



**Interakt**  
PROFESSIONAL WEB TOOLS

**PHAkt 2 MX  
Quick Reference  
Manual**

version 2.8.1

## Table of Contents

<b>Introduction.....</b>	<b>3</b>
<b>Overview.....</b>	<b>3</b>
<b>Prerequisites.....</b>	<b>3</b>
<b>Requirements.....</b>	<b>3</b>
<b>Typographic Conventions.....</b>	<b>4</b>
<b>Some Specifications.....</b>	<b>4</b>
<b>Installations.....</b>	<b>4</b>
<b>Configuring the site in Dreamweaver MX.....</b>	<b>4</b>
<b>Creating the Database.....</b>	<b>5</b>
<b>Search function.....</b>	<b>6</b>
<b>Results page.....</b>	<b>7</b>
<b>Best practice.....</b>	<b>11</b>
<b>Insert Form.....</b>	<b>12</b>
<b>Best practice.....</b>	<b>14</b>
<b>Delete Record.....</b>	<b>15</b>
<b>Update Form.....</b>	<b>16</b>
<b>Login Form.....</b>	<b>18</b>
<b>Dynamic items.....</b>	<b>19</b>
<b>Miscellaneous.....</b>	<b>21</b>

## Introduction

One of InterAKT's most successful tools is PHAkt, a GPL Server Model for Dreamweaver MX, allowing visual PHP development.

Continuing our efforts for PHAkt1 (PHP support in Dreamweaver Ultradev 4), we have created the only alternative Server Model for Dreamweaver MX, PHAkt2 MX. It allows database driven website development with ease, adding new server behaviors over the original PHP\_MySQL server model and allowing connection to multiple databases.

The main advantages of "PHAkt 2" over "PHP\_MySQL" are:

- Support for multiple databases in a transparent way
- Support for MySQL, PostgreSQL and MsSQL date locales
- User authentication Server Behaviors
- Master/Detail Server Behaviors
- Go to detail Page Server behaviors
- Improved code generation (all reusable code is kept in a functions.inc.php file)
- Multiple and extensible recordset support.
- IIS and Apache 2 support by using absolute HTTP redirects.

PHAkt2 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.

## Overview

This tutorial will offer you the guidelines in order to 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. All is presented as a small project in ten steps, to help you learn something useful. The web application will manipulate an employee list, and you will find the finished application in the attached archive.

## Prerequisites

### *Requirements*

This tutorial requires basic knowledge of Macromedia Dreamweaver MX development practices.

To use PHAkt2, you will have to install the following software programs:

Windows 95-2000	<a href="http://www.microsoft.com">http://www.microsoft.com</a>
or	
MAC OS	<a href="http://www.apple.com">http://www.apple.com</a>
Macromedia Dreamweaver MX	<a href="http://www.macromedia.com/">http://www.macromedia.com/</a>
PHAkt2	<a href="http://www.interaktonline.com/products/PHAkt/">http://www.interaktonline.com/products/PHAkt/</a>
Web server with PHP support	<a href="http://www.php.net">http://www.php.net</a>
PostgreSQL	<a href="http://www.postgresql.com/">http://www.postgresql.com/</a>
phpPgAdmin	<a href="http://phppgadmin.sourceforge.net/">http://phppgadmin.sourceforge.net/</a>
MySQL database (ATS version)	<a href="http://www.mysql.com">http://www.mysql.com</a>
optionally phpmyadmin	<a href="http://www.phpwizard.net">http://www.phpwizard.net</a>



You can also use other webservers like IIS, PWS or OmniHTTPd, as for the PHP version we recommend PHP 4.0.6 and above.

We have set up a complex discussion groups for **PHAkt2** related problems at <http://www.interaktonline.com/products/bbs/?prodId=0>

## Typographic Conventions

The notations and text formats used in this tutorial are found below :

- database name will be displayed using a bold, monospaced font: "**phakt**"
- database tables will be displayed using an italic font: "*employees\_emp*"
- database fields will be displayed using a bold, italic font "***id\_emp***"
- site page : monospaced, italic "*search.php*"
- recordset name : underlined italic "*recordset*"
- application button, menu or panel : bold font "**Button**"

## Some Specifications

To be able to fully understand this tutorial you need in any case a certain knowledge of the software we use. One should know how to create dynamic web pages using Dreamweaver MX and the appropriate Server Behaviors, or a rough knowledge of how a web application server and the database operate. A more detailed guidance would make this tutorial way too complicated, but you can always check out the Dreamweaver help, as PHAkt2's GUI is very similar to the PHP\_MySQL's one.

At the end of this tutorial you should be able to publish your project on a public site. Practically, the adjustments made here are not portable, and you might encounter problems especially with the file and database access rights. This tutorial will not deal with this issues, but further help can be found at: <http://www.mysql.com/doc/>. Database connection security is however a very important issue and should be taken seriously.

We will also not deal with the database fundamental mode of operation and configuration, 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 Dreamweaver MX & the ASP Server Model.

All the files generated in this tutorial are present in the tutorial.zip file that is located in the same archive as this document. The database creation script is also available in the archive, to ease the database creation. However, you will have to configure your site and connection to suit your real configuration.

The current bugs in PHAkt2 are available online at "PHAkt bug tracker" located at : <http://www.interaktonline.com/products/bt/>

## Installations

Install Macromedia Dreamweaver MX, a suitable web server with PHP 4.0.3p11 minimum support (we use here the very stable and fast Apache web server, other programs of this kind are available (IIS, OmniHttps)), Mysql and the Dreamweaver extension "**PHAkt2**". To manage the MySQL database we use the "phpmyadmin" web tool, which has to be installed in the server root folder (htdocs).

We will not supply installation guidance of the related products here, please use their provided Readme files or on-line resources.

First of all, you have to start with a correctly configured site. Name your site *phakt*.

## Configuring the site in Dreamweaver MX

Create in the general folder of the Web Servers (htdocs) a directory with the name "phakt". Create in Dreamweaver a new project, and we'll call it from now on "phakt".

To enable PHP support, please select the Server Model to PHP\_ADODB (in the Testing server section).

