READYDEV

[GitHub](https://github.com/gkesse) | [LinkedIn](https://www.linkedin.com/in/tia-gerard-kesse/) | [SiteWeb](http://readydev.freeboxos.fr/)

Par Gérard KESSE

Plateforme de Développement Continu

Comprendre la Théorie pour mieux Pratiquer

Sciences de l'Ingénieur

Cours - Tutoriels

READYDEV

[GitHub](https://github.com/gkesse) | [LinkedIn](https://www.linkedin.com/in/tia-gerard-kesse/) | [SiteWeb](http://readydev.freeboxos.fr/)

Par Gérard KESSE

J'aime, Je partage

Montez en Compétences

Qt

Apprendre la création d’interface homme-machine en C++ avec Qt

# [Auteur](#_Sommaire)

Je suis **Gérard KESSE**,

Ingénieur en Développement Informatique C/C++/Qt,

Avec à la fois des compétences en Systèmes Embarqués et en Robotique.

Formé à Polytech'Montpellier, Je suis un professionnel de conception de projets logiciel applicatif ou embarqué dans les secteurs de l'Aéronautique, de la Robotique, des Drones et de la Vision par Ordinateur. Aussi, Je reste ouvert à d'autres types de secteurs tels que l'Énergie et les Finances.

Les Sciences de l’Ingénieur sont au cœur du métier d’ingénieur. Sur le site

**ReadyDev**, la Plateforme de Développement Continu, dont j'en suis le concepteur, vous trouverez des cours et des tutoriels adaptés aux sciences de l’ingénieur.

J'aime, Je partage.

Gérard KESSE

[GitHub](https://github.com/gkesse) | [LinkedIn](https://www.linkedin.com/in/tia-gerard-kesse/) | [SiteWeb](http://readydev.freeboxos.fr/)

[](http://readydev.freeboxos.fr/)

# Sommaire

[Auteur 2](#_Toc514235287)

[Sommaire 3](#_Toc514235288)

[Introduction 4](#_Toc514235289)

[Installation sous Windows avec MinGW 4](#_Toc514235290)

[Téléchargements 4](#_Toc514235291)

[Installation de Notepad++ 4](#_Toc514235292)

[Installation de MinGW 4](#_Toc514235293)

[Créer des IHM avec le QML 5](#_Toc514235294)

[Apprendre à communique avec le QML 5](#_Toc514235295)

[Créer un nouveau fichier Ui QML 18](#_Toc514235296)

[Apprendre les composants graphiques QML 19](#_Toc514235297)

[Importer un élément QML 20](#_Toc514235298)

[Rendre visible un élément QML 20](#_Toc514235299)

[Ajouter une couleur de fond d’écran 21](#_Toc514235300)

[Afficher un objet dans la fenêtre 24](#_Toc514235301)

[Ajouter une caméra à la scène 3D 28](#_Toc514235302)

[Ajouter de la lumière à une scène 3D 31](#_Toc514235303)

[Utiliser OpenGL avec SDL et GLEW 34](#_Toc514235304)

[Afficher une fenêtre 34](#_Toc514235305)

[Ajouter une couleur de fond d’écran 37](#_Toc514235306)

[Afficher un objet dans la fenêtre 41](#_Toc514235307)

# [Introduction](#_Sommaire)

Le C++ est un langage de programmation orienté objet. Qt est une bibliothèque de création d’interface homme-machine. Le but de ce tutoriel est de vous apprendre la création d’interface homme-machine en C++ avec Qt.

**Prérequis :**

Apprendre à compiler un projet C++ avec QMake.

# [Installation sous Windows avec MinGW](#_Sommaire)

## [Téléchargements](#_Sommaire)

**Notepad++ :**

<https://notepad-plus-plus.org/fr/>

**MinGW :**

<http://mingw.org/>

**Qt Creator :**

<https://download.qt.io/archive/qt/>

## [Installation de Notepad++](#_Sommaire)

**Plugins Notepad++ :**

TextFX

NppExport

## [Installation de MinGW](#_Sommaire)

**Packages MinGW :**

mingw32-base

mingw32-gcc-g++

# [Créer des IHM avec le QML](#_Sommaire)

QML est un langage déclaratif pour la conception d'applications centrées sur l'interface utilisateur.

## [Apprendre à communique avec le QML](#_Sommaire)

**Créer une fenêtre :**

//===============================================

// GWindowQmlSimple.qml

//===============================================

import QtQuick 2.5

import QtQuick.Window 2.2

//===============================================

Window {

}

//===============================================

**Rendre visible la fenêtre :**

//===============================================

// GWindowQmlSimple.qml

//===============================================

import QtQuick 2.5

import QtQuick.Window 2.2

//===============================================

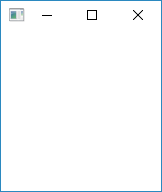
Window {

visible: true

}

//===============================================

**Afficher la fenêtre :**



**Redimensionner la fenêtre :**

//===============================================

// GWindowQmlSimple.qml

//===============================================

import QtQuick 2.5

import QtQuick.Window 2.2

//===============================================

Window {

visible: true

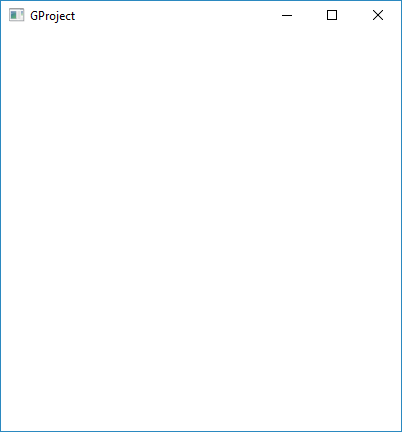
width: 400

height: 400

}

//===============================================

**Afficher la fenêtre :**



**Créer un formulaire :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

//===============================================

Rectangle {

}

//===============================================

**Ajouter le formulaire dans la fenêtre :**

//===============================================

// GWindowQmlSimple.qml

//===============================================

import QtQuick 2.5

import QtQuick.Window 2.2

//===============================================

Window {

visible: true

width: 400

height: 400

//===============================================

GWindowQmlSimpleForm {

}

}

//===============================================

**Considérer une image de fond d’écran :**

//===============================================

// login\_bg.png

//===============================================



**Ajouter l’image dans le formulaire :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

//===============================================

Rectangle {

Image {

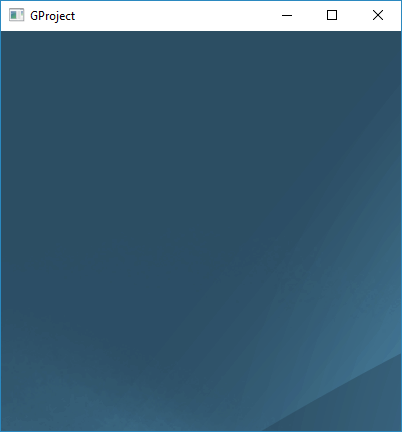
source: "qrc:/img/login\_bg.png"

