



# Section 1 – Graphic Modeling - Course Introduction

## Course Objectives

This course overviews the responsibilities of the graphic design team in NeuArchitect application development, with a focus on the technical features of NeuArchitect and the benefits to the owner of the web application.

In this course you will cover:

1. Team Roles in Style Design
2. Analysis and Design – a High Level Overview of NeuArchitect Graphical Tools
3. NeuArchitect Features – a Closer Look at NeuArchitect Graphical Tools
  - Style Repository
  - Themes
  - Control Styles
  - Page Styles
4. Style Modeling Concepts in NeuArchitect
5. Page Construction from Style Models
6. Integrating Multi-Media into Your Application



### **eLearning Flash:**

If you're looking for a quick, 10,000 foot view of the technical material in this section [click here to go to the Section Reference](#), otherwise, continue through the section.

## Overview

This course is for graphic designers who will be working with NeuArchitect. The graphic designer working with NeuArchitect may be responsible for designing the user interface and overall customer experience for a web site. This includes overall navigation flow, layout of specific pages, and creation of individual graphic elements. The emphasis in this course will be on creating sites that are graphically rich yet highly functional and easy-to-use.

## Course Prerequisites

As this course assumes knowledge of NeuArchitect, the student should have already taken the following courses offered by NeuVis NeuUniversity:

Introduction to NeuArchitect  
NeuArchitect Visual Web Development.

## Roles in Style Design and Modeling with NeuArchitect



***The following is provided as a background for those who are new to the NeuArchitect development environment.***

Website development requires a unique blend of artistic and technical talents to deliver content accuracy, usability and branding requirements in an attractive package.

The key responsibilities for a graphic designer using NeuArchitect may differ from project to project but usually involve some level of each of the following:

### **Key Responsibilities of the Graphic Designer**

Coordination, design and production of web sites

Maintenance of existing sites

Using state-of-the art design

Synchronization with other teams in the development process

Other responsibilities include graphic design support, website and intranet design, redesign and continuous graphic changes of websites. Experience usually includes expertise in the areas of information design, navigation design principles, graphic design, user interface design, web usability tools and methods, content analysis and creative concept development. The designer should also have expertise in branding as it relates to web site development, as well as an understanding of web personalization from a marketing and technical perspective.

The graphic designer should have in-depth knowledge of HTML/ Java/ JavaScript and their support across browsers, browser versions, and operating systems; knowledge of multimedia interactivity, gif animation, Flash animation/interactivity and its integration in web content, as well as a thorough understanding of web graphic optimization and file formats. Included in the skills inventory of the graphic designer would be knowledge of web design/production software programs on PC Windows computing platforms such as Adobe PhotoShop, Adobe Image Ready, Macromedia Flash, Adobe Illustrator, Macromedia Dreamweaver, FrontPage, Corel Draw and Flash, etc.

The table below outlines the key and supporting team roles that are typically involved in decision making for the style and design of a website created with NeuArchitect:

<b>Key Roles</b>	<b>Responsibilities/Skills</b>
<b>Graphical Support Roles</b>	
<b>Graphic Designer</b>	<p>The Web Page Designer is the individual responsible for the overall appearance of the user interface using themes and styles. The Web Designer should have experience with designing user-friendly interfaces. The key role of the designer is to create visually appealing web pages using complimentary colors and graphics as well as defining easy to navigate page controls.</p> <p>The designer is typically skilled in the use of third party graphical applications (i.e. Adobe Photoshop, Macromedia Director/Flash, etc.) to aide in the design phase. Web Designers are the counterparts to the Application Developer; the developer will primarily focus on the coding and functionality of the application and relay that information to the designer. Web Designers may also work with a Graphic Artist and the Business Analyst to assist with design and business requirements.</p>
<b>Graphic Artist</b>	<p>The Graphic Artist will typically assist the Web Designer with complex graphical images. The Graphic Artist should be skilled with designing, manipulating, and/or editing graphics to meet the designer's business requirements. Depending on the nature of the application being designed, the Graphic Artist and Web Designer may be the same individual in a dual role.</p>
<b>Other Supporting Roles</b>	
<b>Business Analyst</b>	<p>The Business Analyst is responsible to gain an in-depth knowledge of the business domain as it pertains to new application development. The information is typically obtained by conducting interviews with business experts and company power-users. The Business Analyst will review the gathered information and translate the appropriate business requirements to the responsible project resources, including the Web Designer.</p>
<b>Application Developer</b>	<p>The Application Developer is primarily responsible for the coding and functionality aspects of the application. The Web Designer would normally work closely with the developer to understand what may be required on the user interface and as well as how the web pages should integrate according to the programmed application.</p>

## Why You Should Apply NeuArchitect Style Modeling Techniques



***In the following, you will learn about the benefits of NeuArchitect Style Modeling and be introduced to the style modeling components.***

By creating and applying a consistent framework for visual design via NeuArchitect style modeling, the following benefits can be achieved:

<b><i>Benefit</i></b>	<b><i>Description</i></b>
<b><i>Consistent Visual Design</i></b>	The visual design of your application will be consistent and will not vary with the visual preferences of each developer.
<b><i>Rapid Development</i></b>	Page developers do not have to constantly change visual parameters. By implementing pre-built control styles, page models, and graphical styles, developers can finish page design and development in a fraction of the time.
<b><i>Compliance</i></b>	You can enforce compliance with visual design requirements. For example, you can enforce the use of web-safe colors, or any other color schemes constrained by a device. As another example, you can enforce your corporate brand identity in your application.
<b><i>Overcome Shortage of Graphical Expertise</i></b>	Developers with minimal graphic skills can rapidly design very appealing sites by using pre-designed themes and styles.
<b><i>Ease of Change</i></b>	If you need to change the visual look of your application, the change can be accomplished quickly, reliably, and with consistency. You can even implement style changes dynamically, personalized to individual user's preferences.
<b><i>Separation of Roles</i></b>	Your graphic artists can focus their time on designing the visual look of the entire application along with the shared models and graphical styles, and your page developers and application programmers can focus their effort on providing the functional interaction in a page.
<b><i>Re-Usability</i></b>	The visual look, and portions thereof, designed for an application can be re-used in any other application you may want to apply it to.

## Style Modeling with NeuArchitect

NeuArchitect provides the graphic designer with a complement of style modeling components to accelerate the development process. These components provide an application with a consistent look and feel by means of colors, fonts, page controls, images, and page designs.

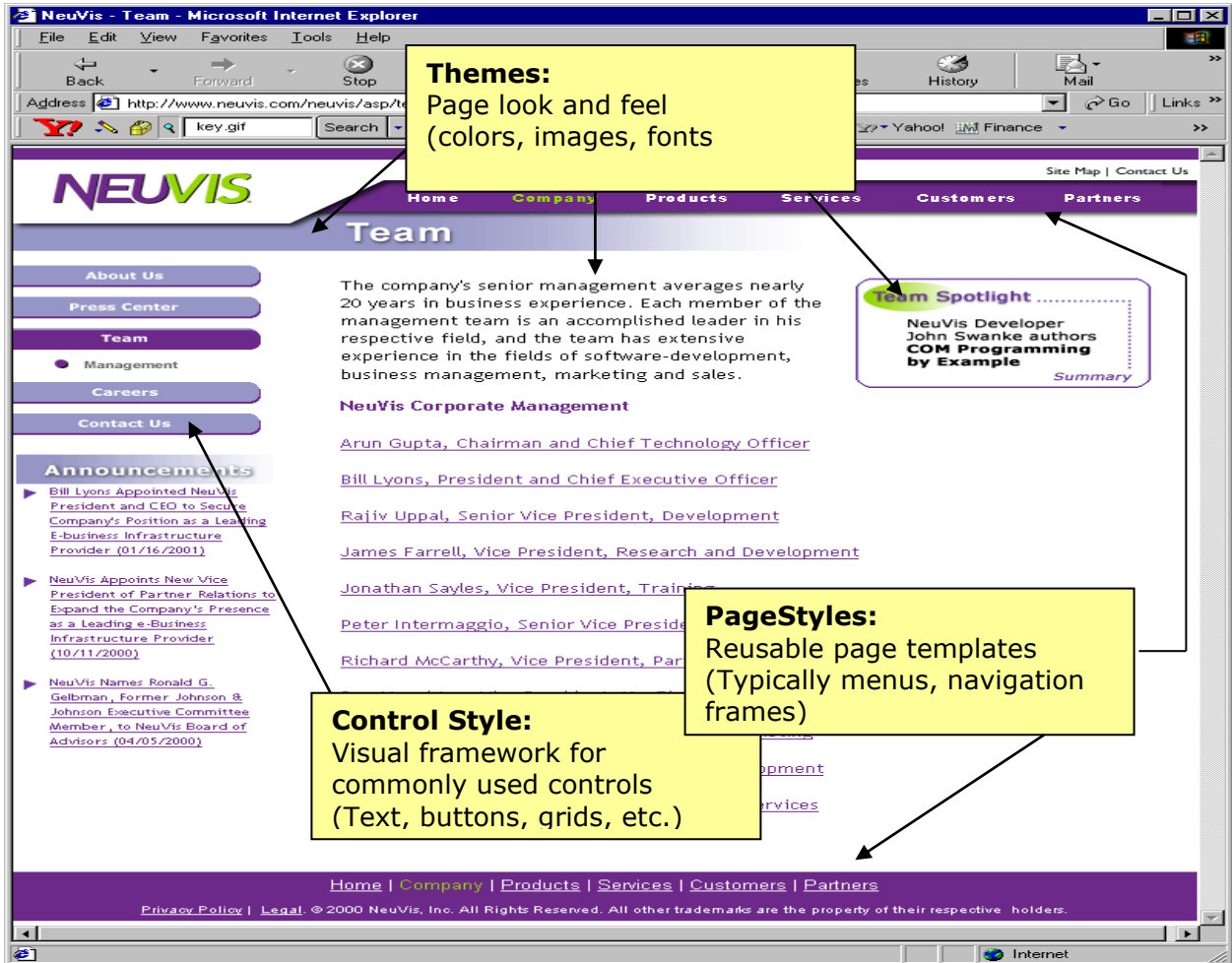


Figure 1 – Style Modeling Components – Web Page Representation



Check out the figure above. Which style component is controlling the placement of the NeuVis logo?

## Style Modeling in the NeuArchitect Development Model

While there are a few settings that are applied in the Development Defaults of a NeuArchitect application, the NeuArchitect developer will apply most of the style modeling parameters as the Site Model is developed. The graphic below depicts where the components of Style Modeling impact the development process.

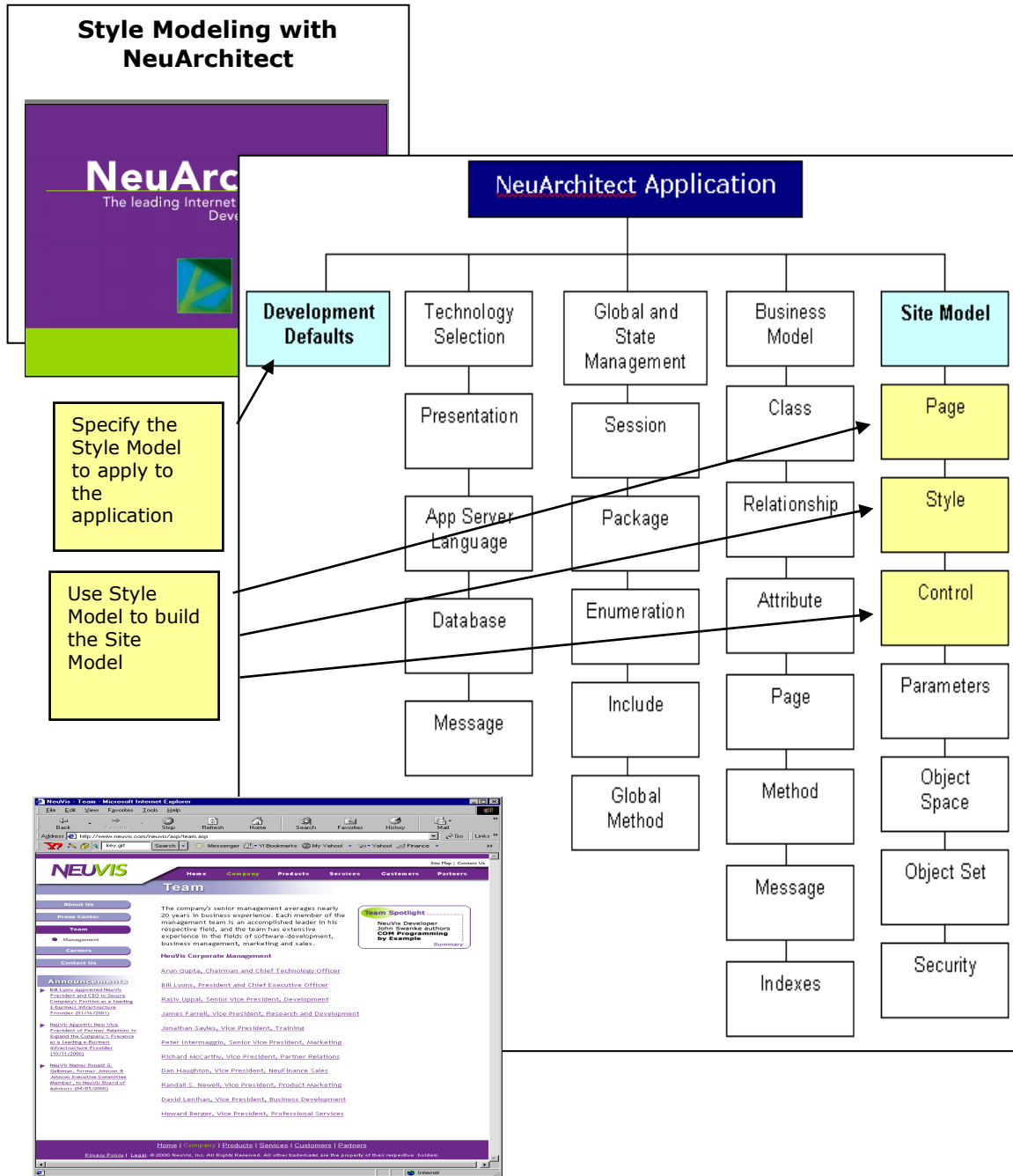


Figure 2 – Style Modeling in the NeuArchitect Development Environment

## Style Modeling Components

Style Modeling in NeuArchitect enables the designer with a powerful interface for website design and a vehicle to manage visual change efficiently via three inter-related styling components. NeuArchitect provides this support via a style repository delivering critical functionality in three components:

- Themes – colors, images and fonts
- Control Style – web page controls
- Page Styles – templates to accelerate page design

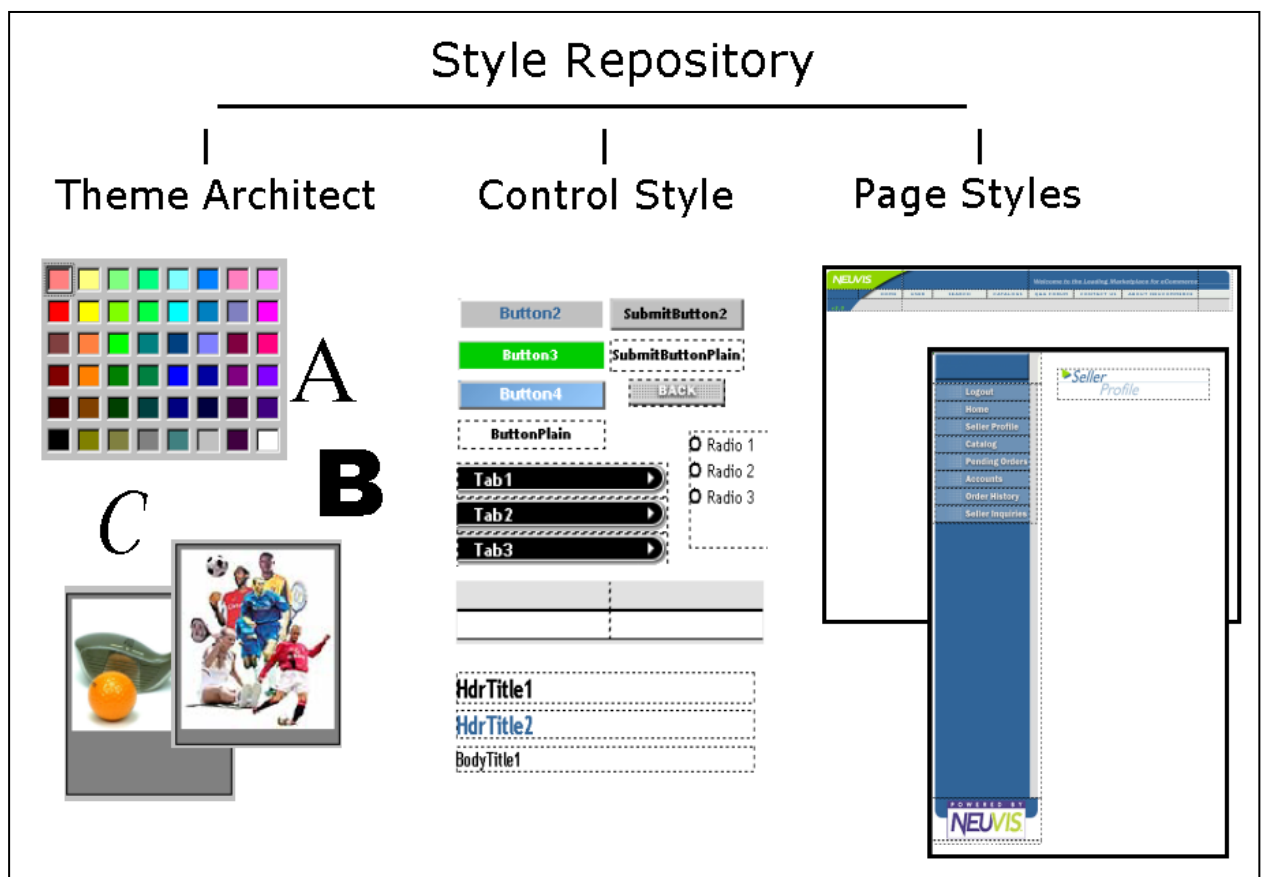


Figure 3 – NeuArchitect6 Style Modeling Components

## Style Modeling Cycle in NeuArchitect

Note in the diagram below that there is a flow when using NeuArchitect Style Components.... Themes are used to set ControlStyle properties, those controls are used to build the Page Styles and the Page Styles are templates for finished pages.

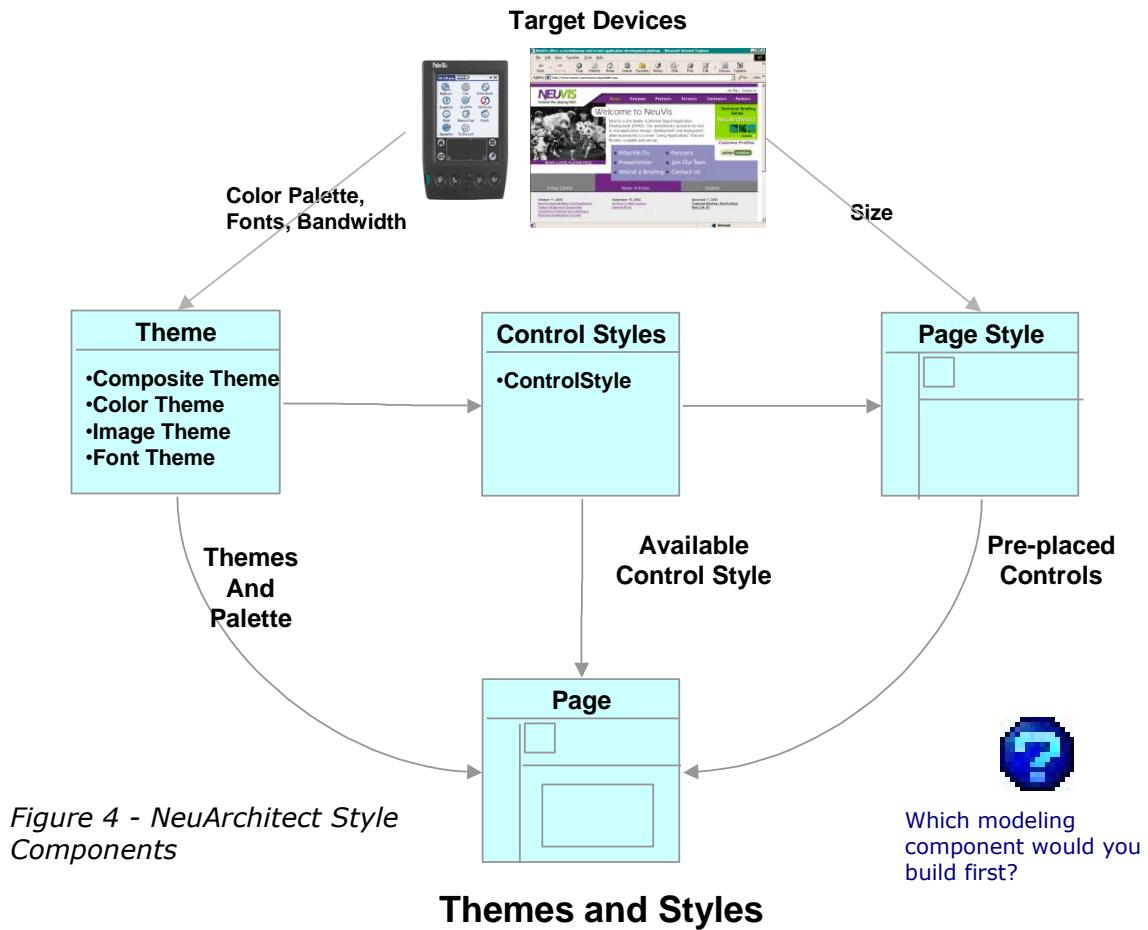


Figure 4 - NeuArchitect Style Components

Style Component	Description	Considerations
<b>Themes</b>	Create pre-defined set(s) of colors, images, and fonts to be used throughout an application	Target device (browsers, handhelds) will effect the color, fonts, images that can be used in Themes
<b>ControlStyle</b>	Delineate the visual framework for commonly used controls, (buttons, grids, textboxes, etc.), in an application	Control design is will have the same target device considerations, but, are usually created within the framework of the Themes
<b>PageStyles</b>	Create page templates that provide the ability to use pre-place controls on a page (or pages) in the application	Usually created within the constraints of the defined Themes and controls in the ControlStyle with the added consideration of target device size.



## The NeuArchitect Style Repository

Parameters for these style modeling components are captured in a **Style Repository**. The style repository is a NeuArchitect application (.OMD file) and is created and/or modified like any other NeuArchitect application. The figure below depicts an open style repository.

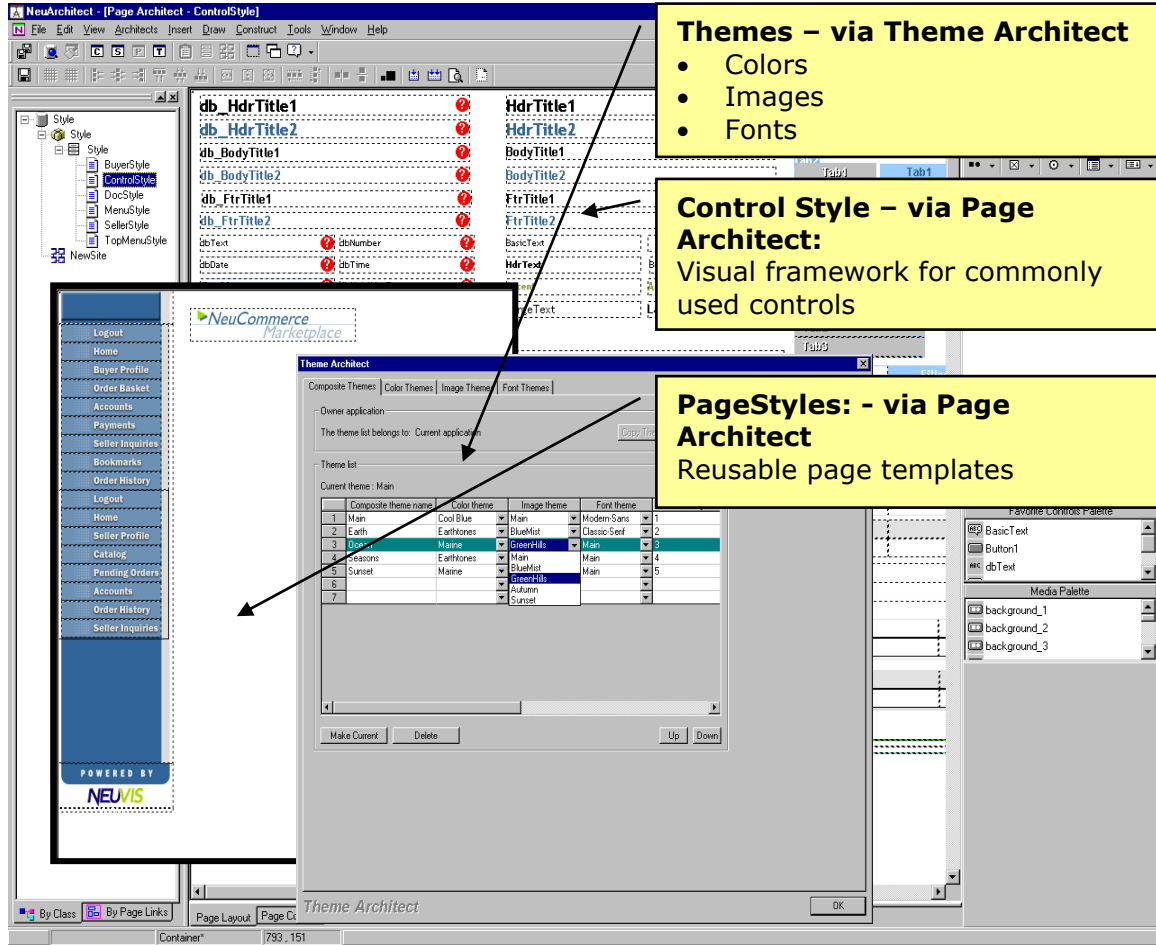
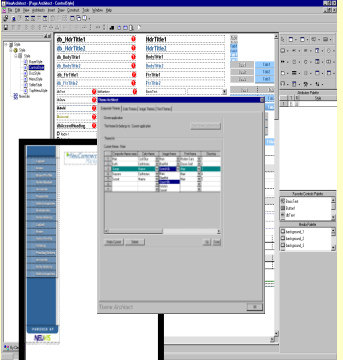
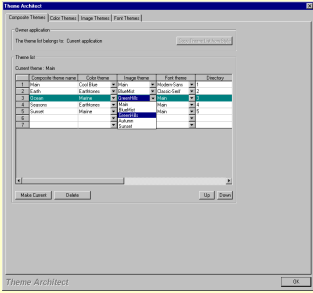
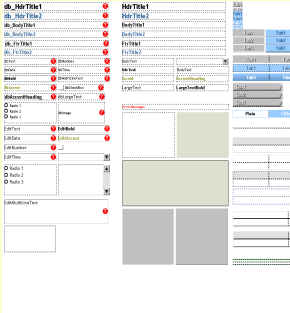



Figure 4 – Style Repository Modeling Components

Modeling Component	Description
<p><b>Style Repository</b></p> 	<p>The style repository is a NeuArchitect application (.OMD file). The style repository is the host for the styling elements:</p> <ul style="list-style-type: none"> <li>• Themes</li> <li>• ControlStyle</li> <li>• PageStyle</li> </ul> <p>Each NeuArchitect application is allowed one current style for accessing the design attributes contained in a style repository. Corporate requirements may necessitate more than one overall web presentation style, requiring maintenance of several style repositories.</p> <p><b>The graphic designer may either:</b></p> <ul style="list-style-type: none"> <li>• <b>Use an existing style repository</b></li> <li>• <b>Create a new style repository</b></li> </ul>

# NeuArchitect Style Repository Modeling Components

The style repository modeling components provide the graphic designer with control over the web presentation. The designer not only has the ability to build the graphical framework that will assure a consistent web presence, but also the flexibility to apply standards globally for the application or selectively within the application.

<b>Modeling Componens of the Style Repository</b>	<b>Description</b>
<p><b>Theme</b></p> 	<p>Themes are created and maintained via Theme Architect. Themes are typically created and maintained in the style repository providing for a consistent 'look and feel' for:</p> <ul style="list-style-type: none"> <li>• Colors</li> <li>• Fonts</li> <li>• Images</li> </ul> <p><b>Theme Architect provides the graphic designer the flexibility to:</b></p> <ul style="list-style-type: none"> <li>• <b>Create and maintain themes in the style repository</b></li> <li>• <b>Create and maintain themes directly in the business application using the option to Copy Theme List from Style</b></li> </ul>
<p><b>ControlStyle</b></p> 	<p>Page controls are the visual elements of the business application (buttons, grids, text, etc.) serving to:</p> <ul style="list-style-type: none"> <li>• Present information</li> <li>• Format data</li> <li>• Imply action to take</li> <li>• Allow for graphical images to be placed on a page.</li> </ul> <p><b>The Control Style page is created/modified by opening the Style .OMD and using the Page Architect feature.</b></p>
<p><b>PageStyle</b></p> 	<p>Page styles provide the ability to pre-place controls, as defined within the Control style page, on a page or pages to create templates. Creating page styles will provide the ability to set the default page size and page background for your page. Using page styles allows you to maintain a consistent look and feel across all pages within the application.</p> <p><b>PageStyles are created/modified by opening the Style .OMD and using the Page Architect feature.</b></p>



True or False? Themes, ControlStyles and Page Styles are created using Page Architect.

## Steps in Site Modeling and Page Modeling for NeuArchitect Application Development



**The steps shown below for site modeling are critical. Following the steps ensures an accelerated development process. Additional recommendations and considerations are also presented.**

In previous NeuUniversity courses, you have been working, more or less, in a vacuum. You have been the designer, developer, the QA technician, etc. In a corporate development environment the following should be completed before a developer even starts to work on page development:

- All modeling decisions should be made
- NeuArchitect site models should be created

The following table maps the suggested process for setting up the NeuArchitect modeling components. Keep in mind that there is a logical flow as each step enables the functionality of the next step. The following is also the order in which you will learn about the NeuArchitect modeling components.

<b>Process for setting up and using the NeuArchitect modeling components.</b>	
<b>1</b>	<b>Establish Naming and Directory Setup Conventions</b> - for style repositories, image access and storage preferences on your development machine
<b>2</b>	<b>Create Style Repositories</b> - can be created from scratch or by cloning existing repositories
<b>3</b>	<b>Define Themes</b> - use Theme Architect to define all the required themes in the repository for colors, images, fonts and composite themes
<b>4</b>	<b>Create the ControlStyle</b> - specify the properties and attributes of controls using Theme Architect definitions
<b>5</b>	<b>Design PageStyles</b> - map out various page styles using the properties and attributes defined in Theme Architect and controls created in the ControlStyle
<b>6</b>	<b>Specify the 'Current Style' in your NeuArchitect application</b> - NeuArchitect allows the flexibility of using the style repository of choice
<b>7</b>	<b>Site Modeling and Page Modeling</b> - use the power of NeuArchitect to model the site and individual pages

## NeuArchitect Style Design - Recommendations

To assist you in the design stage, we have a few recommendations you may like to consider. The recommendations are based on knowledge gathered from prior projects and use of the style design tools. Note that the first three items in the table below fall within the scope of Theme Architect while the remainder are designed in Page Architect.

