

PDFManual

Table of Contents

1	<u>ButtonGadget Manual and Online Help</u>	1
2	<u>Getting Started</u>	2
	2.1 <u>Getting Started</u>	2
	2.2 <u>About ButtonGadget 3 minute tutorial</u>	2
	2.3 <u>Using ButtonGadget with the Hemingway Content Management System</u>	3
3	<u>Interface Overview</u>	4
	3.1 <u>Interface Overview(click image to view full size)</u>	4
4	<u>Button Text tab</u>	6
	4.1 <u>Button Text Tab</u>	6
5	<u>Button Image tab</u>	8
	5.1 <u>Button Image tab</u>	8
6	<u>View Button tab</u>	9
	6.1 <u>View Button tab</u>	9

☐☐☐ ButtonGadget Manual and Online Help

Last Updated:5/25/2003



2 ☐☐☐ Getting Started



2.1 Getting Started

First, if you haven't yet, check out the [ButtonGadget Tour](#) .

There's also an [online tutorial](#) to learn how to build your own ButtonSets from scratch.

2.2 About ButtonGadget 3 minute tutorial

Welcome to the About ButtonGadget section of the manual. After installing ButtonGadget, open it to the main screen.

Before we discuss the interface controls, let's quickly build some buttons and see how easy it is! Just click on the **Load Button Set** button and choose the **Brushed Metal** button from the scrolling list. Note the button named **Online Buttons** at the lower left of the button library screen. This will open a browser and take you to a constantly updated online library where more buttons are available including user contributions.

Click the **Select** button and the various button states will automatically load with the text *Brushed Metal* on them. First, Let's change the brushed metal button to say *Hello World*. So, highlight the text *Brushed Metal* in the Button Text field and type *Hello*, press return and type *World*. Press **tab** to render the text to the selected button (indicated by the blue vertical line next to it). Press the **Apply Text to All** button and watch as all buttons are rendered to their respective positions.

Next, let's change the background color. Select the **Button Image tab** and click the **Background Color** button and choose a medium blue color. The *active* button will now display the newly rendered background. Notice the transparent shadow is now blue and not grey. This is a feature called *alpha masking* which allows you to create higher quality images with more subtle effects! Click the **Set Color for All** button to apply the background color settings to all the buttons.

Let's say we only want to use the **Normal, Mouse Over, and Mouse Down** states for this button. *Uncheck* the **Disabled** checkbox to *hide the Disabled button*. Now, the Disabled button will not be rendered or exported.

To render the button, select the **View Button** tab and press the button, **Build Button**. The screen will refresh as the various button states are composited. After a few seconds, the button is built. You can now interact with the button. Roll your mouse over it and press down on it.

Let's now save the buttons as JPG's. Click on the **Export Images** button and choose a folder to put the images in. You will be prompted if you would like to create javascript to go with the buttons you are exporting. Select Yes.

That's all there is to it. You've just built custom professional looking buttons in only a few seconds! There are more powerful and advanced features in ButtonGadget, such as *the ability to make your own ButtonSets from scratch*. To learn more about these and the ButtonGadget interface, please refer to the manual on the ButtonGadget website.

Button thoughts

So, what are buttons and how do you use them? **ButtonGadget** renders various button states. This includes: **Normal, Mouse Over, Mouse Down and Disabled**. A ButtonSet is a file, which contains one or more of the button states. Some ButtonSets contain all four button states, others contain one or two— it depends on the who authored them and their intended use. Most people designing webs will use only two button states, Normal and Mouse Over. Why then four states? In some instances (the top navigation bar of www.buttongadget.com) the Mouse Down state is also used. Application interface designers often need a fourth Disabled state for buttons.

So, how does one use them? Well, a web designer would render Normal and Mouse Over button states as two separate images, then upload to a server and use javascript to swap the images when the cursor rolled over it. In any case, it is beyond the scope of this manual to describe all situations and techniques for using custom rendered buttons. It is sufficient to say that ButtonGadget can solve most all button rendering tasks.

2.3

Using ButtonGadget with the Hemingway Content Management System

To create up/down buttons for Hemingway:

Open ButtonGadget and load a buttonset. Turn off the MouseDown and Disabled (these aren't used for exporting with javascript).

If you want, you can 'switch' the MouseDown and MouseOver buttons -- Do the following:

Turn off all the buttons except for the ones you want to swap images. So, if you wanted to switch the MouseDown with the MouseOver, turn off the Normal and Disabled views. Then highlight the MouseDown state by clicking on it (the blue vertical bar will now be next to the MouseDown state). Then click the 'Copy Image to All' button. This will put the image which was in the MouseDown Position to the MouseOver position.

Next, finish building your button as you would normally. Type in the text and copy it to the other button states and position it.

Then under the View Button Tab, Alt-Click on the * (asterik) button. You will get the message: File Naming set for Hemingway -- which means it will add an _x to the name of the button. We generally up the JPG quality to 90%. Click Export Images button and give it a name like fred. Answer Yes to the Do you wish to create javascript prompt. In the next dialog box, enter in the name of link you wish to go to. If you are linking to a file, then just put the name of the file there. Example: fred.iwz (MagicItem file which has already been uploaded as an item) or fred_x.iwz (MagicItem file which will be uploaded using HemUtils File Manager).

