Tower Defense

First Build - Design Document

Amritansh Mishra	27288883
George Ekow-Daniels	40011883
Baitassov Ulan	40001592
Sajjad Ashraf	40012126

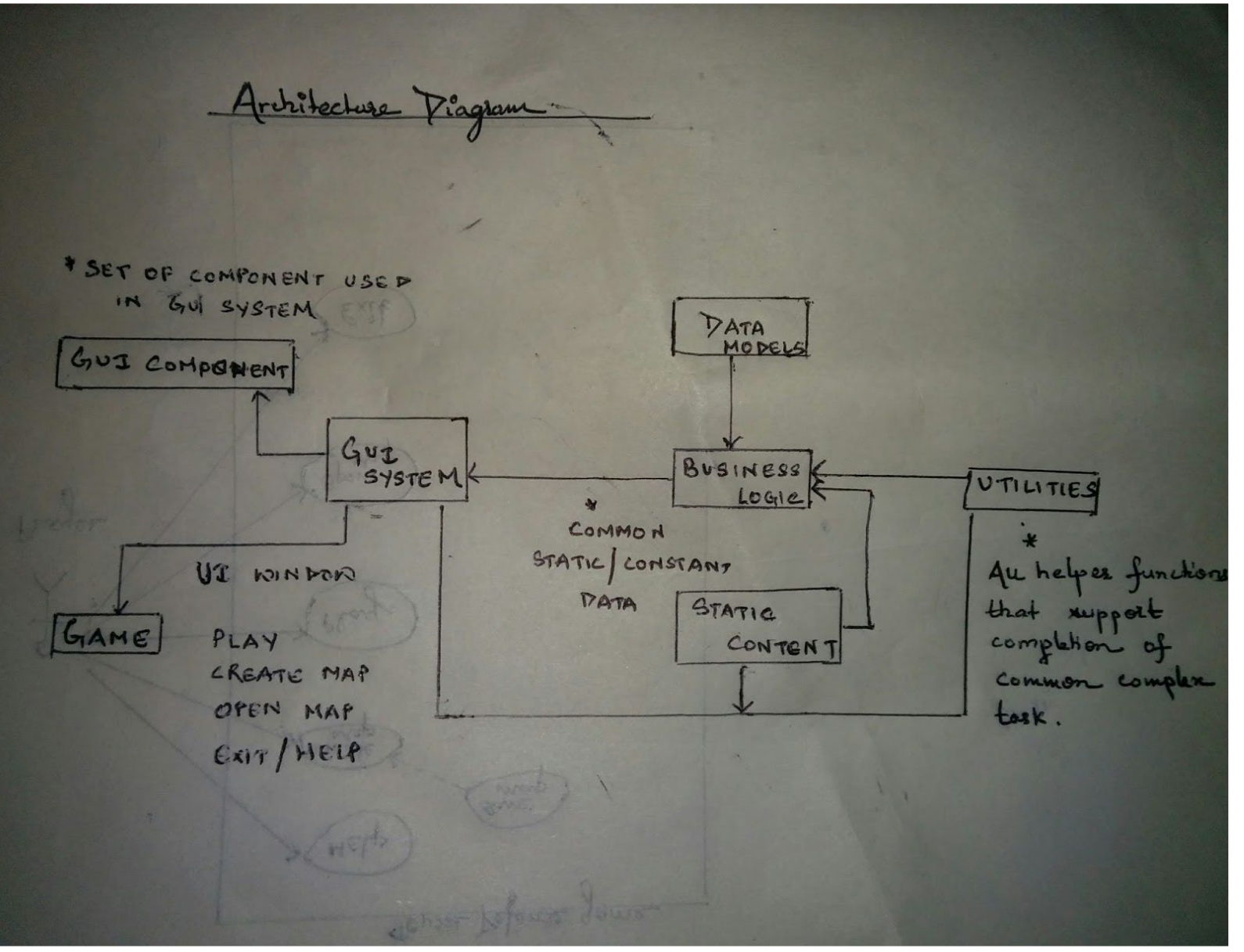
Table of Contents

1.Introduction

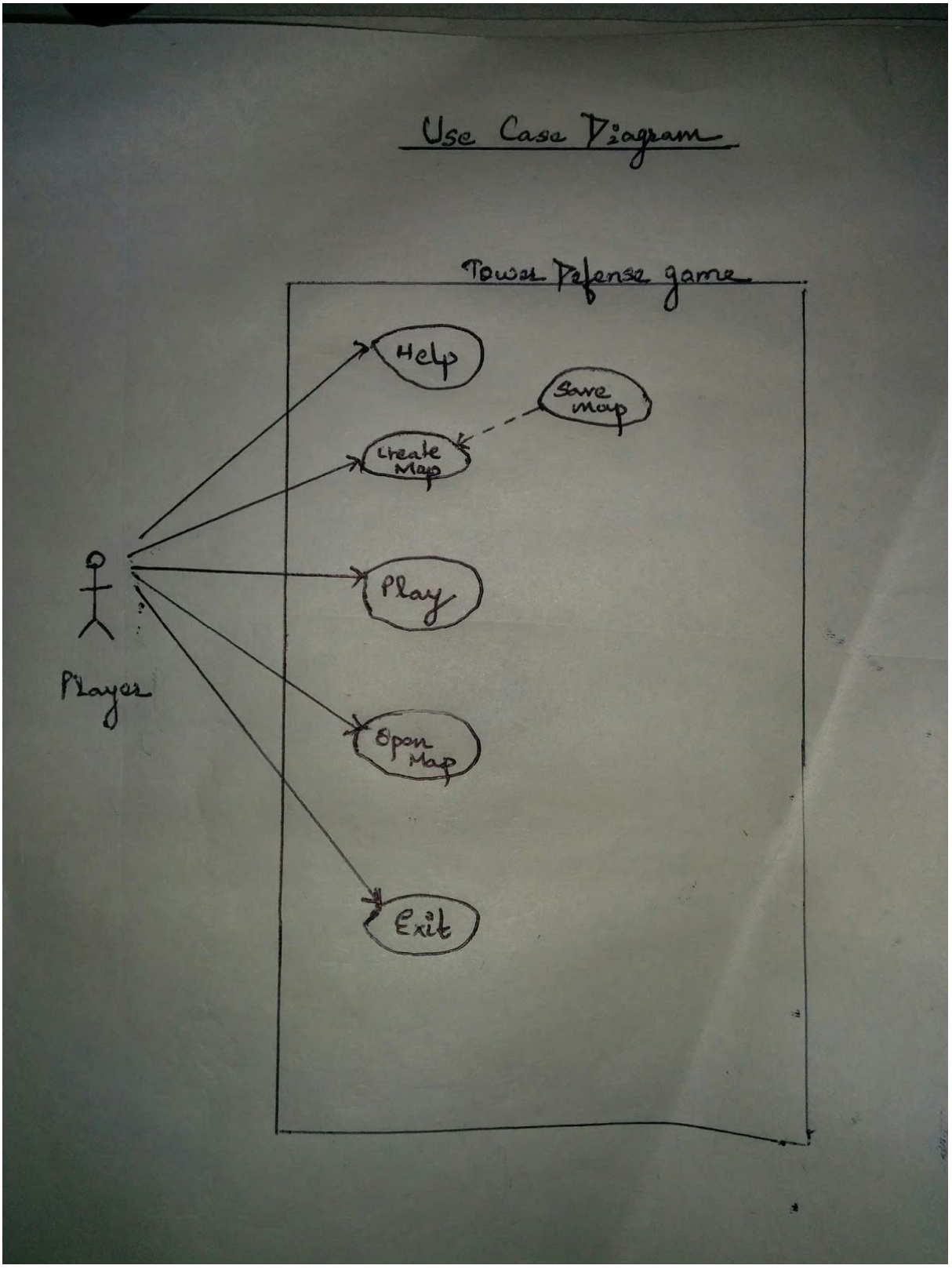
The tower defense game follows the mvc and singleton pattern in parts. The code is organized in seven packages.

1. com.app.towerDefense.guisystem
    a. This Package contains all Windows UI Classes
2. com.app.towerDefense.guiComponents
    a. This package contains the UI Compnents(Buttons, Panels, Images etc) class that give support for guisystem.
3. com.app.towerDefense.bL
    a. This package has all complex Business Logic or Game logic like validating and verifying maps.
4. com.app.towerDefense.models
    a. It contains all Data Object (models) of game that we use in BL and gui components.
5. com.app.towerDefense.staticContent
    a. It contains all Static or constant data most which is not changing very frequently.
    b. All Type of Static or hard code should be declared here.
    c. Easy to Maintain the code. like if want to change the images in over game just need to change in this packages and it will reflect where those variable are being used.
6. com.app.towerDefense.test
    a. All Unit test case defined in this package.
7. com.app.towerDefense.utilities
    a. Here we defined all type of Function that used multiple time from different package.Like saveing file, read file, Json Serialization, deserialization, base64encode decode.

