



# Qt Framework and tools

Developer Guides  
Based on windows (c/c++)

Author : Sahakorn Buangam

Email : [sahakorn.new@gmail.com](mailto:sahakorn.new@gmail.com)

# คำนำ

สำหรับเอกสารนี้จะเป็นการแนะนำเบื้องต้นสำหรับผู้ที่จะพัฒนาซอฟต์แวร์ บน Qt- Framework ซึ่งผู้เขียนจะแนะนำการใช้งานการติดตั้งและแนะนำแนวทางสำหรับการพัฒนาต่อ ซึ่งหากผู้อ่านทุกท่านเข้าใจเนื้อหาในส่วนนี้ จะสามารถนำไปประยุกต์ ต่อยอดการพัฒนาได้ ด้วยตอนนี้สำหรับ Qt- framework ของไทย ยังไม่ค่อยมีใครเผยแพร่ และมีผู้พัฒนายังไม่มาก (แต่มีแนวโน้มจะพัฒนาไกล) แต่ผู้เขียนเห็นว่า Qt นั้นมีความสามารถมาก จึงอยากที่จะทำเป็นเอกสารให้ผู้อ่านที่สนใจเบื้องต้นได้นำไปใช้เริ่มต้นอย่างไม่ต้องเสียเวลา ซึ่งหากผู้อ่านสนใจเนื้อหารายละเอียด แนะนำให้เข้าไปศึกษาได้จาก Qt Documentation : <http://doc.qt.io/> ซึ่งจะมีเนื้อหาที่ละเอียดและครบถ้วน (เป็นภาษาอังกฤษ) สำหรับเอกสารนี้มีได้แสวงหาผลกำไร หรือนำมาหารายได้ แต่มีความมุ่งหวังจะนำเสนอให้ นิสิต นักศึกษา หรือผู้สนใจ ได้เข้าใจหลักการและวิธีการจะเริ่มต้นพัฒนา ซึ่งหากพบว่าเอกสารนี้มีการอ้างอิงถึงบุคคลใด หรือ อ้างอิงจาก Material ใดแล้วทำให้เกิดความผิดพลาดหรือความเข้าใจไม่ตรงกัน ผู้เขียนต้องขออภัยมา ณ ที่นี้

# รู้จักกับ Qt

Qt นั้นเป็นเครื่องมือในการพัฒนาซอฟต์แวร์ หรือ แอปพลิเคชันตัวหนึ่ง ซึ่งข้อดีคือสามารถทำงานแบบข้ามแพลตฟอร์ม {Android, IOS, Windows, Linux/X11, OS X , Windows Runtime , WinCE} และข้อดีอีกหนึ่งอย่างที่น่าสนใจคือการที่ Qt สามารถพัฒนาลงบน คอมพิวเตอร์ขนาดเล็ก หรือ Embedded Devices อย่างพวก Raspberry pi ฯลฯ และหากท่านใดที่พอจะเข้าใจภาษาอังกฤษ Documentation ของ Qt io เองก็มีเนื้อหาที่ครบถ้วน และยังมี Community ที่เยอะ และด้วยความที่ก่อนหน้านี้ Qt เริ่มพัฒนาออกมาเขาทำมาเพื่อ กราฟฟิคเฟรมเวิร์ก ดังนั้น การทำงานทางด้าน กราฟฟิคนั้นยอดเยี่ยมมาก ซึ่งหากเราจะพัฒนาซอฟต์แวร์บน Qt แล้วสามารถทำได้ในหลายๆ ระบบปฏิบัติการ แต่ในเอกสารนี้ ผู้เขียนจะเน้นไปที่ระบบปฏิบัติการ windows เป็นหลัก

# Installation

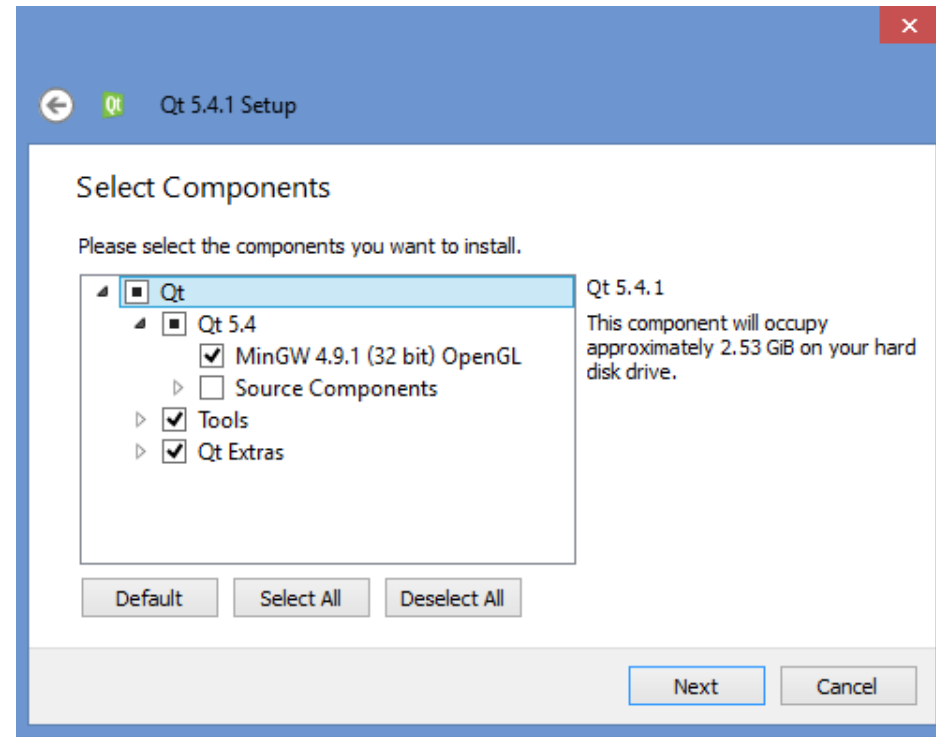
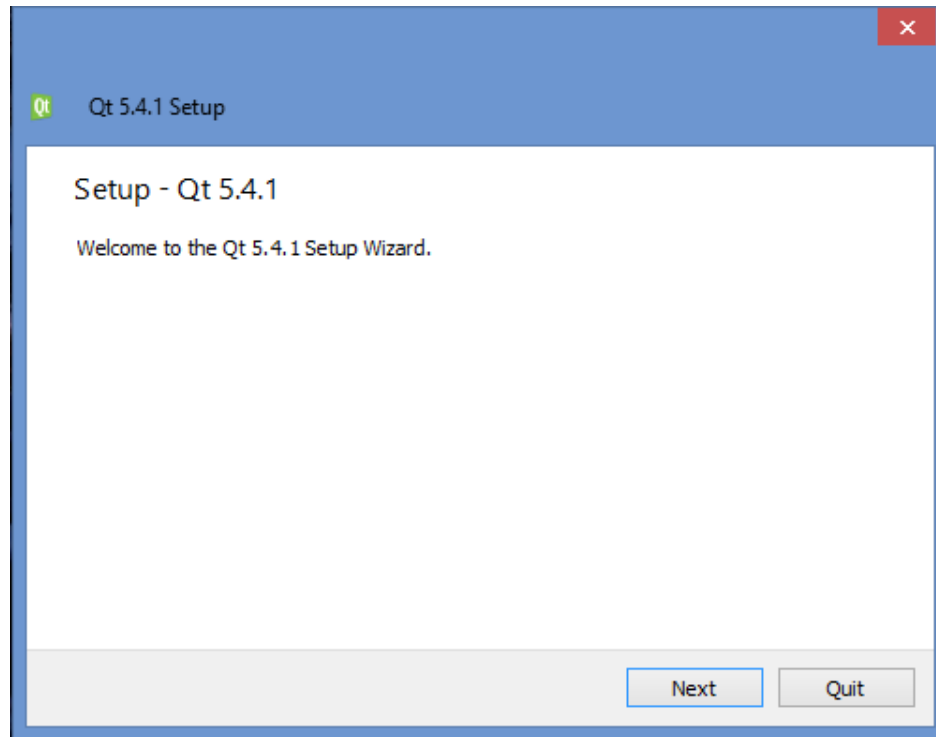
สำหรับการติดตั้งเครื่องมือสำหรับพัฒนาสำหรับบน windows ท่านอาจจะต้องไป ดาวน์โหลดผ่านเว็บของ Qt <http://www.qt.io/download/> ซึ่งหากเป็นระบบปฏิบัติการ Linux อย่าง Ubuntu จะสามารถ ลงผ่าน apt-get ได้ ซึ่งจะสะดวกกว่าหากผู้ที่ชำนาญ

