Manual:

**New Website for Philippine Statistics Authority** using

Metro UI CSS Framework

**Benbinen, Clifford McRey S.**

**Garcia, Danna Marie C.**

**Manaig, Alain Francis L.**

**Sison, Generoso E. III**

**Aldrin Federico R. Bahit, Jr.**

Statistical Coordination Officer IV

Philippine Statistics Authority

Practicum 2 Supervisor

August 2015

**Table Contents**

[**Metro UI CSS Framework Start Up** 4](#_Toc426723890)

[**Layout of the Web Pages** 5](#_Toc426723891)

[**Grid System** 5](#_Toc426723892)

[**1.** **Landing Page** 5](#_Toc426723893)

[**2.** **Sub-pages** 7](#_Toc426723894)

[**Metro UI CSS Framework Classes, Components and JS** 10](#_Toc426723895)

[**1.** **Colors** 10](#_Toc426723896)

[**2.** **Fonts** 10](#_Toc426723897)

[**3.** **Accordion** 14](#_Toc426723898)

[**4.** **Buttons** 16](#_Toc426723899)

[**5.** **Dialog or Modal** 17](#_Toc426723900)

[**6.** **Menu** 19](#_Toc426723901)

[**7.** **List** 20](#_Toc426723902)

[**8.** **Tiles** 21](#_Toc426723903)

[**9.** **Tables** 23](#_Toc426723904)

[Figure 1. Layout of the index.html 5](file:///C:\wamp\www\Benbinen.docx#_Toc426723905)

[Figure 2. Header (logo and icons) 6](file:///C:\wamp\www\Benbinen.docx#_Toc426723906)

[Figure 3. Navigation 6](file:///C:\wamp\www\Benbinen.docx#_Toc426723907)

[Figure 4. Contents 6](file:///C:\wamp\www\Benbinen.docx#_Toc426723908)

[Figure 5. Footer (about us, terms of use, Links, Sitemap, contacts) 7](file:///C:\wamp\www\Benbinen.docx#_Toc426723909)

[Figure 6. Sub-page layout (1) 7](file:///C:\wamp\www\Benbinen.docx#_Toc426723910)

[Figure 7. Sub-page layout (2) 8](file:///C:\wamp\www\Benbinen.docx#_Toc426723911)

[Figure 8. Sub-page layout (1) example. 8](file:///C:\wamp\www\Benbinen.docx#_Toc426723912)

[Figure 9. Sub-page layout (2) example. 9](file:///C:\wamp\www\Benbinen.docx#_Toc426723913)

[Figure 10. Accordion 14](file:///C:\wamp\www\Benbinen.docx#_Toc426723914)

[Figure 11. Dialog 17](file:///C:\wamp\www\Benbinen.docx#_Toc426723915)

[Figure 12. List 20](file:///C:\wamp\www\Benbinen.docx#_Toc426723916)

[Figure 13. Tile with icon 21](file:///C:\wamp\www\Benbinen.docx#_Toc426723917)

[Figure 14. Tile with slide-up label. 21](#_Toc426723918)

# **Metro UI CSS Framework Start Up**

*The front-end framework for developing projects on the web in Windows Metro Style.*

To **get started** in using this framework, there are simple steps to follow before doing anything else.

1. Create a page with HTML5 Doctype. <!DOCTYPE html>
2. Include required styles and libs inside the head element.

<!DOCTYPE html>

<html>

<head>

<link href="/metro.css" rel="stylesheet">

<link href="/css/metro-icons.css" rel="stylesheet">

<scriptsrc="/jquery.js"></script>

<scriptsrc="/metro.js"></script>

<scriptsrc="/js/jquery-2.1.3.min.js"></script>

<scriptsrc="/js/ga.js"></script>

</head>

<body>

</body>

</html>

1. Start coding you contents inside the body element.

<body>

<div class="grid"></div>

</body>

1. See results.

# **Layout of the Web Pages**

# **Grid System**

Grid systems are used to create page layouts through a series of rows and columns that house your content. Here's how the grid system works:

* Rows must be placed within a .grid container
* Use rows to create horizontal groups of columns (container with class .row).
* Content should be placed within columns (container with class .cell).
* Default row contain one cell
* To define cells count use built-inclasses .cells2 ... .cells12 for row.
* To create grid without cell margin, add class .condensed to grid main container
* Cells can extended and take the place of multiple cells using the classes .colspan2 ... .colspan12

\*Example: **Grid with two cells or columns in one row and one cell extended over three size.**

<div class="grid">

<div class="row cells4">

<div class="cell"></div>

<div class="cell colspan3"></div>

</div>

</div>

1. **Landing Page(index.html) –** the image (*Figure 1. Layout of the index.html*)shows the layout of the landing page.This layout is consist of four (4) rows. The **first row** has 2 cells or columns. The **second row** doesn’t have any cells or columns. The **third row** has 3 columns and the last row of the **fourth row** has 4 columns. This whole layout is placed inside a **grid** container.



