PHsPeed
V3.0

CourseWare - Create a todo manager

Table of contents

Introduction

Welcome to the world of PHsPeed!

In our course-ware, we will teach you how to work with PHsPeed and how the system can help you achieve your development goals faster, consistently, and with fun. If you are new to PHsPeed, then we advise you to follow lesson 0: an introduction first. This lesson will teach you the very basic concepts of PHsPeed. If you are also new to PHP, then you will find a crash course for PHP here too.

To get started you need to have PHsPeed installed. If you haven't already, then download, and install it from our website https://www.phspeed.com. Don't forget to register on our forum, so that you can drop your questions. Use a valid email address, not some fake ten minute mail or so. The European privacy laws belong to the most severe in the world, we do NOT share your data to anyone, unless we are forced to do so by law.

Disclaimer:

We show you in these lessons all kinds of features that you can use to support your development. Because of that, we make deliberate mistakes to show you how to correct when things go wrong. We also do not do the things always in the most efficient way, just to be able to show you some feature. Never the less, even in this non-optimized development, you will notice a significant increase of production and decrease of development time. When you work with PHsPeed, then you will find your best practice just by doing and learning.
It is possible that the presented screenshots differ from the version you are running. PHsPeed gets regular updates that sometimes changes the appearances of certain forms. These series of lessons have been written, based on version 3, but it should not be hard to implement these on higher versions.

Building a todo list.

Part 1. Simple to-do list

Before starting to build your application, you need to be aware of the way you want to save the data. A good designed database model will make development of your application the most efficient.

The to-do list that we will use looks as follows:

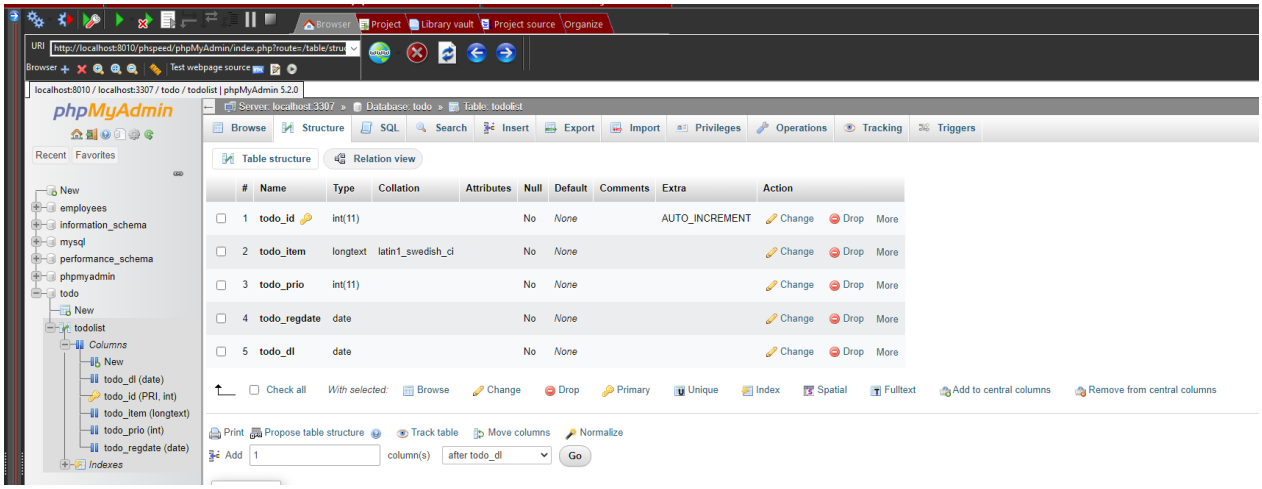
TodoList

This is the main table that contains the todo items.

report (registers the bug, enhancement request, etc.		
todo_id	integer primary key, autonumbering	Unique identifier of the to do item
todo_item	longtext	The to-do item in text
todo_prio	integer	Priority
todo_regdate	date	Date of registration
todo_regdl	date	Deadline

It is easy to enter this data using the embedded PHPMyAdmin, but you can also use the following script (If you are not using MySQL, you might need to change datatypes). You can create this table in the existing

sample database, but - better - is to create a new database. We called it 'todo'.



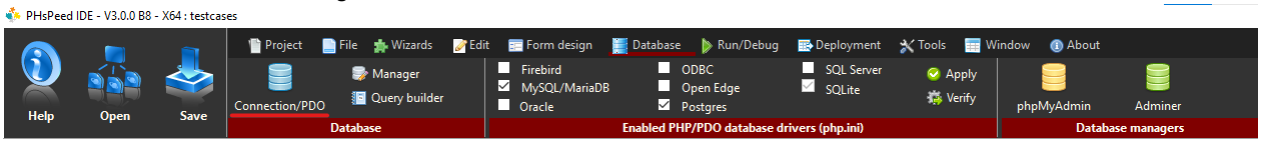

```
CREATE TABLE `todolist` (
  todo_id int(11) NOT NULL AUTO_INCREMENT,
  todo_item longtext NOT NULL,
  todo_prio int(11) NOT NULL,
  todo_regdate date NOT NULL,
  todo_dl date NOT NULL,
  PRIMARY KEY (todo_id)
)
```

Create database connection

PHsPeed applications communicate with a database to store and retrieve information. Depending on the situation, you might have a single development database containing all your projects, or create separate databases for your applications. Either way, PHsPeed requires that you define a data base connection *independent* from your projects. The idea is that you can define a certain database connection and reuse that over (one or more) projects.

Not relevant for this project, but you can have multiple connections in your project, that can point to different databases or brands (combine oracle and mysql for example).

To create a new connection, go to database -> Connection/PDO



Two connections?

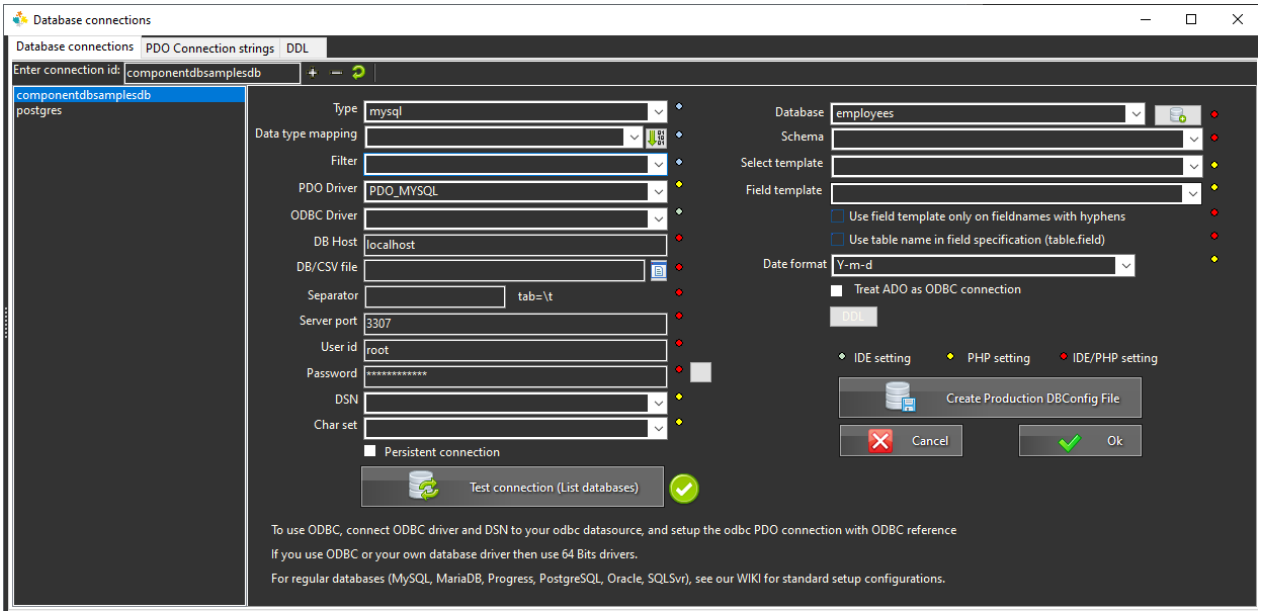
To work with PHsPeed, you require two connection definitions:

PHsPeed IDE

The ide needs to be able to access your database so that it can retrieve metadata, like your database structure, tables and fields

PHP

PHP needs to be able to access your database, and to be able to it requires a PDO specification. PDO is a database-independent layer that binds PHP to the database. Theoretically, PDO makes your applications database independent and that you can change database flavor by just pointing the PDO driver into another database. Unfortunately this is not always the case. For instance, using auto numbering on fields works differently for all database brands, and SQL is not that standard as well. PHsPeed, will generate database specific code when required, so if you want to migrate to another brand, than you must bind the database connection to another connector and regenerate your code.



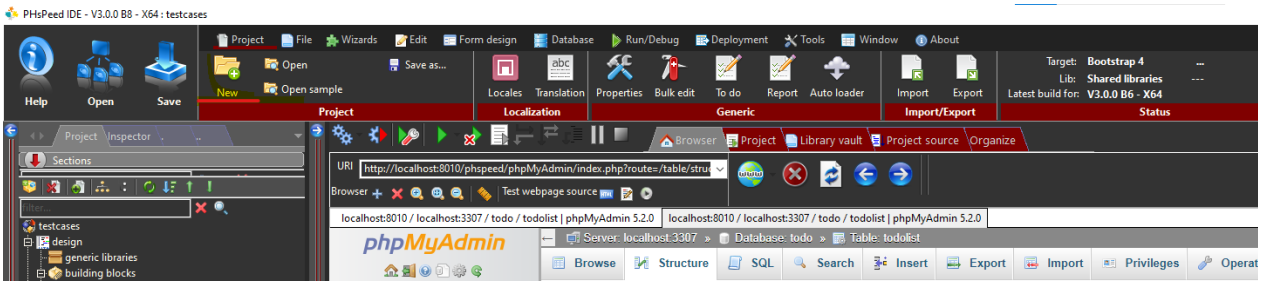
In this screenshot you can see that the PDO_MYSQL (the pdo driver) is already available. If you have another database, then you must create a new PDO connection string.

To add a new connection, enter a name as id and click on +, then enter the required data. Click on 'test connection' to confirm the connection, then the database drop-down will show you the new todo database. Use the correct select template, and click on ok. You have now created a valid database connection.

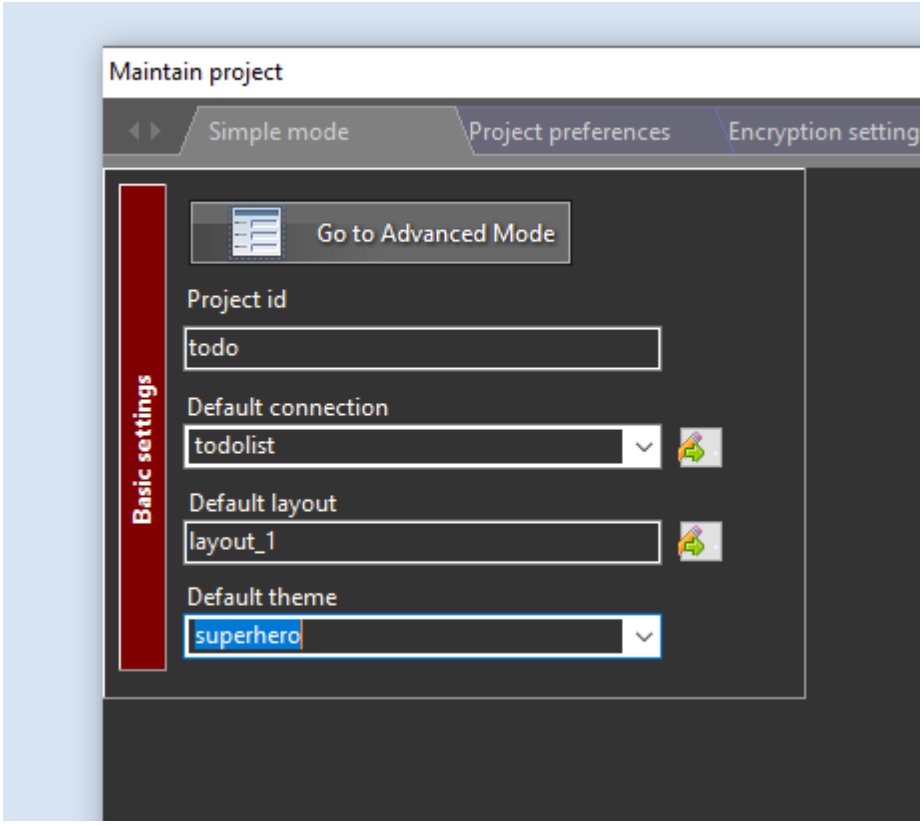
Create Project

Before creating applications, you need to create a project. A project is a container that contains all information that is required to generate your code. To create a new project go to the Project tab, and select

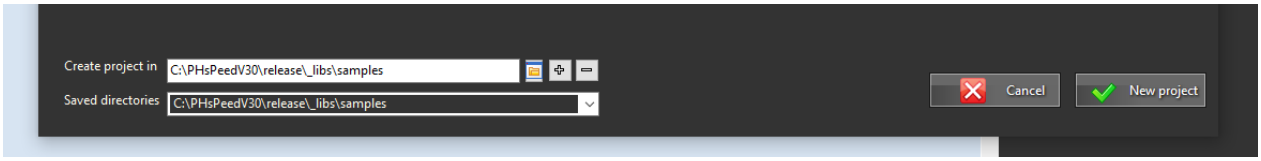
'New'.



