

Introduction to Low Code Mobile Apps Development

22 - 23 Sept 2022

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

- 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

Afternoon

- 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

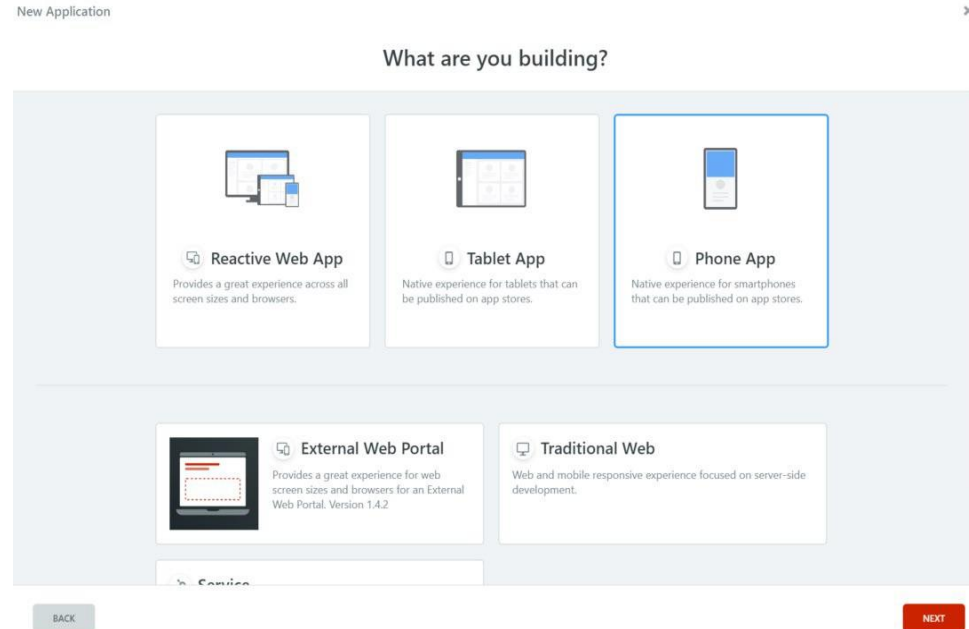
Mobile Applications in OutSystems

Topics

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

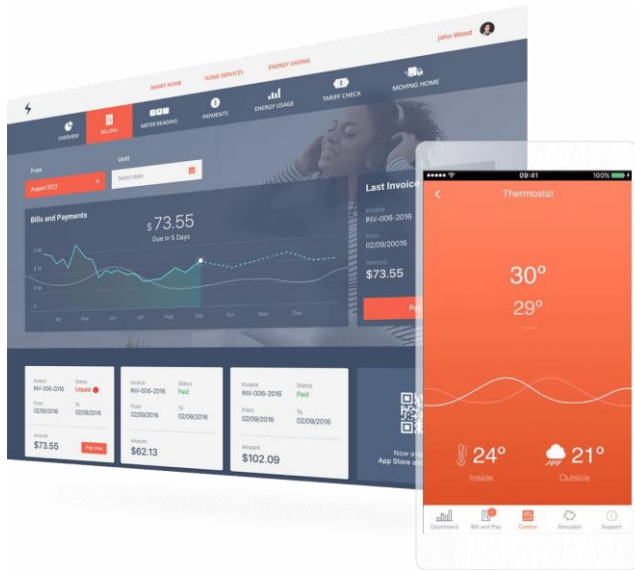
Applications types in OutSystems

- 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



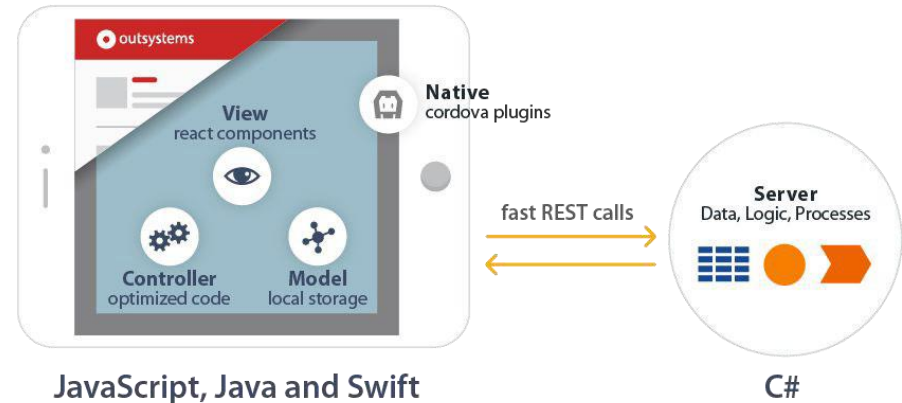
Mobile apps in OutSystems

Run on Android and iOS devices

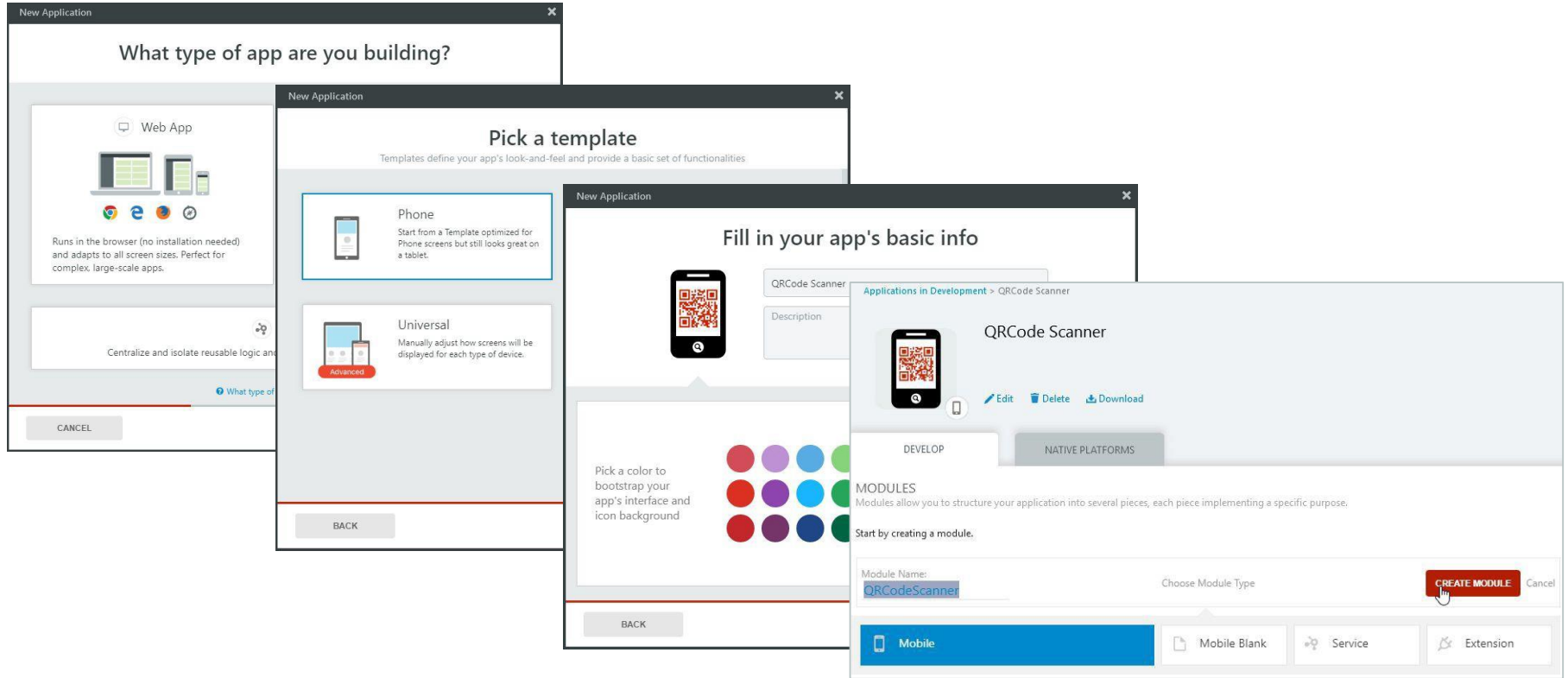


Single Page Applications (SPAs)

