



**Qt dans l'Enseignement**

## Les grandes lignes...







## Traduction française et Adaptation des supports pour l'enseignement

© 2012 Digia Plc.

The enclosed Qt Materials are provided under the Creative Commons Attribution-Share Alike 2.5 License Agreement.



The full license text is available here:  
<http://creativecommons.org/licenses/by-sa/2.5/legalcode>.

Digia, Qt and the Digia and Qt logos are the registered trademarks of Digia Plc. in Finland and other countries worldwide.

Qt Essentials - Fundamentals of Qt Module  
Training Course

Visit us at <http://qt.digia.com>  
Produced by Digia Plc.  
Material based on Qt 5.0, created on September 27, 2012

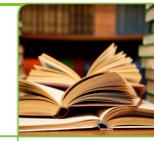
  
Digia Plc.

<http://www.qt.io/training-materials/>  
<http://www.qt.io/qt-essentials-widget-edition/>

 2



## Qu'est-ce que Qt ?



*"Qt est une librairie de développement multiplateforme écrite en C++."*

- C++ de base mais adaptation pour d'autres
  - Python, Ruby, C#, etc.
- Multiplateforme pour tous les aspects
  - Interfaces, bases de données, XML, Web,
  - multimedia, réseau, OpenGL, types de base...

digia 3



## Qu'est-ce que Qt?

- C++ avec macros + introspection

```
foreach (int value, intList) { ... }
```

```
QObject *o = new QPushButton;
o->metaObject()->className(); // returns "QPushButton"
```

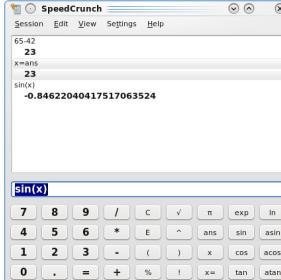
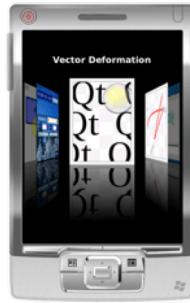
```
connect(button, SIGNAL(clicked()), window, SLOT(close()));
```

digia 4

# Qt Objectifs de Qt



- Code unique quelle que soit la plateforme
- Applications avec « look and feel » natif

- API facile à (ré-)utiliser, haute productivité, outils simples à utiliser

**digia** 5

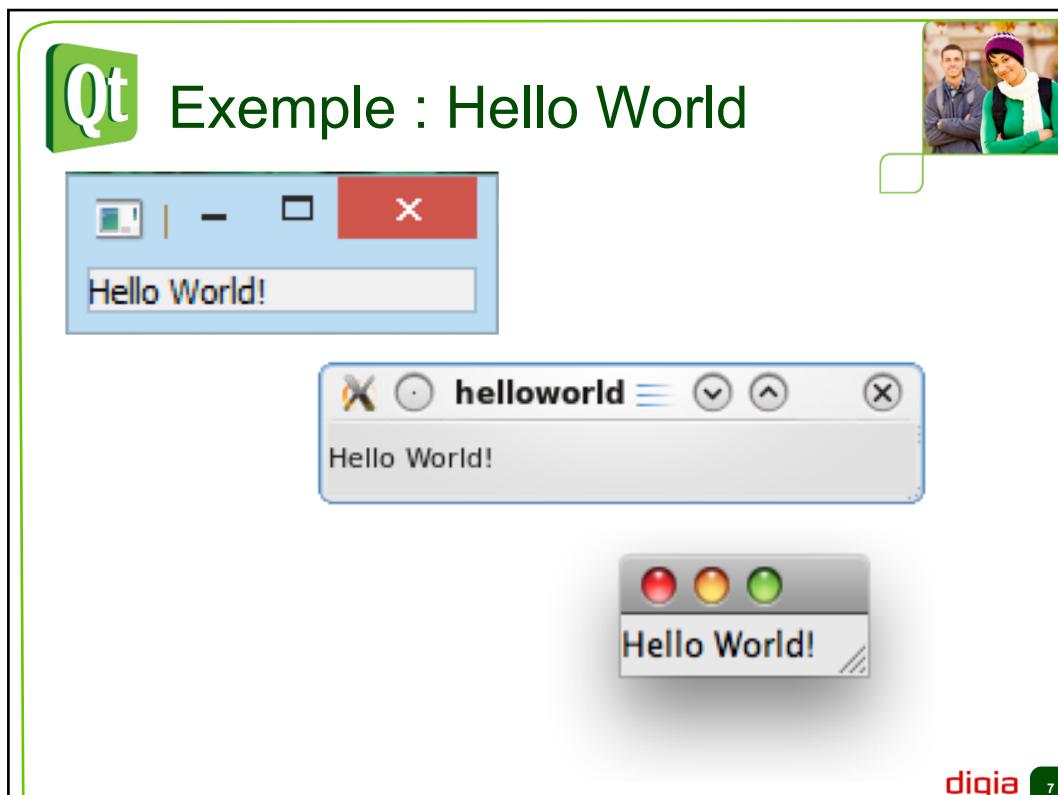
# Qt La communauté Qt

- <http://www.qt.io>
- <http://www.qt.io/developers/>
- <http://qt.developpez.com/>
- <http://www.qtfr.org/>
- etc.





**digia** 6



The image shows a code editor window with the following C++ code:

```
#include <QApplication>
#include <QLabel>

int main( int argc, char **argv )
{
    QApplication app( argc, argv );
    QLabel l( "Hello World!" );
    l.show();
    return app.exec();
}
```

A callout bubble points from the text "Simplicité du code" to the code editor area.



## Hello World : le code

```
#include <QApplication>
#include <QLabel>

int main( int argc, char **argv )
{
    QApplication app( argc, argv );
    QLabel l( "Hello World!" );
    l.show();
    return app.exec();
}
```

digia 9



## Hello World : le code

```
#include <QApplication>
#include <QLabel>
int main( int argc, char **argv )
{
    QApplication app( argc, argv );
    QLabel l( "Hello World!" );
    l.show();
    return app.exec();
}
```

Instanciation  
obligatoire

Autre accès :  
pointeur qApp

digia 10



## Hello World : le code

```
#include <QApplication>
#include <QLabel>

int main( int argc, char **argv )
{
    Widget sans parent ⇒ affiché dans une fenêtre
    QApplication app( argc, argv );
    QLabel l( "Hello World!" );
    l.show();
    return app.exec();
}
```

digia 11



## Hello World : le code

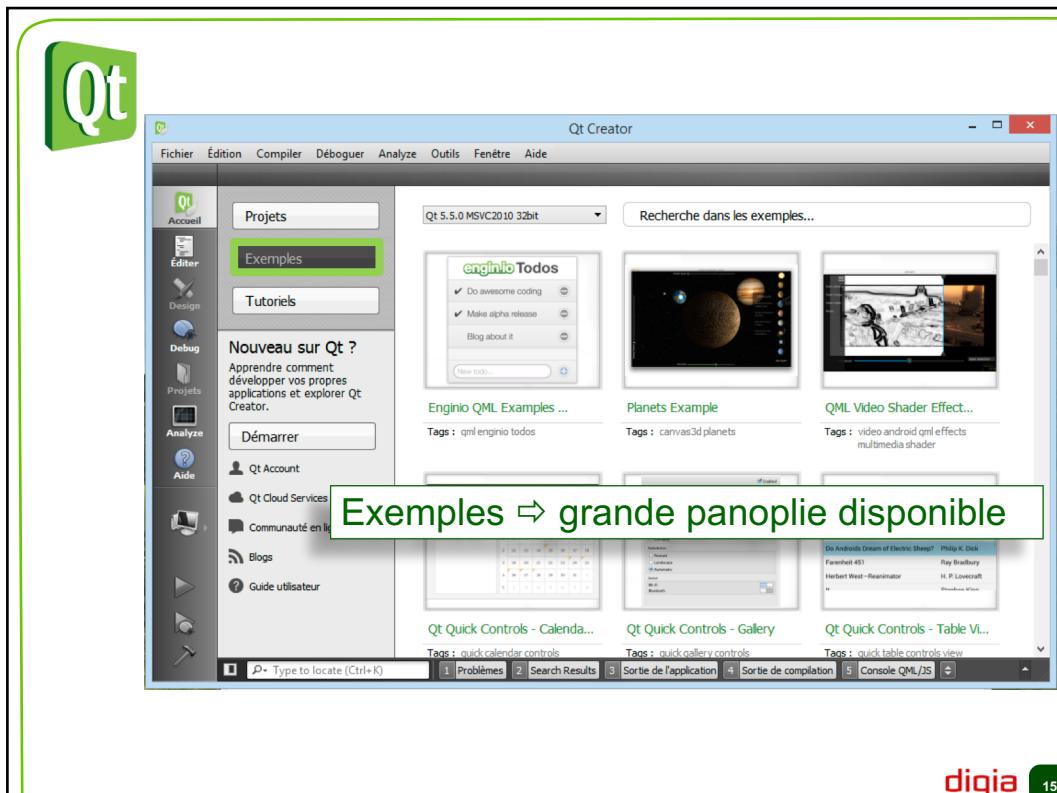
```
#include <QApplication>
#include <QLabel>

int main()
{
    QApplication app;
    QLabel l( "Hello World!" );
    l.show();
    return app.exec();
}
```