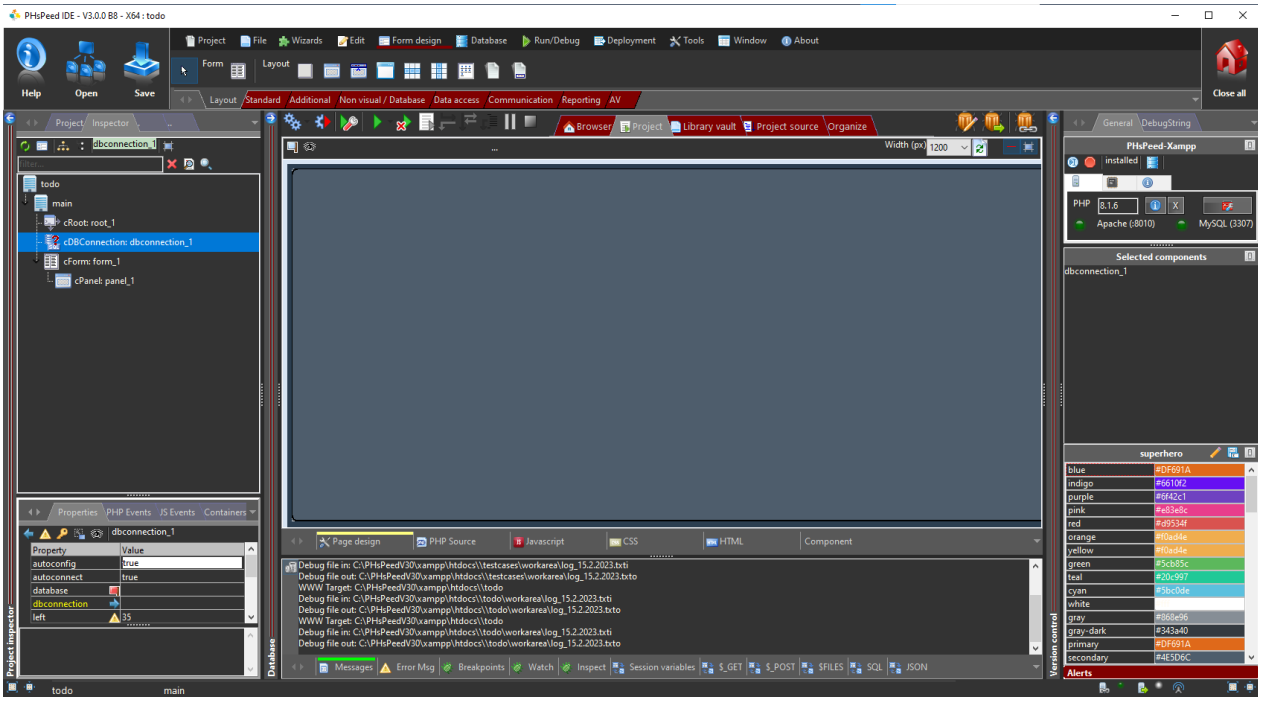
In this case you will use the simple form:



Select the required folder to store the project, and click on 'new project'



Confirm and the main form will appear:

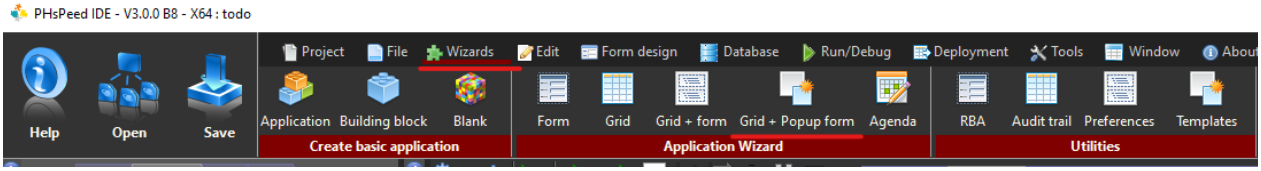


By default PHsPeed will create a form called 'main'. This form is usually used to create your login form, and in this part of the course-ware, we will not use it (but will in part 2).

Create ToDo Grid Crud form

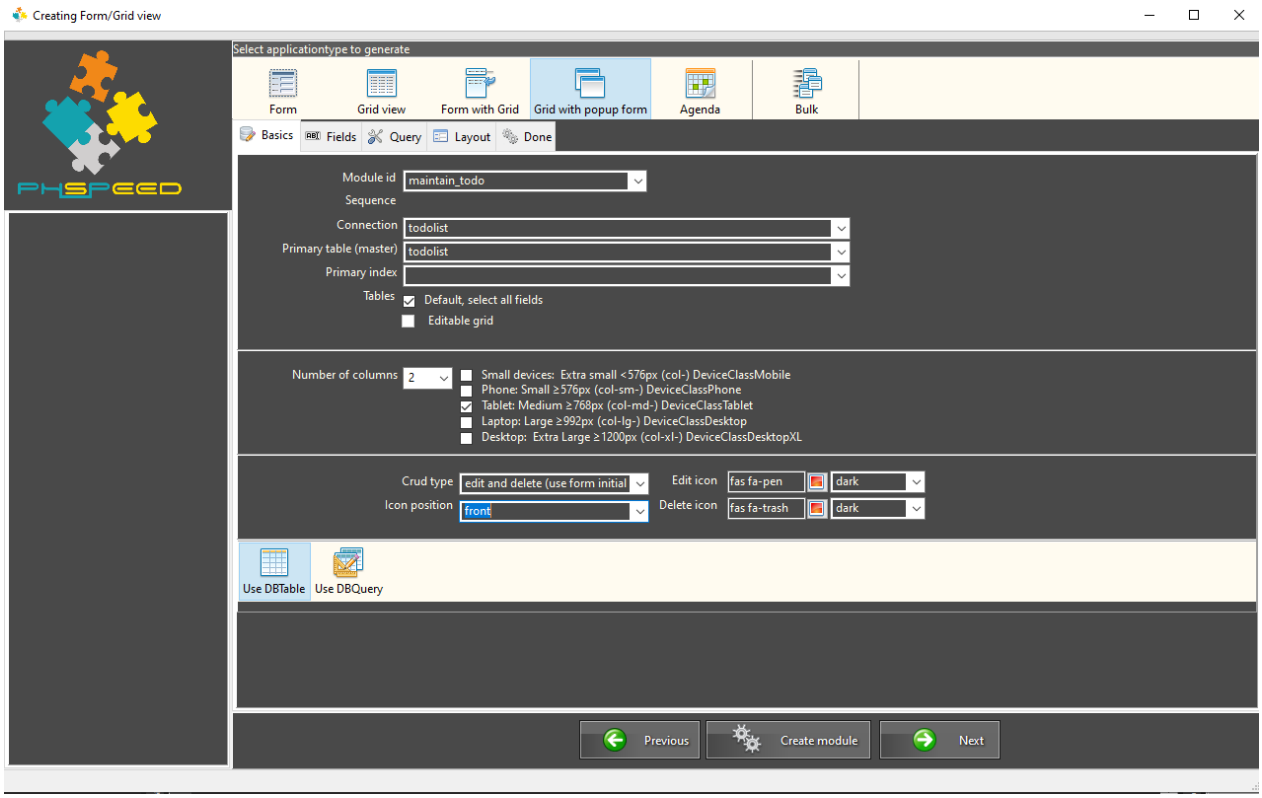
In this part1, we will create a grid, that allows us to add new items, to set the status of an item. For history purposes, we will remove the delete option, so you will always be able to search for old items and when you have completed it.

First, select 'Wizards', and hten 'Grid + Popup form'.

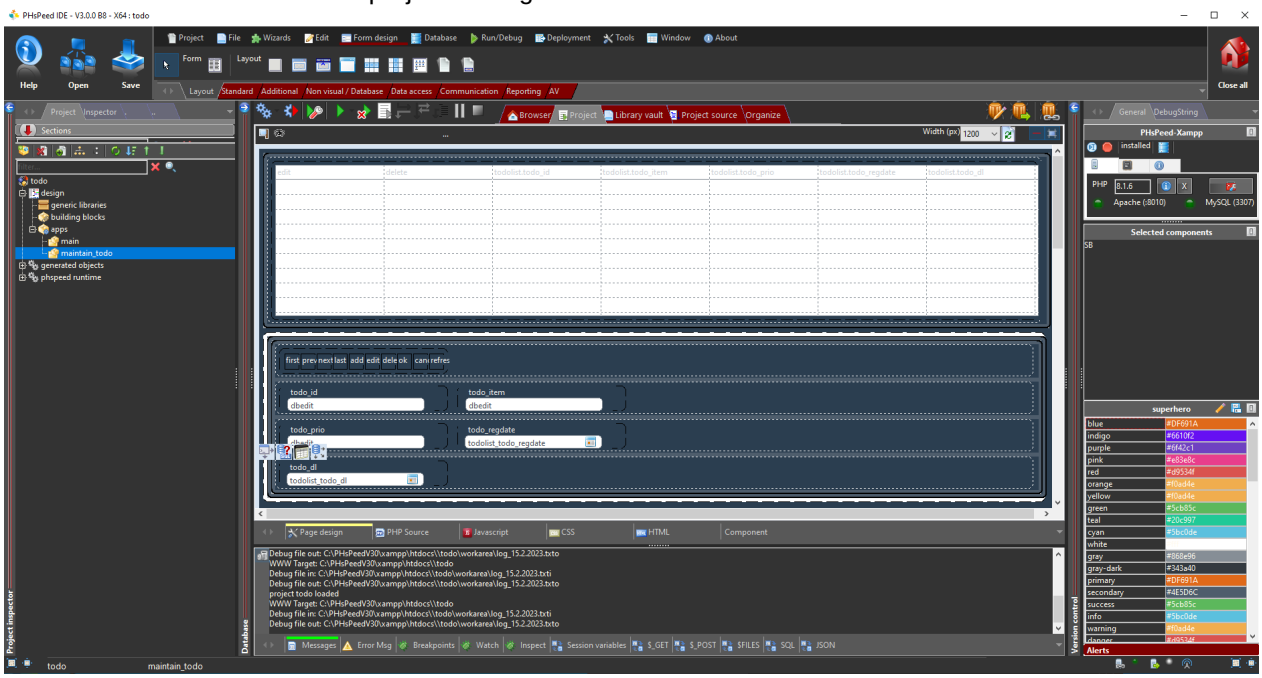


The Wizard form requires some minimal information to create your application. There are many options, but here we keep it simple:

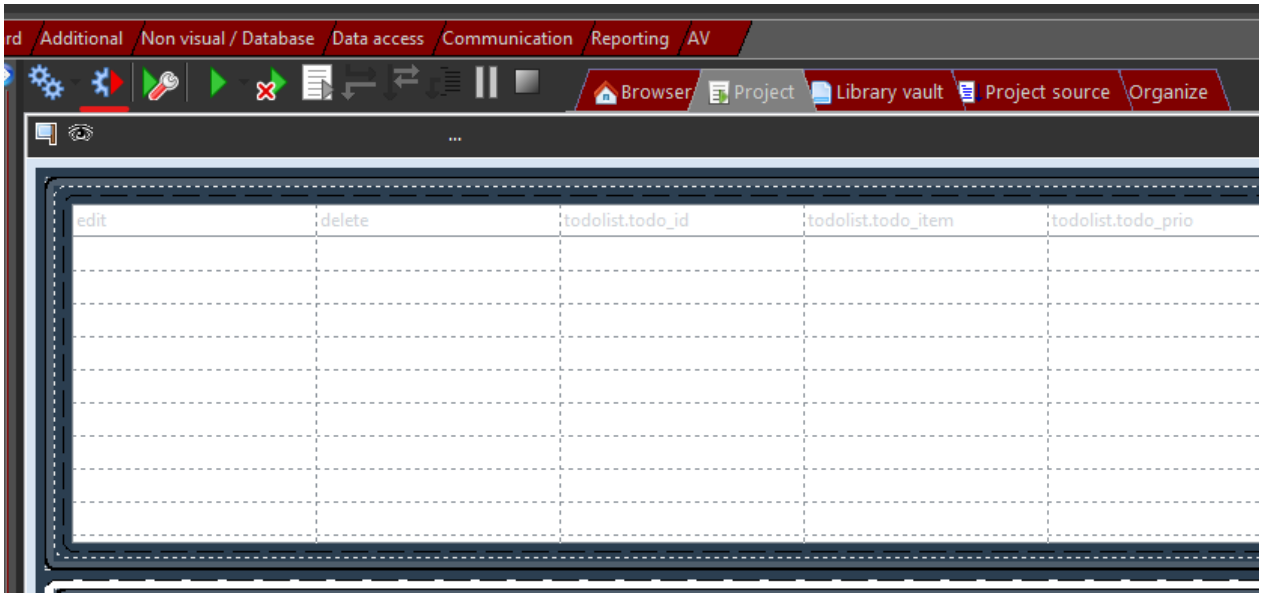
Module id: this is a unique name for your module. (If you select an existing module then the application will be **appended** to your application (which is, i..e, very useful for master-detail relations)
Connection is filled by default by the name in the project, change it if required.
Primary table: the drop down will show you all the available tables in your connection (here todolist).
Crud type: change it to 'edit and delete' so that you get separate columns for edit and delete.



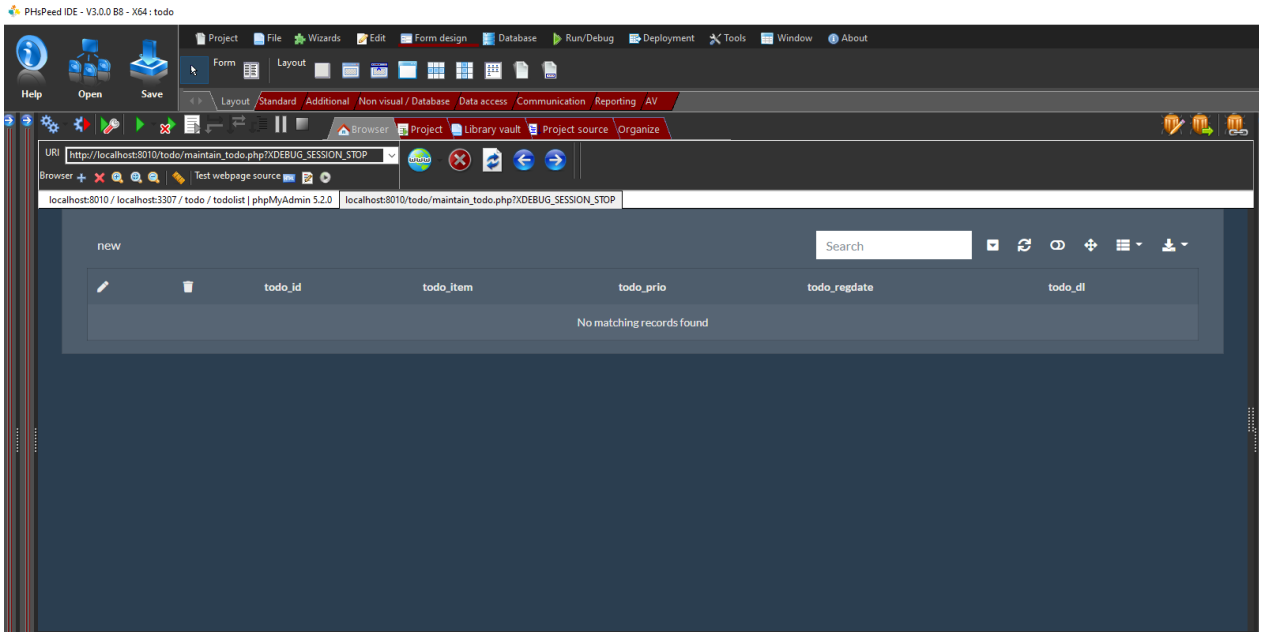
Click on 'Create module'. In the project manager the new module will be added. Click on the new module:



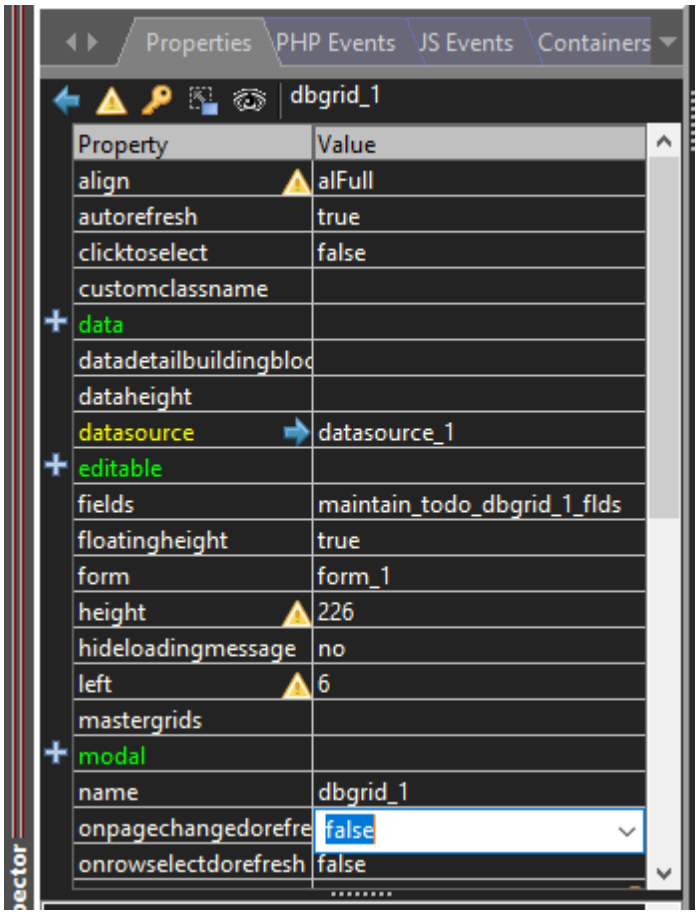
The fastest way to generate and run your applicaiton is to click on the 'generate and run button':\



You will see a result similar to this.

