**Requirements and Format for Term Project Documentation**

Due: **December 7 Monday, 11:59PM** (via D2L dropbox)

Submission:

* Accepted file format: **pdf**
* Filename: **project\_***Firstname*\_*LastnameInitial***.pdf (e.g., project\_Hong\_S.pdf)**

**In the following, blue headings are intended for the template of the term project document. Each section must begin on a new page. Delete all instructional part of this template.**

**Cover Page:**

* Course Title: CMSC 3103 Object Oriented Software Design & Construction
* Semester: Fall 2015
* Name: Younouss Thiam
* UCO ID: 20342920
* Major (Computer Science, CS – Applied, CS – Information Science, Software Engineering)

Major: CS-Applied

Minor: Mathematics

**Section I: User Stories**

* Game Interface:
  + Game Interface:
* As a player I want to have a game interface so that I can play.
  + Game Figures:
* As a player, I want to have 3 types of enemy so that I can make the game play more interesting.
* As a player I want to have 2 types of friend figures.
  + Bombs:
    - As a player I want to see the total number of bombs added so that I know how many bombs I added in total.
    - As a player I want to see the total number of bombs killed so that I know how many bombs I killed in total.
    - As a player I want to see the total number of bombs alive so that I know how many bombs are still alive in total.
  + UFOs:
* As a player I want to see the total number of UFOs added so that I know how many UFOs I added in total.
* As a player I want to see the total number of UFOs killed so that I know how many UFOs I killed in total.
* As a player I want to see the total number of UFOs alive so that I know how many UFOs are still alive in total.
  + Aliens:
* As a player I want to see the total number of Aliens added so that I know how many Aliens I added in total.
  + As a player I want to see the total number of Aliens killed so that I know how many aliens I killed in total.
  + As a player I want to see the total number of Aliens alive so that I know how many aliens are still alive in total.
* Game Play:
  + Bombs:
    - As a player I want to be able to click on an “Add 10” button so that I can add 10 bombs on the screen.

The bombs are added at random positions and bounces off the wall automatically

* + As a player, I want to kill the Bombs with the missiles so that I can increase the total number of killed Bombs.

Bombs explode when there is a collision with missiles.

* + As a player, I want to kill the Bombs with the shooter so that I can increase the total number of killed Bombs.

Bombs explode when there is a collision with the shooter.

* + UFOs:
* As a player I want to be able to click on an “Add UFO” button so that I can add 1 UFO on the screen.

The UFO is added at random position and flies horizontally back and forth.

* As a player, I want to kill the UFOs with the missiles so that I can increase the total number of killed UFOs.

UFOs fall down when there is a collision with missiles.

* As a player, I want to kill the UFOs with the shooter so that I can increase the total number of killed UFOs.

UFOs fall down when there is a collision with the shooter.

* + Aliens:
* As a player I want to be able to click on an “Add Alien” button so that I can add 1 Alien on the screen.

The Alien is added at random position and sparkles.

* As a player, I want to kill the Aliens with the missile so that I can increase the total number of killed Aliens.

Aliens shrink when there is a collision with missiles

* As a player, I want to kill the Aliens with the shooter so that I can increase the total number of killed Aliens.

Aliens shrink when there is a collision with the shooter.

* Missiles:
* As a player I want to click with the mouse so that I can launch missiles.

The missile travels to the mouse press location and then explodes to a certain size.

* Shooter:
* As a player I want to have a shooter so that I can launch missiles from it.
* As a player, I want to press the [Left-Key] so that I can move the shooter to the left by 10 pixels.
* As a player, I want to press the [Right-Key] so that I can move the shooter to the right by 10 pixels.
* As a player, I want to press the [Up-Key] so that I can move the shooter up by 10 pixels.
* As a player, I want to press the [Down-Key] so that I can move the shooter down by 10 pixels.

.

* General:
* Asa player I want to be able to click on a quit button so that I can quit the game.

