

Tutorial
Classified Ads



Table of contents

Introduction	3
Plan the Classified Ads Site	4
Post and List Ads	8
User Authentication	8
List classified ads by category	10
Creating the recordset that contains the classified ads	10
Displaying classified ads	13
Create a tree menu with categories	15
Create a login nugget	18
Post classified ads	21
Create site front-end with MX Includes	26
Improve the Classified Ads Site	30
Create advanced search form	31
Register to receive ads by e-mail	39
Add breadcrumbs navigation trail	45
Other Resources	47
Copyright	48

Introduction

In this tutorial, you will create a web application which will allow the posting and the viewing of classified ads. Ads can be browsed by category, with the help of a tree menu, and only registered users will be able to post ads. Users of your web application will also be able to search ads that match specific criteria.

Pre-requisites:

In order to complete this tutorial, you need to have **Macromedia Dreamweaver** and the **MX Kollection 3** bundle installed. Alternatively, if you do not have the whole bundle, you will need the following separate products:

- ImpAKT - for all of the basic database operations.
- MX Send E-mail - to be able to send e-mail messages from the site.
- MX User Login - to implement the user authentication.
- MX Includes - to display content from different pages in the same file.
- MX Query Builder

The final part of the tutorial also requires these additional free extensions from InterAKT Online:

- MX Breadcrumbs - to create a navigation trail, on top of your pages.
- MX Tree Menu - to create the category listing as a tree menu.

You can download the two free extensions by visiting our website.

The estimated completion time for this tutorial is about 60 minutes. It depends on your web authoring knowledge with **Macromedia Dreamweaver** (MX or MX 2004) and **MX Kollection 3**.

It is recommended that you follow this tutorial in the intended order to avoid potential problems.

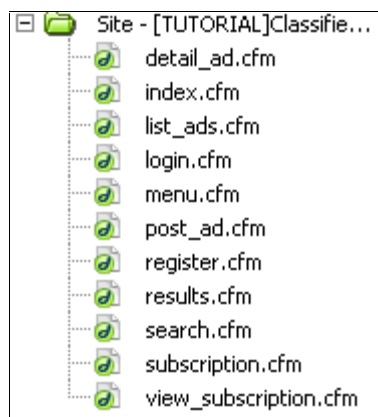
Plan the Classified Ads Site

This first section of this tutorial helps you create all the files and database tables needed for the application.

Before you start building this application, make sure you have a correctly configured **Dreamweaver** site, and a working database connection. For more instructions regarding these actions, consult the *Getting started* help file, which can be found in **Help -> InterAKT -> Getting Started**.

In the tutorial, you will have to create several files in your site root folder. You can create them at the very beginning, so that you will not waste time with this operation again. To create files and folders in the site's root, use the corresponding options in the **File** menu of the **Files** tab.

All files created in this tutorial can also be found inside the downloaded package and you can use them to compare your work with what has been already done. The file structure will look as in the example below, and you can create it easily by unpacking the *zip* file corresponding to your server model from `\tutorials\Classified Ads\` in your site root:

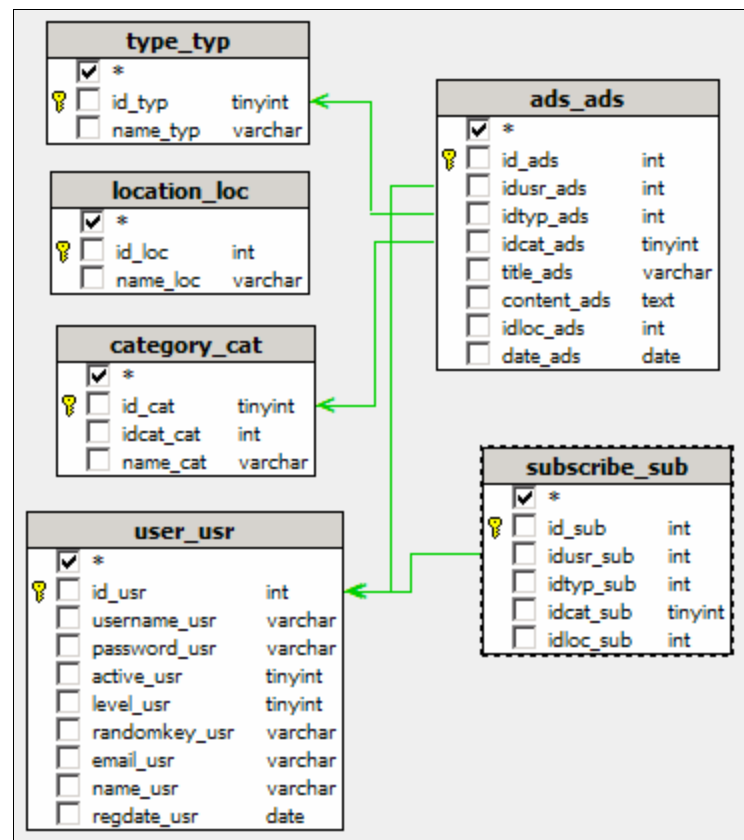


After having created the files for your pages, it is time to set up the database that will hold the information for the application. For this tutorial you will use 6 tables containing data about the ads and users. These tables are:

1. *ads_ads* - to store the classified ads.
 - **id_ads** - the primary key for the ads table. No two ads can have the same ID.
 - **idusr_ads** - foreign key to the users table, so you'll know which message belongs to which user. Login and access restrictions are stored there.
 - **idtyp_ads** - a foreign key to the type table. Each classified ad must have an associated type (For-sale, Wanted etc.)
 - **idcat_ads** - a foreign key to the category table. Stores the category where the classified ad goes in (Cars, Books etc).
 - **title_ads** - stores the ad subject.
 - **content_ads** - stores the actual contents of the classified ad.
 - **idloc_ads** - foreign key to the table that stores the location of the seller or buyer (Denver, Omaha etc.).
 - **date_ads** - stores the date when the ad was posted.
2. *user_usr* - stores information about the site users.
 - **id_usr** - the primary key for the users table. No two users can have the same ID.
 - **username_usr** - the name used by the user to authenticate to the website.

- **password_usr** - the user's password.
Note: In the database created for this tutorial, the passwords are encrypted so to offer the application a higher level of security.
 - **active_usr** - a boolean flag (1/0) showing if the user account has been activated or not.
 - **level_usr** - a boolean flag (1/0) which stores the level of access the user is allowed. Level 1 corresponds to administrators and level 0 to common users.
 - **randomkey_usr** - stores a random alpha-numeric key generated for secure account activation.
 - **email_usr** - the user's email address.
 - **name_usr** - the user's real name.
 - **regdate_usr** - the date when the user registered to the classified ads site.
3. *type_typ* - stores the ad types: For sale, Wanted, Exchange, or Don't care.
- **id_typ** - the primary key for the type table. No two types can have the same ID.
 - **name_typ** - stores the actual type of the message.
4. *category_cat* - stores the ad categories.
- **id_cat** - the primary key for the category table. No two categories can have the same ID.
 - **idcat_cat** - a self-foreign key to the parent category, if any. For example, for the mobile accessories category, this field would contain the id of the Mobile category (the parent). For main categories, that is categories which do not fall under another parent category, the idcat_cat field value is 0. You can find more details on how self-foreign keys here.
 - **name_cat** - the category name (Automotive, Electronics etc.).
5. *location_loc* - table that stores the location where the ad is available.
- **id_loc** - the primary key for the location table. No two locations can have the same ID.
 - **name_loc** - the location name (Chicago, Paris etc.).
6. *subscription_sub* - stores subscription data for the users. Users can subscribe to received new classified ads for their categories of interest.
- **id_sub** - the primary key for the subscription table.
 - **idusr_sub** - a foreign key to the users table. Keeps track of which user has a subscription.
 - **idtyp_sub** - foreign key to the type table, indicating the type of ads the user wants to receive.
 - **idcat_sub** - foreign key to the categories table, indicating the category the user subscribed to.
 - **idloc_sub** - foreign key to the locations table, indicating the locations where the users is interested in receiving ads from.

Below is a layout of the tables using **InterAKT's** visual query editor, **MX Query Builder**:



Note: The database diagram in the image above was built with **MX Query Builder** (also referred as **QuB**) to better illustrate the database structure. You do not need to build it in order to complete this tutorial.

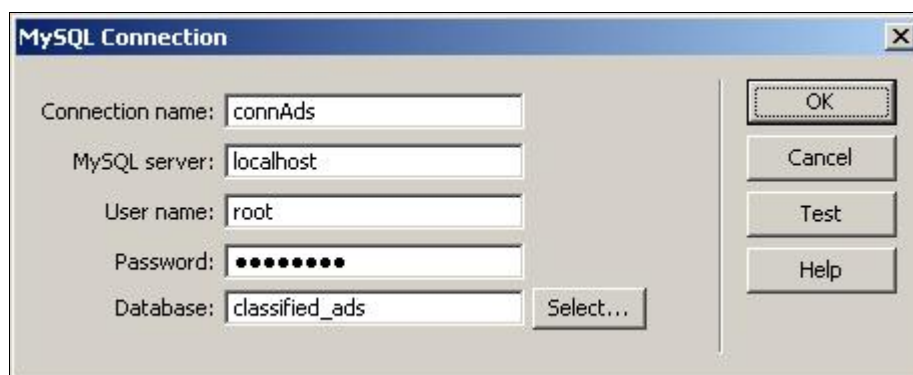
You can find the scripts needed to create an identical table structure inside the downloaded package, in the `\tutorials\Classified Ads\db\` folder, as an `sql` or `mdb` file, depending on the database server you intend to use. Import them in your database server management software (e.g. **PHPMyAdmin**, **Microsoft Access** etc).

The scripts already contain data so that you can easily view the results. In order to access all parts of the application, you can use the default user accounts contained in the `sql` scripts. The accounts for the classified ads application are:

- username: **john**; password: **root**.
- username: **jane**; password: **root**.

These user accounts will be used for authenticating to the web application, to allow you view and post ads. In the database, the passwords are encrypted, as mentioned above.

Before proceeding with the tutorial, open the main `index` page and create a new connection named `connAds` and configure it to connect to your newly created database:



If you use **ColdFusion**, you will need to define a valid data source using the ColdFusion Administrator interface. For more details, check out the Getting Started book, accessible from the Help menu > **InterAKT -> Getting Started.**

In the next chapter, you'll learn how to post and list classified ads.

Post and List Ads

In this first part of the tutorial you will build the basic functionality of the Classified Ads site. The pages you will create will perform the following actions:

- Display a tree menu of categories for visitors to choose.
- Display the ads in the selected category.
- Allow visitors to register to the site.
- Allow registered users to log in.
- Logged in users will be able to post new ads.

User Authentication

The first thing to build is the user registration and authentication system. To do so, follow the steps in the User Authentication tutorial. The main steps of building the user authentication system are:

- Configure the Login Settings in the InterAKT Control Panel > Login Settings, as shown here.
- Build the user registration to let users create an account.
- Restrict access to the pages *post_ad* and *subscription*, so that unauthenticated users will not be able to post classified ads, or register to receive them on e-mail:



Note: Keep in mind that you will have to modify some of the interface fields to reflect your current database settings. Also, user levels are not absolutely necessary for the *Classified Ads Site*. Unless you intent to extend this application by building an administration area, where an administrator can manage categories, locations, ads and so on, you can safely ignore them in the User Authentication tutorial.

If you have the **MX Kollection 3** bundle installed, then you have all the needed tools. Otherwise, the following separate products should be installed on your computer in order to complete this section of the tutorial:

- MX User Login

- MX Includes
- MX Query Builder
- MX Tree Menu - It's a free extension that has to be installed separately from **MX Kollection 3**. Download it from the **InterAKT** web site.

The estimated completion time for this section is about 30 minutes. It depends on your authoring knowledge with **Macromedia Dreamweaver** (MX or MX 2004) and **MX Kollection 3**.

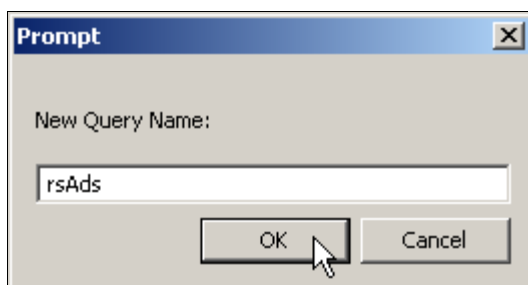
List classified ads by category

In this section, you will create the page that lists the ads in a selected category. The ads will be retrieved from the `ads_ads` table by a recordset that is filtered after the category ID. This ID is passed as an URL parameter called `id_cat`, from the tree navigation menu, when the user clicks on a category.