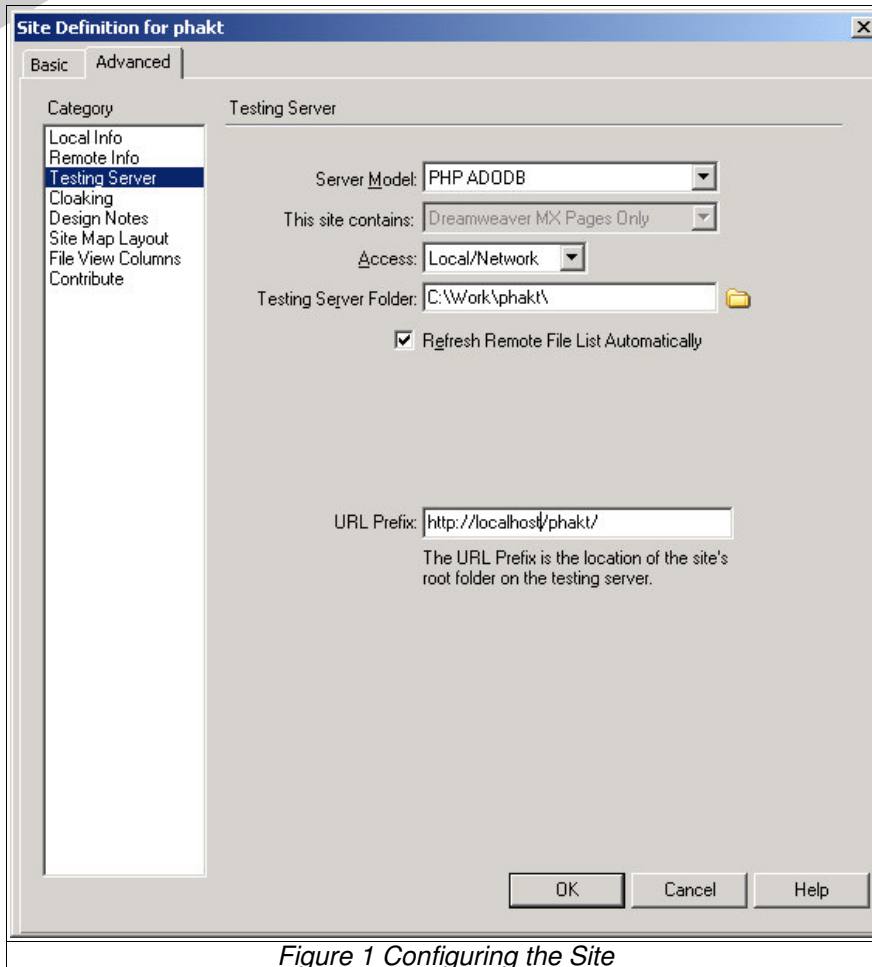


Figure 1 Configuring the Site

You also have to correctly configure the rest of the sections in the site administration wizard.

Now create 6 empty files in the site root : *menu.php*, *update.php*, *search.php*, *list.php*, *input.php*, *delete.php*.

In the *menu.php* file, create links to the other files. We'll use this menu to easily point to a page we've just created and want to test (you could skip the link to *delete.php* file – you have nothing to see there).

Save all the pages and upload on the server the entire site.

## Creating the Database

Make sure that the MySQL server is started then load "phpMyAdmin" in the browser. Create a new database named **phakt**.

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

*employees\_emp*: *id\_emp* (primary key, auto\_increment), *firstname\_emp* (varchar(100)), *lastname\_emp* (varchar(100)), *address\_emp* (Text), *code\_emp* (int), *email\_emp* (varchar(100)), *phon\_emp* (varchar(40)), *fax\_emp* (varchar(40)), *childreno\_emp* (int), *marital\_emp* (int).

*marital\_mar*: *id\_mar* (primary key, auto\_increment), *status\_mar* (varchar(100)) that will contain: single, married, divorced, separated, *radio\_mar* (tinyint) with the values 1 and 0, and *checkbox\_mar* (tinyint) with the values 1 and 0.

*users\_usr*: *id\_usr* (primary key, auto\_increment), *username\_usr* (varchar(16)), *password\_usr* (varchar(16)).

Please find below a database visual representation (generated with our commercial product QuB).

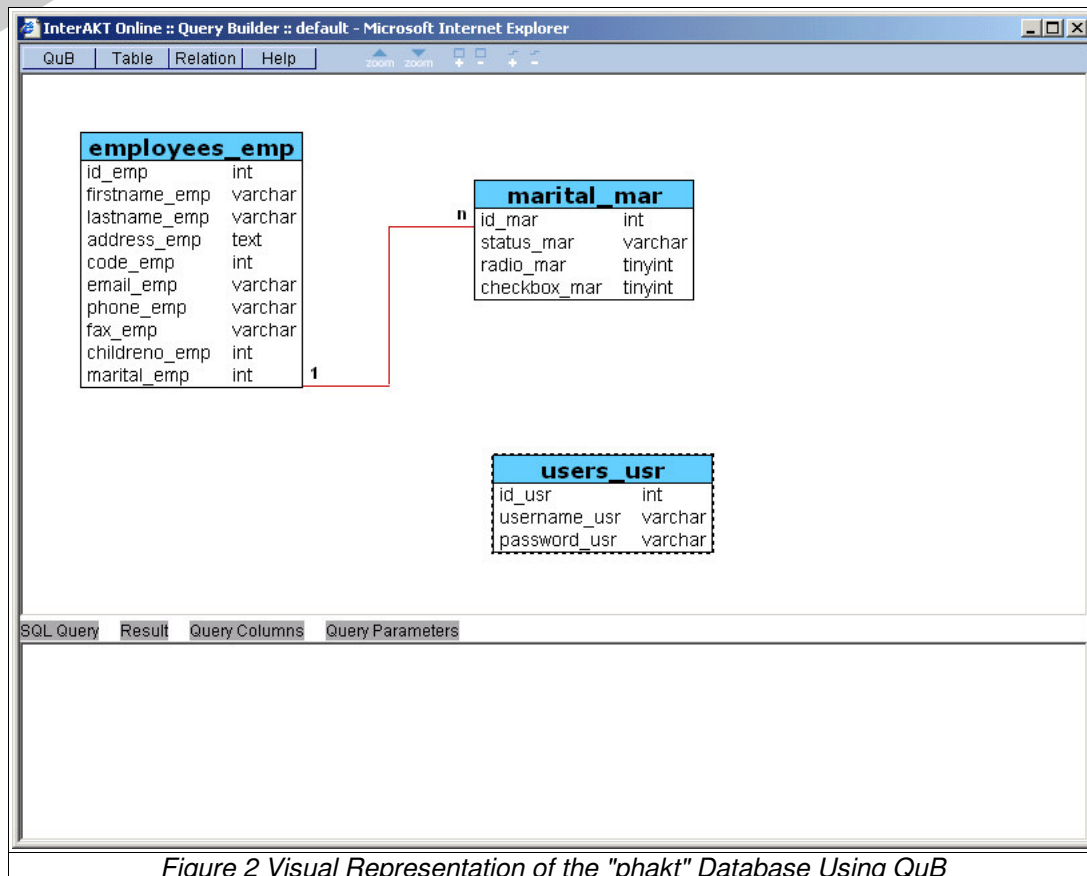


Figure 2 Visual Representation of the "phakt" Database Using QuB

We use a naming convention to be able to manage the database fields with ease, and this consists in adding to each table name an underscore and then a short name (3 letters), that will be also added in the fields name. As you can see from the picture, all employees have a marital status, so we have created a foreign key to the `marital_mar` table.

To easily create the database, run the `create_db.sql` script from the tutorial folder in phpMyAdmin.

## Search function

We will start by creating a search function, that will have as input an employee name, code and address and which will send this information to `list.php`. Its role is to create a lookup by name, code and address.

Open the `search.php` file and create a form containing three text input fields named "`name`", "`code`" and "`address`", and also a **Submit** and a **RESET** button. Make sure the form action is `list.php` and its method is "GET".

