

# Incorporating Advertising with Ad Server

---

Displaying ads on a web site is a common practice. Besides generating additional revenue from the simple display of the business ad of another company, you can also target your customers and display ads or promotions regarding your own business. Couple this capability with customer personalization, which facilitates the storage of pertinent customer information, and you can target ads to specific customers. For example, a customer who has attended systems administration classes at UCI will most likely not be interested in a Visual Studio 6 upgrade class. However, a student that has attended Visual C++ version 5 training would be a good candidate. The ad server component of the Commerce Server software makes the ad publication process very simple.

## Ad Server Features

The features of the Ad Server component of Commerce Server aid greatly in the design of a sophisticated web site that is appealing to both customers and potential advertising clients. These features include:

- The ability to display complex ads that include text, sound, and animations.
- Click-through ads; a customer clicking on the ad is taken directly to the advertising company's web site.
- Setting a limit on the number of times an ad appears. This allows advertising clients to pay for a fixed number of ad displays.

## 154 Chapter 6 • Incorporating Advertising with Ad Server

- The ability to control whether an ad should be displayed more than once during a customer session. Arguments can be made for and against this. Customers will probably not forget about an ad while still connected to the web site. However, the more times customers see an ad, the more likely they may be to access the advertised site.
- Personalizing the ad process by displaying ads that match target customer profiles, such as a bicyclist or rock climber.
- Tracking and reporting the ad process.

### An Advertising Scenario

UCI Corporation, our sample software training company, would like to start an advertising campaign. The first ad that is to be displayed will be directed to all potential customers in that it offers a blanket \$50 per day off of any training class. The marketing department has supplied a graphics file to use in the ad.

The first step in implementing the advertising process is to mark the root directory for the web site as the root directory of an ASP application. We use the Internet Service Manager tools to modify the properties for the site.

In Internet Service Manager, open the properties page for the target web site. In the directory tab, enter a name for the application. This name will be used to configure the ad object in the `global.asa` file. See Figure 6-1.

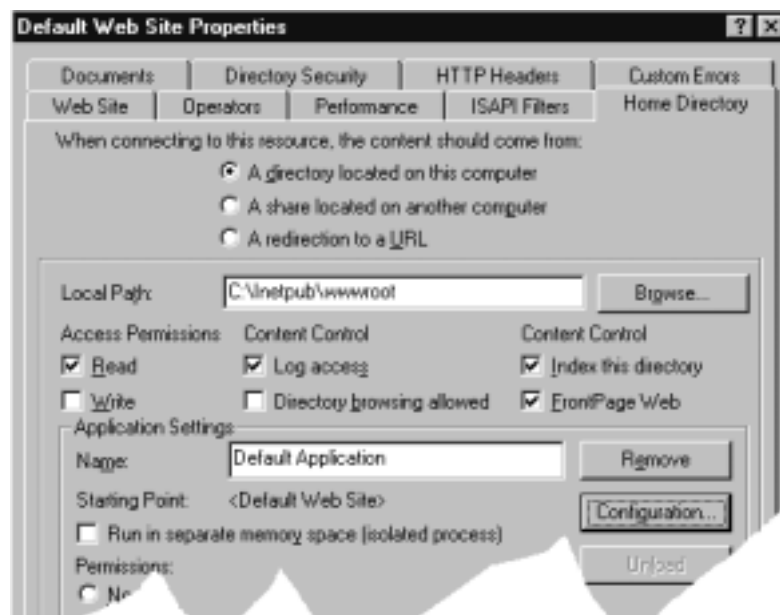


FIGURE 6-1

Setting the ASP Application root directory in Internet Service Manager

### ***Modifications to the global.asa File***

Several changes must be made to the `global.asa` file to support Ad Server. First, we must create an instance of an Ad Server object. We do this by adding the following lines.

```
Set Ad = Server.CreateObject("Commerce.AdServer")
Set Application("Ad") = Ad
```

The first statement creates an instance of an `AdServer` object named `Ad`. This object is then stored in the `Application` object in the variable `Ad`, which will have global scope, thus making it available to every ASP file in the application.

The second modification we make to the `global.asa` file is to specify the connection string for the Ad database. This can be accomplished in one of two ways. The first method is to hardcode the connection string information directly in the `global.asa` file as follows:

```
Ad.ConnectionString = "DSN=Training; UID=sa; PWD="
```

The risk here is that anyone who can read the `global.asa` file now has a user name and password to connect to the database. A more secure method of establishing the connection string is to pull the information from the `site.csc` file. The connection information is written to this file during the site creation process. Use the following statements to extract the connection string:

```
pathConfig = Server.MapPath("config/site.csc")
set dictConfig = Server.CreateObject("Commerce.Dictionary")
set fileDoc = Server.CreateObject("Commerce.FileDocument")
Call fileDoc.ReadDictionaryFromFile(pathConfig, "ASConfig", dictConfig)
Ad.ConnectionString = dictConfig.MASConnectionString
```

A dictionary object named `fileDoc` is first created. Then, the `ReadDictionaryFromFile` method is invoked. As the name implies, this method reads from a stream file and loads a dictionary object, in this case, `dictConfig`. `ReadDictionaryFromFile` takes three arguments. The first is a string containing the full path to the file. The second argument is the name of the data to be extracted. In this case, the string `"ASConfig"` is passed to extract the Ad Server configuration data. The last argument is a pointer to a dictionary to receive the data read. Finally, the `MASConnectionString` property is copied into the `ConnectionString` property for our Ad object.

The next modification to the `global.asa` file is optional. If click-through ads are to be supported, we must set the `RedirectURL` property of the Ad object. This property does not specify the final target for a click-through ad, but specifies an intermediate script that redirects the customer to the appropriate page. The actual targets for click-through ads are defined in the Ad Manager tool. A redirector page `adredir.asp` can be found in the `Ads` folder on the accompanying CD.

## 156 Chapter 6 • Incorporating Advertising with Ad Server

Set the `RedirectURL` property as follows:

```
Ad.RedirectURL = "http://www.ucicorp.com/ads/adredir.asp"
```

The final required modification to `global.asa` is to set the `Application` property of the Ad object:

```
Ad.Application = "Ad"
```

The `Application` property should be set to the URL for the application's root virtual directory, as it is used when Internet Information Server writes logging information. If multiple Ad Server objects are created, giving each one a different name helps when you troubleshoot an error recorded in the IIS log.

