CSCI 201L Assignment #4
Spring 2015
5.0% of course grade

Overview

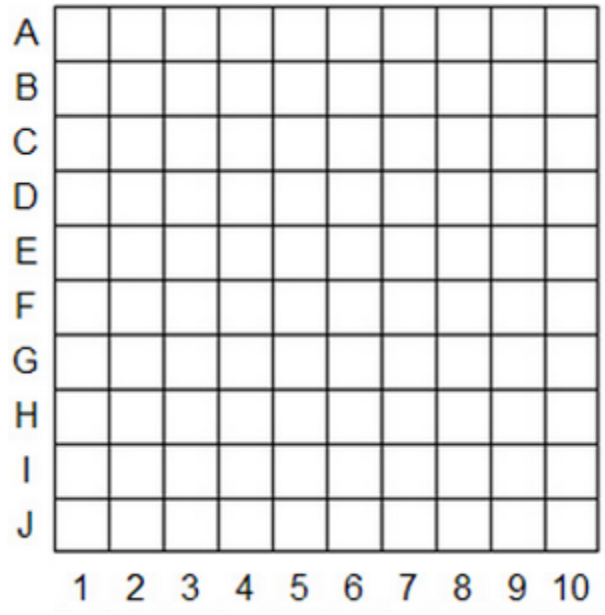
This assignment will build off your previous assignment to improve your Battleship game. Luckily, all of the major components are already done from the last assignment. You will need to add some animations and sounds to make your Battleship game more aesthetically pleasing. Your game will still be a 1-player game, however, some steps will be in preparation to make the game multiplayer.

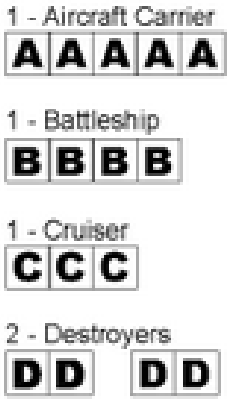
Introduction

If you are worried that your code isn't up to par to continue, don't worry. We have provided you a solution to the previous assignment in order to complete this assignment. We highly recommend that you use your own code though – it is an important skill to be able to build off old code and maintain good reusable code. In addition to this, it will be easier since you should already be familiar with the code you built.

Background

Battleship is a two-player game where each player has a 10x10 grid playing field. This field represents a sea in which players place their ships. One aircraft carrier, one battleship, one cruiser, and two destroyers. Each type of ship is of a different size, but all ships must be placed vertically or horizontally – there is no diagonal placement of ships. The players take turns guessing grid locations of their opponent's ships. Once all grid spaces that are occupied by a ship have been guessed, the ship sinks. The first player to sink all of the opponent's ships wins.
NOTE: This is the same background as last assignment – this is for easy reference.