Cross-Platform, Standards-Based

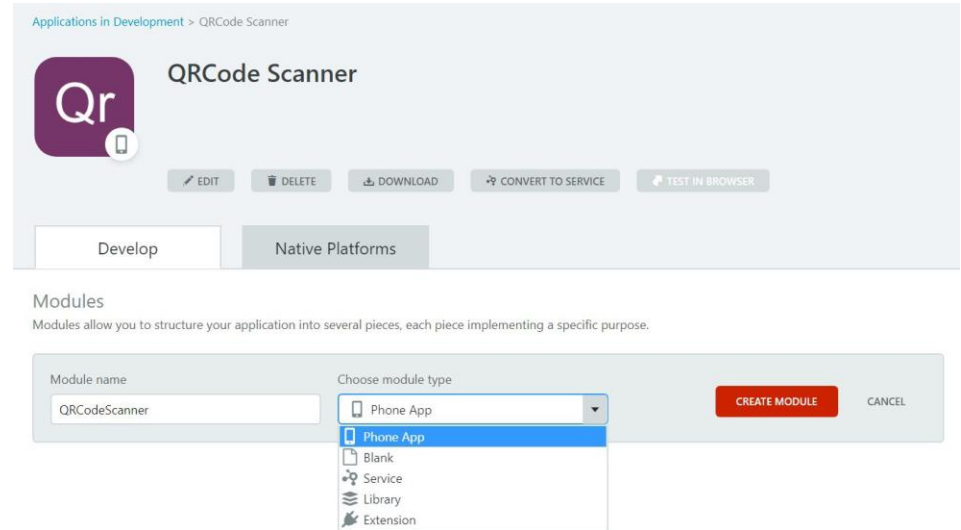


Creating a Mobile Application

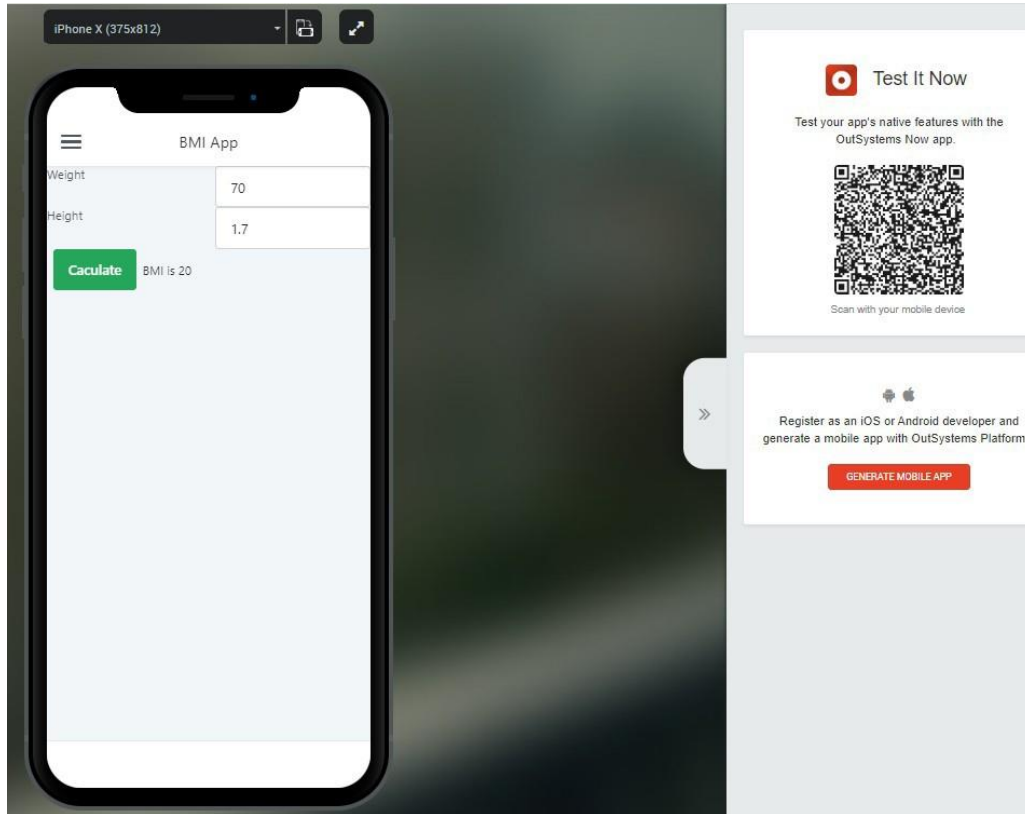


Mobile Applications

- 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

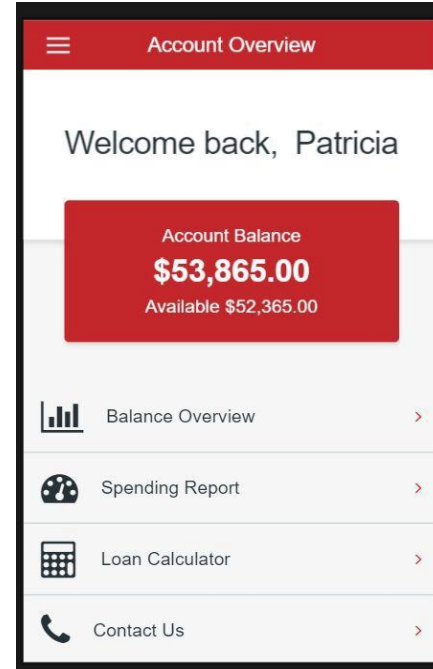
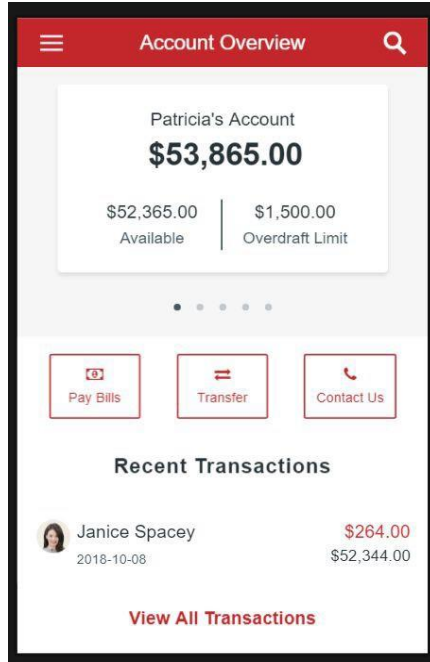
OutSystems Mobile Apps

- 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

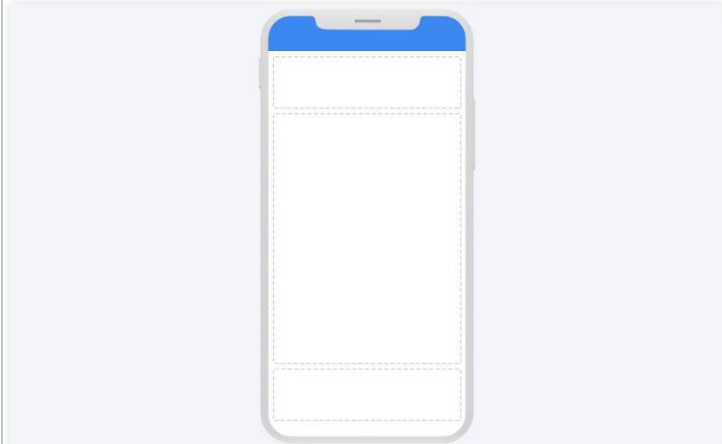
Screens define the user interface end-users interact with



