Inheritance Model

Form inheritance is used to extend a form's functions and UI with those of another form (the super form).
This will give the original form all the methods/variables/elements/properties/parts of the form that it extends.

The inheritance is not limited to 1 level, for example there is a form a, form b and form c, form b is extended with a and c with b. The result is that form c will then have all the methods/variables/elements/properties/parts of form a, b and c.

When using form inheritance, the properties and methods can still be overwritten in the sub forms. This is not the case for the datasource, the datasource can not be changed from one table to another, it can be changed from empty to a table.

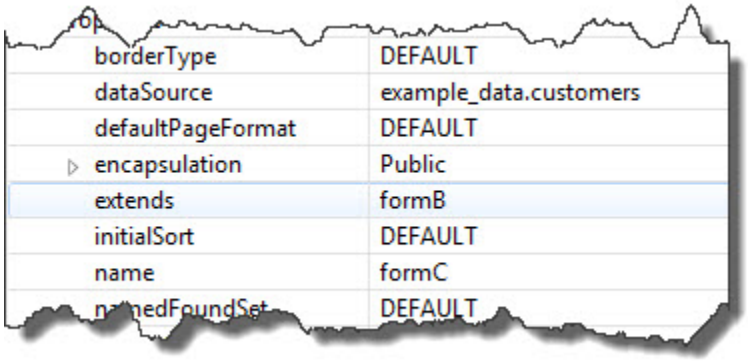
In This Chapter

Sample of Inheritance

Form 'code_base': Here are all the functions like 'new record', 'delete record' ect. On this form there are no parts, it is strictly code. This form will have no datasource.
Form 'data_controller': Here are buttons for the actions 'new record' and 'delete record' ect. There could be more than one, so there can be multiple designs. This form will inherit the 'code_base' form and uses all it's functions. This form will have no datasource.
Form 'customers': This will inherit the 'data_controller' because the functions and buttons are already inherited only the datasource and fields have to be added and there is a working form.

Setting a Super Form

On an existing form, the super form can be set by going to the properties and setting the "extends" property to the super form.



To do this in runtime, it can be done using the same property with the solutionModel.

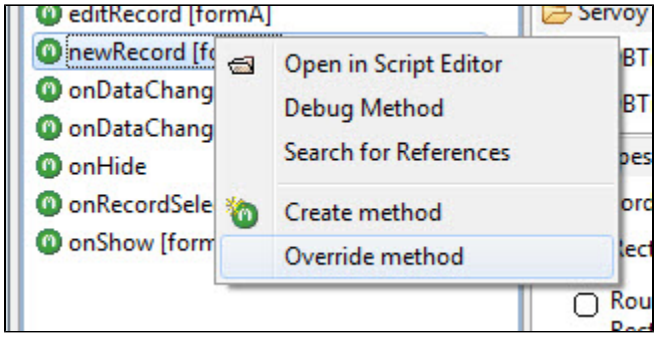
Sample method:

```
function extending() {
        var _jsForm = solutionModel.getForm('formC')
        _jsForm.extendsForm = solutionModel.getForm('formB')
}
```

All the elements on the form are accessible in the solutionModel also the elements that are inherited from a other form they can be accessed as if they are on the sub form.

Overwrite Methods

A sub form will inherit all the methods of the super form. These methods can be overwritten. When this happens, Servoy will generate a method with a call to it's super in it.




```
/** *
 * @param event
 *
 * @properties={typeid:24,uuid:"CC2EC5AB-3433-4000-A195-5E86607A630A"}
 */
function newRecord(event) {
    return _super.newRecord(event)
}
```

In the generated method there will be '_super.newRecord(event)' this will call the original method on the super form. This can be removed if the super code doesn't need to be called or code can be inserted before of after the call, if it is inserted after the super call don't forget to move the 'return'.

