Qt 4 en une heure

Nicolas Arnaud-Cormos

Plan

- Présentation de Qt (Qt 4) :
  - la bibliothèque, les outils fournis
- Nouveautés de Qt 4 :
  - généralités, graphismes, programmation modèle/vue, thread, ...
  - démonstrations

Partie 1

Présentation de Qt (Qt4)

Deux produits, une API

- **Qt** : framework de développement C++, comprenant une bibliothèque et des outils pour le développement d'applications multi-plateformes

- **Qtopia** : plateforme de développement et interface utilisateur pour l'embarqué (Linux)

Double licence

- Qt est disponible sous licence **GPL** :
  - Nombreux tests, transparence du code (Linux, Mac OSX et maintenant Windows),
  - GPL != gratuit : possibilité de se faire payer pour produire du code GPL
- Licence **commercial** :
  - Code source non dévoilé
  - Possibilité de transfert de licence
  - Accès au support de Trolltech !

Le planète Qt

- Communauté (grâce à la licence GPL) :
  - KDE : environnement de bureau Linux
  - Mailing-list, tutoriels, forums :
    - **qtfr.org** : communauté francophone
- Bibliothèques externes (Coin3D, OpenInventor...)
- Partenaires (formation sur site, consulting) :
  - KDAB (depuis plus de 10 ans), Nedyse
- Bibliothèque utilisée et reconnue :
  - Google Earth, Skype, Adobe (Photoshop)

Les plus de Qt

- Développement d'applications multi-plateformes et natives
- Performances et robustesse
- Internationalisation des applications
- Nombreux outils pour le développeurs : *assistant*, *designer*, *linguist*, *qmake*...
- Création simplifiée d'interfaces utilisateurs
- API simple et cohérente : prise en main rapide
- Documentation complète et nombreux exemples

La bibliothèque Qt

- Contient l'essentiel pour la création d'application
- Séparée (depuis Qt 4) en plusieurs modules :
  - Module de base (Qt Core)
  - Module d'interface utilisateur (Qt GUI)
  - Module XML (Qt XML)
  - Module OpenGL (Qt OpenGL)
  - Module SQL (Qt SQL)
  - Module réseau (Qt Network)
  - et d'autres encore...

La bibliothèque Qt

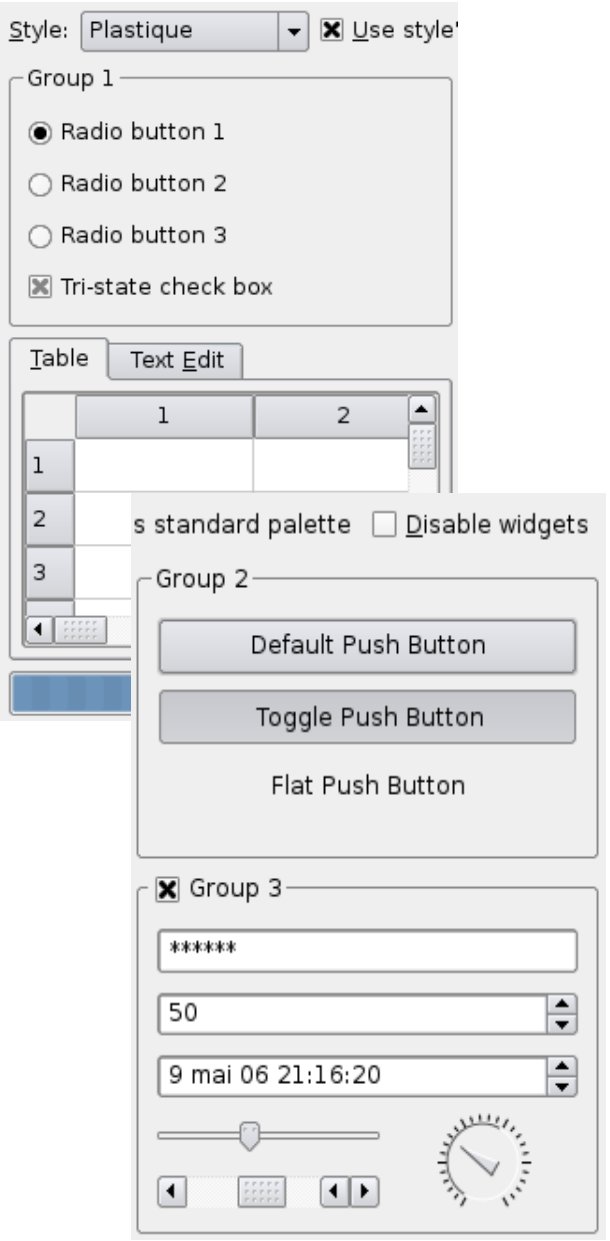
Qt Core

- Classes de base et utilitaires
  - Modèle objet de Qt, gestionnaire d'évènements (système de signaux/slots)
  - Classes d'entrées/sorties : gestion des fichiers, des flux, des répertoires...
  - Gestion des threads
  - Classes de conteneurs : listes, vecteurs, maps...
  - Outils nécessaires à Qt : *qmake*, *moc*, *uic*

La bibliothèque Qt

Qt GUI

- Classes permettant la création d'interface graphique
  - Widgets classiques (boutons, case à cocher, ligne d'édition, menus...)
  - Visualisation de listes/tableaux/arbres avec une architecture MVC
  - Style natif sur toutes les plateformes
  - Accessibilité des interfaces



