Introduction to Low Code Mobile Apps Development

22 - 23 Sept 2022

outsystems

REPUBLIC POLYTECHNIC

Administrative Instruction

- If you have not done so, do sign up for OutSystems account at **https://outsystems.com**

- If you like to try this out on your laptop, download and install OutSystems Service Studio at **https://outsystems.com/downloads/**

- Download training material at **https://bit.ly/LoCoMAD-sept2022**

Low Code Mobile Apps Development

This course is focused on the fundamentals of mobile app development in OutSystems 11

At the end of this course
you will have the...

- Fundamental knowledge & skills
- Hands-on experience

...to start building your own mobile apps with OutSystems

Programme Day One

Morning	Afternoon
○ Quick overview of OutSystems Software ○ User Interface Design and Development ■ Use of common UI components ■ Variables ■ Handling User input ■ Basic Logic operation ■ Handling Event ○ Multi screen Application ■ Navigation between screens ■ Passing data between screens	○ Data Driven Application ■ Import Data from Excel ■ Create screens from imported Data ■ Customization of UI Screen ■ CRUD Operations ○ Basic User Management ■ Check Login User ■ Get User Specific Data ○ App Distribution

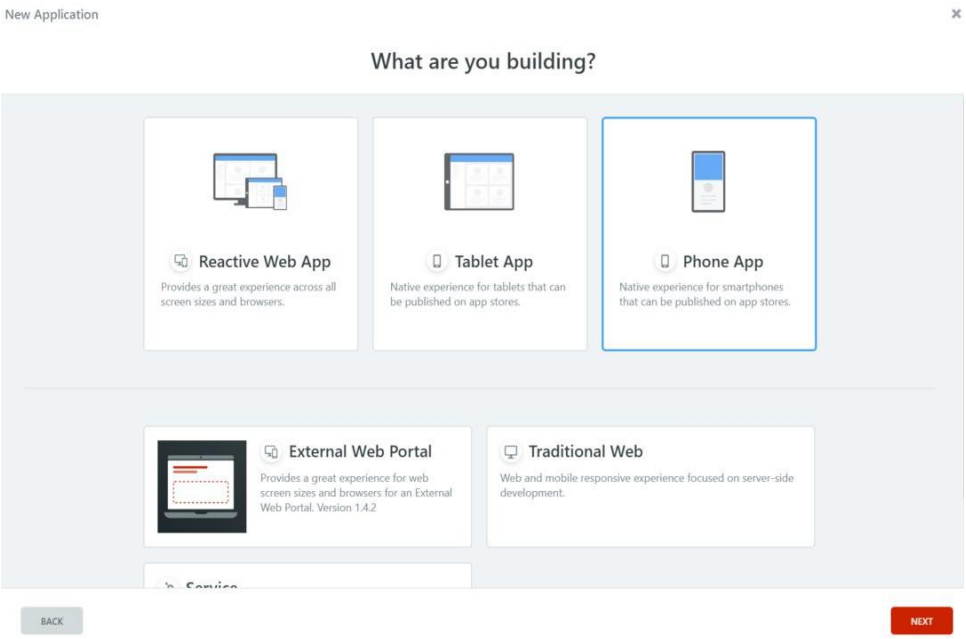
Mobile Applications in OutSystems

Topics

- Mobile Applications in OutSystems
  - Creating a mobile application
  - Application modules

Applications types in OutSystems

- Applications can be
  - Web App
  - Mobile App
  - Service
- Web and Mobile applications follow different programming models
- Applications are the deployment unit in OutSystems platform
  - Can be versioned and tagged for easier management

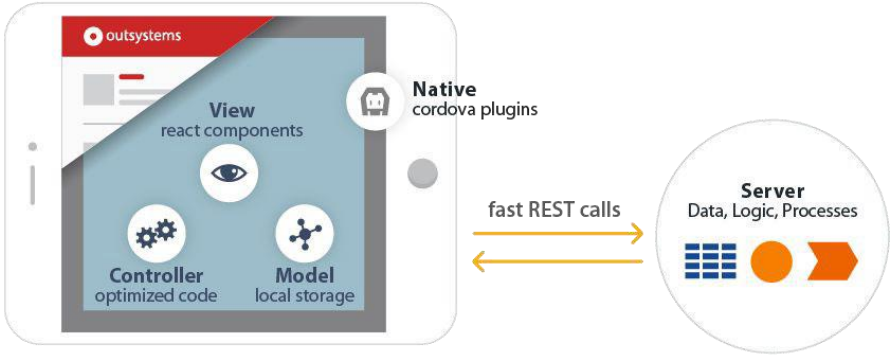


Mobile apps in OutSystems

Run on Android and iOS devices

Single Page Applications (SPAs)

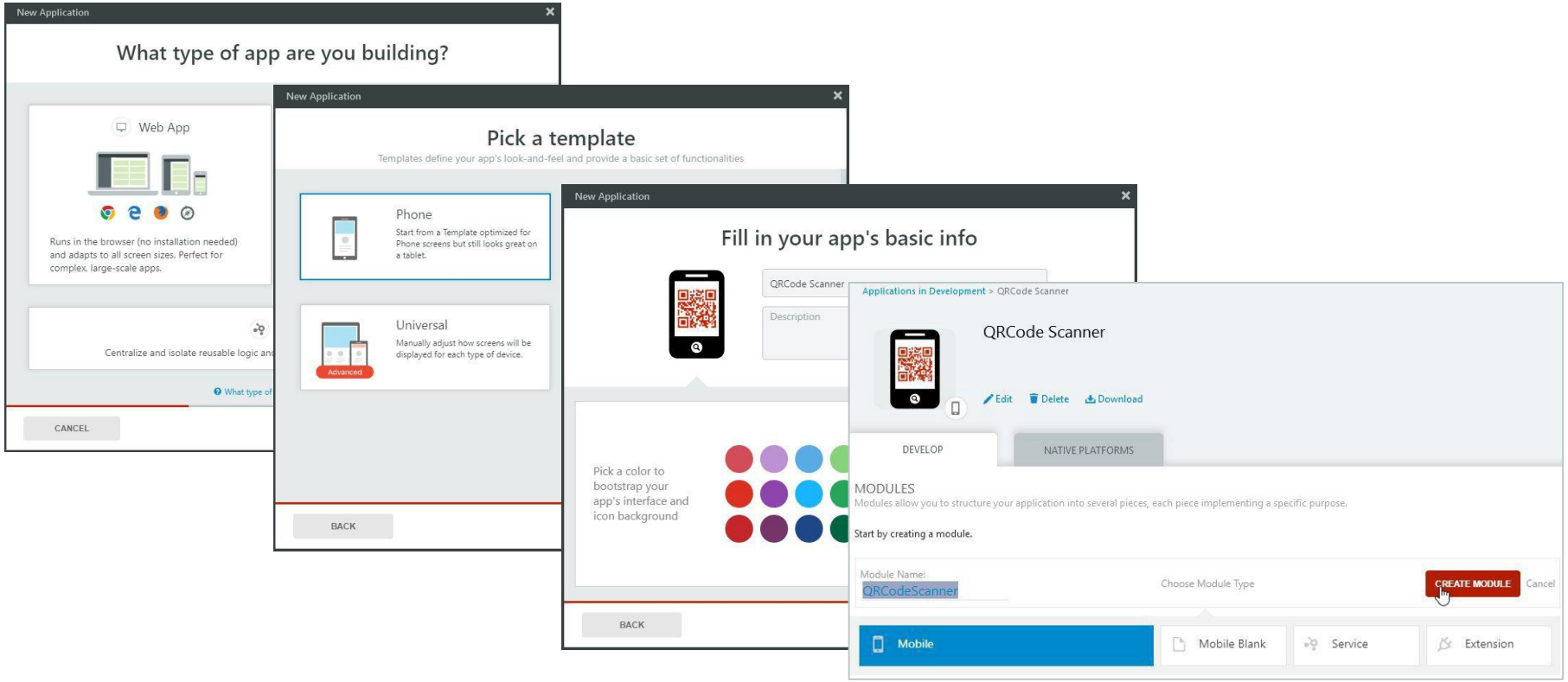
Cross-Platform, Standards-Based



JavaScript, Java and Swift

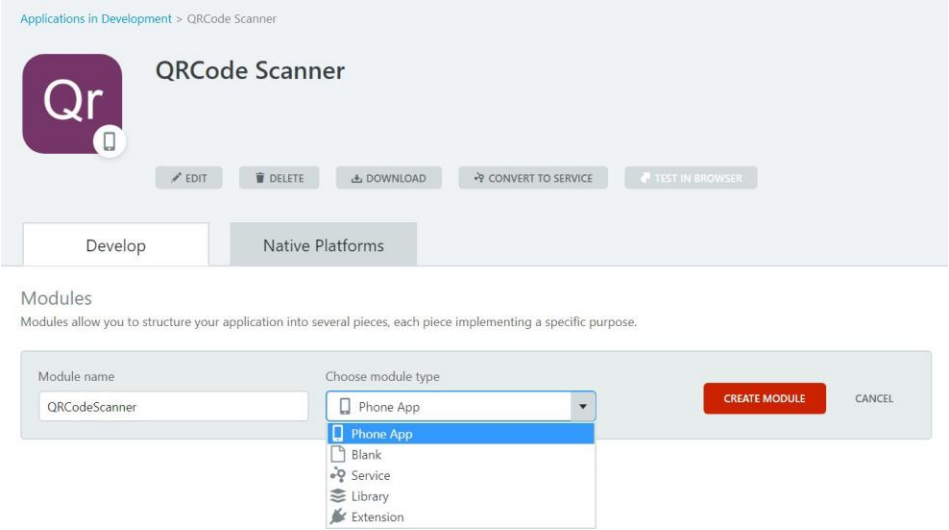
C#

Creating a Mobile Application

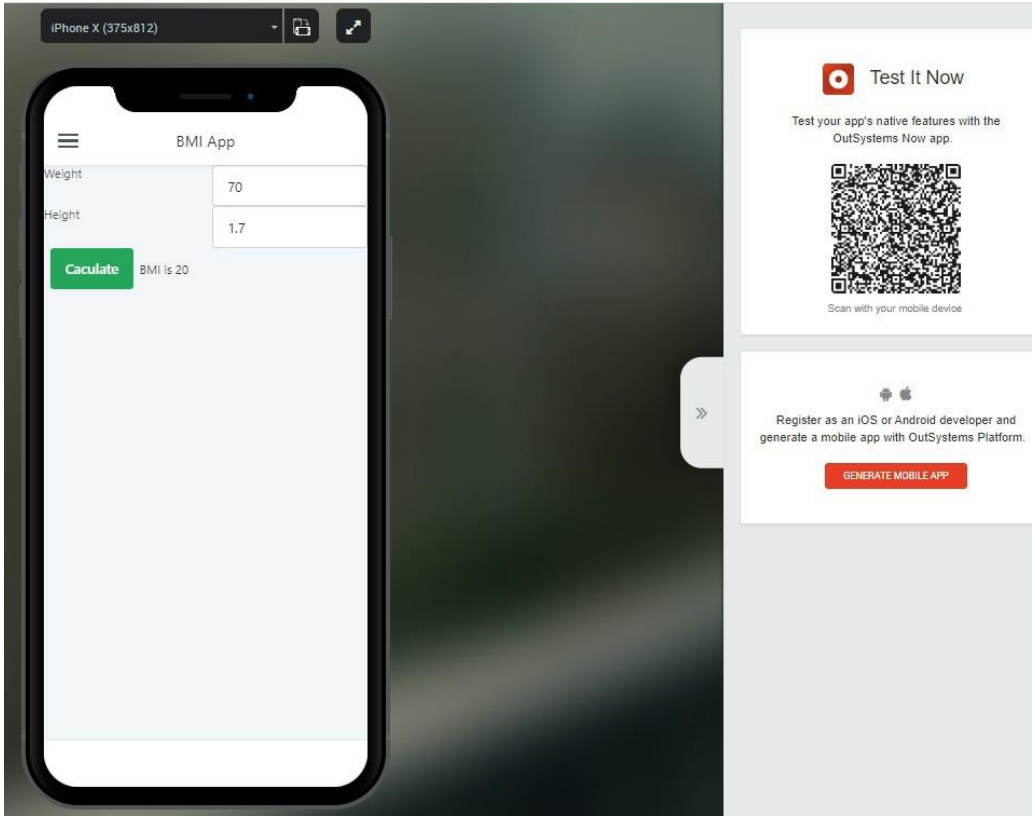


Mobile Applications

- Applications should have at least one modules
- Mobile application modules can be:
  - Phone App
  - Blank
  - Service
  - Library
  - Extension