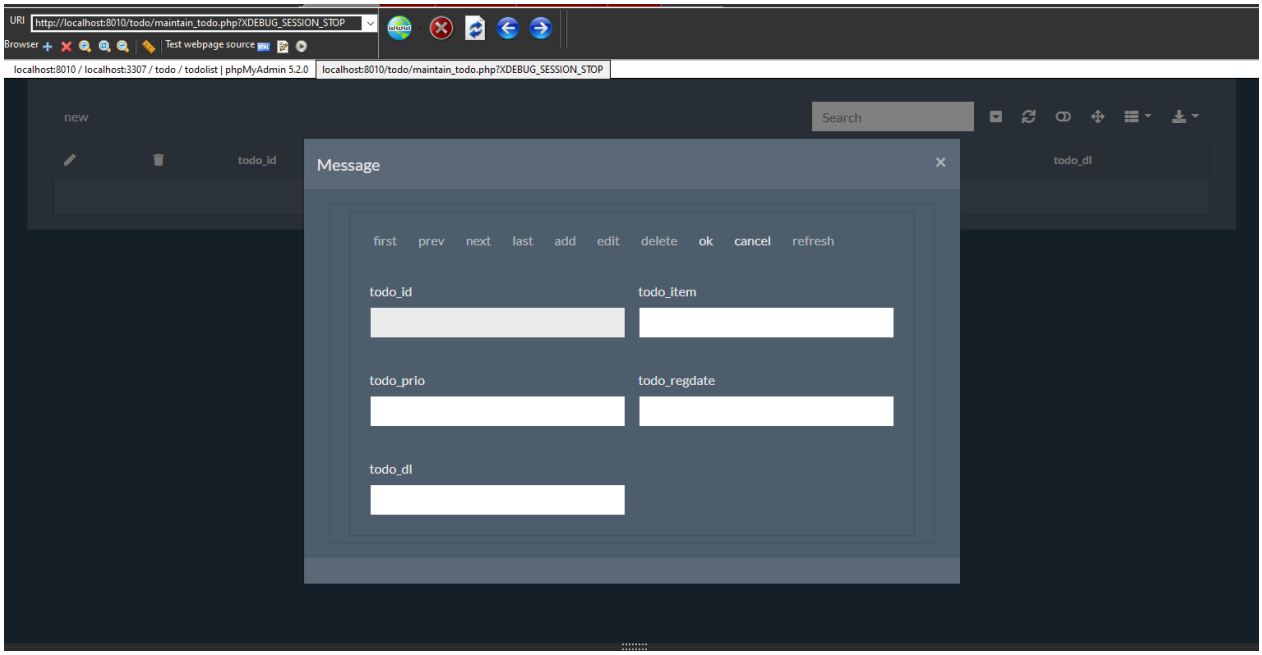


A final step is to change the 'onRowSelectDoRefresh' property of the dbgrid, and set it to false. This is required to avoid two events happening at the same time, while updating a record, and which can cause conflicts:

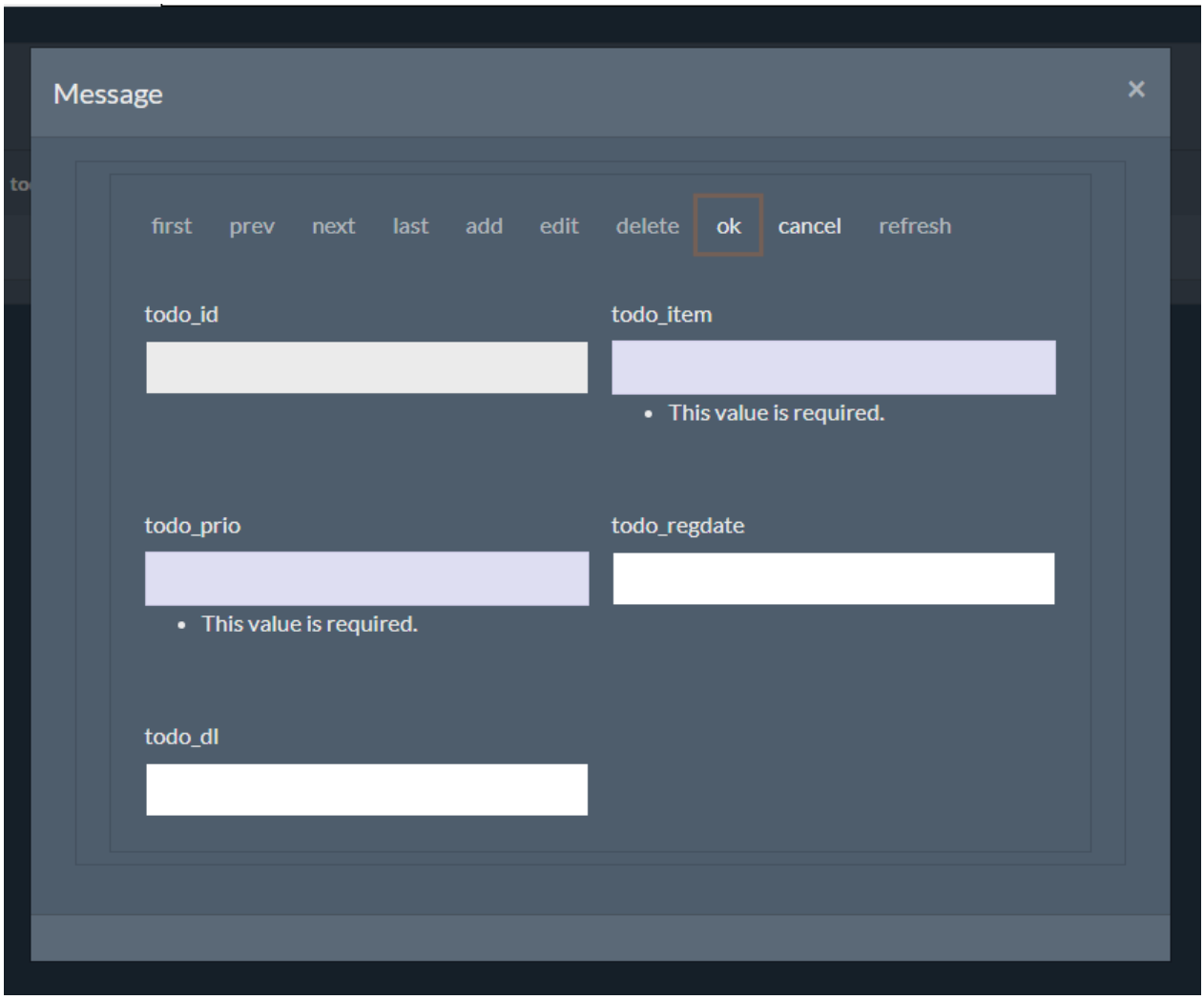


Test the application

As the application is generated we can test it, and make a list of modifications that we want to achieve. As there are no records yet, click on 'new'

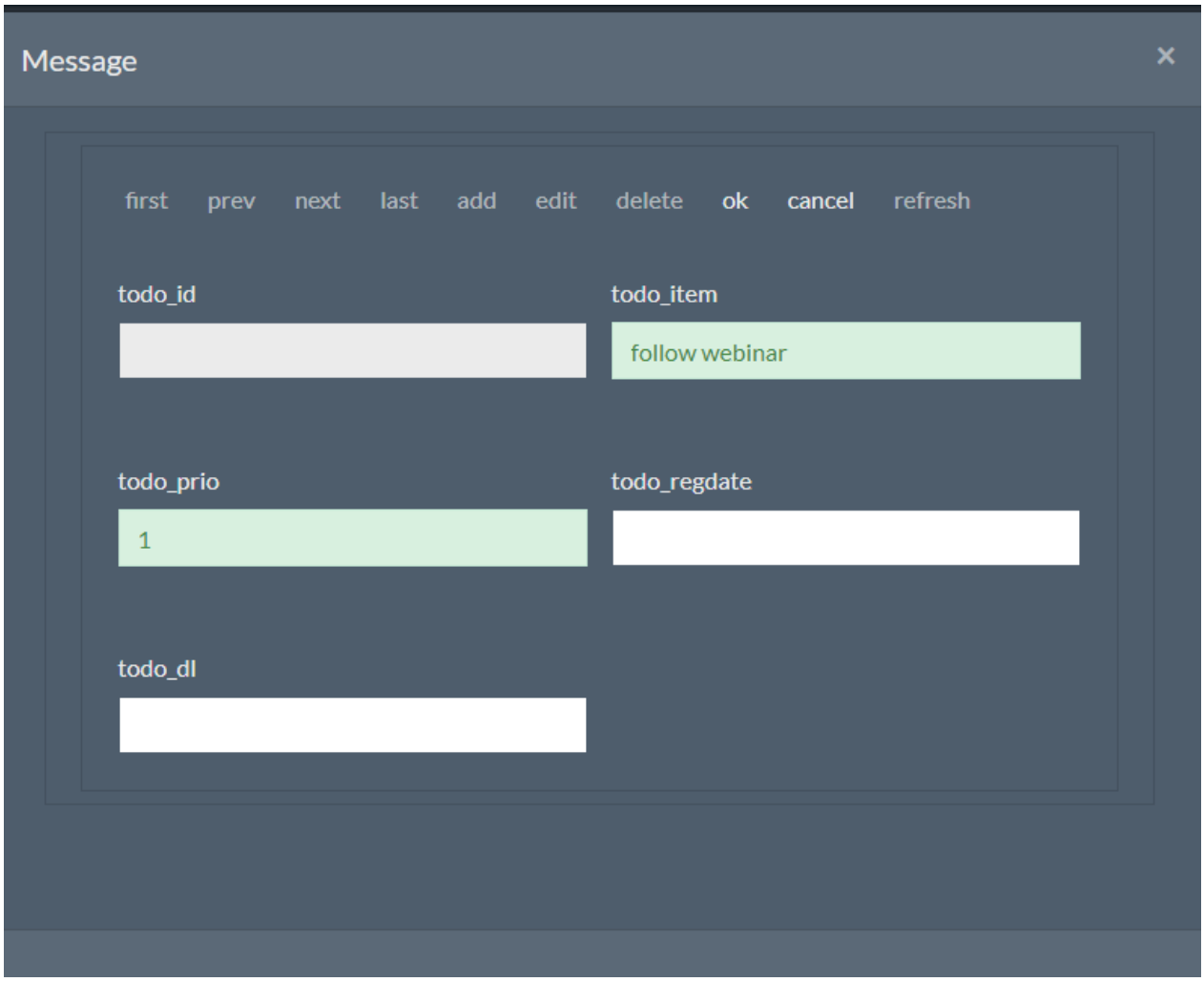


The first thing that you see is that the labels of the table fields are used as a label for the fields. We'll put that on our list of changes. Do not enter data and click on 'ok'.



- The todo_id cannot be entered, which is ok, as this was defined as autonumbering.
- The text fields are required, and this is ok too.
- The date fields are not required, which needs to change in our situation.
- The todo_regdate is not initially filled with the date of today, and should be read-only
- The todo_item is a simple text field, and this should be a wysiwyg field.
- The title 'message' is too generic.

Enter some initial data and click 'ok'.



The data is stored, the dates will have an initial value of 0000-00-00 (in this case we use the MySQL format which is yyyy-mm-dd)

- The icons to edit or to delete are barely visible in this theme, we need to change that.
- The toolbar of the grid contains many items that are not required, we'll change that.

Beautifying the form

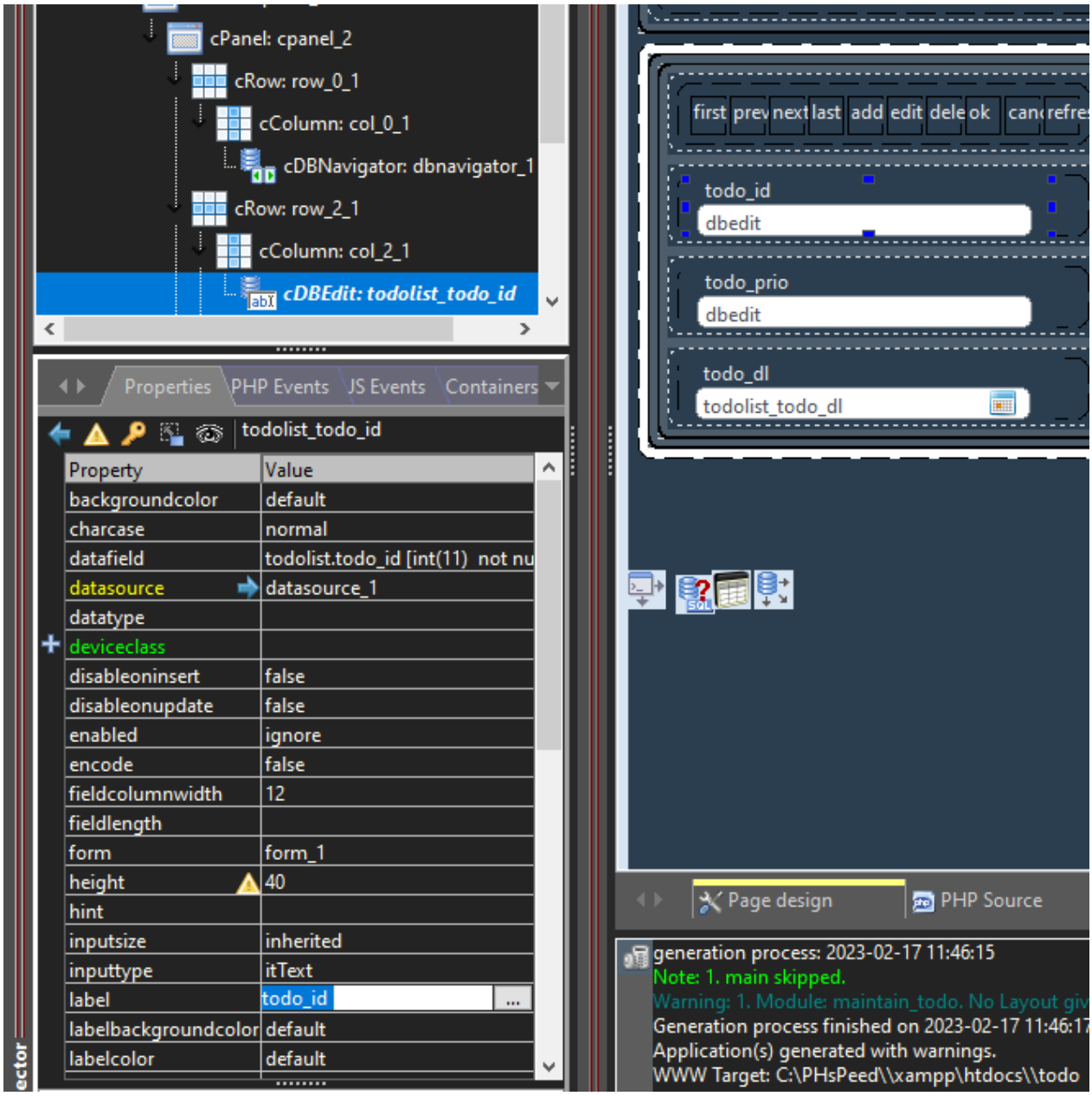
Now that we have created our first form, let's beautify the form by applying changes to the reported items.