La bibliothèque Qt

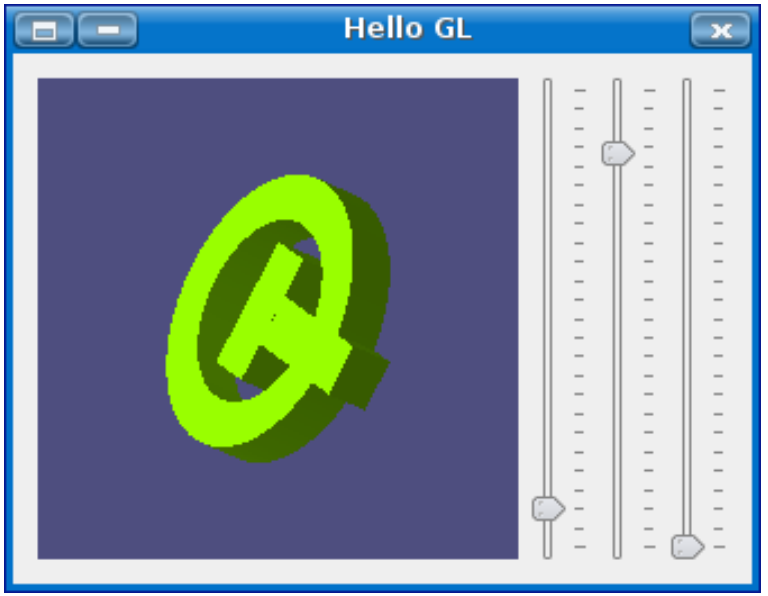
Qt XML

- Parser **SAX2** (Simple API for XML) : lecture par évènements
- Implémentation **DOM** (Document Object Model) : vue hiérarchique d'un document XML

La bibliothèque Qt

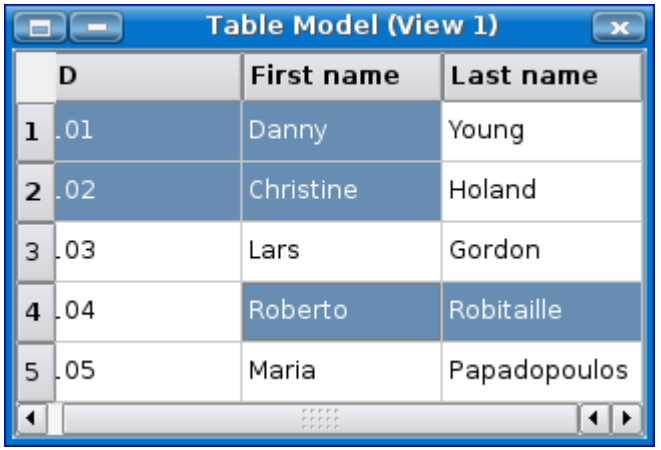
Qt OpenGL

- Intégration d'une fenêtre OpenGL dans les applications Qt
- QGLWidget : s'utilise comme un widget (avec notamment les évènements clavier/souris)



La bibliothèque Qt

Autres modules



- Module SQL : couche d'abstraction pour des bases de données, widgets de visualisation
- Module réseau: fonctionnalité réseau TCP/IP, FTP, HTTP...
- Module de support Qt3 (Qt3 Support) : classes de compatibilités avec Qt3
- Module SVG (Qt SVG) : lecture et affichage de fichiers SVG
- Module de tests (Qt Test) : framework de tests unitaires

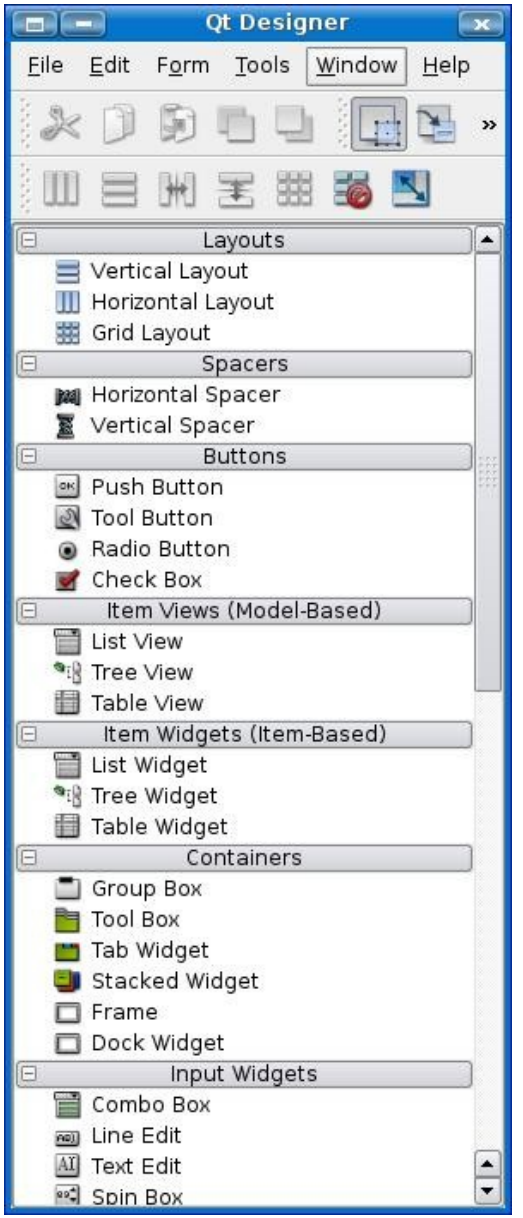
Les outils de Qt

- Qt Designer
  - outil graphique de création d'IHM (par glisser/déposer)
- Qt Assistant
  - outil d'affichage d'aide personnalisée (par défaut, l'aide de Qt)
- Qt Linguist
  - outil permettant l'internationalisation des applications de façon simple
- qmake
  - générateur de projets multi-plateformes

