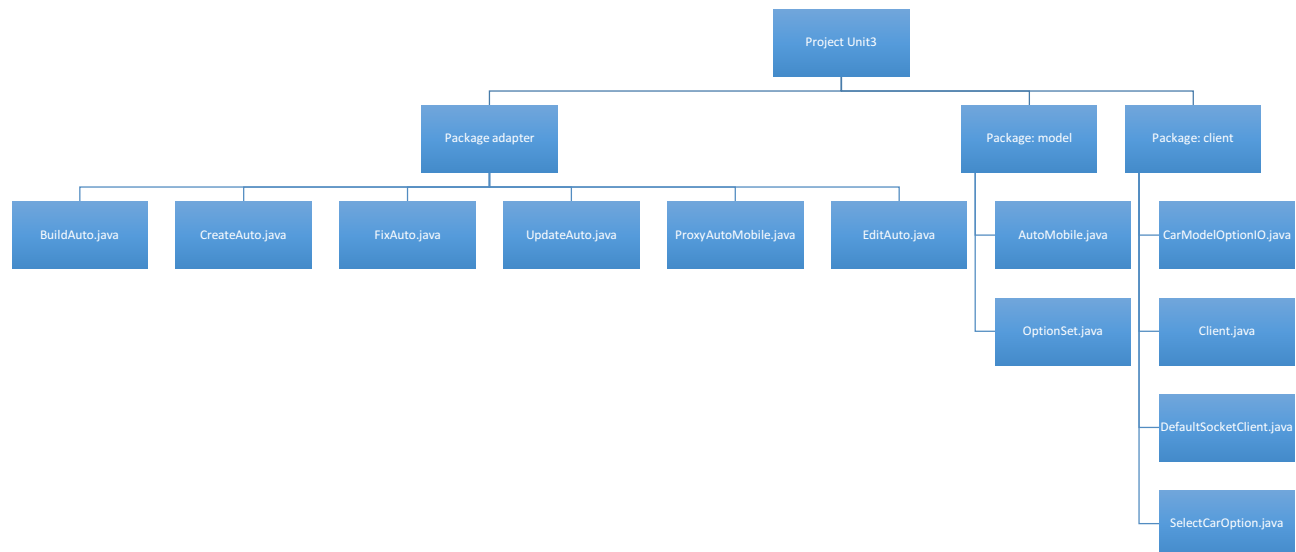
## Attention:

This documentation is for the project 1 unit 4 of the course 18641 in CMU. Design details and test details are included.

## 1: Client Design Details

In this part, I add scale package named client. It includes four classes file: CarModelOptionIO.java, Client.java, DefaultSocketClient.java and SelectCarOption.java as required in project handout.

The following diagram is the package and class relationship in client part. For your convenience, I only draw out the associated packages in the part. Other packages such as exception won't be included.

