DbSchema Forms and Reports Tutorial

Main Features

DbSchema Forms and Reports features:

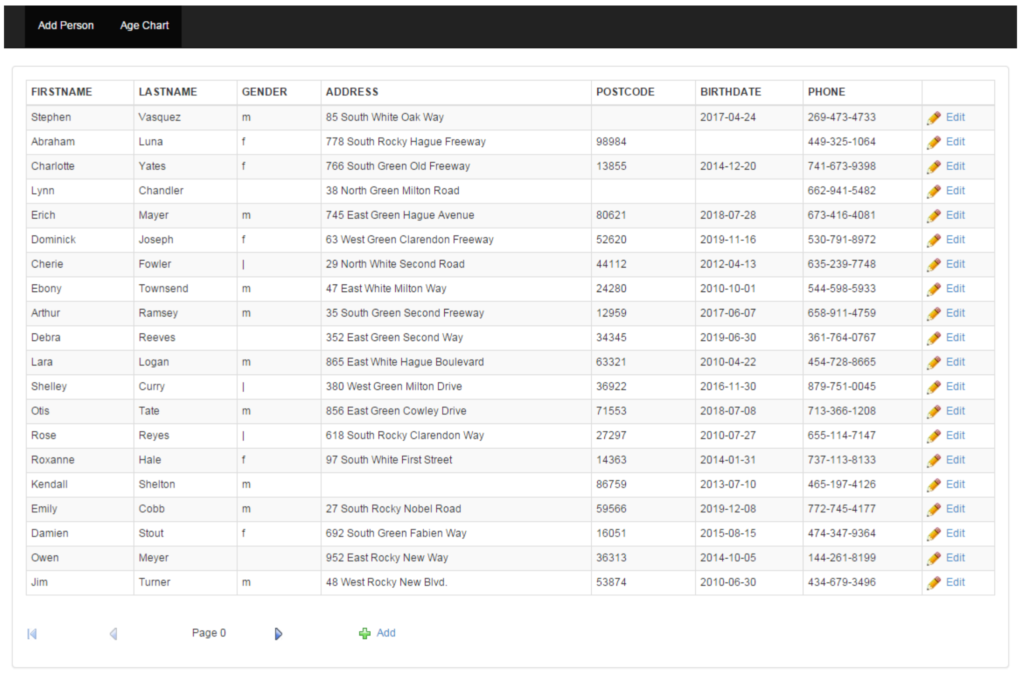
- Build for HTML, native Swing or PDF
- Coding with a modern Java –based scripting language : Groovy
- Can be deployed as stand-alone Tomcat war application
- Customizable HTML template mechanism
- Unlimited master/detail level
- Can build applications integrating input fields, buttons, charts, etc.
- Are the newest most innovative forms and reporting engine

We use the same engine for forms and reports; we make no distinction between them. In DbSchema a report is a form without input fields or buttons which will be executed as PDF or HTML.

DbSchema forms can be packed as a Tomcat war application and be deployed on a stand-alone server.

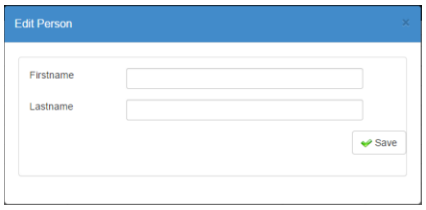
Purpose of this document

Read this tutorial to learn how to use DbSchema forms and reports. After this tutorial you will be able to build forms as below.



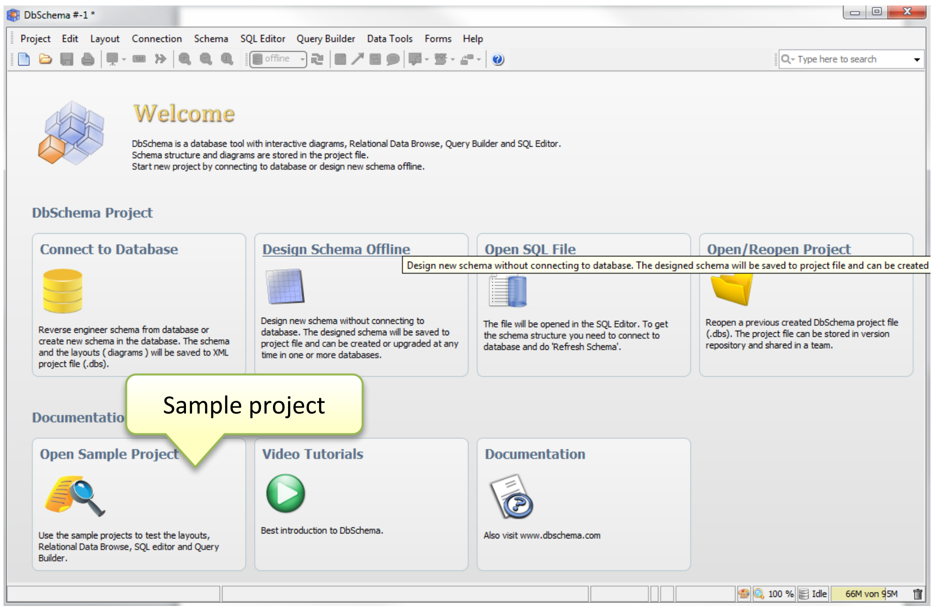
FIRSTNAME	LASTNAME	GENDER	ADDRESS	POSTCODE	BIRTHDATE	PHONE	
Stephen	Vasquez	m	85 South White Oak Way		2017-04-24	269-473-4733	Edit
Abraham	Luna	f	778 South Rocky Hague Freeway	98984		449-325-1064	Edit
Charlotte	Yates	f	766 South Green Old Freeway	13855	2014-12-20	741-673-9398	Edit
Lynn	Chandler		38 North Green Milton Road			662-941-5482	Edit
Erich	Mayer	m	745 East Green Hague Avenue	80621	2018-07-28	673-416-4081	Edit
Dominick	Joseph	f	63 West Green Clarendon Freeway	52620	2019-11-16	530-791-8972	Edit
Cherie	Fowler	|	29 North White Second Road	44112	2012-04-13	635-239-7748	Edit
Ebony	Townsend	m	47 East White Milton Way	24280	2010-10-01	544-598-5933	Edit
Arthur	Ramsey	m	35 South Green Second Freeway	12959	2017-06-07	658-911-4759	Edit
Debra	Reeves		352 East Green Second Way	34345	2019-06-30	361-764-0767	Edit
Lara	Logan	m	865 East White Hague Boulevard	63321	2010-04-22	454-728-8665	Edit
Shelley	Curry	|	380 West Green Milton Drive	36922	2016-11-30	879-751-0045	Edit
Otis	Tate	m	856 East Green Cowley Drive	71553	2018-07-08	713-366-1208	Edit
Rose	Reyes	|	618 South Rocky Clarendon Way	27297	2010-07-27	655-114-7147	Edit
Roxanne	Hale	f	97 South White First Street	14363	2014-01-31	737-113-8133	Edit
Kendall	Shelton	m		86759	2013-07-10	465-197-4126	Edit
Emily	Cobb	m	27 South Rocky Nobel Road	59566	2019-12-08	772-745-4177	Edit
Damien	Stout	f	692 South Green Fabien Way	16051	2015-08-15	474-347-9364	Edit
Owen	Meyer		952 East Rocky New Way	36313	2014-10-05	144-261-8199	Edit
Jim	Turner	m	48 West Rocky New Blvd.	53874	2010-06-30	434-679-3496	Edit

And a pop-up dialog :



Design using Wizard the First Form

To build our first form we will use the DbSchema sample project. Start DbSchema and if no project is open the welcome screen should show up.


DbSchema #-1 *
Project Edit Layout Connection Schema SQL Editor Query Builder Data Tools Forms Help
offline
Type here to search
Sample project
66M von 95M
Idle
100 %

Welcome

DbSchema is a database tool with interactive diagrams, Relational Data Browse, Query Builder and SQL Editor.
Schema structure and diagrams are stored in the project file.
Start new project by connecting to database or design new schema offline.

DbSchema Project

Connect to Database

Reverse engineer schema from database or create new schema in the database. The schema and the layouts (diagrams) will be saved to XML project file (.dbs).

Design Schema Offline

Design new schema without connecting to database. The designed schema will be saved to project file and can be created or upgraded at any time in one or more databases.

Design new schema without connecting to database. The designed schema will be saved to project file and can be created

Open SQL File

The file will be opened in the SQL Editor. To get the schema structure you need to connect to database and do 'Refresh Schema'.

Open/Reopen Project

Reopen a previous created DbSchema project file (.dbs). The project file can be stored in version repository and shared in a team.

Documentatio

Open Sample Project

Use the sample projects to test the layouts, Relational Data Browse, SQL editor and Query Builder.

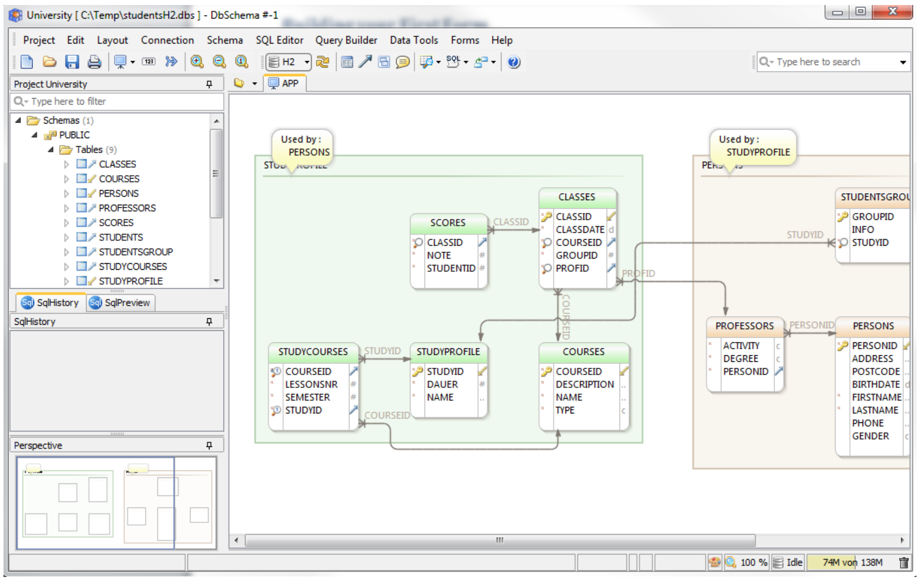
Video Tutorials

Best introduction to DbSchema.

