

CSE 1325-002
Spring 2015
Class Project
Assigned March 31st, 2015, 2:00 pm
Due: April 28th, 2015, 2:00 pm

This file is a homework assignment for a college course that includes individual work.
If this file should submitted to a newsgroup or answer page, please contact me at
becker@uta.edu.
Thank you.

1. Introduction

By this time, everyone in the class should be acquainted with the Java Language, and Object-Oriented programming techniques. Encapsulation, Polymorphism, Generics, member functions and member variables should all have been used. The next stage of the class is using Graphical User Interfaces (GUIs) for the project. The first three homework's are the background for this project, with a twist. Ships come into docks in a port, and they are all on the map. In turn, monsters exist on the sea that can destroy the ships.

2. Objectives

- Creating a GUI
- Creating the Menus
- Creating Dialog boxes
- Using the mouse
- Creating Pop-up menus
- Being able to manage the 2-dimensional graphics elements
- Be aware that Netbeans and other IDE's have many built-in tools to make a programmer's life easier.

3. Requirements to turn in assignment

Netbeans has an export project to a zip file ability. When turning in the assignment, please zip the entire project and upload to Blackboard, or other instructions as needed. Be sure to complete the first menu task, for this is where your name is on your paper.

Required Comments:

At the beginning of each file, you should ensure your name and student id number appear at the top of the file. At the beginning of each class, you should write a short description of the class At the beginning of each function, you should write a short description, and include the parameters and returns of the function. If there are no parameters or returns, leave that part blank.

Good Naming Standards

Writers of Java code have established some simple standards for how a program should be written, addressing both the functionality and the readability of the code. You can find several standards online by simply searching for Java coding standards, including ones by Oracle and Google. Additionally many corporations who use Java have established their own, slightly modified, set of coding standards.

1. Variable names will be written in camel case (e.g. mapData or mainMapMenu). This includes both the member variables of your classes and the internal variables used by a method, such as the parameters.
2. Class names will be written with the first letter of each word capitalized (e.g. MainMenu, MapNode).
3. Class and variable names will be in plain language and will not be abbreviated (e.g. instead of calling an instance of MainMenu simply mm you would call it mainMenu).
4. Any member variables of a class will be set as "private" variables. In order to access those variables getter and setter methods will be created.
e.g.

```
public class MapData
{
    private MapNode mapNode;

    public MapNode getMapNode()
    {
        return mapNode;
    }
    public MapNode setMapNode(MapNode newMapNode)
    {
        this.mapNode = newMapNode;
    }
}
```

4. The Code and the Demo

For the code, as before, the program will need to be turned into a zip file. The name of the file should be in the form of **project.teamname.zip**, preferably generated by Netbeans after a clean build of your program.

For the demo, on the last week of classes, you and your team will be asked to demonstrate your GUI to a grader, either the instructor or one of the teaching assistants. At that time, the grader will bring a fresh set of files of the same format as those from the homework assignments. The Demonstration will be the major part of the grade.

5. Instructions

Procedure

For this assignment, you will be participating in the creation of a Graphical User Interface. Actually, creating a GUI is not difficult with today's modern tools. A key issue, however, is the sheer amount of low-level work necessary to connect all the components of the GUI. As a result, you will work on a team, and will need to keep a log on Blackboard.

Team

By now, you will have formed a team or the instructor will have created a team for you on Blackboard. You will be working in groups of three to complete the assignment. If you need to change your team in mid-project, you will have to come in person to the instructor's office hours.

Log

Blackboard has a large number of student tools available, including a group Team Blog for keeping track of work. While the log will not be required for grading, it is highly recommended that you and your team keep a record of your project, a log or journal.

6. Additional Classes

In order to complete the project, previous classes from the homework and new classes have to be included.

Previous Classes

- The Map, The Port, the FileHandler, and the Main
- Ships-Cargo Ships, Container Ships, and Oil Tankers
- Docks-Docks, Cranes, and Piers
- Cargoes-Cargo, Boxes, and Oil

New Classes

Position Class

Sea Monsters will have a shared Position class. The Position class will contain the current longitude and latitude of each item, the current row and column of each item, and the current pixel location of each item. When you are setting or comparing locations on the map use the Position class to reference the location.

