**Compiling PySide for IDA Pro 6.1 on Windows**

**25/12/1013**

**https://www.facebook.com/luzxs**

If you've ever wanted to use IDA's native GUIs, you'll need PySide. Unfortunately PySide is annoying to compile, and the only available binaries are for Python 2.6. This should work for any version.

Note: The python module installer executable at the end of this guide did not work for me. The compiled pyd and dll files, however were completely usable from within IDA.

**Prequisites**

* Python 2.7
  + <http://www.python.org/ftp/python/2.7/python-2.7.msi>
    - Run and install, I’ll be using C:\Python27
* Perl 5.18
  + <http://strawberry-perl.googlecode.com/files/strawberry-perl-5.18.1.1-32bit.msi>
* Visual Studio 2010
* Visual Studio 2010 Service Pack 1
* Windows SDK v7.0 is required when building against Python 2.6, 2.7 or 3.2
  + <http://www.microsoft.com/en-us/download/details.aspx?id=3138>
* Qt 4.8.5 Source Code (Qt libraries 4.8.5 for Windows (VS 2010, 235 MB))
  + <http://download.qt-project.org/official_releases/qt/4.8/4.8.5/qt-win-opensource-4.8.5-vs2010.exe>
    - Run and install this in your computer, I’ll be using C:\Qt\4.8.5
* setuptools python module
  + <https://pypi.python.org/pypi/setuptools>
    - Get it at [https://bitbucket.org/pypa/setuptools/get/default.tar.gz#egg=setuptools-dev](https://bitbucket.org/pypa/setuptools/get/default.tar.gz" \l "egg=setuptools-dev)
    - Extract and run **setup.py install**
* cmake
  + <http://www.cmake.org/cmake/resources/software.html>
    - Get it at <http://www.cmake.org/files/v2.8/cmake-2.8.12.1-win32-x86.exe>
    - Run and install, I used C:\Cmake 2.8
* git for windows
  + <http://git-scm.com/download/win>
* gnuwin32
  + <http://sourceforge.net/projects/getgnuwin32/files/>
    - Download and install it (will be usefull !)
* After setup these tool, you need to add some key in the **Environment Variables**
  + Right Click Computer > Properties > Advanced > Environment Variables…
    - INCLUDE=C:\Program Files\Microsoft Visual Studio 10.0\VC\include;C:\Program Files\Microsoft SDKs\Windows\v7.0A\Include;C:\Qt\4.8.5\include
    - LIB=C:\Python27\Lib;C:\Qt\4.8.5\lib
    - PYTHONHOME=C:\Python27
    - QMAKESPEC=win32-msvc2010
    - QTDIR=C:\Qt\4.8.5
    - QT\_NAMESPACE=QT
    - PATH=%PATH%;%QTDIR%\bin;C:\Python27;C:\Python27\Scripts;C:\Qt\4.8.5\qmake

**Compiling QT 4**

It is necessary to build the Qt environment, because IDA is shipped with a custom version of Qt which wraps its classes inside the QT namespace (we’ll see later why that is so).

Download the latest version of jom from <http://qt-project.org/wiki/jom> It will accelerate the build process by using all available cores at build time, more on this further down. Extract jom files to

**C:\Qt\jom**

Start **Visual Studio 2010 Command Prompt** as an administrator.

On Win7 this can be done by right-clicking **Start > Programs > Microsoft Visual Studio 2010 > Visual Studio Tools > Visual Studio Command Prompt (2010)** and choosing "**Run as Administrator**" from the popup menu.

On Win8 you can reach the shortcut by typing "Visual Studio" to get a list of app and shortcuts known to the system, it should list this particular shortcut to. Right-click it and choose to run as administrator from the bottom ribbon.

Run the following commands in it (every line is a different command. Type it, then press Enter):

**>> cd c:\Qt\4.8.5**

To compile both debug and release versions type or copy/paste:

**C:\Qt\4.8.5>configure -debug-and-release -platform win32-msvc2010 -no-qt3support -qtnamespace QT**

I chose to only compile the release version like so:

**C:\Qt\4.8.5>configure -release -opensource -platform win32-msvc2010 –no-qt3support –qtnamespace QT**

Note:

1. Don't get irritated by the -platform flag reading "win32-msvc2010". It simply means that the MS Visual Studio compiler will be used and does not specify the architecture (x64 or x32, which was in fact already chosen by starting the Visual Studio x64 command prompt above). In other words: "-platform" is always "win32-msvc2010" for us in the context of this example, even for win64 builds.
2. If you customize the configure line, make sure you leave in the -**qtnamespace QT**! Otherwise your binaries will be unusable from within IDA.