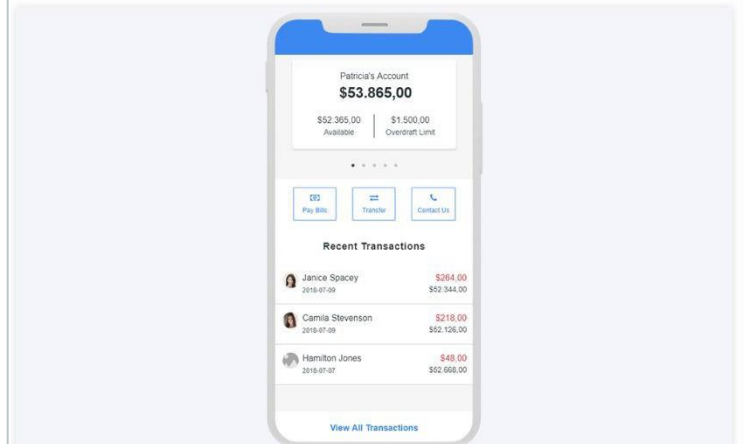
Screen Templates

Screens can be Empty or based on a Template

Empty

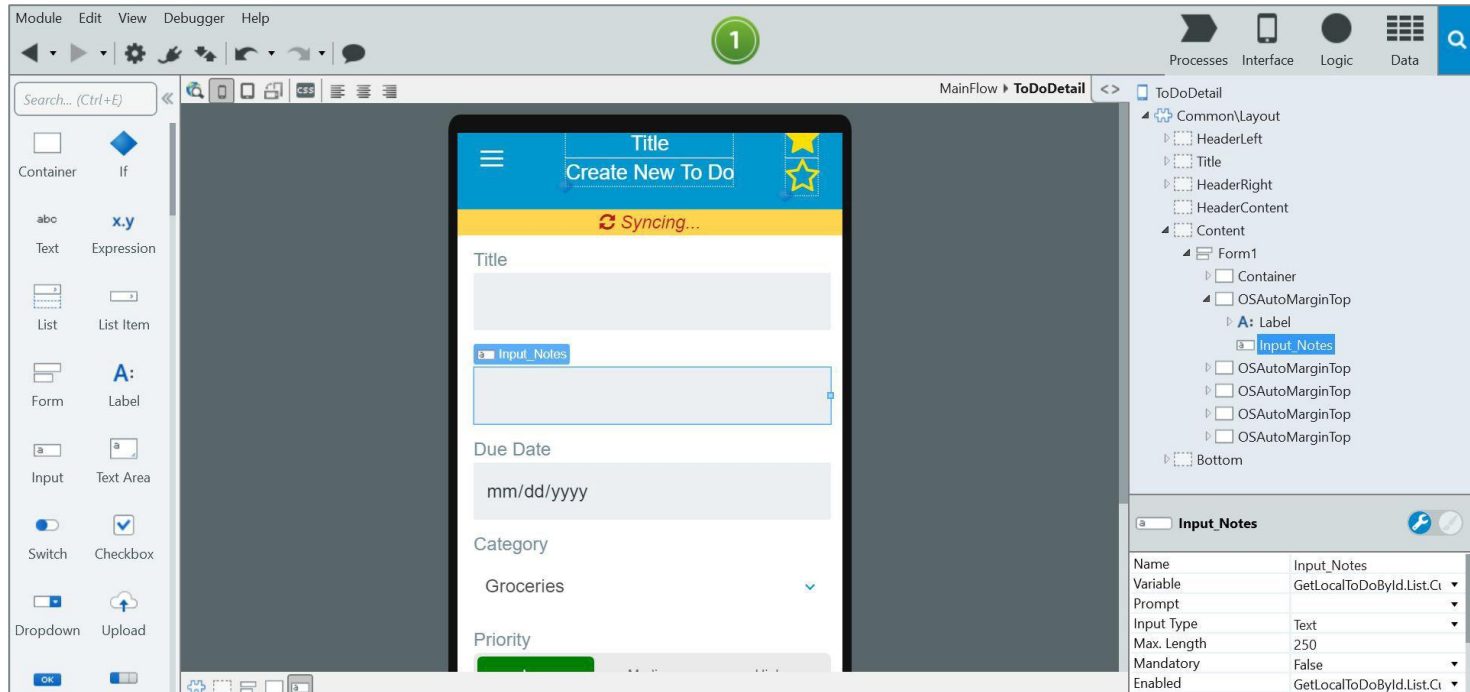


Dashboard With Carousel



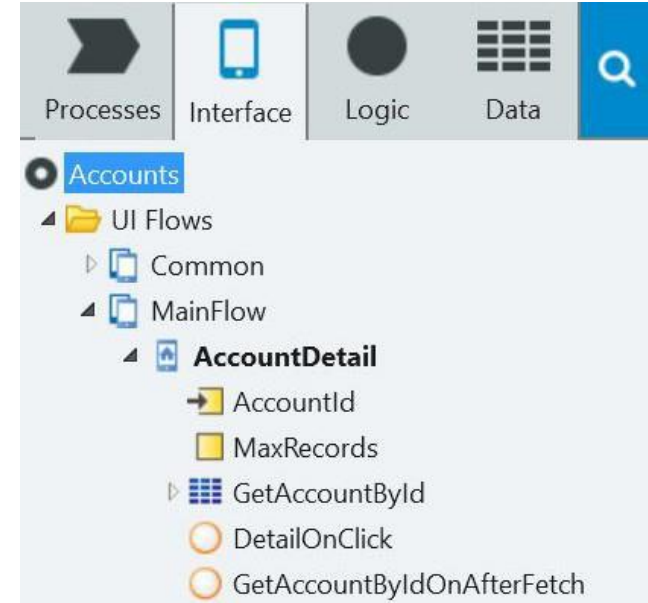
Screen Content

OutSystems Screens are built based on widgets (UI elements)



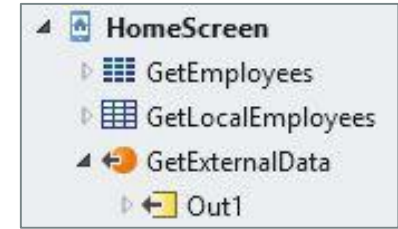
Screen Variables

- 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

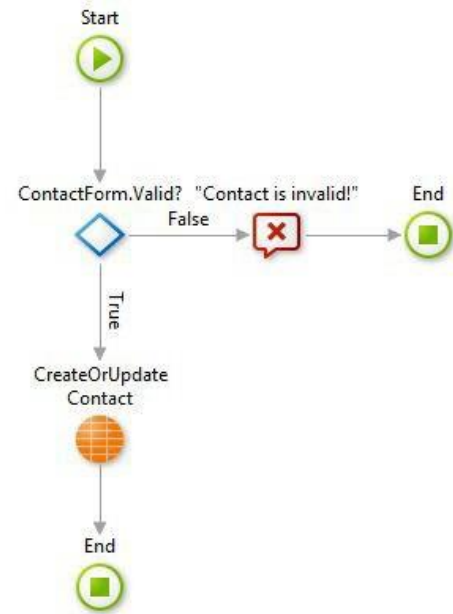
- Screens may need to display data available elsewhere
- Aggregates in the Screen
 - Database or Local Storage Entities
- Data Action for advanced cases
 - Server Action
 - Output Data Type by default is Text, but can be changed
- Queries performed asynchronously and in parallel
- Screen Lifecycle Event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



GetEmployees Aggregate	
Name	GetEmployees
Description	
Server Request Ti...	(Module Default Timeout)
Max. Records	50
Events	
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



