

# **PHAkt Quick Reference Manual**

<b>1. INTRODUCTION:</b> .....	<b>2</b>
<b>2.PREREQUISITES :</b> .....	<b>3</b>
<b>3.PROJECT CREATION:</b> .....	<b>3</b>
<b>4.DATABASE CREATION:</b> .....	<b>3</b>
<b>5.SEARCH FUNCTION:</b> .....	<b>5</b>
<b>6.RESULTS PAGE:</b> .....	<b>6</b>
6.1. FOR THOSE IN A HURRY: .....	<b>8</b>
<b>7. INSERT FORM:</b> .....	<b>9</b>
7.1. FOR THOSE IN A HURRY: .....	<b>10</b>
<b>8.DELETE FORM:</b> .....	<b>10</b>
<b>9.UPDATE FORM:</b> .....	<b>11</b>
9.1.FOR THOSE IN A HURRY: .....	<b>12</b>
<b>10. LOGIN FORM:</b> .....	<b>13</b>
<b>11. DYNAMIC ITEMS:</b> .....	<b>14</b>
<b>12. MISCELLANEOUS:</b> .....	<b>14</b>

# 1. Introduction:

**Ultradev** offers the practical use for the user to create dynamic websites in a WISWIG interface very comfortably. Unfortunately up to the current version (4.0) the **php** support is not implemented in the main product

A Freeware open source project named php4Ud, under the direction of Dan Radigan, presented an extension for UD in early 2001, and allowed PHP development only for the Windows version of Ultradev and only for the MySQL database. Many important functions, which exist also for the ASP/JSP/CF versions, were implemented.

Due to the small public support, in co-operation with the company Interakt the project was renamed "**PHAkt**" and therefore placed under the GPL. The extension was developed further as open source user community project. On 26.03.2001 the first beta version of "PHAkt" appeared.

This extension uses the ADODB extension (Active Data Objects for generic DataBase connectivity from PHP), a database connection library for PHP, which allows connection to a lot of databases, similar with the Microsoft's ADO library. Among others, the following databases are supported: MySQL, PostgreSQL, Interbase, Oracle, Ms SQL 7, Foxpro, ACCESS, ADO, Sybase and ODBC.

As a prerequisite for the creation of dynamic websites with PHAkt, the following software is necessary:

Windows 95-2000	<a href="http://www.microsoft.com">http://www.microsoft.com</a>
MAC OS	<a href="http://www.apple.com">http://www.apple.com</a>
Dreamweaver Ultradev 4	<a href="http://www.macromedia.com">http://www.macromedia.com</a>
The UD Extension "PHAkt"	<a href="http://www.interakt.ro/products/Phakt/index.php">http://www.interakt.ro/products/Phakt/index.php</a>
Web Server e.g. Omnihhttpd	<a href="http://www.omnicron.ab.ca/httpd">http://www.omnicron.ab.ca/httpd</a>
	<a href="http://www.apache.org">http://www.apache.org</a>
Mysql database (WIN version)	<a href="http://www.mysql.com">http://www.mysql.com</a>
Myodbc database driver	<a href="http://www.mysql.com/downloads/api/myodbc.html">http://www.mysql.com/downloads/api/myodbc.html</a>
MySQL JDBC driver (MAC)	<a href="http://www.mysql.com/downloads/api-jdbc.html">http://www.mysql.com/downloads/api-jdbc.html</a> (the 2.0.4 version)
optionally phpmyadmin	<a href="http://www.phpwizard.net">http://www.phpwizard.net</a>

Various resources offer based extensions on "PHAkt", e.g. Massimos php Form analysis. (check: [http://www.udzone.com/.](http://www.udzone.com/))

Now let's start. We will create a small dynamic web application which contains a search function, a login system, a data query, an insert, a delete and an update page. The whole is arranged as small project in ten steps, so that the practice reference is not lost.

To be able to fully understand this tutorial you need in any case a certain knowledge of the used software. One should know how you can create interactive pages using Ultradev and the appropriate Server Behaviors and call these over the web server, or a rough knowledge of how a web application server and the database operate. A more detailed guidance would lead here too far and can be looked up in the Macromedia help, as most things are very similar.

At the end of this tutorial you should be able to publish your project on a public accessible server. In practice the adjustments made here are not portable, which drops back particularly to the file access rights and access rights within the database. This tutorial does not intend to deal with this issues, and further help can be found at: <http://www.mysql.com/doc/>. Security when connecting to the database is however a very important issue and should be taken seriously.

We will also not deal with the fundamental mode of operation and configuration of the database, that could be done here too far. We assume this knowledge is already available. All methodologies described starting from point 2 can be transferred to a large extent also to the work with UD & the ASP Server Model.

The current bugs in PHAkt are available in the PHAkt distribution in the TODO.txt file. One user should