Exercise Lab 1: Create and Test a Mobile App



Mobile Screens

Topics

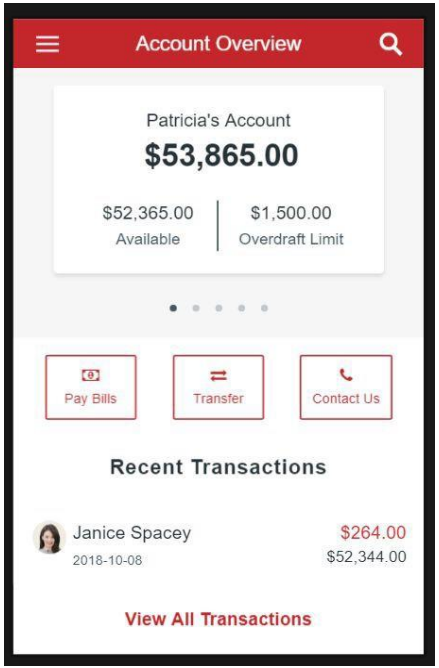
- OutSystems Mobile Apps
  - Single Page Applications
  - Runtime Architecture
- Screens
  - Screen Templates
  - Screen Content
- Screen Variables
- Fetching Data to Display on Screen
- Client-side Logic

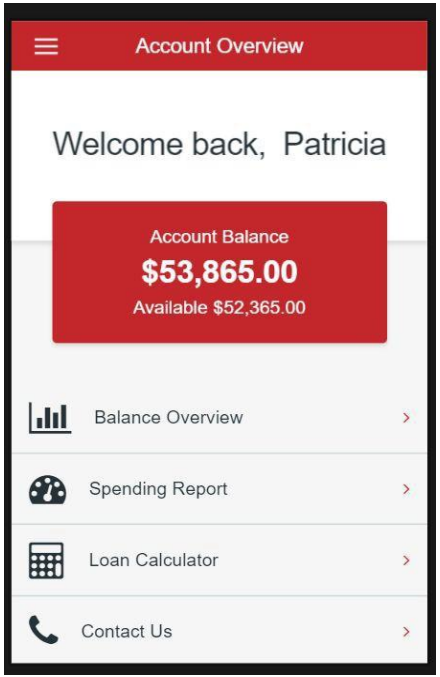
OutSystems Mobile Apps

- OutSystems mobile apps can be installed and run on iOS and Android devices
- Generated with optimized JavaScript application at its core
  - Single Page Applications
  - Screen logic runs on the client side
- Requests to server are only made when necessary
  - Data, Server-side logic (.NET)
  - Automatically created REST Calls

Screens

Screens define the user interface end-users interact with



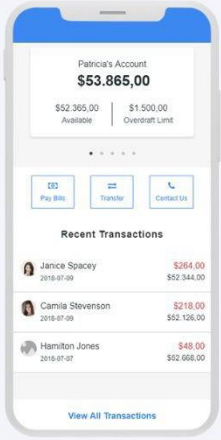


Screen Templates

Screens can be Empty or based on a Template

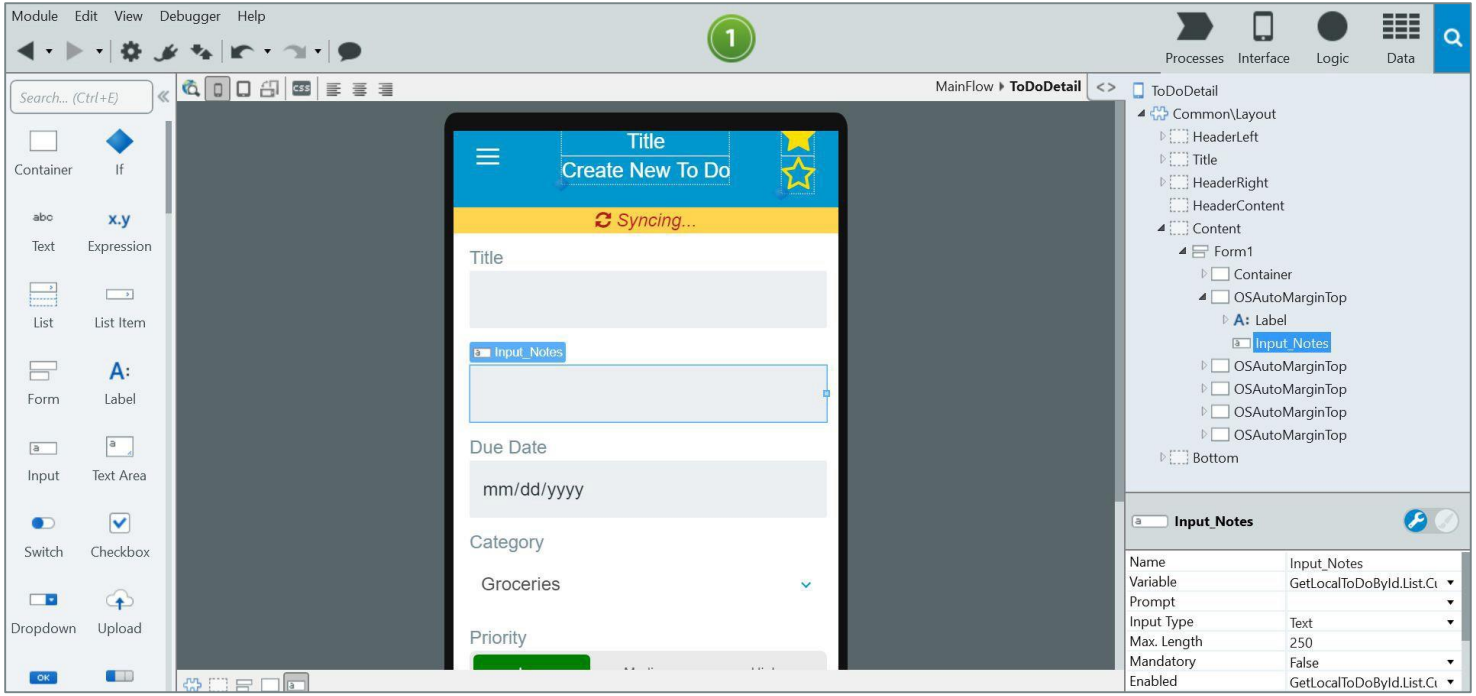
Empty

Dashboard With Carousel



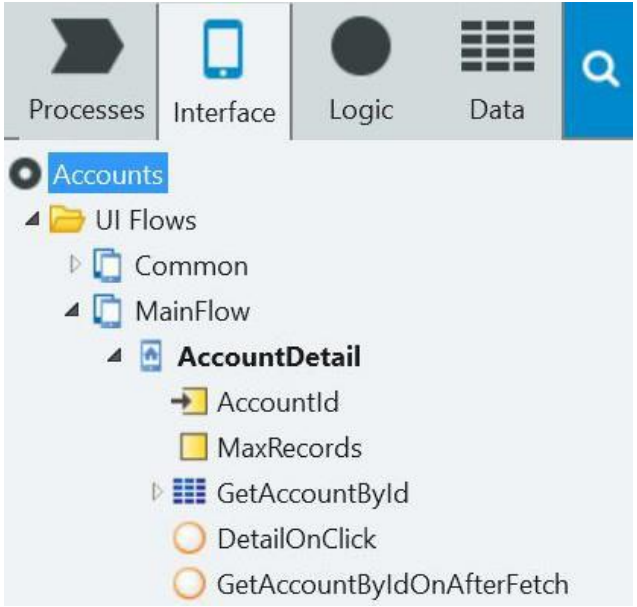
Screen Content

OutSystems Screens are built based on widgets (UI elements)



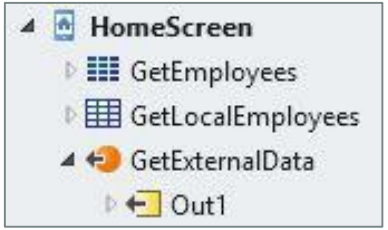
Screen Variables

- What is displayed to the end-user can depend on data
- Some data can come be passed to the Screen
  - Input Parameters
  - When transitioning to a new Screen, a value must be passed to mandatory Inputs
- Screens can also have Local Variables
  - Initialized in the scope of the Screen
- These variables only exist in the scope of the Screen



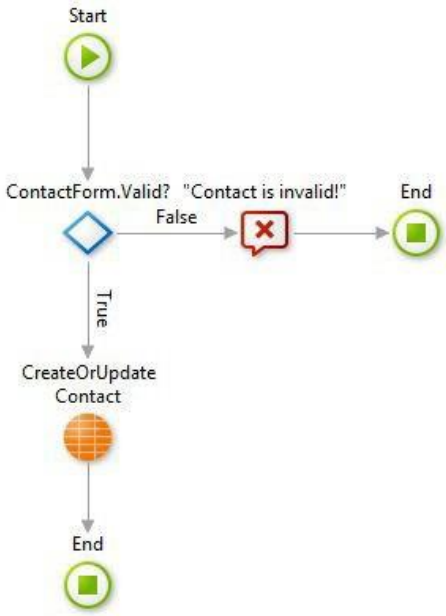
Fetching Data to Display on Screen

- Screens may need to display data available else where
- Aggregates in the Screen
  - Database or Local Storage Entities
- Data Action for advanced cases
  - Server Action
  - Output Data Type by default is Text, but can be changed
- Queries performed asynchronously and in parallel
- Screen Lifecycle Event
  - On After Fetch



Client-side Logic

- Screen Actions run client-side logic in the scope of the Screen
  - Triggered within the Screen
- Client Actions
  - Visually modeled logic and data
  - Easy to call server-side logic
    - Drag & drop
    - REST API generated automatically
- Responsiveness
  - UI elements react to data changes
    - Updates occur immediately
  - UI responds while calling server