Les outils de Qt

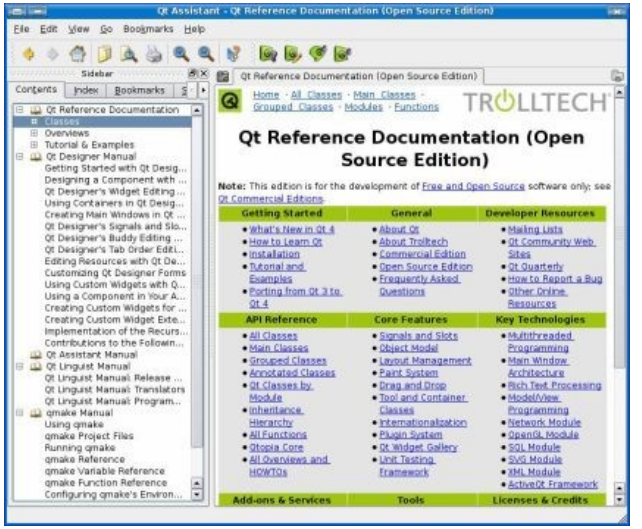
Qt Designer

- Création visuelle d'interface graphique (par glisser/déposer)
  - Large panoplie de widgets
  - Support des widgets personnalisés
  - Intégration à Visual .Net
- Avantages :
  - Gain de temps
  - Fichiers d'interface multi-plateformes
  - Simple à modifier



Les outils de Qt

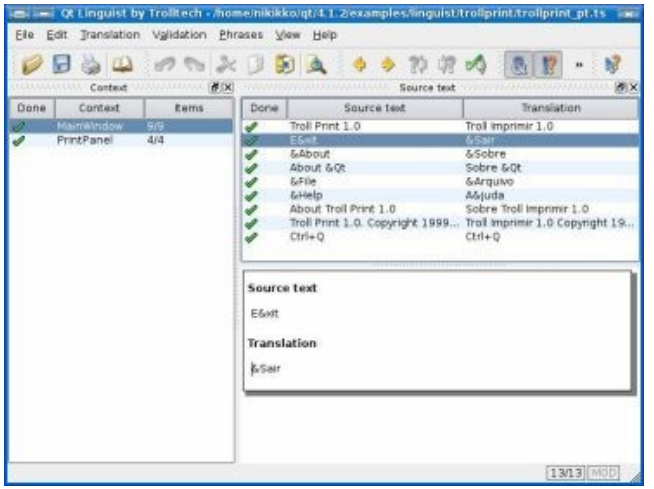
Qt Assistant



- Visualisation d'aide personnalisée
  - Navigation par pages HTML
  - Recherche par mots-clés ou dans tout le texte
- Avantages :
  - Solution multi-plateformes d'aide en ligne
  - Intégration aisée pour une aide contextuelle
  - Nécessite juste la connaissance du HTML
- Par défaut, affichage de l'**aide Qt**, très **bien faite** et très **précieuse**

Les outils de Qt

Qt Linguist



- Outil d'aide à l'internationalisation des applications
  - Extraction automatique des chaînes dans les interfaces
  - Prise en charge de tous les langages (avec le sens de lecture)

- Avantages
  - Internationalisation facile (pas besoin d'être développeur)
  - Pas besoin de recompiler l'application pour ajouter une nouvelle langue

Les outils de Qt

qmake

- Générateur de projets multi-plateformes
  - Un seul fichier projet à créer
  - Génération de projets ou makefile spécifique sur la plateforme cible (Makefile/projet Visual/projet XCode)
  - Prise en compte des dépendances
- Avantages :
  - Ceux qui ont déjà eu à faire un Makefile à la main comprendront l'intérêt ;)
  - Non dépendant du compilateur

Portage d'application de Qt 3 vers Qt 4

- **qt3to4** : outils d'aide au portage
  - Ce n'est pas l'outil magique
- **Qt3Support** : bibliothèque de compatibilité Qt3
  - Même API qu'avec Qt3
  - Les classes sont préfixés par Q3 :
    - QWidget => Q3Widget
    - QCanvas => Q3Canvas
- Travail nécessaire pour supprimer les classes de compatibilité

Et aussi...

- **Qtopia** : plateforme de développement et interface utilisateur pour l'embarqué (Linux)
  - Qtopia Core : Linux embarqué
  - Qtopia Phone : version pour téléphone

- **QSA** (Qt Script for Application)
  - langage de script pour les applications Qt, proche du Javascript, avec des extensions propres à Qt
- Tests automatiques de logiciels :
  - **KD Executor** (KDAB)
  - **Squish** (Frologic)

A venir...

