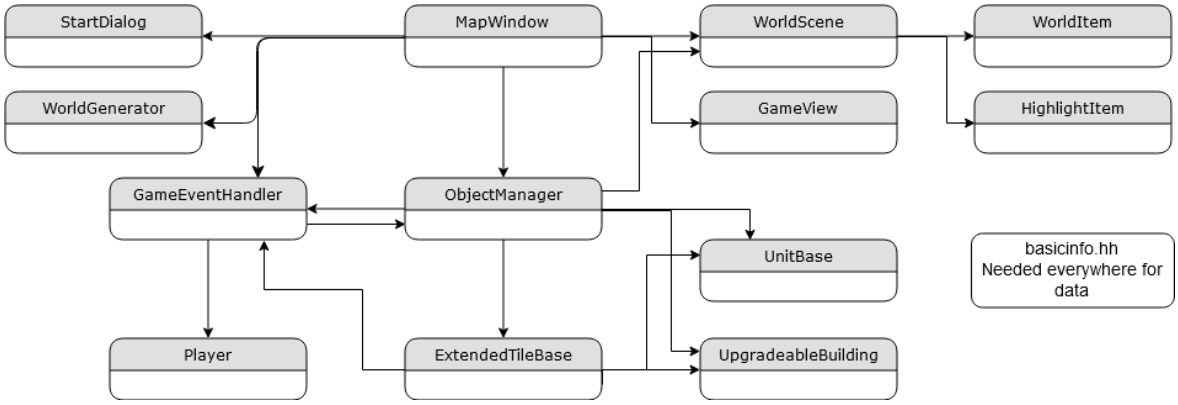
Project documentation



Tile classes: Forest, Grassland, Sand, Stone, Swamp and Water are all inherited from ExtendedTileBase, and they're all referred only by their base class name in the diagram for simplicity's sake.

Building classes: HQ, Campus, Ranch, Fishery, Sawmill, Mine, Market are all inherited from UpgradeableBuilding, and are also skipped from the diagram.

Worker classes: Worker, and Scout, inherited from UnitBase.

Key class responsibilities

GameEventHandler

GameEventHandler is most often called by MapWindow to perform gameplay functions based on GUI button presses. GameEventHandler performs a bunch of sanity testing to ensure gameplay rules are followed, before finally calling other classes such as Player or ObjectManager to do the actions. GameEventhandlers main task is to act as a connecting piece between many different sections of code.

Aside from this, GameEventHandler also keeps track of which players turn it is, and handles turn changing.

ObjectManager

ObjectManagers job is to keep and catalog all tiles, buildings and units in the game. It implements a handful of functions to search and find specific classes as needed elsewhere. In most cases ObjectManagers functions are called from GameEventHandler after testing has already been done, and ObjectManager mainly performs the task with minimal checking.

Player

Player class contains all of players info, including name, chosen color and current resource amounts. Player class implements a few different methods to adjust its resources and a check function to verify if the given player can afford purchases during gameplay.

ExtendedTileBase

ExtendedTileBase is the base class for all tiles in the game, the main difference between it and its base class Course::TileBase is that it redefines resource generation calculations and allows checking how much a tile would produce on the following turn.

UpgradeableBuilding

UpgradeableBuilding is the base class for all buildings in the game, and aside from the stuff handled by its base class Course::BuildingBase it also handles upgrading buildings. By paying extra resources, players can make their buildings more efficient. UpgradeableBuilding keeps track of all the functions related to upgrading and upgrade tiers.

UnitBase

UnitBase is the base class for all units in the game, and besides the stuff from its base class Course::WorkerBase it implements functions for handling and checking movement.

Worker

Worker gets its own mention as it is the only UnitBase inherited class that implements unit specialization. By paying resources, the players can make their workers more efficient at gathering different materials.

GameView

GameView is derived from QGraphicsView and it handles events on the view. Events handled by GameView are mouse scroll for zooming and right click + drag for scrolling.

WorldScene

WorldScene is derived from QGraphicsScene and it is responsible for keeping track of WorldItems and HighlightItems on the map as well as handling mouse events on drawn objects. WorldScene emits signals to MapWindow about clicked objects.

WorldItem

WorldItem is derived from QGraphicsItem and it takes care of drawing individual objects correctly on the scene. WorldItems represent tiles, buildings and workers visual look in the game.

HighlightItem

HighlighItem is derived from QGraphicsItem and it is used to highlight selected tiles on the scene.

MapWindow

MapWindow is responsible for the graphical user interface. It communicates with WorldScene, GameEventHandler and ObjectManager.

StartDialog

StartDialogs job is to get starting values from the user so that the game can be initialized. It does a handful of sanity checking before returning the values to make sure the game can be launched.