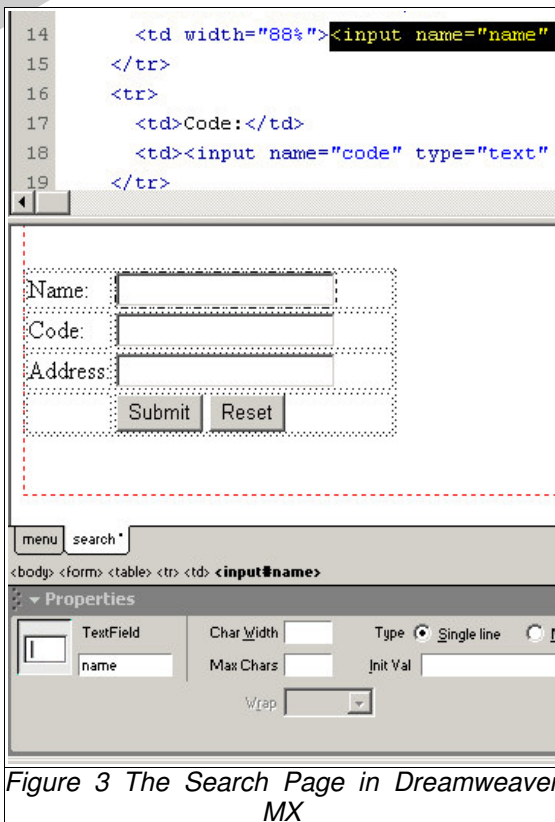


Figure 3 The Search Page in Dreamweaver MX

## Results page

We have to create now a search result page, that will filter the employees table after the sent parameters. Open the "list.php" file. We will create a connection in the first place, that will instruct the generated pages to connect to the database. In the Application panel, the Databases tab, click on the "+" sign then create a new **ADODB Connection**. Fill the form as described below:

- **Connection name**: Name of the connection e.g. "phaktconn"- Connection type: Type of the connection (permanent connection or normal connection)
- **Database type**: Give the type of the database e.g. MySQL
- **Database server**: Indicate the database host name: if you operate locally then put "localhost "
- **User name**: the name of the user that is allowed to connect to the databases
- **Password** : the user's password
- **Database name**: the name of the database is **phakt** (note that after filling in all the textfields correctly, you can click on the **Select** button and select the database name from the list with the name of all databases).

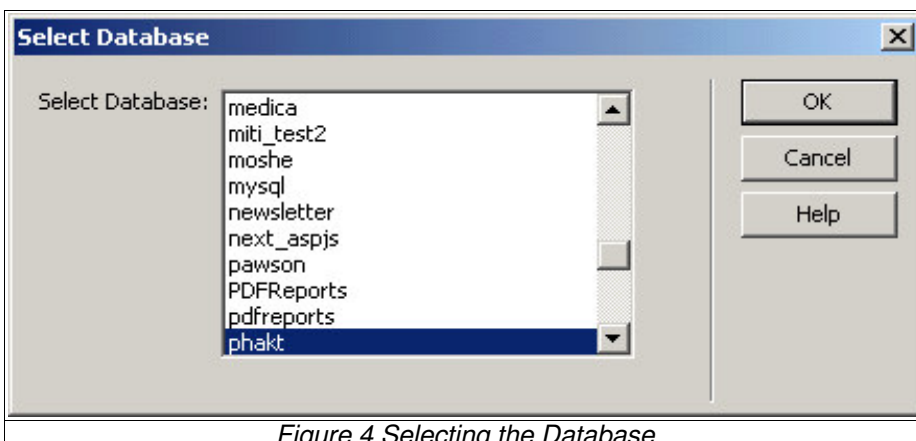


Figure 4 Selecting the Database

- **Date locale:** the format of the locale date (English, French, etc.).  
This option brings the use of formatted date display to recordsets: in Insert/Update forms the date input can now be made according to your locale settings (e.g. "01-19-2002" for English, "19-01-2002" for French). If the database server doesn't accept local dates, this functionality is emulated by PHAkt through the use of regular expressions.
- **Messages locale:** The language of some displayed messages
- **Connection type :** this will make the generated code connect to the database using Pconnect or Connect (Pconnect is the best connection type, but it's not supported by all Internet Service Providers)

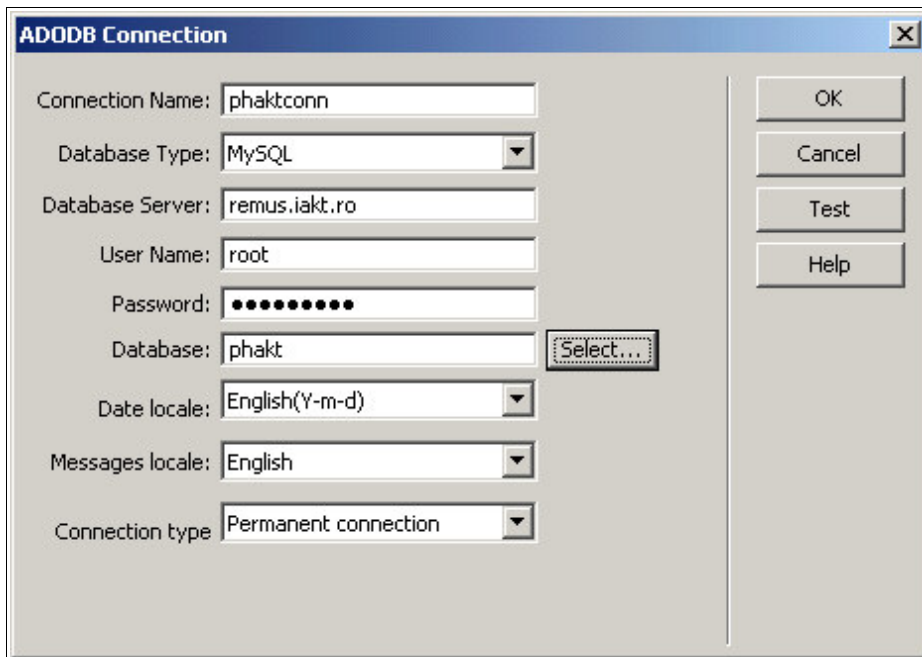


Figure 5 Configuring an ADODB connection

Test the connection by clicking on the **Test** button to check if the database binding is working properly. Then click **OK**, and some new folders will be created in your site root. They are "Connections" (the connection files are kept here), "adodb" (the ADOdb related files are stored here) and "includes" (where it is kept the file that contains all reusable PHAkt2 functions, *functions.inc.php*).

You are in the *list.php* file. In the **Application** panel, **Binding** tab, create a new **Recordset**. As name for the recordset indicate "*rsEmployees*". As connection select "phaktconn".

In the recordset name right side there is an additional menu that allows you to select the recordset subtype: Safe enables a "better" error handling of the recordset through the use of `KT_error` variable (a catching of the recordset errors) for Insert/Update actions. If "**Standard**" is chosen, the default PHAkt 2 recordset is used.

