Association d'un objet C++ au document principal QML

```
#include <QGuiApplication>
#include <QQmlApplicationEngine>
#include <QQmlContext>

#include "maclasse.h"

int main(int argc, char *argv[])
{
    QGuiApplication app(argc, argv);
    QQmlApplicationEngine engine;

    engine.rootContext()->setContextProperty("monObjet", new MaClasse());

    engine.load(QUrl(QStringLiteral("qrc:/main.qml")));

    return app.exec();
}
```

Accède à la déclaration de la **classe C++**

Ajoute un **objet C++** au contexte **QML**

Instancie un **objet C++**

Charge le **document principal QML** dans l'application

Appel d'une méthode d'un objet C++ à partir de QML

maclasse.h

```
#ifndef MACLASSE_H
#define MACLASSE_H

#include <QObject>
#include <QString>

class MaClasse : public QObject
{
  Q_OBJECT

public:
  explicit MaClasse(QObject *parent=nullptr);
  virtual ~MaClasse();

  Q_INVOKABLE bool lire(QString type, int nb=0);
};

#endif // MACLASSE_H
```

La **méthode** doit être déclarée avec `Q_INVOKABLE` dans une classe qui hérite de `QObject` pour pouvoir l'appeler à partir de QML

main.qml

```
Button {
   id: btLire
   text: "Lire"
   onClicked: {
      if(monObjet.lire("a"))
      {
      }
   }
}
```

On peut appeler la **méthode** de l'objet **monObjet** de type **MaClasse** à partir de **QML/JS.**

Il est aussi possible d'appeler un ***slot***.

Déclarer une propriété C++ pour l'utiliser en QML

maclasse.h

```
#ifndef MACLASSE_H
#define MACLASSE_H

#include <QObject>
#include <QString>

class MaClasse : public QObject
{
  Q_OBJECT
  Q_PROPERTY(bool erreurConnexion MEMBER erreurConnexion NOTIFY erreurChanged)
  Q_PROPERTY(QString moyenne READ getMoyenne NOTIFY moyenneUpdated)

public:
  QString getMoyenne(); // accesseur

private:
    bool erreurConnexion;
    QString moyenne;

signals:
    void erreurChanged();
    void moyenneUpdated();
};

#endif // MACLASSE_H
```

Une **propriété** est déclarée **Q_PROPERTY()** dans une **classe** qui hérite de **QObject**.

On peut exporter un **attribut** sous forme de **propriété** Qt avec **MEMBER**. L'attribut sera accessible en **lecture/écriture** sans avoir besoin d'accesseurs **READ** et **WRITE**.

Il faut toujours spécifier un ***signal*** avec **NOTIFY** pour autoriser la liaison de la **propriété** avec **QML**.

On peut exporter un **attribut** sous forme de **propriété** Qt accessible seulement en lecture avec l'accesseur **READ** sans utiliser **MEMBER**.

Utiliser les signaux C++ et les propriétés dans QML

maclasse.cpp

```
void MaClasse::connecter()
{
    erreurConnexion = ...
    emit erreurChanged();
}

void MaClasse::calculerMoyenne()
{
    moyenne = ...;
    emit moyenneUpdated();
}

QString MaClasse::getMoyenne()
{
    return moyenne;
}
```

Émet un ***signal*** Qt

L'élément **Connections** crée une connexion aux ***signaux*** de Qt.

main.qml

```
Connections {
  target: monObjet
  onMoyenneUpdated: {
   console.log("Moyenne : " + monObjet.moyenne);
  }
  onErreurChanged: {
   console.log("Erreur : " + monObjet.erreurConnexion);
  }
}
```

L'**objet** Qt qui émet le ***signal***

Il est aussi possible ce créer un signal dans QML et de le connecter à un slot C++ (voir Cours QML).

Les ***signaux*** Qt sont préfixés par **on** : **onSignal**

Accès à la **propriété** C++ d'un **objet**.