}

}

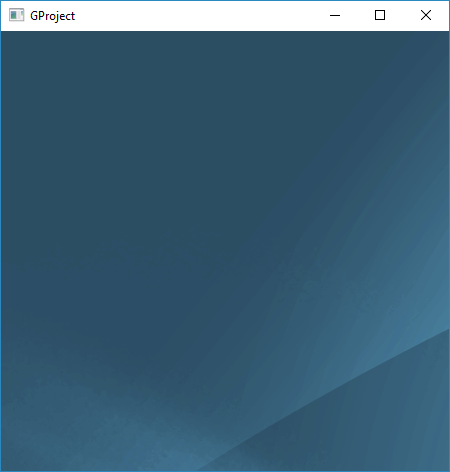
//===============================================

**Afficher la fenêtre :**



**Modifier les dimensions de la fenêtre affichée :**

L’image ne remplit pas le formulaire



**Redimensionner le formulaire :**

**Demander à l’image de remplir le formulaire :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

//===============================================

Rectangle {

width: 400

height: 400

//===============================================

Image {

anchors.fill: *parent*

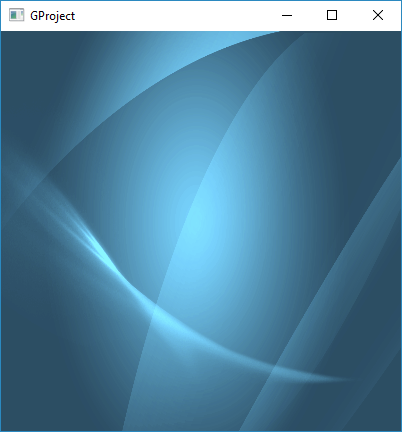
source: "qrc:/img/login\_bg.png"

}

}

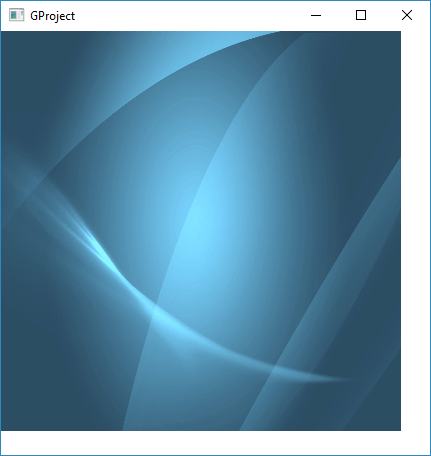
//===============================================

**Afficher la fenêtre :**



**Modifier les dimensions de la fenêtre affichée :**

Le formulaire ne remplit pas la fenêtre



**Demander au formulaire de remplir la fenêtre :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

import QtQuick.Window 2.2

//===============================================

Window {

visible: true

width: 400

height: 400

//===============================================

GWindowQmlSimpleForm {

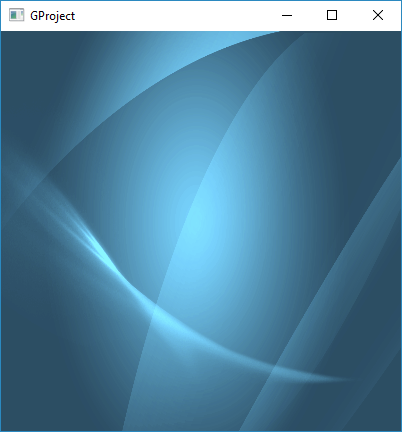
anchors.fill: *parent*

}

}

//===============================================

**Afficher la fenêtre :**



**Créer une entête dans le formulaire :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

//===============================================

Rectangle {

width: 400

height: 400

//===============================================

Image {

anchors.fill: *parent*

source: "qrc:/img/login\_bg.png"

}

//===============================================

Rectangle {

width: 200

height: 200

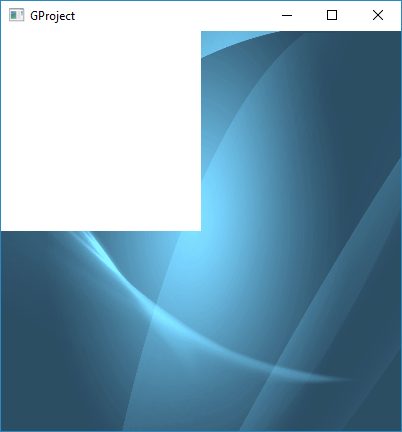
color: "#ffffff"

}

}

//===============================================

**Afficher la fenêtre :**



**Modifier la couleur de fond de l’entête dans le formulaire :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

//===============================================

Rectangle {

width: 400

height: 400

//===============================================

Image {

anchors.fill: *parent*

source: "qrc:/img/login\_bg.png"

}

//===============================================

Rectangle {

width: 200

height: 200

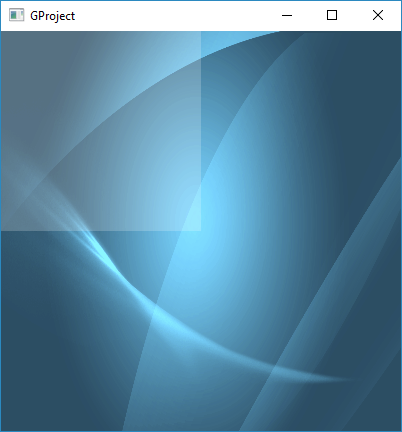
color: "#33ffffff"