Sea Monster Class

Previously, with Docks, Cargos, and Ships, the only forms of inheritance was concrete inheritance, that is that each child class had all the given properties of the parent class. This time, Sea Monster will be a parent **abstract** class. The abstract class will contain an abstract function “battlecry” which is overridden in each child class. In addition, the Sea Monster Class will have an object of the Position class as a member variable that will be shared amongst its children. Each Sea Monster will also have a label that is the type of monster and the current count of the monster (Ex "Leviathan 1"). In addition, the Sea Monster children have special rules when interacting with ships on the map. These additions will be detailed in the map description below.

Kraken Class

The Kraken Class is a child of the Sea Monster Class. The Kraken has a battle cry “RELEASE ME!” whenever it occupies the same square as a ~~Cargo Ship~~ any ship.

Leviathan Class

The Leviathan Class is a child of the Sea Monster Class. The Leviathan has a battle cry of “Come! Ahab beckons!” whenever it occupies the same position as a ~~Container Ship~~ any ship.

Sea Serpent Class

The Sea Serpent Class is a child of the Sea Monster Class. The Sea Serpent has a battle cry of “Suddenly, you hear bagpipes!” whenever it occupies the same position as an ~~oil tanker~~ any ship.

Godzilla Class

The Godzilla Class is a child of the Sea Monster Class. Godzilla has a battle cry of “Baraaaawr-rompf!” whenever it occupies the same position as another sea monster. The Godzilla class is also unique (a singleton). Your Godzilla Class should include static features to ensure that only one Godzilla ever exists at a time. If a second Godzilla should be created, the first Godzilla should be removed from the map.

7. The Map

The GUI will consist of a set of menus, a map, pop-up menus, and a status window. The menus will be contained in a menubar: File, Ship, Port, Sea Monster and About. These menus can open up submenus, and then dialog boxes to update the settings. The second component will be a map, a panel that can be used to draw the different type of sub-images: Docks, Sea Monsters, Ships and Terrain as shown in the table. The third component is a status message box at the bottom of the GUI, where the battle-cries of the sea monsters can be displayed. This status message box is also useful as a debugging tool.