This will take a few minutes, depending on your hardware. When it's done it should report back by printing:

"*Qt is now configured for building. Just run nmake.To reconfigure, run nmake confclean and configure*."

**Build Qt using jom**

Jom is an nmake replacement for Qt compilation on multi-core CPUs. Its parameter -j N allows to setup the number of parallel processes for compilation. Larger is better, where N is the number of CPU cores you want to utilize for Qt compilation. The number of physical CPU cores is a good choice for N, e.g. 4 on a typical quad-core CPU.

Usually compilation takes about 10 hours on a single core, with jom and an SSD drive it took only a bit more than 1 hour on my machine (Windows 8, 64bits)

Still in the same command prompt, enter and execute the following:

**C:\Qt\4.8.5>..\jom\jom.exe -j 4**

You might want to use a different number in the end if you have a system with more or less physical CPU cores.

This stage took around 2 hours on my PC

Example build may be fail, but it’s OK

**Compiling PySide (We’ll use C:\ as Root Directory)**

* Get the packaging code
  + cd /d c:\
  + git clone git://gitorious.org/pyside/packaging.git
* Change directory to packaging/setuptools/
  + cd /d c:\packaging\setuptools
* Edit build.py to comment out the Tools repository

|  |
| --- |
| # Modules  modules = {  'dev': [  ["Apiextractor", "master", "https://github.com/PySide/Apiextractor.git"],  ["Generatorrunner", "master", "https://github.com/PySide/Generatorrunner.git"],  ["Shiboken", "master", "https://github.com/PySide/Shiboken.git"],  ["PySide", "master", "https://github.com/PySide/PySide.git"],  #["Tools", "master", "https://github.com/PySide/Tools.git"],  ],  'stable': [  ["Apiextractor", "0.10.8", "https://github.com/PySide/Apiextractor.git"],  ["Generatorrunner", "0.6.14", "https://github.com/PySide/Generatorrunner.git"],  ["Shiboken", "1.0.9", "https://github.com/PySide/Shiboken.git"],  ["PySide", "1.0.8", "https://github.com/PySide/PySide.git"],  #["Tools", "0.2.13", "https://github.com/PySide/Tools.git"],  ],  } |

* Open the Visual Studio 2010 Comand Prompt
* Add the directory of qmake to your path
  + **set path=%path%;C:\Qt\4.8.5\qmake**
* Change Directory to C:\packaging\setuptools\modules
  + **cd /d c:\packaging\setuptools\modules**
    - I use build.py –d to download all the repositories, but It get error, so I manual download package from it repo
    - In Command Prompts
      * **git clone https://github.com/PySide/Apiextractor.git && git clone https://github.com/PySide/Generatorrunner.git && git clone https://github.com/PySide/Shiboken.git && git clone https://github.com/PySide/PySide.git**
      * ! Notes : All command is in oneline and you must install gnuwin32 to this command to run, or you can type one command each line (ex: git clone <https://github.com/PySide/Apiextractor.git>)
* Change directory into each of the repositories and run the following commands to checkout repo:
  + Apiextractor
    - cd /d \packaging\setuptools\modules\Apiextractor
    - git checkout 0.10.8
  + Generatorrunner
    - cd /d \packaging\setuptools\modules\Generatorrunner
    - git checkout 0.6.14
  + Shiboken
    - cd /d \packaging\setuptools\modules\Shiboken
    - git checkout 1.0.9
  + PySide
    - cd /d \packaging\setuptools\modules\PySide
    - git checkout 1.0.8
* Download **this patch** and apply it using git
  + Copy **pyside.diff** to C:\packaging\setuptools\
    - **cd /d c:\packaging\setuptools**
    - **git apply --ignore-whitespace pyside.diff**

Notes : If this task fail, you can manual apply to source by open pyside.diff and read instruction