- The todo_item is a simple text field, and this should be a wysiwyg field.
- The title 'message' is too generic.
- The icons to edit or to delete are barely visible in this theme, we need to change that.
- The toolbar of the grid contains many items that are not required, we'll change that.

Change labels of fields

To change the labels of the fields we have to change those of the editable fields and the labels of the grid-columns.

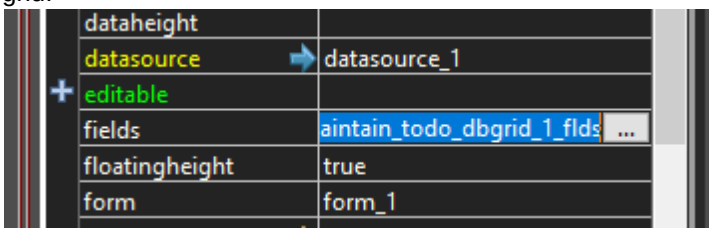
Changing the form fields

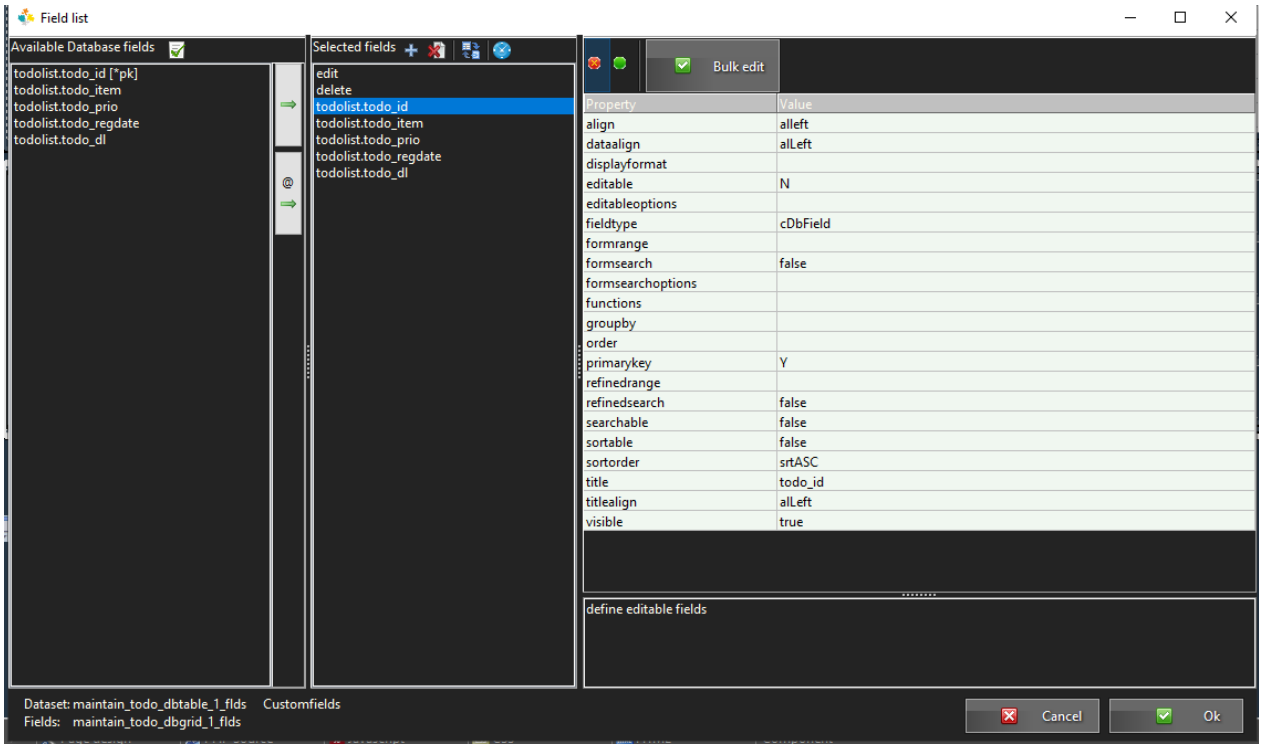


Changing the labels of the form fields is fairly simple. Just select the component and change the label.

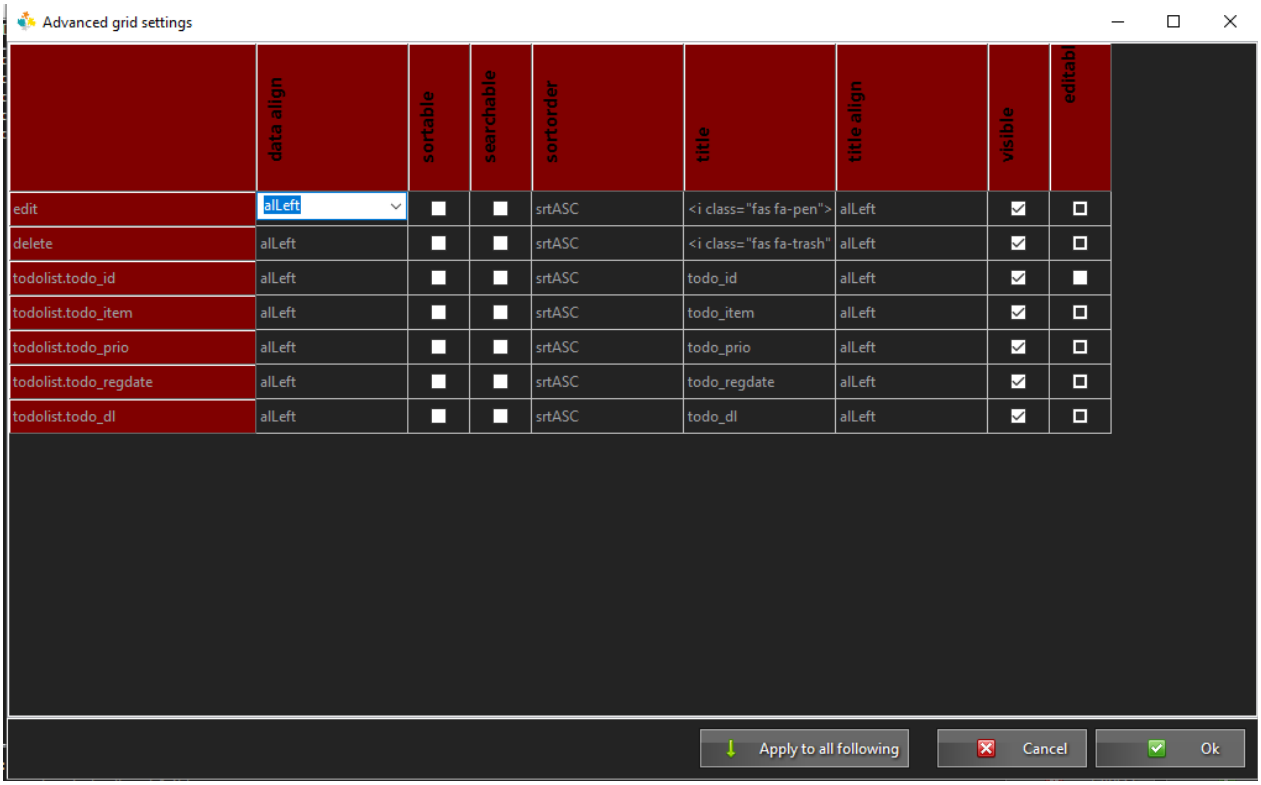
Changing labels in the grid

Changing the labels of the grid can be done by selecting the grid and then open the fields property of the grid.

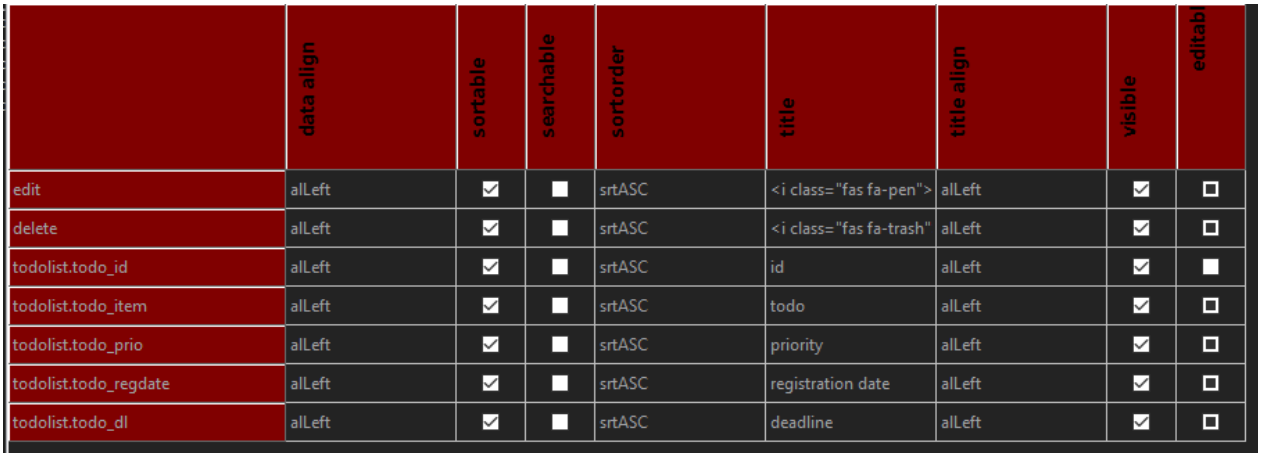




From this form you can manage many properties of the grid columns. It is also possible to set sort options so you can sort on selected columns. Like the form fields, you can apply the changes one-by-one, but a more convenient way is to use the bulk edit option:



You can now apply your changes of all the fields in the grid on one form. If you require to have all fields sortable, then click the sortable of the first field and then click 'apply to all following' which will copy the selected value into the succeeding elements.

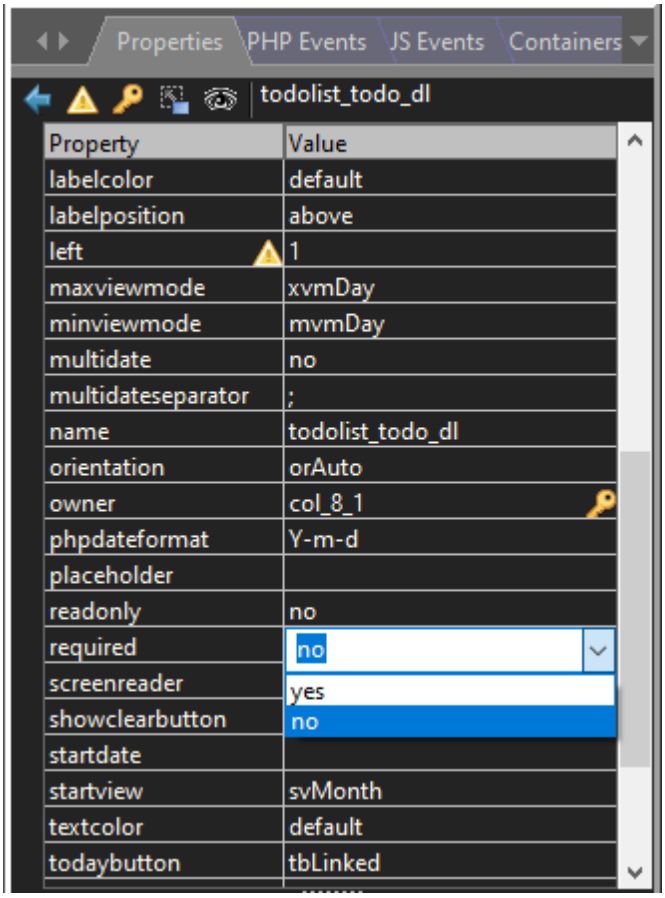


Click ok, and run the application, to verify the result.

The date fields should be required

The date fields are not required, which needs to change in our situation.

To make a field mandatory, you have to set the required option. In this case you need to set the deadline as required

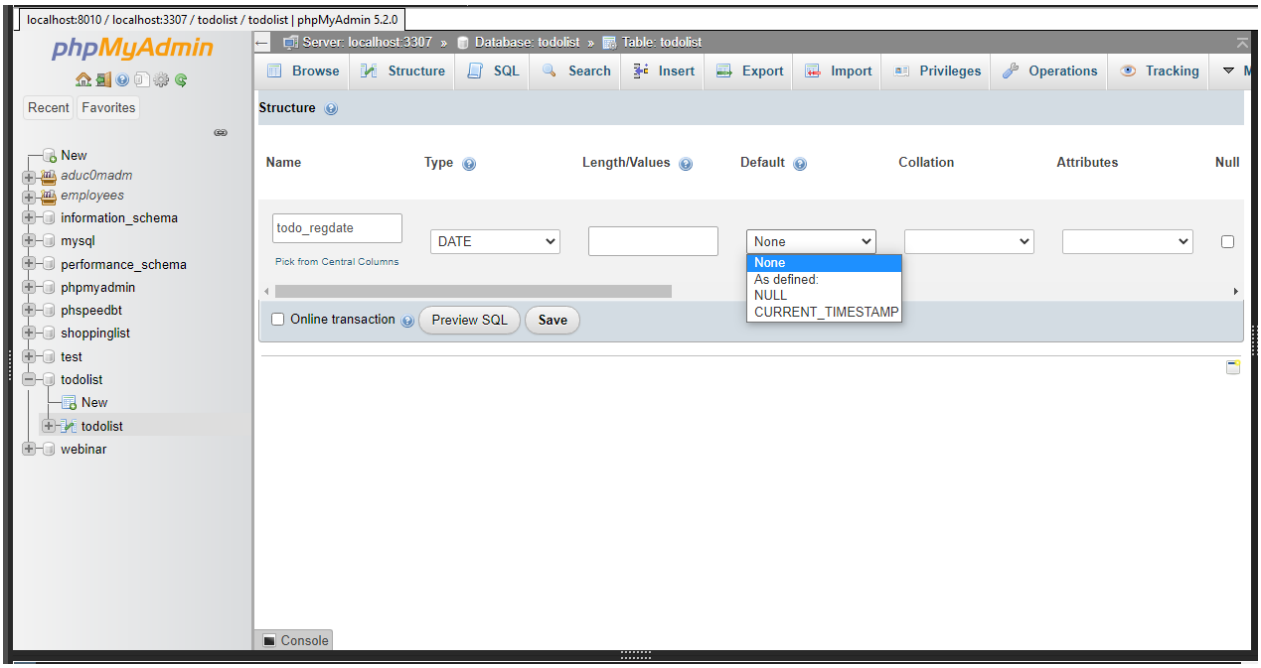


The todo_regdate is not initially filled with the date of today

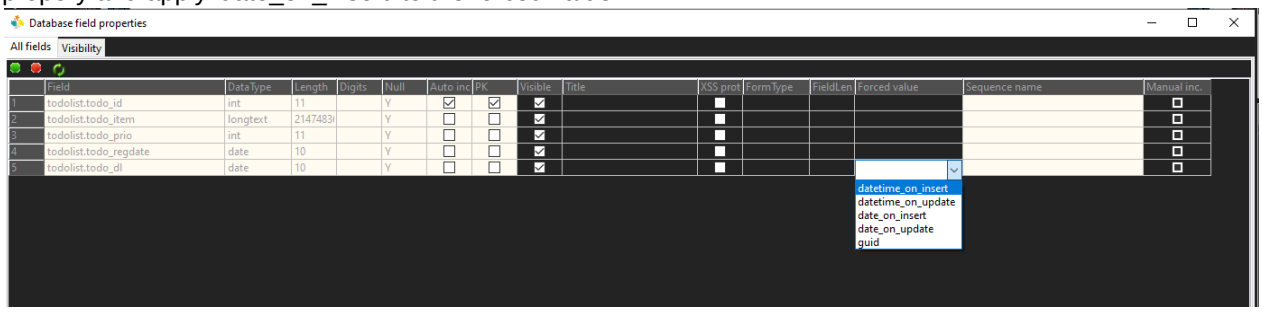
The todo_regdate is not initially filled with the date of today, and should be read-only

There are different ways of doing this.

