WebComponents

Servoy NGClient uses a client side MVC library called AngularJS, which provides all means for building webcomponents.

For example, Servoy NGClient uses an AngularJS based web component `textfield` from the `servoydefault` package by letting Servoy generate a AngularJS directive (for resolving the custom HTML tag) like:

template

```
<servoydefault-textfield dataProviderID="model.orderid" />
```

which Angular resolves run-time to:

result

```
<div class="xyz"><input type="text" ng-model="model.orderid"></div>
```

A directive like `textfield` is the basic building block for a webcomponent, see "Create Components" section at: http://angularjs.org (especially see the component.js tab there)

The component name must follow the angular naming restrictions imposed by angular directives. When using a directive in the html it resolves to the directive name by converting from snake-case to camelCase, for example when defining a directive with name 'myDir' one must use it in ht html as 'my-dir' see 'Matching Directives' section of https://docs.angularjs.org/guide/directive.

Inside each webcomponent Angular automatically takes care of the binding between model and view, in the example above the `ng-model` parameter on the `textfield` binds the value of the `input` element to the *orderid* in the model.

Every webcomponent consists out of at least 3 parts/files:

1. Specification (.spec file) defines all properties and types (internally called beanInfo)
2. Angular template (.html file) contains the html with directives like for example ng-model or "ng-click" handler to webcomponent logic
3. Client side logic (.js file) part does the adding of directive, and likely contains controller javascirpt to-do for example remote calls and expose api functions
4. optionally icon to be used in Servoy developer palette
5. optionally resources like css/images used by your components

> ✓ When starting building webcomponents, the default webcomponents provided by Servoy could serve as an example.