Instigate Design

Parallel Systems Development

Development Specification

Battleship

Version 1.4
Last modified on 29 May 2018
Number of pages 5

Table of Contents

1.   Introduction

1.1.   Document Identifier

This is a development specification document for Battleship game.

1.2.   Scope

Introduction section provides overview of the document content as well as the brief description of the project requirements. Definitions of terms and acronyms section includes description of all the terms and acronyms used in the document. Section References has links to the documentation and sources used to build this document. The architecture of the project is described in Overview of Solution section. Source Code Organization section describes the structure of the project files (sources, documentation), including the folder hierarchy, description of the build and test environments. Description of the solution section includes details on the solution.

1.3.   Definitions of Terms and Acronyms

Glossary

Sea   **–**   the playing board.

Ships   **–**   targets of the player

Shooting **–**   the process of entering the coordinates of a targeted ship.

1.4.   References

Морской бой (игра) This article describes the rules of the game.

2.   Problem Statement

The project should have the following features:

1. Create and print to console a 10x10 playing board.
2. Randomly set 10 ships on the board non visible to the player.
3. Give the player the feature of entering coordinates of the targeted ship.
4. Print messages and update the sea (playing board) according to the entered coordinates.

2.1.   Overview of Solution

Twenty one functions are used in this project.

create_sea();

random_number_generator();

set_ships();

set_ship_length_health();

set_ship_direction();

set_ship_coordinates();

first_ship_coordinates();

next_ship_coordinates(int);

horizontal_ship_coordinates(int);

vertical_ship_coordinates(int);

check_clashes(int);

go_through_new_ship_points(int, int);

go_through_reserved_points(int, int, int);

play();

shoot();

verify_coordinates(std::string, std::string);

convert_x(std::string);

convert_y(std::string);

hit_check(int, int);

health_check(int);

update_sea();

To learn more about each function please type "make doxygen" and "make html" commands in your terminal to see the descriptions of the functions and their arguments.

2.2.   Source Code Organization

- Battleship – the top directory

> - bin – contains the executable file
>
> - docs – contains the documentation of the project
>
> - obj – contains the object files
>
> - src – contains the source code of the project
>
> - test_result –
>
> - tests –
>
> - Doxyfile
>
> - makefile – the top makefile for building the project
>
> - README – short description of the program

3.   Description of the Solution

For the creation of the sea a two dimensional 10x10 char type array is used. Blank points are marked with '0's, hit ship points with 'X's and missed points with '.'s. For the creation of the ships a two dimensional 10x7 int type array is used. Each ship has 7 parameters:

1. Length
2. Direction
3. X1 – the x coordinate of the first point of the ship
4. Y1 – the y coordinate of the first point of the ship
5. X2 – the x coordinate of the last point of the ship
6. Y2 – the y coordinate of the last point of the ship
7. Health

Therefore all the information about ship placement, size and state is stored in the ships[][] array.

The program randomly sets the ships so they don't clash with each other and there is at least one point gap between them.

The program receives coordinates from the player to check if they match with any of the points of the ships set by computer updates the sea and prints the appropriate message. When all the ships are sunk the game is over.