Select the table `employees_emp` now and click on the **Advanced** button, to be able to create an advanced SQL statement to filter the fields. Fill in the SQL textarea with the following SQL statement:

```
SELECT * FROM employees_emp WHERE firstname_emp like '%MMColParam1%' and code_emp like '%MMColParam2%' and address_emp like '%MMColParam3%'
```

Fill in the field Variables as shown below:

Name	Default VALUE	Run-Time VALUE
MMColParam1	' % '	<code>\$_GET["name"]</code>
MMColParam2	' % '	<code>\$_GET["code"]</code>
MMColParam3	' % '	<code>\$_GET["address"]</code>

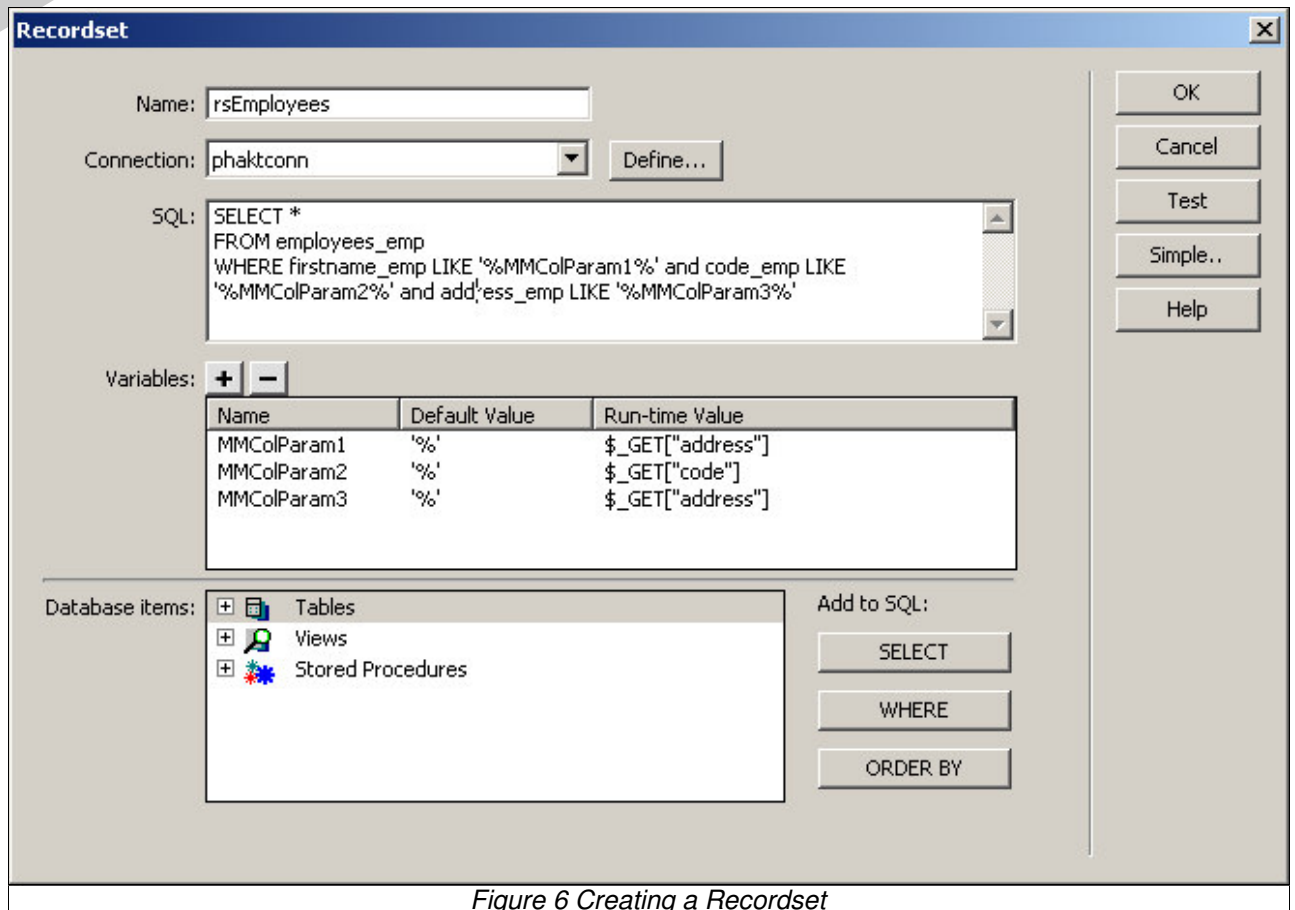


Figure 6 Creating a Recordset

The described syntax defines the SQL variables obtained from a form. The result of the filtering depends, of course, on your personal conditions written in the HTML form.

**Tip:** If you want to use another variables as filters, use the "Simple" mode for creating recordsets, and use the "Filter" inputs to customize 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.

Let's continue with the recordset result display. In the "*list.php*" file create now a table with two rows (one for the headings and one for the information) and twelve columns. From the **Bindings** tab, expand the recordset you've just created, drag-and-drop the recordset fields in each of the cells from the second line of the table, and fill in the cell above with the name of the field. In the last two columns of the second row write "Update" and "Delete". We'll transform these two cells in links to the corresponding pages in the following sections. Save the file.

Test the filtering by loading in a browser the *search.php* file, but only after you have inserted some records in the *employees\_emp* table. Note 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 that match a specific filter, with a maximum number of records, we have to create a "**Repeat Region**". Select the second row of the table (this can be easily done by a single click in the left side of the row) then from the "**Server Behaviors**" tab add the "**Repeat Region**" behavior. Insert the number of records to be displayed (e.g. 10), then click "**OK**". Test again the filtering. Notice that the specified number of records is now shown (or all the records, depending of what you've selected in the "**Repeat Region**" dialog-box).

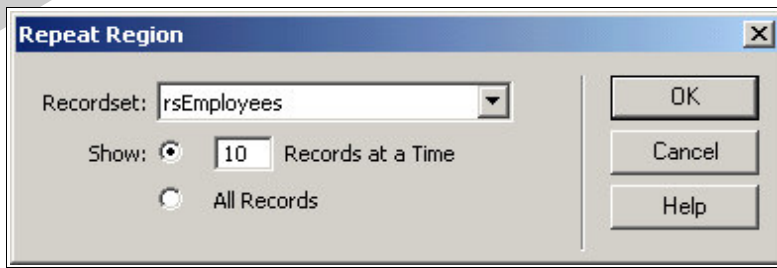


Figure 7 Configuring the Repeat Region

To be able to navigate between the recordset records (if there are more than 10 records), we'll create a navigation bar. With the **"Recordset Paging"** Server Behaviors you have the possibility to navigate through the filtered records. Click on the **"Recordset Paging/Move to Previous Page"**, 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.

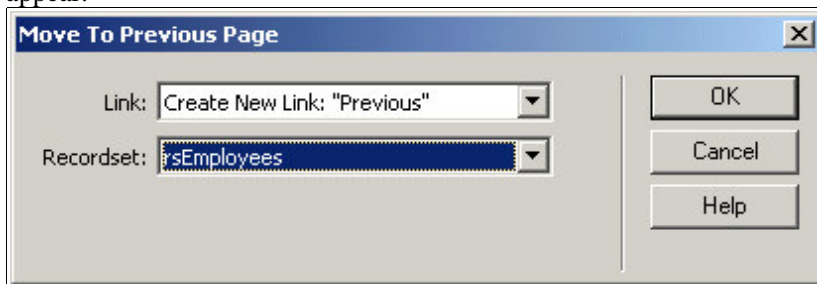


Figure 8 Adding a Move to Previous Page Server Behavior

Because the "Previous" and "Next" links do not have to be shown when there are no supplemental records to show, we'll use the **"Show Region"** Server Behaviors to display the "Previous" and "Next" links depending on a real need. For example, the **"Previous"** button must be shown only if the current Page is not the first page, and the **"Next"** button must be shown only if there are other record pages to be shown. To add those Server Behaviors over the Previous and Next links, simply select the desired area then click in the **"Server Behaviors"** menu on **"Show Region"** submenu and select the appropriate condition (**"Show if not First Page"** and **"Show if not Last Page"**).

