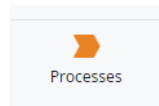# OutSystems Language Concepts

The OutSystems® Platform provides developers with a domain-specific language for developing and changing web business applications. This developer cheat sheet describes the OutSystems language main concepts and their mappings to other development languages, including .NET.
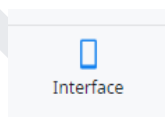
Processes Tab

Business Processes and Batch Processing

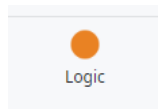| OutSystems	Description	Traditional/.NET/Javascript
Process	Integrate business processes into an application by defining activities within process flows	
Timer	Actions that execute asynchronously on a defined schedule.	Batch job
Process Actions		
Launch‹Process›	Explicitly launch an instance of a process	
Timer Actions		
Wake‹Timer›	Explicitly wake a Timer, ignoring its schedule	

Interface Tab

User Interface

OutSystems	Description	Traditional/.NET/Javascript
UI Flow	User interaction diagram of screens	N/A
UI Flow Tools		

Screen	HTML pages	
Block	Reusable group of widgets for other blocks/screens	
Email	Send an email asynchronously via a sending queue.	
External Site	Connects to a site external to your module through a well-known URL you specify	
Screen and Block Tools		
Widget	UI Controls	**UI Control Classes**
Input Parameter	Send data to a screen	Query string variable of an HTTP request, or an **HttpContext Item** in Server request
Local Variable	Variables with Screen scope	Instance variable of the web page
Fetch Data from Database	Asynchronous Action that retrieves data to the Client using a Server-side Aggregate	
Fetch Data from Other Sources	Asynchronous Action that retrieves data to the Client using a Server Action flow	
Client Action	Actions only available in Screen; compiles to Javascript on screen	
Screen and Block Widgets		
Container	A "box" to contain other widgets and containers	**Panel**
If	Control displayed content based on condition(s) with "then/else"	**if/else**
Expression	Render value computed at run-time	**Label**
(See Documentation for Others)	Table, List, List Item, Form, Label, Input, Text Area, Switch, Checkbox, Radio Group, Dropdown, Upload, Button,	

	Button Group, Link, Popover Menu, Image, Icon, Popup, Block, HTML Element, etc	

Logic Tab

Business Logic

OutSystems	Description	Traditional/.NET/Javascript
Client/Server Action	Business Logic; Server Actions compile to .NET and run on the Server; Client Actions compile to Javascript and run on the Client.	Method
Input Parameter	Send data into Action	Method parameter
Output Parameter	Return data from Action	Method **out** parameter
Local Variable	Variables with Action scope	Method Local Variable
User-Defined Function	Actions that can be invoked in Expressions	**Static** Method
Client Action Tools (Compiles to **Javascript** on the Client)		
Start	Start of Action flow	
Message	Pop-up an Alert message	
Run Client Action	Call a Client-Side Method	
Run Server Action	Call a Server-Side Method	
Aggregate (Mobile)	SELECT Query, using visual editor; for JOIN, WHERE, SORT BY, GROUP BY, etc; DB agnostic (DB type is abstracted	

	away)	
Refresh Data	Refreshes a Screen Aggregate to update the resultset for the Client *Only available on Screen Actions	
Comment	Add a comment	
If	Conditional for branching Action flow into 2 paths	**If** statement
Switch	Conditionals for branching Action flow into multiple paths	**Switch/case/break** statement
For Each	Repeats Action path execution	**Foreach** statement
Assign	Assigns 1 or more expressions to 1 or more variables	Series of **=** statements
JSON Serialize	Convert a Record or a List of Records into JSON data-interchange format text.	
JSON Deserialize	Convert a JSON string into a Structure or List	
Exception Handler	Handle exceptions by catching them and allowing you to define a parallel error handling flow	**Catch** statement with **catch** block
Raise Exception	Explicitly launch a user exception	**Throw** statement
Javascript	Add in in-line Javascript code to execute	
End	Ends the Action flow. More than one End element in the same Action flow is allowed for organization.	
Destination	Jumps to a specific screen *Only available on Screen Actions	
Download	Download facility *Only available on Screen Actions *Not on Mobile	Method that writes the request response to a file, by setting the request content type, its header, and writing the file content to its output

		stream, all processed on the Client side.
Server Action Tools (Compiles to **.NET** on the Server)		
Start	Start of Action flow	
Run Server Action	Call a Server-Side Method	
Aggregate	SELECT Query, using visual editor; for JOIN, WHERE, SORT BY, GROUP BY, etc; DB agnostic (DB type is abstracted away)	Method with SQL query returning a **dataset**
SQL	SELECT Query, using DB-specific text-based SQL	Method with SQL query returning a **dataset**
If	Conditional for branching Action flow into 2 paths	**If** statement
Switch	Conditionals for branching Action flow into multiple paths	**Switch/case/break** statement
For Each	Repeats Action path execution	**Foreach** statement
Assign	Assigns 1 or more expressions to 1 or more variables	Series of **=** statements
Record List To Excel	Exports contents of a record list to an Excel file	Method that writes a list of objects into an Excel file using a library to write Excel files
Excel To Record List	Imports content of an Excel file into a record list	Method that reads a list of objects from an Excel file using a library to read Excel files
JSON Serialize	Convert a Record or a List of Records into JSON data-interchange format text.	
JSON Deserialize	Convert a JSON string into a Structure or List	
Exception Handler	Handle exceptions by catching them and allowing you to define a parallel error handling flow	**Catch** statement with **catch** block

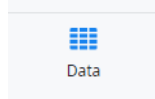|		
Raise Exception	Explicitly launch a user exception	**Throw** statement
Comment	Add a comment	
Send Email	* Does not exist in Data Action	
End	Ends the Action flow. More than one End element in the same Action flow is allowed for organization.	