<b>Themes and Styles</b>	<b>Theme and Style Recommendation</b>
<b>Colors Themes</b> <b>Image Themes</b> <b>Font Themes</b>	<p>In designing your themes with Theme Architect, use logical names similar to those in the SampleStyle repository. You can modify the actual values for the established logical names, thereby saving you time from having to re-enter names and values for a brand new theme. If your application has multiple target devices, you should design at least one theme for each type of target device.</p>
<b>Control Styles</b>	<p>When designing your control styles, make sure that you only use one set of logical names for colors, images and themes and just change the values when creating different themes. Using logical names for the themes will allow the control styles to be automatically updated just by changing the assigned theme.</p>
<b>Page Styles</b>	<p>Typically pages in an application will consist of a fixed part (i.e. frames) and a variable part (i.e. content). The fixed part, generally to the left and top, has the same controls for a large number of pages. You should define a page style that contains these fixed controls. The number of page styles used will depend on how many variations of fixed controls exist within your application. If your application has multiple target devices, you should design a set of page styles for each type of target devices.</p>

## NeuArchitect Style Design - Considerations

Again, based on knowledge gathered from prior projects and use of the NeuArchitect style design tools, you may save time and possible frustration by ensuring you consider the informational input for style design outlined below:

<b><i>Inputs</i></b>	<b><i>Conidérations for the Style Design Stage</i></b>
<b><i>Questions to Ask</i></b>	<p><b>In style design, you need to answer the following questions:</b></p> <ul style="list-style-type: none"> <li>• What are the inputs to style design?</li> <li>• What color themes should be used?</li> <li>• What image themes should be used?</li> <li>• What font themes should be used?</li> <li>• What composite theme or themes should be established?</li> <li>• What control styles should be used?</li> <li>• What page styles should be used?</li> </ul>
<b><i>Style Design Requirements</i></b>	<p><b>In style design, you need to take into account the following requirements:</b></p> <ul style="list-style-type: none"> <li>• Brand Identity</li> <li>• Formal look and feel requirements</li> <li>• Device constraints</li> <li>• User and stakeholder preferences</li> <li>• Target audience</li> </ul>
<b><i>How Many Themes to Use</i></b>	<p><b>You may decide to use one or multiple themes in your application:</b></p> <p>You may use one theme if the look of your application does not vary. You should select or design a theme that is consistent with your company’s brand identity, as well the taste and preferences of designers and users of the site.</p> <p>You may use multiple themes under the following circumstances:</p> <ul style="list-style-type: none"> <li>• You want to change the theme of the application on a periodic basis.</li> <li>• You want to change the theme without reconstructing your application.</li> <li>• You want various sites in the application to use different themes.</li> <li>• You want the users of the site to be able to customize their visual look.</li> <li>• Your site represents multiple stakeholders and you want to provide each stakeholder the option to use a different theme.</li> </ul>

## Workshop



### Additional Software – Netscape 6.0x

In addition to the hardware/software requirements of the previous courses, the student will also need Netscape 6.0x. This can be installed from the Netscape site at: [www.netscape.com](http://www.netscape.com)

It is suggested that you let Netscape install a typical setup directly from the website.



### Course Preparation Workshop - 1



## The Application Files

In order to complete the exercises in this course, you will need to create a directory and a NeuArchitect application. When you have finished this exercise, the files should be resident in:  
    \applications\projects\GMApp

### The Database and ODBC Driver

To save setup time, you will be working in a previously created database eliminating the need for creating the database and an ODBC driver.

### IIS Configuration

In order to preview pages created in the GMApp, you will need to specify a new virtual directory using the Internet Service Manager Console:

1. From the default website folder, specify a new virtual directory name of GMApp
2. Set the physical directory to x:\applications\projects\GMApp
3. Check off all but the last permission and click Finish
4. Save the settings when you close the ISM Console

### Specify the NeuArchitect Settings for GMApp

After starting NeuArchitect and opening the GMApp.OMD file, you'll need to check all your settings:

5. If you are working with IIS, you will need to append the alias of your virtual directory to the Logical directory under Web server settings. When finished, it should appear as follows:
  - http://localhost/GMApp
6. Next, you will want to select the Application Server tab under Technology Selections. Specify your Application server technology.  
Note: If you Application server technology settings require you to specify an EJB Server, you must click the Properties button next to your EJB server selection and set the HTTP Server Port and Secure Port to 7001.
7. Click next on the Database tab. Specify your ODBC data source name as Sports.  
Note: If you are using SQL Server, you will need to specify a User ID and password. If SQL Server has been installed locally and you have full administrative rights, keying **sa** for the User ID will be sufficient.
8. Create one Class in this application called GM\_Site with the following attributes:
  - GM\_Site\_Info – Text 10
  - GM\_Description – Text 50
9. On the Settings tab for the new class, set Sample rows to 10
10. You will need to Construct:
  - Persistence – remember to click Resolve All and then Construct All
  - Sample Data – be sure to click Construct scripts and add to database
11. At this point it would be logical to Save all your work.

## Graphic Modeling Introduction - Reference

### The Style Repository

The Style Repository is a NeuArchitect application (.OMD file). The style repository is the host for the styling elements:

- Themes
- ControlStyle
- PageStyle

Each NeuArchitect application is allowed one current style for accessing the design attributes contained in a style repository. The graphic designer may either:

- Use an existing style repository
- Create a new style repository

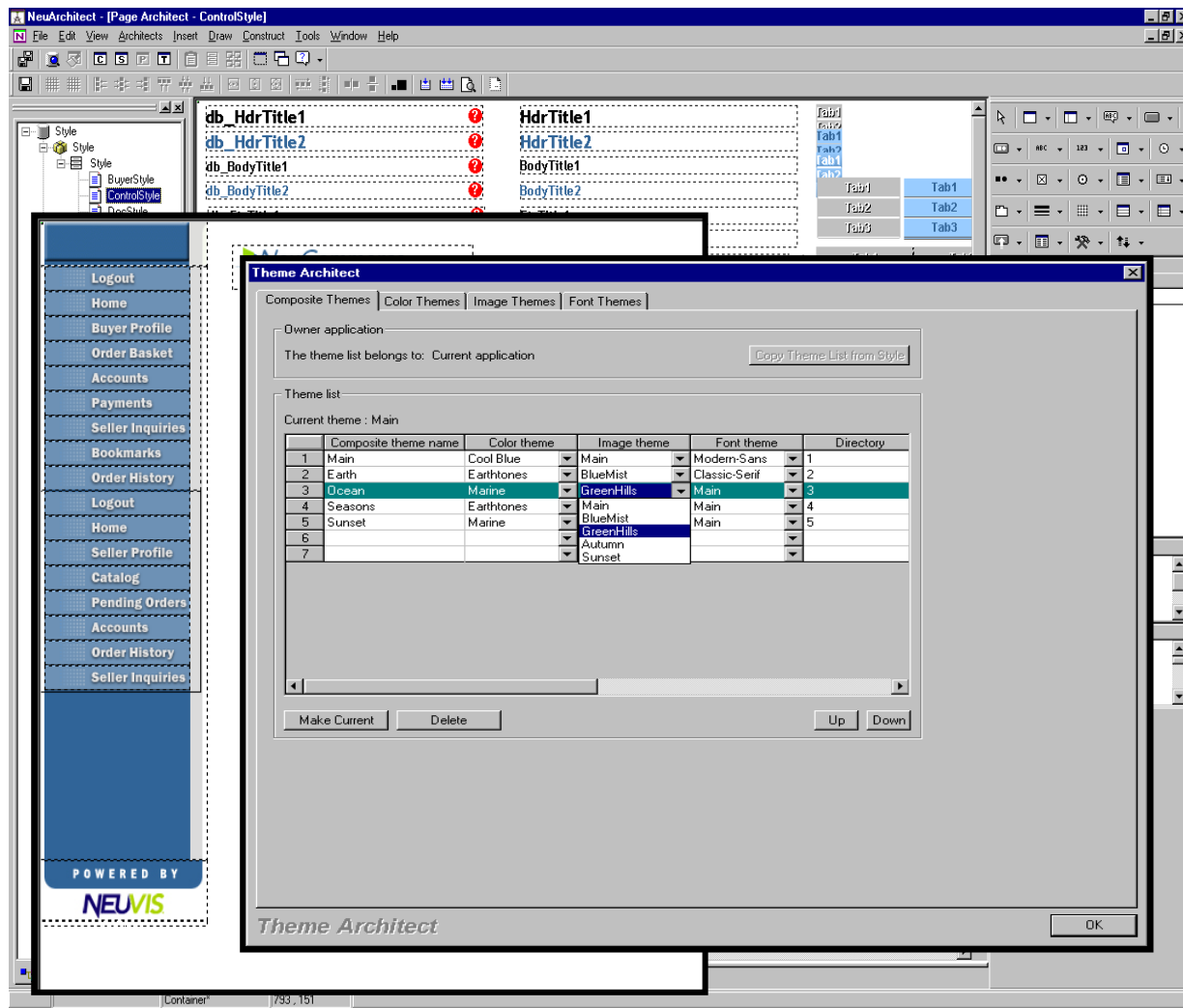


Figure 5 – The Style Repository



## Themes

Themes are created and maintained via Theme Architect. Themes are created and typically maintained in the style repository providing for a consistent 'look and feel' for:

- Colors
- Fonts
- Images

Theme Architect provides the graphic designer the flexibility to:

- Create and maintain themes in the style repository
- Create and maintain themes directly in the business application using the option to Copy Theme List from Style

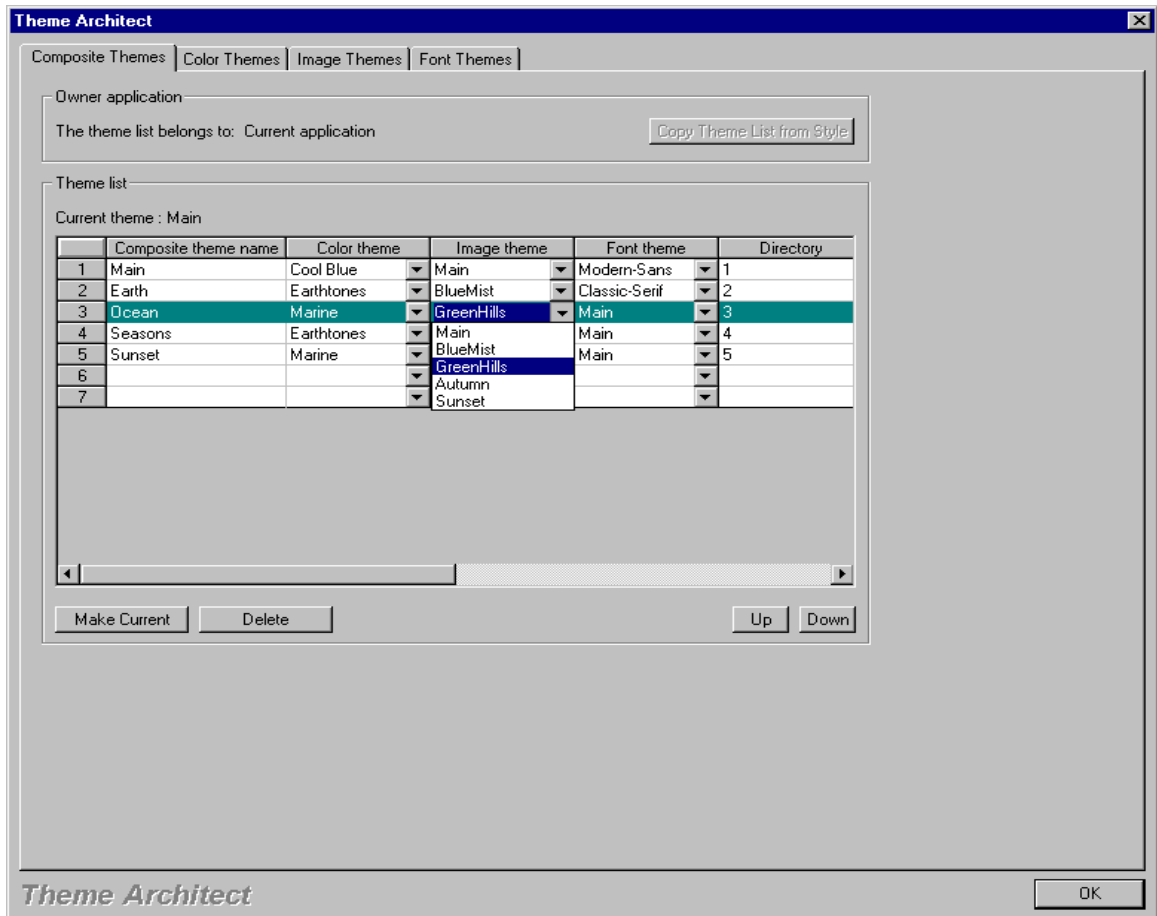


Figure 6 – The Theme Component of the Style Repository

## Page Controls

**Page Controls are the visual elements of the business application (buttons, grids, text, etc.) serving to:**

- **Present information**
- **Format data**
- **Imply action to take**
- **Allow for graphical images to be placed on a page.**

**The Control Style page is created/modified by opening the Style .OMD and using the Page Architect feature.**

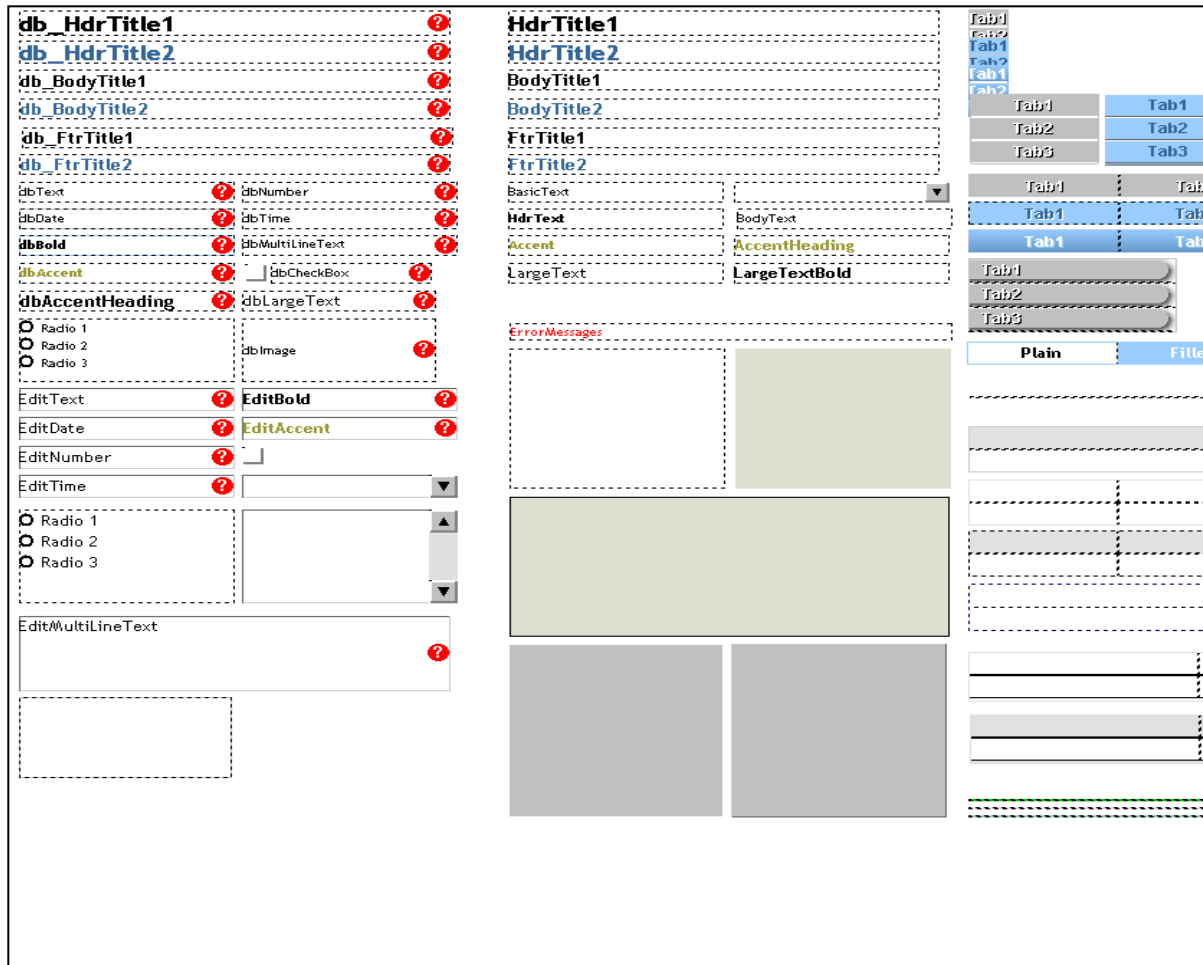


Figure 7 – The Page Control Component of the Style Repository

## Page Styles

**Page Styles provide the ability to pre-place controls, as defined within the Control style page, on a page or pages to create templates. Creating page styles will provide the ability to:**

- Set the default page size and page background
- Maintain a consistent look and feel across all pages within the application.

**Page Styles are created/modified by opening the Style .OMD and using the Page Architect feature.**

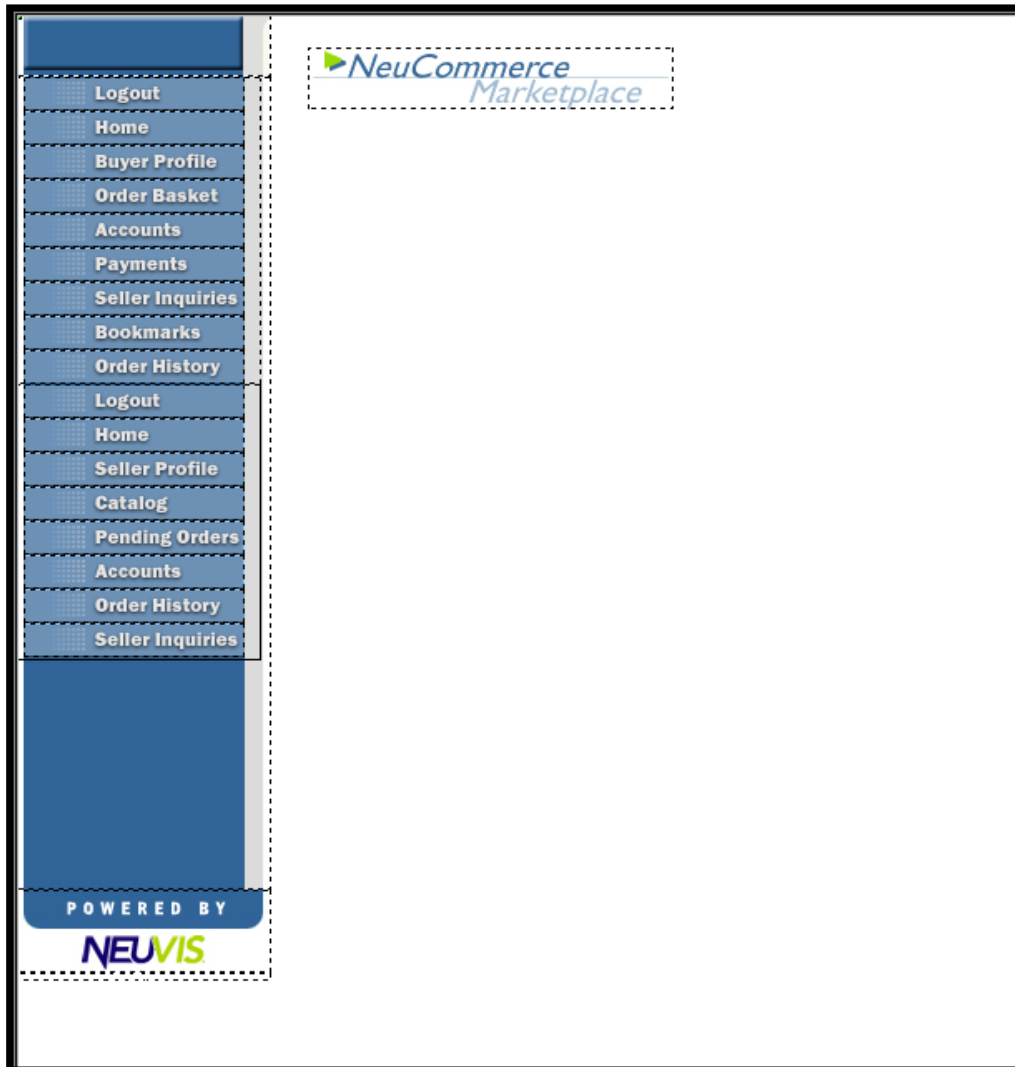


Figure 8 – The Page Style Component of the Style Repository

## Using the Style Repository - Process

The following table maps the suggested process for setting up the NeuArchitect Style Repository modeling components.

<b>Process for setting up and using the NeuArchitect modeling components.</b>	
1	Establish Naming and Directory Setup Conventions - for style repositories, image access and storage preferences on your development machine
2	Create Style Repositories – can be created from scratch or by cloning existing repositories
3	Define Themes – use Theme Architect to define all the required themes in the repository for colors, images, fonts and composite themes
4	Create the ControlStyle – specify the properties and attributes of controls using Theme Architect definitions
5	Design PageStyles – map out various page styles using the properties and attributes defined in Theme Architect and controls created in the ControlStyle
6	Specify the 'Current Style' in your NeuArchitect application – NeuArchitect allows the flexibility of using the style repository of choice
7	Site Modeling and Page Modeling – use the power of NeuArchitect to model the site and individual pages

## Section 2 - NeuArchitect Style Modeling Components

### The Style Repository



***As sections go, this is a pretty long one. You will be learning how to create a Style Repository and all the modeling components***

A style repository is stored like a standard NeuArchitect application. However, it contains styles (i.e. Themes, Controls Style, and Page Styles) to be used in another NeuArchitect business application.

The style repository is only useful when used in conjunction with a NeuArchitect application and only one style repository can be active within the application.



#### **eLearning Flash:**

If you're looking for a quick, 10,000 foot view of the technical material in this section [click here to go to the Section Reference](#), otherwise, continue through the section.

## The Default Style Repository

A default Style repository is supplied when NeuArchitect is installed. The default style repository (SampleStyle.OMD) can be found in its own subdirectory:

\NeuVis\NeuArchitect\Style\SampleStyle.

The figure below shows the Style subdirectory structure created for you application when you save it for the first time. The Style subdirectory is copied from the NeuArchitect install directory. It includes the SampleStyle.OMD, images and source code for the Page Styles.

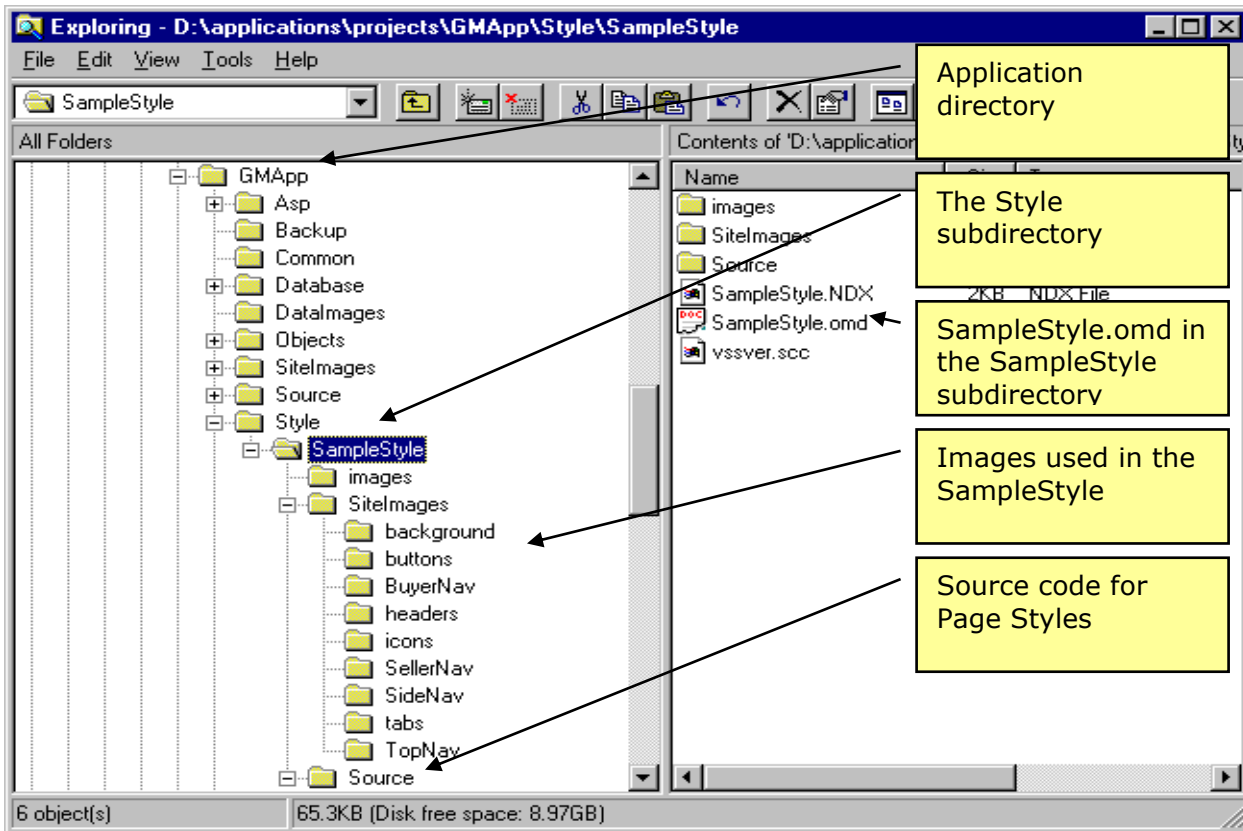


Figure 1 - The Style Repository directory structure

## Style Repository – Naming Conventions

NeuArchitect is installed with a default style repository that is automatically appended to a new NeuArchitect application. The default file is named SampleStyle and is copied to your application directory under the Style subdirectory with the same subdirectory name. If you will be creating several applications requiring different style repositories, you will need to establish meaningful names for the repositories and the directories in which they reside. The names could relate to the specific application or to business functions. The following are samples names for style repositories:

- \Style\BaseStyle\BaseStyle.OMD
- \Style\SportsStyle\SportsStyle
- \Style\CMBStyle\CMBStyle
- \Style\PatrioticStyle\PatrioticStyle

As with a standard NeuArchitect application, each OMD file must be stored in its own directory. The directory name **must** correspond to the name you assign to the style repository application.

When the time comes to start development work on a new NeuArchitect application, any repository that can be accessed by the development machine can be set to the current style for that application. The files will be copied from the host style repository into the subdirectory structure of the new application.

Note that any change to the current style will completely replace the existing style for that application.

## Style Repository Images

Images that are Common to a style are can be made available to an application via a logical name assigned in Theme Architect. The images can be:

- Resident in an image library
- Copied into the Siteimages subdirectory of the style repository.

**Important Note:** When copying a style repository using this technique for storing images, your application will have two paths of Siteimages – one from the style repository and one from the application. NeuArchitect will search the higher path under your application for the images. In order to access the images in the Page Architect Media Palettes and the Display selection tabs for controls, you will have to copy the style repository SiteImages to the Siteimages subdirectory in your application directory.



True or False? The name of the style OMD file and the directory name it resides in must match.

## Creating a Style Repository

When saving a NeuArchitect application for the first time, the default style folder and all it's accompanying files and folders are copied into the file structure of the new application. You can, however, create a new Style Repository as opposed to using or modifying the supplied Style Repository OMD file. Creating a new Style Repository is similar to creating a new application but with fewer steps:

1. Create the new application file
2. Setup the NeuArchitect application infrastructure
3. Use Theme Architect to:
  - Define attributes and files for the primary themes (colors, fonts and images)
  - Map out the composite themes
4. Create the ControlStyle Page in PageArchitect
5. Design the PageStyle templates in PageArchitect

Typically, the designer creates the application and then uses Theme Architect to define the themes. The attributes and files assigned to the colors, fonts and images in Theme Architect can then be applied in designing the ControlStyle and PageStyles.

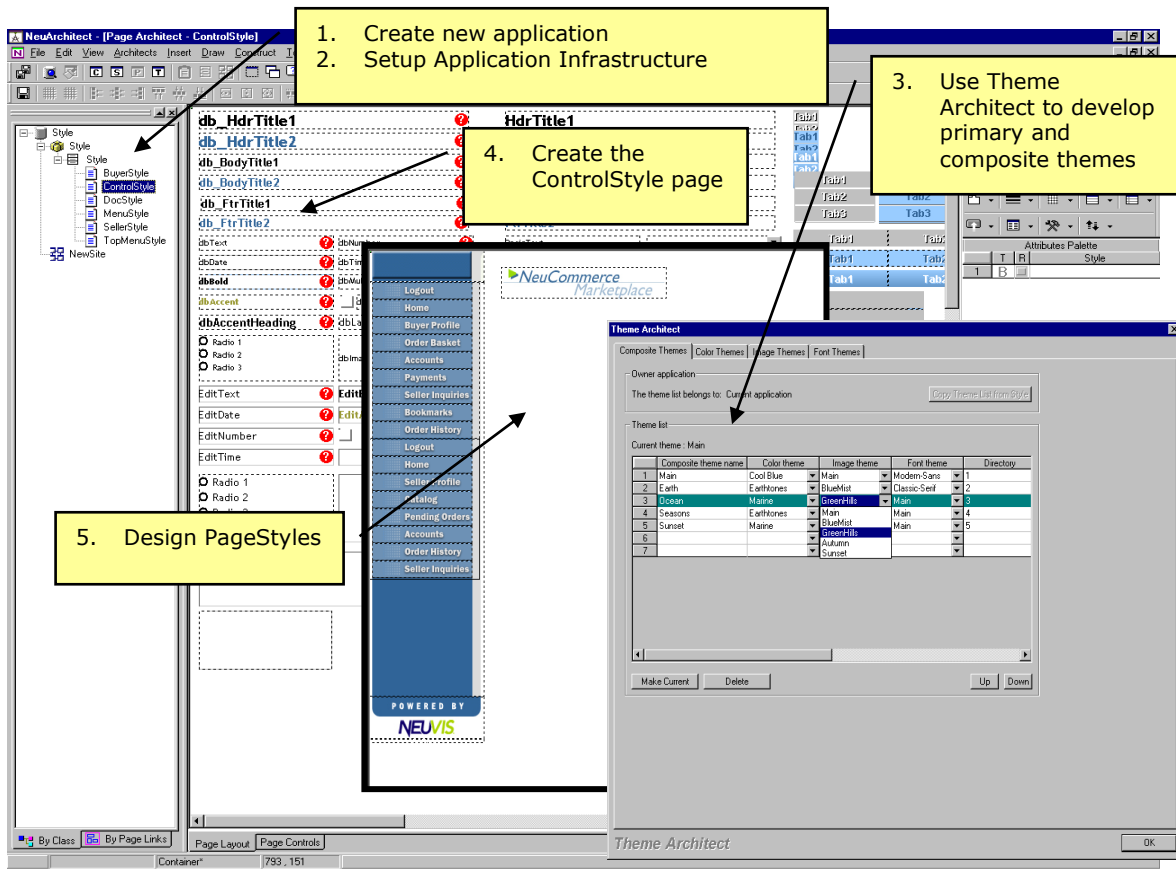


Figure 2 – Creating a New Style Repository



## Creating a New Style Repository for Your NeuArchitect Application - Summary

### First - Create the New Application File

Like any other NeuArchitect application, you will need to create a directory for the new Style Repository. No ODBC driver is required as this application not data connected.