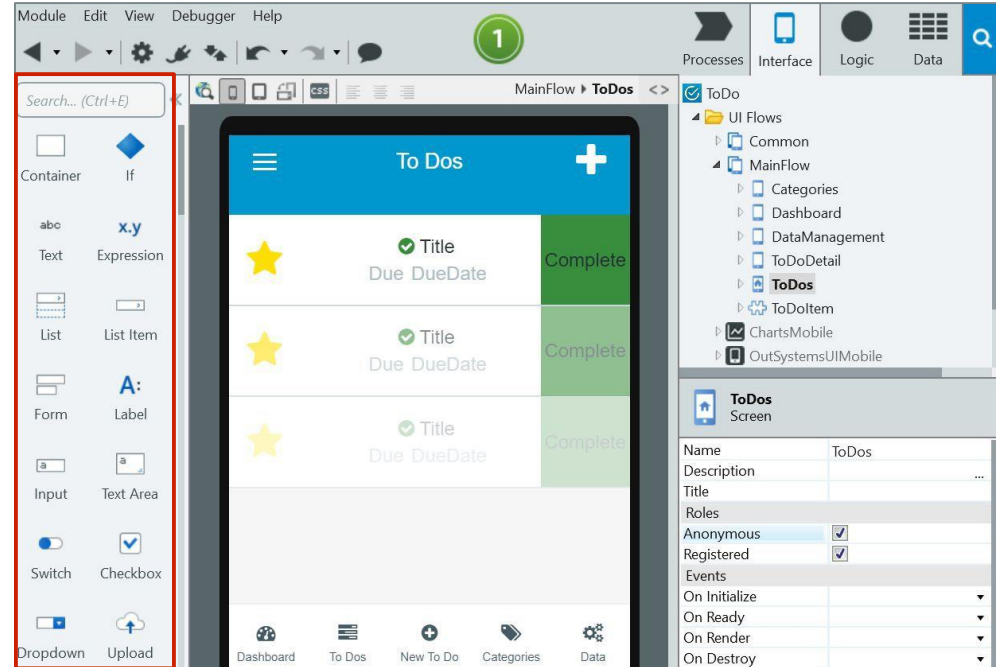
Mobile Widgets

Topics

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

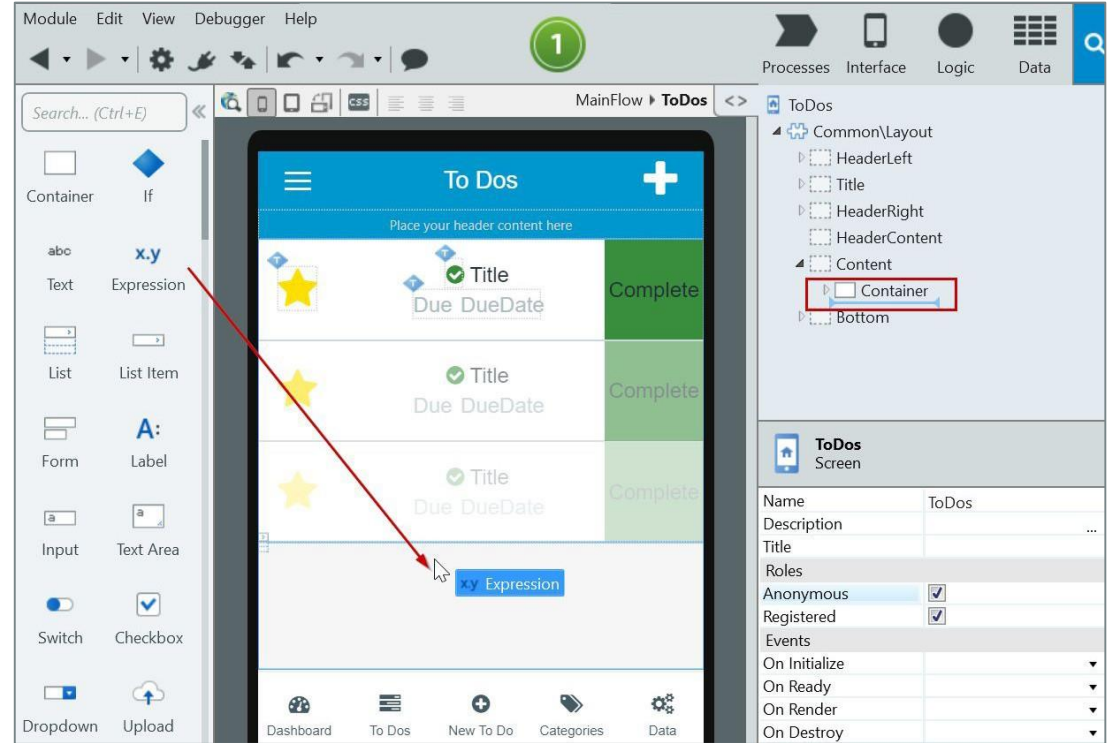
Widgets

- 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

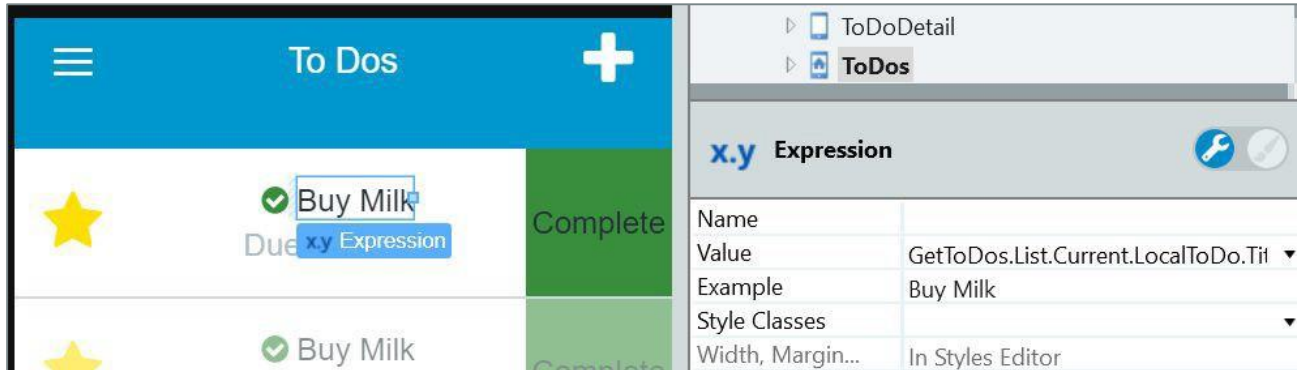
- Drag & Drop Widgets...
 - To the Canvas
 - Opens Widget Tree
 - Help guide placement
 - To Widget Tree
 - For precise placement
- Widget hierarchy
 - Near bottom of canvas
 - Shows related Widgets
 - Select to view properties



Basic Widgets

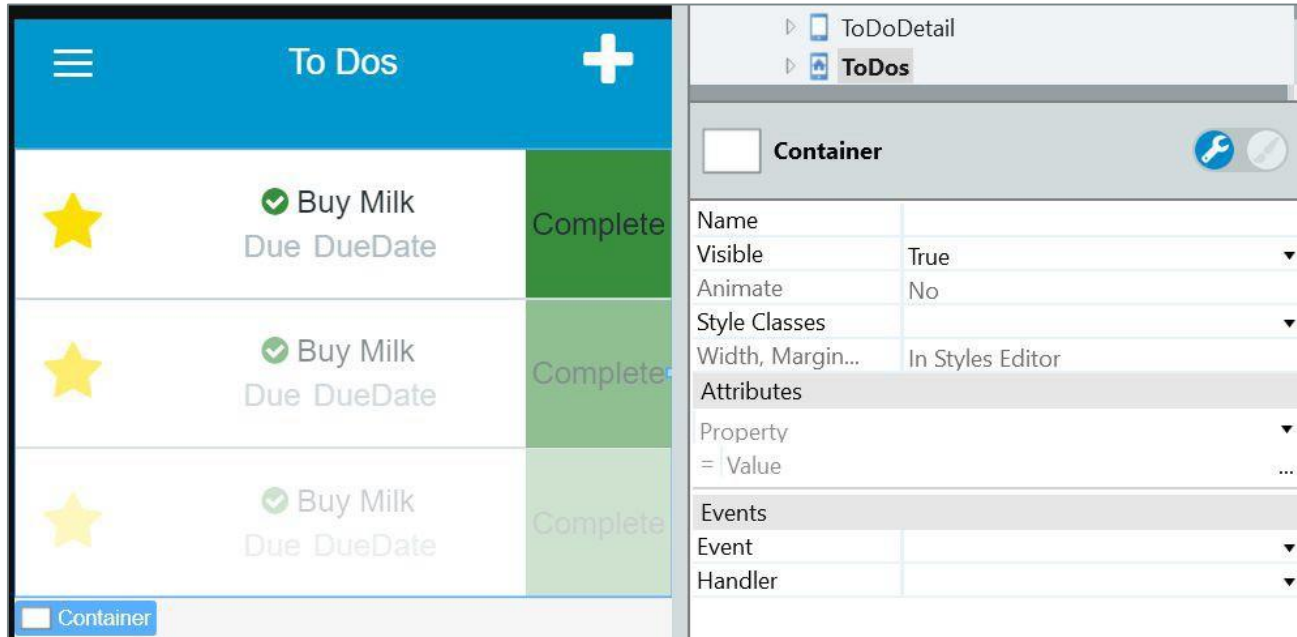
x.y Expression Widget

- 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



The image shows a mobile application interface on the left and its corresponding design tool properties panel on the right.

Mobile App Interface:

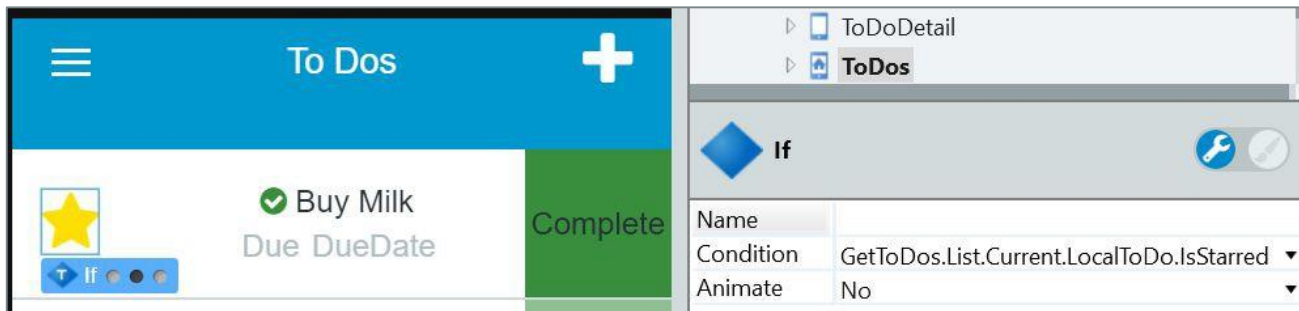
- Title Bar:** Blue header with a hamburger menu icon, the text "To Dos", and a plus icon.
- Content:** A list of three items, each consisting of a yellow star icon, a green checkmark, the text "Buy Milk", and "Due Date". Each item is followed by a green bar with the word "Complete".
- Bottom Bar:** A light blue bar with a "Container" label and a plus icon.

Design Tool Properties Panel:

- Navigation:** A tree view showing "ToDoDetail" and "ToDoS".
- Widget Header:** A grey bar with a "Container" label, a plus icon, and a settings icon.
- Properties:**
 - Name:** Container
 - Visible:** True
 - Animate:** No
 - Style Classes:** (Dropdown menu)
 - Width, Margin...:** In Styles Editor
 - Attributes:** (Section header)
 - Property:** (Dropdown menu)
 - Value:** (Text input)
 - Events:** (Section header)
 - Event:** (Dropdown menu)
 - Handler:** (Dropdown menu)

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)

The screenshot shows a mobile application interface on the left and its configuration panel on the right.

Mobile App Interface:

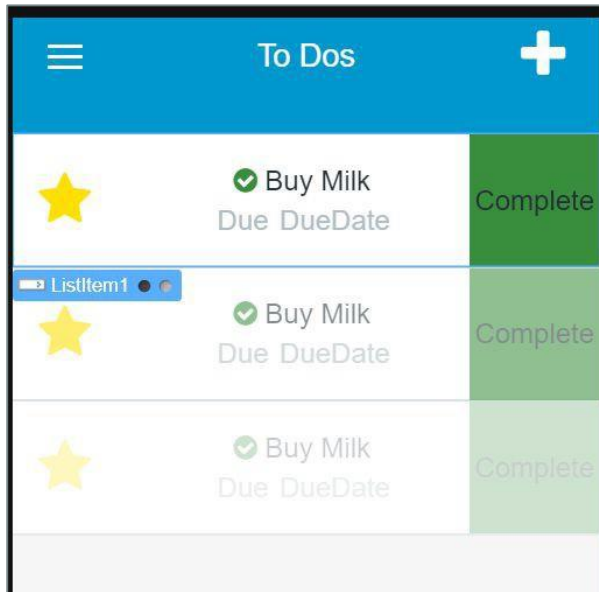
- Title Bar:** Blue header with a hamburger menu icon, the text "To Dos", and a plus icon.
- List:** A scrollable list of three items. Each item has a yellow star icon, a green checkmark, the text "Buy Milk", and "Due DueDate". A green bar on the right of each item indicates it is "Complete".
- Bottom Bar:** A blue bar with a "List" label and three colored dots (blue, black, grey).

Configuration Panel:

- Navigation:** Shows a hierarchy with "ToDoDetail" and "ToDos".
- Widget:** A "List" widget is selected, shown with its icon and a settings icon.
- Properties:**
 - Name: (empty)
 - Source: GetToDos.List
 - Animate Items: Yes
 - Mode: Default
 - Tag: div
 - Style Classes: "list list-group"
 - Width, Margin...: In Styles Editor
- Attributes:**
 - Property: (dropdown)
 - = Value: (dropdown)
- Events:**
 - On Scroll Ending: (dropdown)

ListItem Widget

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



The screenshot shows an Android application interface with a blue header bar labeled "To Dos". Below the header, there is a list of three items. Each item consists of a yellow star icon, a green checkmark, the text "Buy Milk", and a "Due Date" label. The list is titled "To Dos" and has a blue header bar. A "ListItem1" widget is highlighted in the list.

