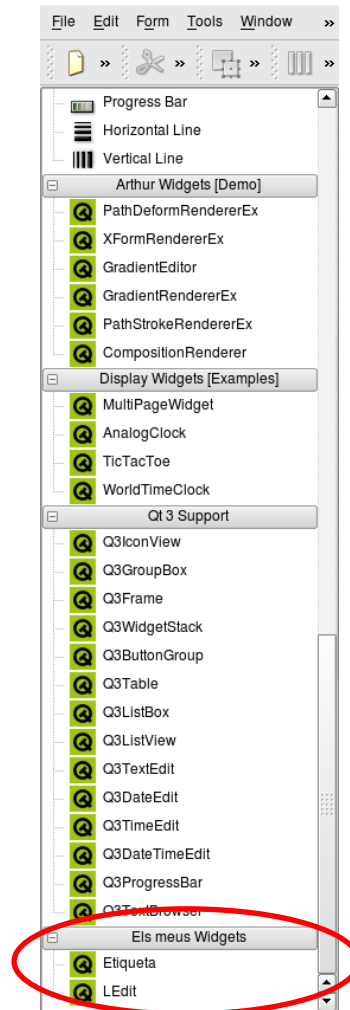


# Crear els nostres propis Widgets i incorporar-los al Designer

Objectiu:



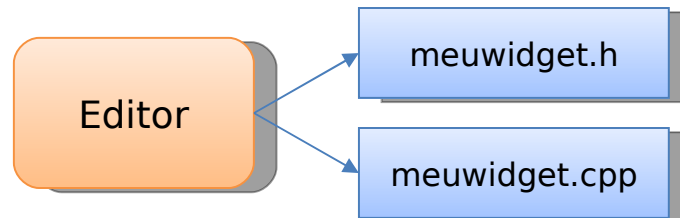
# Crear els nostres propis Widgets i incorporar-los al Designer

## Pasos:

1. Crear la classe per al nostre Widget.
2. Crear una classe que implementa un Plugin. El plugin contindrà el nostre widget i servirà per incorporar-lo al Designer.
3. Compilar les classes del Widget i del Plugin.
4. Instal·lar el plugin.
5. Configurar el sistema per tal que el Designer trobi el plugin.

# Crear els nostres propis Widgets i incorporar-los al Designer

## 1. Crear la classe per al nostre Widget.



### //Fitxer meuwidget.h

```
#include <QtDesigner/QDesignerExportWidget>

class QDESIGNER_WIDGET_EXPORT meuwidget :
public QObject
{
    Q_OBJECT

public:
    meuwidget(QWidget *parent = 0);

public slots:
    //slots de la classe
signals:
    //signals de la classe
};
```

### // Fitxer meuwidget.cpp

```
#include "meuwidget.h"

meuwidget::meuwidget(QWidget* parent):
QObject(parent)
{
}

// Implementació dels slots
```

# Crear els nostres propis Widgets i incorporar-los al Designer

2. Crear la classe del Plugin.



# Crear els nostres propis Widgets i incorporar-los al Designer

```
// Fitxer meuwidgetplugin.h

#include <QDesignerCustomWidgetInterface>

class meuwidgetplugin: public QObject, public QDesignerCustomWidgetInterface
{
    Q_OBJECT
    Q_INTERFACES(QDesignerCustomWidgetInterface)

public:
    meuwidgetplugin(QObject *parent=0);
    bool isContainer() const;
    bool isInitialized() const;
    QIcon icon() const;
    QString domXml() const;
    QString group() const;
    QString includeFile() const;
    QString name() const;
    QString toolTip() const;
    QString whatsThis() const;
    QWidget *createWidget(QWidget *parent);
    void initialize(QDesignerFormEditorInterface *core);
private:
    bool initialized;
};
```

