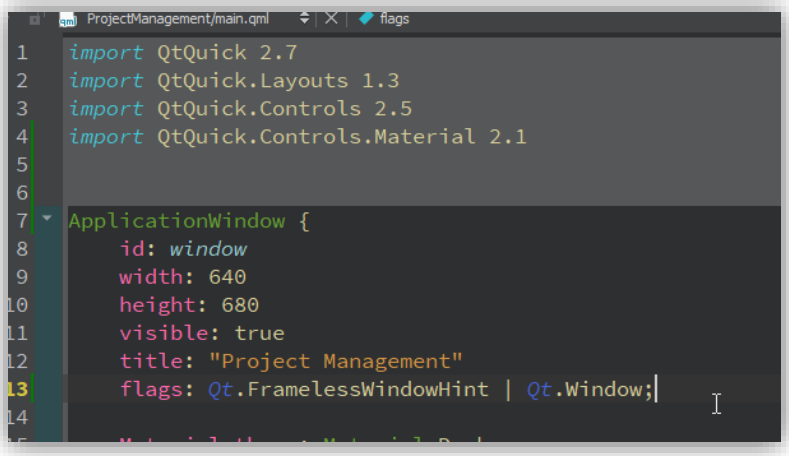
Frameless ApplicationWindow in Qml.net with aero snap functionalities

✓ Problem Description

Using frameless window in qml (ApplicationWindow object) made the application to lose windows aero snap features. Functionalities such as, docking to left/right by pressing win-left/right keys, full screen by moving the window to the top or minimizing other windows by shaking. All these features will no longer be available if use frameless window in qml:

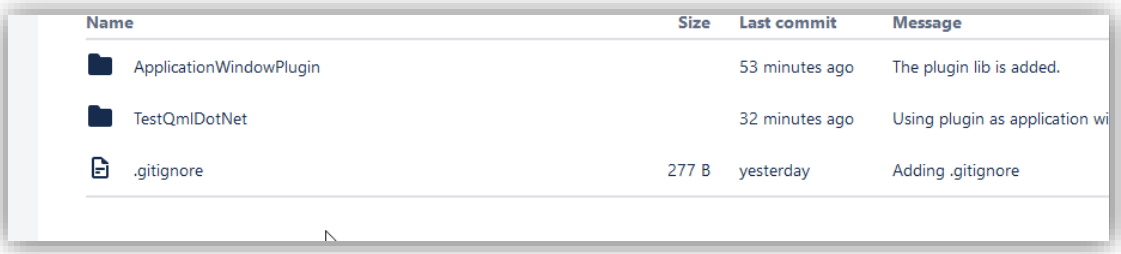


✓ Solution (Managed Hosting)

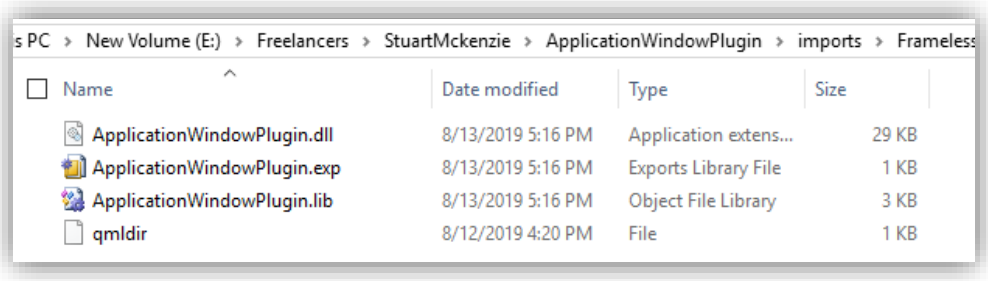
Since qml.net is just a thin layer, which calls Qt API to run the qml stuff, then if we fix this issue in qt/c++, then, it will be fixed in qml.net, too.

The idea is to write and customize our own ApplicationWindow object and add it to the qml plugins. First, we have to use qt/qtcreator to write a custom qml plugin. After that, we should use windows API (dwmapi.h) to make the window act like other normal windows.

