

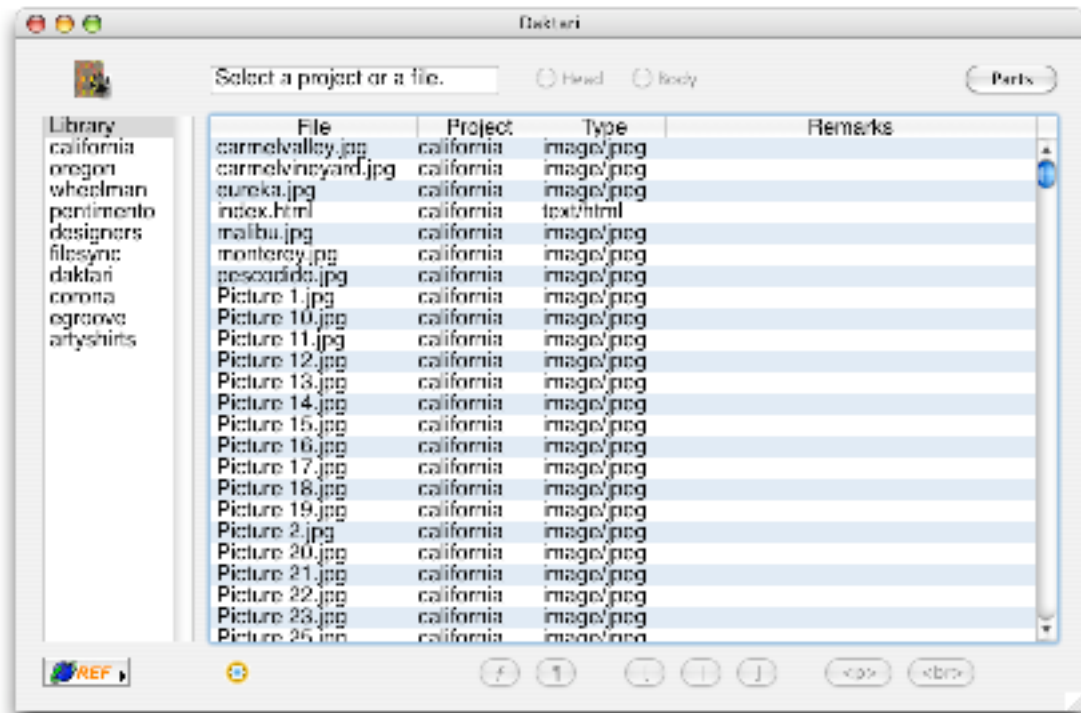
Daktari

© 2002-2006 R Charles Flickinger
All Rights Reserved

Table of Contents

Overview	2-5
Installation	3
Adding Projects	3
Editing Web Pages	4
A Simple Web Page	5-32
Style Basics	7
Format and Links	10
Images and Content	15
Forms and Scripting	18
More About the Main Window	33
The Parts Drawer	33
The Window Menu	36
Additional Edit Menu Commands	37
Limitations and Issues	38
Summary	39
Legal Statement and Disclaimer	40
Quick Reference	41-48
Daktari Main Window	41
File Menu	43
Edit Menu	44
Format Menu	45
Content Menu	46
Forms Menu	47
JavaScript Menu	48

Daktari Guide & Reference



Overview

Daktari brings a familiar metaphor to managing web site projects with unique features for editing web pages. As web site projects are added to Daktari's Library, all of the web pages and image files are listed in a Library List view. Double-clicking a web page in the Library List view opens the page into Daktari's Head and Body views for editing. While editing, the page may be saved and previewed by clicking Daktari's View icon.

In addition to organizing and quick access to the pages and images of web site projects, Daktari provides unique features for writing web pages. Daktari is designed to allow copy to be pasted or entered into a new web page and subsequently select portions of the text for markup, choosing markup commands from Daktari's menus. Drop-down panels simplify creating CSS style definitions, images, and scripts. Daktari colors the web page code to make reading and recognizing the page's parts easier.

Another unique feature is Daktari's Parts drawer. As styles, functions and elements are created, each is listed in a view of the Parts drawer. These lists allow these parts to be quickly located in the web page by double-clicking on them in the Parts drawer. The Parts drawer also may store selections of text from web pages as clips that may be reused in other web pages. The Parts drawer also displays the images of the web page's project, allowing them to be inserted by double-clicking them.

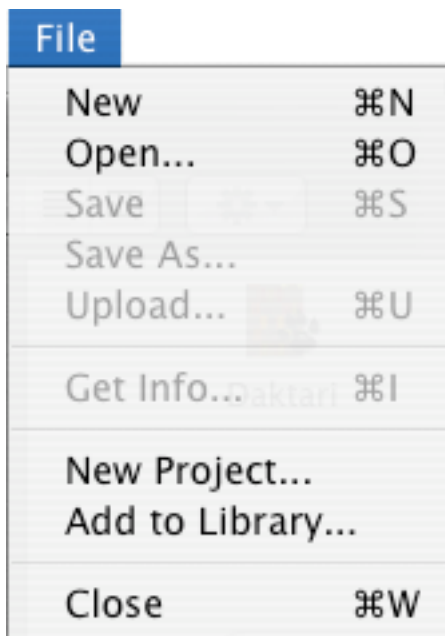
Installation

To install Daktari on the computer, drag and drop the Daktari folder to copy it to the Applications folder on the computer's hard drive. Inside the Daktari folder, are the Daktari application, a Daktari Library file and a Daktari Clips folder. Keep these files in the same folder as the Daktari application.

Open the Daktari application by double-clicking it in the Daktari folder window, or select it and choose Open... from the Finder's File menu.

Adding Projects

Begin building Daktari's Library by adding projects with the New Project... or Add to Library... commands under the File menu.



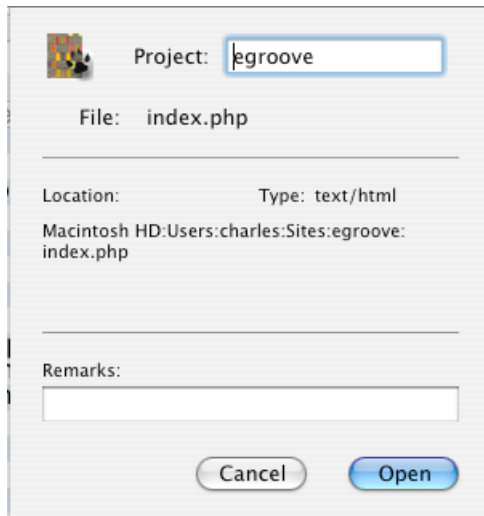
New Project... will prompt for entering a name for the new project and creating a new folder on the computer's hard drive for the new project's files (preferably in the Sites folder of the Home folder).

Add to Library... will prompt for selecting a folder on the computer's hard drive that already has images and some web pages in it. Daktari uses the folder's name for the project and reads information for each file, adding them to the Library.

Daktari lists projects in the Library view on the left side of Daktari's main window. All of the web pages and images are listed in the Library List view of the main window. Each file shows its name, the project it is related to, its type, and a space for entering remarks about the file.

Click a project in the Library view and the project's web pages are listed in a Project List view in Daktari's main window. The project's images are listed in the Images view of the Parts drawer. The Parts drawer is opened and closed by clicking the Parts button of the main window.

The Get Info... command enables when a web page in the Library List or Project List views is clicked to select it. The Get Info... command shows information about the file.



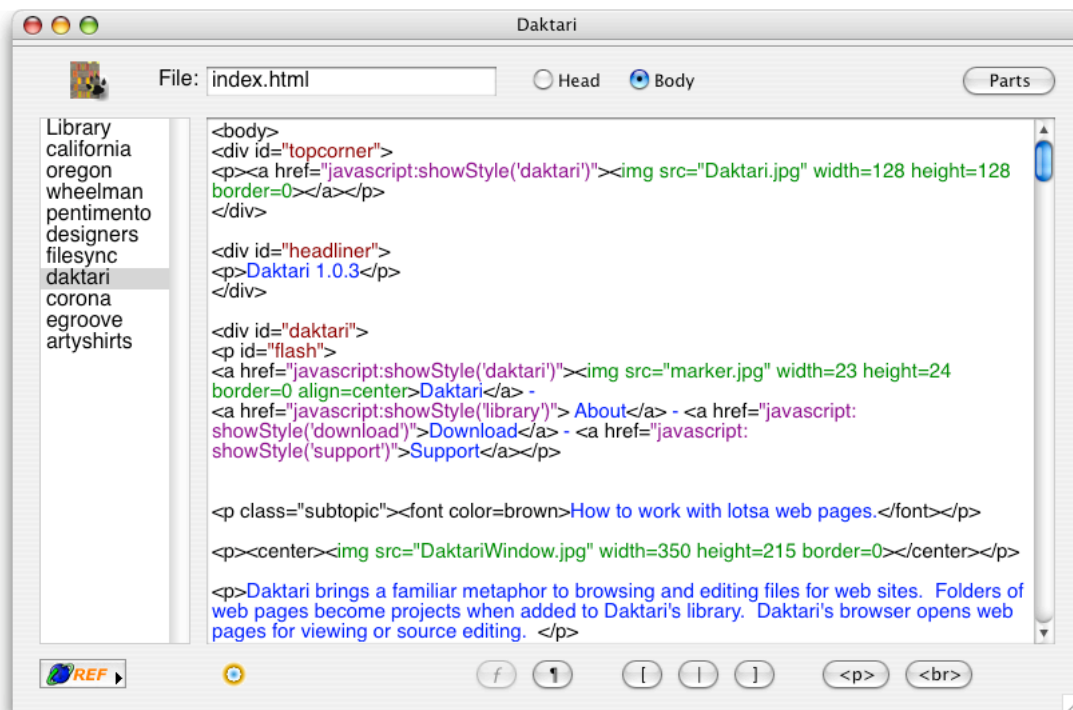
If the selected web page is not already open, the Get Info panel's Open button will open the page for editing.

If the selected web page is already currently open, the Get Info's Open button changes to an Update button. While in this state, the project's name maybe be changed, or remarks may be added for the file in the Library List.

Clicking the Update button confirms these changes to the Library List.

Editing Web Pages

When a web page is selected, it may also be opened with the Open... command under the File menu. Web pages in the Library List or Project List views may also be double-clicked and they will open for editing. Daktari colors the text of the web page as it opens into separate Head and Body views for these respective elements of a web page.



The Head and Body portions of a web page contain completely different content. The upper Head portion generally contains the web page's CSS styles and JavaScript functions. The actual elements of the page (paragraphs, forms, buttons, etc.) are laid out and defined in the lower Body portion. The Head and Body radio buttons of Daktari's main window facilitate moving between these portions of the web page.



Daktari's Edit menu has all the standard Macintosh edit commands for copy, cut, paste, undo and select all. Additionally, Daktari's Head and Body views support standard drag and drop text to and from the Finder.

Edit	
Undo	⌘Z
Cut	⌘X
Copy	⌘C
Paste	⌘V
Select All	⌘A
Clear	
Show Browser	⌘B
Find...	⌘F
Find Again	⌘G

Within the view, selected text may be moved by dragging it to a new location in the text; holding the option-key while dragging a text selection will move a duplicate copy to another location in the text.

Drag and drop text is also useful for selecting and saving clips, or the names of styles, functions and elements to the Parts drawer. Conversely, clips and styles may be dragged from the Parts drawer and dropped into the web page's text. The Parts drawer section of this guide discusses these features more thoroughly.

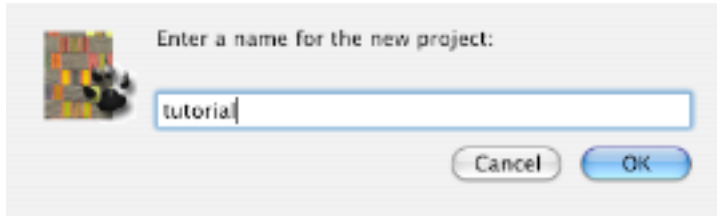
The Edit menu also has standard Find... and Find Again commands.

A Simple Web Page

Building a simple web page provides a good way to describe how to use Daktari's menu commands for editing web pages. While this may seem as a good introduction for persons first learning how to write web pages, more advanced persons will appreciate the features described that make Daktari a useful application. In addition to a web browser, for viewing the page as it is built, a graphics application will be useful for creating graphics that will be used in the web page.

Begin by locating some text, about four or five paragraphs worth, from any document readily available. Four or five paragraphs of text from this guide will work, for example. For fun, drag and drop the text to the Finder desktop to create a text clipping; later we'll drag and drop the clipping into our page's Body view.

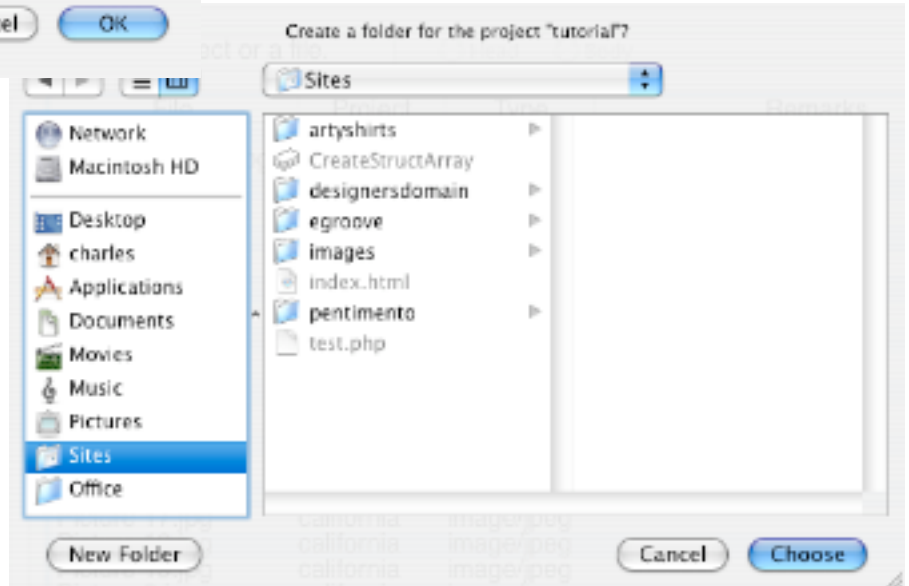
Open Daktari and choose New Project... under the File menu. give the project a name, tutorial will do fine. Daktari will drop down a prompt to create a new folder for the project. Navigate to the Sites folder and create the folder there.



When the panel's Choose button is clicked, a

new folder is created. In Daktari's main window, the project is selected in the Library view, and the empty Project List view is displayed.

Next, choose New... under the File menu to open a new untitled web page. The page's Head view first appears with initial Head and Title tags.



