# Sommaire

[Sommaire 1](#_Toc151538241)

[Introduction 3](#_Toc151538242)

[Recommandations techniques 3](#_Toc151538243)

[Création du programme principal 3](#_Toc151538244)

[Chargement de la fenêtre principale 3](#_Toc151538245)

[Affichage de l’emplacement de la base de données SQLite 3](#_Toc151538246)

[Modification de l’emplacement de la base de données SQLite 4](#_Toc151538247)

[Création de la fenêtre principale 5](#_Toc151538248)

[Initialisation de la fenêtre principale 5](#_Toc151538249)

[Importation d’un module JS 6](#_Toc151538250)

[Définition des propriétés couleurs 6](#_Toc151538251)

[Définition de la propriété base de données 7](#_Toc151538252)

[Chargement de police personnalisée 7](#_Toc151538253)

[Initialisation de la pile de vues graphiques 8](#_Toc151538254)

[Récupération du signal d’initialisation de la fenêtre 8](#_Toc151538255)

[Chargement de la page de connexion 9](#_Toc151538256)

[Appel d’une fonction JS 10](#_Toc151538257)

[Création d’un pop-up 11](#_Toc151538258)

[Gestion de la connexion d’un utilisateur 13](#_Toc151538259)

[Création de la page de connexion 16](#_Toc151538260)

[Initialisation de la page 16](#_Toc151538261)

[Création du signal d’inscription 16](#_Toc151538262)

[Création du fond de la page 16](#_Toc151538263)

[Création du logo 17](#_Toc151538264)

[Création du champ du nom d’utilisateur 19](#_Toc151538265)

[Création du champ du mot de passe 21](#_Toc151538266)

[Création du bouton de connexion 24](#_Toc151538267)

[Gestion de la connexion d’un utilisateur 27](#_Toc151538268)

[Création du bouton d’inscription 30](#_Toc151538269)

[Gestion de l’inscription d’un utilisateur 32](#_Toc151538270)

[Création de la page d’inscription 35](#_Toc151538271)

[Initialisation de la page 35](#_Toc151538272)

[Création des propriétés de la page 36](#_Toc151538273)

[Création du fond de la page 36](#_Toc151538274)

[Création du titre de la page 37](#_Toc151538275)

[Création de l’interface graphique de la page 39](#_Toc151538276)

[Gestion de l’annulation de l’inscription 43](#_Toc151538277)

[Gestion de l’inscription d’un utilisateur 43](#_Toc151538278)

[Création du module Backend JS 45](#_Toc151538279)

[Création de la variable de gestion des erreurs 45](#_Toc151538280)

[Création de la variable de gestion de la base de données 45](#_Toc151538281)

[Gestion des erreurs 45](#_Toc151538282)

[Gestion de la base de données 46](#_Toc151538283)

[Gestion des utilisateurs 47](#_Toc151538284)

[Création d’un bouton personnalisé 48](#_Toc151538285)

[Initialisation du bouton 49](#_Toc151538286)

[Création de la propriété alias du nom 49](#_Toc151538287)

[Création des propriétés des couleurs 50](#_Toc151538288)

[Gestion du texte du bouton 50](#_Toc151538289)

[Gestion du fond et de la bordure du bouton 50](#_Toc151538290)

[Création d’un champ de saisie 51](#_Toc151538291)

[Initialisation du champ 51](#_Toc151538292)

[Création d’un pop-up personnalisé 51](#_Toc151538293)

[Initialisation du pop-up 52](#_Toc151538294)

[Création du fond du pop-up 52](#_Toc151538295)

[Création du texte du pop-up 53](#_Toc151538296)

[Création de la propriété alias du texte 53](#_Toc151538297)

[Fermeture du pop-up après un certain temps 54](#_Toc151538298)

[Création d’un timer 54](#_Toc151538299)

[Création du timer 54](#_Toc151538300)

[Démarrage du timer 55](#_Toc151538301)

[Utilisation de FontAwesome 55](#_Toc151538302)

[Référence complète des icônes FontAwesome 55](#_Toc151538303)

[Utilisation de scripts Batch 56](#_Toc151538304)

[Affichage de l’arbre des répertoires et fichiers 56](#_Toc151538305)

# Introduction

Le but de ce tutoriel est de vous apprendre à créer une application mobile Android - iOS en C++ - Qt – QML - JS permettant la gestion d’un système de connexion et d’inscription de nouveaux utilisateurs.

# Recommandations techniques

Pour tester les extraits de code dans ce tutoriel, vous aurez besoin des éléments suivants:

* Une bibliothèque d’interface graphique Qt:

<https://www.qt.io/>

* Un environnement de développement intégré Android Studio:

<https://developer.android.com/studio>

# Création du programme principal

## Chargement de la fenêtre principale

Programme C++:

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

// main.cpp

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

int **main**(int **\_argc**, char\*\* **\_argv**) {

QGuiApplication **lApp**(*\_argc*, *\_argv*);

QQmlApplicationEngine **engine**;

const QUrl **url**("qrc:/qml/main.qml");

QObject::connect(&engine, &QQmlApplicationEngine::objectCreationFailed,

&lApp, []() { QCoreApplication::exit(-1); },

Qt::QueuedConnection);

engine.load(url);

return lApp.exec();

}

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

## Affichage de l’emplacement de la base de données SQLite

Résultat:



Programme C++:

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

// main.cpp

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

int **main**(int **\_argc**, char\*\* **\_argv**) {

QGuiApplication **lApp**(*\_argc*, *\_argv*);

QQmlApplicationEngine **lEngine**;

const QUrl **lUrl**("qrc:/qml/main.qml");

QObject::connect(&lEngine, &QQmlApplicationEngine::objectCreationFailed,

&lApp, []() { QCoreApplication::exit(-1); },

Qt::QueuedConnection);

lEngine.load(lUrl);

qDebug() << lEngine.offlineStoragePath();