The right side of the image shows the configuration panel for the "ListItem1" widget. The panel includes a search bar, a list of widgets, and a table of properties and events.

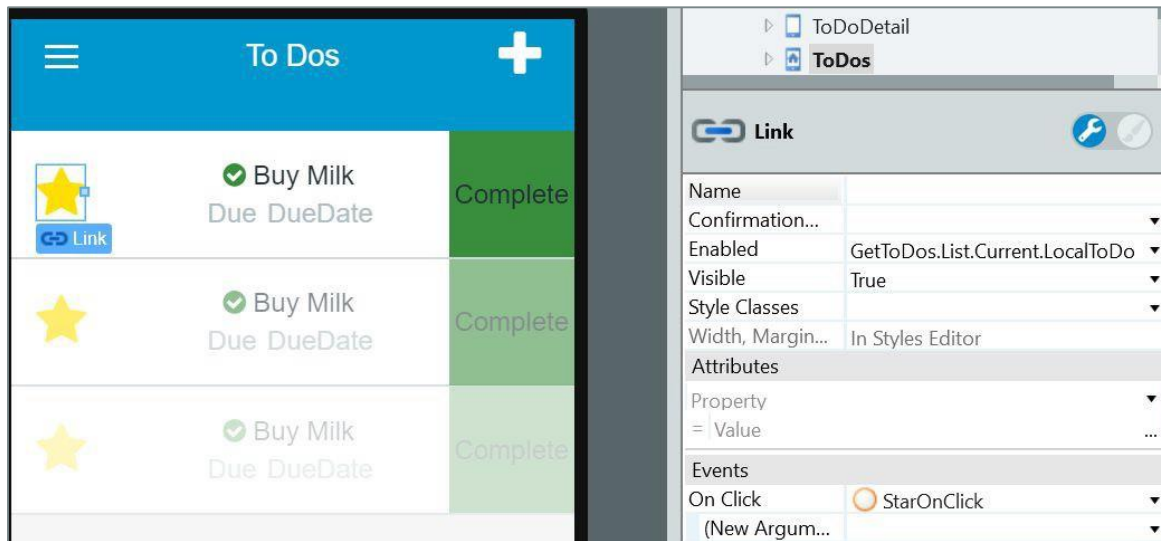
Properties	
Name	ListItem1
Full Swipe Left	Yes
Full Swipe Right	Yes
Style Classes	"list-item"
Width, Margin...	In Styles Editor

Attributes	
Property	Value

Events	
On Click	MainFlow\ToDoDetail
LocalToDoId	GetToDos.List.Current.LocalToDo.Id
(New Argument)	
Transition	(Module Transition)

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

A large blue capital letter 'A' followed by a colon, representing a label widget.

Label

A rectangular text input field containing the lowercase letter 'a', representing an input widget.

Input



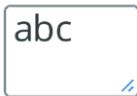
Dropdown



Button Group



Button



Text Area



Checkbox



Switch



Upload

a

Input Widget

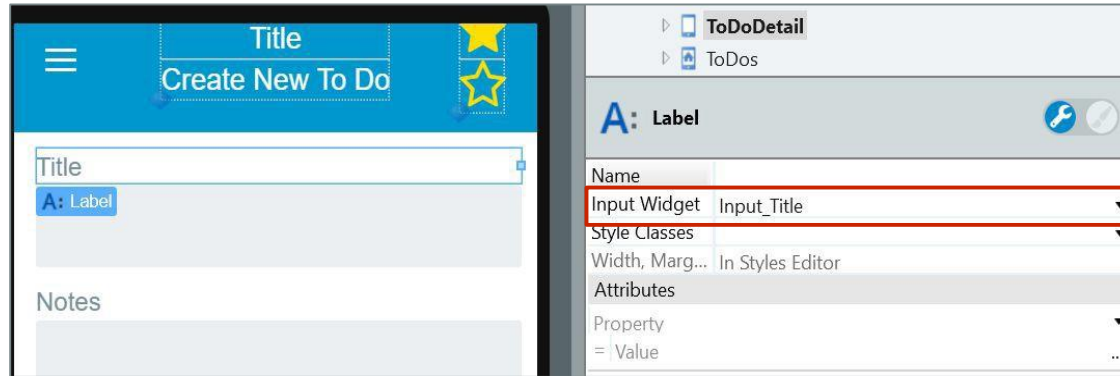
- Field that allows a user to submit input
- Value entered is stored in the *Variable* property (Type)
- Can be mandatory or not

The screenshot displays a mobile application interface for creating a new to-do item. The interface includes a blue header with a menu icon, a title 'Title', and a 'Create New To Do' button. Below the header, there is an input field for the title, a text area for notes, and a date picker for the due date. The configuration panel on the right shows the settings for the 'Input Title' widget.

Name	Input Title
Variable	GetLocalToDoById.List.Current.LocalToDo.Title
Prompt	
Input Type	Text
Max. Length	50
Mandatory	True
Enabled	GetLocalToDoById.List.Current.LocalToDo.Com...
Style Classes	"form-control" + " OSAutoMarginTop"
Width, Marg...	In Styles Editor
Attributes	
Property	= Value

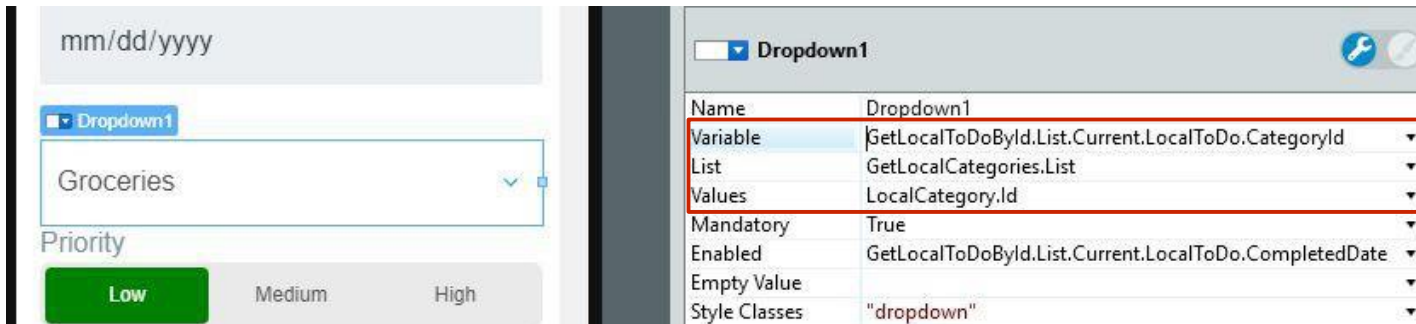
A: Label Widget

- 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



Dropdown1	
Name	Dropdown1
Variable	GetLocalToDoById.List.Current.LocalToDo.CategoryId
List	GetLocalCategories.List
Values	LocalCategory.Id
Mandatory	True
Enabled	GetLocalToDoById.List.Current.LocalToDo.CompletedDate
Empty Value	
Style Classes	"dropdown"

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

The screenshot displays a mobile application interface on the left and a configuration panel for a 'Button' widget on the right.

Mobile App Interface (Left):

- Notes:** A text input field.
- Due Date:** A date input field with the placeholder 'mm/dd/yyyy'.
- Category:** A dropdown menu with 'Groceries' selected.
- Priority:** Three radio button options: 'Low' (selected), 'Medium', and 'High'.
- Buttons:** Two buttons are visible at the bottom: 'OK Button' (blue) and 'Save' (blue).

Widget Configuration Panel (Right):

The panel shows a tree view of the app structure with 'ToDoDetail' selected. Below the tree, the 'Button' widget is configured.