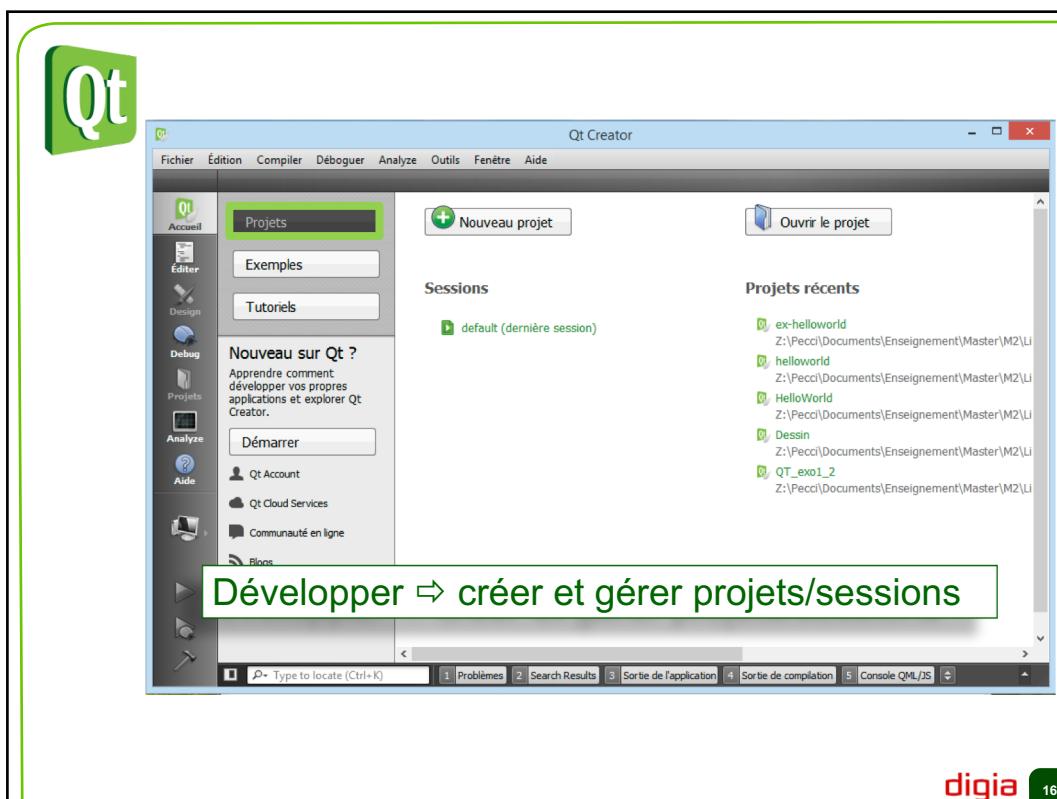
• Lance la boucle d'attente des événements (interaction utilisateur, réseau ou chronomètre)  
 • Arrêt quand dernière fenêtre fermée

digia 12

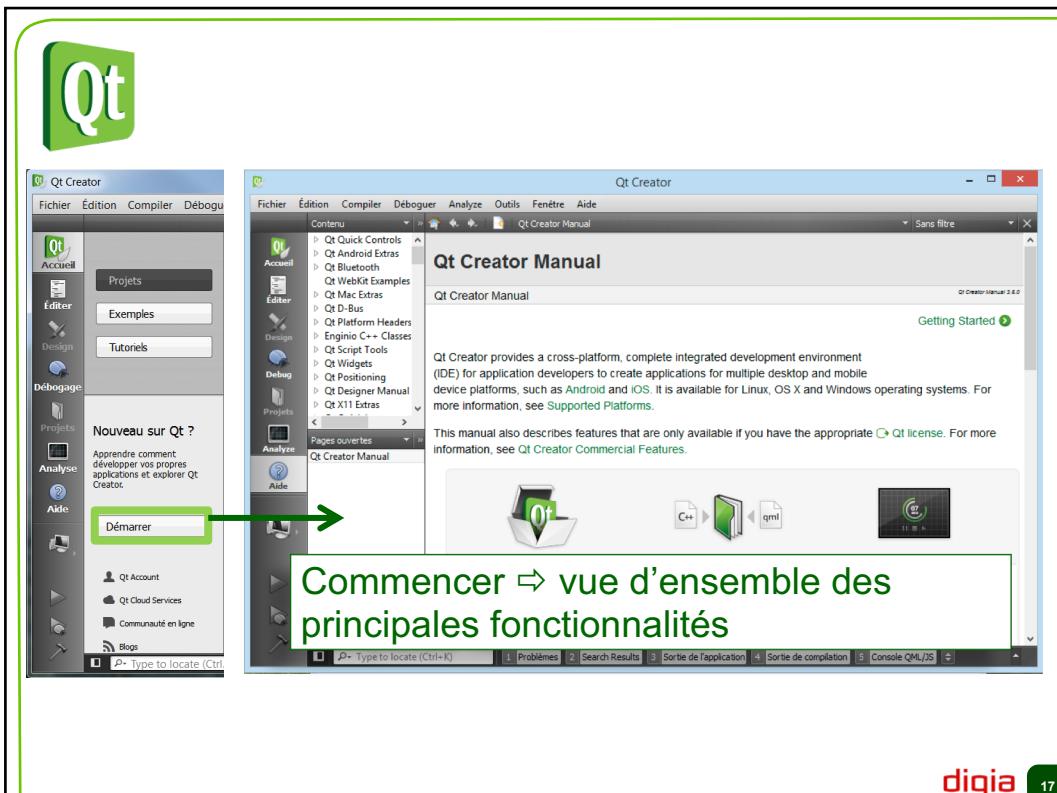




digia 15

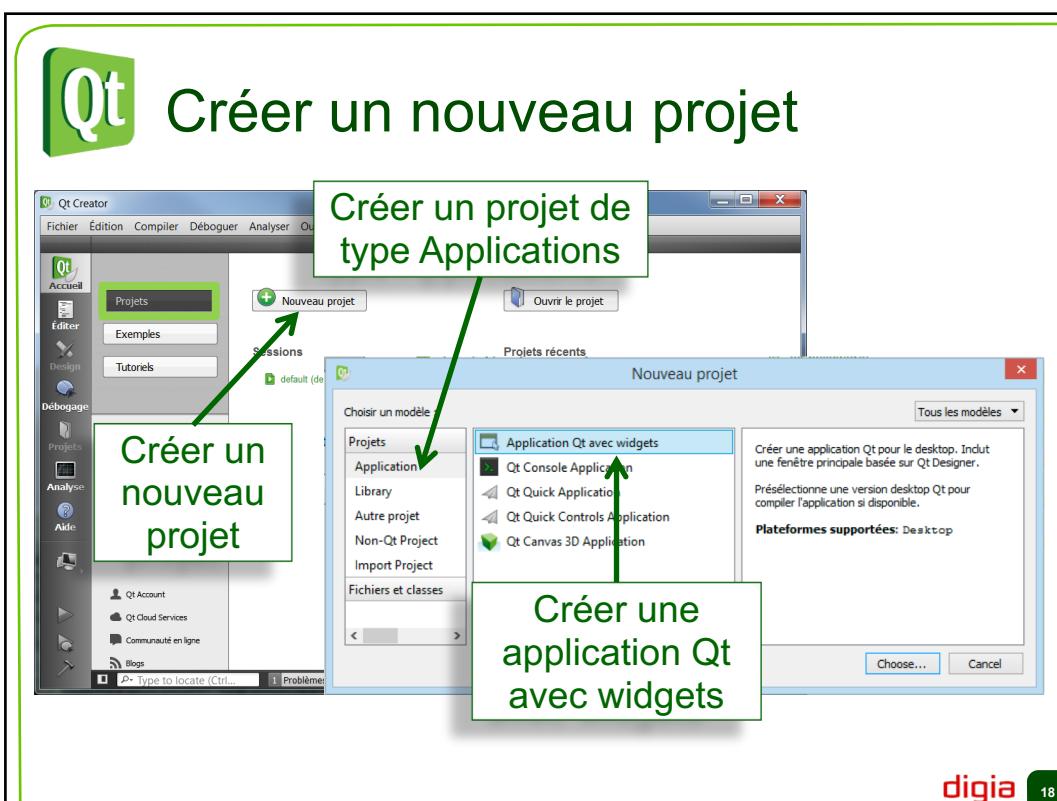


digia 16



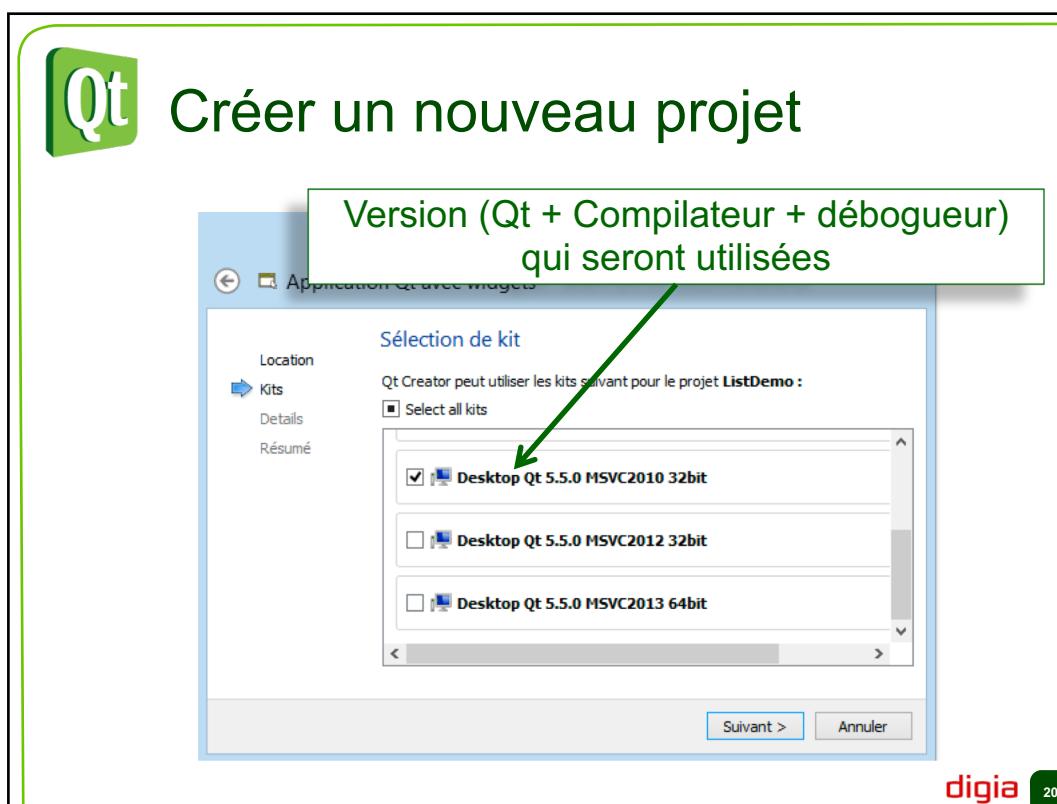
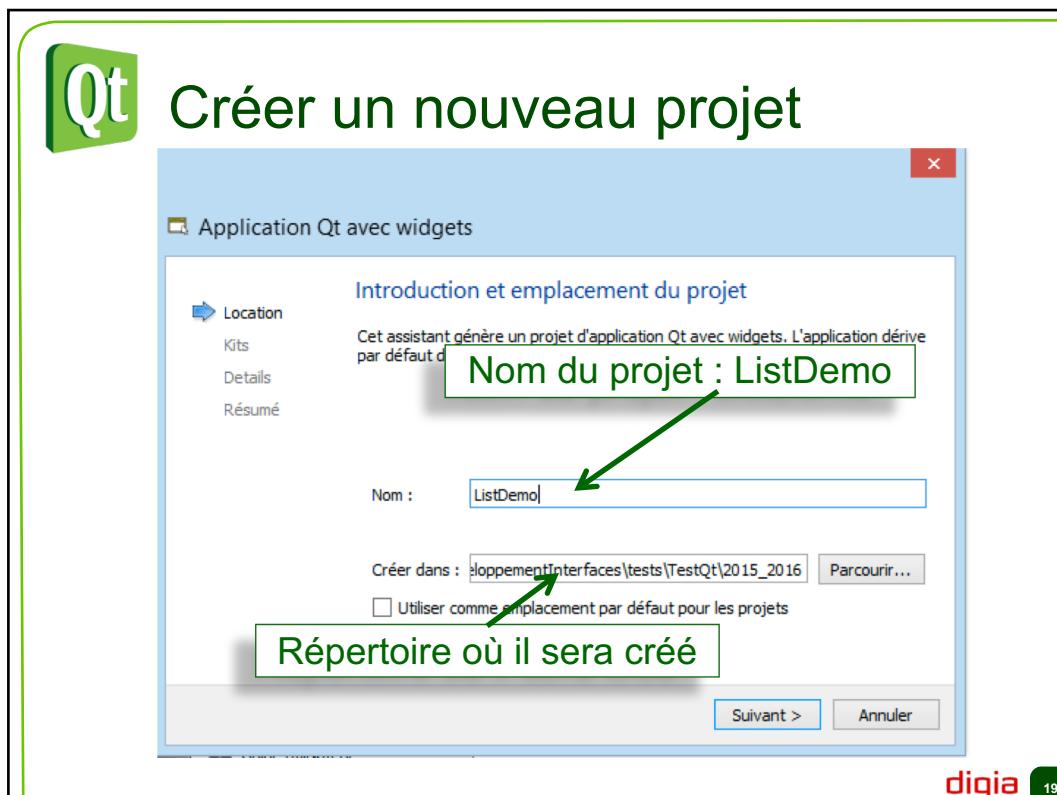
digia