<b>Function</b>	<b>Purpose</b>	<b>Steps</b>
<b>Creating a New Style Repository – Create the application file</b>		
Create the directory	A style repository, like any NeuArchitect must reside as an OMD in its own directory	<ol style="list-style-type: none"> <li>1. Create the application directory (must be the same name as you palm for your OMD file)</li> <li>2. Start NeuArchitect</li> <li>3. Choose Create new application</li> </ol>

### Second - Setup the NeuArchitect Application Infrastructure

Although there are some specific naming requirements and conventions to follow in creating a Style Repository, the process is the same as for any other NeuArchitect application:

<b>Function</b>	<b>Purpose</b>	<b>Steps</b>
<b>Creating a New Style Repository – Setup the NeuArchitect application infrastructure</b>		
Setup the Style Repository application infrastructure	As with any NeuArchitect application, the infrastructure must be setup. In the case of the Style Repository, there are guidelines that must be adhered to specifically (refer to the steps in this table)	<ol style="list-style-type: none"> <li>1. Define the <b>Application name</b>.</li> <li>2. Click the <b>Remove Style</b> button located under the <b>Application tab Import style application</b> section.                       Note: the NeuArchitect supplied SampleStyle will default as the Current style. To avoid confusion when building a new Style Repository <i>always</i> remove any Import style application before saving the new style application.</li> <li>3. Define the Package name as <b>Style</b>.</li> <li>4. Define the Component name as <b>Style</b>.</li> <li>5. Insert a Class and name it <b>Style</b></li> </ol>

## Workshop



### Create a New Style Repository

1. Create a new subdirectory naming it as shown below:  
    \applications\projects\Style\GMStyle
2. Open NeuArchitect
3. Define the **Application name** as GMStyle.
4. Click the **Remove Style** button located under **Import in the style application** on the **Application tab**
5. Define the Package name as **Style**.
6. Define the Component name as **Style**.
7. Insert a Class and name it Style
8. Save the Application as **GMStyle** in the folder you created above.
9. Copy the following image files from your Sports subdirectory into the SiteImages subdirectory under your new **GMStyle** subdirectory.
  - Goaslie.jpg
  - Driver.jpg
  - Glove42737.jpg
  - back.jpg

*NeuArchitect Style Modeling Tools – Workshop 1*

## Defining Themes



**At this point you have created a NeuArchitect application for your Style Repository. Next, you will learn about the first of three modeling components – Theme Architect which itself just happens to have three components.**

A theme provides a pre-defined set of colors, images, and fonts to be used throughout your application. Themes generally evoke an ambiance that you want to convey to users. For example, a:

- Season (spring, summer, fall, winter)
- Holiday (Christmas, 4<sup>th</sup> of July)
- Corporate identity (NeuVis, Coca Cola)
- Genre (industrial, art deco)
- Nation (USA, Italy)
- Demographic group (children, teenagers)
- Market segment (health, insurance).

You may define multiple themes for your application. The colors, fonts, and images used in a theme are constrained only by the abilities of the target device or browser. If multiple themes are defined you may select a theme statically that is applied to all pages within the application, or dynamically program themes to be personalized for individual users. The tool you will use to create, modify, and select themes is called the **Theme Architect**.

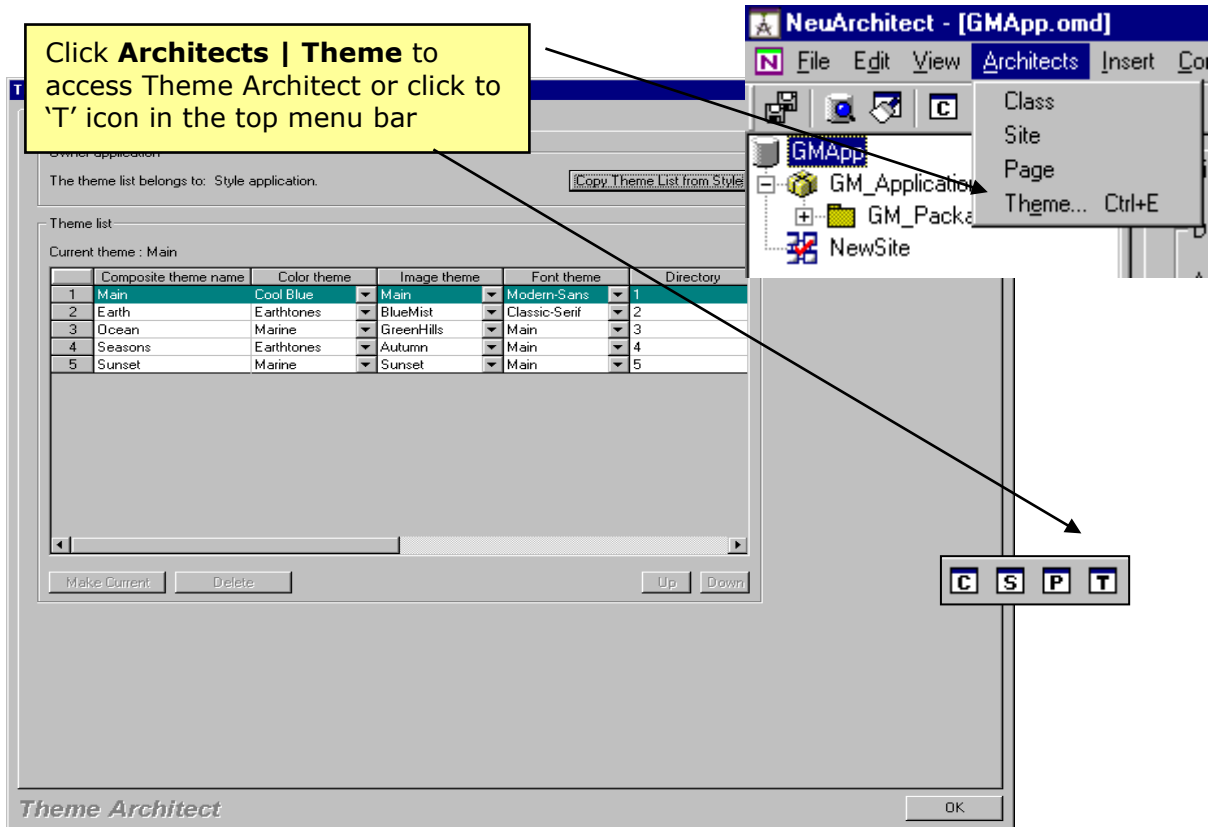
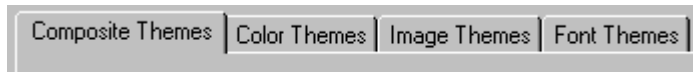


Figure 3 – Accessing Theme Architect

## Creating and Maintaining Themes with Theme Architect

Typically, you work with the three types of primary themes, corresponding to the tabs in Theme Architect: Once you have defined your primary themes, you can then establish composite themes that are defined using various combinations of the primary themes.



While more details will follow in later sections, know now that all three types of primary theme architects allowing you to first define logical names within each primary (for colors, images or fonts). You then any number of themes within the primary theme where each theme is defined with different values (again colors, images or fonts) being assigned to these logical names.

For instance, a logical name for a row color in an ObjectGrid might be RowFill1. For a Marine theme, you might select a light blue for RowFill1. An Earthtone theme might call for a shade of grey to fill your rows.

Color values are RGB colors. Image values are image file names (i.e. .gif, .jpg). Font values consist of font face, size and other font characteristics.

A Composite Theme typically consists of one color theme, one image theme, and one font theme, but, a single primary theme can define a composite theme.

	Composite theme name	Color theme	Image theme	Font theme
1	Main	Cool Blue	Main	Modern-Sans
2	Earth	Earthtones	BlueMist	Classic-Serif
3	Ocean	Marine	GreenHills	Main
4	Seasons	Earthtones	Autumn	Main
5	Sunset	Marine	Sunset	Main

By using different combinations of these primary themes, you can define any number of composite themes. If you create more than one Composite theme you will have the ability to select the desired theme and make it the Current theme for the application. There is one Current theme per application, but you have the option to select different themes at the page level.



True or False? A single composite theme **MUST** have a color, image and font theme.

### Target Device Prerequisites

Whether you are defining Control Styles, Page Styles, or Themes you need to evaluate how the designed controls and pages will be displayed and from where they will be displayed. During the design and layout stage you will need to take into consideration how your website may be viewed such as monitor settings (i.e. 800 x 600 or 1024 x 768, and hand-held devices), web safe colors vs. custom colors, image sizes, and browser compatibility (i.e. Netscape vs Microsoft Internet Explorer). All these considerations can impact the visual appearance as well as user acceptance of the site.

## Theme Architect

Theme Architect enables you to create and store visual elements such as control colors, font styles, and graphic images that can be applied to the global web application, sub-sites off a web application, or whatever your business needs dictate. You can have one Composite Theme or several Composite Themes comprised of one Color, one Font, and one Image Theme. For example: you could create a Color Theme called Summer defined with blues and greens and with a click set the theme to Fall which is defined with browns, yellows, and oranges.

Theme Architect is where you will add, modify, delete, define, and select the themes you want to use within your application. The Theme Architect window is made up of four tabs: Composite Themes, Color Themes, Image Themes, and Font Themes. Going forward we will cover how to access the Theme Architect window and describe each of the tabs features and function. The figure below illustrates what the Composite Theme encompasses in regards to Color, Image, and Font Themes.

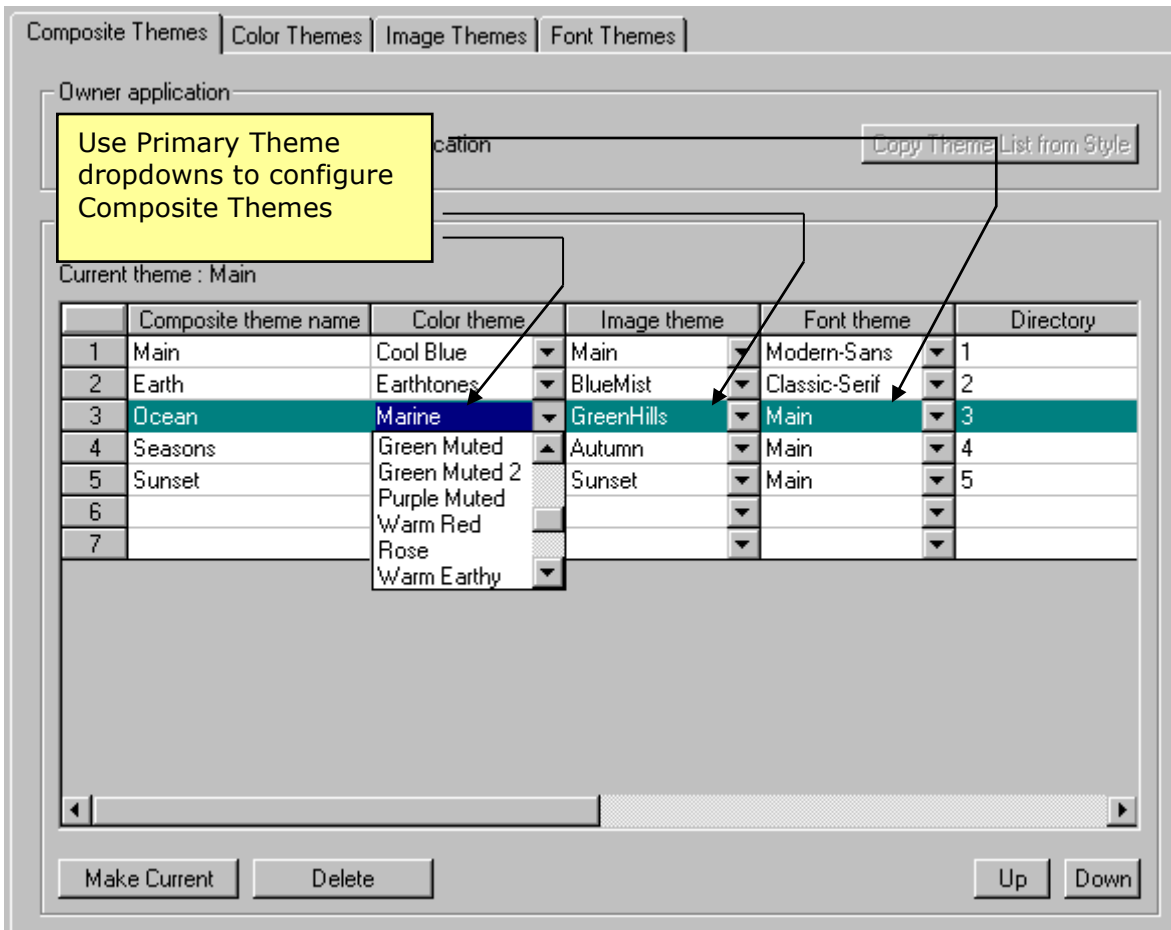


Figure 4 – Primary and Composite Themes in Theme Architect

## Define Themes with Theme Architect

Use Theme Architect to define attributes and files for the primary themes (colors, fonts and images) and to map out the composite themes.

<b>Function</b>	<b>Purpose</b>	<b>Steps</b>
<b>Define Themes with Theme Architect</b>		
Setup primary and composite Themes with Theme Architect	A theme provides a pre-defined set of colors, images, and fonts to be used throughout your application. Themes generally evoke an ambiance(s) that you want to convey to users.	<ol style="list-style-type: none"> <li>1. Click the <b>Architects</b> icon from the NeuArchitect top menu</li> <li>2. Select Theme</li> <li>3. Define primary themes                             <ul style="list-style-type: none"> <li>• Select the Colors Tab - for each primary color theme                                     <ul style="list-style-type: none"> <li>○ Supply a logical color theme name</li> <li>○ Complete the list of colors associating logical color names with desired attributes</li> </ul> </li> <li>• Select the Images tab - for each primary image theme                                     <ul style="list-style-type: none"> <li>○ Supply a logical image theme name</li> <li>○ Complete the list of images associating logical image names with image files</li> </ul> </li> <li>• Select the Fonts tab - for each primary font theme                                     <ul style="list-style-type: none"> <li>○ Supply a logical font theme name</li> <li>○ Complete the list of fonts associating logical font names with font attributes</li> </ul> </li> </ul> </li> <li>4. Select the Composite Themes tab - for each composite theme                             <ul style="list-style-type: none"> <li>• Supply a logical composite theme name</li> <li>• Use the dropdowns to associate a:                                     <ul style="list-style-type: none"> <li>○ Primary color theme</li> <li>○ Primary image theme</li> <li>○ Primary font theme</li> <li>○ Directory name (optional) to be resident under the ASP directory for constructing multiple themes</li> </ul> </li> </ul> </li> </ol>

## Theme Architect - Color Themes



**This is the first modeling component of Theme Architect.**

Color themes are logical names specified for a given color (RGB) combination. The Color Themes tab will enable you to define the colors you want to apply to as page backgrounds, control fills, borders, and shadows.

You may create one or several color themes. Keep in mind once you have established the first color theme, the logical color names you have established will be used in creating any additional color themes. The actual color values can be changed for different themes as well as dynamically changed to personalize your application.

**Color Themes**

Composite Themes | **Color Themes** | Image Themes | Font Themes

Themes

- 1 Nature
- 2 Earthtones
- 3 **Cool Vibrant**
- 4 Cool Electric
- 5 Pastel
- 6 Warm Vibrant

Delete Theme Duplicate

Override the color defined with a Java expression

	Color Name	Red	Green	Blue	R G B	Actual Color	Browse	Color Expression
1	Background	255	255	255	FFFFFF		...	
2	Shine	232	232	232	E8E8E8		...	
3	Shade	102	102	102	666666		...	
4	Border	0	0	0	000000		...	
5	Accent	51	102	255	3366FF		...	
6	Required	255	0	0	FF0000		...	
7	Font-std	0	0	0	000000		...	
8	Font-light	255	255	255	FFFFFF		...	
9	Font-dark	102	102	102	666666		...	
10	HdrFill-light	128	0	64	800040		...	
11	HdrFill-dark	128	0	64	800040		...	
12	BodyFill-light	128	64	64	804040		...	
13	BodyFill-dark	128	64	64	804040		...	
14	FtrFill-light	181	125	175	857DAF		...	
15	FtrFill-dark	173	132	174	AD8AAE		...	
16	RowFill1	204	204	153	CCCC99		...	
17	RowFill2	241	241	241	F1F1F1		...	
18	RowFill3	175	220	220	AFDCDC		...	
19	ALINK	51	102	255	3366FF		...	

Color Attributes to associate with the logical color names

Logical Color Names

Color selector:  Custom  Web safe

Theme Architect




Figure 5 - Theme Architect - Color Themes



True or False? Once an RGB color combination is specified for a logical color name, the color must remain the same through all the color themes.

## Color Theme Features


The following table lists the features for Theme Architect Color Themes.

<b>Feature</b>	<b>Color Theme Function</b>
<b>Color Themes</b>	
<b>Color Theme List</b>	The name of the Color Themes created for the application.
<b>Delete Theme</b>	Select the desired Color Theme and click the Delete Theme button to remove it from the Color Theme List.
<b>Duplicate</b>	Select the desired Color Theme and click Duplicate button to create a copy. The duplicated Color Theme will be given a unique name that can be changed.
<b>Theme colors</b>	
<b>Current theme:</b>	Name of the Composite Theme that is currently being used by the application. You may select only one Composite Theme as Current per application.
<b>Color Name</b>	Logical color name that applies to a control and is applied across the application. All Color Themes will use the same Color name, but may be defined with different color values. The Color Name must be unique.
<b>RGB</b>	The alphanumeric value for the actual color defined.
<b>Actual Color</b>	The visual color for the defined RGB value.
<b>Browse</b>	Click the browse button  to select RGB value from the color palette.
<b>Color Expression</b>	An optional setting that can contain a Java expression specifying parameters in which to override the color value defined.
<b>Delete Color</b>	Select the desired Color name and click the Delete Color button to permanently remove it from the application.
<b>Color selector</b>	
<b>Custom</b>	When selected and the browse button  is launched the custom color palette will display.
<b>Web safe</b>	When selected and the browse button  is launched the web safe color palette will display.




## Creating and Maintaining Color Themes in Theme Architect

Use the following table as a guideline for building and maintaining color themes in Theme Architect

<b>Maintaining Color Themes in Theme Architect</b>	
<p><b>Add a Color Theme</b> - The name entered for the <b>Color Name</b> will be the same across all Color Themes. You may select different color values for each theme. The color palette that appears when the browse button  is applied is dependant on the Color selector that is enabled Custom vs. web safe.</p>	
<b>1</b>	Click on an empty line under <b>Themes</b>
<b>2</b>	Enter the <b>Color Theme Name</b>
<b>3</b>	From the <b>Theme colors</b> section enter the <b>Color Name</b> (i.e. Background, Border, Shade, Fill). For gradient and pattern fills you must indicate two colors, e.g. Background and Background1 or Fill2-dark and Fill2-light.
<b>4</b>	<p>Establish the color by entering values for one of the following:</p> <ul style="list-style-type: none"> <li>• Red, Green, Blue RGB values.</li> <li>• RGB alphanumeric value.</li> </ul> <p>Click Browse and select the actual color from the color palette.</p>
<p><b>Delete a Color Theme</b></p>	
<b>1</b>	Click to highlight the <b>Color Theme Name</b> under the <b>Themes</b> section
<b>2</b>	Click the <b>Delete Theme</b> button
<p><b>Duplicate a Color Theme</b> – Important Note: Deleting a <b>Color Name</b> from the Theme colors section of your duplicate theme will delete that name for all Color Themes within the application. Adding a <b>Color Name</b> to the Color Themes section will apply to all Color Themes established and/or added within the application. The color value default will equal black unless otherwise specified for each Color Theme</p>	
<b>1</b>	Click to highlight the <b>Color Theme Name</b> under the <b>Themes</b> section
<b>2</b>	Click the <b>Duplicate</b> button. The duplicated <b>Color Theme</b> retains the original name of the theme and a numeric value is appended at the end to keep it unique. You can change the name to something more relevant to your application
<b>3</b>	Modify the duplicated Color Theme as needed



## Theme Architect – Adding Color Themes to the Style Repository

1. With the GMStyle application open, click the **Architects** icon from the NeuArchitect top menu 
2. Select the Color Themes tab
3. Use Theme Architect to define 4 colors for each of the primary color themes specified below.

Note, Your own choice of colors may be substituted for the Red, Green and Blues shown in the table.

Primary Color Theme Name	Color Name	Red	Green	Blue
<b>Nature</b>	Fill-Light	182	182	146
	Fill-Dark	129	129	86
	Shade	102	102	102
	Shine	204	204	204
	Background	255	255	255
<b>Earthtones</b>	Fill-Light	188	188	88
	Fill-Dark	124	124	124
	Shade	102	102	102
	Shine	255	255	255
	Background	255	255	255
<b>Vibrant</b>	Fill-Light	228	153	153
	Fill-Dark	128	64	64
	Shade	102	102	102
	Shine	204	204	204
	Background	255	255	255

**Be sure to save your work!**

*NeuArchitect Style Modeling Tools – Workshop 2*

## Theme Architect - Image Themes



**This is the second modeling component of Theme Architect.**

Image themes are logical names specified for a given image files (.gif, .jpg). The **Image Themes** tab will enable you to change and/or update images used in your application in a batch process. This feature will greatly assist you in maintaining images used within your application that may change on a frequent basis. For example: you may have a web page that holds product images that are updated each season, you can establish seasonal Image Themes and apply that new theme under the Composite Theme. By using this process you would avoid having to manually change each of the images page by page.

Keep in mind once you have established the first Image Theme the logical Image names you have established will be used in creating any additional image themes. For other image themes, you can set the Image Path to different folders that house the desired graphic file as well as dynamically changed to personalize your application.


The screenshot shows the Theme Architect interface with the 'Image Themes' tab selected. An 'Open File' dialog is open, showing a list of image files in the 'TopNav' folder. The 'Image Themes' tab contains a list of themes and an 'Images' table. The 'Images' table has columns for 'Image Name', 'Image Path', 'Browse', and 'Image Expression'. Annotations highlight the logical names for images and how to override them with Java expressions.

Image Name	Image Path	Browse	Image Expression
20 TopNav_aboutus_on	.\Siteimages\TopNav\TopNav_about_on.jpg	...	
21 TopNav_catalogs	.\Siteimages\TopNav\TopNav_catalog.jpg	...	
22 TopNav_catalogs_mo	.\Siteimages\TopNav\TopNav_catalog_on.jpg	...	
23 TopNav_catalogs_on	.\Siteimages\TopNav\TopNav_catalog_on.jpg	...	
24 TopNav_contactus	.\Siteimages\TopNav\TopNav_contact.jpg	...	
25 TopNav_contactus_mo	.\Siteimages\TopNav\TopNav_contact_on.jpg	...	
26 TopNav_contactus_on	.\Siteimages\TopNav\TopNav_contact_on.jpg	...	
27 TopNav_forum	.\Siteimages\TopNav\TopNav_forum.jpg	...	
28 TopNav_forum_mo	.\Siteimages\TopNav\TopNav_forum_on.jpg	...	
29 TopNav_forum_on	.\Siteimages\TopNav\TopNav_forum_on.jpg	...	
30 TopNav_home	.\Siteimages\TopNav\TopNav_home.jpg	...	
31 TopNav_home_mo	.\Siteimages\TopNav\TopNav_home_on.jpg	...	
32 TopNav_home_on	.\Siteimages\TopNav\TopNav_home_on.jpg	...	
33 TopNav_login	.\Siteimages\TopNav\TopNav_login.jpg	...	
34 TopNav_login_mo	.\Siteimages\TopNav\TopNav_login_on.jpg	...	
35 TopNav_login_on	.\Siteimages\TopNav\TopNav_login_on.jpg	...	
36 TopNav_quicksearch	.\Siteimages\TopNav\TopNav_srch.jpg	...	
37 TopNav_quicksearch_m	.\Siteimages\TopNav\TopNav_srch_on.jpg	...	
38 TopNav_quicksearch_or	.\Siteimages\TopNav\TopNav_srch_on.jpg	...	
39 TopNav_accountinfo	.\Siteimages\TopNav\TopNav_accountinfo.jpg	...	

Figure 6 - Theme Architect Font Themes

## Image Theme Features

The following table lists the features for Theme Architect Image Themes.

<b>Feature</b>	<b>Image Theme Function</b>
<b>Themes</b>	
<b>Image Theme List</b>	The names of the Image Themes created for the application.
<b>Delete Theme</b>	Select the desired Image Theme and click the Delete Theme button to remove it from the Image Theme List.
<b>Duplicate</b>	Select the desired Image Theme and click the Duplicate button to create a copy. The duplicated Image Theme will be given a unique name that can be changed.
<b>Images</b>	
<b>Image Name</b>	Logical Image name that applies to an image and is applied across the application. All Image Themes will use the same image name, but may be defined with different image files.
<b>Image Path</b>	The address or URL where the image file resides.
<b>Browse</b>	Click the browse button  to select the address/URL for the Image Path.
<b>Image Expression</b>	An optional setting that can contain a Java expression specifying parameters in which to override the image defined.
<b>Delete Image</b>	Select the desired Image name and click the Delete Image button to permanently remove it from the application.



True or False? An image can can either be a file or a URL..

## Creating and Maintaining Image Themes in Theme Architect

Use the following table as a guideline for building and maintaining Image themes in Theme Architect

<b>Maintaining Color Themes in Theme Architect</b>	
<b>Add an Image Theme</b> - The name entered for the <b>Image Name</b> will be the same across all Image Themes, you will use the <b>Image Path</b> to distinguish where to access new images for each Image theme	
<b>1</b>	Click on an empty line under <b>Themes</b>
<b>2</b>	Enter the <b>Image Theme Name</b>
<b>3</b>	From the <b>Images</b> section enter the <b>Image Name</b>
<b>4</b>	Enter the <b>Image Path</b> or use <b>Browse</b> to locate the correct path where the image file resides.
<b>Delete an Image Theme</b>	
<b>1</b>	Click to highlight the <b>Image Theme Name</b> under the <b>Themes</b> section
<b>2</b>	Click the <b>Delete Theme</b> button
<b>Duplicate an Image Theme</b> - Important Note: Deleting an image from the Theme image section of your duplicate theme will delete that name for all Image Themes within the application. Adding an <b>Image Name</b> to the Image Themes section will apply to all Image Themes established and/or added within the application.	
<b>1</b>	Click to highlight the Image Theme Name under the <b>Themes</b> section.
<b>2</b>	Click the <b>Duplicate</b> button. The duplicated Image Theme will retain the original name of the theme and append a numeric value at the end to keep the <b>Image Theme Name</b> unique. You may change the name to something more relevant to your application
<b>3</b>	Modify the duplicated Image Theme as needed

## Workshop



### Theme Architect – Adding Image Themes to the Style Repository

1. With the GMStyle application open, click the **Architects** icon from the NeuArchitect top menu  menu
2. Select the Image Themes tab
3. Use Theme Architect to define 2 images for each of the primary image themes specified below.

Primary Image Theme Name	Image Name	Image
<b>Hockey</b>	Main-Image	/GMStyle/SiteImages/goalie.jpg
	Back-Button	/GMStyle/SiteImages Back.jpg
<b>Golf</b>	Main-Image	/GMStyle/SiteImages Driver.jpg
	Back-Button	/GMStyle/SiteImages Back.jpg
<b>Baseball</b>	Main-Image	/GMStyle/SiteImages Glove42737.jpg
	Back-Button	/GMStyle/SiteImages Back.jpg

**Be sure to save your work!**

*NeuArchitect Style Modeling Tools – Workshop 3*

## Theme Architect - Font Themes



**This is the third modeling component of Theme Architect.**

Font themes are logical names specified for a given font characteristics. The Font Themes tab will enable you to define the font styles (i.e. Arial, Verdana) and characteristics (i.e. font size, bold) that you want to apply to the control labels in your application.

Keep in mind once you have established the first Font Theme the logical Font Names you have established will be used in creating any additional font themes. The actual font values can be changed for different themes as well as dynamically changed to personalize your application.

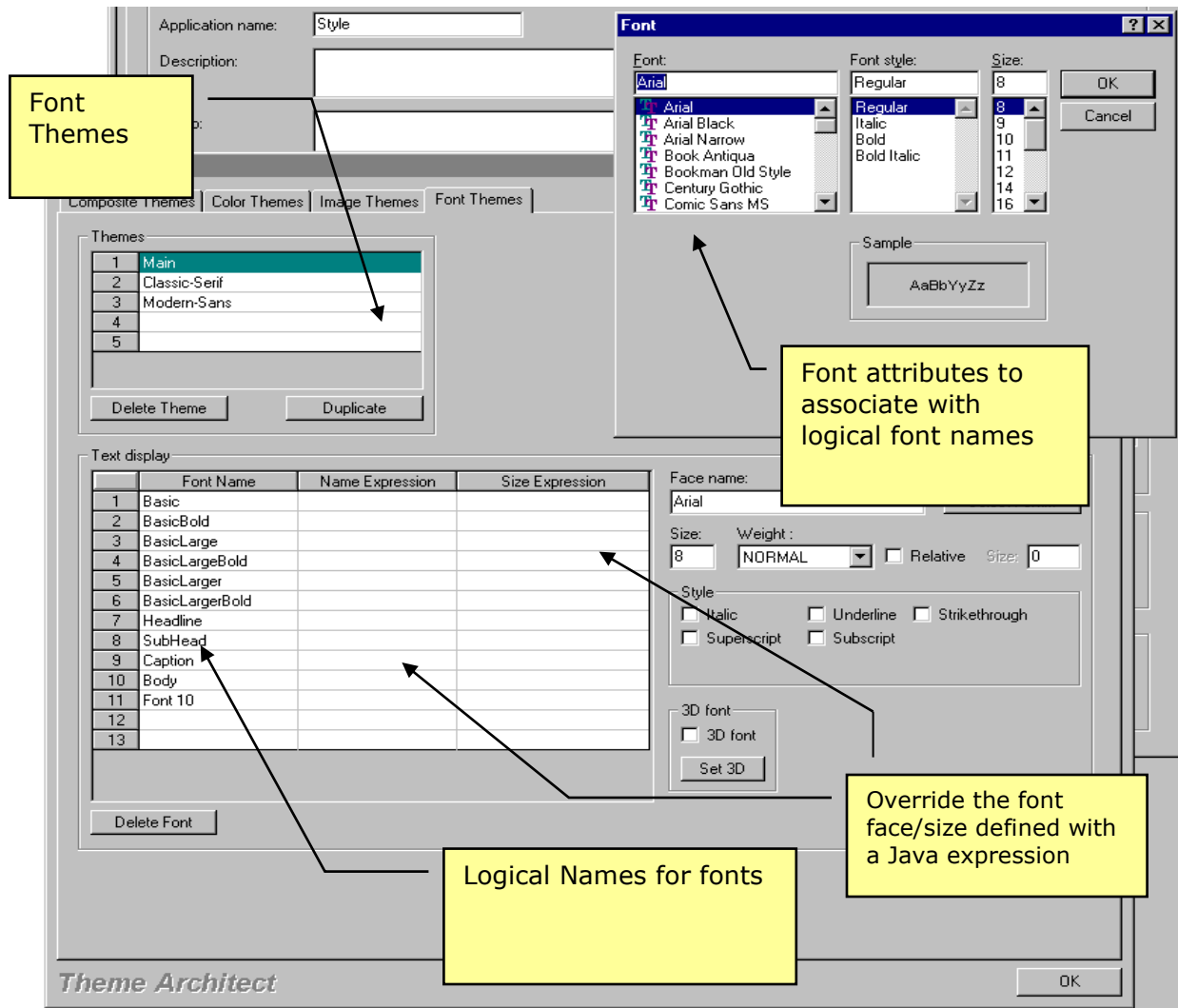


Figure 7 – Theme Architect Font Themes

## Image Theme Features

The following table lists the features for Theme Architect Image Themes.

<b>Feature</b>	<b>Ont Theme Function</b>
<b>Themes</b>	
<b>Font Theme List</b>	The name of the Font Themes created for the application.
<b>Delete Theme</b>	Select the desired Font Theme and click the Delete Theme button to remove it from the Font Theme List.
<b>Duplicate</b>	Select the desired Font Theme and click Duplicate to create a copy. The duplicated Font Theme will be given a unique name that can be changed.
<b>Text display</b>	
<b>Font Name</b>	Logical Font Name that applies to a control label and is applied across the application. All Font Themes will use the same Font Name, but may be defined with different font styles.
<b>Name Expression</b>	An optional setting that can contain a Java expression specifying parameters in which to override the font face defined.
<b>Size Expression</b>	An optional setting that can contain a Java expression specifying parameters in which to override the font size defined.
<b>Delete Font</b>	Select the desired Font Name and click the Delete Font button to permanently remove it from the application.
<b>Text Characteristics</b>	
<b>Face Name</b>	Name of the Font (i.e. Arial, Verdana).
<b>Select Font</b>	Click Select Font button to select the desired Font from the standard windows font dialog box.
<b>Size</b>	The absolute font size.
<b>Weight</b>	The weight of the font selected: Bold, Semi-Bold, Heavy, etc.
<b>Relative</b>	Click the Relative check box to enable the Relative Size. This will override the absolute size.
<b>Relative Size</b>	The relative font size as defined for HTML as a number value 1 – 7 (where 1 is small and 7 is large).
<b>Style</b>	Click the desired Style check boxes: Italic, Underline, Strikethrough, Superscript, and/or Subscript.
<b>3D font</b>	Click the 3D font check box to enable the use of 3D fonts.
<b>Set 3D</b>	Click the Set 3D button to select the 3D font to apply.



