Trong video này chúng ta sẽ tìm hiểu cách để tạo ra một ListModel và QStringListModel đơn giản và truy xuất dữ liệu trong model này đưa lên view.

Như đã đề cập trước đây về mô hình Model – View, chúng ta sẽ phân tách phần model và phần view riêng biệt.

Chúng ta sẽ thử nghiệm mô hình này với trường hợp Table và QSqlTableModel và bạn sẽ thấy rằng chúng thật dễ dàng để thực hiện. Nếu bạn đã sẵn sang chúng ta hãy bắt đầu.

Mở chương trình creator và mở project “tableModelView”

Kéo thành phần List View và thả nó vào vùng ở giữa, bên cạnh đó chúng ta cũng kéo phần tử Table View vào vùng đó.

Tiếp theo chúng ta mở file mainwindow.h và khai báo các hàm ở chế độ private để khởi tạo QStringModel.

Ở đây thay vì trả về chính xác con trỏ đối tượng QStringModel chúng ta trả về kiểu dữ liệu của lớp cha của nó

Đây là cách để thực hiện tính đa hình.

Chúng ta hãy xem chi tiết hơn bên trong hàm này tại file cpp.

Đầu tiên chúng ta khởi tạo đối tượng QStringList là một danh sách chuỗi cơ bản sử dụng hàm khởi tạo của cú pháp C++ 11

Sau đó chúng ta khởi tạo QStringListModel với danh sách vừa tạo ở trên

Để giới thiệu với các bạn cách truy vấn dữ liệu

của Model này

Tôi gọi hàm setData để cập nhật dữ liệu cho model này với vị trí model index.

Chúng ta có thể lấy được vị trí index của model bằng cách gọi hàm model->index và truyền số thự tự hàng và cột vào.

Bởi vì đây là một ListModel.

Nên tham số thứ 2 có thể bỏ qua

Vì vậy nên câu lệnh này thay đổi giá trị dữ liệu của dòng thứ 3 trong model, nghĩa là thay đổi “Thyme” thành “Cinnamon”

Hãy quay trở lên hàm khởi tạo

Và gán model của View là StringModel mà bạn vừa khởi tạo

Và sau đó chạy chương trình

Bạn sẽ thấy dữ liệu trong ListModel được hiển thị ở thành phần ListView của giao diện và phần tử thứ 3

của nó đã được thay đổi thành “Cinnamon”

Hãy nhớ chúng ta đã nói rằng chúng ta sẽ dùng QSqlTableModel như một Model Table của thành phần giao diện ListView

Chỉ cần thay đổi một dòng, gọi hàm setModel của listView và thay đổi tham số truyền vào

là con trỏ QSqlTableModel thay cho hàm initStringModel()

Bây giờ chúng ta chạy lại chương trình

Hãy nhìn giao diện ListView hiển thị cột thứ nhất của Table.

Lý do của điều này là do nó là một StringModel.

Đây là một trường hợp đặc biệt của Table mà chỉ có một cột duy nhất và bởi vì giao diện sử dụng chỉ số thứ tự (index) của model

Để truy xuất dữ liệu

Chúng ta không cần phải lo lắng về vấn đề cấu trúc của dữ liệu hay kiểu dữ liệu quá dài bởi vì nó có thể

Lấy được dữ liệu thông qua giao thức chuẩn tách biệt với thành phần giao diện ở trong view.

Và làm thế nào để thay đổi phần tử trong danh sách này

Và bạn thấy rằng sự thay đổi được cập nhật trực tiếp qua giao diện

Và nếu bạn double click vào đối tượng đại diện trên giao diện, thành phần mà đại diện cho một item của dữ liệu thì

Dữ liệu bạn nhập vào sẽ được cập nhật vào data model.

Bởi vì cả hai view này đều truy xuất đến cùng một data model nên sẽ nhận cùng các tín hiệu liên quan được phát ra và lệnh gọi của vị trí.

Cái mà giúp bật chế độ lắng nghe thay đổi của các sự kiện để thay đổi giá trị trên giao diện.

Trong phần này chúng ta đã học kiến trúc Model-View và làm thế nào để sử dụng TableModel

và View, cũng như cách sử dụng ListModel với View, trong phần tiếp theo

bạn sẽ bắt đầu quá trình tìm hiểu về Qt Quick và QML.