- **LSB 3.1** (Linux Standard Base) : Qt 3 est dans la LSB, Qt 4 en option
- **Qt 4.2** :
  - Amélioration des graphismes (rapidité, précision)
  - Amélioration du rendu des polices
  - GraphicsView : le remplaçant du Canvas de Qt 3
- **Qt/Java** : API Qt en Java
- **Qt/Coco** : pour des utilisations serveur/terminaux
- **Programming with Qt4** : livre officiel (français ?)
- ...

Partie 2

Les nouveautés de Qt 4

Grand nettoyage

Code is only written once, but is read many times

- Simplification des paramètres et noms de fonctions
  - Plus de paramètre `name` dans le constructeur des `QObject`
  - `QString::stripWhiteSpace()` => `QString::simplified()`
- Renommage de classes pour plus de cohérence
  - `QButton` => `QAbstractButton`
- Réorganisation de certaines classes
  - `QButtonGroup` n'est plus un widget

Grand nettoyage

Exemple

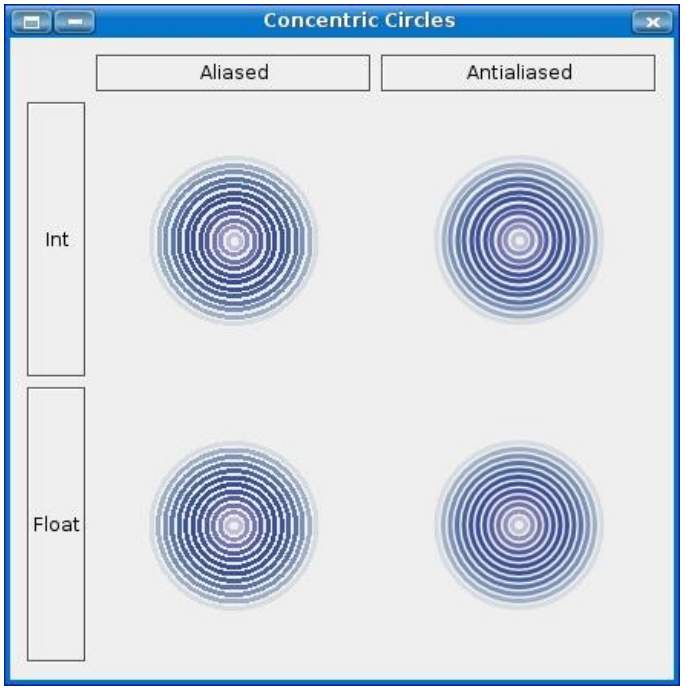
- Qt3

```
QSlider * slider = new QSlider( 0, 100, 10,
  50, Qt::Vertical, this, "mySlider" );
```

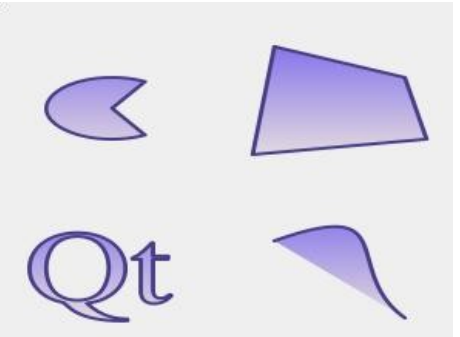
- Qt4

```
QSlider * slider = new QSlider( Qt::Vertical );
slider->setRange( 0, 100 );
slider->setPageStep( 10 );
slider->setValue( 50 );
```