## Creating and Maintaining Font Themes in Theme Architect

Use the following table as a guideline for building and maintaining Image themes in Theme Architect

<b>Maintaining Color Themes in Theme Architect</b>	
<b>Add a Font Theme</b> - The name entered for the <b>Font Name</b> will be the same across all Font Themes. You may select different Font values for each Font theme	
<b>1</b>	Click on an empty line under <b>Themes</b>
<b>2</b>	Enter the Font Theme Name
<b>3</b>	From the <b>Text display</b> section enter the <b>Font Name</b>
<b>4</b>	Apply the appropriate Font styles and characteristics using the options to the right of the <b>Text display</b> (i.e. Font face, Size, Style, etc.).
<b>Delete a Font Theme</b>	
<b>1</b>	Click to highlight the Font Theme Name under the <b>Themes</b> section
<b>2</b>	Click the <b>Delete Theme</b> button.
<b>Duplicate a Font Theme</b> - Deleting a <b>Font Name</b> from the Text display section will delete that name for all Font Themes within the application. Adding a <b>Font Name</b> to the Text display section will apply to all Font Themes established and/or added within the application. The font value default will equal Arial size 10 regular unless otherwise specified for each Font Theme	
<b>1</b>	Click to highlight the Font Theme Name under the <b>Themes</b> section.
<b>2</b>	Click the <b>Duplicate</b> button. The duplicated <b>Font Theme</b> will retain the original name of the theme and append a numeric value at the end to keep it unique. You may change the name to something more relevant to your application
<b>3</b>	Modify the duplicated Font Theme as needed




True or False? To duplicate a font, select the desired Font Theme and click Duplicate to create a copy. The duplicated Font Theme will be given a unique name that can be changed.

## Workshop



### Theme Architect – Adding Font Themes to the Style Repository

1. With the GMStyle application open, click the **Architects** icon from the NeuArchitect top menu  menu
2. Select the Font Themes tab
3. Use Theme Architect to define 2 fonts for each of the primary font themes specified below.

Note, Your own choice of fonts may be substituted for the those suggested in the table.

Primary Font Theme Name	Font Name	Font
<b>Arial</b>	Header	Arial Black 16 point
	Text	Arial 10 point
<b>TimesRoman</b>	Header	Arial Black 16 point
	Text	Times New Roman 10 point
<b>Verdana</b>	Header	Arial Black 16 point
	Text	Verdana 10 point

**Be sure to save your work!**

*NeuArchitect Style Modeling Tools – Workshop 4*

## Defining Composite Themes



**This is where you will learn how to wrap colors, images and fonts into Composite Themes.**

A Composite Theme is a combination of a Color Theme, an Image Theme, and a Font Theme. You may create one or several Composite Themes as dictated by your business needs. You must first create the Color, Image, and/or Font Themes in order to associate them to the Composite Theme. You may assign only one current theme at a time, but you can assign themes to individual pages in Page Properties. To enable multiple themes to be constructed and saved for an application, each Composite Theme can be assigned its own construction directory.

The screenshot shows the 'Theme Architect' dialog box with several tabs: 'Composite Themes', 'Color Themes', 'Image Themes', and 'Font Themes'. The 'Composite Themes' tab is active. It contains a section for 'Owner application' with a dropdown set to 'Current application' and a 'Copy Theme List from Style' button. Below this is a 'Theme list' section showing a table of composite themes. The 'Current theme' is set to 'Main'. The table has columns for 'Composite theme name', 'Color theme', 'Image theme', 'Font theme', and 'Directory'. A dropdown menu is open for the 'Color theme' column of the first row, showing options like 'Cool Electric', 'Earthtones', 'Nature', 'Cool Vibrant', 'Cool Electric', 'Pastel', and 'Warm Vibrant'. At the bottom, there are 'Make Current', 'Delete', 'Up', and 'Down' buttons. Annotations with arrows point to various parts of the interface, explaining their functions.

	Composite theme name	Color theme	Image theme	Font theme	Directory
1	Main	Cool Electric	Main	Modern-Sans	Main
2	Earth	Earthtones	Main	Classic-Serif	Earth
3	Ocean	Nature	Main	Main	Ocean
4		Earthtones			
5		Cool Vibrant			

**Annotations:**

- Current theme for your application
- Makes Theme list part of the current application
- List of Composite Names
- For multiple Themes, the directory path in which the Theme is to be constructed
- Dropdown selectors for specifying primary themes within each composite theme
- Make Current button to set the current theme for your application
- Up and Down Buttons are used to navigate the list of composite themes
- Delete Button to remove a composite style from the repository

Figure 8 – Setting Composite Theme Parameters for Your Application

## Composite Theme Features

The following table lists the features for Theme Architect Composite Themes.

<b>Feature</b>	<b>Function</b>
<b>Owner application</b>	
<b>The theme list belongs to</b>	The name of the application that contains the Theme definition(s).
<b>Copy Theme List from Style</b>	Enables the defined Themes contained within the owning application.
<b>Theme list</b>	
<b>Current theme:</b>	Name of the Composite Theme that is currently being used by the application. You may select only one Composite Theme as Current per application.
<b>Composite theme name</b>	Where the Composite Theme name is entered, this is a user-defined name. The Composite Theme name must be unique.
<b>Color theme</b>	The name of the established Color Theme to be associated with the Composite Theme, accessed by selecting from the drop-down list box. You may select only one Color Theme per Composite Theme.
<b>Image theme</b>	The name of the established Image Theme to be associated with the Composite Theme, accessed by selecting from the drop-down list box. You may select only one Image Theme per Composite Theme.
<b>Font theme</b>	The name of the established Font Theme to be associated with the Composite Theme, accessed by selecting from the drop-down list box. You may select only one Font Theme per Composite Theme.
<b>Directory</b>	The directory path in which the Composite Theme is to be constructed. This should be used when the application is being constructed simultaneously for multiple themes.
<b>Make Current</b>	Select the desired Composite Theme and click on the Make Current button to enable the theme to serve as the application's overall theme.
<b>Delete</b>	Select the Composite Theme and click on the Delete button to permanently remove the Composite Theme from the application.
<b>Up</b>	Select the Composite Theme and click the Up button to change the order of the Composite Themes.
<b>Down</b>	Select the Composite Theme and click the Down button to change the order of the Composite Themes.

## Creating and Maintaining Composite Themes in Theme Architect

Use the following table as a guideline for building and maintaining Composite themes in Theme Architect

<b>Function</b>	<b>Purpose</b>	<b>Steps</b>
<b>Managing Composite Themes in Theme Architect</b>		
Remove a composite theme	Discard a Theme from your application or from the Style OMD file	<ol style="list-style-type: none"> <li>1. From the NeuArchitect top menu, select Architects</li> <li>2. Select Themes</li> <li>3. Select the theme from the Theme list</li> <li>4. Click Delete Button</li> </ol>
Create a new composite theme	Give your application a second, third, etc. look	<ol style="list-style-type: none"> <li>1. From the NeuArchitect top menu, select Architects</li> <li>2. Select Themes</li> <li>3. Click in the first empty row in the Theme list</li> <li>4. Supply the composite theme name and use dropdowns to select primary themes</li> </ol>
Modify a composite theme	Change the primary color, font or image theme for the current composite theme	<ol style="list-style-type: none"> <li>1. From the NeuArchitect top menu, select Architects</li> <li>2. Select Themes</li> <li>3. Select the theme to be modified</li> <li>4. Use dropdowns to select primary themes</li> </ol>
Specify directory path for the composite theme	Supplies the directory path in which the composite theme is to be constructed. This should be used only when the application is being constructed simultaneously for multiple themes.	<ol style="list-style-type: none"> <li>1. From the NeuArchitect top menu, select Architects</li> <li>2. Select Themes</li> <li>3. Select the theme to be constructed simultaneously</li> <li>4. Supply the directory name – should be a meaningful name which will be constructed as a subdirectory of your application ASP directory</li> <li>5. Follow Page Level instructions in <b>Managing Themes at the Page Level in Page Architect</b></li> </ol>




True or False? Double click on the desired Composite Theme or click on the Make Current button to enable the theme to serve as the application's overall theme.

## Workshop



### Theme Architect – Adding Composite Themes to the Style Repository

1. With the GMStyle application open, click the **Architects** icon from the NeuArchitect top menu  menu
2. Select the Composite Themes tab
3. Use Theme Architect to define 3 composite themes as shown below.

Note, Your own choice of primary themes may be substituted for the those suggested in the table.

Composite Theme Name	Color Theme	Image Theme	Font Theme
<b>Hockey</b>	Vibrant	Hockey	Arial
<b>Golf</b>	Earthtones	Golf	TimesRoman
<b>Baseball</b>	Nature	BaseBall	Verdana

4. Using the Make Current button, set the current theme to **Golf**.

**Be sure to save your work!**

*NeuArchitect Style Modeling Tools – Workshop 5*

## Control Styles



**This is the second modeling component of the Style Repository. Remember, in this phase you will reference logical names specified in Theme Architect for your controls, unleashing more of the dynamic power of NeuArchitect Styles!**

Control Styles are pages that specify the styles of commonly used controls to build the pages in your web application. Controls created and added to the Control Style page will be available from the Control palette for designing your web pages.

Controls on a page are the visual elements of the application. A few of the functions that controls serve are to

- Present information
- Format data
- Imply action to take

Allow for graphical images to be placed on a page.

The Control Style page is created/modified by opening the Style .OMD and using the Page Architect feature. You cannot change this Page from within your business application. Page Architect supports over two-dozen types of controls and for each NeuArchitect control you may define one or more control styles.

Control styles describe the visual framework for commonly used controls. Control styles also set control specific characteristics such as size, shape, border, and fill styles. A number of control styles are generally specified for each type of control. For example, some of the control styles for a label control may be: BasicText, BodyText, HdrTitle1, etc. These styles are accessed via dropdowns when you select a control.

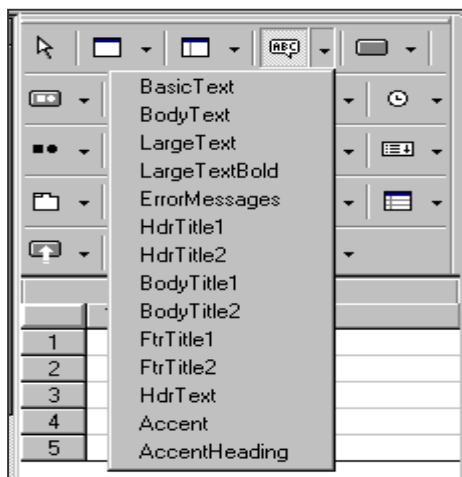


Figure 9 – Styles available for the textbox control

These control styles use the logical names from Theme Architect for color, image and fonts. Therefore, by changing the theme, you can change the look of the control.

Control styles are designed by placing all the styles on a control style page. When you are modeling a page in your application, you are linked to the control style page as defined in the imported style repository, and will have use of all the control styles defined within the control style page. There is one control style page per application.

## Control Style Concepts

The overall concept of creating a Control Style page is to eliminate the need to re-create controls for each page you will design. By applying the Control Styles you will have the necessary control types you need throughout the application. Two major benefits afforded to you is consistency of the look and feel as well as increased productivity by having pre-defined controls at your fingertips. The figure below illustrates some of the types of controls that can be housed in the Control Style page.

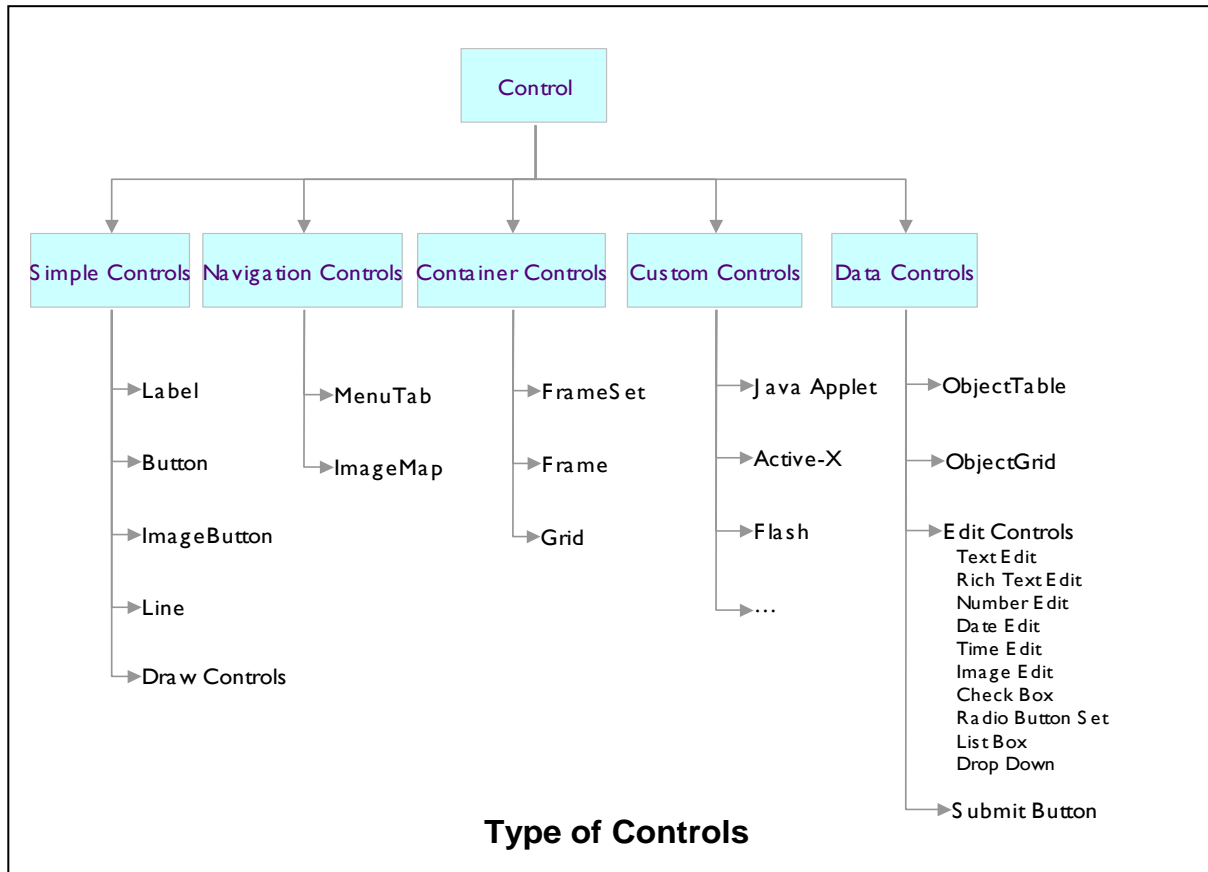


Figure 10 – Types of controls housed in the Control Style



## Accessing the ControlStyle Page

The ControlStyle page is accessed in your Style .OMD file using Page Architect. Refer to the figure below for development functions.

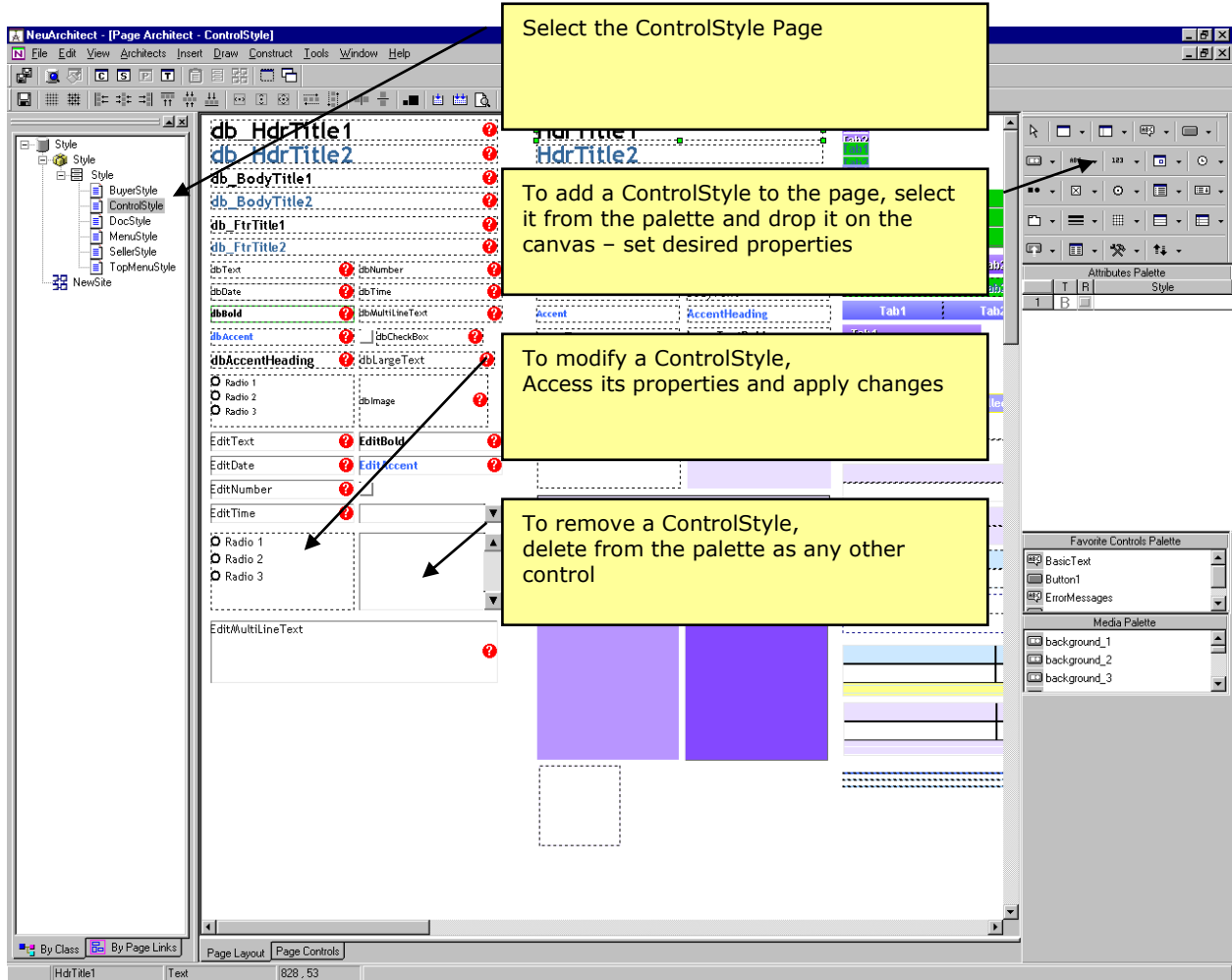


Figure 11 – the ControlStyle page created in NeuArchitect

## Create the ControlStyle Page in PageArchitect

Control styles describe the visual framework for commonly used controls. Control styles also set control specific characteristics such as size, shape, border, and fill styles. A number of control styles are generally specified for each type of control.

<b>Function</b>	<b>Purpose</b>	<b>Steps</b>
<b>Create the ControlStyle Page in PageArchitect</b>		
Build the visual framework for common controls	A number of control styles are generally specified for each type of control. For example, some of the control styles for a label control may be: BasicText, BodyText, HdrTitle1, etc. These styles are accessed via dropdowns when you select a control.	<ol style="list-style-type: none"> <li>1. With the Class name highlighted, click the <b>Pages</b> tab.</li> <li>2. Click <b>New</b></li> <li>3. Enter the Page name as <b>ControlStyle</b> under the Page list.</li> <li>4. Click the <b>Page Architect</b> button.</li> <li>5. Use the Control palette by clicking the desired control and dropping control to the page canvas.</li> <li>6. Access the <b>Control properties</b> window of the newly added control to: <ul style="list-style-type: none"> <li>• Define a logical control name or accept the established control name that is automatically generated.</li> <li>• Use the primary theme logical names to set the desired attributes for the control <ul style="list-style-type: none"> <li>o Color</li> <li>o Image</li> <li>o Font</li> </ul> </li> </ul> </li> </ol>



True or False? Control styles set control specific characteristics such as size, shape, border, and fill styles.

## Adding Controls to the new ControlStyle Page

Use the following table as a guideline for building and maintaining the ControlStyle page in Page Architect.

<b>Process for adding controls to the ControlStyle page</b>	
<b>1</b>	<b>Drop a new control on the Page Architect canvas</b>
<b>2</b>	Access the <b>Control properties</b> window of the control
<b>3</b>	From the <b>Control tab</b> enter the name of the control in the <b>Name</b> field (i.e. Header1)
<b>4</b>	<p>Click the <b>Display</b> tab to define custom control characteristics (i.e. Fill colors, Border colors, Font face and size, etc.) or apply established Theme characteristics as follows:</p> <p><b>Style:</b> select a previously created control style name from the drop-down list that you want your new control to import the characteristics (i.e. Fill colors, Gradient type, Border, etc.) from. The characteristics for the Style drop-down options are set in the Theme Architect\Color Themes and enabled for the Current Theme.</p> <p><b>Text display:</b> select either the <b>Custom</b> or <b>Theme</b> radio button to define the text characteristics. Selecting the <b>Custom</b> radio button indicates you want to manually establish the font characteristics for the control in the Display tab. Selecting the <b>Theme</b> radio button indicates you want to select a previously defined control style name from drop-down list and import the font characteristics (i.e. font face, font size, etc). The characteristics for Theme drop-down options are set in the Theme Architect\Font Themes and enabled for the Current Theme</p>
<b>5</b>	Click <b>OK</b> to capture defined settings
<b>6</b>	Repeat the above steps to define all the controls you will apply throughout your application.
<p>When you add controls using the Control palette types (i.e. Label), the newly defined controls will be added to the drop-down list options for the corresponding Control Category. You may use the Custom Control available at the bottom of the palette for those controls that may not apply to the pre-set categories.</p>	



## Theme Architect – Adding a ControlStyle page to the Style Repository

1. With the GMStyle application open go to the pages tab of the Style Class.
2. Click **New**
3. Enter the Page name as **ControlStyle** under the Page list.
4. Click the **Page Architect** button.
5. Right click in the canvas and set the Fill to **Solid** and **Background**
6. Using the Control palette drop the following controls into the new container and assign the control attributes specified in the table below. Keep in mind as you are setting up these controls that you are using parameters specified in Theme Architect.
7. Open Theme Architect and change the Color Ttheme. Click the canvas to refresh. Note differences.
8. If time permits, feel free to add additional controls like menus and text boxes.

Composite Theme Name	Control Tab	Display Tab
<b>SubmitButton_1</b>	Width 92 Height 22	<b>Fill Properties</b> Fill: Type <b>Solid</b> Color <b>Fill-Light</b> Border: Type <b>Raised</b> Color <b>Shade</b> Shine color <b>Shine</b> Depth <b>3</b> Under Text display use the drop sown and select <b>Text</b> Specify Alignment <b>Center</b>
<b>SubmitButton_2</b>	Width 92 Height 22	<b>Fill Properties</b> Fill: Type <b>Solid</b> Color <b>Fill-Dark</b> Border: Type <b>Raised</b> Color <b>Shade</b> Shine color <b>Shine</b> Depth <b>3</b> Under Text display use the drop sown and select <b>Text</b> Specify Alignment <b>Center</b>
<b>ImageButton_1</b>	Width 66 Height 18	<b>Fill Properties</b> Fill: Type <b>None</b> Border: Type <b>None</b> Select <b>Back-Button</b> from the drop down for Back mage

**Be sure to save your work!**

## Page Styles



**This is the third modeling component of the Style Repository bringing together the look and feel of the Themes and ControlStyles on reusable Page Style templates.**

The last design element is Page Styles. Once the desired controls are defined in the Control Style page and the themes defined in Theme Architect, you can create style template pages. The use of Page Styles will assist in keeping consistency within your web application.

### Page Style Concepts

Page styles are pages that have been defined with pre-placed controls, similar to templates. When a new page is created and uses this Page style, all pre-placed controls are inherited to the new page. You can create and name as many Page Style templates you may need. For example: you may be creating pages with framesets and custom menu tabs. The best way to maintain a consistent look among the pages is by creating templates that have the pre-defined frameset and menu tab controls positioned on the page. The figure below illustrates a page template created as a Page Style.

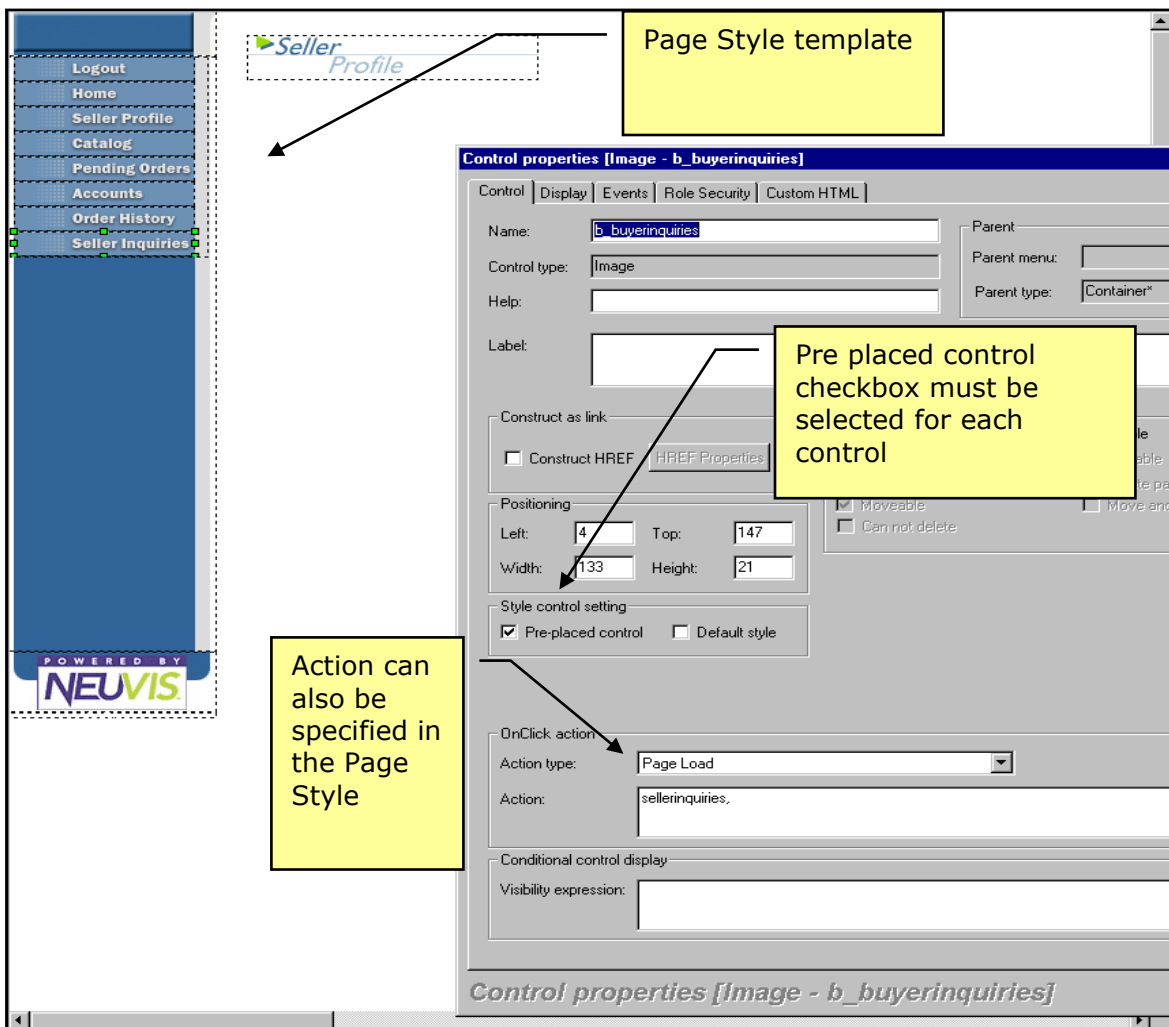


Figure 12 – a page template created as a Page Style

## Create the Style Pages in PageArchitect

The following guidelines are applicable for all Page Styles you will create.

<b>Function</b>	<b>Purpose</b>	<b>Steps</b>
<b>Design the PageStyle templates in PageArchitect</b>		
PageStyle templates for your application	Page styles provide the ability to set the default page size, page background, similar design and functionality on a reusable template	<ol style="list-style-type: none"> <li>1. From the Application Navigator window, highlight the <b>Style</b> class and click the <b>Pages</b> tab.</li> <li>2. Add the new page or pages to the <b>Page list</b> in the Style repository and name them according to function.</li> <li>3. Click the <b>Page Architect</b> button</li> <li>4. Place controls on the page for                             <ul style="list-style-type: none"> <li>• Navigation</li> <li>• Images</li> <li>• Text.</li> </ul> </li> <li>5. Double click the control added to access <b>Control properties</b> define the control characteristics (i.e. <b>Label</b> text).</li> <li>6. From the Control properties\Control tab, locate the <b>Style control setting</b> and click the <b>Pre-placed control</b> checkbox.</li> <li>7. Optionally specify an action to associate with the control (Pageload, etc.)</li> </ol>
Important - In order for the controls placed on template pages to appear when a Page Style is enabled, you must have a check in the <b>Pre-placed control</b> check box located in the Control properties\Control tab		



True or False? You are allowed only one Page Style in a Style Repository.



## Theme Architect – Adding Page Styles to the Style Repository

1. With the GMStyle application open go to the pages tab of the Style Class.
2. Click **New**
3. Enter the Page name as **MainPage** under the Page list.
4. Click the **Page Architect** button.
5. Right click in the canvas and set the Fill to **Solid** and **Background**
6. Drop and stretch an Image Button on the canvas.
7. From the Display tab of the Image Button properties, Select **Main-Image** from the drop down for Back mage
8. Click the **Stretch** radio button
9. Click **OK**.
10. From the Control properties\Control tab, locate the **Style control setting** and click the **Pre-placed control** checkbox.
11. Open Theme Architect and change the Color Theme. Click the canvas to refresh. Note differences.

