

# Qt – Built-in Dialogs

Yih-Chuan Lin

CSIE Windows Programming Class

National Formosa University

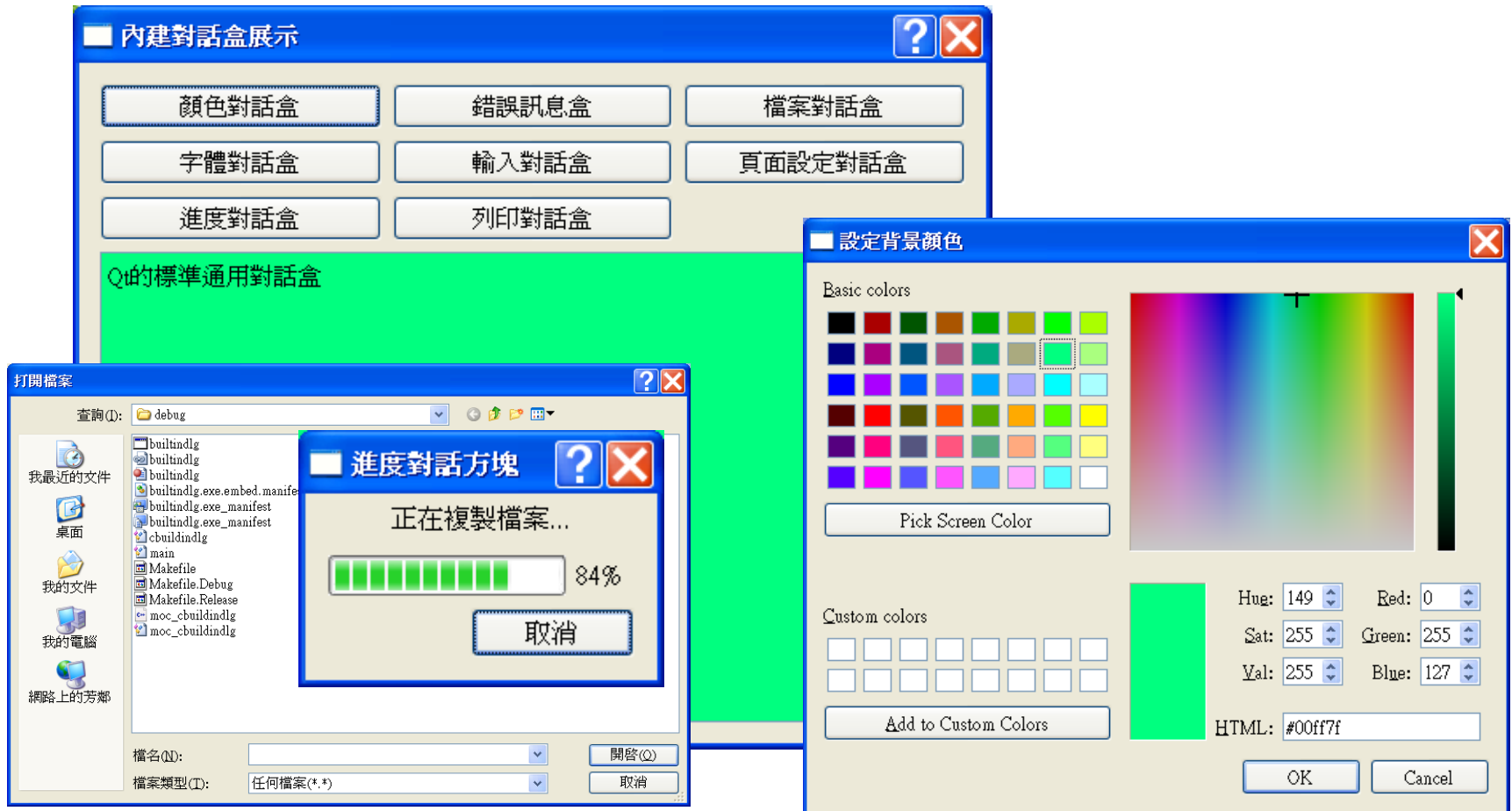
# Qt- Built-in Dialog

- 學習目的
  - 認識Qt各種內建之對話盒視窗
  - 練習使用這些對話盒視窗
  - 學習如何整合內建對話盒視窗於應用程式中



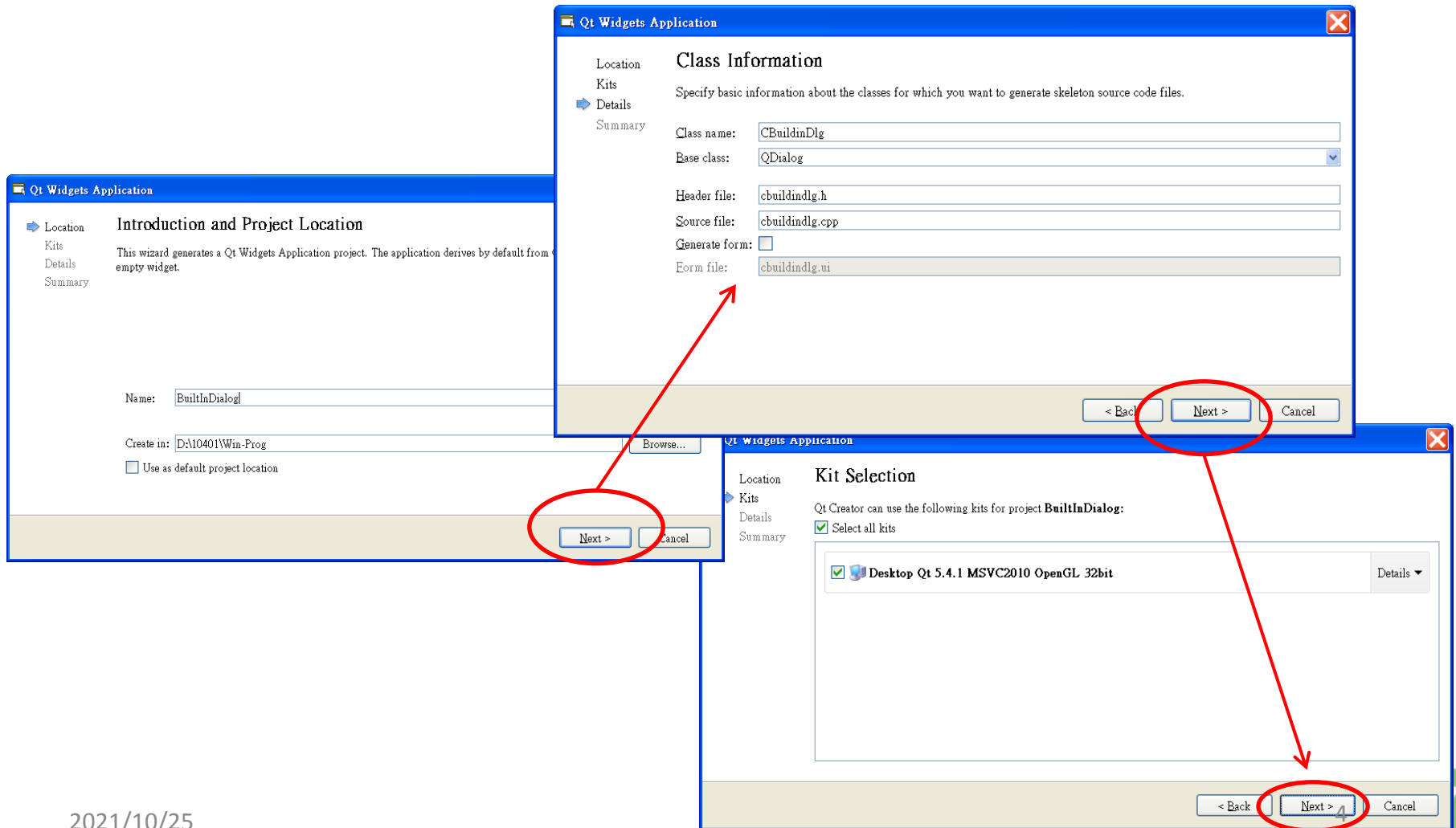
Code less.  
Create more.  
Deploy everywhere.

# Qt- Built-in Dialog



# Qt- Built-in Dialog

- Create a dialog-based application:



The image displays three sequential screenshots of the Qt Widgets Application wizard, illustrating the steps to create a dialog-based application. Red circles and arrows highlight the 'Next >' buttons in each step.

**Step 1: Introduction and Project Location**

This wizard generates a Qt Widgets Application project. The application derives by default from empty widget.

Name:

Create in:

☐ Use as default project location

**Step 2: Class Information**

Specify basic information about the classes for which you want to generate skeleton source code files.

Class name:

Base class:

Header file:

Source file:

Generate form: ☐

Form file:

**Step 3: Kit Selection**

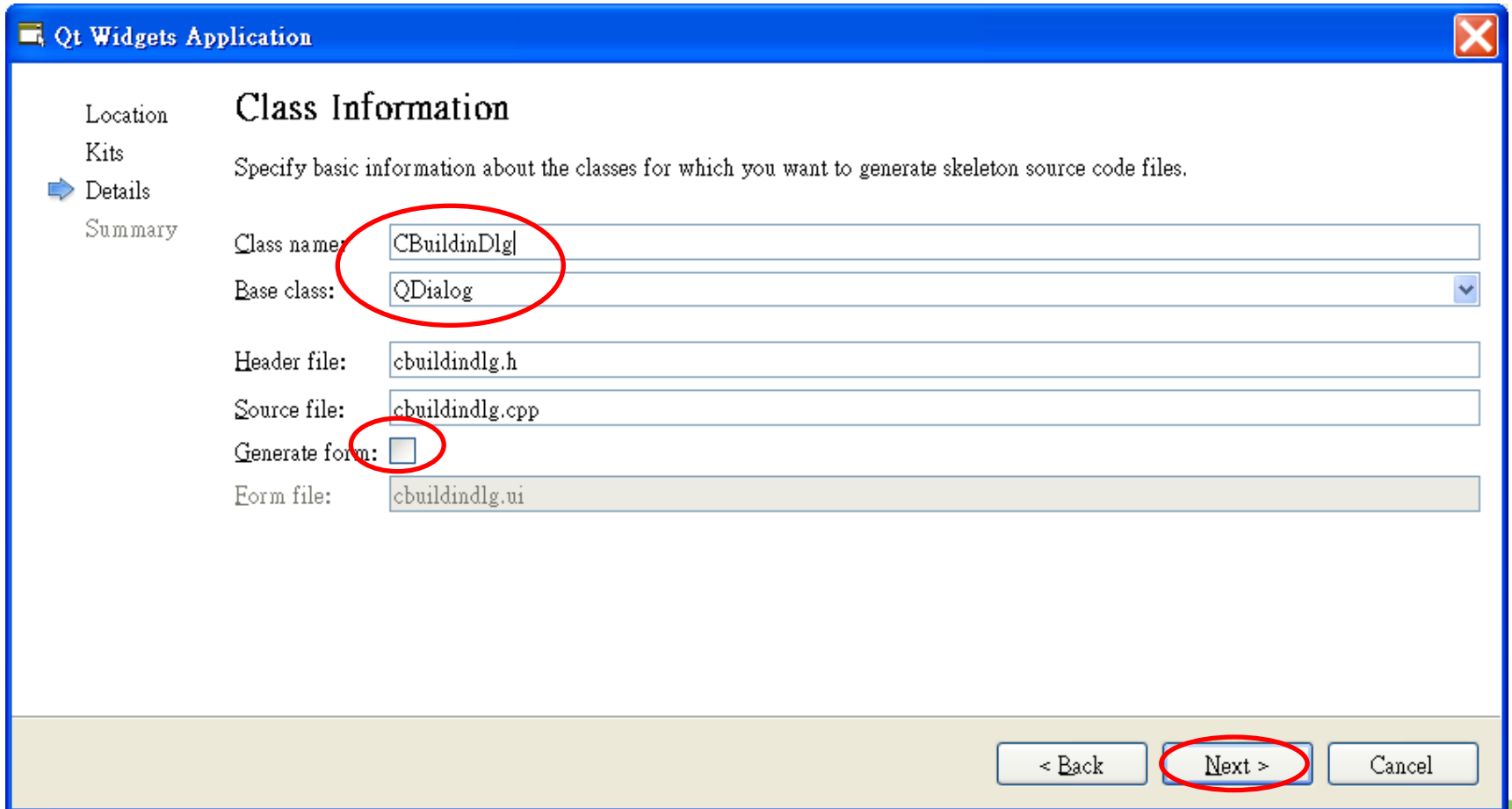
Qt Creator can use the following kits for project **BuiltInDialog**:

☒ Select all kits

☒ Desktop Qt 5.4.1 MSVC2010 OpenGL 32bit

# Qt- Built-in Dialog

- Create a dialog-based application:



The image shows a screenshot of the "Qt Widgets Application" dialog box. The "Details" tab is selected in the left sidebar. The "Class Information" section is active, with the instruction "Specify basic information about the classes for which you want to generate skeleton source code files." The following fields are visible:

- Class name:
- Base class:
- Header file:
- Source file:
- Generate form: ☐
- Form file:

At the bottom right, there are three buttons: "< Back", "Next >", and "Cancel". The "Next >" button is highlighted with a red circle.

