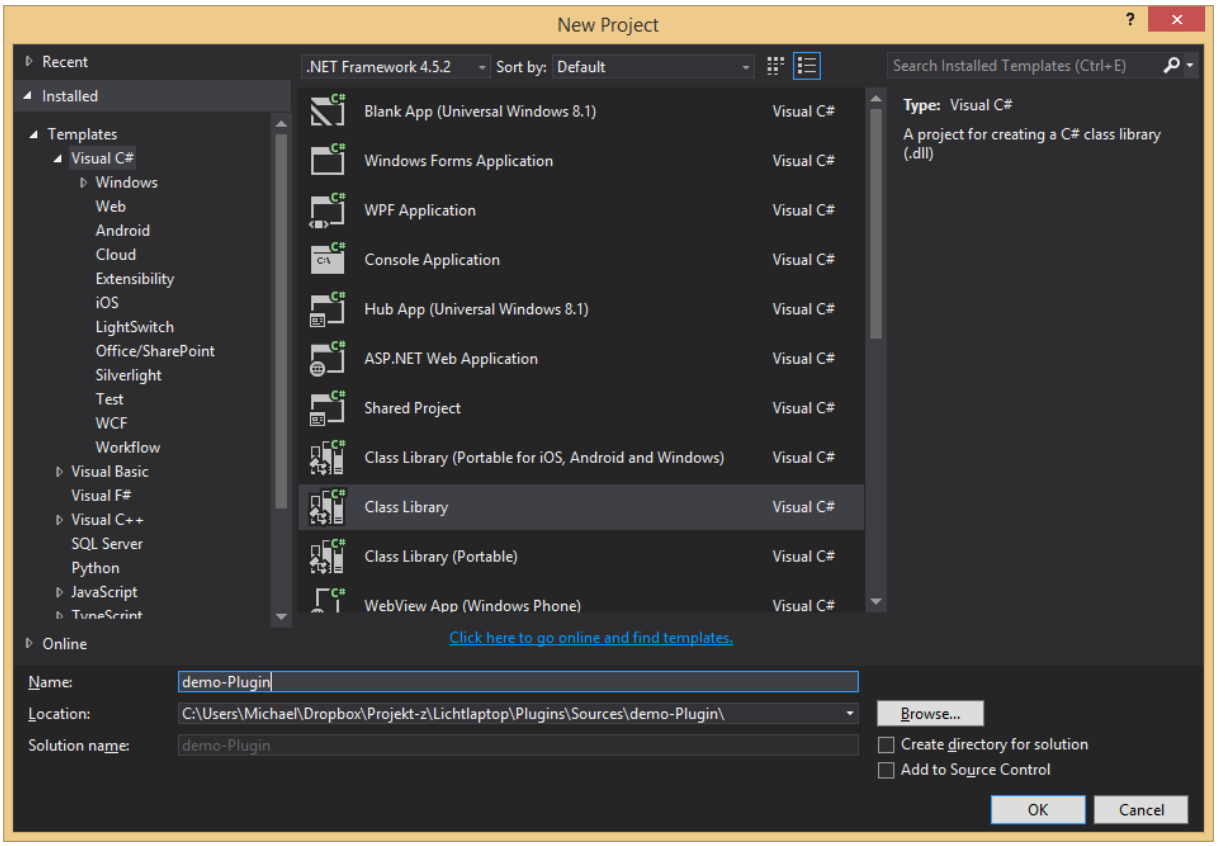
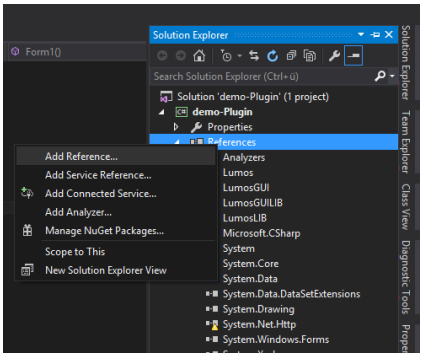