**Be sure to save your work!**

*NeuArchitect Style Modeling Tools – Workshop 7*

## NeuArchitect Style Modeling Components – Reference

### Creating a New Style Repository for Your NeuArchitect Application

**When saving a NeuArchitect application for the first time, the default style folder and all its accompanying files and folders are copied into the file structure of the new application.**

**You can create a new Style Repository as opposed to using or modifying the supplied Style Repository OMD file.**

**Creating a new Style Repository is similar to creating a new application but with fewer steps:**

- 1. Create the new application file**
- 2. Setup the NeuArchitect application infrastructure**
- 3. Use Theme Architect to:**
  - Define attributes and files for the primary themes (colors, fonts and images)**
  - Map out the composite themes**
- 4. Create the ControlStyle Page in PageArchitect**
- 5. Design the PageStyle templates in PageArchitect**



## Steps for Creating a New Style Repository

### First - Create the New Application File

Function	Purpose	Steps
Creating a New Style Repository – Create the application file		
Create the directory	A style repository, like any NeuArchitect must reside as an OMD in its own directory	<ol style="list-style-type: none"> <li>4. Create the application directory (must be the same name as you paln for your OMD file)</li> <li>5. Start NeuArchitect</li> <li>6. Choose Create new application</li> </ol>

### Second - Setup the NeuArchitect Application Infrastructure

Function	Purpose	Steps
Creating a New Style Repository – Setup the NeuArchitect application infrastructure		
Setup the Style Repository application infrastructure	As with any NeuArchitect application, the infrastructure must be setup. In the case of the Style Repository, there are guidelines that must be adhered to specifically (refer to the steps in this table)	<ol style="list-style-type: none"> <li>6. Define the Application name.</li> <li>7. Click the Remove Style button located under the Application tab Import style application section.                             <p style="margin-left: 40px;">Note: the NeuArchitect supplied SampleStyle will default as the Current style. To avoid confusion when building a new Style Repository always remove any Import style application before saving the new style application.</p> </li> <li>8. Define the Package name as Style.</li> <li>9. Define the Component name as Style.</li> <li>10. Insert a Class and name it Style</li> </ol>

## Theme Architect

**Theme Architect, a Style Repository component, enables you to create and store visual elements that can be applied to the global web application such as:**

- **Colors**
- **Graphic images**
- **Font styles**

**The Theme Architect window is made up of four tabs: Composite Themes, and the Primary Themes listed below:**

- **Color Themes**
- **Image Themes**
- **Font Themes.**

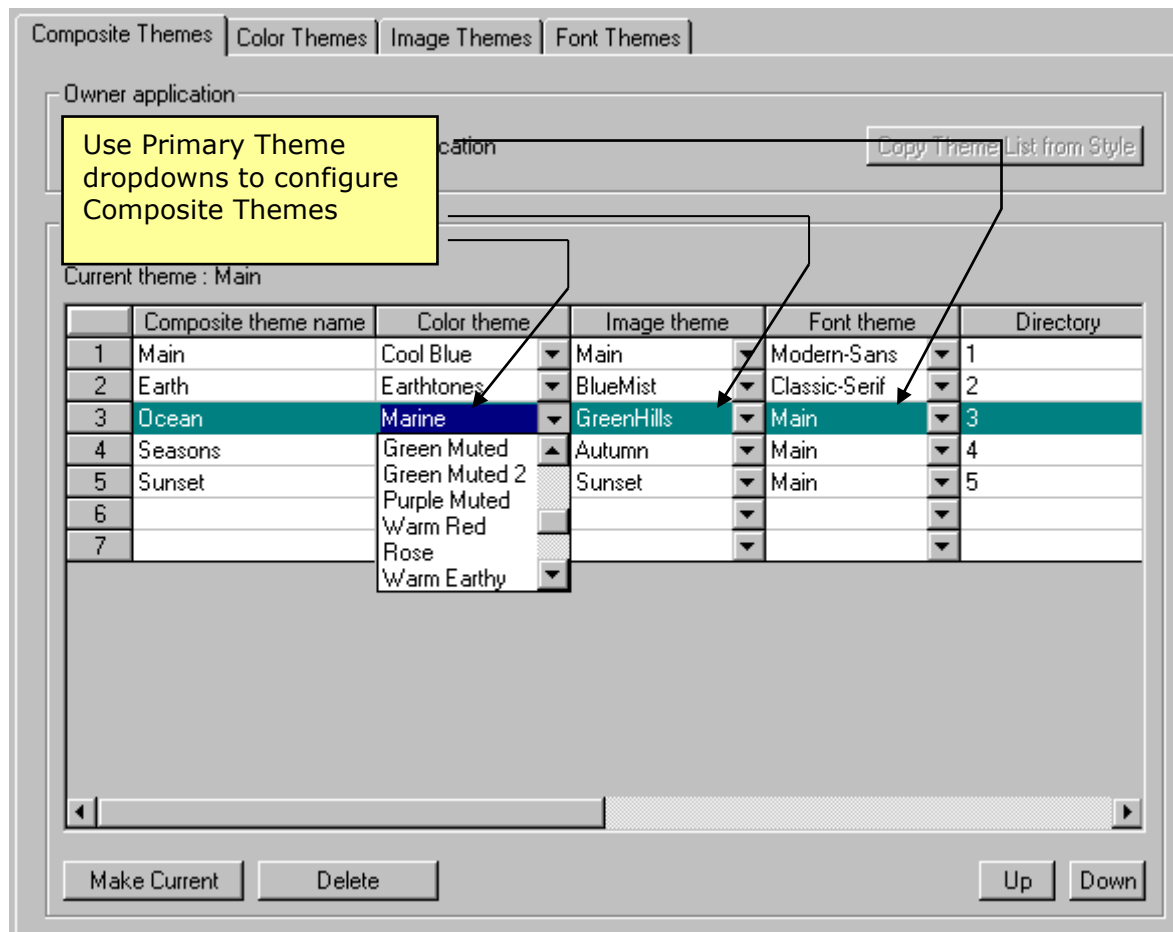


Figure 4 – Primary and Composite Themes in Theme Architect

## Color Themes

Color themes are logical names specified for a given color (RGB) combination.

- The Color Themes tab will enable you to define the colors you want to apply to as page: backgrounds
- control fills
- borders
- shadows.

You may create one or more color themes.

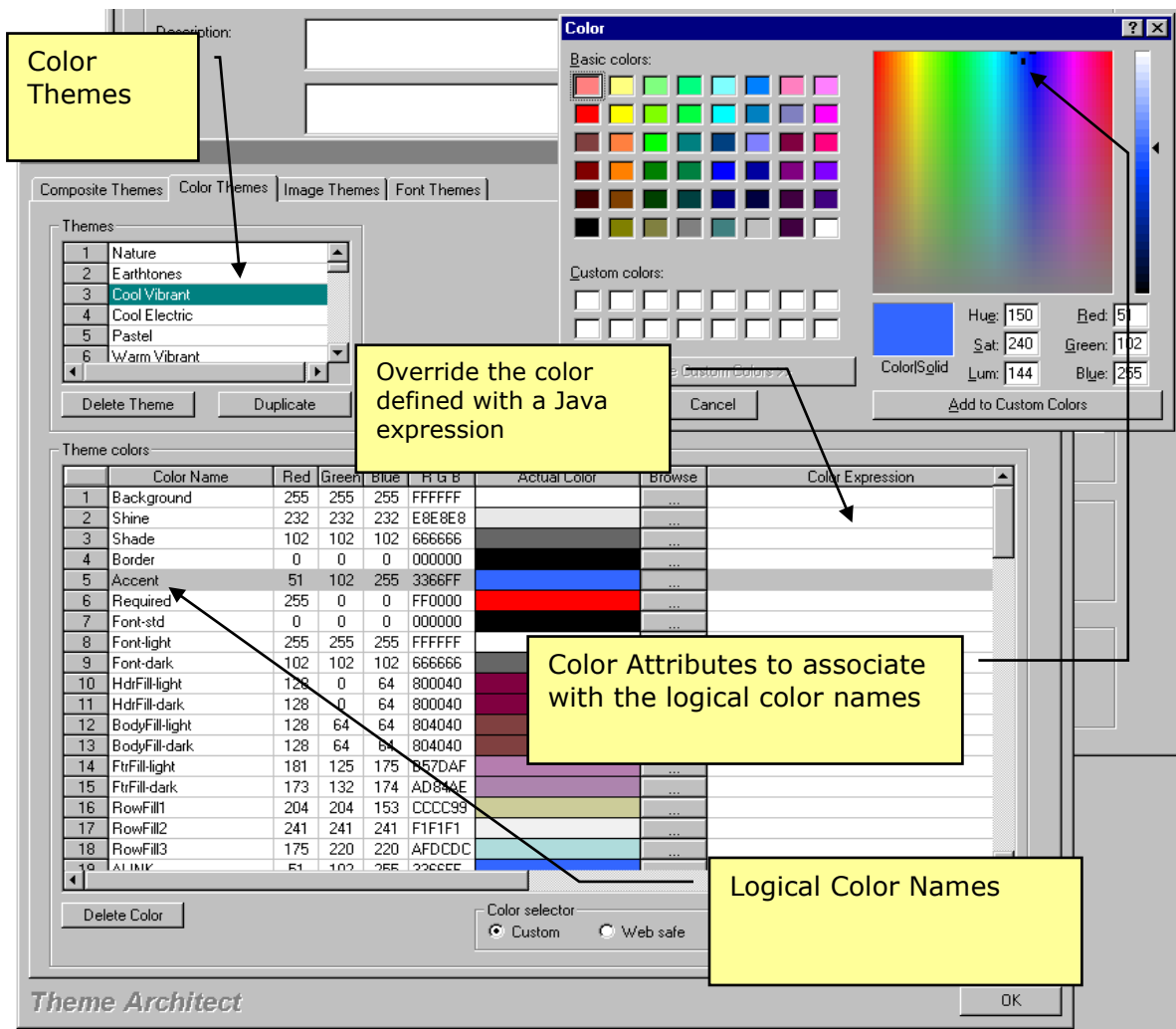



Figure 5 – Theme Architect – Color Themes

## Steps for Creating Color Themes

**Use the following table as a guideline for building and maintaining color themes in Theme Architect**

### Maintaining Color Themes in Theme Architect

Add a Color Theme - The name entered for the Color Name will be the same across all Color Themes. You may select different color values for each theme. The color palette that appears when the browse button  is applied is dependant on the Color selector that is enabled Custom vs. web safe.

- 1 Click on an empty line under Themes
- 2 Enter the Color Theme Name
- 3 From the Theme colors section enter the Color Name (i.e. Background, Border, Shade, Fill). For gradient and pattern fills you must indicate two colors, e.g. Background and Background1 or Fill2-dark and Fill2-light.
- 4 Establish the color by entering values for one of the following:
  - Red, Green, Blue RGB values.
  - RGB alphanumeric value.Click Browse and select the actual color from the color palette.

#### Delete a Color Theme

- 1 Click to highlight the Color Theme Name under the Themes section
- 2 Click the Delete Theme button

Duplicate a Color Theme – Important Note: Deleting a Color Name from the Theme colors section of your duplicate theme will delete that name for all Color Themes within the application. Adding a Color Name to the Color Themes section will apply to all Color Themes established and/or added within the application. The color value default will equal black unless otherwise specified for each Color Theme

- 1 Click to highlight the Color Theme Name under the Themes section
- 2 Click the Duplicate button. The duplicated Color Theme retains the original name of the theme and a numeric value is appended at the end to keep it unique. You can change the name to something more relevant to your application
- 3 Modify the duplicated Color Theme as needed

## Image Themes

**Image themes are logical names specified for a given image files (.gif, .jpg).**

**The Image Themes tab will enable you to change and/or update images used in your application in a batch process.**

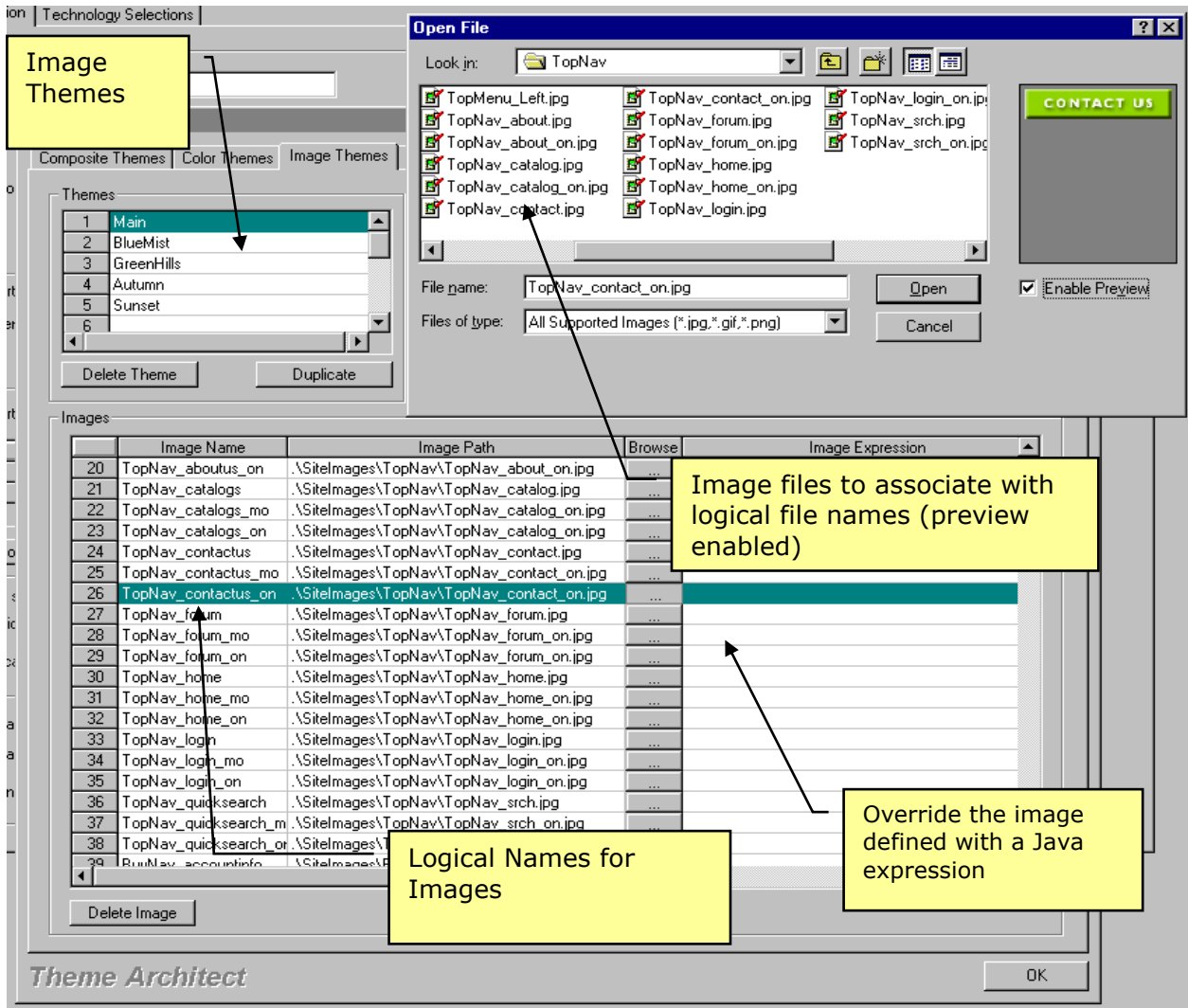


Figure 6 – Theme Architect Font Themes

## Steps for Creating Image Themes

**Use the following table as a guideline for building and maintaining Image themes in Theme Architect**

### **Maintaining Color Themes in Theme Architect**

Add an Image Theme - The name entered for the Image Name will be the same across all Image Themes, you will use the Image Path to distinguish where to access new images for each Image theme

- 1 Click on an empty line under Themes
- 2 Enter the Image Theme Name
- 3 From the Images section enter the Image Name
- 4 Enter the Image Path or use Browse to locate the correct path where the image file resides.

Delete an Image Theme

- 1 Click to highlight the Image Theme Name under the Themes section
- 2 Click the Delete Theme button

Duplicate an Image Theme – Important Note: Deleting an image from the Theme image section of your duplicate theme will delete that name for all Image Themes within the application. Adding an Image Name to the Image Themes section will apply to all Image Themes established and/or added within the application.

- 1 Click to highlight the Image Theme Name under the Themes section.
- 2 Click the Duplicate button. The duplicated Image Theme will retain the original name of the theme and append a numeric value at the end to keep the Image Theme Name unique. You may change the name to something more relevant to your application
- 3 Modify the duplicated Image Theme as needed

## Font Themes

Font themes are logical names specified for a given font characteristics.

The Font Themes tab will enable you to define the font styles (i.e. Arial, Verdana) and characteristics:

- Font style
- Font size
- Bold, light, etc.

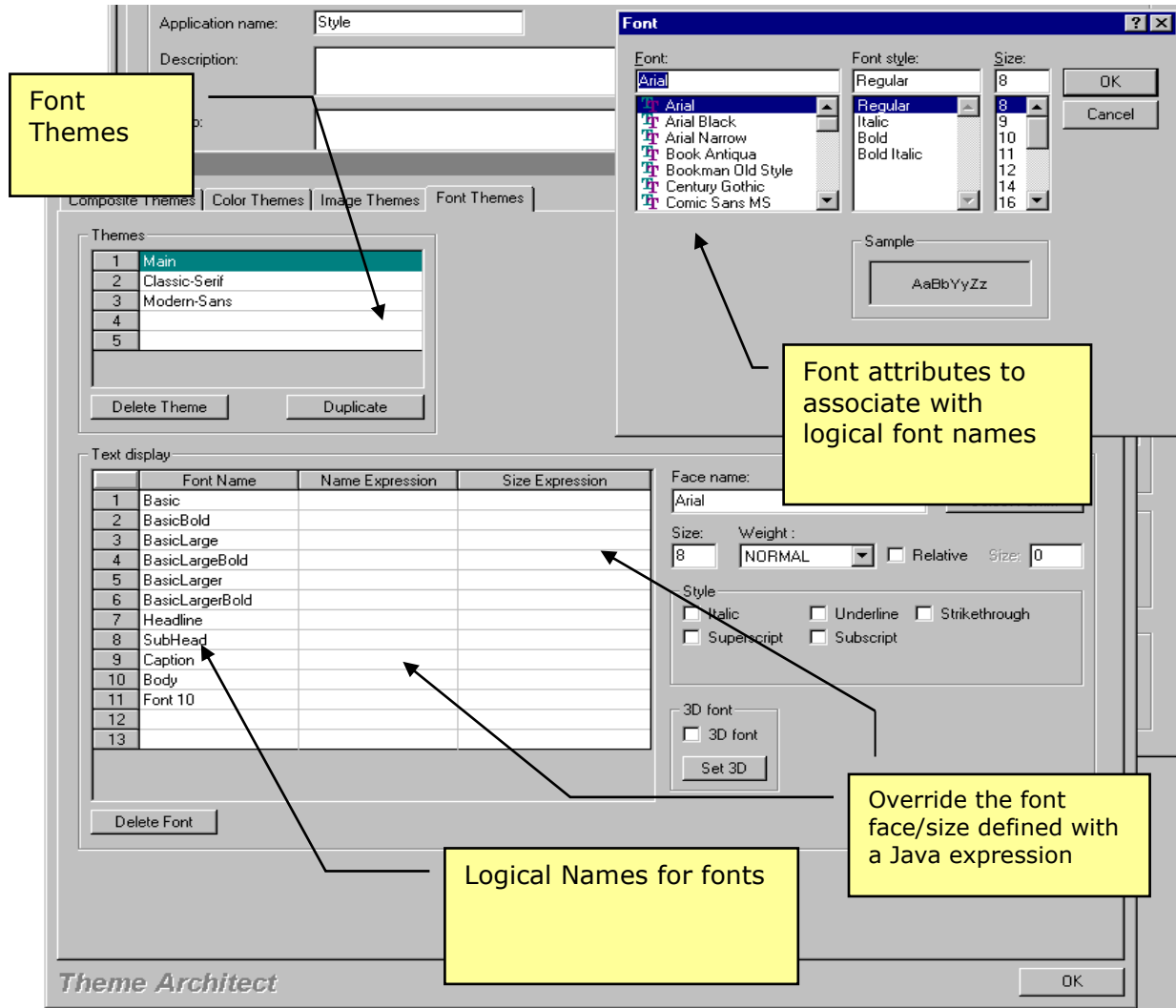


Figure 7 – Theme Architect Font Themes

## Steps for Creating Font Themes

**Use the following table as a guideline for building and maintaining Image themes in Theme Architect**

### **Maintaining Color Themes in Theme Architect**

Add a Font Theme - The name entered for the Font Name will be the same across all Font Themes. You may select different Font values for each Font theme

- 1 Click on an empty line under Themes
- 2 Enter the Font Theme Name
- 3 From the Text display section enter the Font Name
- 4 Apply the appropriate Font styles and characteristics using the options to the right of the Text display (i.e. Font face, Size, Style, etc.).

Delete a Font Theme

- 1 Click to highlight the Font Theme Name under the Themes section
- 2 Click the Delete Theme button.

Duplicate a Font Theme – Deleting a Font Name from the Text display section will delete that name for all Font Themes within the application. Adding a Font Name to the Text display section will apply to all Font Themes established and/or added within the application. The font value default will equal Arial size 10 regular unless otherwise specified for each Font Theme

- 1 Click to highlight the Font Theme Name under the Themes section.
- 2 Click the Duplicate button. The duplicated Font Theme will retain the original name of the theme and append a numeric value at the end to keep it unique. You may change the name to something more relevant to your application
- 3 Modify the duplicated Font Theme as needed



## Composite Themes

You may create one or several Composite Themes as dictated by your business needs. A Composite Theme is a combination of a:

- Color Theme
- Image Theme
- Font Theme.

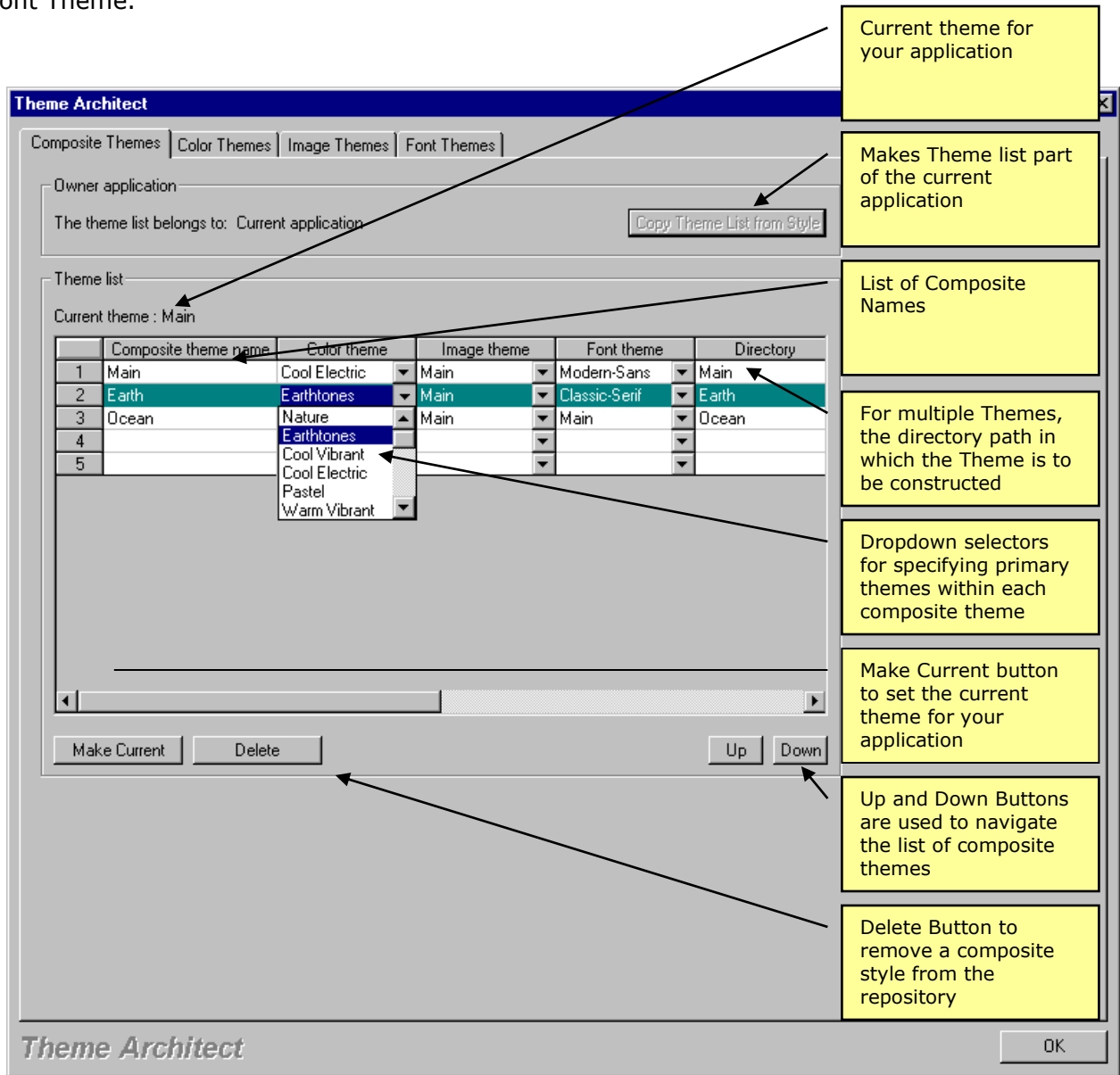


Figure 8 – Setting Composite Theme Parameters for Your Application

## Steps for Creating Composite Themes

Use the following table as a guideline for building and maintaining Composite themes in Theme Architect

Function	Purpose	Steps
Managing Composite Themes in Theme Architect		
Remove a composite theme	Discard a Theme from your application or from the Style OMD file	<ol style="list-style-type: none"> <li>5. From the NeuArchitect top menu, select Architects</li> <li>6. Select Themes</li> <li>7. Select the theme from the Theme list</li> <li>8. Click Delete Button</li> </ol>
Create a new composite theme	Give your application a second, third, etc. look	<ol style="list-style-type: none"> <li>5. From the NeuArchitect top menu, select Architects</li> <li>6. Select Themes</li> <li>7. Click in the first empty row in the Theme list</li> <li>8. Supply the composite theme name and use dropdowns to select primary themes</li> </ol>
Modify a composite theme	Change the primary color, font or image theme for the current composite theme	<ol style="list-style-type: none"> <li>5. From the NeuArchitect top menu, select Architects</li> <li>6. Select Themes</li> <li>7. Select the theme to be modified</li> <li>8. Use dropdowns to select primary themes</li> </ol>
Specify directory path for the composite theme	Supplies the directory path in which the composite theme is to be constructed. This should be used only when the application is being constructed simultaneously for multiple themes.	<ol style="list-style-type: none"> <li>6. From the NeuArchitect top menu, select Architects</li> <li>7. Select Themes</li> <li>8. Select the theme to be constructed simultaneously</li> <li>9. Supply the directory name – should be a meaningful name which will be constructed as a subdirectory of your application ASP directory</li> <li>10. Follow Page Level instructions in Managing Themes at the Page Level in Page Architect</li> </ol>

## Control Style

**Control Styles are pages that specify the styles of commonly used controls to build the pages in your web application.**

**Controls created and added to the Control Style page will be available from the Control palette for designing your web pages.**

**Controls on a page are the visual elements of the application. A few of the functions that controls serve are to**

- **Present information**
- **Format data**
- **Imply action to take.**

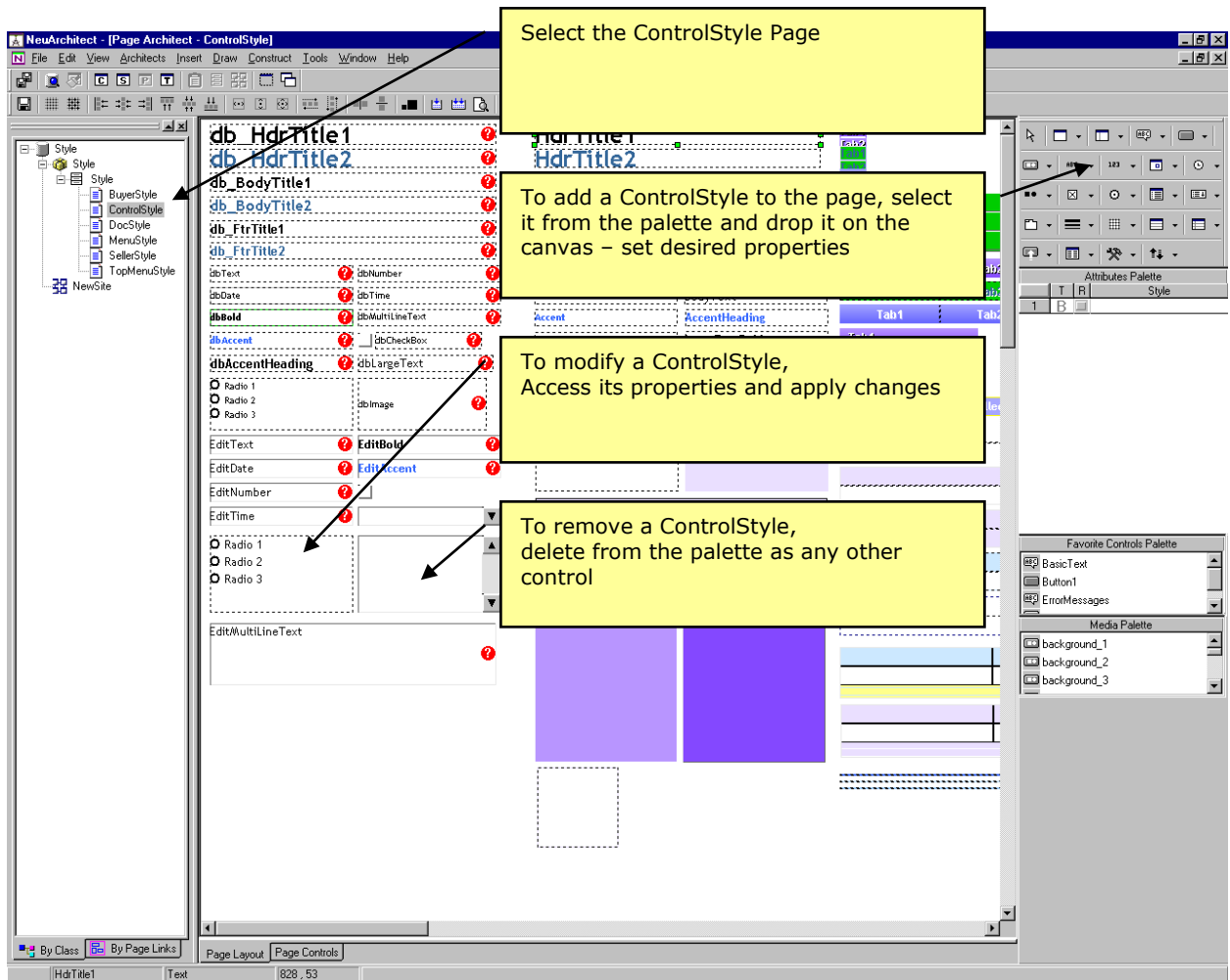


Figure 11 – the ControlStyle page created in NeuArchitect

## Steps for Creating the ControlStyle

**Control styles describe the visual framework for commonly used controls. A number of control styles are generally specified for each type of control.**

**Control styles also set control specific characteristics such as:**

- **Size**
- **Shape**
- **Border**
- **Fill styles.**

<b>Function</b>	<b>Purpose</b>	<b>Steps</b>
<b>Create the ControlStyle Page in PageArchitect</b>		
Build the visual framework for common controls	A number of control styles are generally specified for each type of control. For example, some of the control styles for a label control may be: BasicText, BodyText, HdrTitle1, etc. These styles are accessed via dropdowns when you select a control.	<ol style="list-style-type: none"> <li>7. With the Class name highlighted, click the <b>Pages</b> tab.</li> <li>8. Click <b>New</b></li> <li>9. Enter the Page name as <b>ControlStyle</b> under the Page list.</li> <li>10. Click the <b>Page Architect</b> button.</li> <li>11. Use the Control palette by clicking the desired control and dropping control to the page canvas.</li> <li>12. Access the <b>Control properties</b> window of the newly added control to: <ul style="list-style-type: none"> <li>• Define a logical control name or accept the established control name that is automatically generated.</li> <li>• Use the primary theme logical names to set the desired attributes for the control <ul style="list-style-type: none"> <li>o Color</li> <li>o Image</li> <li>o Font</li> </ul> </li> </ul> </li> </ol>

## Steps for Adding Controls to the new ControlStyle Page

**Use the following table as a guideline for building and maintaining the ControlStyle page in Page Architect.**

<b>Process for adding controls to the ControlStyle page</b>	
1	Drop a new control on the Page Architect canvas
2	Access the Control properties window of the control
3	From the Control tab enter the name of the control in the Name field (i.e. Header1)
4	<p>Click the Display tab to define custom control characteristics (i.e. Fill colors, Border colors, Font face and size, etc.) or apply established Theme characteristics as follows:</p> <p>Style: select a previously created control style name from the drop-down list that you want your new control to import the characteristics (i.e. Fill colors, Gradient type, Border, etc.) from. The characteristics for the Style drop-down options are set in the Theme Architect\Color Themes and enabled for the Current Theme.</p> <p>Text display: select either the Custom or Theme radio button to define the text characteristics. Selecting the Custom radio button indicates you want to manually establish the font characteristics for the control in the Display tab. Selecting the Theme radio button indicates you want to select a previously defined control style name from drop-down list and import the font characteristics (i.e. font face, font size, etc). The characteristics for Theme drop-down options are set in the Theme Architect\Font Themes and enabled for the Current Theme</p>
5	Click OK to capture defined settings
6	<p>Repeat the above steps to define all the controls you will apply throughout your application.</p> <p>When you add controls using the Control palette types (i.e. Label), the newly defined controls will be added to the drop-down list options for the corresponding Control Category. You may use the Custom Control available at the bottom of the palette for those controls that may not apply to the pre-set categories.</p>

## Page Styles

Page styles are pages that have been defined with pre-placed controls, similar to templates.