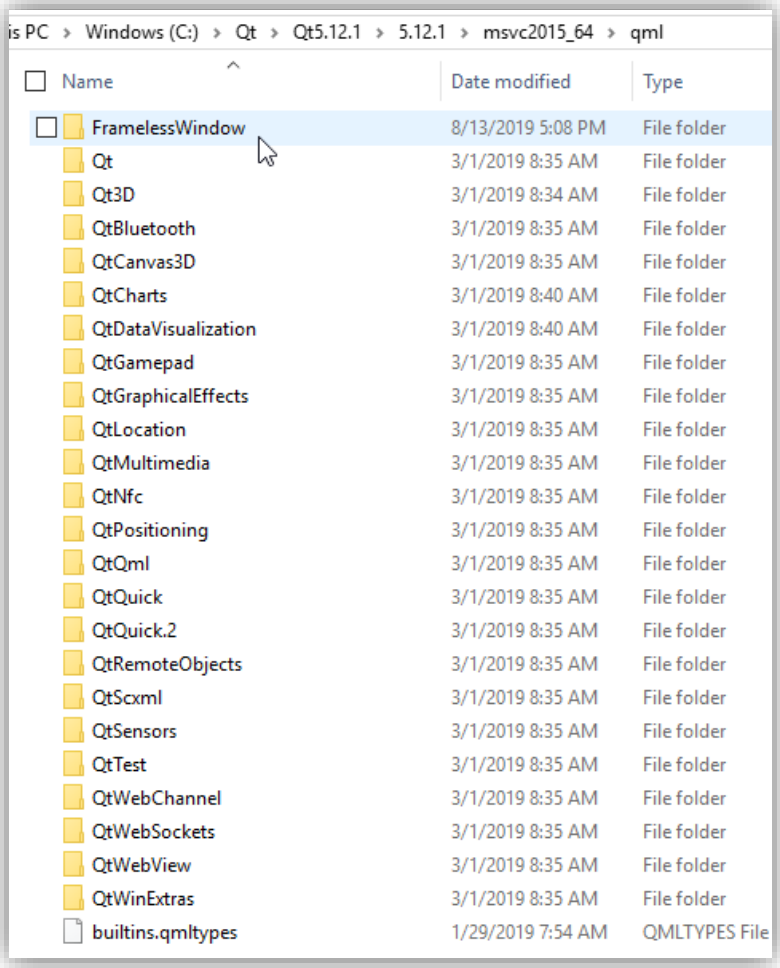
Currently the plugin (ApplicationWindowPlugin) is developed and exists in the repository:



Let me skip the details of the plugin, but after building the plugin with qtcreator, a folder containing some files will be created within the plugin directory:



These are the plugin files (Release version only). We have to copy the folder (FramelessWindow folder) into plugins directory (in Qt installation path):



As soon "FramelessApplication" is pasted in this folder (all the qml plugins are stored), we can import it in Qt/Qml and use this plugin. So, anyone else who is using the same Qt version (5.12 msvc2015 64bit) can use this library files (doesn't need to compile the lib again).