always check the PHAkt bug tracker located at <http://www.interakt.ro/phakt/bt/>

## 2.Prerequisites :

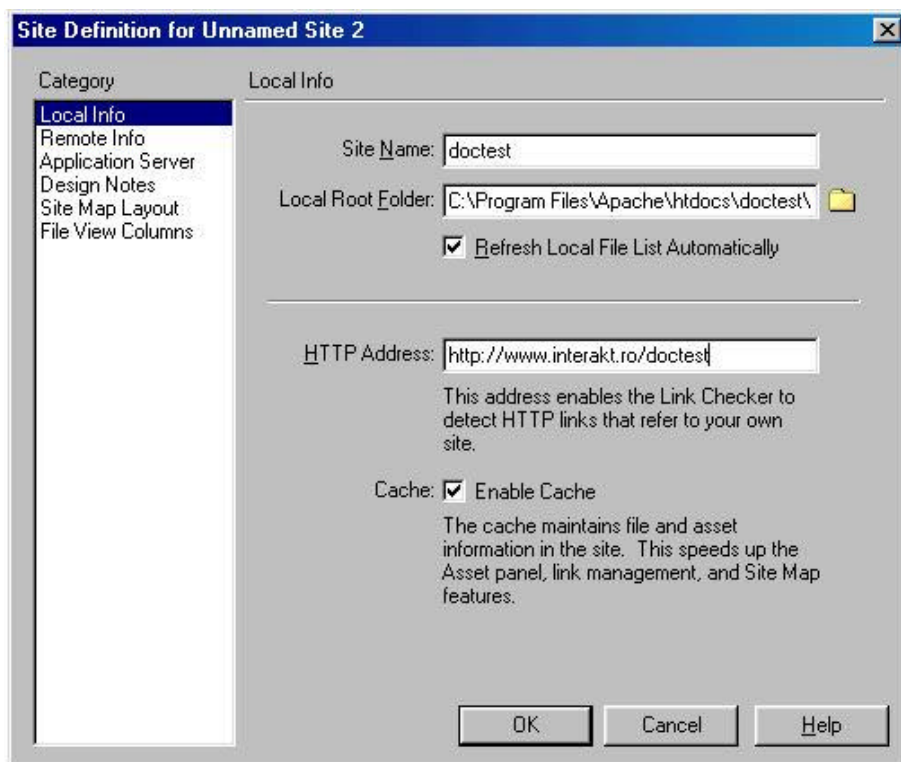
Install Macromedia Ultradev 4.0, a suitable web server with php support (we use here the very stable and fast Omnihttpd Freeware web server, other programs of this kind are available (Apache, PWS)), Myodbc, Mysql and the UD extension "PHAkt". To work with the databases and tables in MySQL we use the web tool "phpmyadmin", which has to be installed in the server root folder (htdocs).

We will supply installation guidance of the related products here, please use their provided Readme files or on-line resources.

Create a document with the name menu.php and other files named update.php, such.php, input.php, delete.php. In menu.php create links to the other files.

## 3.Project creation:

Create in the general statement of the Web Servers (htdocs) a directory with the name "myname". Create in Ultradev a new project, and we'll call it from now on "myname". In the project management of UD you indicate this as general statement, then adjust in the project management "Application server" to be PHP. Adjust also the rest of the settings.

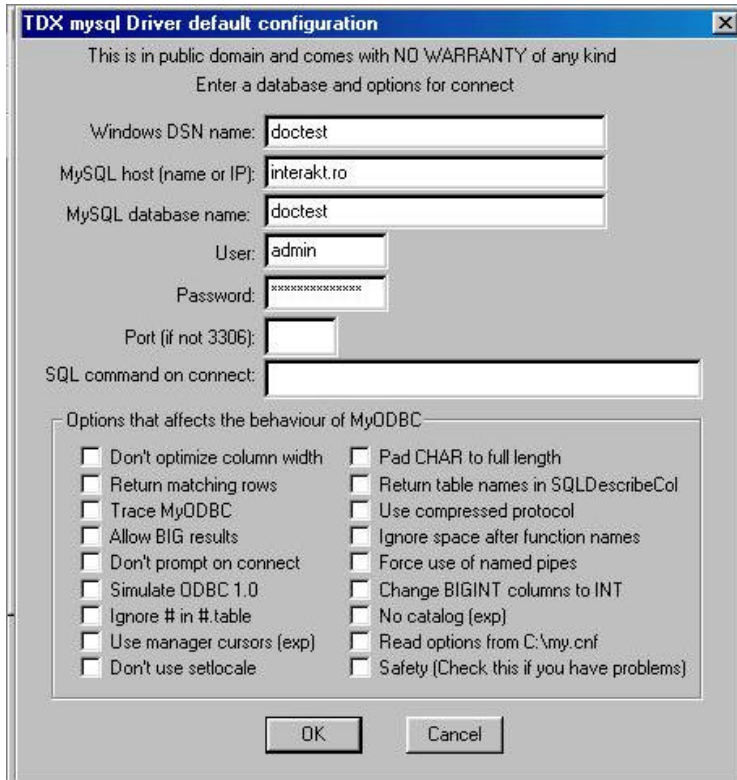


## 4.Database creation:

Start "Mysql" and load "phpmyadmin" in the browser. Create a new database with name "myname". Enter in the "Windows" Control Panel/ODBC drivers/ (control panel ODBC, or system administration ODBC with W2K) and create a new system DSN with the following specifications:

- Windows DSN Name = myname
- Mysql hostname = localhost (if you operate locally, otherwise the IP of the Web Servers, on which the

database is located)  
- Database name = myname.



"Mysql" runs by default on port 3306, so if your configuration differs you should indicate this by changing the port value in the Myodbc menu. Now we'll create the individual tables for the person data.

The first table contains simple person data, while second table serves as dynamic menu on the page, in order to enable to the user given values. The third table contains the acces data for the Login area.

Tbl\_PA: Ident (Primary key, auto\_increment), Name (Text), Surname (Text), Address (Text), Code (mediumint), Email (Text), Phone (Text), Fax (Text), No\_Children (Number), Family (Text).

Tbl\_Fam: Ident (Primary key, auto\_increment), Family (varchar(100)) with contents: single, married, divorced, separated, Radio (Tinyint) with the values 1 and 0 in each case of a line, Checkbox (Tinyint) with the values 1 and 0 in each case of a line.

Tbl\_Log: Ident (Primary keys , auto\_increment), User name (Text), Password (Text).