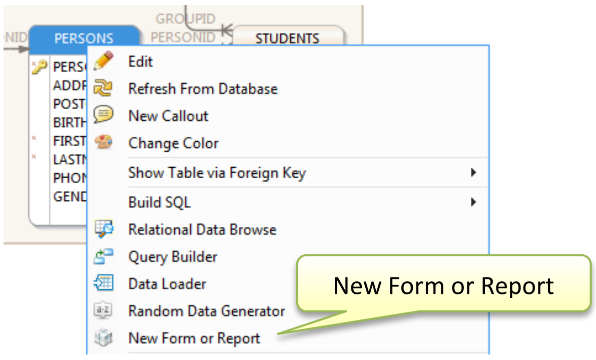
Documentation

Also visit www.dbschema.com

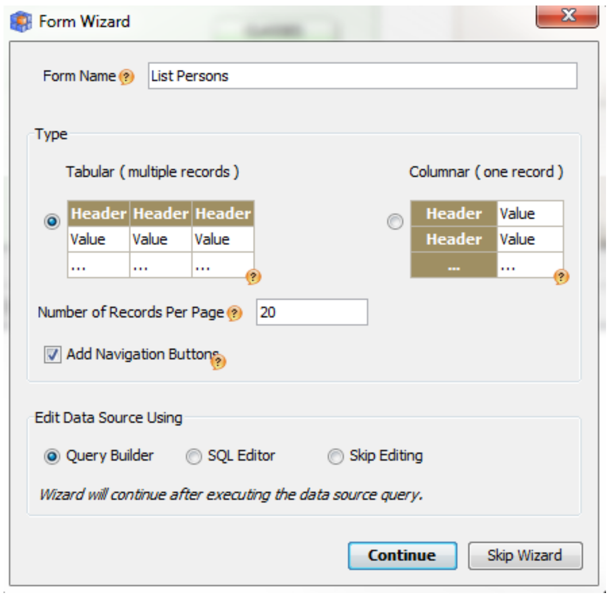
Open the forms sample project.



Right-click the 'persons' table and choose 'New Form or Report'. The Forms Wizard will start.



In the wizard dialog enter *'List Persons'* as form name and keep the default settings. Press the *'Continue'* button.


Form Wizard

Form Name: List Persons

Type

[✓ Tabular (multiple records)] Columnar (one record)

Number of Records Per Page: 20

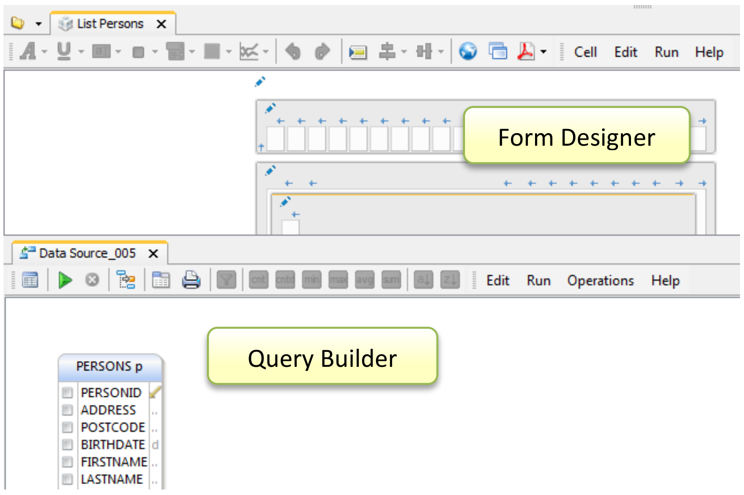
[✓ Add Navigation Buttons]

Edit Data Source Using

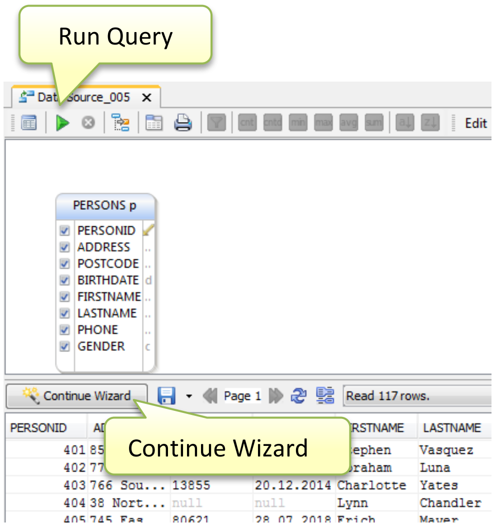
[✓ Query Builder] SQL Editor Skip Editing

Wizard will continue after executing the data source query.

The form designer will open with a Visual Query Builder on the table *'Persons'*.

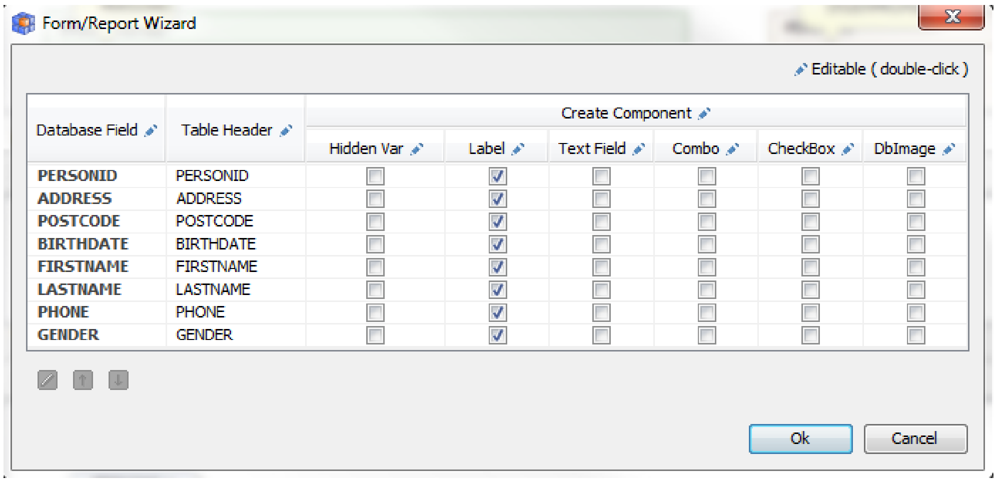


Select all checkboxes (right-click the table header and choose to select all) and run the query. In the result pane press ‘Continue Wizard’.



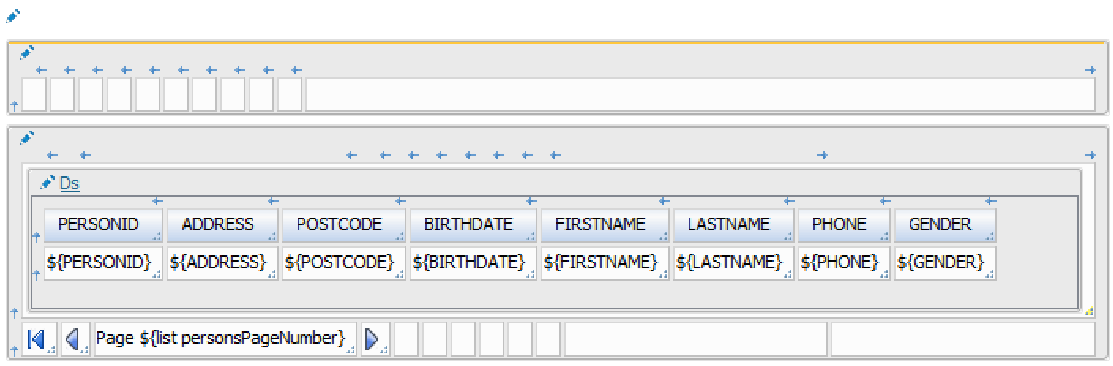
PERSONID	AD			RSTNAME	LASTNAME
401	85			ephen	Vasquez
402	77			raham	Luna
403	766 Sou...	13855	20.12.2014	Charlotte	Yates
404	38 Nort...	null	null	Lynn	Chandler
405	745 Eas	80621	28.07.2018	Erich	Mayer

In the next dialog you can choose the components to attach for each database field. Use labels for all database fields and press ‘Ok’.



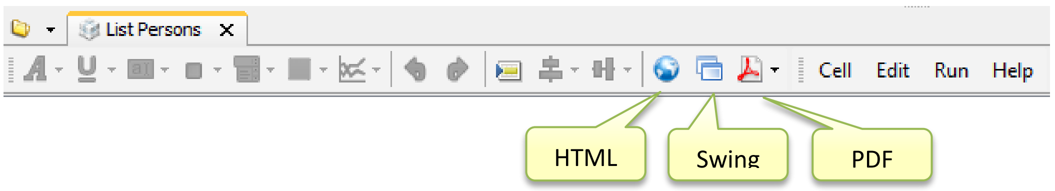
Database Field	Table Header	Create Component					
		Hidden Var	Label	Text Field	Combo	CheckBox	DbImage
PERSONID	PERSONID	☐	✓	☐	☐	☐	☐
ADDRESS	ADDRESS	☐	✓	☐	☐	☐	☐
POSTCODE	POSTCODE	☐	✓	☐	☐	☐	☐
BIRTHDATE	BIRTHDATE	☐	✓	☐	☐	☐	☐
FIRSTNAME	FIRSTNAME	☐	✓	☐	☐	☐	☐
LASTNAME	LASTNAME	☐	✓	☐	☐	☐	☐
PHONE	PHONE	☐	✓	☐	☐	☐	☐
GENDER	GENDER	☐	✓	☐	☐	☐	☐

Our first form is ready. The form will look like bellow.