Various optional properties of the Ad object can be set in the `global.asa` file. For example, Ad Server attempts to balance the delivery rate of different ads, based on settings defined when the ad is registered. The information used to adjust the delivery rates of ads is based on the number of ad requests and the number of times various ads have been delivered. This information is written to the ad database. Ad Server periodically reads the database to recalculate the delivery rate for the pool of ads. The default time period between this recalculation is every 15 minutes. You can adjust the period by setting the `RefreshInterval`, for example, the following line sets the rate to 10 minutes.

```
Ad.RefreshInterval = 600
```

There may be situations where the Ad Server is unable to deliver an ad. Maybe the ad database is unavailable. As a fail-safe, you can define a simple ad in the `global.asa` to be displayed when no actual ad is available by setting the `DefaultAd` property. The following statement creates a basic marquee ad.

```
Ad.DefaultAd =  
<B <MARQUEE WIDTH = 50%>The Best Training Anywhere!</MARQUEE> </B>
```

### ***Displaying the Ad***

To display the ad on your web page involves very little programming. Ad Server does most of the work. The first step is to copy the reference to the Ad object into a page variable, as follows:

```
<% Set Ad = Application("Ad" )%>
```

To display an ad, call the `GetAd` method of the `Ad` object. The `GetAd` method selects the most suitable advertisement and returns the HTML code, which is then inserted into the document. While the `GetAd` method can take three arguments, the most basic call can be made as follows:

```
<% =Ad.GetAd() %>
```

This call assumes that ads have been entered into the Ad database through the Ad Manager tool, which we now look at in detail.

## Ad Manager

Ad Manager is the administration tool to interface with Ad Server. There are four required steps in Ad Manager to configure the advertising process:

- Add an advertising customer.
- Add an advertisement campaign.
- Add campaign items to the campaign.
- Define targets for each campaign item.

To accomplish these tasks, either start the Ad Manager tool located in the Commerce program group or navigate to <http://localhost/admanager/default.asp>.

### *Adding an Advertising Customer*

A customer in this context can be defined as an individual or company that has contracted with the owner of your web site to display ads to expose the customers that visit your web site to their own, possibly related, business. Ad Server allows your web site to be a direct revenue source, as opposed to just an overhead source. As a direct revenue source, you may find it easier to justify new, more powerful equipment to management when it becomes necessary.

To add a customer, click the **Customers** link in the main Ad Manager page, then select **Add New Customer**. Fill out the **New Customer** page that appears. See Figure 6-2. All the information is straightforward, except customer type. There are two choices, Advertiser and Agency. The Ad Server software makes no distinction between the two types; they are solely for record tracking and reporting purposes.

The screenshot shows a web browser window with the address bar containing `http://t2student2/AdManager/customer_new.asp`. The page title is "New Customer". The form contains the following fields:

- Customer Name\*: YouSeeEye Corporation
- Customer Type\*: Advertiser (dropdown menu)
- Contact Name: Agency
- Address: 92 Montvale Ave, Suite 3950, Stoneham, MA 02180
- Phone: 18008841772
- Contact E-mail: abmac@msn.com
- Company URL: http://www.ucicorp.com

FIGURE 6-2

Adding a new customer with Ad Manager

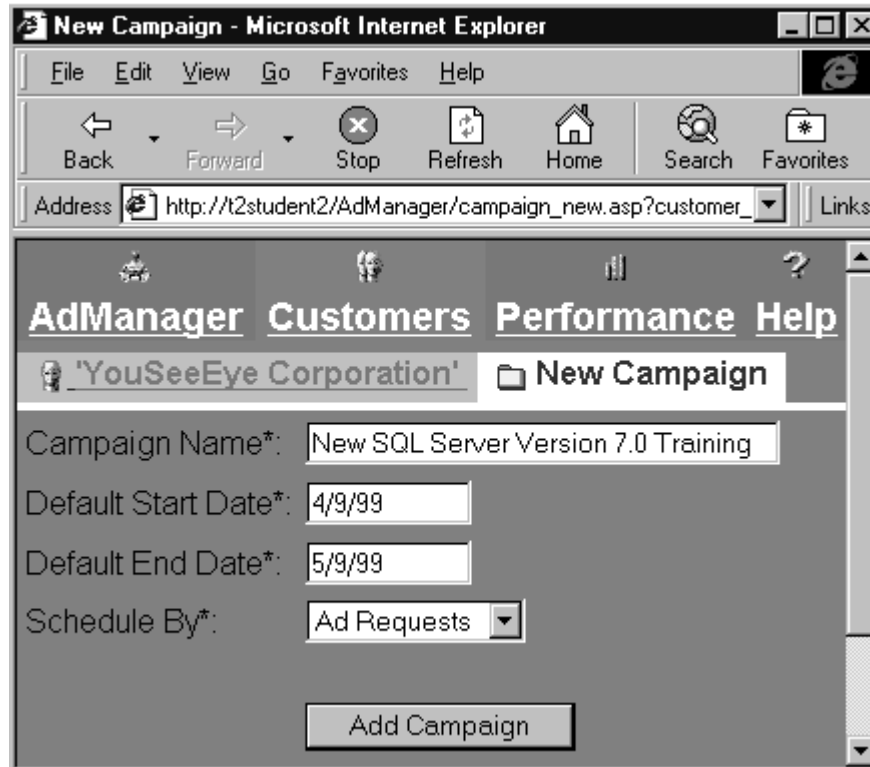
### ***Adding an Advertising Campaign***

A campaign is a container of advertisements to be delivered to retail customers on a configurable schedule. The campaign defines begin and end dates and the method by which Ad Server determines when to display the next ad in the campaign.

To add a campaign, select the **Add New Campaign** link from the Customers properties page. See Figure 6-3.