# Crear els nostres propis Widgets i incorporar-los al Designer

## // Fitxer meuwidgetplugin.cpp

```
#include "meuwidget.h"
#include "meuwidgetplugin.h"
#include <QtPlugin>

meuwidgetplugin :: meuwidgetplugin(QObject *parent):
QObject(parent)
{
    initialized = false;
}

void meuwidgetplugin ::initialize(QDesignerFormEditorInterface * core)
{
    if (initialized)
        return;
    initialized = true;
}

bool meuwidgetplugin ::isInitialized() const
{
    return initialized;
}

QWidget * meuwidgetplugin ::createWidget(QWidget *parent)
{
    return new meuwidget(parent);           // Construir el Widget
}

QString meuwidgetplugin ::name() const
{
    return "meuwidget";                    // El nom de la classe del Widget
}

QString meuwidgetplugin ::group() const
{
    return "Els meus Widgets";
}
```

```
QIcon meuwidgetplugin ::icon() const
{
    return QIcon();
}

QString meuwidgetplugin ::toolTip() const
{
    return "";
}

QString meuwidgetplugin ::whatsThis() const
{
    return "";
}

bool meuwidgetplugin ::isContainer() const
{
    return false;
}

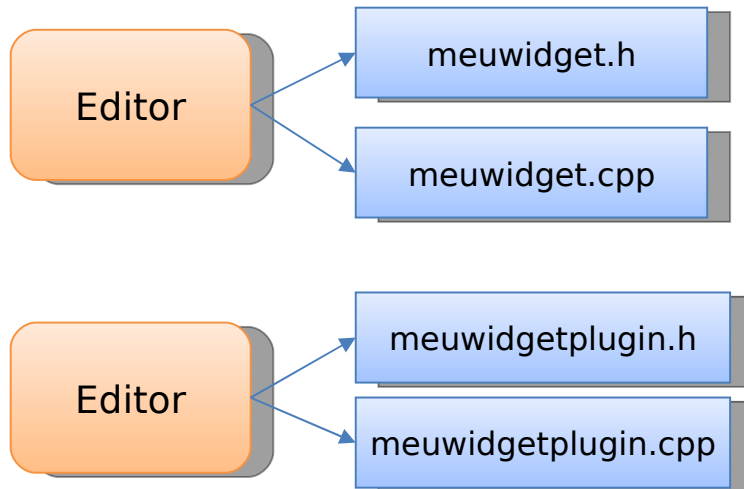
QString meuwidgetplugin ::domXml() const
{
    return "<widget class=\"meuwidget\" name=\"ObjectNameDelWidgetAlDesigner\">\n"
        "  <property name=\"geometry\">\n"
        "    <rect>\n"
        "      <x>0</x>\n"
        "      <y>0</y>\n"
        "      <width>100</width>\n"
        "      <height>100</height>\n"
        "    </rect>\n"
        "  </property>\n"
        "</widget>\n";
}

QString meuwidgetplugin ::includeFile() const
{
    return "meuwidget.h";
}

Q_EXPORT_PLUGIN2(meuwidget, meuwidgetplugin)
```

# Crear els nostres propis Widgets i incorporar-los al Designer

## 3. Compilar les classes del Widget i del Plugin.



```
// Fitxer meuwidgetplugin.pro
```

```
CONFIG+=designer plugin release
```

```
TEMPLATE=lib
```

```
TARGET =
```

```
DEPENDPATH += .
```

```
INCLUDEPATH += .
```

```
HEADERS+=meuwidget.h meuwidgetplugin.h
```

```
SOURCES += meuwidget.cpp meuwidgetplugin.cpp
```

```
target.path=~/.designer/plugins/designer
```

```
INSTALLS+=target
```

Fem

```
qmake; make
```

Es generaran uns fitxers `moc_*.cpp` i el fitxer `libmeuwidget.so`

# Crear els nostres propis Widgets i incorporar-los al Designer

## 4. Instal·lar el plugin.

```
// Fitxer meuwidgetplugin.pro

CONFIG+=designer plugin release
TEMPLATE=lib
TARGET =
DEPENDPATH += .
INCLUDEPATH += .

HEADERS+=meuwidget.h meuwidgetplugin.h
SOURCES += meuwidget.cpp meuwidgetplugin.cpp

target.path=~/.designer/plugins/designer
INSTALLS+=target
```

Directori d'instal·lació.

La primera part ha de ser igual al valor de la variable **\$QT\_PLUGIN\_PATH**

Fem

**make install**

S'instal·larà el fitxer **libmeuwidgetplugin.so** al target.path

# Crear els nostres propis Widgets i incorporar-los al Designer

## 5. Configurar el sistema

El Designer busca plugins al seu directori per defecte i també a

`$QT_PLUGIN_PATH/designer`

Per tant hem de configurar la variable `QT_PLUGIN_PATH`

Al fitxer `~/.bashrc`:

```
QT_PLUGIN_PATH=~/.designer/plugins
export QT_PLUGIN_PATH
```

O al fitxer `~/.tcshrc`:

```
setenv QT_PLUGIN_PATH ~/.designer/plugins
```

depenent de quin és el nostre shell (`echo $SHELL`).