ซึ่งหากใครไม่สามารถดาวน์โหลดได้ ให้เข้าไปที่ (เป็นเวอร์ชัน 5.4.1)

<http://ftp.jaist.ac.jp/pub/qtproject/archive/qt/5.4/5.4.1/>

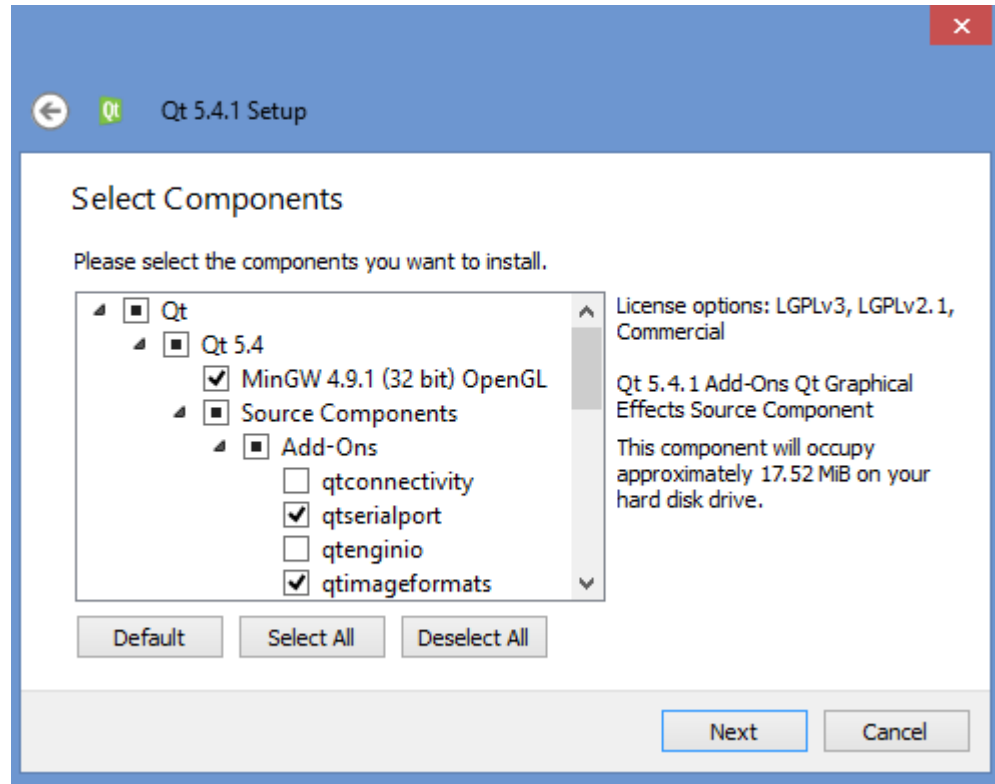
เลือกดาวน์โหลด qt-opensource-windows-x86-mingw491\_opengl-5.4.1.exe

