B) Tutorial Overview

Created by Jacques Marais, last modified on Oct 09, 2018

How to Use This Tutorial

This tutorial was developed as a step by step guide on how to use Helium and the Helium DSL to develop a sample application. The intended use is to work through the lessons in order and to use the Supplementary Documentation section as needed. The supplementary documentation also includes a list of best practices, and it's important to be aware of these before moving on from the tutorial.

All the Helium DSL features covered in the tutorial lessons are also mentioned in the DSL Reference, View Reference and Quick Reference. If more detail is needed on any topic, these documents can be consulted.

Each tutorial lesson consists of the following:

- Downloadable source code for all lessons up to that point included at the end of each lesson page as a zip file attachment. Note that although the tutorial wiki pages themselves contains some code snippets, this is mainly for demonstration/reference purposes and does not represent all the source code for the lesson. For all the source code related to a specific lesson, please refer to the downloadable zip attachment linked at the end of each lesson.
- A written lesson that highlights learning outcomes and the steps required to reach them.

Introduction to the Helium DSL

The Helium DSL is a domain specific language with Java-like syntax that runs on top of the Helium platform. The basic constructs of the language are listed in the Quick Reference.

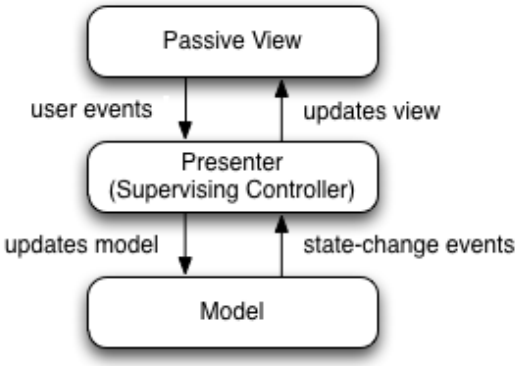
Helium applications subscribe to the model view presenter pattern. Model components are created as Helium script files using the .mez file extension. The model files describe data validators, enum types, custom objects and relationships between objects.

Presenters are also described in .mez script files and contain variables and functions grouped together in a construct called a unit. These variables and functions contain the application logic which can be invoked in various ways.

Firstly, application logic can be invoked by views. View files contain xml and uses the .vxml file extension. Views represent the user interface where user interaction results in the execution of application logic.

In addition to views, Helium also provides special callback functions and scheduled functions which can be used to invoke code on the back-end.

The diagram below shows the model view presenter pattern that Helium apps subscribe to.



Introduction to the Tutorial App

The application that will be developed as the base of this tutorial is an app that is centered around a seed and fertiliser store system. This system includes user management, capturing an reporting on different types of data, management of purchases, invoicing and various other notification and monitoring features.

This topic was chosen as a simplified real world example that demonstrates most, if not all, of the DSL and Helium features available to developers. The individual requirements for the application can be found below.

Lesson Plan

Roles	Requirement	Covered in lesson
System Admin	As a System Admin I want to see a welcome page with my details when I log into the app.	Lesson 1
System Admin	As a System Administrator I want to view a list of System Administrators.	Lesson 2
System Admin	As a System Administrator I want to invite and edit other System Administrators.	Lesson 3
System Admin	As a System Administrator I want to delete other System Administrators.	Lesson 4
System Admin	As a System Admin I want to be able to view, create, edit and delete shops.	Lesson 5
System Admin	As a System Admin I want to invite, view and edit Shop Owners and link them to a shop.	Lesson 6
System Admin	As a System Admin I want to invite Farmers.	Lesson 7
Farmer	As a farmer, I want to specify for my profile the crops I cultivate.	Lesson 8
Shop Owner	As a Shop Owner I want to do bulk updates to my stock levels by uploading CSV files.	Lesson 9
Farmer	As a Farmer I want to set user preferences and receive stock level notification messages accordingly.	Lesson 10

Roles	Requirement	Covered in lesson
Farmer	As a Farmer I want to upload my Government Assistance Recipient Certificate.	Lesson 11
Farmer	As a Farmer I want to purchase items from a shop and make payments using the Helium app.	Lesson 12
Farmer	As a Farmer I want to receive an invoice for each purchase. This is to be sent as an E-mail attachment. In addition the E-mail should make use of a custom template.	Lesson 13
System Admin, Shop Owner, Farmer	As any user, I want to log a support ticket by sending an SMS message.	Lesson 14
System Admin, Shop Owner	As a System Admin or Shop Owner, I want to see farmer locations on a map.	Lesson 15
Farmer	As a Farmer, I want to view a customizable table comparing stock prices at different shops.	Lesson 16
System Admin	As a System Admin I want to view a report of purchases made.	Lesson 17
System Admin	As a System Admin I want to be notified via email about consecutive failures of SMS or e-mail messages.	Lesson 18
System Admin	As a System Admin I want to view details of failed messages.	Lesson 19
System Admin	As a System Admin I want to see my role name differently depending on whether I'm a developer or a client.	Lesson 20
System Admin	As a System Admin I want to view the names and surnames of System Admins as they have entered it on their user profiles.	Lesson 21
Shop Owner	As a Shop Owner I would like to see a data table with the number of purchases per week for a specified period and shop.	Lesson 22
System Admin	As a System Admin I would like to resolve all current service tickets or mark all current service tickets as spam.	Lesson 23
System Admin, Shop Owner, Farmer	As any app user I want to have the app initialised upon deployment to include the default stock items As a System Admin I want to have access to a data table that summarises all purchases in the system	Lesson 24

Conventions in This Tutorial

Where keywords, variable names, etc. from code examples or for code explanations appear in the text, it is formatted in monospace and, to distinguish further, in a particular colour:

token / entity	example
file names and file paths in gray and bold	**SystemAdminHome.mez**
keywords and HeliumDev client commands in black and bold	**string**
non-keyword identifiers, e.g. variables, functions in green	**function** getSystemAdmins() **string** systemAdminName
presenter unit names in green	**unit** SystemAdminHome
user role names in green and quotes	"System Admin"
view names in blue	SystemAdminHome
view XML elements in blue and angle brackets	<info>
other view XML excerpts in blue	label="info.label"
properties file excerpts in orange	menu_item = Manage Users

No labels