2.Architecture Design



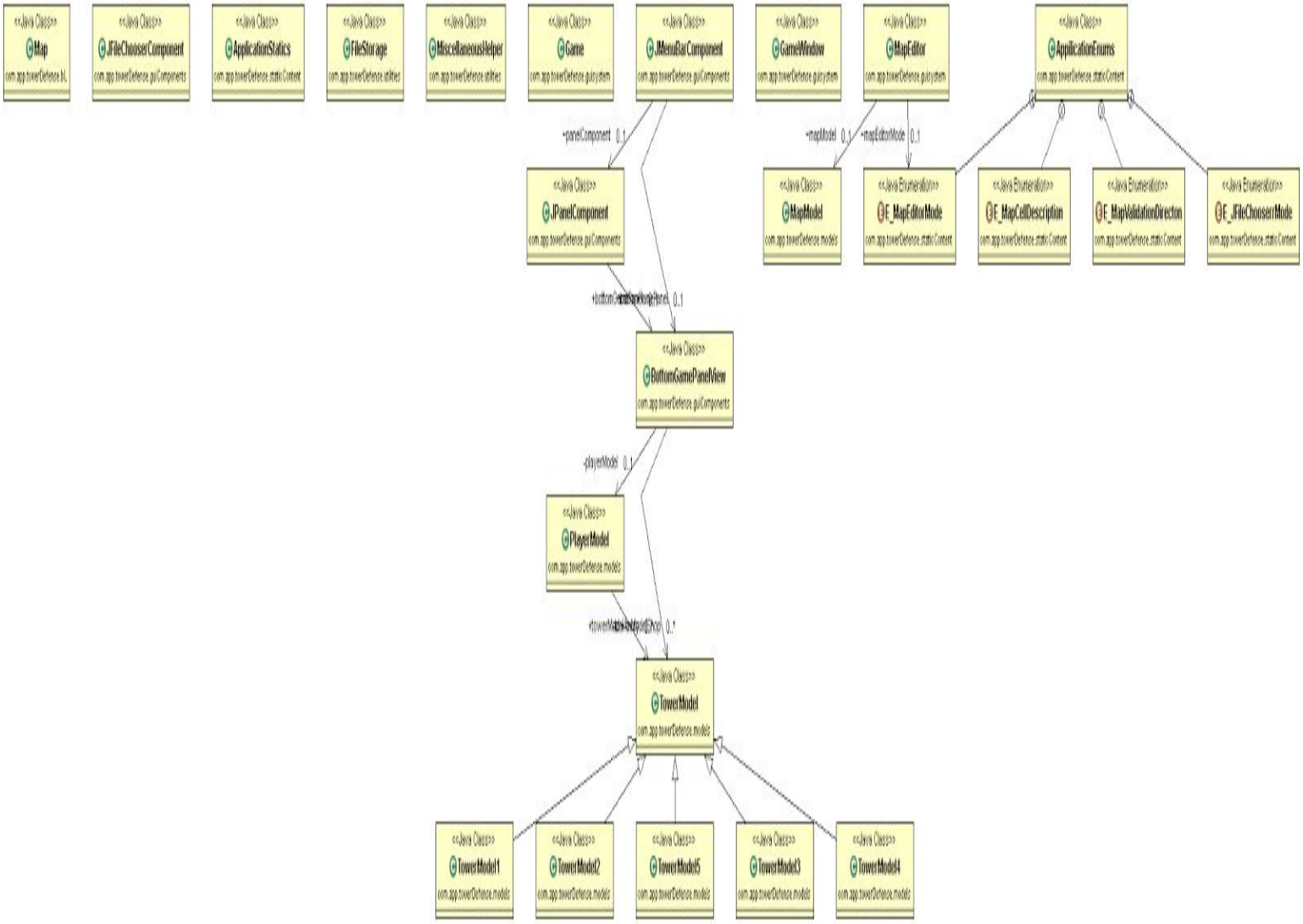
3. Use Case Diagram



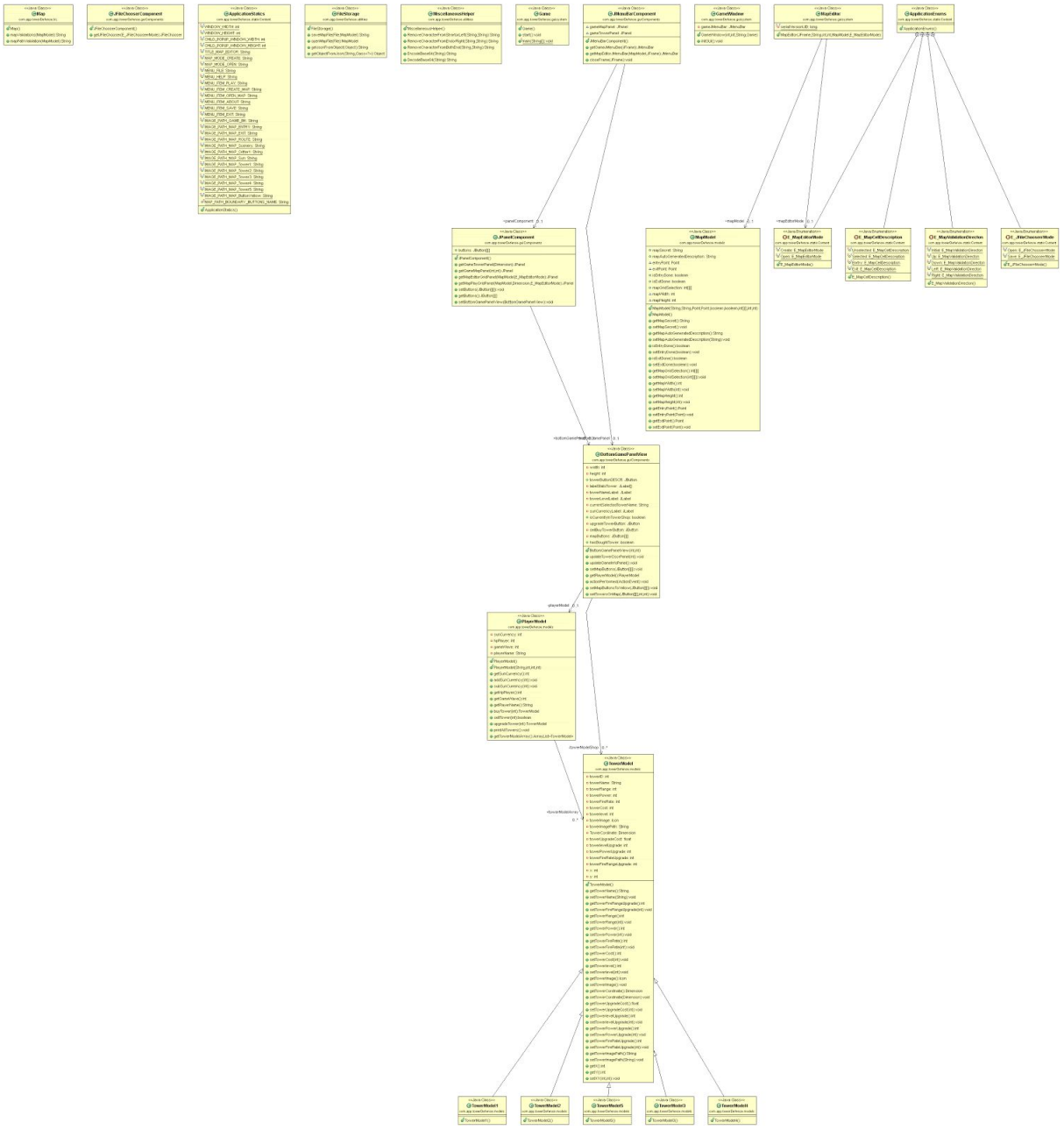
4. Class Diagram

Tower Model

Class Relationship Diagram



Detailed Class Diagram



5.Coding Standards

To build enterprise Java applications, which are reliable, scalable and maintainable, it is important for development teams to adopt proven design techniques and good coding standards. The adoption of coding standards results in code consistency, which makes it easier to understand, develop and maintain the application. In addition by being aware of and following the right coding techniques at a granular level, the programmer can make the code more efficient and performance effective.

Commenting

Single Line Comments

```
/* Handle the condition. */
//if (bar > 1) {
```

Multiple Line Comments

```
/*
 * Here is a block comment.
 */
```

Documentation Comments

```
/**
 * The Example class provides ...
 */
```

Class Declaration

Java source are named as *. java while the compiled Java byte code is named as *.class file. Each Java source file contains a single public class or interface. Each class must be placed in a separate file. This also applies to non-public classes too.

Variable Declaration

Class Variables - All class variables are declared in camelcase.

Example :

```
public String mapSecret;
```

Local Variables- All local variables are declared in camelcase.

Example :

```
TowerModel tempTM = null;           ;
```

Parameters - All parameter are prefixed with new_.

Example :

```
public TowerModel buyTower(int new_towerID){}
```

Constants - All constants are written in capital letters and the ones which need to be reused are declared in the file AppicationStatics.java

Example :

```
public static final int CHILD_POPUP_WINDOW_WIDTH = (int) WINDOW_WIDTH - 100;
```

File Organization

All classes are declared as part of a package.

Example:

Models are organized under the package :

```
com.app.towerDefense.models
```

Method Names

- All method names are written in camelcase. All method names begin with a strong action verb.

  Example :

  ```
  public TowerModel buyTower(int new_towerID){}
  ```

- Use of the prefixes get and set for getter and setter methods.

  Example :

  ```
  public String getPlayerName() {}
  ```

- If the method returns a boolean value, use is or has as the prefix for the method name.

  Example :

  ```
  public boolean isEntryDone() {}
  ```