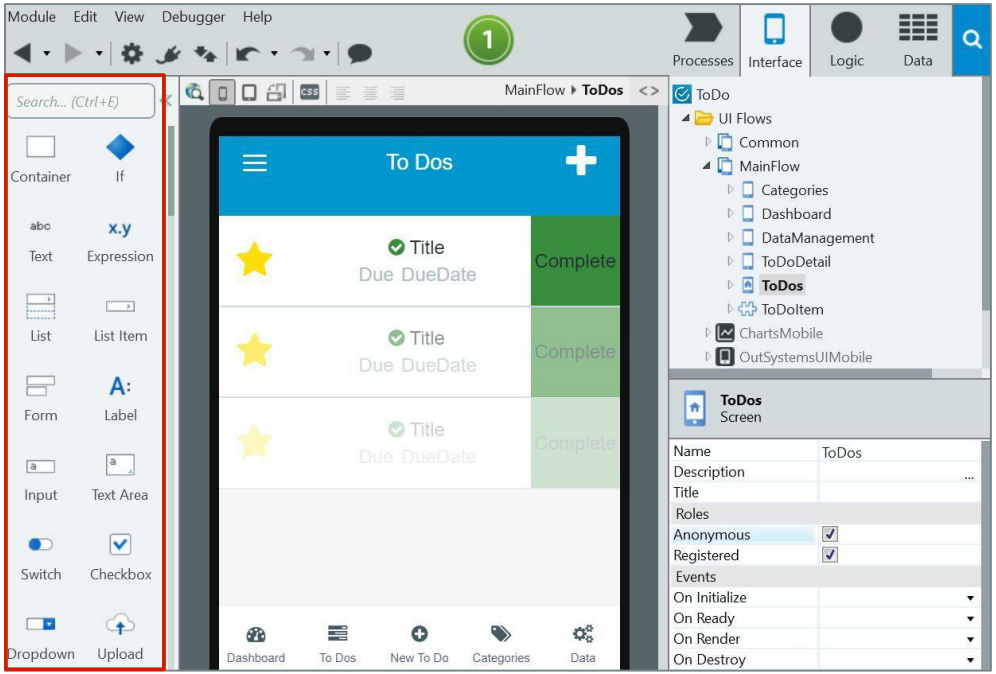
Mobile Widgets

Topics

- Widgets
  - Using Widgets
- Basic Widgets
  - Expression
  - Container
  - List and ListItem
  - Link
- Input Widgets
  - Form
  - Input
  - Dropdown
  - Button

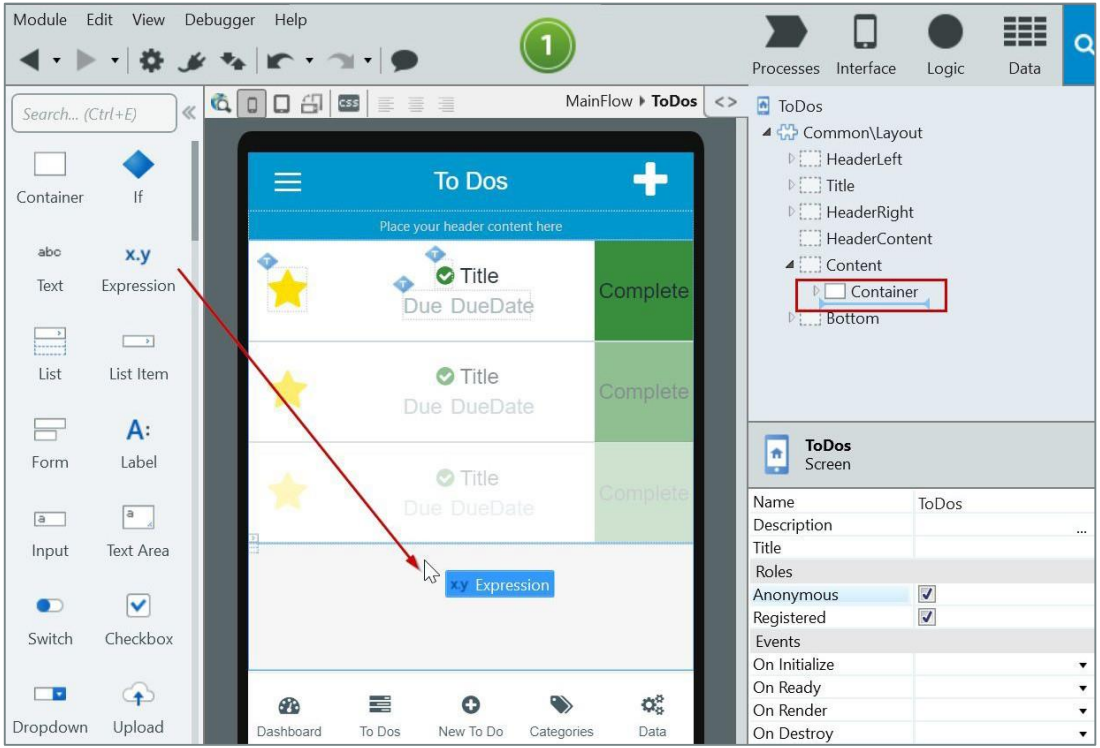
Widgets

- Simple built-in Patterns that represent a Screen component
  - Widgets can be found in the Toolbox
- What can Widgets do?
  - Basic Widgets: Lists, selections, navigation, display content, & popups
  - Input Widgets: Form and inputs
- Why are there Widgets?
  - Common components that are highly reusable



Using Widgets

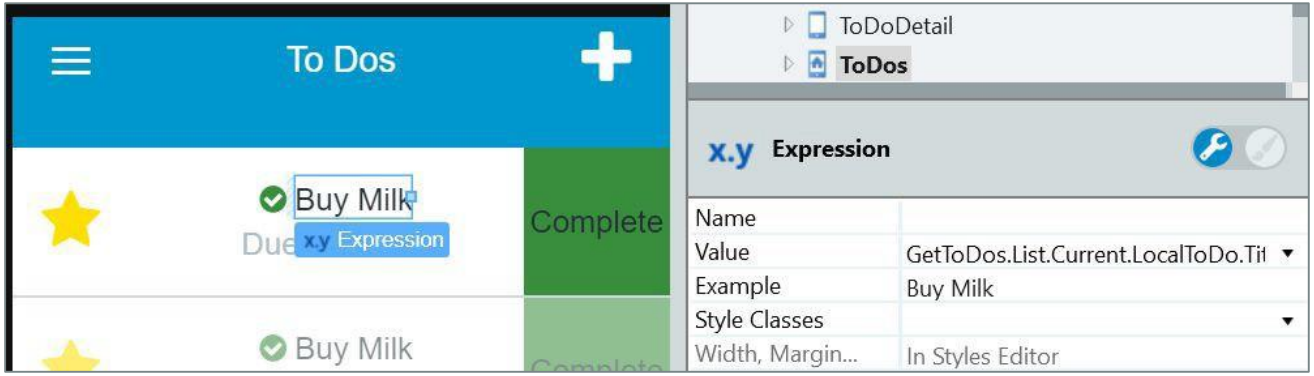
- Drag & Drop Widgets…
  - To the Canvas
    - Opens Widget Tree
    - Help guide placement
  - To Widget Tree
    - For precise placement
- Widget hierarchy
  - Near bottom of canvas
  - Shows related Widgets
  - Select to view properties



Basic Widgets

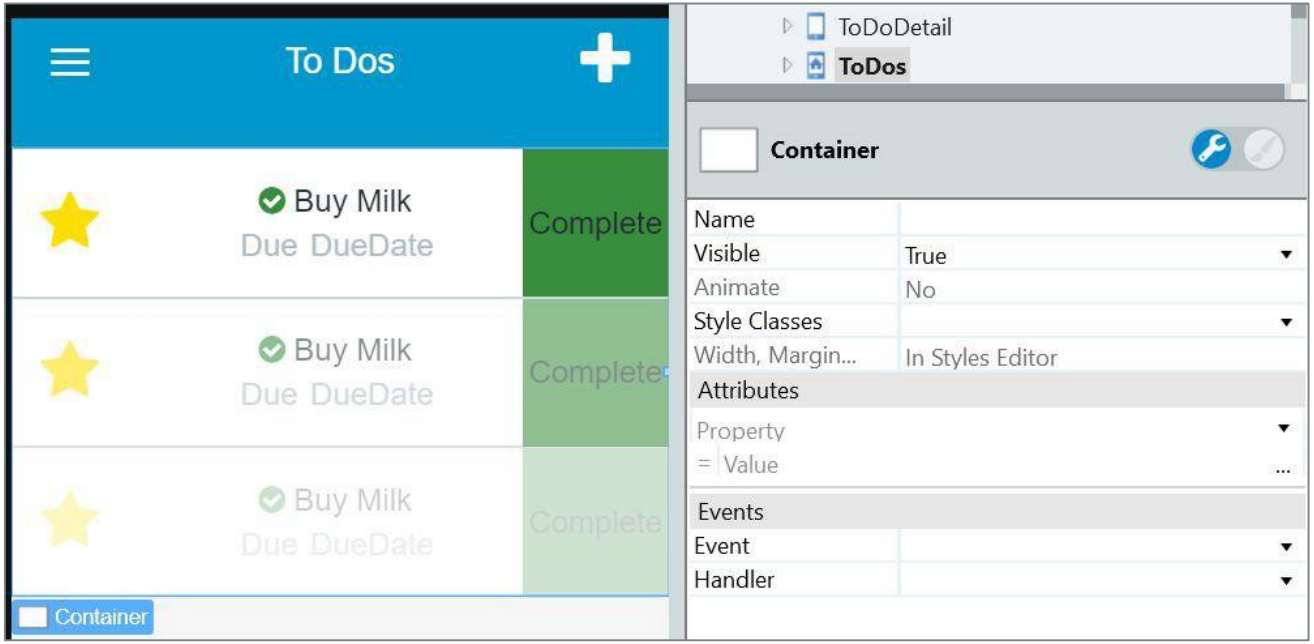
x.y Expression Widget

- Displays the Value of the *Value* property on the Screen
- *Example* property displays an example Expression on preview



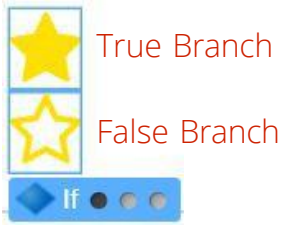
Container Widget

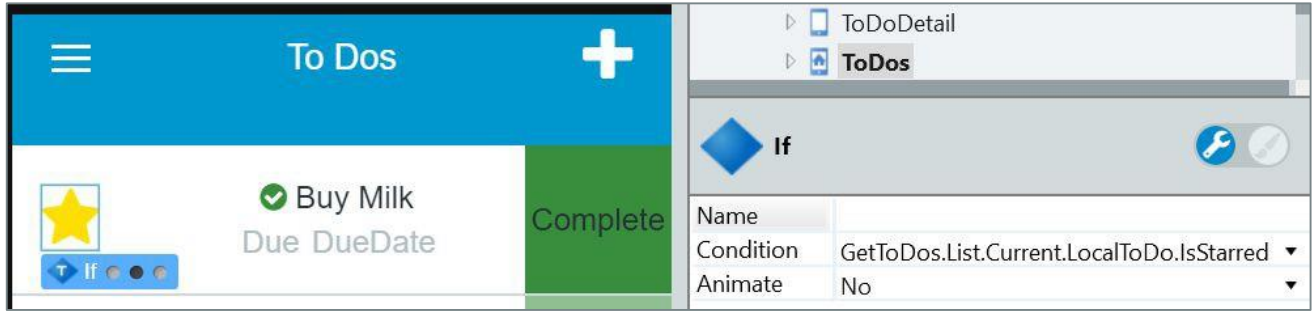
- Groups Widgets together and applies styles to them
- *Visible* property determines if the content is displayed or not