}

}

//===============================================

**Afficher la fenêtre :**



**Redimensionner l’entête dans le formulaire :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

//===============================================

Rectangle {

width: 400

height: 400

//===============================================

Image {

anchors.fill: *parent*

source: "qrc:/img/login\_bg.png"

}

//===============================================

Rectangle {

height: 67

color: "#33ffffff"

transformOrigin: Item.Top

anchors.right: *parent*.right

anchors.rightMargin: 0

anchors.left: *parent*.left

anchors.leftMargin: 0

anchors.top: *parent*.top

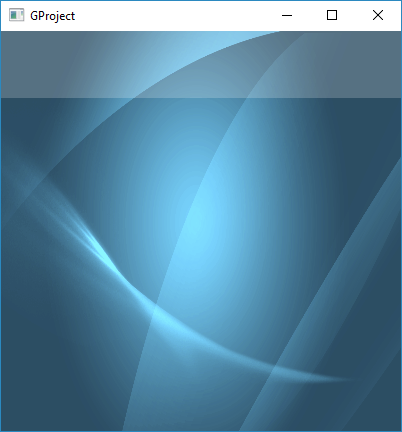
anchors.topMargin: 0

}

}

//===============================================

**Afficher la fenêtre :**



**Créer un bouton éteindre dans l’entête du formulaire :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

//===============================================

Rectangle {

width: 400

height: 400

//===============================================

Image {

anchors.fill: *parent*

source: "qrc:/img/login\_bg.png"

}

//===============================================

Rectangle {

height: 67

color: "#33ffffff"

transformOrigin: Item.Top

anchors.right: *parent*.right

anchors.rightMargin: 0

anchors.left: *parent*.left

anchors.leftMargin: 0

anchors.top: *parent*.top

anchors.topMargin: 0

Rectangle {

width: 200

height: 200

color: "#ffffff"

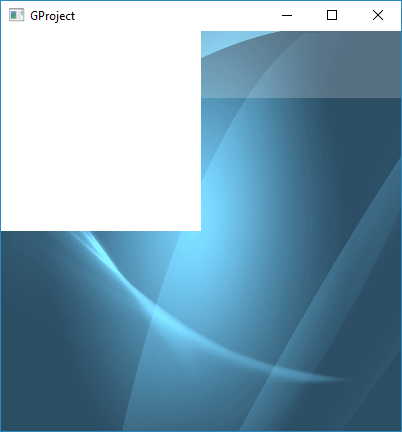
}

}

}

//===============================================

**Afficher la fenêtre :**



**Modifier la couleur de fond du bouton éteindre :**

**Redimensionner bouton éteindre :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

//===============================================

Rectangle {

width: 400

height: 400

//===============================================

Image {

anchors.fill: *parent*

source: "qrc:/img/login\_bg.png"

}

//===============================================

Rectangle {

height: 67

color: "#33ffffff"

transformOrigin: Item.Top

anchors.right: *parent*.right

anchors.rightMargin: 0

anchors.left: *parent*.left

anchors.leftMargin: 0

anchors.top: *parent*.top

anchors.topMargin: 0

Rectangle {

width: 50

height: 50

color: "#33ffffff"

anchors.right: *parent*.right

anchors.rightMargin: 10

anchors.top: *parent*.top

anchors.topMargin: 10

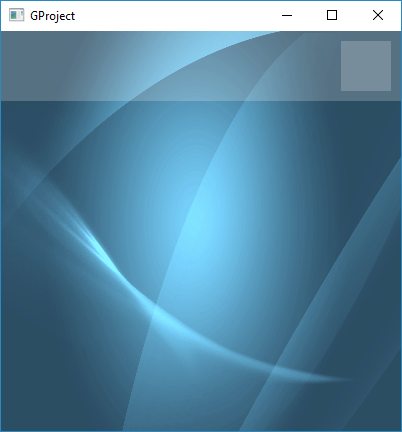
}

}

}

//===============================================

**Afficher la fenêtre :**



**Considérer une image de fond du bouton éteindre :**



**Ajouter l’image de fond du bouton éteindre :**

//===============================================

// GWindowQmlSimpleForm.ui.qml

//===============================================

import QtQuick 2.5

//===============================================

Rectangle {

width: 400

height: 400

//===============================================

Image {

anchors.fill: *parent*

source: "qrc:/img/login\_bg.png"

}

//===============================================

Rectangle {

height: 67

color: "#33ffffff"

transformOrigin: Item.Top

anchors.right: *parent*.right

anchors.rightMargin: 0

anchors.left: *parent*.left

anchors.leftMargin: 0

anchors.top: *parent*.top

anchors.topMargin: 0

Rectangle {

width: 200

height: 200

color: "#ffffff"

}

}

}

//===============================================

**Afficher la fenêtre :**

## [Créer un nouveau fichier Ui QML](#_Sommaire)

**Créer un nouveau fichier Ui QML :**

Fichier

Nouveau fichier ou Projet

Fichiers et Classes

Qt

QtQuick Ui File

Choose

Component Name

Main

Component Form Name

MainForm

Suivant

Terminer

**Fichiers Crées :**

Main.qml

MainForm.ui.qml

**Main.qml**

//===============================================

import QtQuick 2.4

//===============================================

MainForm {

}

//===============================================

**MainForm.ui.qml**

//===============================================

import QtQuick 2.4

//===============================================

Item {

width: 400

height: 400

}

//===============================================

## [Apprendre les composants graphiques QML](#_Sommaire)

**Ouvrir le fichier Ui QML :**

MainForm.ui.qml

**Eléments de base :**

Border Image

Flickable

Focus Scope

Image

Item

Mouse Area

Rectangle

Text

Text Edit

Text Input

**Eléments de position :**

Column

Flow

Grid

Row

**Eléments de vue :**

Grid View

List View

Path View

## [Importer un élément QML](#_Sommaire)

**Importer l’élément Window :**

//===============================================

import QtQuick 2.4

import QtQuick.Window 2.0

//===============================================

Window {

MainForm {

}

}

//===============================================

## [Rendre visible un élément QML](#_Sommaire)

**Rendre visible l’élément Window :**

//===============================================

import QtQuick 2.4

import QtQuick.Window 2.0

//===============================================

Window {

visible: true

//===============================================

MainForm {

}

}

//===============================================

## [Ajouter une couleur de fond d’écran](#_Sommaire)

**Objectif :**

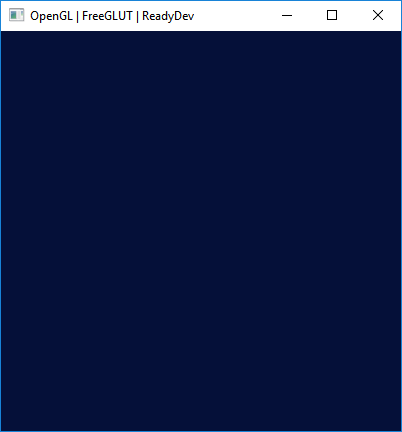
Ajouter une couleur de fond d’écran à une fenêtre.