To easily create the database, run the following SQL in phpMyAdmin:

```
# Server version: 3.23.36
# PHP Version: 4.0.6
# Database : `doctest`
# -----
#
# Table structure for table `Tbl_Fam`
#
DROP TABLE IF EXISTS Tbl_Fam;
CREATE TABLE Tbl_Fam (
  ident bigint(20) NOT NULL auto_increment,
  family varchar(100) NOT NULL default '',
  radio tinyint(4) NOT NULL default '0',
  checkbox tinyint(4) NOT NULL default '0',
  PRIMARY KEY (ident),
  UNIQUE KEY ident (ident),
  KEY ident_2 (ident)
) TYPE=MyISAM;
# -----
```

```

#
# Table structure for table `Tbl_Log`
#

DROP TABLE IF EXISTS Tbl_Log;
CREATE TABLE Tbl_Log (
  ident bigint(20) NOT NULL auto_increment,
  username text NOT NULL,
  password text NOT NULL,
  PRIMARY KEY (ident),
  UNIQUE KEY ident (ident),
  KEY ident_2 (ident)
) TYPE=MyISAM;
# -----

#
# Table structure for table `Tbl_PA`
#

DROP TABLE IF EXISTS Tbl_PA;
CREATE TABLE Tbl_PA (
  ident bigint(20) NOT NULL auto_increment,
  name text NOT NULL,
  surname text NOT NULL,
  address text NOT NULL,
  code int(11) NOT NULL default '0',
  email text NOT NULL,
  phone text NOT NULL,
  fax text NOT NULL,
  no_children int(11) NOT NULL default '0',
  family text NOT NULL,
  PRIMARY KEY (ident),
  UNIQUE KEY ident (ident),
  KEY ident_2 (ident)
) TYPE=MyISAM;
# End

```

## 5. Search function:

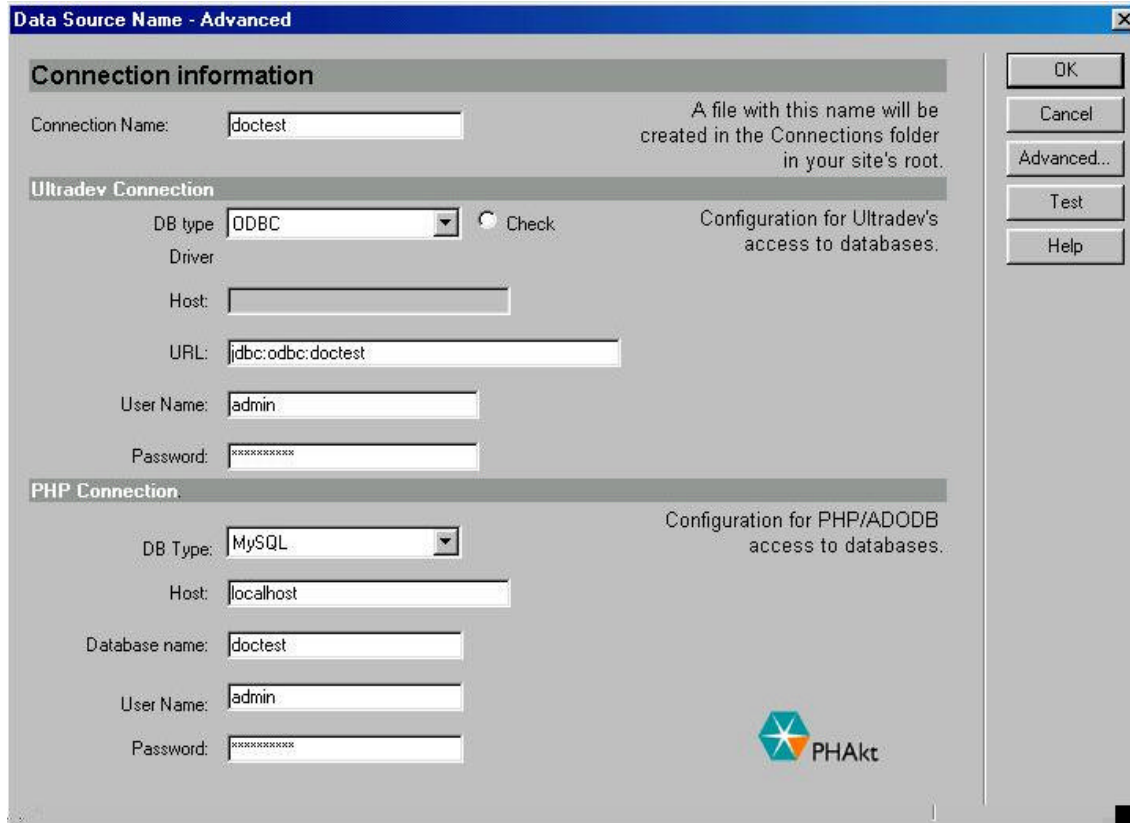
Its role is to create a lookup by name, code and address. Create now a file in UD with name "such.php" and a form containing three text input fields named "name", "code" and "address", and also a Submit and a RESET Button. Make the form action is "list.php" and its method is "POST".

The image shows a web form and its configuration in a design tool. The form has three text input fields labeled "Name", "Code", and "Address", and two buttons labeled "Submit" and "Reset". The design tool shows the configuration for the "name" field, including "Char\_width" and "Max Chars" settings.

## 6.Results page:

A result page with the appropriate filtering is now to be created. Create a new document with the name "list.php". Create in the menu "Data Bindings" a connection by clicking on the "Define Connection" button. In the "PHAkt" Pop UP windows input :

- Connection name: Name of the connection e.g. "myconname"
- Database type: Give the type of the database e.g. ODBC
- Host: Indicate the host name: if you operate locally then put "localhost "
- URL: the URL of the database e.g. jdbc:odbc:myname
- Database type: The type of the database e.g. Mysql
- Host: the host names of the Web Servers e.g. local host (if local)
- Database name: the name of the database e.g. myname



Test the connection by clicking on the "Test" button to check if the database binding is working properly.