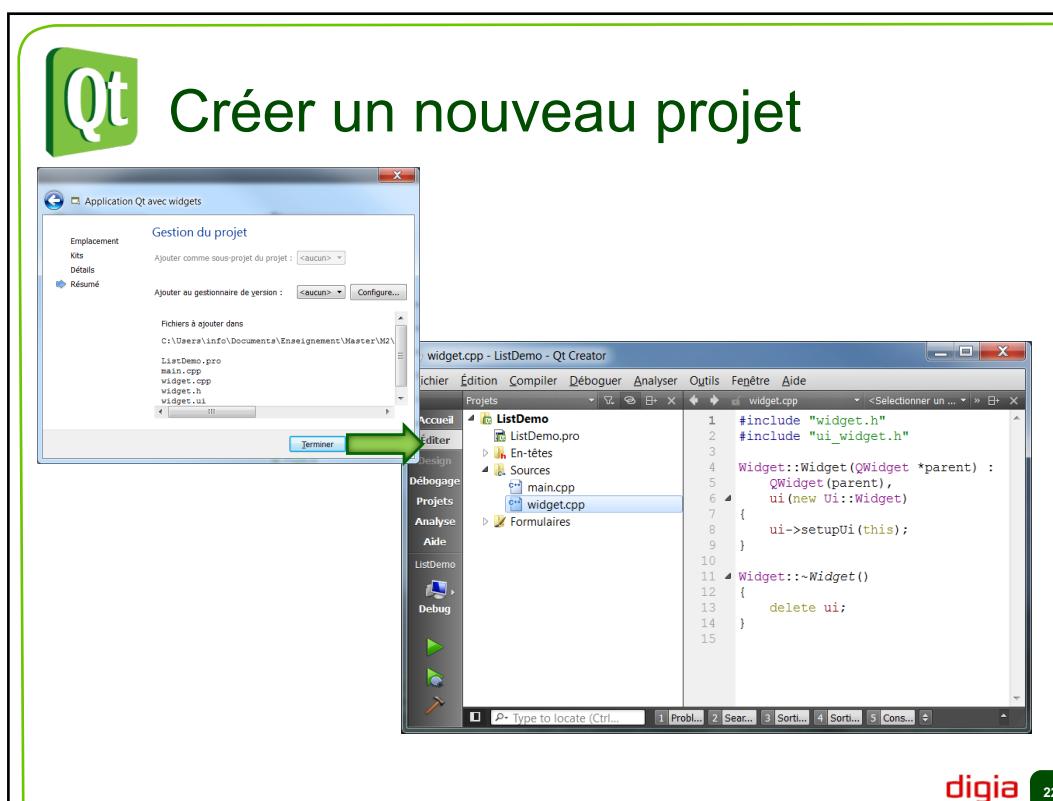
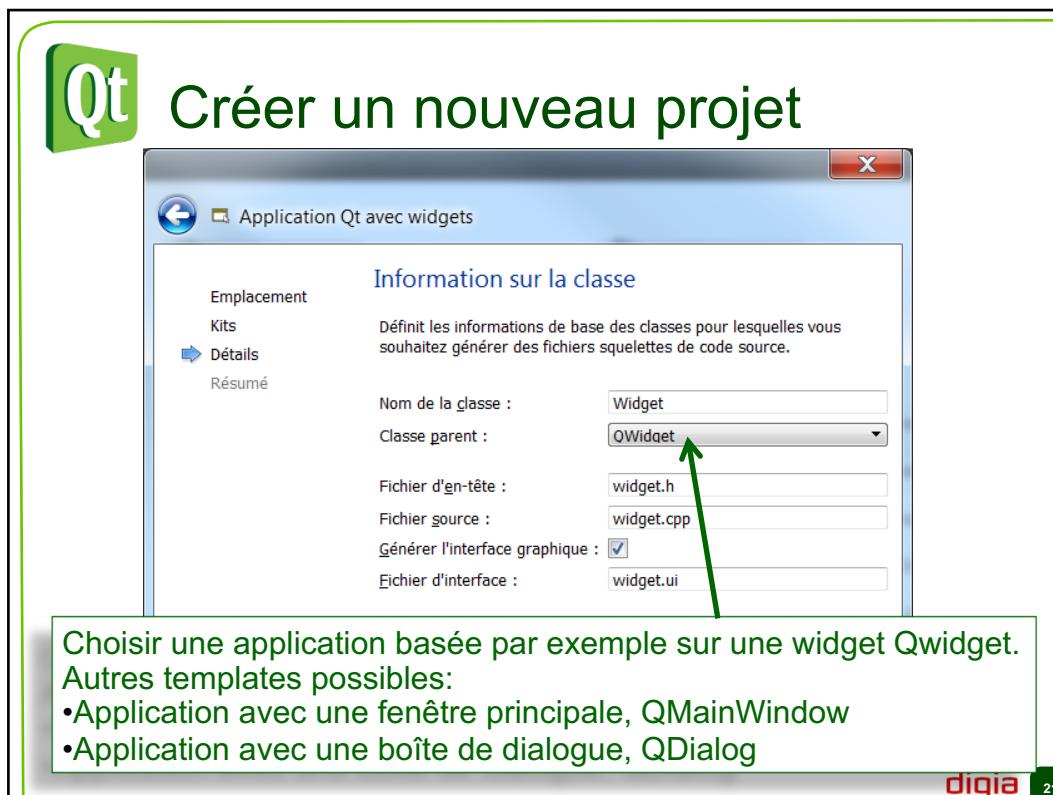
17