If Widget

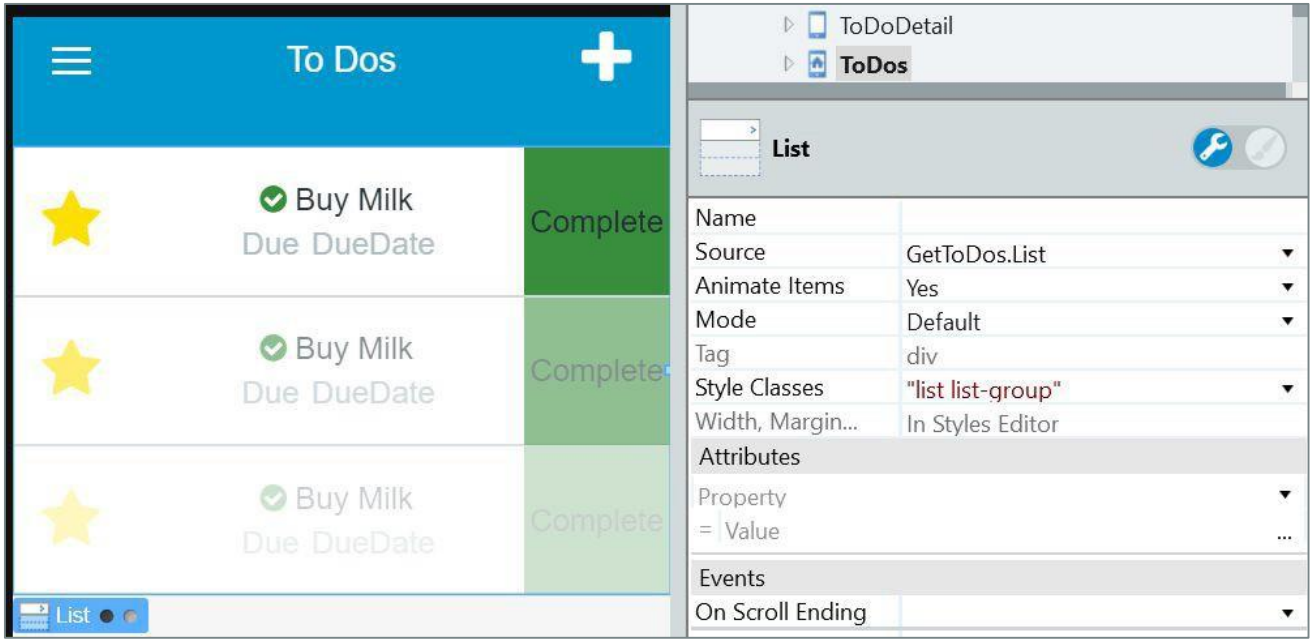
- Displays content based on a Boolean condition
- Two branches with Screen Contents, True or False
  - Boolean condition determines which branch is displayed
  - Both branches (or just one) can be displayed in Service Studio





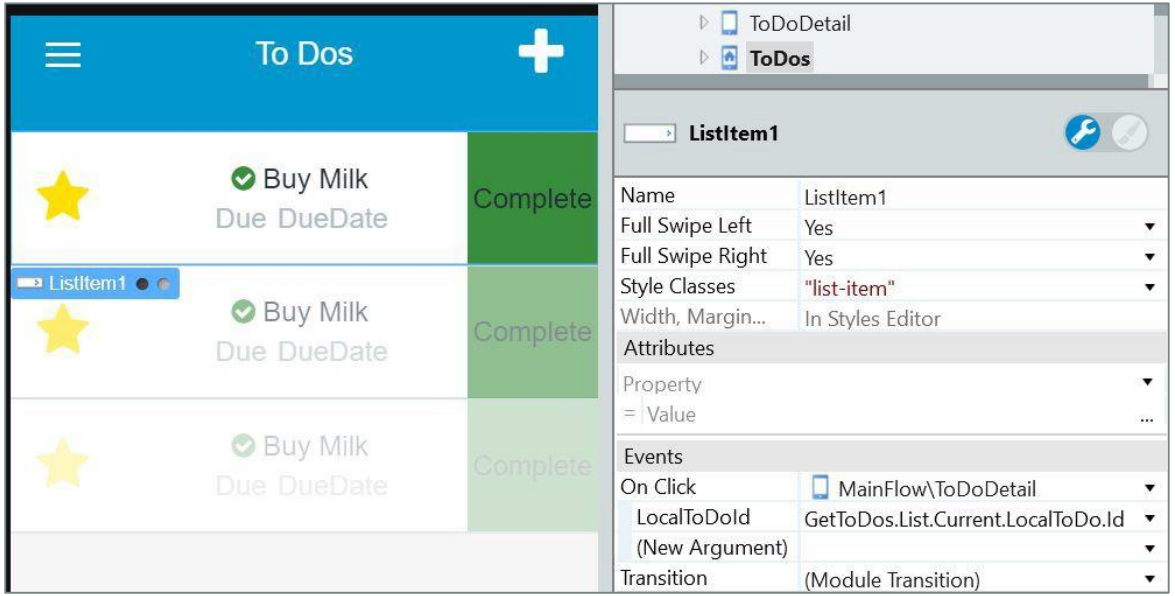
List Widget

- Displays a scrollable List of records
- *Source* property expects a List of records to display (e.g: output of an Aggregate)



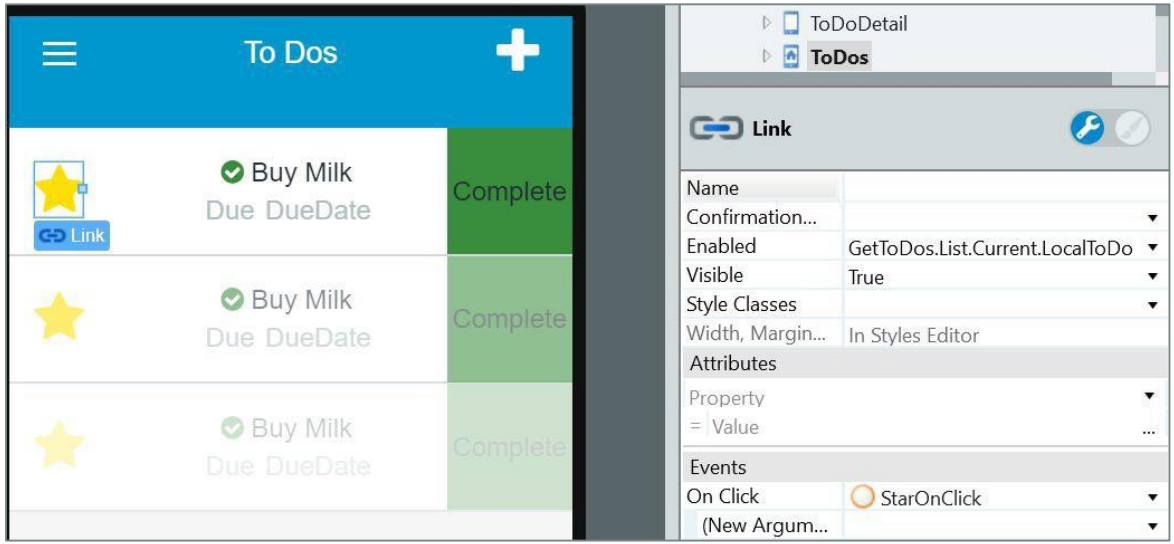
ListItem Widget

- Displays a record (inside or outside a List)
- On Click and Swipe Behavior
  - Full Swipe Left or Right



Link Widget

- Provides a Link to a Destination
  - Screen Action
  - Screen
- Can enclose other widgets (e.g: Container, Expressions, Icon…)



More Basic Widgets

Icon: Displays a scalable vector picture

Image: Displays an image. Source is resource, URL, or binary data

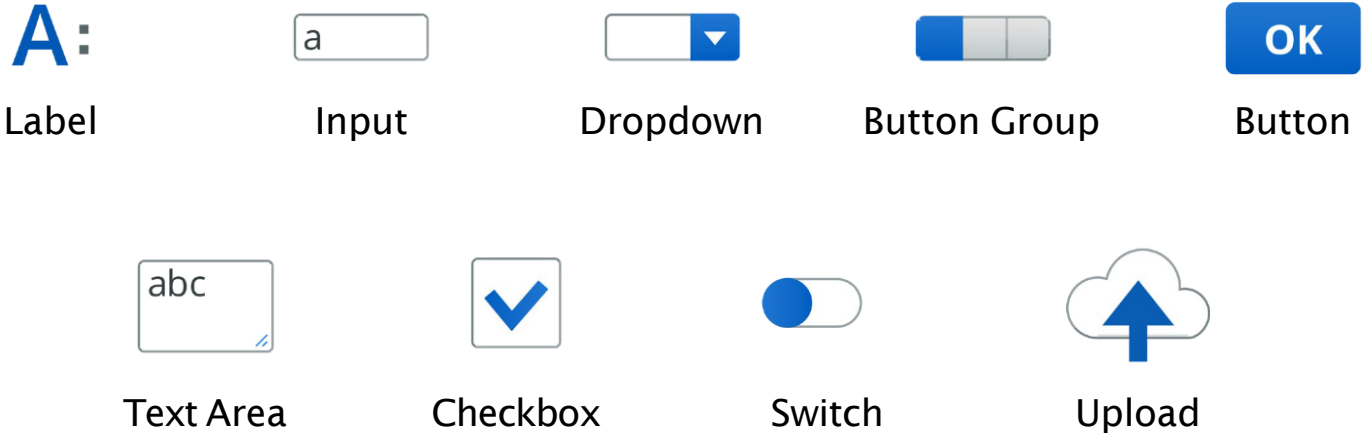
HTML Element: Allows adding a custom HTML element to the Screen