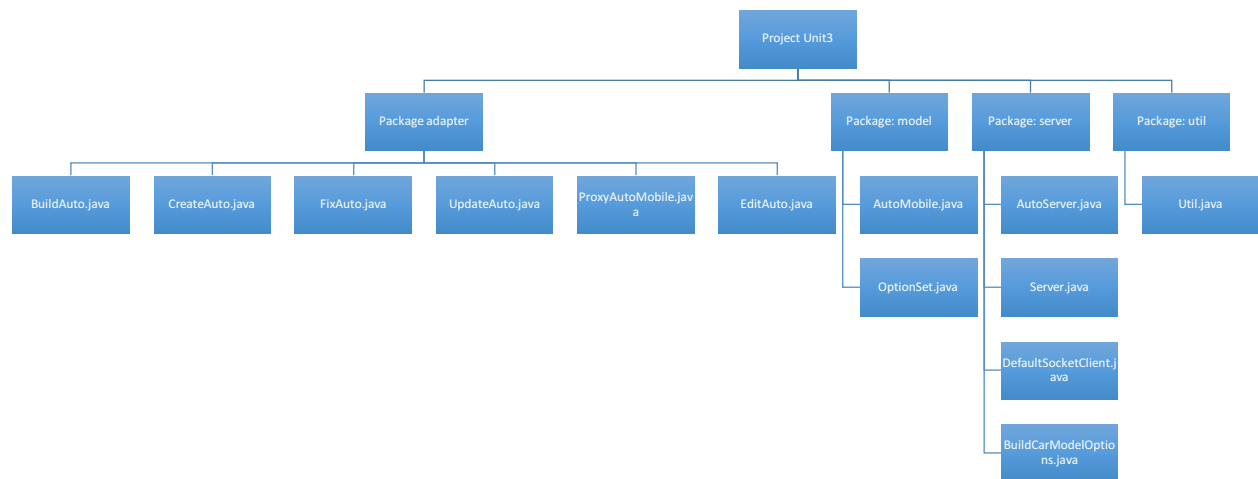


In summary, Client.java is just a wrapper and all specific implementations are done in DefaultSocketClient.java. CarModelOptionIO.java is to read all car model file names from a directory named fileSets.txt in this project. SelectionOption.java is to customize a car configuration when user choose a specific car model.

## 2: Server Design Details

In this part, I add scale package named server. It includes four classes file: AutoServer.java (interface), Server.java, DefaultSocketClient.java and BuildCarModelOptions.java as required in project handout.

The following diagram is the package and class relationship in server part. For your convenience, I only draw out the associated packages in the part. Other packages such as exception won't be included.



In summary, Server.java is just a wrapper and all specific implementations are done in DefaultSocketClient.java. AutoServer.java is an interface and it supports three methods that can make server can handle different input file types as required in handout. BuildModelOption.java implements AutoServer.java interface and call three methods that are specifically defined in ProxyAutoMobile as required in handout.

Just like what is demonstrated above, ProxyAutoMobile.java should implement AutoServer.java interface and give specific definition of these three methods and be called in BuildAutoOption.java.

Also in package util, I add a method called readAutoFromProperty() in Util.java to upload car model information with input argument is properties file type.

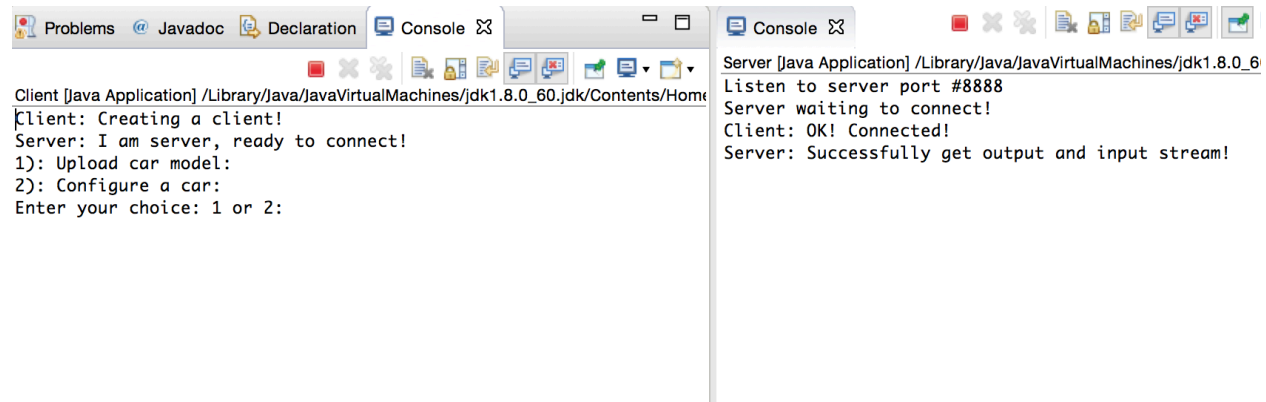
All client and server design details are strictly following project handout.

### 3: Test details:

In this part, I test client and server communication by uploading four car models information and customize two cars for user.

Please read following instruction carefully and for each step I will give my test snapshot:

1: First build connection between server and client.