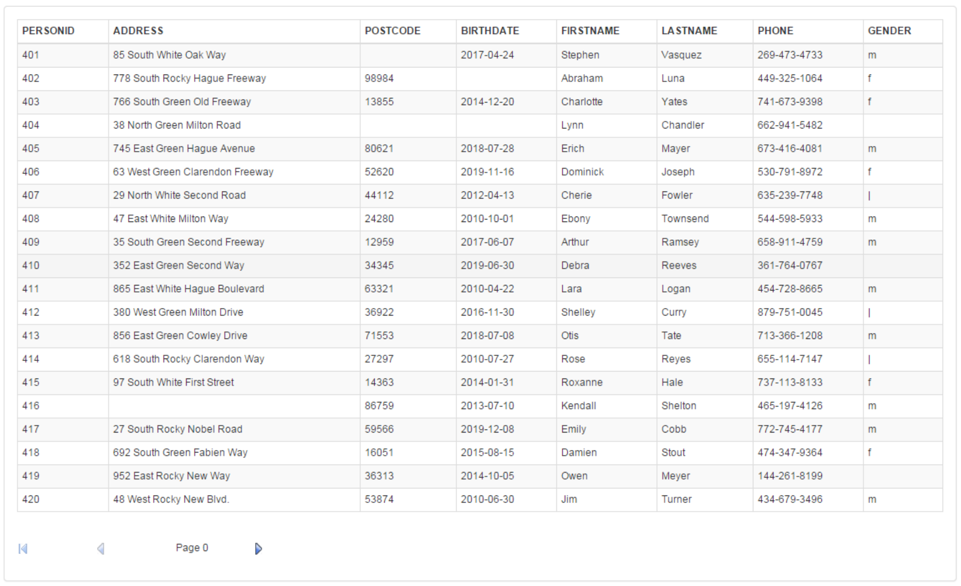


PERSONID	ADDRESS	POSTCODE	BIRTHDATE	FIRSTNAME	LASTNAME	PHONE	GENDER
${PERSONID}	${ADDRESS}	${POSTCODE}	${BIRTHDATE}	${FIRSTNAME}	${LASTNAME}	${PHONE}	${GENDER}

From the forms menu we can execute the form as HTML, Swing or PDF :



Executing as HTML you should get the following output.

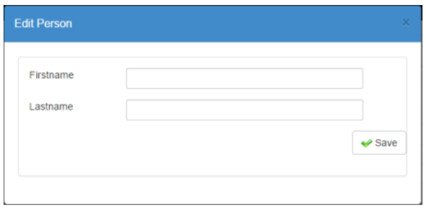


PERSONID	ADDRESS	POSTCODE	BIRTHDATE	FIRSTNAME	LASTNAME	PHONE	GENDER
401	85 South White Oak Way		2017-04-24	Stephen	Vasquez	269-473-4733	m
402	778 South Rocky Hague Freeway	98984		Abraham	Luna	449-325-1064	f
403	766 South Green Old Freeway	13855	2014-12-20	Charlotte	Yates	741-673-9398	f
404	38 North Green Milton Road			Lynn	Chandler	662-941-5482	
405	745 East Green Hague Avenue	80621	2018-07-28	Erich	Mayer	673-416-4081	m
406	63 West Green Clarendon Freeway	52620	2019-11-16	Dominick	Joseph	530-791-8972	f
407	29 North White Second Road	44112	2012-04-13	Cherie	Fowler	635-239-7748	l
408	47 East White Milton Way	24280	2010-10-01	Ebony	Townsend	544-598-5933	m
409	35 South Green Second Freeway	12959	2017-06-07	Arthur	Ramsey	658-911-4759	m
410	352 East Green Second Way	34345	2019-06-30	Debra	Reeves	361-764-0767	
411	865 East White Hague Boulevard	63321	2010-04-22	Lara	Logan	454-728-8665	m
412	380 West Green Milton Drive	36922	2016-11-30	Shelley	Curry	879-751-0045	l
413	856 East Green Cowley Drive	71553	2018-07-08	Otis	Tate	713-366-1208	m
414	618 South Rocky Clarendon Way	27297	2010-07-27	Rose	Reyes	655-114-7147	l
415	97 South White First Street	14363	2014-01-31	Roxanne	Hale	737-113-8133	f
416		86759	2013-07-10	Kendall	Shelton	465-197-4126	m
417	27 South Rocky Nobel Road	59566	2019-12-08	Emily	Cobb	772-745-4177	m
418	692 South Green Fabien Way	16051	2015-08-15	Damien	Stout	474-347-9364	f
419	952 East Rocky New Way	36313	2014-10-05	Owen	Meyer	144-261-8199	
420	48 West Rocky New Blvd.	53874	2010-06-30	Jim	Turner	434-679-3496	m

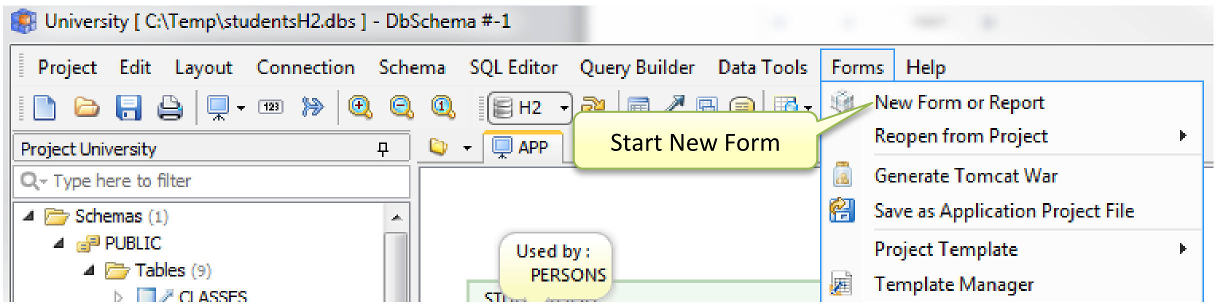
Read the next chapter to understand the form designer and how to use it.

Manual Design Second Form

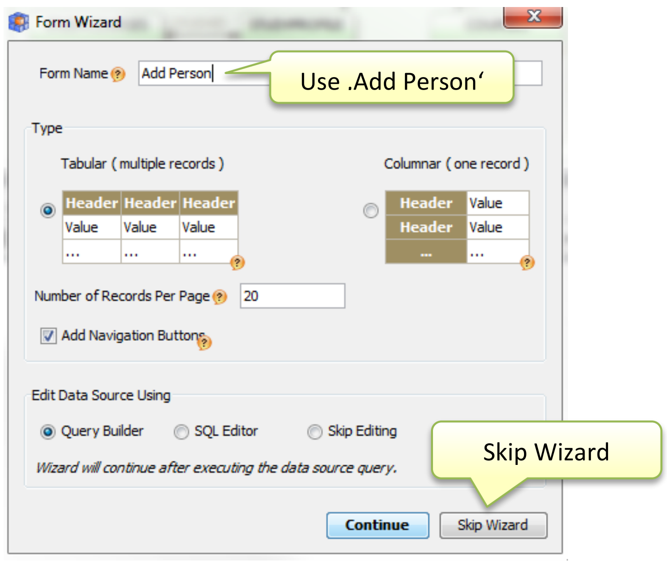
In this chapter we will build together a new form called 'Add Person'. This form will be used for adding new persons into the *'persons'* database table. The form will look like bellow.



From DbSchema main menu choose 'New Form or Report'.



The wizard will show up. Enter form name 'Add Person' and press Skip Wizard button.



Form Wizard

Form Name: Add Person

Type

Tabular (multiple records) [✓]

Header	Header	Header
Value	Value	Value
...	...	...

Columnar (one record)

Header	Value
Header	Value
...	...

Number of Records Per Page: 20

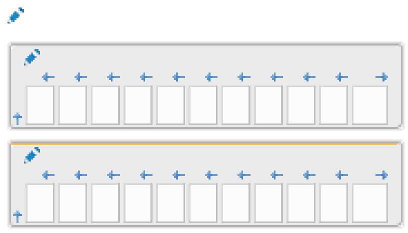
[✓] Add Navigation Buttons

Edit Data Source Using

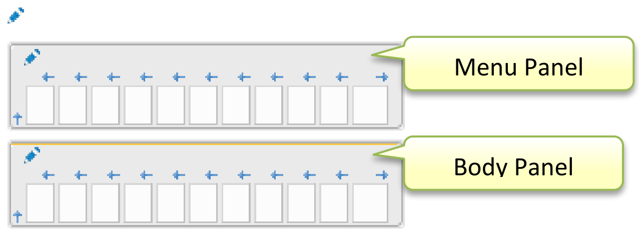
[✓ Query Builder] SQL Editor Skip Editing

Wizard will continue after executing the data source query.

An empty form will show-up.

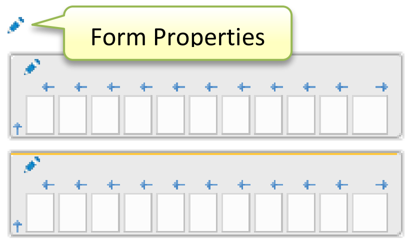


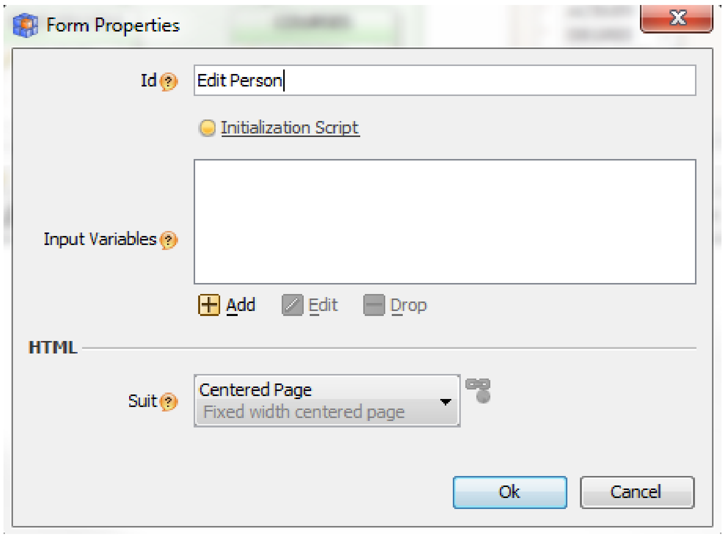
The form consists of two panels: one for menus and one for body.



Form Properties

The first pencil button is for the form properties. Click it to open the Form Properties Dialog.





