What is the source code?

This is a Visual Studio 2013 solution that includes all needed to build the application. It was created and tested for Prepar3d version 3.1 and SDK version 3.1.2.15831 (actually on January 2016). This is not trivial multithread application that met all Prepar3D SDK requirements. To build the application just load the solution in Visual Studio and press ‘F7’, that all.

I advise to look at fsqar.h at the beginnig

The code is work correctly. This is not trivial multithread application that met all Prepar3D SDK requirements.

This is a very bad code! This is not C++ code but "C with objects" one. It breaks many basic principles of Objected-Oriented Programming. It is badly designed and ugly written. I do not advise to use it directly without any modifications. This code may be used only as a starting point for developing more advanced application. Let me explain why it is. In the first, this code does not use standard C++ library (STL) and other templates (i.e. it does not use general programming paradigm). In the second, it does not use any exception instead of it the code use return code like Win32 API. The code is mixt from plain C and objects. The design of the objects is not comprehensible. Can we say that the code is true C++ if it is not build on the base of C++ philosophy?

How to improve the program?

There is an infinite set of ways to improve the code. But why? The program has a narrow target and the target is achieved.

Description of the main idea

The basic idea is to create a multithreaded application in which the main thread is UI thread and the second working thread is SimConnect dispatcher. These threads are exchanging of synchronization events. It is quite clear how to create the main thread. I propose the next solution for the working thread. The flight simulator is a message server. Our application is its client. In this code, the working thread is running into a message loop from the simulator. This loop is a copy of the flight simulator SDK examples. However, in any moment the 'Exit' message from the main UI thread can be received. The function "IsTerminated" was placed in the loop for checking of the "Exit" event. When this event is occurred the command "Quit" is sending and the client escape from the message loop from the server. The client should be having two handles: The SimConnect handle and the working thread handle. CSimClient wraps these handles.