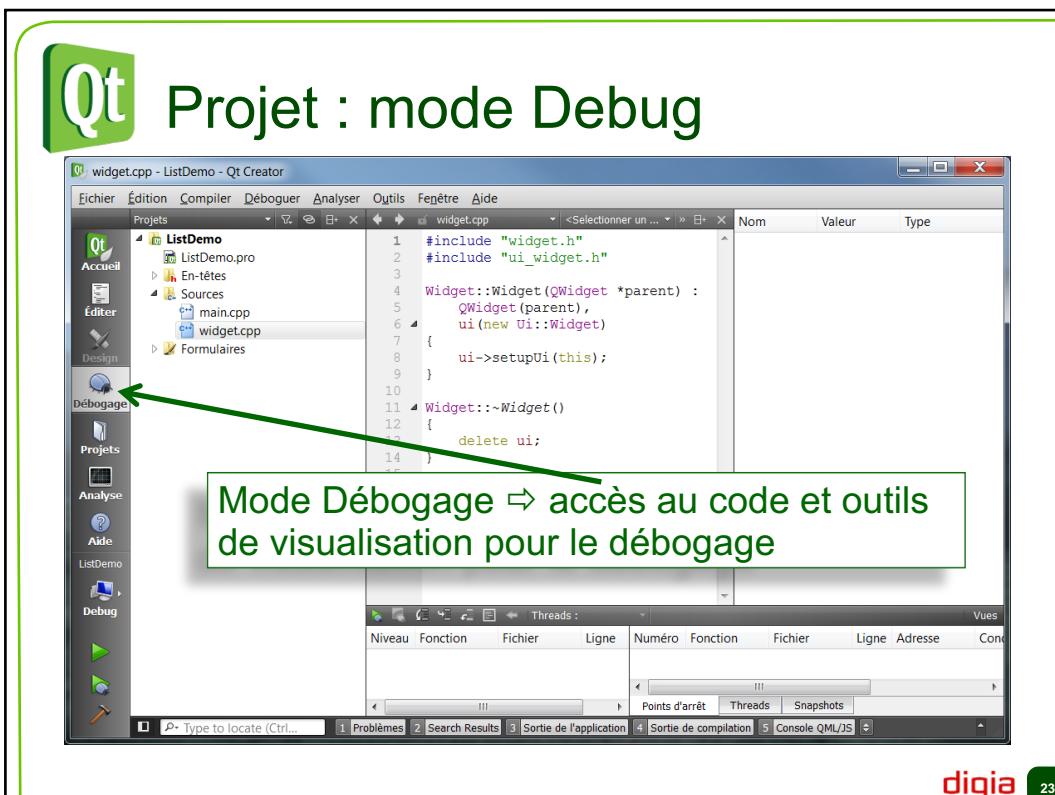


digia

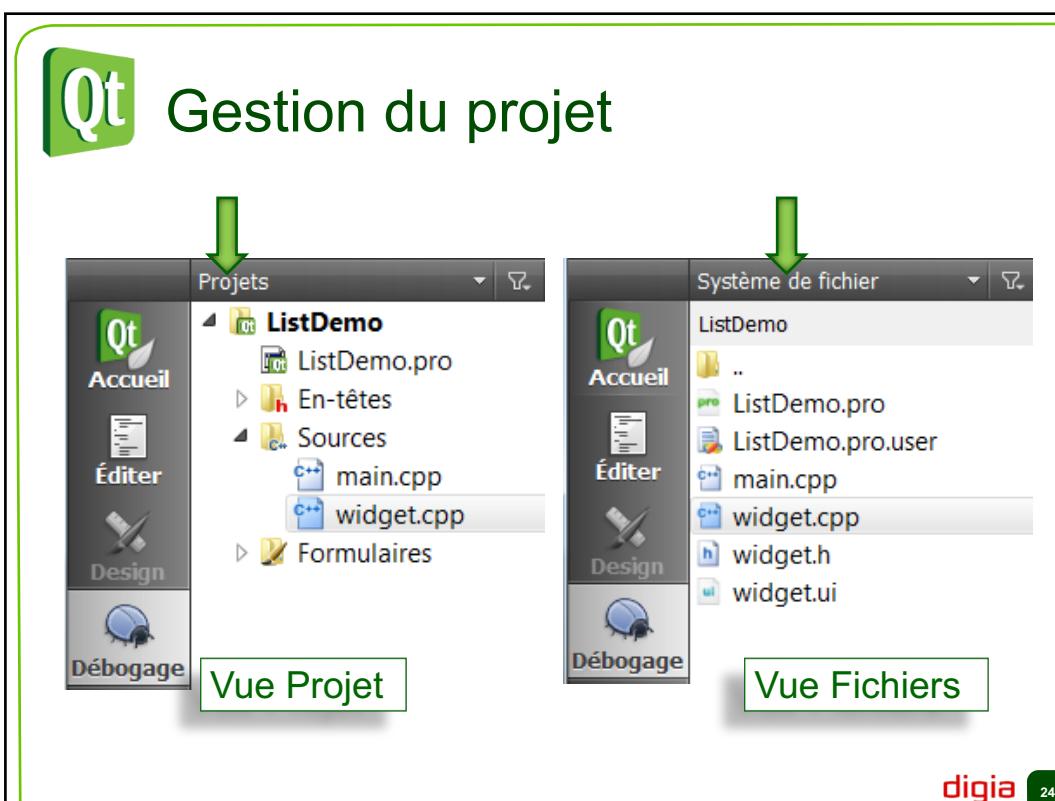
18



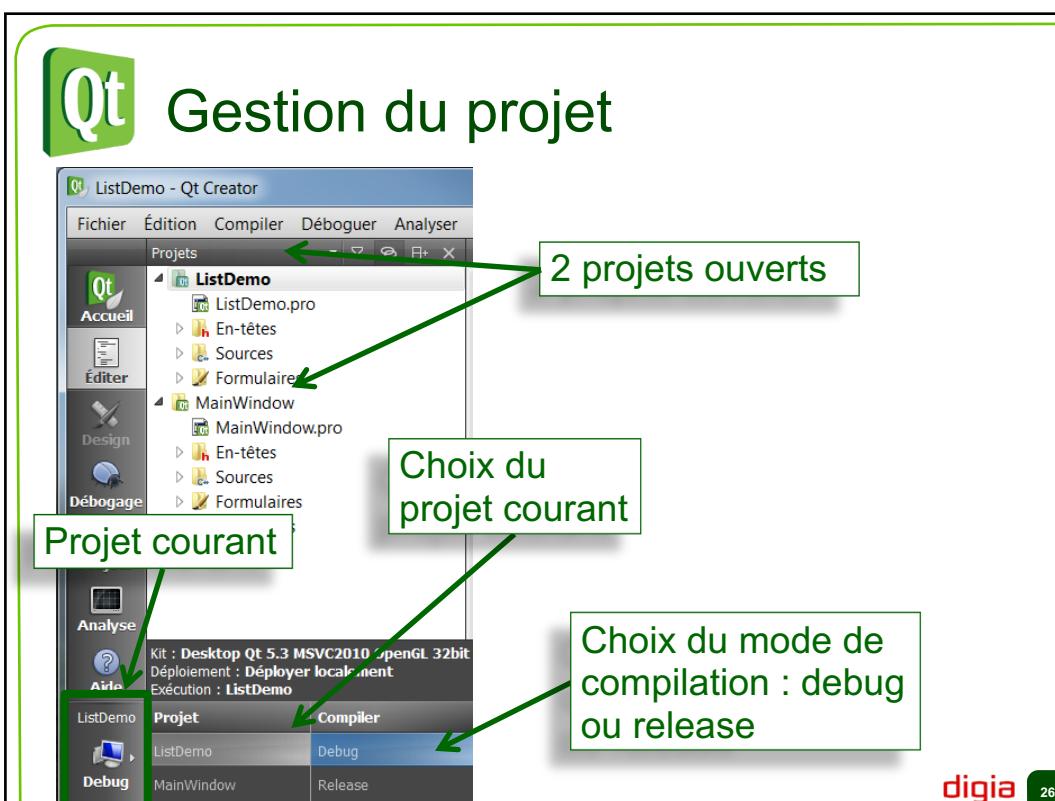
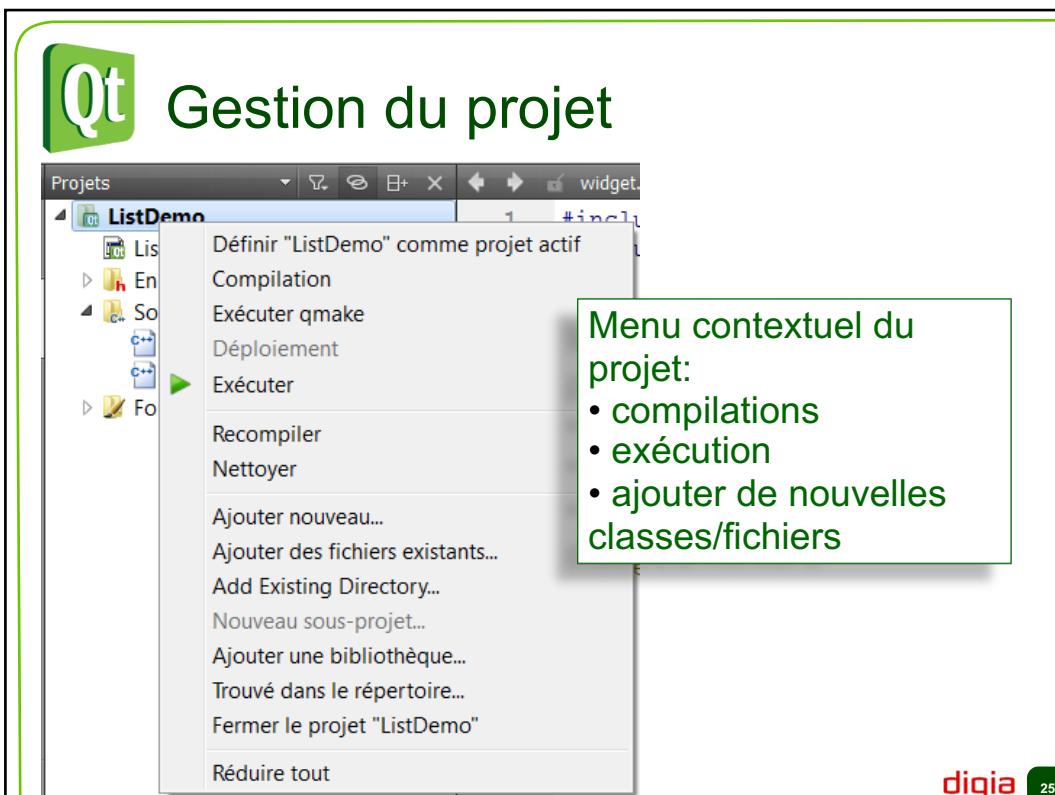




digia 23



digia 24



# Qt Utiliser un kit de développement

- Vérifier si détecté
  - Projets | Gérer les kits...
  - Compiler & Exécuter → (3)
  - Choisir le kit autodétecté → (4)

digia 27

# Qt Si kit pas détecté

- Définir la version de Qt

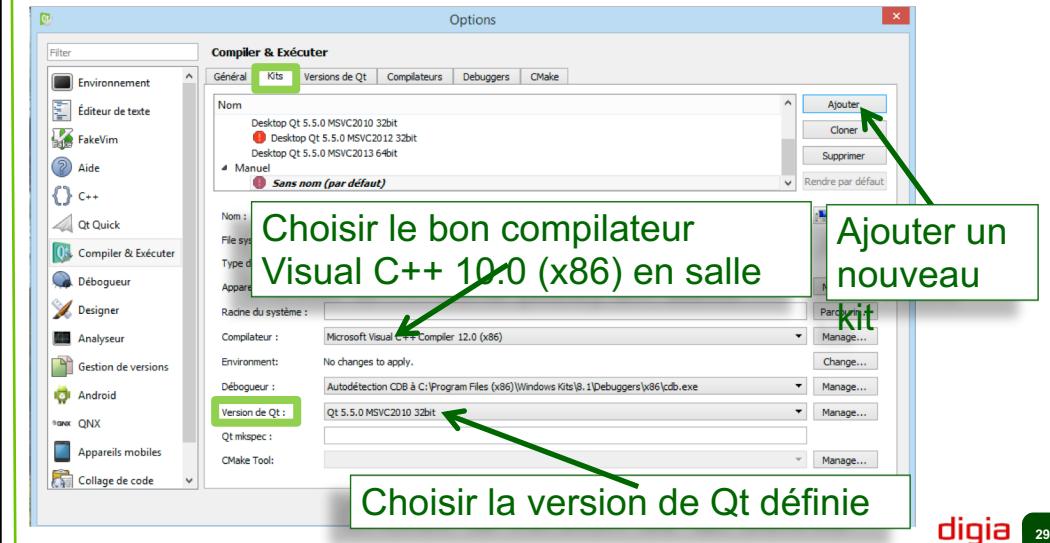
En ajouter un si pas auto-détecté  
A priori dans C:\Qt5.5\msvc2010\bin\qmake.exe

digia 28



## Si kit pas détecté

- Créer le kit à partir de la version de Qt définie

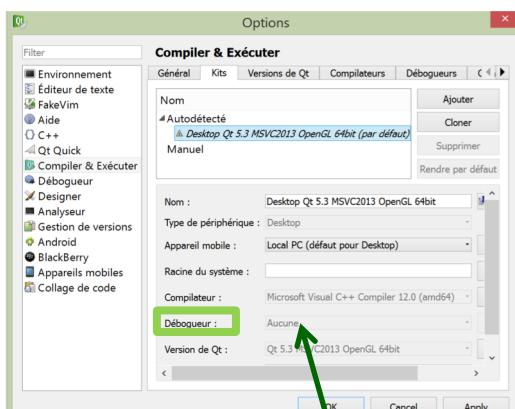


digia 29



## Utiliser débogueur Windows

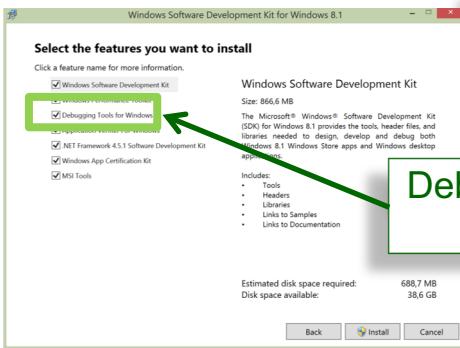
- Vérifier s'il est détecté



digia 30

## Qt Utiliser débogueur Windows

- Si pas détecté : fichier cdb.exe doit être installé
  - Quitter QtCreator
  - Télécharger le kit de développement logiciel
    - Windows SDK pour votre version de Windows