To create this page, follow the next steps:

Creating the recordset that contains the classified ads

1. Open the `list_ads` page in **Dreamweaver**.
2. Next, you need to create a recordset that retrieves all the ads, their associated categories and locations, their type and the users who posted them. You need all this information in order to display all the details for a classified ad. You also need to filter this recordset by an URL parameter that is passed from the navigation menu, which you will create later. When visitors click on a category from the menu, they need to see only those ads that fall under that category. To create the recordset, go to the **Bindings** tab in the **Application** panel, click the + button, and select **Recordset** from the displayed menu. In the displayed **Recordset** window, click the QuB3 button, to go to the QuB Visual Recordset interface.
3. Enter `rsAds` in the recordset name field, select the database connection you have created (`connAds`) and click the **New Query** button, to create the query that will extract the information you want.
4. In the displayed interface, enter a name for the SQL query you will create, such as "rsAds".

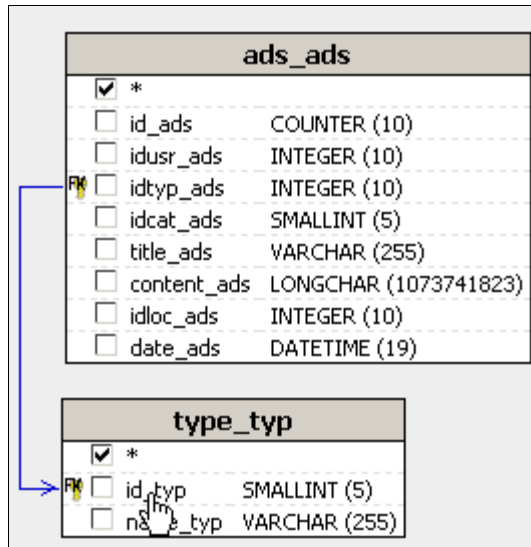


5. After clicking OK, the MX Query Builder web interface will open, where you will be able to visually generate the recordset.
6. In the Tables panel, select the tables that will be included in the query: `ads_ads`, `location_loc`, `user_usr`, `type_typ`.

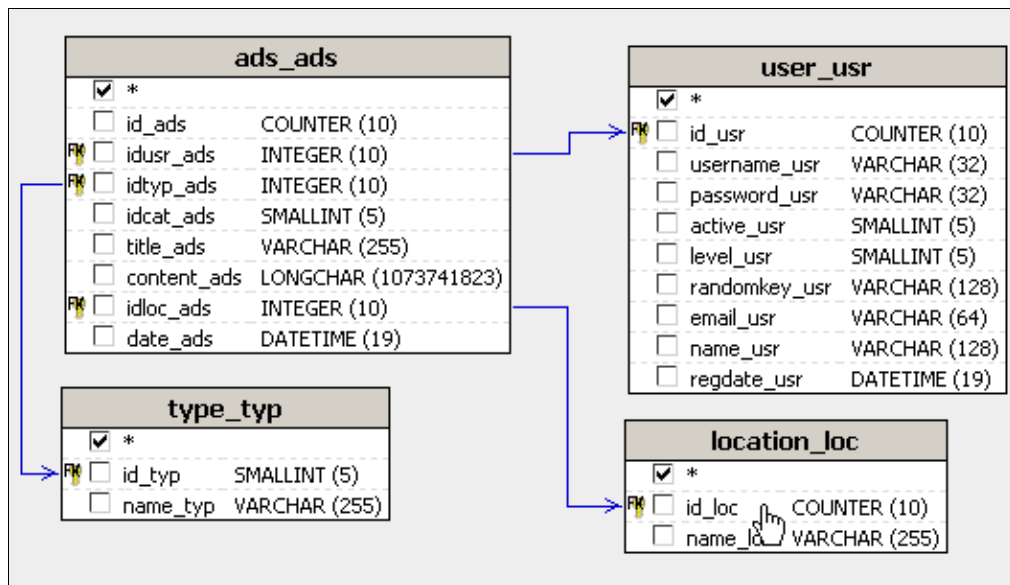


The corresponding tables will be displayed in the Database Diagram, where you can arrange them in any way you like.

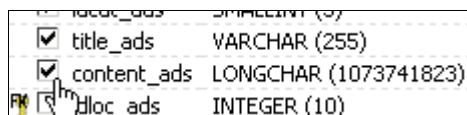
- Next, link the tables, by dragging each foreign key onto the primary key it references. For instance, drag the `idtyp_ads` column from the `ads_ads` table over the `id_typ` column in the `type_typ` table:



You also need to define relations between `idusr_ads` and `id_usr`, between `idloc_ads` and `id_loc`. The database diagram should look like this:



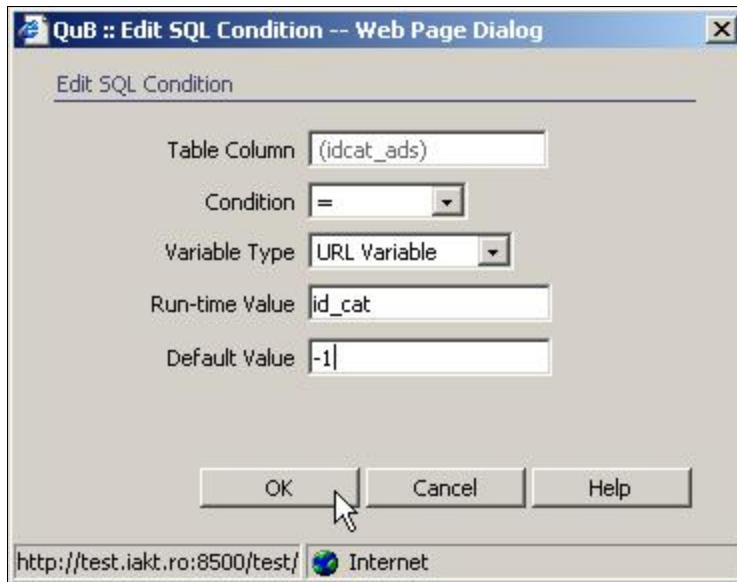
- Next, select the columns you need for your recordset: `title_ads`, `content_ads`, `name_loc`, `name_typ`, `email_usr`, `idcat_ads`, `date_ads`.



- The `idcat_ads` needs to be equal to the URL parameter received from the navigation menu, when users click on a category. To define this condition, click the button next to the Condition text field, corresponding to the `idcat_ads` column in the Query Management Panel:

Column Name	Alias	Sort	Output	Grouping	Condition
ads_ads.idcat_ads			<input checked="" type="checkbox"/>		

10. Configured the displayed dialog box as in the following image (the column must be equal to the URL parameter id_cad):



11. The last thing you need to do before saving the query is to order the classified ads by their date, in descending order, such that the latest ones are always displayed first.

location_loc.name_loc			<input checked="" type="checkbox"/>		
ads_ads.date_ads		DESC	<input checked="" type="checkbox"/>		

Select DISTINCT

ASC
DESC

12. Your query is now complete. You can view the query results or the generated SQL code in the SQL/Results Preview panel:

```

SELECT ads_ads.idcat_ads, ads_ads.title_ads, ads_ads.content_ads, user_usr.email_usr, type_typ.name_typ,
location_loc.name_loc, ads_ads.date_ads
FROM ((ads_ads
LEFT JOIN location_loc ON location_loc.id_loc=ads_ads.idloc_ads)
LEFT JOIN type_typ ON type_typ.id_typ=ads_ads.idtyp_ads)
LEFT JOIN user_usr ON user_usr.id_usr=ads_ads.idusr_ads)
WHERE ads_ads.idcat_ads=#URL.id_cat#

ORDER BY ads_ads.date_ads DESC
    
```

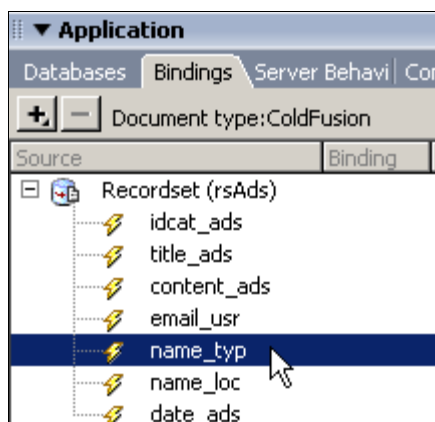
Click the **Save** button from the **Query List** panel, to save your settings.



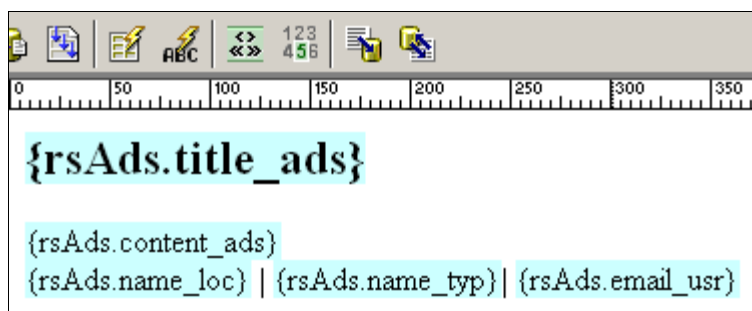
13. To close the **MX Query Builder** web interface and return to Dreamweaver, select Close from the **QuB** menu.
14. Back in the **MX Query Builder Visual Recordset** interface, click the Refresh button to load the generated SQL code. Your recordset is ready. Click OK.

Displaying classified ads

1. Now that you have all data available in the **Bindings** tab, it is time to put it up for display. Or in other words, you need to actually show the ads in your page. The most important things regarding an ad are its title, content, type, location and contact e-mail address. So these will be the fields to be displayed. Drag and drop each of them from the *rsAds* recordset in the **Bindings** tab onto the page.



2. The layout is yours to choose, although for this tutorial's purpose, a simple one is enough:

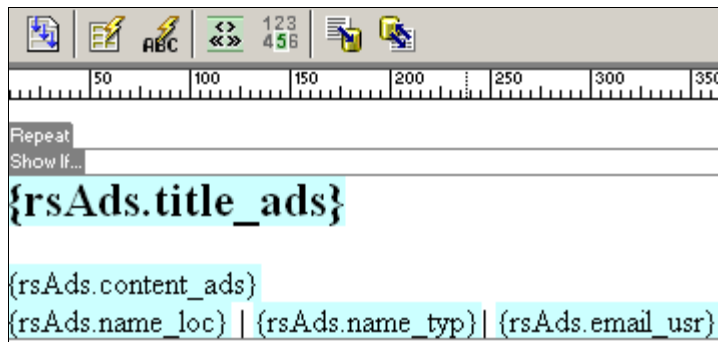


3. To prevent seeing the separator bars, or even errors when no category is selected or when there are no ads in the current category, the entire region shown above will have to be enclosed in a conditional region, allowing it to be visible only if the *rsAds* recordset contains data. For this you will use a standard **Dreamweaver** server behavior, called **Show If Recordset Is Not Empty**. It is accessible from **Server Behaviors -> Show Region**.

4. The dialog box that opens contains only a drop-down menu, allowing you to select the recordset to check for data.
5. Now the text will display only if the recordset contains data, but it will only display one record. To make it display all retrieved ads, you must apply a **Repeated region**. This command is accessible from the **Application** tab of the **Insert** panel.
6. Configure the dialog box that opens to use the *rsAds* recordset, and to display all records.



7. Once you click the **OK** button, the repeated region server behavior will apply on the selected region, which will be marked by a second translator (the first one was put in by the conditional region):



8. Now if you preview the page in the browser, providing it a correct category ID for which there are ads in the database, you will see a list of the ads, instead of only one.

This page will be included together with a tree-like menu of categories and a login nugget on the site's first page. In the next section you will create the tree menu containing links to the specific categories.

Create a tree menu with categories

To ease navigation through the ads, you can place a menu on the home page, allowing visitors to browse the categories on your site. Though it can be done with simple tables and some conditional regions, the easiest (and best looking) way is to use another **InterAKT** extension: **MX Tree Menu**. This extension is not included in **MX Kollection**, but you can download it for free from the InterAKT web site.