**Section II: CRC Cards**

Using the same format of CRC cards, write CRC cards for

|  |  |
| --- | --- |
| **GamePanel** | |
| Assure bouncing of bombs | Bomb |
| Assure flying of UFOs | UFO |
| Assure sparkling of Aliens | Alien |
| Assure traveling of Missiles | Missile |

|  |  |
| --- | --- |
| **Animator** | |
| Detect collisions between missiles and bombs | Missile, Bomb |
| Detect collisions between missiles and UFOs | Missile, UFO |
| Detect collisions between missiles and aliens | Missile, Alien |
| Detect collisions between launcher and bombs | Launcher, Bomb |
| Detect collisions between launcher and UFOs | Launcher, UFO |
| Detect collisions between launcher and aliens | Launcher, Alien |
| **MainWindow** | |
| Display total number of bombs | Bomb |
| Display number of bombs killed | Bomb |
| Display number of bombs alive | Bomb |
| Display total number of UFOs | UFO |
| Display number of UFOs killed | UFO |
| Display number of UFOs alive | UFO |
| Display total number of Aliens | Alien |
| Display number of Aliens killed | Alien |
| Display number of Aliens alive | Alien |

|  |  |
| --- | --- |
| **Main** | |
| Create Animator | Animator |
| Create Game Panel | GamePanel |
| Create Game Data | Game Data |

|  |  |
| --- | --- |
| **GameData** | |
|  |  |

|  |  |
| --- | --- |
| **Collision Area** | |
| Create collision area |  |

|  |  |
| --- | --- |
| **Game Figures** | |
| Create Game Figures |  |
| Get collision area | Collision Area |

|  |  |
| --- | --- |
| **Bomb** | |
| Create Bombs | Game Figure |
| Get collision area | Collision Area |

|  |  |
| --- | --- |
| **Launcher** | |
| Create Launcher | Game Figure |
| Get collision area | Collision Area |

|  |  |
| --- | --- |
| **UFO** | |
| Create UFOs | Game Figure |
| Get collision area | Collision Area |

|  |  |
| --- | --- |
| **Alien** | |
| Create Aliens | Game Figure |
| Get collision area | Collision Area |

|  |  |
| --- | --- |
| **Missile** | |
| Create Missiles | Game Figure |
| Get collision area | Collision Area |

|  |  |
| --- | --- |
| **Button Listener** | |
| Add 10 Bombs | Bomb |
| Add 1 UFO | UFO |
| Add 1 Alien | Alien |
| Quit the game |  |

|  |  |
| --- | --- |
| **Mouse Controller** | |
| Launch missiles | Missile, Launcher |

|  |  |
| --- | --- |
| **Key Controller** | |
| Move launcher up | Launcher |
| Move launcher down | Launcher |
| Move launcher left | Launcher |
| Move launcher right | Launcher |

**Section III: UML Class Diagram**



**Section IV: Implementation of Required Features**

**(1) Implementation of One Additional Enemy Type**

*Explain in detail attributes and behaviors of the additional enemy type that you implemented*

* *Attributes:*
* *collision Box: rectangle form that represents the contour of the Alien type.*
* *width: width of the rectangle representing the Alien type.*
* *height: height of the rectangle representing the Alien type.*
* *color: color of the rectangle representing the Alien type.*
* *cloneWidth: copy of the width of the rectangle representing the Alien type.*
* *Behaviors:*
* *Alien: constructor of the Alien type.*
* *render: responsible for the graphic rendering.*
* *update: responsible for the graphic rendering update.*
* *getCollisionBox: return the collision box.*
* *deadBehavior: reduce the height and the weight from 1.*

**(2) Destroying Effects of Enemies**

*Explain in detail the effects you implemented when each enemy is destroyed.*

* *Bomb:*
* *Basically the size of the bomb increases until it reaches a maximum size and the bomb disappears.*
* *The bomb increases in size and disappears through the deadBehavior method.*
* *FlyingSaucer:*
* *The flyingsaucer falls down.*
* *This movement is implemented trough the dead behavior method.*
* *Alien:*
* *The size of the alien is reduced until it disappears.*
* *The alien decreases in size and disappears through the deadBehavior method.*

**(3) Implementation of the State design pattern for three Enemy types**

*For each enemy type, explain in detail how you utilized the State pattern (In Section V and VI, you will give UML diagrams and proof. You may refer to them when you explain here in detail.)*