# Installation (อ่านขั้นตอนดีๆ)

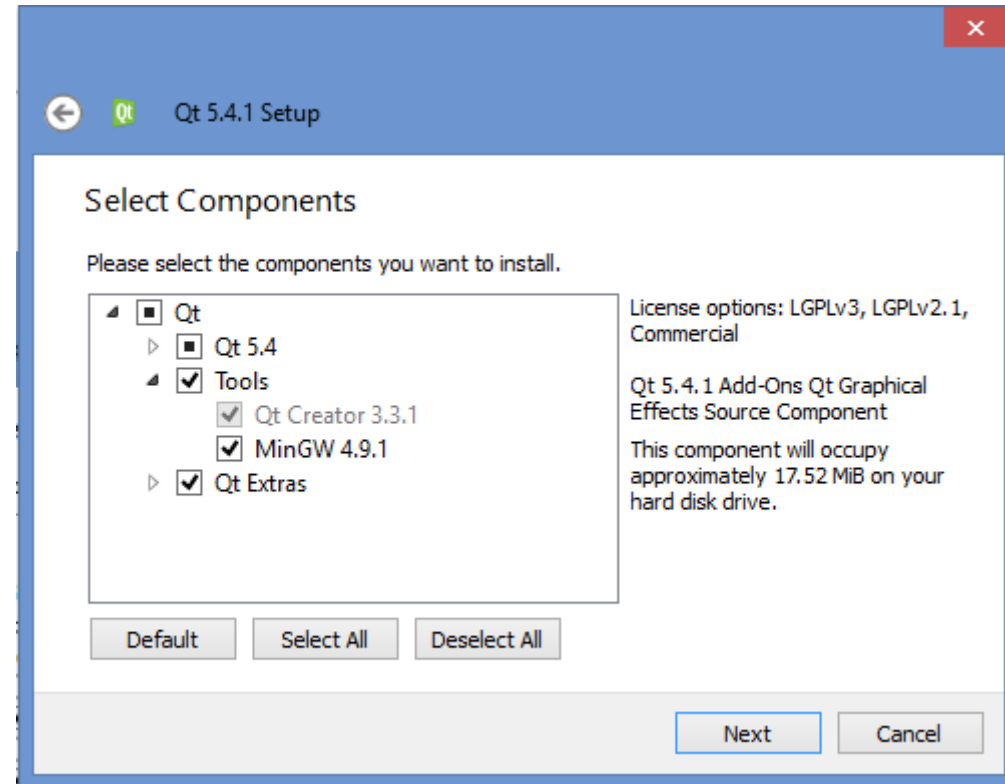


ให้ทำการเลือก MinGW 4.9.1 ด้วยที่แถบ Qt 5.4

# Installation



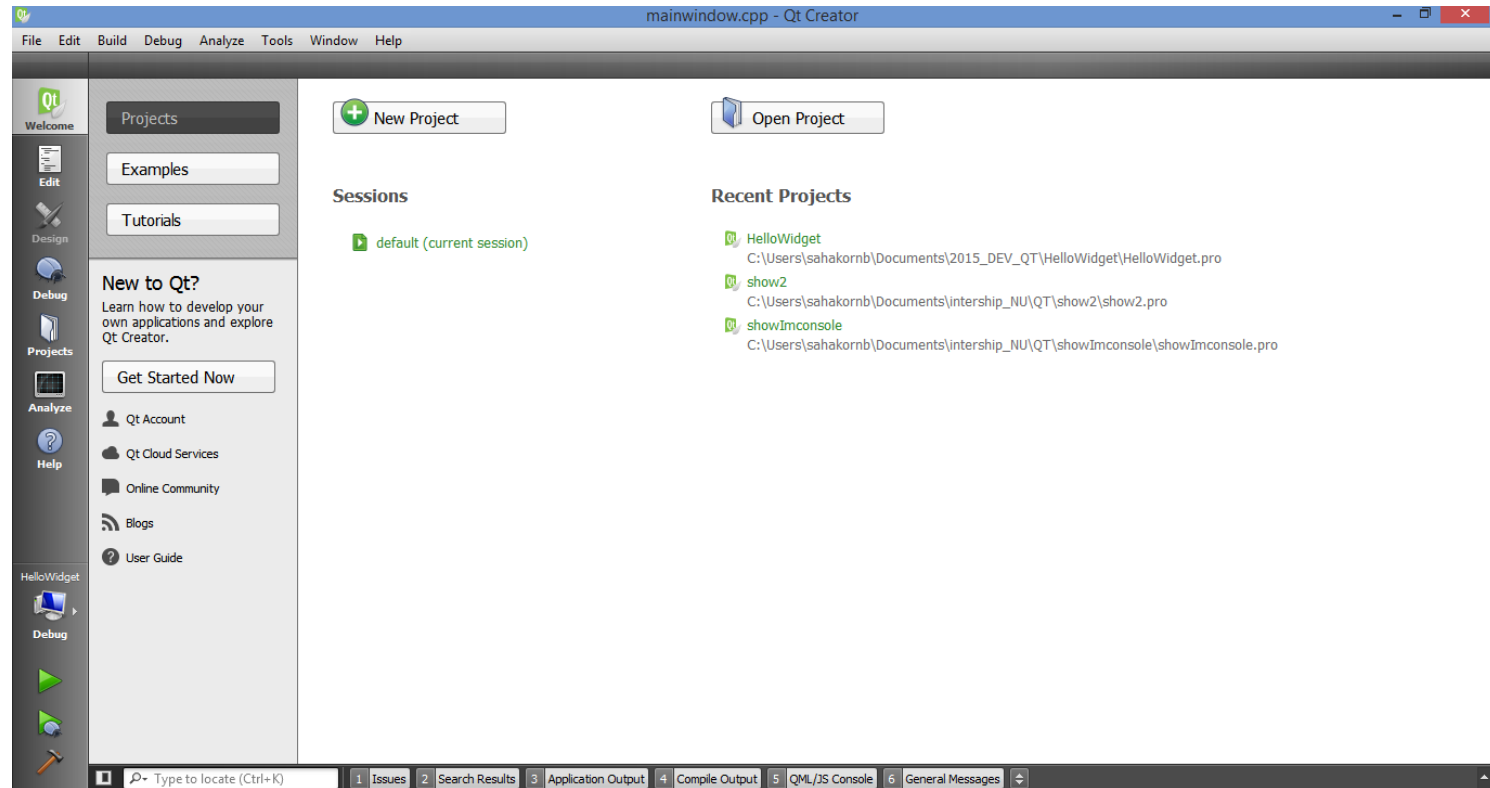
แถบ Source Components หากท่านต้องการ Add-Ons  
เพิ่มก็เลือกที่ท่านต้องการ แต่หากไม่เลือกไม่เป็นไร ติดตั้งที่หลังได้



แถบ Tools ถ้าไม่มี MinGW 4.9.1 ก็ให้เลือกด้วย (แนะนำให้เลือก)  
ส่วน Qt Extras ถ้าใช้ก็เลือก

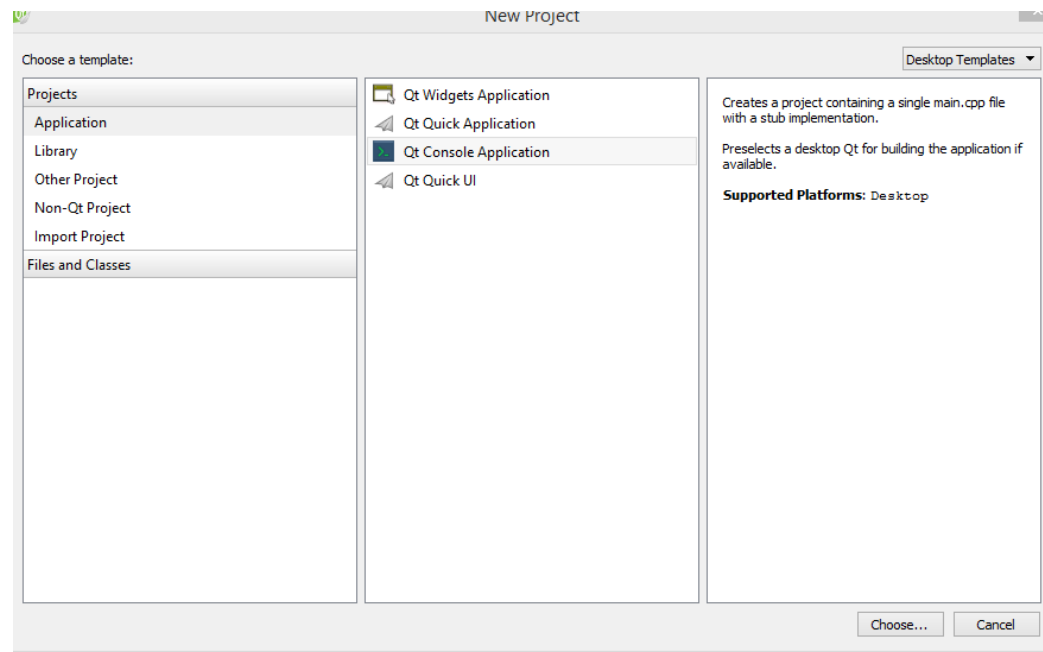
# Installation

เมื่อท่านติดตั้งเสร็จแล้วจะมีหน้าต่าง Qt Creator ขึ้นมานี่คือหน้าต่างที่เราจะพัฒนาโปรแกรมกัน



# แนะนำเบื้องต้น

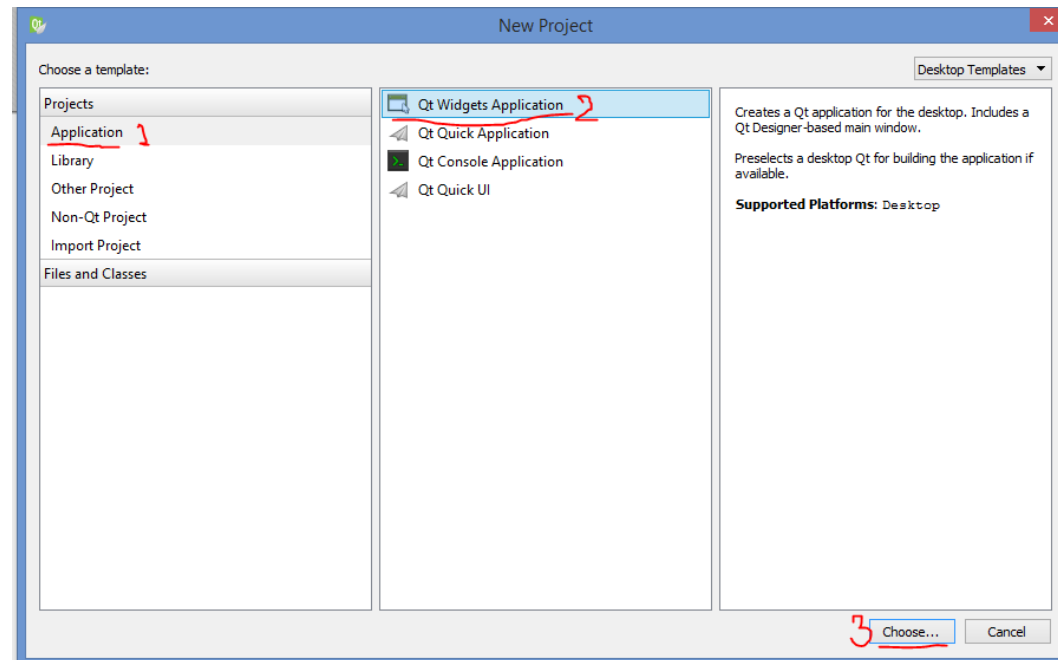
สำหรับ Qt Creator นั้นจะสามารถพัฒนาโปรแกรมได้หลายแบบซึ่งสามารถทำได้ทั้ง Application Library หรือว่าจะเป็น Non-Qt Project (สำหรับเป็น Compiler พวก c/c++) ซึ่งโดยทั่วไปผมจะพัฒนา Application เป็น 2 แบบ คือ Qt Widgets Application และ Qt Quick Application (QML) แต่ยังมี Qt Quick UI หรือจะเป็น Console Application ด้วย ซึ่งหากผู้อ่านเข้าใจหลักการ คิดว่าประยุกต์การพัฒนาไม่ยากซึ่งสำหรับภาษาที่ใช้พัฒนาจะเป็น c/c++