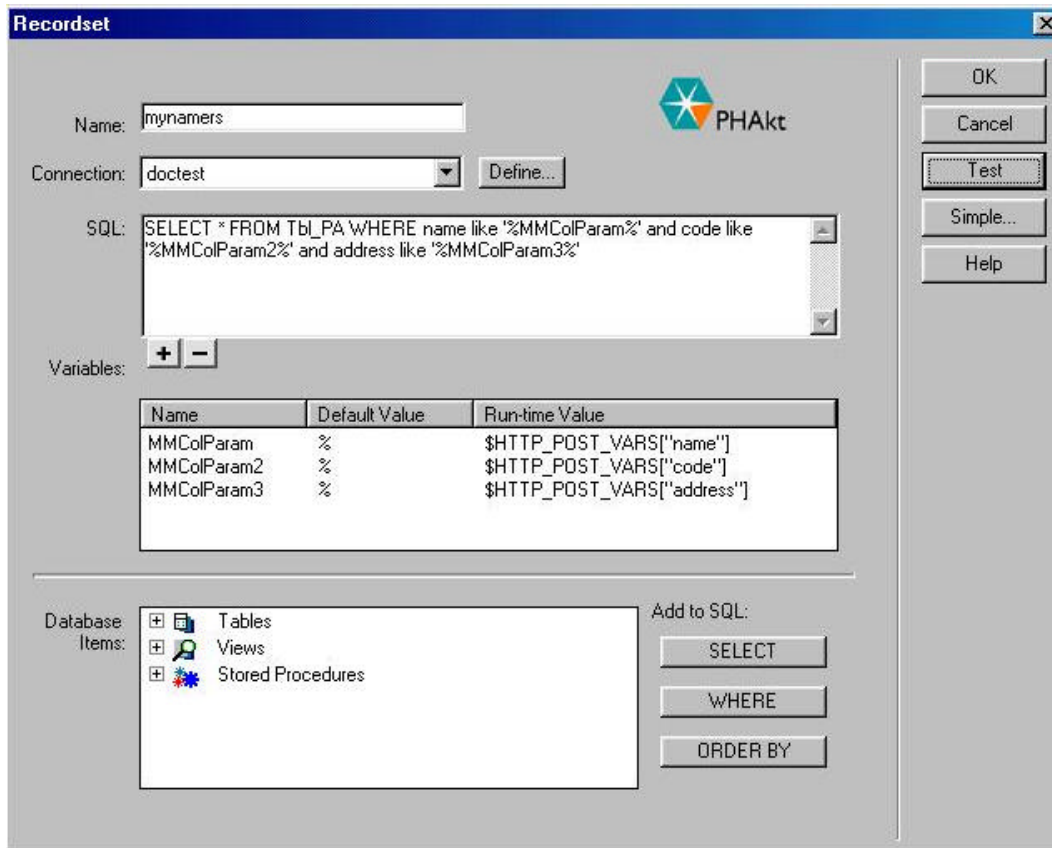
In the menu "Data Bindings" create a new recordset. As name for the recordset indicate "mynamers". As connection select "myconname".

Select the table Tbl\_PA now and click on the Button "Advanced", where a advanced SQL statement is to be created for filtering the fields. E.g. input the following into the field SQL:

```
SELECT * FROM Tbl_PA WHERE name like '%MMColParam%' and code like '%MMColParam2%' and address like '%MMColParam3%'
```

Into the field variables input:

Name	Default VALUE	Run-Time VALUE
MMColParam %	%	\$HTTP_POST_VARS["name"]
MMColParam2 %	%	\$HTTP_POST_VARS["code"]
MMColParam3 %	%	\$HTTP_POST_VARS["address"]



With this SQL statement the desired filtered resultset is obtained. The described syntax defines the variable transfer from a form. The result of the filtering depends, of course, on your personal conditions.

### Tip:

If you want to use another variables as filters, use the “Simple” mode for creating the recordsets, and use the “Filter” inputs to customise your filter: the field to be filtered. In the next input, the operator to be applied to the expression (=,<,>, etc), the type of variable used in the expression (URL GET variable, Form variable, cookie, etc.) and finally the name of the variable to be used.

In the document "list.php" create now a table with two lines (one for the headings and one for the data) and ten columns. From the “Data Bindings” menu, from the recordset you’ve created, drag-and-drop the fields of the recordset in each of the cells corresponding to the second line of the table you’ve created, and write in the cell above the name of the field. Save the file.

Test the filtering by using the “such.php” file, but only after you’ve inserted some data in the table. Notice that only the first line of the recordset is displayed, and not all the lines that match the filter.

In order to view all the records, of a certain number of records, create a "Repeat Region". Select the second line of the table then from the menu “Server Behaviours” add the behaviour “Repeat Region”. Insert the number of records to be displayed, then click “OK”. Test again the filtering. Notice that the specified number of records is now shown (or all the records, depending what you’ve selected in the “Repeat Region” dialog-box) are now shown.

With the "Move to Record" Server Behaviour you have the possibility to navigate in the filtered data records. Click on the “Move to Previous Record”, and select the recordset you’ve created then click “OK”. A “Previous” link will appear. Do the same for the “Move to Next Record”. A “Next” link will appear. With the " Show Region " Server Behaviour you have the possibility, of displaying the “Previous” and “Next” buttons depending on the presence of data. For example, the “Previous” button must be shown only if the current record is not the first record, and the “Next” button must be shown only if the current record is not the last record. To add this Server Behaviour, simply select the desired area then click in the "Server Behaviors" menu on " Show Region" and select the appropriate condition.



**Tip:**

In the menu " Objects->Live " you'll see a menu option "Insert Recordset Navigation Bar PHP", that automatically creates a navigation bar. Simply indicate there the appropriate recordset and determine whether it is to create links or use images. The buttons "First, Previous, etc are created. If you use images, the pictures are created automatically. However, the default ones can be replaced by others.