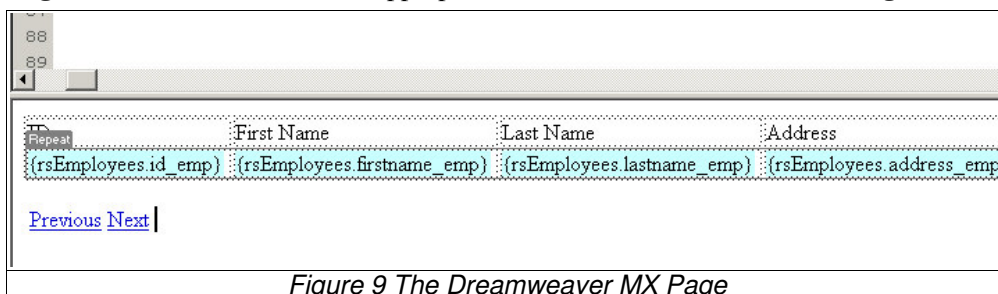


Figure 9 The Dreamweaver MX Page

**Tip:** In the upper **Insert** menu you'll see a tab called **Application**. Clicking on it will present you the application objects that you can use to automatize some of your tasks. You can click in **"Recordset Navigation Bar"** to automatically create a navigation bar. Simply indicate there the appropriate recordset and determine whether it is to create text or image links for the **First**, **Previous**, etc. buttons. If you use images, the picture files are created automatically.



Figure 10 Recordset Navigation Bar

Because usually you will create lists with only a subset of the information shown (for example our list should have contained only the first name and the last name of the employee), it's a good practice to create a link from the list

to the real detail page, where all record information is shown.

In order to display record details, first create a file named *detail.php* containing a recordset *rsEmp* that filters the *employees\_emp* table after the *id\_emp* parameter received as **URL** Parameter. This will load in the recordset a specific record from the *employees\_emp* table. Drag the fields from the recordset into the page.

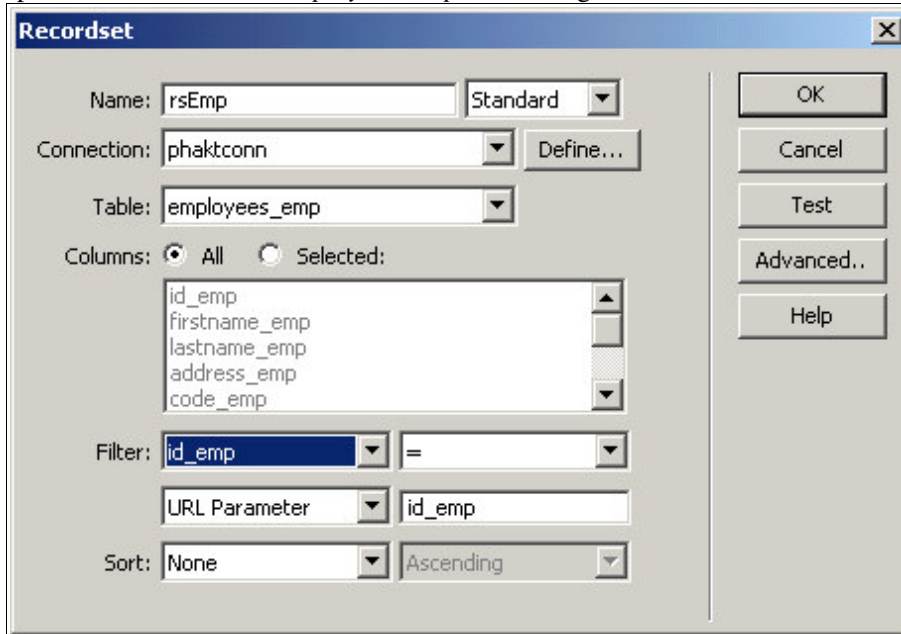


Figure 11 Creating a Recordset

In the *list.php* page, select the field *id\_emp* from the second table row, and then in the menu "**Server Behaviors**" the option "**Go To Detail Page**". You'll find there the transfer parameters to the detail page. 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). See the screenshot below for detailed configuration information.

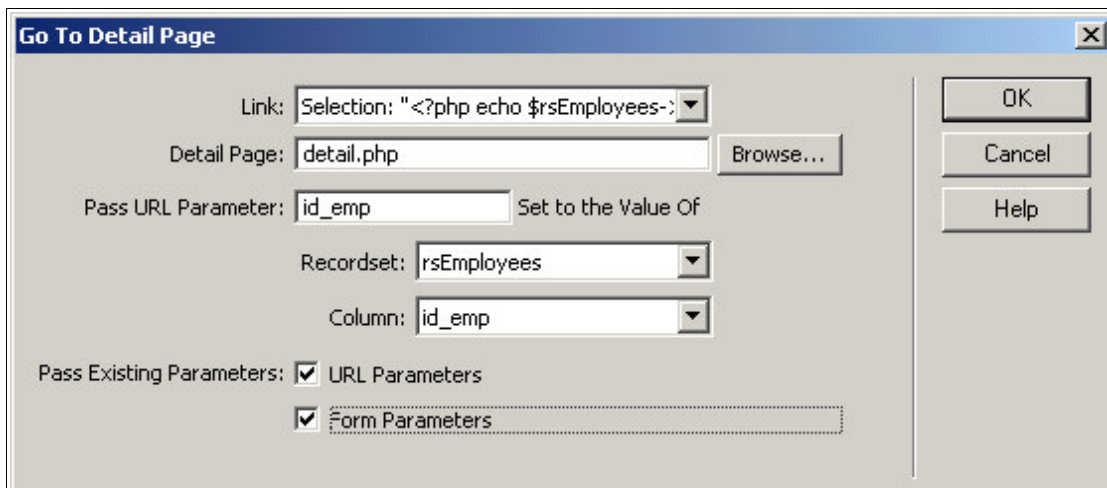


Figure 12 Adding a Go To Detail Page Server Behavior

Please analyze the *detail.php* page from the tutorial for more details on this topic.

### Best practice

**PHAkt 2** already has a feature to create a result page including a navigation border and a detail page very fast. Open the *list.php* file. The basic condition is to create your recordset yourself. After you create it (you can start with a simple recordset on the *employees\_emp* table), click in the menu "**Application**" tab on "**Master/Detail Set**". A form will be shown, please fill it in as instructed below:

- **Recordset:** Indicate the recordset to use.

- **Master Page Fields:** Select which fields to be added in the List page.
- **Link to Detail From:** Field used for linking the Master and the Detail Page
- **Pass Unique Key:** An 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:** The name of the Detail Page
- **Detail Page Fields:** Specify which fields to be displayed in the Detail page.

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

Figure 13 Adding a Master Detail Page Set Server Behavior

## Insert Form

We'll continue by creating an insert form, to allow us to add records in the *employees\_emp* table.

Open the *input.php* file. Create a new form and add text input fields for all table fields (except *id\_emp* and *marital\_emp*) and set their names exactly the same as the related columns in the *employees\_emp* table. Don't forget to add the **Submit** and **Reset** buttons. For the *marital\_emp* field, don't insert a text field, but a **List/Menu** named "*marital\_emp*". Create a recordset named *rsMar* that loads all fields from the *marital\_mar* table ordered by the *status\_mar* field. Use the already created connection "phaktconn".