# Qt- Built-in Dialog

## ● Create a dialog-based application:

The screenshot shows the Qt Creator interface during the creation of a dialog-based application. The 'Qt Widgets Application' wizard is open, and the 'Edit' button is highlighted with a red circle. The wizard's 'Files to be added in' section lists the following files:

- D:\10401\Win-Prog\BuiltInDialog:
- BuiltInDialog.pro
- cbuildindlg.cpp
- cbuildindlg.h
- main.cpp

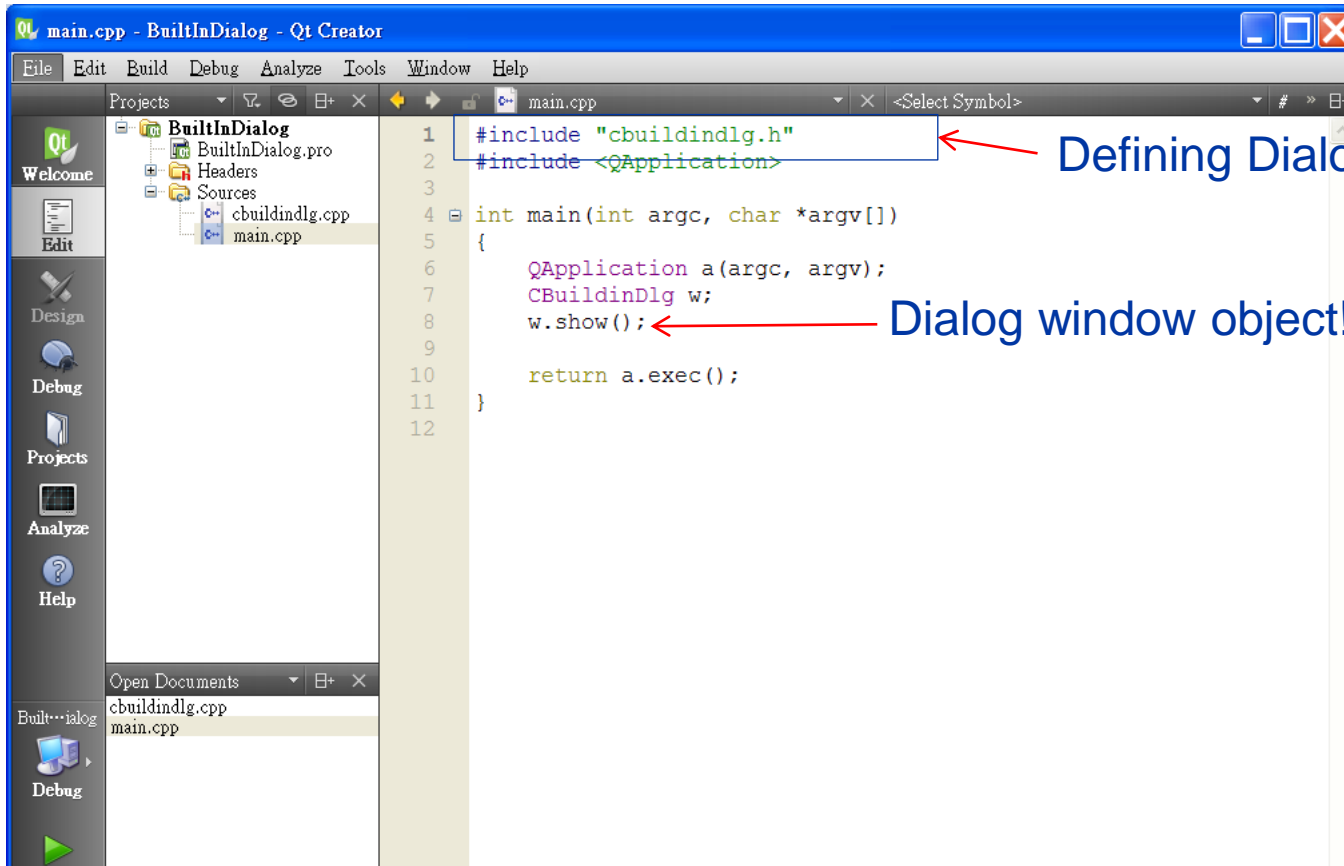
The 'Finish' button is also highlighted with a red circle. The 'Build Settings' dialog is open, showing the 'Debug' configuration selected in the 'Edit build configuration' dropdown. The 'Shadow build' checkbox is highlighted with a red circle. The 'Build directory' is set to D:\10401\Win-Prog\BuiltInDialog.

```

1 #include "cbuildindlg.h"
2 #include <QApplication>
3
4 int main(int argc, char *argv[])
5 {
6     QApplication a(argc, argv);
7     CBuildinDlg w;
8     w.show();
9
10    return a.exec();
11 }
12
  
```

# Qt- Built-in Dialog

- Check out the main function:



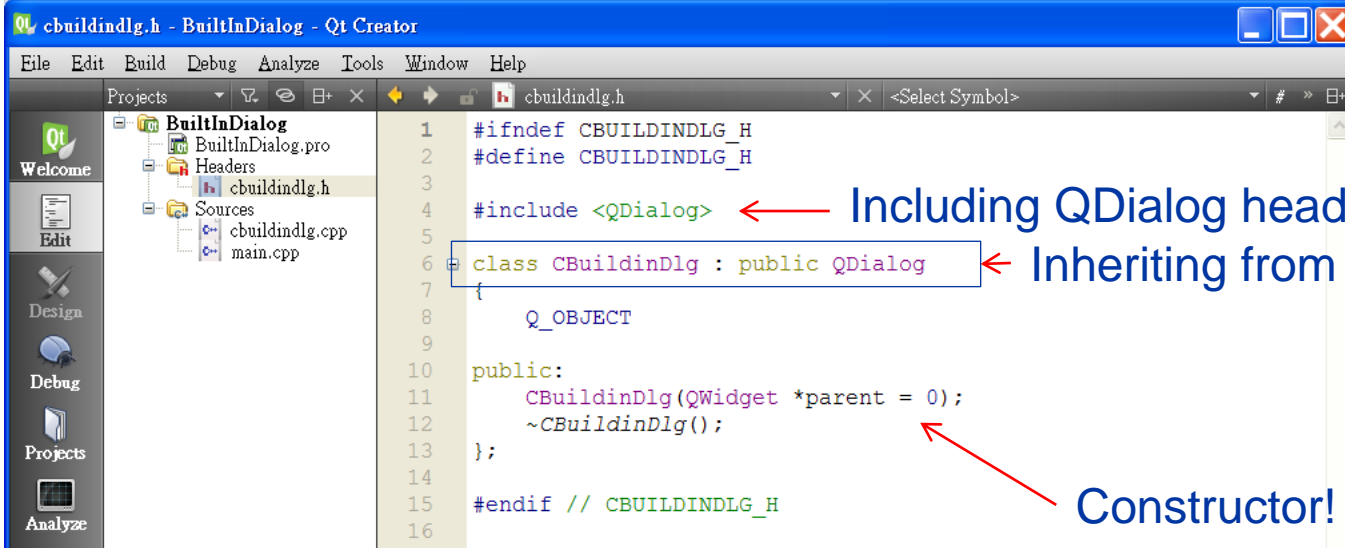
```
1 #include "cbuildindlg.h"
2 #include <QApplication>
3
4 int main(int argc, char *argv[])
5 {
6     QApplication a(argc, argv);
7     CBuildinDlg w;
8     w.show();
9
10    return a.exec();
11 }
12
```

Defining Dialog window!

Dialog window object!

# Qt- Built-in Dialog

- Check out the definition of CBuildinDlg class:



The screenshot shows the Qt Creator IDE with the file `cbuilddlg.h` open. The code defines the `CBuildinDlg` class, which inherits from `QDialog`. Red arrows and text annotations highlight key parts of the code:

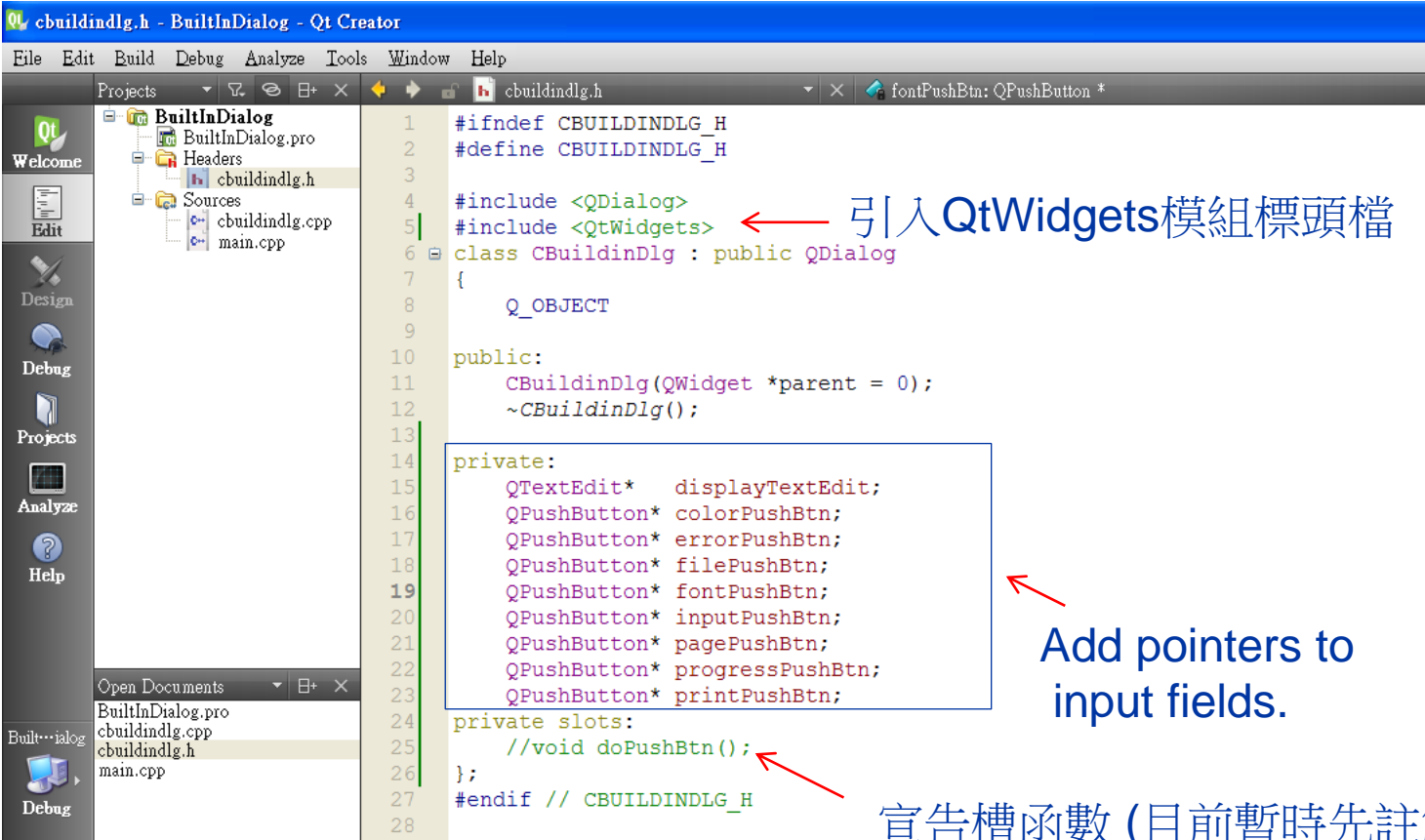
- `#include <QDialog>` is annotated with "Including QDialog header file!".
- `class CBuildinDlg : public QDialog` is annotated with "Inheriting from QDialog Class!".
- The constructor `CBuildinDlg(QWidget *parent = 0);` is annotated with "Constructor!".

```
1  #ifndef CBUILDINDLG_H
2  #define CBUILDINDLG_H
3
4  #include <QDialog>
5
6  class CBuildinDlg : public QDialog
7  {
8      Q_OBJECT
9
10 public:
11     CBuildinDlg(QWidget *parent = 0);
12     ~CBuildinDlg();
13 };
14
15 #endif // CBUILDINDLG_H
16
```