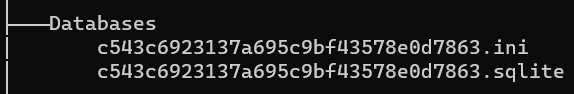
return lApp.exec();

}

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

## Modification de l’emplacement de la base de données SQLite

Résultat:



Contenu du fichier INI:

#================================================

# Fichier.ini

#================================================

**[General]**

Name**=**db\_readydev.dat

Version**=**1.0

Description**=**Base de données de ReadyDev

EstimatedSize**=**1000000

Driver**=**QSQLITE

#================================================

Programme C++:

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

// main.cpp

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

int **main**(int **\_argc**, char\*\* **\_argv**) {

QGuiApplication **lApp**(*\_argc*, *\_argv*);

QQmlApplicationEngine **lEngine**;

const QUrl **lUrl**("qrc:/qml/main.qml");

QObject::connect(&lEngine, &QQmlApplicationEngine::objectCreationFailed,

&lApp, []() { QCoreApplication::exit(-1); },

Qt::QueuedConnection);

lEngine.setOfflineStoragePath(".");

lEngine.load(lUrl);

qDebug() << lEngine.offlineStoragePath();

return lApp.exec();

}

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

# Création de la fenêtre principale

## Initialisation de la fenêtre principale

Résultat:



Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

}

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

## Importation d’un module JS

Résultat:



Programme QML:

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

// main.qml

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

import "qrc:/js/GBackend.js" as GBackend

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

}

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

Programme JS:

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

// GBackend.js

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

*console*.log("GBackend.js...")

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

## Définition des propriétés couleurs

Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

property color backGroundColor : "#394454"

property color mainAppColor: "#6fda9c"

property color mainTextCOlor: "#f0f0f0"

property color popupBackGroundColor: "#b44"

property color popupTextCOlor: "#ffffff"

}

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

## Définition de la propriété base de données

Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

property color backGroundColor : "#394454"

property color mainAppColor: "#6fda9c"

property color mainTextCOlor: "#f0f0f0"

property color popupBackGroundColor: "#b44"

property color popupTextCOlor: "#ffffff"

property var dataBase

}

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

## Chargement de police personnalisée

Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

property color backGroundColor : "#394454"

property color mainAppColor: "#6fda9c"

property color mainTextCOlor: "#f0f0f0"

property color popupBackGroundColor: "#b44"

property color popupTextCOlor: "#ffffff"

property var dataBase

FontLoader {

id: *fontAwesome*

source: "qrc:/font/fontawesome-webfont.ttf"

}

}

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

## Initialisation de la pile de vues graphiques

Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

property color backGroundColor : "#394454"

property color mainAppColor: "#6fda9c"

property color mainTextCOlor: "#f0f0f0"

property color popupBackGroundColor: "#b44"

property color popupTextCOlor: "#ffffff"

property var dataBase

FontLoader {

id: *fontAwesome*

source: "qrc:/font/fontawesome-webfont.ttf"

}

StackView {

id: *stackView*

focus: true

anchors.fill: *parent*

}

}

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

## Récupération du signal d’initialisation de la fenêtre

Résultat:



Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

property color backGroundColor : "#394454"

property color mainAppColor: "#6fda9c"

property color mainTextCOlor: "#f0f0f0"

property color popupBackGroundColor: "#b44"

property color popupTextCOlor: "#ffffff"

property var dataBase

FontLoader {

id: *fontAwesome*

source: "qrc:/font/fontawesome-webfont.ttf"

}

StackView {

id: *stackView*

focus: true

anchors.fill: *parent*

}

Component.onCompleted: {

*console*.log("Component.onCompleted...")

}

}

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

## Chargement de la page de connexion

Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

property color backGroundColor : "#394454"

property color mainAppColor: "#6fda9c"

property color mainTextCOlor: "#f0f0f0"

property color popupBackGroundColor: "#b44"

property color popupTextCOlor: "#ffffff"

property var dataBase

FontLoader {

id: *fontAwesome*

source: "qrc:/font/fontawesome-webfont.ttf"

}

StackView {

id: *stackView*

focus: true

anchors.fill: *parent*

}

Component.onCompleted: {

*stackView*.push("qrc:/qml/GLoginUi.qml")

}

}

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

// GLoginUi.qml

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

Page {

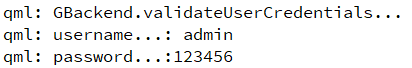
id: *loginPage*

}

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

## Appel d’une fonction JS

Résultat:



Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

property color backGroundColor : "#394454"

property color mainAppColor: "#6fda9c"

property color mainTextCOlor: "#f0f0f0"

property color popupBackGroundColor: "#b44"

property color popupTextCOlor: "#ffffff"

property var dataBase

FontLoader {

id: *fontAwesome*

source: "qrc:/font/fontawesome-webfont.ttf"

}

StackView {

id: *stackView*

focus: true

anchors.fill: *parent*

}

Component.onCompleted: {

*stackView*.push("qrc:/qml/GLoginUi.qml")

}

function *loginUser*(\_username, \_password) {

*GBackend*.validateUserCredentials(*\_username*, *\_password*)

}

}

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

Programme JS:

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

// GBackend.js

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

import "qrc:/js/GBackend.js" as *GBackend*

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

function *validateUserCredentials*(\_username, \_password) {

*console*.log("GBackend.validateUserCredentials...")