+++ : File need to edit

@@ 8, 9 … Line to edit

* delete line

+ add line

* Patch the calls to **deallocData**in **Siboken** so they pass in two arguments (second argument should be True)
  + c:\packaging\setuptools\modules\Shiboken\libshiboken\basewrapper.cpp

|  |
| --- |
| extern "C" {  void deallocData(SbkObject\* self, bool cleanup);  static void SbkObjectTypeDealloc(PyObject\* pyObj); |

|  |
| --- |
| void DtorCallerVisitor::done() {  Shiboken::Object::deallocData(m\_pyObj, true);  std::list<std::pair<void\*, SbkObjectType\*> >::const\_iterator it = m\_ptrs.begin(); |

c:\packaging\setuptools\modules\Shiboken\libshiboken\basewrapper\_p.h

|  |
| --- |
| \* Destroy internal data  \*\*/  void deallocData(SbkObject\* self);  void deallocData(SbkObject\* self, bool cleanup);  } // namespace Objectging |

c:\packaging\setuptools\modules\Shiboken\CmakeLists.txt

|  |
| --- |
| set(shiboken\_VERSION "${shiboken\_MAJOR\_VERSION}.${shiboken\_MINOR\_VERSION}.${shiboken\_MICRO\_VERSION}")  IF (NOT $ENV{QT\_NAMESPACE} STREQUAL "")  add\_definitions(-DQT\_NAMESPACE=$ENV{QT\_NAMESPACE})  ENDIF (NOT $ENV{QT\_NAMESPACE} STREQUAL "")  option(BUILD\_TESTS "Build tests." TRUE) |

C:\packaging\setuptools\modules\shiboken\generator\cppgenerator.cpp

|  |
| --- |
| Bug : error C2661: 'QFileWrapper::open' : no overloaded function takes 3 arguments <http://comments.gmane.org/gmane.comp.lib.qt.pyside/2995>  **With QT > 4.8.0** |

|  |
| --- |
| **Locate :**  --------  && (func->name() != "qt\_metacall"))  overloads.append(func);  ---------  **and replace with:**  ……………  && (func->name() != "qt\_metacall")) {  if (func->toString().contains("open")) {  if (func->isVirtual())  overloads.append(func);  }  else  overloads.append(func);  }  …………. |

* Open up another Visual Studio 2010 Command Prompt
* Add cmake/qmake/git/qt to your PATH and set the Qt namespace

|  |
| --- |
| * set path=%path%;C:\CMake 2.8\bin * set path=%path%;C:\Qt\4.8.5\qmake * set path=%path%;C:\Git\bin * set path=%path%;C:\Qt\4.8.5\bin * set QT\_NAMESPACE=QT |

* Start the build process
  + Build.py
  + This takes around an hour on my PC

Notes : If you get error while linking with generatorrunner.exe, please copy QtCore4.dll, QtGui4.dll, QtXml4.dll from C:\Qt\4.8.5\bin to C:\packaging\setuptools\install-py2.7-qt4.8.5\bin, this error cause by generatorrunner.exe can’t get it dependency library.

* Your new PySide installer should be available in packaging\setuptools\dist

**Another way: Build newest version of Pyside at Offical Source**

This way will introduce you to build Pyside from Orginal Source at Qt Homepage.

Repair

* Download PySide Source from this address.
  + <http://download.qt-project.org/official_releases/pyside/PySide-1.2.1.tar.gz>
  + After download, let’s extract it to your directory to Compile, in my example is C:\PySide-1.2.1
* We need manual Patch some part of source file (will be replace with **RED** color)
  + <DIR>PySide-1.2.1\sources\pyside\libpyside\pysidemetafunction\_p.h

|  |
| --- |
| #include <QByteArray>  class QObject;  namespace PySide { namespace MetaFunction { |

|  |
| --- |
| #include <QByteArray>  **namespace QT {**  class QObject;  }  namespace PySide { namespace MetaFunction { |

|  |
| --- |
| void init(PyObject\* module);  /\*\*  \* Does a Qt metacall on a QObject  \*/  bool call(QObject\* self, int methodIndex, PyObject\* args, PyObject\*\* retVal = 0);  } //namespace MetaFunction |

|  |
| --- |
| void init(PyObject\* module);  /\*\*  \* Does a Qt metacall on a QObject  \*/  bool call(**QT::QObject\*** self, int methodIndex, PyObject\* args, PyObject\*\* retVal = 0);  } //namespace MetaFunction |

* + <DIR>PySide-1.2.1\sources\pyside\libpyside\signalmanager.h

|  |
| --- |
| #include <QMetaType>  class QObject;  namespace PySide  { |

|  |
| --- |
| #include <QMetaType>  #include <QtCore/qnamespace.h>    **QT\_BEGIN\_NAMESPACE**  class QObject;  **QT\_END\_NAMESPACE**  namespace PySide  { |