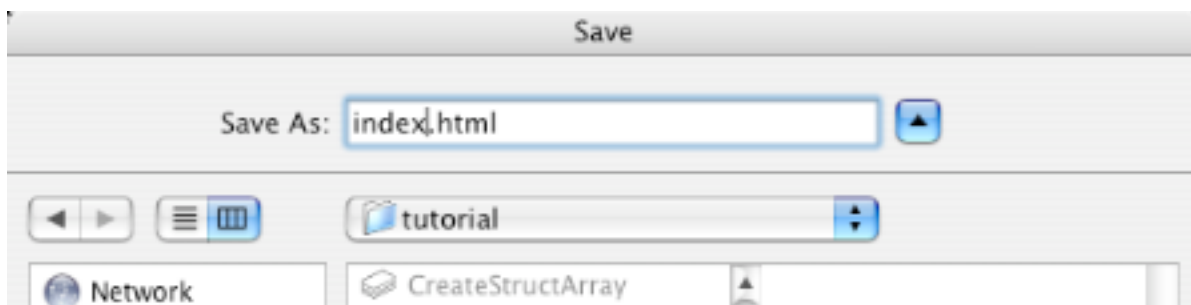
Click between the Title tags

```
<html>
<head>
<title>My Simple Web Page</title>
</head>
```

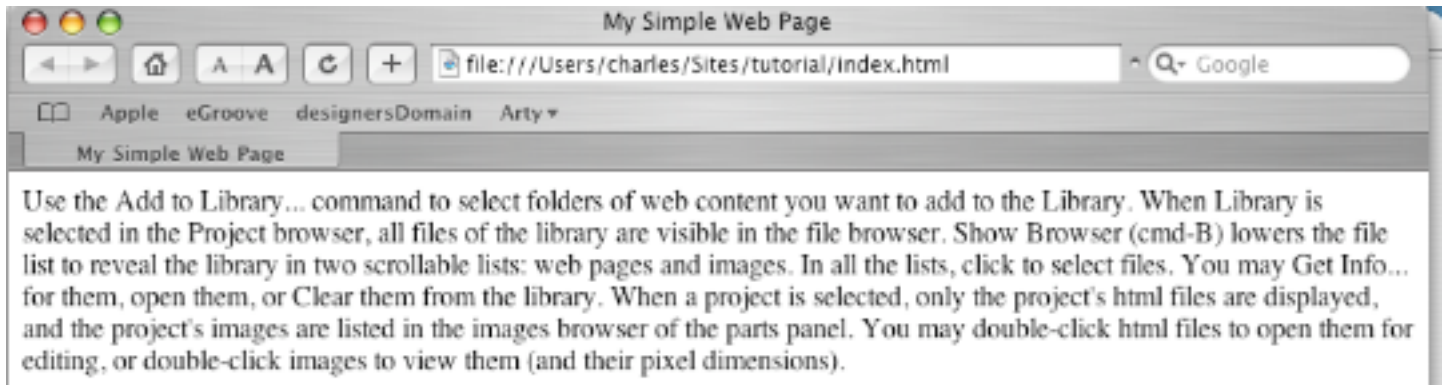
and enter a title for the web page. This title appears in the titlebar of the web page when it is viewed with the web browser.

Now click the Body radio button to switch to the web page's Body view. Body tags and the ending HTML tag are present. Drag the text clipping saved earlier

the desktop and drop it in front of the ending Body tag in the Body view. This places the text between the Body tags. Next, click the Daktari icon in the upper left of the main window to view the page in the web browser. Daktari saves the page before loading it into the web browser; new documents display the standard Save panel. Name the file index.html and save it in the Tutorial project folder that was created in the Sites folder.



After the page is saved, the Get Info panel appears confirming that the file is related to the tutorial project. Click the Update button and it will load into the web browser. Notice that while the text of the clipping may have had perfectly formed paragraphs, the web browser does not display them this way. Later, we will use Daktari's paragraph command to from these paragraphs so the web browser will display them properly.

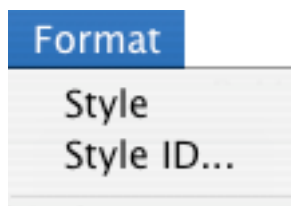


Style Basics

More often, web pages are using Cascading Style Sheets (CSS) to format and style web pages. Styles are handy because they may be copied and reused in other web pages to give them a uniform, standardized look. We will see how Daktari helps create these style definitions easily.

Many web pages on the world wide web have an area across the top of the page called a mast-head. Often, web pages present links to other web pages in a sidebar, usually on the left side of the page. The remainder of the page often contains the main content of the page. We will create these three styles for our simple web page using Daktari's style definition panel.

Styles are defined within the Head portion of the web page between Style tags. Click the Head radio button to switch to the web page's Head view. Insert a new line after the ending Title tag (press return on the keyboard) and choose Style under the Format menu. A pair of Style tags are inserted into the web page.



```
<html>
<head>
<title>My Simple Web Page</title>
<style>

</style>
```

We'll define the Style for the masthead first. Click on the empty line between the Style tags and choose Style ID... from the Format menu. The Style definition panel will display.

Along the top of the panel is where the name of the style is entered. Type masthead into the style name field. We want this style to be visible, so leave the visibility checkbox hilited.

The next portion tells the web browser how to position the style. Web browsers use screen pixels to position styles, the top left of the page being 0,0. We'll place our masthead ten pixels from the top corner of the page; set the top and left values to 10, width to 640 and the height to 128 pixels.

Because our masthead will be a graphic im-

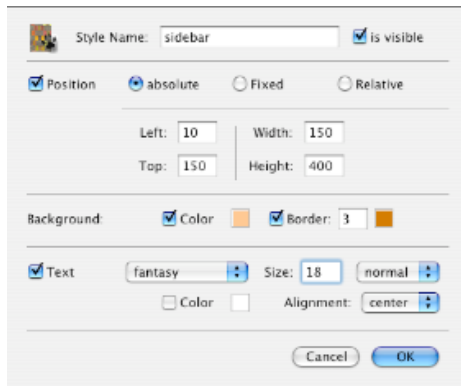
age, we'll leave the Background section alone. Our masthead also will have no text, so click the Text checkbox so it is not hilited. Click OK to close the panel and insert the masthead style into the web page.

```
<html>
<head>
<title>My Simple Web Page</title>
<style>
#masthead {position:absolute; left:10; top:10; width:640; height:128; visibility:visible}
</style>
</head>
```

The Style definition panel eliminates a lot of typing. It places the “#” in front of the style's name, and all of the properties that define the style are between braces.

Insert a new line after the ending brace of the masthead style. Note it is important that all the style definition are kept within the Style tags. Choose Style ID... from the Format menu to create a style definition for our sidebar. Enter sidebar for the name and set the left to 10, the top to 150, and the width to 150.

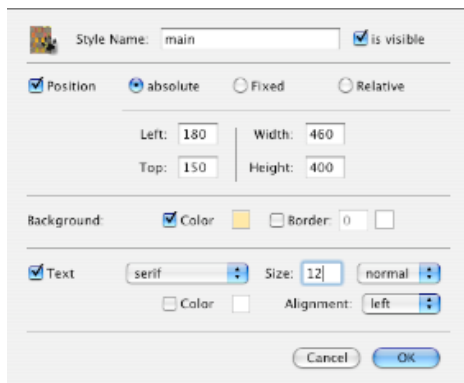
In the background section, hilitate the Color checkbox and click the white box next to it to select a light background color for the style. To add a stronger border color to our sidebar, click the border checkbox, enter 3 for a border width of three pixels, and click the white box to choose a darker color for the border.



In the text section of the panel, choose “fantasy” from the popup, set the size to 18, and the alignment to center. The style definition for the sidebar is now complete, click OK to insert it into the web page.

Press enter on the keyboard to insert a new line after the sidebar style definition and choose Style ID... under the Format menu to create the main style.

After entering the style’s name, we set the main style’s positioning. To have 30 pixels of space between this style and the sidebar style, enter 180 for the left. Set the top to 150, same as the top of the sidebar style. We don’t want this style to go beyond the width of our masthead, so enter 460 for the width (we subtract 180 from the width of our masthead style).



For the background of our main style, hilite the color checkbox and choose a light color that complements the sidebar’s background color. Leave the border color off, our main style will not use a border. In the text section, change the size to 12.

The style definition for our main style is complete, click OK to add it to our style definitions in the web page.

We have seen how Daktari’s style definition panel makes adding style definitions to web pages much simpler to typing them from scratch. There is more about CSS that goes beyond the purpose of our simple web page. Some of these additional features are discussed later in this guide.

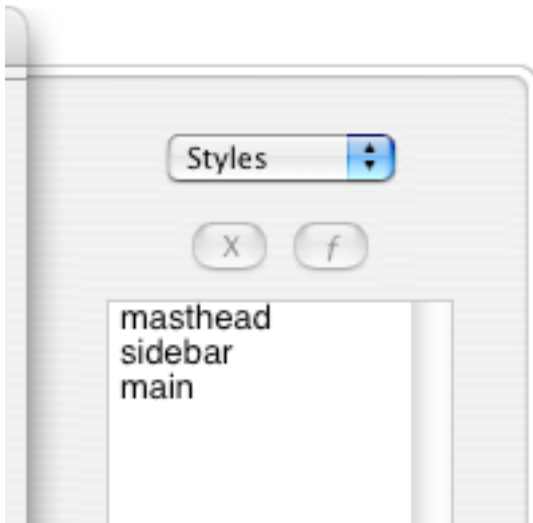
We are now ready to go back to the body view of our web page to see how to use these styles. Choose the Save command under the File menu to save the changes we’ve made to our simple web page. To bring the Body portion back into view, click the Body radio button.

Format and Links

One might say the Head portion is the “brains” of a web page, and the Body portion is its “muscle” (and possibly some vital “organs”). The Body portion is where text and elements for forms, buttons and fields are written. We have seen earlier that the web browser has no idea how to display these elements, or rather, the web browser does not render our text the same way as we typed it. We have to tell our web page, using “language” the web browser understands, how to display the elements we put in our web page.

In the previous section, we defined styles for three major portions of our simple web page. In this section we will insert references to these styles and place text within them. One thing Daktari does when creating styles (and also functions and input elements) is it adds their names to lists in the Parts drawer. Instead of trying to remember and typing out style names, we can drag and drop our styles into the Body portion of our web page.

With the Body portion in view, click the Parts button on Daktari’s main window to open the Parts drawer. What first appears in the drawer is the Clips list. We will learn about Clips later in this guide. At the top of the Parts drawer is a popup control for selecting to view each of the five lists in the Parts drawer. One of these lists is Images. It is currently empty for our tutorial project because we have not added any images to our project as yet.



For now, choose the Styles list. When it is displayed, the names of the three styles we defined in the previous section will be visible in the list.

Insert a new line in the Body view, after the beginning Body tag. Then, drag and drop the masthead style name from the Parts drawer to this new line. Daktari inserts what needs to be typed to define this style in the Body for the web browser.

```
<body>
<div id="masthead">

</div>
<div id="sidebar">

</div>
<div id="main">

</div>
Use the...
```

Insert another new line, after the ending </div> tag and drag and drop the sidebar style name from the Parts drawer to this new line. Do the same for the main style name. All three should appear just above the text we brought into the web page earlier.

In essence, what we’ve created here are three “boxes,” or layers, in which to put text and other elements, which will be displayed by the positioning values we defined in their respective style definitions. The first thing we’ll do is fill the main layer with our body text below. Select all of it, drag and drop it into the line in between the main

tags.

Now, the web browser will know to display the text within the area defined by the main style definition. But the browser will still render the text as we saw earlier until we markup each paragraph with paragraph tags. To do this, select the first paragraph of the text in the main layer and choose

Paragraph under the Format menu. The text will now be surrounded by starting and ending paragraph tags (<p>the first paragraph</p>). Do the same for each of the paragraphs in the main layer.

After each paragraph has been surrounded with paragraph tags, click the Daktari icon in the upper left of the main window to save, colorize



and view the page in the web browser. Notice that the paragraphs are now recognized by the web browser, instead of one giant glob of words. Also, the colors of the sidebar and main layers are visible; the masthead layer has nothing in it yet, so it does not appear visible. Notice the words in the main layer line up all the way to the very edges. We will go back to our style definitions in a moment to fix this, but first, we'll add four or five words to our sidebar layer.

Earlier, we noted that sidebars are generally used to contain navigational elements, so we want to enter each word on its own line and tell the browser to do the same. Within the sidebar layer tags, type each word, and after each, choose the Break command under the Format menu before pressing return to type the next word. After one of the words, try the Rule command, instead.

```
<div id="sidebar">
Breakfast<br>
Lunch<br>
Dinner<hr>
Desserts<br>
Snacks
</div>
```

Before we view the web page again, let's fix the problem of the words rendering to the edges of the main layer. Click the Head radio button to bring the web page's Head portion back into view.

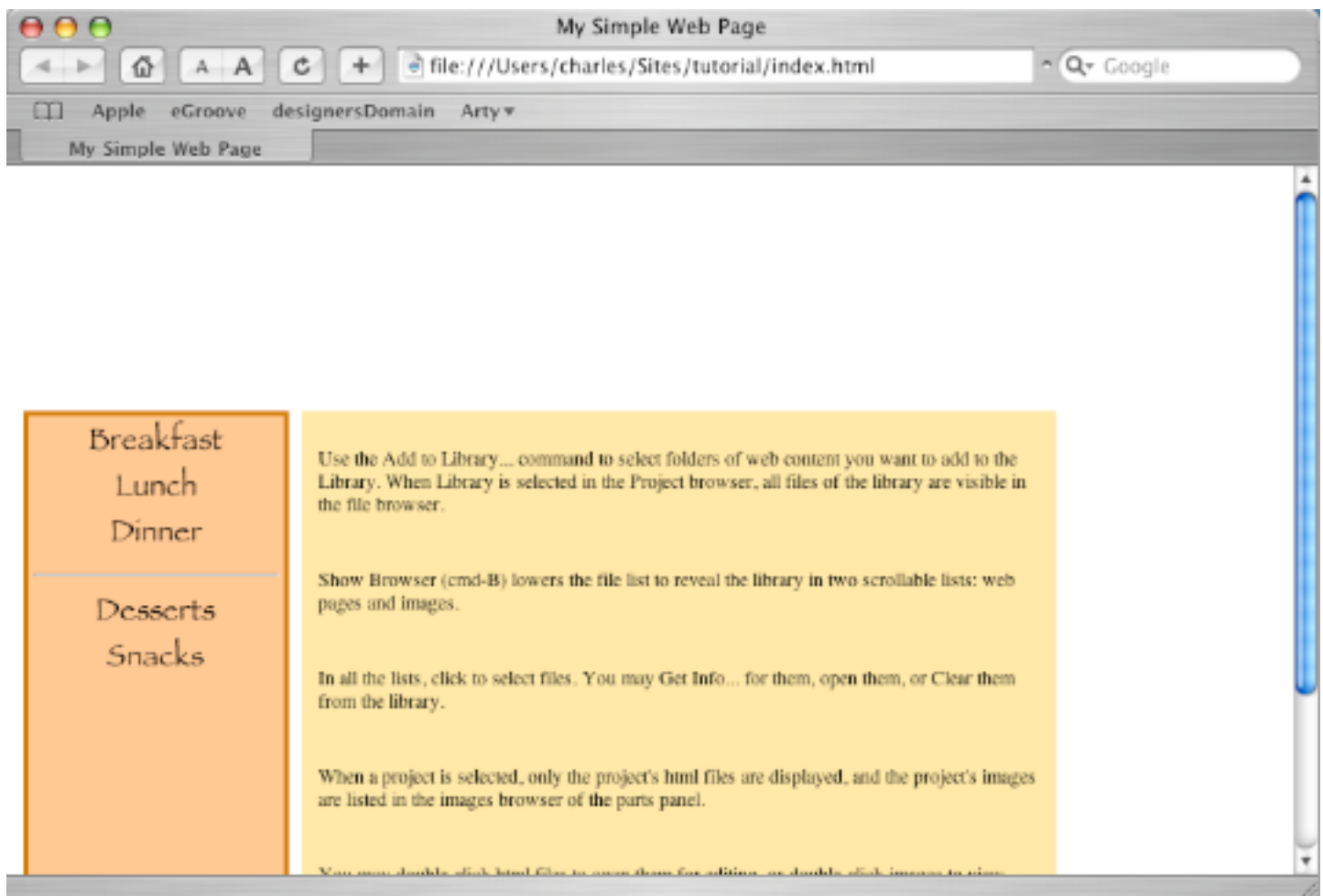
CSS is not limited to defining styles for layers. Styles may also be defined for standard HTML elements, such as paragraphs. We will enter a short style definition for specifically for paragraphs that puts space between them and any layers they are within.

```
<head>
<title>My Simple Web Page</title>
<style>
p {padding:10}
#masthead {position:absolute; ...
```

Make a new line after the beginning style tag and type p {padding:10} being sure to use braces and a colon.

