PyQt for Autodesk Maya 2017 64bit

Written by Cyrille Fauvel – Autodesk Developer Network (April 2013)
Updated by Vijaya Prakash – Autodesk Developer Network (November 2016)

Building SIP, and PyQt for Maya 2017

PyQt [http://www.riverbankcomputing.co.uk] is a python binding to the Qt library. Because Maya uses Qt internally, you can use the PyQt modules in Maya python scripts to create custom UI. PyQt does not have the same licensing as Maya, Qt, or Python. Please consult the PyQt website for information about licensing for PyQt.

Download PyQt: http://www.riverbankcomputing.com/software/pyqt/download

Download SIP: http://www.riverbankcomputing.com/software/sip/download

The following are instructions for building a copy of the PyQt modules that have been known to work with Maya.

Maya 2017 uses Qt5.6.1 which is binary compatible with the latest version of PyQt – 5.7 / SIP - 4.18.1

Note that it's important to use the Maya modified version of the Qt source code. A copy of the customized Qt 5.6.1 source is available from Autodesk's Open Source web-site (http://www.autodesk.com/lgplsource) and includes text files describing how to configure, build and install Qt for each platform supported by Maya.

However, be aware that with Maya 2017, there is no more need to build PySide as it is coming by default in Maya, nor you have to rebuild Qt itself as the main Qt tools to build PyQt are now included in the Maya distributions (I.e. qmake, moc, …)

libxml, openSSL, OpenAL, python2.7, qt-5.6.1, and tbb are also coming by default in the Maya include and lib folders, so unless you have a very specific need, you would not need to rebuild any of those libraries like before. Note as well that there is a 'MAYA_LOCATION/support/opensource' folder now which contains some of the community source.

Important: Maya 2017 now ships without the devkit, include and mkspecs folders. You can get the Maya 2017 devkit from the Autodesk Apps Exchange Store here for Windows, OSX, and Linux. Download the devkit and unzip the files into your Maya root folder. Make sure to read the instructions to install the devkit, include and mkspecs folders properly on your system.

The scripts used in this document are now also posted on Github.

Download SIP and PyQt source from 'http://www.riverbankcomputing.co.uk' - here I downloaded 'sip-4.18.1' and 'PyQt5_gpl-5.7'. Unzip them in one folder, then you should get something like this:

Mac

/Users/cyrille/Documents/_Maya2017Scripts/sip-4.18.1
/Users/cyrille/Documents/_Maya2017Scripts/PyQt5_gpl-5.7
'/Users/cyrille/Documents/_Maya2017Scripts' being my local folder. Now the instructions, and bash scripts to build that SIP and PyQt.

Follow the instructions from the API docs to setup your environment (Developer Resources > API Guide > Setting up your build environment > Mac OS X environment, in the Maya Documentation)

Untar the /devkit/include/qt-5.6.1-include.tar.gz into /devkit/include/Qt

The qt.conf file uses MAYA_LOCATION and DEVKIT_LOCATION to locate the expected header/library files. Therefore, users must set both environment variables before building the PyQt5.

DEVKIT_LOCATION should point to the directory where the devkit include, mkspecs, cmake directories are located.

Edit …/bin/qt.conf like this:

```
[Paths]
Prefix=
Libraries=$(MAYA_LOCATION)/MacOS
Binaries=$(DEVKIT_LOCATION)/devkit/bin
Headers=$(DEVKIT_LOCATION)/include/Qt
ArchData=$(DEVKIT_LOCATION)
Data=$(DEVKIT_LOCATION)
HostData=$(DEVKIT_LOCATION)
HostBinaries=$(DEVKIT_LOCATION)/devkit/bin
HostLibraries=$(MAYA_LOCATION)/MacOS
```