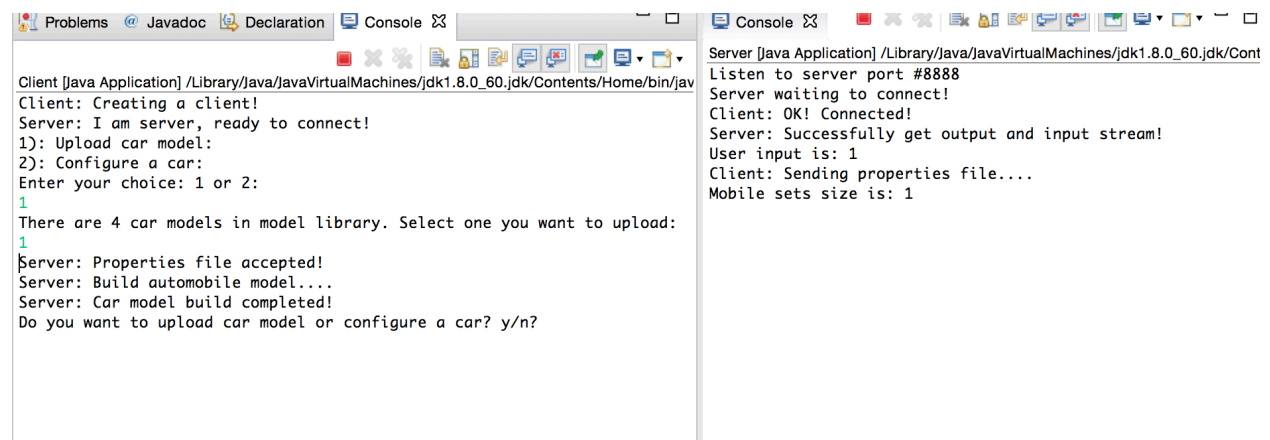


The screenshot shows two IDE console windows. The left window is titled 'Client [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0\_60.jdk/Contents/Home' and contains the following text: 'Client: Creating a client!', 'Server: I am server, ready to connect!', '1): Upload car model:', '2): Configure a car:', and 'Enter your choice: 1 or 2:'. The right window is titled 'Server [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0\_60.jdk/Contents/Home' and contains the following text: 'Listen to server port #8888', 'Server waiting to connect!', 'Client: OK! Connected!', and 'Server: Successfully get output and input stream!'.

In this step, I firstly run Server.java and Client.java (the order can not be reversed.) Then from the two consoles information (left one is the client console and right one is the server console), I know that server and client connect successfully.

Then upon the client console information, next step is to ask user to input a choice, here since no car models have been uploaded so I first upload car models. So, go to step 2.

2: Upload car models



The screenshot shows two IDE console windows. The left window is titled 'Client [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0\_60.jdk/Contents/Home/bin/jav' and contains the following text: 'Client: Creating a client!', 'Server: I am server, ready to connect!', '1): Upload car model:', '2): Configure a car:', 'Enter your choice: 1 or 2:', '1', 'There are 4 car models in model library. Select one you want to upload:', '1', 'Server: Properties file accepted!', 'Server: Build automobile model....', 'Server: Car model build completed!', and 'Do you want to upload car model or configure a car? y/n?'. The right window is titled 'Server [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0\_60.jdk/Contents/Home' and contains the following text: 'Listen to server port #8888', 'Server waiting to connect!', 'Client: OK! Connected!', 'Server: Successfully get output and input stream!', 'User input is: 1', 'Client: Sending properties file....', and 'Mobile sets size is: 1'.

In this step, I input 1 to upload a car model, from the information, I know that there are 4 car models in the model library. So I first choose the first one to upload.

Then we can see from the client console, server send information to tell the client car model has been successfully uploaded.

From the server console, we know that the mobile sets size is increasing indicates that the car model has been uploaded successfully. Then I continue input y and digit to upload all these 4 car models into the mobile sets.