*console*.log("username...: " + *\_username*)

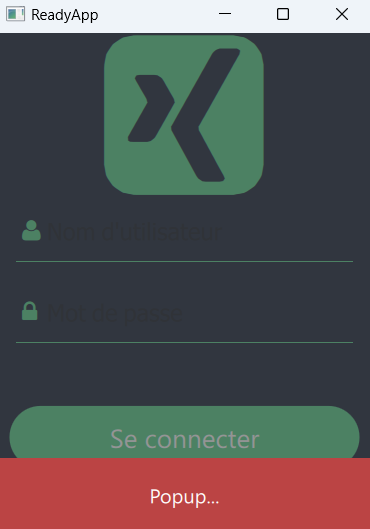
*console*.log("password...:" + *\_password*)

}

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

## Création d’un pop-up

Résultat:



Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

property color backGroundColor : "#394454"

property color mainAppColor: "#6fda9c"

property color mainTextCOlor: "#f0f0f0"

property color popupBackGroundColor: "#b44"

property color popupTextCOlor: "#ffffff"

property var dataBase

FontLoader {

id: *fontAwesome*

source: "qrc:/font/fontawesome-webfont.ttf"

}

StackView {

id: *stackView*

focus: true

anchors.fill: *parent*

}

Component.onCompleted: {

*stackView*.push("qrc:/qml/GLoginUi.qml")

}

Popup {

id: *popup*

property alias popMessage: *message*.text

background: Rectangle {

implicitWidth: *rootWindow*.width

implicitHeight: 60

color: *popupBackGroundColor*

}

y: (*rootWindow*.height - 60)

modal: true

focus: true

closePolicy: Popup.CloseOnPressOutside

Text {

id: *message*

anchors.centerIn: *parent*

font.pointSize: 12

color: *popupTextCOlor*

}

onOpened: popupClose.start()

}

Timer {

id: *popupClose*

interval: 2000

onTriggered: *popup*.close()

}

function *loginUser*(\_username, \_password) {

*popup*.popMessage = "Popup..."

*popup*.open()

}

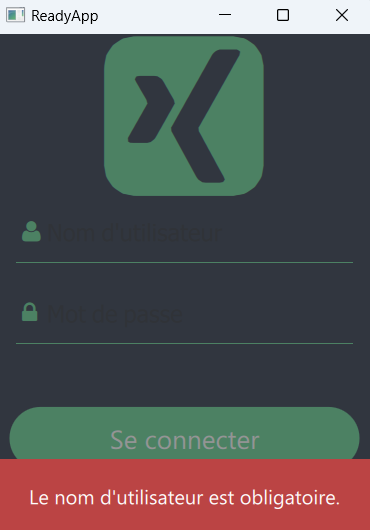
}

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

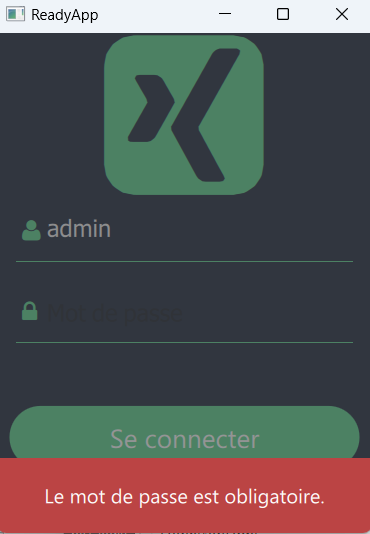
## Gestion de la connexion d’un utilisateur

Résultat:

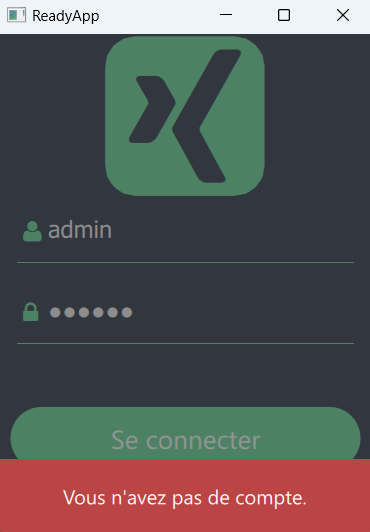
Résultat du nom d’utilisateur vide:



Résultat du mot de passe vide:



Résultat du nom d’utilisateur incorrect:



Résultat du mot de passe incorrect:

Programme QML:

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 400

visible: true

title: *qsTr*("ReadyApp")

property color backGroundColor : "#394454"

property color mainAppColor: "#6fda9c"

property color mainTextCOlor: "#f0f0f0"

property color popupBackGroundColor: "#b44"

property color popupTextCOlor: "#ffffff"

property var dataBase

FontLoader {

id: *fontAwesome*

source: "qrc:/font/fontawesome-webfont.ttf"

}

StackView {

id: *stackView*

focus: true

anchors.fill: *parent*

}

Component.onCompleted: {

*stackView*.push("qrc:/qml/GLoginUi.qml")

}

Popup {

id: *popup*

property alias popMessage: *message*.text

background: Rectangle {

implicitWidth: *rootWindow*.width

implicitHeight: 60

color: *popupBackGroundColor*

}

y: (*rootWindow*.height - 60)

modal: true

focus: true

closePolicy: Popup.CloseOnPressOutside

Text {

id: *message*

anchors.centerIn: *parent*

font.pointSize: 12

color: *popupTextCOlor*

}

onOpened: popupClose.start()

}

function *loginUser*(\_username, \_password) {

if(!*GBackend*.validateUserCredentials(*\_username*, *\_password*)) {

*popup*.popMessage = *GBackend*.getErrors()

*popup*.open()

return

}

}

}

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

# Création de la page de connexion

## Initialisation de la page

Programme QML:

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

// GLoginUi.qml

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

Page {

id: *loginPage*

}

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

## Création du signal d’inscription

Programme QML:

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

// GLoginUi.qml

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

Page {

id: *loginPage*

signal registerClicked()

}

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

## Création du fond de la page

Résultat:



Programme QML:

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

// GLoginUi.qml

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

Page {

id: *loginPage*

signal registerClicked()

background: Rectangle {

color: backGroundColor

}

}

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

## Création du logo

Résultat:



Programme QML:

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

// GLoginUi.qml

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

Page {

id: *loginPage*

signal registerClicked()

background: Rectangle {

color: backGroundColor

}

Rectangle {

id: *iconRect*

width: *parent*.width

height: *parent*.height / 3

color: backGroundColor

Text {

id: *icontext*

text: *qsTr*("\uf169")

anchors.centerIn: *parent*

font.pointSize: 112

font.family: "fontawesome"

color: mainAppColor

}

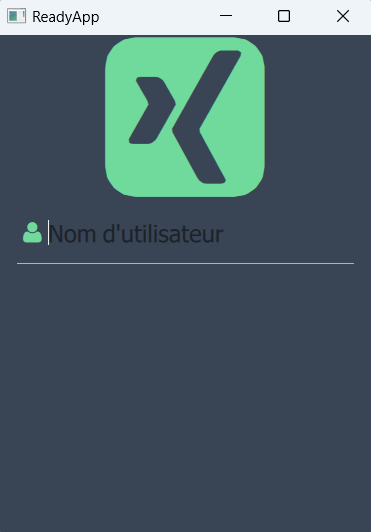
}

}

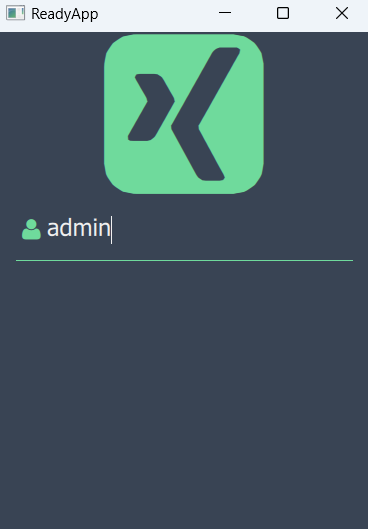
//===============================================

## Création du champ du nom d’utilisateur

Résultat:



Résultat de saisie du nom d’utilisateur:



Programme QML:

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

// GLoginUi.qml

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

Page {

id: *loginPage*

signal registerClicked()

background: Rectangle {

color: backGroundColor

}

Rectangle {

id: *iconRect*

width: *parent*.width

height: *parent*.height / 3

color: backGroundColor

Text {

id: *icontext*

text: *qsTr*("\uf169")

anchors.centerIn: *parent*

font.pointSize: 112

font.family: "fontawesome"

color: mainAppColor

}

}

ColumnLayout {

width: *parent*.width

anchors.top: *iconRect*.bottom

spacing: 15

TextField {

id: *loginUsername*

placeholderText: *qsTr*("Nom d'utilisateur")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf007"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

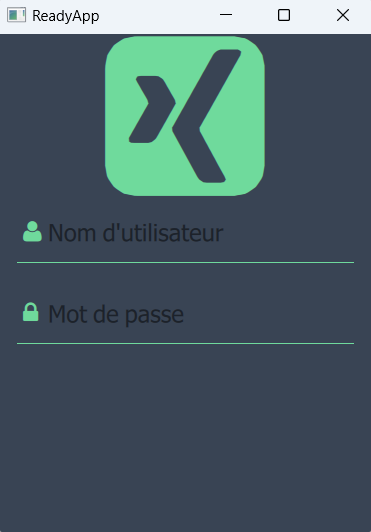
}

}

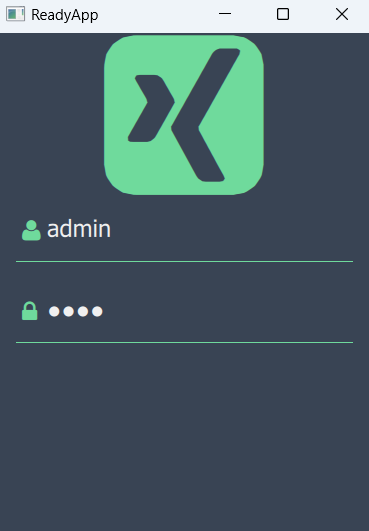
//===============================================

## Création du champ du mot de passe

Résultat:



Résultat de saisie du mot de passe:



Programme QML:

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

// GLoginUi.qml

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

Page {

id: *loginPage*

signal registerClicked()

background: Rectangle {

color: backGroundColor

}

Rectangle {

id: *iconRect*

width: *parent*.width

height: *parent*.height / 3

color: backGroundColor

Text {

id: *icontext*

text: *qsTr*("\uf169")

anchors.centerIn: *parent*

font.pointSize: 112

font.family: "fontawesome"

color: mainAppColor

}

}

ColumnLayout {

width: *parent*.width

anchors.top: *iconRect*.bottom

spacing: 15

TextField {

id: *loginUsername*

placeholderText: *qsTr*("Nom d'utilisateur")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf007"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

TextField {

id: *loginPassword*

placeholderText: *qsTr*("Mot de passe")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

echoMode: TextField.Password

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf023"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

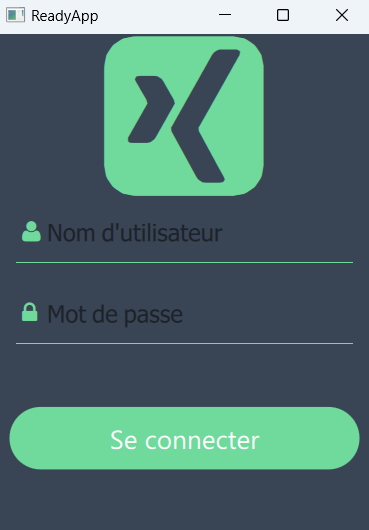
}

}

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

## Création du bouton de connexion

Résultat:



Programme QML:

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

// GLoginUi.qml

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

Page {

id: *loginPage*

signal registerClicked()

background: Rectangle {

color: backGroundColor

}

Rectangle {

id: *iconRect*

width: *parent*.width

height: *parent*.height / 3

color: backGroundColor

Text {

id: *icontext*

text: *qsTr*("\uf169")

anchors.centerIn: *parent*

font.pointSize: 112

font.family: "fontawesome"

color: mainAppColor

}

}

ColumnLayout {

width: *parent*.width

anchors.top: *iconRect*.bottom

spacing: 15

TextField {

id: *loginUsername*

placeholderText: *qsTr*("Nom d'utilisateur")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf007"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