Click the Daktari icon in the upper left of the main window to save, colorize and view the web page in the web browser to

see the results of our changes.



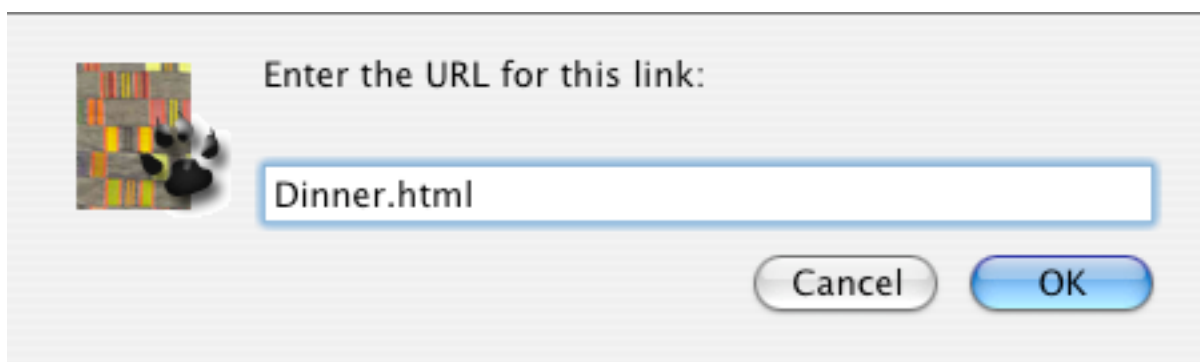
So far, we have done a lot for our simple web page without a lot of typing or remembering actual HTML tags or style properties; mostly entering names or selecting text and applying menu commands. Daktari works the same way when creating links to other web pages. Click the Body radio button to bring our web page's Body portion back into view.

We will change the words in our sidebar layer and make links to them. Double-click the first word and change it to "MacCentral." Then, double-click on it again and choose the Link... command under the Format menu. Enter `http://www.maccentral.com` into the Link panel that drops down and click OK to confirm it. Daktari surrounds the text with the Link tags and includes the URL you entered for the href property.



Change the second word of our sidebar to "Daktari" and create a Link to Daktari's web site the same way as above, selecting it before choosing the Link... command from the Format menu. The URL to Daktari's web site is `http://www.designersdomain.com/daktari`.

The two URLs above are complete URLs, called absolute references. Links to web pages that are in the same folder as the web page only require the file name of the web page; these are called relative references. It is easiest to work with relative references because if the site should ever require relocation, the references will not become broken and they'll continue to work. This saves a lot of time not having to retype them to get them working again. For an example of a relative reference link, double-click the third word in our sidebar and copy it. Then, choose the Link command and paste it into the Link panel, typing ".html" after it.



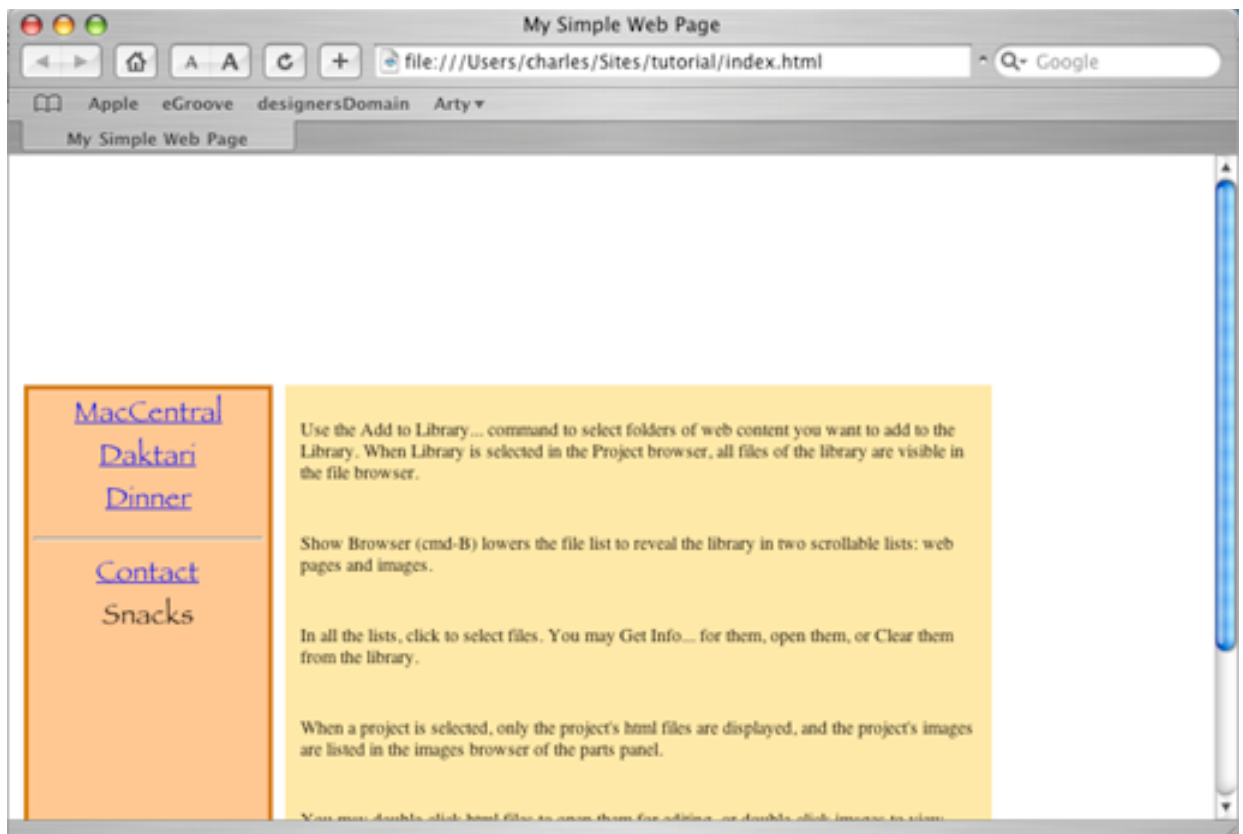
The Link... command is also useful for entering references to JavaScript functions and URLs to email addresses. In the Link command's drop down panel, just enter the appropriate syntax:

for a JavaScript function: javascript:functionName()
for an email address: mailto:recipientName@domain.com

We will see later how the link command will execute JavaScript functions entered in the Head portion of a web page. The mailto URL for email opens a new message in the end-users web browser. Use the Link command on the fourth word of our sidebar to try it out.

Use the Daktari icon to save, colorize and view our web page in the web browser. Try the newly created links to see how well they work. Before continuing on to the next section about images, create and save two graphics JPEG files. One, for the masthead, should be 640 x 128 pixels, and saved to the Finder desktop. The other, for the sidebar, should be 150 pixels wide, and saved inside our tutorial folder in the Sites folder.

```
<div id="sidebar">  
<a href="http://www.maccentral.com">MacCentral</a><br>  
<a href="http://www.designersdomain.com/daktari">Daktari</a><br>  
<a href="Dinner.html">Dinner</a><hr>  
<a href="mailto:recipientName@domain.com">Contact</a><br>  
Snacks  
</div>
```



Images and Content

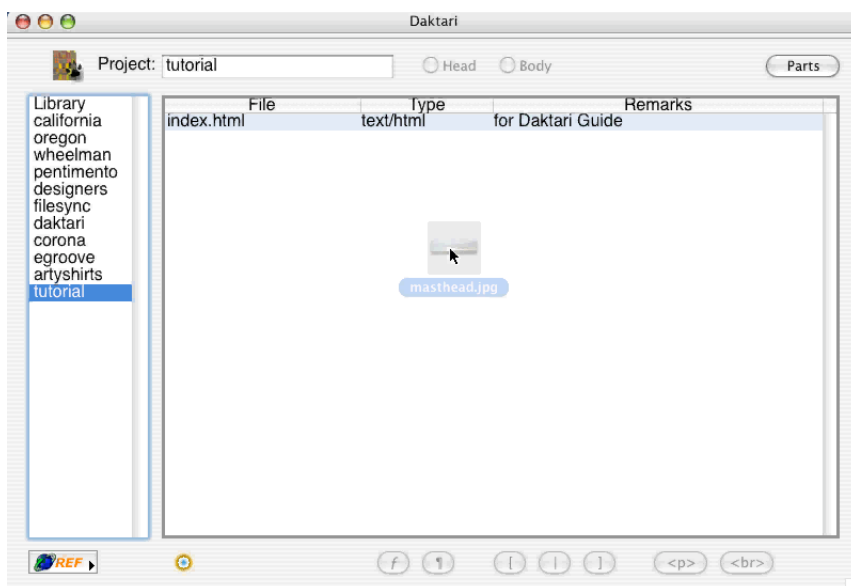
Images and graphics have been a vital part of web pages since the World Wide Web's beginnings. Images are not only the larger photographs you see on web pages, but often the smaller icons and buttons that beg you to click on them. This section primarily discusses images, and using Daktari's image panel to place them in web pages. There are many file formats for images, Daktari recognizes JPEG, PNG and GIF format.

Ideally, many of the initial images and graphics for a web page may be created and placed early on into a project folder. Using the Add to Library... command will add the project and all of the images in the folder to Daktari's Library, ready to use with the project's web pages.

The images for our simple web page, however, have not been added to the Daktari library as yet. We've saved the graphic for the masthead on the Finder desktop and the graphic for the sidebar is already in the same folder as our simple web page. The problem is no different than adding a new song to Apple's iTunes. If a song is dragged into the iTunes music folder, iTunes has no idea the new song is there. If iTunes is used to open and play the song, it may be added to the music library, but the play lists don't know about it.

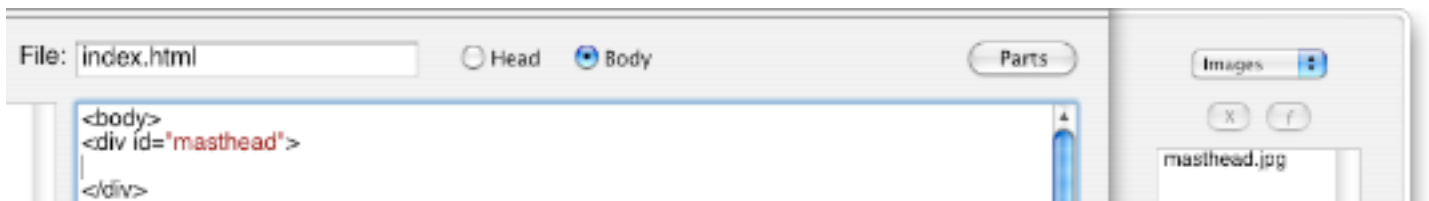
In a nutshell, web pages have to tell web browsers where the images are located. Daktari uses relative file references by default, so when an image is not in the same folder as the web page being edited, Daktari will ask whether to move the file to the project's folder. Daktari tries to help address these issues as we will now see as we begin to work with images.

New files for an existing project may be dragged and dropped singly to a project's list in the Daktari main window to add them to the Daktari Library. Click on our tutorial project in the Library list, then drag and drop the masthead graphics file from the desktop to the tutorial's Project List in the main window. A panel will drop down to asking to confirm the file will be added to the tutorial project; click OK.

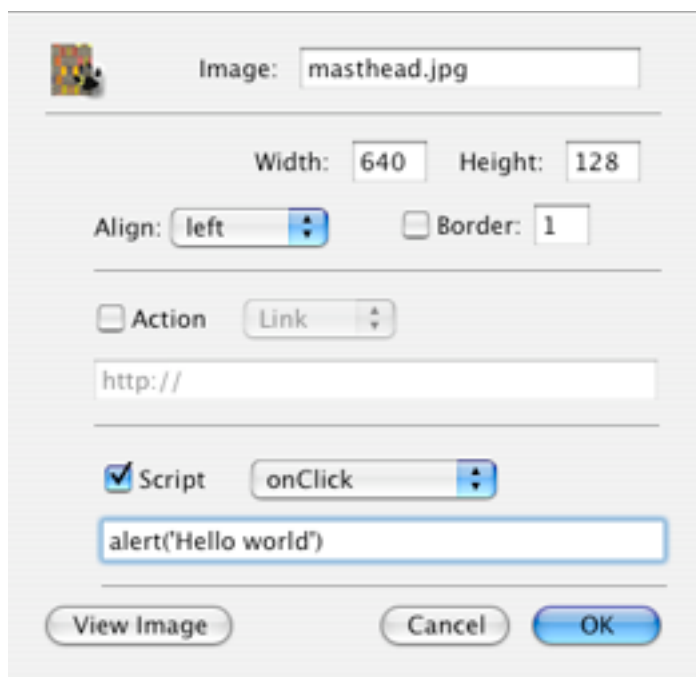


Now, double-click our web page in the Project List to open it, and click the Parts drawer to open it. Choose the Images list from the Parts drawer's popup.

We'll now place our masthead image into the masthead layer. Click the Body radio button on the main window to bring the Body portion of the web page into view, and click between the masthead layer tags. Double-click on the masthead.jpg file in the Parts drawer.