# Qt- Built-in Dialog

- Modify the definition of CBuildinDlg class:



```
1 #ifndef CBUILDINDLG_H
2 #define CBUILDINDLG_H
3
4 #include <QDialog>
5 #include <QtWidgets>
6 class CBuildinDlg : public QDialog
7 {
8     Q_OBJECT
9
10 public:
11     CBuildinDlg(QWidget *parent = 0);
12     ~CBuildinDlg();
13
14 private:
15     QTextEdit* displayTextEdit;
16     QPushButton* colorPushBtn;
17     QPushButton* errorPushBtn;
18     QPushButton* filePushBtn;
19     QPushButton* fontPushBtn;
20     QPushButton* inputPushBtn;
21     QPushButton* pagePushBtn;
22     QPushButton* progressPushBtn;
23     QPushButton* printPushBtn;
24 private slots:
25     //void doPushBtn();
26 };
27 #endif // CBUILDINDLG_H
28
```

引入QtWidgets模組標頭檔

Add pointers to input fields.

宣告槽函數 (目前暫時先註解)

# Qt- Built-in Dialog

- **Modify the constructor of CBuildinDlg class:**

cbuildindlg.cpp - BuiltInDialog - Qt Creator

```

1  #include "cbuildindlg.h"
2
3  CBuildinDlg::CBuildinDlg(QWidget *parent)
4      : QDialog(parent)
5  {
6
7      QGridLayout* gridLayout = new QGridLayout;
8      displayTextEdit = new QTextEdit(QStringLiteral("Qt的標準通用對話盒"));
9      colorPushBtn = new QPushButton(QStringLiteral("顏色對話盒"));
10     errorPushBtn = new QPushButton(QStringLiteral("錯誤訊息盒"));
11     filePushBtn = new QPushButton(QStringLiteral("檔案對話盒"));
12     fontPushBtn = new QPushButton(QStringLiteral("字體對話盒"));
13     inputPushBtn = new QPushButton(QStringLiteral("輸入對話盒"));
14     pagePushBtn = new QPushButton(QStringLiteral("頁面設定對話盒"));
15     progressPushBtn = new QPushButton(QStringLiteral("進度對話盒"));
16     printPushBtn = new QPushButton(QStringLiteral("列印對話盒"));
17 }
18
19 CBuildinDlg::~CBuildinDlg()
20 {
21 }
22
23

```

Establish layout managers!  
Generate pushbutton objects!

# Qt- Built-in Dialog

- **Modify the constructor of CBuildinDlg class:**

```

cbuildindlg.cpp - BuiltInDialog - Qt Creator
File Edit Build Debug Analyze Tools Window Help
Projects cbuildindlg.cpp* CBuildinDlg::CBuildinDlg(QWidget *)
5 {
6     QGridLayout* gridLayout = new QGridLayout;
7     displayTextEdit = new QTextEdit(QStringLiteral("Qt的標準通用對話盒"));
8     colorPushBtn = new QPushButton(QStringLiteral("顏色對話盒"));
9     errorPushBtn = new QPushButton(QStringLiteral("錯誤訊息盒"));
10    filePushBtn = new QPushButton(QStringLiteral("檔案對話盒"));
11    fontPushBtn = new QPushButton(QStringLiteral("字體對話盒"));
12    inputPushBtn = new QPushButton(QStringLiteral("輸入對話盒"));
13    pagePushBtn = new QPushButton(QStringLiteral("頁面設定對話盒"));
14    progressPushBtn = new QPushButton(QStringLiteral("進度對話盒"));
15    printPushBtn = new QPushButton(QStringLiteral("列印對話盒"));
16
17    gridLayout->addWidget(colorPushBtn,0,0,1,1);
18    gridLayout->addWidget(errorPushBtn,0,1,1,1);
19    gridLayout->addWidget(filePushBtn,0,2,1,1);
20    gridLayout->addWidget(fontPushBtn,1,0,1,1);
21    gridLayout->addWidget(inputPushBtn,1,1,1,1);
22    gridLayout->addWidget(pagePushBtn,1,2,1,1);
23    gridLayout->addWidget(progressPushBtn,2,0,1,1);
24    gridLayout->addWidget(printPushBtn,2,1,1,1);
25    gridLayout->addWidget(displayTextEdit,3,0,3,3);
26
27    setLayout(gridLayout);
28    setWindowTitle(QStringLiteral("內建對話盒展示"));
29    resize(400,300);
30 }
31
32
33 CBuildinDlg::~CBuildinDlg()
34 {

```

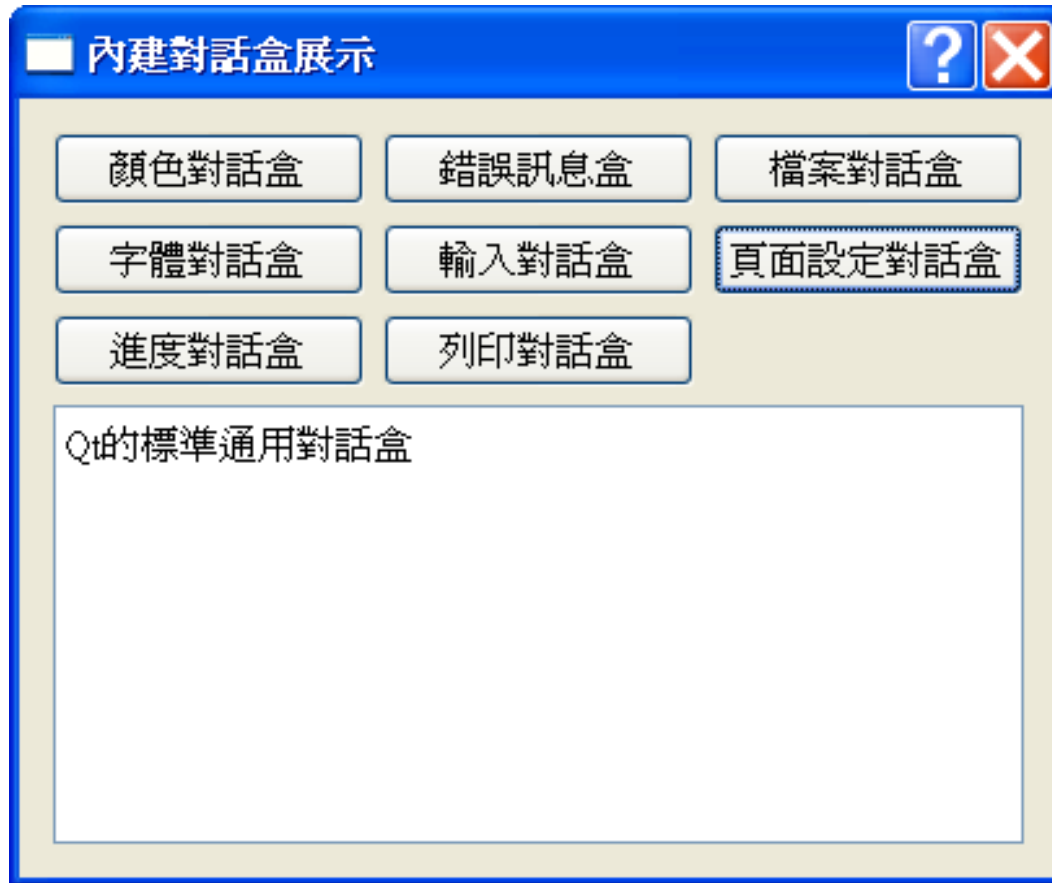
Put all the buttons and textedit into the grid layout manager!



Code less.  
Create more.  
Deploy everywhere.

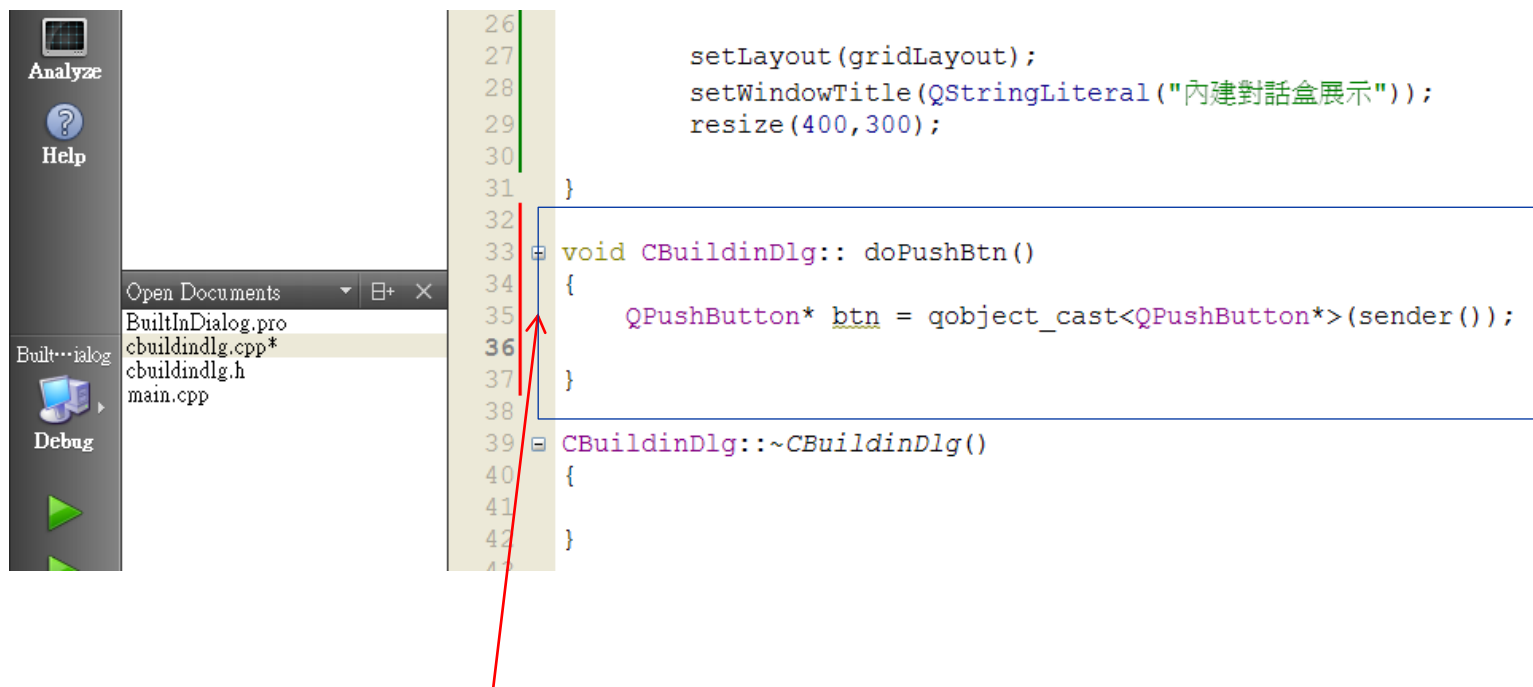
# Qt- Built-in Dialog

- 建置專案並執行程式、測試人機介面:



# Qt- Built-in Dialog

- 實作槽函數 `doPushBtn()`:



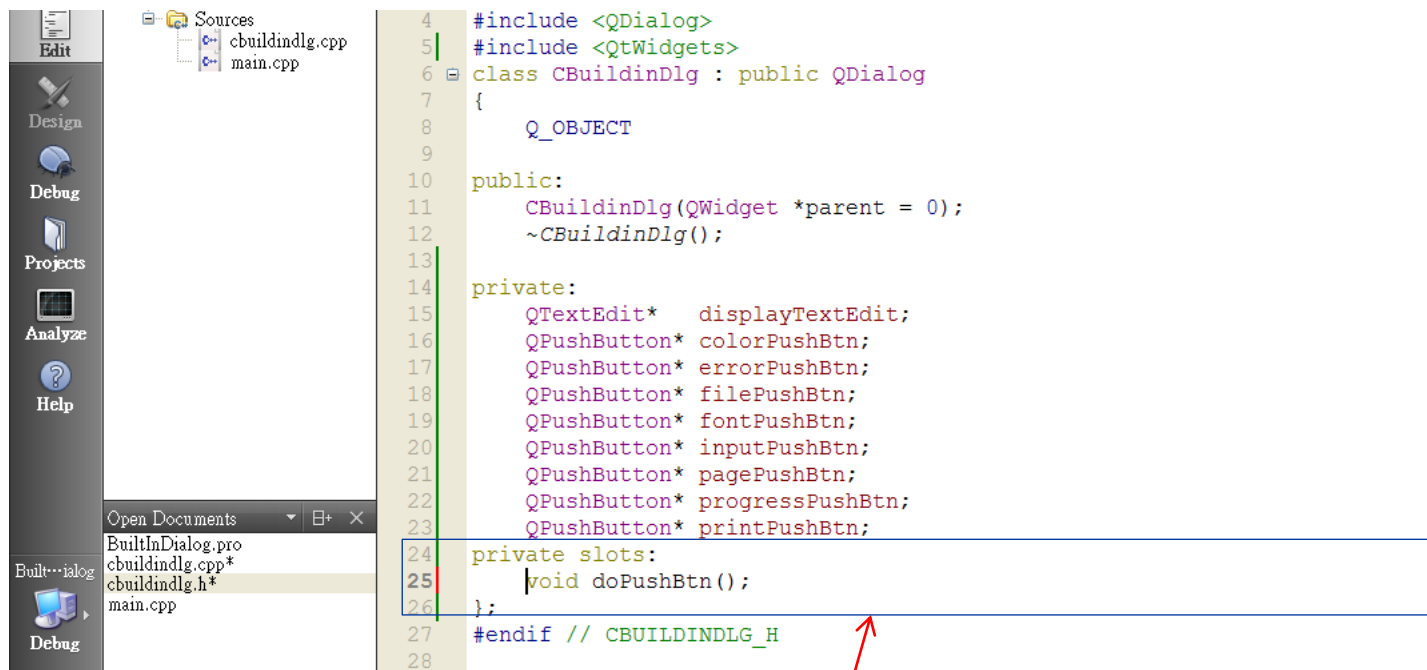
Implement the slot function `doPushBtn()`.

`sender()`: 擷取信號來源物件指標。

`qobject_cast<T>()`: Qt物件強迫轉型樣版函數

# Qt- Built-in Dialog

- 取消註解，回復槽函數宣告：



```
4 #include <QDialog>
5 #include <QtWidgets>
6 class CBuildinDlg : public QDialog
7 {
8     Q_OBJECT
9
10 public:
11     CBuildinDlg(QWidget *parent = 0);
12     ~CBuildinDlg();
13
14 private:
15     QTextEdit* displayTextEdit;
16     QPushButton* colorPushBtn;
17     QPushButton* errorPushBtn;
18     QPushButton* filePushBtn;
19     QPushButton* fontPushBtn;
20     QPushButton* inputPushBtn;
21     QPushButton* pagePushBtn;
22     QPushButton* progressPushBtn;
23     QPushButton* printPushBtn;
24 private slots:
25     void doPushBtn();
26 };
27 #endif // CBUILDINDLG_H
28
```

槽函數宣告

# Qt- Built-in Dialog

## ● 訊號與槽函數之相連:



```

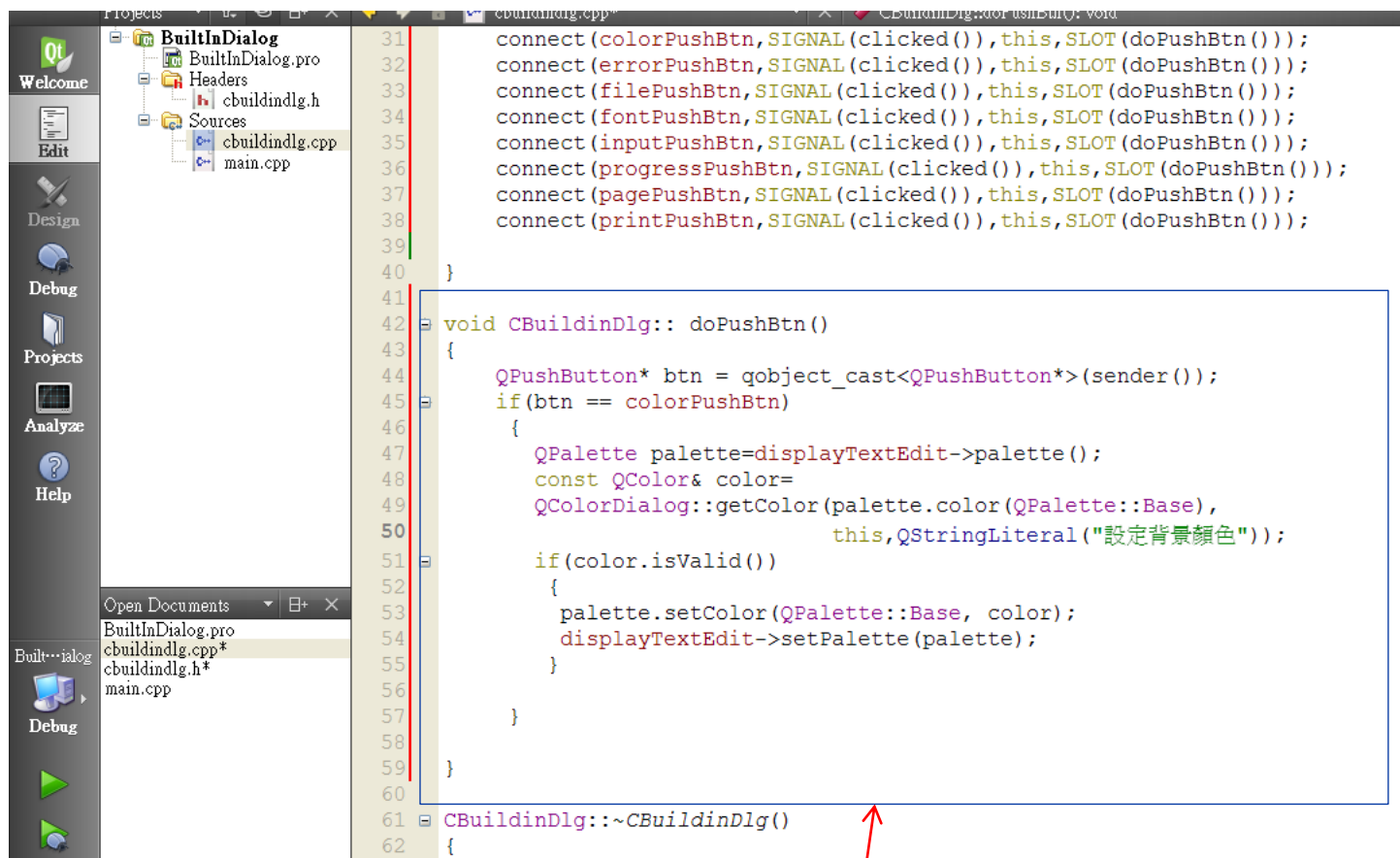
18  gridLayout->addWidget(errorPushBtn,0,1,1,1);
19  gridLayout->addWidget(filePushBtn,0,2,1,1);
20  gridLayout->addWidget(fontPushBtn,1,0,1,1);
21  gridLayout->addWidget(inputPushBtn,1,1,1,1);
22  gridLayout->addWidget(pagePushBtn,1,2,1,1);
23  gridLayout->addWidget(progressPushBtn,2,0,1,1);
24  gridLayout->addWidget(printPushBtn,2,1,1,1);
25  gridLayout->addWidget(displayTextEdit,3,0,3,3);
26
27  setLayout(gridLayout);
28  setWindowTitle(QStringLiteral("內建對話盒展示"));
29  resize(400,300);
30
31  connect(colorPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
32  connect(errorPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
33  connect(filePushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
34  connect(fontPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
35  connect(inputPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
36  connect(progressPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
37  connect(pagePushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
38  connect(printPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
39
40 }
41
42 void CBuildeInDlg:: doPushBtn()
43 {
44     QPushButton* btn = qobject_cast<QPushButton*>(sender());
45 }
46

```

多個訊號連接至相同槽函數

# Qt- Built-in Dialog

- 實作doPushBtn():