Button	
Name	
Confirmation Message	
Enabled	True
Visible	True
Style Classes	"btn btn-primary"
Width, Margin...	In Styles Editor
Attributes	
Property	
= Value	...
Events	
On Click	SaveOnClick
(New Argument)	
Built-in Validations	Yes

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



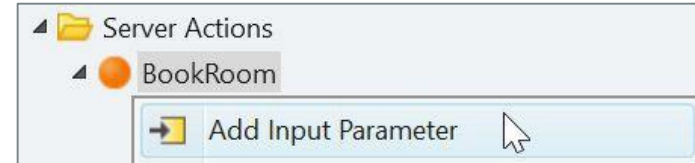
Input Parameter

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



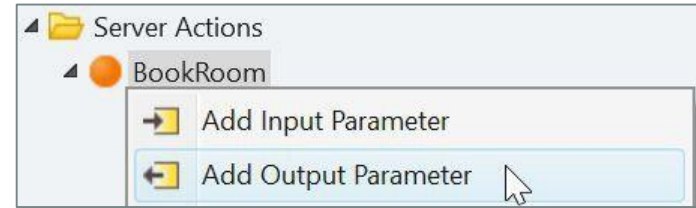
CheckInDate Input Parameter	
Name	CheckInDate
Description	...
Data Type	Date ▼
Is Mandatory	No ▼
Default Value	CurrDate()


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



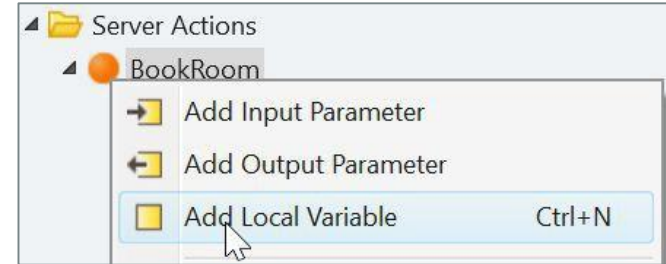
	RoomNumber Output Parameter
Name	RoomNumber
Description	...
Data Type	Integer ▼
Default Value	

Local Variable

Exists exclusively within the scope of its parent element

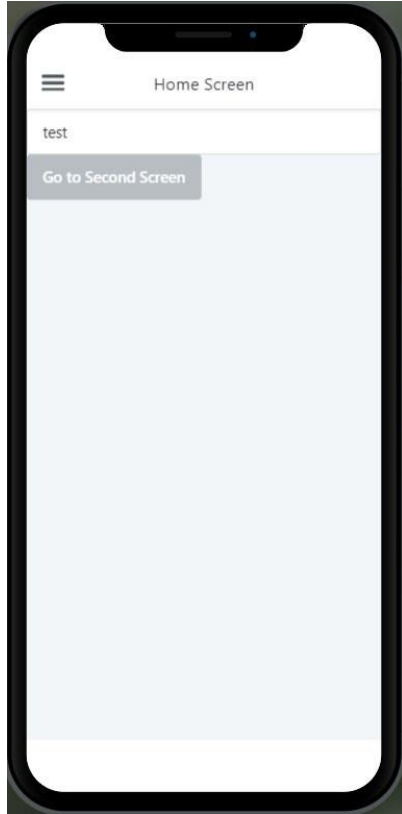
Can be assigned and used “locally” inside that scope

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

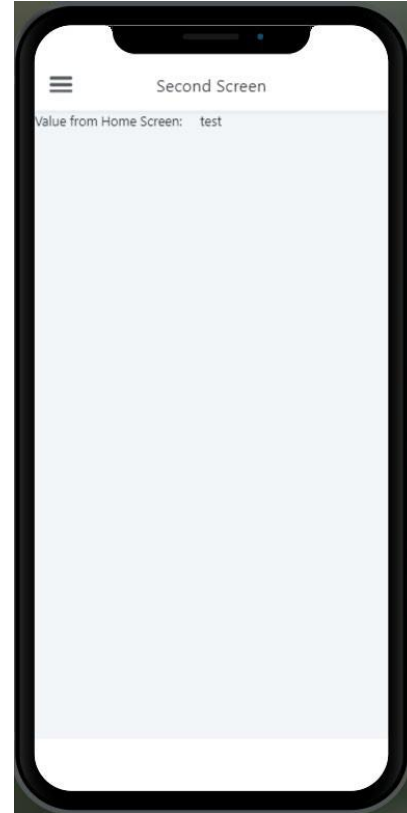


Room Local Variable	
Name	Room
Description	
Data Type	Room
Default Value	

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



20 Mins

A mockup of the 'First Screen' of the app. It features a hamburger menu icon in the top left corner. The title 'First Screen' is centered at the top. Below the title, there are two input fields: 'Number 1' with the value '1' and 'Number 2' with the value '2'. At the bottom, there is a green button with the text 'Add Numbers and Show Result in next Screen'.

- Second Screen
 - An expression showing the result of adding the 2 numbers from first screen
 - A button to go back to first screen

A mockup of the 'Second Screen' of the app. It features a hamburger menu icon in the top left corner. The title 'Second Screen' is centered at the top. Below the title, there is a 'Result:' label followed by the value '3'. At the bottom, there is a green button with the text 'Go Back to Home Screen'.

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

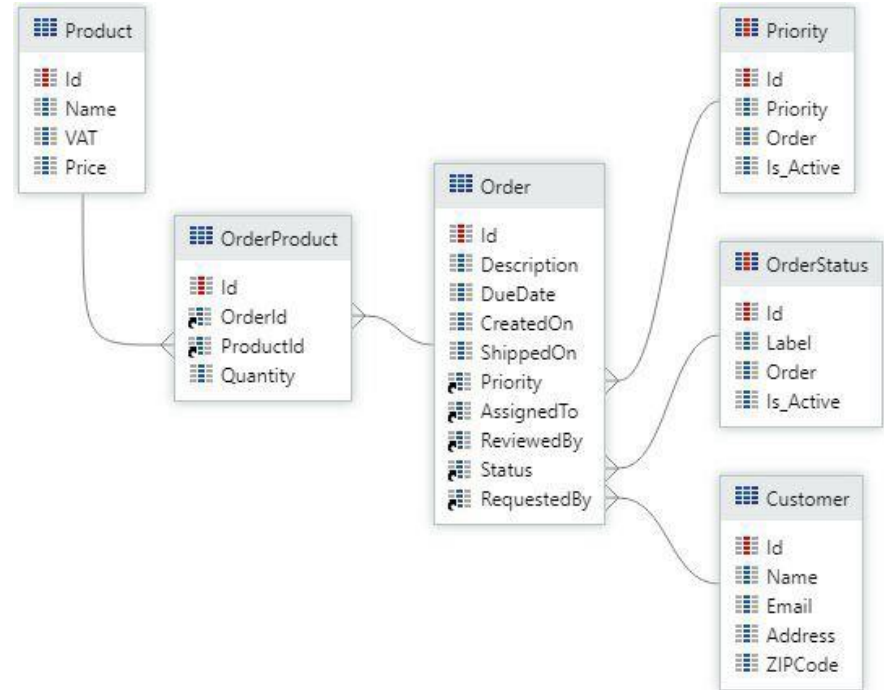
- Compound data with multiple fields

OutSystems enables persisting data with:

- Entities

Entities are created in Service Studio

- OutSystems manages the creation of the underlying Database tables



Entities

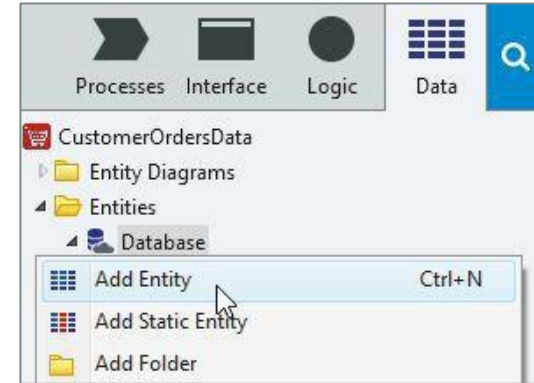
An Entity is persisted in a **Database Table**

- 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**



Customer Entity	
Name	Customer
Description	...
Public	No
Expose Read Only	Yes
Indexes	...
More...	...

Entity Attributes

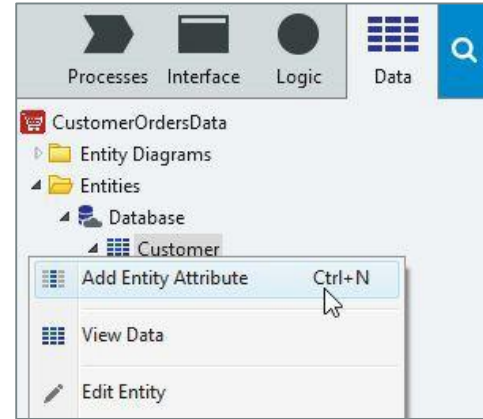
Entity data is stored in its **attributes**

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



Name Entity Attribute	
Name	Name
Description	...
Label	Name
Data Type	Text ▼
Length	50
Is Mandatory	No ▼
Default Value	

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()

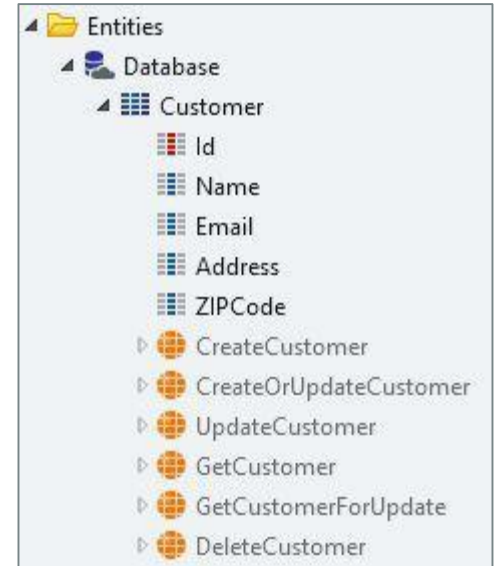
Entity Actions

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

Data operations refer to the Entity by its Name:

- **Create:** CreateCustomer
- **Retrieve:** GetCustomer
- **Update:** UpdateCustomer
- **Delete:** DeleteCustomer