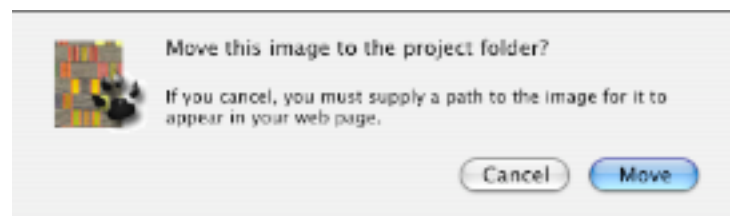
The image definition panel drops down for setting the image's properties. The image's size is displayed and should already be 640 x 128 pixels. The width and height values are editable, and when either of them are changed, Daktari recalculates the other to maintain the image's aspect ratio. The Align popup is used to indicate where any nearby text should appear. For now, click the Border checkbox and set it to 1, which will put a thin black line around the image. We'll also enter a simple script; hilite the Script checkbox and type alert('Hello World'). Click OK to insert the image definition into our web page.



Because the masthead.jpg file is not in the same folder as our web page, Daktari drops down another panel asking whether to move the file. Click Move to confirm and move the file.

The image definition is now inserted into the masthead layer. Also, the masthead.jpg file is now located in the same folder as our web page

so web browsers will find it when the page is viewed. Click Daktari icon on the main window to save, colorize and view these changes in the web browser.



```
<div id="masthead">

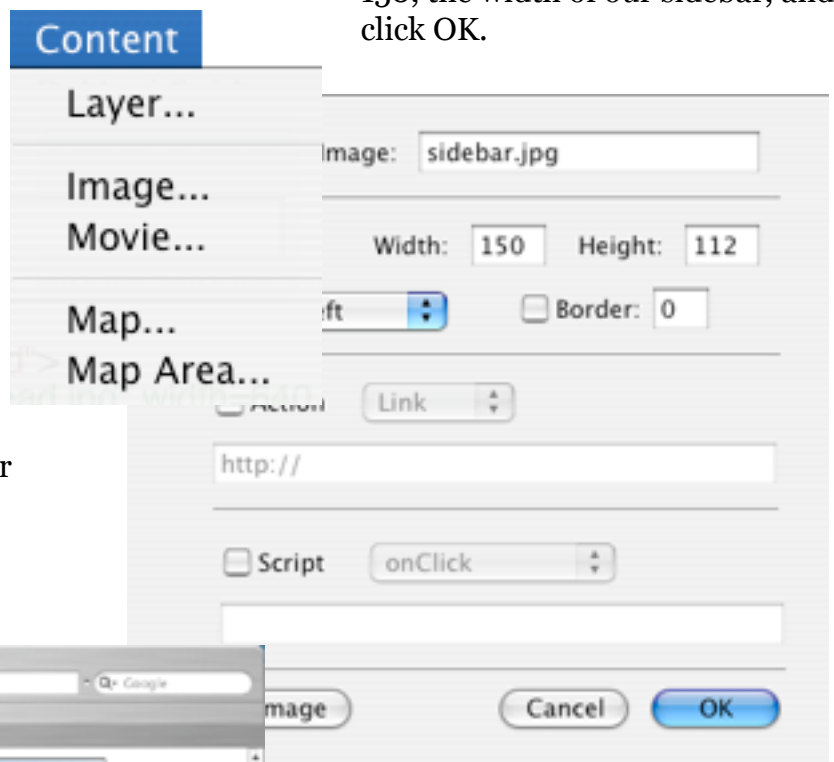
</div>
```

Next, we'll add the image for our sidebar. It's not listed in the Parts drawer, so we can't double-click on it to drop down the image properties panel. Instead, we'll use the Image... command under the Content menu. Before choosing the command, click and enter a new line after the beginning sidebar layer tag.

```
<div id="sidebar">
<a href="http://www.maccentral.com">MacCentral</a><br>
<a href="http://www.designersdomain.com/daktari">Daktari</a><br>
<a href="Dinner.html">Dinner</a><hr>
<a href="mailto:recipientName@domain.com">Contact</a><br>
Snacks
</div>
```

The Image... command drops down a standard open panel for choosing files. Navigate to the sidebar.jpg file in our tutorial project folder in the Sites folder. Select it and click the panel's Open button. Set the width to 150, the width of our sidebar, and click OK.

Because the sidebar.jpg is already in the same folder as our simple web page, Daktari notices this and adds the image to our tutorial project list in the library. The images list for our tutorial project in the Parts drawer is also updated.



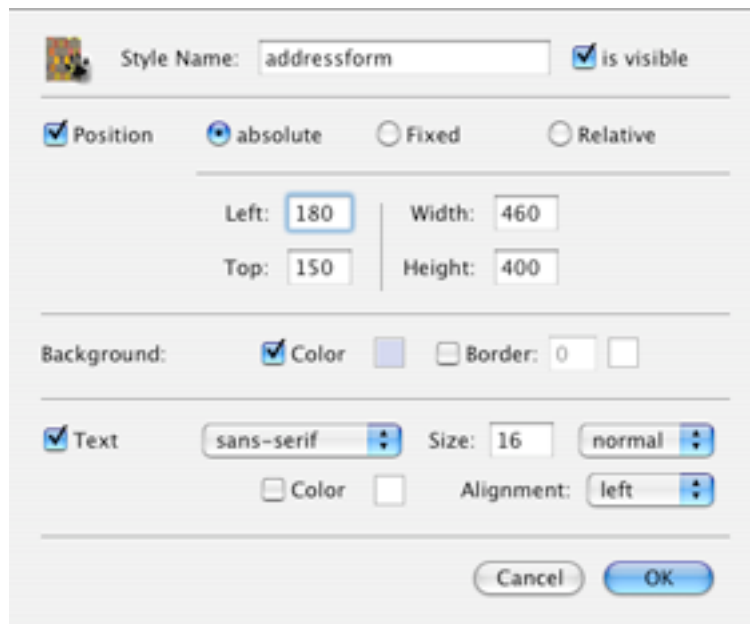
Click Daktari's view icon to save our changes and view the web page in the web browser.



The Movie... command works like the Image.. command, but without a properties dialog. It inserts a default set of movie properties that you can change. The controller property sets whether the movie's controller is visible, autostart sets whether the movie plays automatically, and kioskmode protects the movie from being downloaded.

Forms and Scripting

So far, the interactivity of our simple web page is limited to the links we made in our sidebar. Forms provide the means to go beyond offering just simple hypertext links. A form consists of a set of Form tags, with all of the forms fields, buttons and selectors arranged between the tags. Many form elements use a scripting language, such as JavaScript or PHP, to give the form its ability to respond to user input. In this section we will create an example form and use both these scripting languages. Note that PHP requires uploading our simple web page to your domain server. Most domain providers that use the Apache web server have PHP enabled so that web pages will use PHP.



We'll begin by making a new layer for our form. We could easily create our form in the main layer, but we will use JavaScript to show and hide our form. Daktari provides two clips that are JavaScript functions that help us do this. Begin by clicking the Head radio button to bring the Head portion of our web page into view. Click after the sidebar layer's definition and press Return on the keyboard to enter a new line, then choose Style ID... from the Format menu. Name the new layer "addressform" and set the the left to 180, top to 150, and width to 460. Choose a light color for the background and set the text to "sans-serif". Click OK to insert the addressform layer's definition into the web page, then click the Body radio button to bring the Body portion into view.

```
#sidebar {position:absolute; left:10; top:150; width:150; height:400; background:rgb(255,189,129);
border:solid 3 rgb(200,105,8); padding:3; font: normal 18 fantasy; text-align:center; visibility:visible}
#addressform {position:absolute; left:180; top:150; width:460; height:400;
background:rgb(188,189,231); font: normal 16 sans-serif; text-align:left; visibility:visible}
#main {position:absolute; left:180; top:150; width:460; height:400; background:rgb(255,229,152);
font: normal 12 serif; text-align:left; visibility:visible}
```

Snacks

```
</div>
<div id="addressform">

</div>
<div id="main">
```

Click the Parts button and choose the Styles list from the drawer's popup; our new addressform layer appears in the list. In the body of our web page, enter a new line after the sidebar layer, then drag and drop addressform from the Parts drawer in between the sidebar and main layers.

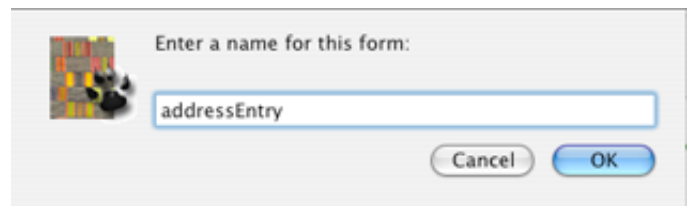
Now we'll begin building our form. Click between the start and end tags of the addressform layer and choose Form... from the Forms menu. A panel will drop down requesting a name for the form; enter "addressEntry" and click OK. A set of form tags are entered into the addressform layer.

Some initial properties are included within the form's start tag. First is the name of the form, which may be referenced by scripts. Next are method, action and enctype properties that the domain server will use to handle the information in the form. The last one, onSubmit, identifies a JavaScript function we will write that will validate the form before submitting it to the server.

```
<div id="addressform">
<form name="addressEntry" method="post" action="serverURL" enctype="multipart/form-data" on-
Submit="return submitClicked(this)">
```

```
</form>
```

Often, it can be a challenge to create a form that looks professional. Using a table can help simplify the layout of the form's elements. Click in the blank line between the form's start and end tags and choose the Table... command from the Format menu.



Our addressform layer's width is 460, and we want approximately a 50 pixel margin on each side, so set the table's width to 360. Set the number of columns to 1 and the number of rows to 3. Set the column alignment to "right" because the second and third rows of our table will be right-justified. Click OK to enter the table into our addressEntry form.

```
<div id="addressform">
<form name="addressEntry" method="post" action="serverURL" enctype="multipart/form-data" on-
Submit="return addressEntryClicked(this)">
```

```
<table width=360 align=center>
<tr valign=top>
<td width=100% align=right></td>
</tr>
<tr valign=top>
<td width=100% align=right></td>
</tr>
<tr valign=top>
<td width=100% align=right></td>
</tr>
</table>
```

```
</form>
```

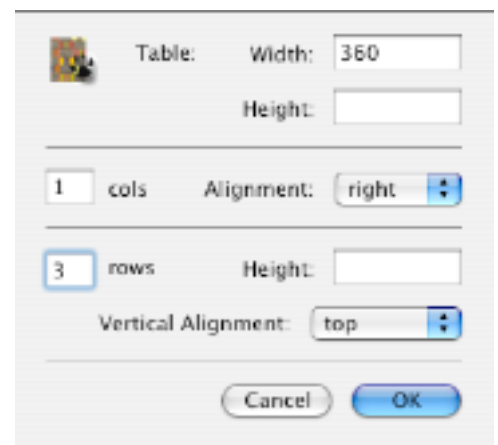
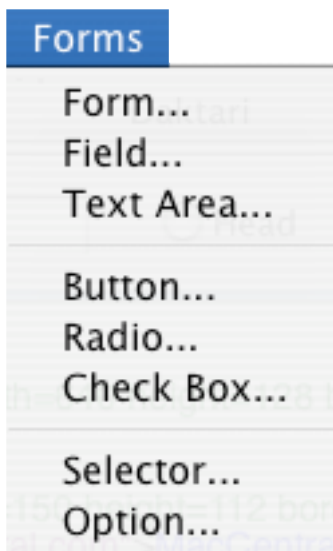


Table code can be a bit daunting to look at because it doesn't look like a table at all. It takes some practice. Basically, each table cell is defined by `<td>` tags. Our table only has one column, so there is only one set of `<td>` tags between the `<tr>` tags that define the rows. The table is contained within `<table>` tags.

Click between the first set of `<td>` tags and type our forms title, "Address Entry". Within this table cell's start tag, change the align property to "left"; this will left-justify our table's title in the first row.

We'll place our fields and their labels in the second row. All of our field labels and elements will be between the second table cell's `<td>` tags. There's three steps for each field; enter a label for the field, define the field, and insert an optional `
` break tag.



Click between the second row's `<td>` tags and enter "Name:" and a space. Immediately choose Field... from the Forms menu to drop down the field definition panel. Enter "person" for the field's name, and "35" for the field's size, then click OK. The field's code is inserted into the table cell. Next, choose the Break command from the Format menu; the break tag tells the web browser to render the next field on a new line.

Press return on the keyboard to enter a new line. Keeping within this table cell's tags, the `</td>` end tag will be on the new line. Enter "Address:" and a space. Choose Field... again from the Format menu. Enter "address" for the field's name, and set the size to "35", then click OK. Choose the Break command from the Format menu to insert a break tag.

```
<table width=360 align=center>
<tr valign=top>
<td width=100% align=left>Address Entry</td>
</tr>
<tr valign=top>
<td width=100% align=right>Name: <input type="text" name="person" size=35><br>
Address: <input type="text" name="address" size=35><br>
City: <input type="text" name="city" size=35><br>
Phone: <input type="text" name="phone" size=35></td>
</tr>
<tr valign=top>
<td width=100% align=right><input type="button" value="submit" onClick="submitClicked()"></td>
</tr>
```

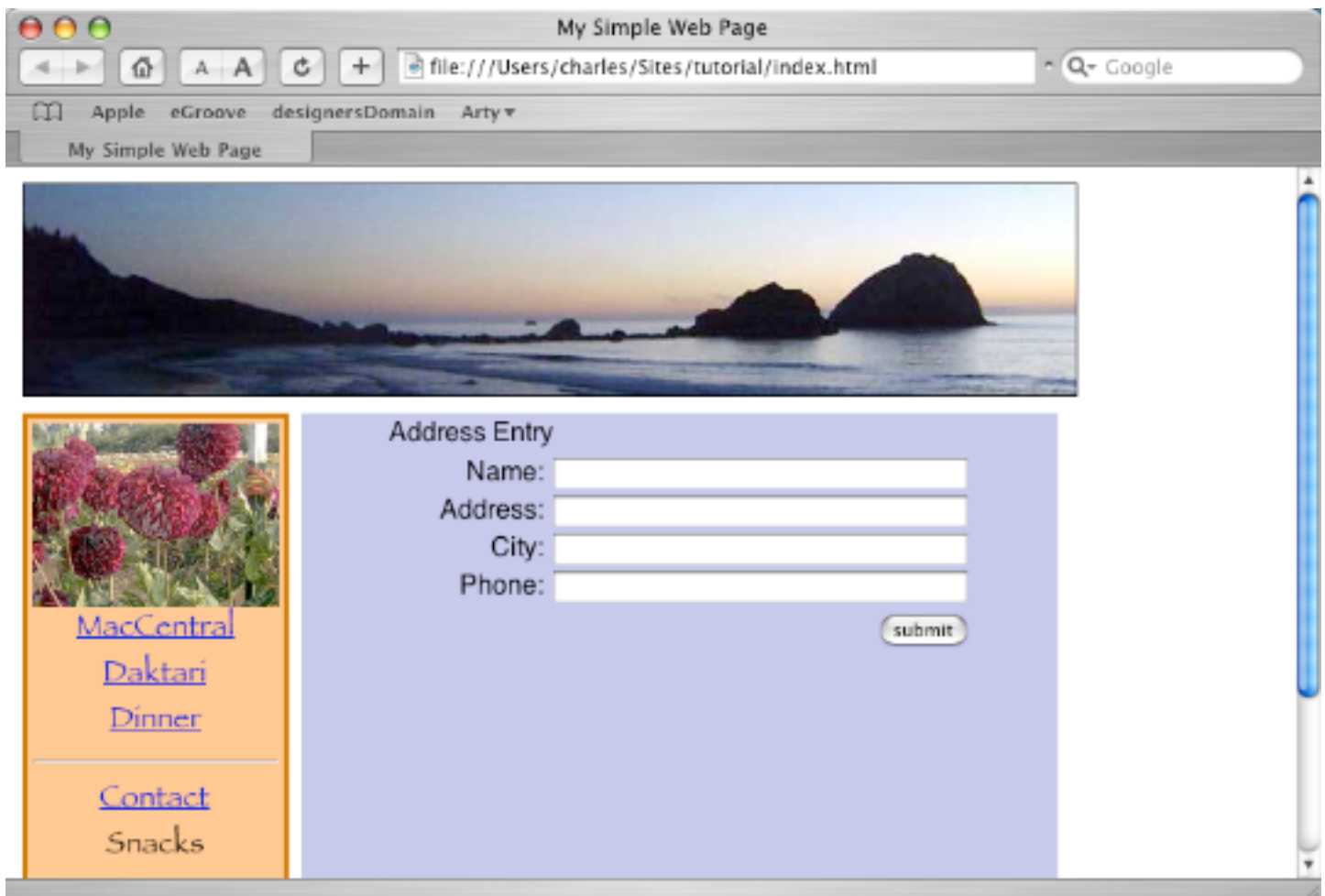
Repeat the steps two more times, creating fields for "city" and "phone". The fields' sizes should all be the same, set to "35". Because it is the last field, a break tag is not necessary after the phone field.