Block: Displays a reusable Screen element

Input Widgets

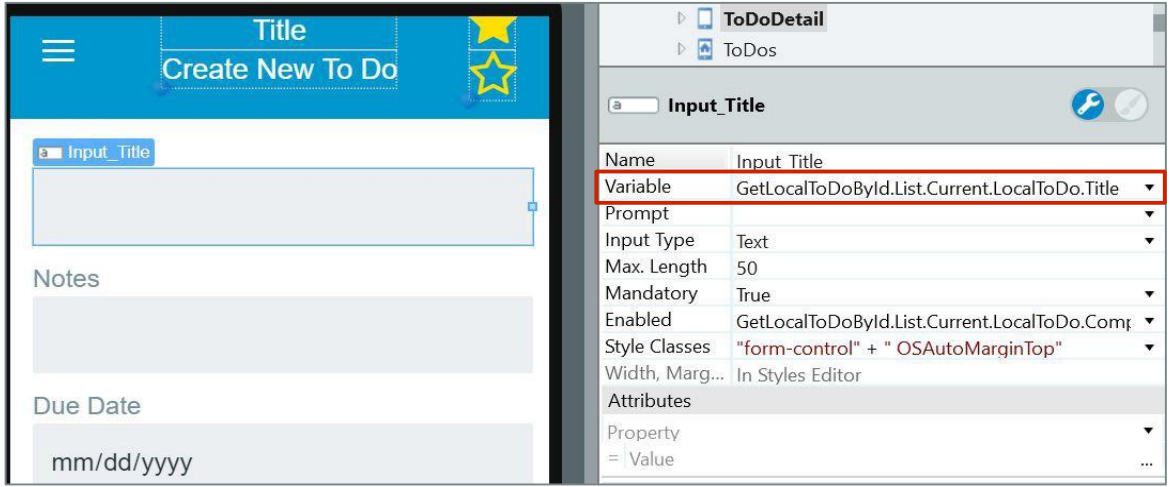
Form Widget

- Groups input widgets together
- Useful to validate user data



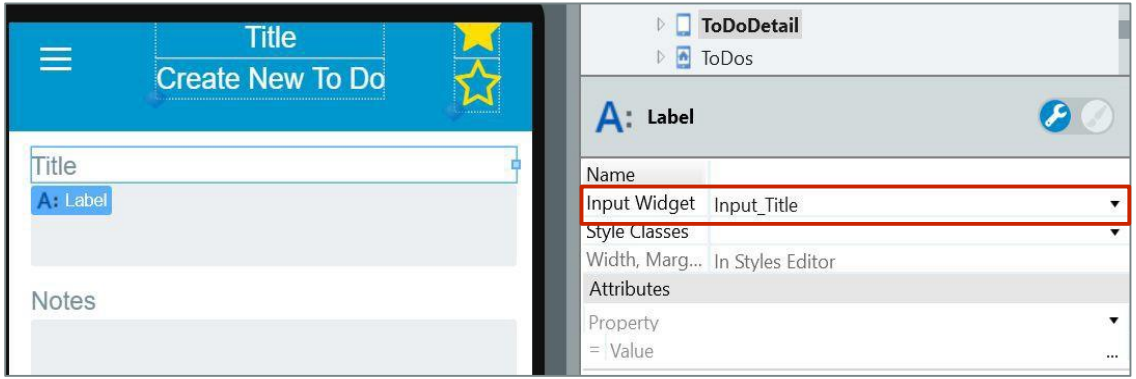
Input Widget

- Field that allows a user to submit input
- Value entered is stored in the *Variable* property (Type)
- Can be mandatory or not



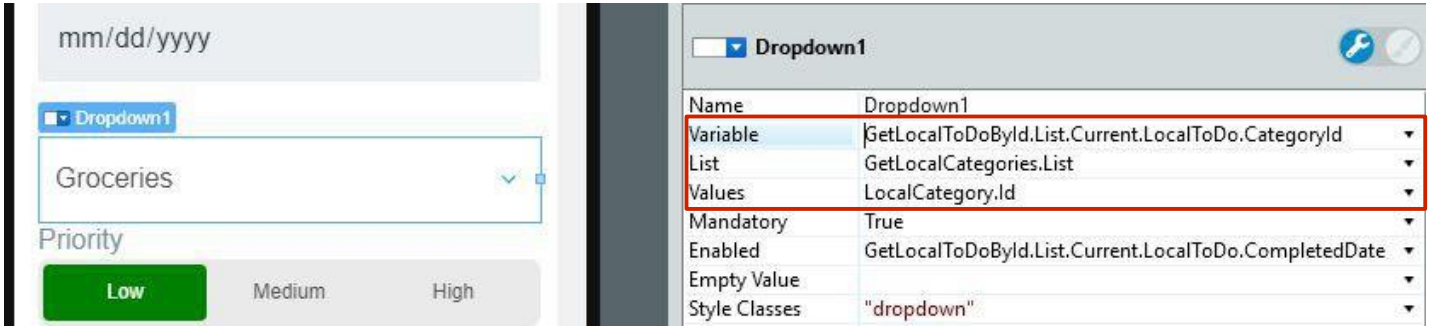
A: Label Widget

- Displays text associated with an input field
- The Input Widget property creates an association with the selected Input
  - Provides a visual cue when inputs are mandatory



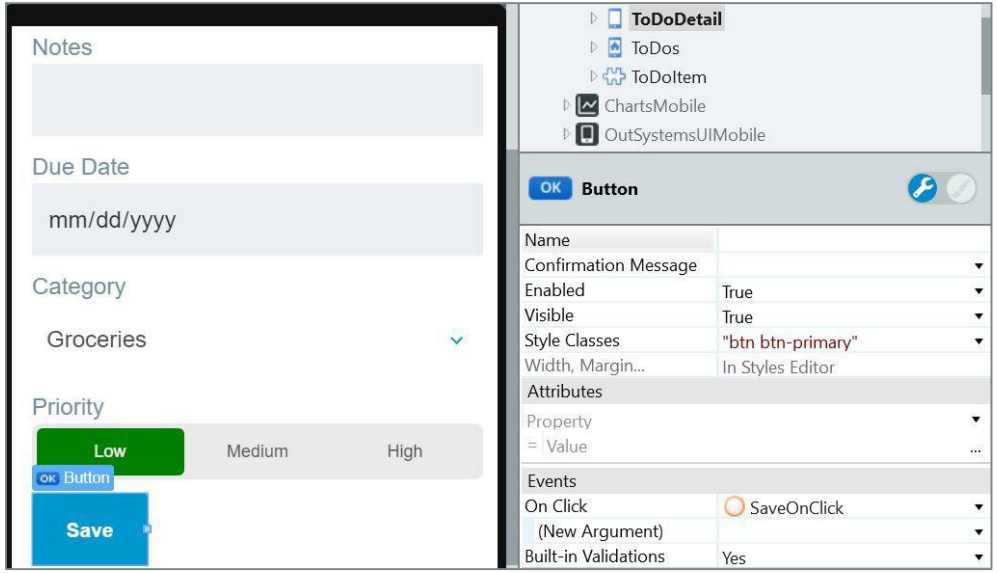
Dropdown Widget

- Dropdown list from which a user can select a value
- Value selected is stored in the *Variable* property
- *List* property specifies the list of records to show in the dropdown
  - *Values* property defines the identifier attribute of the selected value



OK Button Widget

- On Click property defines what occurs when the user clicks the Button
  - Executes a Screen Action
  - Navigates to a Screen / External URL



Other Input Widgets

Text Area: Multi-line input field for user to input data

Checkbox: Field allowing a user to check or uncheck an option

Switch: Toggle control to select between 2 options

Upload: Control allowing users to select a file

Variables in OutSystems

Topics

- Variables
  - Input Parameter
  - Output Parameter
  - Local Variable

Variables

Variables are locations **in memory** that can hold data

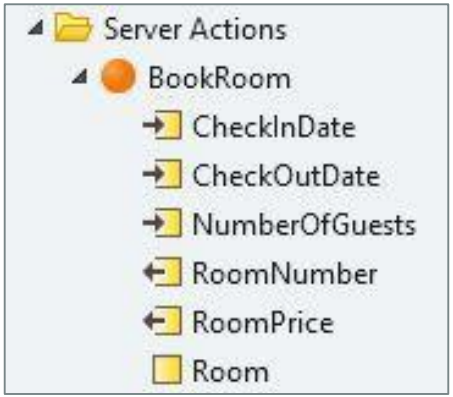
- Hold data of a particular data type
- Can be any data type

Variables are defined and exist in a particular **scope**

- Values can be accessed and modified in that scope
- If execution leaves that scope, the variable is destroyed

Variables can be:

- Input Parameters
- Output Parameters
- Local Variables



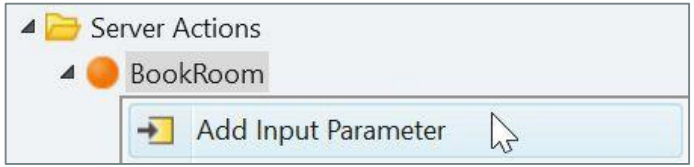
Input Parameter

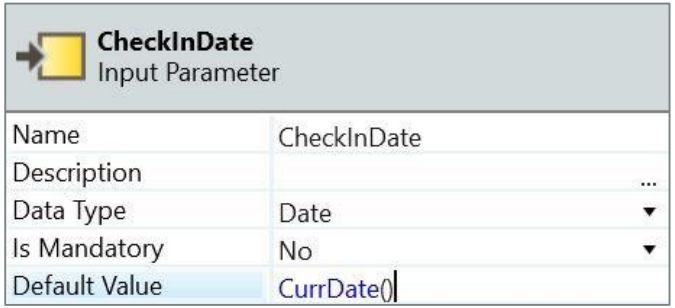
Passes a value **into** its parent's scope from the outside scope

Can be set as **Mandatory**

- Requires that the parameter must have a value assigned it

The variable is destroyed when execution leaves the scope of the parent element



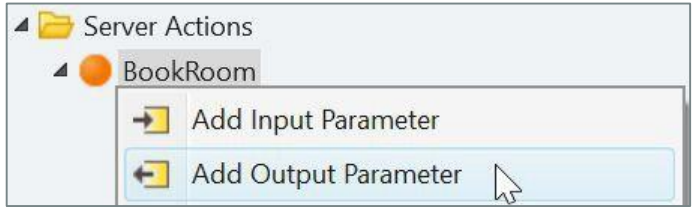


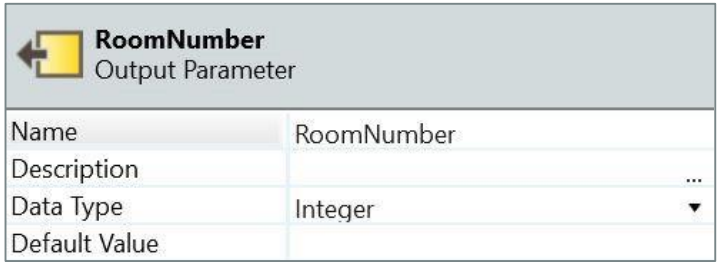
Output Parameter

Returns a value from inside its parent's scope to the outside scope

A value **must be assigned** to the Output Parameter inside its scope

The variable **continues to exist** in the outside scope even after its parent element's scope is gone



