Servoy Concepts

This chapter provides an overview of the most important concepts in the Servoy patform

Concept	Basic Description
Solution	A Solution is a single application that can be run in any of the Servoy Clients. It contains forms, business logic and datalayer definitions.
Modules	A Module is solution that is contained within another Solution.
Form	A Form is an object that provides a UI and/or contains business logic. The UI of a Form is built up of Form Parts, which in turn contain Elements.
Abstract Form	An Abstract Form is a Form that does not implement any Form Parts and as such can not be shown to the user, but can contain business logic and can act as a Super Form for Child Forms
Super Form	A Form that is used as a Parent for a Child Form through inheritance
Parent Form	Alias for Super Form
Child Form	A Form that extends another Form through inheritance
Form Parts	Horizontal bands on the Form that together make up the vertical dimensions of a Form, like a header, body and footer.
Form View	The 'view' mode of the Form. There are three modes available: • TableView: the body part of the Form shows multiple Records in a grid of data, with each record occupying one row • RecordView: the body part shows 1 record and the elements can positioned any way the developer thinks suit • ListView: the body part shows multiple records and the display of each record is equal to how the record would be displayed in RecordView
Element	A UI widget that can be placed on a Form, like a Field, button, Label or TabPanel for example
Container	A Element that can contain another another Form, like a TabPanel or SplitPane
Bean	A UI widget that is not a core Element in Servoy, but a drop in extension to the built-in list of Elements
UI	UI is short for User Interface. In Servoy a UI is provided by Forms with their Form Parts which contain Elements
Media	A file object, usually images.
Media Library	The library in which Media's are stored.
Inheritance	The mechanism of creating new Forms (Child Form) that extending existing Forms (Super or Parent Form).
Business Logic	The logic written by developers in JavaScript
JavaScript	The scripting language used in Servoy to write business logic
Scripting Layer	The virtual layer in the Solution where all the execution of business logic takes place
Scripting API	The predefined set of functions by Servoy to facilitate common actions, for example the API of all the datalayer components for easy datamanipulation
Plugin	A drop-in extension to the Scripting API
Function	A single bit of executable logic in JavaScript
Method	A JavaScript function that is tied to an object is called a method of that object
Variable	A named property that can contain a value

Scope	A JavaScript execution context within the Solution. A scope contains variables and functions. When the functions get executed, they are executed within that scope. Within Servoy there are a few types of scopes: • Solution / top-level scopes: These scopes are not tied to anything but the solution itself. A default 'globals' scope is automatically available when a solution is created (but can be removed by the developer). Other top-level scopes can be created by the user. • Form scope: each form has a scope attached to it, in which the JavaScript logic that is tied to that form is executed. If the Form is extending another Form through inheritance, the JavaScript scope also inherits the variables and functions defined in the parent Form's scope. A form has access to the calculations scope through a record. • Foundset scope: the scope in which Entity Methods and Table Events are defined and execute. The methods from the foundset scope have access to the calculation scope through a record. • Calculation scope: the scope in which Calculations are defined and execute. The calculation scope is a special scope that is meant just for the record itself.
Globals	The default top-level scripting scope inside a Solution or a Module.
Datalayer	The exposed objects in the Form scopes that are tied to columns and tables in the database.
Datasource	A datasource is the identifier for an external source of tabular data, in most cases a table from a database.
Dataprovider	A named property that contains data. The property can be tied to a element in the UI or can be interacted with in scripting. A dataprovider can either be: • a variable in the scope of a Form, • a variable in the global scope or • the dataproviders of the selected record in a FoundSet
Relation	A object from the datalayer that provides the link from one datasource to another, based on one or more matches between the dataproviders in the datasource and an operator
ValueList	An object from the datalayer that is used in the UI to provide selection lists and translations of internal values to display values
FoundSet	A FoundSet is a represention of a datasource as an object in the Solution. Each record from the datasource is contained in the FoundSet as a Record, making the FoundSet a collection of Records. A FoundSet is always linked to just one datasource and can be interacted with through it's API in the Scripting Layer and can also be tied to UI through Forms. A change made to the Foundset or any of it's containing Records is, when a save occurs, automatically persisted to the external source of data. The FoundSet exposes the columns of the selected Record in the Foundset as Dataproviders. FoundSets are tied to a datasource and retrieve records automatically in batches from the datasource when they are accessed through the FoundSet's scripting API or when they needs to be displayed in the UI. The FoundSet automatically manages the retrieval, caching and removal of records, in order to prevent high memory consumption or long loading times.
Record	A Record is a representation of a record in a datasource as an object in the Solution. Records are contained in FoundSets
DataSet	A DataSet is an object with inmemory
Calculation	A dataprovider on datasource level, who's value is calculated by business logic and is exposed as a dataprovider on each Record in Foundsets based on the datasource
Unstored Calculation	A calculation that is not stored in the external source behind the datasource
Stored Calculation	A calculation that is stored in the external source behind the datasource. Calculations automatically become Stored Calculations if defined with the same name as one of the columns in the external source of data behind a datasource
Aggregation	A dataprovider that is an aggregation of a column in the datasource
Databroadcasting	The automated mechanism of Servoy to notify Servoy Clients that have data in memory to update their cache if the data was altered by another ServoyClient
TableFilters	The facility in Servoy to set a filter on a datasource, after which Servoy will automatically apply the filter to all interaction with that datasource
Designtime	Designtime refers to everything the developer creates inside Servoy Developer that becomes part of the Solution. These are the Forms, the business logic written, the datalayer definitions etc.
Runtime	Runtime refers to everything that happens while a Solution is running in a Servoy Client
SolutionModel	The scripting API that allows developers to modify the design of the Solution at runtime
ClientDesign	The mode in which a form can be set that allows user to resize and rearrange the Elements on the Form
Internationalization	The act of and tools for adding multi language and multi timezone support to a Solutions
i18n	Short for Internationalization
Smart Client	A native Servoy Client that automatically installs and updated itself on the client machine

Web Client	A browser based, fully Ajax aware Servoy Client, that executes all it's business logic on the Servoy Application Server, while rendering the UI using just HTML and CSS in a browserNG
NG Client	A browser-based application of which the UI is rendered in the browser using pure HTML, CSS and a bit of JavaScript for event handling and communication with the Servoy Application Server (HTML5, CSS3 and websockets communication is used). The business logic of the Solution that is running in the NGClient is executed on the Application Server for security reasons: the code of the business logic is not exposed in the webpage markup.
Headless Client	A Servoy Client that has no UI attached to it and runs on the Application Server and provides a Java API for interacting with all the business logic and forms contained in the Solution it runs
Batch Processor	A Headless Client that is configured on the Servoy Application Server to automatically start when the Application Server is launched