In this part, another part should be explained.

```
Client [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Contents/Home/bin/jav
Client: Creating a client!
Server: I am server, ready to connect!
1): Upload car model:
2): Configure a car:
Enter your choice: 1 or 2:
1
There are 4 car models in model library. Select one you want to upload:
1
Server: Properties file accepted!
Server: Build automobile model....
Server: Car model build completed!
Do you want to upload car model or configure a car? y/n?
y
Client: Creating a client!
Server: I am server, ready to connect!
1): Upload car model:
2): Configure a car:
Enter your choice: 1 or 2:
1
There are 4 car models in model library. Select one you want to upload:
2
Server: Properties file accepted!
Server: Build automobile model....
Server: Car model build completed!
Do you want to upload car model or configure a car? y/n?

Server [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Conte
Listen to server port #8888
Server waiting to connect!
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 1
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 2
```

Now I upload two car models, then continue to upload next car models. Finally, we can get:

```
Client [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Contents/Home/bin/jav
Do you want to upload car model or configure a car? y/n?
y
Client: Creating a client!
Server: I am server, ready to connect!
1): Upload car model:
2): Configure a car:
Enter your choice: 1 or 2:
1
There are 4 car models in model library. Select one you want to upload:
3
Server: Properties file accepted!
Server: Build automobile model....
Server: Car model build completed!
Do you want to upload car model or configure a car? y/n?
y
Client: Creating a client!
Server: I am server, ready to connect!
1): Upload car model:
2): Configure a car:
Enter your choice: 1 or 2:
1
There are 4 car models in model library. Select one you want to upload:
4
Server: Txt file accepted!
Server: Build automobile model....
Server: Car model build completed!
Do you want to upload car model or configure a car? y/n?

Server [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/C
Listen to server port #8888
Server waiting to connect!
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 1
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 2
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 3
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending txt file....
Accepting data....
Data Accepted!
Mobile sets size is: 4
```

As you can see from the 4 input, this is a txt file not a properties file, so some different information is printed. After upload all car models, now I try to customize user's car. See below:

```
Client [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Contents/Home/bin/jav
Do you want to upload car model or configure a car? y/n?
y
Client: Creating a client!
Server: I am server, ready to connect!
1): Upload car model:
2): Configure a car:
Enter your choice: 1 or 2:
1
There are 4 car models in model library. Select one you want to upload:
4
Server: Txt file accepted!
Server: Build automobile model....
Server: Car model build completed!
Do you want to upload car model or configure a car? y/n?
y
Client: Creating a client!
Server: I am server, ready to connect!
1): Upload car model:
2): Configure a car:
Enter your choice: 1 or 2:
2
Available car model:
Focus Wagon ZTW Base price is: 18445.0
Copper Mini Copper Base price is: 15000.0
Toyota Crola Base price is: 20000.0
BMW X5 Base price is: 30000.0
Input your wanted car model:

Server [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Cont
Listen to server port #8888
Server waiting to connect!
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 1
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 2
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 3
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending txt file....
Accepting data....
Data Accepted!
Mobile sets size is: 4
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 2
```

From the client console, we can know that there are 4 available car models, indicates that the previous steps success. Then go to step 3:

### 3: Customize two cars and print their information

Firstly, I choose (capital x here) X5 model, then the client console will print out X5 model information:

```
Client [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Contents/Home/bin/jav
X5
Model received!
Here is the model information:
The following information is model basic information:

# Car model: X5
# Base price: $30000.00

The following information is model additional information:

#Color:
Fort Knox Gold Clearcoat Metallic: $0.00
Liquid Grey Clearcoat Metallic: $0.00
Infra-Red Clearcoat: $0.00
Grabber Green Clearcoat Metallic: $0.00
Sangria Red Clearcoat Metallic: $0.00
French Blue Clearcoat Metallic: $0.00
Twilight Blue Clearcoat Metallic: $0.00
CD Silver Clearcoat Metallic: $0.00
Pitch Black Clearcoat: $0.00
Cloud 9 White Clearcoat: $0.00

#Transmission:
automatic: $0.00
manual: $-815.00

#Brakers:
Standard: $0.00
ABS: $400.00
ABS with Advance Trac: $1625.00

#Side Impact Air Bags:
present: $350.00
not present: $0.00

#Power Moonroof:
present: $595.00
not present: $0.00

Server [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Co
Listen to server port #8888
Server waiting to connect!
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 1
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Server: Fail to upload car model, press y to upload again!
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 2
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 3
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending txt file....
Accepting data....
Data Accepted!
Mobile sets size is: 4
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 2
User choose car model X5
```