As items are added to the campaign, the campaign default start and end dates become the initial dates for each campaign item. It is possible to modify the dates for a specific campaign item. As we will see shortly, it is possible to configure an individual ad to be displayed a fixed number of times. The **Schedule By** property defines the method used to determine when an ad has been “displayed,” and can be counted against the number of times an ad has been delivered. The two options are:

- Ad Requests
- Click-through



The screenshot shows a Microsoft Internet Explorer browser window titled "New Campaign - Microsoft Internet Explorer". The address bar contains the URL "http://t2student2/AdManager/campaign\_new.asp?customer\_". The main content area displays the "AdManager" interface with navigation links for "Customers", "Performance", and "Help". Below these links, there is a breadcrumb trail: "'YouSeeEye Corporation' > New Campaign". The form contains the following fields:

- Campaign Name\*:
- Default Start Date\*:
- Default End Date\*:
- Schedule By\*:

An "Add Campaign" button is located at the bottom of the form.

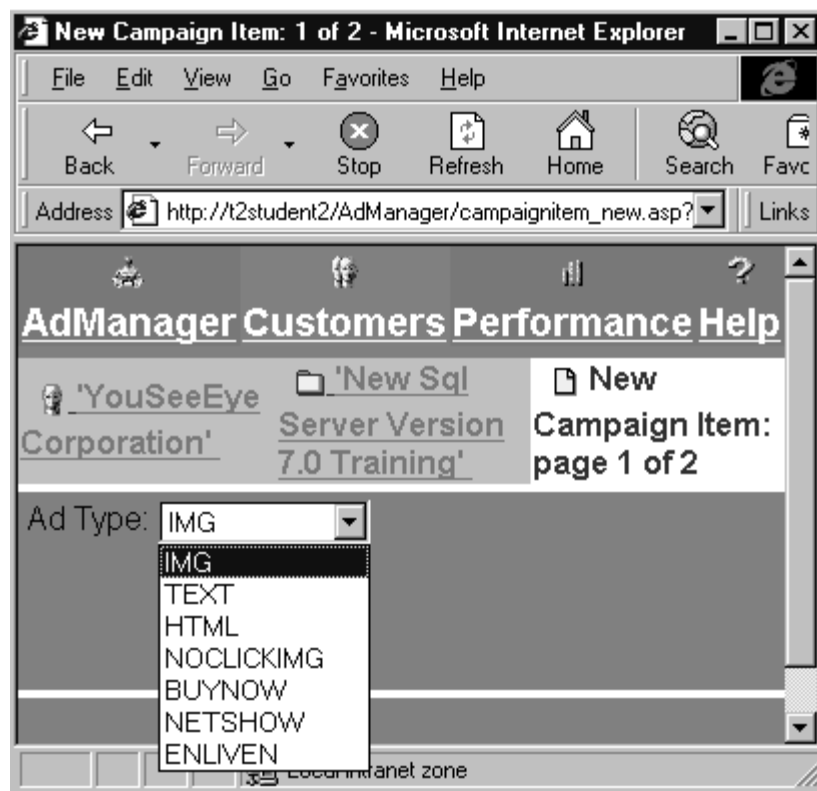
**FIGURE 6-3** Adding a campaign

For example, assume your site displays an ad that is a link to the advertised company's web site. If a campaign is configured to use ad requests as the **Schedule By** property, the delivery of the ad to the customer web page meets the requirements of the ad campaign and the number of times the ad has been delivered is updated. However, if the **Schedule By** property is configured as click-through, the ad is not considered successfully delivered unless the user actually clicks on the link to the company web site.

To summarize, using Ad Requests as the **Schedule By** property enforces the display of the ad a fixed number of times, whereas the click-through option enforces redirecting a fixed number of users to the advertised company's web site. Obviously, not every customer who is exposed to a click-through ad will follow the link. Therefore, the ad could potentially be displayed many more times than the ad is configured for. If these are advertising campaigns sold to other companies, click-through ads should command a higher fee than a campaign based on ad requests.

### ***Adding a Campaign Item***

A campaign item is the actual ad along with supporting information such as start and end dates and the number of times that the ad has been contracted to be delivered. To create an ad item, select the **Add New Campaign Item** option from the Campaign properties page. When a new campaign item is added, the first step is to define the ad type. See Figure 6-4. There are various choices here, each with different features.

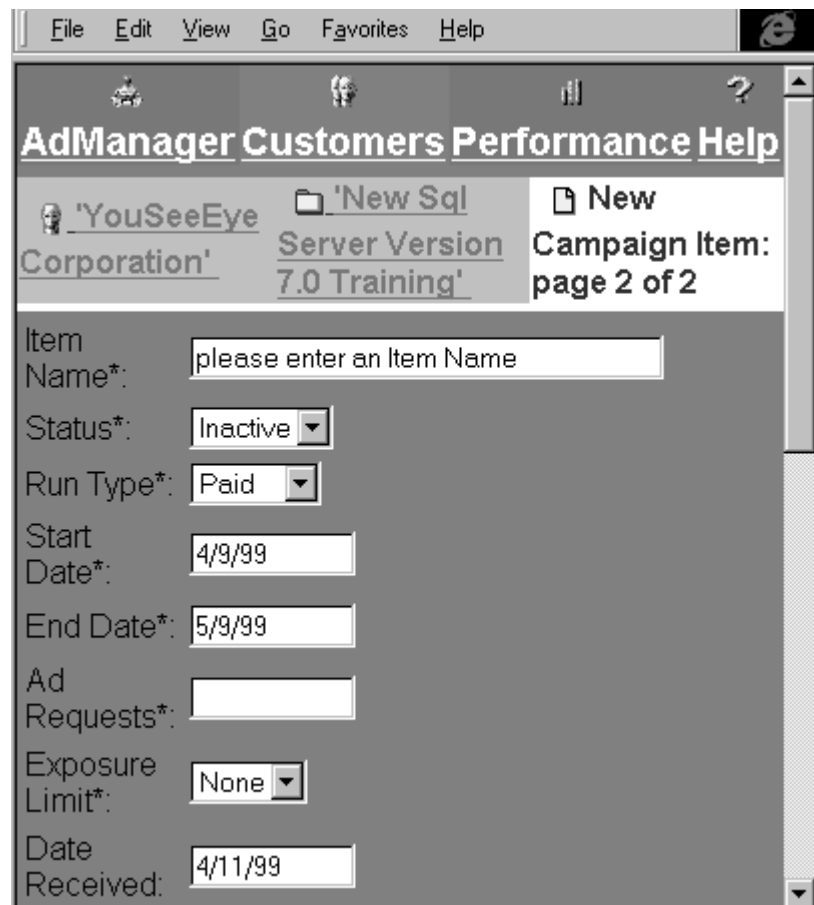


**FIGURE 6-4** Adding a campaign item (part 1)

### ***Campaign Item Properties***

Each campaign item property type has a slightly different property page. Selecting a campaign item type, in this case, IMG, next displays the campaign item properties page. For convenience we'll look at this page in three parts. The first set of parameters for a campaign item is displayed in Figure 6-5.