Figure 1. Layout of the index.html

\*The code for the landing page’s layout looks like this:

<div class="grid">

<div class="row cells12">

<div class="cell colspan6">*…*</div>

<div class="cell colspan6">…</div>

</div>

<div class="row">…</div>

<div class="row cells12">

<div class="cell colspan3">…</div>

<div class="cell colspan7">…</div>

<div class="cell colspan2">…</div>

</div>

<div class="row cells12">

<div class="cell colspan3">…</div>

<div class="cell colspan3">…</div>

<div class="cell colspan3">…</div>

<div class="cell colspan3">…</div>

</div>

</div>

First row

Second row

Third row

Fourth row

* 1. **First row (logo, icons)** – The first row consists of the **logo** that is placed in the first cell or column **and the icons** for Facebook, twitter and e-mail that is placed in the second cell.

Figure 2. Header (logo and icons)



* 1. **Second Row (navigation) –** Inside the second row which doesn’t have any columns is the navigation part of the webpage.



Figure 3. Navigation

* 1. **Third row (contents) –**This row has three (3) cells or columns. This is where the contents of the landing page are placed. In the first column, **STATBOARD** is placed, second column **HEADLINES** are placed, and in the last column are **advisory, transparency seal, Citizen’s Charter** and etc.



Figure 4. Contents

* 1. **Fourth row (footer) –**This row has 4 columns which consists of about us, terms of use, links and sitemap and lastly the copyright, address etc.

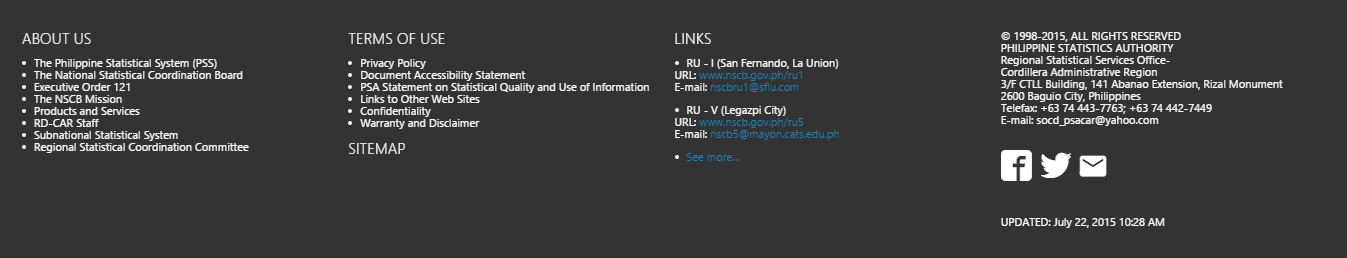


Figure 5. Footer (about us, terms of use, Links, Sitemap, contacts)

1. **Sub-pages–** These pages has a **different layout in its third row** from the landing page (index.html). Since the other rows (first, second and fourth) are same with the previous discussion, this will focus on the third row part. As you can observe, the third row has only **2 cells or columns**. In the first column, the contents are placed while in the second column, different (depending on the web page) navigations are placed. This layout (*see figure 6 – Sub-pages layout (1))* is applied in most of the pages.



Figure 6. Sub-page layout (1)

\*The code for **figure 6’s** layout looks like this:

<div class="grid">

<div class="row cells12">

<div class="cell colspan6">*…*</div>

<div class="cell colspan6">…</div>

</div>

<div class="row">…</div>

<div class="row cells12">

<div class="cell colspan8">…</div>

<div class="cell colspan4">…</div>

</div>

<div class="row cells12">

<div class="cell colspan3">…</div>

<div class="cell colspan3">…</div>

<div class="cell colspan3">…</div>

<div class="cell colspan3">…</div>

</div>

</div>

*This* ***code*** *is the third row.*

1. ***cells12 –*** *this means that the row have 12 cells or columns.*

***\*note: the maximum cell is 12 (cells12)***

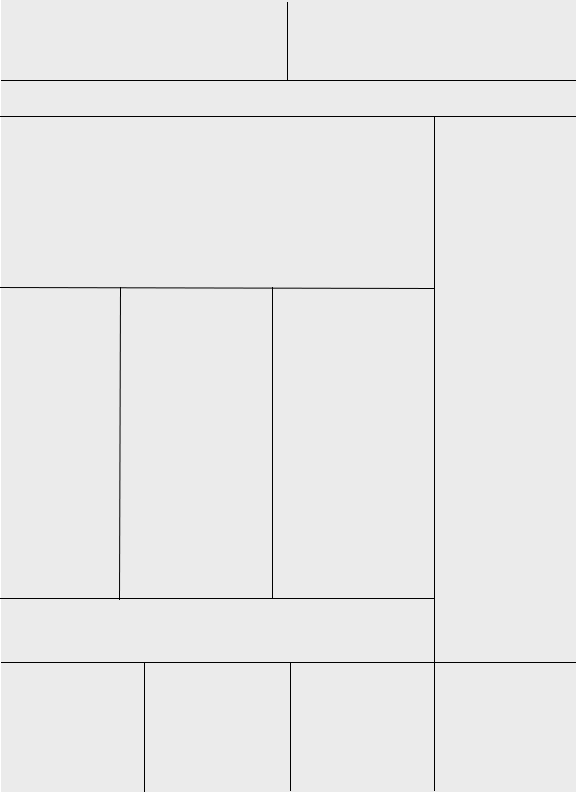
1. ***colspan8, colspan4 –*** *colspan8 means that it gets* ***8cells*** *from the cells12 while colspan4 gets only 4 cells.*

*As you can see from the figure 6, the third row’s columns has different sizes from each other. The first column is wider than the second one.*

There is another layout exclusive for the other subpages. This is a little complicated compared to other layout because **there is a row inside a column**. In this way, we can add another series of columns. (*see Figure 7 – Sub-page layout (2)).*You can see this layout to pages Quick Stat, Agricultural Situationer, Stattrivia etc. The three columns are used for the divisions of Years. Under each year there are articles.

You can insert the following code inside any of the cells or columns (e.g<div class="cell colspan8">…</div>). In figure 7’s case, the code will be added at the third row’s first column. As you can see, there are three class="cell colspan4"under the row. This means that you are going to add three columns inside the row.

Figure 7. Sub-page layout (2)



*This part is the* ***row*** *added. Inside the row there are* ***three columns****.****See figure 6*** *to know the difference.*

\*The code should look like this:

<div class="cell colspan8">

<div class="row cells12">

<div class="cell colspan4">...</div>

<div class="cell colspan4">...</div>

<div class="cell colspan4">...</div>

</div>

</div>

*This part is the code you are going to add inside a column* (<div class="cell colspan8">*) to* ***create a row inside the column****. Inside the row you created, you can create columns. In the layout above (figure 7). There are three columns. So, this code has* ***three cell****.*

*\*Note: You can* ***nest*** *the rows and columns.*

The screenshots below shows the actual layout of the page:

1. **Sub-page layout (1)**– You can see this layout in the majority of the pages.