TextField {

id: *loginPassword*

placeholderText: *qsTr*("Mot de passe")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

echoMode: TextField.Password

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf023"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

Item {

height: 20

}

GButtonUi {

height: 50

Layout.preferredWidth: *loginPage*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "Se connecter"

baseColor: mainAppColor

borderColor: mainAppColor

}

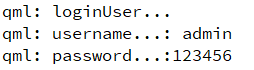
}

}

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

## Gestion de la connexion d’un utilisateur

Résultat:



Programme QML:

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

// GLoginUi.qml

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

Page {

id: *loginPage*

signal registerClicked()

background: Rectangle {

color: backGroundColor

}

Rectangle {

id: *iconRect*

width: *parent*.width

height: *parent*.height / 3

color: backGroundColor

Text {

id: *icontext*

text: *qsTr*("\uf169")

anchors.centerIn: *parent*

font.pointSize: 112

font.family: "fontawesome"

color: mainAppColor

}

}

ColumnLayout {

width: *parent*.width

anchors.top: *iconRect*.bottom

spacing: 15

TextField {

id: *loginUsername*

placeholderText: *qsTr*("Nom d'utilisateur")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf007"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

TextField {

id: *loginPassword*

placeholderText: *qsTr*("Mot de passe")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

echoMode: TextField.Password

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf023"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

Item {

height: 20

}

GButtonUi {

height: 50

Layout.preferredWidth: *loginPage*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "Se connecter"

baseColor: mainAppColor

borderColor: mainAppColor

onClicked: {

loginUser(*loginUsername*.text, *loginPassword*.text)

}

}

}

}

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

// main.qml

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

ApplicationWindow {

function *loginUser*(\_username, \_password) {

*console*.log("loginUser...")

*console*.log("username...: " + *\_username*)

*console*.log("password...:" + *\_password*)

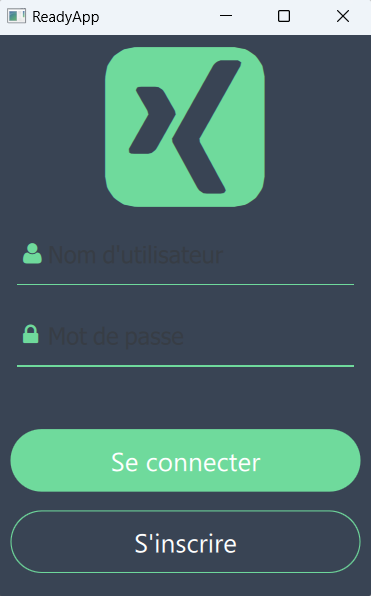
}

}

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

## Création du bouton d’inscription

Résultat:



Programme QML:

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

// GLoginUi.qml

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

Page {

id: *loginPage*

signal registerClicked()

background: Rectangle {

color: backGroundColor

}

Rectangle {

id: *iconRect*

width: *parent*.width

height: *parent*.height / 3

color: backGroundColor

Text {

id: *icontext*

text: *qsTr*("\uf169")

anchors.centerIn: *parent*

font.pointSize: 112

font.family: "fontawesome"

color: mainAppColor

}

}

ColumnLayout {

width: *parent*.width

anchors.top: *iconRect*.bottom

spacing: 15

TextField {

id: *loginUsername*

placeholderText: *qsTr*("Nom d'utilisateur")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf007"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

TextField {

id: *loginPassword*

placeholderText: *qsTr*("Mot de passe")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

echoMode: TextField.Password

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf023"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

Item {

height: 20

}

GButtonUi {

height: 50

Layout.preferredWidth: *loginPage*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "Se connecter"

baseColor: mainAppColor

borderColor: mainAppColor

onClicked: {

loginUser(*loginUsername*.text, *loginPassword*.text)

dataBase

}

}

GButtonUi {

height: 50

Layout.preferredWidth: *loginPage*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "S'inscrire"

baseColor: "transparent"

borderColor: mainAppColor

}

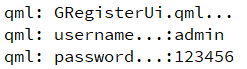
}

}

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

## Gestion de l’inscription d’un utilisateur

Résultat:



Programme QML:

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

// GRegisterUi.qml

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

Page {

id: *loginPage*

signal registerClicked()

background: Rectangle {

color: backGroundColor

}

Rectangle {

id: *iconRect*

width: *parent*.width

height: *parent*.height / 3

color: backGroundColor

Text {

id: *icontext*

text: *qsTr*("\uf169")

anchors.centerIn: *parent*

font.pointSize: 112

font.family: "fontawesome"

color: mainAppColor

}

}

ColumnLayout {

width: *parent*.width

anchors.top: *iconRect*.bottom

spacing: 15

TextField {

id: *loginUsername*

placeholderText: *qsTr*("Nom d'utilisateur")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf007"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

TextField {

id: *loginPassword*

placeholderText: *qsTr*("Mot de passe")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

echoMode: TextField.Password

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf023"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

Item {

height: 20

}

GButtonUi {

height: 50

Layout.preferredWidth: *loginPage*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "Se connecter"

baseColor: mainAppColor

borderColor: mainAppColor

onClicked: {

loginUser(*loginUsername*.text, *loginPassword*.text)

dataBase

}

}

GButtonUi {

height: 50

Layout.preferredWidth: *loginPage*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "S'inscrire"

baseColor: "transparent"

borderColor: mainAppColor

onClicked: {

stackView.push("qrc:/qml/GRegisterUi.qml", {"username": "admin", "password": "123456"})

}

}

}

}

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

// GRegisterUi.qml

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

Page {

id: *registerPage*

property string username: ""

property string password: ""

Component.onCompleted: {

*console*.log("GRegisterUi.qml...")