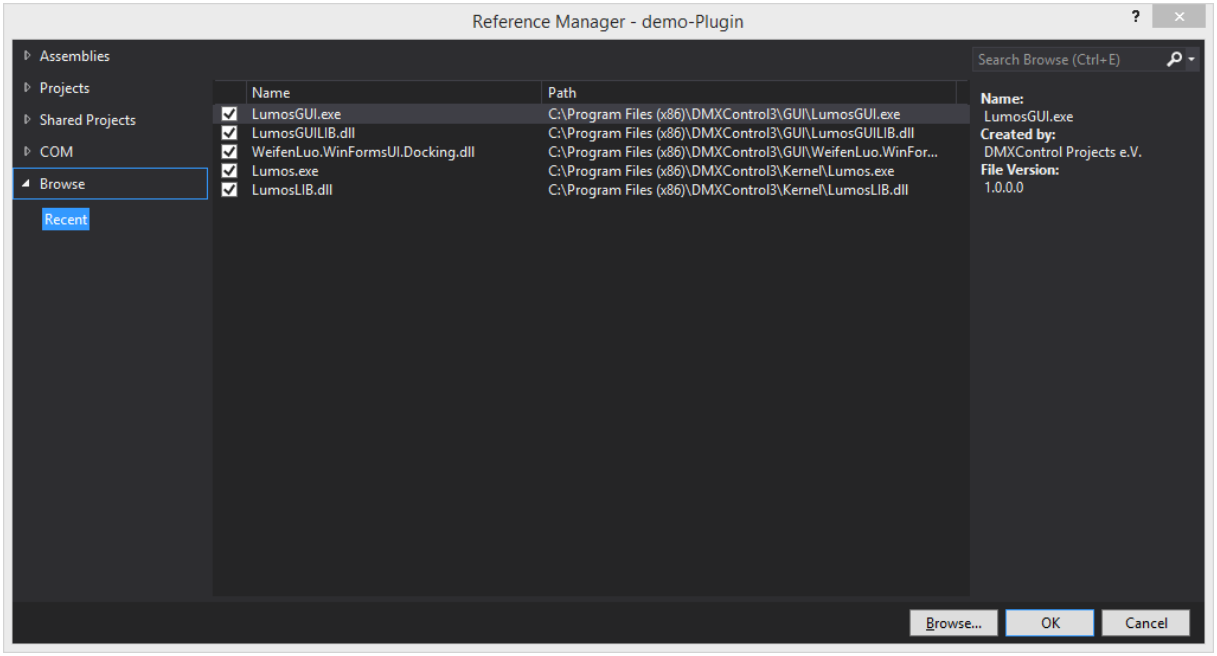
Erstellen eines Plugins