Then the client console prompts that press any key to continue to customize your own car:

```
Client [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Contents/Home/bin/jav
Sangria Red Clearcoat Metallic: $0.00
French Blue Clearcoat Metallic: $0.00
Twilight Blue Clearcoat Metallic: $0.00
CD Silver Clearcoat Metallic: $0.00
Pitch Black Clearcoat: $0.00
Cloud 9 White Clearcoat: $0.00

#Transmission:
automatic: $0.00
manual: $-815.00

#Brakers:
Standard: $0.00
ABS: $400.00
ABS with Advance Trac: $1625.00

#Side Impact Air Bags:
present: $350.00
not present: $0.00

#Power Moonroof:
present: $595.00
not present: $0.00

-----
Press any key to set up your own car!

Please choose the Color
1. Fort Knox Gold Clearcoat Metallic. Price: $0.0
2. Liquid Grey Clearcoat Metallic. Price: $0.0
3. Infra-Red Clearcoat. Price: $0.0
4. Grabber Green Clearcoat Metallic. Price: $0.0
5. Sangria Red Clearcoat Metallic. Price: $0.0
6. French Blue Clearcoat Metallic. Price: $0.0
7. Twilight Blue Clearcoat Metallic. Price: $0.0
8. CD Silver Clearcoat Metallic. Price: $0.0
9. Pitch Black Clearcoat. Price: $0.0
10. Cloud 9 White Clearcoat. Price: $0.0

Server [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk
Listen to server port #8888
Server waiting to connect!
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 1
Client: Sending properties file....
Mobile sets size is: 2
Client: Sending properties file....
Mobile sets size is: 3
Client: Sending txt file....
Accepting data....
Data Accepted!
Mobile sets size is: 4
The car model has been added
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 2
User choose car model X5
```

Then the client console asks for Color selection and I randomly choose 4 (digit required):

```
Problems Javadoc Declaration Console
Client [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Contents/Home/bin/jav
Pitch Black Clearcoat: $0.00
Cloud 9 White Clearcoat: $0.00

#Transmission:
automatic: $0.00
manual: $-815.00

#Brakers:
Standard: $0.00
ABS: $400.00
ABS with Advance Trac: $1625.00

#Side Impact Air Bags:
present: $350.00
not present: $0.00

#Power Moonroof:
present: $595.00
not present: $0.00

-----
Press any key to set up your own car!

Please choose the Color
1. Fort Knox Gold Clearcoat Metallic. Price: $0.0
2. Liquid Grey Clearcoat Metallic. Price: $0.0
3. Infra-Red Clearcoat. Price: $0.0
4. Grabber Green Clearcoat Metallic. Price: $0.0
5. Sangria Red Clearcoat Metallic. Price: $0.0
6. French Blue Clearcoat Metallic. Price: $0.0
7. Twilight Blue Clearcoat Metallic. Price: $0.0
8. CD Silver Clearcoat Metallic. Price: $0.0
9. Pitch Black Clearcoat. Price: $0.0
10. Cloud 9 White Clearcoat. Price: $0.0
4
Please choose the Transmission
1. automatic. Price: $0.0
2. manual. Price: $-815.00
1

Server [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_60.jdk/Cont
Listen to server port #8888
Server waiting to connect!
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 1
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Server: Fail to upload car model, press y to upload again!
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 2
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Mobile sets size is: 3
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 1
Client: Sending properties file....
Accepting data....
Data Accepted!
Mobile sets size is: 4
Client: OK! Connected!
Server: Successfully get output and input stream!
User input is: 2
User choose car model X5
```

Then the client console asks for Transmission selection, I randomly choose 1. And also the next is other features selection, and I will skip this snapshot. Finally, the client console will print out you customize car information and your total price



And finally input n to exit this program.

Test completed.

#### **4: Special cases**

During your test, if prompt port already in use error, remember close both client and server session. Then test again.