*console*.log("username...:" + *username*)

*console*.log("password...:" + *password*)

}

}

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

# Création de la page d’inscription

## Initialisation de la page

Programme QML:

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

// GRegisterUi.qml

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

Page {

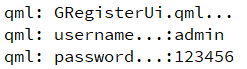
id: *registerPage*

}

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

## Création des propriétés de la page

Résultat:



Programme QML:

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

// GRegisterUi.qml

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

Page {

id: *registerPage*

property string username: ""

property string password: ""

Component.onCompleted: {

*console*.log("GRegisterUi.qml...")

*console*.log("username...:" + *username*)

*console*.log("password...:" + *password*)

}

}

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

## Création du fond de la page

Résultat:



Programme QML:

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

// GRegisterUi.qml

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

Page {

id: *registerPage*

property string username: ""

property string password: ""

background: Rectangle {

color: backGroundColor

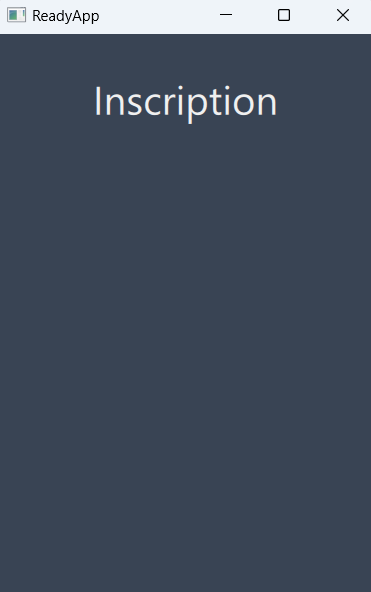
}

}

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

## Création du titre de la page

Résultat:



Programme QML:

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

// GRegisterUi.qml

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

Page {

id: *registerPage*

property string username: ""

property string password: ""

background: Rectangle {

color: backGroundColor

}

Text {

id: *signupText*

text: *qsTr*("Inscription")

font.pointSize: 24

anchors.top: *parent*.top

anchors.topMargin: 30

anchors.horizontalCenter: *parent*.horizontalCenter

color: mainTextCOlor

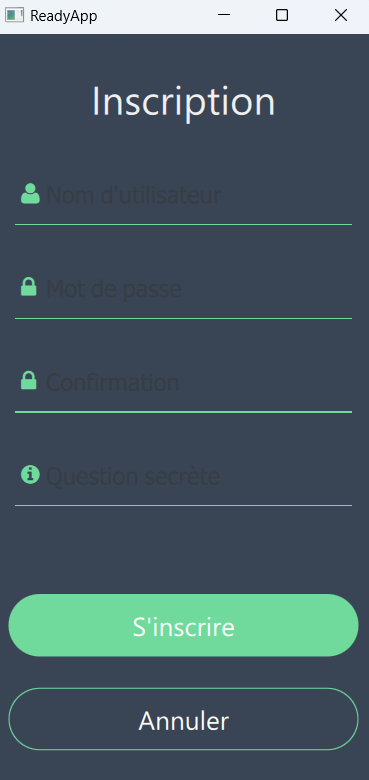
}

}

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

## Création de l’interface graphique de la page

Résultat:



Programme QML:

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

// GRegisterUi.qml

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

Page {

id: *registerPage*

property string username: ""

property string password: ""

background: Rectangle {

color: backGroundColor

}

Text {

id: *signupText*

text: *qsTr*("Inscription")

font.pointSize: 24

anchors.top: *parent*.top

anchors.topMargin: 30

anchors.horizontalCenter: *parent*.horizontalCenter

color: mainTextCOlor

}

ColumnLayout {

width: *parent*.width

anchors.top: *signupText*.bottom

anchors.topMargin: 30

spacing: 25

TextField {

id: *registerUsername*

placeholderText: *qsTr*("Nom d'utilisateur")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf007"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

TextField {

id: *registerPassword*

placeholderText: *qsTr*("Mot de passe")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

echoMode: TextField.PasswordEchoOnEdit

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf023"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

TextField {

id: *registerPassword2*

placeholderText: *qsTr*("Confirmation")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

echoMode: TextField.PasswordEchoOnEdit

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf023"

font.pointSize: 14

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

TextField {

id: *passwordHint*

placeholderText: *qsTr*("Question secrète")

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

color: mainTextCOlor

font.pointSize: 14

font.family: "fontawesome"

leftPadding: 30

background: Rectangle {

implicitWidth: 200

implicitHeight: 50

radius: *implicitHeight* / 2

color: "transparent"

Text {

text: "\uf05a"

font.pointSize: 13

font.bold: true

font.family: "fontawesome"

color: mainAppColor

anchors.left: *parent*.left

anchors.verticalCenter: *parent*.verticalCenter

leftPadding: 10

}

Rectangle {

width: *parent*.width - 10

height: 1

anchors.horizontalCenter: *parent*.horizontalCenter

anchors.bottom: *parent*.bottom

color: mainAppColor

}

}

}

Item {

height: 20

}

GButtonUi {

height: 50

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "S'inscrire"

baseColor: mainAppColor

borderColor: mainAppColor

}

GButtonUi {

height: 50

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "Annuler"

baseColor: "transparent"

borderColor: mainAppColor

}

}

}

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

## Gestion de l’annulation de l’inscription

Programme QML:

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

// GRegisterUi.qml

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

Page {

id: *registerPage*

property string username: ""

property string password: ""

ColumnLayout {

width: *parent*.width

anchors.top: *signupText*.bottom

anchors.topMargin: 30

spacing: 25

GButtonUi {

height: 50

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "Annuler"

baseColor: "transparent"

borderColor: mainAppColor

onClicked: stackView.pop()

}

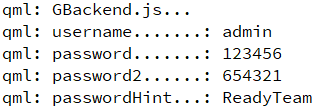
}

}

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

## Gestion de l’inscription d’un utilisateur

Résultat:



Programme QML:

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

// GRegisterUi.qml

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

Page {

id: *registerPage*

property string username: ""

property string password: ""