Graphique

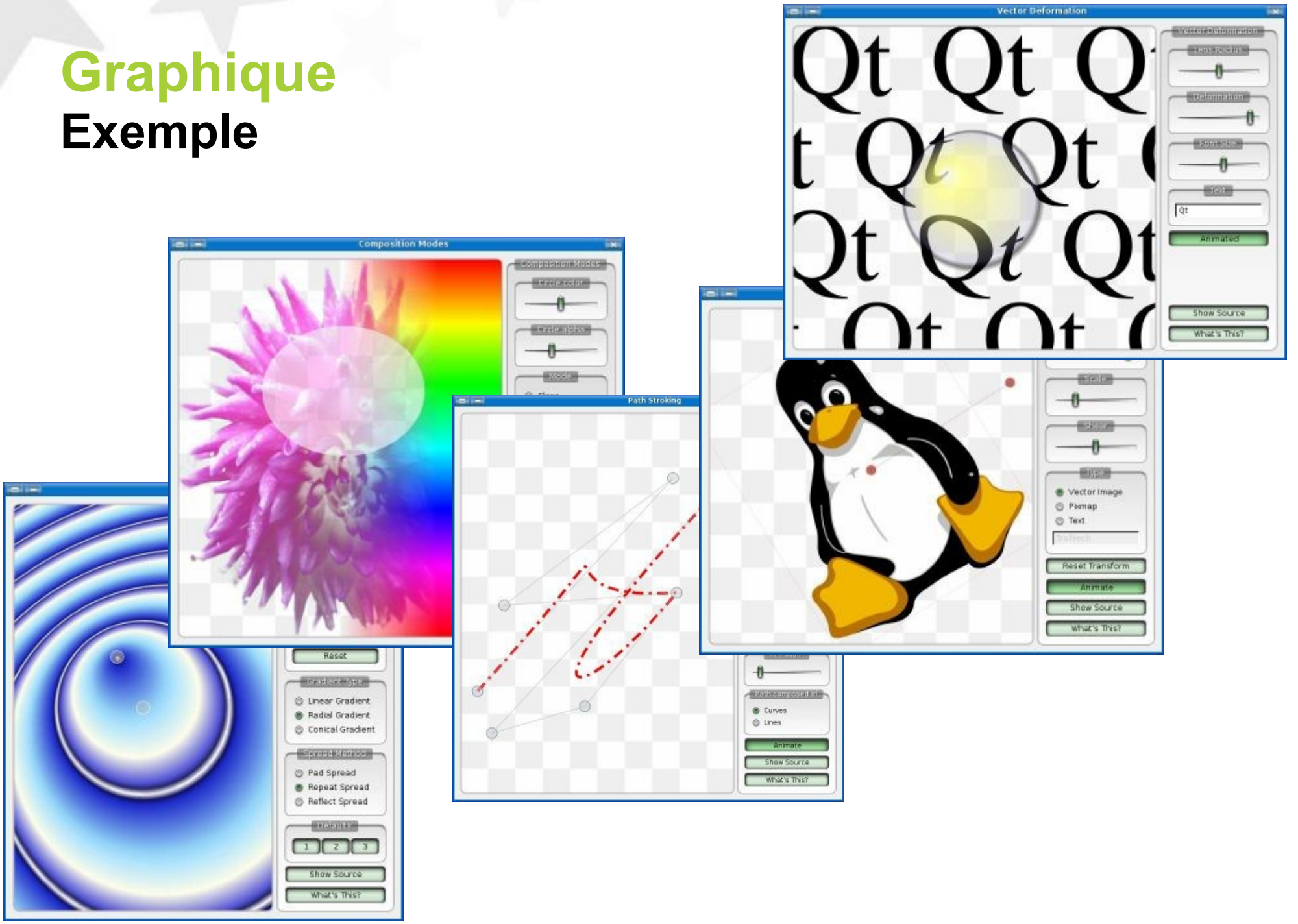


- Nouveau moteur de rendu
  - support de la transparence
  - précision flottante
  - anti-aliasing
  - ajout de dégradés (linéaire, radial et conique)
- Différents modes de composition (avec transparence)
- `QPainterPath` : création et réutilisation de formes géométriques
  - ajout de lignes/arc/courbes/textes
  - notion d'intérieur/extérieur



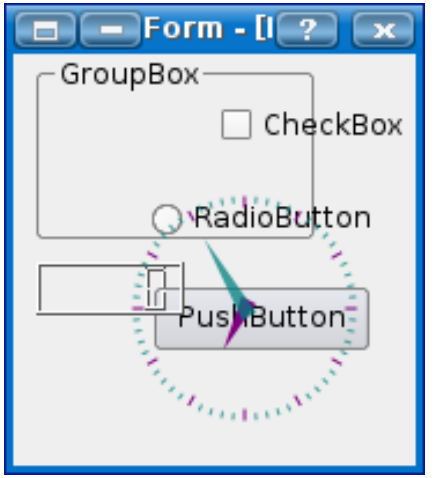
Graphique

Exemple

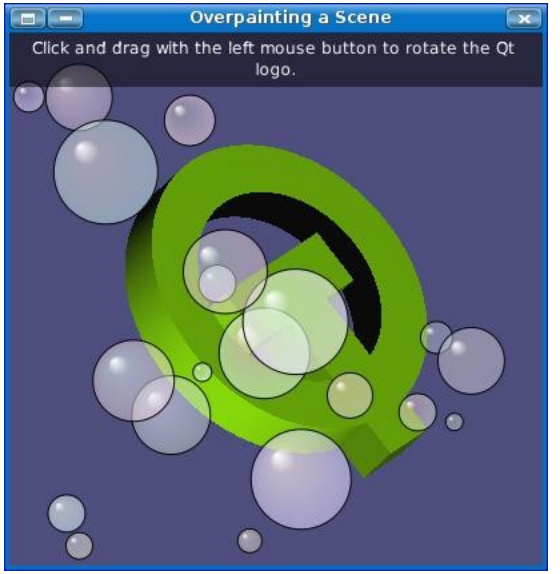


Graphique

Et bien d'autre encore...



- Double-buffering
  - par défaut pour tous les widgets
- Backing-store (depuis Qt 4.1)
  - vraie transparence des widgets
- Amélioration des moteurs de rendu
  - Possibilité d'ajouter son propre moteur
  - Dessin sur un `QGLWidget` avec un `QPainter`