Form Properties

Id	Edit Person
	Initialization Script
Input Variables	
Suit	Centered Page — Fixed width centered page

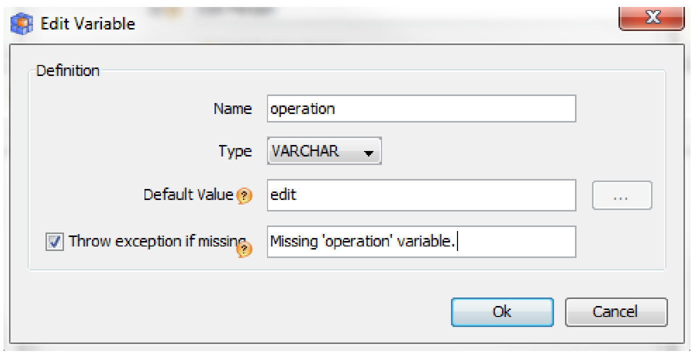
The **Initialization Script** can is a groovy script executed when this form is displayed. The script can contain logic for validating input data received, to implement authentication, etc.

Forms works on a similar basis with HTML applications. When a user modifies some data in a page, the effective database modification will be done in the next page. Forms work similar: data modifications are done in initialization script of the next form. Pressing a button may call the same form (where the button belongs to), but the form will be rendered again and its initialization script will be called.

The **Input Variables** are used to validate or define input data. This because each form (A) may send some variables to the next form (B), in a HTML format :

operation=add&firstname=John&lastname=Turner

In form (B) we define a variable *'operation'* with the default value *'edit'*. If the variable is missing, the form will throw an exception with the given text.



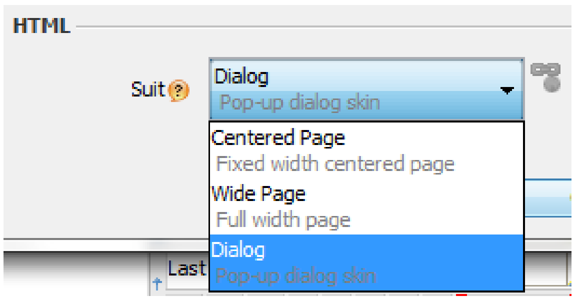
Edit Variable

Definition

Name	operation
Type	VARCHAR
Default Value	edit
☑ Throw exception if missing	Missing 'operation' variable.

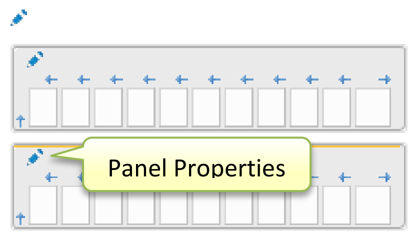
The **Suit** is the HTML representation of the form page or of a component. *'Centered Page'* is used for a web page with left and right margins, *'Wide Page'* for no margins and *'Dialog'* if the form will show inside a dialog.

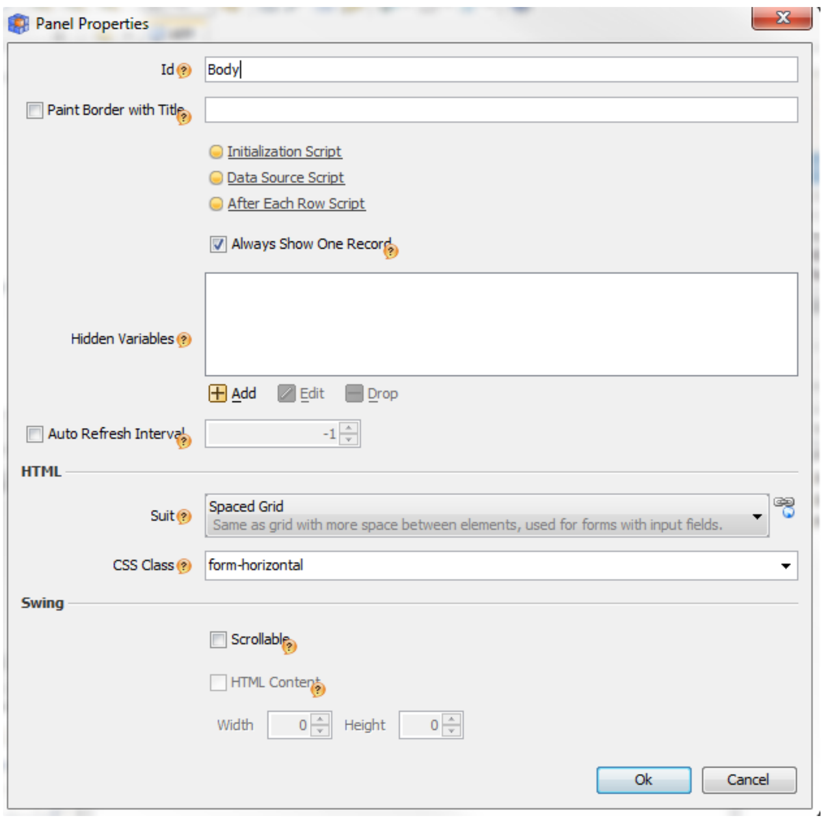
In our case we choose the suit *'Dialog'*, since we plan to show this form inside a dialog on top of the form *'List Persons'*.



Panel Properties

Similar with form properties, we can edit the panel properties. Click it to open the Form Properties Dialog.





Panel Properties

Id	Body
☐ Paint Border with Title	
	Initialization Script
	Data Source Script
	After Each Row Script
	☑ Always Show One Record
Hidden Variables	
	Add Edit Drop
☐ Auto Refresh Interval	-1

HTML

Suit	Spaced Grid — Same as grid with more space between elements, used for forms with input fields.
CSS Class	form-horizontal

Swing

	☐ Scrollable
	☐ HTML Content
	Width 0 Height 0

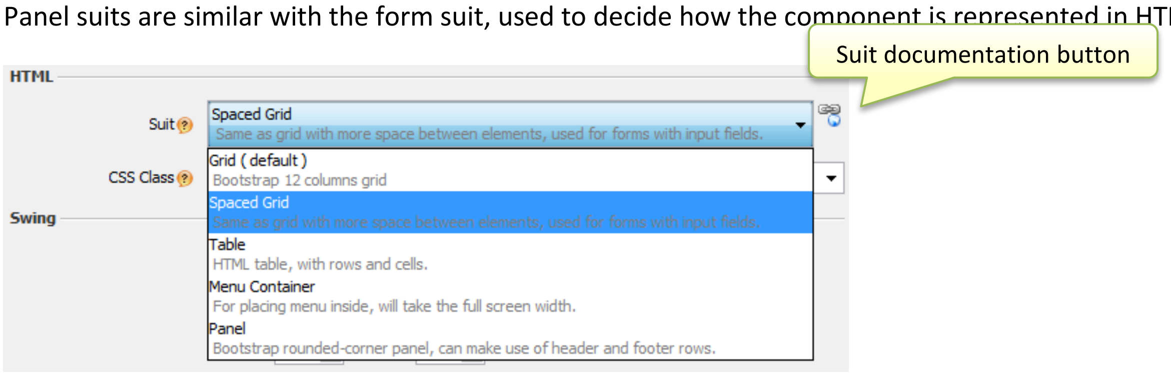
Each panel may use three **scripts**: *initialization, data source* and *after each row*. The initialization script is executed before the panel is rendered. The data source script provides data from the database to be shown inside the panel. The *after each row* script is used to compute statistics over the data coming from the data source.

The **always show one record** flag can make a distinction for the columnar forms. If the data source returns zero records, the panel will behave as it would have got an empty record and display the components inside one time. The same if the data source has three records, the panel will display only the first record.

The **hidden variables** are used to remember data without displaying it on screen. As example let's use an *edit person* form. The form receives a variable '*personid*' from the previous form and has to remember it without showing it on the screen. In this case we create a hidden variable *'personid'* which will be sent together with the other data to the next form. Components which are storing itself the variable values are input fields, combo boxes, radio buttons and checkboxes.

Panel Suit

Panel suits are similar with the form suit, used to decide how the component is represented in HTML.



Spaced Grid
Same as grid with more space between elements, used for forms with input fields.

Grid (default)
Bootstrap 12 columns grid

Spaced Grid
Same as grid with more space between elements, used for forms with input fields.

Table
HTML table, with rows and cells.

Menu Container
For placing menu inside, will take the full screen width.

Panel
Bootstrap rounded-corner panel, can make use of header and footer rows.

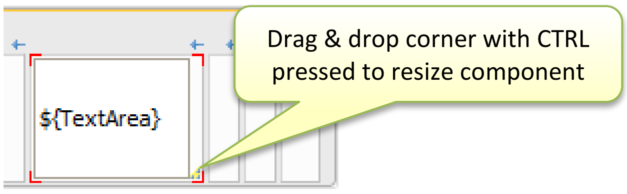
The Grid is using a 12-column Bootstrap grid documented on http://getbootstrap.com/ . You can press the small button near suit combo to get directly to the documentation. The predefined panel suits are:

- Grid – is a 12 equal-size bootstrap columns grid. In the designer the columns are not equal sized because we use a different representation for swing. Read the cell sizing chapter for details.
- Spaced Grid – same as grid, but with more space between cells. Use this for forms showing text fields, radio buttons, etc., where more space is required between components
- Table – the content will be represented as an HTML table
- Panel – is a bootstrap panel with rounded corner

Swing Settings

The Swing settings are used when the form is executed as Swing application.

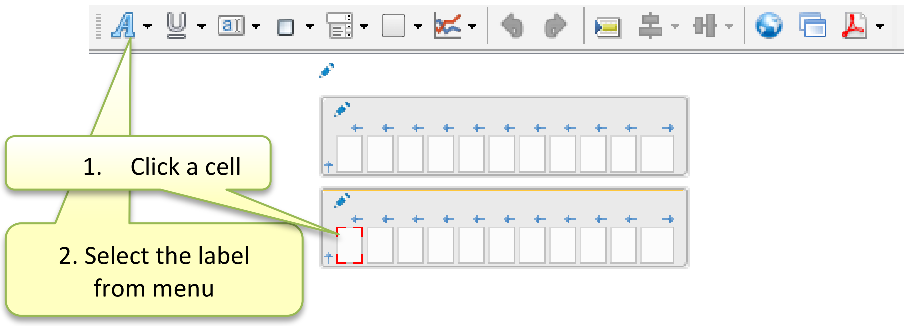
The Swing settings include the scrollable property, which will wrap the component inside a scroll pane. This will allow using scroll bars for the actual panel. If you select this option the panel can have a custom size. Only some components can be resized. You can resize them also using the mouse, by drag & drop of the grid corner while CTRL key is pressed.

