Language Reference for “Shape” HTML

The “Shape” HTML language is written for PHP to make the generation of HTML forms easy for PHP developers & other developers who do not wish to learn all the semantics of the HTML5 standard. Instead these developers can use a familiar coding method to create HTML pages.

**Why Would I Use “Shape” HTML for PHP**

If you are a developer already using PHP then this language will improve your accuracy in terms of outputting correct HTML and eliminate obvious syntax errors. If you are new to HTML and have a strong C /C++ background then switching to PHP and coding as you are familiar will allow you to quickly put together HTML pages and allow you to focus on the code, not the mechanics of how HTML, Java-script and the Web Server work together.

**Tag Methods and How They Work**

Each tag method can be called with one or more parameters, the parameters are dynamic in the sense that they are recognized as either another element, a textual caption or content for the tag and lastly attributes that can enhance the tag. The parameters can be called in any sequence and this allows for an easy flow of the code.

**How to Use “Shape” HTML**

The easiest way to illustrate the functionality is to take you through a step by step thought process of writing a simple html page with a form. You can try this step by step example at <http://mitov.dyndns.org/playbox/> Please clear out the sample code before trying this example.

**First Step**

I'm thinking we need to make a web page so I start with the html enclosing tag which would be done as follows, I'm going to use a variable to hold the HTML output in so I can work with it later or simply output it when I'm done.

//Lets start programmming!

$html = htmltag (

);

//output the generated HTML  
 echo $html;

I've spread out my enclosing brackets because I'm going to want to put HTML tags inside the whole outer tag.

**Second Step**  
Every good HTML page should have a header and a title to make it read properly in the browser. So next step is to add that, which is quite simply moving my cursor to the correct place in my layout, my new code is in bold.

//Lets start programmming!

$html = htmltag (  
   
 **headtag (   
 title ( “My new page” )   
 )**

);

//output the generated HTML  
 echo $html;

The title is a sub tag put inside the headtag which will change the caption in the browser tag. Now this currently is very boring because there is nothing displaying on the screen and the page appears to be blank, I quickly view the source to see this:

<!DOCTYPE html><HTML><HEAD><TITLE>My New Page</TITLE></HEAD></HTML>

**Third Step**

As you saw in the previous example, I have a perfectly structured HTML page and I've hardly had to write any code. I do need to add a body element to the page and I want to make the background green for fun, here I will introduce a simple way of introducing properties to the HTML element.

//Lets start programmming!

$html = htmltag (  
   
 headtag (   
 title ( “My new page” )   
 )**,**

**bodytag ( array ( “bgcolor” => “green” )**

**)**

);

//output the generated HTML  
 echo $html;

As you can see this was not too difficult to achieve using an array property, if you're thinking like me you could have predefined properties for similar elements.

**Fourth Step**I'm thinking now I would like to add a form tag for some inputs and a field set to contain the inputs in a good looking way for the end user. Notice the creation of more attributes to the form tag, I needed to specify the name of the form and an action, the action would allow me to divert the form to a URL for processing. If I left that out it would come back to the page I'm working on.

//Lets start programmming!

$html = htmltag (  
   
 headtag (   
 title ( “My new page” )   
 ),

bodytag ( array ( “bgcolor” => “green” )**,  
 formtag ( array ( “name” => “myform”, “action” => ”http://google.com/” ),  
 fieldset ( “Input some data”**

**)   
 )**  
 )

);

//output the generated HTML  
 echo $html;

If you have done HTML programming before I trust that you are now getting excited about the look and feel for this framework, you should be picking up by now that making mistakes should become a thing of the past because the code is simple and easy to read.

**Fifth Step**I hope so far you have not been left behind and this is all making perfect sense. We're now going to add our first input for an Email address, I will need to give it a name property so that I can have the information submitted to the web server.

//Lets start programmming!