Untar the qt-5.6.1-mkspecs.tar.gz into $MAYA_LOCATION/Maya.app/Contents/bin/mkspecs.
Make sure the qconfig.pri looks like this:

qconfig.pri

```
#configuration
CONFIG += release def_files_disabled exceptions no_mocdepend stl
x86_64 qt #qt_framework
QT_ARCH = macosx
QT_EDITION = OpenSource
QT_CONFIG += minimal-config small-config medium-config large-
config full-config no-pkg-config dwarf2 phonon phonon-backend
accessibility opengl reduce_exports ipv6 getaddrinfo ipv6ifname
getifaddrs png no-freetype system-zlib nis cups iconv openssl
corewlan concurrent xmlpatterns multimedia audio-backend svg
script scripttools declarative release x86_64 qt #qt_framework

#versioning QT_VERSION = 5.6.1
QT_MAJOR_VERSION = 5
QT_MINOR_VERSION = 6
QT_PATCH_VERSION = 1
```

```
QT_NAMESPACE =
QT_NAMESPACE_MAC_CRC =
```

Build & Install SIP

```
#!/usr/bin/env bash
MAYAPYQTBUILD="`dirname \"$0\"`" # Relative
export MAYAPYQTBUILD="`( cd \"$MAYAPYQTBUILD\" && pwd )`" #
Absolutized and
normalized
cd $MAYAPYQTBUILD

export SIPDIR=$MAYAPYQTBUILD/sip-4.18.1
export MAYA_LOCATION=/Applications/Autodesk/maya2017

cd $SIPDIR
$MAYA_LOCATION/Maya.app/Contents/bin/mayapy ./configure.py --
arch=x86_64
make
sudo make install
```

Build & Install PyQt

```
#!/usr/bin/env bash

MAYAPYQTBUILD="`dirname \"$0\"`" # Relative
export MAYAPYQTBUILD="`( cd \"$MAYAPYQTBUILD\" && pwd )`" #
Absolutized and
normalized
cd $MAYAPYQTBUILD

export MAYA_LOCATION=/Applications/Autodesk/maya2017
export QTDIR=$MAYA_LOCATION/Maya.app/Contents
export QMAKESPEC=$QTDIR/mkspecs/macx-clang
export INCDIR_QT=$MAYA_LOCATION/devkit/include/Qt
export LIBDIR_QT=$QTDIR/MacOS

if [ ! -f $QMAKESPEC/qmake.conf ];
then
  echo "You need to install qt-5.6.1-mkspecs.tar.gz in
$QTDIR/mkspecs
!"
  exit
fi
if [ ! -f $INCDIR_QT/QtCore/qdir.h ];
then
echo "You need to uncompress $MAYA_LOCATION/devkit/include/qt-
5.6.1- include.tar.gz in $INCDIR_QT !"
  exit
```

```
/Applications/Autodesk/maya2017/Maya.app/Contents/Resources
if [ ! -f $QTDIR/bin/qt.conf ];
then
  echo "You need to copy $QTDIR/Resources/qt.conf in $QTDIR/bin !"
  exit
fi

test=`grep "Data=../.." $QTDIR/bin/qt.conf`
if [ ! -z "$test" ];
then
  echo "You need to edit $QTDIR/bin/qt.conf to use 'Data=..'"
exit
fi
test=`grep "Headers=../../include" $QTDIR/bin/qt.conf`
if [ ! -z "$test" ];
then
echo "You need to edit $QTDIR/bin/qt.conf to use
'Headers=../../../devkit/include/Qt'"
exit
fi
test=`grep "Libraries=../lib" $QTDIR/bin/qt.conf`
if [ ! -z "$test" ];
then
  echo "You need to edit $QTDIR/bin/qt.conf to use
'Libraries=../MacOS'"
  exit
fi
test=`grep "Plugins = qt-plugins" $QTDIR/bin/qt.conf`
if [ ! -z "$test" ];
then
  echo "You need to edit $QTDIR/bin/qt.conf to use 'Plugins=../qt-
plugins'"
  exit
fi
test=`grep "Translations = qt-translations" $QTDIR/bin/qt.conf`
if [ ! -z "$test" ];
then
  echo "You need to edit $QTDIR/bin/qt.conf to use
'Translations=../qt-
translations'"
  exit
fi

for mod in Core Designer DesignerComponents Gui Help Multimedia
Network OpenGL Script ScriptTools Sql Svg WebKit Xml XmlPatterns
do
  if [ ! -f $QTDIR/MacOS/libQt5${mod}.dylib ];
  then
    echo "You need to copy a fake Qt$mod dylib - cp
$QTDIR/MacOS/Qt5$mod
$QTDIR/MacOS/libQt5${mod}.dylib !"
    #cp $QTDIR/MacOS/Qt5$mod $QTDIR/MacOS/libQt5${mod}.dylib
  exit
fi
done
```

```
export DYLD_LIBRARY_PATH=$QTDIR/MacOS
export DYLD_FRAMEWORK_PATH=$QTDIR/Frameworks
export SIPDIR=$MAYAPYQTBUILD/sip-4.18.1
export PYQTDIR=$MAYAPYQTBUILD/PyQt5_gpl-5.7
     export
SIP_EXE=$QTDIR/Frameworks/Python.framework/Versions/2.7/bin/sip
      export
SIP_INCLUDE=$QTDIR/Frameworks/Python.framework/Versions/2.7/includ
e/python2.7


cd $PYQTDIR
export PATH=$QTDIR/bin:$PATH
$QTDIR/bin/mayapy ./configure.py LIBDIR_QT=$LIBDIR_QT
INCDIR_QT=$INCDIR_QT MOC=$QTDIR/bin/moc --sip=$SIP_EXE --sip-
incdir=$SIP_INCLUDE -w --no-designer-plugin
make -j 8
sudo make install
```