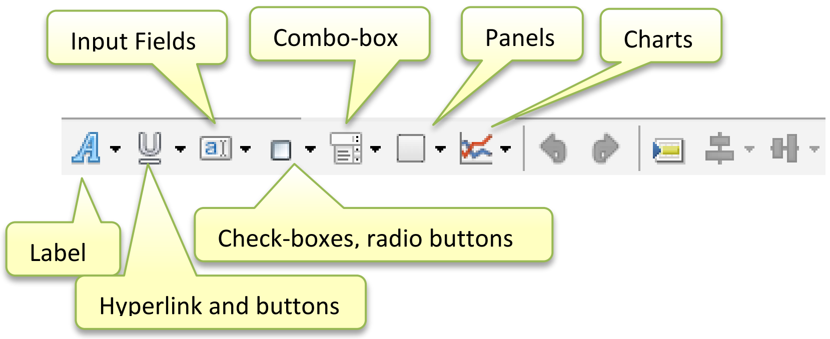


The HTML content determines the rendering of the panel as HTML, displayed inside a JEditorPane.

Add Components to Panel

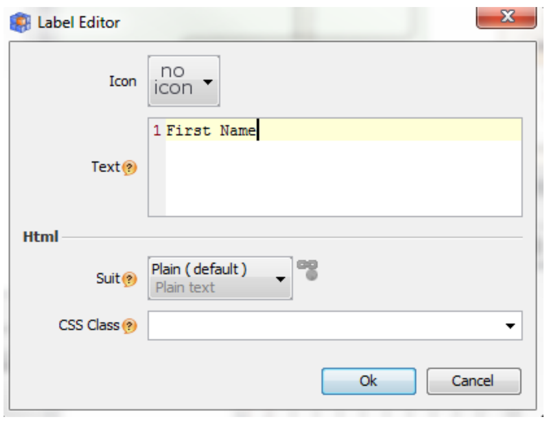
In this chapter we started a new empty form. Let's add some components to it. First click an empty cell and choose a label from the menu. Right-clicking the cell you can choose the component from pop-up.





Remember that panels can be created inside panel as each any other component. An unlimited chain of panels inside each other can be created.

When you choose the label component the label dialog will show up. Enter '*First Name*' as label text.


Label Editor

Icon: no icon

Text: First Name

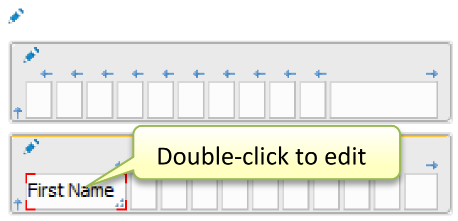
Html

Suit: Plain (default) Plain text

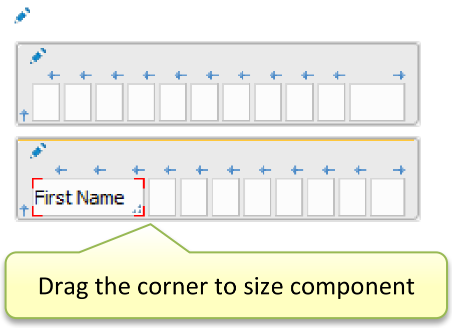
CSS Class:

Ok Cancel

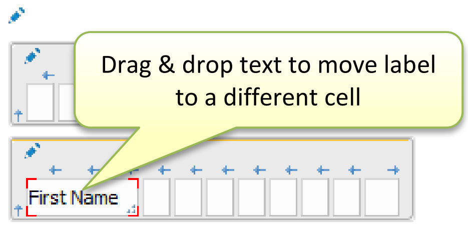
After the label is created, re-open the label editor by double-clicking the label.



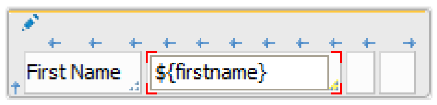
One component can cover more than one cell. Drag and drop cell from the right-side bottom grip to size it.



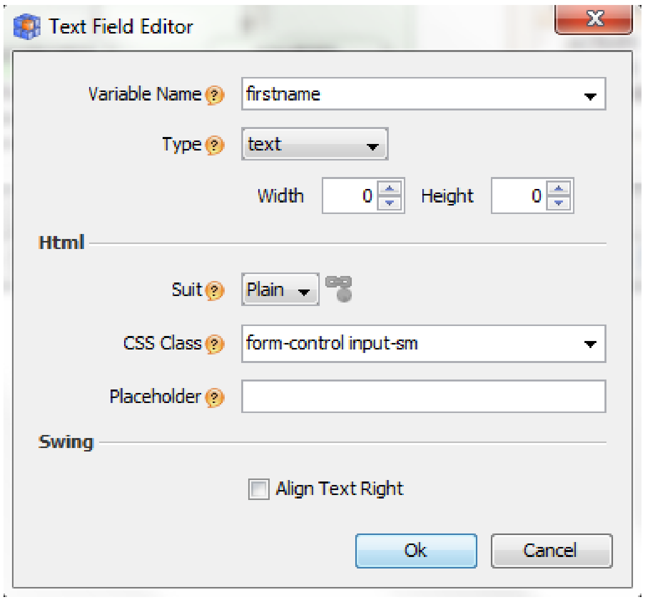
Similar you can move the label to a different cell. Drag and drop the text inside.



Using the same procedure create a text field beside the label, with *‘firstname’* as variable name.



The text field will show the text defined by the variable with this name and will also send to the next form the text in a variable with this name (*'firstname'*).



Text Field Editor

Variable Name: firstname

Type: text

Width: 0 Height: 0

Html

Suit: Plain

CSS Class: form-control input-sm

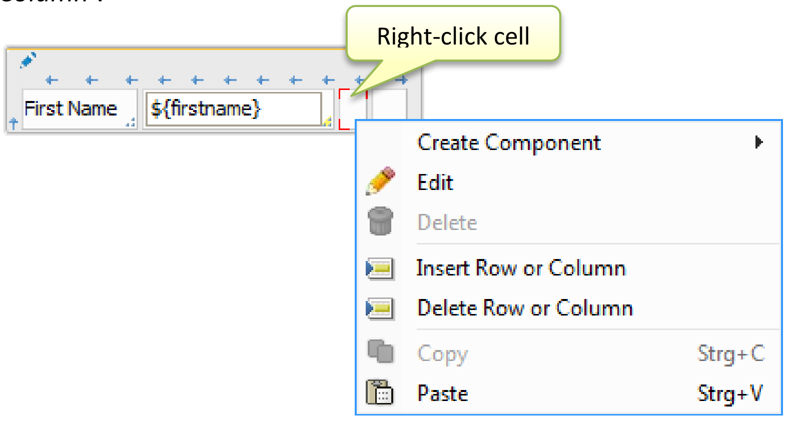
Placeholder:

Swing

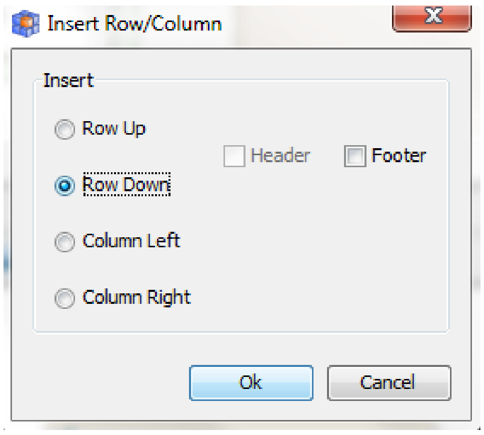
☐ Align Text Right

Ok Cancel

Next step add one more row to the panel. Right-click and empty cell and choose '*Insert Row or Column'*.



Use *'Row Down'* to insert the row under the current selected cell.



Insert Row/Column

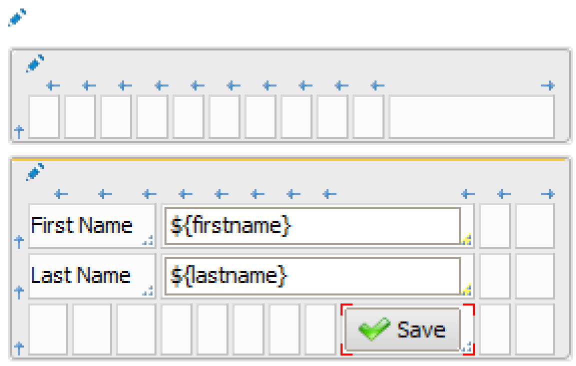
Insert

○ Row Up
● Row Down
○ Column Left
○ Column Right

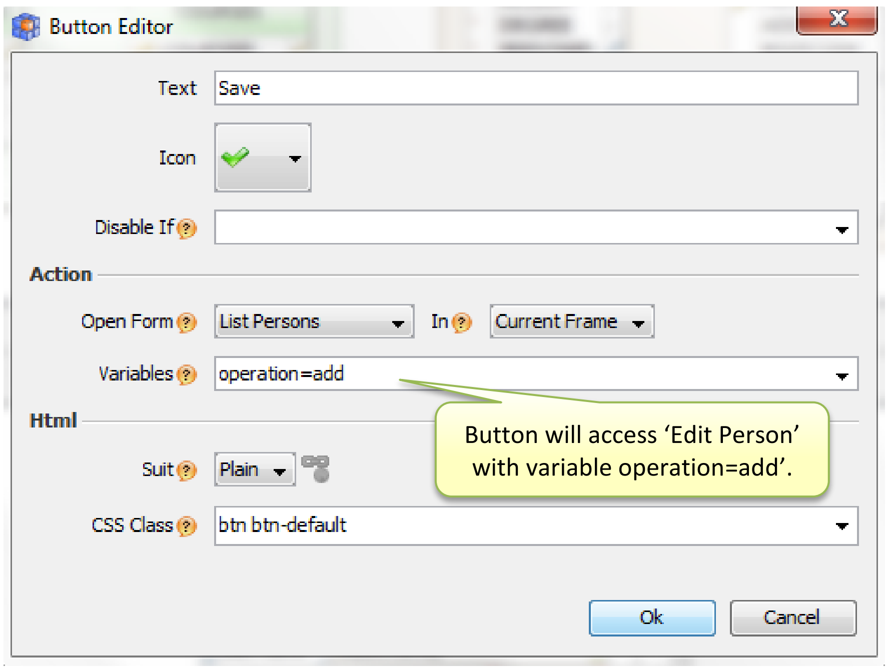
☐ Header ☐ Footer

Ok Cancel

Similar add label and text field for *'lastname'* and then add a button.