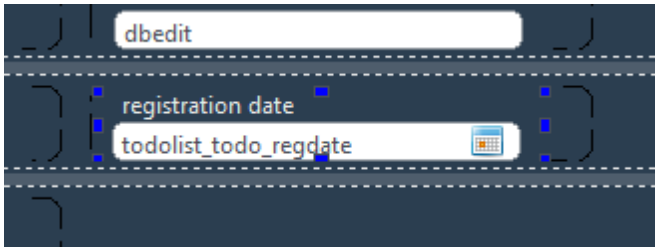
The first way is to define a rule at database level, so that when a field is inserted blank that it gets the default value.



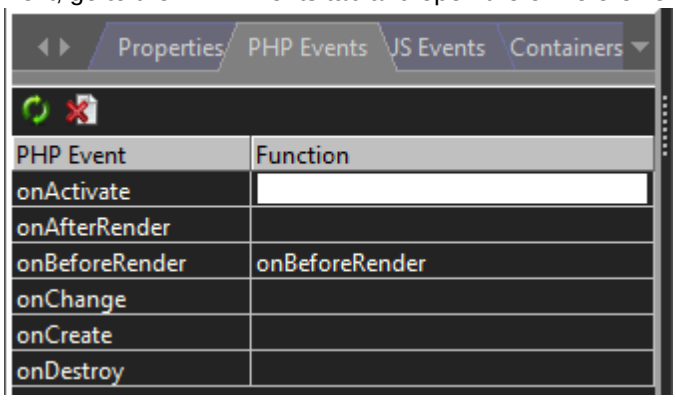
The second method is to apply a rule to the field. To do this, select the DBTable component, open the fields propery and apply 'date_on_insert' to the forced value.



The third option is to set the default value of the field and make the field read-only.
To make a field read-only, select it in the form designer and set the read-only property. To set a default value, you can use the 'value' property of the field. But as the current date is a PHP function and is not static, you need to apply a line of PHP. The best event to do that is by using the onBeforeRender event of the registration date field:

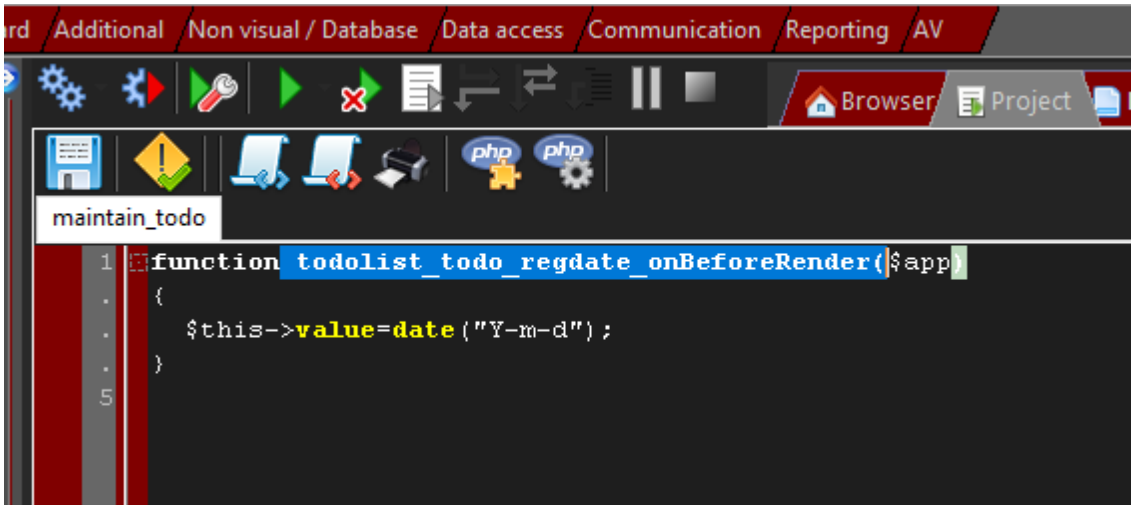


Next, go to the PHP Events tab and open the onBeforeRender event:



This will open the PHP editor and apply

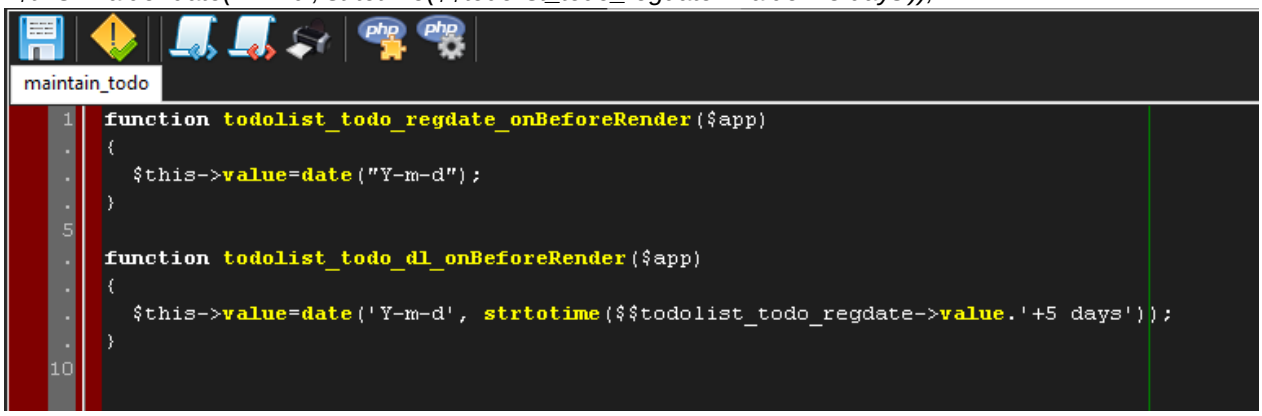
$this->value=date("Y-m-d");



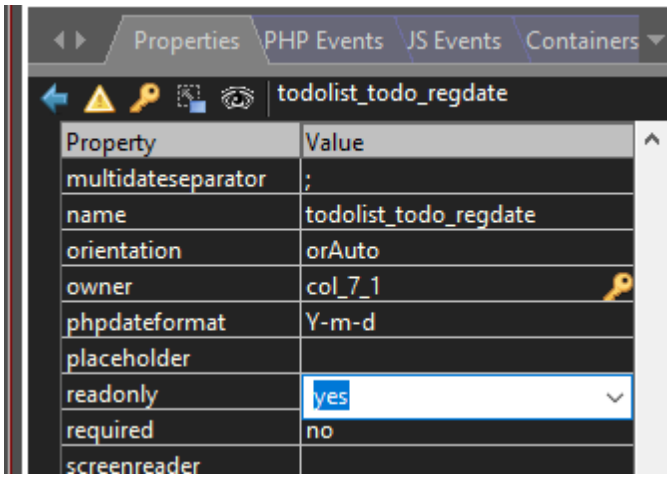
As you are changing the value of the selected component itself, you can use $this->
If you want to change the value of another component user $$

For instance, you want a default deadline of 5 days after the registration date. To set this value you could add the following line in the onBeforeRender event of the deadline field:

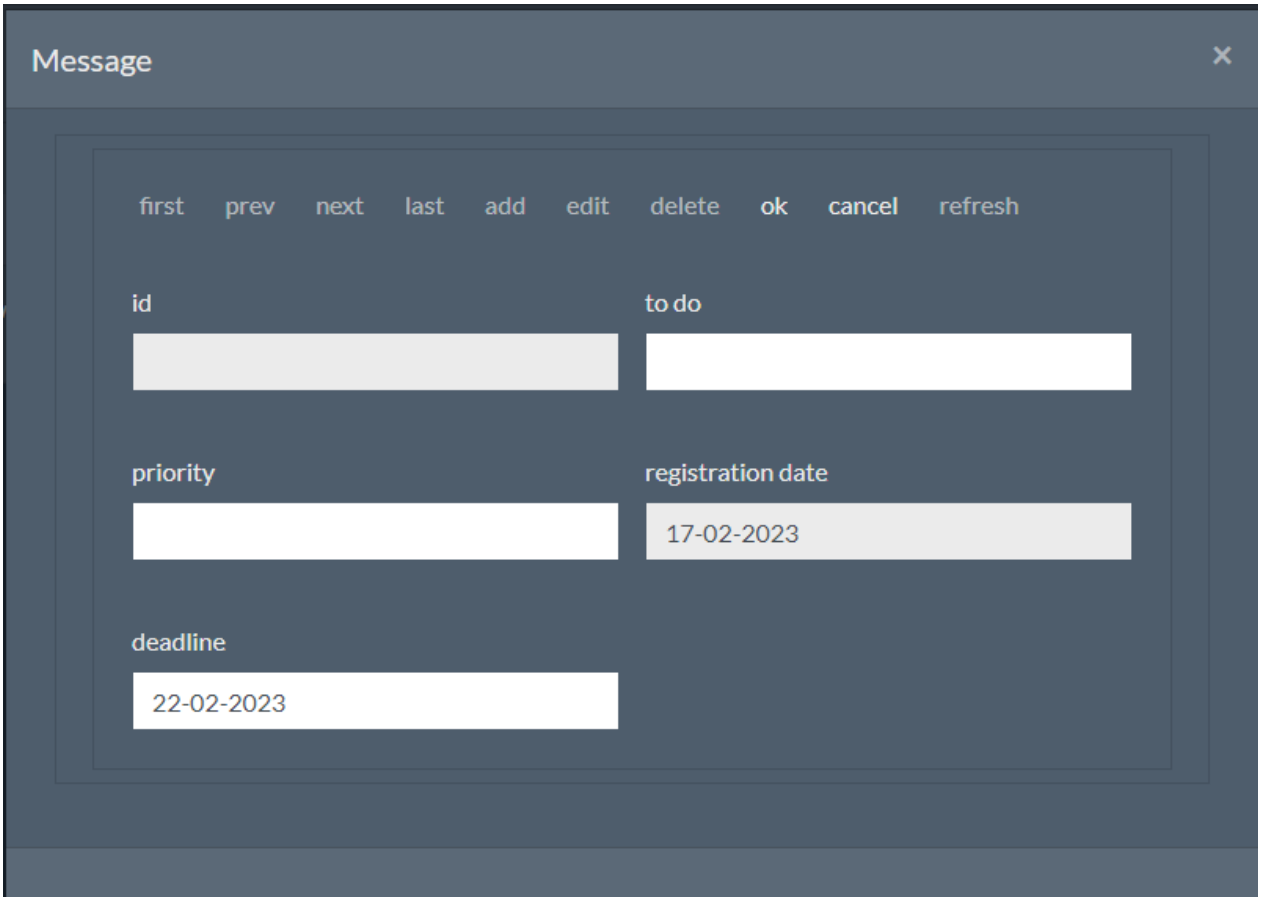
$this->value=date('Y-m-d', strtotime($$todolist_todo_regdate->value.'+5 days'));



To make the registration date read-only:



Now you can generate and run the code:

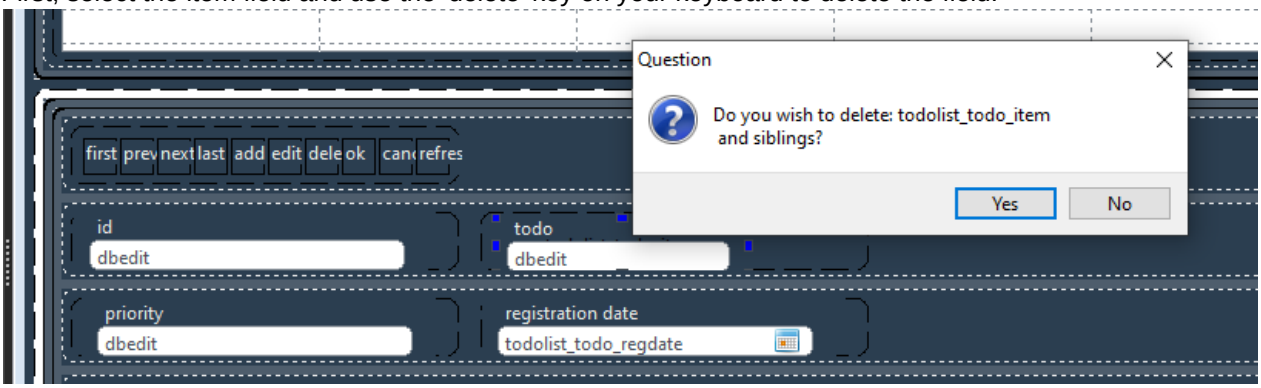


- The

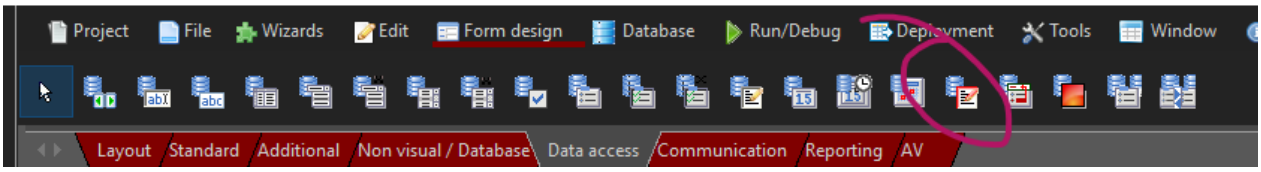
The todo_item is a simple text field, and this should be a wysiwyg field.