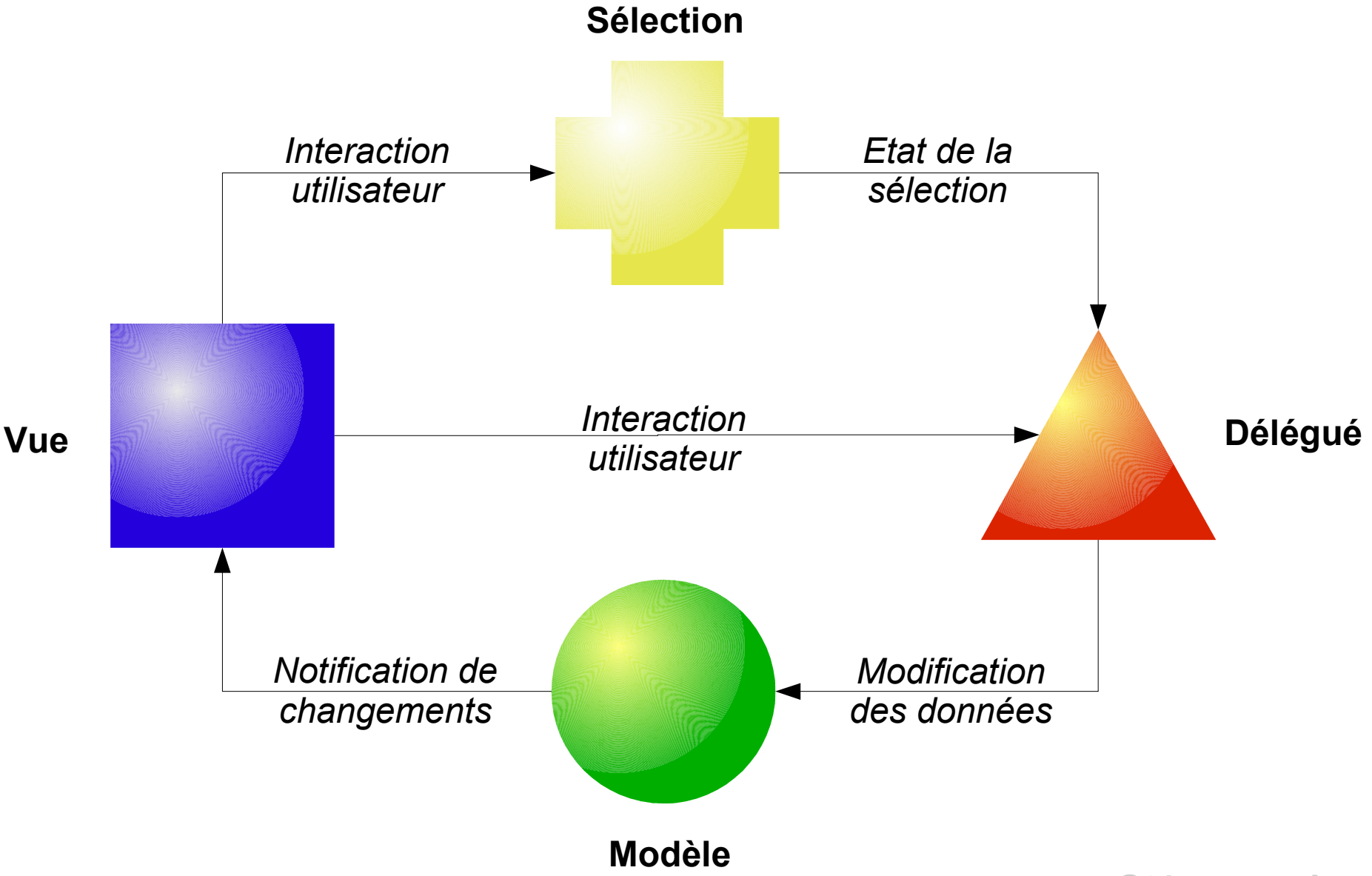


Programmation Modèle/Vue

- Grande nouveauté de Qt 4 !
  - Changement important dans la façon de programmer
- Utilisation d'un pattern MVC (légèrement adapté) pour la visualisation d'éléments :
  - liste, arbre, tableau...
  - affichage beaucoup plus performant
  - possibilité de modifier facilement l'affichage (des éléments, des en-têtes)
  - ...