ColumnLayout {

width: *parent*.width

anchors.top: *signupText*.bottom

anchors.topMargin: 30

spacing: 25

GButtonUi {

height: 50

Layout.preferredWidth: *parent*.width - 20

Layout.alignment: Qt.AlignHCenter

name: "S'inscrire"

baseColor: mainAppColor

borderColor: mainAppColor

onClicked: onRegister(*registerUsername*.text, *registerPassword*.text, *registerPassword2*.text, *passwordHint*.text)

}

}

}

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

// main.qml

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

ApplicationWindow {

id: *rootWindow*

width: 300

height: 600

visible: true

title: *qsTr*("ReadyApp")

function *onRegister*(\_username, \_password, \_password2, \_passwordHint) {

if(!*GBackend*.onRegister(*\_username*, *\_password*, *\_password2*, *\_passwordHint*)) {

*popup*.popMessage = *GBackend*.getErrors()

*popup*.open()

return

}

}

}

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

Programme JS:

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

// GBackend.js

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

function *onRegister*(\_username, \_password, \_password2, \_passwordHint) {

*clearErrors*()

*console*.log("GBackend.js...")

*console*.log("username.......: " + *\_username*)

*console*.log("password.......: " + *\_password*)

*console*.log("password2......: " + *\_password2*)

*console*.log("passwordHint...: " + *\_passwordHint*)

return !*hasErrors*()

}

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

# Création du module Backend JS

## Création de la variable de gestion des erreurs

Programme JS:

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

// GBackend.js

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

var *m\_errors* = []

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

## Création de la variable de gestion de la base de données

Programme JS:

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

// GBackend.js

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

var *m\_errors* = []

var *m\_dbSQL* = null

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

## Gestion des erreurs

Programme JS:

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

// GBackend.js

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

var *m\_errors* = []

var *m\_dbSQL* = null

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

// errors

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

function *clearErrors*() {

*m\_errors* = []

}

function *addError*(\_msg) {

*m\_errors*.push(*\_msg*)

}

function *getErrors*() {

return *m\_errors*.join("\n")

}

function *hasErrors*() {

return (*m\_errors*.length !== 0)

}

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

## Gestion de la base de données

Programme JS:

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

// GBackend.js

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

var *m\_errors* = []

var *m\_dbSQL* = null

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

// errors

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

function *clearErrors*() {

*m\_errors* = []

}

function *addError*(\_msg) {

*m\_errors*.push(*\_msg*)

}

function *getErrors*() {

return *m\_errors*.join("\n")

}

function *hasErrors*() {

return (*m\_errors*.length !== 0)

}

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

// database

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

function *isOpenDatabase*() {

if(!*m\_dbSQL*) {

*addError*("Erreur 2 lors de la connexion au données.")

return false

}

return !*hasErrors*()

}

function *openDatabase*() {

*m\_dbSQL* = LocalStorage.openDatabaseSync("db\_readydev.dat", "1.0", "Base de données de ReadyDev", 1000000);

try {

*m\_dbSQL*.transaction(function(tx) {

*tx*.executeSql("create table if not exists \_users ( " +

"\_username text, " +

"\_password text " +

")")

})

}

catch(*e*) {

*addError*(*e*)

}

return !*hasErrors*()

}

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

## Gestion des utilisateurs

Programme JS:

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

// GBackend.js

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

var *m\_errors* = []

var *m\_dbSQL* = null

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

// errors

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

function *clearErrors*() {

*m\_errors* = []

}

function *addError*(\_msg) {

*m\_errors*.push(*\_msg*)

}

function *getErrors*() {

return *m\_errors*.join("\n")

}

function *hasErrors*() {

return (*m\_errors*.length !== 0)

}

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

// database

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

function *isOpenDatabase*() {

if(!*m\_dbSQL*) {

*addError*("Erreur 2 lors de la connexion au données.")

return false

}

return !*hasErrors*()

}

function *openDatabase*() {

*m\_dbSQL* = LocalStorage.openDatabaseSync("db\_readydev.dat", "1.0", "Base de données de ReadyDev", 1000000);

try {

*m\_dbSQL*.transaction(function(tx) {

*tx*.executeSql("create table if not exists \_users ( " +

"\_username text, " +

"\_password text " +

")")

})

}

catch(*e*) {

*addError*(*e*)

}

return !*hasErrors*()

}

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

// users

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

function *isLoginUser*(\_username, \_password) {

*clearErrors*()

if(*\_username* === "") {

*addError*("Le nom d'utilisateur est obligatoire.")

return false;

}

if(*\_password* === "") {

*addError*("Le mot de passe est obligatoire.")

return false;

}

try {

*m\_dbSQL*.transaction(function(tx) {

var *lResults* = *tx*.executeSql("" +

"select \_password from \_users " +

"where \_username = ? " +

"", *\_username*)

if(*lResults*.rows.length === 0) {

*addError*("Vous n'avez pas de compte.")

}

else if(*lResults*.rows.item(0).\_password !== *\_password*) {

*addError*("Le mot de passe est incorrect.")