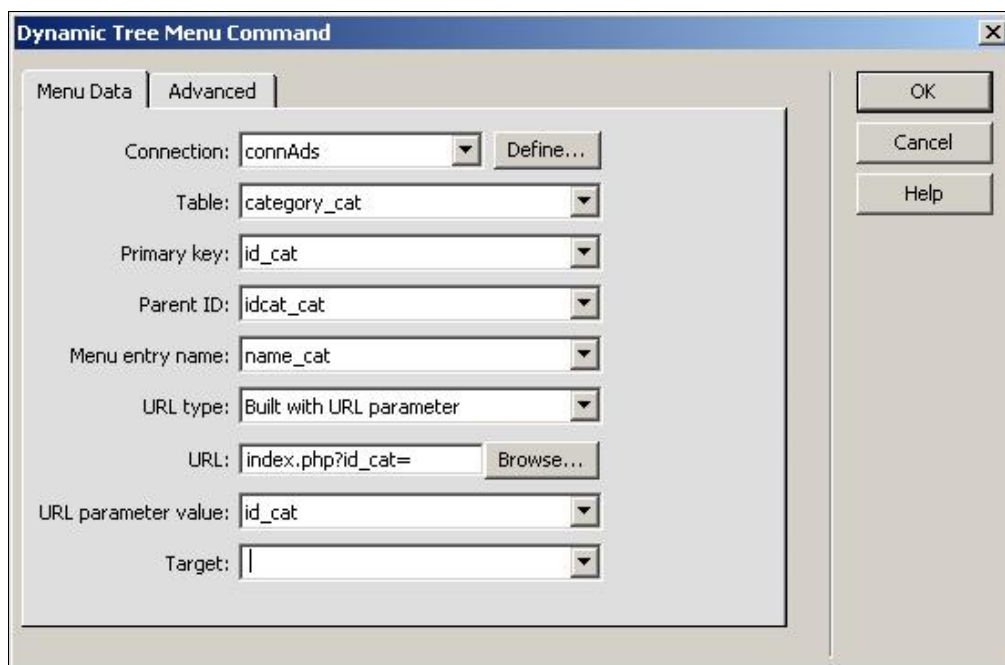
Before you start creating the page described in this section of the tutorial, you must download and install the extension mentioned above.

What this extension does, as its name suggests, is to create a tree-like menu, from the category table. The field that determines which category is on which level, is the self foreign-key **idcat_cat**. If this field contains the value 0, then the category is a top-level one; otherwise, it will be placed under the category whose id it contains.

After you've installed this extension, a new tab called **InterAKT** will appear on the **Insert** panel. On it there are two buttons, allowing you to insert a tree menu based on database fields, or on manually entered values.

If you have installed the **MX Tree Menu** extension, and understood what the table structure is for the *category_cat* table, you can proceed and create the page:

1. First open the *menu.php* page in **Dreamweaver**.
2. Apply the **Dynamic Tree Menu Command** from the **InterAKT** tab of the **Insert** panel.
3. Configure the dialog box that opens as follows:
 - In the **Connection** drop-down menu select the database connection created at the beginning of the tutorial: *connAds*.
 - The table containing the items to be displayed is *category_cat*.
 - The Primary key is selected automatically, and in the **Parent ID** drop-down menu, select the table field storing the upper category: *idcat_cat*.
 - In the **Menu entry name** select the labels for the menu elements: *name_cat*.
 - Each element is in fact a link to a page that will display the ads in the respective category. Since the menu will have to be available all the time, the easiest way is to create the main page with **MX Includes**, allowing you to combine several pages. This way, you are using the same page, which receives the category ID as an URL parameter.
 - The dialog box options that sets this is the **URL type**. Select **Built with URL parameter**. In the following fields, enter the **URL**, complete with the URL parameter name, and in the drop-down menu, select the table field that will provide the values.
 - Leave the **Target** drop-down menu at its default value, because it is not necessary for this page.



Dynamic Tree Menu Command

Menu Data | **Advanced**

Connection: connAds Define...

Table: category_cat

Primary key: id_cat

Parent ID: idcat_cat

Menu entry name: name_cat

URL type: Built with URL parameter

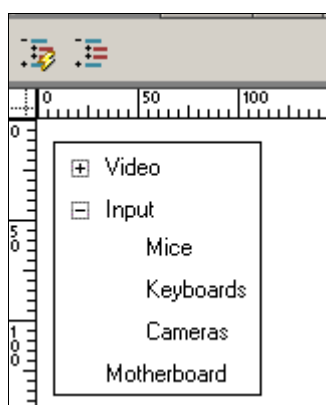
URL: index.php?id_cat= Browse...

URL parameter value: id_cat

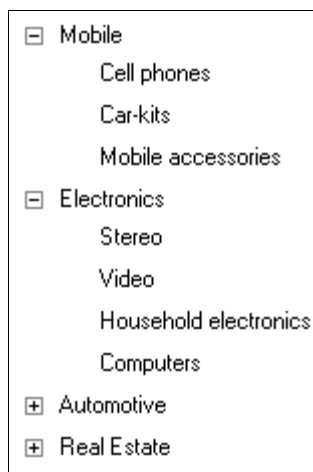
Target: |

OK
Cancel
Help

- Once you click on the **OK** button, the menu will be added to the page, and a translator will be shown in **Dreamweaver**:



- You can save the page and preview it in the browser by pressing the **F12** key. The page will display a tree-like menu, with the categories that have the **idcat_cat** 0 (zero) as top-categories:



Now that you have created the menu that will help you navigate through the site, you can proceed and integrate it along with the other components of the main page in the next part of the tutorial.

Create a login nugget

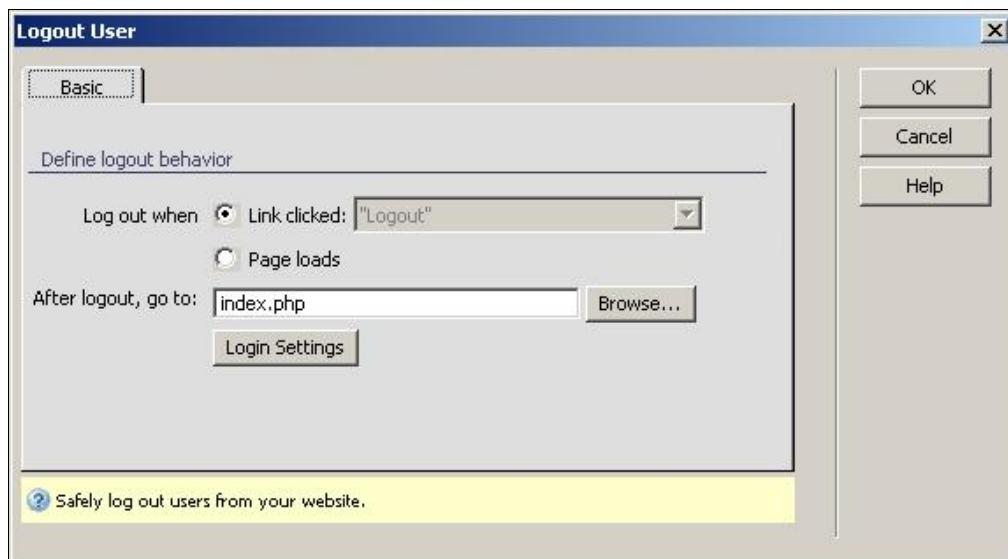
In this tutorial section you will create a page that will be shown as a nugget in the site's main page. The page will display some menu options for the users that are already logged in, or the login form for the users that are not logged in yet.

To create this page, you will use the **Show If User Is Logged In** conditional region, with the else option enabled. This will allow displaying a region if the user is logged in, and another if he isn't. Follow the next steps to create the page:

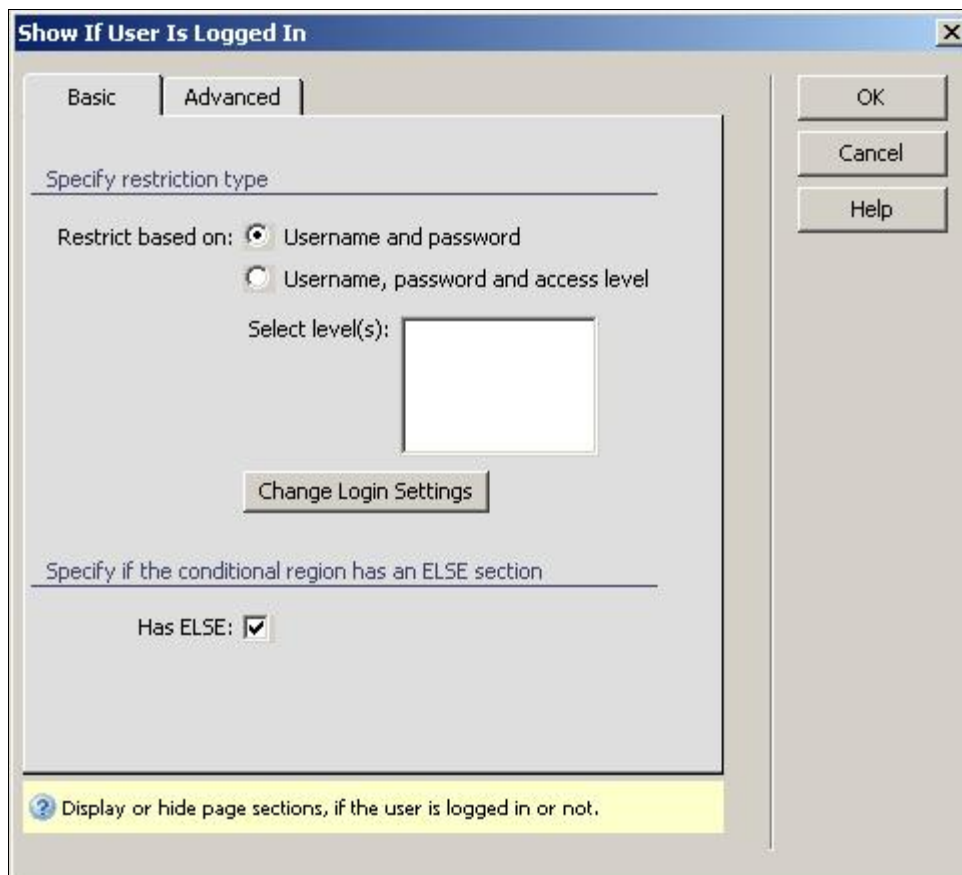
1. First open the *login.php* file in **Dreamweaver**.
2. Type in the menu options that the user will have one by one, and also turn them into links, as explained below:
 - **Post Ad** - link to *index.php?act=post*
 - **Logout** - no link.
3. Next you will turn the logout text into an actual logout link. Select the text and apply the **Logout User** server behavior from **Server Behaviors > MX Kollection > User Login**.

Note: Before applying this server behavior, make sure you have the login settings correctly configured.

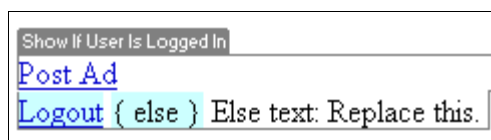
The configuration is straightforward, as you can see below:



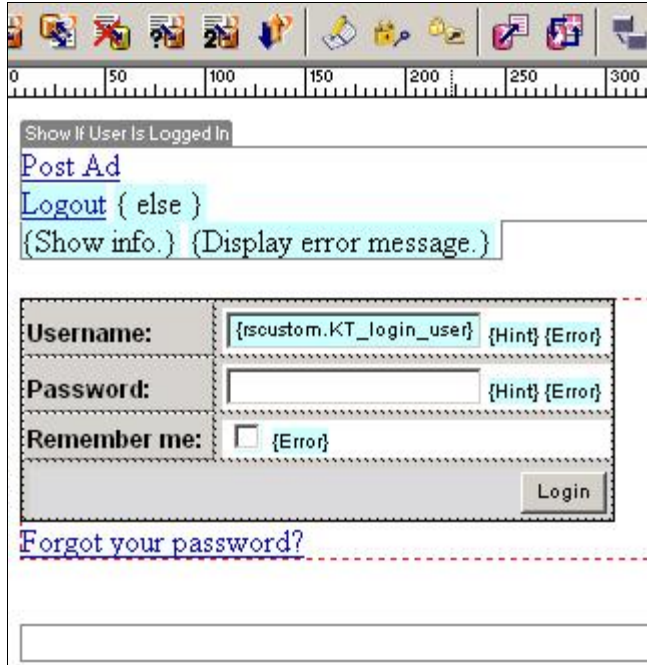
4. Once you're done with the **Logout User** dialog box, click the **OK** button to apply the server behavior.
5. Now select the entire page contents (the two links) and apply the **Show If User Is Logged In** server behavior from the **Server Behaviors tab -> MX Kollection -> Conditional regions**. Configure it as shown below:



6. This will determine if the user is logged in based on the user name and the password, and also has an else region. The else region contents are displayed when the user is not logged in.
7. Now the **Dreamweaver** page looks like the following:



8. To add the login form, select the text after the `{else}` mark-up: *Else text: Replace this.* Apply the **User Login Form Wizard** from the **MX Kollection** tab of the **Insert** bar, and configure it similar to the User Authentication login page. It will automatically add the login form into the page. Once completed, the nugget page will look like the following in **Dreamweaver**:



9. Save the page and upload it to the server. This was the last piece needed in order to create the site's main page.

In the next part of the tutorial, you will assemble all the pages you've created so far into a single page, the site's main index.