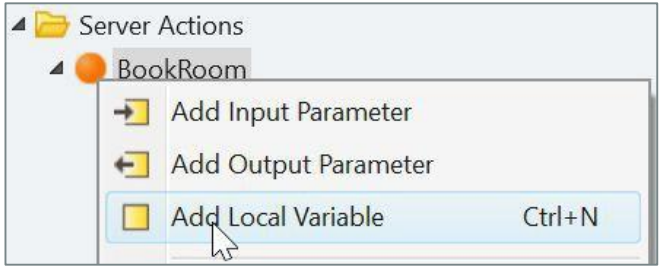


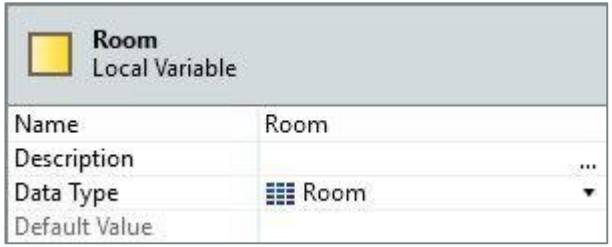
Local Variable

Exists exclusively within the scope of its parent element

Can be assigned and used “locally” inside that scope

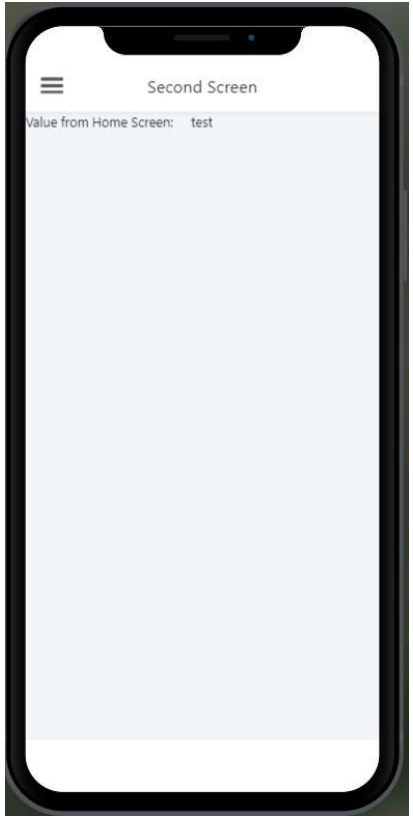
The variable is destroyed when execution leaves the scope of the parent element





Exercise Lab 2: MultiScreen App

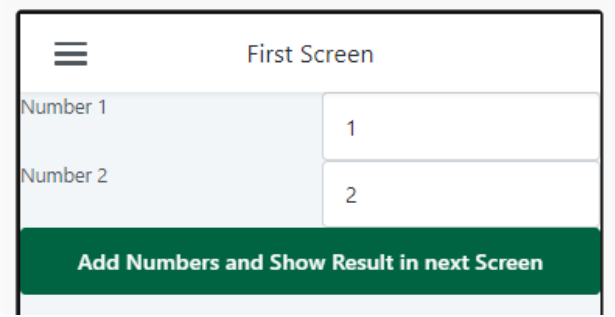
value passed from first screen to second screen

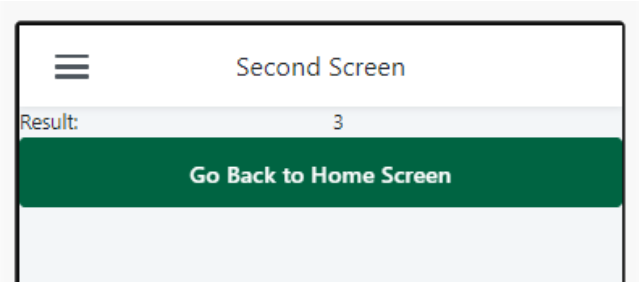


Challenge 1: Maths App

- Create an app with 2 screens
- First Screen
  - Local Variables: num1 and num2
  - 2 **input** widgets
  - A button to add the 2 numbers and display the result in 2nd screen
- Second Screen
  - An expression showing the result of adding the 2 numbers from first screen
  - A button to go back to first screen

20 Mins





OutSystems Data Modeling

Topics

- Data Modeling in OutSystems
- Entities
  - Entity Attributes
  - Basic Data Types
  - Entity Actions
- Aggregates overview
- Create Aggregates
- Aggregates
  - Sources
  - Filter
  - Sorting

Data Modeling in OutSystems

Many applications work with data that needs to be persisted in the Database

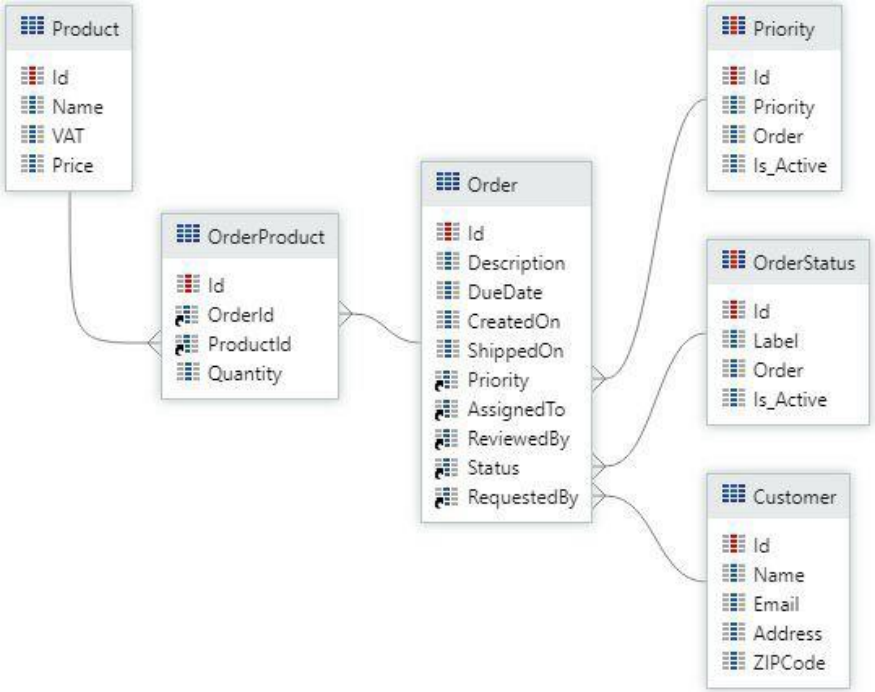
- Compound data with multiple fields

OutSystems enables persisting data with:

- Entities

Entities are created in Service Studio

- OutSystems manages the creation of the underlying Database tables



Entities

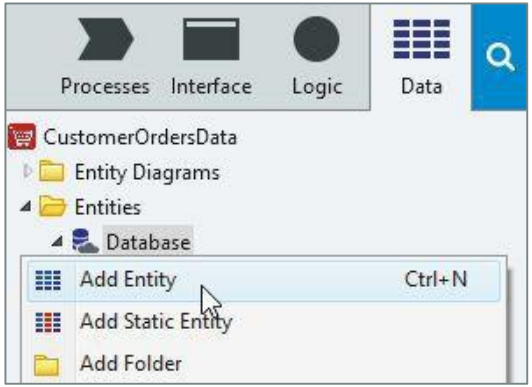
An Entity is persisted in a **Database Table**

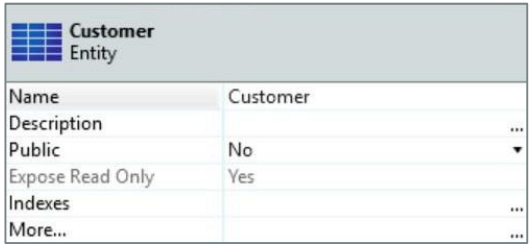
- Each new instance or record of an Entity is **inserted as a row** on the corresponding table

A different Entity should be created for each application concept

- Customer, Order and Order Item

An Entity is defined by fields called **Attributes**





Entity Attributes

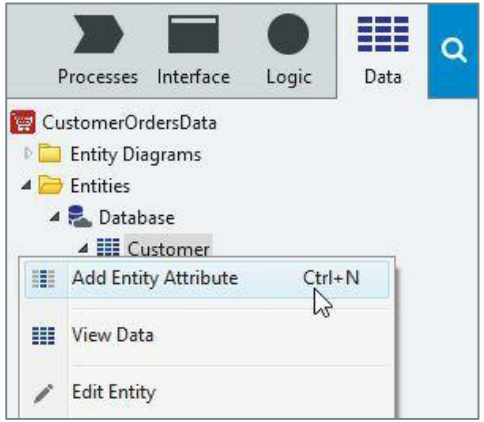
Entity data is stored in its **attributes**

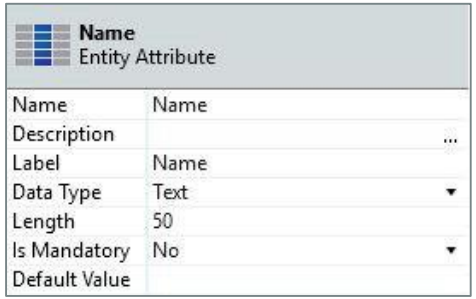
Each Attribute

- Is **a Column** on the respective database Table
- Must be one of the Basic Types

Every Entity is created with a special Id Attribute

- **Primary key** in the database
- Supports relational database operations





Basic Data Types

Family	OutSystems Data Type	Example	Default Value
Alphanumeric	**Text**	“OutSystems”	“”
	Phone Number	“+1 555 111 222”	“”
	Email	“support@mail.com”	“”
Numeric	**Integer**	42	0
	Long Integer	4200000000	0
	Decimal	11.08	0.0
	Currency	9.99	0.0
Logic	**Boolean**	True	False
Dates and Times	**Date Time**	#2011-12-13 14:15:16#	#1900-01-01 00:00:00#
	Date	#2011-12-13#	#1900-01-01#
	Time	#14:15:16#	#00:00:00#
Large Object	**Binary Data**	<Unprintable>	<0 bytes>
Referential	**Entity Identifier**	<Positive Integer or Text>	NullIdentifier()

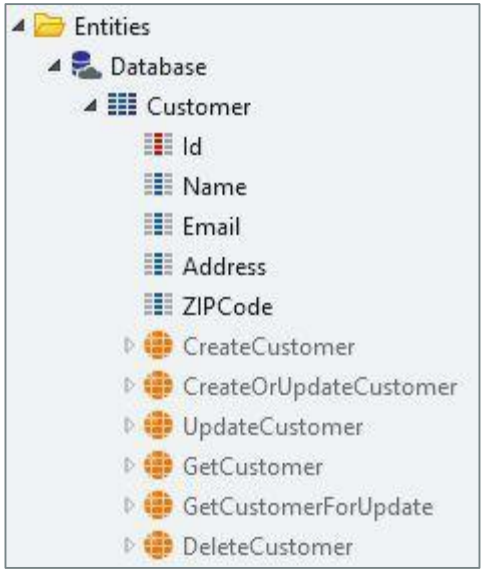
Entity Actions

OutSystems automatically creates **Entity Actions** for each of the CRUD data operations

Data operations refer to the Entity by its Name:

- **C**reate: CreateCustomer
- **R**etrieve: GetCustomer
- **U**pdate: UpdateCustomer
- **D**elete: DeleteCustomer