You're done! go to the testing paragraph at the end of the article.

Linux

/home/cyrille/Documents/_Maya2017Scripts/sip-4.18.1
/home/cyrille/Documents/_Maya2017Scripts/PyQt5_gpl-5.7

'/home/cyrille/Documents/_Maya2017Scripts' being my local folder. Now the instructions, and bash scripts to build SIP and PyQt.

Follow the instructions from the API docs to setup your environment (Developer Resources > API Guide > Setting up your build environment > Linux environments (64 bit), in the Maya Documentation).

The qt.conf file uses MAYA_LOCATION and DEVKIT_LOCATION to locate the expected header/library files. Therefore, users must set both environment variables before building the PyQt5.

DEVKIT_LOCATION should point to the directory where the devkit include, mkspecs, cmake directories are located.

Edit your qt.conf file (/usr/autodesk/maya2017/bin) like below

```
[Paths]
Prefix=
Libraries=$(MAYA_LOCATION)/lib
Binaries=$(MAYA_LOCATION)/bin
Headers=$(DEVKIT_LOCATION)/include/Qt
ArchData=$(DEVKIT_LOCATION)
Data=$(DEVKIT_LOCATION)
HostData=$(DEVKIT_LOCATION)
HostBinaries=$(MAYA_LOCATION)/bin
```

Untar the /include/qt-5.6.1-include.tar.gz into /include/Qt

Untar the /mkspecs/qt-5.6.1-mkspecs.tar.gz into /mkspecs

Make qmake, moc executables from the Maya bin directory

```
sudo chmod aog+x /usr/autodesk/maya2017/bin/moc
sudo chmod aog+x /usr/autodesk/maya2017/bin/qmake
```

Build & Install SIP

```
#!/usr/bin/env bash
MAYAPYQTBUILD="`dirname \"$0\"`" # Relative
export MAYAPYQTBUILD="`( cd \"$MAYAPYQTBUILD\" && pwd )`" #
Absolutized and normalized
cd $MAYAPYQTBUILD

export SIPDIR=$MAYAPYQTBUILD/sip-4.18.1
export MAYA_LOCATION=/usr/autodesk/maya2017

cd $SIPDIR
$MAYA_LOCATION/bin/mayapy ./configure.py
make
sudo make install
```