Post classified ads

Posting a new ad to the site is reduced to a simple insert operation, that will populate the *ads_ads* table with the entered data. The file that contains the elements to allow users post a new ad is called *post_ad* and is stored in the site root.

Now, to create the actual insert form, you will use the **Insert Record Form Wizard**. Open it by pressing its corresponding button on the **MX Kollection** tab of the **Insert** panel, and configure it as follows:

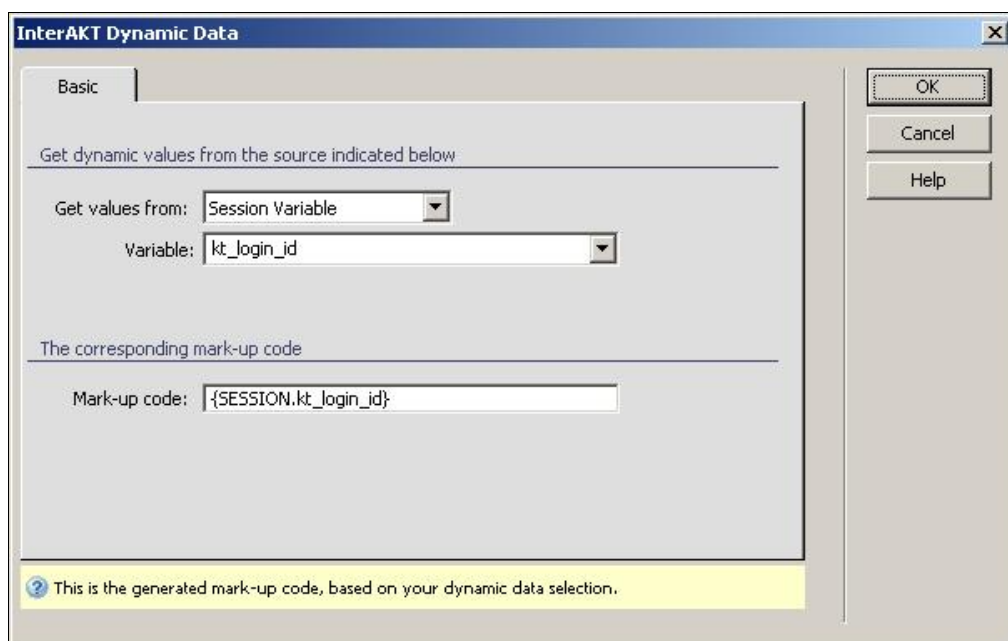
1. First of all, in the wizard's first step, select the database connection created at the beginning of this tutorial, the table into which the insert operation will take place (*ads_ads*) and the page that will be opened if the transaction is successful.

The screenshot shows the 'Insert Record Form Wizard' dialog box at Step 1/3: 'Select a table to insert into'. The dialog is titled 'Specify database information' and contains the following fields and controls:

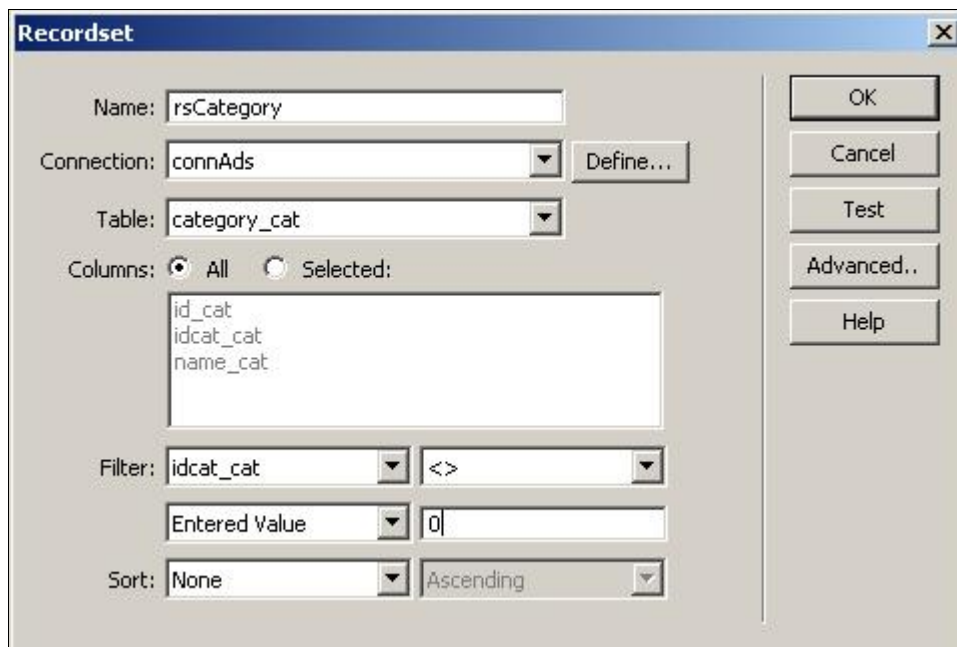
- Connection:** A dropdown menu set to 'connAds' with a 'Define...' button to its right.
- Insert into table:** A dropdown menu set to 'ads_ads'.
- Primary key column:** A dropdown menu set to 'id_ads' with a checked checkbox labeled 'Numeric' to its right.
- Specify redirect page:** A section with the label 'Specify redirect page' and a text box containing 'index.php' with a 'Browse...' button and a file icon to its right.

At the bottom of the dialog, there is a yellow status bar with the text 'Specify the page where to go after inserting a record.' and a set of navigation buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

2. In the second step of the wizard, you will have to define what table column to display in the form, how they will be represented and submitted. When you move on to the second step by pressing the **Next >** button, the **Form fields** grid will already be populated with almost all the columns in the *ads_ads* table. The one that is missing is the primary key.
3. As you have noticed in the application planning section, the *ads_ads* table stores the following data: the user, ad type, ad category and ad location ID's, the title and the content of the ad. The ID's are stored as numeric values, and in order to display them conveniently for the users, you should use drop-down menus that retrieve their values from the type, category and location table. The user ID is most easily taken from the session variable.
4. Select the fields from the grid, and change their properties as follows:
 - For the *idusr_ads* field, set the **Display as** to **text**, and in order to retrieve the currently logged in user's ID, in the **Default Value** text-box select the *kt_login_id* session variable from the InterAKT Dynamic Data dialog box:



- The next field is the **idtyp_ads**, that stores the id of the ad type. Types are stored in a separate table, *type_typ*. To avoid having the users enter the numeric value, set the form element to be used to Menu (from the **Display as** drop-down menu). Two more buttons became available: **Add recordset** and **Menu Properties**. As their names suggest, the first will be used to create a recordset that will retrieve the types from the database, and the second will configure the menu to use the correct columns to display labels, and store values.
- Click on the **Add Recordset** button. The standard new recordset dialog box will open. Configure it to retrieve all records from the *type_typ* table, through the database connection you've created at the beginning of this tutorial. When you're done configuring it, click **OK** to return to the **Insert Record Wizard**.
- Now configure the menu to use the recordset you've just created. Click the **Menu properties** button, and select the appropriate options: the recordset name, the label column (**name_typ**) and the values column (**id_typ**). Then click on **OK** to finish configuring the menu.
- For the category and location fields, you will proceed in a similar manner, changing the recordset each time to retrieve data from the desired tables: *category_cat* and *location_loc*. For the category recordset, there is one more thing to consider: the main categories cannot have ads, so you must filter the recordset so that records that have the **idcat_cat** field equal to the entered value **0** will not be retrieved, as you can see in the image below:



The screenshot shows the 'Recordset' dialog box in Microsoft Access. The 'Name' field is 'rsCategory', 'Connection' is 'connAds', and 'Table' is 'category_cat'. Under 'Columns', the 'All' radio button is selected, and the list contains 'id_cat', 'idcat_cat', and 'name_cat'. The 'Filter' is set to 'idcat_cat' with a '<>' operator. The 'Entered Value' is '0'. The 'Sort' is set to 'None' with an 'Ascending' order. On the right side, there are buttons for 'OK', 'Cancel', 'Test', 'Advanced..', and 'Help'.

- There are only three more fields left to configure: for the title, you can use the default settings, but for the content, change the form element from text-field into a text area.
- The last field, the **date_ads**, which stores the date when the ad was posted will not be visible in the page, as it will be automatically filled in. Change its associated form element to text, and enter in the **Default value** text field the following mark-up: `{NOW}`. This will be automatically replaced at runtime by the **Transaction Engine** with the current date in the correct format.

Note: If you are using a **Microsoft Access** database, in the **Submit as** drop-down menu, for the date_ads column, another option will be available: **Date MS Access**. Select this option when submitting the date.

- Now the **Insert Record Form Wizard** step 2 dialog box looks like:

Insert Record Form Wizard

Step 2/3: Configure the form fields

Set form fields properties

Form fields: + -

Column	Label	Display as	Submit as
idusr_ads	Idusr:	Text	Numeric
idtyp_ads	Ad type:	Menu	Numeric
idcat_ads	Ad category:	Menu	Numeric
idloc_ads	Ad location:	Menu	Numeric
title_ads	Title:	Text field	Text
content_ads	Content:	Text area	Text

Label:

Display as:

Submit as:

Default value:

Interakt

Enter the default value for the current form field.

< Back Next > Finish Cancel Help

- The third step of the wizard is only available if you have **MX Form Validation** installed, either as a separate product, or as a part of **MX Kollection**. It allows you to set rules for the form elements, submitting them only if they pass validation. You will not need validation on these fields, so you can safely skip this step.
- If you forgot something, or did it wrong, you can click on the **< Back / Next >** buttons to move through the wizard's steps. When you consider the configuration to be finished, click the **Finish** button to close the wizard.
- The wizard will add some server behaviors, for the transaction and the associated triggers, as well as all form elements required. After closing the wizard, remove the first row of the table, the one containing the idusr text. The **Dreamweaver** editable area will look as follows:

{Display error message.}

Idusr:	{text}
Ad type:	<input type="text"/> {Error}
Category:	<input type="text"/> {Error}
Location:	<input type="text"/> {Error}
Title:	{rsads_ads.title_ads} {Hint} {Error}
Content:	{text} {Hint} {Error}
Date:	{text}

Due to the **Restrict Access to page** server behavior you've added to the page in the first step, you must first login with the correct user name and password before accessing the actual insert form. After you log in however, open the post ad page, and complete it with some test data. Open the database server management software, and check if the submitted data has been correctly inserted into the database.

The next part shows you how to use **MX Includes** in order to improve your site.

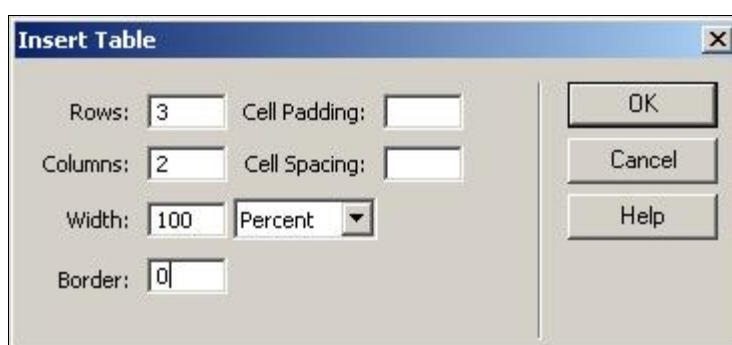
Create site front-end with MX Includes

In this tutorial chapter you will combine all elements created until now into a single page, allowing visitors accomplish every possible task in the shortest amount of time. To do so, you will use **MX Includes**, which allows adding other pages in a static or dynamic manner.