* + PySide-1.2.1\sources\pyside\PySide\QtCore\typesystem\_core\_common.xml

|  |
| --- |
| <inject-code class="native" position="beginning">  extern bool  qRegisterResourceData(int,  const unsigned char \*,  const unsigned char \*,  const unsigned char \*);  extern bool  qUnregisterResourceData(int,  const unsigned char \*,  const unsigned char \*,  const unsigned char \*);  </inject-code> |

|  |
| --- |
| <inject-code class="native" position="beginning">  **QT\_BEGIN\_NAMESPACE**  extern bool  qRegisterResourceData(int,  const unsigned char \*,  const unsigned char \*,  const unsigned char \*);  extern bool  qUnregisterResourceData(int,  const unsigned char \*,  const unsigned char \*,  const unsigned char \*);  **QT\_END\_NAMESPACE**  </inject-code> |

* <DIR>PySide-1.2.1\sources\pyside\PySide\QtDeclarative\CMakeLists.txt

|  |
| --- |
| ${QT\_QTSQL\_LIBRARY}  ${QT\_QTXMLPATTERNS\_LIBRARY}  ${QT\_QTOPENGL\_LIBRARY}  ${QT\_QTDECLARATIVE\_LIBRARY}) |

|  |
| --- |
| ${QT\_QTSQL\_LIBRARY}  ${QT\_QTXMLPATTERNS\_LIBRARY}  **~~${QT\_QTOPENGL\_LIBRARY}~~**  ${QT\_QTDECLARATIVE\_LIBRARY}) |

* <DIR>PySide-1.2.1\sources\pyside\PySide\QtGui\typesystem\_gui\_common.xml

|  |
| --- |
| <extra-includes>  <include file-name="QIcon" location="global"/>  <include file-name="QMessageBox" location="global"/>  </extra-includes>  <inject-code class="native" file="glue/qwidget\_glue.cpp" position="beginning" /> |

|  |
| --- |
| <extra-includes>  <include file-name="QIcon" location="global"/>  <include file-name="QMessageBox" location="global"/>  </extra-includes>  **<add-function signature="FromCObject(PyObject\*)" return-type="PyObject\*" static="yes">**  **<inject-code class="target" position="beginning">**  **<![CDATA[ %PYARG\_0 = Shiboken::createWrapper<QWidget>((const QWidget \*)PyCObject\_AsVoidPtr(%1), false, false); ]]>**  **</inject-code>**  **</add-function>**  <inject-code class="native" file="glue/qwidget\_glue.cpp" position="beginning" /> |

* \PySide-1.2.1\sources\pyside\PySide\CMakeLists.txt

|  |
| --- |
| HAS\_QT\_MODULE(QT\_QTTEST\_FOUND QtTest)  HAS\_QT\_MODULE(QT\_QTOPENGL\_FOUND QtOpenGL)  HAS\_QT\_MODULE(QT\_QTSQL\_FOUND QtSql) |

|  |
| --- |
| HAS\_QT\_MODULE(QT\_QTTEST\_FOUND QtTest)  **#HAS\_QT\_MODULE(QT\_QTOPENGL\_FOUND QtOpenGL)**  HAS\_QT\_MODULE(QT\_QTSQL\_FOUND QtSql) |

|  |
| --- |
| HAS\_QT\_MODULE(QT\_QTMULTIMEDIA\_FOUND QtMultimedia)  HAS\_QT\_MODULE(QT\_PHONON\_FOUND phonon)  HAS\_QT\_MODULE(QT\_QTDECLARATIVE\_FOUND QtDeclarative) |

|  |
| --- |
| HAS\_QT\_MODULE(QT\_QTMULTIMEDIA\_FOUND QtMultimedia)  **#HAS\_QT\_MODULE(QT\_PHONON\_FOUND phonon)**  HAS\_QT\_MODULE(QT\_QTDECLARATIVE\_FOUND QtDeclarative) |

* PySide-1.2.1\sources\pyside\PySide\global.h.in

|  |
| --- |
| //QtHelp need be included after QtSql  #include "@QT\_QTHELP\_INCLUDE\_DIR@/QtHelp"  #ifndef QT\_NO\_OPENGL  #include <@GL\_H@>  #include <@QT\_QTOPENGL\_INCLUDE\_DIR@/QtOpenGL>  #endif // QT\_NO\_OPENGL |

|  |
| --- |
| //QtHelp need be included after QtSql  #include "@QT\_QTHELP\_INCLUDE\_DIR@/QtHelp"  **//#ifndef QT\_NO\_OPENGL**  **//#include <@GL\_H@>**  **//#include <@QT\_QTOPENGL\_INCLUDE\_DIR@/QtOpenGL>**  **//#endif // QT\_NO\_OPENGL** |