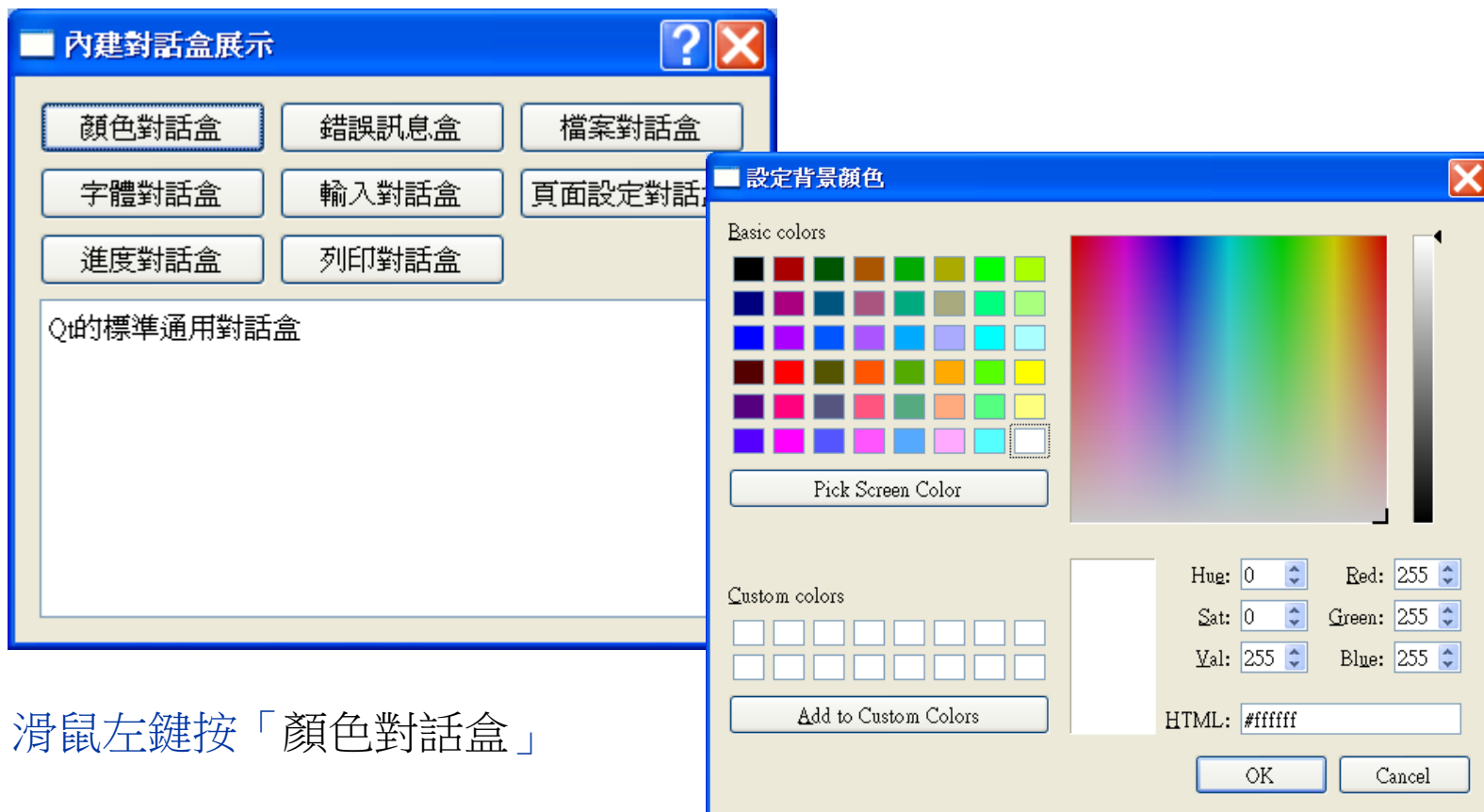
```
31 connect(colorPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
32 connect(errorPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
33 connect(filePushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
34 connect(fontPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
35 connect(inputPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
36 connect(progressPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
37 connect(pagePushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
38 connect(printPushBtn,SIGNAL(clicked()),this,SLOT(doPushBtn()));
39
40 }
41
42 void CBuildinDlg:: doPushBtn()
43 {
44     QPushButton* btn = qobject_cast<QPushButton*>(sender());
45     if(btn == colorPushBtn)
46     {
47         QPalette palette=displayTextEdit->palette();
48         const QColor& color=
49         QColorDialog::getColor(palette.color(QPalette::Base),
50                               this,QStringLiteral("設定背景顏色"));
51         if(color.isValid())
52         {
53             palette.setColor(QPalette::Base, color);
54             displayTextEdit->setPalette(palette);
55         }
56     }
57 }
58
59
60
61 CBuildinDlg::~CBuildinDlg()
62 {
```

實作 colorPushBtn之功能



# Qt- Built-in Dialog

- 建置及執行程式:



滑鼠左鍵按「顏色對話盒」

# Qt- Built-in Dialog

## ● 實作doPushBtn():

```

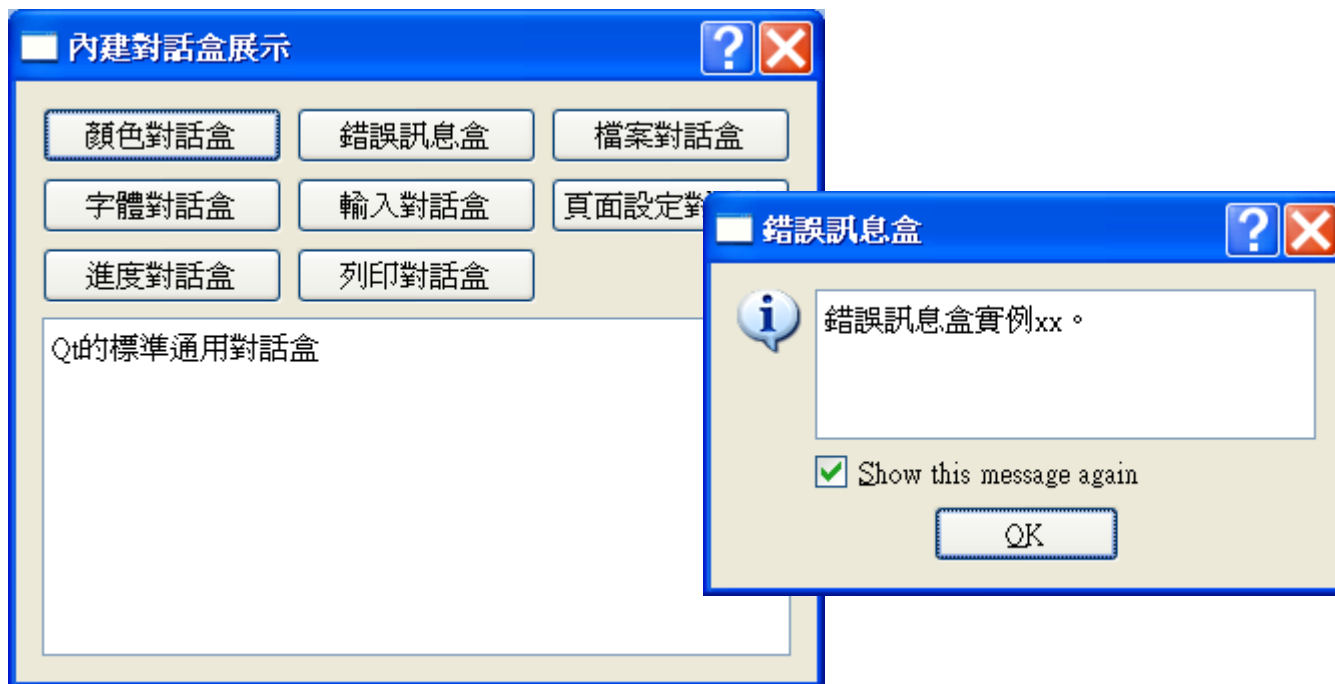
44  QPushButton* btn = qobject_cast<QPushButton*>(sender());
45  if(btn == colorPushBtn)
46  {
47      QPalette palette=displayTextEdit->palette();
48      const QColor& color=
49      QColorDialog::getColor(palette.color(QPalette::Base),
50                          this,QStringLiteral("設定背景顏色"));
51  if(color.isValid())
52  {
53      palette.setColor(QPalette::Base, color);
54      displayTextEdit->setPalette(palette);
55  }
56  }
57
58  if(btn == errorPushBtn)
59  {
60      QMessageBox box(this);
61      box.setWindowTitle(QStringLiteral("錯誤訊息盒"));
62      box.showMessage(QStringLiteral("錯誤訊息盒實例xx:"));
63      box.showMessage(QStringLiteral("錯誤訊息盒實例yy:"));
64      box.showMessage(QStringLiteral("錯誤訊息盒實例zz:"));
65      box.exec();
66  }
67
68
69
70  CBuildinDlg::~CBuildinDlg()
71  {
72
73

```

實作 errorPushBtn之功能

# Qt- Built-in Dialog

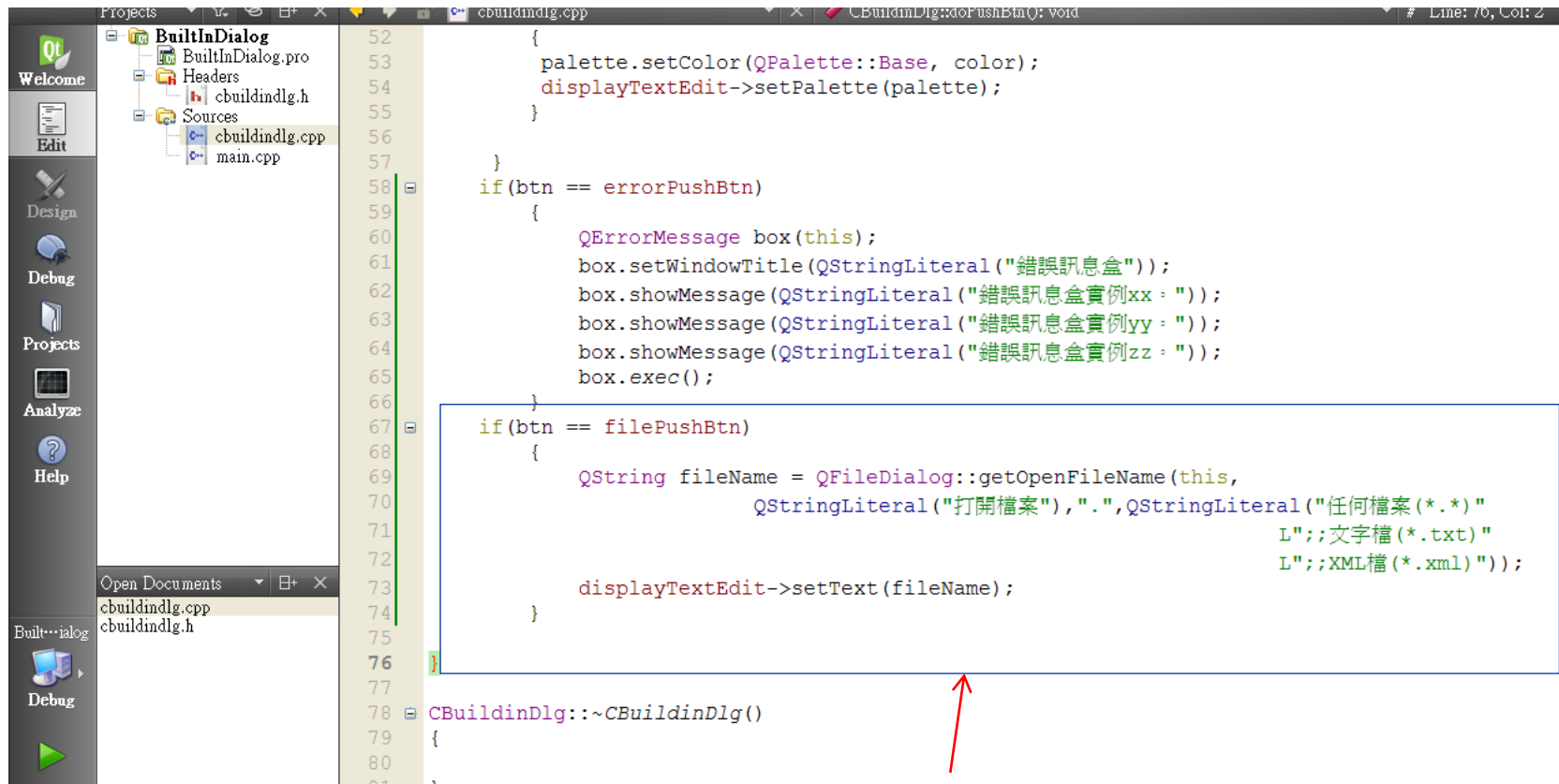
- 建置及執行程式:



滑鼠左鍵按「錯誤訊息盒」三次

# Qt- Built-in Dialog

## ● 實作doPushBtn():



```

52     {
53         palette.setColor(QPalette::Base, color);
54         displayTextEdit->setPalette(palette);
55     }
56
57 }
58
59 if(btn == errorPushBtn)
60 {
61     QMessageBox box(this);
62     box.setWindowTitle(QStringLiteral("錯誤訊息盒"));
63     box.showMessage(QStringLiteral("錯誤訊息盒實例xx: "));
64     box.showMessage(QStringLiteral("錯誤訊息盒實例yy: "));
65     box.showMessage(QStringLiteral("錯誤訊息盒實例zz: "));
66     box.exec();
67 }
68
69 if(btn == filePushBtn)
70 {
71     QString fileName = QFileDialog::getOpenFileName(this,
72                                                     QStringLiteral("打開檔案"), ".",
73                                                     QStringLiteral("任何檔案 (*.*) "
74                                                         "L";;"文字檔 (*.txt) "
75                                                         "L";;"XML檔 (*.xml) "));
76     displayTextEdit->setText(fileName);
77 }
78
79 CBuildinDlg::~CBuildinDlg()
80 {
81