1. After entering a campaign name, specify whether the campaign item is **Active** or **Inactive** in the **Status** field. An **Active** ad is available to Ad Server for display to customers, while an inactive ad is not. It might be useful to make an ad inactive if it is in the process of being redesigned or if the site the ad refers to is offline or unavailable for some reason.



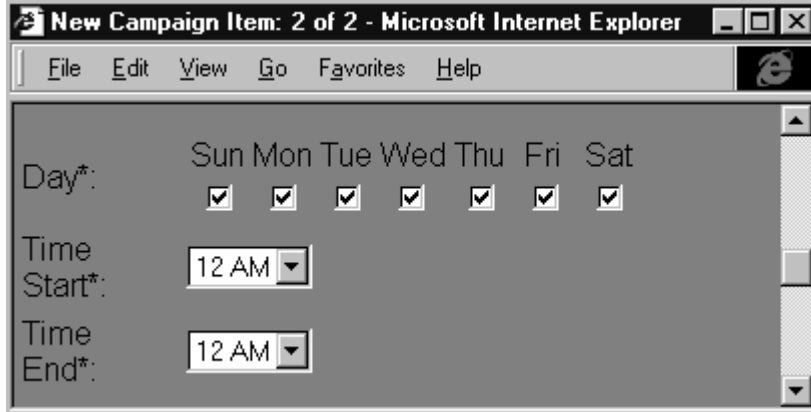
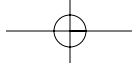
The screenshot shows a web browser window with the title 'AdManager Customers Performance Help'. The browser's address bar shows the URL 'http://www.youseeyeye.com/new\_sql\_server\_version\_7.0\_training/campaign\_item\_page\_2\_of\_2.html'. The page content includes a navigation menu with 'YouSeeEye Corporation', 'New Sql Server Version 7.0 Training', and 'New Campaign Item: page 2 of 2'. The main form area contains the following fields:

Item Name*:	<input type="text" value="please enter an Item Name"/>
Status*:	<input type="text" value="Inactive"/>
Run Type*:	<input type="text" value="Paid"/>
Start Date*:	<input type="text" value="4/9/99"/>
End Date*:	<input type="text" value="5/9/99"/>
Ad Requests*:	<input type="text"/>
Exposure Limit*:	<input type="text" value="None"/>
Date Received:	<input type="text" value="4/11/99"/>

**FIGURE 6-5** Adding a campaign item (part 2)

2. For the **Run Type** parameter, the options are **Paid** or **House**. A **Paid** ad is an ad that an advertising customer has contracted with you for delivery, or it is an ad for your business.
3. Campaign items have a **Start Date** and an **End Date**. An ad is only eligible to be displayed between these dates. The date parameters play an important role in helping Ad Server determine which ad should be selected from the pool of available ads. Ad Server is constantly calculating whether each ad is behind or ahead of schedule, based on the number of days the ad is available and the number of **Ad Requests** configured (see Step 4). If all ads are ahead of schedule, the display mechanism needs to be slowed down. This is the purpose for a **House** ad. **House** ads are basically filler ads. They consume the ad slots between the display of the various **Paid** ads.
4. The parameter **Ad Requests** defines the total number of times that this ad should be displayed. For example, you might sell advertising space to a business based on the fact that you will display their ad 1,000 times. It is not guaranteed that an ad will be displayed the number of times defined by the **Ad Requests** parameter. Suppose that an ad campaign is established with 5,000 ad requests defined, but your site is only visited 4,000 times during the period the ad runs.
5. If the same ad is displayed constantly to a customer, eventually the customer will ignore the ad. In a worse scenario, the customer can actually get annoyed with the ad and the company or product the ad contains. To avoid this situation, define an **Exposure Limit** for each campaign item. The exposure limit controls the number of times an ad can be displayed to a customer during a single session.
6. The Ad Server process ignores the **Date Received** parameter. This information is for the Commerce Site administrator.

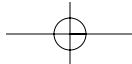
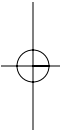
The next set of parameters for a campaign item controls the hours of the day and the days of the week the campaign item is available for display. See Figure 6-6. This control allows flexibility to target ads with a more sophisticated approach. For example, it may be beneficial to advertise the local coffee shop during the morning hours, while advertising the local restaurants later in the day. Likewise, it may make more sense to advertise weekend getaways at a resort during the first half of the week, perhaps Monday through Wednesday, when a customer will have plenty of time to make a reservation and any necessary travel arrangements. A different ad could target filling up any remaining rooms as the weekend nears.



**FIGURE 6-6** Adding a campaign item (part 3)

### ***Campaign Item Ad Types***

Seven different types of campaign items can be added to the campaign, and each is configured differently. The final portion of the campaign item properties screen varies according to the type of campaign item being configured.



### THE IMG AD TYPE

The IMG ad type displays an image that is a hyperlink to another site. For example, if you are advertising bicycles, an ad for an exercise equipment company could be displayed to take the customer to the equipment company's web site when the ad is clicked. To configure an IMG ad, see Figure 6-7.

The **Image URL** is the path to the image to be displayed in the ad. It corresponds to the SRC attribute in the <IMG> tag. The **Click URL** defines the page to be displayed when the customer clicks on the ad. From the previous example, this could be the home page for the company that sells exercise equipment. The **Alt Text** parameter provides text to be displayed when a browser for whatever reason cannot display the image. This field corresponds to the ALT attribute in the <IMG> tag.