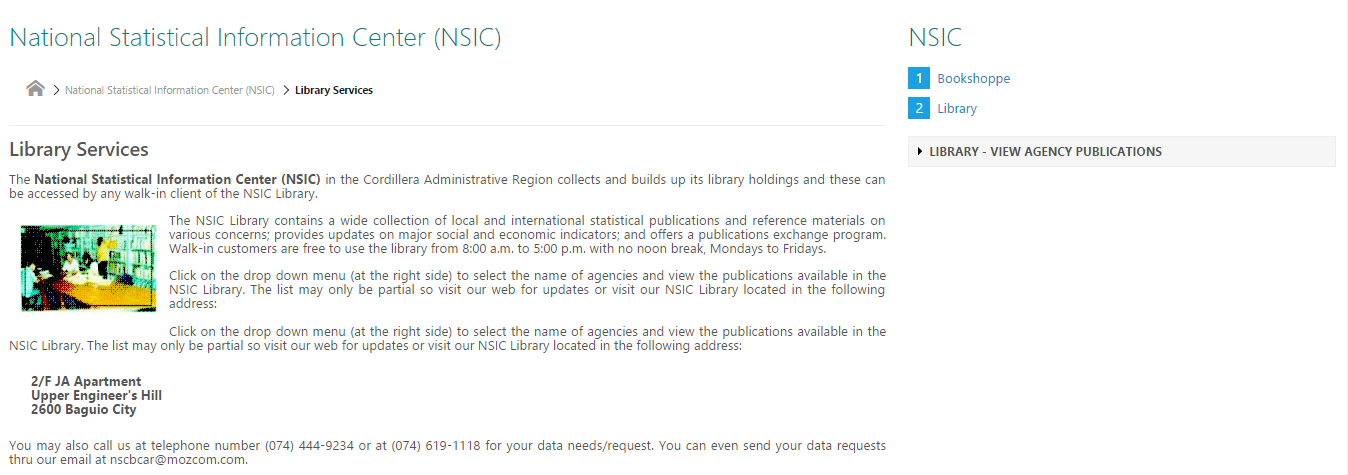
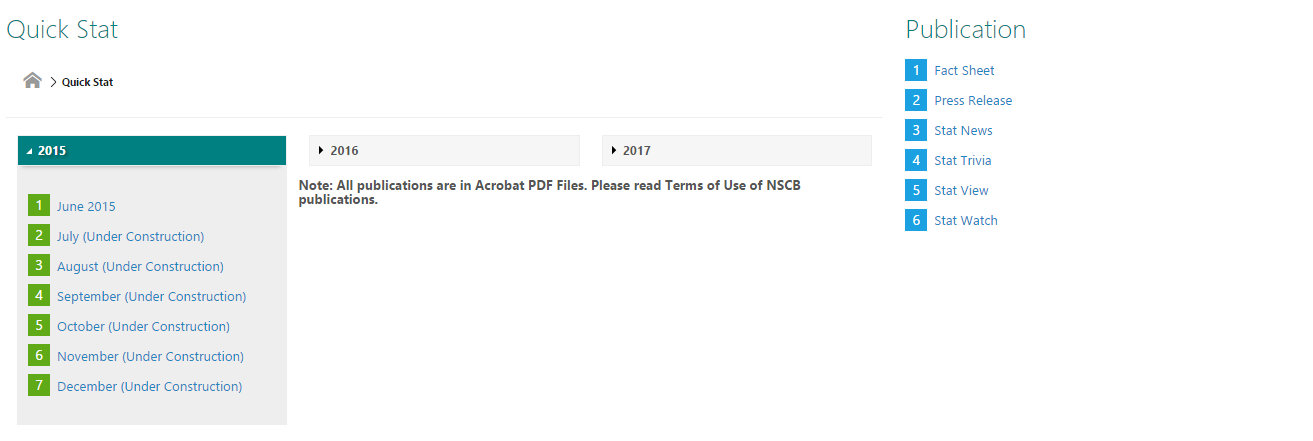


Figure 8. Sub-page layout (1) example.

*First Column*

*Second Column*

1. **Sub-page layout (2)**– Imagine the above layout (*sub-page layout (1) example*). Add a row inside its first column and you get to create this layout but don’t forget adding cells or columns inside your row to completely do the following layout.



*First Column*

*Second Column*

*Third Column*

***These three columns are inside the row.***

Figure 9. Sub-page layout (2) example.

\*The whole code for the sub-page layout 2 should look like this:

<div class="grid">

<div class="row cells12">

<div class="cell colspan6">*…*</div>

<div class="cell colspan6">…</div>

</div>

<div class="row">…</div>

<div class="row cells12">

<div class="cell colspan8"

<div class="row cells12">

<div class="cell colspan4">...</div>

<div class="cell colspan4">...</div>

<div class="cell colspan4">...</div>

</div>

</div>

<div class="cell colspan4">…</div>

</div>

<div class="row cells12">

<div class="cell colspan3">…</div>

<div class="cell colspan3">…</div>

<div class="cell colspan3">…</div>

<div class="cell colspan3">…</div>

</div>

</div>

# **Metro UI CSS FrameworkClasses, Componentsand JS**

When you **followed the start-up**, you can already use the available classes and JavaScript codes provided by the Metro UI CSS Framework **to style your html elements**. The framework also provides declarative codes to **create components** such as navigations, tiles, accordion, table and etc. You can see available components by visiting their website <http://metroui.org.ua/> or by downloading its documentation found in their website if you intend to code without using internet connection. This can also be helpful if you have no stable or consistent internet connection.

1. **Colors**- To set background color: use prefix bg-, to set foreground color: use prefix fg-, to set outline color: use prefix ol-, to set border color: use prefix bd-. Also you can set active, hover and focus colors with classes bg(fg)-hover-\*, bg(fg)-active- and bg(fg)-focus-. Also you can change :before and :after color and background for element with classes: before-bg-\*, before-fg-\*, after-bg-\*, after-fg-\*.

***\*Note: The colors provided are limited. You can see the list of colors available in their website or documentation. If you think you miss any color, you can modify or add colors as many as you want in the framework’s CSS (metro.css).***

\*The example below uses **class bg-\* and fg-\*.** The code for this should be:



<div class="bg-darkTealfg-white">

<span>Back to top</span>

</div>

1. **bg-darkTeal** – sets the div element’s background color.
2. **fg-white** – sets the text’s color inside the div element.