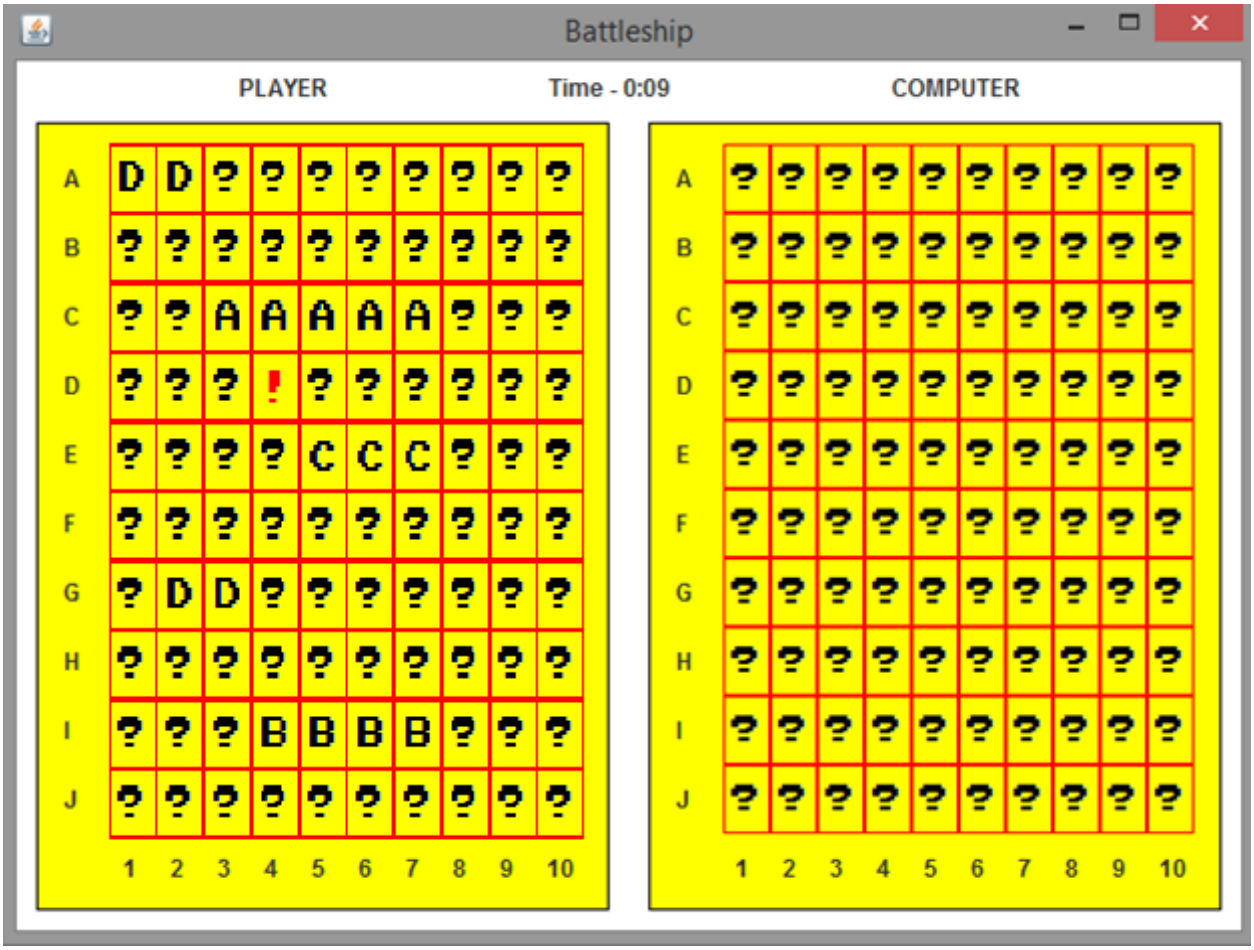
Implementation

Step 1

We need to limit the amount of time a player can take to make a move. Add a timer with a graphical display so the player can see how much time he has left to make a move. Start the timer at 0:15, representing 15 seconds. Once the timer runs out, the round will be over, regardless of whether the player made a move or not. If both players make a move before the time runs out, the timer will reset to 0:15 and the players will be able to make a move again.