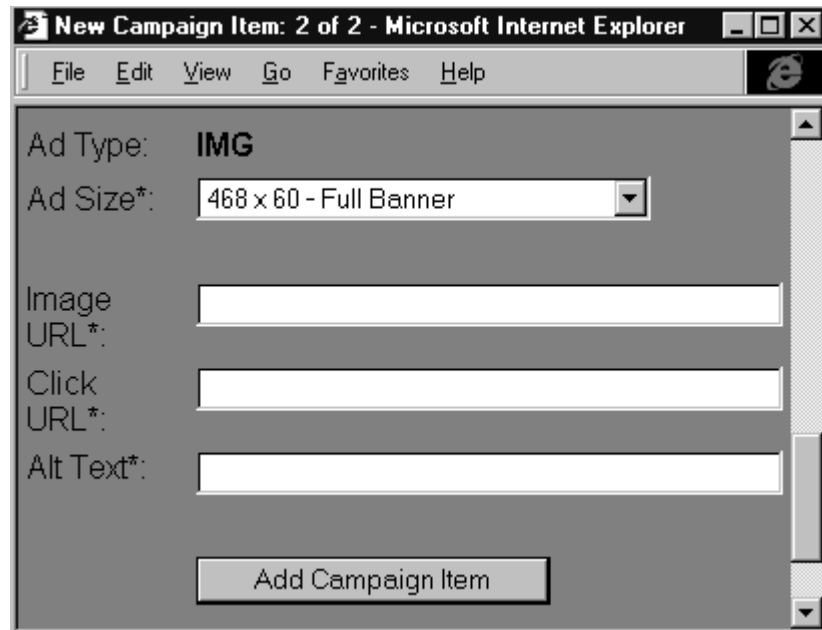
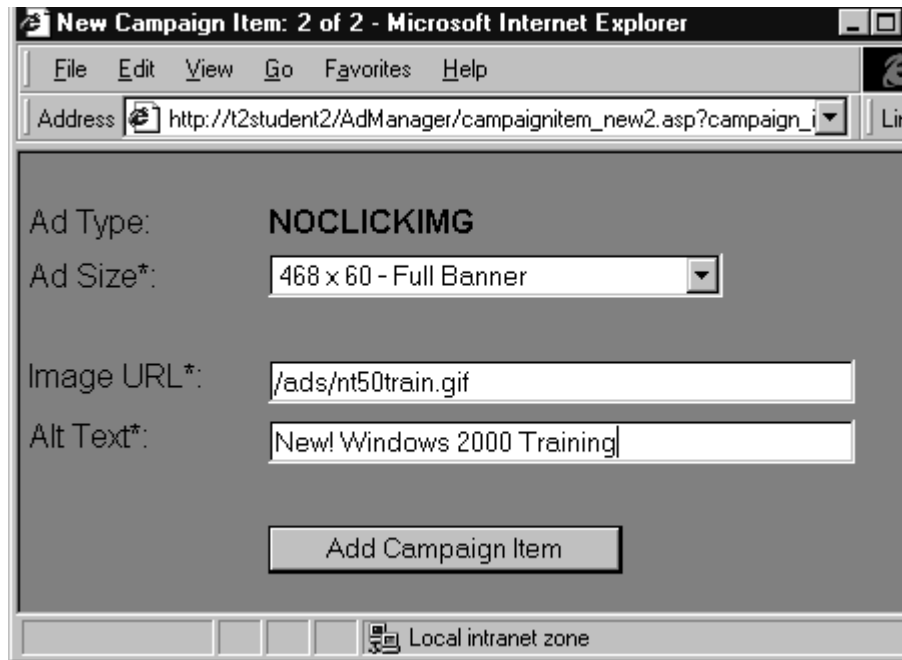


FIGURE 6-7

Configuring an IMG ad type

### THE NOCLICKIMG AD TYPE

The NOCLICKIMG ad type simply displays an image. The **Image URL** is the path to the image representing the ad. The **Alt Text** parameter provides text to display in situations where the image cannot be displayed. See Figure 6-8.



**FIGURE 6-8** Configuring a NOCLICKIMG ad type

### THE TEXT AD TYPE

The TEXT ad type displays text that is a hyperlink to another site. It is a variation of the IMG ad type. There are two parameters to configure a TEXT ad. See Figure 6-9. The **Text** parameter is the text string to display as the ad. The **Click URL** is the page to be displayed when the customer clicks on the ad.



FIGURE 6-9

Configuring a TEXT ad type

### THE HTML AD TYPE

The HTML ad type passes raw data to the web page; the data is then interpreted, and the HTML commands are executed. The ad type has one parameter, **HTML Text**, which holds the raw HTML to pass to the page. See Figure 6-10.