***\*note: Remember that if you want to use the classes, you should always put in the attribute “class”. You can add as many classes as you want.***

1. **Fonts (Metro Icon Fonts)and Typography**

**Typography**

|  |  |
| --- | --- |
| @fontName: | "Segoe UI", "Open Sans", serif; |
| @fontNameLight: | "Segoe UI Light", "Open Sans Light", serif; |
| @fontNameBold: | "Segoe UI Bold", "Open Sans Bold", serif; |

*\*The signature user interface font of Metro UI CSS, Segoe, Open Sans and PT Serif Caption. It's a sans serif design drawn in the tradition of signage and way-finding typefaces. For non-windows system used Open sans fonts, loaded from google fonts.*Paragraphs can be styled using these classes:

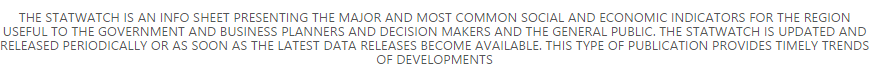
* 1. text-default
  2. text-accent
  3. text-secondary
  4. text-small
  5. text-dashed
  6. lowercase
  7. uppercase
  8. capital
  9. align-left
  10. align-right
  11. align-justify
  12. align center

\*Example:

1. **Using classes align-center and uppercase.**

<p class=”**align-center upper-case**”>The StatWatch is an info sheet presenting the major and most common social and economic indicators for the region useful to the government and business planners and decision makers and the general public. The StatWatch is updated and released periodically or as soon as the latest data releases become available. This type of publication provides timely trends of developments</p>

The code above should look like this:



2. If you **add class fg-red**, it will look like this:

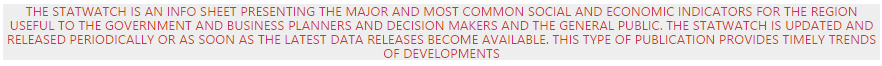
<p class=”align-center upper-case**fg-red**”>The StatWatch is an info sheet presenting the major and most common social and economic indicators for the region useful to the government and business planners and decision makers and the general public. The StatWatch is updated and released periodically or as soon as the latest data releases become available. This type of publication provides timely trends of developments</p>



As you can observe, it set its text color to **red**

3. if you **add classbg-grayLighter**, it will look like this

<p class=”align-center upper-casefg-red **bg-grayLighter**”>The StatWatch is an info sheet presenting the major and most common social and economic indicators for the region useful to the government and business planners and decision makers and the general public. The StatWatch is updated and released periodically or as soon as the latest data releases become available. This type of publication provides timely trends of developments</p>



As you can observe, it set its background color to **grayLighter**

**Fonts (Metro Icon Font)**

*Metro Icon Font gives you scalable vector icons that can instantly be customized — size, color, drop shadow, and anything that can be done with the power of CSS. Metro Icon Font includes over 400 icons.*

If you remembered at the start-up, you can see that you included this link<link href="metro-icons.css" rel="stylesheet"> inside the head element. This is to access the metro icon fonts available. Metro UI CSS Framework provides different categories of icons such as application icons, mobile icons, device icons, player icons, brand icons, weather icons, chart icons and spinner icons. The best part here is that it provides you the classes to animate your icons.

To get started in using the fonts, see the example below:

\*Example:

<span class="mif-*icon\_name*"></span>

***You can choose icons from their list. You can view them by visiting their website:*** [***http://metroui.org.ua/font.html***](http://metroui.org.ua/font.html) ***or by downloading their documentation for offline use.***

Copy the code above and change the “*icon-name*” to any icon you have chosen from the list. For example:

1. <span class="mif-file-pdf"></span>This code will result to:



1. <span class="mif-music"></span><span>music</span>This code will result to:



To **change size icon** you can use next classes: **mif-lg, mif-2x, mif-3x, mif-4x.**

1. <span class="mif-earth mif-2x"></span>



To **change icon color** you can use **fg-\*** classes or use css property **color**.

1. **class**: <span class="mif-earth mif-2xfg-green"></span>
2. **css property**:<span class="mif-earth mif-2x"style=”color:green;”></span>



The framework provides different kinds of how the icons are going to be animated. Since this is document file, there is no way to show you how the animations of icons look like. Visit this link to see: [***http://metroui.org.ua/font.html***](http://metroui.org.ua/font.html)***.*** To use the animation, see the code below.

\*To use font animation:

<span class="mif-earth mif-2xfg-greenmif-ani-spin"></span>This code will result to **mif-earth** icon to **spin**.

<span class="mif-heart mif-ani-heartbeat"></span>This code will result to **mif-heart** icon to **beat**.

**You can check the animations in the pages: Agricultural Situationer and Quick Stat.**

### **Accordion**

\*Note: You can **use class colors** that we discussed previously.