It is not possible to use the feature to change the edit type of cDBEdit to cDBWysiwyg. The reason is that the properties of both components do not have enough similarities. Instead we need to delete the edit component and apply the new one.

First, select the item field and use the 'delete' key on your keyboard to delete the field.

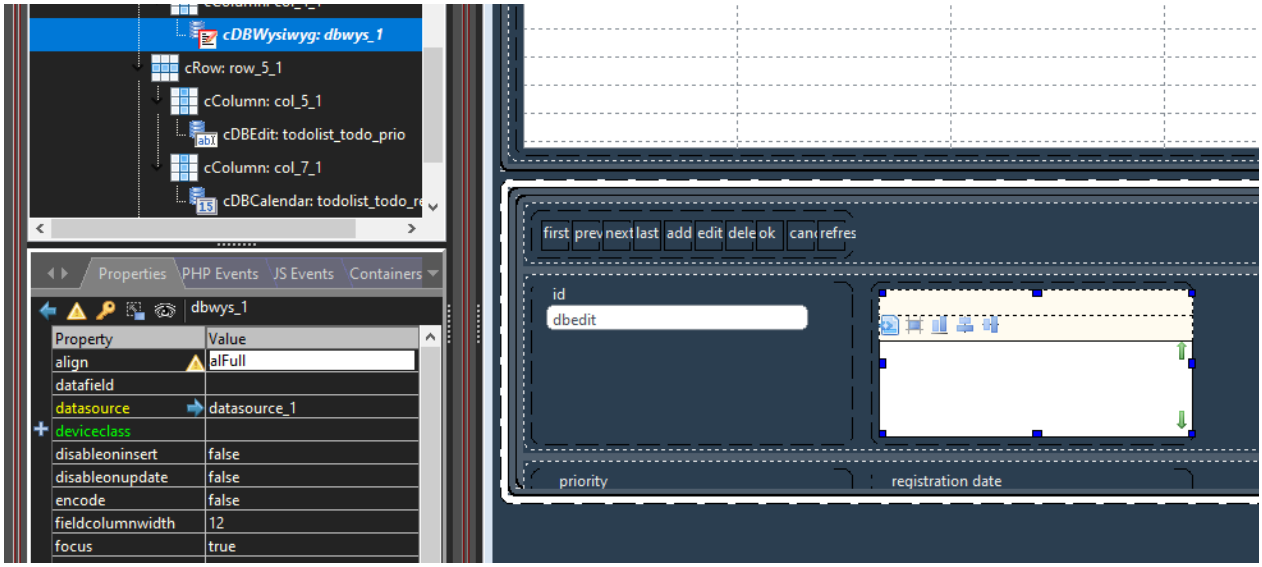


Next, select the DBWysiwyg from the component panel and place it in the (now empty) column.

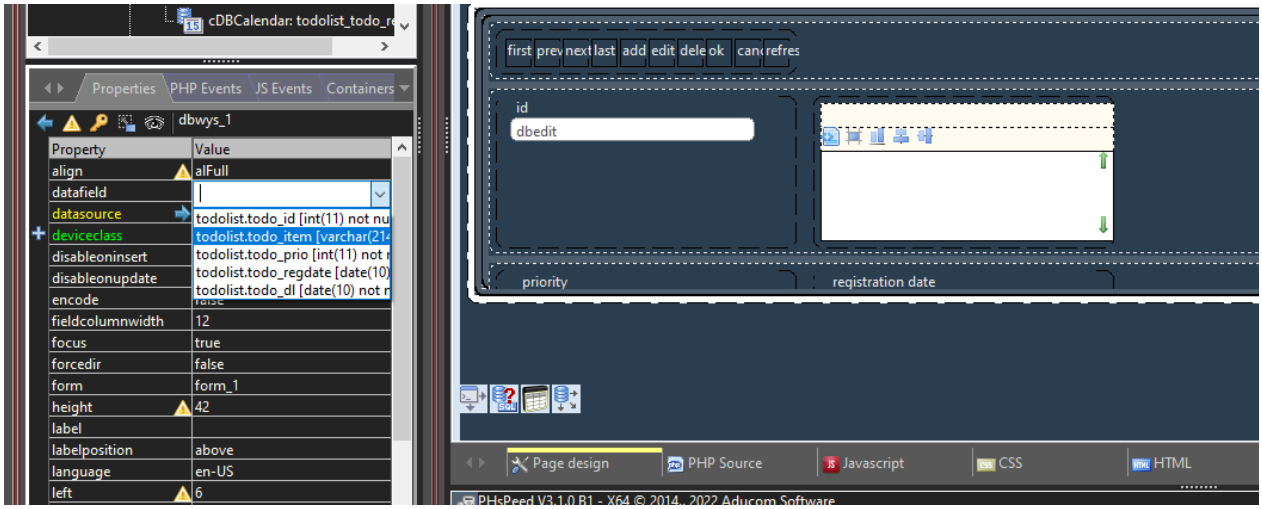


Click on the row and use the mouse to make the row heigher, so that you can see the wysiwyg field better.

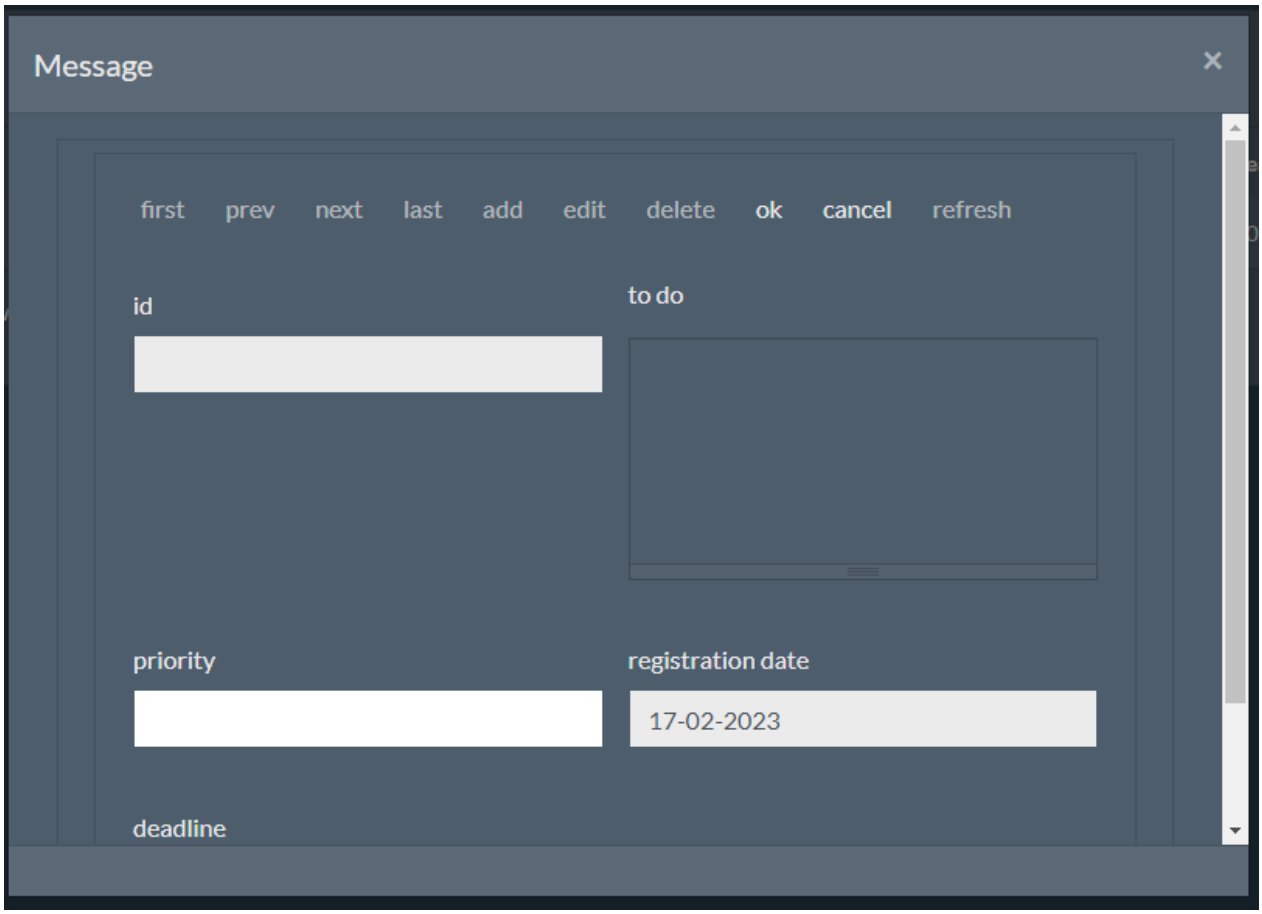
Also set the align property of the field to alfull.



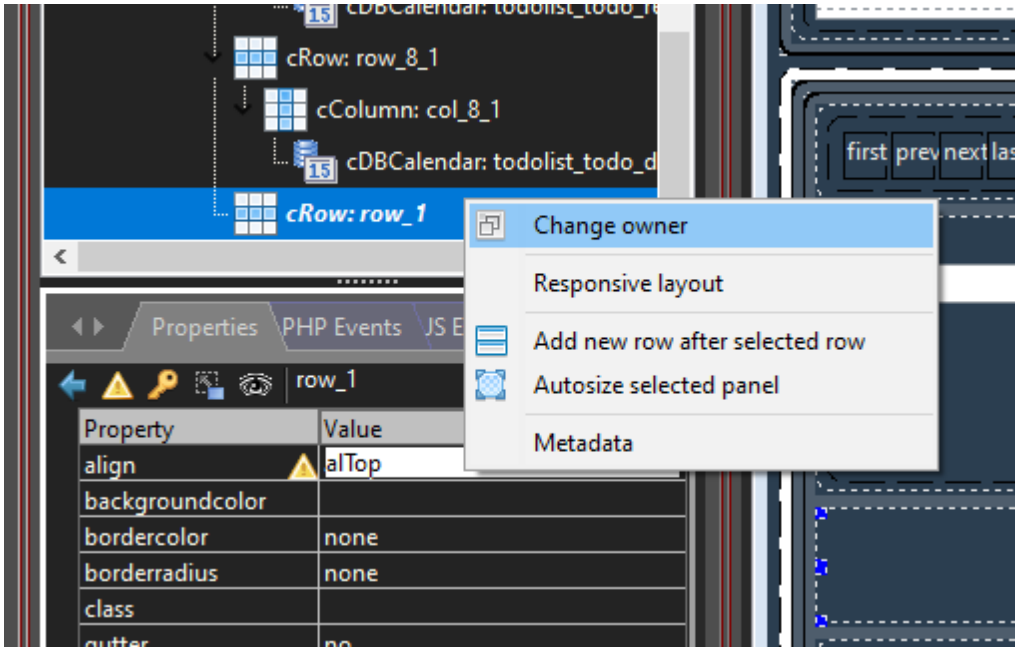
Verify that the datasource of the wysiwyg component points to datasource_1 and set the correct datafield



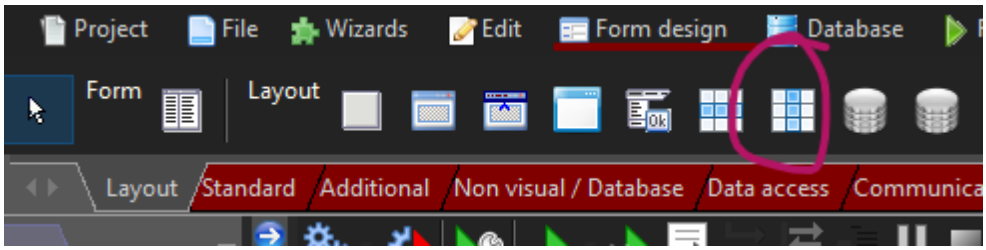
Now you can test the application by generating and running the code.



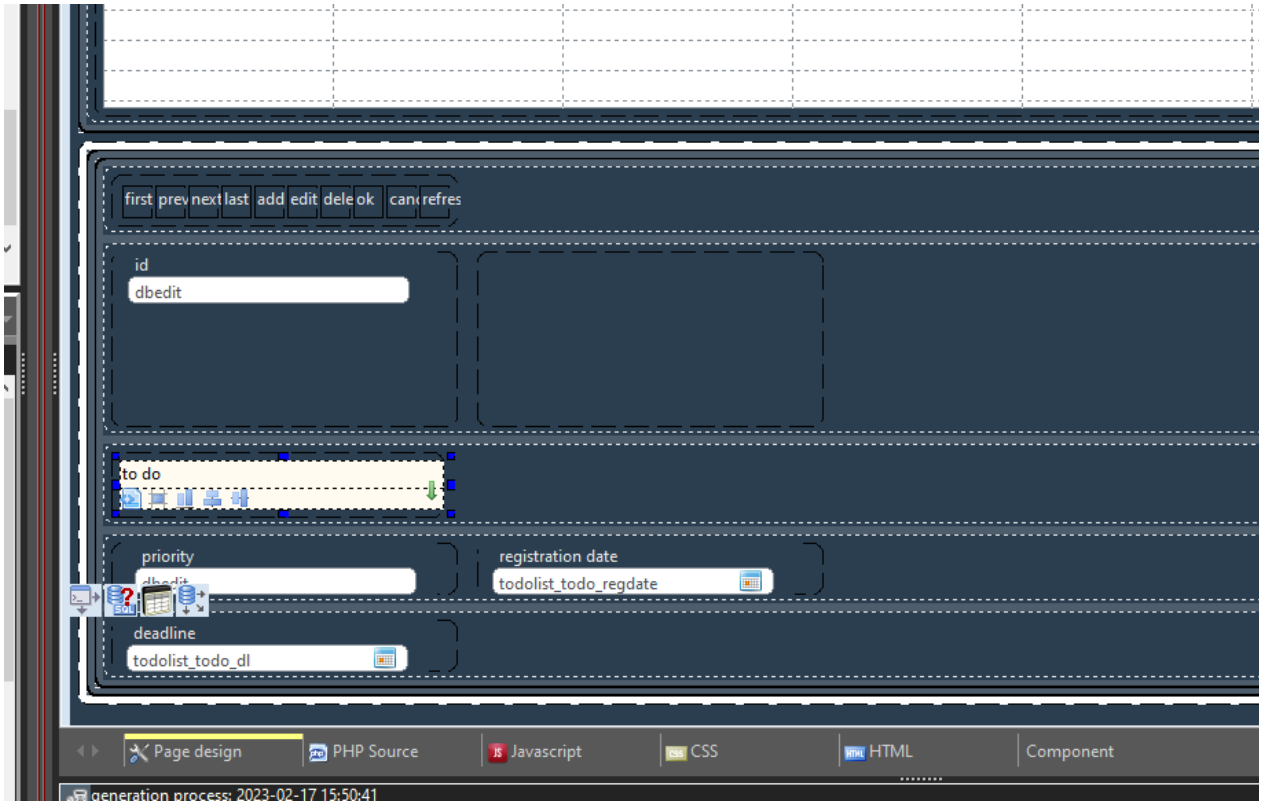
Note: the edit area of the wysiwyg editor is depending on the Bootstrap theme that you are using. This form can be improved further by expanding the todo field over the full width of the form. Select the row with the edit field. In the component tree use the right mouse click to open the dialog and select 'add new row after selected row. .