**Implémentation :**

Créer un gestionnaire de fenêtre (GWindow).

Ajouter une couleur de fond d’écran (setBackground()).

**Résultat :**



**Dossier projet :**

src/main.cpp

src/manager/GConfig.h

src/manager/GWindow.cpp

src/manager/GWindowGLUT.cpp

**Programme principal :**

//===============================================

#include "GWindow.h"

#include "GConfig.h"

//===============================================

int main**(**int argc**,** char**\*\*** argv**)** **{**

GConfig**::**Instance**()->**setData**(**"WINDOW\_TYPE"**,** "OPENGL\_GLUT"**);**

GConfig**::**Instance**()->**setData**(**"BACKGROUND\_FLAG"**,** "TRUE"**);**

GConfig**::**Instance**()->**setData**(**"BACKGROUND\_COLOR"**,** "5;16;57;255"**);**

GWindow**::**Instance**()->**show**(&**argc**,** argv**);**

**return** 0**;**

**}**

//===============================================

**Affichage de la fenêtre :**

//===============================================

void GWindowGLUT**::**show**(**int**\*** argc**,** char**\*\*** argv**)** **{**

glutInit**(**argc**,** argv**);**

glutInitDisplayMode**(**GLUT\_DOUBLE **|** GLUT\_RGBA **|** GLUT\_DEPTH**);**

glutInitWindowPosition**(**m\_x**,** m\_y**);**

glutInitWindowSize**(**m\_w**,** m\_h**);**

glutCreateWindow**(**m\_title**.**toStdString**().**c\_str**());**

glutSetOption **(**GLUT\_ACTION\_ON\_WINDOW\_CLOSE**,** GLUT\_ACTION\_GLUTMAINLOOP\_RETURNS**);**

setBackground**();**

glutIdleFunc**(**IdleFunc**);**

glutDisplayFunc**(**DisplayFunc**);**

glutMainLoop**();**

**}**

//===============================================

**Ajout d’une couleur de fond d’écran :**

//===============================================

void GWindow**::**setBackground**()** **{**

QString m\_backgroundFlag **=** GConfig**::**Instance**()->**getData**(**"BACKGROUND\_FLAG"**);**

**if(**m\_backgroundFlag **==** "FALSE"**)** **return;**

QString m\_backgroundColor **=** GConfig**::**Instance**()->**getData**(**"BACKGROUND\_COLOR"**);**

QStringList m\_colorMap **=** m\_backgroundColor**.**split**(**";"**);**

float m\_red **=** QString**(**m\_colorMap**[**0**]).**toFloat**()/**255.0**;**

float m\_green **=** QString**(**m\_colorMap**[**1**]).**toFloat**()/**255.0**;**

float m\_blue **=** QString**(**m\_colorMap**[**2**]).**toFloat**()/**255.0**;**

float m\_alpha **=** QString**(**m\_colorMap**[**3**]).**toFloat**()/**255.0**;**

glClearColor**(**m\_red**,** m\_green**,** m\_blue**,** m\_alpha**);**

**}**

//===============================================

## [Afficher un objet dans la fenêtre](#_Sommaire)

**Objectif :**

Afficher un objet dans la fenêtre.

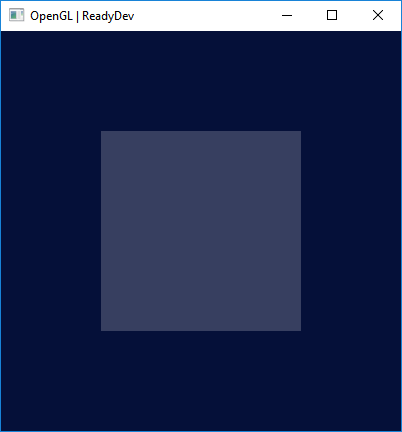
**Implémentation :**

Créer un gestionnaire de fenêtre (GWindow).

Créer un gestionnaire de dessin (GDraw).

Dessiner un objet dans la fenêtre (draw()).

**Résultat :**



**Dossier projet :**

src/main.cpp

src/manager/GConfig.cpp

Src/manager/GWindow.cpp

Src/manager/GWindowGLUT.cpp

**Programme principal :**

//===============================================

#include "GWindow.h"

//===============================================

int main**(**int argc**,** char**\*\*** argv**)** **{**

GWindow**::**Instance**()->**show**(&**argc**,** argv**);**

**return** 0**;**

**}**

//===============================================

**Affichage de la fenêtre :**

//===============================================

void GWindow**::**show**(**int**\*** argc**,** char**\*\*** argv**)** **{**

glutInit**(**argc**,** argv**);**

glutInitDisplayMode**(**GLUT\_DOUBLE **|** GLUT\_RGBA **|** GLUT\_DEPTH**);**

glutInitWindowPosition**(**m\_x**,** m\_y**);**

glutInitWindowSize**(**m\_w**,** m\_h**);**

glutCreateWindow**(**m\_title**.**toStdString**().**c\_str**());**

glutSetOption **(**GLUT\_ACTION\_ON\_WINDOW\_CLOSE**,** GLUT\_ACTION\_GLUTMAINLOOP\_RETURNS**);**

setBackground**();**

glutIdleFunc**(**IdleFunc**);**

glutDisplayFunc**(**DisplayFunc**);**

glutMainLoop**();**

**}**

//===============================================

**Fonction de rappel de la tâche de fond :**

//===============================================

void GWindow**::**IdleFunc**()** **{**

glClear**(**GL\_COLOR\_BUFFER\_BIT **|** GL\_DEPTH\_BUFFER\_BIT**);**

GDraw::Instance()->draw();