```

實作 filePushBtn之功能

# Qt- Built-in Dialog

- 建置及執行程式:



滑鼠左鍵按「檔案對話盒」

# Qt- Built-in Dialog

## ● 實作doPushBtn():

```

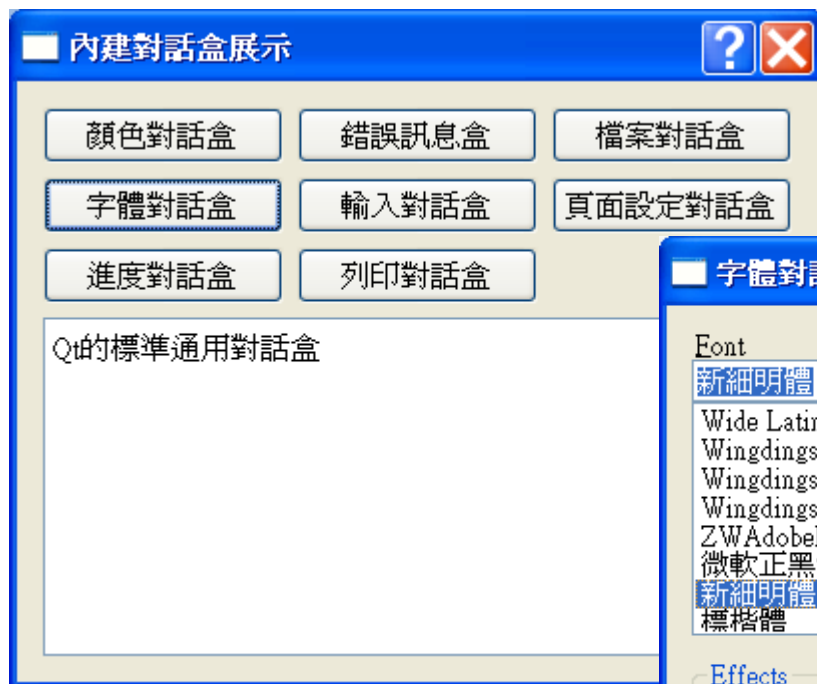
59 {
60     QMessageBox box(this);
61     box.setWindowTitle(QStringLiteral("錯誤訊息盒"));
62     box.showMessage(QStringLiteral("錯誤訊息盒實例xx : "));
63     box.showMessage(QStringLiteral("錯誤訊息盒實例yy : "));
64     box.showMessage(QStringLiteral("錯誤訊息盒實例zz : "));
65     box.exec();
66 }
67 if(btn == filePushBtn)
68 {
69     QString fileName = QFileDialog::getOpenFileName(this,
70     QStringLiteral("打開檔案"), ".", QStringLiteral("任何檔案 (*.*)"
71     L";;文字檔 (*.txt)"
72     L";;XML檔 (*.xml)"));
73     displayTextEdit->setText(fileName);
74 }
75 if(btn == fontPushBtn)
76 {
77     bool ok;
78     const QFont& font = QFontDialog::getFont(&ok,
79     displayTextEdit->font(),
80     this,
81     QStringLiteral("字體對話盒"));
82     if (ok) displayTextEdit->setFont(font);
83 }
84 }
85
86 }
87

```

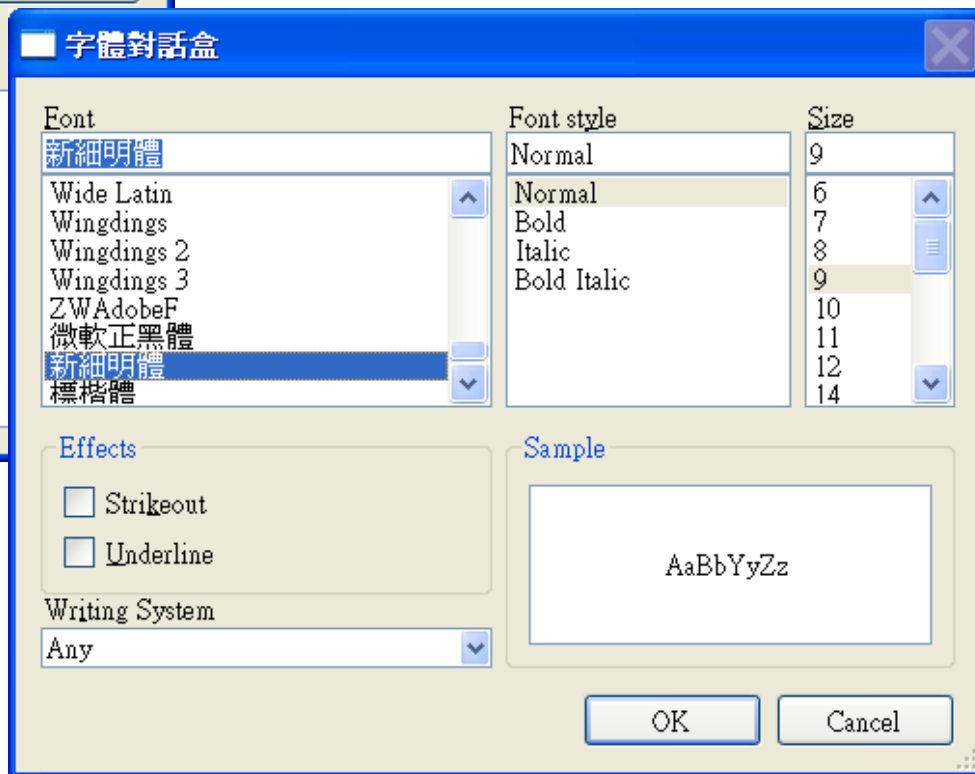
實作 fontPushBtn之功能

# Qt- Built-in Dialog

- 建置及執行程式:



滑鼠左鍵按「字體對話盒」



# Qt- Built-in Dialog

## ● 實作doPushBtn():

```

71
72
73     displayTextEdit->setText(fileName);
74 }
75 if(btn == fontPushBtn)
76 {
77     bool ok;
78     const QFont& font = QFontDialog::getFont(&ok,
79                                             displayTextEdit->font(),
80                                             this,
81                                             QStringLiteral("字體對話盒"));
82     if (ok) displayTextEdit->setFont(font);
83 }
84
85 if (btn == inputPushBtn)
86 {
87     bool ok;
88     QString text = QInputDialog::getText(this,
89                                         QStringLiteral("輸入對話盒"),
90                                         QStringLiteral("輸入文字"),
91                                         QLineEdit::Normal,
92                                         QDir::home().dirName(),
93                                         &ok
94                                         );
95     if (ok && !text.isEmpty()) displayTextEdit->setText(text);
96 }
97
98 }
99
100 CBuildinDlg::~CBuildinDlg()
101 {
102

```

實作 inputPushBtn之功能

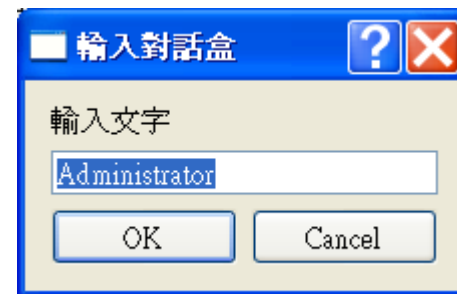
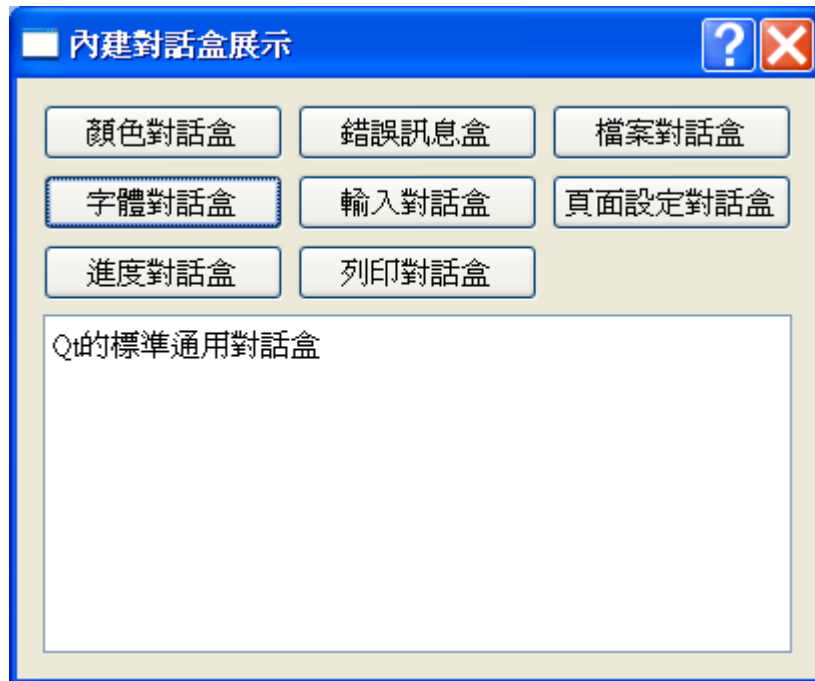




Code less.  
Create more.  
Deploy everywhere.

# Qt- Built-in Dialog

- 建置及執行程式:



滑鼠左鍵按「輸入對話盒」

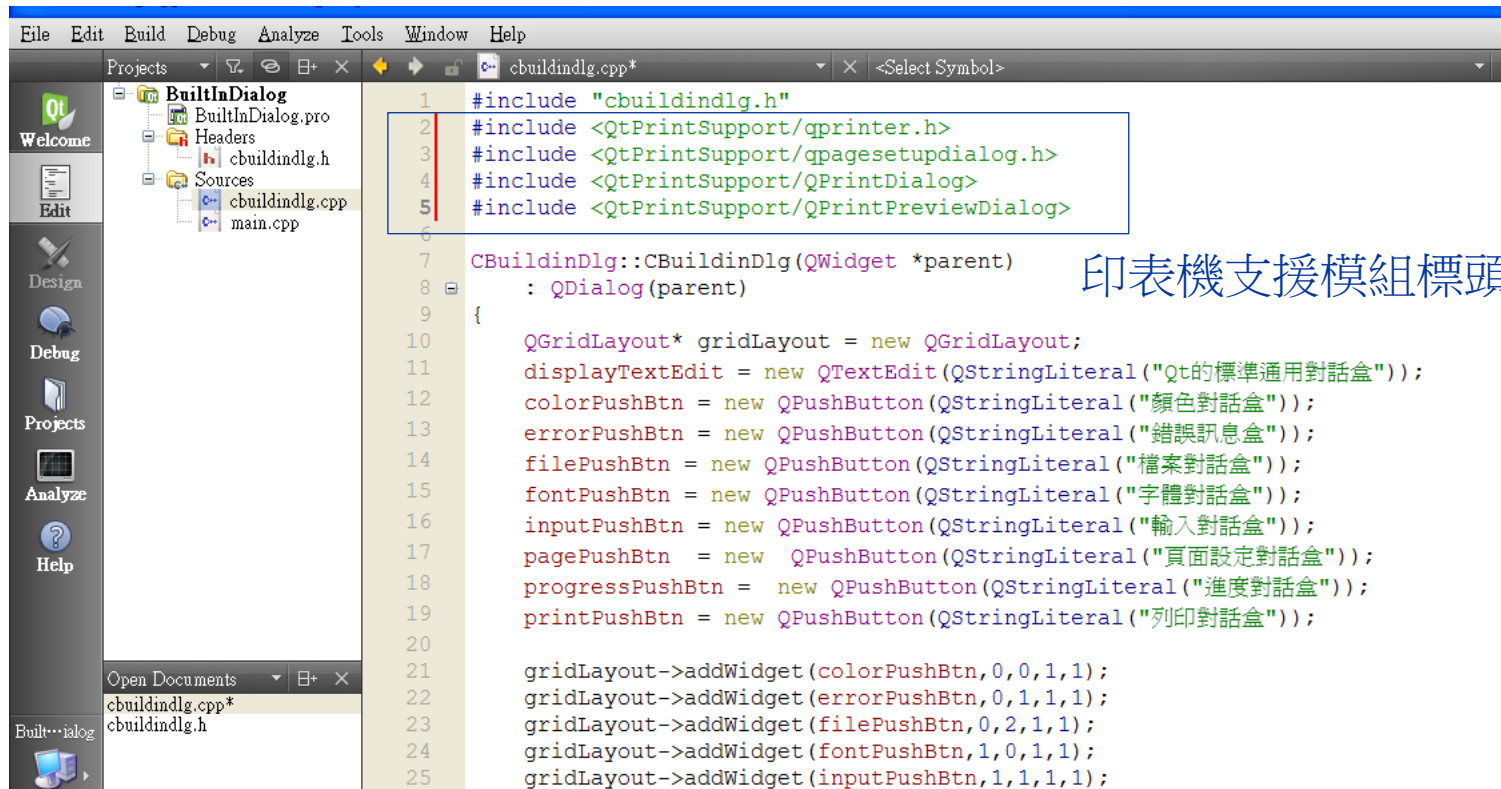




Code less.  
Create more.  
Deploy everywhere.