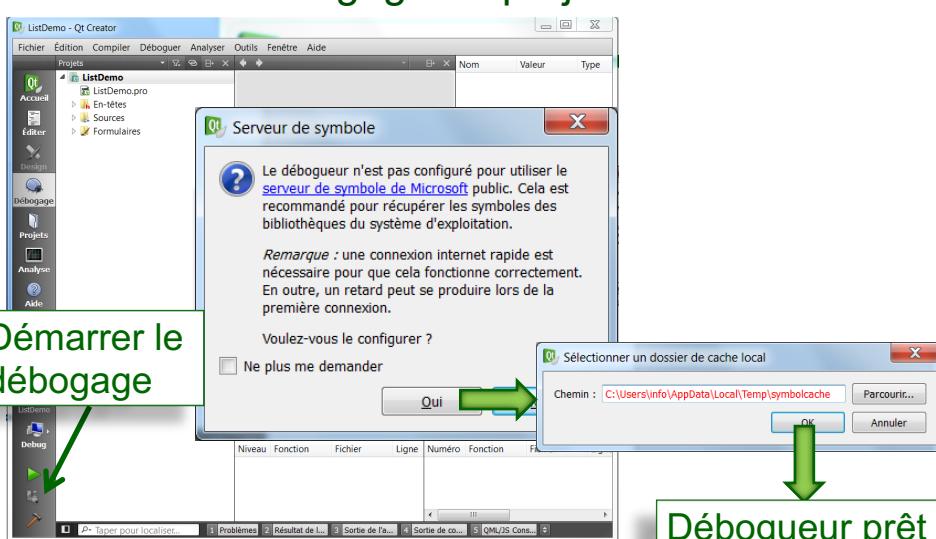
<http://www.microsoft.com/fr-fr>

Debugging Tools for Windows doit être coché

digia 31

## Qt Configurer le débogueur

- Démarrer le débogage du projet



Démarrer le débogage

Le débogueur n'est pas configuré pour utiliser le serveur de symbole de Microsoft public. Cela est recommandé pour récupérer les symboles des bibliothèques du système d'exploitation.

Voulez-vous le configurer ?

Ne plus me demander

Qui

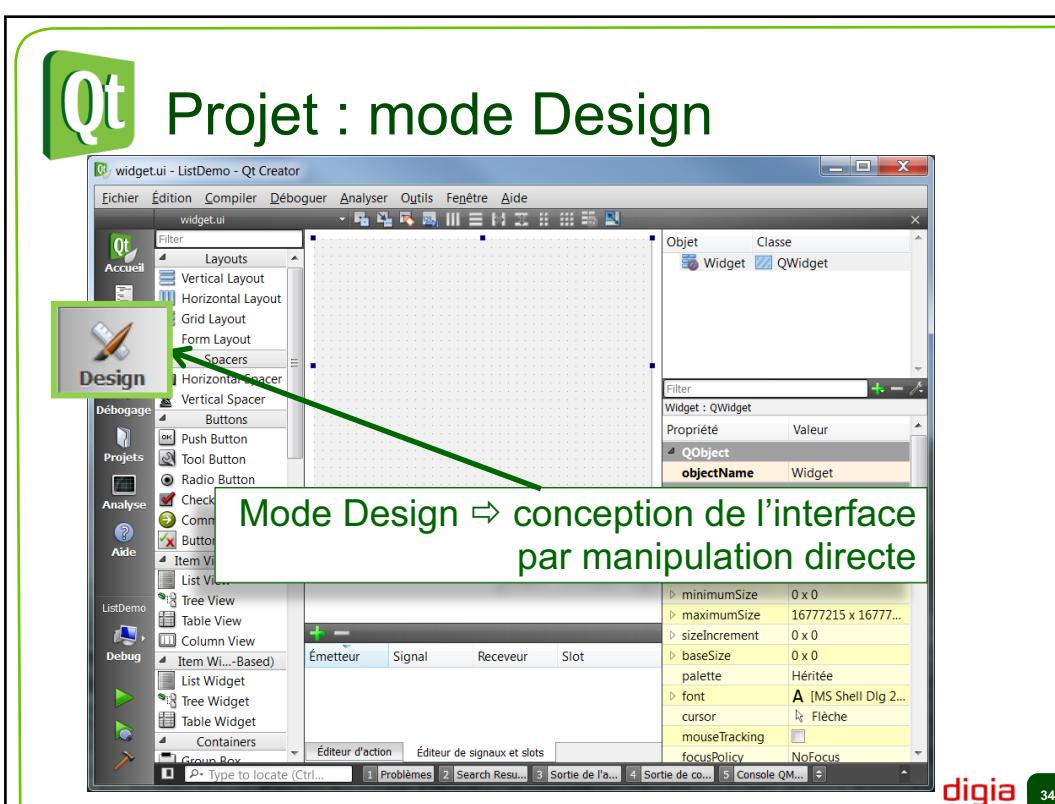
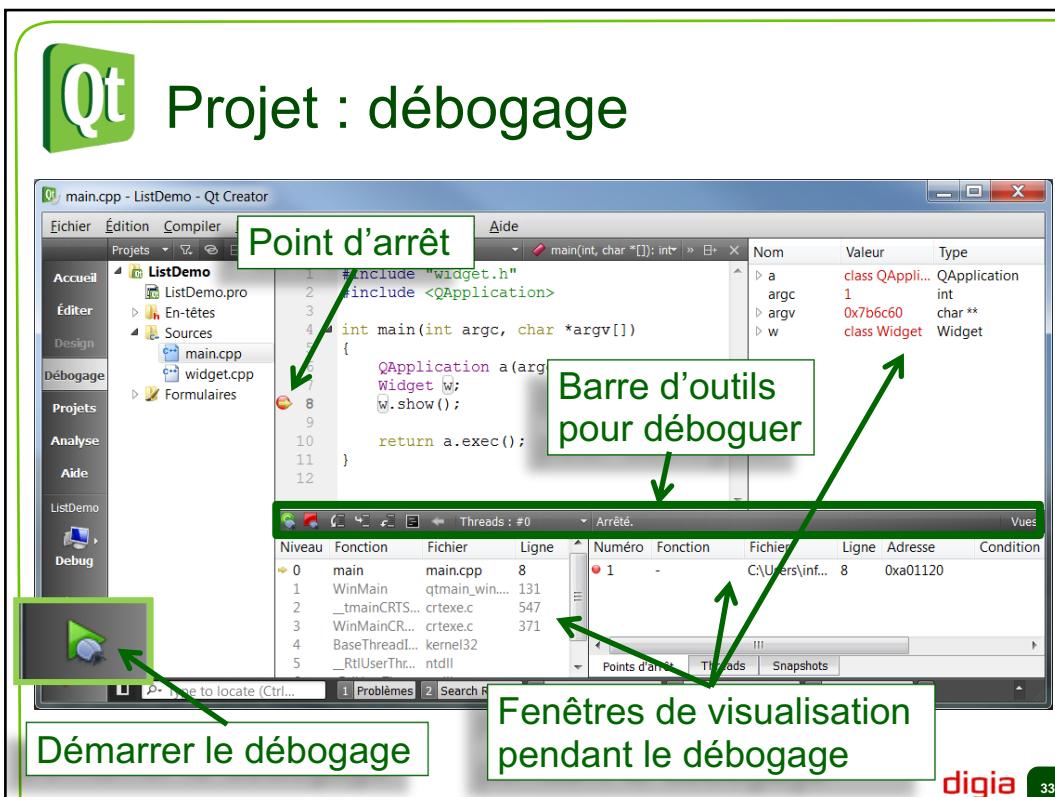
Selectionner un dossier de cache local

Chemin : C:\Users\info\AppData\Local\Temp\symbolcache

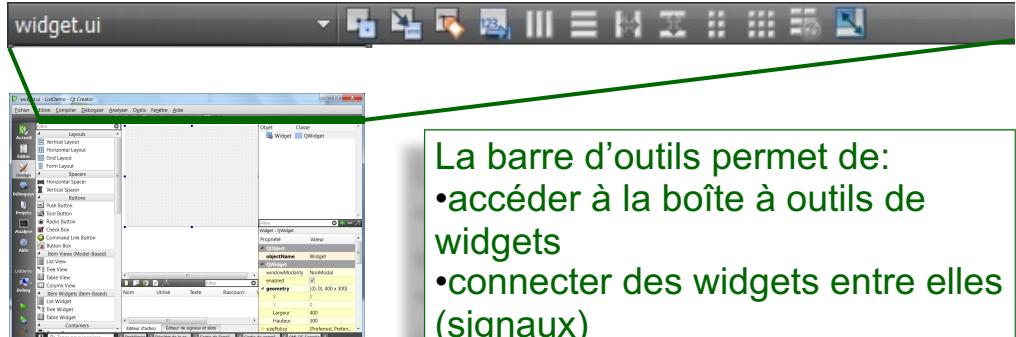
OK Annuler

Débogueur prêt !