Project functionality

During MapWindow initialization, it calls for StartDialog to get important setup information from the user, such as map size or player names. Once suitable starting values are given, MapWindow calls WorldGenerator to generate the board and passes information forward to GameEventHandler.

GameEventHandlers task is then to create the Player classes according to player-chosen parameters. After this, it calculates starting locations for each player and initializes them with HQ and two starting units. After this, the game is ready to start.

MapWindow class is the graphical user interface that the player uses to control the game. It is derived from QMainWindow class. MapWindow contains buttons, text areas and other Qt widgets. MapWindow communicates mainly with WorldScene, GameEventHandler and ObjectManager. Most of the functionality of MapWindow is in slots that are connected to user interface buttons and WorldScene. MapWindow uses basicinfo.hh for showing the information about buildings and units.

Most of the user interaction happens in GameView and in QStackedWidget which is located in the right panel of graphical user interface. QStackedWidget is called menuWidget and contains six different menu pages: mainMenu, tileMenu, buildingMenu, workerMenu, specializeMenu and buildMenu. Four of those menus show information about the game or the currently selected object in a text area. Those menus contain buttons which are enabled or disabled depending on possible actions. BuildMenu and specializeMenu buttons are created dynamically according to the parameters in basicinfo.hh.

Buttons have tooltips that gives hints about button actions. Buttons for building, upgrading, hiring and specializing show the resource cost of action in a tooltip. All tooltips are activated by hovering the cursor over a button.

Tiles, buildings and workers are pixmap images. Image files are handled by a resource file graphicsresources.qrc. Images are drawn to the game in WorldItem class using QPainter class.

GameEventHandler is responsible for checking if users commands can be executed before calling other classes functions to actually perform them. GameEventHandler also handles switching turns and automating resource generation between turns.

Each of the other classes are called as requested by user inputs, and their documentation was already explained above in the key class responsibilities section.

Extra feature documentation

Each tile, building and worker has their own graphics. Graphics were drawn using paint.net. GameView is implemented to allow smooth zooming and moving on the map.

Campus research system was implemented to have a main goal in the game. Game winner is the player who builds the campus and finishes the OHJ3 project first. Research is done by assigning units to work at the campus.

Building upgrade system is a new game mechanic that allows players to upgrade their buildings up to tier three. Upgrading increases the amount of resources generated by the building.

Unit specialization allows players to promote basic workers to different professions. There is one profession for each game resource and they are more efficient in producing the resources.

Moving workers out of the captured area is prohibited so we created a Scout class that allows capturing new areas outside already captured area. Scouts can move longer distances than workers but they do not produce any resources.

The map size, the number of players and player names are adjustable prior to starting the game.

Division of work

At the beginning work was split so that each would take one interface class to implement, but as project moved forward we switched so that Valto took over MapWindow and graphics side of code, and Jaakko handled GameEventHandler and ObjectManager.

Gameplay manual

Win Condition

The first player to build a campus and finish the OHJ3 research project within wins! Research progresses by having any worker on the campus, but Teekkaris are particularly good at research.

Special Features

- Workers and buildings can be upgraded to increase their effectiveness. In the case of Outposts, upgrading expands the area the building captures around itself.
- Outposts can be built on scout locations to capture faraway pieces of land.
- Buildings can be sold for half of its original build cost

Building costs

Farm, Fishery, Sawmill, Mine, Market, Outpost: 100 Wood

Campus: 1000 money, 1000 food, 1000 wood, 1000 stone and 1000 ore

Building upgrades

Upgrades to tier 2: 100 money, 100 stone

Upgrades to tier 3: 400 money, 100 ore

Unit costs

Worker: 100 food

Scout: 100 money, 100 food

Unit upgrades

Farmer, Lumberjack, Miner, Merchant: 200 money, 100 food

Teekkari: 1000 money, 100 food

How to earn resources

Tiles and buildings only generate resources if they have workers on them. Each building and worker specialization focuses on one particular resource type. Specialized units are twice as effective for their particular focus, but only half as good for everything else. Each building upgrade tier roughly doubles the materials earned from that tile.

Farm & Fishery & Farmer = food

Market & Merchant = money

Sawmill & Lumberjack = wood

Mine & Miner = stone & ore

Bugs & missing features

Certain button texts and unit names don't fit within their constraints and get clipped.

UI is lacking styling and more game information should be added to be visible for the user.

Random world generation occasionally places Headquarters and workers on tiles they're not normally supposed to be allowed on, such as water tiles.

Ability to sell excess materials at Marketplaces was planned but not implemented due to time constraints.

Some graphics were planned to be animated but finding good sprite sheets licensed for free use was hard and drawing them would take a lot of time.