The inserted List/Menu "*marital\_emp*" will be loaded with records from the *rsMar* recordset. Select the List/Menu control in form and from the "Servers Behaviors"-"Dynamic Form Elements" click on "Dynamic List/Menu". Select the recordset *rsMar* into the *Options From Recordset* interface field. Select the field you want to designate the values actually to be inserted into database and the field you want to represent the data displayed in the

List/Menu (*Values and Labels*):

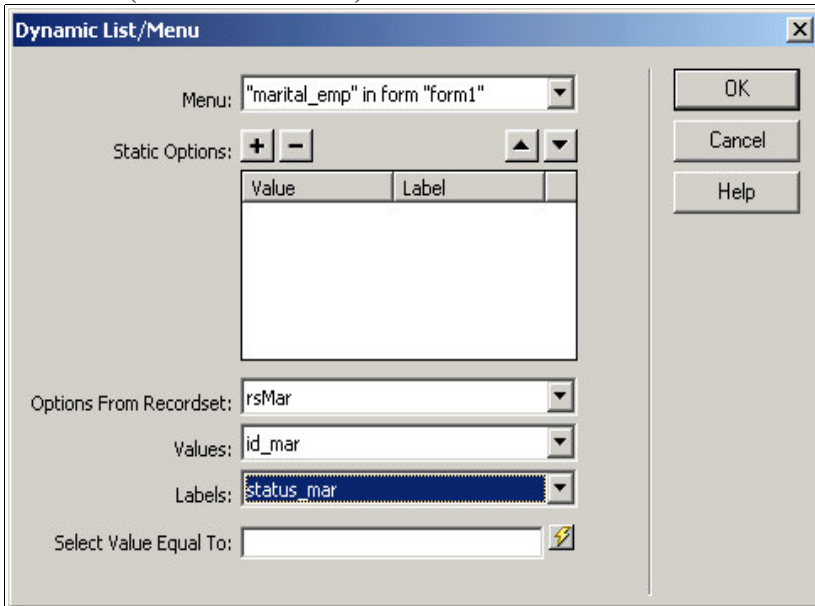


Figure 14 Configuring the Dynamic List/Menu

Click now in the "Server Behaviors" menu on "Insert Record". A new window will appear, please fill in the widgets as instructed below:

- **Submit Values From:** Gives the name of the form containing the data to be inserted.
- **Connection:** the connection to the database.
- **Insert Table:** Specifies the table to insert into
- **Columns:** Sets the column in which the corresponding form element is stored.
- **Value:** specify the form input element that will be stored in the selected table column
- **Submit As:** Sets the type in which data from the corresponding form element is stored (text, date, number, etc.)
- **After Inserting Go To:** Sets the page to be displayed after the insert is made.

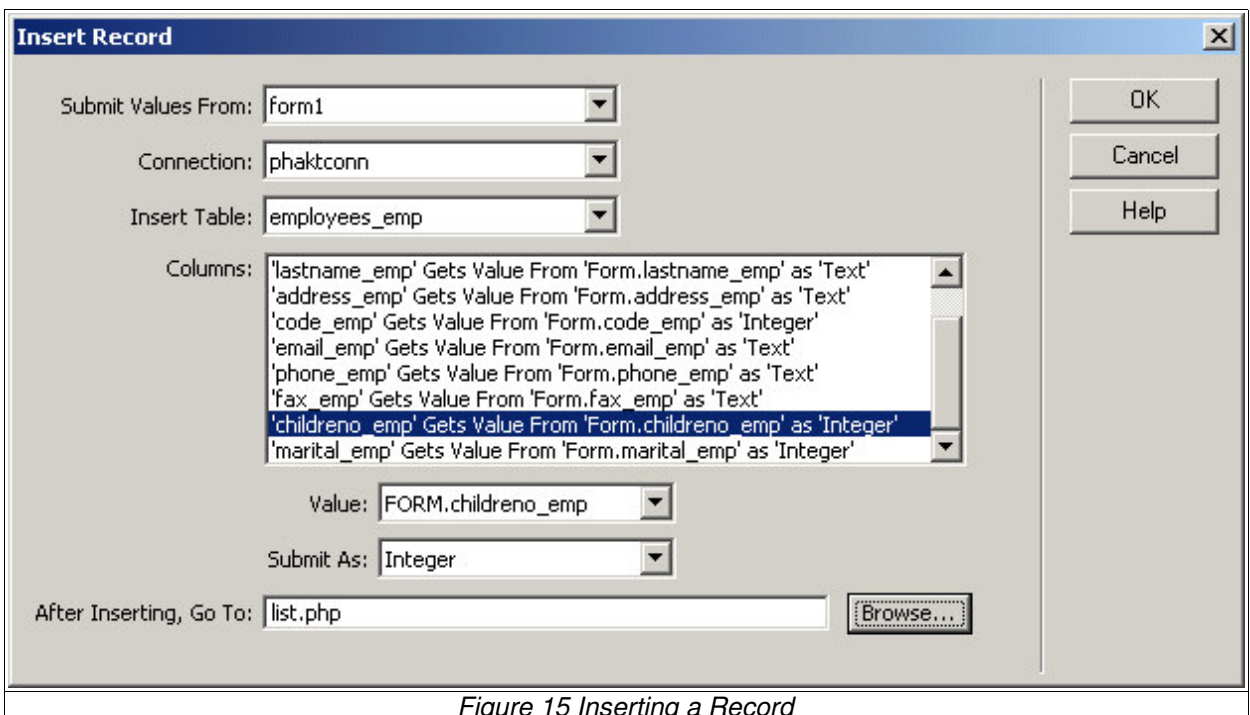


Figure 15 Inserting a Record

Save and upload the page you've just created and test it into the browser – try to input some data. Look into the data list and check whether the insertion took place.

## Best practice

PHAkt 2 already has a feature implemented, that can create an Insert Form in a very efficient way.

In order to compare the results, create a new insert file (e.g. `input1.php`). Open the new file and create a recordset (you can leave the "Recordset1" default name) with the same characteristics as the "rsMar" recordset created in the `input.php` file. Click on the **Application** tab in the upper panel, then click on the "Record Insertion Form". A window will be opened, into which you must input the following:

- **Connection:** Indicates the connection, e.g. "phaktconn"
- **Table:** Indicate the table to insert into
- **After Inserting Go to:** Indicate the page that will be loaded after the insert is made
- **Form Fields:** Allow you to specify which fields will be part of 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.

