

Class details are described in following pages

Class: Option

```
▼ G Option
  SF serialVersionUID : long
  name : String
  price : int
  C Option()
  C Option(String, int)
  replaceOption(Option) : void
  getName() : String
  getPrice() : int
  setName(String) : void
  setPrice(int) : void
  toString() : String
```

Class: OptionSet

```
▼ G OptionSet
  SF serialVersionUID : long
  name : String
  options : ArrayList<Option>
  C OptionSet()
  C OptionSet(String)
  C OptionSet(String, int)
  insertOption(Option) : void
  setName(String) : void
  insertOptionSet(OptionSet) : void
  setOptions(ArrayList<Option>) : void
  setOption(int, String, int) : void
  setOption(String, int) : void
  setOption(String, String) : void
  getOption(String) : Option
  getOptions() : ArrayList<Option>
  getName() : String
  getOptionPrice(String) : int
  findOption(String) : int
  deleteOption(String) : void
  toString() : String
  find(Object) : Object
```

Class: Automobile

```

G Automobile
  SF serialVersionUID : long
  □ modelName : String
  □ optionSetList : ArrayList<OptionSet>
  C Automobile()
  ● getMake() : String
  ● setMake(String) : void
  ● insertOptionSet(OptionSet) : void
  ● insertOption(OptionSet, Option) : void
  ● getOptionSetList() : ArrayList<OptionSet>
  ● getOptions(String) : ArrayList<Option>
  ● getOption(String, String) : Option
  ● getModelName() : String
  ● getBasePrice() : int
  ● setModelName(String) : void
  ● setBasePrice(int) : void
  ● setOptionSetList(ArrayList<OptionSet>) : void
  ● setOptionNameInOptionSet(OptionSet, String) : void
  ● setOptionsInOptionSet(OptionSet, ArrayList<Option>) : void
  ● updateOption(OptionSet, String, String) : void
  ● updateOption(OptionSet, String, int) : void
  ● deleteOptionSet(String) : void
  ● deleteOption(String, String) : void
  ● △ find(Object) : Object
  ● △ toString() : String
  ● getTotalPrice() : int
  ● △ calculatePrice() : int
```

Class: Util

```
▼ Util
  Util()
  serialize(Automobile, String) : void
  readSerializedFile(String) : Automobile
  writeFile(Automobile, String) : void
  readFile(String) : Automobile
```

Class: kbbSystem

```
▼ kbbSystem
  S autoLinkedHashMap : LinkedHashMap<String, Automobile>
  S insertionCount : int
  kbbSystem()
  getAutos() : LinkedHashMap<String, Automobile>
  insertAuto(Automobile) : void
  toString() : String
  find(Object) : Object
```

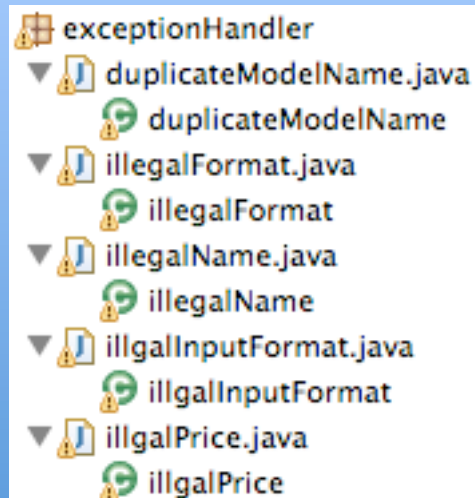
Abstract Class: Product

```
▼ Product
  make : String
  price : int
  calculatePrice() : int
```

Interface:

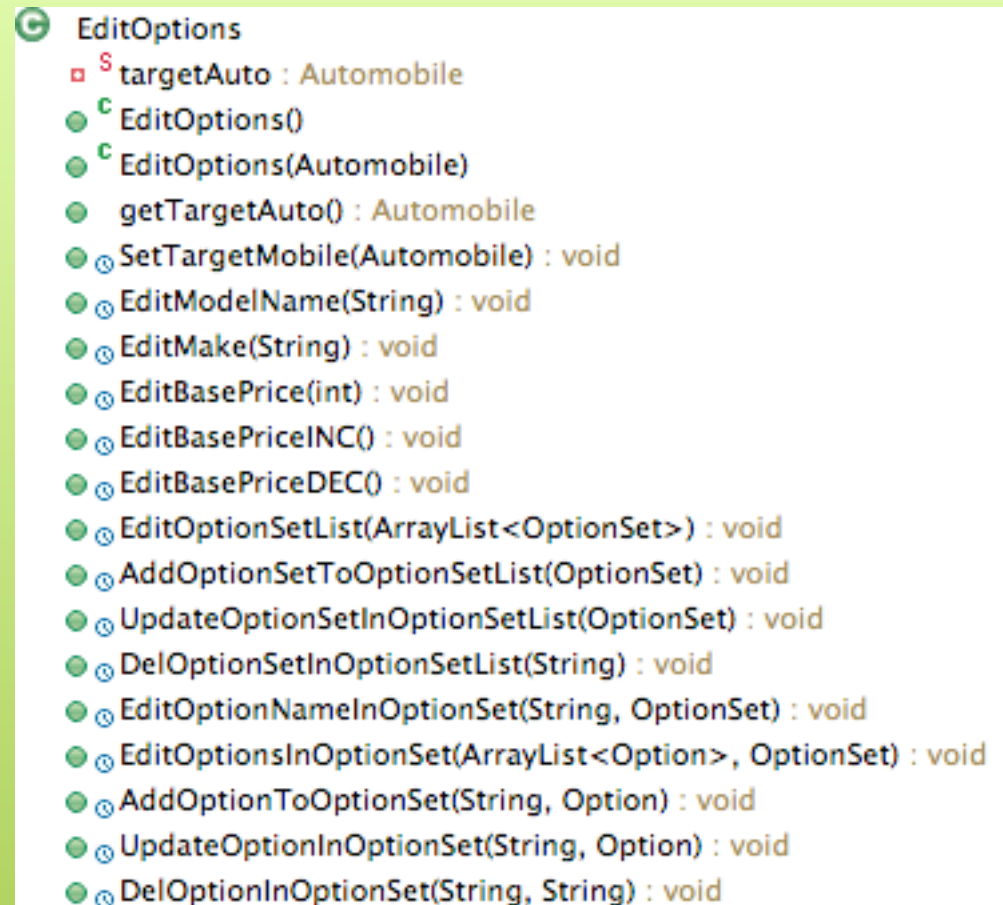
```
▼ CanModifyAll.java
  CanModifyAll
    modifyAll(kbbSystem) : void
▼ CanModifySelf.java
  CanModifySelf
    modifySelf(Automobile) : void
▼ CanRead.java
  CanRead
    read(Automobile) : String
▼ Find.java
  Find
    find(Object) : Object
```

Exception Handler



```
exceptionHandler
├── duplicateModelName.java
│   └── duplicateModelName
├── illegalFormat.java
│   └── illegalFormat
├── illegalName.java
│   └── illegalName
├── illegalInputFormat.java
│   └── illegalInputFormat
└── illegalPrice.java
    └── illegalPrice
```

EditOptions



```
class EditOptions {
    private Automobile targetAuto;

    EditOptions()
    EditOptions(Automobile)

    getTargetAuto() : Automobile
    SetTargetMobile(Automobile) : void
    EditModelName(String) : void
    EditMake(String) : void
    EditBasePrice(int) : void
    EditBasePriceINC() : void
    EditBasePriceDEC() : void
    EditOptionSetList(ArrayList<OptionSet>) : void
    AddOptionSetToOptionSetList(OptionSet) : void
    UpdateOptionSetInOptionSetList(OptionSet) : void
    DelOptionSetInOptionSetList(String) : void
    EditOptionNameInOptionSet(String, OptionSet) : void
    EditOptionsInOptionSet(ArrayList<Option>, OptionSet) : void
    AddOptionToOptionSet(String, Option) : void
    UpdateOptionInOptionSet(String, Option) : void
    DelOptionInOptionSet(String, String) : void
}
```

Unit 3 Changes

Classes in EditOptions are synchronized.

The Runner class generated two threads to change a value in a static Automobile object. The value was increased and decreased by the thread. With the help of synchronization, threads are not racing with each other to get the original value and the write back behavior is stable.

The text file uses <> and </> to separate OptionSet names, uses comma to separate option names and prices.

The program parse the file and construct an Automobile object, serialize and deserialize the data to recreate Automobile object.

Program is implemented to accept extra space at beginning or ending of each line or invalid price (e.g. No price or negative price). Duplicated option will be overwritten by the latest record. Duplicated OptionSet will be merged into one OptionSet.

To run the program, the user need to type in a certain format in car.in and get result from car.out.