# Qt- Built-in Dialog

- 新增印表機支援模組標頭檔:



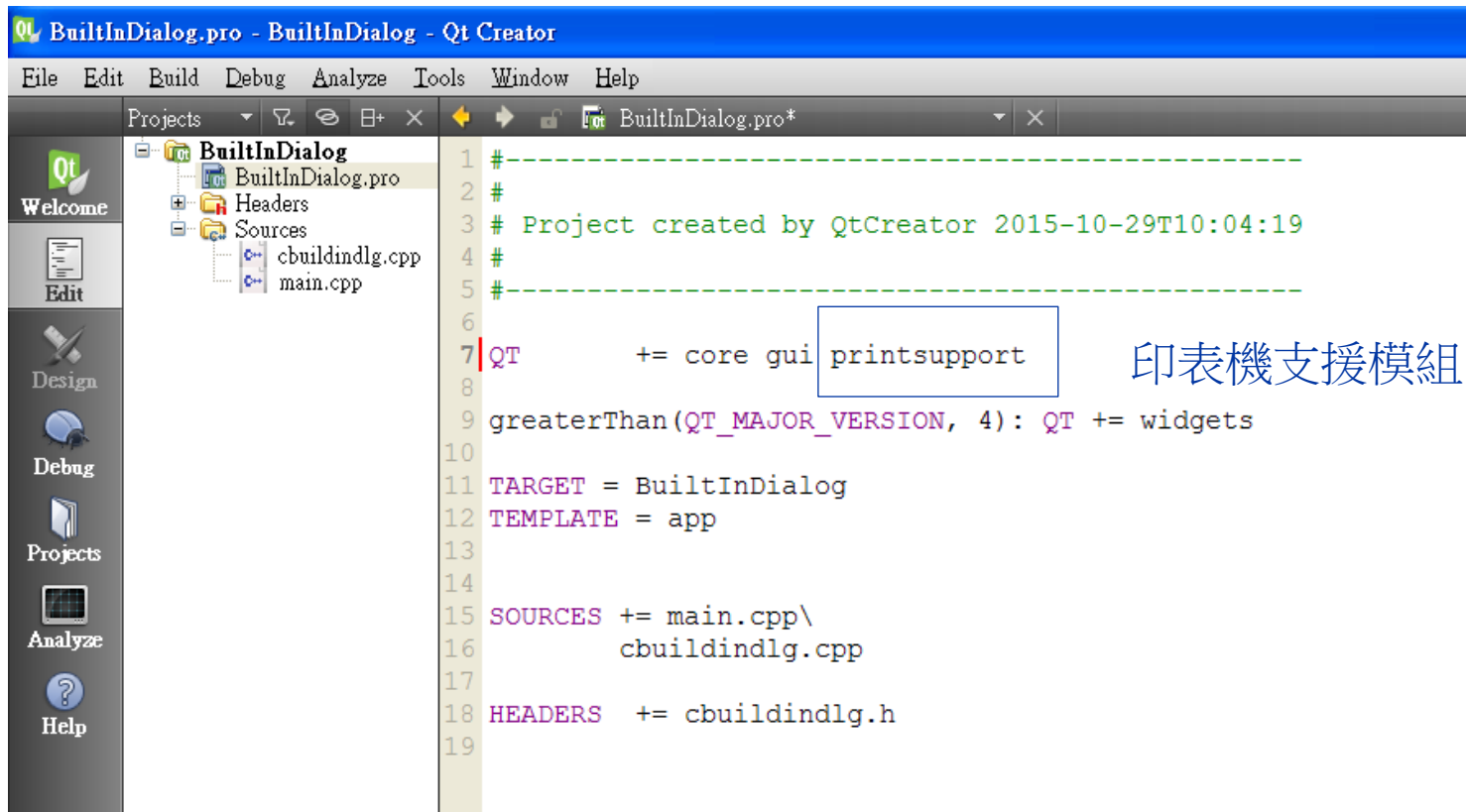
```
1 #include "cbuildindlg.h"
2 #include <QtPrintSupport/qprinter.h>
3 #include <QtPrintSupport/qpagesetupdialog.h>
4 #include <QtPrintSupport/QPrintDialog>
5 #include <QtPrintSupport/QPrintPreviewDialog>
6
7 CBuildinDlg::CBuildinDlg(QWidget *parent)
8     : QDialog(parent)
9 {
10     QGridLayout* gridLayout = new QGridLayout;
11     displayTextEdit = new QTextEdit(QStringLiteral("Qt的標準通用對話盒"));
12     colorPushBtn = new QPushButton(QStringLiteral("顏色對話盒"));
13     errorPushBtn = new QPushButton(QStringLiteral("錯誤訊息盒"));
14     filePushBtn = new QPushButton(QStringLiteral("檔案對話盒"));
15     fontPushBtn = new QPushButton(QStringLiteral("字體對話盒"));
16     inputPushBtn = new QPushButton(QStringLiteral("輸入對話盒"));
17     pagePushBtn = new QPushButton(QStringLiteral("頁面設定對話盒"));
18     progressPushBtn = new QPushButton(QStringLiteral("進度對話盒"));
19     printPushBtn = new QPushButton(QStringLiteral("列印對話盒"));
20
21     gridLayout->addWidget(colorPushBtn,0,0,1,1);
22     gridLayout->addWidget(errorPushBtn,0,1,1,1);
23     gridLayout->addWidget(filePushBtn,0,2,1,1);
24     gridLayout->addWidget(fontPushBtn,1,0,1,1);
25     gridLayout->addWidget(inputPushBtn,1,1,1,1);
```

印表機支援模組標頭檔



# Qt- Built-in Dialog

- 專案檔新增印表機支援模組:



```
1 #-----
2 #
3 # Project created by QtCreator 2015-10-29T10:04:19
4 #
5 #-----
6
7 QT       += core gui printsupport
8
9 greaterThan(QT_MAJOR_VERSION, 4): QT += widgets
10
11 TARGET = BuiltInDialog
12 TEMPLATE = app
13
14
15 SOURCES += main.cpp\
16          cbuildindlg.cpp
17
18 HEADERS  += cbuildindlg.h
19
```

# Qt- Built-in Dialog

- 實作頁面設定功能:



```

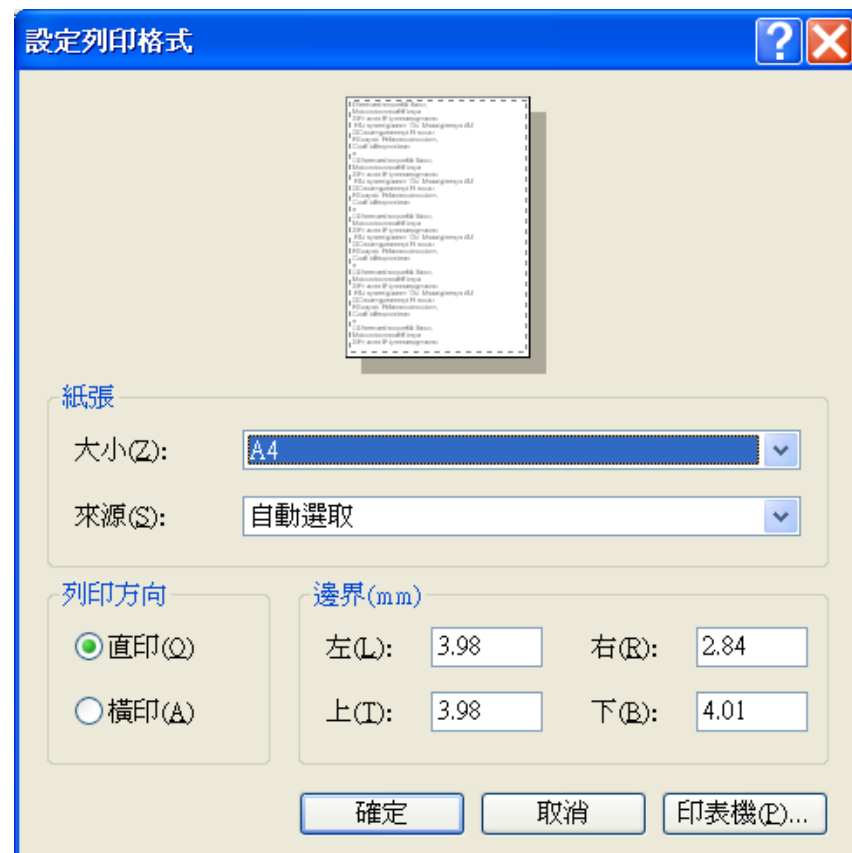
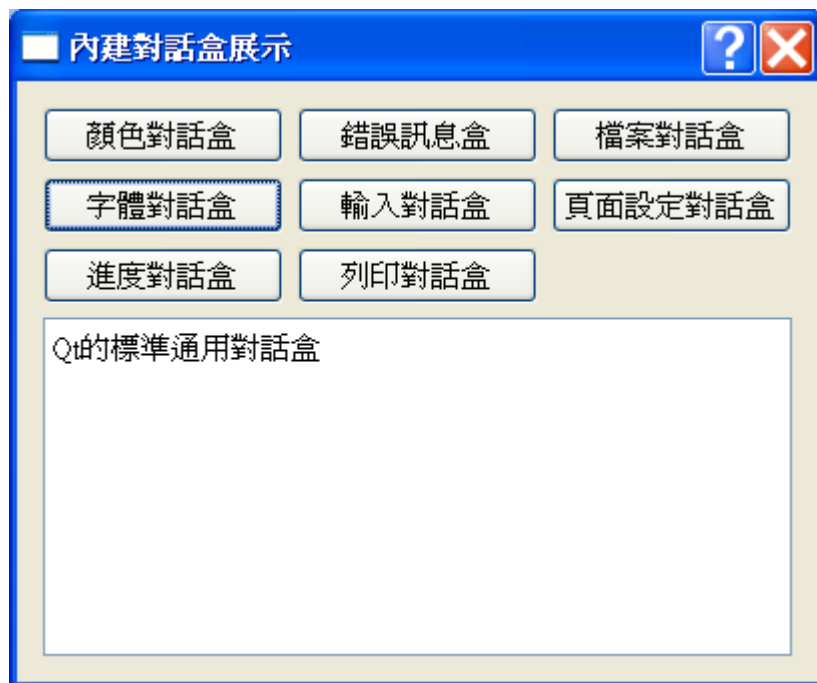
100                                     );
101         if (ok && !text.isEmpty()) displayTextEdit->setText(text);
102     }
103     if (btn == pagePushBtn)
104     {
105         QPainter printer(QPrinter::HighResolution);
106         QPageSetupDialog* dlg = new QPageSetupDialog(&printer, this);
107         dlg->setWindowTitle(QStringLiteral("頁面設定話方塊"));
108         if (dlg->exec() == QDialog::Accepted)
109         {
110
111         }
112     }
113
114
115     if (btn == pagePushBtn)
116     {
117         QPainter printer(QPrinter::HighResolution);
118         QPageSetupDialog* dlg = new QPageSetupDialog(&printer, this);
119         dlg->setWindowTitle(QStringLiteral("頁面設定話方塊"));
120         if (dlg->exec() == QDialog::Accepted)
121         {
122
123         }
124     }
125
126
127 }
128
129 CBuildinDlg::~CBuildinDlg()

