

# Lesson 1. Hello λ·vue!

In this lesson, we demonstrate how to setup the environment necessary to start writing programs using  $\lambda$  vue magnification engine (**MagEngine**).

## Environment Preparation

In order to use **MagEngine**, we need to configure project by specifying the **MagEngine's** *INCLUDEPATH* and *LIBS* in the .pro file. Assuming Lambda Vue SDK is installed in the default location, add the following lines.

For Windows, use:

```
INCLUDEPATH += C:/LambdaVueSDK/include  
LIBS += -LC:/LambdaVueSDK/lib -lMagEngine -lavformat -lavcodec -lavutil -lswscale
```

For Linux (Ubuntu), use:

```
INCLUDEPATH += /opt/LambdaVue/include/  
LIBS += -L/opt/LambdaVue/include/ -lMagEngine -lavformat -lavcodec -lavutil -lswscale  
-lcurl -lgomp
```

For Mac OS, use:

```
INCLUDEPATH += /Library/LambdaVueSDK/sdk/include/  
LIBS += -L/Library/LambdaVueSDK/sdk/lib/ -lMagEngine -lstdc++
```

The updated .pro file now looks like this:

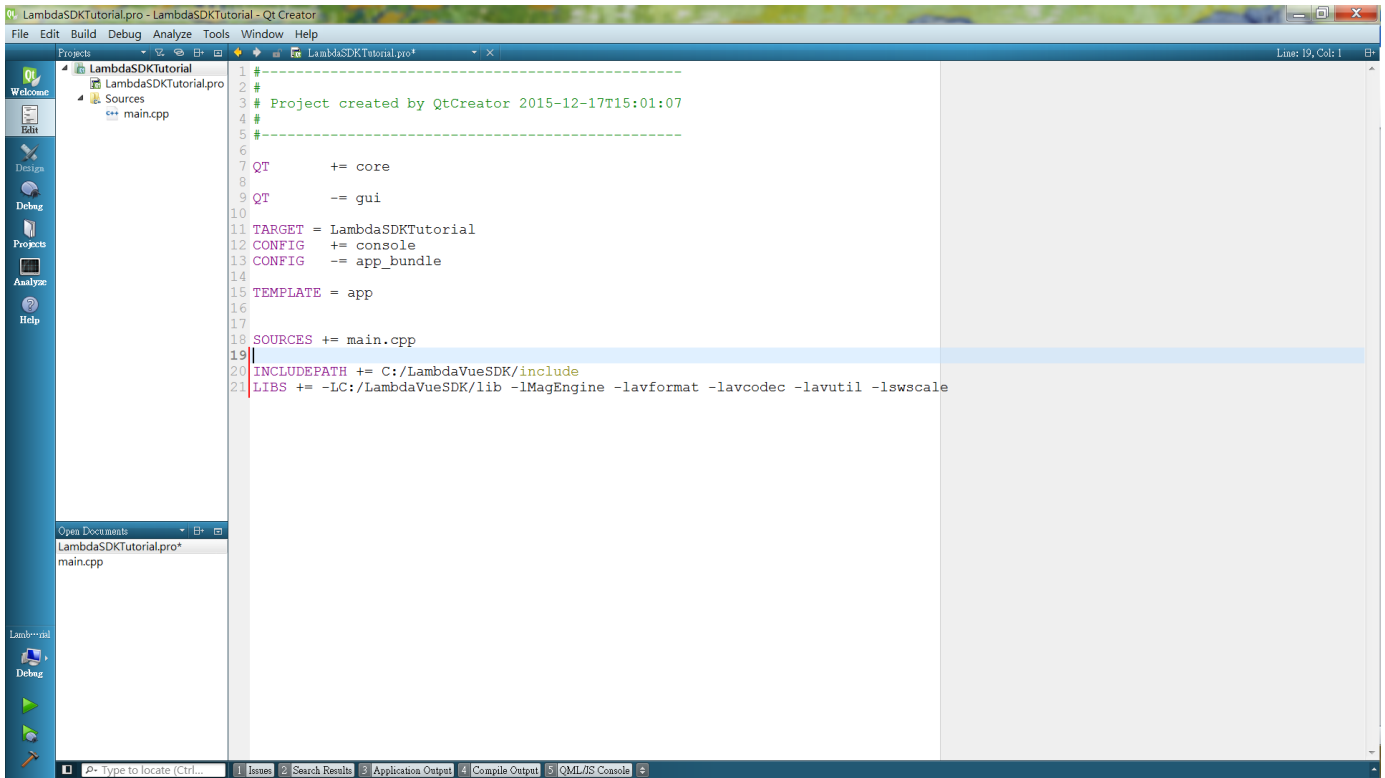


Figure 1. Updated .pro file

## My First $\lambda$ -vue Program

Here is what our first program looks like. Essentially, this program does two things:

1. Print out the current version of  $\lambda$ -vue SDK
2. Check the validity of  $\lambda$ -vue license

```

#include <QCoreApplication>
#include <iostream>
/* Include the main Lambda SDK header file */
#include <MagEngineAPI.h>

using namespace std;
using namespace geko;

/* Callback function to handle various Lambda events */
void cbFunction(struct EventValue value) {
    cout << "Event " << value.event << " triggered" << endl;
}

int main(int argc, char *argv[]) {
    QCoreApplication a(argc, argv);

    /* Display the current Lambda SDK version */
    cout << "Hello World, I am using Lambda SDK version: "
         << MAG_ENGINE_VERSION() << endl;

    /* Initilize Lambda engine using defined Callback function as parameter
     * and return the license status (See SDK manual for states)
     */
    enum LicenseResponse license_status = initMagEngine(cbFunction);

    /* Check the license state, continue only if the license is valid */
    cout << "Lambda license status: " << license_status << endl;
    if (license_status <= 0) {
        cout << "Valid license" << endl;
    } else {
        cout << "Invalid license" << endl;
    }

    /* Destroy Lambda Engine before exiting the program */
    destroyMagEngine();

    return a.exec();
}