32



# Qt Projet : mode Design

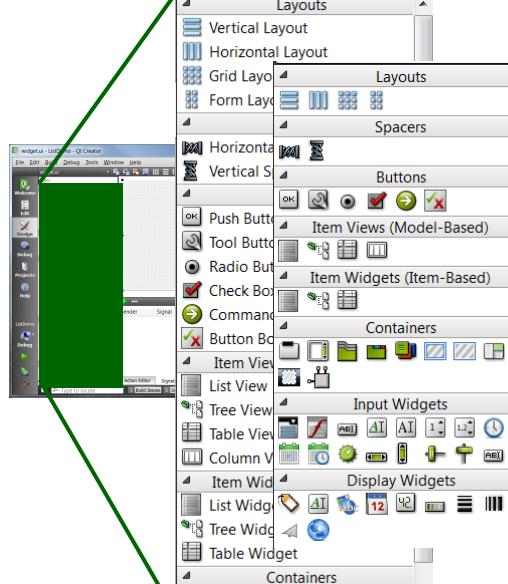


La barre d'outils permet de:

- accéder à la boîte à outils de widgets
- connecter des widgets entre elles (signaux)
- connecter des libellés amies (« buddies ») à d'autres widgets
- définir l'ordre de tabulation
- placer/aligner des widgets

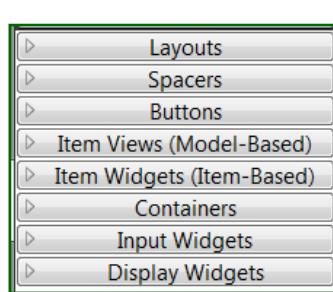
**digia** 35

# Qt Projet : mode Design



La boîte des widgets

- liste les widgets disponibles par catégorie
- 3 vues possibles (menu contextuel)



**digia** 36

# Qt Projet : mode Design

**La vue arborescente  
⇒ hiérarchie des widgets sur le formulaire**

Objet	Classe
Widget	QWidget
listWidget	QListWidget
pushButton	QPushButton
pushButton_2	QPushButton
pushButton_3	QPushButton

**digia 37**

# Qt Projet : mode Design

**La fenêtre des propriétés ⇒ propriété du/des widgets sélectionnés sur le formulaire**

Propriété	Valeur
windowTitle	Widget
windowIcon	
windowOpacity	1.000000
toolTip	
statusTip	
whatsThis	
accessibleName	
accessibleDescri...	

**digia 38**

**Qt Projet : mode Design**

3 QPushButtons  
1 QListWidget

objectName = addButton  
text = Add...

objectName = deleteButton  
text = Delete

objectName = clearButton  
text = Clear...

en utilisant...  
 • la fenêtre des Propriétés  
 • Le menu contextuel du QPushButton

selectionMode= MultiSelection

**dicia** 39

**Qt Prévisualisation de l'interface**

- Vue rapide de l'interface ⇒ pas de compilation
  - pas de réactions personnalisées sur les widgets
  - Outils|Form Editor|Prévisualisation...

**dicia** 40

## Qt Prévisualisation de l'interface

- Prévisualisation sous différents styles
  - Outils|Form Editor|Aperçus dans

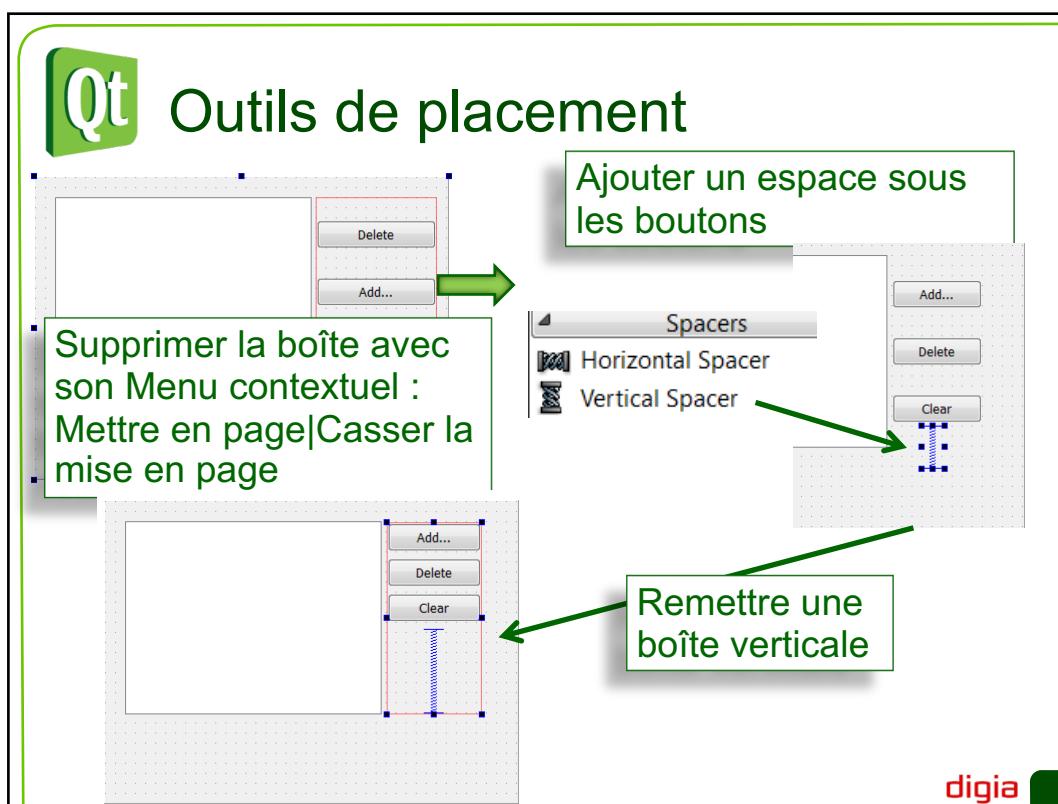
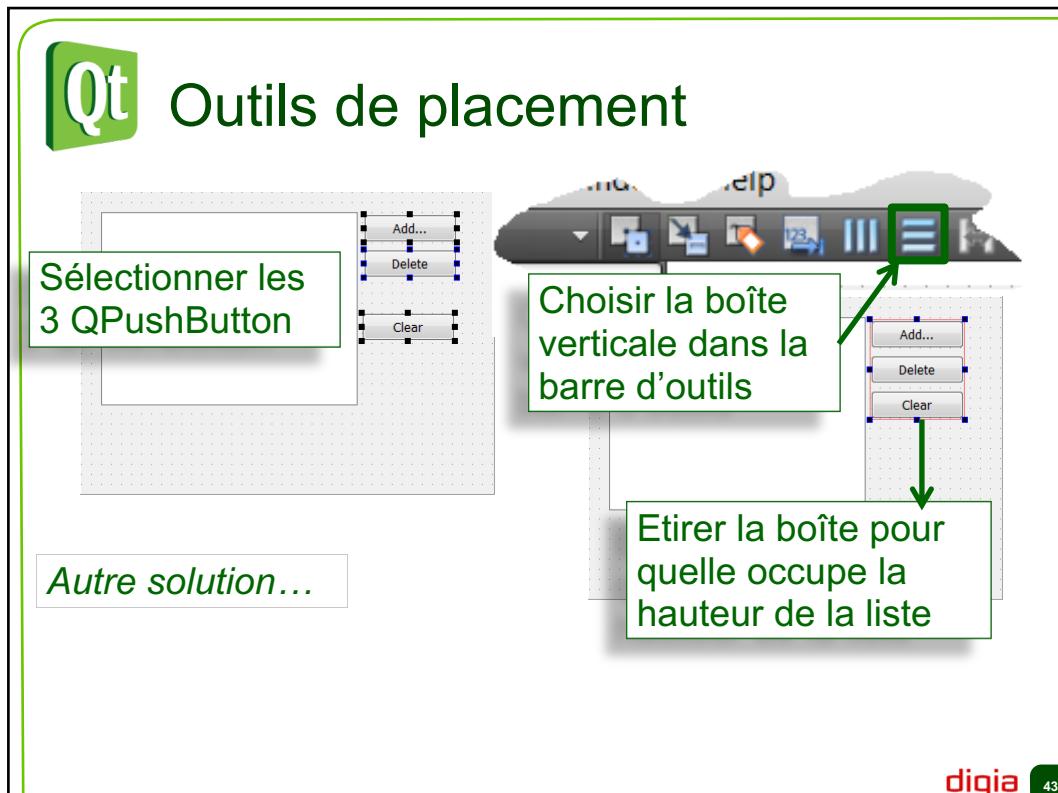
The diagram illustrates the Qt Form Editor's preview feature. It shows three windows side-by-side. On the left is the standard design view of a form with three buttons ('Add...', 'Delete', 'Clear') and a central area. Two green arrows point from this view to two preview windows on the right. The top-right preview window is titled 'Widget - [prévisualisation]' and shows the same form with the word 'Windows' in the central area. The bottom-right preview window is also titled 'Widget - [prévisualisation]' and shows the same form with the words 'Windows XP' in the central area. A small ellipsis (...) is positioned between the two preview windows.