```

頁面設定程式碼

# Qt- Built-in Dialog

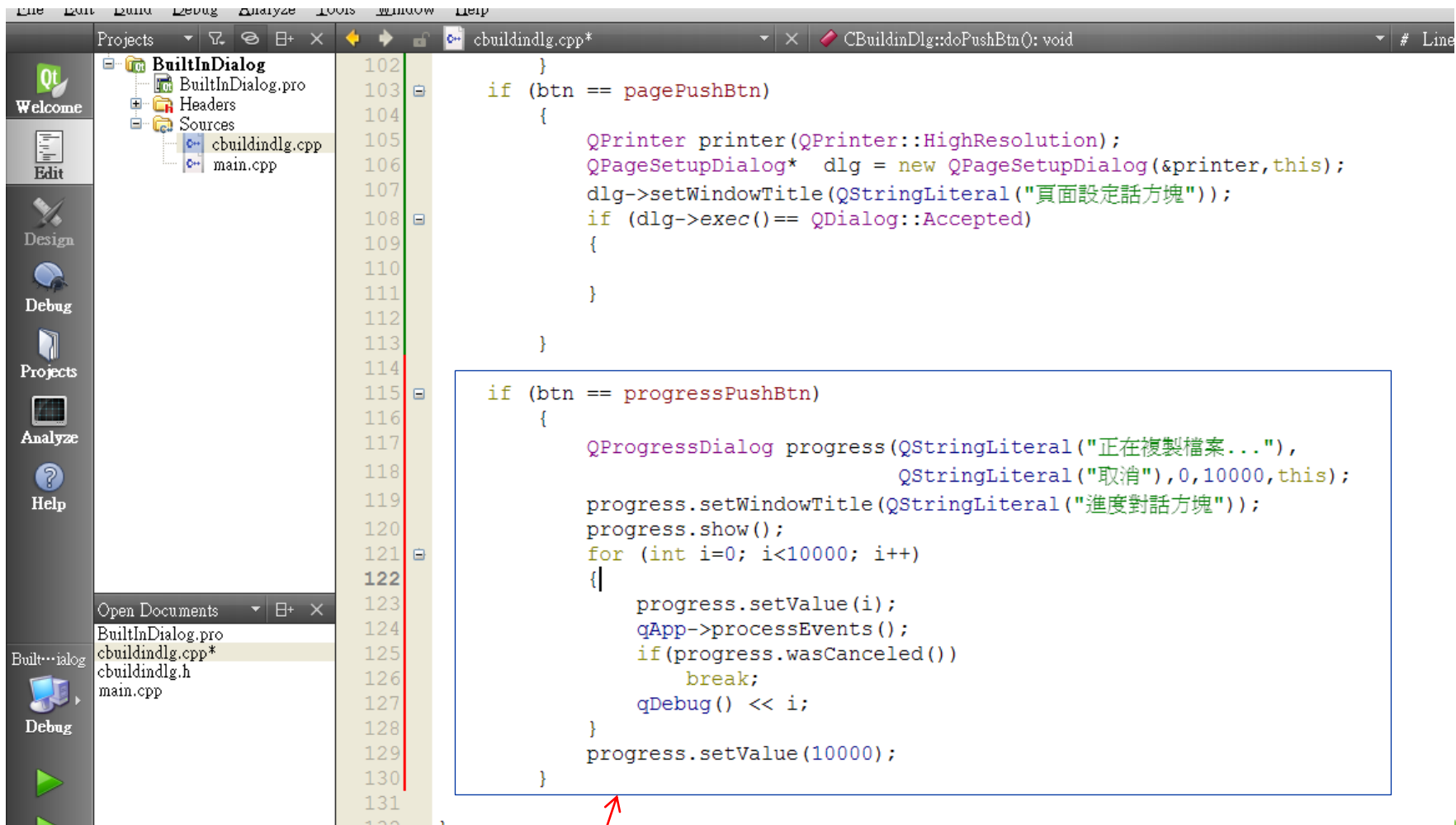
- 建置及執行程式:



滑鼠左鍵按「頁面設定對話盒」

# Qt- Built-in Dialog

## ● 實作進度對話盒功能:

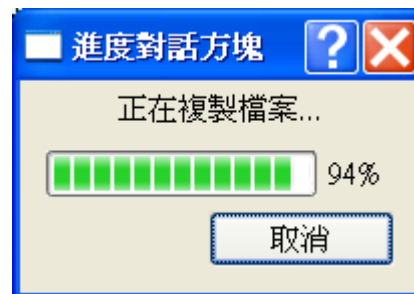
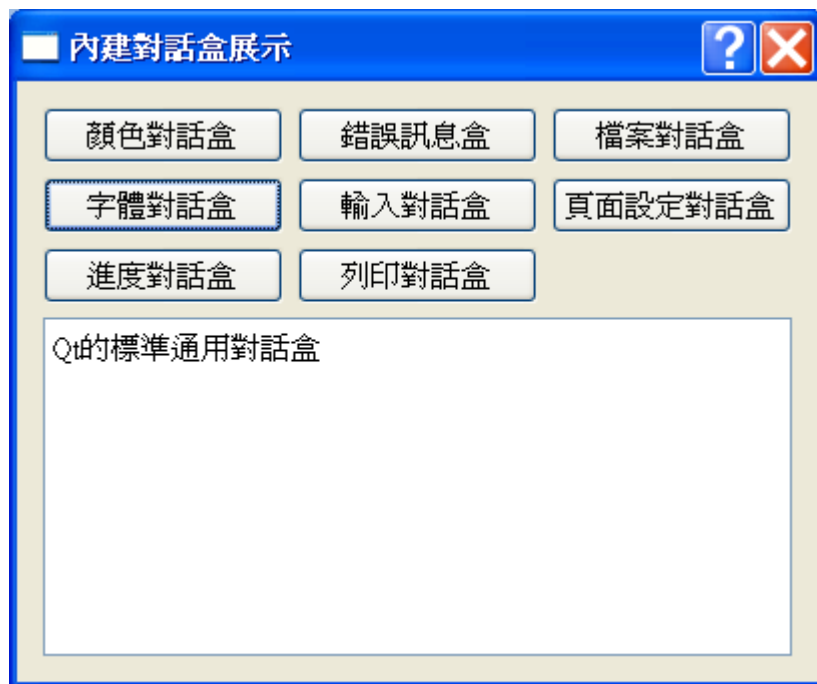


```
102     }
103     if (btn == pagePushBtn)
104     {
105         QPrinter printer(QPrinter::HighResolution);
106         QPageSetupDialog* dlg = new QPageSetupDialog(&printer, this);
107         dlg->setWindowTitle(QStringLiteral("頁面設定對話方塊"));
108         if (dlg->exec() == QDialog::Accepted)
109         {
110
111         }
112     }
113
114
115     if (btn == progressPushBtn)
116     {
117         QProgressDialog progress(QStringLiteral("正在複製檔案..."),
118                                 QStringLiteral("取消"), 0, 10000, this);
119         progress.setWindowTitle(QStringLiteral("進度對話方塊"));
120         progress.show();
121         for (int i=0; i<10000; i++)
122         {
123             progress.setValue(i);
124             QApplication->processEvents();
125             if (progress.wasCanceled())
126                 break;
127             qDebug() << i;
128         }
129         progress.setValue(10000);
130     }
131
132
```

進度對話盒功能程式碼

# Qt- Built-in Dialog

- 建置及執行程式:



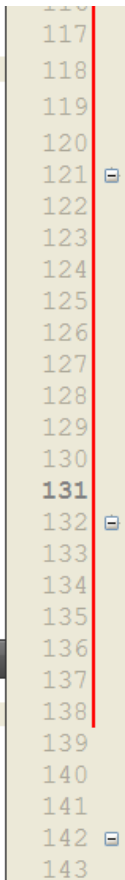
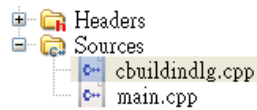
滑鼠左鍵按「進度對話盒」



Code less.  
Create more.  
Deploy everywhere.

# Qt- Built-in Dialog

## ● 實作列印對話盒功能:



```
117     QProgressDialog progress(QStringLiteral("正在複製檔案..."),
118                             QStringLiteral("取消"), 0, 10000, this);
119     progress.setWindowTitle(QStringLiteral("進度對話方塊"));
120     progress.show();
121     for (int i=0; i<10000; i++)
122     {
123         progress.setValue(i);
124         QApplication->processEvents();
125         if (progress.wasCanceled())
126             break;
127         qDebug() << i;
128     }
129     progress.setValue(10000);
130 }
131
132 if (btn == printPushBtn)
133 {
134     QPrinter printer(QPrinter::HighResolution);
135     QPrintDialog dialog(&printer, this);
136     if (dialog.exec() != QDialog::Accepted)
137         return;
138 }
139
140 }
141
142 CBuildinDlg::~CBuildinDlg()
143 {
```

↑  
列印對話盒功能程式碼



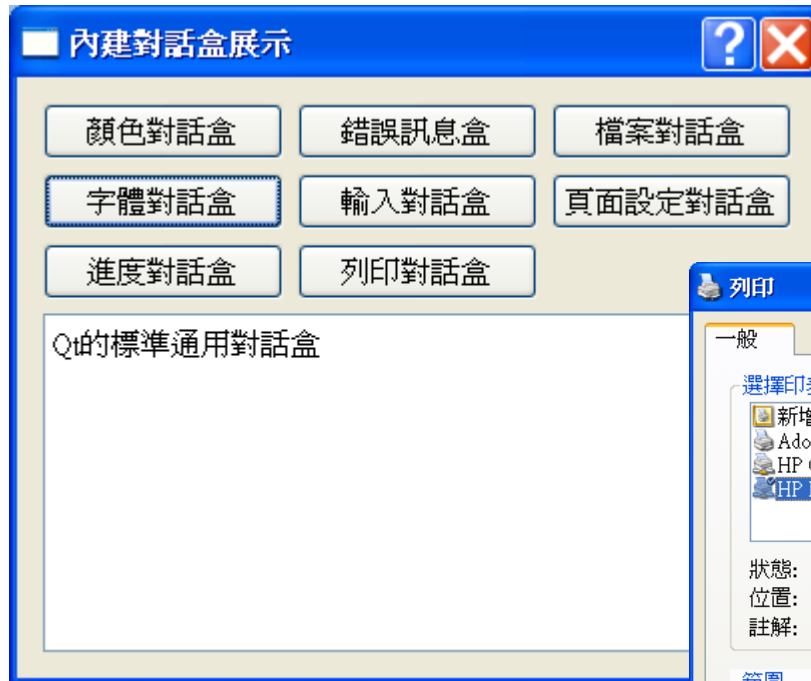




Code less.  
Create more.  
Deploy everywhere.

# Qt- Built-in Dialog

- 建置及執行程式:



滑鼠左鍵按「列印對話盒」