In the third row's table cell tags, we will have our submit button here. Choose Button... from the Forms menu to drop down a button properties panel. Enter "submit" for the button's name, select the Submit radio button, and click OK. The button's code should insert between the third row's table cell tags.

Before clicking Daktari's view icon to save and view our changes, click the Head radio button to bring the headportion into view. In the main layer's style definition, change the visible property to "hidden". Because the position properties of the addressform and main layers are the same, this will hide the main layer so they do not appear on top of one another.

```
#main {position:absolute; left:180; top:150; width:460; height:400; background:rgb(255,229,152); font:normal 12 serif; text-align:left; visibility:hidden}
```

After setting the main layer to be hidden, click the Daktari view icon to save, colorize and view our changes. You can enter text in the form's fields, but we're not finished as yet, so don't click the Submit button.



To get our form to work, we must enter some JavaScript code in the Head portion of our web page. We also want to write a set of functions that will display our page’s main layer or addressform layer when we want to view them.

We’ll first create two buttons in our sidebar layer that will show or hide our main and addressform layers. Click the Body radio button to bring the Body portion of our web page into view. Enter a new line after the last word in the sidebar layer and choose the Paragraph command from the Format menu.

Click between the paragraph tags that are inserted and enter a new line, then choose the Button... command from the forms menu. In the panel that drops down, enter “View Page” for this button’s name and click OK. Next, choose Break from the Format menu to insert a break tag after the button, then enter a new line for the next button’s code. Choose Button... from the Forms menu and enter “View Form” for this button’s name and click OK.

Snacks

```
<p>
<input type="button" name="ViewPage" value="View Page" onClick="ViewPageClicked()"><br>
<input type="button" name="ViewForm" value="View Form" onClick="ViewFormClicked()"></p>
</div>
```

Now we move to the Head portion of our web page. Click the Head radio button to bring it into view. First, we’ll make our addressform layer hidden by changing its visible property to “hidden”. Next, make our main layer visible again by setting its visible property to “visible”.

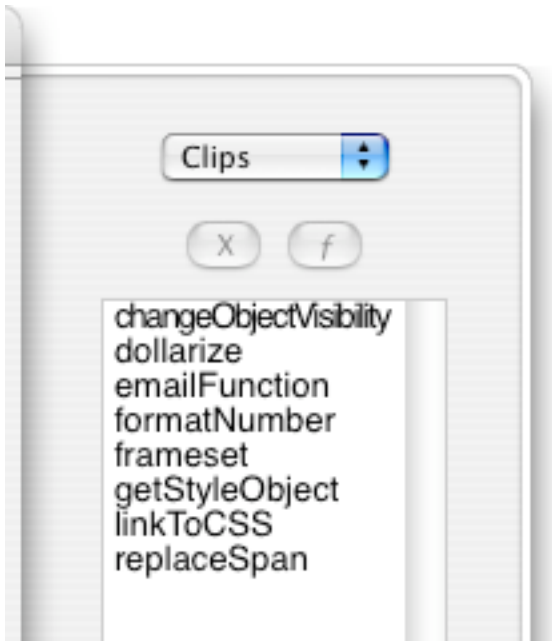
```
#addressform {position:absolute; left:180; top:150; width:460; height:400;
background:rgb(188,189,231); font: normal 16 sans-serif; text-align:center; visibility:hidden}
#main {position:absolute; left:180; top:150; width:460; height:400; background:rgb(255,229,152);
font: normal 12 serif; text-align:left; visibility:visible}
```

Now, we’ll enter script tags and some scripts. Click after the ending </style> tag and press enter on the keyboard to create a new line. Choose the Script command from the JavaScript menu to insert our script tags. All of our scripts will go between these tags. Our first two scripts are in our Parts drawer; choose the Clips list from the Parts drawer’s popup selector.

```
<script language="JavaScript">
</script>
```

Daktari provides a handful of useful scripts for you. Two of them work with changing the visibility of style layers. The changeObjectVisibility function is the script that actually changes the visibility property for a style layer, but it requires the second getStyleObject function to do its work.

Drag and drop the `changeObjectVisibility` and `getStyleObject` clips in between the script tags from the Clips list. Use the Return key to create an empty line between the scripts; it's easier to read scripts if there's a blank line between each of them.



Now we create three functions that will use the `changeObjectVisibility` and `getStyleObject` functions. One will hide both layers, one will show the main layer, the other will show the addressform layer.

Click after the second function, just above the ending `</script>` tag and create a couple of new lines. Choose the function... command from the JavaScript menu. In the panel that drops down, call this new function "hideAll" and click OK to insert the new function.

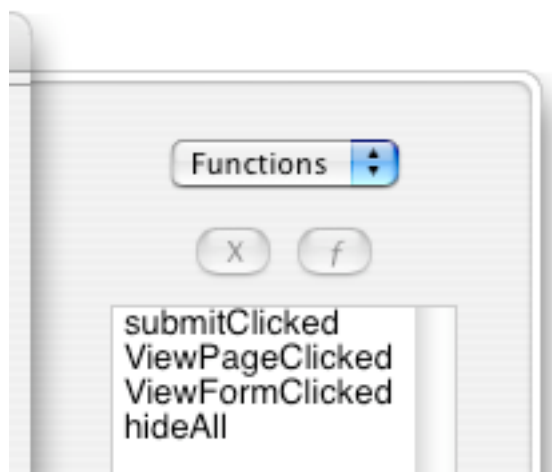
Notice the structure of the function. All functions have a name followed by parentheses and a set of opening and closing braces. This is very similar to syntax used in many programming languages.

```
function hideAll() {
}
```

Click in the line between the opening and closing braces of the `hideAll` function and enter the following two lines of code:

```
changeObjectVisibility("main","hidden");
changeObjectVisibility("addressform","hidden");
```

Our next two functions were added to the Functions list of our Parts drawer when we created our View Page and View form buttons. Choose the Functions list from the Parts drawer's popup selector to see them.



Create two new lines after the closing brace of the `hideAll` function, then drag the `ViewPageClicked` function from the Parts drawer and drop it after the `hideAll` function. In the blank line between the braces, enter the following JavaScript code:

```
hideAll();
changeObjectVisibility("main","visible");
```

Again, create two new lines after our last function, then drag the ViewFormClicked function from the Parts drawer and drop it after the ViewPageClicked function. Add the following code into the blank line between the braces:

```
hideAll();
changeObjectVisibility("addressform","visible");
```

Now we turn to our submitClicked function. Our form in the addressform layer has a JavaScript event handler called "onSubmit" that calls the submitClicked function.

```
onSubmit="return submitClicked(this)"
```

Two things to notice are the return and this keywords. The return keyword is a special significance; it allows us to return a value to the form to let it know whether to actually submit the form. The this keyword passes a reference to the form's name to the submitClicked function so it knows what form to return the value. In this way, we are able to validate all fields of the form have been completed before actually submitting it to the domain server. We will write our submitClicked function to return true if all the fields are complete, or false if any of them are empty.

When we created our form, Daktari added a submitClicked function to our Functions list in the Parts drawer. Create two new lines after our ViewFormClicked function, then drag the submitClicked function from the Parts drawer and drop it after the ViewFormClicked function.

Before we enter code between the function's braces, we want to enter a variable between the parentheses to accept the this value passed to the function from the form's onSubmit event handler.

```
function submitClicked(theForm) {
}
```

The variable can be any word we want to use, or even a single letter. For clarity in our submitClicked function, type "theForm" between the parentheses.

Click between the braces to enter the code for this function. First, we must declare a variable to hold the value we return to the form. Type "var isValidated = true;" which sets the variable isValidated to true. Should all the fields of the form test to contain information, this variable will remain unchanged and return true.

```
function submitClicked(theForm) {
  var isValidated = true;
}
```

Now we enter code that tests whether any of the fields are empty. We could write the function to look at each of the four fields by name, or we can write one line of code that loops or repeats for four times. Press return on the keyboard to create a new line in the function and choose for... from the JavaScript menu. The for loop construction is inserted along with its own set of braces.

The portion of the for loop within the parentheses establishes a loop counter, expressed in three parts, each separated by commas. JavaScript indexes the five elements in our addressEntry form from 0 to 4, so the first part sets *i* to zero. The second part establishes the value *i* reaches to exit the loop. However, *x* is not the variable we passed the form's name in our submitClicked function; change *x* to "theForm". The third part, *i++*, increments the counter *i* by adding 1 to it as the for loop executes for each of the elements in the addressEntry form.

```
function submitClicked(theForm) {
  var isValidated = true;
  for (i = 0; i<theForm.length; i++) {
  }
}
```

We use an if statement to test whether a field contains information or is empty. Click between the braces of the for loop and choose if... from the JavaScript menu. The if statement syntax is inserted in between the for loop's braces. Within its parentheses, the two equals signs indicate the statement will test for equality; values only need be entered on either side of them.

```
function submitClicked(theForm) {
  var isValidated = true;
  for (i = 0; i<theForm.length; i++) {
    if (theForm.elements[i].value == "") {
      isValidated = false;
    }
  }
}
```

On the left side of the two equals signs, within the parentheses, type "theForm.elements[i].value". This is actually a reference to each of the fields of our form. Because theForm refers to the form's name in our web page, we can use it here to refer to it. What appears in the brackets is the index number that refers to an element of our addressEntry form, so we

place our counter, *i*, between the brackets. Because we are testing whether fields are empty, enter two quote marks on the right side of the two equals signs. Values that are text are always stated within quotes; these indicate nothing is between them.

Between the if statement's braces, enter "isValidated = false;". The if statement now indicates that for each element *i*, set *isValidated* to false if element's value has no text.

To finish our submitClicked function, we need to return the result back to the addressForm form to indicate whether to submit the form. To do this, insert a new line between the last two braces and type "return isValidated;" What this means is that during the for loop, if any of the fields are

```
function submitClicked(theForm) {
  var isValidated = true;
  for (i = 0; i<theForm.length; i++) {
    if (theForm.elements[i].value == "") {
      isValidated = false;
    }
  }
  return isValidated;
}
```

empty, *isValidated* is changed to false, otherwise it remains true as we set it before the for loop. After the for loop is finished, the value of *isValidated* is returned to the form.

We now have our set of scripts in the Head portion of our web page to perform tasks required to hide and show the main and addressform layers, and to submit our addressEntry form to a domain server. The complete listing of our scripts and their code is below:

```
<script language="JavaScript">

function getStyleObject(objectId) {
if(document.getElementById && document.getElementById(objectId)) {
return document.getElementById(objectId).style;
} else if (document.all && document.all(objectId)) {
return document.all(objectId).style;
} else if (document.layers && document.layers[objectId]) {
return document.layers[objectId];
} else {
return false;
}
}

function changeObjectVisibility(objectId, newVisibility) {
var styleObject = getStyleObject(objectId);
if(styleObject) {
styleObject.visibility = newVisibility;
return true;
} else {
return false;
}
}

function hideAll() {
changeObjectVisibility("main","hidden");
changeObjectVisibility("addressform","hidden");
}

function ViewPageClicked() {
hideAll();
changeObjectVisibility("main","visible");
}

function ViewFormClicked() {
hideAll();
changeObjectVisibility("addressform","visible");
}

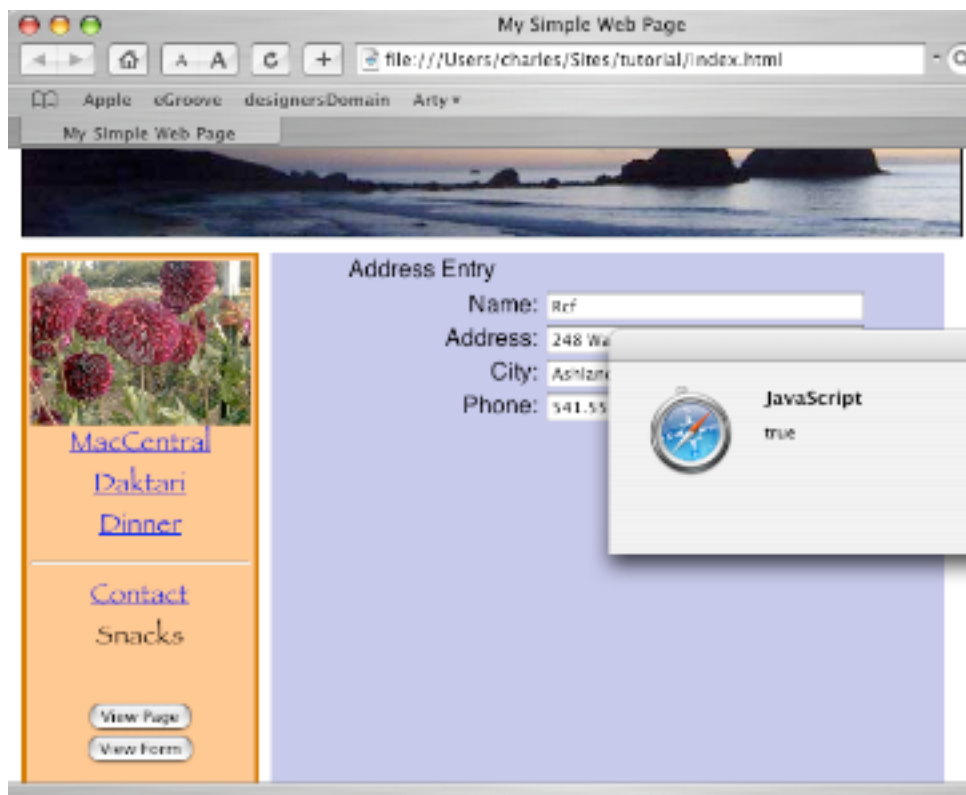
function submitClicked(theForm) {
var isValidated = true;
for (i = 0; i<theForm.length; i++) {
if (theForm.elements[i].value == "") {
isValidated = false;
}
}
return isValidated;
}
</script>
```

We can now view our web page to see if these buttons actually do their work. Click Daktari's view icon to save, colorize and render the page in the web browser. Test the buttons by clicking them. Note that our Submit button for the form is not finished yet, so don't click on it yet.