This will create a set of files: fred_x.jpg, fred_over_x.jpg, and fred.htm. Open the fred.htm file in Notepad and copy all the text from it and put it into a Hemingway Item -- *make sure and check the use HTML box!*

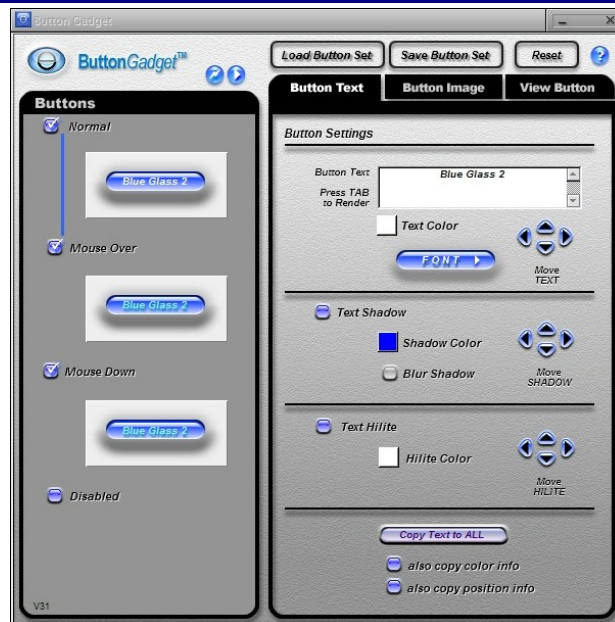
Now, open up HemUtils, launch File Manager, and upload the images fred_x.jpg and fred_over_x.jpg (and the MagicItem file if you're using one). Now build the website and everything should work just fine! BTW, this only works for ONE button per page. If you wish to put multiple buttons on a page, then you'll need to edit the javascript code per the instructions on this faq:

How can I modify the Javascript so multiple buttons on a page will work?

at

<http://www.buttongadget.com/buttongadget/FAQs.htm>

3 ☐ Interface Overview



3.1

Interface Overview

(click image to view full size)

The **ButtonGadget** interface is divided into three parts:

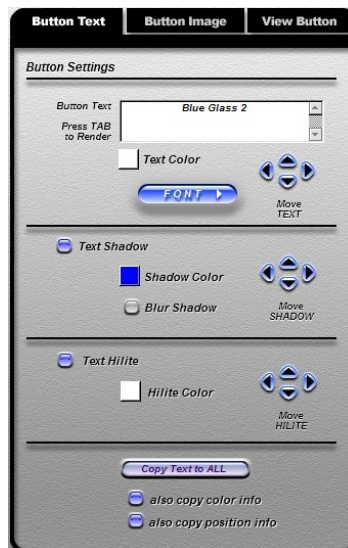
- 1)The **Button States panel** is on the left and titled **Buttons**. It contains the four button states: **Normal**, **MouseOver**, **MouseDown** and **Disabled**. Here you can view the changes made to the different button states. **Click on a button state to set it 'Active.'** All changes are made to the 'Active' button state. To 'turn off' a button state, click the check box next to the button state and it will be hidden.
- 2)The **Editing Panel** is on the right and has three tabs named: **Button Text**, **Button Image**, and **View Button**. This is where most of the button design and customization is done. More on how this works later.
- 3)The **Top Area** contains application controls, which are always visible. From top left to right:
 - a.**Online Update Check control** – checks for latest version of ButtonGadget and prompts to download. Note: Please close ButtonGadget before downloading a newer version.
 - b.**Shrink / Grow interface control** – toggles between a large and small interface
 - c.**Load Button Set control** – opens the button library where you can choose new ButtonSets to load. This is generally the place to start when designing new buttons. The button library shows a list of the installed ButtonSets on your local computer. You may choose to go online and download new ButtonSets which will be automatically added to your local library.
 - d.**Save Button Set control** – allows you to save customized version of an existing ButtonSet to your local library.
 - e.**Reset control**– clears out all button images and resets all parameters to their default state. Use this control to design ButtonSets from scratch.
 - f.**Help Control** – used to do the following:

- i. Access the version number of ButtonGadget
- ii. Register ButtonGadget
- iii. View the online help manual

4 ☐ Button Text tab

4.1

Button Text Tab



Editing Buttons

Buttons are edited using the **Editing Panel**. First, click the **Load Button Set** control and select a ButtonSet to customize. The states available for editing are rendered into the **Button States panel**. There are three tabs to the **Editing Panel: Button Text, Button Image and View Button**. Following is a description of the tabs and their respective controls.