In order to display further details, create first a file named "detail.php" with appropriate dynamic contents (see above). Then select one link in the document "list.php" (for example the ident fields from the second line) and in the menu "Server Behaviours" the option "Go To Detail PAGE ". You'll find there transfer parameters similar to an asp binding. Select the target, the recordset, the column of the recordset to be used, and the modality of passing the existing parameters: as URL or as Form. For a very good reason the column should contain values which uniquely identify the selected record (generally a key used as index, an ID). On the detail page another filtering of the appropriate column must take place now, whose values are transferred just like above. In our case a simple filtering is sufficient. Select there also the type of the transfer (in this case " URL GET Paramter ").

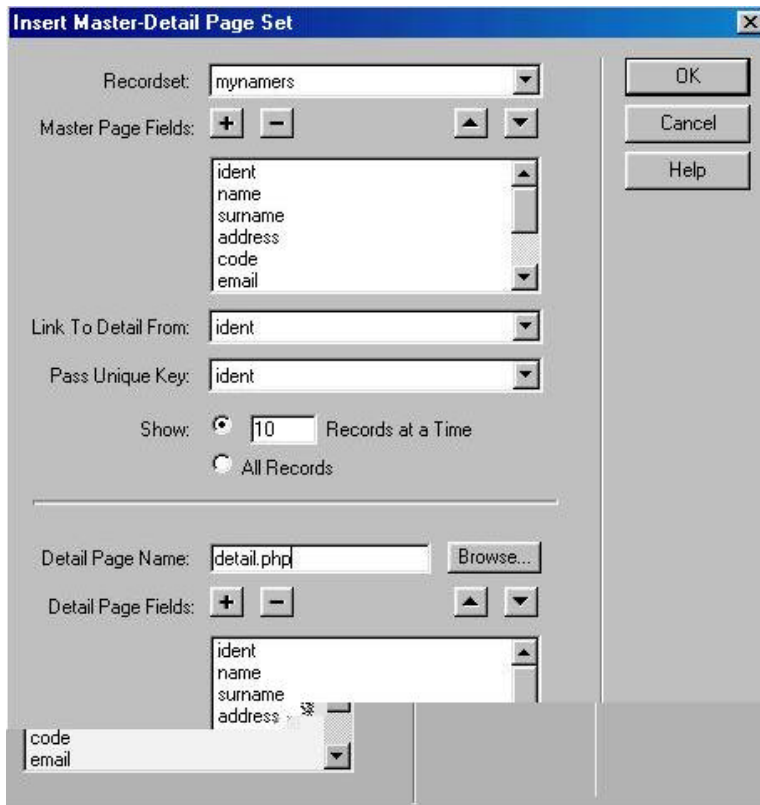
### **6.1. For those in a hurry:**

There is a function to create a result page including a navigation border and a detail page very fast. The basic condition is that the document must include at least one recordset. If this condition is satisfied, click in the menu "Objects->Live" on "Insert Master-Detail Page Set PHP". It opens a form, into which you must input the following:

- Recordset: Indicate the record set to use.
- Master Page Fields: Determine by adding and removing fields by clicking on the " + " and " - " buttons, which fields of the recordset are to be displayed in the Master Page (the list page).
- Link to Detail From: Field used for linking the Master and the Detail Page
- Pass Unique Key: A unique value identifying the record (generally an ID, a unique key)
- Show: Number of records to be displayed in the Mater Page's list. (all or a specified number)
- Detail Page Name: Path to the Detail Page
- Detail Page Fields: Determine by adding and removing fields by clicking on the " + " and " - " buttons, which fields of the recordset are to be displayed in the Detail Page.

After clicking on "OK" the appropriate Master & Detail pages are generated.





## 7. Insert Form:

Create a new document named "input.php". If you add form text input fields, designate their names exactly the same as the column headings of the database. Do not forget to add the Submit and Reset buttons. Insert at the end a List/Menu named "Fam". Create two new recordsets "mynamers" with contents of the table "Tbl\_PA" and "mynamers2" with contents of the table "Tbl\_Fam". Use the connection "myname" already created.

The inserted List/Menu "Fam" serves pre-allocating certain form contents. Click on the List/Menu "Fam" you've inserted. In the menu "Server Behaviors" "Dynamic Elements" click "Dynamic List/Menu". Select the recordset "mynamers2", to display data from the "tbl\_fam". Select the field you want to represent the data displayed in the List/Menu, and the field you want to designate the values actually to be inserted (Label and Value).

Click now in the "Server Behaviors" menu on "Insert Record". A Pop UP window in that appears where you have to make the following specifications:

- Connection: the connection to the database.
- Insert Into Table: Sets into which table the data is to be stored.
- After Inserting Go to: Sets the page to be displayed after the insert is made.
- Get Values From: Gives the name of the form containing the data to be inserted.
- Form Elements: Sets the form elements that are to be used in the insert.
- Column: Sets the column in which the corresponding form element is stored.
- Submit as: Sets the type in which data from the corresponding form element is stored (text, date, number, etc.).

In the menu "Server Behaviors" click on "User Authentication" "Check New User Name", that checks if the new entry already exists in the table.

Test the page you've just created by calling the document over the web server or the menu and input some data. Call the data list and check whether the insert took place.

## 7.1. For those in a hurry:

There is a function to create an Insert Form very fast. Basic condition is the document includes at least a recordset. If this condition is satisfied, click in the menu "Objects->Live" on "Insert Record Insertion Form PHP". It opens a form, into which you must input the following:

- Connection: Indicates the connection, e.g. myname
- Insert Into Table: Indicate the database table into which the new data is to be inserted
- After Inserting Go to: Indicate a page, which is to be opened, after the record data was inserted
- Form Fields: Determine by adding and removing fields by clicking on the " + " and " - " buttons, which fields of the recordset are to be contained in the form.
- Label: Indicates the name of the corresponding field.
- Display As: Sets the format in which the data is to be displayed.
- Submit As: Sets the format in which the data is stored in the table.
- Default Value: Sets a default value for the corresponding field. This value can be static, or originate from a database.

