JUCE — Exemple d'apprentissage : Slider + Label

Ce document explique comment créer une première application graphique avec **JUCE**, contenant :

- un **Slider** (curseur),
- un **Label** qui affiche la valeur du slider,
- un composant simple pour apprendre la structure d'un projet JUCE moderne.

Le tout est basé sur CMake.

---

1. Structure du projet

Créer un dossier de projet :

```
mkdir JuceSliderDemo
cd JuceSliderDemo
mkdir Source
```

Contenu final :

```
JuceSliderDemo/
 ├── CMakeLists.txt
 └── Source/
      ├── Main.cpp
      ├── MainComponent.h
      └── MainComponent.cpp
```

---

2. Fichier `CMakeLists.txt`

```
cmake_minimum_required(VERSION 3.15)

project(JuceSliderDemo VERSION 0.0.1)

# Modifier avec le bon chemin vers JUCE
add_subdirectory(/chemin/vers/JUCE JUCE)
```

```
juce_add_gui_app(JuceSliderDemo
    PRODUCT_NAME "Juce Slider Demo"
)

juce_generate_juce_header(JuceSliderDemo)

target_sources(JuceSliderDemo
    PRIVATE
        Source/Main.cpp
        Source/MainComponent.h
        Source/MainComponent.cpp
)

target_link_libraries(JuceSliderDemo
    PRIVATE
        juce::juce_gui_extra
        juce::juce_audio_utils
        juce::juce_gui_basics
        juce::juce_graphics
        juce::juce_core
)
```

---

3. Code : `MainComponent.h`

```
#pragma once

#include <JuceHeader.h>

class MainComponent : public juce::Component
{
public:
    MainComponent();
    ~MainComponent() override = default;

    void paint(juce::Graphics& g) override;
    void resized() override;

private:
    juce::Slider slider;
    juce::Label valueLabel;

    void updateLabel();
```

```
    JUCE_DECLARE_NON_COPYABLE_WITH_LEAK_DETECTOR(MainComponent)
};
```

---

4. Code : `MainComponent.cpp`

```
#include "MainComponent.h"

MainComponent::MainComponent()
{
    slider.setRange(0.0, 100.0, 1.0);
    slider.setValue(50.0);
    slider.setSliderStyle(juce::Slider::LinearHorizontal);
    slider.setTextBoxStyle(juce::Slider::TextBoxBelow, false, 80, 20);

    slider.onValueChange = [this]() { updateLabel(); };

    addAndMakeVisible(slider);

    valueLabel.setText("Valeur : 50", juce::dontSendNotification);
    valueLabel.setJustificationType(juce::Justification::centred);
    valueLabel.setFont(juce::Font(18.0f));

    addAndMakeVisible(valueLabel);

    setSize(400, 200);

    updateLabel();
}

void MainComponent::paint(juce::Graphics& g)
{
    g.fillAll(juce::Colours::darkgrey);

    g.setColour(juce::Colours::white);
    g.setFont(juce::Font(20.0f, juce::Font::bold));
    g.drawFittedText("Demo JUCE : Slider + Label",
                     getLocalBounds().removeFromTop(40),
                     juce::Justification::centred,
                     1);
}

void MainComponent::resized()
{
```

```
    auto area = getLocalBounds().reduced(20);
    area.removeFromTop(40);

    auto labelArea = area.removeFromTop(40);
    valueLabel.setBounds(labelArea);

    auto sliderArea = area.removeFromTop(60);
    slider.setBounds(sliderArea);
}

void MainComponent::updateLabel()
{
    auto value = slider.getValue();
    valueLabel.setText("Valeur : " + juce::String(value),
juce::dontSendNotification);
}
```

---

5. Code : `Main.cpp`

```
#include <JuceHeader.h>
#include "MainComponent.h"

class JuceSliderDemoApplication : public juce::JUCEApplication
{
public:
    const juce::String getApplicationName() override       { return "Juce
Slider Demo"; }
    const juce::String getApplicationVersion() override    { return "0.0.1";
}
    bool moreThanOneInstanceAllowed() override             { return true; }

    void initialise(const juce::String&) override
    {
        mainWindow = std::make_unique<MainWindow>(getApplicationName());
    }

    void shutdown() override
    {
        mainWindow = nullptr;
    }

    void systemRequestedQuit() override
    {
```

```
        quit();
    }

    void anotherInstanceStarted(const juce::String&) override {}

    class MainWindow : public juce::DocumentWindow
    {
    public:
        MainWindow(juce::String name)
            : juce::DocumentWindow(name,
                                   juce::Colours::black,
                                   DocumentWindow::allButtons)
        {
            setUsingNativeTitleBar(true);
            setResizable(true, true);

            setContentOwned(new MainComponent(), true);

            centreWithSize(getWidth(), getHeight());
            setVisible(true);
        }

        void closeButtonPressed() override
        {
            juce::JUCEApplication::getInstance()->systemRequestedQuit();
        }
    };

private:
    std::unique_ptr<MainWindow> mainWindow;
};

START_JUCE_APPLICATION(JuceSliderDemoApplication)
```

---

6. Compilation

Depuis le dossier du projet :

```
cmake -B build
cmake --build build
```

L'exécutable sera généré dans :

```
build/
```