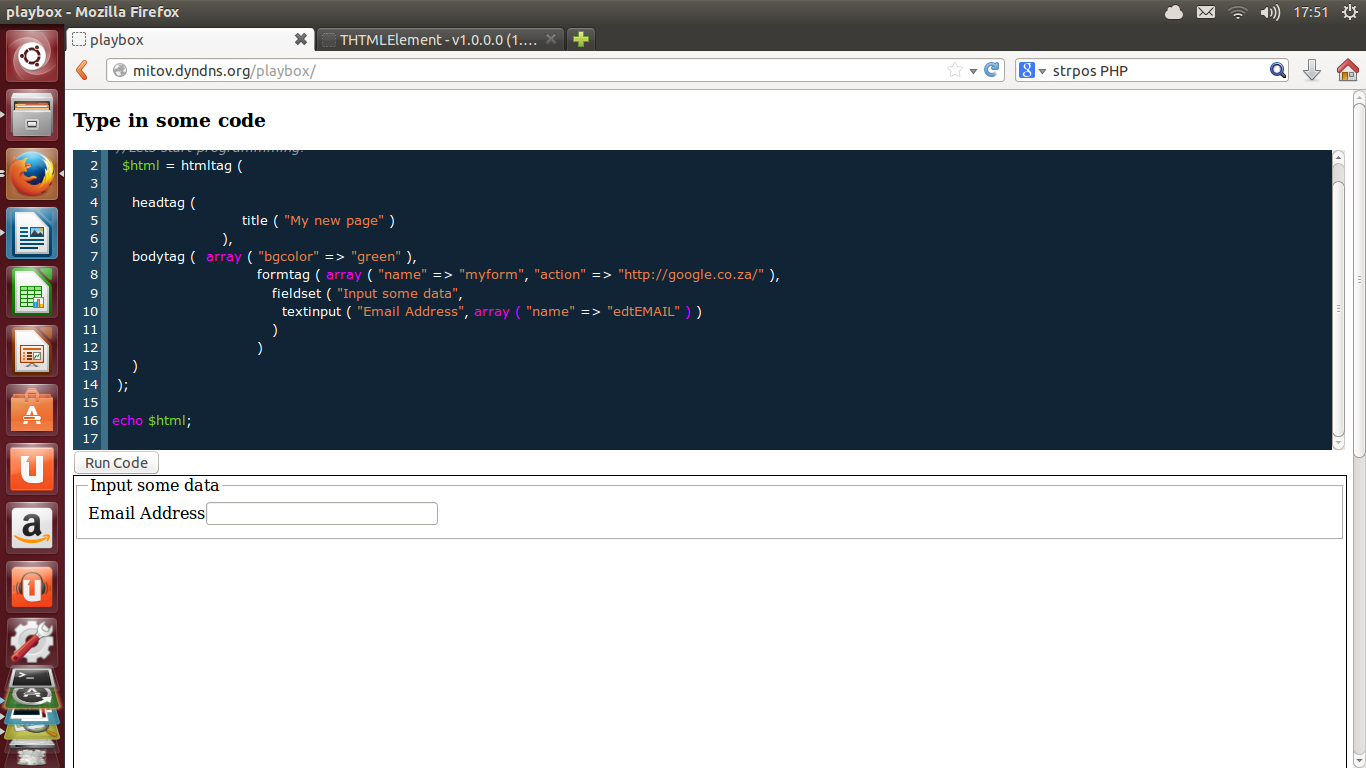
$html = htmltag (  
   
 headtag (   
 title ( “My new page” )   
 ),

bodytag ( array ( “bgcolor” => “green” ),  
 formtag ( array ( “name” => “myform”, “action” => ”http://google.com/” ),  
 fieldset ( “Input some data”**,  
 textinput ( “Email Address”, array ( “name” => “edtEMAIL”) )**  )   
 )  
 )

);

//output the generated HTML  
 echo $html;

If you're following along properly you should have the following on the screen :



**Sixth Step**The cherry on this cake will be the submit button which will send our email input to the Google website.

//Lets start programmming!

$html = htmltag (  
   
 headtag (   
 title ( “My new page” )   
 ),

bodytag ( array ( “bgcolor” => “green” ),  
 formtag ( array ( “name” => “myform”, “action” => ”http://google.com/” ),  
 fieldset ( “Input some data”,  
 textinput ( “Email Address”, array ( “name” => “edtEMAIL”) )**,  
 submit ( “Send to Google” )**   
 )   
 )  
 )

);

//output the generated HTML  
 echo $html;

After running the code we can click on the “Send to Google” button to see what happens. Obviously this is a nonsense request but you can decide for yourself the power of this language.

**Tag / Method Reference**

htmltag  
headtag  
title  
h1tag through h7tag  
ptag  
fieldset  
article  
divtag  
formtag  
bodytag  
textinput  
button  
submit  
line

**Disclaimer**This proposed “language” is in its infancy and may change literally from day to day. The document is intended to act as a proofing ground for the syntax and allow programmers to test the language as it evolves. Any suggestions should be sent to [andre@spiceware.co.za](mailto:andre@spiceware.co.za) for further analysis.