After clicking on "OK" the input form is generated.

Column	Label	Display As	Submit As
phone	Phone:	Text Field	Text
fax	Fax:	Text Field	Text
no_chil...	No_children:	Text Field	Numeric
family	Family:	Menu	Text

## 8.Delete Form:

According to the same principle as the Input Form , a Delete Form can also be produced. Create a new document named "delete.php". If you add form text input fields, set their names exactly the same as the column headings of the database. Do not forget to add the Submit and Reset Buttons. Create a new recordset "mynamers", using the already created connection "myname". Instead of text input fields you can also use of course Lists/Menus or Radio Buttons. This depends on the handled data.

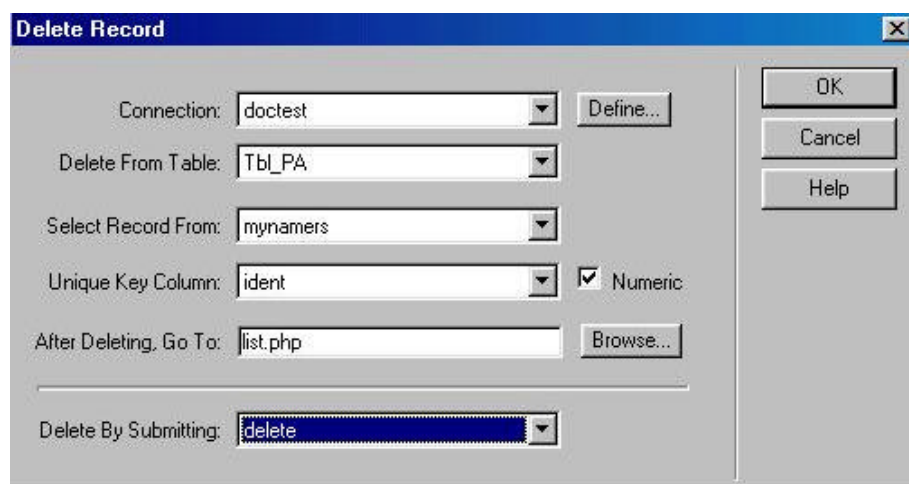
In order to display the data to be deleted, the form fields must contain the appropriate script codes. Draw now by dragging & dropping the names from the menu "Data Binding" into the appropriate form field

Create, as already described above, a "Previous" and a "Next" button, so that you can also navigate in the individual data records. With the "Show Region" options you have the possibility to display the navigation buttons depending on presence of data just as described above in section 5.

Tip: See Above(###\_link: ###)

Click now in the "Server Behaviors" menu on "Delete Record". A Pop UP window in that appears where you have to make the following specifications:

- Connection: Give here the connection to the database e.g. "myname".
- Delete From Table: Sets the table from which the data is to be deleted.
- Select Record From: Set the recordset which stores the data displays and that is to be deleted e.g. mynamers.
- Unique Key Column: Set the column that designates a unique key for the records in the table, and also indicate id this values is numeric.
- After Inserting Go to: Sets the page to be displayed after the delete is made.
- Delete By Submitting: Sets the form that must be submitted for the delete.



Test your page by calling the document over the web server or the menu. After clicking on " Submit ", the displayed data record should be deleted. Also call the data list and check whether the deletion really took place.

## 9.Update Form:

According to the same principle as the Delete Form, a Update Form can also be produced. Create a new document named "update.php". If you add form text input fields, set their names exactly the same as the column headings of the database. Do not forget to add the Submit and Reset Buttons. Insert at the end another List/Menu field named "Fam". Create two new recordsets "mynamers" with contents of the table "Tbl\_PA" and "mynamers2" with contents of the table "Tbl\_Fam", using the already created connection "myname".

In order to display the updated data, the form fields must contain the appropriate script codes. Draw now by dragging & dropping the names from the menu "Data Bindings" of the corresponding recordset into the appropriate form field. The inserted cunning menu "Fam" serves pre-allocating certain form contents. In the menu "Server Behaviors" -> " Dynamic Elements" click on "Dynamic List/Menu". Select the recordset "mynamers2", set the data to be used as labels and values as described before.

Create, as already described above, a "Previous" and a "Next" button, so that you can also navigate in the individual data records. With the "Show Region" options you have the possibility to display the navigation buttons depending on the presence of data to be displayed next (display the "Previous" button if you're not on the first recordset and display the "Next" button if you're not on the last).

Tip: See Above(###\_link: ###)

Click now in the "Server Behaviors" menu on "Update Record". A Pop UP window will appear where you have to input the following:

- Connection: Specifies the connection to the database
- Table to Update: Sets into which table the updates will be made.
- Select Record From: Sets the record that designates the fields to be updated.

- Unique Key Column: Sets the fields designating a unique key for the records in the table (generally a ID, a key)
- After Updating Go To: Sets the page to be displayed after the update.
- Get Values From: Sets the name of the form containing the new values of the fields to be updated.
- Form Elements, Column, Submit as: Sets for each value from the form what field in the database updates and in what format the values will be submitted (as text, number, date, etc).

Test the page by calling the document over the web server. Modify some data. After clicking on the "Submit" button the data should be updated. Check if the update really took place by calling the data list.