When a new page is created and uses this Page style, all pre-placed controls are inherited to the new page.

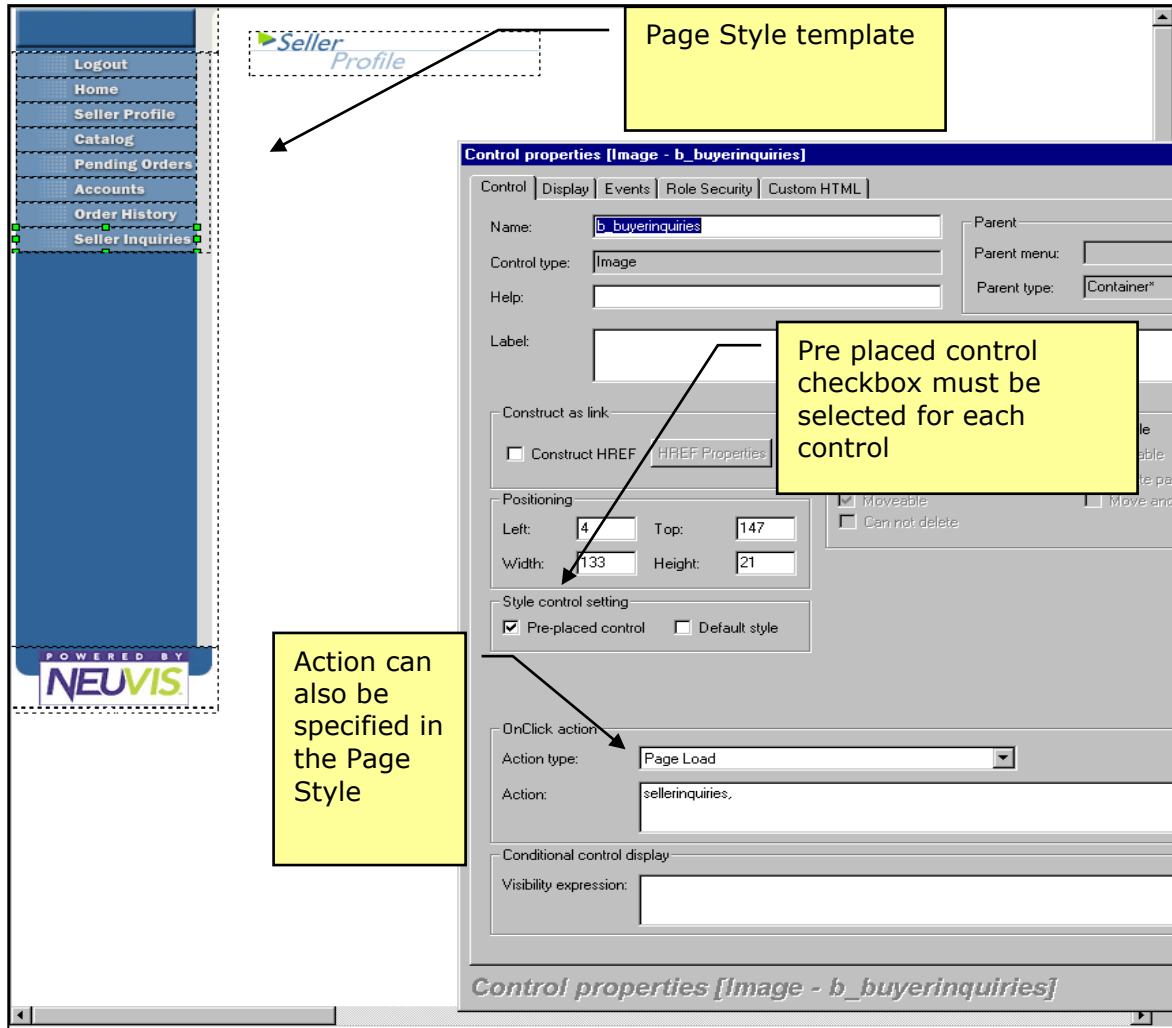


Figure 12 – a page template created as a Page Style

## Steps for Creating Page Styles

**The following guidelines are applicable for all Page Styles you will create.**

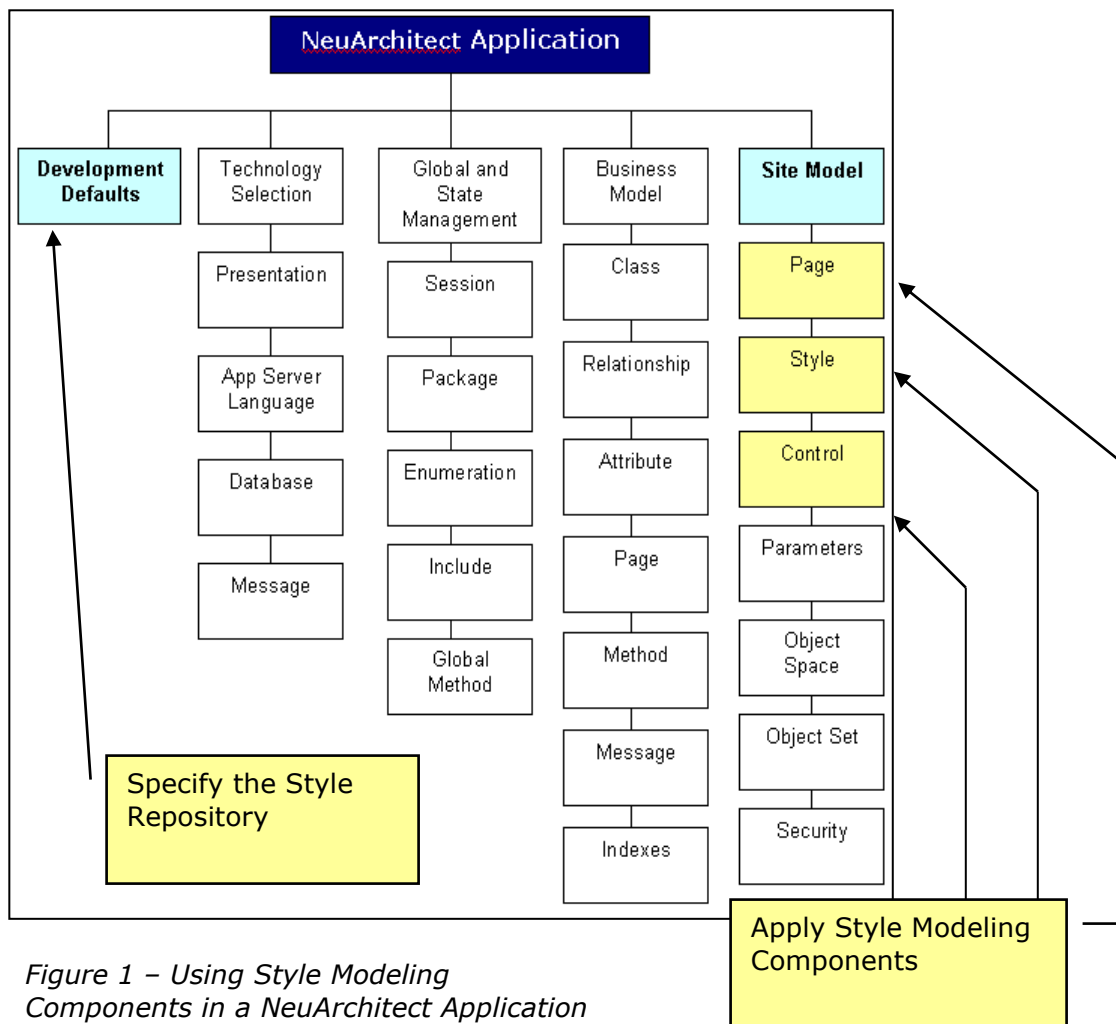
Function	Purpose	Steps
Design the PageStyle templates in PageArchitect		
PageStyle templates for your application	Page styles provide the ability to set the default page size, page background, similar design and functionality on a reusable template	<ol style="list-style-type: none"> <li>8. From the Application Navigator window, highlight the Style class and click the Pages tab.</li> <li>9. Add the new page or pages to the Page list in the Style repository and name them according to function.</li> <li>10. Click the Page Architect button</li> <li>11. Place controls on the page for                             <ul style="list-style-type: none"> <li>• Navigation</li> <li>• Images</li> <li>• Text.</li> </ul> </li> <li>12. Double click the control added to access Control properties define the control characteristics (i.e. Label text).</li> <li>13. From the Control properties\Control tab, locate the Style control setting and click the Pre-placed control checkbox.</li> <li>14. Optionally specify an action to associate with the control (Pageload, etc.)</li> </ol>
<p>Important - In order for the controls placed on template pages to appear when a Page Style is enabled, you must have a check in the Pre-placed control check box located in the Control properties\Control tab</p>		

# Section 3 - Using the Style Repository in a NeuArchitect Application



**What follows is a brief discussion of how to select the correct Style Repository for an application. We'll then get into some additional details of working with the Style Repository later in this section.**

In the previous section of this course, you created a Style Repository. You used Theme Architect to design a combination of color, image and font themes, built a Control Style page and created a Page Style. In this section you will learn how these are applied in a NeuArchitect application.



*Figure 1 - Using Style Modeling Components in a NeuArchitect Application*



## Specifying the Style Repository

When saving a new NeuArchitect application for the first time, the default style folder and all its accompanying files and folders are copied into the file structure of the new application. This operation is reflected on the application tab. Before saving an application, the **Current style path:** points to your NeuArchitect install directory. After saving the application, the **Current style path:** is changed to your application directory where the SampleStyle files have been copied. The SampleStyle repository contains a control style, themes, and page styles that can be used in your application.

Whether you use the SampleStyle without change or create your own style repository will clearly be a business decision dependent on the types and application styles you are developing. One thing to keep in mind is that style repositories can be used and re-used by many applications.

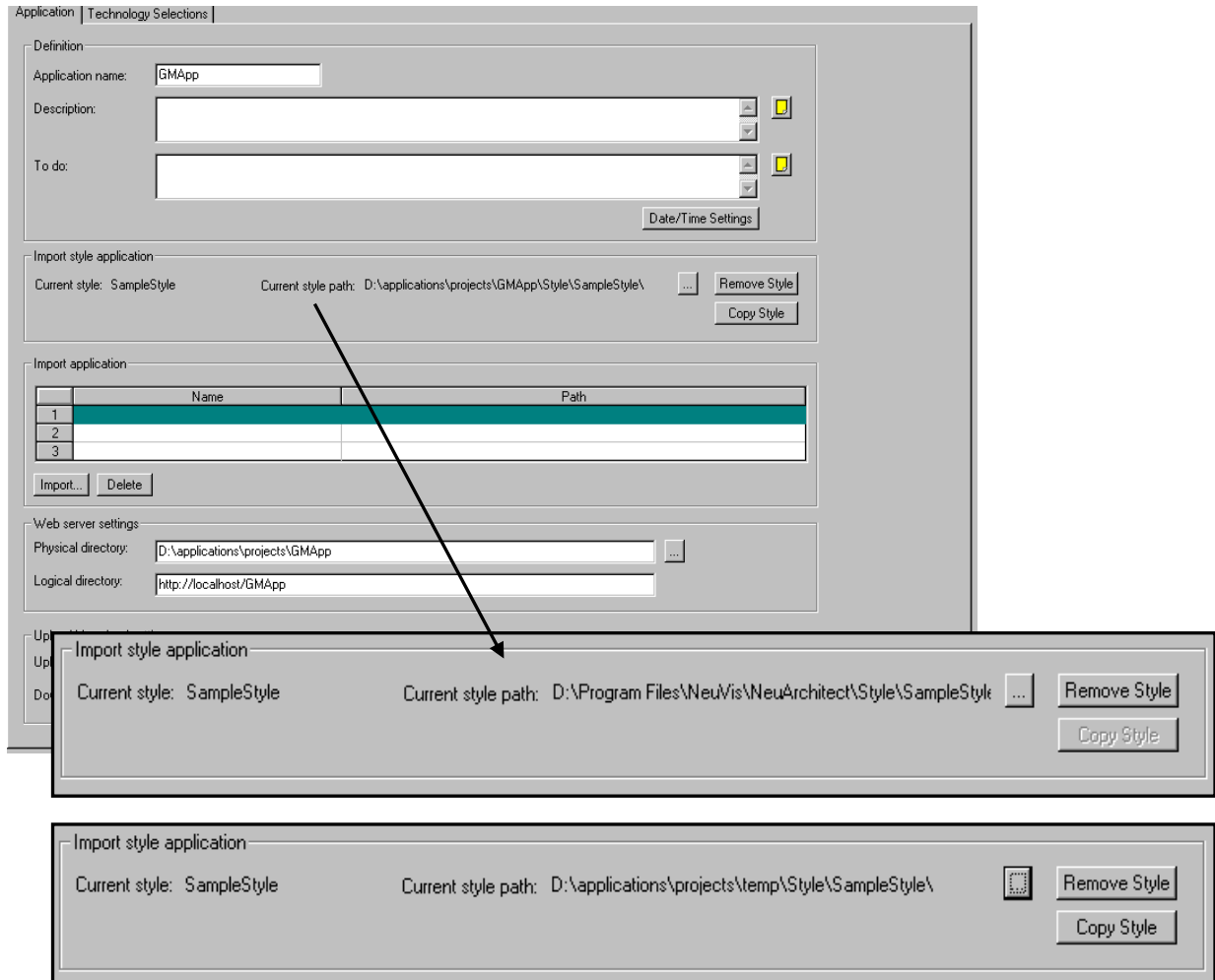
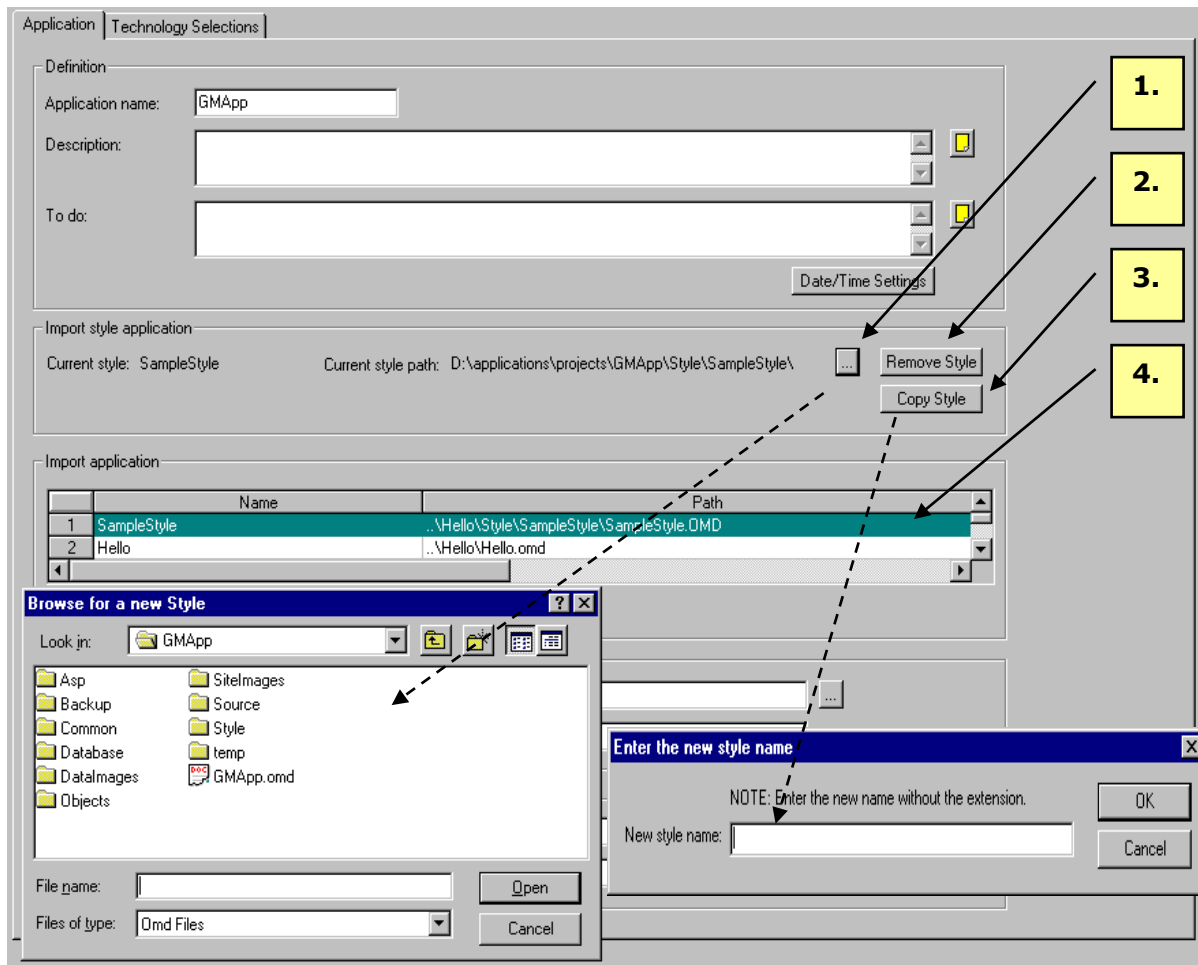


Figure 2 – The Current style path before and after saving an application

The figure and table shown below refer to the Application tab showing you where to (and where not to) import, remove, or copy a style repository for an application within NeuArchitect. Items in the table refer to corresponding numbers in the figure.



<i>Item</i>	<i>Option</i>	<i>Description</i>
<b>1</b>	Browse Button	Opens Browse window to select a different style to apply in your application
<b>2</b>	Remove Style Button	Completely clears the Style from your application
<b>3</b>	Copy Style Button	Prompts for a new Style name Makes an exact copy of everything in the current SampleStyle directory Assigns the supplied name to the new OMD file and to the new subdirectory under the Style subdirectory Sets the Current style path to the copied Style
<b>4</b>	Import application	This should <b>not</b> be used to try to import Styles! Styles are only referenced there if you are importing another application. While this style will display in the Page Architect Navigator window, the current application will not reference it!

Figure 3 – Selecting the Current Style for your Application



You can verify your answers to the true/false questions below by moving your mouse pointer over the underlined area after before question (*online versions only*).


1.        When creating a NeuArchitect application, a Style is automatically added to your application.
2.        The three main components in a Style are a ControlStyle, Themes and ImageStyles.
3.        Once you have saved your application, it is necessary to make a copy of the Style using the Control tab.
4.        Business decisions usually guide changes to the default style.
5.        The SampleStyle.OMD file specified on the Control tab is located in the SiteImages subdirectory.

*Optional Review*

## Using the Style Repository in Your NeuArchitect Application - Summary

Whether you use the SampleStyle without change or create your own style repository will clearly be a business decision dependent on the types and application styles you are developing. The following table summarizes to create a new Style Repository.

The following table summarizes the functions of managing a Style Repository for a NeuArchitect application.

<b>Function</b>	<b>Purpose</b>	<b>Steps</b>
<b>Managing Style Repositories - Application Level</b>		
Copy the current Style Repository	Use the Copy Style option to: <ul style="list-style-type: none"> <li>• Make an exact copy of everything in the current SampleStyle directory</li> <li>• Assign the supplied name to the new OMD file and to the new subdirectory under the Style subdirectory</li> <li>• Set the Current style path to the copied Style</li> </ul> Note: This option cannot be used to copy a Style Repository from a different application	<ol style="list-style-type: none"> <li>1. Select the NeuArchitect Application tab</li> <li>2. Click the Copy Style button under Import style application</li> <li>3. Answer Yes to the remove style prompt</li> <li>4. Supply the new style name (do not supply a file extension)</li> <li>5. Click OK</li> </ol>
Remove a Style Repository	Totally remove the Style Repository and its graphical parameters from your application	<ol style="list-style-type: none"> <li>1. Select the NeuArchitect Application tab</li> <li>2. Click the Remove Style button under Import style application</li> </ol>
Import a different Style Repository	Copy a Style Repository from another application and make it the current Style Repository for your application	<ol style="list-style-type: none"> <li>1. Select the NeuArchitect Application tab</li> <li>2. Click the browse button  under Import style application</li> <li>3. Locate and select the Style OMD file that you wish to import</li> </ol>

## Applying Themes in Page Design



**Here, we will look at the 'Dynamics' of Theme Architect in NeuArchitect Application Development.**

When a page from your application is constructed, Page Architect will either:

- Apply default characteristic from the current composite style set in the SampleStyle
- Override the current style, applying the composite style selected for that page.

The following table indicated the steps required to change the composite theme for the entire application.

Function	Purpose	Steps
<b>Changing the Composite Theme for an entire application</b>		
Change the current composite theme for your entire application	Reconstruct under a different composite theme (Winter, Spring, etc.)	<ol style="list-style-type: none"> <li>1. From the NeuArchitect top menu, select Architects</li> <li>2. Select Themes</li> <li>3. Select new theme from the Theme list</li> <li>4. Click Make Current Button</li> </ol>

The composite style is set and displayed in Page Architect on the Page Properties Page tab. When first creating a page in Page Architect, the Style is set to current. The current theme applies default characteristic from the current composite style set in the Theme Architect. Using the dropdown list, you can change the style for that page. Keep in mind, however, that you are overriding the default to the current theme. If you go back to theme Architect and change the current composite style (for example, Main to Ocean), the pages with the overrides will not change. In the example shown below, selecting the Earth theme for the page will always override the current theme. Note that the dropdown options allow you to switch back to the Current theme of the application.

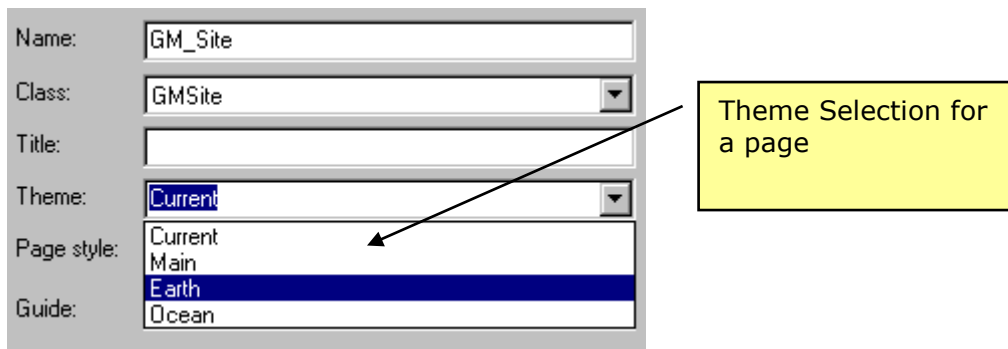


Figure 4 – Specifying a style to be applied to a specific page

## Managing Themes at the Page Level in Page Architect

Page Architect provides several options to enhance the development process with access to the themes developed in Theme Architect. It should be noted that all options summarized in the table below apply on a page-by-page basis. **None of the theme functionality that you set in Page Architect is applied globally across all pages.**

### Theme Specification at the Page Level

This option is used to veer from the current theme selected for your application. Consider the situation where an application is constructed regularly for changing themes, but, the design of the site requires that certain pages should retain their own style characteristics when constructed. In this case you would set the **Theme:** on the Page Properties Page tab using the dropdown.

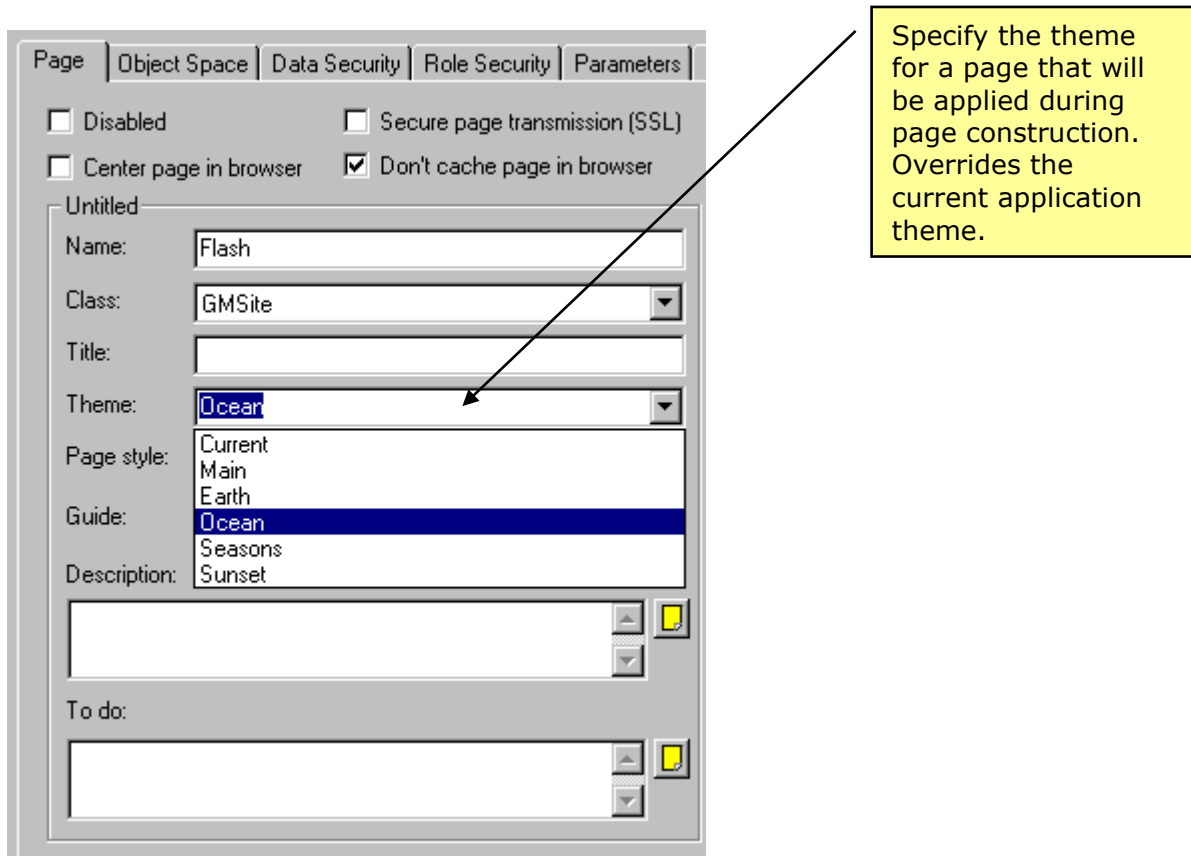


Figure 5 – Setting Theme Specification at the Page Level

## Dynamic (Multiple) Theme Construction

This option allows you to construct multiple composite themes simultaneously for the current page. The page will be constructed in the subdirectories of your application's executable directory with names corresponding to the directory names specified in Theme Architect. This option is used when you need to dynamically control the theme presentation based on input criteria or demographics of the user. Can be controlled, for instance, via a SessionVariable in your application.

In this case you would set the option on the Page Properties Page Themes tab.

	Composite theme name	Color theme	Image theme	Font theme	Directory
1	Main	Cool Blue	Main	Modern-Sans	Main
2	Earth	Earhtones	BlueMist	Classic-Serif	Earth
3	Ocean	Marine	GreenHills	Main	Oceans
4	Seasons	Earhtones	Autumn	Main	Seasons
5	Sunset	Marine	Sunset	Main	Sunset

Page | Object Space | Data Security | Role Security | Parameters | Page Themes

Construct this page for the themes specified below.

Current theme only  
 All themes in application  
 As specified for this site  
 As checked in the list below

		Theme Name
1	<input checked="" type="checkbox"/>	Main
2	<input type="checkbox"/>	Earth
3	<input type="checkbox"/>	Ocean
4	<input type="checkbox"/>	Seasons
5	<input type="checkbox"/>	Sunset
6	<input type="checkbox"/>	
7	<input type="checkbox"/>	

Note: As specified for site is not currently used

**Current theme only**  
Default – can be used to abandon multiple page construction

**All themes in application**  
Construct the current page for all composite themes

**As checked in the list below**  
Construct the composite themes checked in the list

```

    GMApp
    ├── Asp
    │   ├── Earth
    │   ├── GenImages
    │   ├── Images
    │   ├── Main
    │   ├── Oceans
    │   ├── Seasons
    │   └── Sunset
    
```

Figure 6 – Multiple Theme Construction

Managing Themes at the Page Level in Page Architect - Summary

<b>Function</b>	<b>Purpose</b>	<b>Steps</b>
<b>Managing Themes at the Page Level in Page Architect</b>		
Change the theme for a specific page	Veer from the current theme selected for your application. Used when the current style for the application might be seasonal but certain pages should retain the their own style characteristics when constructed	<ol style="list-style-type: none"> <li>1. Open the Page Properties window in Page Architect for the selected page</li> <li>2. Click the Theme dropdown on the Page tab</li> <li>3. Select the theme to be applied to the current page</li> <li>4. Click OK</li> </ol>
Implement dynamic multiple theme construction	Construct multiple composite themes simultaneously for the current page. The page will be constructed in a subdirectory of your application executable directory. Used when you need to dynamically control the theme presentation based on input criteria or demographics of the user. Can be controlled, for instance, via a SessionVariable in your application.	<p>To construct the current page for all composite themes:</p> <ol style="list-style-type: none"> <li>1. Open the Page Properties window in Page Architect</li> <li>2. Click the Page Themes tab</li> <li>3. Click the <b>All themes in application</b> radio button</li> <li>4. Click OK</li> </ol> <p>To construct the current page for selected composite themes:</p> <ol style="list-style-type: none"> <li>1. Open the Page Properties window in Page Architect</li> <li>2. Click the Page Themes tab</li> <li>3. Click the <b>As Checked in the list below</b> radio button</li> <li>4. Use checkboxes to indicate which themes to construct</li> <li>5. Click OK</li> </ol>
Abandon multiple page construction	A function required only if you have implemented multiple theme construction and your business requirements for the page change back to a single theme	<ol style="list-style-type: none"> <li>1. Open the Page Properties window in Page Architect</li> <li>2. Click the Page Themes tab</li> <li>3. Click the <b>Current Theme only</b> radio button</li> <li>4. Click OK</li> </ol>



## Importing a Theme into Your NeuArchitect Application

NeuArchitect allows you to import the themes from your current Style Repository enabling easier access to Theme Architect. The advantage to copying the themes into your current application is ease of maintenance saving you the need to open and close files to access the themes. The disadvantage is that the changes you make to the themes will be unique to the current application. Any changes to imported composite themes will be made in your application and not reflected in your Style Repository. Also you still need to access the Style OMD file to make changes to the ControlStyle and PageStyles.

