Ikasan Studio

Contents

[Introduction 1](#_Toc85034066)

[Installing the plugin 1](#_Toc85034067)

[Basic onscreen regions 1](#_Toc85034068)

[Code Editor Panel 1](#_Toc85034069)

[Component Configuration Panel 2](#_Toc85034070)

[Module Canvas 2](#_Toc85034071)

[Component Palette 2](#_Toc85034072)

[Creating a simple project 2](#_Toc85034073)

# Introduction

This document has been produced to assist new developers in learning how to use the IDE, and for new IDE developers to get an overview of the major components.

This document intends to be a hands-on, action driven tutorial, rather than a monologue of features. The guide can be used in conjunction with the videos, instead of the videos or the videos used instead of the user guide. Different people assimilate knowledge in different ways, no one way is any better or worse than the other.

# Installing the plugin

# Basic onscreen regions

Once installed, the plugin should be available from the standard rightnav menu, clicking on the logo will expand the components of the plugin.

Window is split into 4 sections

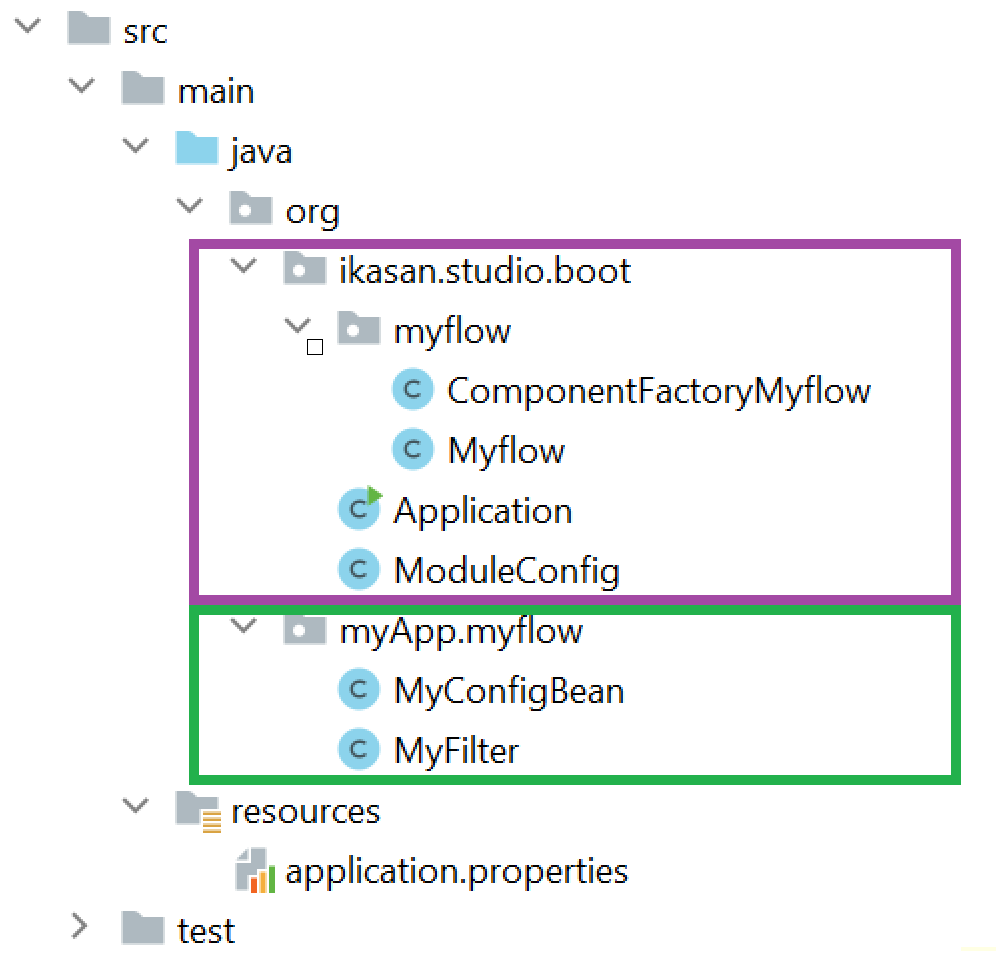
* Code editor panel (a standard component of InteliJ)
* Component configuration panel (displays configuration and support update of configuration for the current component)
* Module Canvas (contains the graphical representation of the module)
* Component palette (contain all the Ikasan components available to the developer)