glutSwapBuffers**();**

**}**

//===============================================

**Dessin de l’objet 3D :**

//===============================================

void GDraw**::**draw**()** **{**

float m\_red **=** 55.0**/**255.0**;**

float m\_green **=** 63.0**/**255.0**;**

float m\_blue **=** 96.0**/**255.0**;**

glColor3f**(**m\_red**,** m\_green**,** m\_blue**);**

float m\_width **=** 0.5**;**

float m\_height **=** 0.5**;**

float m\_depth **=** 0.5**;**

btVector3 m\_vertices**[**8**]** **=** **{**

btVector3**(**m\_width**,** m\_height**,** m\_depth**),**

btVector3**(-**m\_width**,** m\_height**,** m\_depth**),**

btVector3**(**m\_width**,** **-**m\_height**,** m\_depth**),**

btVector3**(-**m\_width**,** **-**m\_height**,** m\_depth**),**

btVector3**(**m\_width**,** m\_height**,** **-**m\_depth**),**

btVector3**(-**m\_width**,** m\_height**,** **-**m\_depth**),**

btVector3**(**m\_width**,** **-**m\_height**,** **-**m\_depth**),**

btVector3**(-**m\_width**,** **-**m\_height**,** **-**m\_depth**)**

**};**

int m\_indices**[**36**]** **=** **{**

0**,**1**,**2**,**

3**,**2**,**1**,**

4**,**0**,**6**,**

6**,**0**,**2**,**

5**,**1**,**4**,**

4**,**1**,**0**,**

7**,**3**,**1**,**

7**,**1**,**5**,**

5**,**4**,**7**,**

7**,**4**,**6**,**

7**,**2**,**3**,**

7**,**6**,**2

**};**

glBegin**(**GL\_TRIANGLES**);**

**for(**int i **=** 0**;** i **<** 36**;** i **+=** 3**)** **{**

const btVector3 **&**m\_vert1 **=** m\_vertices**[**m\_indices**[**i**]];**

const btVector3 **&**m\_vert2 **=** m\_vertices**[**m\_indices**[**i**+**1**]];**

const btVector3 **&**m\_vert3 **=** m\_vertices**[**m\_indices**[**i**+**2**]];**

btVector3 m\_normal **=** **(**m\_vert3**-**m\_vert1**).**cross**(**m\_vert2**-**m\_vert1**);**

m\_normal**.**normalize **();**

glNormal3f**(**m\_normal**.**getX**(),** m\_normal**.**getY**(),** m\_normal**.**getZ**());**

glVertex3f**(**m\_vert1**.**x**(),** m\_vert1**.**y**(),** m\_vert1**.**z**());**

glVertex3f**(**m\_vert2**.**x**(),** m\_vert2**.**y**(),** m\_vert2**.**z**());**

glVertex3f**(**m\_vert3**.**x**(),** m\_vert3**.**y**(),** m\_vert3**.**z**());**

**}**

glEnd**();**

**}**

//===============================================

## [Ajouter une caméra à la scène 3D](#_Sommaire)

**Objectif :**

Ajouter une caméra à la scène 3D.

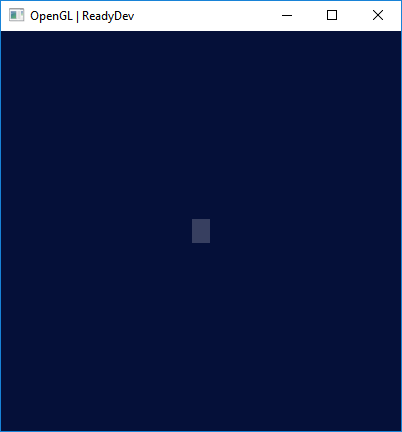
**Implémentation :**

Créer un gestionnaire de fenêtre (GWindow).

Créer un gestionnaire de camera (GCamera).

Afficher le champ de vision de la camera (update()).

**Résultat :**



**Dossier projet :**

src/main.cpp

src/GWindow.h

src/GWindow.cpp

win/bin/

win/build/

**Programme principal :**

//===============================================

#include "GWindow.h"