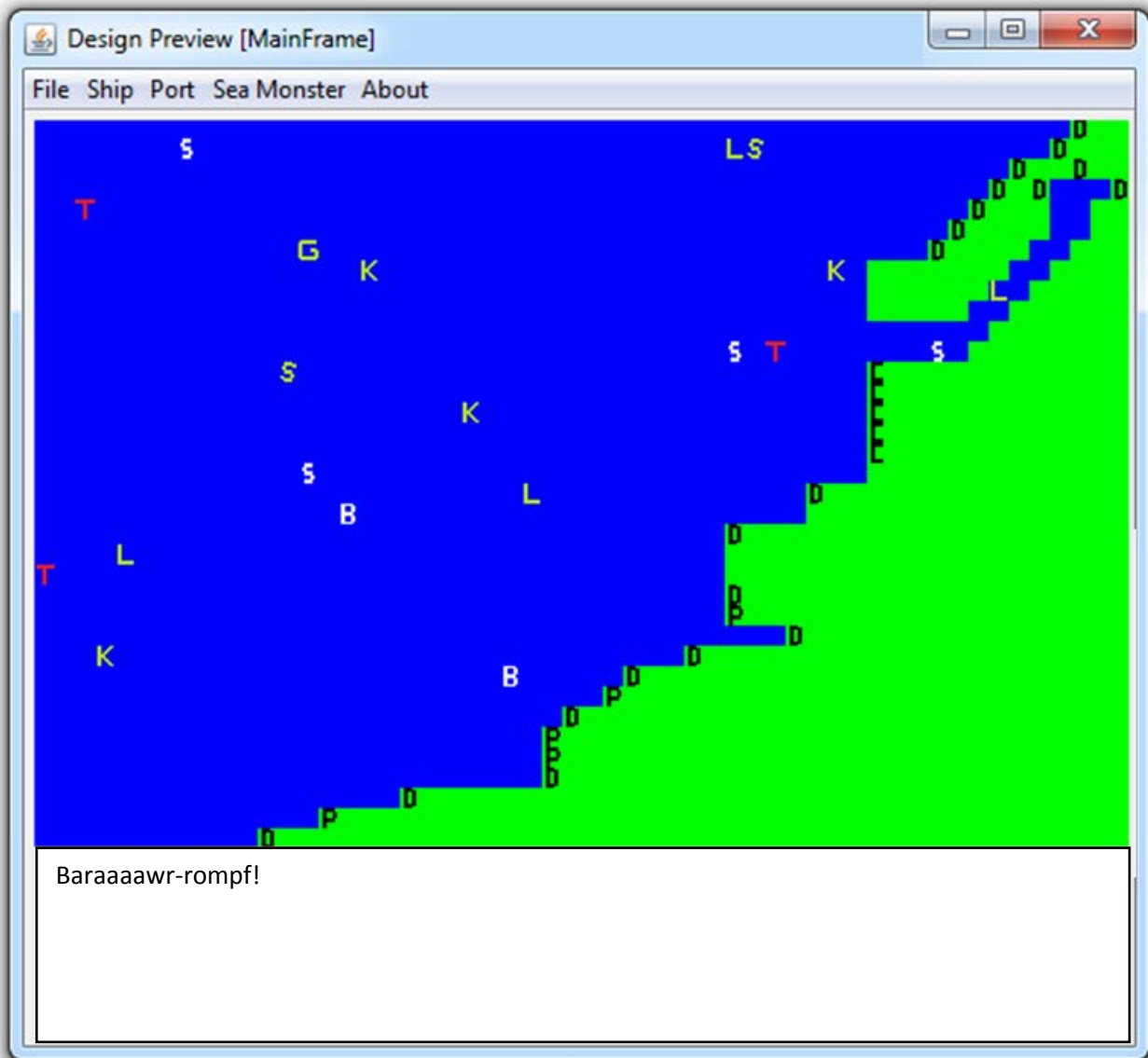












Figure 1. Example Screen

Table of Tiles for Map

Type	Icon	PNG
Dock	Crane	
Dock	Dock	
Dock	Pier	
Sea Monster	Godzilla!	
Sea Monster	Kraken	
Sea Monster	Leviathan	
Sea Monster	Sea Serpent	
Ship	Cargo Ship	
Ship	Container Ship	
Ship	Oil Tanker	
Ship	Safely Docked	
Ship	Unsafe Ship	
Terrain	Land	
Terrain	Sea	

8. Conversion to GUI

The time for ASCII text interfaces have passed on this project. The time has come to talk of many things: Menus, Dialog Boxes, Actions, Items, and Mouse. This next stage of the project will be GUI based, using the SWING and AWT components of the Java API.

The main menubar will have four menus and one menu item. File, Ship, Port, Sea Monsters and an About menu Item

File Menu

- **Open**-prompt the user for a tag, such as simple or complex, to load the current set of files into the system.
- **Close**-will erase the current map and all ships, docks, cargoes, and sea monsters
- **Snap Shot**- opens a file dialog to prompt the user for a filename and directory, then the program will write the current list of ships, docks, sea monsters, and cargoes to a file using a comma separated value format.
- **Exit**-ends the program

Ship Menu

- **Generate Ships** Prompt the user with a dialog box, and request a number of ships to be generated. The ships will then be placed on the map in the sea.
- **Update Ships** Prompt the user with a dialog box with a list item containing the names of all available ships. Once a ship has been selected, open a second dialog box that will allow the user to update the current ship properties.
- **Display all Ships**-Show the current ships in the status message box at the bottom of the screen.
- **Remove all Ships**-Remove all ships from the current map.