Function	Purpose	Steps
<b>Importing a Theme into you Application</b>		
Importing themes into your NeuArchitect application	Gives you access to the application theme(s) without opening the Style OMD. However, the theme becomes unique to the current application	<ol style="list-style-type: none"> <li>1. From the NeuArchitect top menu, select Architects</li> <li>2. Select Themes</li> <li>3. Click the Copy Theme List from Style button</li> </ol>

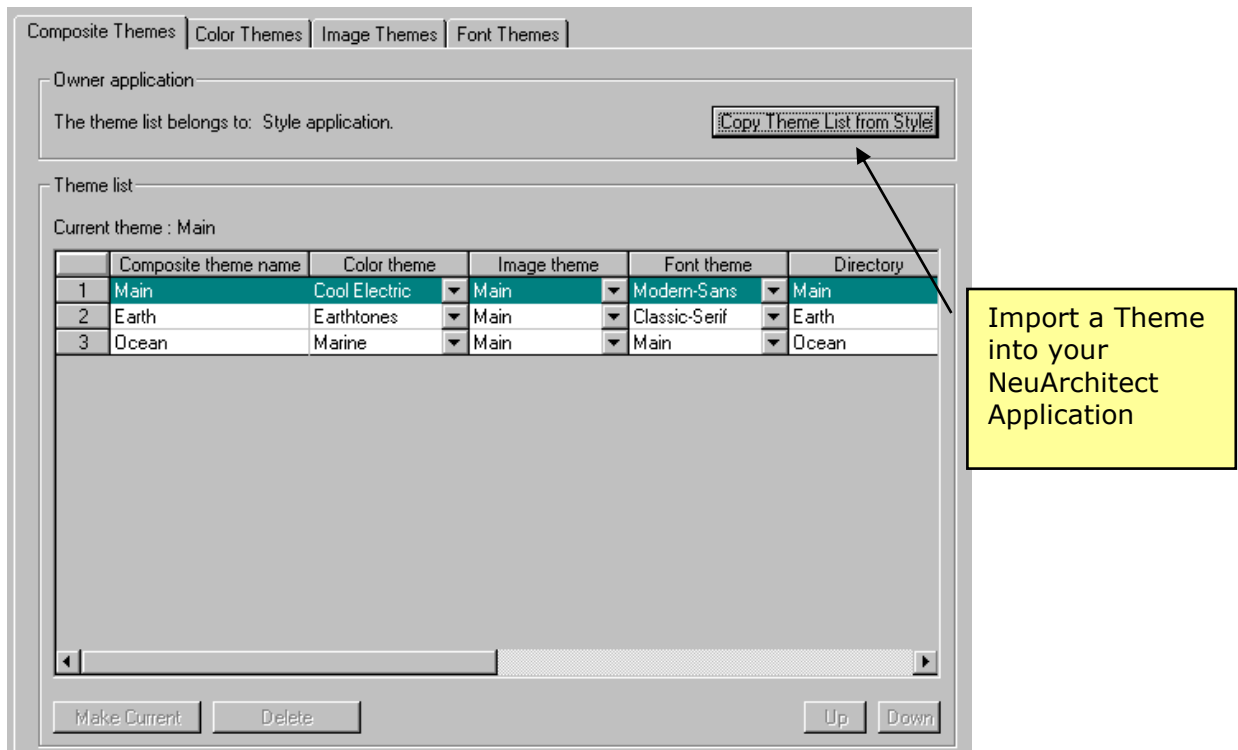


Figure 7 - Importing a Theme into a NeuArchitect Application

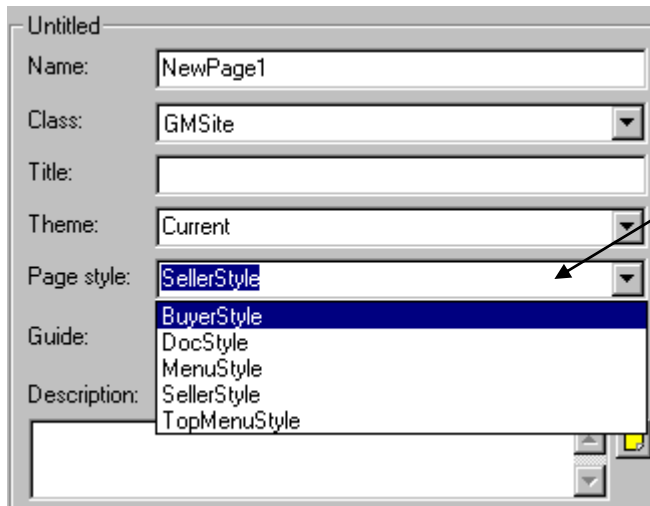
## Using a Page Style in Your NeuArchitect Application



***This is the second modeling component of Theme Architect.***

You have a few options as to where Page Styles can be applied throughout the NeuArchitect application. From Site Architect you can access the page Properties window that has a field to define the desired page style, and from the Page Architect window you may access Page properties and select the page style to enable.

<b><i>Process for using a Page Style</i></b>	
<b>1</b>	Access Page Architect
<b>2</b>	From the tree view, click to highlight desired template page
<b>3</b>	From the Page Properties Page tab, click the Page style drop-down list box and select the desired page style to apply.
<b>4</b>	Click <b>OK</b> button



Dropdown list of Page Styles on the Page Properties **Page** tab

*Figure 8 – Selecting a Page Style in Page Architect*

## Using Control Styles in a Page

When designing a page you will typically access the controls from the Control Palette. The controls that are accessed from the palette have been defined in the ControlStyle page thereby making it easy to click and drop them on the page canvas. The controls available will depend on the ControlStyle page defined and stored in the style repository you have imported.

<b>Process for Using Control Styles</b>	
<b>1</b>	Access Page Architect
<b>2</b>	From the Control Palette, click the control category and select the control type to apply.
<b>3</b>	Move your mouse to the page canvas (the cursor will appear as a plus "+" sign).
<b>4</b>	Drop the control on the canvas – Note, if you drag and size the control the default size properties of the control defined in the ControlStyle will be lost.
<b>5</b>	To modify how the control will display, double-click the control to access the Control properties.
<b>6</b>	Click <b>OK</b> to retain any changes

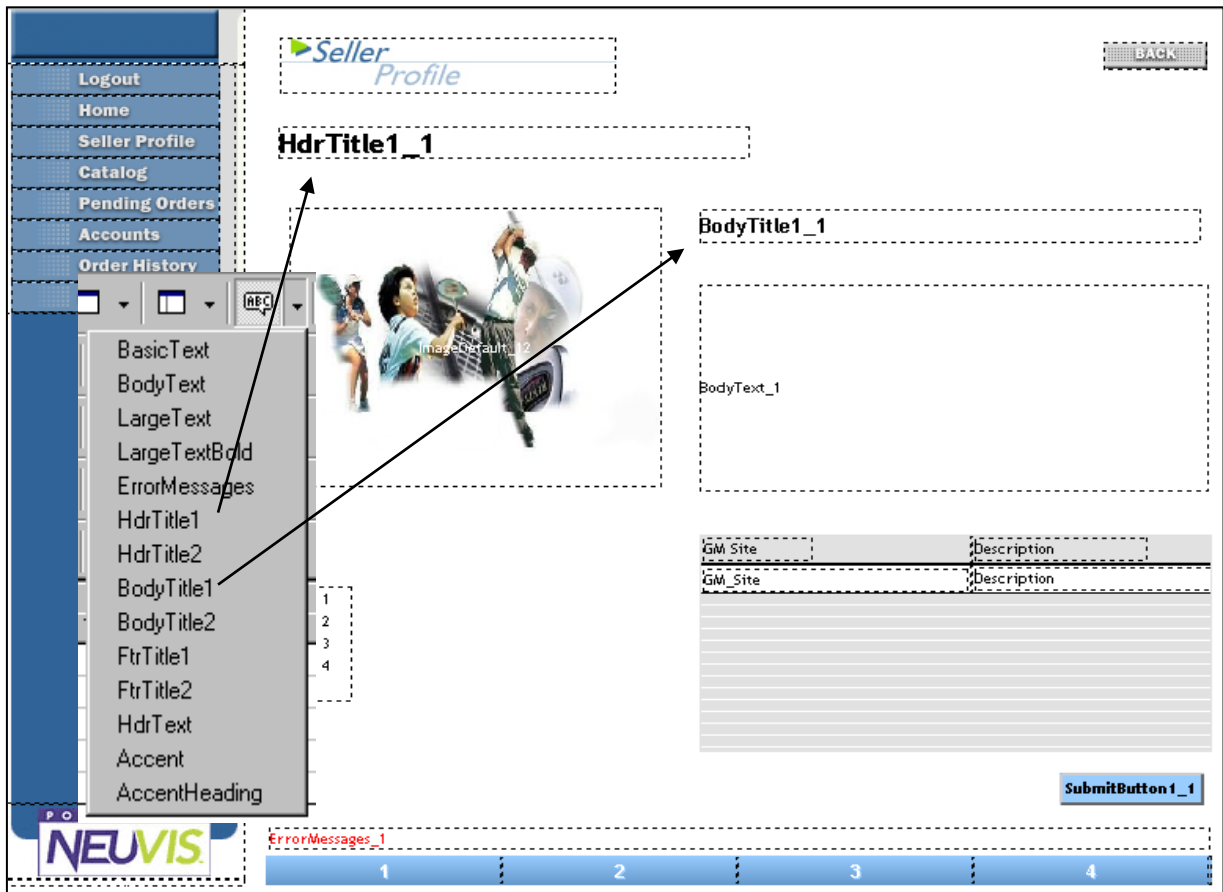


Figure 9 – Using Control Styles on a page



You can verify your answers to the true/false questions below by moving your mouse pointer over the underlined area after before question (*online versions only*).




1.        A theme provides a pre-defined set of colors, images, and fonts to be used throughout your application.
2.        Once you have defined your composite themes, you can then establish primary themes.
3.        When a page from your application is constructed, Page Architect will either apply default characteristic form the current composite style set in the SampleStyle or override the current style, applying the composite style selected for that page.
4.        Themes can be modified directly while in your application.
5.        You may define multiple themes for your application.

*Optional Review*



## Managing Styles in Your Application

In this workshop you will experiment with Importing the Style Repository you created in an earlier exercise into your GMApp application. You will experiment with changing Primary Themes, Composite Themes and controlling themes with Page Properties. You will also create a page that uses your Page Style from the Style Repository. You should also note a problem with the images used that show up when you change themes.

1. Copy the images from:
    - \applications\projects\Style\GMStyle\Siteimages to
    - \applications\projects\GMApp\Siteimages
  2. Open the Gmapp application in NeuArchitect.
  3. Click the **Import style application** browse button  and select the GMStyle.OMD file you created in an earlier exercise.
  4. Select the GMSite class, add a page named TestStyle and click the Page Architect button.
  5. Select the button and submit buttons that you created in an earlier exercise using the dropdowns on the controls in the Control Palette and place them on the canvas.
  6. In the Media Palette, Click to **highlight Main-Image** and drop it on the canvas.
  7. Click the Page Properties icon  and use the **Theme** dropdown to change from Current to **Hockey**. You should see the change in button colors, fonts and the image. If not, click the canvas to refresh.
  8. Go back into Page Properties and change the Page Theme back to **Current**.
  9. Open Theme Architect using the  icon.
  10. Click the **Copy Theme List from Style** button.
  11. Change the Image Theme for the Current Composite Theme to BaseBall. Note that button colors and fonts do not change – only the image changed.
  12. Go back into Theme Architect and change the Image Theme for the Current Composite Theme back to Golf.
  13. Go back into Page Properties and change the Theme back to **Hockey**.
  14. Go back into Theme Architect and change the Image Theme for the Current Composite Theme to Baseball.
  15. Go back into Page Properties and change the Page Theme back to **Current**. Experiment further with Theme Architect, swapping around primary themes and composite themes observing the results carefully until you feel comfortable with the way this works.
  16. From the Navigator Window, right click on the GMSite class and create a new page.
  17. Go into Page Properties on the new page and name it TestPageStyle.
  18. Using the Page Style dropdown select the MainPage Style.
  19. Experiment with adding a few additional controls to this page.
- Optional** – If you have time, set the Page Themes tab to **All themes in application**. Reconstruct the page and use Explorer to verify creation of the dynamic subdirectories. Switch back to **Current theme only** and reconstruct.
- Optional** – If you have Photoshop or Paint Shop Pro available, resize and optimize the three images used for Main-Image and retest.

*Managing the Style Repository in Your Application – Workshop 1*

# Section 4 - Netscape vs IE

## Section Objectives



**This is the second modeling component of Theme Architect.**

After completing this section you should be able to:

- List the browsers supported in NeuArchitect
- Specify the differences between browsers
- Identify some of the rendering differences between Netscape and Internet Explorer
- Preview a NeuArchitect application in Netscape and resolve browser differences



Figure 1 – Netscape and Internet Explorer

## Overview

This section explores some of the issues in designing applications for contrasting interfaces with technological differences. Typically, developers will either be constructing applications destined for an Internet browser (HTML) or a digital appliance such as a cell phone (WML). The following table lists the NeuArchitect supported browsers and platforms:

<b>Browsers</b>	<b>Operating Platforms</b>
Internet Explorer 4.0 and above	PC and Macintosh
Netscape 3.0 and above	PC and Macintosh
Netscape 6.0 (Beta)	PC only
WAP	Nokia Simulator Toolkit Phone.com Simulator Motorola StarTac ST7860W phone

## Rendering Differences – The Need to Test

When a web page is written, or scripted, the goal is to create a visually attractive, and informative presentation. Many elements such as font type and size, colors, graphics, animation, sound, page structure, and others, are taken into consideration, and incorporated into a final design. All of these design considerations are based on the computer language referred to as HTML, (Hyper Text Markup Language,) which is the accepted standard communication format for the Internet and networks that interface with those systems.

Unfortunately, browser programs do not conform to any established HTML standard. There are cases where basic HTML elements or code may not be read or accepted by specific browsers. Some browsers have developed their own extensions and responses to HTML, which are not recognized by other browser programs, and are not standard HTML code. These "browser specific " commands are usually only responded to by that one browser (Netscape or Microsoft Internet Explorer, for example,) and are completely ignored by other browsers, and in such cases, have no effect. The following is an example of the behavior of different effects:

Depicted in Figure 1 below is the Page Architect canvas and the resultant renderings of the generated page in Netscape and Internet Explorer. The designer of the page used two Label controls: one for the page heading and one for some custom code for a marquee. The designer also used an Object Grid (defaults unchanged) to display some data on the page. While you would be much more likely to be using Object Grids, the custom code for the marquee helps to illustrate the point – always test in common browsers.

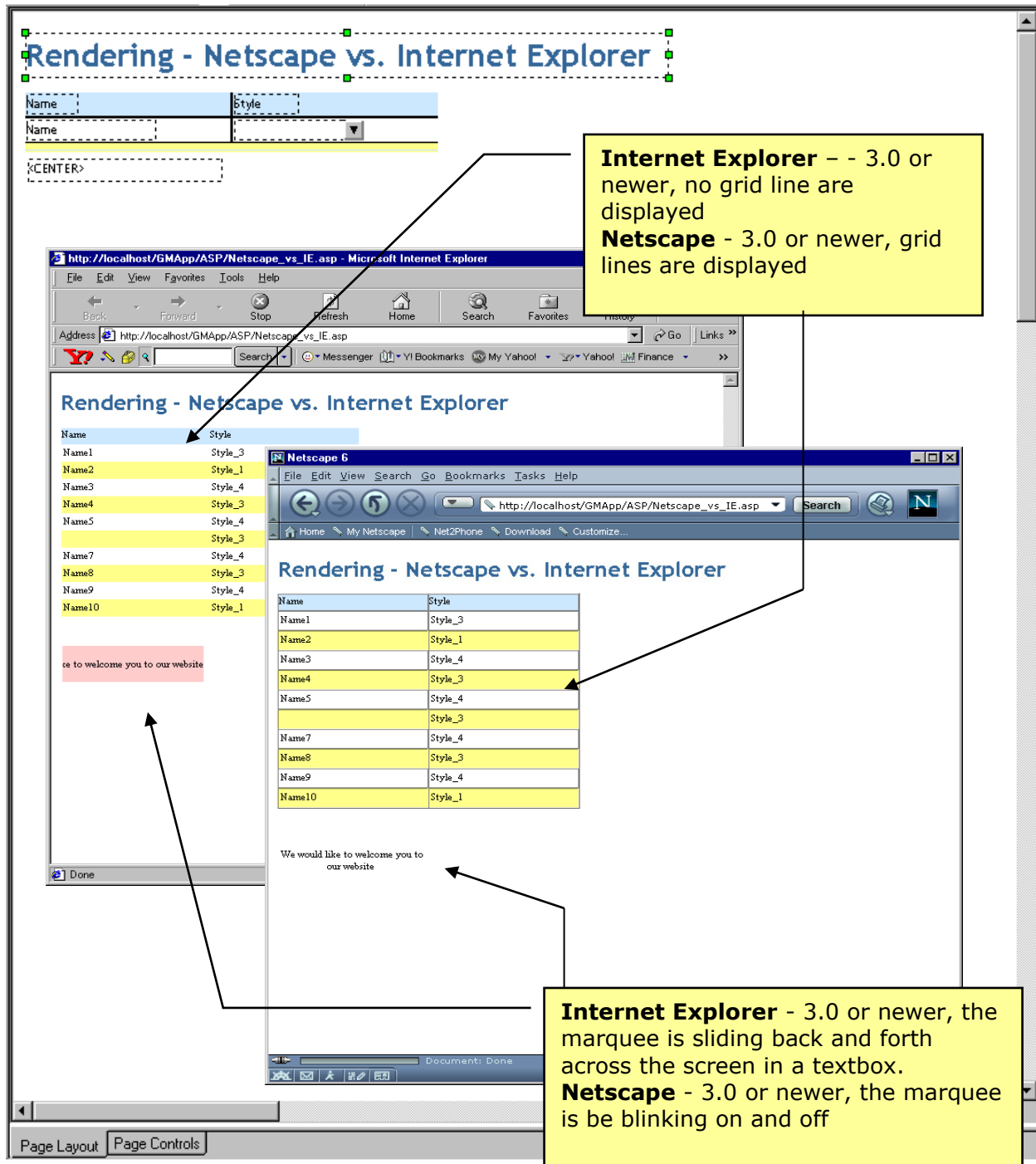


Figure 1 – Rendering - Netscape vs Internet Explorer



True or False? Unfortunately, browser programs do not conform to any established HTML standard.





## Differences in Browser Renderings

In this workshop you will be creating a simple data connected page that illustrates some of the browser differences you might find when developing an application. You'll first check out the differences and then take action to modify the defaults of the Object Grid (this is one browser disparity you can easily resolve). For this workshop, you will need Netscape, Internet Explorer and your GMApp ready to go.

1. Add a new page to the GMSite Class
2. Go to Page Properties  
Name it Netscape\_vs\_IE  
Add GMSite to the ObjectSpace
3. Select an Object Grid from the Control palette and drop it on the canvas
4. Access the Object Grid properties and set the Object Set to ObjectSpace.GM\_Site1 on the Object Set tab
5. Click OK to close the Object Grid properties
6. Click on both Name and Style in the Attributes Palette to add them to the Object Grid
7. Select a Label from the Control palette and drop it on the canvas
8. Access the label properties
9. From the Control tab, select multi line for the label and copy and paste in the following:

```
<CENTER>  
<BLINK>  
<MARQUEE BGCOLOR="ffcccc" BEHAVIOR="alternate">  
We would like to welcome you to our website  
</MARQUEE>  
</BLINK>  
</CENTER>
```

10. Generate and preview in Internet Explorer
11. Start up Netscape
12. Type (or copy/paste from IE) the URL in Netscape:
13. Note the differences
14. Back in Page Architect, access the Control properties for the Object Grid again
15. On the Display tab, change the Border Depth to 0
16. Regenerate
17. Preview again in Internet Explorer
18. Refresh in Netscape

*Managing the Style Repository in Your Application – Workshop 1*

## Section 5 HTML and WML

## HTML Notes









**This is the second modeling component of Theme Architect.**








This code is included here is for sample purposes only. It can and will change as part of the normal development process.

- [...]: Represents values that can be changed by the developer.
- Any control will be generated as an <IMG ...> tag if **Generate image** checkbox in the Display property page is checked.

	Control	Description	Special Considerations
	Container	Used to keep together a set of controls	
<code>&lt;TABLE BORDER=0 CELLSPACING=0 CELLPADDING=0 WIDTH=[width]&gt;</code>			
	FrameSet	A frameset of rows and columns, dividing the control into frame containers that contain other page. Only control in a page..	
<pre> &lt;FRAMESET FRAMEBORDER=No FRAMESPACING=0 Border=0 Rows="[row1 height],[row2 height]" Cols="[column1 width],[column2 width]"&gt; &lt;FRAME marginheight=0 marginwidth=0 FrameBorder=No Scrolling=[scrolling-Auto/Yes/No] SRC="[file name]" NAME="[frame name]"&gt; ...other frames... &lt;/FRAMESET&gt;                     </pre>			
	Label	Used for static text	Can be constructed as Link.
<pre> &lt;Span Name="[label name]" Title="" STYLE="background-color:[color];font-family:[font family],Arial ;font-size:[font size]pt;font-weight:[font weight];text-align:[alignment];text-decoration: line-through underline;color:[color];WIDTH:[width]px;HEIGHT:[height]px;Z-INDEX:1;"&gt;Label&lt;/SPAN&gt;                     </pre> <p>“text-decoration: line-through underline;” is generated when Underline and Strikethrough in Text display section in Display property page are checked;</p> <p>“WIDTH:[width]px;HEIGHT:[height]px;Z-INDEX:1;” is generated when Alignment is not left.</p>			
	Button	Used as an event-button to launch pages and other actions	Can be constructed as Link.
<pre> "&lt;A HREF = "#" onmousedown="flipImage([button name], '[clicked image]')" onmouseup="OnMenuBUp([button name], '[selected image]')" onmouseover="flipImage([button name], '[mouseover image]')" onmouseout="flipImage([button name], '[normal image]')"&gt;"                     </pre> <p>is generated when “Mouse effects” in Display property page are specified;</p> <p>Usually, a user either specify back images or check Generate image so that &lt;IMG...&gt; tag is generated; otherwise, an &lt;SPAN...&gt; tag is generated.</p>			
	Image Button	A static image that may be used as a button.	This control always generates <IMG...> tag whether or not Generate image is checked. Can be constructed as Link.
<code>&lt;IMG SRC="[image name]" Border="0" ALT="" Name="[image button name]" Title="" &gt;</code>			
	Text Field	Used to hold static text	Can be constructed as Link when set to Read-Only. When set to Read-Only, this control generates same as a Label control.

	Control	Description	Special Considerations
Input		<p>&lt;Input TYPE= TEXT Name="[text field name]" Title="" SIZE=[text control size in characters] maxLength=[maximum number of characters that can be entered in a text control] Value = "[text field value]" &gt;</p> <p>"Multi-line" in the Control property page is unchecked by default. If it is checked, it generates the same code as a Rich Text Field generates;</p> <p>If "Read only" in the Projection property page is checked, it generates the same code as a Label generates.</p>	
R/O		<p>&lt;Span Name="[ text field name]" Title="" STYLE="background-color:[color];font-family:[font family],Arial ;font-size:[font size]pt;font-weight:[font weight];text-align:[alignment];text-decoration:line-through underline;color:[color];WIDTH:[width]px;HEIGHT:[height]px;Z-INDEX:1;" &gt; text field data&lt;/SPAN&gt;</p> <p>"text-decoration: line-through underline;" is generated when Underline and Strikethrough in Text display section in Display property page are checked;</p> <p>"WIDTH:[width]px;HEIGHT:[height]px;Z-INDEX:1;" is generated when Alignment is not left.</p>	
	Rich Text Field	Used to hold RTF text fields	Can be constructed as Link when set to Read-Only. When set to Read-Only, this control generates same as a Label control.
Input		<p>&lt;TEXTAREA WRAP Name="[rich text field name]" Title="" COLS = [text control size in characters] ROWS = [number of rows]&gt;[text field value]&lt;/TEXTAREA&gt;</p> <p>"Multi-line" in the Control property page is checked by default. If it is unchecked, it generates the same code as a Text Field generates;</p> <p>If "Read only" in the Projection property page is checked, it generates the same code as a Label generates.</p>	
R/O		<p>&lt;Span Name="[ text field name]" Title="" STYLE="background-color:[color];font-family:[font family],Arial ;font-size:[font size]pt;font-weight:[font weight];text-align:[alignment];text-decoration:line-through underline;color:[color];WIDTH:[width]px;HEIGHT:[height]px;Z-INDEX:1;" &gt; text field data&lt;/SPAN&gt;</p> <p>"text-decoration: line-through underline;" is generated when Underline and Strikethrough in Text display section in Display property page are checked;</p> <p>"WIDTH:[width]px;HEIGHT:[height]px;Z-INDEX:1;" is generated when Alignment is not left.</p>	
	Number Field	Used to hold numeric fields	Same as Text Field
Input	See Text Field		
R/O	See Text Field		
	Date Field	Used to hold date fields	Same as Text Field
Input	See Text Field		
R/O	See Text Field		
	Time Field	Used to hold time fields	Same as Text Field
Input	See Text Field		
R/O	See Text Field		
	Image Field	Used to hold dynamic graphic images	Image Field is very similar to Image with the exception that Image Field can read the image source from a database. Can be constructed as Link.
<p>&lt;IMG SRC="[image name]" Border="0" ALT="[control name]" Name="[control name]" Title="" &gt;</p>			
	CheckBox	Used to select a binary option	Can be constructed as Link.
<p>&lt;Input TYPE=CHECKBOX ["CHECKED" or "" ] Name="[checkbox name]" Title="" &gt;&lt;Span STYLE="background-color:[color];font-family:[font family],Arial ;font-size:[font size]pt;font-weight:[font weight];text-align:[alignment];color:[color];"&gt;[text]&lt;/SPAN&gt;</p>			
R/O	<p>&lt;Input TYPE=CHECKBOX ["DISABLED CHECKED" or "DISABLED"] Name="[checkbox name]" Title="" &gt;&lt;Span STYLE="background-color:[color];font-family:[font family],Arial ;font-size:[font size]pt;font-weight:[font weight];text-align:[alignment];color:[color];"&gt;[text]&lt;/SPAN&gt;</p>		

	Control	Description	Special Considerations
	<b>Radio Buttons</b>	Used to select from a mutually exclusive set of options	Can be constructed as Link.
<p>For each Radio Button, NA generates:</p> <pre>&lt;Input TYPE=RADIO Value="[radio button value (incremented number)]" Name="[radio button name]" Title="" &gt;&lt;Span STYLE="background-color: [color];font-family: [font family],Arial ;font-size: [font size]pt;font-weight: [font weight];text-align: [alignment];color: [color];"&gt;[text]&lt;/SPAN&gt;</pre>			
R/O		For each Radio Button in Read-Only mode, NA generates:	
<pre>&lt;Input TYPE=RADIO DISABLED Value="[radio button value (incremented number)]" Name="[radio button name]" Title="" &gt;&lt;Span STYLE="background-color: [color];font-family: [font family],Arial ;font-size: [font size]pt;font-weight: [font weight];text-align: [alignment];color: [color];"&gt;[text]&lt;/SPAN&gt;</pre>			
	<b>List Box</b>	Used to select from a mutually exclusive set of options	Can be constructed as Link. When set to Read-Only, this control generates same as a Label control.
<p>For a List Box, NA generates:</p> <pre>&lt;SELECT Name="[list box name]" Title="" STYLE="background-color: [color];font-family: [font family],Arial ;font-size: [font size]pt;font-weight: [font weight];text-align: [alignment];color: [color];WIDTH: [width]px;HEIGHT: [height]px;Z-INDEX:1;" SIZE=[number of rows] &gt;</pre> <p>For each list item, NA generates:</p> <pre>&lt;OPTION VALUE ="[list item value (incremented number)]" &gt;[text].</pre> <p>For a List Box, NA generates:</p> <pre>&lt;/SELECT&gt;</pre>			
R/O			
<pre>&lt;Span Name="[list box name]" Title="" STYLE="background-color: [color];font-family: [font family],Arial ;font-size: [font size]pt;font-weight: [font weight];text-align: [alignment];text-decoration: line-through underline;color: [color];WIDTH: [width]px;HEIGHT: [height]px;Z-INDEX:1;" &gt; list box select data&lt;/SPAN&gt;</pre> <p>When Alignment is not left, the following is generated</p> <pre>"WIDTH: [width]px;HEIGHT: [height]px;Z-INDEX:1;"</pre>			
	<b>Drop Down</b>	Used to select from a mutually exclusive set of options	Can be constructed as Link. When set to Read-Only, this control generates same as a Label control.
<p>For a Drop Down, NA generates:</p> <pre>&lt;SELECT Name="[drop down name]" Title="" STYLE="background-color: [color];font-family: [font family],Arial ;font-size: [font size]pt;font-weight: [font weight];text-align: [alignment];color: [color];WIDTH: [width]px;HEIGHT: [height]px;Z-INDEX:1;" SIZE=1 &gt;</pre> <p>For each drop down item, NA generates:</p> <pre>&lt;OPTION VALUE ="[drop down item value (incremented number)]" &gt;[text]</pre> <p>For a Drop Down, NA generates:</p> <pre>&lt;/SELECT&gt;</pre>			
R/O			
<pre>&lt;Span Name="[ drop down name]" Title="" STYLE="background-color: [color];font-family: [font family],Arial ;font-size: [font size]pt;font-weight: [font weight];text-align: [alignment];text-decoration: line-through underline;color: [color];WIDTH: [width]px;HEIGHT: [height]px;Z-INDEX:1;" &gt; drop down select data&lt;/SPAN&gt;</pre> <p>When Alignment is not left, the following is generated</p> <pre>"WIDTH: [width]px;HEIGHT: [height]px;Z-INDEX:1;"</pre>			
	<b>Menu</b>	A menu control consists of a number of menu items, each one causing an action. Menu controls also provide code for supporting pop-up menus.	

	Control	Description	Special Considerations
		For each menu item, NA generates: <pre>&lt;A HREF=[page to load] onmousedown="flipImage([menu item name], '[clicked image]') " onmouseup="OnMenuBUp([menu item name], '[selected image]') " onmouseover="OnMenuBOver([menu item name], '[mouseover image]') " onmouseout="OnMenuBOut([menu item name], '[normal image]') "&gt;&lt;IMG SRC="[normal image]" Border="0" WIDTH="[width]" HEIGHT="[height]" Name="[menu item name]" Title="" &gt;</pre>	
	Line	A horizontal line used to separate sections in a page.	
		<pre>&lt;HR Width=[width] Size=[height] Color="[color]" &gt;</pre>	
	Grid	Used to present a uniform and tabular view of data	
		If Layout type is WYSIWYG, Grid is generated as a <b>&lt;TABLE...&gt;</b> tag and each cell in the Grid is generated as a <b>&lt;TABLE...&gt;</b> tag also; If Layout type is Expand vertically, Grid is generated as a <b>&lt;TABLE...&gt;</b> tag and each cell in the Grid is generated as a <b>&lt;TABLE...&gt;</b> tag if there is more than one control in the cell or either the left margin or the top margin of the only control in the cell is greater than 5. Otherwise, each cell is generated as a <b>&lt;TD...&gt;</b> tag; If Layout type is Paragraph flow, Grid is generated as a <b>&lt;TABLE...&gt;</b> tag and each cell in the Grid is generated as a <b>&lt;TD...&gt;</b> tag.	
	ObjectTable	Used to present data in a free-form table – that can be arranged in a number of views	Same as Grid. Except the user will define how a cell should appear and NeuArchitect will dynamically generate all cells required based on data in the database.
		See Grid	
	ObjectGrid	Used to present data in a spreadsheet-form table	Same as Grid. Except the user will define how a table row should appear and NeuArchitect will dynamically generate all rows required based on data in the database.
		See Grid	
	Submit Button	Used to submit page data and launch server-side logic to process it.	
		<pre>&lt;A href="#" OnClick="SubmitButton([form name], [submit button name]);return false;" &gt;&lt;IMG SRC="[image for submit button]" ALT="" Border="0" WIDTH="[width]" HEIGHT="[height]" Name="[submit button name]" Title="" &gt;&lt;/A&gt;</pre>	
	Custom Control	A control that may include external custom code.	The user specifies parametrized custom code. The language in which the code is written must be consistent with the technology selected, i.e. Java or JavaScript.
		The generated code is the custom code, with Parameter names replaced by the Parameter Values .	
	Upload Control	A control that can be used to upload files to the server	
		<pre>&lt;Input type=FILE Name="[upload control name]" &gt;</pre>	

# Wireless Markup Language



**This is the second modeling component of Theme Architect.**

Wireless Markup Language is a tag-based language like HTML, designed for hardware constrained, narrow-band wireless devices with limited input/output capabilities. WML pages use a card-and-deck metaphor, where a card is a single unit of user interaction and a deck is a related set of cards. Like an HTML page, a card typically contains some viewable content and perhaps some user choices for selecting an option, entering some data, or navigating to another card. Instructions in a card may invoke new static or dynamic decks from content servers.