Build & Install PyQt

```
#!/usr/bin/env bash
MAYAPYQTBUILD="`dirname \"$0\"`" # Relative
export MAYAPYQTBUILD="`( cd \"$MAYAPYQTBUILD\" && pwd )`" #
Absolutized and normalized
cd $MAYAPYQTBUILD

export MAYA_LOCATION=/usr/autodesk/maya2017
export QTDIR=$MAYA_LOCATION
export QMAKESPEC=$QTDIR/mkspecs/linux-g++-64
export INCDIR_QT=$MAYA_LOCATION/include/Qt
export LIBDIR_QT=$QTDIR/lib

if [ ! -f $QMAKESPEC/qmake.conf ];
then
  echo "You need to install qt-5.6.1-mkspecs.tar.gz in
$QTDIR/mkspecs !"
  exit
fi
if [ ! -f $INCDIR_QT/QtCore/qdir.h ];
then
  echo "You need to uncompress $MAYA_LOCATION/include/qt-5.6.1-
include.tar.gz in $INCDIR_QT !"
  exit
fi
# qt.conf - /usr/autodesk/maya2017/bin
if [ ! -f $QTDIR/bin/qt.conf ];
then
  echo "You need to copy $QTDIR/Resources/qt.conf in $QTDIR/bin !"
  exit
fi
```

```
     test=`grep 'Headers=$(DEVKIT_LOCATION)/include/Qt'
$QTDIR/bin/qt.conf`
      if [ -z "$test" ];
      then
        echo "You need to edit $QTDIR/bin/qt.conf to use
      'Headers=$(DEVKIT_LOCATION)/include/Qt'"
        error=1
      fi

     export SIPDIR=$MAYAPYQTBUILD/sip-4.18.1
      export PYQTDIR=$MAYAPYQTBUILD/PyQt5_gpl-5.7

      cd $PYQTDIR
      export PATH=$QTDIR/bin:$PATH
      $QTDIR/bin/mayapy ./configure.py LIBDIR_QT=$LIBDIR_QT
      INCDIR_QT=$INCDIR_QT MOC=$QTDIR/bin/moc -w --no-designer-plugin -g
      make -j 8
      sudo make install
```

You're done! go to the testing paragraph at the end of the article.

Windows

D:__sdkext_Maya2017 Scripts\sip-4.18.1
D:__sdkext_Maya2017 Scripts\PyQt5_gpl-5.7

'D:__sdkext_Maya2017 Scripts' being my local folder. Now the instructions and scripts to build SIP and PyQt.

Follow the instructions from the API docs to setup your environment (Developer Resources > API Guide > Setting up your build environment > Windows environment (64-bit), in the Maya Documentation)

Edit your qt.conf file (C:\Program Files\Autodesk\Maya2017\bin) like below

```
[Paths]
Prefix=$(MAYA_LOCATION)
Libraries=lib
Binaries=bin
Headers=include/Qt
Data=.
Plugins=qt-plugins
Translations=qt-translations
Qml2Imports=qml
```

Unzip the /include/qt-5.6.1-include.tar.gz into /include/Qt

Unzip the /mkspecs/qt-5.6.1-mkspecs.tar.gz into /mkspecs

Build & Install SIP

```
@echo off

set MAYAPYQTBUILD=%~dp0
set MAYAPYQTBUILD=%MAYAPYQTBUILD:~0,-1%
```

```
subst v: "%MAYAPYQTBUILD%"

set SIPDIR=v:\sip-4.18.1
set MSVC_DIR=C:\Program Files (x86)\Microsoft Visual Studio 11.0
if ["%LIBPATH%"]==[""] call "%MSVC_DIR%\VC\vcvarsall" amd64

set MAYA_LOCATION=C:\Program Files\Autodesk\Maya2017

set INCLUDE=%INCLUDE%;%MAYA_LOCATION%\include\python2.7
set LIB=%LIB%;%MAYA_LOCATION%\lib

pushd %SIPDIR%
"%MAYA_LOCATION%\bin\mayapy" configure.py
nmake
nmake install
popd
```

