HEALTHGOAL

User Interface Documentation

NOVEMBER 14, 2017

HEALTHGOAL V 1.0

Contents

Introduction

The project is a purely Qt Quick application and no C++ backend is used as per requirement.

Qt Versions

- Qt 5.9.2
- Qt Creator: 4.4.1
- Compiler MingW 5.3.2

Supported Platforms

- Windows
- Linux
- Android
- Others: So far, there is no limitations on other Qt supported platforms.

GUI organization

The Window object creates a new top-level window for a Qt Quick scene. contentItem is the invisible root item of the scene. Overview page/item is created and loaded into the Window in main.qml.

Overview Page contains Scrollview as container. On top of the view ProfileArea is present. Below ProfileArea, item “p” is present which holds the tiles. Below “p”, fadedHexagon is anchored.

Tile class is inherited from QML Image. It has MouseArea to handle mouse and touch interactions.The class has signals like newHexagonAdded() and hexagonRemoved().

The Tiles have certain extent of overlapping to fit the edges. The Tile width is calculated based on Window width.

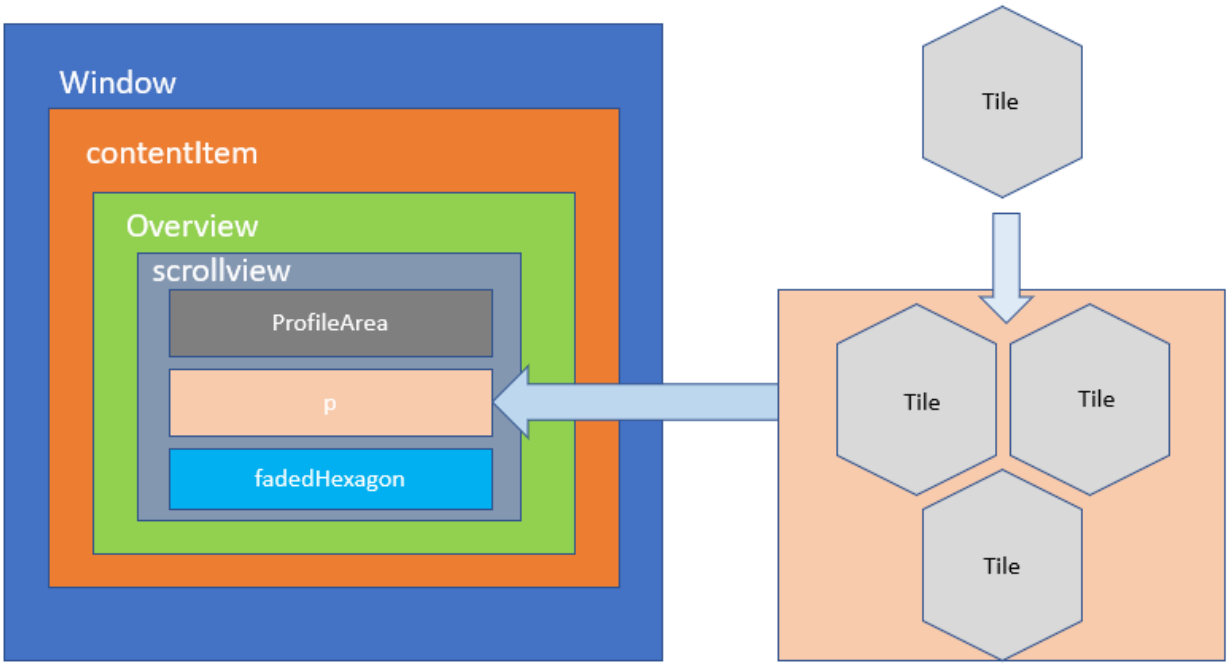

Figure 1 : UI organization

Model-View Implementation

The Tiles are created and updated as per Qt's Model-View implementation. The model communicates with a source of data, providing an interface for the other components in the architecture. The nature of the communication depends on the type of data source, and the way the model is implemented.

The view obtains model indexes from the model; these are references to items of data. By supplying model indexes to the model, the view can retrieve items of data from the data source.

In standard views, a delegate renders the items of data. When an item is edited, the delegate communicates with the model directly using model indexes.

In current implementation, ListModel is used for our model and Repeater is used as View. We have used customDelegate as delegate.

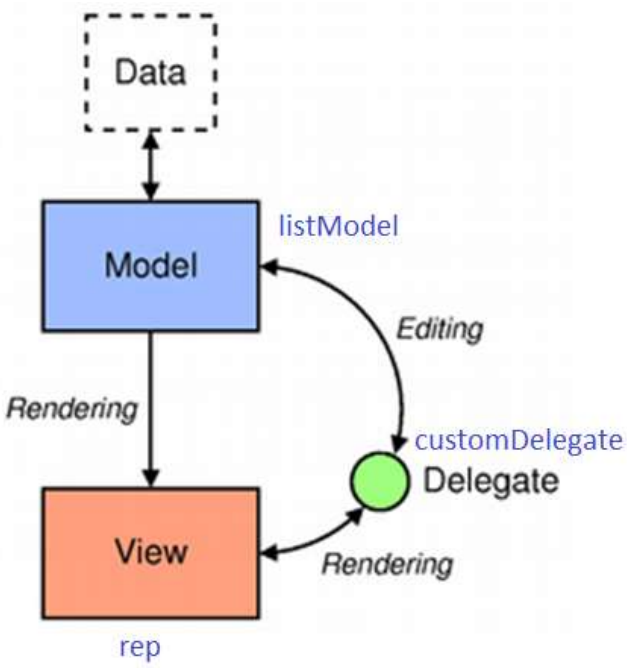

Figure 2 : M-V implementation

User Actions

When fadedHexagon is tapped single or two hexagons are added based on earlier row. When user does a long press, or press and hold action, then close icon is shown. On tapping close icon, the tile is removed. This is achieved by removing the element from model. Similarly, an element is added to model when add icon is tapped on fadedHexagon. The Tiles auto arrange themselves. The user can drag an item and dop it on another tile to swap the tiles.

Drag & Drop

For intuitive swap between tiles, Drag and Drop infrastructure is used. dragItemComponent is a drag Item which creates the drag illusion instead of real drag of original item. Drag and drop provides a simple visual mechanism which users can use to transfer information between and within applications. Drag and drop is similar in function to the clipboard's cut and paste mechanism. DropArea is used to catch the dragItem and perform the swap action.

Animations

NumberAnimation is used in adding and removal of tiles. NumberAnimation is a specialized PropertyAnimation that defines an animation to be applied when a numerical value changes. In this project, we are manipulating opacity to show the animation.

- *removeAnimation* is used when a tile is removed.
- *addAnimation* is used when a tile is added.
- *swapAnimation* is used while performing a swap.