//===============================================

int main**(**int argc**,** char**\*\*** argv**)** **{**

GWindow**::**Instance**()->**show**(&**argc**,** argv**);**

**return** 0**;**

**}**

//===============================================

**Affichage de la fenêtre :**

//===============================================

void GWindow**::**show**(**int**\*** argc**,** char**\*\*** argv**)** **{**

glutInit**(**argc**,** argv**);**

glutInitDisplayMode**(**GLUT\_DOUBLE **|** GLUT\_RGBA **|** GLUT\_DEPTH**);**

glutInitWindowPosition**(**m\_x**,** m\_y**);**

glutInitWindowSize**(**m\_w**,** m\_h**);**

glutCreateWindow**(**m\_title**.**toStdString**().**c\_str**());**

glutSetOption **(**GLUT\_ACTION\_ON\_WINDOW\_CLOSE**,** GLUT\_ACTION\_GLUTMAINLOOP\_RETURNS**);**

setBackground**();**

glutReshapeFunc**(**ReshapeFunc**);**

glutIdleFunc**(**IdleFunc**);**

glutDisplayFunc**(**DisplayFunc**);**

glutMainLoop**();**

**}**

//===============================================

**Fonction de rappel de la tâche de fond :**

//===============================================

void GWindow**::**IdleFunc**()** **{**

glClear**(**GL\_COLOR\_BUFFER\_BIT **|** GL\_DEPTH\_BUFFER\_BIT**);**

GDraw**::**Instance**()->**draw**();**

GCamera**::**Instance**()->**update**();**

glutSwapBuffers**();**

**}**

//===============================================

**Fonction de rappel du redimensionnement de la fenêtre :**

//===============================================

void GWindow**::**ReshapeFunc**(**int w**,** int h**)** **{**

glViewport**(**0**,** 0**,** w**,** h**);**

GCamera**::**Instance**()->**setWindowSize**(**w**,** h**);**

GCamera**::**Instance**()->**update**();**

**}**

//===============================================

**Affichage du champ de vision de la camera :**

//===============================================

void GCamera**::**update**()** **{**

**if(**m\_w **==** 0 **&&** m\_h **==** 0**)** **return;**

glMatrixMode**(**GL\_PROJECTION**);**

glLoadIdentity**();**

float m\_ratio **=** **(**float**)**m\_w**/**m\_h**;**

float m\_nearPlane **=** 1.0**;**

float m\_farPlane **=** 1000.0**;**

float m\_left **=** **-**m\_ratio**\***m\_nearPlane**;**

float m\_right **=** m\_ratio**\***m\_nearPlane**;**

float m\_top **=** **-**m\_nearPlane**;**

float m\_bottom **=** m\_nearPlane**;**

float m\_zNear **=** m\_nearPlane**;**

float m\_zFar **=** m\_farPlane**;**

glFrustum**(**m\_left**,** m\_right**,** m\_top**,** m\_bottom**,** m\_zNear**,** m\_zFar**);**

glMatrixMode**(**GL\_MODELVIEW**);**

glLoadIdentity**();**

float m\_eyeX **=** 10.0**;**

float m\_eyeY **=** 5.0**;**

float m\_eyeZ **=** 0.0**;**

float m\_centerX **=** 0.0**;**

float m\_centerY **=** 0.0**;**

float m\_centerZ **=** 0.0**;**

float m\_upX **=** 0.0**;**

float m\_upY **=** 1.0**;**

float m\_upZ **=** 0.0**;**

gluLookAt**(**m\_eyeX**,** m\_eyeY**,** m\_eyeZ**,** m\_centerX**,** m\_centerY**,** m\_centerZ**,** m\_upX**,** m\_upY**,** m\_upZ**);**

**}**

//===============================================

## [Ajouter de la lumière à une scène 3D](#_Sommaire)

**Objectif :**

Ajouter de la lumière à une scène 3D.

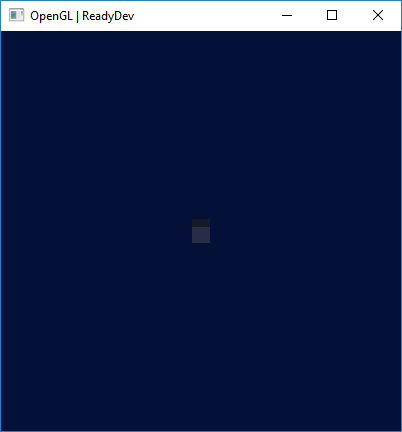
**Implémentation :**

Créer un gestionnaire de fenêtre (GWindow).

Créer un gestionnaire de lumière (GLight).

Ajouter de la lumière à la scène (initLight()).

**Résultat :**



**Dossier projet :**

src/main.cpp

src/GWindow.h

src/GWindow.cpp

win/bin/

win/build/

**Programme principal :**

//===============================================

#include "GWindow.h"

//===============================================

int main**(**int argc**,** char**\*\*** argv**)** **{**

GWindow**::**Instance**()->**show**(&**argc**,** argv**);**

**return** 0**;**

**}**

//===============================================

**Affichage de la fenêtre :**

//===============================================

void GWindow**::**show**(**int**\*** argc**,** char**\*\*** argv**)** **{**

glutInit**(**argc**,** argv**);**

glutInitDisplayMode**(**GLUT\_DOUBLE **|** GLUT\_RGBA **|** GLUT\_DEPTH**);**

glutInitWindowPosition**(**m\_x**,** m\_y**);**

glutInitWindowSize**(**m\_w**,** m\_h**);**

glutCreateWindow**(**m\_title**.**toStdString**().**c\_str**());**

glutSetOption **(**GLUT\_ACTION\_ON\_WINDOW\_CLOSE**,** GLUT\_ACTION\_GLUTMAINLOOP\_RETURNS**);**

setLight**();**

setBackground**();**

glutReshapeFunc**(**ReshapeFunc**);**

glutIdleFunc**(**IdleFunc**);**

glutDisplayFunc**(**DisplayFunc**);**

glutMainLoop**();**

**}**

//===============================================

**Ajout de la lumière à la scène 3D :**

//===============================================

void GWindow**::**setLight**()** **{**

GLight**::**Instance**()->**initLight**();**

**}**

//===============================================

**Création de la lumière ambiante, diffuse, spéculaire :**

//===============================================

void GLight**::**initLight**()** **{**

GLfloat m\_ambient**[]** **=** **{**51.0**/**255.0**,** 51.0**/**255.0**,** 51.0**/**255.0**,** 255.0**/**255.0**};**

GLfloat m\_diffuse**[]** **=** **{**255.0**/**255.0**,** 255.0**/**255.0**,** 255.0**/**255.0**,** 255.0**/**255.0**};**

GLfloat m\_specular**[]** **=** **{**255.0**/**255.0**,** 255.0**/**255.0**,** 255.0**/**255.0**,** 255.0**/**255.0**};**

GLfloat m\_position**[]** **=** **{**5.0**,** 10.0**,** 1.0**,** 0.0**};**

glLightfv**(**GL\_LIGHT0**,** GL\_AMBIENT**,** m\_ambient**);**

glLightfv**(**GL\_LIGHT0**,** GL\_DIFFUSE**,** m\_diffuse**);**

glLightfv**(**GL\_LIGHT0**,** GL\_SPECULAR**,** m\_specular**);**

glLightfv**(**GL\_LIGHT0**,** GL\_POSITION**,** m\_position**);**

glEnable**(**GL\_LIGHTING**);**

glEnable**(**GL\_LIGHT0**);**

glEnable**(**GL\_COLOR\_MATERIAL**);**

glMaterialfv**(**GL\_FRONT**,** GL\_SPECULAR**,** m\_specular**);**

glMateriali**(**GL\_FRONT**,** GL\_SHININESS**,** 15**);**

glShadeModel**(**GL\_SMOOTH**);**

glEnable**(**GL\_DEPTH\_TEST**);**

glDepthFunc**(**GL\_LESS**);**

**}**

//===============================================

# [Utiliser OpenGL avec SDL et GLEW](#_Sommaire)

## [Afficher une fenêtre](#_Sommaire)

**Objectif :**

Afficher une fenêtre.