Entity Actions can be used directly in the application logic



Aggregates Overview

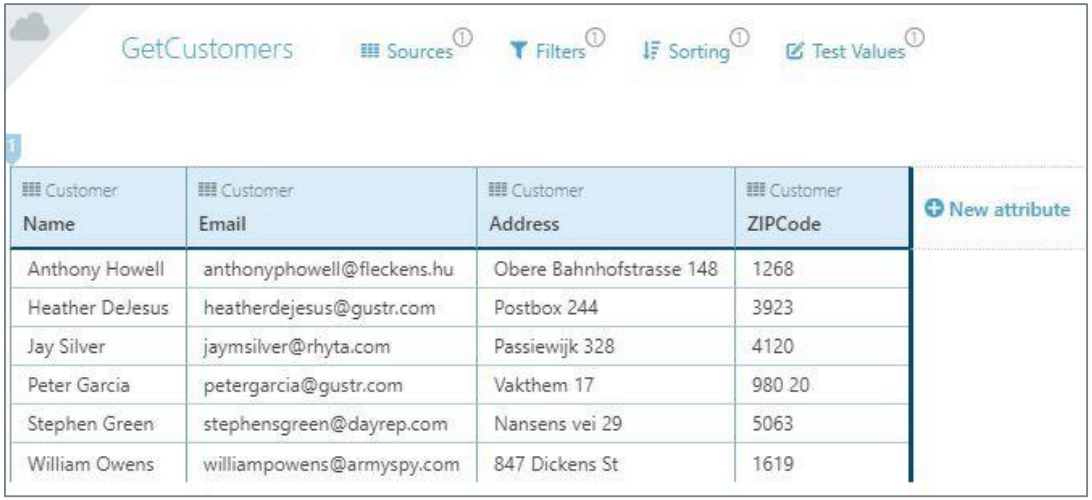
Most applications need to fetch data from the database

Aggregates allow us to define database queries in a visual way

- Add Sources
- Create filters
- Define sorting

Aggregates are easy to create and maintain

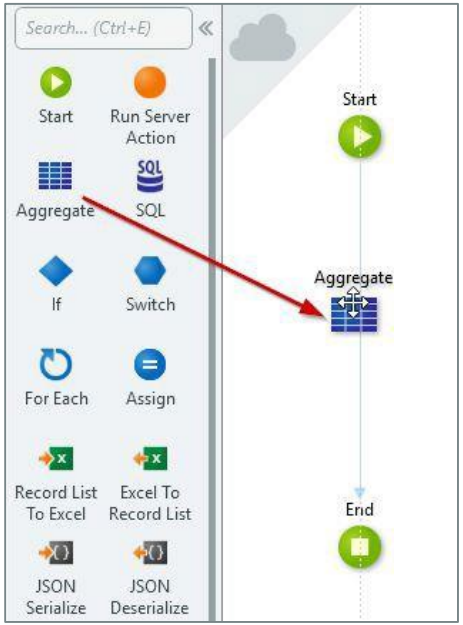
- Excel-like display of real data
- SQL knowledge is NOT required



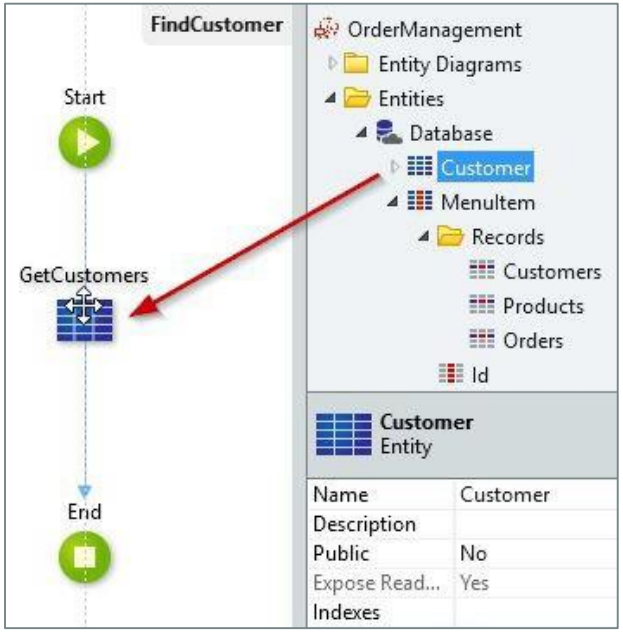
Customer Name	Customer Email	Customer Address	Customer ZIPCode
Anthony Howell	anthonyphowell@fleckens.hu	Obere Bahnhofstrasse 148	1268
Heather DeJesus	heatherdejesus@gustr.com	Postbox 244	3923
Jay Silver	jaymsilver@rhyta.com	Passiewijk 328	4120
Peter Garcia	petergarcia@gustr.com	Vakthem 17	980 20
Stephen Green	stephensgreen@dayrep.com	Nansens vei 29	5063
William Owens	williampowens@armyspy.com	847 Dickens St	1619

Creating an Aggregate

Drag an Aggregate from the Toolbox to an Action flow



Drag an Entity to an Action Flow (accelerator)

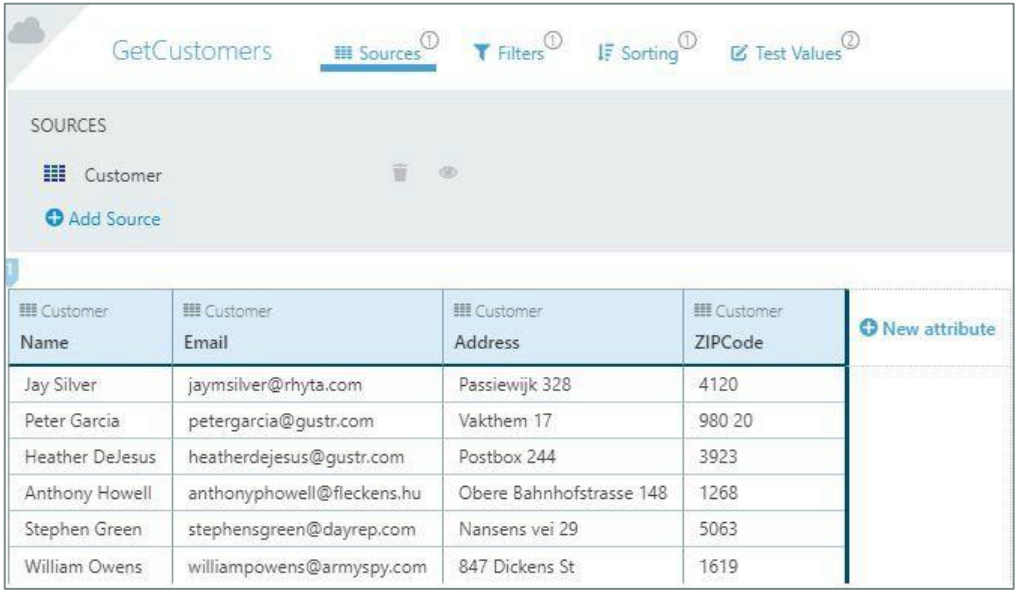


Aggregate Sources

Aggregates support one or more source Entities

The sources determine the type of the Aggregate's Output List

- e.g. Customer List

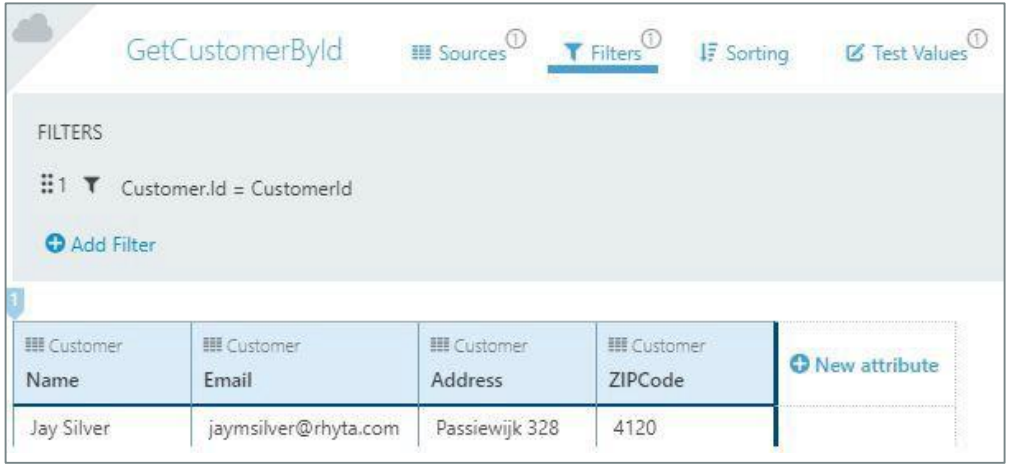


Aggregate Filters

Adds one or more conditions to the query to filter the output records

- Support for multiple filters
- Supports logical operators
  - =, <>, and, or, …
- Support for some built-in functions
  - CurrDateTime()
  - If(Condition, True, False)