For button choose the text 'Save'. Button should open the 'Edit Person' form in current frame.


Button Editor

Text	Save
Icon	✓
Disable If	

Action

Open Form	List Persons
In	Current Frame
Variables	operation=add

Html

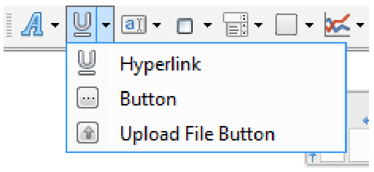
Suit	Plain
CSS Class	btn btn-default

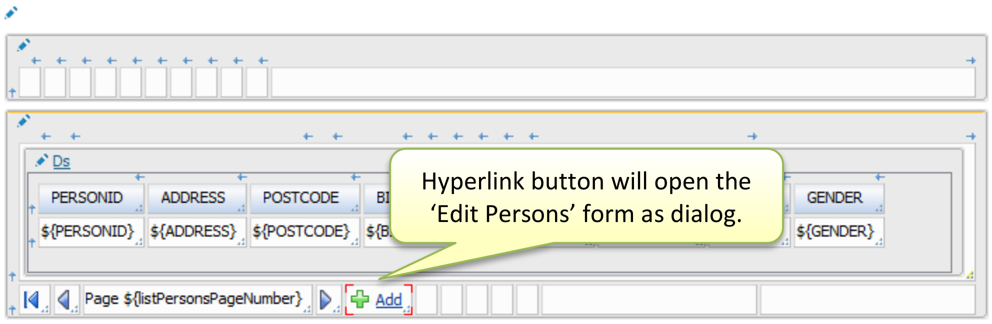
Button will access 'Edit Person' with variable operation=add'.
Ok
Cancel

The variable operation with value 'add' will be sent to the 'List Persons' form when pressing this button. In 'List Persons' we will create an initialization script. The script will check for this variable, and if it has the value 'add' will insert a new person in the database.

We see here a field '**disable if**'. The text inside will be evaluated as groovy variable, and if the value is *true* the text field will be disabled. Sample: *${page < 0 }*. It can contain plain text as well, like true or false.

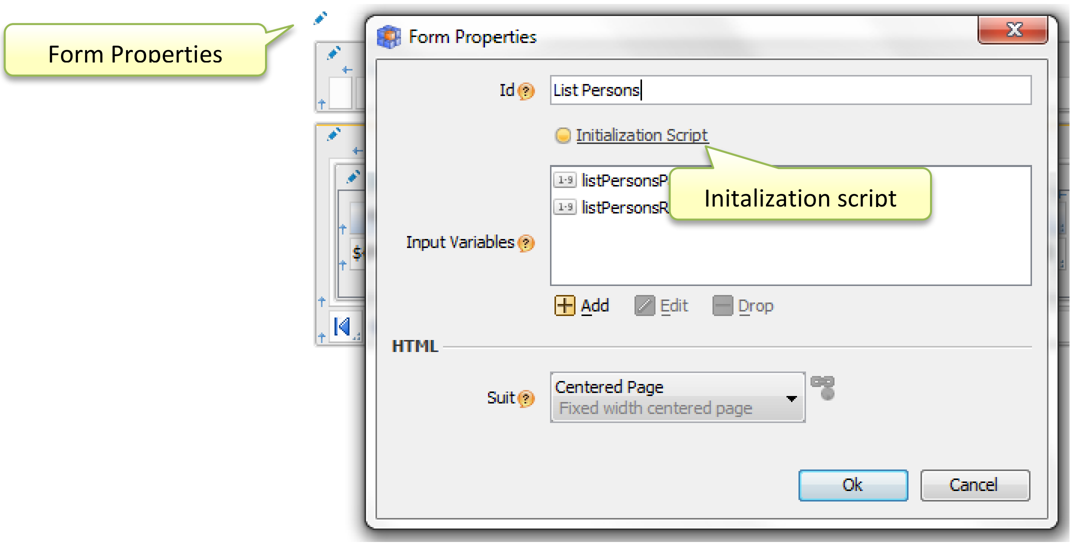
This form will be accessed from *'List Persons'*, therefore let's create a hyperlink button in *'List Persons'*.




Ds
PERSONID
ADDRESS
POSTCODE
GENDER
${PERSONID}
${ADDRESS}
${POSTCODE}
${GENDER}
Page ${listPersonsPageNumber}
Add
Hyperlink button will open the 'Edit Persons' form as dialog.

The Initialization Script

Now is the moment to add the logic in the 'List Persons' which will add the new created person into the database. Click the form properties dialog and then the Initialization script.



Form Properties

Id: List Persons

Input Variables

Suit: Centered Page — Fixed width centered page

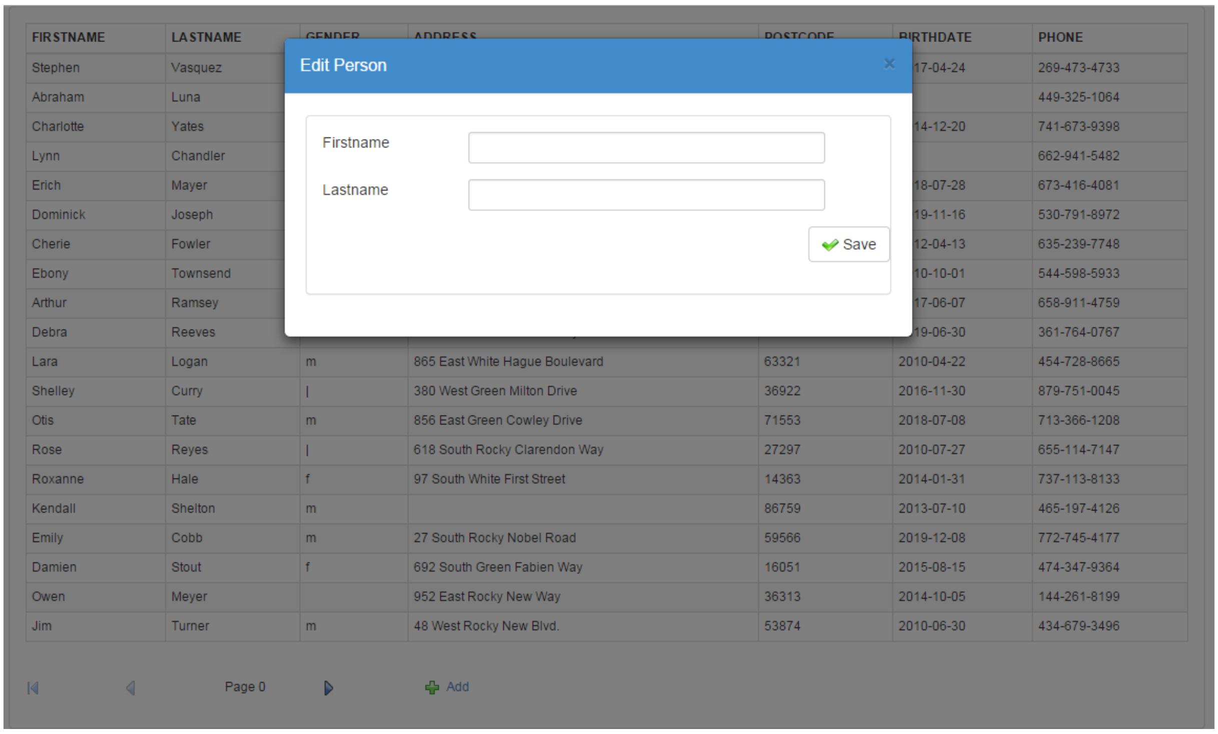
The script editor will open. Enter the following text in the editor.

```
if ( 'add'.equals( operation )){
 sql.execute( "insert into person( firstname, lastname ) values
${firstname}, ${lastname} ) " )
 sql.commit()
}
```

The script is checking if the variable *'operation'*, and if its value is 'add' will insert a new person into the database *person* table.

The script uses the Groovy language. Check in Google for Groovy tutorials.

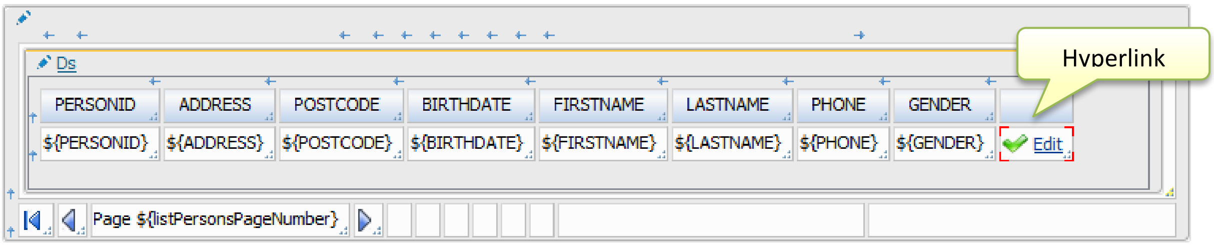
Now execute the *'List Persons'* form. Clicking the *'Add'* button you should get something like bellow.