Our next task will be to change our submitClicked function in order to test the form without actually submitting it. Type two slash marks in front the last line of code in the submitClicked function that reads "return isValidated". This effectively comments the line so JavaScript will not execute it. Enter a new line after this line of code, before the last brace of the function. Type the following code into this new line:

```
function submitClicked(theForm) {
    var isValidated = true;
    for (i = 0; i<theForm.length; i++) {
        if (theForm.elements[i].value == "") {
            isValidated = false;
        }
    }
    //return isValidated;
    alert(isValidated);
    return false;
}
```

Click the Daktari view icon to test our addressEntry form's submit button. If one or more of the fields are empty, the JavaScript alert message should display "false". Only when all the fields contain some text should the JavaScript alert message display "true".



We now know our addressEntry form works accurately and only returns true when all the fields contain some kind of text. Now we will turn our web page into a PHP document. Choose Save As... from the File menu and save our web page as “tutorial.php” in our tutorial folder.

Working with PHP documents is different than with HTML. While they use HTML throughout, they may not be opened locally on the computer as HTML documents are. Most domain servers have PHP enabled. It is simpler to upload PHP documents to the server and test them there. We will use PHP to see how it passes form data by displaying the form data in our main layer after submitting the form. To see the results of our work, we will use Daktari’s Upload command to place our PHP document immediately on the domain server.

First, remove the two lines of test code in our submitClicked function and remove the comments from the return value line. Click above it, enter a new line and choose if... from the JavaScript menu. On the left side of the double equals signs, type “isValidated” and on the right side of them, type “false”. Click between the if statement’s braces and type:

```
alert("Please complete all of the fields to submit the form for processing.")
```

The submitClicked function is now complete.

```
function submitClicked(theForm) {
var isValidated = true;
for (i = 0; i<theForm.length; i++) {
if (theForm.elements[i].value == "") {
isValidated = false;
}
}
If (isValidated == false) {
alert("Please complete all of the fields to submit the form for processing.")
}
return isValidated;
}
```

Click the Body radio button to bring the Body portion of our tutorial.php document into view.

Our addressEntry form’s starting <form> tag contains properties specific to submitting the form’s data to the domain server for processing. The action property indicates the URL that contains the processing instructions. Daktari places a generic “serverURL” for the action property when using the Form command from the Forms menu. Double-click to select it and change it to “tutorial.php”, the name of our PHP document. Often, the action URL is a different web page, but in this example we will see that PHP may be written send form data to the same page.

```
<form name="addressEntry" method="post" action="tutorial.php" enctype="multipart/form-data" on-Submit="return submitClicked(this)">
```

One of the interesting features of PHP is that it can be written to escape in and out of HTML, allowing statements within PHP tags to be sprinkled throughout the document.

```
<?php php statements ?>
```

Another feature to notice is that all variables begin with a dollar sign (\$). In our addressEntry form, all of our text input fields have names. These same names are the variable names that are passed to the URL indicated in the form's action property. Let's see how this works.

Click after the main layer's start tag, create a new line and type "<?php". Press return on the keyboard to create another new line and choose if... from the JavaScript menu. Within the if statement's parentheses, remove the double equals signs and replace them with "\$person". Click between the braces, type ">" to close the PHP statement, and press return to create a new line. Type the following code:

```
<?php
if ($person) {
?>

}
```

```
<p><center>
<?php print($person); ?><br>
<?php print($address); ?><br>
<?php print($city); ?><br>
<?php print($phone); ?>
</center></p>
```

The above code is essentially HTML except for escaping to insert the form variables. Notice each of the variables are the names of our text input elements in our addressEntry form, preceded by "\$". The <center> tags will center these values, and everything is neatly wrapped in a set of paragraph tags.

```
<div id="main">
<?php
if ($person) {
?>
<p><center>
<?php print($person); ?><br>
<?php print($address); ?><br>
<?php print($city); ?><br>
<?php print($phone); ?>
</center></p>
<?php
}
?>
<p> Use the...
```

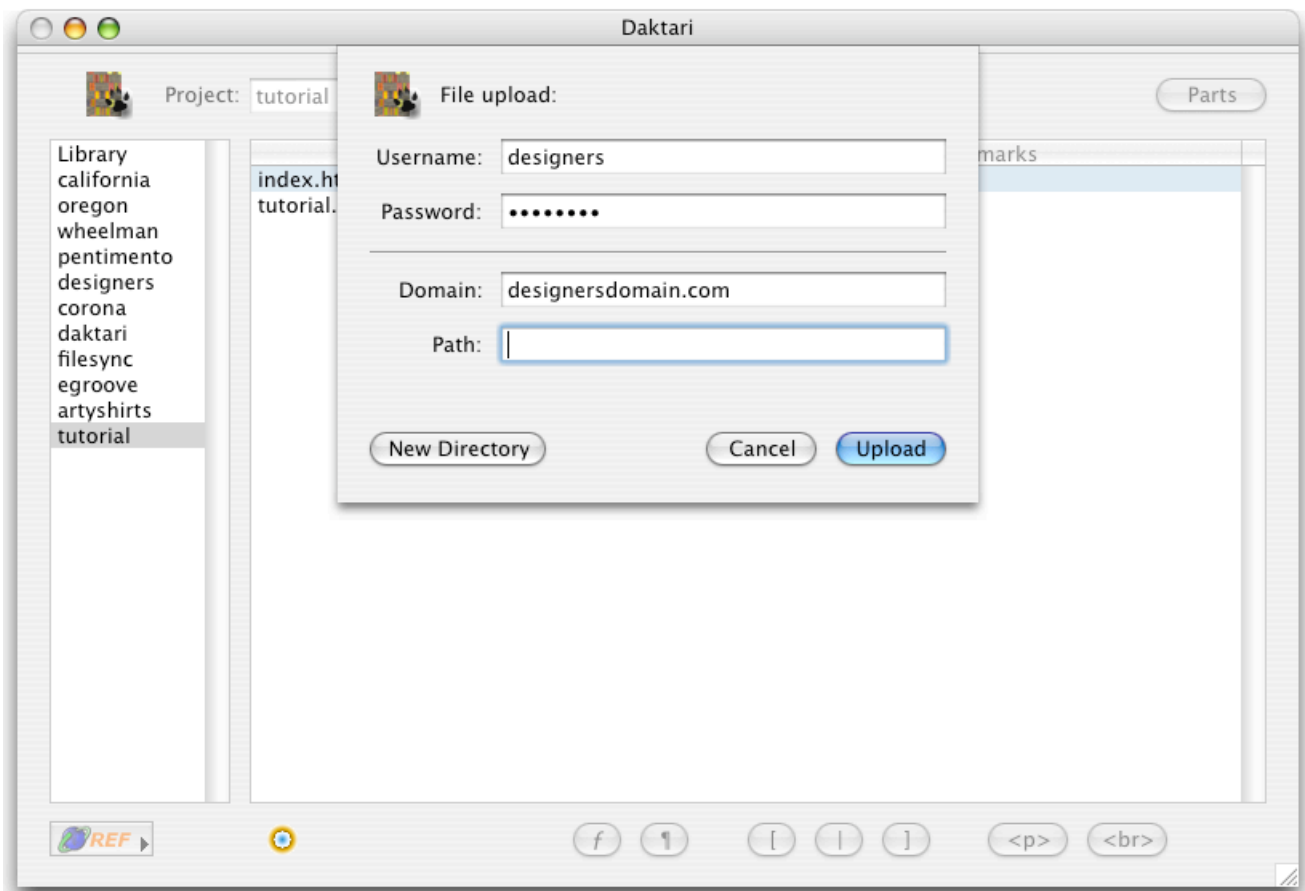
Finally, we need to put the if statement's ending brace within a PHP statement. Press return on the keyboard to create a new line and type "<?php", then click after the if statement's ending brace, enter a new line and type ">".

At left is the entire code as it should appear above the paragraphs of our main layer. What it means is that when the page loads, if a \$person variable is passed to the page, all of the values of the form data will be centered in a paragraph above the main layer's paragraphs. If, there is no \$person variable passed to the page, there is no form data to display, and only the main layer's paragraphs will be displayed.

To see whether this really works, we must save and upload the page to the domain server. While editing a web page, the Upload... command is always available to do this, however this is the first time we are uploading our page to the domain, and there are two images to go along with it. To place everything on the domain at once, we will upload the entire project from the project view.

Click the tutorial project in the Library view. A panel will drop down asking whether to save the changes made to our web page; click Save. The web page will close and the tutorial project list will be in view.

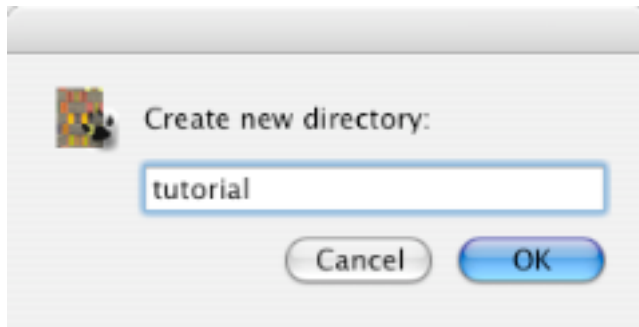
From the File menu, choose the Upload command. In the Upload panel, we generally complete all four of the information fields the domain server will require. All domain servers require a username and password, as provided by your domain host's system administrator.



The panel's domain field should contain only your domain name. Do not include "www" or "ftp", Daktari handles this for you.

We will create a tutorial directory at the root of the domain, so it is not necessary to enter an initial path. Daktari looks at the responses from the domain server and determines the path for us, and places it in the Path field.

Click the New Directory button on the Upload panel. A panel appears, prompting for the name of the new directory. Enter “tutorial” and click “OK”. Daktari sends FTP commands to the operating system’s UNIX shell that connects to the domain and makes the new directory. The path that the domain server returns is entered into the Upload panel’s path field for you.



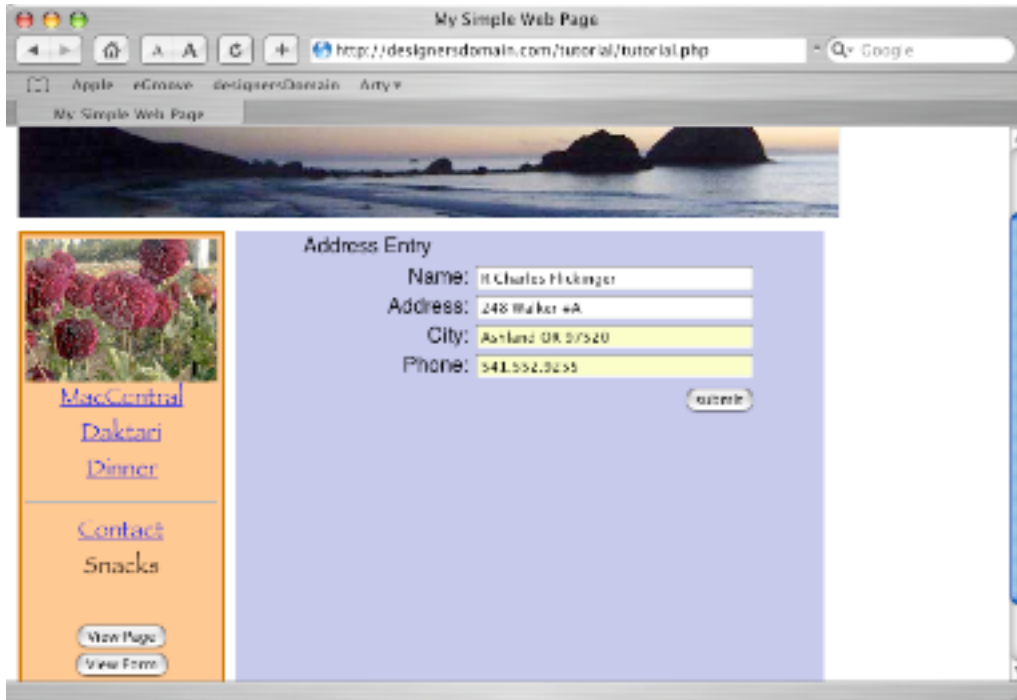
Note the part of the path that leads to the root of the domain. This is everything in front of *tutorial* in the path name. Knowing the path to the root of the domain will help when entering paths to other directories on the domain for other web site projects.

The required information for the Upload panel is complete; click the Upload button. Daktari already knows the file paths to the tutorial project’s pages on the computer’s hard drive. Daktari connects with the domain server, and the entire contents of our project folder is uploaded to the tutorial directory on the domain. When the panel closes, open the web browser and access the page on the domain to test it:

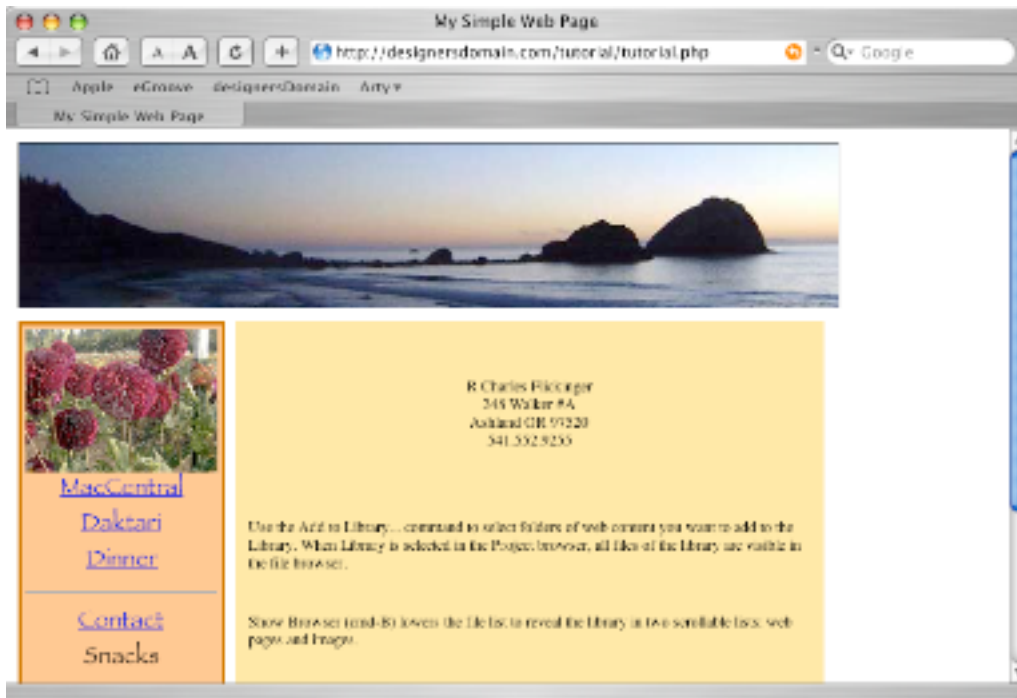
www.yourdomain.com/tutorial/tutorial.php



To test the page, click the “View Form” button. The main layer will become hidden, and the addressform layer will appear. Complete the form’s information fields.