Port Menu

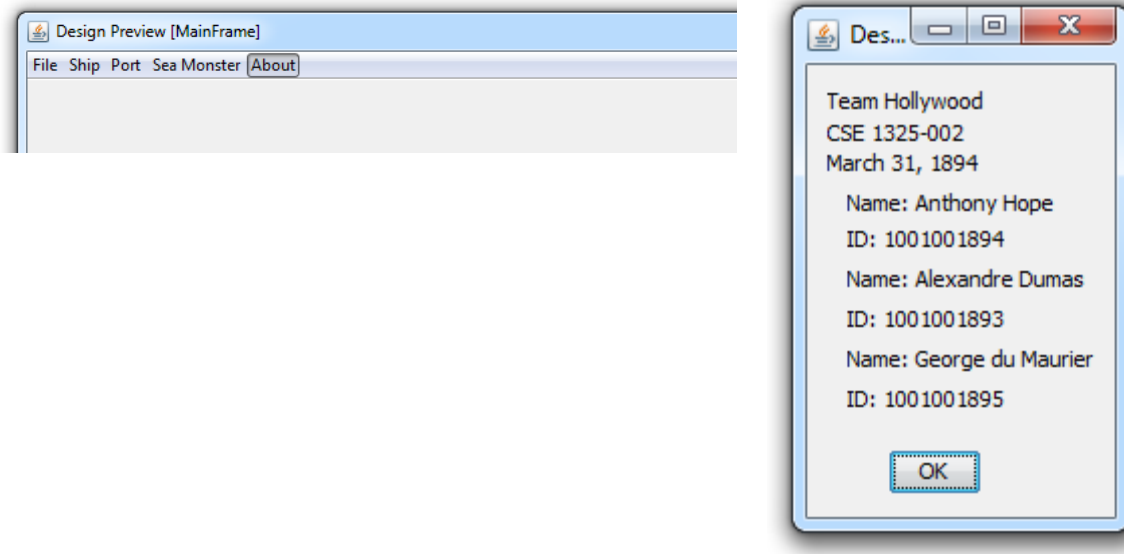
- **Unload Ship**-Prompt the user with a dialog box, showing a list of all the ships safely in dock
- **Update Dock**-Prompt the user with a dialog box with a list item containing the names of all available docks. Once a dock has been selected, open a second dialog box that will allow the user to update the current dock properties.
- **Display All Docks**-Show the current docks in the status message box at the bottom of the screen.
- **Display All Cargos**-Shows the current cargos in the port in the status message box at the bottom of the screen.

Monster Menu

- **Generate Monsters:** Prompts the user with a dialog box, and request a number of monsters (Sea Serpents, Leviathans, and Krakens) to be generated. The monsters will then be placed on the map in the sea.
- **Update Monsters**-Prompt the user with a dialog box with a list item containing the current monsters. Once a monster has been selected, open a second dialog box that will allow the user to update the current monster properties.
- **Display all Monsters**-Show the current monsters in the status message box at the bottom of the screen.
- **Remove all Monsters**-Remove all monsters from the map.
- **Summon Godzilla**-Put Godzilla on the map. Prompt the user for the location of where Godzilla would appear. Godzilla, being an amphibian, can be on either land or water.

About Button

When the about button is selected, it is menu item, not a menu. The button will open up a dialog box and show the information about the team and its members.



9. Drag and Drop

Ships and Monsters should have the ability to be dragged and dropped with a mouse. While the user can move a ship or a sea monster with adjusting its coordinates, a more friendly way is to select the symbol and move it while holding the mouse button. When the monster or ship reaches the new square, releasing the button will drop the object onto the new location.

10. Pop Up Menu

For the ships and monsters, these items should be considered to be locations on the map, and it would be much easier if the ships and monsters can be moved using the mouse. Two additional behaviors are needed. Have the pop-up menu for the Ship or for the Sea Monsters. When selected with the mouse, the Ship Properties or Monster Properties

11. Bonuses

Better Graphics: Higher Resolution and Scroll Bars- (5 Points)

The current layout of the map and screen only allows a minimum graphic resolution. While this makes programming much easier, it is dull and boring. Use scroll panes and better graphics to create a bigger map with nice pictures instead of simple letters and symbols. If you download images from a licensed website, please remember to add a cited works file to your team space on Blackboard.

Sound: Add the sounds of the monsters to play-(5 Points)

Each of the monsters issues a battle cry when they collide with a ship or another monster. Research media playing classes for sound files for Java. Replace the text battlecry with a command to play the sound file. If you download sounds from a licensed website, please remember to add a cited works file to your team space on Blackboard.

Java FX (5 points)

Currently, the assignment in class is using Java Swing. Like AWT before it, Swing is being replaced by JavaFX, the next generation of GUI for Java. Build the project in JavaFX instead of Swing for 5 points.

Threading (10 Points):

Have a “Play” function that will have the monsters hunt the ships, and Godzilla hunt the monsters. Research the “Predator-Prey” problem -15 Points

Final Exam Replacement Option:

If your team creates a truly over-the-top system, using something similar to Google Earth or NASA World Wind to run the map and data points, and you demonstrate this to the class without crashing, you will receive a 100 on the Final Exam.