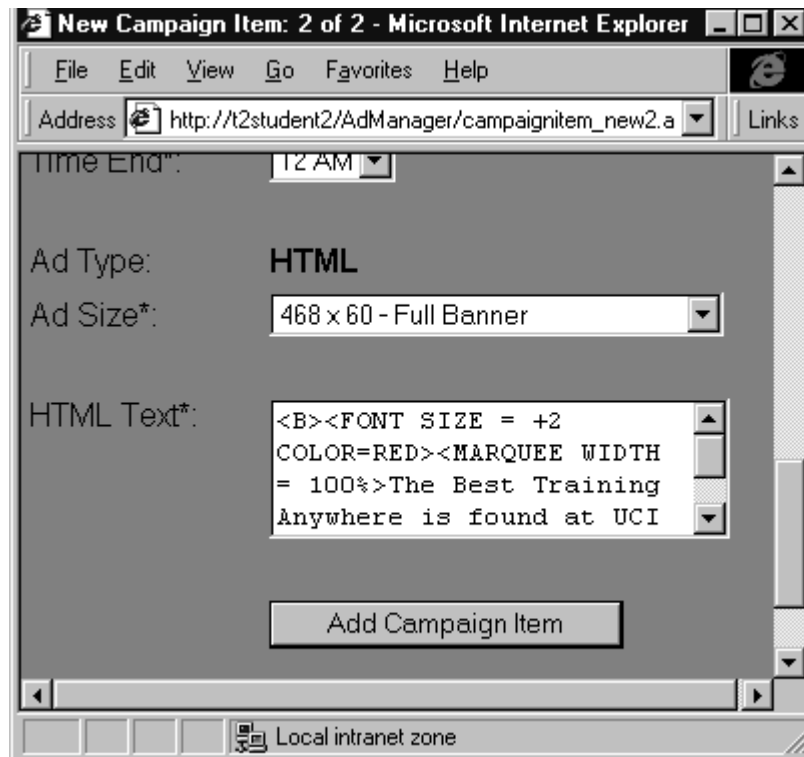


FIGURE 6-10 Configuring an HTML ad type

### THE BUYNOW AD TYPE

Special commerce sites can be created to support impulse buying. A link can be displayed on a web site in such a way that when the customer clicks on the link, an item is purchased from an entirely different site. For example, UCI Corporation may want to sell numerous books that have been written by their instructors, but it does not have a bookstore. UCI could place a **BuyNow** ad on their web site that links to a bookstore Commerce Server site. The `product.asp` page is immediately displayed with the line item already in the order form. This is accomplished by sending the product ID to be purchased as an argument to the `product.asp` page.

Figure 6-11 displays configuring a **BUYNOW** ad to purchase the product whose product ID is 7 from the `www.bookstogo.com` web site.

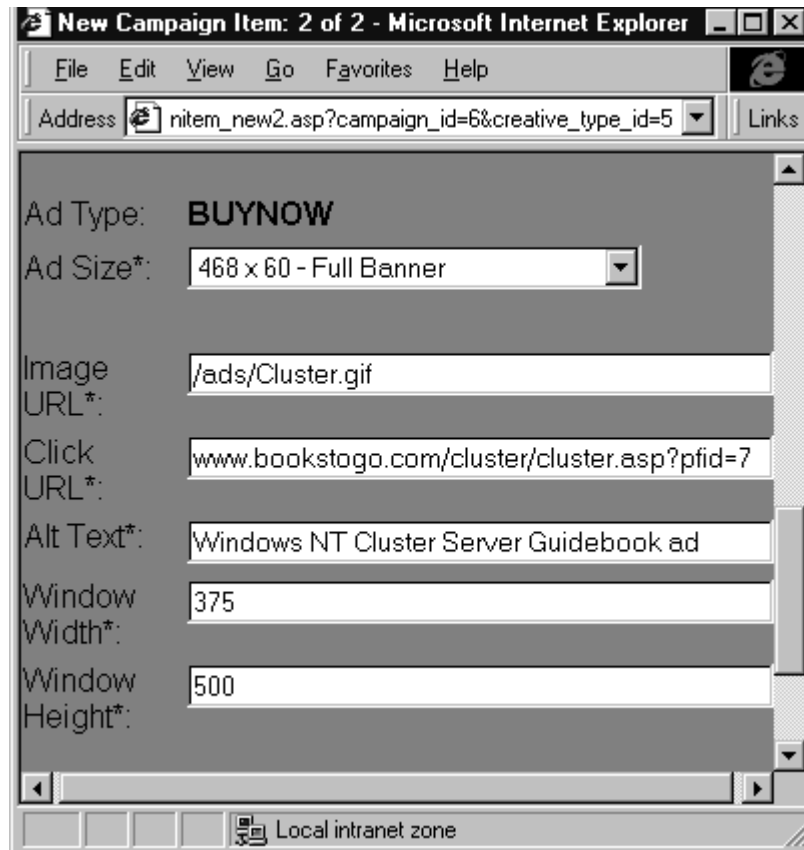
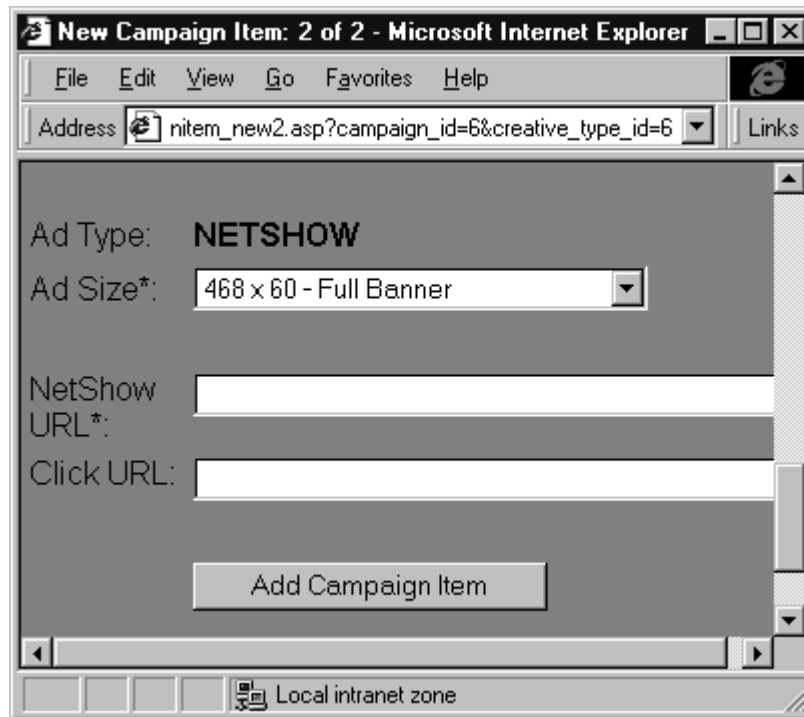


FIGURE 6-11 Configuring a BUYNOW ad

### THE NETSHOW AD TYPE

NETSHOW is software that enables you to provide high-quality audio and video over a network such as the Internet. Customers play NETSHOW content with Microsoft Media Player. Advertisers can use the Commerce Server platform almost in the same context they use television commercials.

To configure a NETSHOW ad, you fill in the fields shown in Figure 6-12.



The screenshot shows a Microsoft Internet Explorer window titled "New Campaign Item: 2 of 2 - Microsoft Internet Explorer". The address bar contains "nitem\_new2.asp?campaign\_id=6&creative\_type\_id=6". The main content area displays a form with the following fields:

- Ad Type: **NETSHOW**
- Ad Size\*: 468 x 60 - Full Banner
- NetShow URL\*: [Empty text box]
- Click URL: [Empty text box]

Below the form is a button labeled "Add Campaign Item". The status bar at the bottom indicates "Local intranet zone".