Entity Actions can be used directly in the application logic



Aggregates Overview

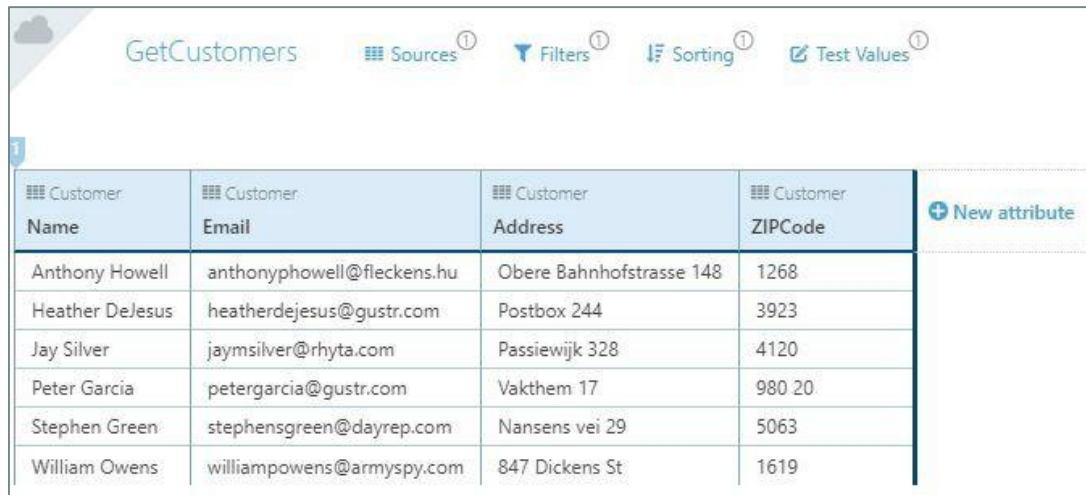
Most applications need to fetch data from the database

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

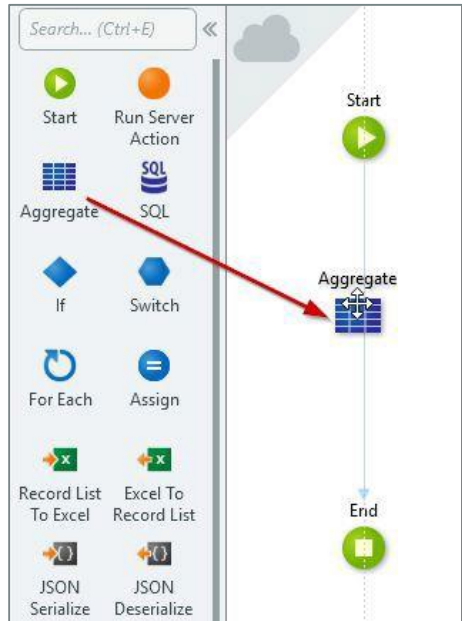


The screenshot shows the 'GetCustomers' application interface. At the top, there are icons for 'Sources', 'Filters', 'Sorting', and 'Test Values', each with a circled '1' indicating a step. Below these is a table with 4 columns: 'Name', 'Email', 'Address', and 'ZIPCode'. The table contains 6 rows of customer data. To the right of the table is a '+ New attribute' button. The table is styled with a light blue header and a light gray background for the data rows.

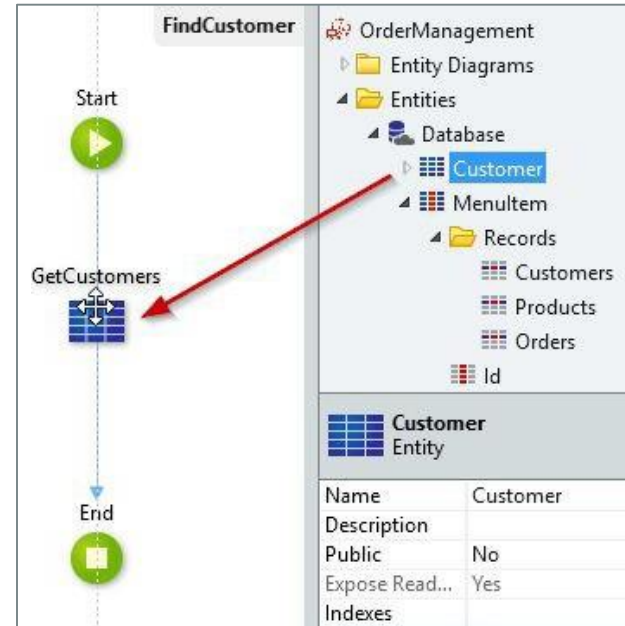
Customer Name	Customer Email	Customer Address	Customer ZIPCode	+ New attribute
Anthony Howell	anthonyphowell@fleckens.hu	Obere Bahnhofstrasse 148	1268	
Heather DeJesus	heatherdejesus@gustr.com	Postbox 244	3923	
Jay Silver	jaysilver@rhyta.com	Passiewijk 328	4120	
Peter Garcia	petergarcia@gustr.com	Vakthem 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)

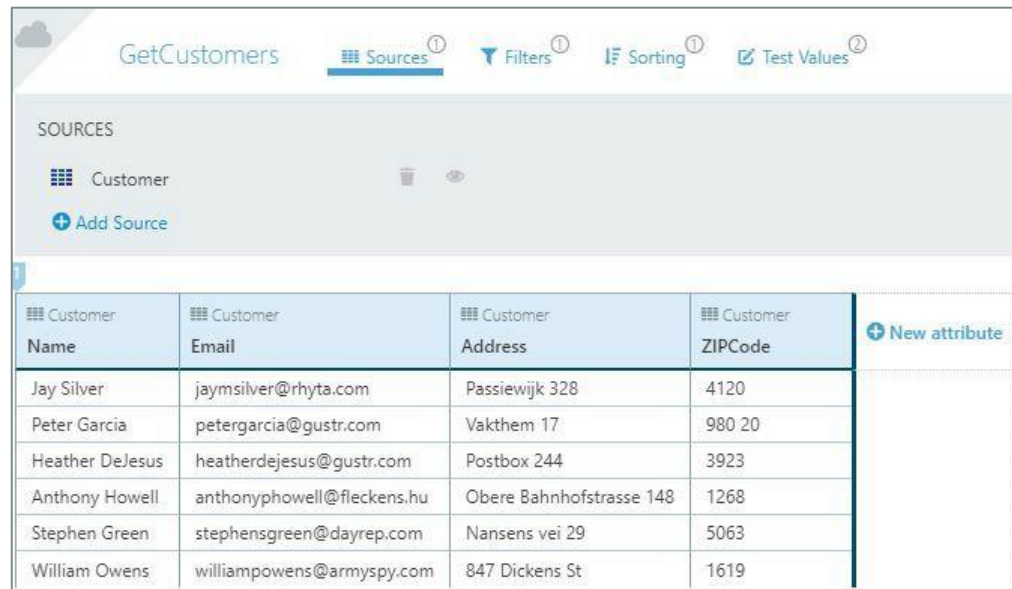


Aggregate Sources

Aggregates support one or more source Entities

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

- e.g. Customer List



The screenshot shows the 'GetCustomers' application interface. At the top, there are navigation tabs: 'Sources' (selected), 'Filters', 'Sorting', and 'Test Values'. Below the tabs, the 'SOURCES' section displays a single source entity 'Customer' with a trash icon and an 'Add Source' button. The main part of the interface is a table with columns for 'Name', 'Email', 'Address', and 'ZIPCode', each preceded by a 'Customer' entity icon. A 'New attribute' button is located to the right of the table. The table contains six rows of customer data.

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

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

The screenshot shows a query editor interface for a query named 'GetCustomerById'. The 'Filters' tab is active, displaying a single filter condition: 'Customer.Id = CustomerId'. Below the filter list, there is a table with four columns: 'Name', 'Email', 'Address', and 'ZIPCode'. The first row of data shows 'Jay Silver', 'jaymsilver@rhyta.com', 'Passiewijk 328', and '4120'. A 'New attribute' button is visible on the right side of the table.

Customer Name	Customer Email	Customer Address	Customer ZIPCode	New attribute
Jay Silver	jaymsilver@rhyta.com	Passiewijk 328	4120	

Aggregate Sorting

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

- Ascending
- Descending

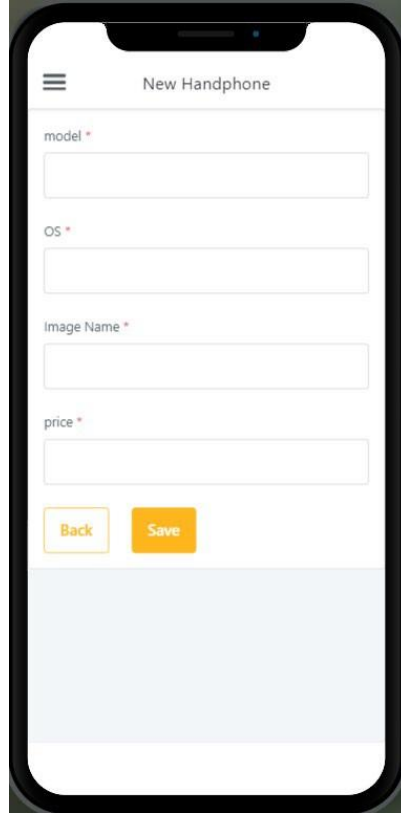
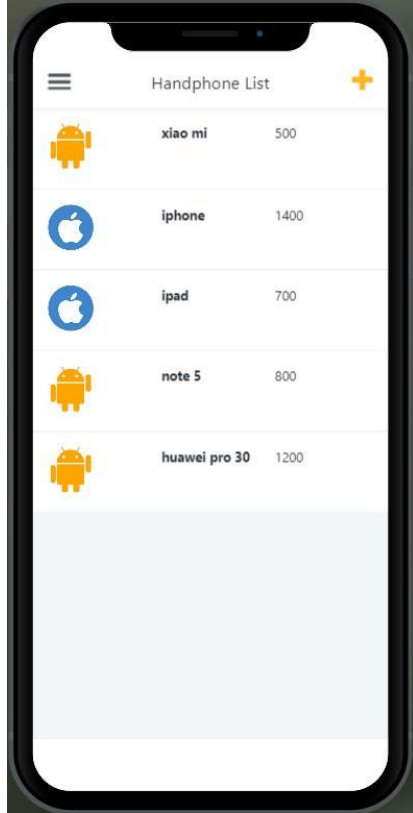
When defining multiple sorts the order is relevant to the result

Equivalent to a SQL **ORDER BY**

The screenshot shows the 'GetCustomers' interface with the 'Sorting' tab selected. The 'SORTING' section displays a single sort rule: '1 Customer.Name' with a dropdown set to 'Ascending'. Below this are buttons for '+ Add Sort' and '+ Add Dynamic Sort'. The table below shows customer data sorted by name.