* PySide-1.2.1\sources\pyside\PySide\\_\_init\_\_.py.in

|  |
| --- |
| \_\_all\_\_ = ['QtCore', 'QtGui', 'QtNetwork', 'QtOpenGL', 'QtSql', 'QtSvg', 'QtTest', 'QtWebKit', 'QtScript']  \_\_version\_\_ = "@BINDING\_API\_VERSION\_FULL@" |

|  |
| --- |
| **\_\_all\_\_ = ['QtCore', 'QtGui', 'QtNetwork', 'QtSql', 'QtSvg', 'QtTest', 'QtWebKit', 'QtScript']**  \_\_version\_\_ = "@BINDING\_API\_VERSION\_FULL@" |

* PySide-1.2.1\sources\pyside\CMakeLists.txt

|  |
| --- |
| if(USE\_XVFB)  find\_program(XVFB\_RUN NAMES xvfb-run)  if (NOT ${XVFB\_RUN} MATCHES "XVFB\_RUN-NOTFOUND")  set(XVFB\_EXEC ${XVFB\_RUN} -a)  message(STATUS "Using xvfb-run to perform QtGui tests.")  endif()  endif()  option(BUILD\_TESTS "Build tests." TRUE) |

|  |
| --- |
| if(USE\_XVFB)  find\_program(XVFB\_RUN NAMES xvfb-run)  if (NOT ${XVFB\_RUN} MATCHES "XVFB\_RUN-NOTFOUND")  set(XVFB\_EXEC ${XVFB\_RUN} -a)  message(STATUS "Using xvfb-run to perform QtGui tests.")  endif()  endif()  **IF (NOT $ENV{QT\_NAMESPACE} STREQUAL "")**  **add\_definitions(-DQT\_NAMESPACE=$ENV{QT\_NAMESPACE})**  **ENDIF (NOT $ENV{QT\_NAMESPACE} STREQUAL "")**  option(BUILD\_TESTS "Build tests." TRUE) |

* + Delete pyside-tools folder
  + PySide-1.2.1\sources\shiboken\ApiExtractor\parser\rpp\preprocessor.h

|  |
| --- |
| #include <QtCore/qstringlist.h>  class QByteArray;  class PreprocessorPrivate; |

|  |
| --- |
| #include <QtCore/qstringlist.h>  **QT\_BEGIN\_NAMESPACE**  class QByteArray;  **QT\_END\_NAMESPACE**  class PreprocessorPrivate; |

* PySide-1.2.1\sources\shiboken\ApiExtractor\parser\ast.h

|  |
| --- |
| #ifndef AST\_H  #define AST\_H  #include "smallobject.h"  #include "list.h"  class QString;  #define DECLARE\_AST\_NODE(k) \ |

|  |
| --- |
| #ifndef AST\_H  #define AST\_H  #include <qglobal.h>  #include "smallobject.h"  #include "list.h"  **QT\_BEGIN\_NAMESPACE**  class QString;  **QT\_END\_NAMESPACE**  #define DECLARE\_AST\_NODE(k) \ |

* PySide-1.2.1\sources\shiboken\ApiExtractor\parser\compiler\_utils.h

|  |
| --- |
| #include "codemodel.h"  class QString;  class QStringList;  struct TypeSpecifierAST; |

|  |
| --- |
| #include "codemodel.h"  **QT\_BEGIN\_NAMESPACE**  class QString;  class QStringList;  **QT\_END\_NAMESPACE**  struct TypeSpecifierAST; |

* R:\PySide-1.2.1\sources\shiboken\ApiExtractor\apiextractor.h

|  |
| --- |
| class AbstractMetaBuilder;  class QIODevice;  class ApiExtractor  { |

|  |
| --- |
| class AbstractMetaBuilder;  **QT\_BEGIN\_NAMESPACE**  class QIODevice;  **QT\_END\_NAMESPACE**  class ApiExtractor  { |

* PySide-1.2.1\sources\shiboken\ApiExtractor\CMakeLists.txt

|  |
| --- |
| add\_definitions(-DRXX\_ALLOCATOR\_INIT\_0)  set(apiextractor\_SRC |

|  |
| --- |
| add\_definitions(-DRXX\_ALLOCATOR\_INIT\_0)  **add\_definitions(-DQT\_NAMESPACE=$ENV{QT\_NAMESPACE})**  set(apiextractor\_SRC |

