WebMowgli – ( Theming )

**General Theming**

**Prerequisites for theming**

1. Template should be a working static website.
2. All front end animations/javascript should be working as a static website
3. Supports Jquery based websites
4. Preferably use newer version of Jquery

**General concept**

WM picks up the elements from an existing template ( as provided by the theme ) and converts it into suitable data blocks. Template snippets from the original theme provided are used to generate dynamic data on the front end when a page is rendered.

Data blocks are marked out using special extraction tags <extract> tags defined in the documentations. Once all the necessary pages of the original template is complete, the entire theme should be zipped ( .zip file ) at the root folder and imported using the WM importer.

Thats it !!! your website is ready to work with dynamic data.

**General syntax**

In order to create a section as a data block you simply have to surround that section of HTML with the below html ( extract tag )

<extract id=’module\_name:[other module params]’ [ block=’optional block name’] > </extract>

*Note: [] brackets indicate optional syntax*

**General tips ( Do’s and Dont’s )**

1. Currently every page needs to be themed separately
2. If a page already exists and another page ( template ) with the same name is imported, then the newer page is renamed with a suffix ( e.g. about-1, about-2 . .. etc )
3. Data blocks should be identified according to the type of control/data to be displayed at a particular place

**General Parse tags**

The above <extract> tags will help WM identify the data blocks and the html to use from the original template, however for actual data to be populated in the right places, simple parse tags need to be used. These tags are like blank values that will be replaced with the correct data *( while rendering the page )* thus providing the final output to be displayed.

Some general structures are available to use parse tags. Each structure works differently and can render data in a specific format.

The structures are as follows.

**Simple tags**

These are simple parse tags that will be replaced by a single string. Single tags will vary from module to module, hence the right tag should be used in the right location of the data block to get the correct data while rendering.  
  
Eg. {title}, {head}, {site:resource}, {url} . . . . etc

**Loop tags**

These tags are used when you expect a list of repetitive information, like a list of categories, list of posts, list of articles, list of events . . . etc.  
this tag has to be defined only once and WM will automatically repeat the structure with all the data in the list one after the other.

these tags begin with a tag ( specific to each module view ), the ending tag will have a ‘/’ to indicate the end of the loop.

Every loop will have simple parse tags in between the loop tags, these parse tags are only available within the loop and will NOT be rendered if used outside the loop tags

Eg.

{posts}

{title}, {slug}, {author}

{/posts}

**Condition tags**

The condition tags can be used to control the display of a variable OR data depending on some simple conditions.

General format . . .

{if variable\_name=value }

. . . . .. some html

{else}

. . . . .. some other html

{/if}

*NOTE:*

1. {if variable\_name } will check if a variable is set to exists OR not.
2. {else} condition is optional, if NOT used it will display the {if} section if condition = true OR display nothing if condition = false
3. The {if} conditions can ONLY support 1 condition, conditions CANNOT be combined with other conditions with an AND, OR operators
4. {If} conditions cannot be used in a nested manner  
    *E.g. the below structure will cause unexpected results*  
     
   {if variable\_name }  
     
    {if variable\_name2 }  
    .......  
    {/if}  
     
   {/if}
5. {if} conditions can be used in a nested manner ONLY if the underlying {if} statement is inside a Loop tag  
     
   *E.g. the below structure will work*  
   {if variable\_name }  
     
    {posts}  
    {if variable\_name2 }  
    .......  
    {/if}  
    {/posts}  
   {/if}