Customer Name	Customer Email	Customer Address	Customer ZIPCode	+ New attribute
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	

Exercise Lab 3: Data Driven App

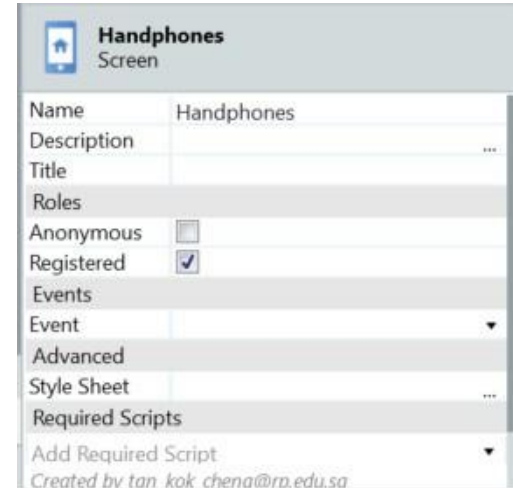


End User Management

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



The screenshot shows the configuration interface for a screen named 'Handphones'. The interface is organized into sections: Name, Description, Title, Roles, Events, Advanced, Style Sheet, and Required Scripts. The 'Roles' section has two checkboxes: 'Anonymous' (unchecked) and 'Registered' (checked). The 'Events' section has a dropdown menu for 'Event'. The 'Advanced' section has a dropdown menu for 'Style Sheet'. The 'Required Scripts' section has a dropdown menu for 'Add Required Script'. The footer of the interface reads 'Created by tan kok.chena@rp.edu.sg'.

Handphones Screen	
Name	Handphones
Description	
Title	
Roles	
Anonymous	<input type="checkbox"/>
Registered	<input checked="" type="checkbox"/>
Events	
Event	
Advanced	
Style Sheet	
Required Scripts	
Add Required Script	

Created by tan kok.chena@rp.edu.sg

End User Management Website

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

- **Create, Edit and Delete Users**

The screenshot shows a web browser window with the URL `kokcheng.outsystemscloud.com/Users/User_Edit.aspx`. The browser's address bar and tabs are visible. The page header includes the Outsystems logo and the text "Users". Below the header, there are three tabs: "Users", "Groups", and "Applications". The "Users" tab is selected. The main content area is titled "Users" and includes links for "Create a new User", "Inactive Users", and "Export to Excel". A search bar with a "Search" button is present. Below the search bar, it says "3 records". A table displays the user records with columns for Name, Username, and Email. The table contains three rows of data. Below the table, it again says "3 records".

Name	Username	Email
alan	alan	alan@alan.com
Kok Cheng Tan	tan_kok_cheng@rp.edu.sg	tan_kok_cheng@rp.edu.sg
kokcheng	kokcheng	

Using the Built-In User Management System

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



UserId Entity Attribute	
Name	UserId
Description	...
Label	User
Data Type	User Identifier ▼
Is Mandatory	No ▼
Delete Rule	Protect ▼

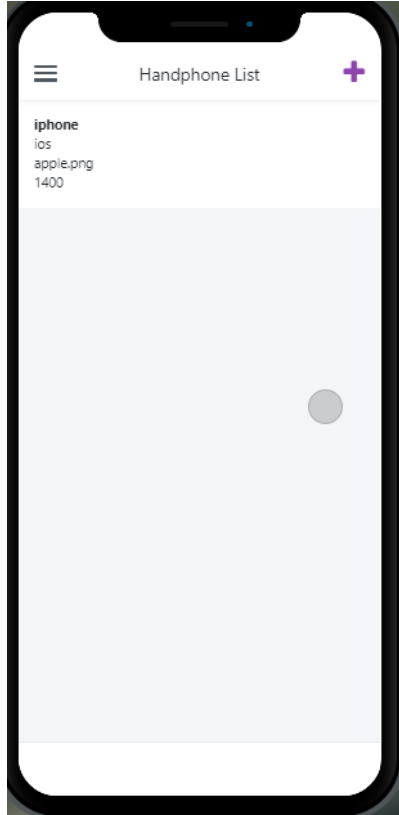
Filtering Data based on Login User

- Make use of **GetUserId()** to get the id of the current login user
- Set up the Filters in Aggregates

The screenshot shows the 'GetExpensesByUserId' interface. At the top, there's a breadcrumb trail: 'MainFlow > Expenses > GetExpensesByUserId'. Below this, there are tabs for 'Sources', 'Filters', 'Sorting', and 'Test Values'. The 'Filters' tab is active. Under the 'FILTERS' heading, there is a single filter rule: 'Expenses.UserId = GetUserId()'. Below the filter rule, there is a '+ Add Filter' button.

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

Exercise Lab 4: LoginUser Mobile App






outsystems | Users

Users Groups Applications

Users

[Create a new User](#) [Inactive Users](#) [Export to Excel](#)

3 records

Name	Username
 alan	alan
 Kok Cheng Tan	tan_kok_cheng@rp.edu.sg
 kokcheng	kokcheng

3 records

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





Configure iOS App

You will need an Apple developer account

Build type The type of build you want to generate.	<input type="text" value="Development"/>
App identifier Unique ID used to identify the App in store and devices.	<input type="text" value="com.outsystemscloud.kokcheng.DataDrivenA"/>
Certificate Certificate used in Apple's iOS Developer Program.	<div><input type="text" value="Upload file"/> SELECT <input type="text" value="Passwoi"/></div>
Provisioning profile A provisioning profile which matches the certificate.	<div><input type="text" value="Upload file"/> SELECT</div>
<div>GENERATE APP BACK</div>	

Generate Android App



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



Configure Android App

You will need to digitally sign your App

Build type

The type of build you want to generate.

Release

App identifier

Unique ID used to identify the App in store and devices.

com.outsystemscloud.kokcheng.DataDrivenA

Keystore

A binary file that contains a set of keys.

Upload file

SELECT

Keystore Pa



Alias

The alias of the private/public key pair to use.

Alias Passw

GENERATE APP

BACK

Progressive Web App (PWA)

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

Progressive Web App (PWA)

Mobile-optimized apps that can be installed from the browser

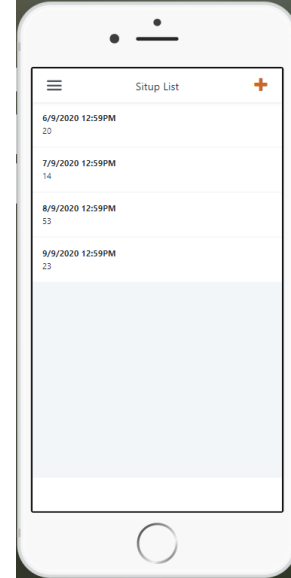
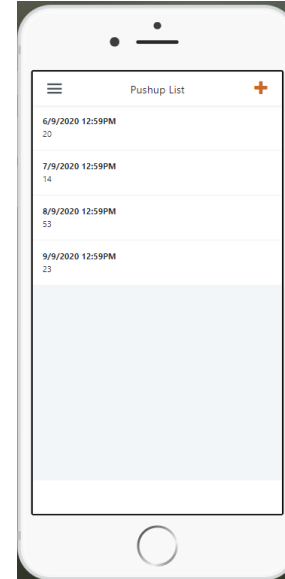
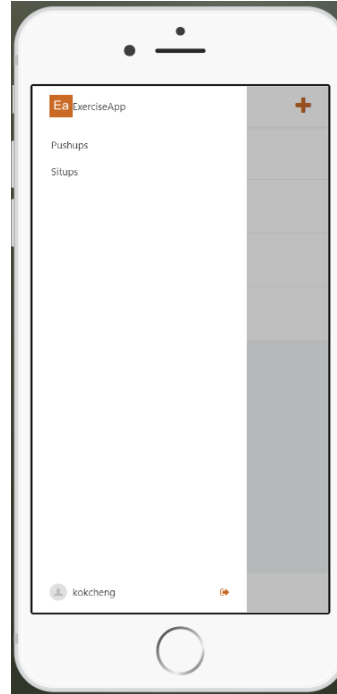
Distribute as PWA



Scan the code to access your app and add it to your device's home screen.
You can also [click here to open in the browser](#)

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