FIRSTNAME	LASTNAME	GENDER	ADDRESS	POSTCODE	BIRTHDATE	PHONE
Stephen	Vasquez				17-04-24	269-473-4733
Abraham	Luna					449-325-1064
Charlotte	Yates				14-12-20	741-673-9398
Lynn	Chandler					662-941-5482
Erich	Mayer				18-07-28	673-416-4081
Dominick	Joseph				19-11-16	530-791-8972
Cherie	Fowler				12-04-13	635-239-7748
Ebony	Townsend				10-10-01	544-598-5933
Arthur	Ramsey				17-06-07	658-911-4759
Debra	Reeves				19-06-30	361-764-0767
Lara	Logan	m	865 East White Hague Boulevard	63321	2010-04-22	454-728-8665
Shelley	Curry	l	380 West Green Milton Drive	36922	2016-11-30	879-751-0045
Otis	Tate	m	856 East Green Cowley Drive	71553	2018-07-08	713-366-1208
Rose	Reyes	l	618 South Rocky Clarendon Way	27297	2010-07-27	655-114-7147
Roxanne	Hale	f	97 South White First Street	14363	2014-01-31	737-113-8133
Kendall	Shelton	m		86759	2013-07-10	465-197-4126
Emily	Cobb	m	27 South Rocky Nobel Road	59566	2019-12-08	772-745-4177
Damien	Stout	f	692 South Green Fabien Way	16051	2015-08-15	474-347-9364
Owen	Meyer		952 East Rocky New Way	36313	2014-10-05	144-261-8199
Jim	Turner	m	48 West Rocky New Blvd.	53874	2010-06-30	434-679-3496

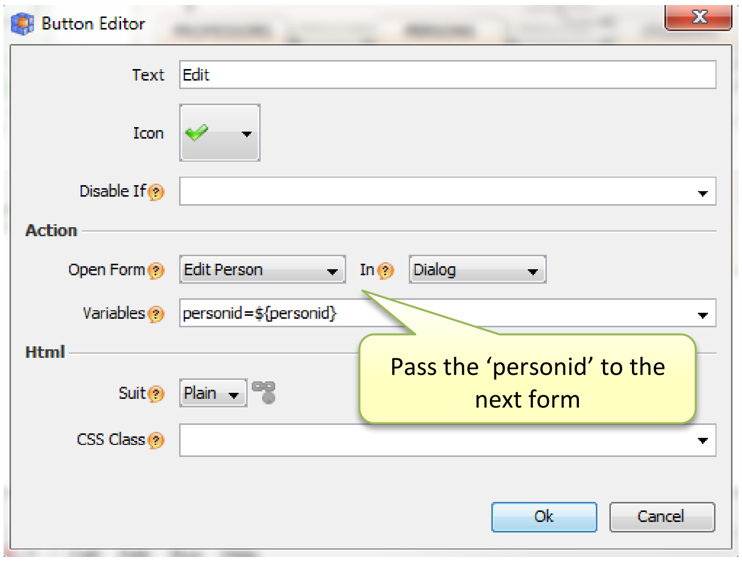
Add Hyperlinks for each record

Now our application works. We can list the persons and we can add a new person to the database. But what if we want to edit a person? We may create a hyperlink for each record, like this:



PERSONID	ADDRESS	POSTCODE	BIRTHDATE	FIRSTNAME	LASTNAME	PHONE	GENDER	
${PERSONID}	${ADDRESS}	${POSTCODE}	${BIRTHDATE}	${FIRSTNAME}	${LASTNAME}	${PHONE}	${GENDER}	✓ Edit

The hyperlink should access another form 'Edit Person' by sending the variable personid like here:



Text: Edit
Icon: ✓
Disable If:

Action

Open Form: Edit Person   In: Dialog
Variables: personid=${personid}

Html

Suit: Plain
CSS Class:

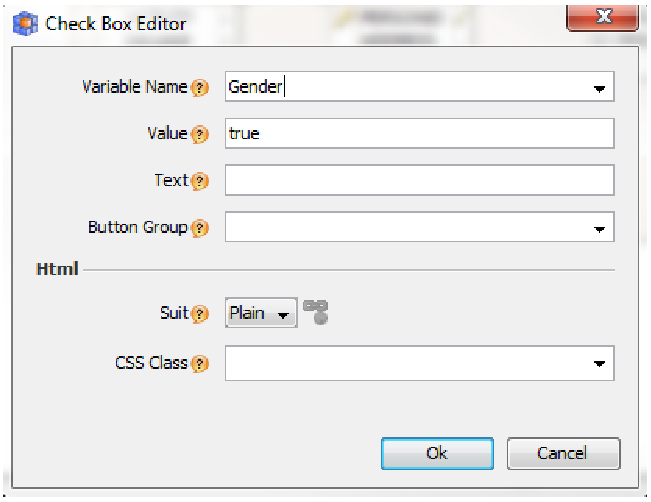
The forms can be designed so the same form can be used for adding new persons or for editing an existing person. That form may have a data source script based on personid, and if the person is new the personid can be -1.

Checkboxes and radio buttons

Checkboxes and radio buttons require a variable name. If the checkbox is selected, this variable will be true in the next form. As for text fields, checkboxes value is automatically being sent to the next form.

The value field can be a groovy expression or some text. This sets the initial value of the checkbox. The text field is similar.

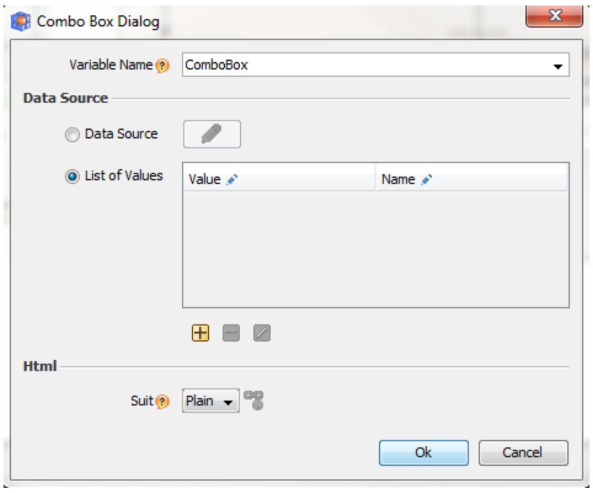
The button group can be any name. In a group of radio buttons only one radio will be selected.



Variable Name	Gender
Value	true
Text	
Button Group	
Suit	Plain
CSS Class	

Combo Boxes

Combo boxes shows the items based on a list of values or a data source script. The SQL data source script should return two columns (value and name). If a single column is returned, both value and name will have the same value.



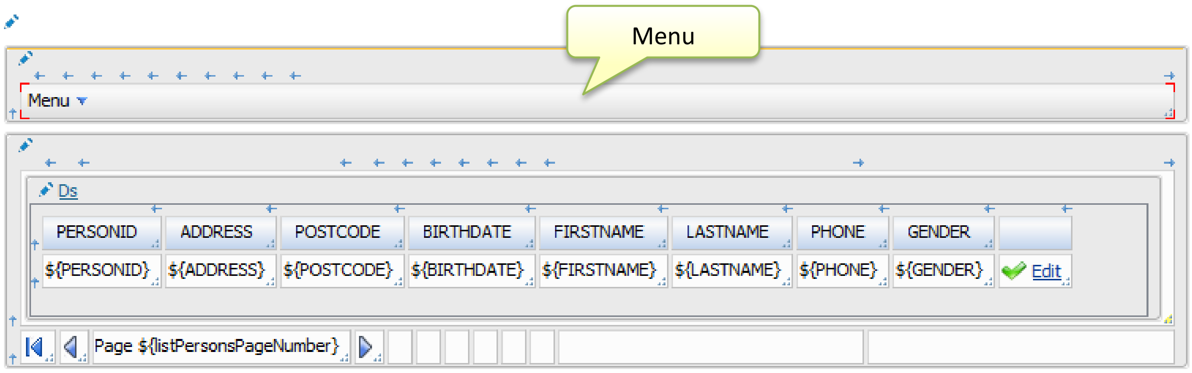
Variable Name: ComboBox

Data Source: ( ) Data Source [✓ List of Values]

Suit: Plain

Menus

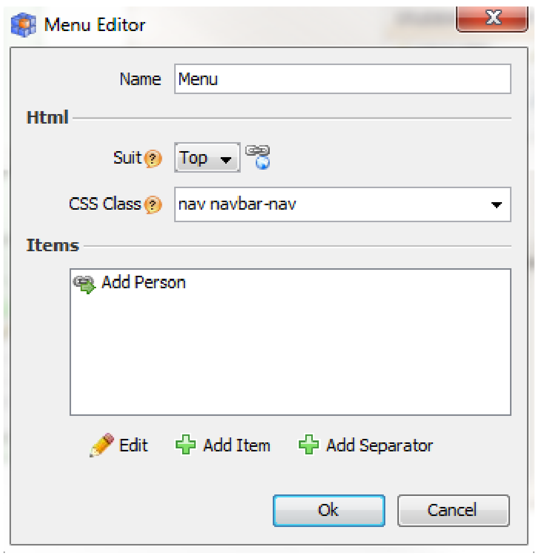
Add menus in the top menu panel. Menu shortcuts are referring another menu, so you don't have to define a menu each time you define a new form and rather refer an existing one.



PERSONID	ADDRESS	POSTCODE	BIRTHDATE	FIRSTNAME	LASTNAME	PHONE	GENDER	
${PERSONID}	${ADDRESS}	${POSTCODE}	${BIRTHDATE}	${FIRSTNAME}	${LASTNAME}	${PHONE}	${GENDER}	✓ Edit

Page ${listPersonsPageNumber}

In the menu dialog add items as links to other forms, similar with hyperlinks.



Menu Editor

Name: Menu

Html

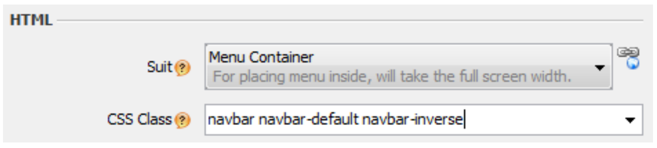
Suit: Top

CSS Class: nav navbar-nav

Items

Add Person

In HTML the menu will show correct if you change the menu panel suit to 'Menu Container' and CSS like here.

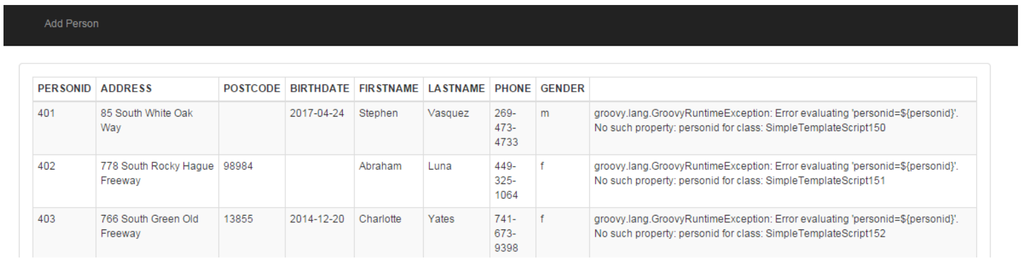


HTML

Suit: Menu Container
For placing menu inside, will take the full screen width.