Button Text tab

- 1) **Button Text field** – to change the text of a button, click on the button state you wish to change and then type into the Button Text field the text you wish to render for it. Press the tab key to view the text in place on the active button state.
- 2) **Text Color button** – click to change the color of the text for the active button.
- 3) **Font button** – sets the font, font size, font style and font alignment for the active button.
- 4) **Move Text controls** – moves the text on the active button. Hold the shift key down while pressing to move text faster.
- 5) **Text Shadow checkbox** – if checked will display shadow text for the active button.
- 6) **Shadow Color button** – click to change the color of the text shadow for the active button.
- 7) **Blur Shadow** – if checked will create a subtle blur for the shadow for the active button.
- 8) **Move Shadow controls** – moves the shadow on the active button. Hold the shift key down while pressing to move shadow faster.
- 9) **Text Hilite checkbox** – if checked will display highlight text for the active button.
- 10) **Hilite Color button** – click to change the color of the text highlight for the active button.
- 11) **Move Hilite controls** – moves the highlight on the active button. Hold the shift key down while

pressing to move highlight faster.

12) **Copy Text to All button** – copies the text of the active button to all the other visible button states (those that are currently checked 'on').

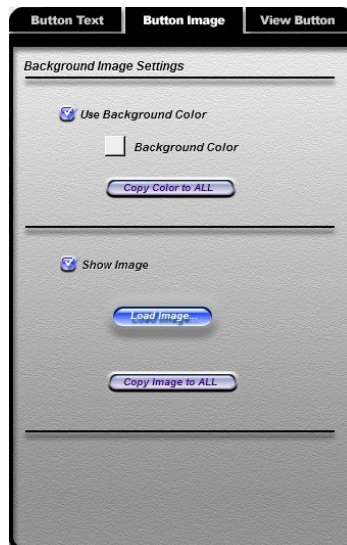
13) **also copy color info checkbox**– works with the Copy Text to All button and when checked will copy not only the text, but also the color settings, the text shadow settings and the text hilite settings to the other visible button states.

14) **also copy position info checkbox** – works with the Copy Text to All button and when checked will copy not only the text, but also the text position settings to the other visible button states.

5 ☐ Button Image tab

5.1

Button Image tab



1)**Use Background Color checkbox** – if checked will display the background color for the active button state. Note: if a button has no transparency channel (i.e. a rectangular button with no drop shadow) this setting will not display any change as there is no transparency for the background to show through. In this case, leave it checked for correct rendering of saved ButtonSets.

2)**Background Color button** – click to change the color of the background for the active button.

3)**Copy Color to All button** – copies the background color of the active button to all the other visible button states (those that are currently checked 'on').

4)**Show Image checkbox** – if checked will display the 'current' loaded image for the active button.

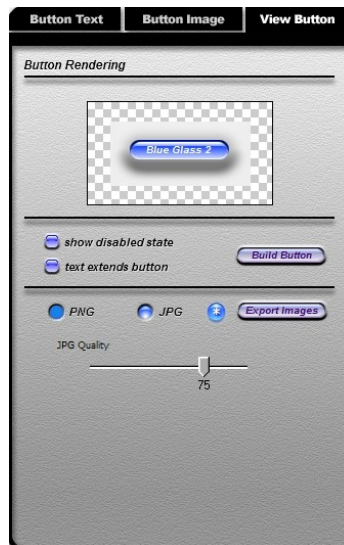
5)**Load Image button** – prompts the user for a JPG, GIF or PNG file to load as the 'current' image for the active button. If a GIF or PNG file with transparency is selected, then the transparency will be preserved in the 'current' image.

6)**Copy Image to All** – copies the 'current' image of the active button to all the other visible button states (those that are currently checked 'on').

6 View Button tab

6.1

View Button tab



1)**Render Area** – gray and white checkerboard rectangle which is used to display finished, composited rendered buttons. The checkerboard pattern allows you to view the 'transparent' areas of your button, which is used when exporting using the PNG format (the JPG format does not support transparency).

2)**Build Button button** – renders the current visible button states to the Render Area. After finished, you can click on the button and view the different button states interactively.

3)**show disabled state checkbox** – if checked displays the currently rendered state for the Disabled state in the Render Area. Used primarily to check the final render of a Disabled button state. If the Disabled button state is inactive, then this button will display nothing in the Render Area.

4)**Text extends button** – used in the rare case where the text goes outside beyond a button outline (ex. checkbox and radio buttons). When checked, it is more computationally intensive, so try first leaving unchecked. Technically, it renders text over the transparency channel, then generates a new transparency channel for the final image(s).

5)**PNG export radio button** – if selected will export the previously rendered button states generated by the Build Button button in PNG format. If a transparency channel exists, it will be preserved in the PNG file(s).

6)**JPG export radio button** – if selected will export the previously rendered button states generated by the Build Button button in JPG format.

7)**JPG Quality slider** – sets the quality level for JPG files exported. The quality range is between 1 and 100 with 100 being the best quality.

8)* **button** – changes the default file naming conventions used when exporting images. You can customize the file names for each of the separate button states as they are exported.

9)**Export Images button** – prompts the user for a file prefix name, then saves the rendered buttons as displayed in the Render Area, one at a time, with the set file naming conventions. If the JPG export is chosen, then ButtonGadget will prompt you if you want to create javascript to go along with the files just exported. Javascript will only be created for the Normal and Mouse Over button states.