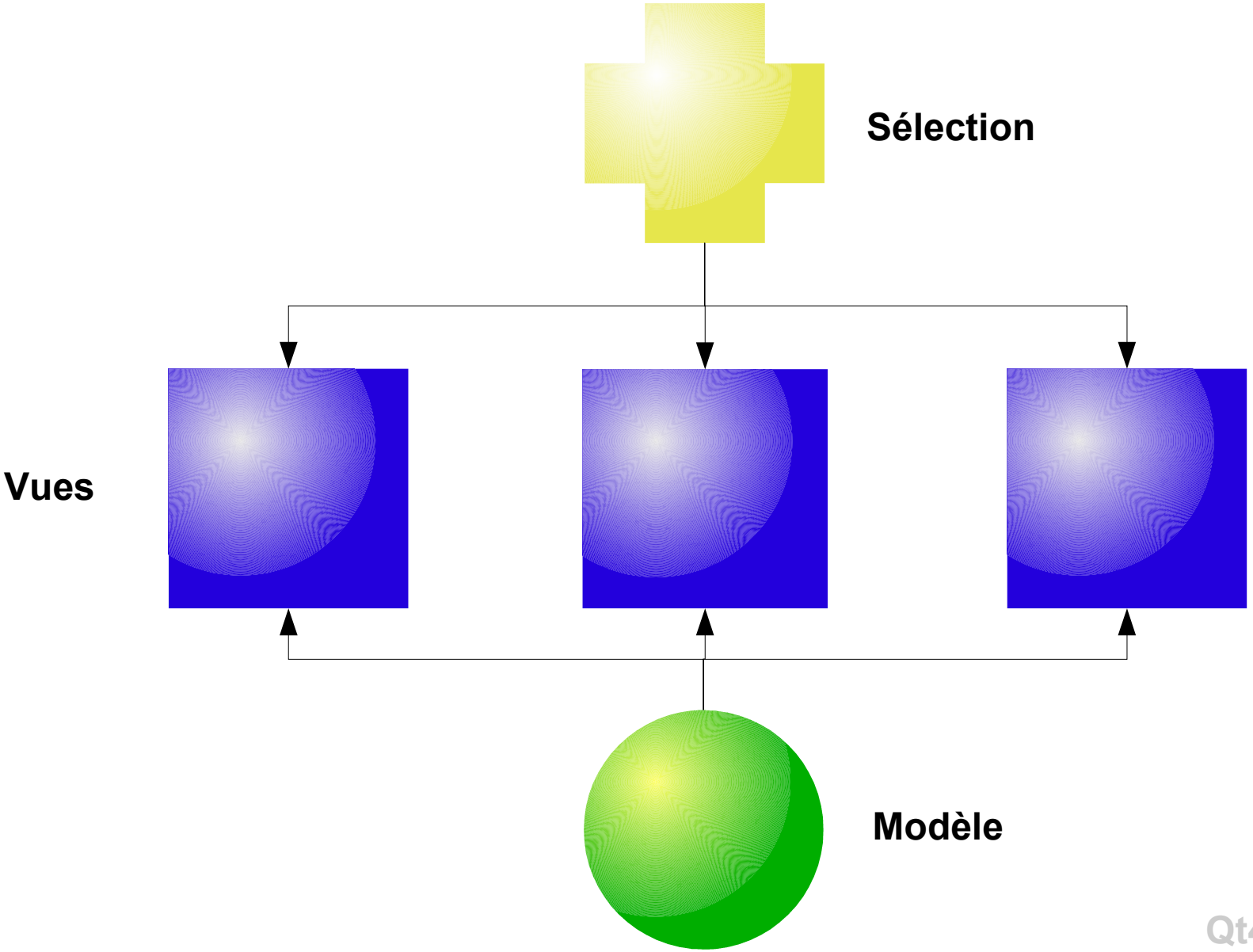
Programmation Modèle/Vue

Comment ça marche ?



Programmation Modèle/Vue

Un modèle, plusieurs vues



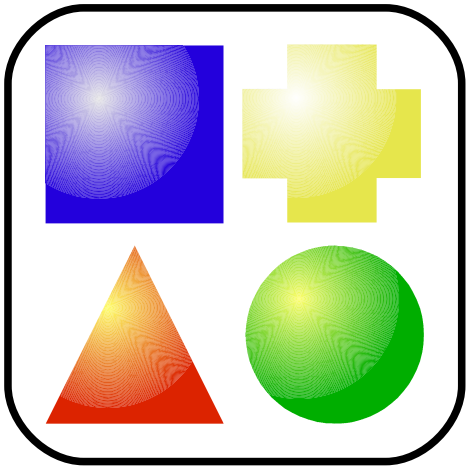
Programmation Modèle/Vue

Exemple

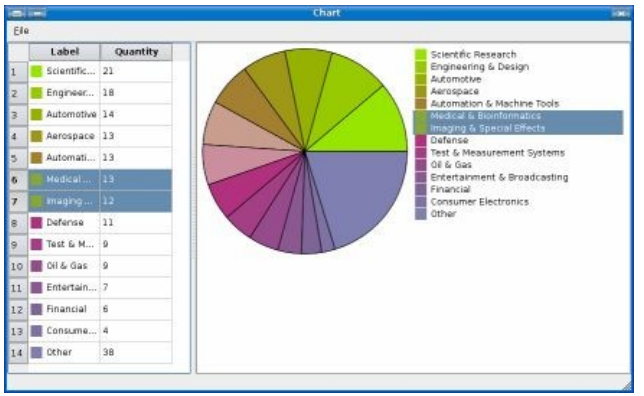
```
#include <QtGui>

int main(int argc, char *argv[]) {
    QApplication app(argc, argv);

    QStandardItemModel model(10, 10);
    // Double boucle for pour remplir le modèle

    QTableView tableView;
    tableView.setModel(&model);

    QTreeView treeView;
    treeView.setModel(&model);

    treeView.setSelectionModel(tableView.selectionModel());

    tableView.show();
    treeView.show();
    app.exec();
}
```

Programmation Modèle/Vue

Et ce n'est pas fini...



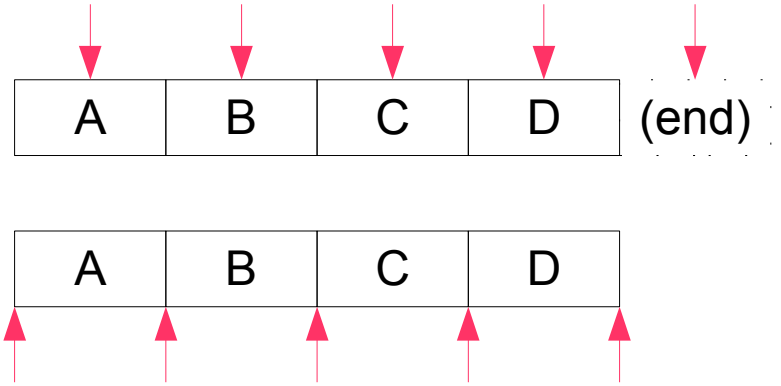
- Classes intégrant vue & modèle :
  - `QTableWidget`, `QTreeWidget`, `QListWidget`
- Les délégués contrôle :
  - le dessin de chaque élément
  - l'interaction avec l'utilisateur



- Définition de différents rôles, pour chaque élément du modèle :
  - texte affichée, icône, couleur, bulle d'aide...