* PySide-1.2.1\sources\shiboken\ApiExtractor\docparser.h

|  |
| --- |
| #include "abstractmetalang.h"  class QDomDocument;  class QDomNode;  class QXmlQuery;  class DocParser  { |

|  |
| --- |
| #include "abstractmetalang.h"  QT\_BEGIN\_NAMESPACE  class QDomDocument;  class QDomNode;  class QXmlQuery;  QT\_END\_NAMESPACE  class DocParser  { |

* R:\PySide-1.2.1\sources\shiboken\ApiExtractor\include.h

|  |
| --- |
| #include <QList>  class QTextStream;  class Include  { |

|  |
| --- |
| #include <QList>  QT\_BEGIN\_NAMESPACE  class QTextStream;  QT\_END\_NAMESPACE  class Include  { |

* R:\PySide-1.2.1\sources\shiboken\ApiExtractor\reporthandler.h

|  |
| --- |
| #ifndef REPORTHANDLER\_H  #define REPORTHANDLER\_H  class QString;  class ReportHandler  {  public:  enum DebugLevel { NoDebug, SparseDebug, MediumDebug, FullDebug };  static void setContext(const QString &context);  static DebugLevel debugLevel();  static void setDebugLevel(DebugLevel level);  static int warningCount();  static int suppressedCount();  static void warning(const QString &str);  template <typename T>  static void setProgressReference(T collection)  {  setProgressReference(collection.count());  }  static void setProgressReference(int max);  static void progress(const QString &str, ...);  static void debugSparse(const QString &str)  {  debug(SparseDebug, str);  }  static void debugMedium(const QString &str)  {  debug(MediumDebug, str);  }  static void debugFull(const QString &str)  {  debug(FullDebug, str);  }  static void debug(DebugLevel level, const QString &str);  static bool isSilent();  static void setSilent(bool silent);  static void flush();  };  #endif // REPORTHANDLER\_H |

|  |
| --- |
| #ifndef REPORTHANDLER\_H  #define REPORTHANDLER\_H  namespace QT {  class QString;  }  class ReportHandler  {  public:  enum DebugLevel { NoDebug, SparseDebug, MediumDebug, FullDebug };  static void setContext(const QT::QString &context);  static DebugLevel debugLevel();  static void setDebugLevel(DebugLevel level);  static int warningCount();  static int suppressedCount();  static void warning(const QT::QString &str);  template <typename T>  static void setProgressReference(T collection)  {  setProgressReference(collection.count());  }  static void setProgressReference(int max);  static void progress(const QT::QString &str, ...);  static void debugSparse(const QT::QString &str)  {  debug(SparseDebug, str);  }  static void debugMedium(const QT::QString &str)  {  debug(MediumDebug, str);  }  static void debugFull(const QT::QString &str)  {  debug(FullDebug, str);  }  static void debug(DebugLevel level, const QT::QString &str);  static bool isSilent();  static void setSilent(bool silent);  static void flush();  };  #endif // REPORTHANDLER\_H |

* R:\PySide-1.2.1\sources\shiboken\ApiExtractor\typesystem.h

|  |
| --- |
| class AbstractMetaType;  class QTextStream;  class EnumTypeEntry; |

|  |
| --- |
| class AbstractMetaType;  QT\_BEGIN\_NAMESPACE  class QTextStream;  QT\_END\_NAMESPACE  class EnumTypeEntry; |

* R:\PySide-1.2.1\sources\shiboken\generator\qtdoc\qtdocgenerator.h

|  |
| --- |
| #include "docparser.h"  class QtDocParser;  class AbstractMetaFunction;  class AbstractMetaClass;  class QXmlStreamReader;  class QtDocGenerator;  class QtXmlToSphinx  { |

|  |
| --- |
| #include "docparser.h"  QT\_BEGIN\_NAMESPACE  class QtDocParser;  class QXmlStreamReader;  class QtDocGenerator;  QT\_END\_NAMESPACE  class AbstractMetaFunction;  class AbstractMetaClass;  class QtXmlToSphinx  { |

* R:\PySide-1.2.1\sources\shiboken\generator\generator.h

|  |
| --- |
| #include <abstractmetalang.h>  class ApiExtractor;  class AbstractMetaBuilder;  class QFile;  QTextStream& formatCode(QTextStream &s, const QString& code, Indentor &indentor); |

|  |
| --- |
| #include <abstractmetalang.h>  #include <QtCore/qnamespace.h>  class ApiExtractor;  class AbstractMetaBuilder;  QT\_BEGIN\_NAMESPACE  class QFile;  QT\_END\_NAMESPACE  QTextStream& formatCode(QTextStream &s, const QString& code, Indentor &indentor); |

