# Brief description.

1. Entry point

GlobalController class of Start scene is entry point of game. It makes some basic initialization and starts StartState.

1. Runtime states.

State pattern is used to isolate logic states of game into separate classes.

There are global states: StartState (loading of game, database), MainMenuState (logic of main menu operations), LevelState (playing one of game levels).

LevelState has substates for each step of level gameplay.

1. Database.

Unity ScriptableObject component is used for database.

There are 2 assets:

* DBLevels is describing game levels. Levels could be modified, added or removed.
* DBScreenIds holds string ids of UI windows and panels.

Singleton pattern is used to provide access to database data at runtime.

1. Scene.

Level specific stuff are located in Level scene. It is asynchronously loaded and unloaded while in LevelState.

1. ObjectPool.

ObjectPool pattern is used to retain performance during spawn and destruction of matching elements.

1. Logic and UI separation.

Logic and UI are separated by means of Observer pattern variation (Bus). No logic state class knows about UI class and vice versa.

1. UI

I’m using side opensource UI framework deVoid to speedup prototyping of games. I would instantiate UI prefabs manually for test task but prefer cut that corner.

1. Commentaries.

MatchFinder.cs is example of appropriate code commentaries format. Basically, all classes should have such clarification information.

Please note I retain intellectual property rights of source code of the test task application.