Step 2

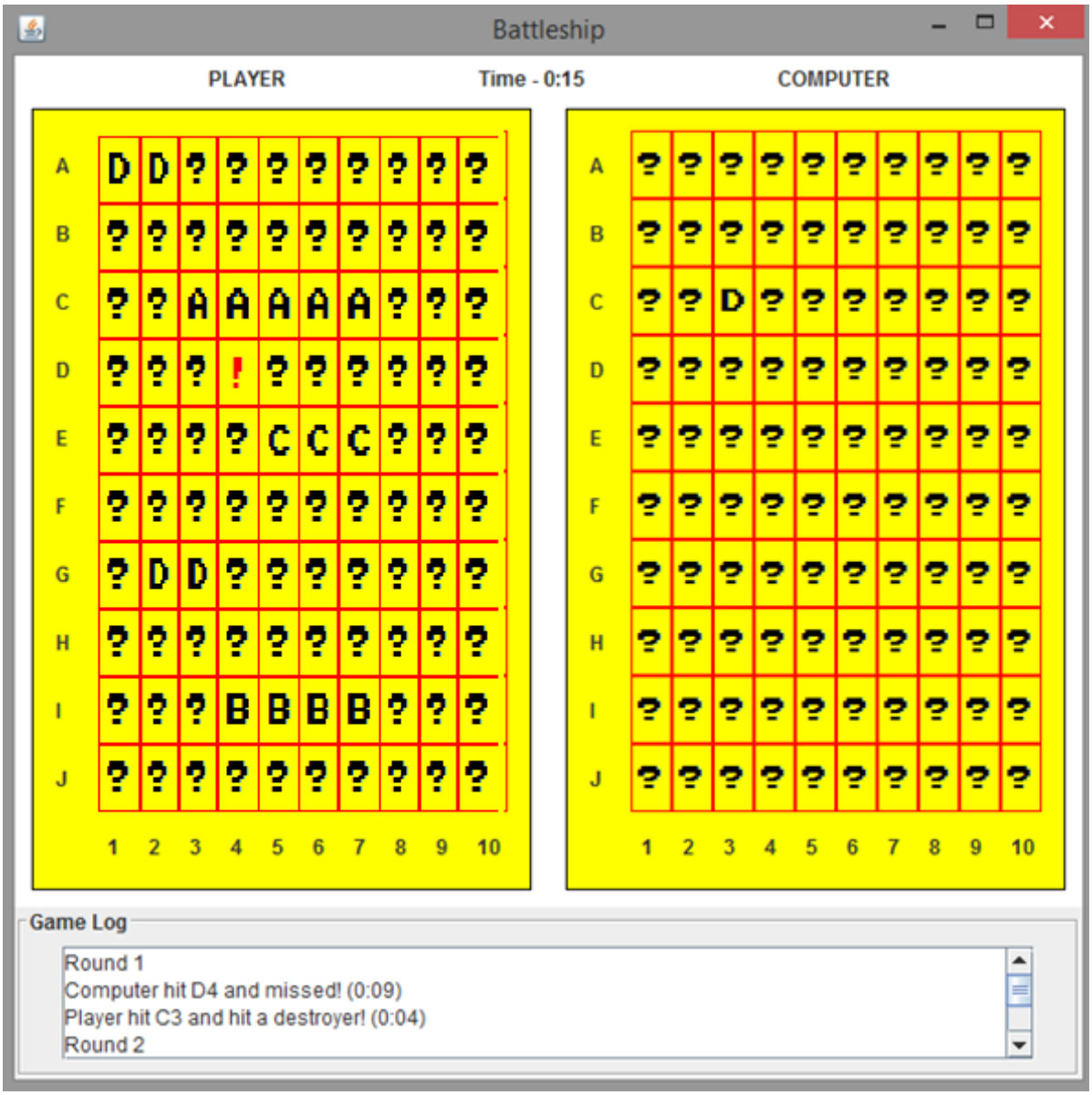
To simulate a real player, add some behavior to the computer. Make the computer wait a random time duration before making a move. The grid spaces the computer picks can still be chosen poorly or more intelligently, but it does not have to be a smart opponent. Make sure the computer can only make one move per round. In the example below, the computer waits 6 seconds before attacking the coordinate D4, and misses.