Record Insertion Form

Connection: phaktconn

Table: employees\_emp

After Inserting, Go To: list.php

Form Fields:

Column	Label	Display As	Submit As
code_emp	Code_emp:	Text Field	Numeric
email_emp	Email_emp:	Text Field	Text
phone_emp	Phone_emp:	Text Field	Text
fax_emp	Fax_emp:	Text Field	Text
childreno_emp	Childreno_emp:	Text Field	Numeric
marital_emp	Marital_emp:	Menu	Numeric

Label: Marital\_emp:

Display As: Menu Submit As: Numeric

Figure 16 Configuring a Record Insertion Form

For the `marital_emp` field you have to use a special form field type that is a Menu type. Set the "Display As" field as "Menu", then configure the Menu as shown below (click on **Menu Properties** button):

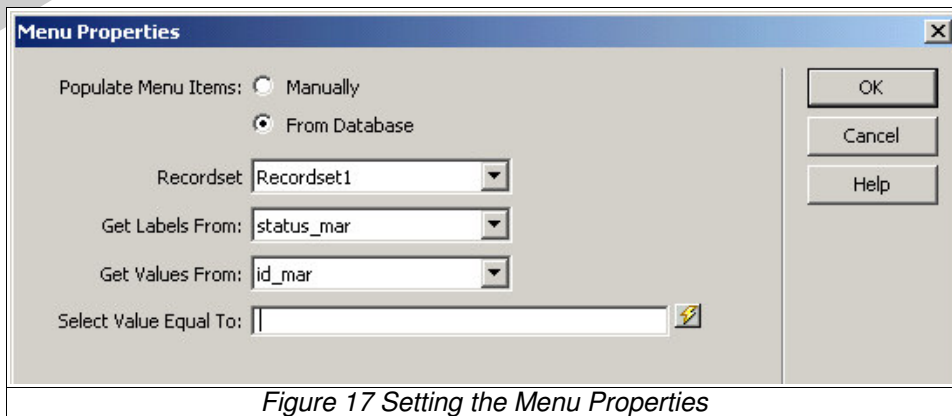


Figure 17 Setting the Menu Properties

After clicking on "OK" the input form is generated. Save and upload the page you've just created and test it into the browser – try to input some data. Look into the data list and check whether the insertion took place.

## Delete Record

According to the same principle as the **Input Record**, a **Delete Record** can also be produced. Open the *list.php* file, select the "Delete" text in the right cells of the table, then add a **Go To Detail Page Server Behavior**.

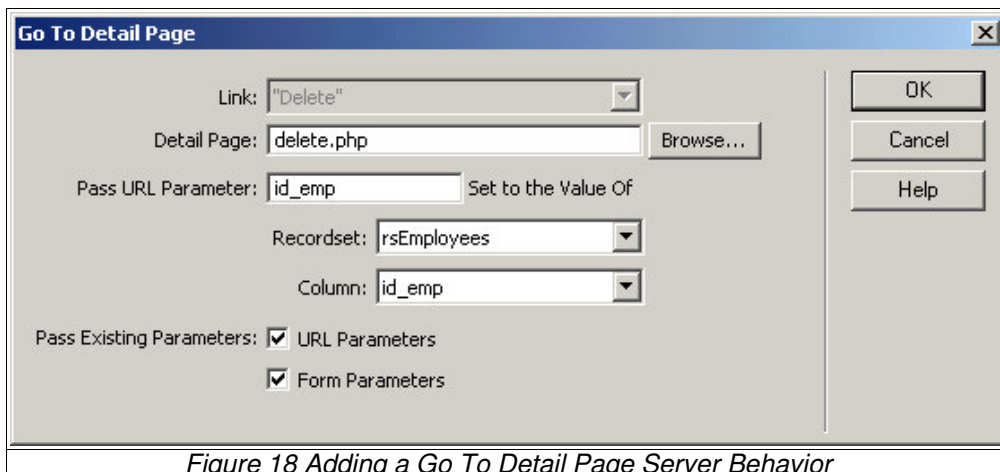


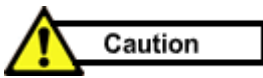
Figure 18 Adding a Go To Detail Page Server Behavior

Then open the *delete.php* file.

Click now in the "Server Behaviors" menu on "Delete Record". A popup window will appear and you have to configure it as following:

- **First Check if variable is defined:** Leave this as the "Primary Key Value"
- **Connection:** Enter here the connection to the database e.g. "phaktconn".
- **Table:** Set the table from to delete from
- **Primary Key Column:** Set the column that designates the unique key for the records in the table, and also indicate id this values is numeric.
- **Primary Key Value:** Specify how the parameter will be passed to this form (in our case it's an URL parameter named *id\_emp*).
- **After Inserting Go to:** Sets the page to be displayed after the delete is made.

Figure 19 Configuring the Delete Record Window

**Caution**

You should pay attention when you add a Delete option into your web application. You should follow the exact steps from above: create a "Delete" link, create a *delete.php* page, add Delete Record behavior in the *delete.php* page and not in the *list.php* page. If you choose not to create a *delete.php* page, and you add the Delete Record behavior in the *list.php* page, the application will go into an infinite loop.

Test your **Delete Record** feature by clicking on the **Delete** link in the *list.php* page.

## Update Form

To be able to update records from the *employees\_emp* table, we'll have to create an Update Record file. Open the *list.php* file, select the "Update" text in the right cells of the table, then add a **Go To Detail Page Server Behavior**.

Figure 20 Adding a Go To Detail Page Server Behavior

Open the *update.php* file, then create a recordset *rsEmp* that filters the *employees\_emp* table on the *id\_emp* URL Parameter. Also create a recordset *rsMar*, that loads all the fields from the *marital\_mar* table ordered by *status\_mar*.

We'll create directly a Record Update form. From the **Application** tab in the **Insert** panel (upper region), click on the **Record Update Form** button. A window will be opened, please configure it as instructed below:

- **Connection:** Indicates the connection, e.g. "phaktconn"
- **Table to Update:** Indicates the database table whose data should be updated (*employees\_emp*).
- **Select Record From:** Indicates the recordset of the data which should be updated (in our case *rsEmp*)

- **Unique Key Column:** Indicates a field that uniquely identifies a row in a table (generally an unique ID, key)
- **After Updating Go To:** Indicates a page, which is to be opened, after the data is updated
- **Form Fields:** Used to map the form fields with the table fields.
- **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.

Please note that the *marital\_emp* field will be generated in a menu created from database (*rsMar*). See the Insert Record explanations above for more details.

The screenshot shows the 'Record Update Form' dialog box with the following configuration:

- Connection: phaktconn
- Table to Update: employees\_emp
- Select Record From: rsEmp
- Unique Key Column: id\_emp (checked as Numeric)
- After Updating, Go To: list.php

The 'Form Fields' section contains the following table:

Column	Label	Display As	Submit As
id_emp	Id_emp:	Text	
firstna...	Firstname_emp:	Text Field	Text
lastna...	Lastname_emp:	Text Field	Text
adres...	Address_emp:	Text Field	Text
code ...	Code_emp:	Text Field	Numeric

Below the table, the 'Label' is set to 'Marital\_emp:', 'Display As' is 'Menu', and 'Submit As' is 'Numeric'. A 'Menu Properties...' button is visible below the 'Display As' dropdown.

Figure 21 Configuring the Record Update Form

After clicking on "OK" the update form is generated. The generated form will look like in the figure below:

Id_emp	{rsEmp.id_emp}
Firstname_emp	{rsEmp.firstname_emp}
Lastname_emp	{rsEmp.lastname_emp}
Address_emp	{rsEmp.address_emp}
Code_emp	{rsEmp.code_emp}
Email_emp	{rsEmp.email_emp}
Phone_emp	{rsEmp.phone_emp}
Fax_emp	{rsEmp.fax_emp}
Childreno_emp	{rsEmp.childreno_emp}
Marital_emp	<input type="text"/>
Update Record	

Figure 22 The Update Page in Dreamweaver MX

## Login Form

The Authentication Server Behaviors are completely implemented and with "PHAkt2" you have the possibility to easily create a database based Login.

Create a new file named "*failed.php*", which will contain a message to show the user that its login attempt failed (e.g. "Login Failed"). You can also create a link to the "*login.php*" page. Save the file.