Start off by opening the *index* file in **Dreamweaver**. To decide what files will be included in the main page, you must decide what functionality this page will offer: you must have the category menu, the ad listing for the currently selected category, and a nugget that will allow registered users to login, and then have links towards the post page.

Once the files to include are set, the page design is also quite easy to figure out: a table with at least 3 rows and 2 columns, of which the second column will have only one row. To begin creating the page, follow the next steps:

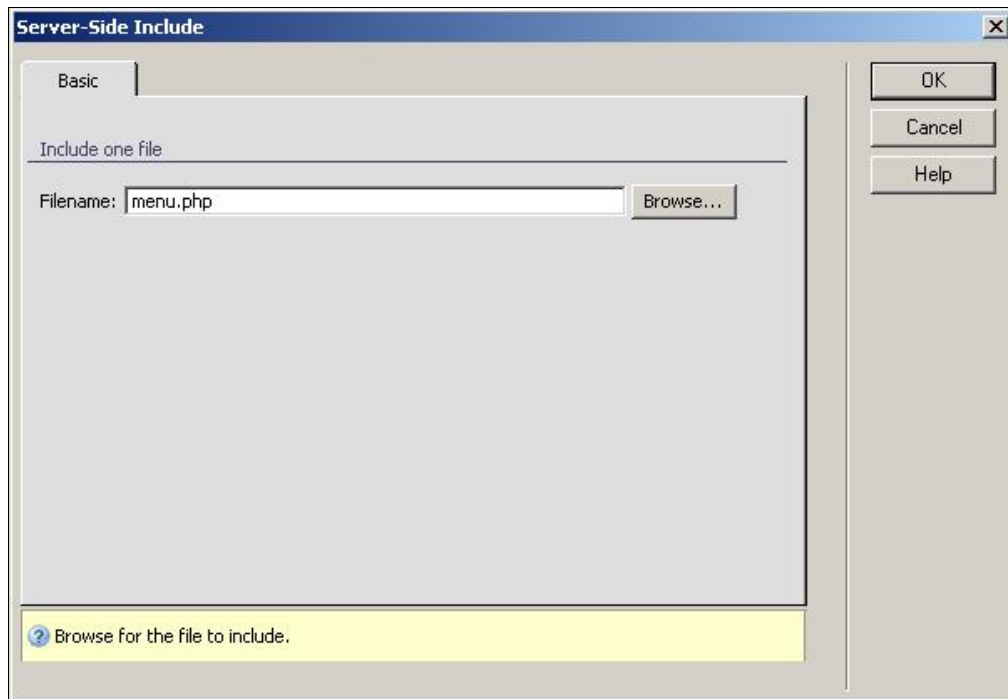
1. Add a new table into the page. Configure it to have 3 rows and 2 columns (let's add the site title as well):



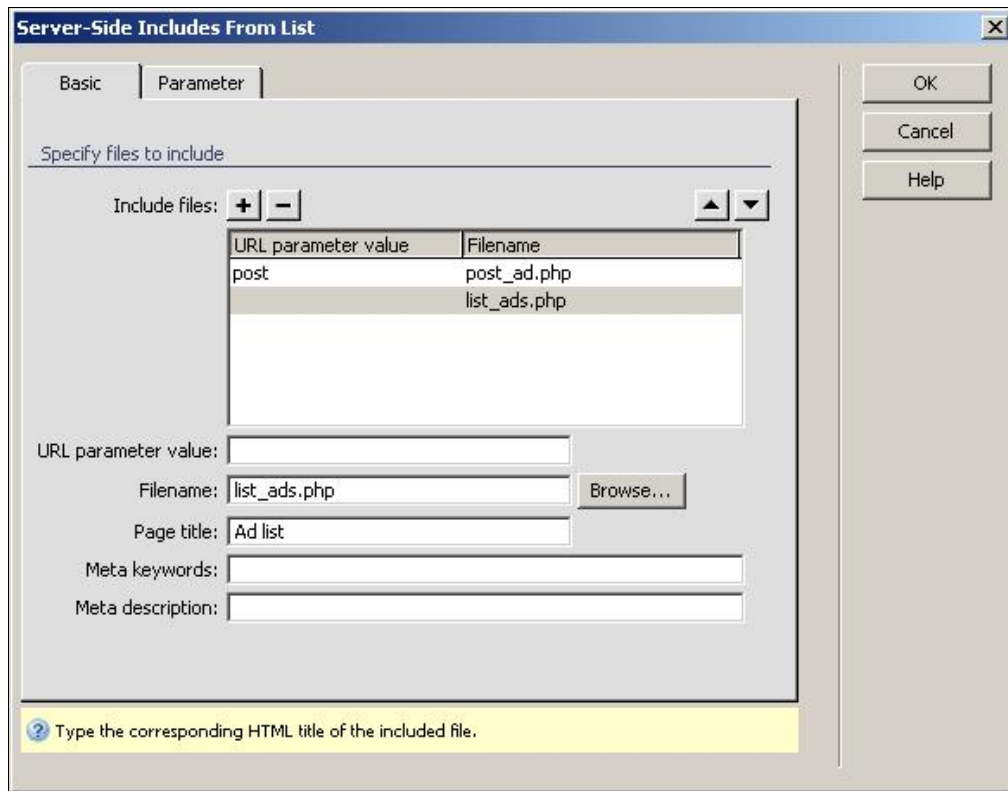
2. After this basic table is added to the page, merge the first row's columns. To do this, select the two columns, and select **Table -> Merge Cells** from the right-click pop-up menu. Now enter a site title into the first row: **Welcome to Classified Ads** for example.
3. Also, merge the second column's rows, in the same manner. Now the *index* page looks like the following image in **Dreamweaver**:



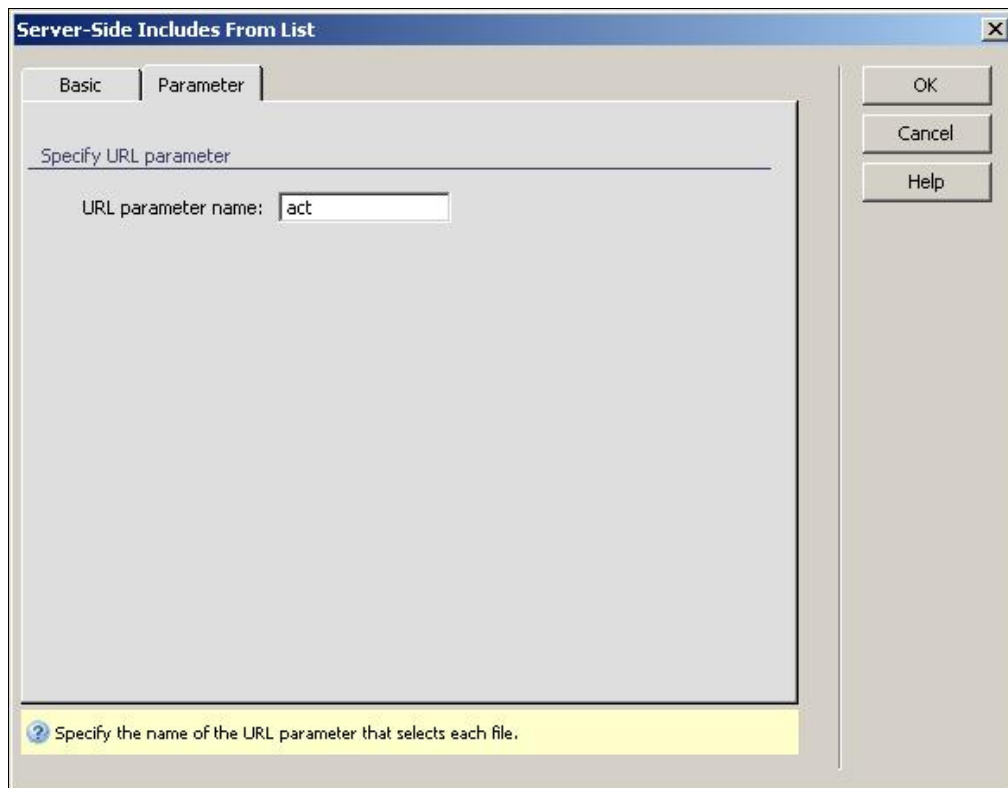
4. Now it is time to start adding content, by including the correct files. The first one is the tree menu created in the previous section. It will reside in the second row's first cell.
5. Place the cursor in the cell, and apply the **Server-Side Include** server behavior. You can access this server behavior from **Server Behaviors -> + -> MX Kollection -> Server-Side Includes**. Configure the dialog box that opens by browsing to the *menu* file. Then click on **OK** to apply the server behavior:



6. Next comes the login/user menu file. It will be displayed as a nugget in the third row's first cell, beneath the menu. This file is also statically included, so the steps to take are the same, the only difference residing in the file you have to select in the **Server-Side Include** dialog box: *login*.
7. The last thing you must do in regards to the *index* page is to define the files that will be included in the second column's cell. This will be a dynamic include, as this will display the ad listing when a category is selected, but also the post ad form when the **Post Ad** link is clicked. To determine which page will be displayed, an URL parameter is used: **act**. Depending on its value, a different page will be loaded.
8. To achieve this, you will use the **Server-Side Includes From List** server behavior, that can be accessed from **Server Behaviors -> + -> MX Kollection -> Server-Side Includes**.
 - To configure the **Basic** tab, consult the following image:



- When configuring the **Parameter** tab, set the name of the URL parameter to **act**:



9. This way, when the **act** parameter is missing, the ad listing will be displayed. The tree menu's links do not use the **act** URL parameter, and as such the correct ad list will be displayed. The **Post ad** link from the nugget does use it however, giving it the post value, and as such, the form will be displayed.

Improve the Classified Ads Site

In this part of the tutorial you will bring several improvements to the basic classified site built earlier. This improvements will:

- Allow the site visitors to search for a particular ad.
- Allow the registered users to subscribe to receive new ads on e-mail.
- Add a breadcrumb navigation trail to the page, allowing visitors to jump between categories faster.

In the first part of this tutorial section, you will create an advanced search form, allowing users find the desired ad faster.

If you have the **MX Kollection 3** bundle installed, then you have all the needed tools. Otherwise, the following separate products should be installed on your computer in order to complete this tutorial section:

- ImpAKT
- MX Send E-mail
- MX Query Builder
- MX Breadcrumbs

The estimated completion time for this section is about 20 - 30 minutes. It depends on your authoring knowledge with **Macromedia Dreamweaver** (MX or MX 2004) and **MX Kollection 3**.

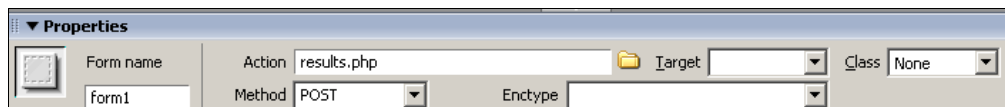
Create advanced search form

In this tutorial section you will add the first improvement to the classified ads site: a way for users to search for ads matching some criteria: the ad type, category and location, as well as some keywords. The keywords are searched both in the ad title and content.

As for the Job site search improvement, you will use two pages for the search: one where the user can select the values of each criteria, and the actual results page, displaying the ad title, and a link to see more.

To create the first part of this enhancement, the *search* form page, follow the next steps:

1. Create a new page in the site root, *search.php*. Open it in **Dreamweaver**.
2. Start the **Custom Form Wizard** from the **MX Kollection** tab of the **Insert** panel. Configure it to use the database connection created at the beginning of this tutorial. You don't need to set a redirect page, as it will not be used in this example.
3. In the second step of the wizard, you can add fields that will be displayed in the page. This is actually a much easier way of adding form elements, as it allows you to manage them in a central place. Also, you can define a SQL query to use them in the wizard's last step. Add the following fields: **Type**, **Category**, **Location** and **Keywords**. For the first three, set them to be displayed as menus, and configure the menus to retrieve data from the respective tables. For an example of how to do that, see the Post Ad section.
4. When all fields are set, you can click the **Finish** button to close the wizard.
5. Now edit the form's submit button, by changing its label from Insert Record to Search, as it is more appropriate.
6. The last thing to do on this page is to set the form's action to point to the *results.php* page. Select the form tag, and in the **Action** text-field of the **Property Inspector** type the name of this page (which will soon be created).