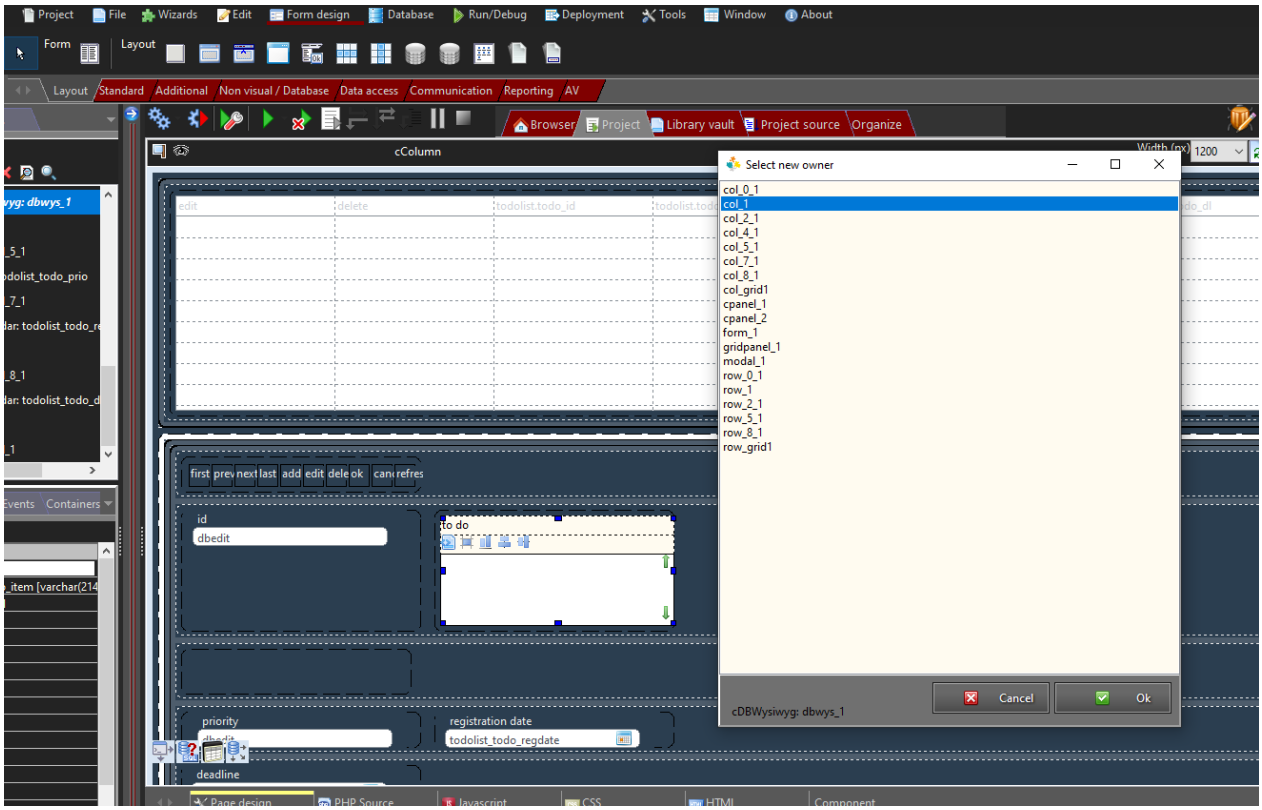
Next apply a new column in the row.



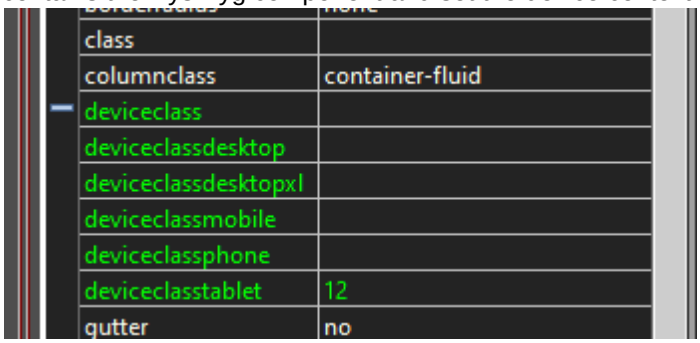
To move the wysiwyg field into the new column you can use the component tree and drag/drop the wysiwyg field into the column:



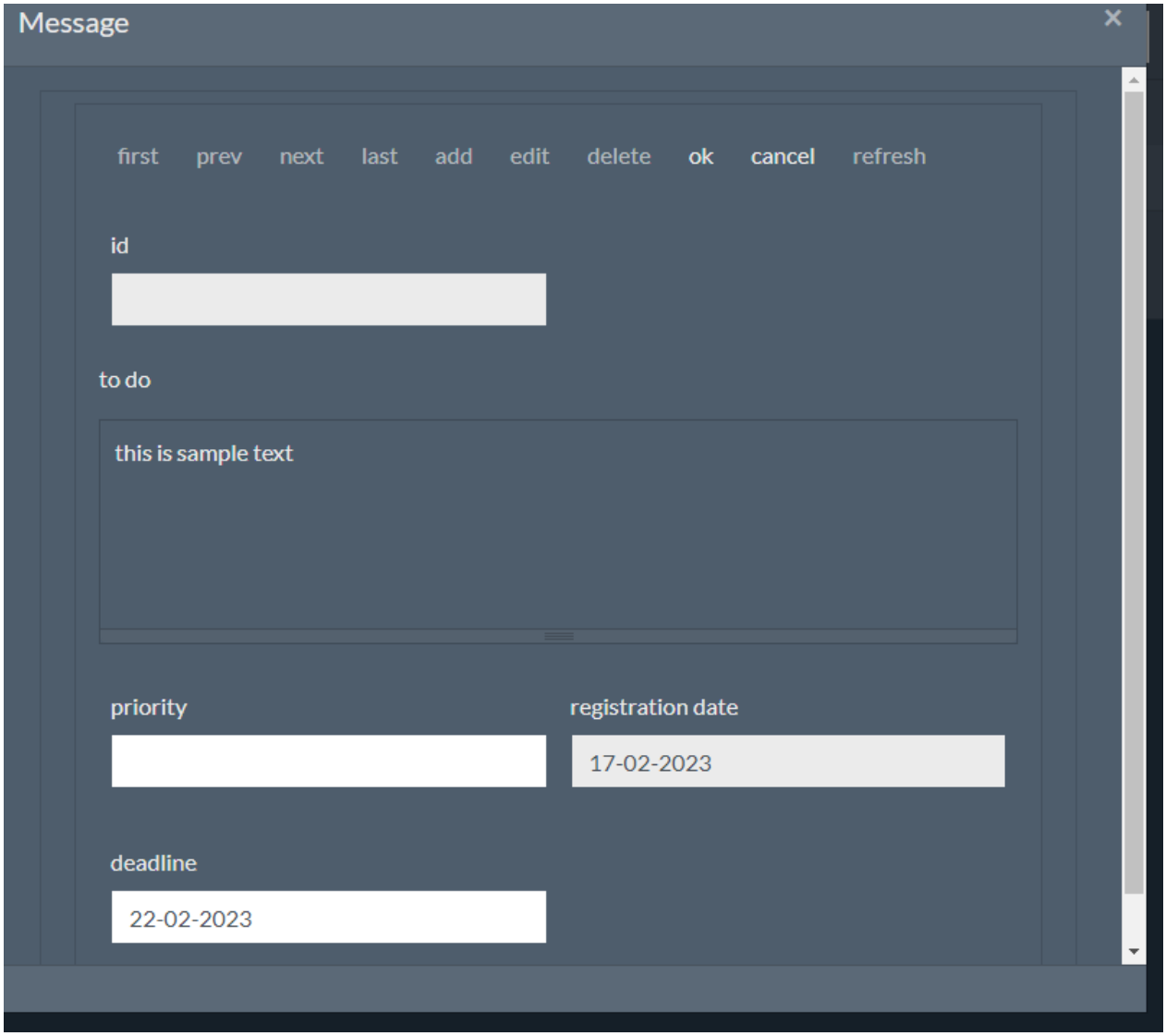
If the component tree is large due to many components on the form, then you can also change the parent of the component



The final thing to do is to specify the full width of the column in the row (col_1). Select the column that contains the wysiwyg component and set the device context to 12 (= full width).



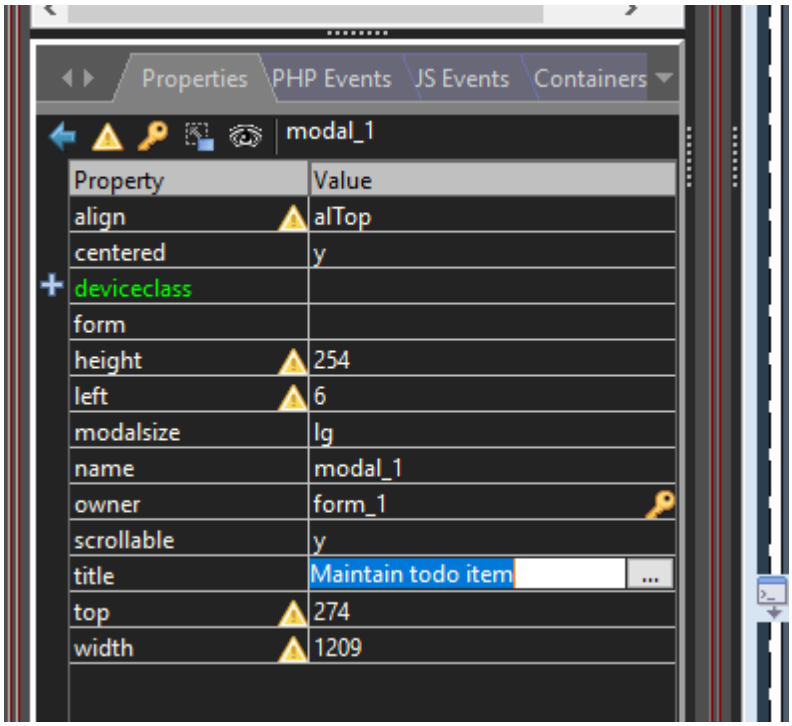
Generate and run the code, it should look something like:

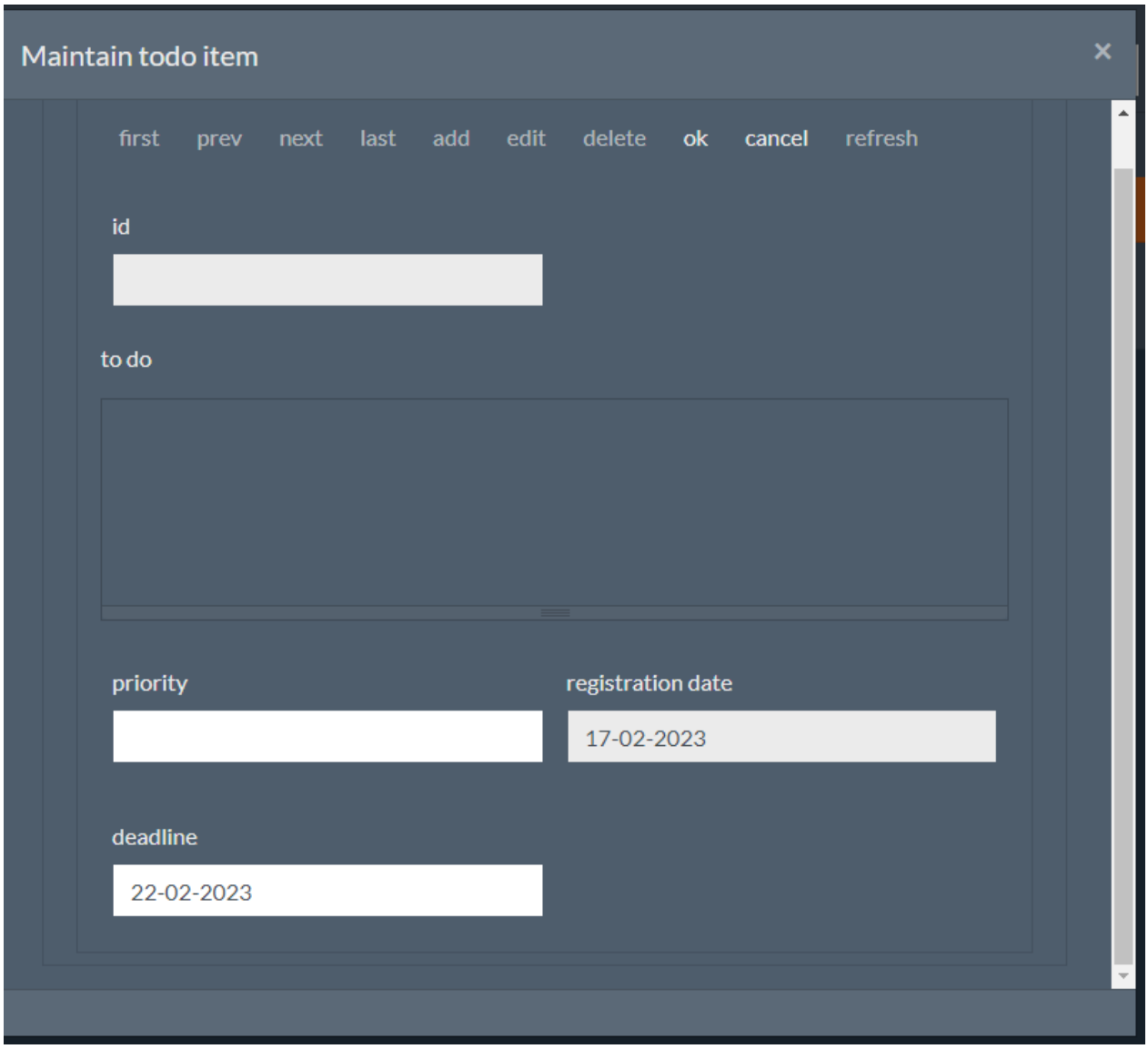


The title 'message' is too generic.

It is also possible to change the title of the popup. One option is to change it to 'maintain todo item'.