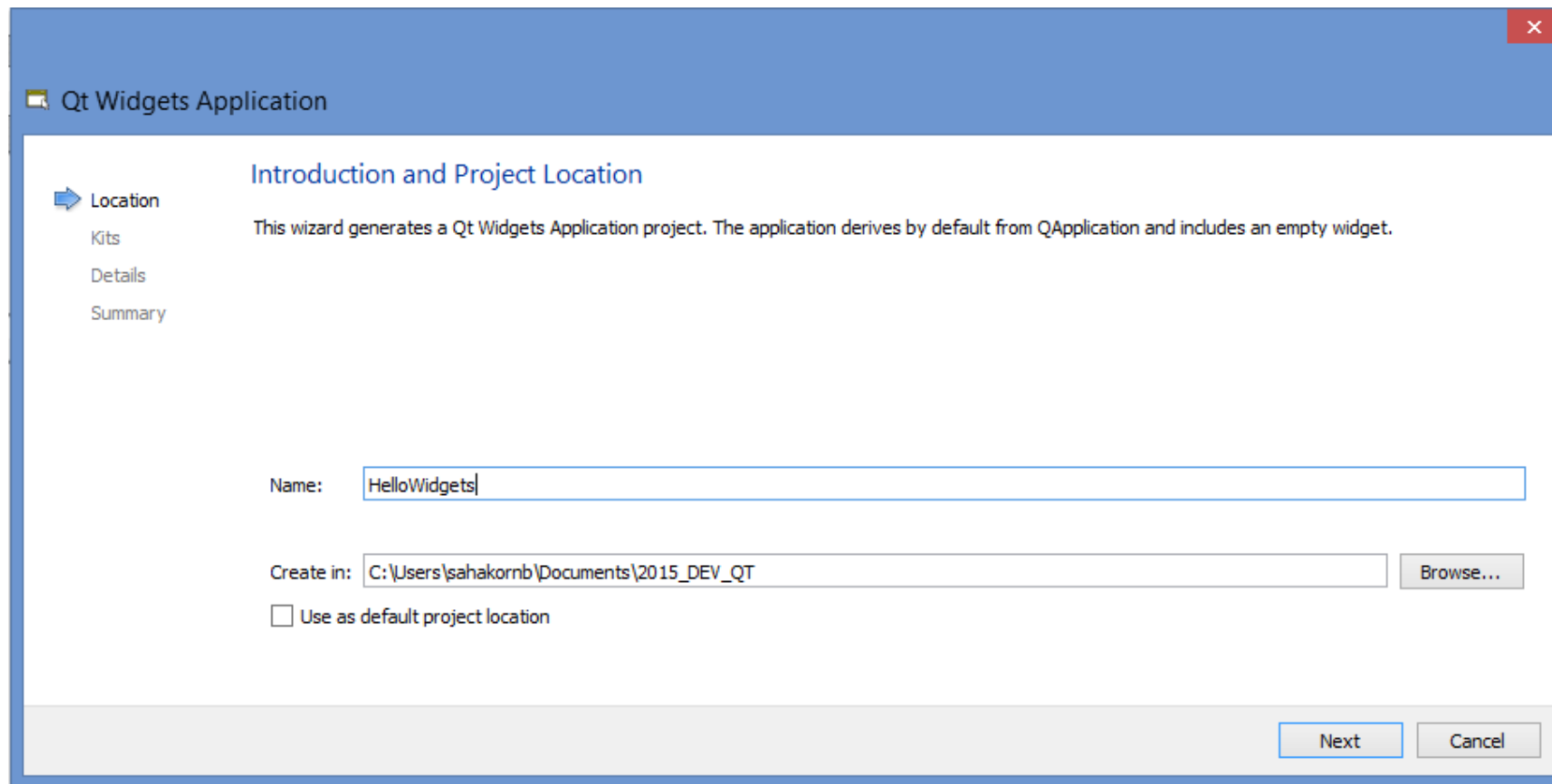
# เริ่มสร้างโปรเจค Qt Widgets Application

หากเข้ามาหน้าแรกจะเห็นปุ่ม New Project ให้คลิกไปที่ปุ่ม หรือไปที่ File → New File of Project  
ที่แถบเลือก Application → Qt Widgets Application → Choose...



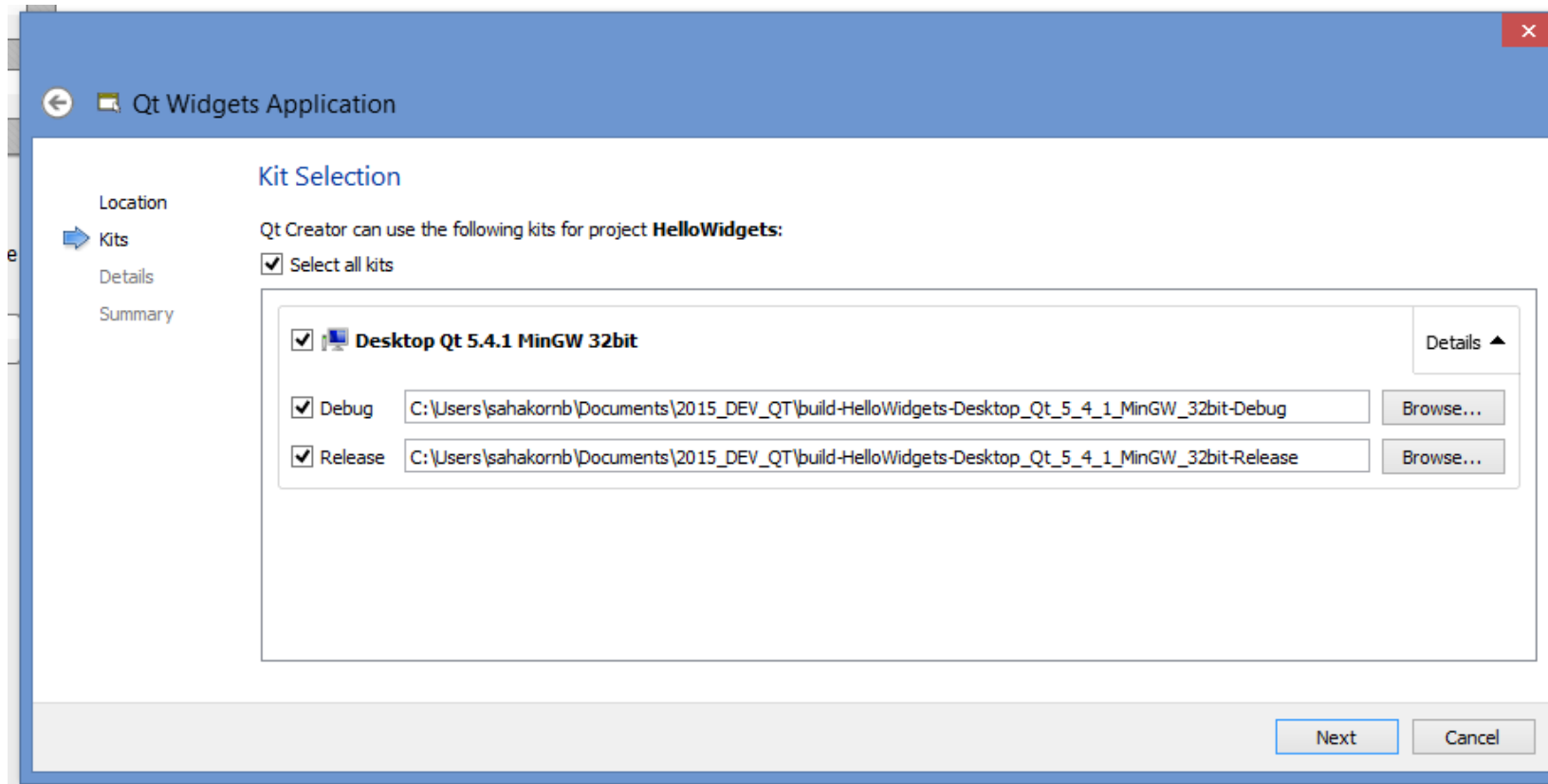
# เริ่มสร้างโปรเจค Qt Widgets Application