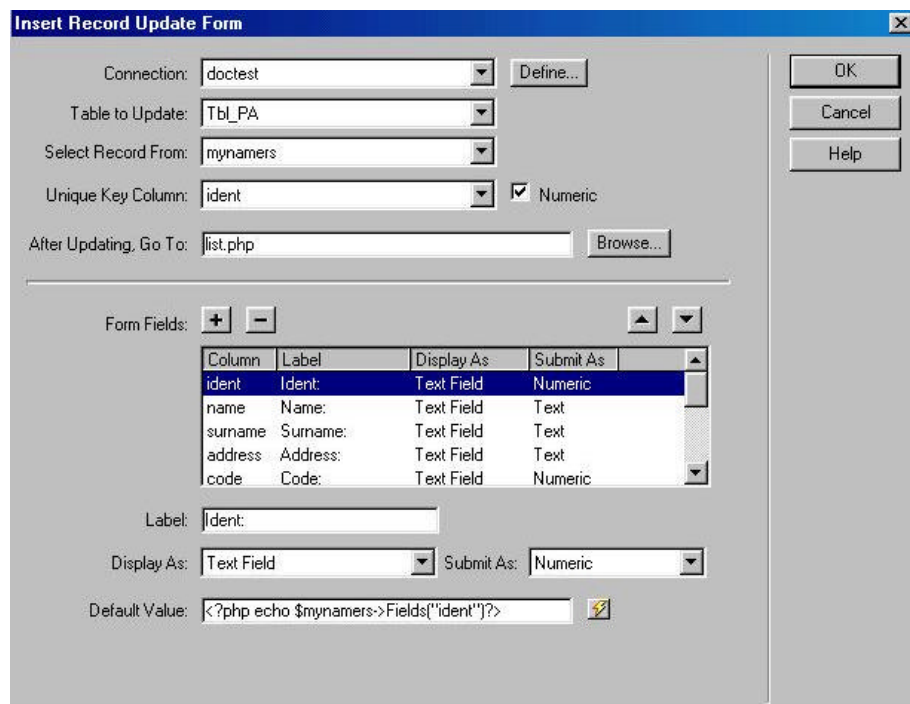
If a field should be displayed but you are prevented to make modifications on its content, then there is a possibility that the form field has a "readonly" property. Simply delete the property "readonly" of the field from the property list of the input.

## 9.1. For those in a hurry:

There is a function to create a form for update very fast. Basic condition is that the document must include at least one recordset. If this condition is satisfied, click in the menu "Objects->Live" on "Insert Record Update Form PHP". It opens a form, into which you must input the following:

- Connection: Indicates the connection, e.g. myname
- Table to Update: Indicates the database table, whose data should be updated.
- Select Record From: Indicate the recordset of the data which can be updated.
- Unique Key Column: Indicates a field that uniquely identifies a row in a table (generally an unique ID, a key)
- After Updating Go To: Indicates a page, which is to be opened, after the data is updated
- Form Fields: Determine by adding and removing fields by clicking on the " + " & " - " buttons, which fields of the recordset are to be contained and updated by the form.
- Label: Sets the names (labels) of the corresponding field.
- Display As: Sets the format in which the data is to be displayed.
- Submit As: Sets the format in which the data is to be stored in the table.
- Default Value: Sets a default value for the corresponding field . This value can be static, or originate from a database.

After clicking on "OK" the update form is generated..



Connection: doctest Define...

Table to Update: Tbl\_PA

Select Record From: mynamers

Unique Key Column: ident  Numeric

After Updating, Go To: list.php Browse...

Form Fields: + -

Column	Label	Display As	Submit As
ident	Ident:	Text Field	Numeric
name	Name:	Text Field	Text
surname	Surname:	Text Field	Text
address	Address:	Text Field	Text
code	Code:	Text Field	Numeric

Label: ident:

Display As: Text Field Submit As: Numeric

Default Value: <?php echo \$mynamers->Fields('ident')?>

OK Cancel Help

## 10. Login Form:

Up to the current version 1.0.2 the Authentication functions are completely implemented,

With "PHAkt" you have the possibility of executing a database based Login. Create a new document named "failed.php", by indicating to the user that its log in attempt failed. Create a new document named "login.php". Add in a Form two text inputs, and set their names to "username" and "password". The password input field can be defined also as password field, so that no plain text display takes place.

Create a new recordset "mynamers", using the already created connection "myname". Select the table "Tbl\_Log". Click now in the menu "Server Behaviors" on "User Authentication" -> "Log In User". A Pop UP window appears, into which you enter the following data:

- Get Input From Form: Name of the form used for log in (see above)
- Username Field: The name of the field for the user name ("username")
- Password Field: The name of the field for the password ("password")
- Validate Using Connection: Name of the connection to the database e.g. mynamers
- Table: The table with the acces data in the database e.g. Tbl\_Log
- Username Column: Column with entries of the user names in the database ("user name")
- Password Column: Column with entries of the passwords in the database ("password")
- If Log In Succeeds, Go To: Page to be opened if the Login attempt runs successfully (e.g. "menu.php")
- If Log In Fails, Go To: Page to be opened if the Login attempt fails (e.g."failed.php")
- Restrict Access Based On: Indicates here whether the Login attempt is to be checked on the basis by user names and password, or whether the Login is to be based additionally on a group affiliation (here it is however necessary to create in the database an additional column in which the appropriate groups the individual Users are assigned, and indicate this column in UD).

For creating a logout link, select a link, click in the menu on "Server Behaviors" --> "User Authentication" -> "Logout User PHP". A dialog-box will open where you specify wether the logout takes place by clicking on a link or on page load, and the page to be displayed after the logout (generally the login page).