* R:\PySide-1.2.1\sources\shiboken\CMakeLists.txt

|  |
| --- |
| set(shiboken\_VERSION "${shiboken\_MAJOR\_VERSION}.${shiboken\_MINOR\_VERSION}.${shiboken\_MICRO\_VERSION}")  option(BUILD\_TESTS "Build tests." TRUE) |

|  |
| --- |
| set(shiboken\_VERSION "${shiboken\_MAJOR\_VERSION}.${shiboken\_MINOR\_VERSION}.${shiboken\_MICRO\_VERSION}")  IF (NOT $ENV{QT\_NAMESPACE} STREQUAL "")  add\_definitions(-DQT\_NAMESPACE=$ENV{QT\_NAMESPACE})  ENDIF (NOT $ENV{QT\_NAMESPACE} STREQUAL "")  option(BUILD\_TESTS "Build tests." TRUE) |

* R:\PySide-1.2.1\MANIFEST.in

|  |
| --- |
| #  # MANIFEST.in  #  # Manifest template for creating the PySide source distribution.  include MANIFEST.in  include CHANGES.rst  include README.rst  include ez\_setup.py  include setup.py  include pyside\_postinstall.py  include popenasync.py  include qtinfo.py  include utils.py  # sources  recursive-include sources/patchelf \*\*  recursive-include sources/shiboken \*\*  recursive-include sources/pyside \*\*  recursive-include sources/pyside-tools \*\*  recursive-include sources/pyside-examples \*\*  recursive-exclude sources/pyside-examples/examples/hyperui \*\*  recursive-exclude sources/pyside-examples/mobility \*\*  # ignore .git  recursive-exclude sources/shiboken/.git \*\*  recursive-exclude sources/pyside/.git \*\*  recursive-exclude sources/pyside-tools/.git \*\*  recursive-exclude sources/pyside-examples/.git \*\*  # PySide package  recursive-include pyside\_package/PySide \*\*  recursive-include pyside\_package/PySide/docs \*\*  recursive-include pyside\_package/PySide/plugins \*\*  recursive-include pyside\_package/PySide/imports \*\*  recursive-include pyside\_package/PySide/translations \*\*  recursive-include pyside\_package/PySide/include \*\*  recursive-include pyside\_package/PySide/typesystems \*\*  recursive-include pyside\_package/PySide/examples \*\*  # pysideuic package  recursive-include pyside\_package/pysideuic \*\*  recursive-include pyside\_package/pysideuic/Compiler \*\*  recursive-include pyside\_package/pysideuic/port\_v2 \*\*  recursive-include pyside\_package/pysideuic/port\_v3 \*\*  recursive-include pyside\_package/pysideuic/widget-plugins \*\* |

|  |
| --- |
| #  # MANIFEST.in  #  # Manifest template for creating the PySide source distribution.  include MANIFEST.in  include CHANGES.rst  include README.rst  include ez\_setup.py  include setup.py  include pyside\_postinstall.py  include popenasync.py  include qtinfo.py  include utils.py  # sources  recursive-include sources/patchelf \*\*  recursive-include sources/shiboken \*\*  recursive-include sources/pyside \*\*  #recursive-include sources/pyside-tools \*\*  recursive-include sources/pyside-examples \*\*  recursive-exclude sources/pyside-examples/examples/hyperui \*\*  recursive-exclude sources/pyside-examples/mobility \*\*  # ignore .git  recursive-exclude sources/shiboken/.git \*\*  recursive-exclude sources/pyside/.git \*\*  #recursive-exclude sources/pyside-tools/.git \*\*  recursive-exclude sources/pyside-examples/.git \*\*  # PySide package  recursive-include pyside\_package/PySide \*\*  recursive-include pyside\_package/PySide/docs \*\*  recursive-include pyside\_package/PySide/plugins \*\*  recursive-include pyside\_package/PySide/imports \*\*  recursive-include pyside\_package/PySide/translations \*\*  recursive-include pyside\_package/PySide/include \*\*  recursive-include pyside\_package/PySide/typesystems \*\*  recursive-include pyside\_package/PySide/examples \*\*  # pysideuic package  #recursive-include pyside\_package/pysideuic \*\*  #recursive-include pyside\_package/pysideuic/Compiler \*\*  #recursive-include pyside\_package/pysideuic/port\_v2 \*\*  #recursive-include pyside\_package/pysideuic/port\_v3 \*\*  #recursive-include pyside\_package/pysideuic/widget-plugins \*\* |