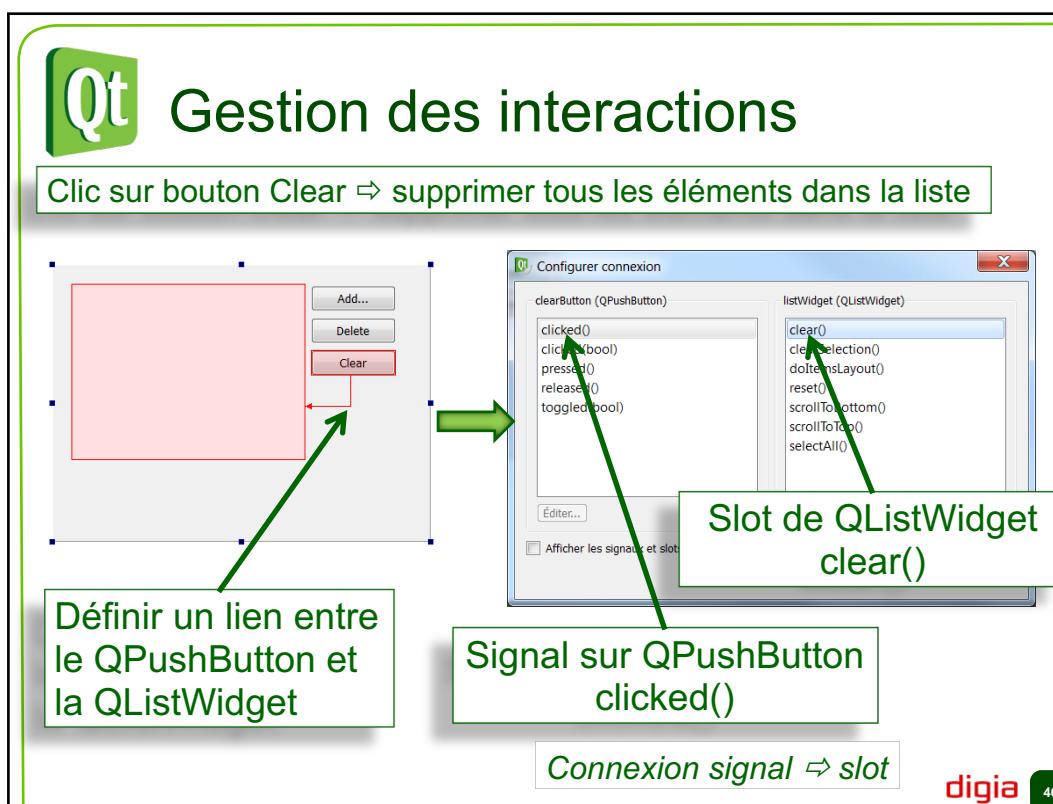
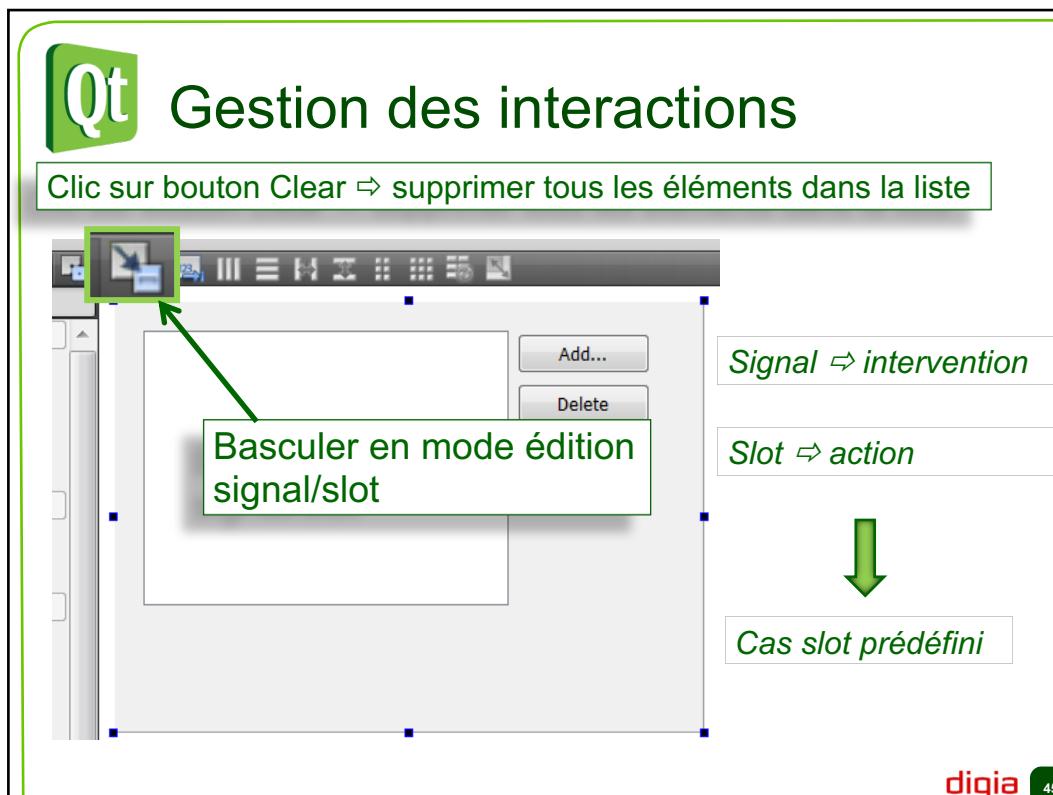
**digia** 41

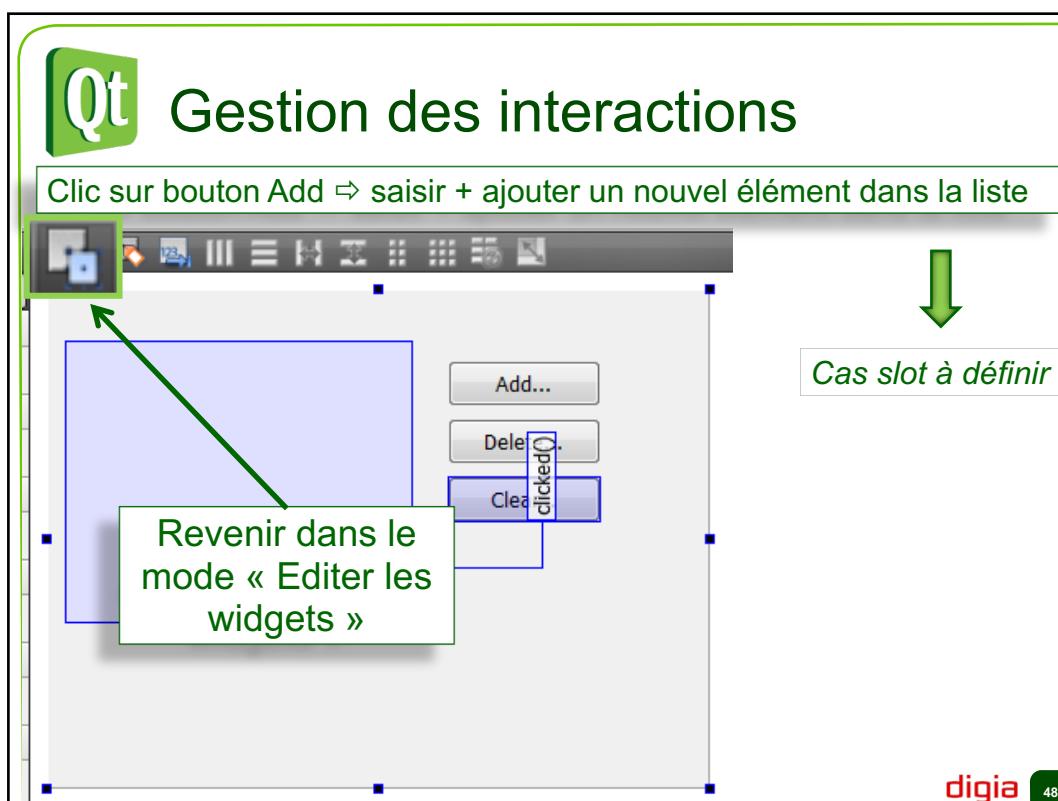
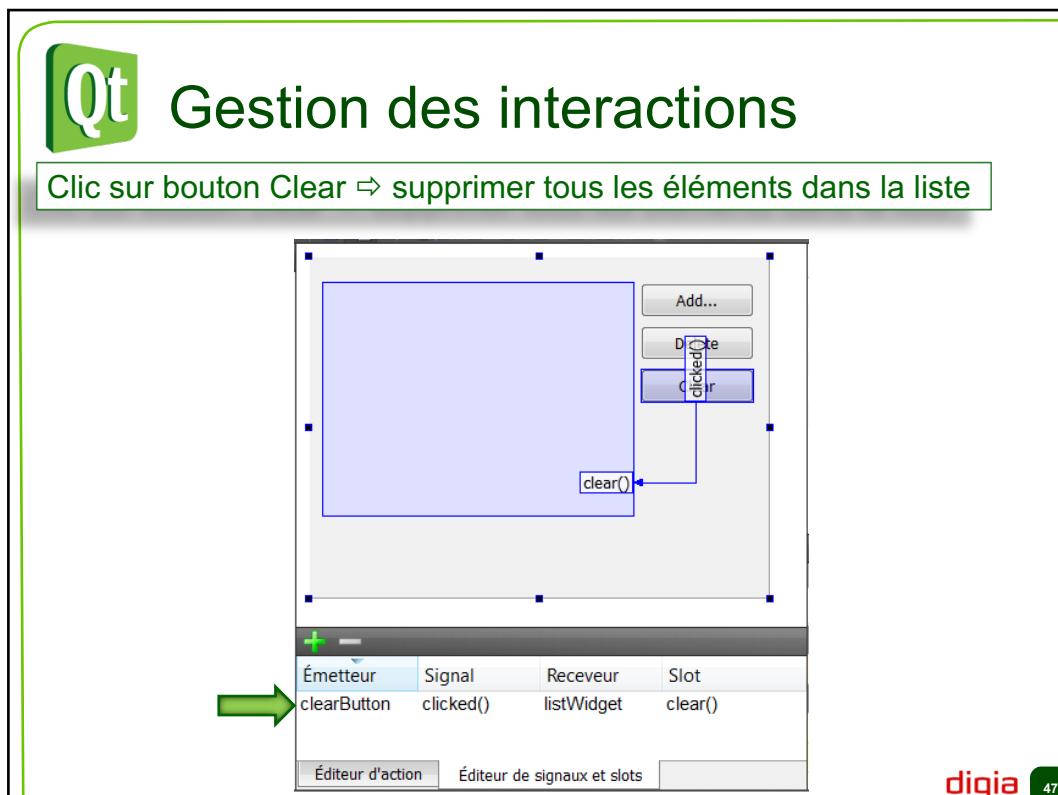
## Qt Outils de placement