Integrations

OutSystems	Description	Traditional/.NET/Javascript
SOAP	Expose/Consume SOAP API	
REST	Expose/Consume REST API	
SAP	Integrate with SAP BAPI	
Extension (.xif)	Wrappers to existing C# libraries, database tables, etc, visually exposed inside the IDE as reusable Actions, Entities, and Structures. Extensions are created/modified with Integration Studio.	**Class Library** project compiled into a native assembly DLL

Security

OutSystems	Description	Traditional/.NET/Javascript
Role	Implement security representing a group of elements that share the same grant policy	Security setting granted to users and roles
Role Actions		
Check<Role>	Checks whether a specific End User has been granted access to a specific permission area	N/A

Grant<Role>	Provides access for a specific End User to a specific permission area.	N/A
Revoke<Role>	Denies access for a specific End User to a specific permission area.	N/A

Data Tab

Database (Server-Side) and Local Storage (Client-Side)

OutSystems	Description	Traditional/.NET/Javascript
Entity	Persistent data storage representing database model	Database Table, similar to **DataSource** or an Object-Relational (OR) Mapping Class
Entity Attribute	Entity storage	Database Table Field, similar to **DataSource** field or a field in an Object-Relational (OR) Mapping Class
Entity Actions	CRUD (Create, Read, Update, Delete) Actions automatically created and managed by the OutSystems platform for each Entity	OR Mapping CRUD Methods or Methods that execute the **CREATE**, **GET**, **UPDATE**, and **DELETE** SQL statements
Static Entity (Server Only)	Entity with static data managed during design time with strong typing. Used for constants and enumerations.	Entity with static and fixed data (**enums**) defined during design-time.
Static Entity Record (Server Only)	Instance of static data that holds literal values for each entity attribute	**Enum** named constant
Entity Actions		
Create<Entity>	Creates a new record in the entity. If the record already exists, i.e. there is a row in the database with the same	OR Mapping library method that creates a given object in the database, or a Method that

	identifier, a database exception is returned and no record is created.	executes the **CREATE** SQL statement to create a new database table record
CreateOrUpdate<*Entity*>	Creates a new record in the entity. If the record already exists, i.e. there is a row in the database with the same identifier, their attributes are updated with the arguments of the action.	OR Mapping library method that creates a given object in the database if it is new, or updates an existing one if it already exists in the database, or a Method that executes a **CREATE** SQL statement in case the entity identifier is null or an **UPDATE** SQL statement in case the entity identifier is not null
Update<*Entity*>	Updates a specific record of the entity.	OR Mapping library method that updates an existing object in the database, or a Method that executes an **UPDATE** SQL statement to update the database table record
Get<*Entity*>	Gets a row from the entity with a specific identifier.	OR Mapping library method that gets an existing object from the database, or a Method that executes a **GET** SQL statement to get the database table record
GetForUpdate<*Entity*>	Gets a row from the entity with a specific identifier and locks it, preventing other processes from accessing this record.	OR Mapping library method that gets an existing object from the database and keeps it locked while your transaction is open to assure no other transaction can change it, or a Method that executes a **GET** SQL statement to get the database table record and includes a **WITH UPDLOCK** SQL statement
Delete<*Entity*>	Deletes a specific record of the entity given an identifier.	OR Mapping library method that deletes an object from the database, or a Method that

		executes a **DELETE** SQL statement to delete a database table record

Session State

OutSystems	Description	Traditional/.NET/Javascript
Client Variable	Variables that hold data that is persistent during the session, stored on the Client side. You can use these variables to save information during the end–user interaction.	

Miscellaneous

OutSystems	Description	Traditional/.NET/Javascript
Structure	Data skeleton that gathers information of the Module. Structures are not persistent, they only exist in memory until the application ends, at which time they are discarded. The information is stored in attributes.	**Struct** type
Structure Attribute	Stores part of the information that concerns the structure.	**Struct** property
Site Property	Variable instantiated before the deployment of the Module. The value can be changed in runtime after deploying the application, using Service Center	**Public static** variable accessible across application that can be changed in runtime using the back office (Service Center)

Projects

OutSystems	Description	Traditional/.NET/Javascript

Solution (.osp)	Groups of Modules and Extensions for storage and staging between Environments.	**Visual Studio Solution** (.sln)
Module (.oml)	Project with UI, Logic, Data, Processes, Security Rules, etc.	**Web Application Project**
Extension (.xif)	Extension of OutSystems Platform capabilities written in C#	**Class Library Project**