**Implémentation :**

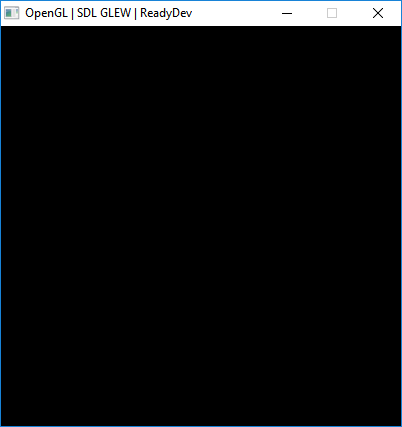
Créer un gestionnaire de paramètres (GConfig).

Créer un gestionnaire de fenêtre (GWindow).

Initialiser les paramètres de l’application (setData()).

Afficher la fenêtre (show()).

**Résultat :**



**Dossier projet :**

src/main.cpp

Src/manager/GConfig.cpp

Src/manager/GWindow.cpp

Src/manager/GWindowSDL.cpp

**Programme principale :**

//===============================================

#include "GWindow.h"

#include "GConfig.h"

//===============================================

int main**(**int argc**,** char**\*\*** argv**)** **{**

GConfig**::**Instance**()->**setData**(**"WINDOW\_TYPE"**,** "OPENGL\_SDL\_GLEW"**);**

GWindow**::**Instance**()->**show**();**

**return** 0**;**

**}**

//===============================================

**Affichage de la fenêtre :**

//===============================================

void GWindowSDL**::**show**(**int**\*** argc**,** char**\*\*** argv**)** **{**

SDL\_Init**(**SDL\_INIT\_VIDEO**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_CONTEXT\_MAJOR\_VERSION**,** 3**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_CONTEXT\_MINOR\_VERSION**,** 1**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_DOUBLEBUFFER**,** 1**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_DEPTH\_SIZE**,** 24**);**

m\_window **=** SDL\_CreateWindow**(**m\_title**.**c\_str**(),** m\_x**,** m\_y**,** m\_w**,** m\_h**,** SDL\_WINDOW\_SHOWN **|** SDL\_WINDOW\_OPENGL**);**

SDL\_GLContext m\_context **=** SDL\_GL\_CreateContext**(**m\_window**);**

glewInit**();**

SDL\_Event m\_event**;**

**while(**1**)** **{**

SDL\_WaitEvent**(&**m\_event**);**

**if(**m\_event**.**window**.**event **==** SDL\_WINDOWEVENT\_CLOSE**)** **break;**

glClear**(**GL\_COLOR\_BUFFER\_BIT**);**

SDL\_GL\_SwapWindow**(**m\_window**);**

**}**

SDL\_GL\_DeleteContext**(**m\_context**);**

SDL\_DestroyWindow**(**m\_window**);**

SDL\_Quit**();**

**}**

//===============================================

**Initialisation des paramètres de la fenêtre :**

//===============================================

GWindowSDL**::**GWindowSDL**()** **{**

m\_x **=** SDL\_WINDOWPOS\_CENTERED**;**

m\_y **=** SDL\_WINDOWPOS\_CENTERED**;**

m\_w **=** 400**;**

m\_h **=** 400**;**

m\_title **=** "OpenGL | SDL GLEW | ReadyDev"**;**

**}**

//===============================================

## [Ajouter une couleur de fond d’écran](#_Sommaire)

**Objectif :**

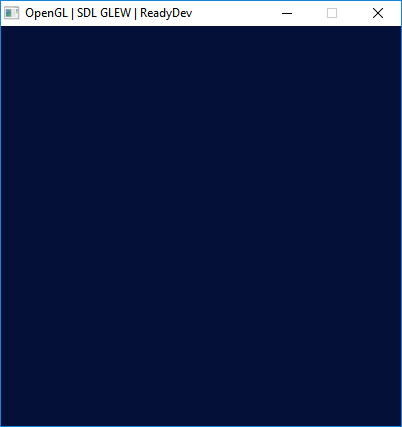
Ajouter une couleur de fond d’écran à une fenêtre.

**Implémentation :**

Créer un gestionnaire de fenêtre (GWindow).

Ajouter une couleur de fond d’écran (setBackground()).

**Résultat :**



**Dossier projet :**

src/main.cpp

Src/manager/GConfig.cpp

Src/manager/GWindow.cpp

Src/manager/GWindowSDL.cpp

**Programme principal :**

//===============================================

#include "GWindow.h"

#include "GConfig.h"

//===============================================

int main**(**int argc**,** char**\*\*** argv**)** **{**

GConfig**::**Instance**()->**setData**(**"WINDOW\_TYPE"**,** "OPENGL\_SDL\_GLEW"**);**

GConfig**::**Instance**()->**setData**(**"BACKGROUND\_FLAG"**,** "TRUE"**);**

GConfig**::**Instance**()->**setData**(**"BACKGROUND\_COLOR"**,** "5;16;57;255"**);**

GWindow**::**Instance**()->**show**(&**argc**,** argv**);**

**return** 0**;**

**}**

//===============================================

**Affichage de la fenêtre :**

//===============================================

void GWindowSDL**::**show**(**int**\*** argc**,** char**\*\*** argv**)** **{**

SDL\_Init**(**SDL\_INIT\_VIDEO**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_CONTEXT\_MAJOR\_VERSION**,** 3**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_CONTEXT\_MINOR\_VERSION**,** 1**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_DOUBLEBUFFER**,** 1**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_DEPTH\_SIZE**,** 24**);**

m\_window **=** SDL\_CreateWindow**(**m\_title**.**c\_str**(),** m\_x**,** m\_y**,** m\_w**,** m\_h**,** SDL\_WINDOW\_SHOWN **|** SDL\_WINDOW\_OPENGL**);**