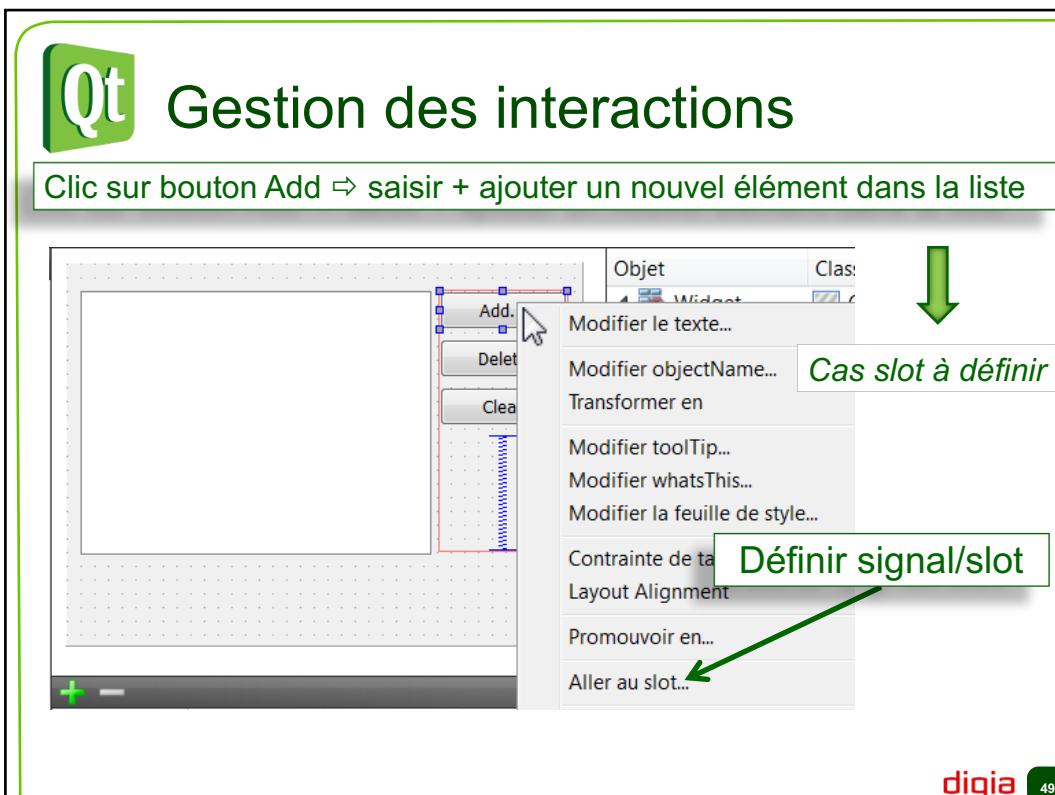
The diagram shows the Qt Form Editor interface with a focus on layout tools. On the left, the 'Layouts' section of the tool palette is open, showing options like 'Vertical Layout', 'Horizontal Layout', 'Grid Layout', and 'Form Layout'. A green callout box labeled '1. Ajouter une boîte verticale' has an arrow pointing to the 'Vertical Layout' option. In the center, a vertical red dashed line indicates where a new layout will be placed. On the right, the main workspace shows a form with three buttons ('Add...', 'Delete', 'Clear') at the top. A green callout box labeled '2. Déplacer les QPushButton' has an arrow pointing to the buttons. Another green callout box labeled '3. Placer chaque QPushButton dans la boîte' has an arrow pointing to the workspace below the buttons, where they are intended to be placed within the new layout.

**digia** 42

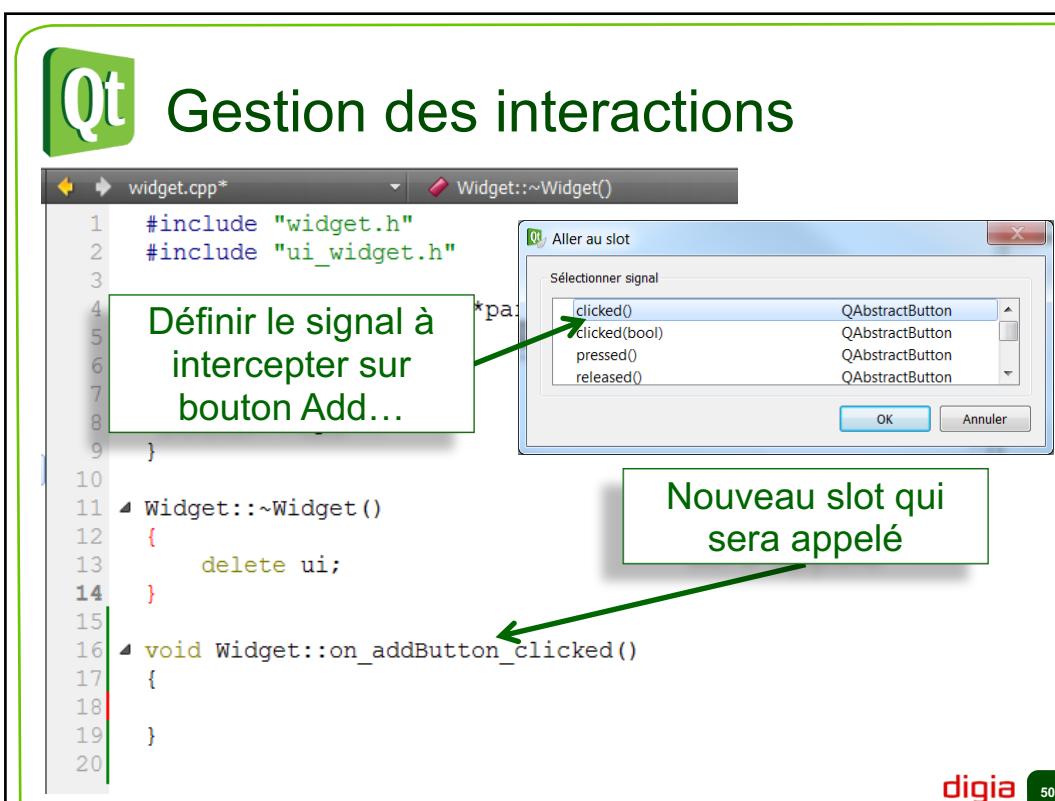








digia 49



digia 50

## Qt L'éditeur de code

- F4 : navigation fichiers entête/implémentation
- F2 : idem F4 avec localisation sur une méthode
- Complétion de code

The screenshot shows a code editor window in Qt Creator. The cursor is at the end of the line '#include "QInput"', and a completion tooltip is displayed. The tooltip contains the following list of suggestions:

```

1 #include "widget.h"
2 #include "ui_widget.h"
3 #include "QIn"
4     • QIncompatible
5     • QWidget
6     • ui_(ne)
7     • ui->
8     { ui->
9     }
10    }
11    }
12    Widget::Widget()
13    {
14        delete ui;
15    }

```

Below the list, the suggestion 'QInputDialog' is highlighted with a blue rectangle. Two green arrows point from the text above towards this highlighted suggestion.

digia 51

## Qt L'éditeur de code

- Ctrl + K : recherche et localisation
  - Classes, méthodes, fichiers, ligne, etc.
  - Exemple : localiser le slot à définir

The screenshot shows the Qt Creator interface with the keyboard shortcut 'Ctrl + K' highlighted in a green box. Below it, a search results panel is open, showing a list of items starting with 'm on'. One item in the list is highlighted with a green box: 'on\_addButton\_clicked()'. To the right of the search results, a tooltip provides instructions: 'Ctrl + K « m on » recherche une méthode commençant par « on »'. The status bar at the bottom of the screen also displays the search term 'm on'.

digia 52

## Qt L'éditeur de code

```
void Widget::on_addButton_clicked()
{
    QString newText = QInputDialog::getText(this, "Enter text", "Text:");
    if( !newText.isEmpty() )
        ui->listWidget->addItem(newText);
}
```

- Compiler, puis exécuter

Aide à la saisie des paramètres de la méthode

Exécution (+ compilation si nécessaire)

Exécution en mode débogage

Compilation

Console Q... 6 Messages g... Compilation

## Qt Exécution

Add...

Enter text

Toto

Toto  
Titi  
Tata

Clear

Delete ↳ Ø

## Qt Gestion du bouton delete

Clic sur bouton Delete ⇒ supprimer les éléments sélectionnés

**Cas slot à définir**

```
void Widget::on_deleteButton_clicked()
{
    foreach (QListWidgetItem *item, ui->listWidget->selectedItems())
        delete item;
}
```

The first screenshot shows a window titled "Widget" containing a QListWidget with three items: "Toto", "Tata", and "titi". Below the list is a horizontal toolbar with "Add...", "Delete", and "Clear" buttons. The "titi" item is currently selected, indicated by a blue selection bar around its text. A large green arrow points from the "Delete" button on the toolbar to the "titi" item. The second screenshot shows the same window after the deletion, with only "Tata" remaining in the list.

**digia 55**

## Qt Gestion du bouton delete

Bouton Delete ⇒ accessible si au moins un élément sélectionné dans la liste

**Cas slot défini par programme**

```
class Widget : public QWidget
{
    Q_OBJECT

public:
    explicit Widget(QWidget *parent = 0);
    ~Widget();

private:
    Ui::Widget *ui;

private slots:
    void on_deleteButton_clicked();
    void on_addButton_clicked();
    void updateDeleteEnabled();
```

**Ajout d'un nouveau slot**

The first screenshot shows a window titled "Widget" containing a QListWidget with three items: "Toto", "Tata", and "titi". A green arrow points from the "Delete" button on the toolbar to the "titi" item, which is highlighted with a blue selection bar. The second screenshot shows the same window after the deletion, with only "Tata" remaining in the list.

**digia 56**

# Qt Gestion du bouton delete

Définition de l'action du slot ⇒ définir l'accessibilité au bouton Delete

```
void Widget::updateDeleteEnabled()
{
    ui->deleteButton->setEnabled(ui->listWidget->selectedItems().count() != 0)
}
```

Compilation ⇒ erreur

Problèmes

C2143: erreur de syntaxe: absence de ';' avant ')'

widget.cpp 33

Sortie de compilation

widget.cpp  
...\\ListDemo\\widget.cpp(33) : error C2143: erreur de syntaxe: absence de ';' avant ')'  
jom: C:\\Users\\info\\Documents\\Enseignement\\Master\\M2\\LibrairieDeveloppementInterfaces\\CoursQt\\Demos\\2014-2015\\build-ListDemo-Desktop\_Qt\_5\_3\_MSVC2010\_OpenGL\_32bit-Debug\\Makefile.Debug  
[debug\\widget.obj] Error 2

# Qt Gestion du bouton delete

Widget::Widget(QWidget \*parent) :

```
    QWidget(parent),
    ui(new Ui::Widget)
```

Connexion signal/slot dans le constructeur

```
{
```

```
    ui->setupUi(this);
    connect(ui->listWidget->selectionModel(),
            SIGNAL(selectionChanged(QItemSelection,QItemSelection)),
            this, SLOT(updateDeleteEnabled()));
```

} Problème : le bouton n'est pas inaccessible au départ  
⇒ appeler le slot après la connexion

```
Widget::Widget(QWidget *parent) :
```

```
    QWidget(parent),
    ui(new Ui::Widget)
```

```
{
```

```
    ui->setupUi(this);
    connect(ui->listWidget->selectionModel(),
            SIGNAL(selectionChanged(QItemSelection,QItemSelection)),
            this, SLOT(updateDeleteEnabled()));
    this->updateDeleteEnabled();
```