**FIGURE 6-12** Configuring a NETSHOW ad

**THE ENLIVEN AD TYPE**

The ENLIVEN ad type displays ads with Narrative's ENLIVEN software. To configure an ENLIVEN ad type, fill in the parameters shown in Figure 6-13.



**FIGURE 6-13** Configuring an ENLIVEN ad

### ***Configuring Campaign Targets***

Situations can arise where it may be inappropriate to display certain ads to specific groups of customers. For example, showing an ad for a company that sells snowboards will not generate a great deal of business when shown to customers over fifty years old. After a campaign item is added, it can be configured to have one or more **tags**. See Figure 6-14.

A tag in this context is a descriptive string that can be included in the logic that Ad Server uses when determining which is the most appropriate ad to display to the customer. When the `GetAd` method of the `Ad` object is invoked, one of the optional parameters is to send a tag list.

The tag list can contain information about the user that you have collected interactively or possibly retrieved from an active user object. The tag list could also contain information about the web page. For example, you might not want to show an ad for life insurance on a web page that allows customers to check the price of airline tickets! Last, the tag list could contain information about the browser that sent the request. You could use this information to determine whether certain ad features will not work and whether or not to display a specific ad.

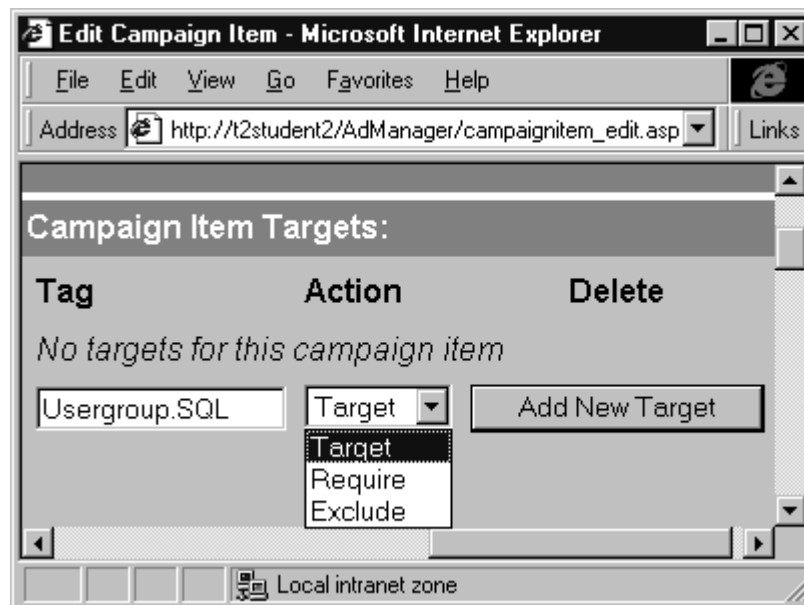


FIGURE 6-14 Configuring campaign item tags

To pass a tag list to Ad Server when an ad is requested, use the following form when invoking the `GetAd` method:

```
Set Ad = Application("Ad")
<% Taglist = Array("SQLCustomers", "CommerceCustomers") %>
<% =Ad.GetAd (Response, TagList ) %>
```

For a campaign item, you can configure three types of tags that are used by Ad Server in its determination of the best ad to display: target tags, require tags, and exclude tags.

### TARGET TAGS

When an entry in the tag list supplied by the `GetAd` method contains an entry that matched a target tag on the ad, the ad is given an increased chance of being selected for display to the customer. For example, an ad might have three target tags defined: male sports, exercise, and baseball. The `GetAd` method is invoked, passing a tag list array of one item, "baseball." All other factors being equal, the ad mentioned above would be displayed since there is a match in the tags.

Up to this point, you might think that the more tags you assign, the better your chances of a match with the customer. The property `PenalizeUnmatchedTargets` offsets this logic. When this property is enabled, which is the default, any tags defined for an ad that are not in the tag list argument decrease the ad's chance of being displayed. For example, using our same ad and three tags from the previous discussion, assume there is now another ad, referred to as ad B, with one tag of "exercise." If the `GetAd` method is invoked with a tag list of one item, "exercise," at first glance it may seem like a tossup as to which ad will be displayed since they both contain the tag "exercise." However, since the first ad has two other tags that were not requested in the `GetAd` call, its chance of being displayed is decreased and most likely ad B will be delivered to the customer.

### REQUIRED TAGS

Required tags are very straightforward. If an ad has a **required** tag defined, the tag must appear in the tag list argument of the `GetAd` method. For example, to ensure that an ad is only displayed to females, a required tag of **gender.female** could be placed on the ad. Only when this tag is supplied with the `GetAd` method will the ad even be considered for display to the customer.

### EXCLUDE TAGS

Exclude tags remove an ad from the available ad pool for display when the tag is matched with an entry in the tag list argument from the `GetAd` method. Another way to handle the previous situation is to place an exclude tag of **gender.male** on the ad. If the same string is included in the tag list argument of the **GetAd** method, the ad will not be displayed.

**THE AD.SIZE TAG**

When a web page is designed, only a fixed amount of space is allocated for ad display purposes. Just as with advertising in other media, it is possible to offer different size ad slots with different features. The size of the ad is specified when the campaign item is created. If the ad pool contains various size ads, then when the `GetAd` method is invoked, it must request an ad with a given size to fit into the ad slot on the web page. This is accomplished by including a special tag in the tag list argument with the `GetAd` method. The tag takes the format of `Ad.Size.type` where *type* is one of the options in Table 6-1. For example, to request a full-size banner ad of 468 x 60 pixels, include the tag `Ad.Size.Banner` in the tag list argument of the `GetAd` method. This tag will be treated as a required tag, meaning that if no ad matches the size requested, no ad will be returned by Ad Server for display to the customer.