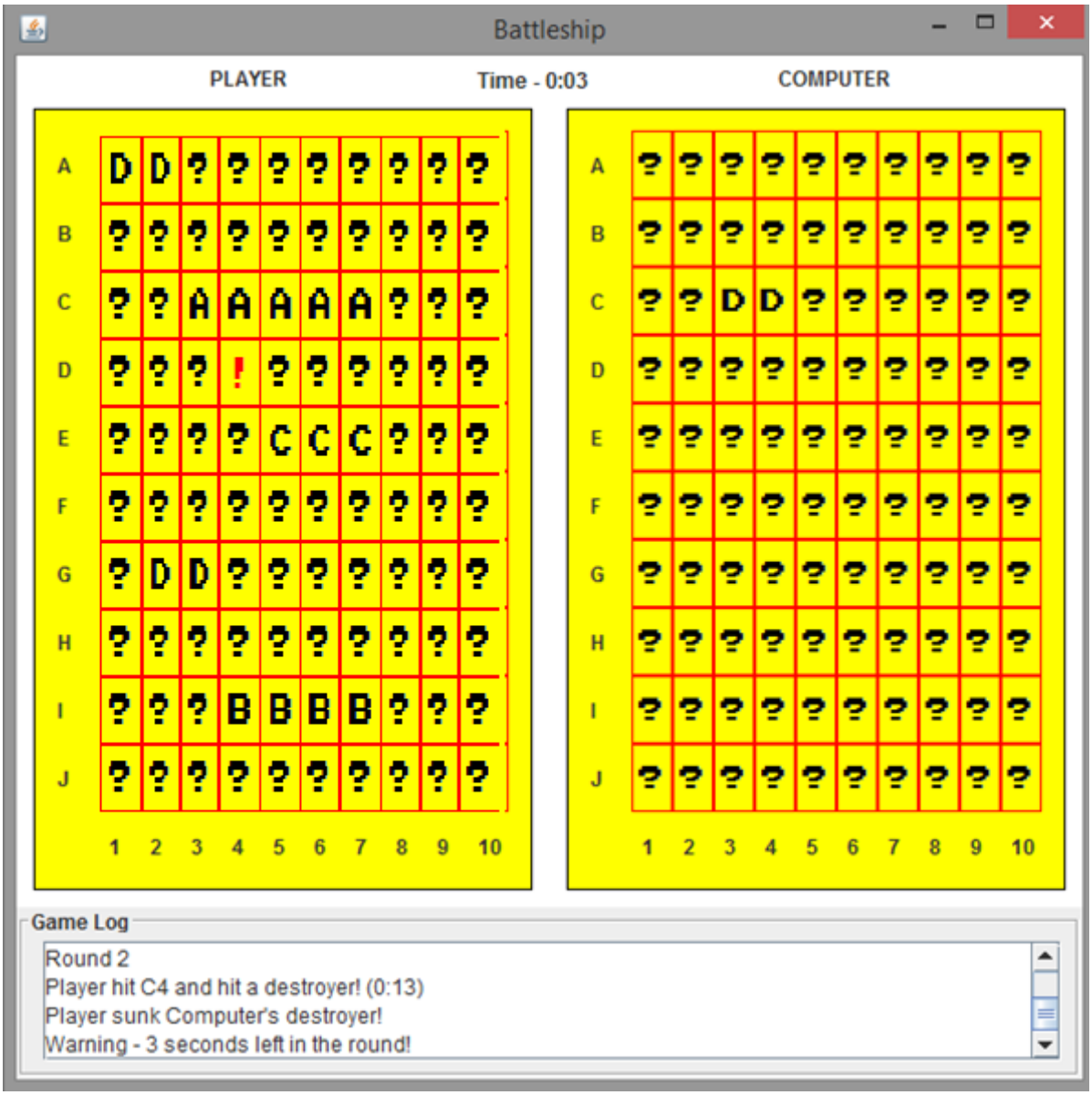
Step 3

Our log from the last assignment was fine for a single-player game since both moves happened simultaneously. Often, this will not be the case, so we need to make a scrollable text log so both players can see the individual moves. In addition to announcing where each player made his move, add in information about when the player made the move, ships being hit, ships being sunk, a warning when there are only 3 seconds left to make a move, and a message for when the round restarts.

Here is an example of two rounds. In round 1, the computer waits 6 seconds before making a move (0:09), and attempts to hit D4. Five seconds later (0:04), the Player makes a move, and hits a destroyer at C3. Even though there were 4 seconds remaining in the round, since both players have attacked, the timer resets to 0:15 and a new round begins.