}

})

}

catch(*e*) {

*addError*(*e*)

}

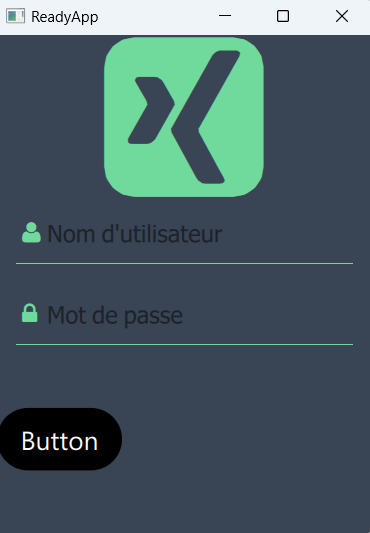
return !*hasErrors*()

}

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

# Création d’un bouton personnalisé

Résultat:



## Initialisation du bouton

Programme QML:

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

// GButtonUi.qml

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

Button {

id: *control*

text: *qsTr*("Button")

font.pointSize: 16

}

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

## Création de la propriété alias du nom

Programme QML:

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

// GButtonUi.qml

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

Button {

id: *control*

text: *qsTr*("Button")

font.pointSize: 16

property alias name: *control*.text

}

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

## Création des propriétés des couleurs

Programme QML:

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

// GButtonUi.qml

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

Button {

id: *control*

text: *qsTr*("Button")

font.pointSize: 16

property alias name: *control*.text

property color baseColor

property color borderColor

}

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

## Gestion du texte du bouton

Programme QML:

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

// GButtonUi.qml

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

Button {

id: *control*

text: *qsTr*("Button")

font.pointSize: 16

property alias name: *control*.text

property color baseColor

property color borderColor

contentItem: Text {

text: *control*.text

font: *control*.font

opacity: *enabled* ? 1.0 : 0.3

color: *control*.down ? "#ffffff" : "#ffffff"

horizontalAlignment: Text.AlignHCenter

verticalAlignment: Text.AlignVCenter

elide: Text.ElideRight

}

}

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

## Gestion du fond et de la bordure du bouton

Programme QML:

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

// GButtonUi.qml

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

Button {

id: *control*

text: *qsTr*("Button")

font.pointSize: 16

property alias name: *control*.text

property color baseColor

property color borderColor

contentItem: Text {

text: *control*.text

font: *control*.font

opacity: *enabled* ? 1.0 : 0.3

color: *control*.down ? "#ffffff" : "#ffffff"

horizontalAlignment: Text.AlignHCenter

verticalAlignment: Text.AlignVCenter

elide: Text.ElideRight

}

background: Rectangle {

id: *bgrect*

implicitWidth: 100

implicitHeight: 50

color: *baseColor*

opacity: *control*.down ? 0.7 : 1

radius: *height*/2

border.color: *borderColor*

}

}

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

# Création d’un champ de saisie

## Initialisation du champ

Programme QML:

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

// GLineEdit.qml

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

TextField {

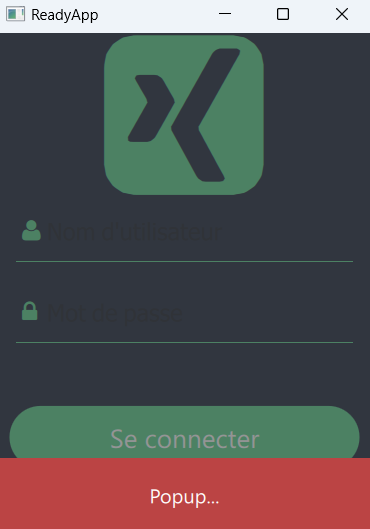
id: *control*

}

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

# Création d’un pop-up personnalisé

Résultat:



## Initialisation du pop-up

Programme QML:

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

// GPopupUi.qml

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

Popup {

id: *popup*

}

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

## Création du fond du pop-up

Programme QML:

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

// main.qml

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

Popup {

id: *popup*

background: Rectangle {

implicitWidth: *rootWindow*.width

implicitHeight: 60

color: *popupBackGroundColor*

}

}

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

## Création du texte du pop-up

Programme QML:

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

// main.qml

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

Popup {

id: *popup*

background: Rectangle {

implicitWidth: *rootWindow*.width

implicitHeight: 60

color: *popupBackGroundColor*

}

y: (*rootWindow*.height - 60)

modal: true

focus: true

closePolicy: Popup.CloseOnPressOutside

Text {

id: *message*

anchors.centerIn: *parent*

font.pointSize: 12

color: *popupTextCOlor*

}

onOpened: popupClose.start()

}

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

## Création de la propriété alias du texte

Programme QML:

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

// main.qml

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

Popup {

id: *popup*

property alias popMessage: *message*.text

background: Rectangle {

implicitWidth: *rootWindow*.width

implicitHeight: 60

color: *popupBackGroundColor*

}

y: (*rootWindow*.height - 60)

modal: true

focus: true

closePolicy: Popup.CloseOnPressOutside

Text {

id: *message*

anchors.centerIn: *parent*

font.pointSize: 12

color: *popupTextCOlor*

}

onOpened: popupClose.start()

}

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

## Fermeture du pop-up après un certain temps

Programme QML:

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

// main.qml

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

Popup {

id: *popup*

property alias popMessage: *message*.text

background: Rectangle {

implicitWidth: *rootWindow*.width

implicitHeight: 60

color: *popupBackGroundColor*

}

y: (*rootWindow*.height - 60)

modal: true

focus: true

closePolicy: Popup.CloseOnPressOutside

Text {

id: *message*

anchors.centerIn: *parent*

font.pointSize: 12

color: *popupTextCOlor*

}

onOpened: *popupClose*.start()

}

Timer {

id: *popupClose*

interval: 2000

onTriggered: *popup*.close()

}

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

# Création d’un timer

## Création du timer

Programme QML:

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

// main.qml

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

Timer {

id: *popupClose*

interval: 2000

onTriggered: *popup*.close()

}

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

## Démarrage du timer

Programme QML:

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

// main.qml

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

Popup {

id: *popup*

onOpened: *popupClose*.start()

}

Timer {

id: *popupClose*

interval: 2000

onTriggered: *popup*.close()

}

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

# Utilisation de FontAwesome

## Référence complète des icônes FontAwesome

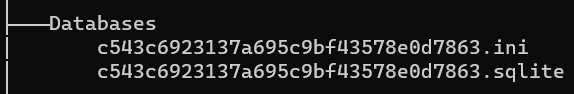
<https://fontawesome.com/v4/cheatsheet/>



# Utilisation de scripts Batch

## Affichage de l’arbre des répertoires et fichiers

Résultat:



Programme Batch:

::===============================================

:: Terminal

::===============================================

tree /f

::===============================================