**TABLE 6-1** Ad size options

Name	Tag	Width (in pixels)	Height (in pixels)
Button #1	Button1	120	90
Button #2	Button2	120	60
Full banner	Banner	468	60
Full banner with navigation bar	NavBanner	392	72
Half banner	Half	234	60
Micro button	Micro	88	31
Square button	Square	125	125
Vertical banner	Vertical	120	240

**THE AD.TARGETFRAME TAG**

Another reserved tag designates the target frame or window for the display of ads. The tag name is `Ad.TargetFrame.frame` where *frame* can be any one of the strings from Table 6-2. If this tag is not supplied, the default is `Ad.TargetFrame._top`.

**TABLE 6-2** Ad location options

Tag	Causes the click-through page to appear
<code>Ad.TargetFrame._top</code>	In the main browser window.
<code>Ad.TargetFrame._self</code>	In the same frame in which the ad itself appears.
<code>Ad.TargetFrame._parent</code>	In the immediate parent frame of the frame containing the ad.
<code>Ad.TargetFrame.frame</code>	In the frame named <i>frame</i> .

## Users and Ad Histories

Earlier in this chapter, we defined the **Exposure Limit** property of a campaign item as the number of times an ad could be displayed to a customer during a single session. To support this feature, it is necessary to keep a history of ads already displayed to the user and to pass this list when the `GetAd` method is invoked. The information contained in this list, along with defined exposure limits for ads, is used by the Ad Server process in determining the appropriate ad to display to the customer. The history list is the third potential argument to the `GetAd` method. Examine the code in Example 6-1.

### EXAMPLE 6-1 Maintaining an ad history

```
<% viewHistory = Session("AdHistory") %>  
<% = Ad.GetAd(Response, Null, viewHistory) %>  
<% Session("AdHistory") = viewHistory %>
```

## Other Uses for Tags

Tags can be used in various ways that may not be immediately evident. For example, let's assume that we have an ad that does not work correctly with all types and versions of the various browsers that customers may be using to access your E-commerce site.

See Example 6-2. The sample code uses the `BrowserType` object to determine the name. It then constructs a tag string that is sent with the `GetAd` method. If the same browser string is used as a required tag on the campaign items, we have guaranteed that ads will not be sent to a browser that does not have the supported functionality.

### EXAMPLE 6-2 Controlling display of ads by browser type

```
<% Set bc = Server.CreateObject( "MSWC.BrowserType" ) %>  
<% browserTag = "Browser." & bc.browser %>  
<% tagList = Array( browserTag ) %>  
<% =Ad.GetAd( Response, tagList, null ) %>
```

## Ad Server Object Reference

Tables 6-3 and 6-4 summarize the properties and methods for the Ad Server object.

TABLE 6-3 Ad Server object methods

Method	Description
DumpSchedule	Returns a table, in HTML format, of the current advertisement delivery schedule of the local instance of the <code>AdServer</code> object.
GetAd	Selects the most appropriate advertisement, and returns HTML text that can be inserted into the calling page.
RecordEvent	Logs events to the Microsoft Internet Information Server (IIS) log, and updates the count in memory.
RefreshSchedule	Connects the <code>AdServer</code> object to the database immediately, receives updated delivery information, and writes ad performance data.

TABLE 6-4 Ad Server object properties

Property	Description
Application	Name used to distinguish separate instances of the <code>AdServer</code> object in error messages in the Microsoft Windows NT Event Viewer.
ConnectionString	Connection string to be used when the <code>AdServer</code> object connects and authenticates itself to the Ad Server database.
DefaultAd	HTML text to be returned by the <code>GetAd</code> method if the <code>AdServer</code> object cannot determine an appropriate advertisement.
DesignMode	Boolean. If true, the <code>AdServer</code> object returns an error message when the <code>AdServer.GetAd</code> method cannot determine an appropriate advertisement to return. If false, the method displays the default ad in this case.
PenalizeUnmatchedTargets	Boolean. When set to true, the selection weight for an ad is decreased if the campaign item contains target tags that do not appear in the ad request.
RedirectURL	URL of the Active Server Pages (ASP) redirector script, which logs the click event and redirects the visitor's browser to the destination page.
RefreshInterval	Number of seconds between refreshes, at which time the <code>AdServer</code> object connects to the database to post its delivered performance data and receive updated delivery schedule information.
RetryInterval	Specifies the number of seconds that the <code>AdServer</code> object waits between attempts to reconnect with the AD database if the connection fails.

## Example: Using Ad Server to Display Ads

The UCI site can be extended to include banner advertisements, using the procedures described in this chapter. The file `global.asa` contains the following modifications of the `Application_OnStart` procedure, as listed in Example 6-3.

### EXAMPLE 6-3 Including banner ads

```
rem -- Configure Ads

Set Ad = Server.CreateObject("Commerce.AdServer")
Set Application("Ad") = Ad

Ad.ConnectionString = "DSN=Ads;UID=sa;PWD="

rem -- Uncomment the following lines to extract the connection string from
site.csc then comment or remove the above line

rem pathConfig = Server.MapPath("config/site.csc")
rem streamConfig = "ASConfig"
rem set dictConfig = Server.CreateObject("Commerce.Dictionary")
rem set fileDoc = Server.CreateObject("Commerce.FileDocument")
rem Call fileDoc.ReadDictionaryFromFile(pathConfig, streamConfig, dictConfig)
rem Ad.ConnectionString = dictConfig.MASConnectionString

rem Replace www.ucicorp.com from the following two lines with your site URL

rem Ad.RedirectURL = "http://www.ucicorp.com/adsamples/aredir.asp"
Ad.Application = "/ucicorp"

Ad.DefaultAd = "<P><B><marquee width=50% border=0>The best training
anywhere!</marquee></B></P>"
Ad.RefreshInterval = 60
```

The `default.asp` file contains the following instructions, which display two targeted ads.

```
<% Set Ad = Application("Ad") %>
<% Taglist = Array("SQLCustomers") %>
<% =Ad.GetAd(Response, TagList) %>
```

Notice the name of the data source is `Ads`. Match the data source name specified during the installation of Commerce Server. Also notice the `Ad.Application` variable refers to the `ucicorp` subdirectory, which corresponds to the virtual directory of the web site. You will need to add the above ads with Ad Manager before the ads will be displayed.