Log In User

Get Input From Form: login

Username Field: username

Password Field: password

Validate Using Connection: doctest

Table: Tbl\_Log

Username Column: username

Password Column: username

User Id Column: ident

If Log In Succeeds, Go To: menu.php

Go To Previous URL (if it exists)

If Log In Fails, Go To: failed.php

Restrict Access Based On:

Username and Password

Username, Password, and Access Level

Get Level From: ident

OK

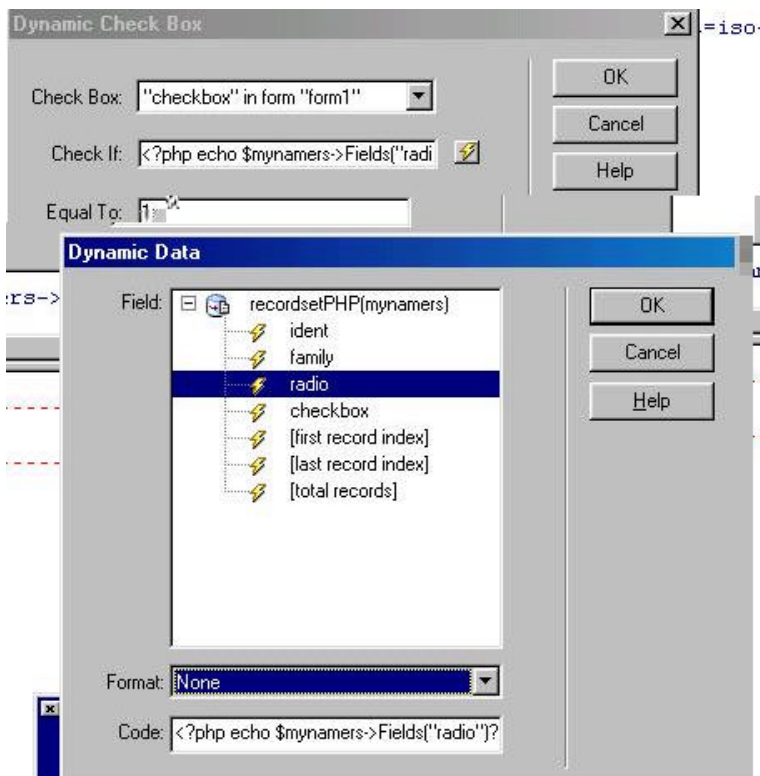
Cancel

Help

## 11. Dynamic items:

This page is independent of the other pages already created and is created only to briefly represent the mode of operation of dynamic form elements. Create a new document named "dynamic.php". Add a RadioButton named "radio" and a CheckBox named "checkbox". Add a recordset named "mynamers", as source for the inputs, select again the well-known connection "myname". The recordset is to contain record from the table "Tbl\_Fam".

Select the CheckBox and click in the "Server Behaviors" menu on "Dynamic Elements" - "Dynamic Check Box". A dialog will appear, where you set the name of the CheckBox to be transformed, the condition for the CheckBox to be checked ("Check If" ... "Equals To" condition). If you click of the "lightning" button, you can easily select the field from the recordset that enters in the condition. If you have more CheckBoxes, repeat the procedure for each of them.



The procedure for the RadioButton is quite similar, and it will not be discussed further. Thus you can very easily create Dynamic RadioButtons, CheckBoxes, depending on the values from a database.

## 12. Miscellaneous:

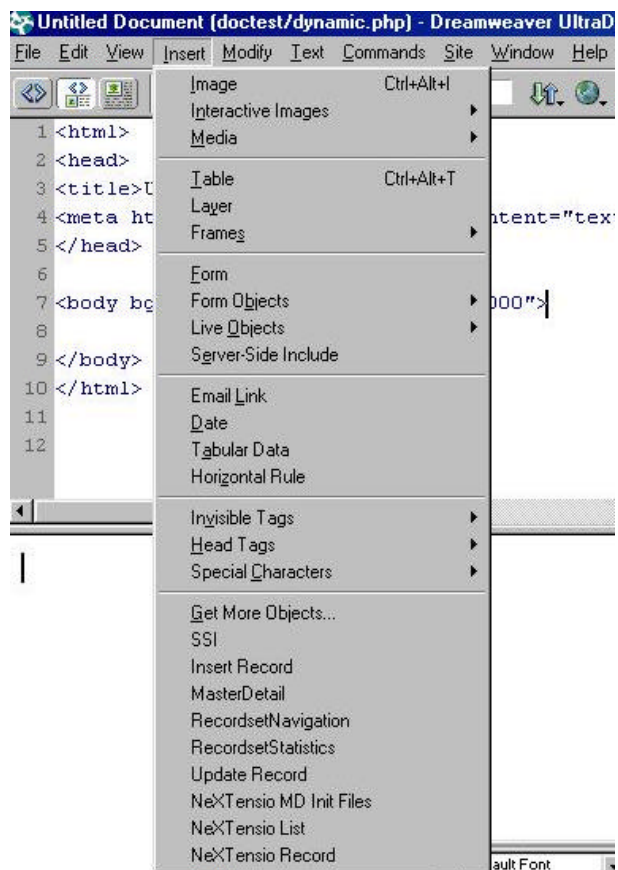
"PHAkt" offers some very useful auxiliary functions.

In the menu "Servers Behaviors" you see a menu option "No Cache", option that prevents the storage of data in the browser's cache. At this point simply click on it to integrate it into the page, no other adjustments are necessary.

In the menu "Insert" you see a menu option "Live-Objects->Recordset Navigation Status PHP", displays the status of the recordset records displayed (e.g. records 1-5 from 10). Simply indicate the recordset used.

In the menu "Insert" you see a menu option "Server-Side Include". With the instruction Require there can be selected contents of an other/external page and merged into the current page. Simply click on the menu option and indicate in the Pop UP window the file name or the path to the file to be included.





This completes our tutorial.

Remember, PHAkt is our common project and we wellcome your suggestions for code and documentation improvements.

Don't forget to visit PHAkt's site for updates : <http://www.interakt.ro/products/PHAkt/>