Go to the component tree and select the cModal component. Change the title property:

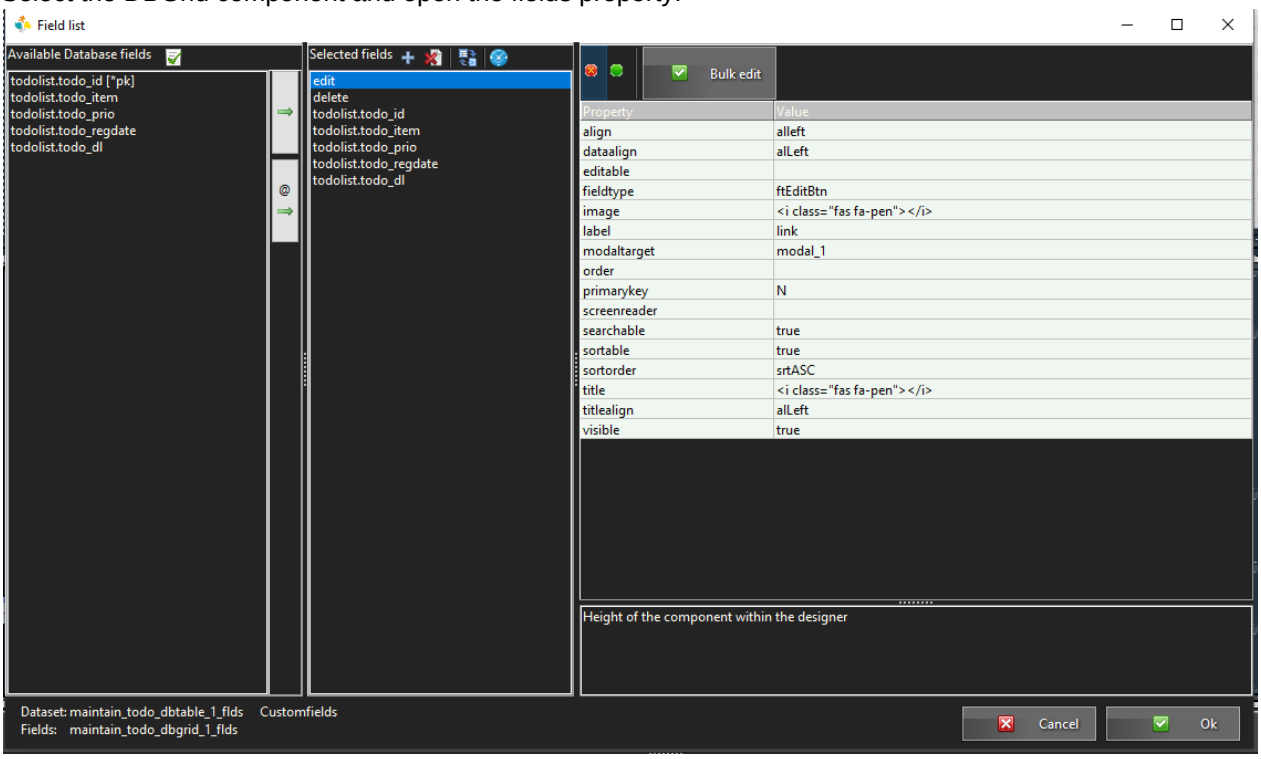

Properties
PHP Events
JS Events
Containers
modal_1
Property
Value
align
alTop
centered
y
deviceclass
form
height
254
left
6
modalsize
lg
name
modal_1
owner
form_1
scrollable
y
title
Maintain todo item
top
274
width
1209


Maintain todo item
first
prev
next
last
add
edit
delete
ok
cancel
refresh
id
to do
priority
registration date
17-02-2023
deadline
22-02-2023

The icons to edit or to delete are barely visible in this theme

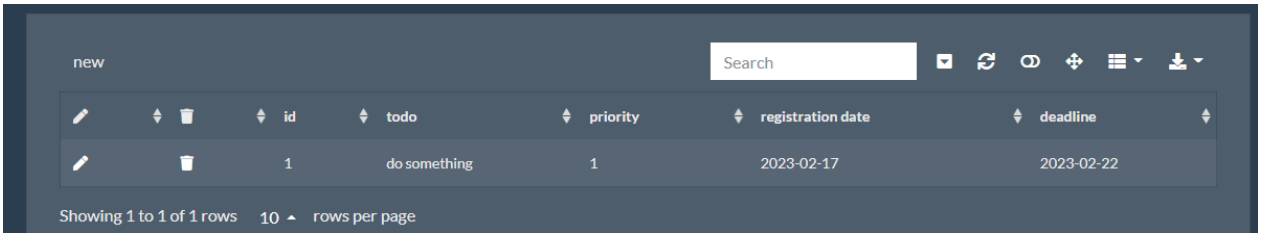
Depending on the used theme, it is possible that the default colors of the icons do not enough contrast. Fortunately, this can be changed easily.

Select the DBGrid component and open the fields property.



In the edit and delete field, you will see a link like:

Remove 'text-dark' , click ok and generate/run the application. (By using the bootstrap colors like text-primary etc, you can also change color).



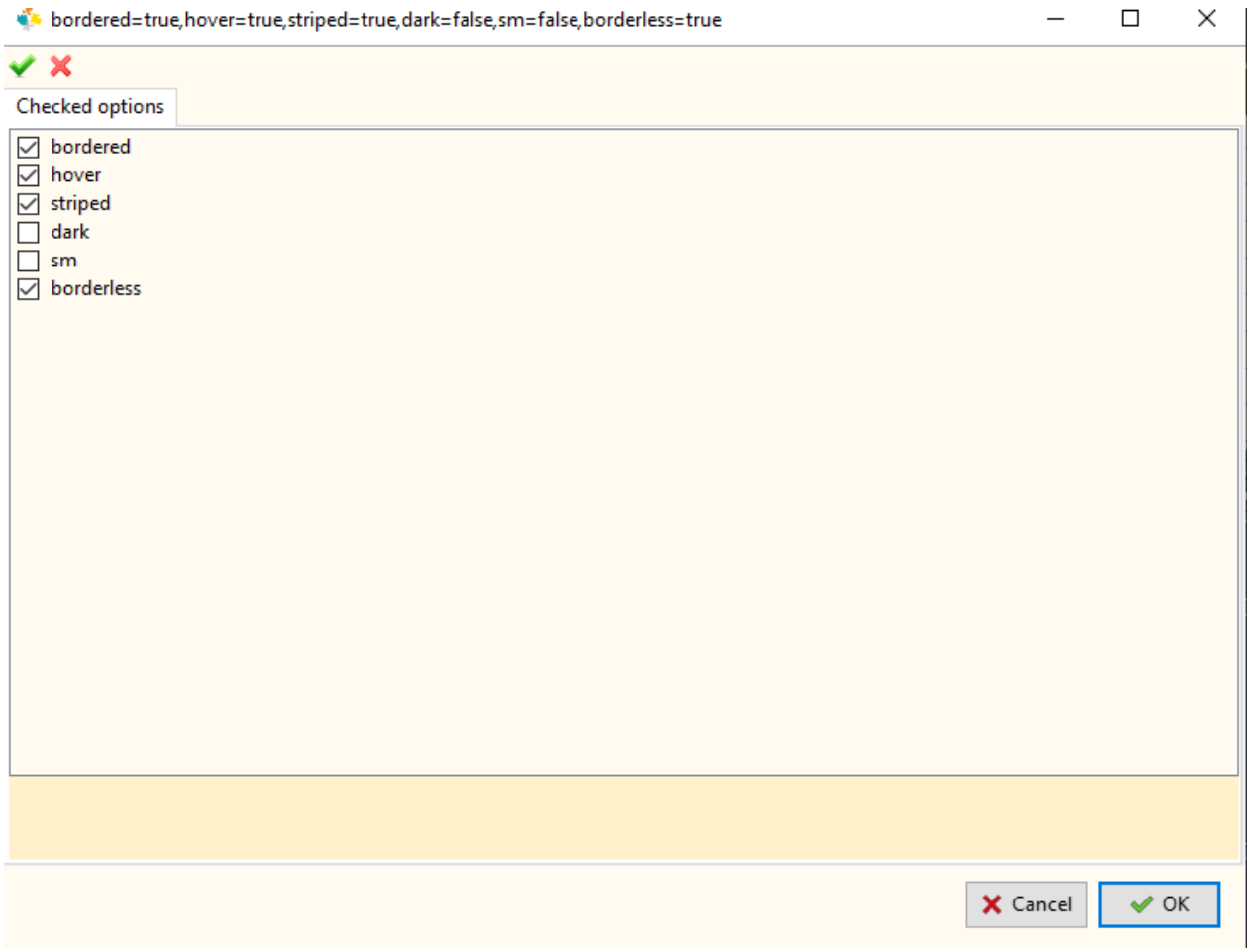
The toolbar of the grid contains many items that are not required, we'll change that.

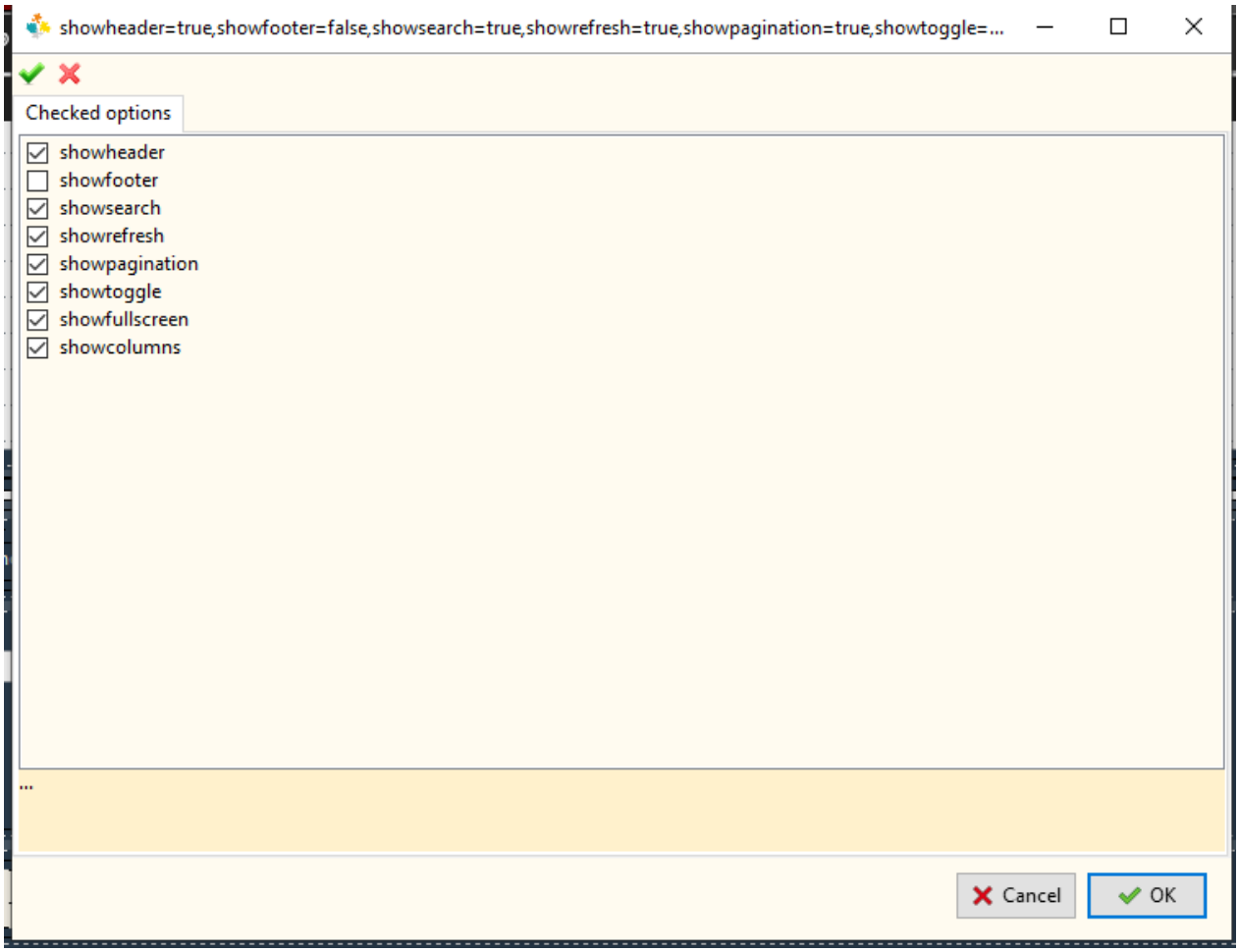
It is also possible to change the look-and-feel of the grid. One of the things you might want to change is the number of options that are not always useful:

Property ShowOptions:

You can enable/disable elements of the grid by checking/unchecking these options.

The appearance of the grid can be changed as well by setting the tableclasses property:


bordered=true,hover=true,striped=true,dark=false,sm=false,borderless=true
Checked options
bordered
hover
striped
dark
sm
borderless
Cancel
OK


showheader=true,showfooter=false,showsearch=true,showrefresh=true,showpagination=true,showtoggle=...
Checked options
showheader
showfooter
showsearch
showrefresh
showpagination
showtoggle
showfullscreen
showcolumns
...
Cancel
OK

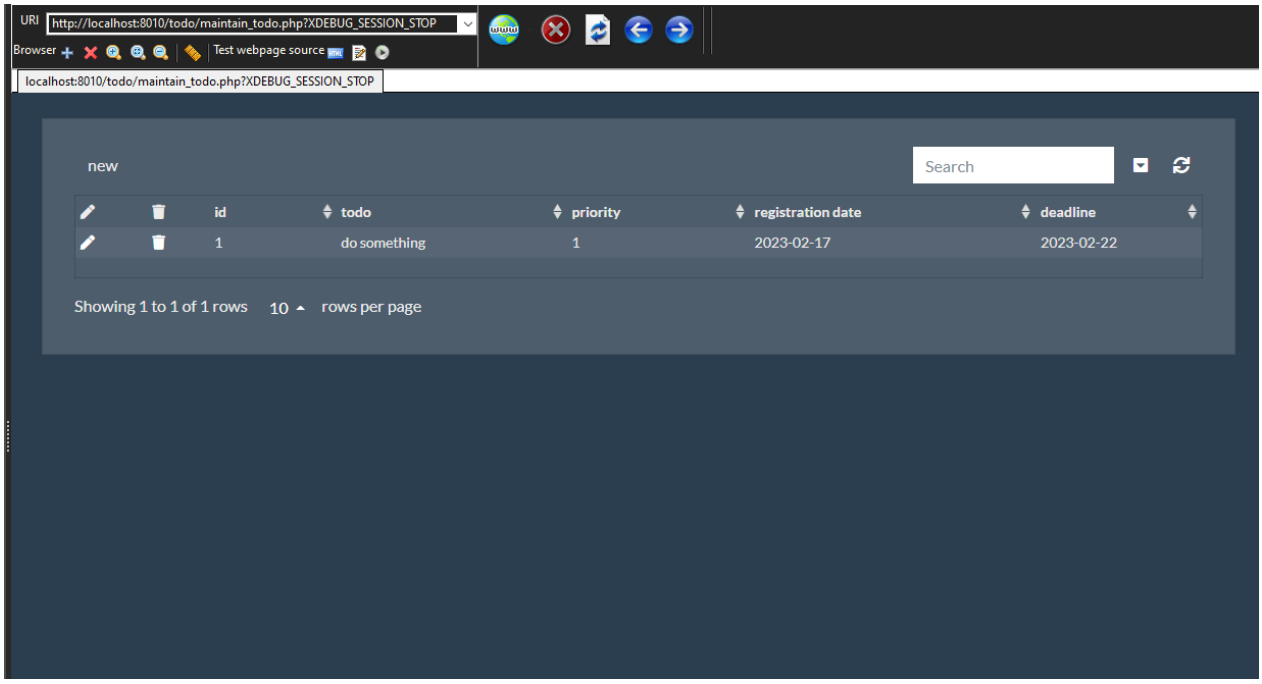
Some of the most used options are part of the component itself

ShowColums : allows the user to enable/disable columns.
ShowExport: allows the user to export data in several formats.

It is easy to find out what these properties do, by changing these and generate/run the application.

Final result

The final result might look something like this:



f

Part 2. Apply features

Soon...