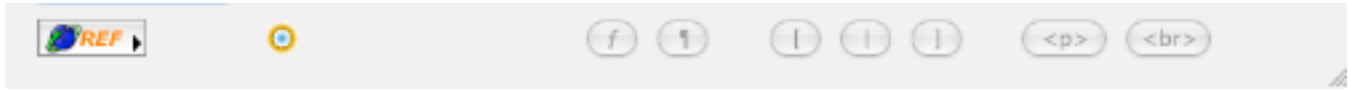


Click the Submit button of the form. The page will reload and display the form data above the main layer’s paragraphs:



More About the Main Window

Throughout the previous tutorial section of this guide, parts of the main window were used to select projects, select and open files related to the projects, switching between the Head and Body views while editing web pages, and opening the Parts drawer. There are other features on Daktari's main window that are useful and described here.



In the section of the tutorial about forms and scripting, we introduced some beginning concepts of working with JavaScript and the PHP languages. In the lower left corner of the main window is a Reference bevel button with instant links to online references for HTML, CSS, PHP and the Daktari web site.

Just right of the Reference bevel button is a rainbow-colored button; when clicked it will refresh the colorization of a web page's code.

In the center, along the bottom of the main window are function and style buttons, designated by “f” and “¶” symbols respectively, that work the same as their counterparts in the JavaScript menu.

Right of these are three justification buttons for left (L), center(|), and right (R) justification tags. To use them, select the text to justify, and click the appropriate button. Tags are placed in before and after the selected text.

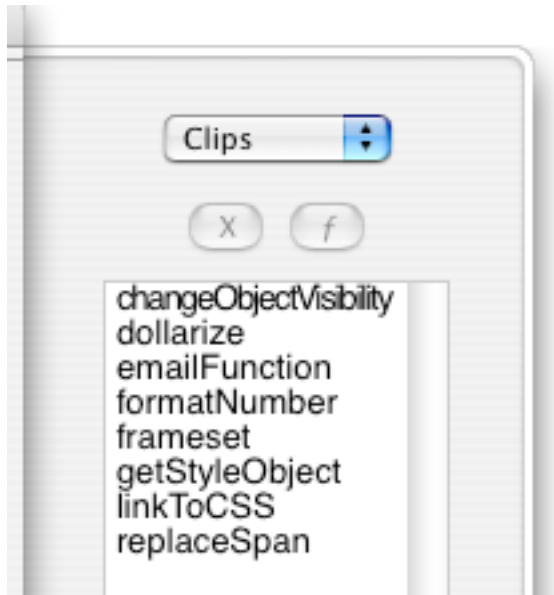
In the lower right are paragraph and break buttons, designated by their tags. These insert a single <p> tag or
 tag respectively. These help minimize the need to visit the menu bar each time for these commands.

The Parts Drawer

We have seen how the Parts drawer can be useful when editing web pages. Having lists of the projects images and the pages styles, functions and elements, not to mention clips saved to the Clips list, keeps them handy and often saves typing.

The Parts button of Daktari's main window opens and closes the Parts drawer. At the top of the Parts drawer is a popup selector for choosing one of the five lists of the parts drawer.

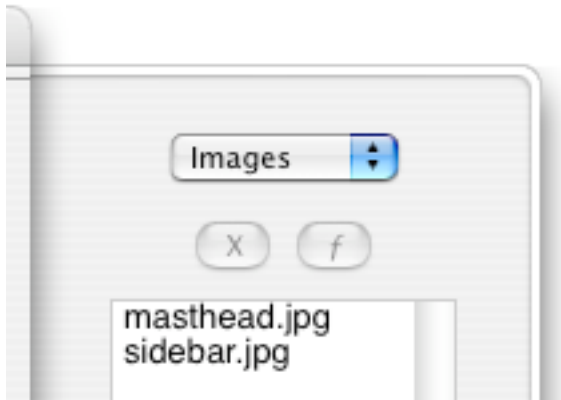
The Clips list stores the names of clips you add to it by dragging selections of text from your web pages to the Clips list. The clips are saved in a Clips folder inside the same folder as the Daktari application.



When a clip is dragged onto a web page, its full text is inserted into the page. This saves a lot of typing and simplifies copying and pasting styles or functions that may be used across several pages of a web site project, as is often the case when achieving a standardized appearance and navigation control in a project.

Daktari provides several useful clips. We've seen two of them, `changeObjectVisibility` and `getStyleObject`, in our tutorial web page. The `dollarize` function works with the `formatNumber` function to format calculated numbers to two decimal places. The `emailFunction` clip is a JavaScript function that helps reduce spam by hiding email addresses in your web page from internet spam bots. The `frameset` clip provides a basic frameset for any web page intended to use frames. The `linkToCSS` clip is a basic link reference to an external CSS stylesheet. The `replaceSpan` clip is basic code for replacing text within a span element; a way to create dynamic text in a web page.

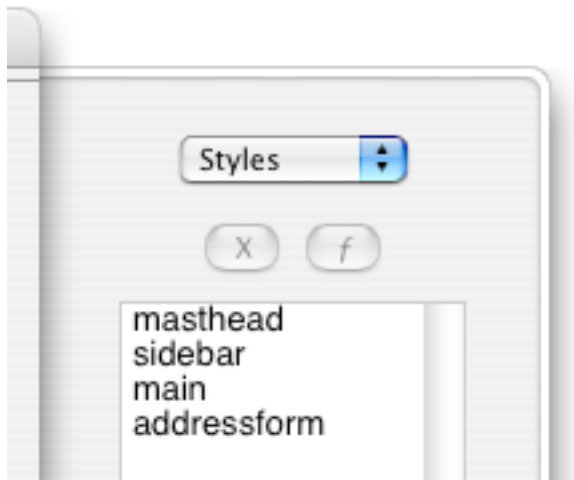
The Images list shows the images related to a selected project. Images are loaded into it when a project is selected in the main window's Library view. We have seen in the tutorial web page, that images are inserted into the Body view of a web page by double-clicking on them from the Images list. If the Body portion of a web page is not in view, Daktari opens a preview of the image.



If an image is selected from the Finder, using the Image... command of the Content menu, Daktari checks to see whether the image is in the project's folder, and asks whether to move it there if it is not. When composing images in a graphics application, images may be saved to a location, such as the desktop, and Daktari will move them to the project's folder when they're used in a web page.

Quicktime movies intended to be included in web pages are also stored in the Images list of the Parts drawer. They work the same way, by double-clicking them in Images list to insert them into the web page.

The Styles list shows the CSS styles defined for the web page. Each time you create a Style ID, the style's name is added to the Styles list of the Parts drawer.



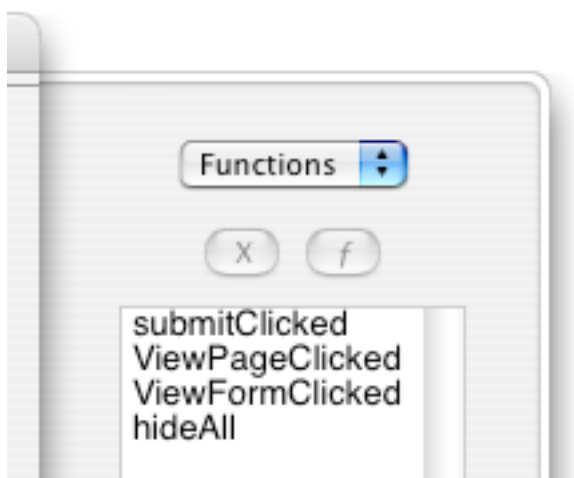
Styles from the Styles list may be dragged and dropped from the Parts drawer into the Body view of a web page. When inserted, they appear as layer tags. Click between the tags and enter the content for the layer.

When a web page has a large collection of styles, they may be quickly located in the page by double-clicking the styles listed in the Parts drawer's Styles list.

The Style ID... command may be used to create styles for classes and specific HTML elements. In the Style properties panel, enter the class or element name (p, body, etc.), and uncheck positioning to exclude those properties from the definition. After the definition is inserted into the web page, change the “#” to a period for a class, or remove it for an element.

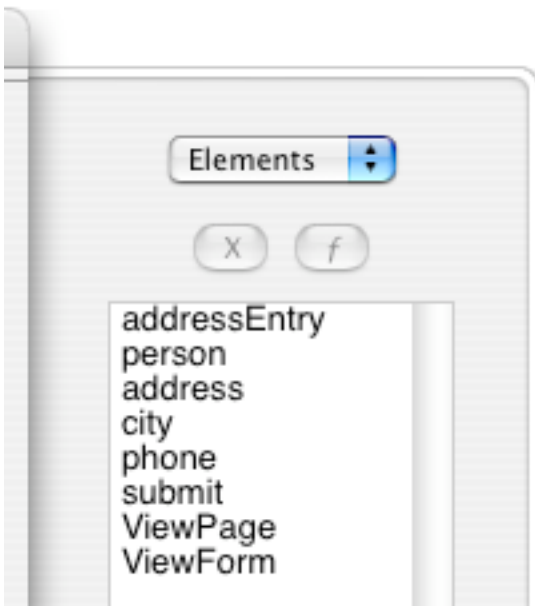
Style classes are handy when a style is needed across different elements in a web page, they're not necessarily tied to specific layers or elements. CSS also offers four pseudo-classes for <a> tags: hover, link, active and visited. A simple way to make links appear hot when the mousepointer is moved over them is to include `a:hover {color:red}` with the style definitions in the Head portion of the page.

The Functions list allows function names to be dragged and dropped from the Parts drawer into either the Head or Body views of a web page. When the function... command from the JavaScript menu is used to create a function, or when certain elements are created such as we've seen with forms and buttons, function names are added to the Functions list. When a function is dragged and dropped into the Head view of a web page, its basic form appears; click between the braces and begin entering the function's code.



As with styles in the Styles list of the Parts drawer, the locations of functions within the Head or Body views of a page may be located quickly by double-clicking the function names in the list.

When certain elements are created in a web page, Daktari adds their names to the Elements list of the Parts drawer. Element names may be dragged and dropped from the Parts drawer into the Head or Body views whenever needed.



Elements may also be located quickly in a web page by double-clicking them in their list in the Parts drawer.

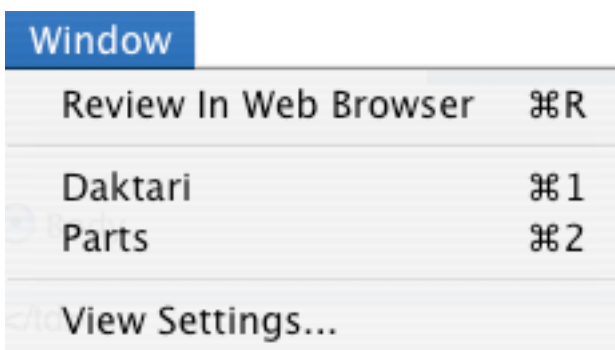
Just above the lists of the Parts drawer, beneath the popup selector, are two buttons. When an item in the styles, functions or elements lists is selected, these buttons are enabled. The button designated with the “x” is a button for removing an item from a list. The button designated with the “f” will locate the item in the web page, similar to double-clicking the item in the list.

Often, when making changes or revisions to a web page, any items that are changed or deprecated remain in the list. While Daktari provides a means to create these lists for styles, functions and elements while building a web page, it doesn’t automatically delete them from these lists.

The lists are for convenience, and the items do not necessarily need to remain in the lists. To update the styles, functions or elements lists, select an item by clicking it, and use the Remove (x) button to remove it from the list. Conversely, any style, function or element name may be dragged from the web page and dropped into these lists to add a name to a list.

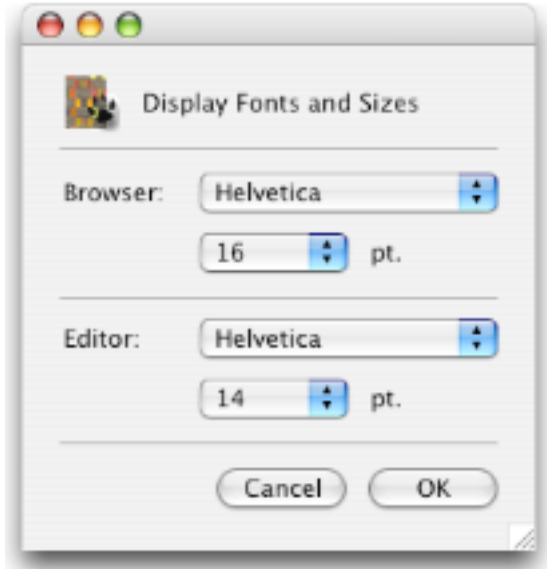
The Window Menu

Daktari’s Window menu has a few commands worth noting. Review in Web Browser offers a command-key equivalent to clicking on Daktari’s view icon in the main window. It saves, colorizes and opens a web page into the web browser.



The Daktari window command will reopen the main window if it was previously closed with the Close command from the File menu. The Parts command works as a toggle as the Parts button of the main window, showing or hiding the Parts drawer.

The View Settings... command allows the fonts and sizes to be set for the browser and editing views. This handy when a larger size font is needed for a large monitor display, or a different font is desired to make codemore readable.



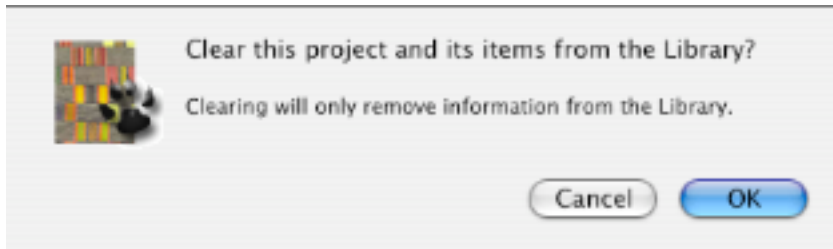
The settings for Browser affect the Library, Library Items, Project List, Pages, and Images views.

The settings for Editor affect the Head and Body views of web pages.

