GFX Library Integration With Tritium

By: Nebojsa Pintaric

Introduction

Library home page: http://maccman.github.com/gfx/

Library provide integration with CSS3 transformation and transitions into jQuery, making it simple to create animation widgets

So far, these three widgets are covered:

1. Flip
2. Overlay
3. Cube

Library usage

Before we go into usage detail of those widgets note that this library depends on jQuery. So if it's not already included into your project you need to import it manually:

go to "[project_name]/assets/javascripts/bundles.yml" and after line that says "input_files:" add this line:

- http://code.jquery.com/jquery-latest.min.js

Ok, next we need to include general part of library that all widgets share. To do that, just copy file gfx.js into folder - "[project_name]/assets/javascripts/main" .

Flip

The flip effect is for showing two elements that be can flipped between.

Steps to setup:

1. Copy gfx.flip.js into “[project_name]/assets/javascripts/main” folder.
   Note that you should use files I’ve provided since there are slightly changes from original.

2. Go to file “[project_name]/functions/main.ts” and add these two functions:

```
@func XMLNode.gfx_flip(Text %name){
	# Set attributes
	attribute("gfx_type", "gfx_flip")
	attribute("gfx_flip_item", %name)

	# Generate and insert script on page
	$("/html/head") {
		insert_bottom("script", type: "text/javascript"){
			attribute("charset", "utf-8")
				inner("jQuery(function($) {$('[gfx_flip_item=" + %name +
"]').gfxFlip();});")
		}
	}
}

@func XMLNode.gfx__flip_button(Text %name){

	# Set attributes
	attribute("gfx_type", "gfx_flip_button")
	$("//*[@gfx_type='gfx_flip_button']") {
		attribute("gfx_flip_button_item", index())
	}
	%selector = fetch("./@gfx_flip_button_item")

	$("/html/head") {
		insert_bottom("script", type: "text/javascript"){
			attribute("charset", "utf-8")
			inner("jQuery(function($) {$('[gfx_flip_button_item=" + %selector +
"]').click(function() { return $('[gfx_flip_item=" + %name + "]').trigger('flip');});});")
		}
	}
}
```

3. Now you need to make this tag structure on page where you want to use widget:
    // This is wrapper div
   Front // Class .front is mandatory
   Back // Class .back is mandatory

4. All there is left to do is to fire gfx_flip(Text %name) function on widget wrapper. And gfx_flip_button(Text %name) on elements you want to flip widget. That can be widget sides (front/back) or any element outside wrapper. Note that name should be same in both function, so script can connect those two.

Example:

Div "nav-mens" is used as whole widget wrapper. There are two child divs with classes front/back. Function gfx_flip_button(mens) is fired

on whole front side of widget and for flipping back we only want to use link a.goback.

Overlay

Overlay widget enables us to put some content in front of rest of content and to darken background.

Steps to setup

1. Copy gfx.overlay.js into [project_name]/assets/javascripts/main" folder.
   Note that you should use files I've provided since there are slightly changes from original
2. Go to file "[project_name]/functions/main.ts" and add these two functions

```
@func XMLNode.gfx_overlay_button(Text %name){

	# Set attributes
	attribute("gfx_type", "gfx_overlay")
	attribute("gfx_overlay_button", %name)

	# Generate and insert script on page
	$("/html/head") {
		insert_bottom("script", type: "text/javascript"){
			attribute("charset", "utf-8")
			inner("jQuery(function($) {$('[gfx_overlay_button=" + %name +
"]').click(function() { return $.gfxOverlay('[gfx_overlay_content=" + %name +
"]');});});")
		}
	}

}

@func XMLNode.gfx_overlay_content(Text %name){
	# Set attributes
	attribute("gfx_type", "gfx_overlay")
	attribute("gfx_overlay_content", %name)
}
```

3. On page where you want to use this widget you need to have two tags:
   - Tag for overlay button (triggering widget)
   - Tag for overlay content