เลือก Location สำหรับโปรเจค



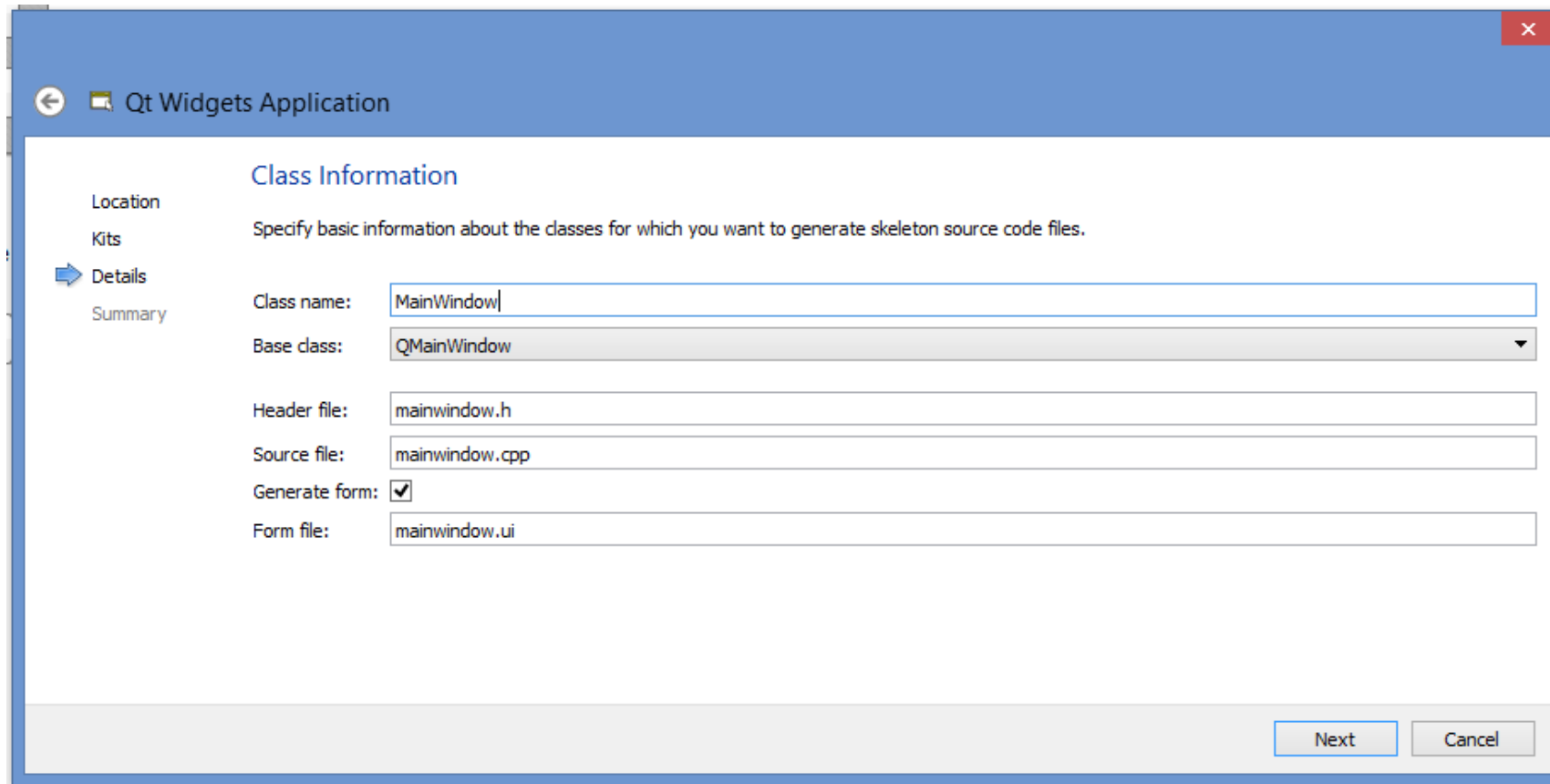
# เริ่มสร้างโปรเจค Qt Widgets Application

โดยหากท่านติดตั้ง MinGW แล้วส่วนของ kits และ Compiler จะได้ดังนี้ (โดยปกติโปรแกรมจะเลือกให้อัตโนมัติ)



# เริ่มสร้างโปรเจค Qt Widgets Application

ตั้งชื่อ Class และ เลือก Base class เป็น QMainWindow (หากไม่ตั้งไม่เป็นไรจะเลือกให้อัตโนมัติ)



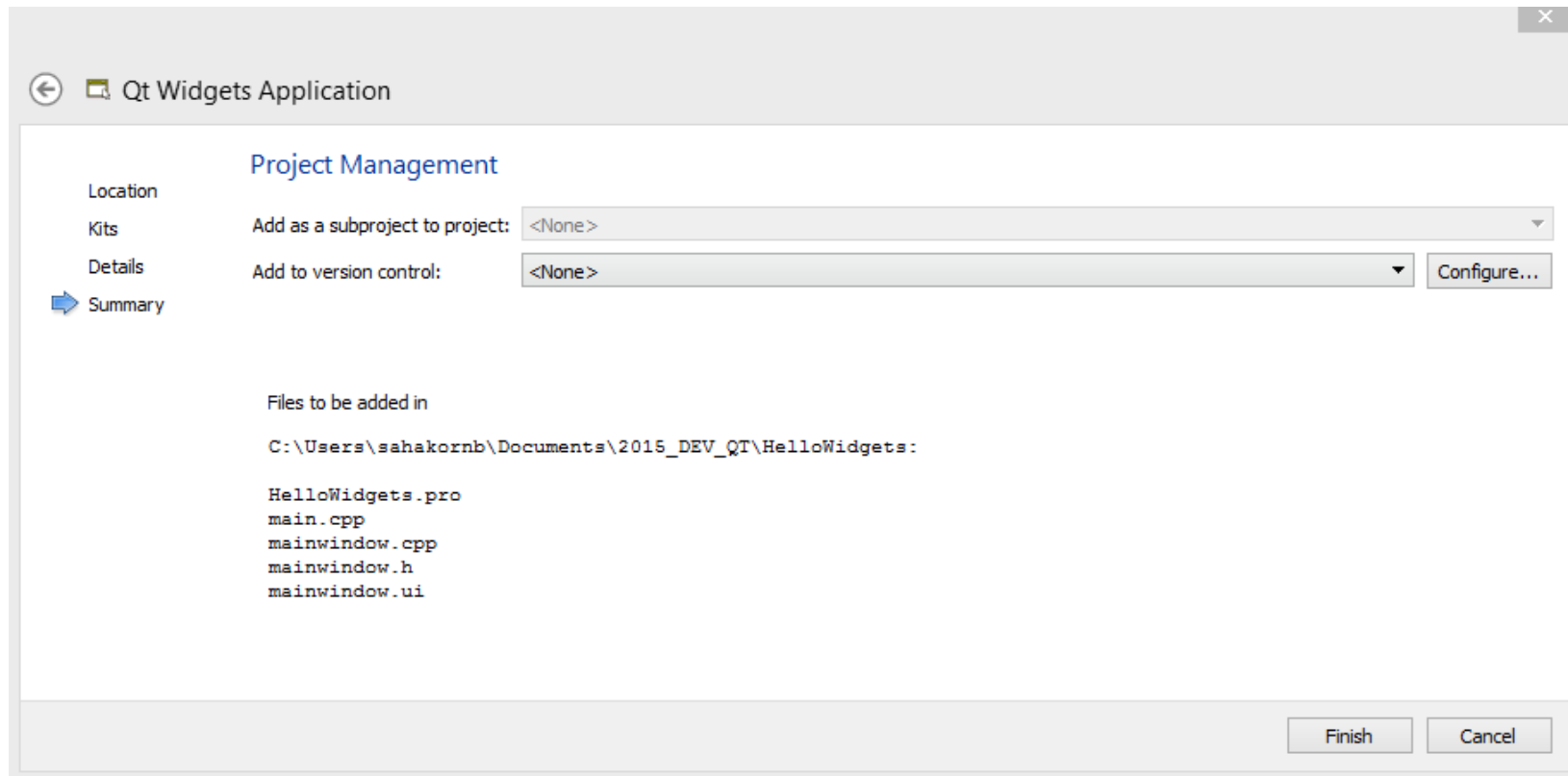
The screenshot shows the 'Qt Widgets Application' wizard window. On the left, there is a sidebar with four tabs: 'Location', 'Kits', 'Details' (which is selected and highlighted with a blue arrow), and 'Summary'. The main area is titled 'Class Information' and contains the instruction: 'Specify basic information about the classes for which you want to generate skeleton source code files.' Below this instruction, there are several input fields and a dropdown menu:

- 'Class name:' with a text box containing 'MainWindow'
- 'Base class:' with a dropdown menu showing 'QMainWindow'
- 'Header file:' with a text box containing 'mainwindow.h'
- 'Source file:' with a text box containing 'mainwindow.cpp'
- 'Generate form:' with a checked checkbox
- 'Form file:' with a text box containing 'mainwindow.ui'

At the bottom right of the window, there are two buttons: 'Next' and 'Cancel'.

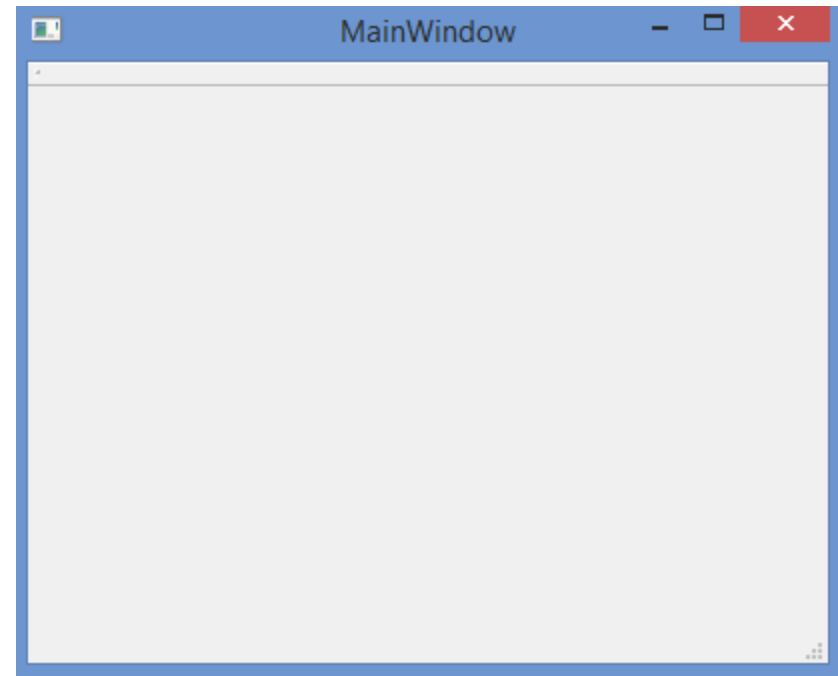
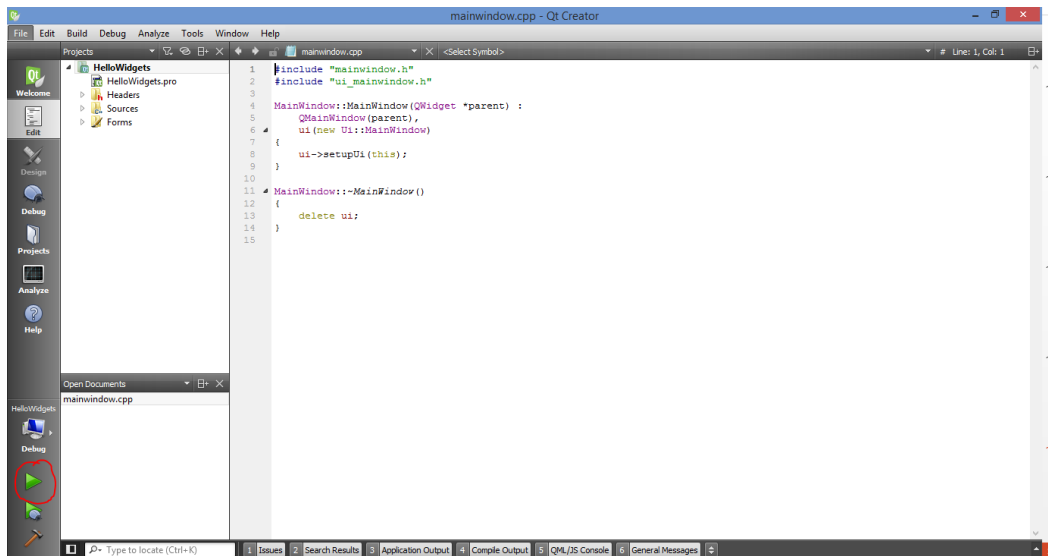
# เริ่มสร้างโปรเจค Qt Widgets Application

ถ้าไม่ใช้ Version Control พวก git ก็เลือกเป็น None ( Version Control เดี๋ยวจะพูดถึงหลังๆ)



# เริ่มสร้างโปรเจค Qt Widgets Application

ทดสอบ Build & run → ปุ่มเขียว ถ้าไม่ผิดพลาดจะมีหน้าต่างเด้งขึ้นมาแสดงว่าติดตั้งสมบูรณ์



หากใครพบปัญหาการ build ให้กลับไปดูการติดตั้งใหม่

# โครงสร้างเบื้องต้น



The image shows the Qt Creator interface for a project named 'HelloWidgets'. On the left, the 'Project Explorer' pane displays the project structure:

- HelloWidgets** (Project)
  - HelloWidgets.pro (Project File)
  - Headers** (Folder)
    - mainwindow.h (Header File)
  - Sources** (Folder)
    - main.cpp (Source File)
    - mainwindow.cpp (Source File)
  - Forms** (Folder)
    - mainwindow.ui (Form File)

Labels with red boxes highlight specific files:

- Header file** points to `mainwindow.h`.
- ไฟล์โค้ด** (Code File) points to `main.cpp` and `mainwindow.cpp`.
- ส่วน UI** (UI Part) points to `mainwindow.ui`.