```

## Line-by-line Explanation

```
#include <MagEngineAPI.h>
```

Include the main header file before we can do anything

```
using namespace geko;
```

Use **MagEngine**'s namespace *geko*

```
void cbFunction(struct EventValue value) {  
    cout << "Event " << value.event << " triggered" << endl;  
}
```

Define a callback function to handle events generated by **MagEngine**. Right now, we just print out the event ID. For a complete description of these events, please refer to the  $\lambda$ -vue SDK API Manual.

```
cout << "Hello World, I am using Lambda SDK version: "  
    << MAG_ENGINE_VERSION() << endl;
```

Print out the Hello World message and show the current SDK version number

```
enum LicenseResponse license_status = initMagEngine(cbFunction);
```

Initialize **MagEngine** and return the current license status value

```
cout << "Lambda license status: " << license_status << endl;  
if (license_status <= 0) {  
    cout << "Valid license" << endl;  
} else {  
    cout << "Invalid license" << endl;  
}
```

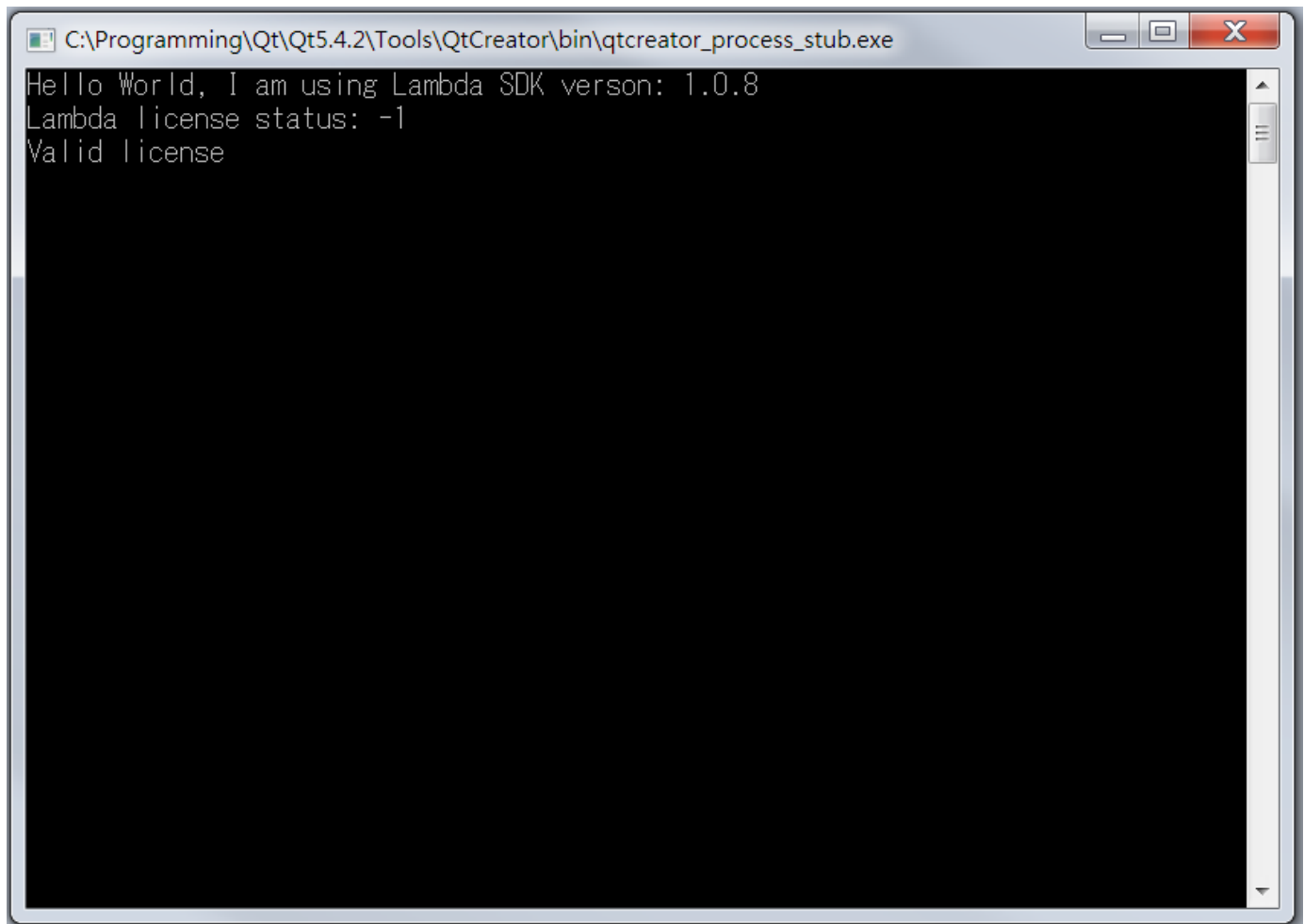
Print out the license status value and check to see if it is valid. For a complete description of what each of these value means, please refer to the  $\lambda$ -vue SDK API Manual.

```
destroyMagEngine();
```

Terminate **MagEngine** before quitting the program

# Program Output

When you run this program, you should get the following output.

A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Programming\Qt\Qt5.4.2\Tools\QtCreator\bin\qtcreator\_process\_stub.exe. The window contains the following text:

```
Hello World, I am using Lambda SDK version: 1.0.8  
Lambda license status: -1  
Valid license
```

*Figure 2. Lesson 1 Output*

Congraduation! You've now completed your first  $\lambda$ -vue program.