4. At overlay button element you need to fire gfx_overlay_button(Text %name) function and on content tag gfx_overlay_content(Text %name) function. Note that name should be same in both function, so script can connect those two.
5. You need to style and position popup. Box here's some example CSS to get you started.

```
#gfxOverlayPanel {
        width: 300px;
        height: 300px;
        margin: 10% auto;
        background: #E3E3E3;
        border: 1px solid white;
        -webkit-border-radius: 2px;
        -moz-border-radius: 2px;
        border-radius: 2px;
        -webkit-box-shadow: 0 0 20px rgba(0, 0, 0, 0.4);
        -moz-box-shadow: 0 0 20px rgba(0,0,0,0.4);
        box-shadow: 0 0 20px rgba(0, 0, 0, 0.4);
}
```

Example:

```
/*** Womens ***/
insert("div", class: "db"){
  insert("div", id: "nav-womens"){

    attribute("class", "block")
    insert("div", class: "front side"){
      gfx_overlay_button("women")
    }
    insert("div", class: "back side"){
      gfx_overlay_content("women")
        insert("div", class: "wrapper"){…
    }
  }
}
```

We have one div that acts as a button and one div that acts as a content.

Cube

The Cube is great for displaying multiple pieces of information, perhaps steps in a tutorial or setup procedure.

Steps to setup

1. Copy gfx.cube.js into [project_name]/assets/javascripts/main" folder.
   Note that you should use files I've provided since there are slightly changes from original
2. Go to file "[project_name]/functions/main.ts" and add these two functions

```
func XMLNode.gfx_cube(Text %name){

	# Set attributes
	attribute("gfx_type", "gfx_cube")
	attribute("gfx_cube_item", %name)

	# Generate and insert script on page
	$("/html/head") {
		insert_bottom("script", type: "text/javascript"){
			attribute("charset", "utf-8")
			inner("
			jQuery(function($) {
			var source =   $('[gfx_cube_item=" + %name + "]');
			var cube_width = source.css('width').slice(0,-2);
			var cube_height = source.css('height').slice(0,-2);

			$('[gfx_cube_item=" + %name + "]').gfxCube({
				width: cube_width,
				height: cube_height
				});
			});")
		}
	}


func XMLNode.gfx_spin_cube(Text %name, Text %side){
	# Set attributes
	attribute("gfx_type", "gfx_cube_button")
	attribute("gfx_cube_button", %side)
	attribute("gfx_cube_item", %name)

	# Generate and insert script on page
	$("/html/head") {
		insert_bottom("script", type: "text/javascript"){
			attribute("charset", "utf-8")
			inner("jQuery(function($) {$('[gfx_cube_button=" + %side + "][gfx_cube_item=" + %name +
).click(function() { return $('[gfx_cube_item=" + %name + "]').trigger('cube', '" + %side + "');});});")
		}
```

3. You need to setup correct structure
    // wrapper div

   note that only .front class is mandatory.
4. You need to fire gfx_cube(Text %name) on widget wrapper. And then gfx_cube_button (Text %name, Text %side) with parameter %name for which cube to rotate and parameter %side to which page to rotate.
5. Cube wrapper tag should have width and height defined.

Example:

We have gfx_cube("mw_cube") fired at wrapper div. Then we use whole front side as click area to switch to right side with function gfx_cube_button("mw_cube", "right"). On right side we gave link .goback on which we fire function gfx_cube_button("mw_cube", "left") which will trigger cube spinning to show left side. Finally on left side we have another link .goback which will bring us to beginning (front) side with function gfx_cube_button("mw_cube", "front").

Known bugs:

- On tritium powered pages script has delay when clicked several times.
- I didn't manage to rotate cube to "back" side.

```
/*** Kids ***/
insert("div", class: "db"){
  gfx_cube("mw_cube")
  attribute("id", "nav-kids")

  insert("div"){
    attribute("class", "front")
    add_class("side")
    gfx_cube_button("mw_cube", "right")
  }

  insert("div"){
    attribute("class", "right")
    add_class("side")

    insert("div", class: "wrapper"){
      insert("div", class: "mw_col1"){…
      insert("div", class: "mw_col2"){…
      insert("div", class: "mw_col3"){
        insert("a", "ACTIVITY")
        insert("a", "COLLECTIONS")
        insert("a", class: "goback"){
          gfx_cube_button("mw_cube", "left")
          inner("GO BACK")
        }
      }
    }

  }
  insert("div", class: "left"){
    add_class("side")

    insert_top("p", "Search 'Kids'")
    insert("form"){…
    insert("a", class: "goback"){
      gfx_cube_button("mw_cube", "front")
      inner("GO BACK")
    }
  }

}
```