Build & Install PyQt

```
@echo off

set MAYAPYQTBUILD=%~dp0
set MAYAPYQTBUILD=%MAYAPYQTBUILD:~0,-1%
if exist v:\nul subst v: /d subst v: "%MAYAPYQTBUILD%"
v:

set MAYA_LOCATION=C:\Program Files\Autodesk\Maya2017
if exist m:\nul subst m: /d
subst m: "%MAYA_LOCATION%"
set MAYA_LOCATION=m:

set QTDIR=%MAYA_LOCATION%
set MSVC_VERSION=2012
set QMAKESPEC=%QTDIR%\mkspecs\win32-msvc%MSVC_VERSION%
set _QMAKESPEC_=win32-msvc%MSVC_VERSION%

set QMAKESPEC=%QTDIR%\mkspecs\%_QMAKESPEC_%
if not exist "%QMAKESPEC%\qmake.conf" (
        echo "You need to uncompress %MAYA_LOCATION%\mkspecs\qt-5.6.1-
mkspecs.tar.gz !"
        goto end
)
if not exist "%MAYA_LOCATION%\include\Qt\QtCore\qdir.h" (
        echo "You need to uncompress %MAYA_LOCATION%\include\qt-5.6.1-
include.tar.gz in %MAYA_LOCATION%\include\Qt !"
        goto end
)
findstr /L /C:"Headers=include/Qt" "%MAYA_LOCATION%\bin\qt.conf" >nul 2>&1
if ERRORLEVEL 1 (
        echo "You need to edit %MAYA_LOCATION%\bin\qt.conf to use
'Headers=include/Qt'"
        goto end
)
```

```
findstr /L /C:"-lqtmain -lshell32" "%QTDIR%\mkspecs\common\msvc-desktop.conf"
>nul 2>&1
if ERRORLEVEL 1 (
        echo "You need to edit %QTDIR%\mkspecs\common\msvc-desktop.conf to use
'QMAKE_LIBS_QT_ENTRY     = -lqtmain -lshell32'"
        goto end
)
if not exist "%MAYA_LOCATION%\include\Qt\qtnfc.disabled" (
        echo "You need to rename %MAYA_LOCATION%\include\Qt\qtnfc to
%MAYA_LOCATION\include\Qt\qtnfc.disabled"
        goto end
)

set SIPDIR=v:\sip-4.18.1
set PYQTDIR=v:\PyQt5_gpl-5.7

set MSVC_DIR=C:\Program Files (x86)\Microsoft Visual Studio 11.0
if ["%LIBPATH%"]==[""] call "%MSVC_DIR%\VC\vcvarsall" amd64

set INCLUDE=%INCLUDE%;%MAYA_LOCATION%\include\python2.7
set LIB=%LIB%;%MAYA_LOCATION%\lib

pushd %PYQTDIR%

set PATH=%QTDIR%\bin;%PATH%
"%MAYA_LOCATION%\bin\mayapy" configure.py --spec %QMAKESPEC%
LIBDIR_QT="%QTDIR%\lib" INCDIR_QT="%QTDIR%\include\Qt"
MOC="%QTDIR%\bin\moc.exe" --sip="%QTDIR%\Python\sip.exe" --sip-
incdir="%QTDIR%\Python\include" -w --no-designer-plugin
nmake
nmake install
popd

:end
```

You're done! go to the testing paragraph at the end of the article.

Testing

Copy and paste this example in the Maya Script Editor (in a Python tab), and execute the code:

```
import sys
from PyQt5.QtWidgets import (QWidget, QToolTip, QPushButton)
from PyQt5.QtGui import QFont


class Example(QWidget):

    def __init__(self):
        super(Example,self).__init__()

        self.initUI()


    def initUI(self):

        QToolTip.setFont(QFont('SansSerif', 10))
```

```
        btn = QPushButton('Button', self)
        btn.setToolTip('This is a <b>QPushButton</b> widget')
        btn.resize(btn.sizeHint())
        btn.move(50, 50)

        self.setGeometry(300, 300, 300, 200)
        self.setWindowTitle('Tooltips')
        self.show()


ex = Example()
```

If you see the dialog is showing, you are all set.