7. Now, when viewed in Dreamweaver, the form looks like:

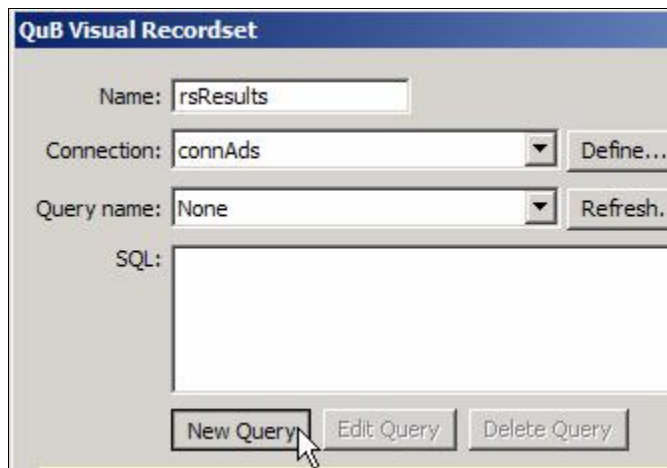
The screenshot shows a web form with the following elements:

- A placeholder for an error message: `{Display error message.}`
- A 'Type:' dropdown menu with a placeholder `{Error}`.
- A 'Category:' dropdown menu with a placeholder `{Error}`.
- A 'Location:' dropdown menu with a placeholder `{Error}`.
- A 'Keywords:' text input field with a placeholder `{text}`, a hint `{Hint}`, and an error placeholder `{Error}`.
- A 'Search' button at the bottom right.

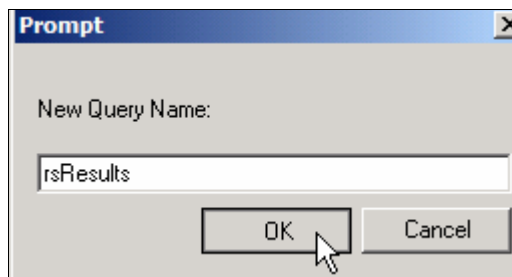
8. Save the page and upload it to the server.

Now you must build the *results* page, which will take the criteria submitted by the form, and retrieve the data that matches from the ads table. The data retrieval is done by a filtered recordset. To create this page follow the next steps:

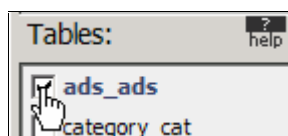
1. First create the *result.php* file in the site's root folder. Open it in **Dreamweaver**.
2. Add a new recordset from the **Bindings** tab, by pressing on the **Plus (+)** button and selecting the **Recordset (Query)** option of the pop-up menu.
3. Switch to the **QuB3** view of the dialog box by clicking the button on the right.
4. A window like the one below will appear. Configure it as shown, then click New Query:



5. Enter a name for the query in the prompt:



6. Now the **MX Query Builder** window will open in a new window. If you're not already familiar with the interface, you may want to give the QuB Workspace orientation page a quick look.
7. In the **Tables** panel, check the box next to the *ads_ads* table.



8. This puts the table in the **Database Diagram** panel. Now check the boxes next the following fields:
 - ***id_ads***
 - ***idtyp_ads***
 - ***idcat_ads***
 - ***idloc_ads***
 - ***title_ads***
 - ***content_ads***

ads_ads		
<input type="checkbox"/>	*	
<input checked="" type="checkbox"/>	id_ads	int (11)
<input type="checkbox"/>	idusr_ads	int (11)
<input checked="" type="checkbox"/>	idtyp_ads	int (11)
<input checked="" type="checkbox"/>	idcat_ads	tinyint (4)
<input checked="" type="checkbox"/>	title_ads	varchar (255)
<input checked="" type="checkbox"/>	content_ads	varchar
<input type="checkbox"/>	idloc_ads	int (11)
<input type="checkbox"/>	date_ads	date

9. These fields will then be listed below in the **Query Management** panel. This is where you set the conditions necessary for obtaining the information requested by the user. Each field has a column labeled **Condition**:

Column Name	Alias	Sort	Output	Grouping	Condition		
ads_ads.id_ads			<input checked="" type="checkbox"/>			...	v ^ Del
ads_ads.idtyp_ads			<input checked="" type="checkbox"/>			...	v ^ Del
ads_ads.idcat_ads			<input checked="" type="checkbox"/>			...	v ^ Del
ads_ads.content_ads			<input checked="" type="checkbox"/>			...	v ^ Del
ads_ads.title_ads			<input checked="" type="checkbox"/>			...	v ^ Del

10. With the exception of the id_ads field, each row will need to have a condition associated with it. Starting with the **idtyp_ads** field, click the ... button in the **Condition** column. Fill in the fields as shown in the image below, then click OK.

QuB :: Edit SQL Condition -- Web Page Dialog

Edit SQL Condition

Table Column: (idtyp_ads)

Condition: =

Variable Type: Form Variable

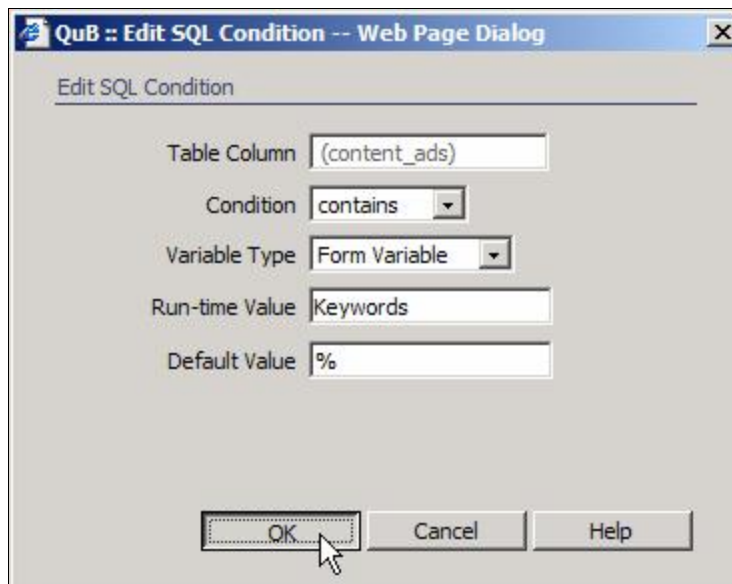
Run-time Value: Type

Default Value: -1

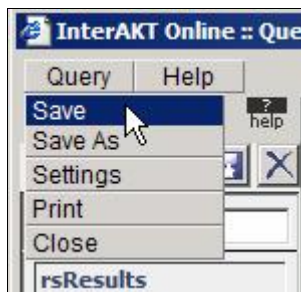
OK Cancel Help

11. Repeat step 10 for the **idcat_ads** and **idloc_ads** fields, only using "Category", respectively "Location" as the **Run-time Value**.

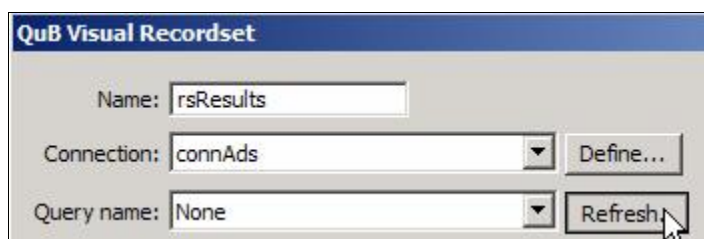
12. Now you will add conditions for both the **title_ads**, and **content_ads** fields, but with different settings for the condition. Since the comparison is between strings, rather than numbers, use the contains option for the Condition. You will also use "Keywords" as the Run-time Value, and "%" for the Default Value, as shown below:



13. Once all the conditions have been added, save the query by going to the **Query** menu and selecting "Save":



14. Close the **MX Query Builder** interface and go back into **Dreamweaver**.
15. Click Refresh next to the **Query name** field in the **QuB Visual Recordset** window.



16. Now the completed recordset from **MX Query Builder** will appear in the list. At this point there's one slight change to make in the query. In the text area containing the SQL code, scroll down to the last line.

QuB Visual Recordset

Name:

Connection: Define...

Query name: Refresh..

SQL: `AND ads_ads.idcat_ads='.$ POST["Category"].'`
`AND ads_ads.content_ads LIKE '%'.$ POST["Keywords"].'%'`
`AND ads_ads.title_ads LIKE '%'.$ POST["Keywords"].'%'`

17. Change the final `AND` to `OR`. You must also put an opening parenthesis after the last "AND", and a closing parenthesis and the end of the query. See the image below for how the query should look after the change, then click **OK**.

QuB Visual Recordset

Name:

Connection: Define...

Query name: Refresh..

SQL: `AND ads_ads.idcat_ads='.$ POST["Category"].'`
`AND (ads_ads.content_ads LIKE '%'.$ POST["Keywords"].'%'`
`OR ads_ads.title_ads LIKE '%'.$ POST["Keywords"].'%')`

18. At this point, the recordset retrieves the correct data, but there are no elements to display it. Since this is only a results page, you do not have to display all details of an ad, but only its title, and a link to a detail page. You can use a dynamic table to show results.
19. Place a dynamic table from the **Application** tab of the **Insert** panel. Configure it to display all records of the *rsResults* recordset. Once you've added the table, it will display all recordset columns. Except for the *title_ads* column, delete the rest. Also remove the header row for the title column.
20. To place the link to details for each ad, create a new column after the *title_ads* one. Enter the text **Read more**, and make it a link. In the Make Link dialog box, browse to the *detail_ad.php* page, and pass it an URL parameter. Name the parameter *id_ads*, and set its value by pressing the dynamic data icon, and choosing the *id_ads* recordset field:

Parameters

Name	Value
id_ads	<?php echo \$row_rsResults[id...

21. Now the page looks like the following when viewed in **Dreamweaver**:



22. To make the results page easier to understand, create a new paragraph before the dynamic table. This is where the page title, as well as the keywords used in the search will be displayed.

23. Type in "Search results for:". Switch to code view, and enter the following code:

- For PHP

```
<?php echo $_POST['Keywords']; ?>
```

- For ColdFusion

```
<cfoutput>#FORM.Keywords#</cfoutput>
```

- For ASP VbScript

```
<% =request("Keywords") %>
```

You can now save the pages and upload them to the server. Before calling it quits, though, you should place an entry in the *login.php* file that will point to the *search* page and create the detail page. This way it will be much easier to find. If you access the page, you can enter the criteria you want in the form, and when you press **Search**, the results will be displayed:

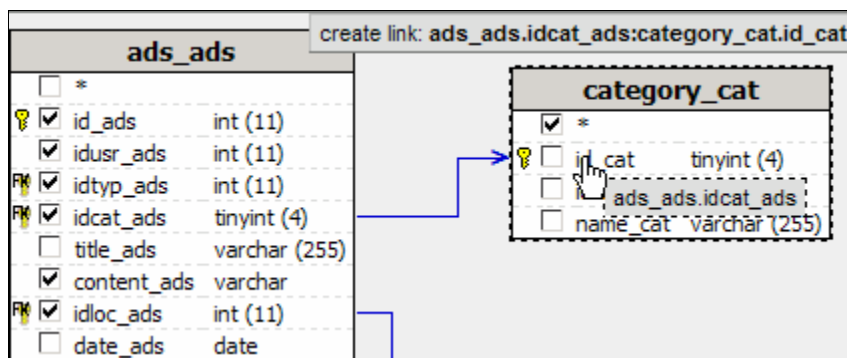
Type:	For Sale
Category:	Cars
Location:	Detroit
Keywords:	BMW
Search	

After performing the search, the results are displayed as in the following image:

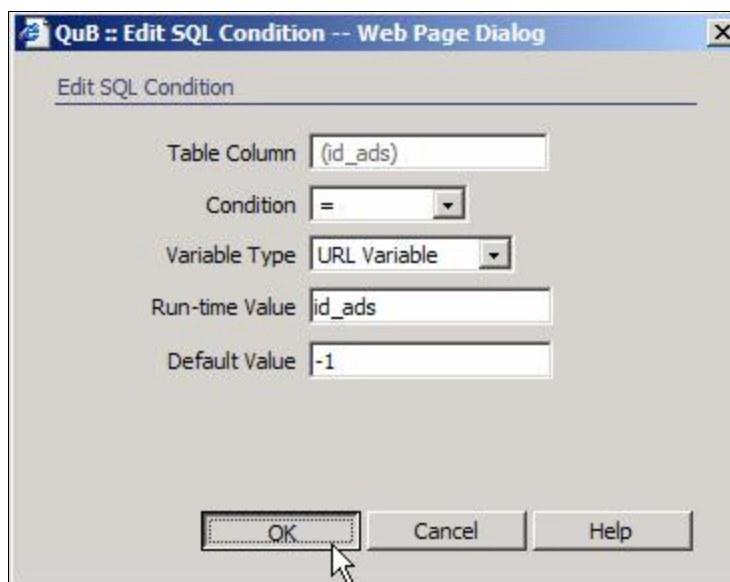
Search results for: BMW	
1992 BMW Cabrio	Read more
2004 BMW 325i	Read more

The last thing to do is create the detail page, which opens when the Read more link is clicked. This page will only display the ad that has been selected, and whose ID has been passed through the *id_ads* URL parameter. To build the detail page, follow these steps:

1. Open the *detail_ad* page in Dreamweaver. For this page, you will be creating a query in MX Query Builder, using steps similar to those used to build the *rsResults* recordset.
2. Create a new recordset, from the Bindings tab of the Application Panel. Name it *rsDetail*, and select as connection the *connAds* database connection.
3. Switch to the **QuB3** view by clicking on the **QuB3** button on the right.
4. Click the **New Query** button on the bottom, and enter *rsDetail* in the name prompt.
5. After the **MX Query Builder** interface opens, go to the **Tables** panel on the left. Select all tables except for the *subscribe_sub* table.
6. With the tables now in the **Database Diagram**, you can move them around with the mouse so that all tables are visible.
7. Once the tables are positioned, add relations between them by dragging the foreign keys from the *ads_ads* table, and dropping them on their corresponding primary key. Take a look at the image below for an example:



8. To get the fields required for this query, check the following boxes from the *ads_ads* table.
 - ***id_ads***
 - ***idusr_ads***
 - ***idtyp_ads***
 - ***idcat_ads***
 - ***title_ads***
 - ***content_ads***
 - ***idloc_ads***
 - ***name_cat***
 - ***name_loc***
 - ***name_typ***
9. With the above fields now listed in the Query Management panel, add a URL parameter condition to ***id_ads*** field as done in steps 9 & 10 of the *rsResults* query. Configure the Condition as shown below:



10. Now repeat steps 13 - 15 from the *rsResults* query to be able to use the new query in **Dreamweaver**.
11. The new SQL code should now appear in the **QuB Visual Recordset** window. Click **OK**.
12. Now close the recordset creation dialog box by clicking the **OK** button.
13. From the **Bindings tab** drag and drop all of the fields that you want displayed onto the page. You can also arrange the fields any way you want them.
14. Once you're done with the table column display, save the page and upload it to the server. You can now try out the ad search.

You can further improve your site using the **MX Kollection 3** suite, by adding an administrative section, where you can control which ads can be published on the web site, which users to accept or to promote as administrators, and a better ad detail page for the search results. And all these can be made with your own design.

Register to receive ads by e-mail

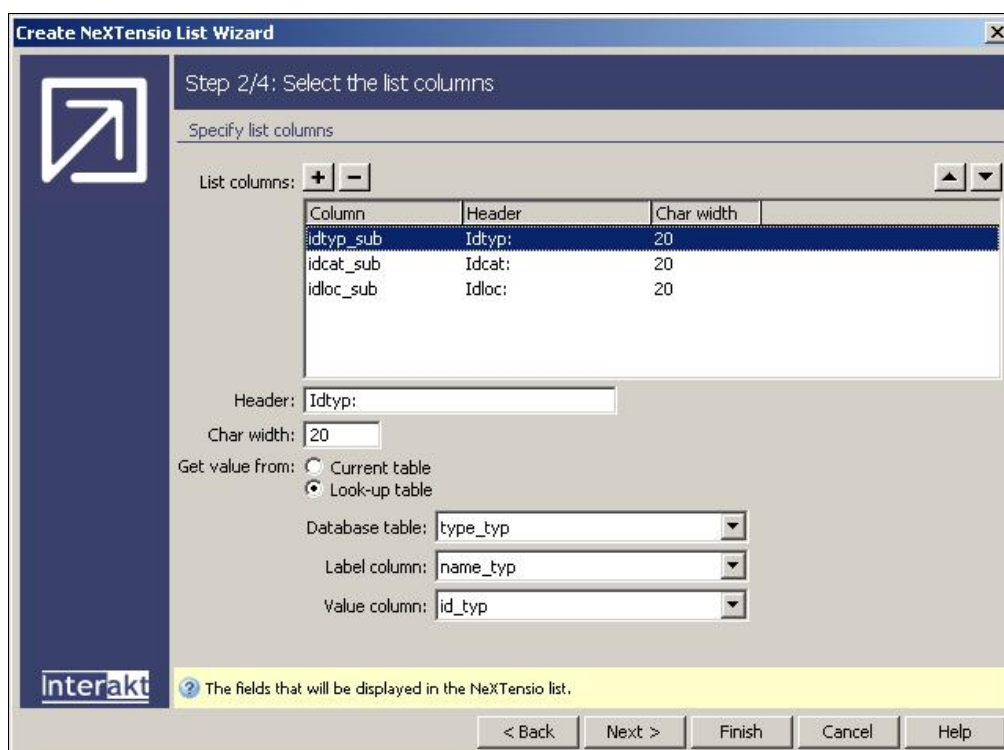
In this tutorial section you will enhance the Classified Ads application by allowing users subscribe to receive email notifications when certain combinations of ad type, category and location are met. This enhancement contains two pages:

1. *view_subscription.php* - where a list with the existing subscriptions is shown. Links to add, edit or delete a subscription are displayed on the page.
2. *subscription.php* - where a form allows adding or editing subscriptions.

Create these two files in the site's root folder. To implement this enhancement, you will use the NeXTensio list and form wizard.

First off, let's create the subscription list page:

1. Open the *view_subscription.php* file in **Dreamweaver**. To ensure that unregistered users cannot nose into other's subscriptions, make sure the **Restrict Access To Page** server behavior was applied, as shown here.
2. Next start the **NeXTensio List Wizard**, to display the subscriptions. Configure each step as shown below:
 - In the first step, simply select the database connection created at the beginning of this tutorial, *connAds*, and the table to list: *subscribe_sub*. The primary key field is auto-detected, and can be left at its default. Also, set the redirect page to the *subscription* file
 - In the second step you must define which table columns will be displayed and how. By default, all fields except the **Primary Key** are displayed in the List Fields grid. You should remove the *idusr_sub* field too, as the only user that would appear is the currently logged in one.
 - For the *idtyp_sub*, *idcat_sub* and *idloc_sub* fields, use a look-up table to retrieve data. For each field select its corresponding table, and the appropriate label and value columns from the appropriate drop-down menus (e.g. for the *idtyp_sub* use the *type_typ* table, with the *name_typ* column for label, and the primary key of the table - *id_typ* - for the value):



- Step 3 of the wizard allows setting up the form elements used in the list filter. Since each field is based on a look-up table, the corresponding filter element will be a menu which retrieves its data from the tables configured above.

- In the wizard's last step, configure the list options: the page containing the add/edit form (*subscription.php*), the number of records, and the table column to sort the list after:

- Now the HTML elements, as well as the NeXTensio recordset and server behaviors are added to the page. The only problem of this list is the fact that, at this point, it displays the subscriptions for all users. To restrict it to the current user only, you have to modify the recordset, and add a condition.

Note: If using ColdFusion, you can skip to step 3.

- To do so, double-click the list recordset in the Bindings tab. Add a new variable, called user, having the runtime value equal to `$_SESSION['kt_login_id']`. To add a variable, click the **Plus (+)** button on top of the **Variables** grid, and set its properties in the dialog box that opens:

- Now edit the SQL, to add the condition of equality between the table column storing the user id, and the newly defined variable. The new code to enter in the SQL text area is:

```
SELECT type_typ.name_typ AS idtyp_sub, category_cat.name_cat AS
idcat_sub, location_loc.name_loc AS idloc_sub, subscribe_sub.id_sub
FROM subscribe_sub LEFT JOIN type_typ ON subscribe_sub.idtyp_sub =
type_typ.id_typ LEFT JOIN category_cat ON subscribe_sub.idcat_sub =
category_cat.id_cat LEFT JOIN location_loc ON subscribe_sub.idloc_sub
= location_loc.id_loc
WHERE NXTfilter AND subscribe_sub.idusr_sub =user
ORDER BY NXTSort
```

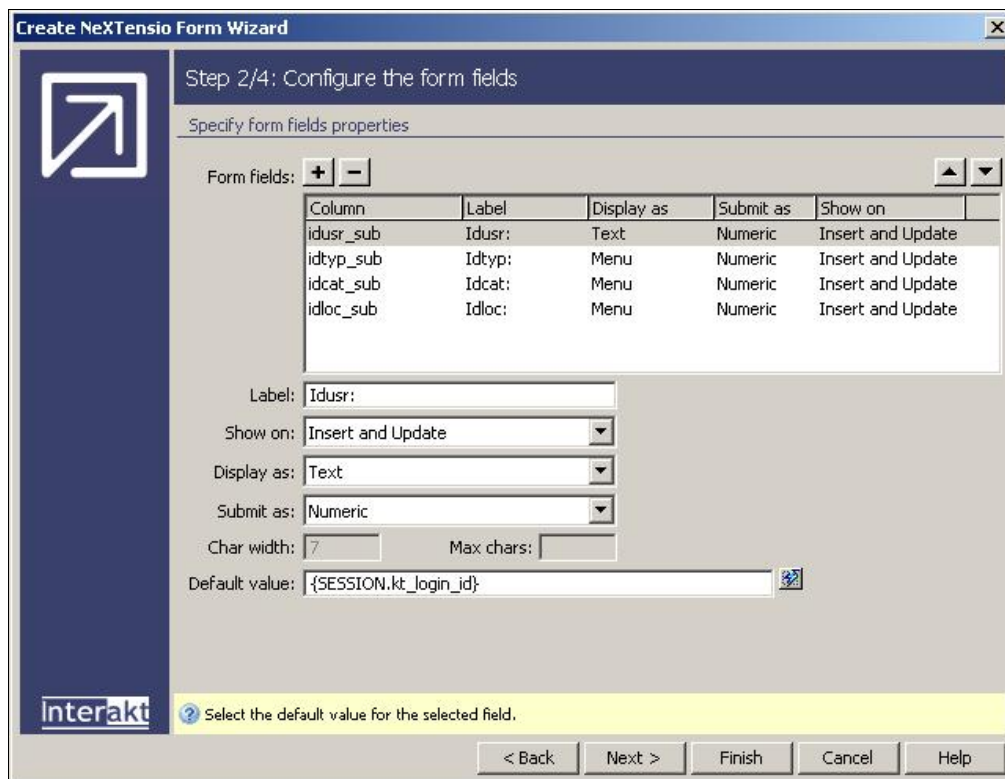
3. If using ColdFusion, you do not need to add the page parameter. Simply add a new condition to the query, which will check the **idusr_sub** column against the *kt_login_id* session variable. Add the following condition after the existing **WHERE** clause, and before the **ORDER BY** clause:

```
AND idusr_sub = #SESSION.kt_login_id#
```

4. The query should look like the following: (page parameter names may vary, based on your particular setup):

```
SELECT type_typ.name_typ AS idtyp_sub, category_cat.name_cat AS
idcat_sub, location_loc.name_loc AS idloc_sub, subscribe_sub.id_sub
FROM ((subscribe_sub LEFT JOIN type_typ ON subscribe_sub.idtyp_sub =
type_typ.id_typ) LEFT JOIN category_cat ON subscribe_sub.idcat_sub =
category_cat.id_cat) LEFT JOIN location_loc ON subscribe_sub.idloc_sub
= location_loc.id_loc
WHERE #PreserveSingleQuotes(SESSION.filter_tfi_listsubscribe_sub1)#
AND idusr_sub = #SESSION.kt_login_id#
ORDER BY #SESSION.sorter_tso_listsubscribe_sub1#
```

5. Simply replace the existing SQL code with this one, and the recordset will filter out any other users than the one that is currently visiting the page.
6. Now all operations on this page are done, and you can save and upload it to the server, and move on to the form.
7. To create the form, you must open the *subscription.php* file in **Dreamweaver**. The page must have the same name as the one specified in the list wizard last step, or the links will not work correctly. This page will contain a NeXTensio form, which will be added and configured through the **NeXTensio Form Wizard**.
8. Open the wizard from the **MX Kollection tab** of the **Insert Panel**, and configure each step as shown below:
 - In the first step select the same connection and table as for the **List wizard**. Since you've applied the List Wizard first, the **UI persistence** should have completed all fields.
 - In the second step, all fields are correctly configured by the persistence. If not in the grid, you will have to add the **idusr_sub** field. Set it as text with the default value of the session variable **kt_login_id**:



- The third step is available only if you have **MX Form Validation** installed, and provides a way to define validation rules for all fields involved in the form. Since all fields are menus, you do not need validation and can just click the **Finish** button to add all elements into the page.
9. The **NeXTensio** form elements, recordset and server behaviors are now inserted into the page. Delete the first row from the table, the one containing the `idusr_sub` text. All that remains is to save the page and upload it to the server. This way, new users can define subscriptions for different types of ads easily.