Step 1: Anlegen eines neuen Projekts in Visual Studio in Form einer Class Library (DLL).

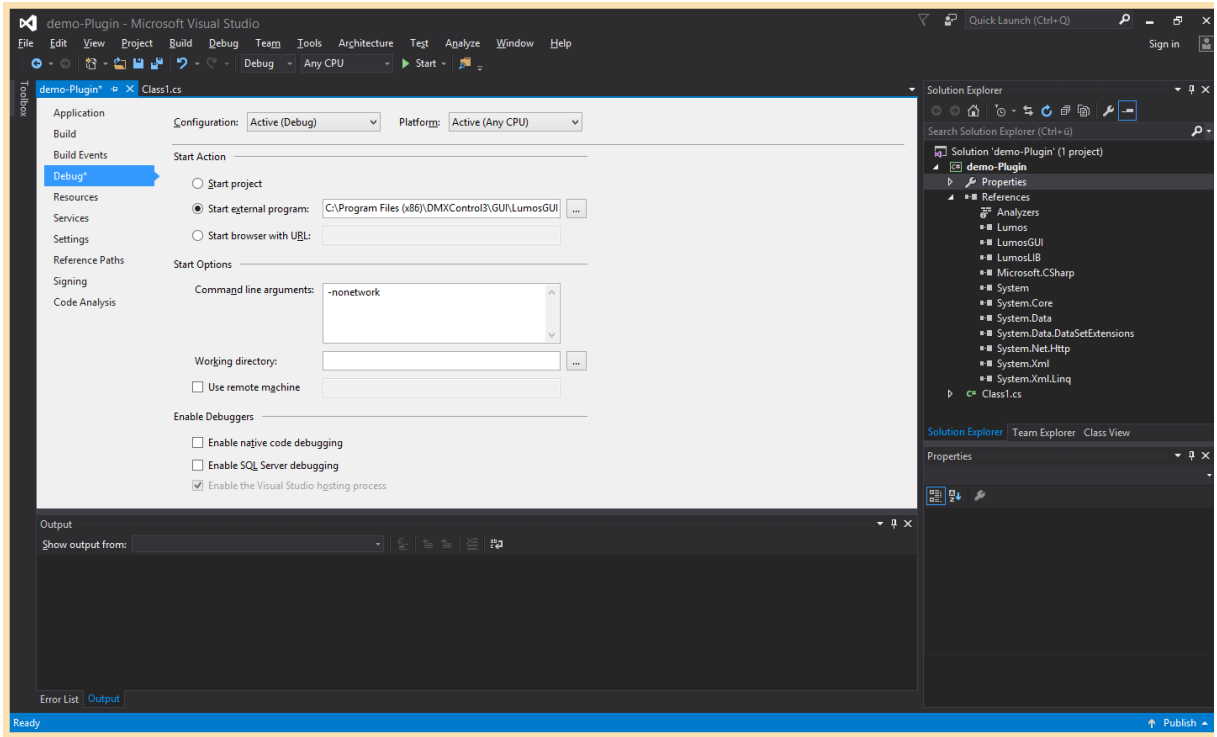


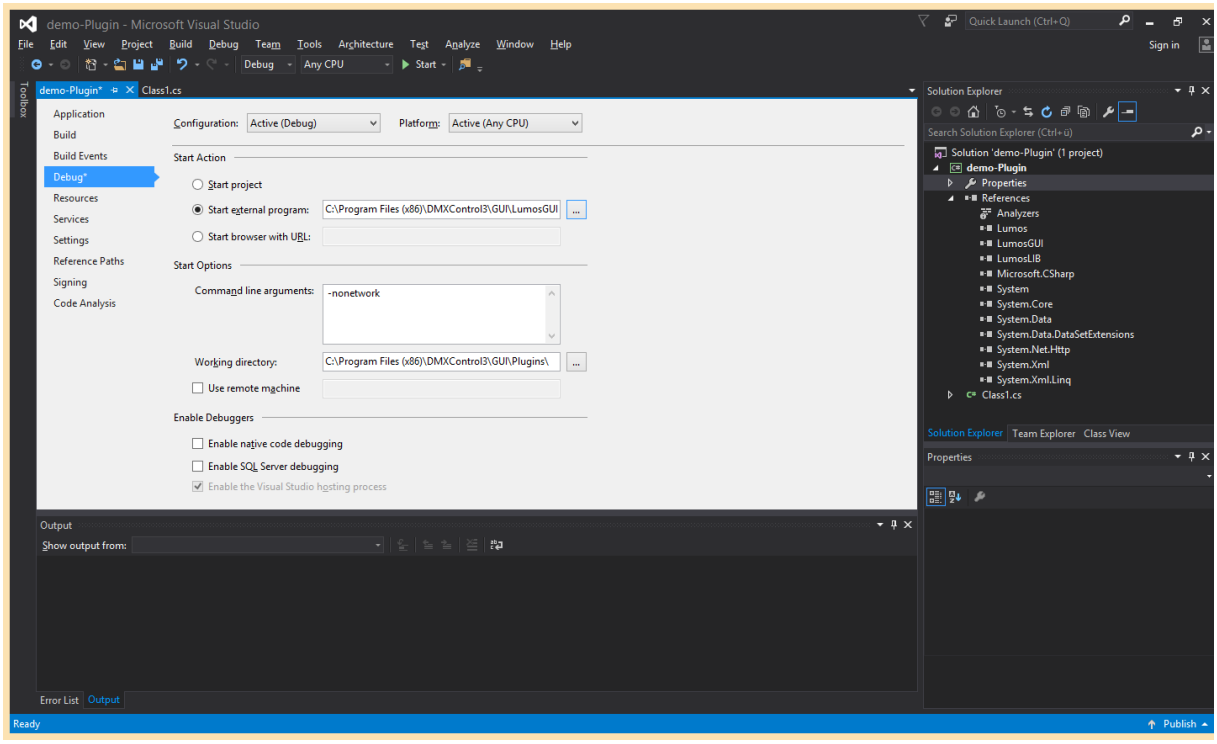
Je nach Plugin (GUI oder Kernel) sollten untenstehende Dateien über „add References“ eingebunden werden.



