Cours QT - deuxième partie

Pierre POMPIDOR

18 mars 2002

Affichage graphique de bas niveau

Gestion des images et des pixmaps

Gestion des images :

Déclaration : QImage *image ;

Définition : image = new QImage(" chemin ") ;

Affichage : p.drawImage (x, y, *image) ;

Gestion des pixmaps :

Déclaration : QPixmap *pixmap ;

Définition : pixmap = new QPixmap (" chemin ") ; ou pixmap->convertFromImage (*image, 0) ;

Affichage : p.drawPixmap (x, y, *pixmap) ;

Les fonctions virtuelles à redéfinir

Les événements bas niveaux, notamment dans la gestion de tracés graphiques, doivent être gérés par l'emploi de méthodes virtuelles. Voici les quatre méthodes les plus couramment redéfinies :

```
protected:
  virtual void mousePressEvent  ( QMouseEvent* );  // Appelée lors des clics souris
  virtual void mouseMoveEvent   ( QMouseEvent* );  // Appelée lors des déplacements de la souris
  virtual void paintEvent       ( QPaintEvent* );  // Appelée lors des (ré)affichages
  virtual void resizeEvent      ( QResizeEvent* ); // Appelée lors des retaillages
```

Gestion de mousePressEvent et mouseMoveEvent

Pour récupérer la nouvelle position du curseur de la souris : `QPoint position = event->pos();`

Gestion de paintEvent

```
void ...::paintEvent(QPaintEvent *)
{
  QPainter p;
  p.begin(this);  // Association du contexte graphique à la widget courante
  ...
  p.drawLine (x, y, x2, y2); ou p.drawLine (position1, position2)  // position1 ou 2 : QPoint
  p.drawRect (x, y, l, h);
  p.drawArc ou p.drawPie  (x, y, l, h, position_départ,    // 0 = 15 heures
                                       angle);             // en seizièmes de degrés
  ...
  p.end();
}
```

Gestion de resizeEvent

1) Les cases de l'échiquier doivent être contenues dans un gestionnaire de positionnement `QGridLayout` :

```
QGridLayout* layout = new QGridLayout (this, 8, 8);
layout->addWidget (cellule[i], x, y);
...
layout->activate();
```

2) Dans la méthode slot `resizeEvent`, appelez `paintEvent` (avec `repaint()`) qui redimensionnera le pixmap affiché.

Gestion des menus

Création d'un menu

Fichiers d'entêtes à inclure :

```
qpopupmenu.h
et éventuellement qkeycode.h
```

Déclaration du menu dans le fichier d'entête : QPopupMenu *menu ;

Création du menu avec ses items :

```
QPopupMenu *parties = new QPopupMenu();

parties->insertItem("Nouvelle partie",    this, SLOT(nouvelle_partie()), CTRL+Key_N);
parties->insertSeparator();
parties->insertItem("Charger une partie", this, SLOT(chargement()),      CTRL+Key_C);
parties->insertItem("Sauver la partie",   this, SLOT(sauvegarde()),      CTRL+Key_S);

ou

parties->insertItem("&Nouvelle partie",    this, SLOT(nouvelle_partie()));
parties->insertSeparator();  // provoquera l'affichage d'un trait de séparation
parties->insertItem("&Charger une partie", this, SLOT(chargement()));
parties->insertItem("&Sauver la partie",   this, SLOT(sauvegarde()));
```

Création d'une barre de menus

Fichier d'entête à inclure :

```
qmenubar.h
```

Déclaration de la barre de menu dans le fichier d'entête :

```
QMenuBar *barre_de_menu;
```

Création de la barre de menu avec ses items :

```
barre_de_menu = new QMenuBar (this);

barre_de_menu->insertItem("&Parties",     parties);
barre_de_menu->insertItem("&Déroulement", deroulement);
barre_de_menu->insertItem("&Temps",       temps);
barre_de_menu->insertSeparator();  // provoquera l'affichage du menu d'aide à droite de la barre de menu
barre_de_menu->insertItem("&Aide",        aide);
```

Création d'une barre d'icônes

```
#include <qtoolbar.h>
#include <qtoolbutton.h>
#include <qtooltip.h>

tool = new QToolBar("Barre D'Outils",this,Top);

QToolButton *btnnouv = new QToolButton(tool);
btnnouv->setIconSet(QPixmap("icon/filenew.xpm"));
connect(btnnouv,SIGNAL(clicked()),interf->nouv,SLOT(show()));
QToolTip::add(btnnouv,"Nouvelle partie (F2)");
```

Création d'une barre de status

```
#include<qstatusbar.h>

status = new QStatusBar(this);
```