On the right, the 'Editor' pane shows the source code for `mainwindow.cpp`, which is also highlighted by a red box:

```
1 #include "mainwindow.h"
2 #include "ui_mainwindow.h"
3
4 MainWindow::MainWindow(QWidget *parent) :
5     QMainWindow(parent),
6     ui(new Ui::MainWindow)
7 {
8     ui->setupUi(this);
9 }
10
11 MainWindow::~MainWindow()
12 {
13     delete ui;
14 }
15
```

Label **ส่วนโค้ด** (Code Part) points to the code editor area.

# ส่วนของ UI

คลิกไปที่ mainwindows.ui

หน้าต่าง เครื่องมือ

หน้าต่าง UI

หน้าต่าง property

Object	Class
MainWindow	QMainWindow
centralWidget	QWidget
menuBar	QMenuBar
mainToolBar	QToolBar
statusBar	QStatusBar

Property	Value
QObject	
objectName	MainWindow
QWidget	
windowModality	NonModal
enabled	<input checked="" type="checkbox"/>
geometry	[(0, 0), 400 x 300]
sizePolicy	[Preferred, Pref..]
minimumSize	0 x 0
maximumSize	16777215 x 1677.
sizeIncrement	0 x 0
baseSize	0 x 0
palette	Inherited
font	A [MS Shell D.



# ส่วนของ UI

ในหน้า mainwindows.ui จะพบว่าลักษณะจะคล้ายกับเครื่องมือของทาง .NET (Visual studio) ซึ่งหากใครประสบการณ์กับการพัฒนามบน Visual Studio มากก็จะใช้งานได้อย่างรวดเร็ว

ส่วนหลักๆที่เราจะเป็นต้องเรียนรู้คือ

- Property

ส่วนนี้จะเป็นส่วนที่ปรับคุณลักษณะของเครื่องมือที่เราใช้ไม่ว่าจะเป็น ขนาดสี รูปร่าง ฯลฯ ซึ่งหากดูดีๆ จะมีส่วนที่เป็น style ซึ่งหากใครเคยเขียนเว็บ ไฟล์ css จะสามารถใช้คำสั่งเดียวกันในการปรับได้

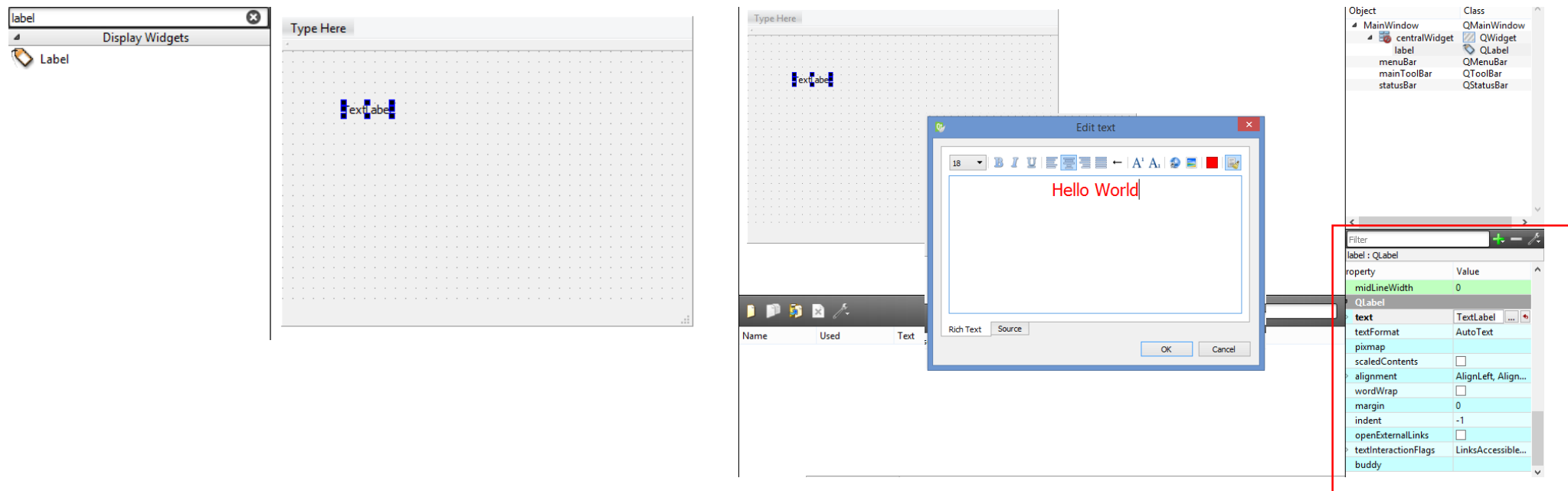
- เครื่องมือต่างๆ

สำหรับเครื่องมือต่างๆ สามารถดูได้จาก แถบด้านซ้าย และลากวางได้ เลย โดยเราสามารถปรับคุณลักษณะได้ จาก Property

# Hello World

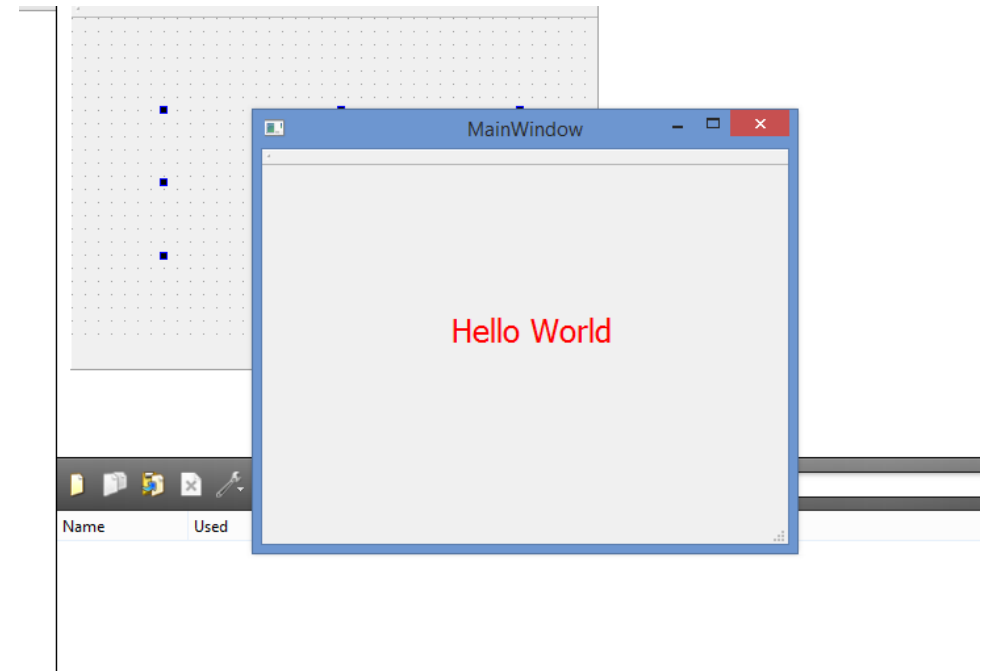
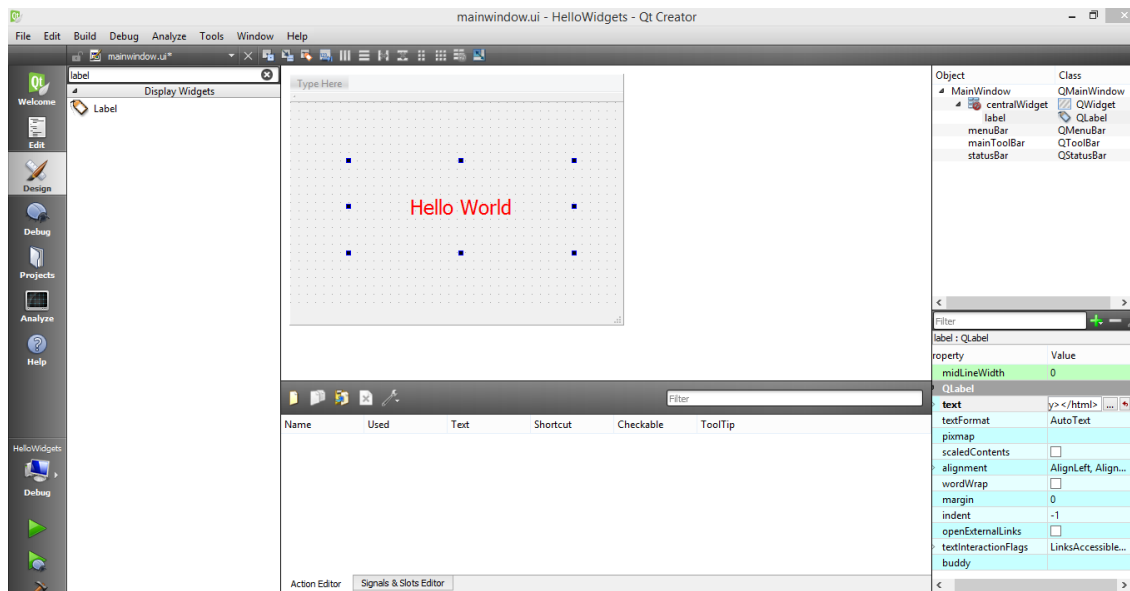
ให้ทำการเลือกเครื่องมือ ชื่อ Label ลาดวางบน panel และปรับขนาดให้เหมาะสม