The WML specification defines the intent of the individual language tags, not how a particular user agent should implement those tags. Each WAP device vendor has a great deal of latitude in deciding how to present WML tags to the user, the data entry mechanism available, the size of the screen and so on. The following table maps NeuArchitect controls to WML tags and any identifies any known special considerations. The code shown in the table is the code that the presentation layer sends to the browser.

### Important

NeuArchitect support for WML does not include multi-theme construction.

- [...]: Represents values that can be changed by the developer.
- Any control will be generated as an `<IMG ...>` tag if **Generate image** checkbox in the Display property page is checked.

	Control	Description	Special Considerations
	Container	Used to keep together a set of controls. In WML it represents a "screen" of data.	
<pre>&lt;card id=["card-name"] newcontext="true" ordered=["true" or "false"] title=[label string] &gt; &lt;/card&gt;</pre>			
	Label	Used for static text	Can be constructed as Link.
<p>A label control will generate paragraph code in WML.</p> <pre>&lt;p mode="wrap" &gt; [Label String] &lt;/p&gt;</pre> <p>The following elements will be used when the label text properties are set accordingly:  <b>&lt;b&gt;</b> Bold, <i>&lt;i&gt;</i> Italic, <u>&lt;u&gt;</u> Underline.  When the label is an HREF (Anchor link), the tag generated code will use:</p> <pre>&lt;a href=["URL"] [text string] &gt; &lt;/a&gt;</pre>			

	Control	Description	Special Considerations
	<b>Image Field</b>	Used to hold dynamic graphic images	The WAP forum has created its own image format. It is called: Wireless Bitmap Format or WBMP. Can be constructed as Link.
<pre>&lt;img align="middle" alt=["alt string"] src=["URL"] height=100% width=100% /&gt;</pre>			
	<b>Text Field</b>	Used to hold dynamic text and accept input from user.	Can be constructed as Link when set to Read-Only.
<b>Input</b>	<pre>&lt;Input type= ["text" or "password"] name=["field name"] title=["label string"] value=[text to load] /&gt;</pre>		
<b>R/O</b>	When set to read-only, the text field will be constructed in the same way as a "Label" control. <pre>&lt;p mode="wrap"&gt;[field name]&lt;/p&gt;</pre>		
	<b>Submit Button</b>	Used to submit page data and launch server-side logic to process it.	
Uses the "do" event to perform an "accept" operation. <pre>&lt;do type='accept' label=["label string"] &gt; &lt;go href=["post page"] method="post" &gt; &lt;postfieldname=["text field name 1"] value=["default value 1"] /&gt; &lt;postfieldname=["text field name 2"] value=["default value 2"] /&gt; . . &lt;postfieldname=["text field name n"] value=["default value n"] /&gt; &lt;/go&gt;</pre>			
	<b>ObjectTable</b>	Used to present data in a free-form table	Due to the limitations of WML, the Object Table is generated as a text paragraph <p> with "rows" separated by   tags.



## Section 6 - Integrating Multi-Media into Your Application

Perhaps the most powerful aspect of the computing technology is the ability to combine text, graphics, sounds, and moving images in meaningful ways. The promise of multimedia has been slow to reach the web because of bandwidth limitations, but each day brings new solutions. The options enumerated here are certainly not the only ones and will surely soon become outdated but they are the solutions we use in our work and have proved to be the most practical and effective for our purposes.

### **Splash vs. content**

Web designers must always be considerate of the consumer. A happy customer will come back, but one who has been made to wait, and is then offered goods that are irrelevant, will very likely shop elsewhere. Since multimedia comes with a high price-tag in terms of bandwidth, it should be used sparingly and judiciously.

Splash screens have become a common location for multimedia elements. Like the cover to a book, splash screens are intended to entice users into a site to open the book and read what's inside. Animations and sound can pique a user's curiosity, compelling them to enter the site and explore. Any page element, however, that is not relevant to the content is simply distracting.

The options for content are essentially defined by bandwidth. Audio files can be compressed so effectively that sound can now be considered for site content, particularly for intranet sites.

Animation files at present are not terribly useful as content because of compression limitations. Most animation file formats require the file to be fully downloaded before it can be played, so file size is a serious limitation. And most popular animation formats do not support compression, so if one content-rich GIF image is 30k, two combined makes 60k, and so on.

If your site will be accessed by people using modems, forget about digital video, at least for the moment. The quality compromises required to deliver video to modems altogether obviate its usefulness. However, if your site is intended for use on an intranet, video content is a definite possibility.

### **Plug-ins**

Each day brings a new plug-in that allows users to see new and exciting things using their favorite browser software. This is especially true of multimedia; the options for encoding and delivering audio, animations, and video are dizzying. It is tempting to create files that utilize the functionality offered by these custom plug-ins, but there are two considerations designers should bear in mind. First, you will lose a large number of users when they hit the "Plug-in not supported, etc..." dialog box. The bother and potential confusion of downloading and installing plug-ins will deter a large percentage of users. Secondly, it is not prudent to create content in a custom file format which could quickly become obsolete. It is best to create your multimedia content in the standard formats for operating systems and browser software.

## NeuArchitect Custom Controls

NeuArchitect controls give the developer the flexibility to supplement code functionality and to override the NeuArchitect generated HTML for the elements and controls of a web page. Utilizing the same functionality, a Custom Control gives the ability to utilize graphic techniques to optimize the look and efficiency of web page graphics. From an open page in PageArchitect, the developer selects the properties dialog of the Custom Control and then the Custom HTML tab.

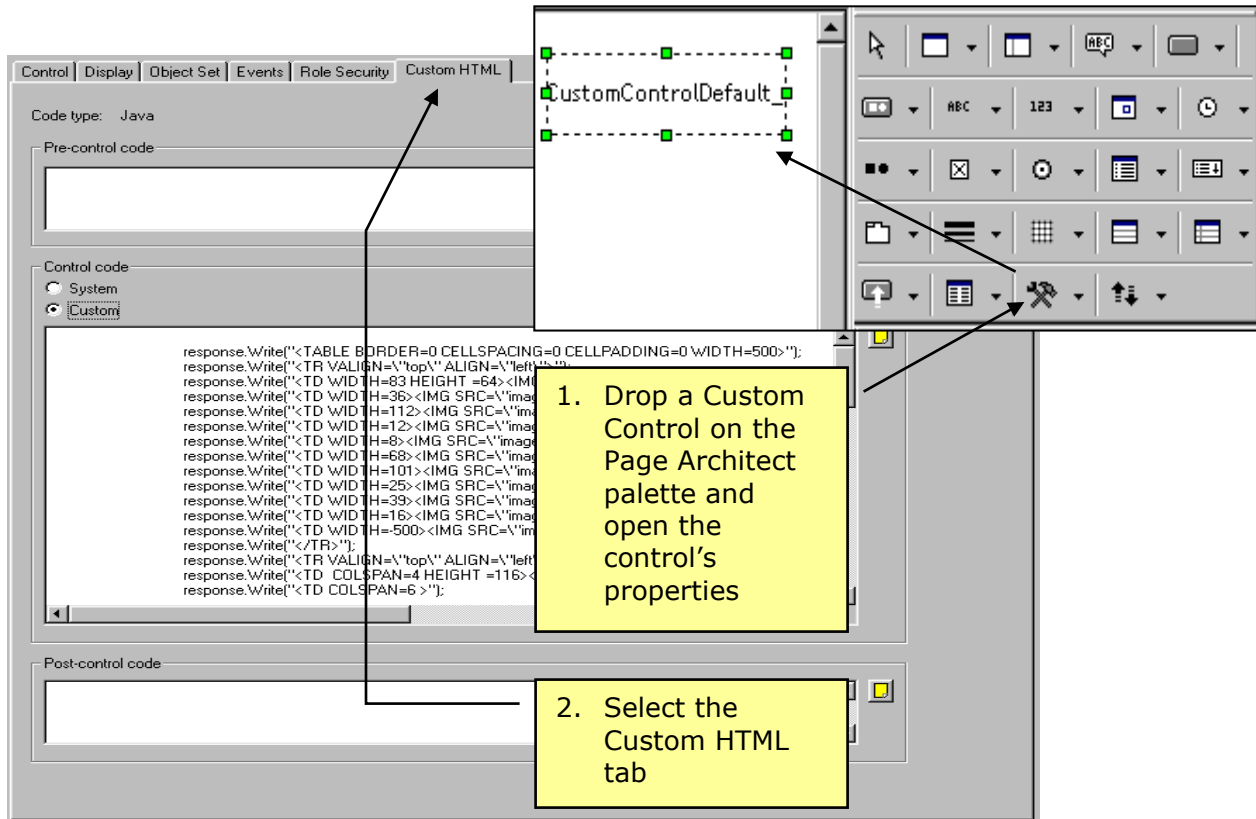


Figure 1 – Custom HTML tab

<b>Code Type</b>	<b>Description</b>
<b>Pre-control code</b>	Code to be executed prior to the execution of the HTML system generated code.
<b>Post-control code</b>	Code to be executed after the execution of the HTML system generated code.
<b>Control-code type</b>	Indicates whether system HTML or custom HTML should be used for the specified control.
<b>Custom Control-code</b>	Code placed in this edit window will override the system HTML if the Control-code-type Custom is chosen.
<b>Get System HTML</b>	Retrieves the NeuArchitect system HTML and displays it in the Control-code edit window. This is a convenient feature for the developer who needs to make a minor change to the existing HTML. Once retrieved, the system code can be copied, pasted as custom and modified as needed.

## Custom HTML – Image Slicing

In the Visual Web Development course, you spent some time working with custom HTML for your image maps. Custom HTML also enables you to apply techniques to make your web application more efficient. Shown below is code generated from Photoshop 5. You will be using this code in your next workshop.

```
<HTML>
<HEAD>
<TITLE>splashmain-01</TITLE>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">
</HEAD>
<BODY BGCOLOR=#FFFFFF>
<!-- ImageReady Slices (splashmain-01.gif) -->
<TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
  <TR>
    <TD>
      <IMG SRC="../../../siteimages/splashmain-01_01.gif" WIDTH=370 HEIGHT=47</TD>
    </TR>
    <TR>
    <TD>
      <IMG SRC="../../../siteimages/splashmain-01_02.gif" WIDTH=370 HEIGHT=47</TD>
    </TR>
    <TR>
    <TD>
      <IMG SRC="../../../siteimages/splashmain-01_03.gif" WIDTH=370 HEIGHT=46</TD>
    </TR>
    <TR>
    <TD>
      <IMG SRC="../../../siteimages/splashmain-01_04.gif" WIDTH=370 HEIGHT=47</TD>
    </TR>
    <TR>
    <TD>
      <IMG SRC="../../../siteimages/splashmain-01_05.gif" WIDTH=370 HEIGHT=47</TD>
    </TR>
  </TABLE>
<!-- End ImageReady Slices -->
</BODY>
</HTML>
```

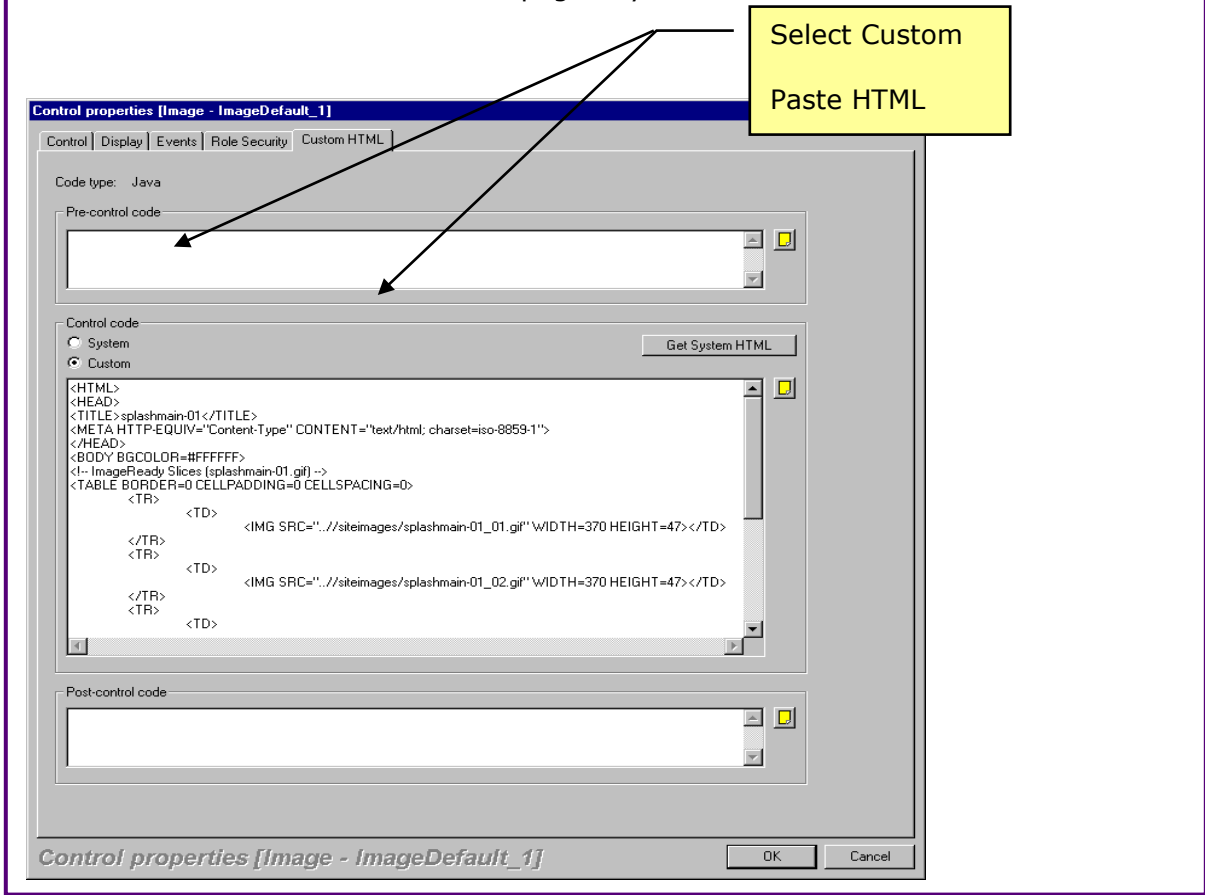
Figure 2 – HTML Generated from Photoshop 5 for Slicing an Image



## Using Custom HTML – Image Slicing

**Before you start:** If working online, copy the Image Slicing HTML from Figure 2 into your Windows text buffer.

1. Open GMSite in Page Architect
2. To simplify the process for this exercise, switch you Application Technology to Microsoft JavaScript with ASP pages.
3. Go to Page Architect.
4. Select an Image Button from the Control Palette and drop it in the Page Architect canvas.
5. Access the properties of the new button and select the Custom HTML tab.
6. Select the Custom button
7. Paste in the custom code from Figure 2.
8. Click OK
9. Construct HTML and Preview the page in your browser.



Custom HTML Image Slicing – Workshop 1

## Custom HTML – Macromedia Flash

Macromedia Flash (<http://www.macromedia.com/software/flash/>) delivers low-bandwidth animations and presentations. It offers scripting capabilities and server-side connectivity for enhancing applications. The online audience will be able to view it with the Macromedia Flash Player.

A few engaging sites are listed for you to explore.....

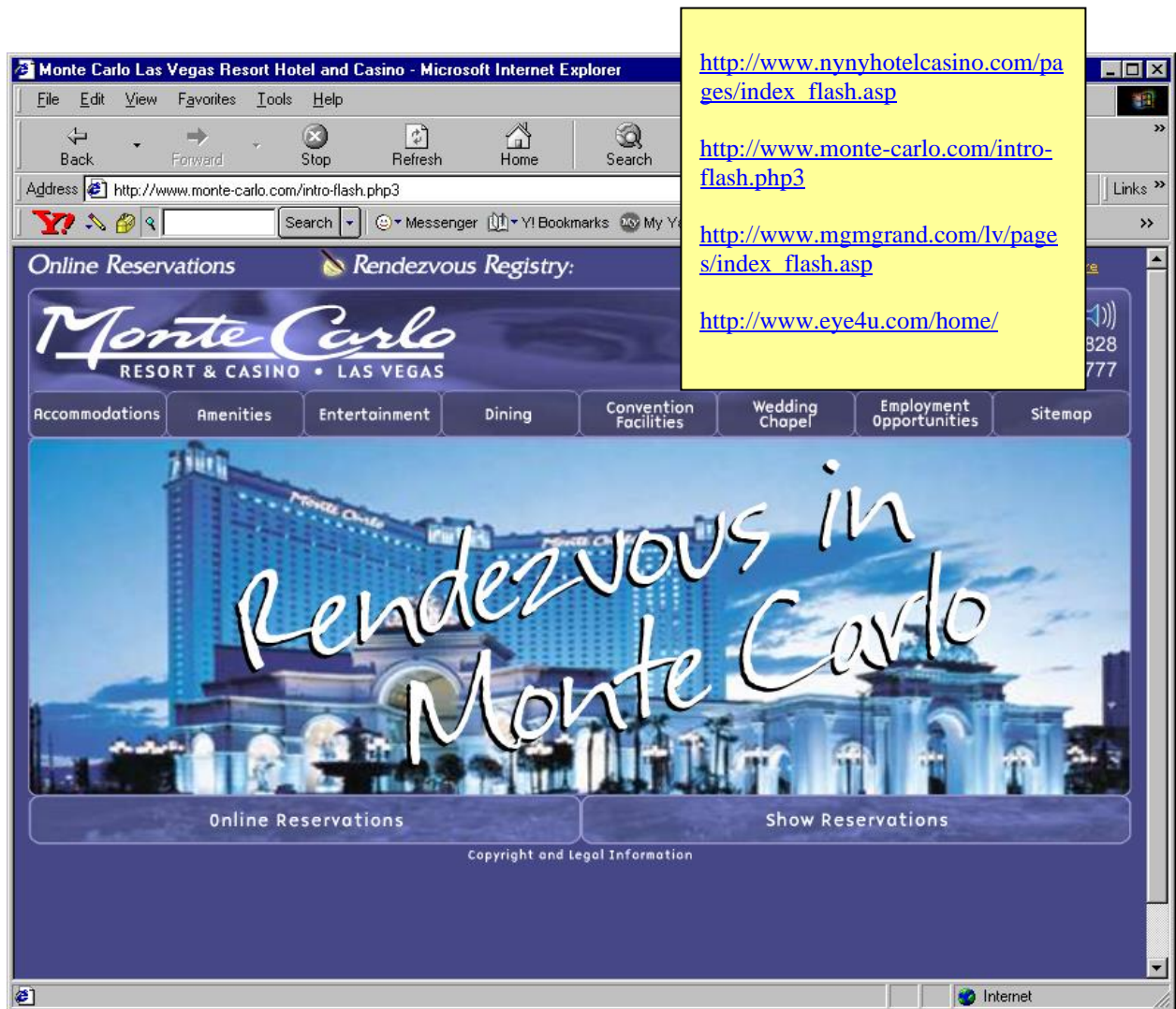


Figure 3 – Sample WWW Flash Page

The code shown below was generated from Macromedia Flash 5. You will be using this code in your next workshop.

```
<HTML>
<HEAD>
<TITLE>SpotlightMask</TITLE>
</HEAD>
<BODY bgcolor="#FFFFFF">
<!-- URL's used in the movie-->
<!-- text used in the movie-->
<OBJECT classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#versi
on=5,0,0,0"
WIDTH=550 HEIGHT=400>
  <PARAM NAME=movie VALUE="SpotlightMask.swf"> <PARAM NAME=quality VALUE=high>
<PARAM NAME=bgcolor VALUE=#FFFFFF> <EMBED src="SpotlightMask.swf" quality=high
bgcolor=#FFFFFF WIDTH=550 HEIGHT=400 TYPE="application/x-shockwave-flash"
PLUGINSOURCE="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_Version
=ShockwaveFlash"></EMBED>
</OBJECT>
</BODY>
</HTML>
```

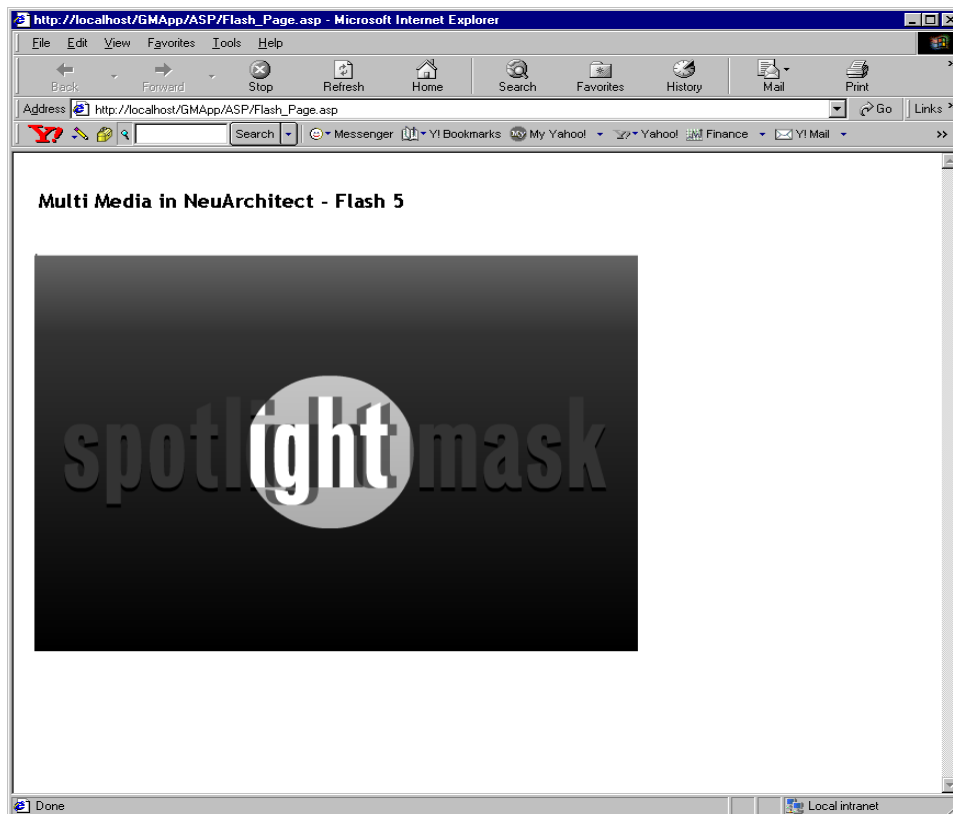
*Figure 4 – HTML Generated from Flash 5*



## Using Custom HTML – Flash 5

**Before you start:** If working online, copy the HTML from Figure 3 into your Windows text buffer.

1. Open GMSite in Page Architect
2. To simplify the process for this exercise, switch you Application Technology to Microsoft JavaScript with ASP pages.
3. Create a page named Flash\_Page and go to Page Architect.
4. Select a Custom Control from the Control Palette and drop it in the Page Architect canvas.
5. Access the properties of the new control and select the Custom HTML tab.
6. Select the Custom button
7. Paste in the custom code from Figure 2.
8. Click OK
9. Construct HTML and Preview the page in your browser.



Custom HTML - Flash – Workshop 2

## Custom HTML – Macromedia Fireworks 4

Macromedia Fireworks (<http://www.macromedia.com/software/fireworks/>) enables you to create, edit, and animate Web graphics using bitmap and vector tools. You can use export controls to optimize your images, give them advanced interactivity, and export them.

A few engaging sites are listed for you to explore.....

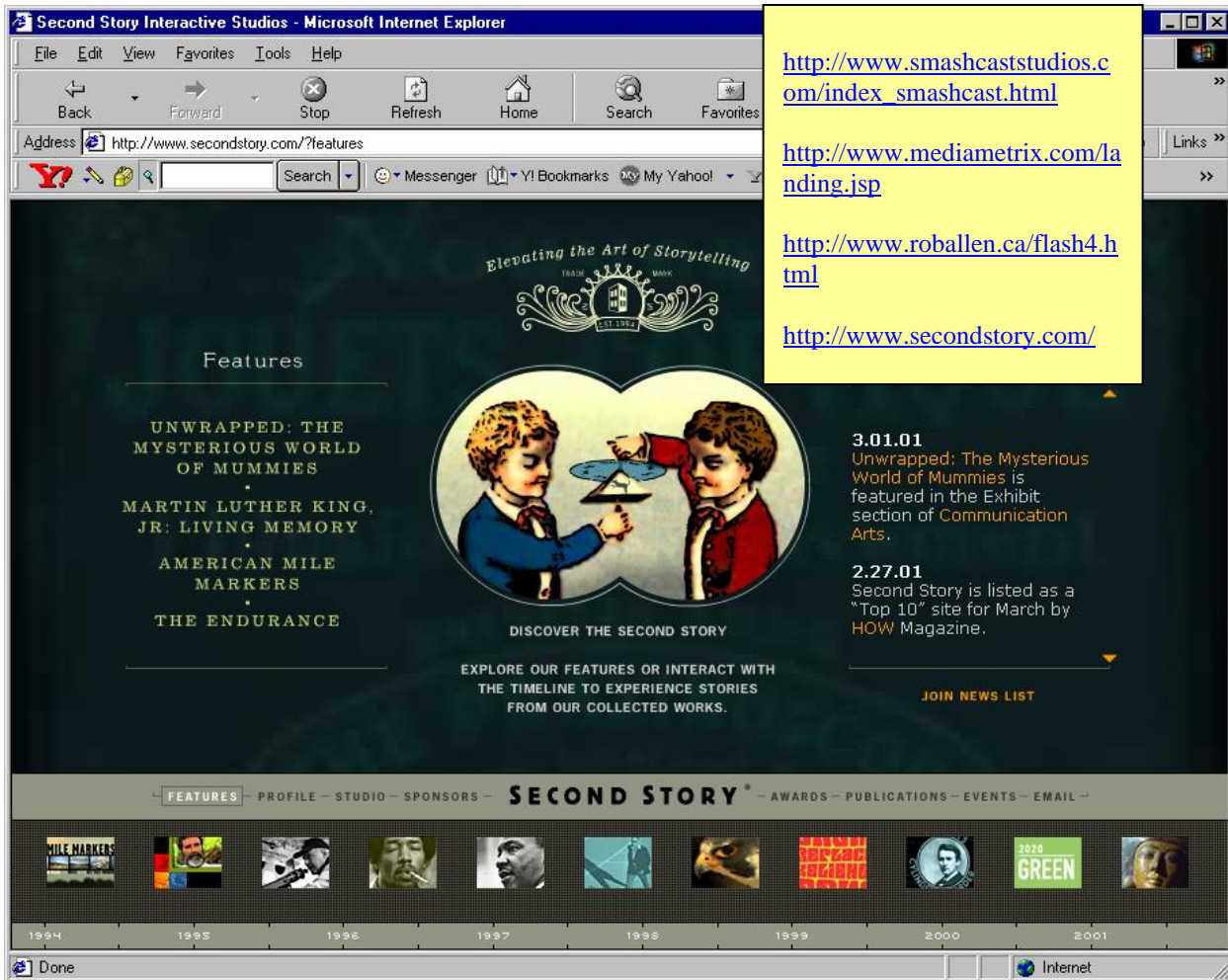


Figure 5 – Sample WWW Fireworks Page

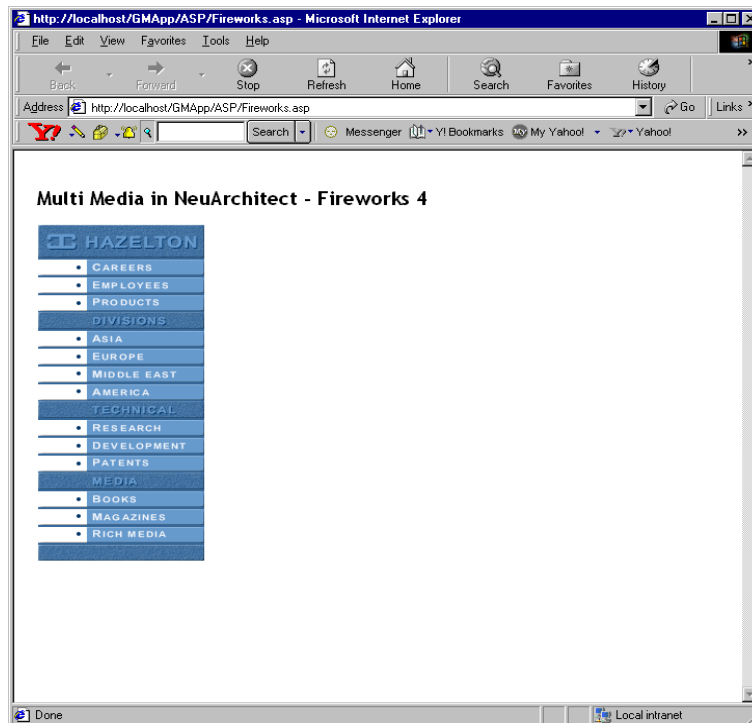




## Using Custom HTML – Fireworks 4

**Before you start:** If working online, copy the HTML from Nav Bar 2-State.htm into your Windows text buffer.

1. Open GMSite in Page Architect
2. To simplify the process for this exercise, switch you Application Technology to Microsoft JavaScript with ASP pages.
3. Create a page named Fireworks\_Page and go to Page Architect.
4. Select a Custom Control from the Control Palette and drop it in the Page Architect canvas.
5. Access the properties of the new control and select the Custom HTML tab.
6. Select the Custom button
7. Paste in the custom code from Figure 2.
8. Click OK
9. Construct HTML and Preview the page in your browser.
10. Try the top button, 'Careers'. It is hooked into one of your application pages already.



Custom HTML - Fireworks – Workshop 3

## Custom HTML – Applets

### Workshop

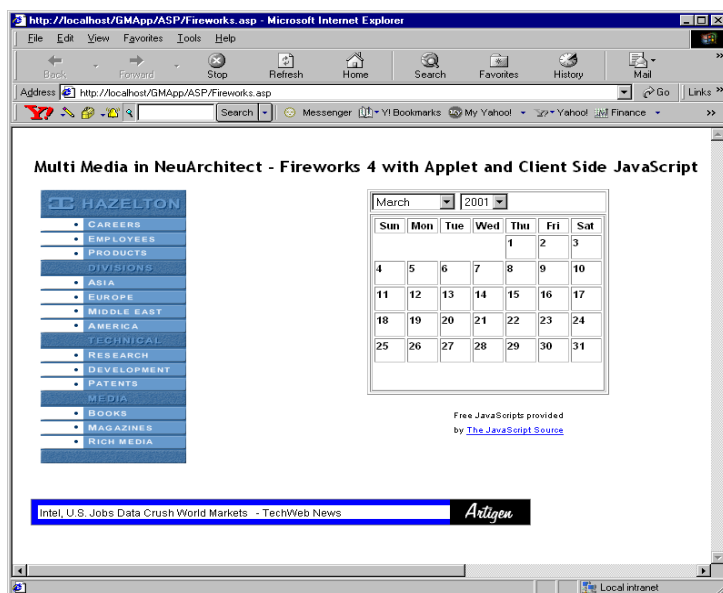


### Using Custom HTML – Applets and Client Side JavaScript

2. Go to Page Architect and open your Fireworks\_Page
3. Select a Custom Control from the Control Palette and drop it in the Page Architect canvas under your Fireworks custom control.
4. Access the properties of the new control and select the Custom HTML tab.
5. Select the Custom button.
6. Paste in the custom code shown below and Click OK

```
<APPLET CODE="INewswireTicker.class" CODEBASE="http://www.artigen.com/newswire/"
WIDTH=550 HEIGHT=33>
<PARAM NAME="channel" VALUE="infotech">
<PARAM NAME="speed" VALUE="medium">
<PARAM NAME="target" VALUE="framename">
</APPLET>
```

7. Select another Custom Control from the Control Palette and drop it in the Page Architect canvas to the right of your Fireworks custom control.
8. Paste in the custom code from the calendar.html file and click OK.
9. Construct HTML and Preview the page in your browser.



### Custom HTML – Applets and Client Side JavaScript – Workshop 4