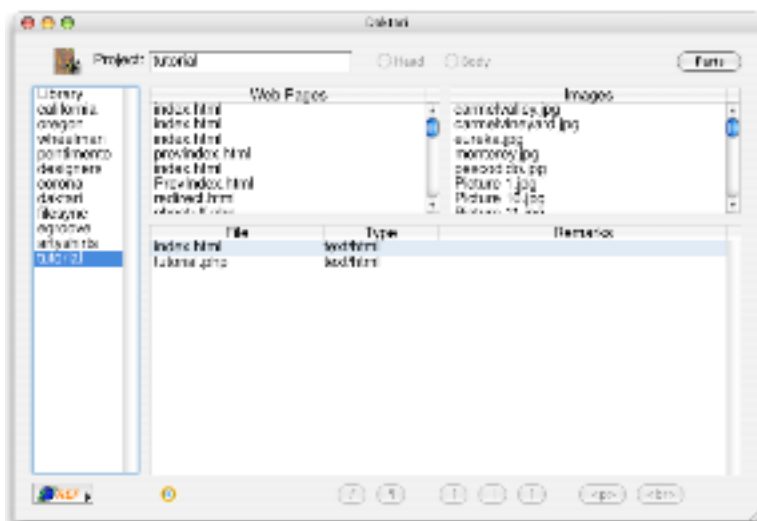
When the Daktari main window is closed, Daktari saves the Library information, and includes the main window's size and view settings so when the main window is re-opened, it will appear as when it was last used.

Additional Edit Menu Commands

The Clear command of the Edit menu works similar to the Clear command in Apple's iTunes.



When a project is selected in the Library view, Daktari will offer to delete the project name and it's related files from the Library. Files remain on the computer's hard drive. Only when clearing a single file does Daktari offer the option to delete a file from the hard drive.



The Show Browser command lowers the Library Items or Project List views to show views of Pages and Images. This command is handy for browsing the files of the library when a single project is selected.

The files in the Pages and Images views work similar to the other Library views. Double-clicking will open them, and Get Info... will present information about them.

Limitations and Issues

Daktari is a great tool that simplifies coding web pages. Daktari can also help simplify organizing your web pages and their related resources such as images, external style sheets and external Javascripts.

Daktari's library is similar to the libraries in iTunes and iPhoto. *If you move or rename files and folders, the Daktari library won't know about it.* If Daktari indicates it was "unable to complete a task", it is most likely it could not find a file that has been moved or renamed. If necessary, a project may be cleared and added to the library again to update it's files' location references.

The Add to Library command does not import subfolders of a folder or their contents; subfolders may be added to a project separately. It is possible to add folders that are not within a project's folder, but this is not a recommended practice.

When inserting images into a web page, Daktari references files by name only, ie. href="filename.html" or src="filename.jpg". If a file is in a subfolder, the subfolder must be entered in front of the filename.

The Upload command is handy for uploading a web page to a domain for testing while editing. It is also available when a project is selected in the Library view, and will upload the content of the selected project's folder, including any subfolders. *Do not upload a project to a domain's root directory unless that is exactly what you intend to do.* Any of the project's files that are not inside the project's folder are not uploaded; it is best to keep all of a project's files within it's project folder whenever possible.

Summary

This guide has presented many of the features and ideas Daktari provides for managing web site projects and their related images. Daktari's menus, panels, and unique Parts drawer, simplify the process of creating and editing web pages.

To learn more about the languages of web pages, Daktari provides links to excellent references on the Internet. These may be chosen from the Reference bevel button on Daktari's main window.

Helpful information about Daktari, including current versions and tips are available at the Daktari web site:

<http://www.designersdomain.com/daktari>

Questions about Daktari, or web pages in general, may be emailed to:

idlewild@designersdomain.com

The Daktari software is provided on a 30-day evaluation, after which the software will not run unless registered. If you find Daktari is a useful tool that saves you time and simplifies your work with web pages and their resources, please consider purchasing a serial number to continue using it.

Online registration fee for Daktari is \$24.95, and serial numbers are issued by email upon receipt of your payment. A convenient link on Daktari's download page is provided that opens a form on Paypal.com to receive online payments.

The Daktari software, registration, Daktari Guide & Reference, and video screenshots of the tutorial presented in this guide, are available on CD-R for \$29.95. Your postal address for shipping is required. Payments for the Daktari CD are accepted on Paypal (use the convenient link on the Daktari web site) or send check or money order (US funds only) to:

Daktari CD-R
c/o R Charles Flickinger
PO Box 983
Talent OR 97540

Daktari

© 2002-2006 by R. Charles Flickinger
All Rights Reserved

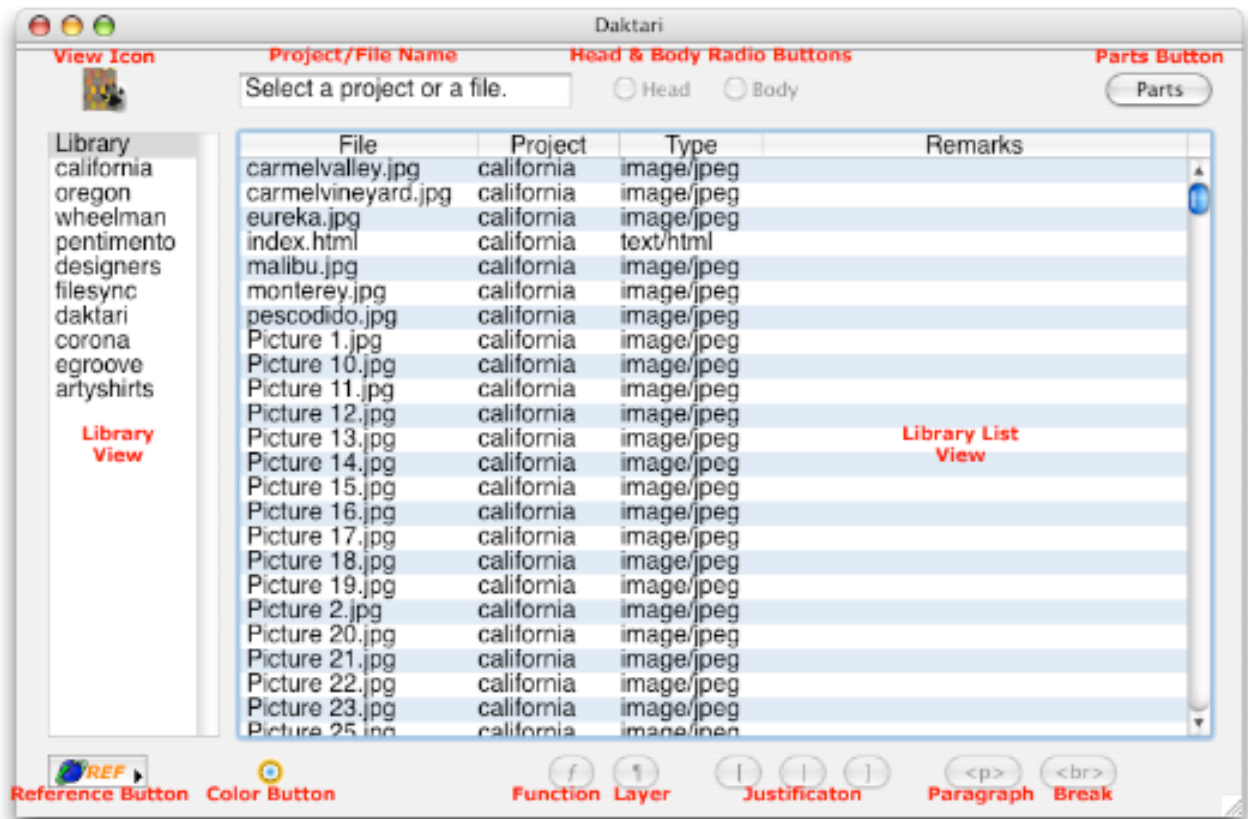
Legal Statement and Disclaimer


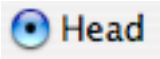
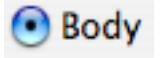
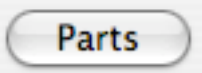
Software authors have no control over what people do with, or on, their computers. While I say Daktari will perform the tasks outlined in this document, it is necessary that I protect myself from those who would negligently use or misuse the software. Because I prefer spending my time creating useful software, rather than defending my intentions in courts of law, the following statement and disclaimer is necessary:




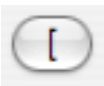


The Daktari software and accompanying files (this documentation and files included in the Daktari Clips folder) are provided “as is” without warranty of any kind to you, the end-user. The author, R Charles Flickinger, does not warrant, guarantee, or make any representations regarding the use of, or the results of the use of the software or accompanying instructions in terms of correctness, accuracy, reliability, currentness, or otherwise. The entire risk as to the results and performance of the software is assumed by you. If the software or instructions are defective, you, the end-user, are responsible for the entire cost of all necessary servicing, repair or correction.

Software companies charge hundreds of dollars for software that has similar disclaimers and legal statements. If after thirty days, you find Daktari a useful software application, please purchase a serial number to register your copy.

Daktari Quick Reference



Daktari Main Window		
	View Icon	Click to save, colorize and view the .html files locally in the web browser.
Project: <input type="text" value="daktari"/>	Project/File Name Field	Displays the name of the current project or web page.
	Head View button	Click to display the Head portion of a web page.
	Body View button	Click to display the Body portion of a web page.
	Parts Drawer button	Click to open or close the Parts drawer.

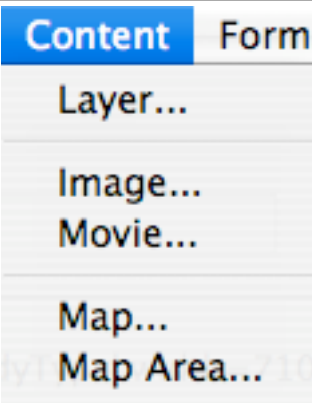
Daktari Main Window (continued)		
	Reference Bevel button	Press to choose quick links to online references for Daktari, HTML, CSS, and PHP.
	Color Refresh button	Click to update text colorization of Head and Body views.
	Function button	Click to create a function in the Head view of a web page.
	Style Layer button	Click to create a style definition in the Head view, or a style layer in the Body view.
	Left Justification button	Click to surround a text selection with <left> justification tags.
	Center Justification button	Click to surround a text selection with <center> justification tags.
	Right Justification button	Click to surround a text selection with <right> justification tags.
	Paragraph Tag button	Click to insert a single <p> paragraph tag.
	Break Tag button	Click to insert a break tag.

File Menu				
File	Edit	Format	New	Begin a new untitled web page document.
New		⌘N		
Open...		⌘O	Open...	Open a selected web page or image from the library or project list. If no file is selected, a file may be chosen with the standard open file panel.
Save		⌘S		
Save As...				
Upload...		⌘U		
Get Info...		⌘I	Save	Saves and colorize the web page opened for editing.
New Project...				
Add to Library...			Save As...	Save the web opened for editing as a different file name.
Close		⌘W		
			Upload...	Upload a project's files to the domain server when a project is selected. Save and upload a web page opened for editing to the domain server.
			Get Info...	Display information about a file selected in the library or project lists, or information about a web page opened for editing.
			New Project...	Create a name and folder on the hard drive for a new project.
			Add to Library...	Select a folder from the computer's hard drive and add its files to a project in the library.
			Close	Close the Daktari main window.

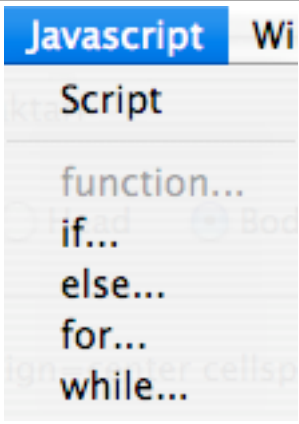
Edit Menu		
Edit	Format	Content
Undo		⌘Z
Cut		⌘X
Copy		⌘C
Paste		⌘V
Select All		⌘A
Clear		
Show Browser		⌘B
Find...		⌘F
Find Again		⌘G
Undo		
Cut		
Copy		
Paste		
Select All		
Clear		
Show Browser		
Find...		
Find Again		

Undo	Return the document back to before the text was recently changed.
Cut	Remove selected text and place it in the clipboard.
Copy	Copy selected text and place it in the clipboard.
Paste	Insert text in the clipboard into the document at the location of the insertion marker.
Select All	Select the entire text of the Head or Body view.
Clear	Remove a selected project and its files from the library, or remove a file selected in the library or project lists, with option to delete it from the computer's hard drive.
Show Browser	Reduce or raise the height of the library or project list views to reveal or hide the Pages and Images views.
Find...	Enter and find occurrences of text in the Head or Body view.
Find Again	Find the next occurrence of text in the Head or Body view.

Format Menu		
Format Cont... Style Style ID... Paragraph Break Rule Table... Anchor... Link... Comment	Style	Insert <style> tags into the Head portion of a web page.
	Style ID...	Create a style definition in the Head portion of a web page, using the Style Properties panel.
	Paragraph	Place starting and ending <p> paragraph tags around selected text.
	Break	Insert a break tag at the location of the insertion cursor.
	Rule	Insert a horizontal <hr> rule at the location of the insertion cursor.
	Table...	Create and insert tags for a set of table cells at the location of the insertion cursor, using the table panel.
	Anchor...	Name and surround selected text with <a> anchor tags.
	Link...	Specify a URL link and surround selected text with <a> link tags.
	Comment	Surround selected text with comment tags.

Content Menu			
	Layer...	Name and create a <div> layer in the Body portion of a web page.	
	Image...	Choose an image from the standard open file panel to add it to a project and insert it into a web page at the location of the insertion cursor using the Image Properties panel.	
	Movie...	Choose a QuickTime movie file from the standard open file panel to add it to a project and insert it into a web page at the location of the insertion cursor.	
	Map...	Name and insert <map> tags at the location of the insertion cursor.	
	Map Area...	Specify the coordinates and URL reference for a map item.	

Forms Menu			
Forms Javascript Form... Field... Text Area... Button... Radio... Check Box... Selector... Option...	Form...	Enter <form> tags at the location of the insertion cursor.	
	Field...	Name and enter a text input element at the location of the insertion cursor.	
	Text Area...	Name and enter a text area element at the location of the insertion cursor.	
	Button...	Name and enter a button input element at the location of the insertion cursor.	
	Radio...	Name and enter a radio input element at the location of the insertion cursor.	
	Checkbox...	Name and enter a checkbox input element at the location of the insertion cursor.	
	Selector...	Name and enter a popup selector element at the location of the insertion cursor.	
	Option...	Name and enter an option item for a popup selector.	

JavaScript Menu		
	Script	Insert starting and ending <code><script></code> tags around selected text in a web page.
	function...	Name and insert basic function syntax into the Head view of a web page at the location of the insertion cursor.
	if...	Insert basic if statement syntax into a web page at the location of the insertion cursor.
	else...	insert basic else statement syntax into a web page at the location of the insertion cursor.
	for...	Insert basic syntax for loop into a web page at the location of the insertion cursor.
	while...	insert basic while loop syntax into a web page at the location of the insertion cursor.