* R:\PySide-1.2.1\setup.py

|  |
| --- |
| submodules = {  '1.2.1': [  ["shiboken", "1.2.1"],  ["pyside", "1.2.1"],  ["pyside-tools", "0.2.15"],  ["pyside-examples", "master"],  ],  '1.2.0': [  ["shiboken", "1.2.0"],  ["pyside", "1.2.0"],  ["pyside-tools", "0.2.14"],  ["pyside-examples", "master"],  ],  '1.1.2': [  ["shiboken", "1.1.2"],  ["pyside", "1.1.2"],  ["pyside-tools", "0.2.14"],  ["pyside-examples", "master"],  ],  '1.1.1': [  ["shiboken", "1.1.1"],  ["pyside", "1.1.1"],  ["pyside-tools", "0.2.14"],  ["pyside-examples", "master"],  ],  } |

|  |
| --- |
| submodules = {  '1.2.1': [  ["shiboken", "1.2.1"],  ["pyside", "1.2.1"],  #["pyside-tools", "0.2.15"],  ["pyside-examples", "master"],  ],  '1.2.0': [  ["shiboken", "1.2.0"],  ["pyside", "1.2.0"],  #["pyside-tools", "0.2.14"],  ["pyside-examples", "master"],  ],  '1.1.2': [  ["shiboken", "1.1.2"],  ["pyside", "1.1.2"],  #["pyside-tools", "0.2.14"],  ["pyside-examples", "master"],  ],  '1.1.1': [  ["shiboken", "1.1.1"],  ["pyside", "1.1.1"],  #["pyside-tools", "0.2.14"],  ["pyside-examples", "master"],  ],  } |

|  |
| --- |
| if not OPTION\_ONLYPACKAGE:  # Build extensions  for ext in ['shiboken', 'pyside', 'pyside-tools']:  self.build\_extension(ext) |

|  |
| --- |
| if not OPTION\_ONLYPACKAGE:  # Build extensions  #for ext in ['shiboken', 'pyside', 'pyside-tools']:  for ext in ['shiboken', 'pyside']:  self.build\_extension(ext) |

* Tạo tập tin cài đặt PySide
  + Mở Visual Studio Command Prompt (hoặc gõ vào vcvars32)
  + cd PySide-1.2.1
  + c:\Python27\python setup.py bdist\_wininst --qmake=c:\Qt\4.8.5\bin\qmake.exe

Chú ý : Trong quá trình Build, có thể các bạn sẽ gặp phải lỗi 139 tại tập tin generatorrunner.exe làm cho Cmake không thể tiếp tục được.

Lỗi này là do Generatorrunner.exe không tìm thấy DLL dependency của nó, bạn có thể copy QtCore4.dll, QtXml4.dll, QtGui4.dll từ thư mục Qt mà bạn đã Compile (ex : C:\Qt\4.8.5\bin) bỏ vào C:\Windows\System32, và Build lại bình thường.

* Sau khi Build thành công, tập tin cài đặt sẽ nằm trong thư mục
  + C:\PySide-1.2.1\dist

**Credits**

[1] Aaron Portnoy for outlining this process for python 2.6, 2012

<http://dvlabs.tippingpoint.com/blog/2012/02/25/mindshare-yo-dawg-i-heard-you-like-reversing>

[2] Daniel Pistelli for his guide on compiling Qt correctly with Visual Studio 2008

<http://www.hexblog.com/?p=250>, 2011

[3] Stepan Kubicek for his awesome tuts to compile QT, 2013

<http://plutometro.blogspot.com/2013/06/compiling-x64-pyside-using-vs2010.html>

[4] Compiling PySide for IDA Pro on Windows, 2013

<https://gist.github.com/ancat/8078106>

[5] [https://groups.google.com/forum/#!forum/pyside](https://groups.google.com/forum/" \l "!forum/pyside)

[6] [http://qt-project.org/wiki/PySideDownloads#a98df7359ac52361af81fa897d556a6a](http://qt-project.org/wiki/PySideDownloads" \l "a98df7359ac52361af81fa897d556a6a)

[7] <http://qt-project.org/wiki/Building_PySide_on_Windows>