Typing Analysis

Dima Trushin

Contents

1 General information

The application uses Qt environment as an ecosystem. It does not use RTTI but uses exceptions. All components of the program must be inside NSApplication namespace. Nested namespaces should be placed in a subfolder of the project.

2 Addressable objects

I use term addressable object. It means an object that can be referenced by other objects. There are several types of addressable objects:

1. QObject
2. CApplicationImpl
3. Observer/Observable (TO DO)

Addressable objects must be created on heap via `std::make_unique` or similar mechanism. An exception to this rule: you may create an addressable object inside another addressable object.

3 Exceptions

The strategy is to catch exception, show a message, and die. Since Qt environment is not totally exception safe, it is not allowed to throw exceptions in an event loop. If a QObject requires to throw an exception and terminate the program it sends a signal to QApp, QApp shows a message, and terminates execution (TO DO).

4 Application Structure

CApplication object initialize all required resources for the application including the ones to interact with Qt ecosystem. It may throw an exception while constructing. In order to minimize stack usage CApplication contains `std::unique_ptr` to CApplicationImpl (it is addressable object). CApplicationImpl consists of four parts:

1. **CApplicationGlobals**. Its purpose is to initialize global resources, e.g., timers, loggers, thread pools, etc. Application initializes all global resources (basically singletons) at the start. This allows not to wast time on the first call. Also, it is inconvenient to initialize timers via the first call.
2. **CApplicationKernel**. Its purpose is to initialize the kernel of the application. The kernel does not depend on the GUI and uses MVC via observer pattern to interact with GUI.

3. **CApplicationGUI**. Its purpose is to provide View wrappers over Qt resources compatible with MVC pattern.
4. **CApplicationImpl**. Its purpose is to connect the kernel and the GUI via MVC.

The order of construction is ensured by the inheritance mechanism.