### Code Editor Panel

The package structure is split into 2 distinct areas

The Ikasan Studio generated package tree, the developer must not edit this section directly, otherwise it is possible that their edits will be overwritten as the source code is refreshed from the ‘internal model’ of the module

The developer maintain package tree, some stubs will be auto-generated into this section when a component is first dragged onto and configured in the module canvas



### Component Configuration Panel

### Module Canvas

### Component Palette

The component palette on the far right of the Studio works space contains icons for flows and components. The developer can select a component and drag it onto the canvas. The canvas will indicate if a component can be dropped at the current location

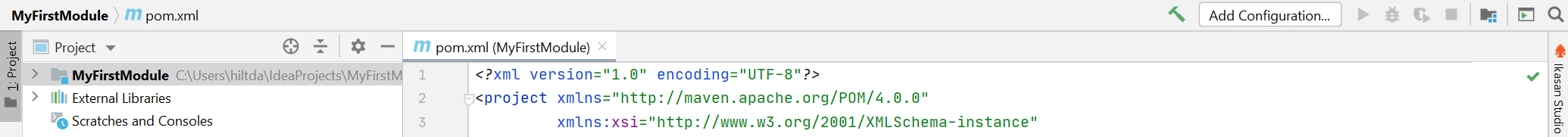
## Creating a simple project

This section can be used in conjunction with the tutorial ‘CreatingASimpleModule’ found in the tutorial section of the distribution. The developer is encourages to watch the video then refer back to this section as a memory aid.

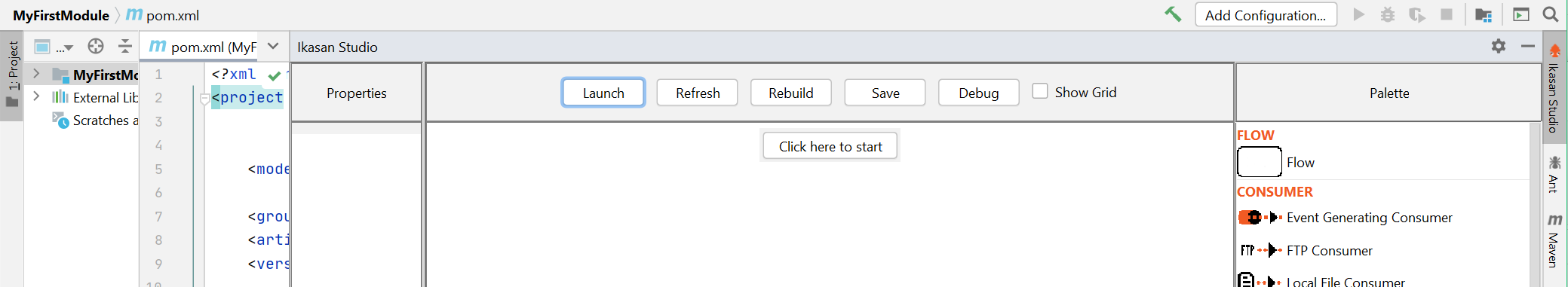
The best was to use Studio is to start a new project (though it is possible to use it with an existing module if the Ikasan standard coding standards have been used).

|  |  |
| --- | --- |
| 1. From the Intellij welcome screen, opt to create a new project |  |
| 1. Currently Maven is the supported build mechanism, other build tools will be added over time. There is no need to select an archetype |  |
| 1. Provide a module name then allow IntelliJ to create the module and complete its standard index operations. |  |

The IDE will typically open the pom.xml and show the main IntelliJ screen



Click on the Ikasan menu and icon on the right navigation board menu to reveal the module start screen:



|  |  |
| --- | --- |
| Once the ‘start button has been clicked, the user may enter  **Project Name** – This will be used to identify the module, it will be used as part of the URL to the module control  **Application Package Name** – This is the top level package structure for your Java code. It will be used by Studio when it generate stubs for you and will be used by you to hold your own code. |  |

**Application Port Number** – This really only affects the local installation, it will be the port that is copied to the local properties file and used to launch Ikasan and launch the console. In your organisation you may have a framework for managing properties in different environments

**Description** – An optional description of the project

Code Generation

Appendix 1 – Intelij Program Structure Interface (PSI)