Guiapp MVC implementation

Overview

This article introduces a Qt MVC implementation. The Qt application is based on SL2.0 guiapp. This MVC framework is derived from Flux project (https://facebookarchive.github.io/flux/). This implementation is also inspired by the project QuickFlux (https://github.com/benlau/quickflux) which is proven to work for Qt/QML applications.

Concepts

The most important concept is that the Flux architecture utilizes a unidirectional data flow. Flux has four major parts: the dispatcher, the store, the action, and the view.

Dispatcher

The dispatcher receives each action and then dispatches the action to stores that have registered with the dispatcher. The dispatcher is singleton in guiapp.

Store

The store in guiapp holds the data or model. The store is an abstract concept. It can be a whole application such as CallUI or Settings. It can also be a smaller entity such as an instance of SLPage.
A store registers a group of actions it wants to handle. The store notifies data changes to views.

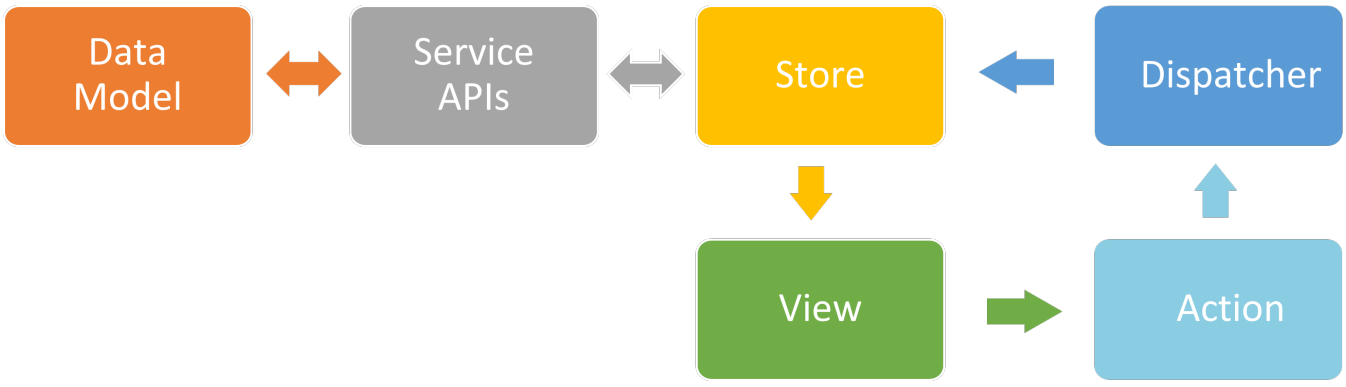
Action

An action defines what should be done when the user performs certain operation. The internal identifier of the action should be unique in the system. One action could be listened by multiple stores, because the user operation may result in multiple data changes. Actions are typically dispatched from the view, but they can also be dispatched from other modules such as a store.

View

The view displays the data with QML items. This is pretty much similar to current SL2.0 view concept. The view is skinny since it does not handle business logic and action dispatching details. The first principle of this MVC architecture is that the view does not update the store directly. Instead, the view sends an action with changed data to the dispatcher.

Data flow



Code

https://sqbu-github.cisco.com/zhijin/sl-main/compare/488e5283..5147dbe0

Examples

TBD

FAQ

TBD