This is how **accordion** looks like. The first image shows only a 2015 with its background. But when the time you click it, it will show its contents. Click it again and it will revert back to its original form. This component is best when you a minimalist type of person. You don’t want to make your whole page look like a book showing all of its content.

Accordion really saves space and it looks clean instead of bombarding the user with all the contents when he only need a portion to see.

See below to know how to code this using the framework.

*You see a types of accordion by visiting their web site:* [*http://metroui.org.ua/accordion.html*](http://metroui.org.ua/accordion.html) *or by downloading*



Figure 10. Accordion

The **red marks** are the **key classes** you need to create your desired accordion.

1. **class=”accordion”** – this is the important class so don’t miss it. This is to **initialize** the accordion.
2. **class=”frame”** – this is where you write your **title** for your accordion.
3. **class=”content”** – this is where you **write your contents** inside the accordion.

**\*This is how to code accordion using the framework.**

<div class="accordion">

<div class="frame">

<div class="heading">2015</div>

<div class="content">

<li>June 2015</li>

<li>July</li>

<li>August</li>

<li>September</li>

<li>October</li>

<li>November</li>

</div>

</div>

</div>

**\*To create 2 or more accordion, there are two ways**

1. Create this code below two times

<div class="accordion">

<div class="frame">

<div class="heading">2015</div>

<div class="content">

<li>June 2015</li>

<li>July</li>

<li>August</li>

<li>September</li>

<li>October</li>

<li>November</li>

</div>

</div>

</div>

2. Copy the whole <div class="frame">and the other elements inside it then paste under the end tag of the <div class="frame">.

<div class="accordion">

<div class="frame">

<div class="heading">2015</div>

<div class="content">

<li>June 2015</li>

<li>July</li>

<li>August</li>

<li>September</li>

<li>October</li>

<li>November</li>

</div>

</div>

<div class="frame">

<div class="heading">2014</div>

<div class="content">

<li>June 2015</li>

<li>July</li>

<li>August</li>

<li>September</li>

<li>October</li>

<li>November</li>

</div>

</div>

</div>

### **Buttons**

*Button styles can be applied to anything with the* ***.button class*** *applied. However, typically you'll want to apply these to only* **<a>** *and* **<button>***elements for the best rendering.*

There are **two ways** to create a button:

1. By using **<a>** element

<a **class="button**bg-darkVioletfg-white">Button</a>



1. By using <button> element

<button **class="button**bg-orange fg-white">Click Me!</button>



In the website, buttons are also used in pages such as Statview.



The above images are coded like this:

<button **class="button**>Fire Statistics, CAR: 2010-2013</button>

<button **class="button"**>Fireworks-related Injuries by Province, CAR: 2010-2012 </button>

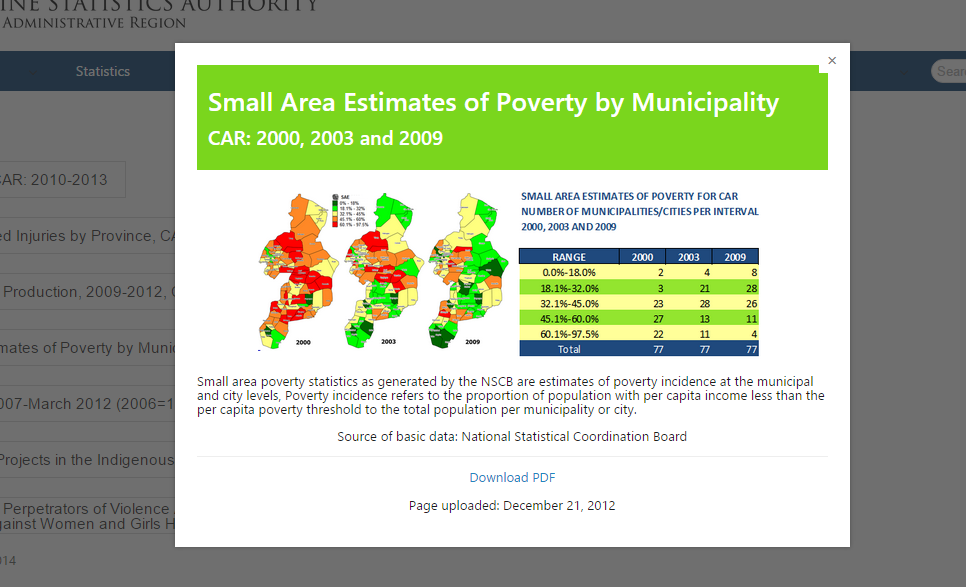
The length of the button is relative to how long the text you are going to write inside the button element.