ปรับขนาด label → Property → เลื่อนลงมาจนเจอ text คลิก ... และปรับขนาดและสี



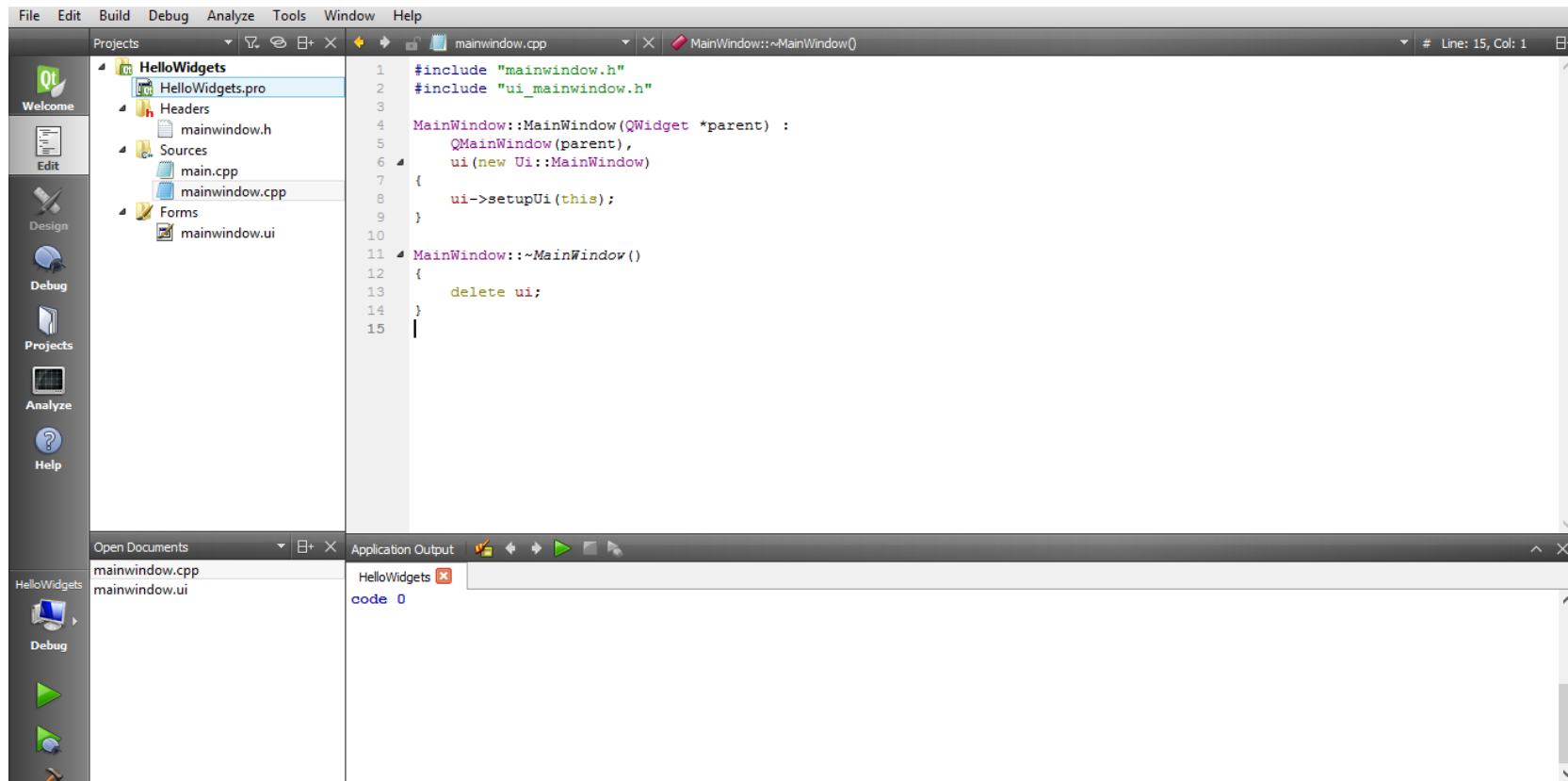
# Hello World

ปรับขนาดให้แสดงข้อความทั้งหมด ทดสอบ build & Run



# ทดสอบเปลี่ยนแปลงข้อความmainwindow.cpp

ให้ไปที่mainwindow.cpp → Edit → sources → mainwindow.cpp



# ทดสอบเปลี่ยนแปลงข้อความmainwindow.cpp

จะเห็นว่าใน mainwindos.cpp จะประกอบไปด้วย constructor [MainWindow::MainWindow]

ซึ่งจะเข้ามาทำงานใน Method นี้ครั้งแรก (เพิ่มเติมศึกษา OOP) ซึ่งเราจะลองเพิ่มคำสั่งในการเปลี่ยนแปลงข้อความ label ซึ่งมีวิธีการเรียกใช้งานดังนี้

```
1  #include "mainwindow.h"
2  #include "ui_mainwindow.h"
3
4  MainWindow::MainWindow(QWidget *parent) :
5      QMainWindow(parent),
6      ui(new Ui::MainWindow)
7  {
8      ui->setupUi(this);
9      ui->label->setText("My First Application");
10 }
11
12 MainWindow::~MainWindow()
13 {
14     delete ui;
15 }
```

ทดสอบการ run program

# ทดสอบเปลี่ยนแปลงข้อความmainwindow.cpp

จะพบว่าตัวอักษรที่เราตั้งไว้ตอนแรกได้ เป็นสีแดง ขนาด 18 หายไป กลายเป็น Default ดังนั้นในการสั่งการจาก cpp ไปสู่หน้า UI เราต้องตั้งค่าขนาดและสีด้วยด้วย ซึ่งผมจะใช้ Stylesheet ในการสั่งงาน ซึ่งสามารถศึกษาเพิ่มเติมได้

```
1  #include "mainwindow.h"
2  #include "ui_mainwindow.h"
3
4  MainWindow::MainWindow(QWidget *parent) :
5      QMainWindow(parent),
6      ui(new Ui::MainWindow)
7  {
8      ui->setupUi(this);
9      ui->label->setText("My First Application");
10     ui->label->setStyleSheet("color:red;font-size:18px");
11 }
12
13 MainWindow::~MainWindow()
14 {
15     delete ui;
16 }
```

ทดสอบการ run program