CSS Class: navbar navbar-default navbar-inverse

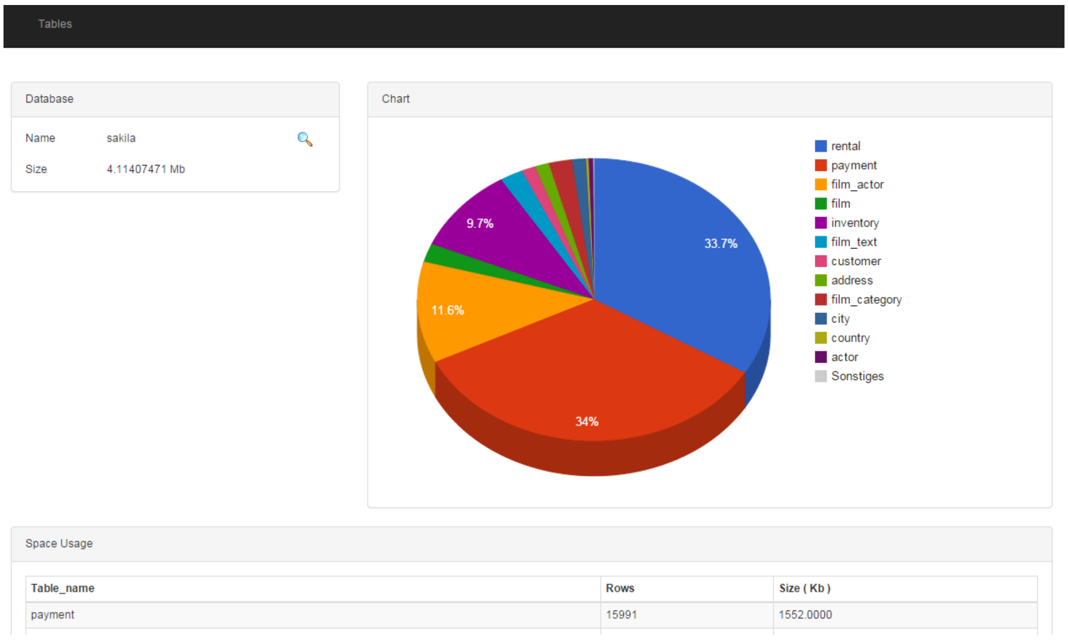
The HTML page will look like:



PERSONID	ADDRESS	POSTCODE	BIRTHDATE	FIRSTNAME	LASTNAME	PHONE	GENDER	
401	85 South White Oak Way		2017-04-24	Stephen	Vasquez	269-473-4733	m	groovy.lang.GroovyRuntimeException: Error evaluating 'personid=${personid}'. No such property: personid for class: SimpleTemplateScript150
402	778 South Rocky Hague Freeway	98984		Abraham	Luna	449-325-1064	f	groovy.lang.GroovyRuntimeException: Error evaluating 'personid=${personid}'. No such property: personid for class: SimpleTemplateScript151
403	766 South Green Old Freeway	13855	2014-12-20	Charlotte	Yates	741-673-9398	f	groovy.lang.GroovyRuntimeException: Error evaluating 'personid=${personid}'. No such property: personid for class: SimpleTemplateScript152

Charts

The component suite includes charts as well. Line, bar and pie charts can be used. Other charts can be implementing them in the forms template.



Table_name	Rows	Size (Kb)
payment	15991	1552.0000

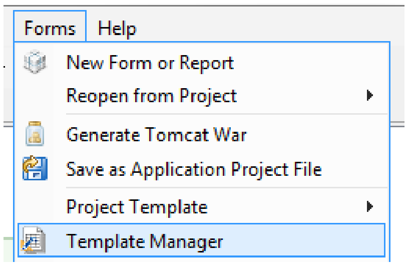
The chart component require a data source script, which should return:

- First column a string (the name of the value)
- Second column the value (number: integer, long, double, etc.)
- Further value columns if the chart is for example a multi-line chart

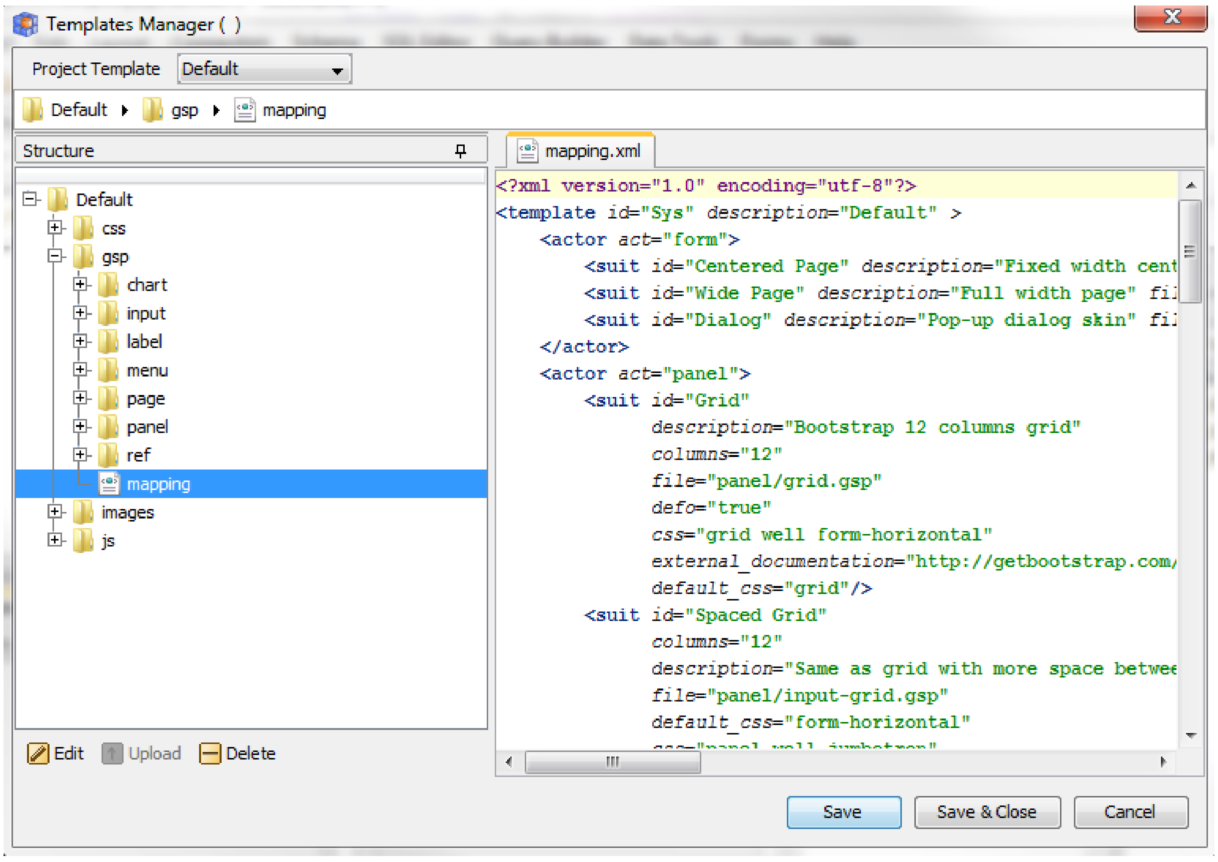
Form Templates

The forms are rendered in HTML using customizable Groovy templates scripts. This means you can modify the look of a component or you can add new representation for a component. We call **'Suit'** a component representation in the template.

The template manager is accessible from the DbSchema main menu.



The template manager is in fact a file explorer. The template files are located on disk in the *C:/Users/<current_user>/.DbSchema/templates* folder. The template include java script files (js), css files and images which will be available in web as they are. The *gsp* folder contains a file named *mapping.xml* which makes the binding of each component to a gsp file.



```
<?xml version="1.0" encoding="utf-8"?>
<template id="Sys" description="Default" >
    <actor act="form">
        <suit id="Centered Page" description="Fixed width cent
        <suit id="Wide Page" description="Full width page" fi
        <suit id="Dialog" description="Pop-up dialog skin" fi
    </actor>
    <actor act="panel">
        <suit id="Grid"
              description="Bootstrap 12 columns grid"
              columns="12"
              file="panel/grid.gsp"
              defo="true"
              css="grid well form-horizontal"
              external_documentation="http://getbootstrap.com/
              default_css="grid"/>
        <suit id="Spaced Grid"
              columns="12"
              description="Same as grid with more space betwee
              file="panel/input-grid.gsp"
              default_css="form-horizontal"
```

As example the label component has few suits: *plain.gsp, h1.gsp, h2.gsp*..., etc. The *h1.gsp* looks like:

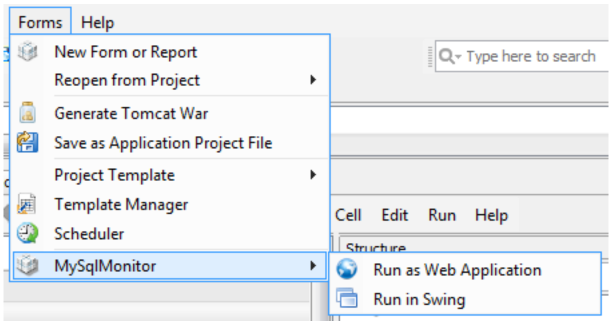
```
<% if (actor.getIconName()!=null){ %><img src='${actor.getIconName()}'>&nbsp;<% }
%><h1>${text}</h1>
```

This will output the <h1>...text...</h1> and place an image tag before if the label has an image. The component is passed under the name 'actor' to the script and has the properties as described in the DbSchema API (read the DbSchema Help).

DbSchema Management Applications

Based on DbSchema forms and reports engine we start implementing management applications for each database. The target features will be space usage, database activity and locks, management of user roles and rights, etc.

This application will be available as **open source project**, so users are free to contribute to their development. Please contact us on support@dbschema.com if you want to contribute to this.



End

We hope you will enjoy the DbSchema forms. Please notice that this is a recent feature in DbSchema, released under the beta version. Please write us back if you find any issues or you wish to get new features.