Create a new document named "*login.php*" and open it in Dreamweaver MX. Add in a new form two text fields, and set their names to "*username*" and "*password*". The password should be defined as password field, to mask the password characters.

Click now in the menu "Server Behaviors" on "User Authentication" -> "Log In User". A window will appear, please configure it as instructed below:

- **Get Input From Form:** Name of the form used for log in (the form you've just created)
- **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. "phaktconn"
- **Table:** The table with the access data in the database e.g. *users\_usr*
- **Username Column:** Column with entries of the user names in the database ("*username\_usr*")
- **Password Column:** Column with entries of the passwords in the database ("*password\_usr*")
- **If Log In Succeeds, Go To:** Page to be opened if the Login attempt runs successfully (e.g. "*list.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 of user name and password, or whether the Login is to be based additionally on a group affiliation (here it is however necessary to create an additional column into the *users\_usr* table, which will contain the appropriate groups the individual Users are assigned, and indicate this column in DWMX).

The 'Log In User' dialog box is configured with the following settings:

- Get Input From Form: form1
- Username Field: username
- Password Field: password
- Validate Using Connection: phaktconn
- Table: users\_usr
- Username Column: username\_usr
- Password Column: password\_usr
- If Login Succeeds, Go To: list.php
- Go To Previous URL (if it exists):
- If Login Fails, Go To: failed.php
- Restrict Access Based On:
  - Username and Password
  - Username, Password, and Access Level
- Get Level From: id\_usr

Figure 23 Configuring the Log In User Server Behavior

To really protect the site pages, you have to add the **Restrict Access to Page** Server Behavior to every page (excepting of course *login.php* and *failed.php*).

To create a logout link (preferable on every page, but you can test it only in one page – e.g. *list.php*), type “Logout”, select the text and click in the menu "**Server Behaviors**" -> "**User Authentication**" -> "**Logout User**". A dialog-box will open where you specify whether 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).

The 'Log Out User' dialog box is configured with the following settings:

- Log Out When:
  - Link Clicked: Selection: "Logout"
  - Page Loads
- When Done, Go To: login.php

Figure 24 Configuring the LogOut User Server Behavior

## Dynamic items

We'll continue by creating a new page, independent of the rest of the site. The goal is to see how we can use dynamic form elements. Create a new document named "*dynamic.php*". Add a **RadioButton** named "*radio*" and a **Checkbox** named "*checkbox*". Add a recordset named "*rsDyn*", as source for the inputs - select again the well-known connection "phaktconn" and set the recordset to load its values from the *marital\_mar* table.

Select the Checkbox and click in the "Server Behaviors" menu on "Dynamic Form Elements" - "Dynamic Check Box". A dialog window will appear, where you can set the name of the Checkbox to be dynamized, the condition to check the Checkbox to ("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 are configuring the Log In User Server Behavior and have more Checkboxes, repeat the procedure for each of them.

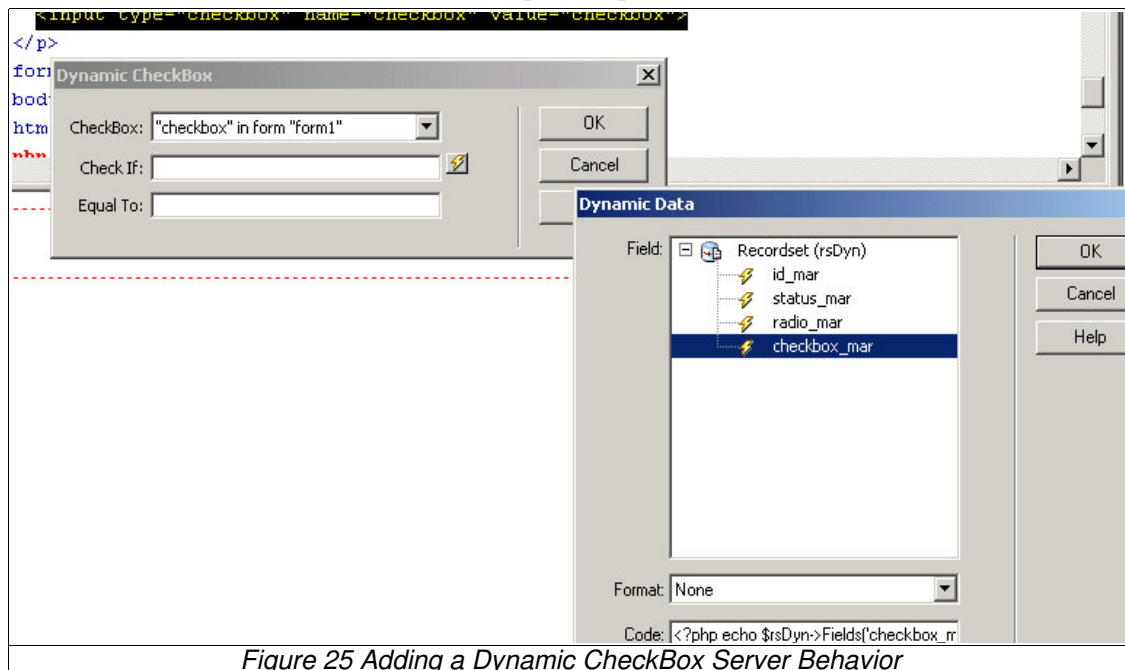


Figure 25 Adding a Dynamic Check Box Server Behavior

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.

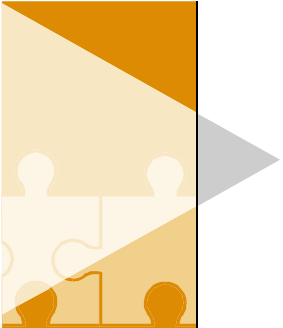
## Miscellaneous

"PHAkt2" also offers some very useful auxiliary functions.

Weekly, **PHAkt2** connects to the **InterAKT** web site (the complete address is "http://www.interaktonline.com/demo/Krysalis/webapps/update/index.php?KTURL=information.pxp&prodId=0") to retrieve its current version number and therefore to inform you if an update is available. This action is performed weekly, only from the Dreamweaver MX pages containing **PHAkt2** recordsets.

You can add a **Recordset Navigation Status** from **Application** tab in the **Insert** panel, that will display the numbers of the records displayed (e.g. "Records 1-5 from 10"). You only have to select the used recordset .

That should be all. For supplemental information, don't forget to check <http://www.interaktonline.com/products/PHAkt/> for updates.



## Copyrights and Trademarks

Copyright - 2000-2003 by InterAKT Online, SRL.

All Rights Reserved. This tutorial is subject to copyright protection.

PHAkt, ImpAKT, NeXTensio, QuB, Transaction Engine are trademarks of InterAKT.

All other trademarks are acknowledged as the property of their respective owners.

This document and the product to which it pertains are distributed under licenses restricting their use, copying, distribution, and decompilation. No part of this document or of the associated product may be reproduced in any form by any means without prior written authorization of InterAKT Online, except when presenting only a summary of the tutorial and then linking to the InterAKT website.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Send comments and suggestions to [products@interaktonline.com](mailto:products@interaktonline.com).

No

**Interakt**

professional web tools

---

InterAKT Online  
web: <http://www.interakt.ro/>  
e-mail: [contact@interakt.ro](mailto:contact@interakt.ro)

tel: +4021 312 5 312