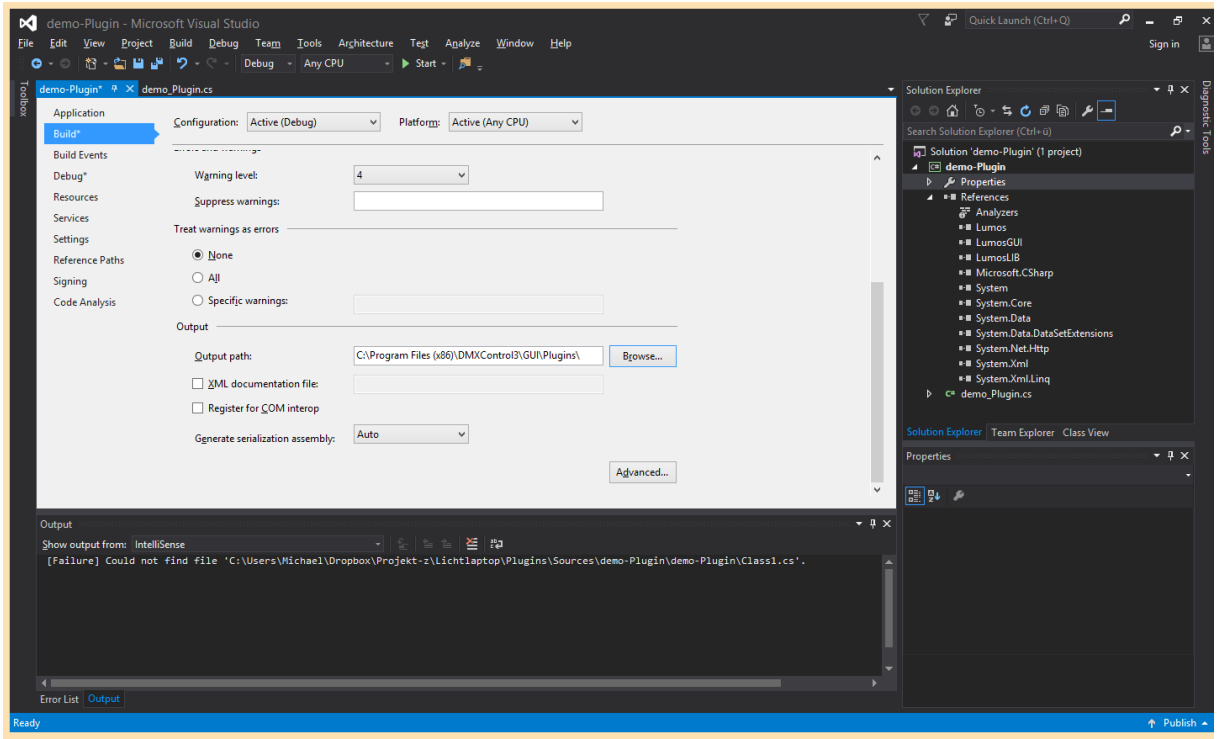


Zum ordentlichen Debuggen habe ich folgende Einstellungen getroffen:





Den Outputpath ebenfalls so spezifizieren, dass er in das Pluginverzeichnis des Kernels oder der GUI zeigt. So kann im Visual Studio Debugging betrieben werden. Je nach Installationspfad von DMXC ist es notwendig VisualStudio mit Administratorrechten auszuführen.



Standardcode für GUI Plugins

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Lumos.GUI;
using Lumos.GUI.Plugin;
using LumosLIB.Kernel.Log;

namespace demo_Plugin
{
    public class demo_Plugin : GuiPluginBase
    {
        /// <summary>
        /// Every Plugin has a unique ID
        /// </summary>
        const string PLUGIN_ID = "{BA54E029-4666-41a1-96C7-455CDF113983}";

        /// <summary>
        /// A logger to write Logfiles
        /// </summary>
        private static readonly ILumosLog log =
LumosLogger.getInstance(typeof(demo_Plugin));

        public demo_Plugin()
            : base(PLUGIN_ID, "demo-Plugin")
        {

        }

        /// <summary>
        /// Called when Plugin is initialized (on Startup)
        /// </summary>
        protected override void initializePlugin()
        {
            return;
        }

        /// <summary>
        /// Called when Plugin is disabled
        /// </summary>
        protected override void shutdownPlugin()
        {
            return;
        }

        /// <summary>
        /// Called when Plugin is activated (enabled in PluginManager)
        /// </summary>
        protected override void startupPlugin()
        {
            return;
        }
    }
}
```

Logger:

Mit

```
        private static readonly ILumosLog log =
LumosLogger.getInstance(typeof(demo_Plugin));
```
wird der Logger initialisiert.

Mit den Befehlen:
```
log.Debug("Debugmessage");
log.Info("Infomessage (shown in GUI)");
log.Warning("Warning (shown in GUI)");
log.Error("Error (shown in GUI)");
```
können entsprechende Infos in den Logfiles hinterlegt werden. Nur die Kategorien Info, Warning und Error werden in der GUI unter Logs dargestellt.

Eigene Einstellungen hinzufügen

```
using org.dmxc.lumos.Kernel.Settings;
using Lumos.GUI.Settings.PE;
```