Equivalent to a SQL **WHERE** Clause



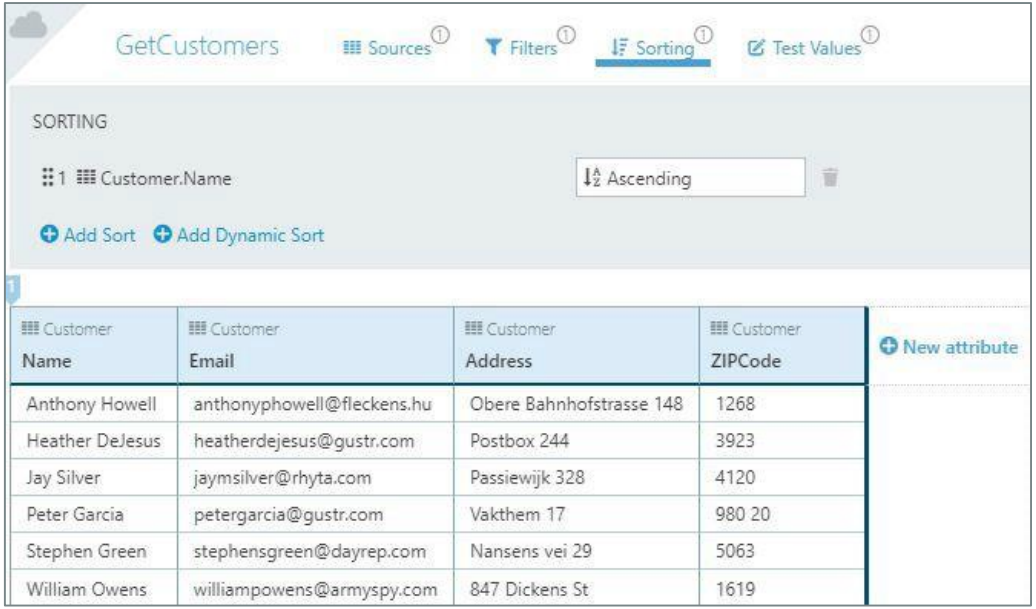
Aggregate Sorting

Defines the Entity's attribute to sort by and in which direction

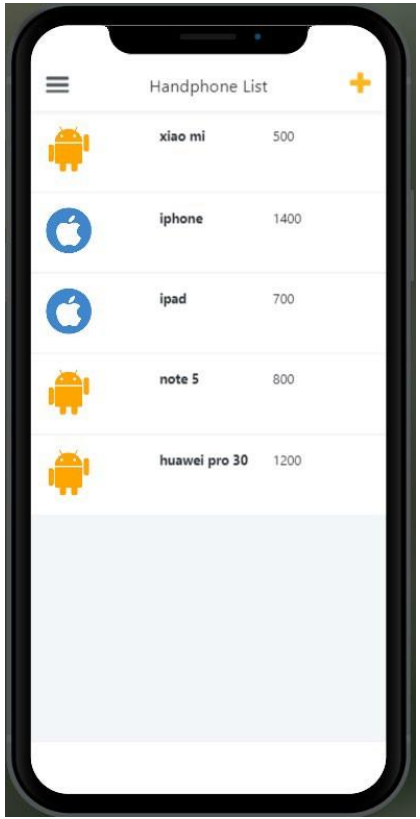
- Ascending
- Descending

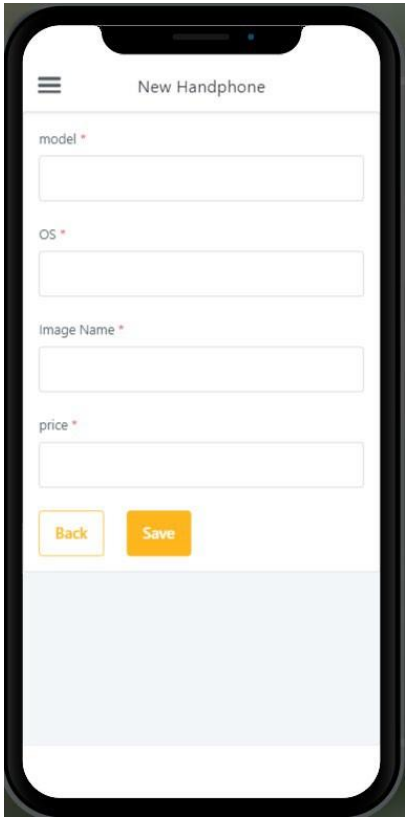
When defining multiple sorts the order is relevant to the result

Equivalent to a SQL **ORDER BY**



Exercise Lab 3: Data Driven App





End User Management

End User Management

End user of OutSystems are considered either Anonymous or Registered User

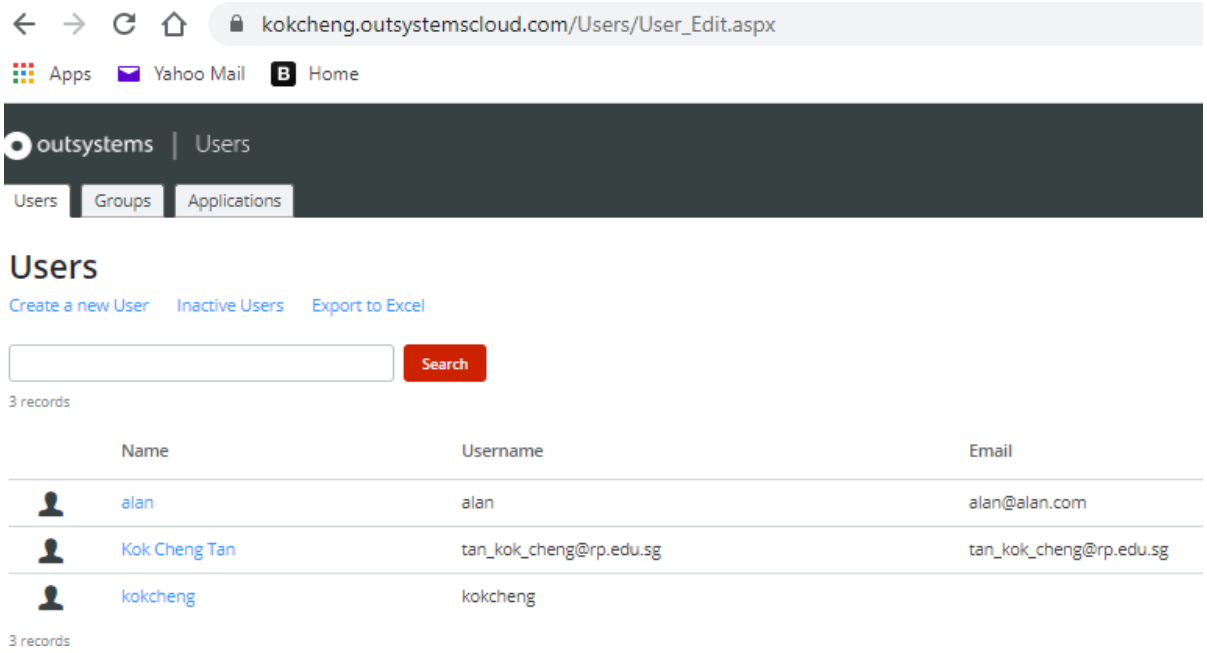
- **Anonymous Users**
  No Login required

- **Registered User**
  Need to be registered

End User Management Website

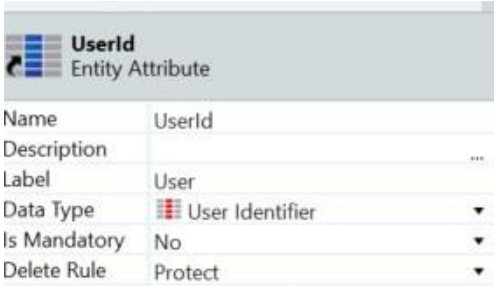
Go to http://<yourenvironmentaddress>.outsystems.com/Users

- **Create, Edit and Delete Users**



Using the Built-In User Management System

- Add UserId an an Attribute the the Database Entity
- Set the Data Type to User Identifer



Filtering Data based on Login User

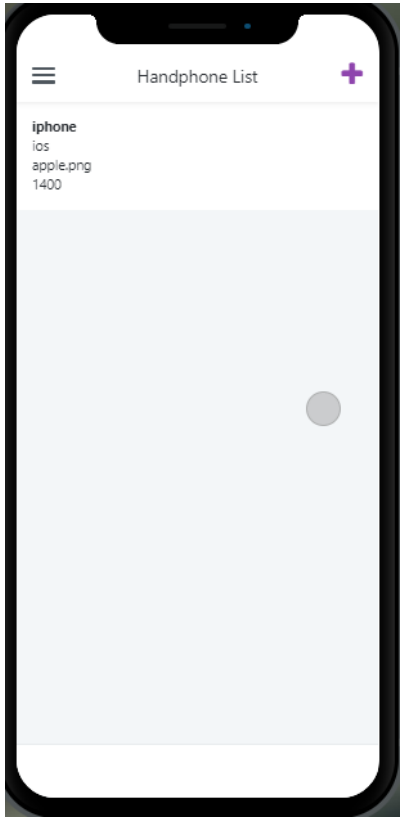
- Make use of **GetUserId()** to get the id of the current login user

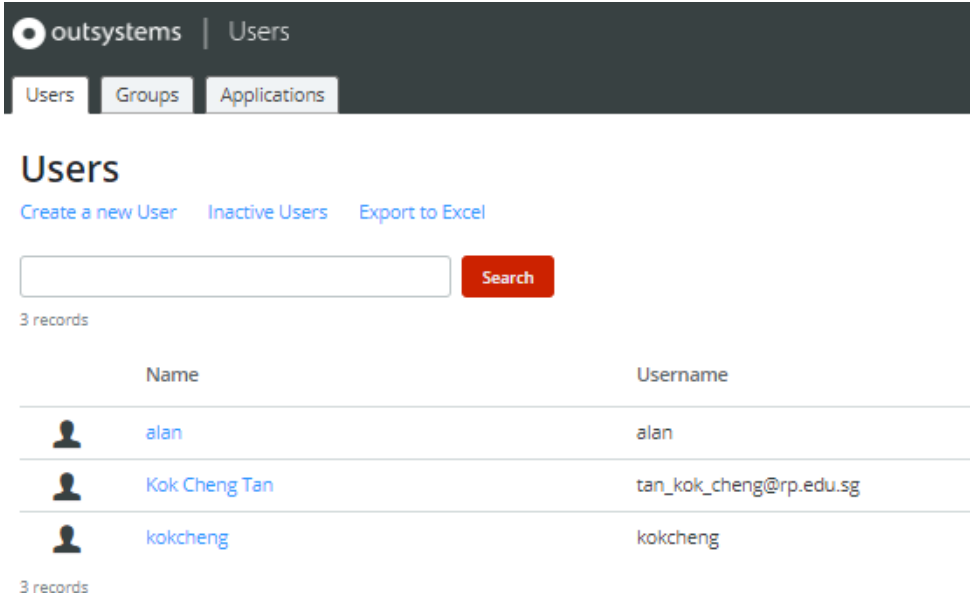
- Set up the Filters in Aggregates

Expenses item	Expenses Id	Expenses price	Expenses UserId
phone	1	500	0
tablet	2	300	0
watch	3	200	0
shoe	4	100	0

New attribute

Exercise Lab 4: LoginUser Mobile App





Distributing App

Distributing App

- Generate iOS App
- Generate Android App
- Progressive Web App (PWA)

Generate iOS App

- Needs Apple Developer Account
- Fairly complex process to generate certificate and provisioning profile

Generate Android App

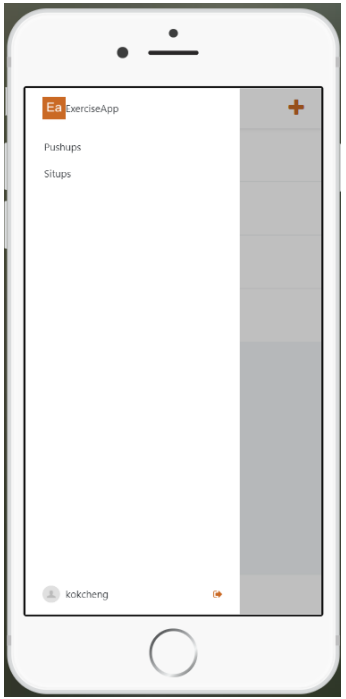
- Generate *Debug* App is fairly simple
- Generate *Release* App will require uploading of the keystore

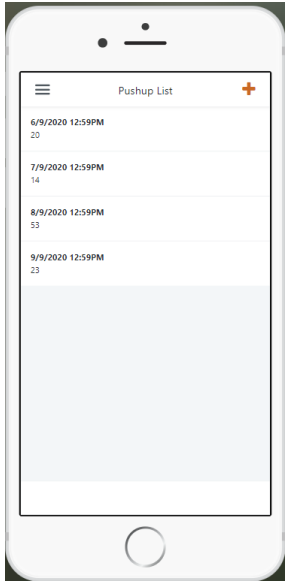
Progressive Web App (PWA)

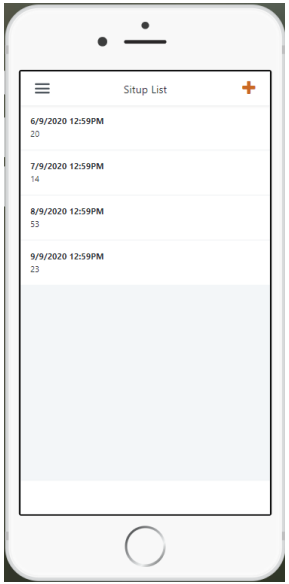
- Most convenient way to distribute app
- Access the App via url or QR code

Challenge 2: Multi-Data Source App

- Use data from 2 Excel Sheets
- Use Common Menu to provide access to different screens







End of Day 1