* *The state design pattern has been implemented in the Game Figure class so each enemy type uses this implementation.*
* *There is 1 State class and 2 children of the classes: the StateAlive class and the StateDone class.*
* *There is 1 constructor called setState for each one of the children of the State class that defines the state of the Game Figure.*
* *There is 1 goNext method in the State class that is overridden by each of the children of the State class which permits to change the state of the Game Figure.*
* *There is 1 setState method in the Game Figure class, which permits to define the state of the Game Figure.*
* *There is 1 pull method in the Game Figure class, which permits to implement the goNext method from a state context.*
* *The pull method is implemented in the animator class and permits to remove the enemy Figures when collisions are detected.*

**(4) Implementation of an additional design pattern**

*Explain in detail how you utilized this design pattern (In Section V and VI, you will give UML diagrams and proof. You may refer to them when you explain here in detail.)*

**(5) Implementation of an optional feature**

*Explain in detail the optional feature you implemented and how you utilized it for the game. If it is an additional design pattern, add it also in Section V and VI as the proof of correct implementation.*

**(6) New Ideas of the Game**

*Explain in detail if your game is not directly based on p8; e.g., different enemy types than UFO and Bombs. different behaviors of the main character (Shooter).*

**Section V: UML Diagrams for Design Patterns Implemented**

**For each design pattern**:

* Draw a UML diagram in detail **including attributes (instance variables) and behaviors (methods)**
* Include classes only if they are the participants of the design pattern.
* Must include relationships and multiplicity based on your finished program.

Note that you need to include 4 or 5 UML diagrams:

* 3 UML diagrams for State pattern implementation for three enemy types.
* 1 UML diagram for an additional design pattern implementation
* 1 more UML diagram if you choose a design pattern as the optional feature.

**Section VI: Proof for the Correctness of Implemented Design Patterns**

For each design pattern (for State pattern, choose one of enemy types), **prove** that your implementation is correct in terms of (1) the intent of the pattern, (2) the responsibilities of each participant in the pattern.

Use the following table for the proof:

|  |  |  |
| --- | --- | --- |
| **Pattern name: XXX** | **The original specification** | **Your implementation** |
| **Intent** | *write the original intent of this pattern* | *Write your intent and show that your intent matches the original intent* |
| **participant’s name: XXX** | *write the original responsibilities of this participant* | ***Your class name*** *for this participant.*  *Explain how responsibilities of this participant are implemented in your class.* |
| ***repeat for the remaining participants*** | *repeat for the remaining participants* | *repeat for the remaining participants* |
| **…** | … | … |

**For example:** This is a proof for the correct implementation of the Chain of Responsibility pattern in DP\_ChainOfResponsibility1 example program.

|  |  |  |
| --- | --- | --- |
| **Chain Of Responsibility** | **The original specification** | **Your implementation** |
| **Intent** | * *Avoid coupling the sender of a request to its receiver by giving more than one object a chance to handle the request.* * *Chain the receiving objects and pass the request along the chain until an object handles it.* | * In the restaurant, the one who complains is decoupled from the ones who receive and processes the complaints. * Waiter, Chef, and Manager would receive the complaints, but they are chained in terms of responsibilities and the complaint is passed along the chain until the complaint is handled. |
| **Handler** | * *defines an interface for handling requests* * *(optional) implements the successor link.* | **class name**: ComplaintHandler   * abstract void processComplaint(String message) method is for the interface * ComplaintHandler successor instance variable is to implement the successor link |
| **ConcreteHandler** | * *handles requests it is responsible for* * *can access its successor* * *if the ConcreteHandler can handle the request, it does so; otherwise it forwards the request to its successor.* | **Class names***: Manager, Chef, Waiter*   * Each concrete handler has its own responsibility to process the complaints. * Each concrete handler is set for its successor * The complaint is forward to the successor if it can not be handled. |
| **Client** | * *initiates the request to a ConcreteHandler object on the chain* | **Class name**: Main   * Complaints are initiated and they are sent to Waiter concrete handler. |