Mediatonic Air Battle Code Test

Client-side Engineering

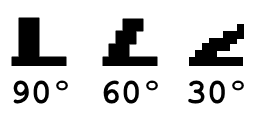
## Introduction

It’s the year 1977 and Betari, a company specialized in creating knockoffs of famous games, has hired you to create a clone of the hugely successful [*Air-Sea Battle*](https://en.wikipedia.org/wiki/Air-Sea_Battle) game. Given the proximity of a famous videogame trading show and the desire to show it to potential clients, the company has asked you to only deliver the first stage of this game and limiting it to 1 player instead of 2.

In this first stage, the player uses a stationary anti-aircraft gun that can be positioned at a 30, 60, or 90 degree angle to shoot down planes. The planes typically appear in groups of three to five, and once every plane in a formation has been destroyed, a new formation appears.

## Requirements

* We’re in an alternate dimension of year 1977 and Unity already exists. Please use version **2019.1.8f1**.
* Use the assets in [this resource pack](https://drive.google.com/a/mediatonic.co.uk/file/d/1UjsWfZBMd4KMiRQdTyRrabbiZatUuUB4/view?usp=sharing).
* Betari is still deciding what’s the best aspect ratio and resolution for the game, so please make it in a way that can work for any possible aspect ratio within the 4:3 to 16:9 range.
* The main menu will show the highest score and a button to start playing. High scores are not saved when the game is closed.
* During gameplay, the stationary gun will be shown at the bottom of the screen, at x = ScreenWidth/4. By default, it’ll point at a 60 degree angle. Holding the *up* arrow key will point at a 30 degree angle, and holding the *down* key will point at 90 degrees. When releasing any of these keys, the gun will return to the default 60 degree angle.



* Upon pressing the *Space* button, the gun will shoot a projectile using its current inclination but in a straight trajectory. To improve the original version of the game, the player will be able to fire multiple projectiles (instead of just 1), but due to the memory restrictions of our target platform, we can only show up to 5 projectiles on screen at the same time.
* Projectiles will either hit one of the enemy planes, or disappear when they cross the screen boundaries.
* The planes will appear in groups of three to five (the number is randomly picked) and will all fly left to right at the same constant speed. Upon reaching the right side of the screen, they will reappear on the left side. We would like to be able to tweak the speed of the planes within the editor (i.e. without having to edit the code)
* There are 3 UI elements shown during gameplay: current score on the top left, time remaining (in seconds) on the top center, and the highest score on the top right.
* Upon hitting a plane, the plane will disappear, an explosion sound effect will be played, and the player score will be increased. When all planes of the same formation are destroyed, a new group of planes is spawned.
* When the time remaining gets to 0, the game will return to the main menu, but apart from the highest score, it’ll also show the last obtained score.
* Betari wants to be able to adjust several parameters remotely. To do this, their central services team has set up a server that allows game developers to retrieve the game configuration via HTTP while the game boots. Your game has been assigned [this](http://content.gamefuel.info/api/client_programming_test/air_battle_v1/content/config/config) endpoint. This configuration comes in Json format and specifies the following: default high score, time limit (in seconds) and number of points to add when destroying a plane, all of them as integers. Because the computer networks of year 1977 are not very stable, upon any connectivity issues or not having a connection at all, please use these default values: default high score: 100; time limit: 30; points per plane: 1.
* Gaming machines from this era are extremely slow. Please do whatever you can to avoid instantiation of gameplay elements while the user is playing the game.
* There are no code style specifications set for this test (despite our constant space vs tab studio wars). However, we will evaluate that you follow consistent rules (e.g. code formatting, variable and function naming conventions…), good programming practices (e.g. meaningful variable and method names, no hardcoded values, error handling…) and produce code that is as clean and easy to read as possible.
* You can include more assets or features if you’d like. However, we recommend you focus on the requested items to produce the best result for those.

## Deliverables

The result should contain the following:

* The source project, contained within a .*zip* file. This file should only contain the minimum amount of files needed to open it and be able to run the game (i.e. avoid including temporary or cache files, as well as any builds of the game)
* A readme file explaining any decisions you made that can help us understand your thought process and any issues you might have come across. If anything wasn’t clear enough in the instructions, please describe any assumptions you’ve made. Do also include an estimate (in hours) of how long it took you to perform the test. There is no set time limit, just take the time you need to produce something you’re proud of.
* A list of any 3rd party assets and tools you used.

Good luck!

Monday - 3 hours

Tuesday - 3 Hours

v1.2 (8th August 2019)