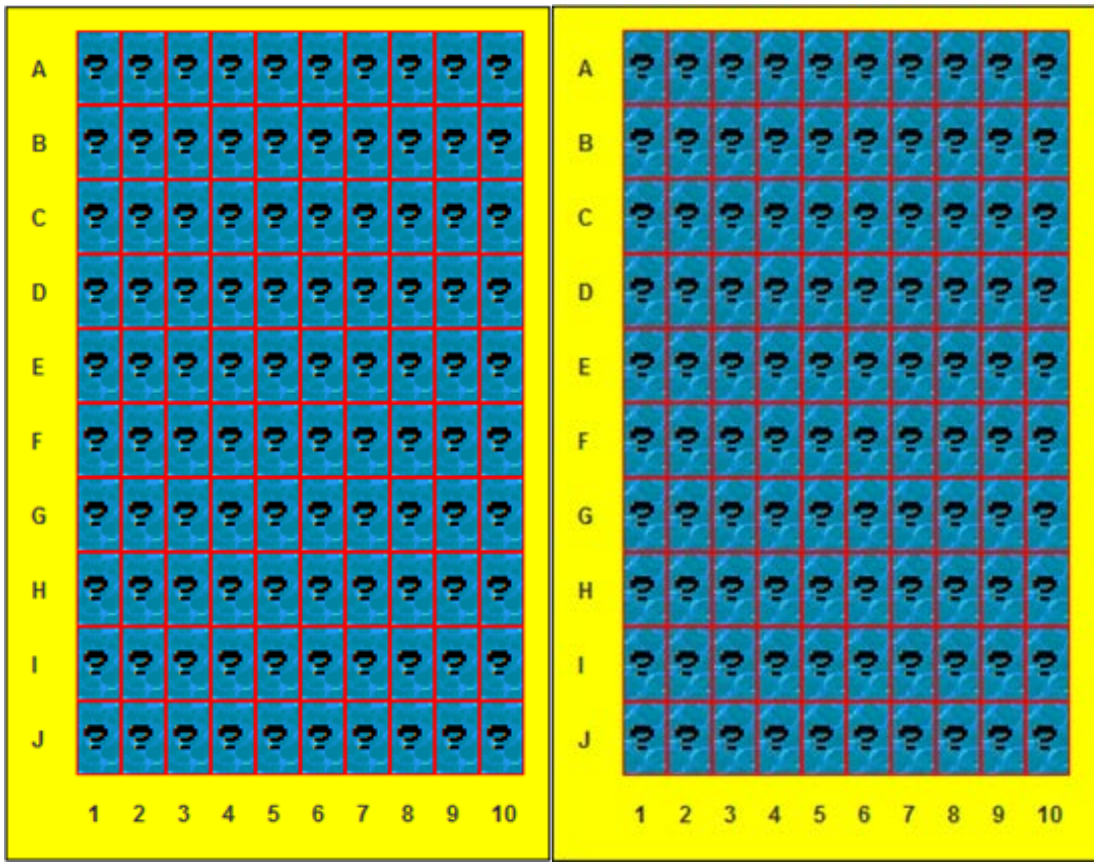


In this next example, the player attacks space C4 and sinks the computer's destroyer. The computer has waited more than 12 seconds, so a log sends a warning that the time is almost up.



Step 4

Right now, the game is fairly boring – we have some cool graphics, but everything is static. Instead of having plain spots all over the grid, use two water textures that alternate back and forth in the background. This way it will look like an ocean with moving water. Some textures have been provided for you, but feel free to make/find your own. Play around with the interval of time in which you switch between the two images to try to make it look fluid. Although it may be hard to see in this document, the below images show two different water images that will oscillate to portray water.