Classes conteneurs

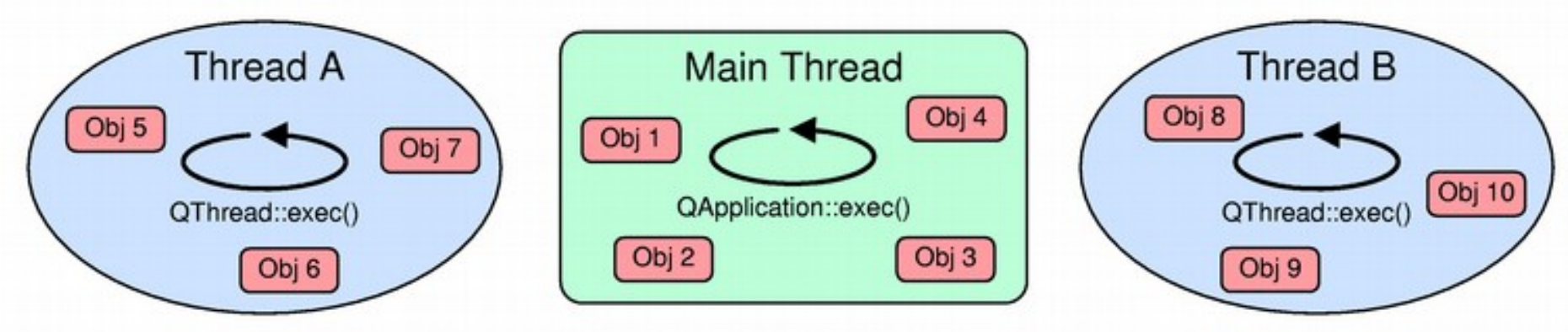
- Conteneurs classiques de la STL :
  - séquence : `QVector`, `QList`, `QQueue`...
  - associatif : `QSet`, `QMap`, `QHash`...
- Deux types d'itérateurs
  - itérateurs STL classiques
  - itérateurs Java
  - macro spéciale `foreach`



- Performant : implicitement partagés et ré-entrants
  - copie seulement lors d'une modification
  - manipulation d'un même objet dans plusieurs threads

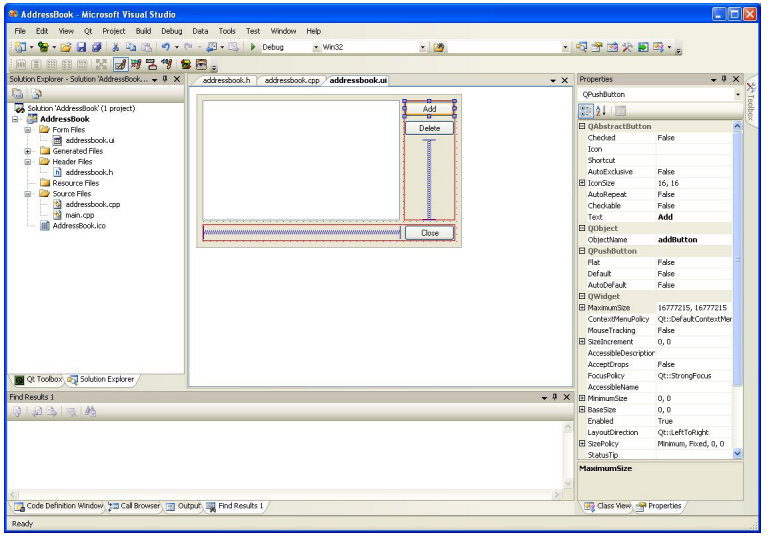
Gestion des threads

- Une boucle d'évènements par thread



- Connexion signal/slot entre deux threads
- Amélioration de la gestion des classes partagées implicitement

Le nouveau Qt Designer



- Plus grande modularité
- Intégration Visual Studio .Net
  - intégration de *Qt Assistant*
  - intégration de *Qt Designer*
  - complétion de code
- Suppression de l'éditeur de code et de projet
- **Attention** : changements importants pour la prise en charge des fichiers ui

Et bien d'autres encore...

- Séparation en plusieurs bibliothèques : QtCore, QtGui, QtXml, QtOpenGL...
- Prise en charge de l'accessibilité des applications
- Meilleur rendu du texte
- Améliorations des performances
- Utilisation mémoire plus faible
- ...

Bibliothèque pensée pour le futur !!

Questions

Références (je n'ai pas fait ça tout seul)

- Introducing Qt (Qt4, portage, QSA)
  - Xavier Ducasse & Jasmin Blanchette (Trolltech)
  - présentation effectuée en février 2006
- Qt RoadShow 2006 à Paris
  - formation effectuée lors du Qt Roadshow 2006
  - présentations disponibles sur le site de Trolltech
- L'aide de Qt, source inépuisable d'informations ;)
- Le site de Trolltech
  - http://www.trolltech.com