SDL\_GLContext m\_context **=** SDL\_GL\_CreateContext**(**m\_window**);**

glewInit**();**

SDL\_Event m\_event**;**

**while(**1**)** **{**

SDL\_WaitEvent**(&**m\_event**);**

**if(**m\_event**.**window**.**event **==** SDL\_WINDOWEVENT\_CLOSE**)** **break;**

setBackground**();**

glClear**(**GL\_COLOR\_BUFFER\_BIT**);**

SDL\_GL\_SwapWindow**(**m\_window**);**

**}**

SDL\_GL\_DeleteContext**(**m\_context**);**

SDL\_DestroyWindow**(**m\_window**);**

SDL\_Quit**();**

**}**

//===============================================

**Ajout d’une couleur de fond d’écran :**

//===============================================

void GWindow**::**setBackground**()** **{**

QString m\_backgroundFlag **=** GConfig**::**Instance**()->**getData**(**"BACKGROUND\_FLAG"**);**

**if(**m\_backgroundFlag **==** "FALSE"**)** **return;**

QString m\_backgroundColor **=** GConfig**::**Instance**()->**getData**(**"BACKGROUND\_COLOR"**);**

QStringList m\_colorMap **=** m\_backgroundColor**.**split**(**";"**);**

float m\_red **=** QString**(**m\_colorMap**[**0**]).**toFloat**()/**255.0**;**

float m\_green **=** QString**(**m\_colorMap**[**1**]).**toFloat**()/**255.0**;**

float m\_blue **=** QString**(**m\_colorMap**[**2**]).**toFloat**()/**255.0**;**

float m\_alpha **=** QString**(**m\_colorMap**[**3**]).**toFloat**()/**255.0**;**

glClearColor**(**m\_red**,** m\_green**,** m\_blue**,** m\_alpha**);**

**}**

//===============================================

## [Afficher un objet dans la fenêtre](#_Sommaire)

**Objectif :**

Afficher un objet dans la fenêtre.

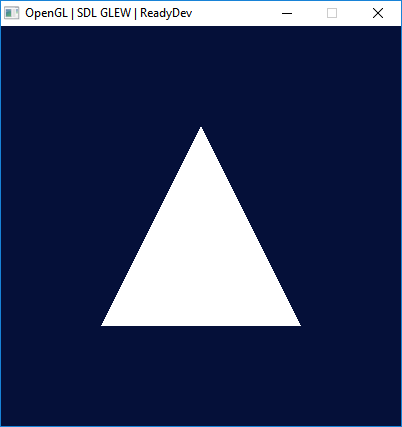
**Implémentation :**

Créer un gestionnaire de fenêtre (GWindow).

Créer un gestionnaire de dessin (GDraw).

Dessiner un objet dans la fenêtre (draw()).

**Résultat :**



**Dossier projet :**

src/main.cpp

src/manager/GConfig.cpp

src/manager/GWindow.cpp

src/manager/GWindowSDL.cpp

src/manager/GDraw.cpp

src/manager/GDrawSDL.cpp

**Programme principal :**

//===============================================

#include "GWindow.h"

#include "GConfig.h"

//===============================================

int main**(**int argc**,** char**\*\*** argv**)** **{**

GConfig**::**Instance**()->**setData**(**"WINDOW\_TYPE"**,** "OPENGL\_SDL\_GLEW"**);**

GConfig**::**Instance**()->**setData**(**"BACKGROUND\_FLAG"**,** "TRUE"**);**

GConfig**::**Instance**()->**setData**(**"BACKGROUND\_COLOR"**,** "5;16;57;255"**);**

GConfig**::**Instance**()->**setData**(**"DRAW\_FLAG"**,** "TRUE"**);**

GWindow**::**Instance**()->**show**(&**argc**,** argv**);**

**return** 0**;**

**}**

//===============================================

**Affichage de la fenêtre :**

//===============================================

void GWindowSDL**::**show**(**int**\*** argc**,** char**\*\*** argv**)** **{**

SDL\_Init**(**SDL\_INIT\_VIDEO**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_CONTEXT\_MAJOR\_VERSION**,** 3**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_CONTEXT\_MINOR\_VERSION**,** 1**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_DOUBLEBUFFER**,** 1**);**

SDL\_GL\_SetAttribute**(**SDL\_GL\_DEPTH\_SIZE**,** 24**);**

m\_window **=** SDL\_CreateWindow**(**m\_title**.**c\_str**(),** m\_x**,** m\_y**,** m\_w**,** m\_h**,** SDL\_WINDOW\_SHOWN **|** SDL\_WINDOW\_OPENGL**);**

SDL\_GLContext m\_context **=** SDL\_GL\_CreateContext**(**m\_window**);**

glewInit**();**

setVertices**();**

SDL\_Event m\_event**;**

**while(**1**)** **{**

SDL\_WaitEvent**(&**m\_event**);**

**if(**m\_event**.**window**.**event **==** SDL\_WINDOWEVENT\_CLOSE**)** **break;**

setBackground**();**

glClear**(**GL\_COLOR\_BUFFER\_BIT**);**

draw**();**

SDL\_GL\_SwapWindow**(**m\_window**);**

**}**

SDL\_GL\_DeleteContext**(**m\_context**);**

SDL\_DestroyWindow**(**m\_window**);**

SDL\_Quit**();**

**}**

//===============================================

**Initialisation des vertices :**

//===============================================

void GDrawSDL**::**setVertices**()** **{**

float m\_vertices**[]** **=** **{-**0.5**,** **-**0.5**,** 0.0**,** 0.5**,** 0.5**,** **-**0.5**};**

int m\_size **=** **sizeof(**m\_vertices**)/sizeof(**float**);**

**for(**int i **=** 0**;** i **<** m\_size**;** i**++)** **{**

m\_verticesMap**[**i**]** **=** m\_vertices**[**i**];**

**}**

**}**

//===============================================

**Dessin de l’objet dans la fenêtre :**

//===============================================

void GDrawSDL**::**draw**()** **{**

glVertexAttribPointer**(**0**,** 2**,** GL\_FLOAT**,** GL\_FALSE**,** 0**,** m\_verticesMap**);**

glEnableVertexAttribArray**(**0**);**

glDrawArrays**(**GL\_TRIANGLES**,** 0**,** 3**);**

glDisableVertexAttribArray**(**0**);**

**}**

//===============================================