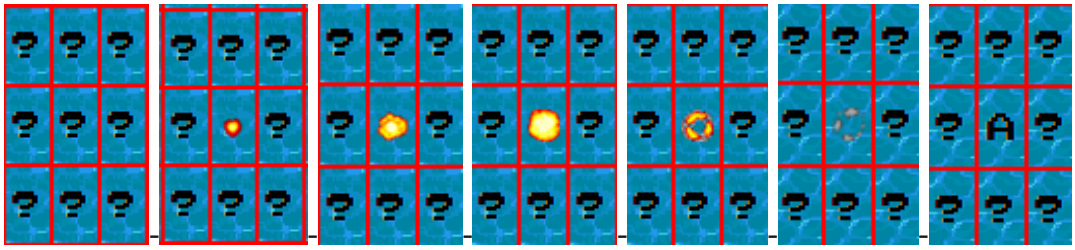


Step 5

Currently, nothing exciting happens when a player clicks on a space - the space simply reveals its associated letter. Instead we want animations to play based on the outcome of the click. If the player hits a ship, we want an explosion animation to play. If the player misses, we want a splash animation to play. These animations will take some time to play, however, the rest of the program should not wait on the animations to finish. If both the computer and the player make a move at the same time, both animations should be able to play at once.

In addition to these animations, add a sinking animation. We will reuse the splash animation for this, but it will happen on all squares of the ship. This animation should play after the explosion animation, and only when the entire ship has been hit.

This is an example of the computer hitting G3, part of an aircraft carrier.



NOTE: Notice how the water background is still updating during this animation.

NOTE: Loading images into memory takes a bit of time, and can take up a lot of space. Make sure all of the images are loaded into memory at the beginning of program execution, and don't load in more than one of the same image.

Step 6

In addition to animation, we want to be able to play sound effects. Play a cannon blast sound each time a player makes a move, an explosion sound when a ship is hit, a splash sound when the player misses, and a sinking sound when a ship is sunk. Playing all the sounds will take some time, but the program should not wait on the sounds to finish. If both the player and the computer make a move at the same time, both sounds should be played on top of one another.

NOTE: Loading sounds into memory takes a bit of time, and can take up a lot of space. Make sure all of the sounds are loaded into memory at the beginning of program execution, and don't load more than one of the same sound.

Step 7

Lastly, we need to make sure that the animations and sounds are synched up. Make sure to slow down or speed up the animations to match the durations of the sounds being played. Also, keep in mind that there is a sound for a player clicking but no animation associated with that. Make sure to delay the start of the explosion animation or splash animation until after the attack sound is played.

In addition to this, we want the animations to finish playing before the game ends. Make sure to delay the win/lose/tie pop-up until after all the animations are done playing.

Grading Criteria

Category and % of grade	Criteria
Timer 0.75%	
0.25%	The timer counts down by 1 every second.
0.25%	The timer resets to 0:15 once both players make a move.
0.25%	The timer resets to 0:15 once the timer runs out of time.
Log 1.25%	
0.25%	The log reports a new round after the timer is reset.
0.25%	The log reports each hit with player name and time as shown in the example.
0.25%	The log reports a warning at 3 seconds remaining in the round.
0.25%	The log reports sunken ships as shown in the example.
0.25%	The log is scrollable and moves automatically as new messages appear.
Updated Graphics 0.5%	
0.25%	The backgrounds of the JLabels/JButtons alternate between two images.
0.25%	The animation of the background does not get interrupted by anything.
Basic Animations 1.0%	
0.25%	A hit animation plays to reveal a boat when one is hit.
0.25%	A splash animation plays to reveal a miss when a boat is not hit.
0.5%	A sinking animation plays over all squares of a ship when sunk.
Sound 0.5%	
0.1%	Cannon blast sound plays when a player makes a move.
0.1%	Explosion sound plays when a player hits a ship.
0.1%	Splash sound plays when a player misses a ship.
0.1%	Sinking sound plays when a ship is sunk.
0.1%	Sounds play over one another.
Syncing 1.0%	
0.2%	Animations can occur together.
0.2%	Program does not wait on any animation.
0.2%	Animations wait for the cannon blast sound to finish.
0.2%	Animations are the same duration as their respective sounds.
0.2%	On Game Over, animation will finish before ending the game.