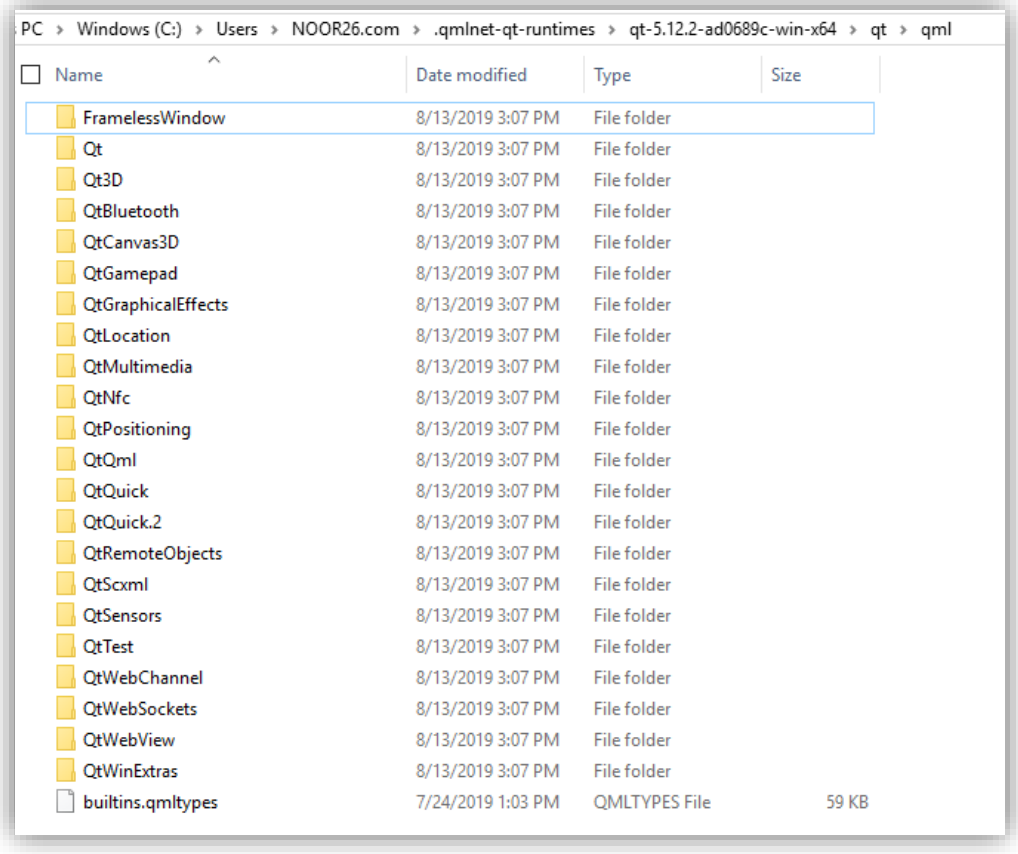
To use the plugin in the qml, we need to do some changes (line 5 is added and line 7 is changed):

```
ProjectManagement/main.qml | title
1  import QtQuick 2.7
2  import QtQuick.Layouts 1.3
3  import QtQuick.Controls 2.5
4  import QtQuick.Controls.Material 2.1
5  import FramelessWindow 1.0
6
7  FramelessAppWindow {
8      id: window
9      width: 640
10     height: 680
11     visible: true
12     title: "Project Management"
13
14     Material.theme: Material.Dark
15     Material.accent: '#4885cc'
16     Material.primary: '#4885cc'
17     property alias footerLabel: footerLabel;
```

Now the window will have aero snap features completely.

- ✓ **Using the plugin in qml.net**

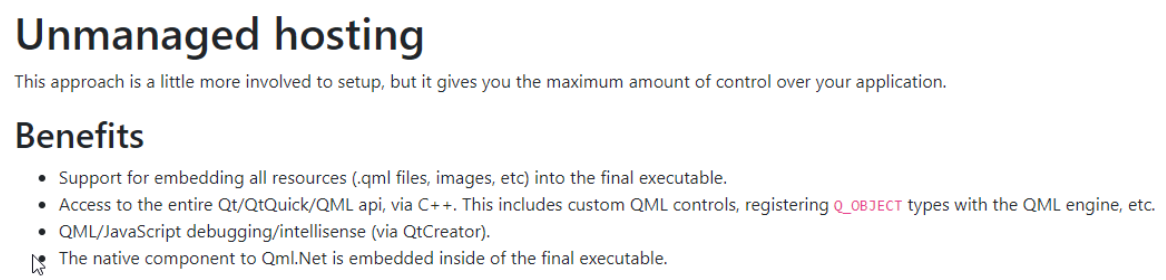
But qml.net will fail to run this qml files since it will search qml plugins in a different directory. All qt runtime files are downloaded at the first time and are stored in username/.qmlnet-qt-runtimes. So we need to copy the plugin folder to this folder, too (please pay attention to the address bar):



After this, qml.net can find this plugin and will work without any issues.

- ✓ **Solution (Unmanaged Hosting)**

Although the issue is fully resolved by using managed hosting, there are reasons that maybe in future, we need to use unmanaged hosting instead of managed hosting. The reasons are well written in the Paul page (Benefits):



Unmanaged hosting

This approach is a little more involved to setup, but it gives you the maximum amount of control over your application.

Benefits

- Support for embedding all resources (.qml files, images, etc) into the final executable.
- Access to the entire Qt/QtQuick/QML api, via C++. This includes custom QML controls, registering `Q_OBJECT` types with the QML engine, etc.
- QML/JavaScript debugging/intellisense (via QtCreator).
- The native component to Qml.Net is embedded inside of the final executable.

Since we have extended the plugin, it's not going to be any problems for us whereas we use managed or unmanaged hosting. So the situation doesn't affect the way we have fixed aero snap features.