The last part of the tutorial is an improvement to be made to the ad posting page. The subscription list is created and managed by the users, but you have to implement the actual sending of the emails.

For this enhancement, you will use **MX Send E-mail**, on the `post_ad.php` page. To modify the page, follow the next steps:

1. Open the `post_ad.php` page in **Dreamweaver**. On the page there is already an **Insert Transaction**, that allows posting new ads.
2. To send the email to all users that are subscribed to the type, category and location of the ad that's being posted, you must add two things: a recordset to retrieve their email addresses, and a **Send email trigger**.
3. First the recordset: it must be filtered, since the user's preferred ad type, category and location must match the ones entered in the post ad form. Create a new recordset from the **Bindings panel**, and when the dialog box opens, switch to the advanced view.
4. In the **Advanced view**, select the database connection created at the beginning of the tutorial.
5. Create three new variables, by clicking the **Plus (+)** button on top of the variables grid. Configure them as follows:
 - Name: **type**; Default value: -1; Runtime value: `$_POST['idtyp_ads']`.
 - Name: **category**; Default value: -1; Runtime value: `$_POST['idcat_ads']`.
 - Name: **location**; Default value: -1; Runtime value: `$_POST['idloc_ads']`.

- When the form is submitted, these variables will retrieve the values of their corresponding form elements, and will be used to compare against the fields in the **subscribe_sub** table. The comparison is done by the SQL query you must enter in the **SQL** text area:

```
SELECT *  
FROM subscribe_sub LEFT JOIN user_usr ON  
subscribe_sub.idusr_sub=user_usr.id_usr  
WHERE idtyp_sub=type AND idcat_sub=category AND idloc_sub=location
```

- As you may notice, aside from comparing the type, category and location, this SQL code performs a **JOIN** operation between the subscription table and the user table, in order to retrieve the e-mail address for each one. The recordset creation is now complete, and you can close the dialog box by clicking the **OK** button.
- Now for the actual sending of the mail messages you will use a trigger provided by **MX Send E-mail: Send E-mail to recipients from recordset**. This trigger will send a mail message to all mail addresses retrieved by a recordset, in our case the *rsMailList* one.
- Before applying the server behavior, you should check out the mail server settings in **Control Panel**, to see if they are correct in your case. To learn more about the mail server settings, click here.
- Apply the **Send E-mail** to recipients from recordset server behavior from the **Server Behaviors tab -> MX Kollection -> Send E-mail**. From the dialog box that opens, you only need to configure the options on the first tab. Set them as follows:
 - In the **Recordset** drop-down menu select *rsMailList*.
 - In the **Email to field** drop-down menu select the recordset field containing the user's mail address.
 - In the **From** text box you can enter an email address specific for your site, or leave it at default, to use the control panel defined sender.
 - In the **Subject** text-box enter the mail's subject: New ad posted.
 - For the mail's content, you have two choices: to use a predefined template, or to write the content in the text area below. Select the latter version, as there is no need for a special template.
 - In the **Content** text area enter the mail text. Usually, a notice that a new ad that matches what the user subscribed for has been added, and a link to the site is enough.
 - The completed dialog box should look like the image below:

- Once all dialog box required options are completed, simply click the **OK** button to add the server behavior into the page.

Note: If using the **ASP VBScript** server model you must configure the E-mail server settings in the Control Panel. You must fill in the server address (or name), the port (by default it is 25), user name and password. Optionally, you can also set the default sender field.

Save the page and upload it to the server. When a user will post an ad, all those that are subscribed to the same ad type, category and location will receive an email message warning them of the new post.

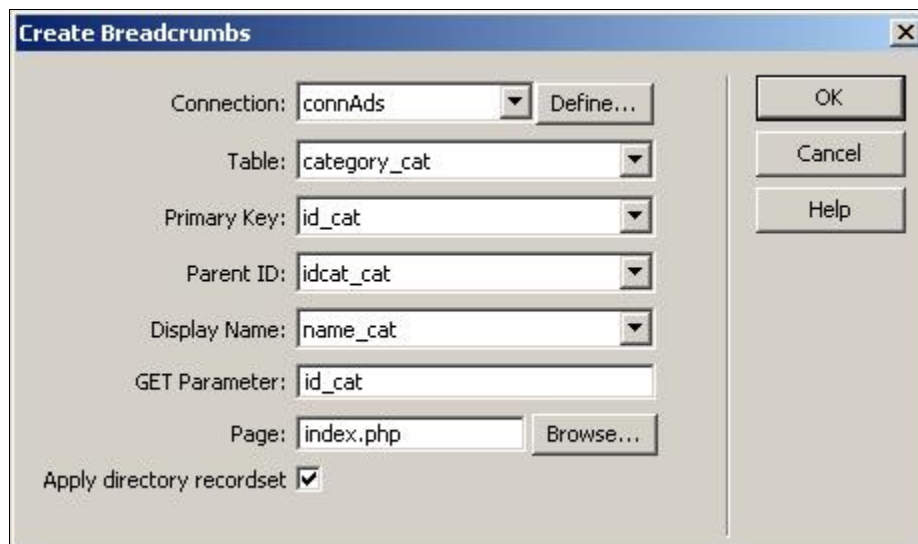
Add breadcrumbs navigation trail

One way to improve navigation in a dynamic content web site is to always allow the visitor retrace his steps. This is accomplished by placing at the top of the page a breadcrumb navigation trail. This is possible for this site, as it displays the ads in different categories on the same page, by using the category ID as URL parameter.

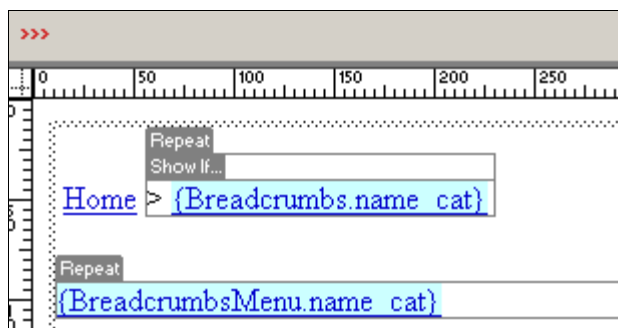
To be able to follow this section of the tutorial, you need to have downloaded and installed **MX Breadcrumbs**. This is a free **Dreamweaver** extension, which you can download [here](#).

Once you've got the extension installed, and you've restarted **Dreamweaver**, you can begin enhancing the site's main page. To do so, follow the next steps:

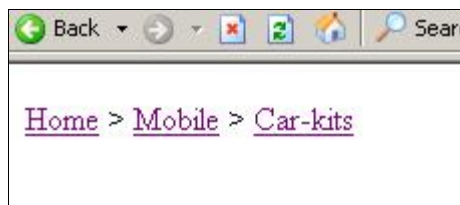
1. Open the *index.php* file from the site's root folder in **Dreamweaver**. As you've seen when the page was created, it contains a table where all data is stored.
2. The breadcrumb navigation trail should be placed above all other elements, welcome message included. So the simplest way of doing it is to add a new row at top of the existing table.
3. To add a new row, simply right-click in the first row and select **Table -> Add rows and columns** from the pop-up menu. In the user interface that opens, select to add a new row, above the current selection.
4. Now click in the newly created row, and apply the **Create Breadcrumbs** command from the **InterAKT** tab of the **Insert** panel.
5. Configure the dialog box that opens as follows:
 - In the **Connection** drop-down menu, select the database connection you've created at the beginning of this tutorial: *connAds*.
 - In the **Table** drop-down menu, select the database table storing the data used to create the breadcrumbs. This should be the same as the one providing the URL parameter, so that navigation will really work. In this case, the table is *category_cat* (the URL parameter is the category id, after all).
 - In the Primary key drop-down menu, select the table's field containing the primary key. By default, the first field found in the table is selected, but you can change it easily.
 - In the **Parent ID** drop-down menu, select the table field containing the self foreign key, which points to the upper category: *idcat_cat*.
 - In the **Display name** drop-down menu, select the table field storing the labels that each breadcrumb element will display. For the categories, it is *name_cat*.
 - In the **GET Parameter** text field you must enter the name of the URL parameter that must be transmitted to the page, so that it will load the appropriate information. Enter *id_cat*, as this is used in the *index* page to select the ads to display.
 - In the **Page** text field, enter the page that receives the URL parameter and shows the ads. You can use the **Browse...** button to search for the *index.php* file.
 - The completed dialog box looks like in the image below:



- Once you've completely configured the user interface, click the **OK** button to apply the command, and insert all the elements into the page.
6. Some page elements are added by the command in the first row, as well as some server behaviors. In **Dreamweaver**, the first row now contains:



This is all that you have to do in order to add a breadcrumb navigation trail to your site. Just save the page, and preview it in the browser by pressing the **F12** key, and you will notice on top of the page the trail that allows moving through the site in an easier manner:



Now that you have an improved version of the classified ads site, you can move on and implement new ways to browse the ads (sorted by different criteria), an administrative section (where you can control what ads are to be shown) or an expire date for each ad.

Other Resources

Other Dreamweaver Extensions from InterAKT

- KTML
- MX Kart
- MX Site Search
- MX RSS Reader-Writer
- MX Dynamic Table Sorter
- MX Coder Pack
- MX Dynamic Charts
- MX CSS Dynamic Menus

Contact Us

- **Address:**
1-11 Economu Cezarescu ST, AYASH Center, 1st floor
Sector 6, ZIP 060754, Bucharest, Romania
- **Web:** <http://www.interaktonline.com/>
- **E-mail:** contact@interaktonline.com
- **Phone:** +4031 401.68.19 or +4021 312.51.91
- **Fax:** +4021 312.53.12

Copyright

Windows is a trademark of Microsoft, Inc.

Dreamweaver MX is a trademark of Macromedia, Inc.

Redhat is a trademark of Redhat, Inc.

Copyrights and Trademarks

Copyright 2000-2005 by InterAKT Online.

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

PHAkt, ImpAKT, NeXTensio, MX Query Builder, Transaction Engine, MX Includes, KHTML, MX Kommerce, MX Kollection, MX Widgets, MX Looper, MX Dynamic Charts, MX CSS Dynamic Menus, MX Tree Menu, MX Form Validation, MX File Upload, MX Send E-mail, MX User Login, MX CSV Import-Export, MX Kart, MX Site Search, MX Dynamic Table Sorter, MX RSS Reader-Writer, MX Coder Pack, MX Dynamic Charts are trademarks of InterAKT Online.

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



InterAKT Online

Web: <http://www.interaktonline.com/>

E-mail: contact@interaktonline.com

Address: 1-11 Economu Cezarescu ST, AYASH Center, 1st floor, Sector 6, ZIP 060754, Bucharest, Romania

Phone: +4021 312.51.91

Fax: +4021 312.53.12