Sample of code before:

```
function newRecord(event) {
        vMode = 'new Record'
        return _super.newRecord(event)
}
```

Sample of code after:

```
function newRecord(event) {
        _super.newRecord(event)
        city = 'Amersfoort'
        return
}
```

Encapsulation

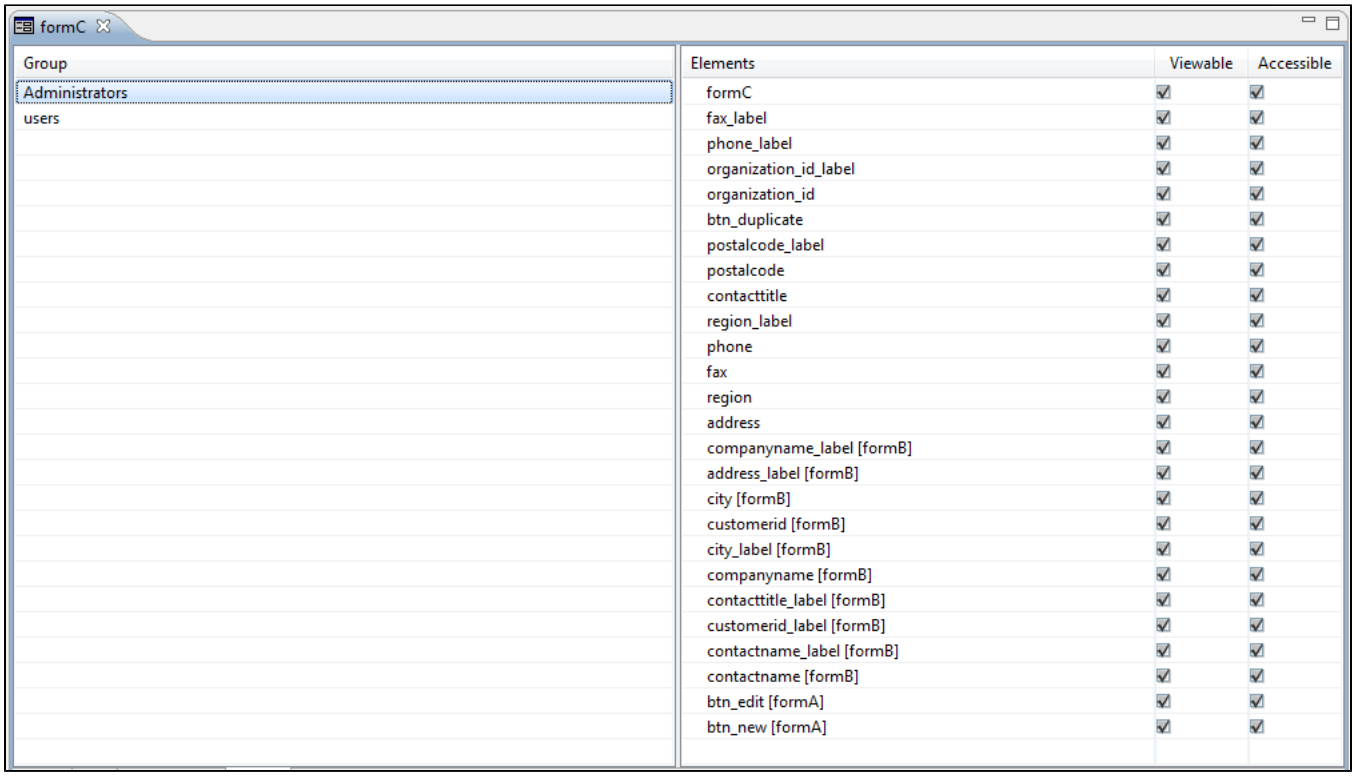
Methods that are defined to be private, will not be available for the subforms. Public methods will be available because they are available everywhere. If a method is protected it will be available for the subform but not for other forms. By creation of a method there is a choice to make it private, public or protected. If a method is already created it can be done by adding '@protected', '@private' to the docs.

Limitations

There are not many limitations to form inheritance. One of the limitations is that if there is an element on the super form, it can not be deleted on the sub form, if it should not show on the sub form the visible property of the element can be unchecked. Another limitation is that there cannot be new parts added in-between existing ones, new parts can only be added below existing parts.

Inheritance and Security

The security of a form that is extended works like it would work if it was not extended, security can be set on all the elements on the form, even if they are inherited. The security has to be set on the form that the user uses and not on one of the super forms. In the security tab it is shown from what form the inherited elements are inherited.


formC
Group
Administrators
users
Elements
Viewable
Accessible
formC
fax_label
phone_label
organization_id_label
organization_id
btn_duplicate
postalcode_label
postalcode
contacttitle
region_label
phone
fax
region
address
companyname_label [formB]
address_label [formB]
city [formB]
customerid [formB]
city_label [formB]
companyname [formB]
contacttitle_label [formB]
customerid_label [formB]
contactname_label [formB]
contactname [formB]
btn_edit [formA]
btn_new [formA]