### **Dialog or Modal**

*Metro UI CSS provides the ability to easily create dialogs. The dialogue can be ANY Your unit, it is sufficient to determine the appropriate role***data-role='dialog'***.*

When you click a button, or anything within the element **<a>.** Example of this component can be found in statview page. The image below shows how it works

Figure 11. Dialog



When you click the button (first image), the second image shows up. The second image is what you call a modal or dialog (in Metro UI CSS Framework).

You can do this with few simple steps using the framework. As you see above images, the first one is a button.

1. So first create a <button>, <div> or <a> elements.

<button class=”button”>Small Area Estimates of Poverty by Municipality</button>or

<div>Small Area Estimates of Poverty by Municipality</button> or

<aclass=”button”>Small Area Estimates of Poverty by Municipality</a>

1. Create the div that will show up when the button is clicked. You can place this part anywhere in you html because it will not be rendered by a web browser.

<div **data-role="dialog"data-overlay="true" data-overlay-color="op-dark" id="dialog"**>

<h1> Small Area Estimates of Poverty by Municipality </h1>

<p>

Put your contents here.

</p>

</div>

1. **data-role=”dialog”** – this means that is a dialog hence is not going to show up unless called.
2. **data-overlay=”true”** – this means that when this div is called you cannot click anything outside the dialog.
3. **data-overlay-color=”op-dark”** – the color outside the dialog
4. **id=”dialog”** – name your id to identify what dialog will be shown.
5. Copy this JQuery code. Just copy and paste anywhere in you html code. This script calls the dialog (see step #2) to show up.

<script>

functionshowDialog(id){

var dialog = $(id).data('dialog');

dialog.open();

}

</script>

1. Go back to step 1 and modify the code.

<button class=”button” onclick="showDialog('#dialog)">Small Area Estimates of Poverty by Municipality</button>or

<divonclick="showDialog('#dialog)">Small Area Estimates of Poverty by Municipality</button> or

<a class=”button” onclick="showDialog('#dialog)">Small Area Estimates of Poverty by Municipality</a>

onclick=”showDialog(“#dialog)” – this triggers the dialog to show up whenever its clicked.

#dialog – you can name it in any way you want. Just don’t drop the # sign. This is the same with the id in step #3.

1. **Menu(Navigation)**

*The framework provides tools to create different cool menu, as horizontal, vertical, tool and any menu types. All menus have a similar structure and are created using lists, list items and anchors. Menu is determined by the class applied to the root list. For drop down menu use the role dropdown.*

<ul class="f-menu">

<li><a href="#">Home</a></li>

<li><a href="#">File</a></li>

<li><a href="#">Products</a></li>

<li>

<a href="#" class="dropdown-toggle">More...</a>

<ul class="d-menu" data-role="dropdown">

<li><a href="#">About</a></li>

<li><a href="#">Partners</a></li>

</ul>

</li>

<li class="place-right no-hovered">

<form>

<div class="input-control text" style="width: 250px; margin-right: 10px">

<input type="text" placeholder="Search...">

<button class="button"><span class="mif-search"></span></button>

</div>

</form>

</li>

</ul>

### **List**

Classes used:

1. **numeric-lis**t–this is a class that makes the list in **ordered**numbers with background color.

***Note****: You can change this to* ***simple-list*** *which has no number to it. But keep in mind that you should change the list element to <ul>*

1. **square-marker** – this makes the **background** of numbers to be in **square**. By removing this class, it will change into circle.
2. **blue-bullet**–this **sets** the **color** of the numbers’ background color.

***Note****: You can change this colors with: black-bullet, green-bullet, red-bullet and yellow-bullet.*

This is how you code figure 12.

<ol class="numeric-list square-marker blue-bullet">

<li>Introduction</li>

<li>FAQs</li>

<li>Indicators</li>

<li>Regional Datasets</li>

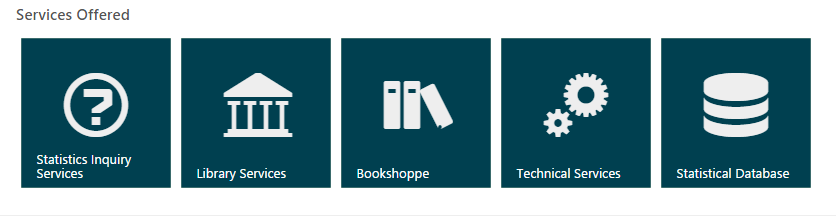
<li>MDG Watch</li>

</ol>

Figure 12. List



### **Tiles**

****

**Tiles** helps in visualization and lessen texts in a page which make the users get bored in visiting pages.

To **createtiles** using the framework:

1. With **icon** and **label.** (*figure 13*)

<div class="tile-content iconic tile-square">

<span class="icon mif-envelop"></span>

<span class="tile-label">Mail</span>

</div>

1. With **image with slide-up label.** (*Figure 14)*

<div class="tile-wide data-role="tile">

<div class="tile-content slide-up">

<div class="slide">

<img width="100%" height="100%" src="/images/pcpcOnline.gif">

</div>

<div class="slide-over op-black fg-white padding10">

Philippine Central Product Classification Online

</div>

</div>

</div>

1. If you want to **create a group of tile,** you should enclose them inside a tile-container.

<div class="tile-container">...</div>

***\*Note: If you want to understand more, you should visit their website and explore it or for offline use, download their documentation***[***http://metroui.org.ua***](http://metroui.org.ua)***.***

Figure 13. Tile with icon

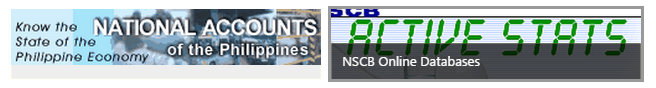


Figure 14. Tile with slide-up label.

**With image and slide-up label**

1. **tile-wide** – This tile will be a rectangle.
2. **data-role=”tile”** – this means that the div will be a tile.
3. **slide-up**– when hovered, the label will show up at the bottom part.

**With icon and label**

1. **tile-content** – this div houses the content of the tile.
2. **iconic** – this means that it should have an icon inside.
3. **tile-square** – the size of this square tile is 150px by 150px
4. **icon** – means that the icon to be placed will be at the center of the tile.
5. **tile-label** – this will place the label of the tile.

### **Tables**

Tables helps sort out data and figure out easily what they represent. Classes for tables are also provided by the framework.

Classes used:

1. **table**– before adding any more classes, this class should be added first.
2. **text-small** – this makes the texts inside the table are small.
3. **hovered** – this makes the rows to change colors when hovered.
4. **bd-grayLighter** – every borders will be colored as grayLighter.
5. **bordered** – means that the table will have borders for every cell

Figure 15. Table

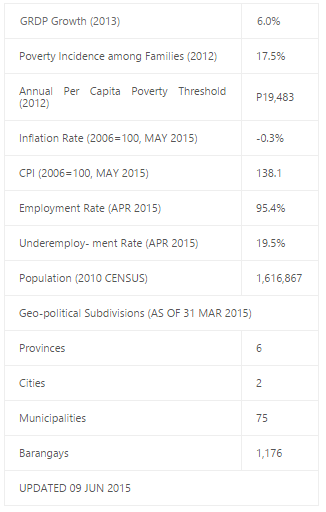


Figure 15 can be coded like this:

<table class="table text-small hovered border bd-grayLighter no-padding bordered">

<tr>

<td class="bd-grayLighter">GRDP Growth (2013)</td>

<td class="bd-grayLighter">6.0%</td>

</tr>

<tr>

<td class="bd-grayLighter">Poverty Incidence among Families (2012)</td>

<td class="bd-grayLighter">17.5%</td>

</tr>

</table>

1. **Breadcrumbs**

Classes used:

1. **breadcrumbs –** a class used to a <ul> tag to make a breadcrumb trail.
2. **icon** – means that the icon to be placed will be at the center of the tile.
3. **mif-home –** used to call the home icon ()

A breadcrumb trail is a navigation aid used to allow users to keep track of their locations within the website.



Figure 16

Figure 16 can be coded like this:

<ul class="breadcrumbs">

<li><a href="/index.html"><span class="icon mif-home"></span></a></li>

<li><a href="/news\_and\_releases/pov\_map/pm\_index.html">Poverty Mapping</a></li>

<li><a href="/news\_and\_releases/pov\_map/provincial/Mt.Province/mtprov\_index.html">Mt. Province</a></li>

<li>Crime Against Property</li>

</ul>

1. **Includes**

Methods used:

1. **$(document).ready(*function*);**

This is used so the ***function***inside will execute if and only if the document is ready.

1. **$(#*id*).load(“*location/sample.html*”);**

This is used to call the separate html file and insert it to a tag that has the same id as the one indicated in the **#id*.***

This is used to call a separate html file and insert it to the designated div tag.

This code is required at the header of the html file:

<script src="/js/jquery-2.1.3.min.js"></script>

\*Jquery-2.1.3.min.js must be located correctly.

Sample:

<div id="mtprov">

<script>

$(document).ready(function(){

$("#mtprov").load("/includes/PM/mtprov\_drop.html");

});

</script>

</div>

*\*In the example, the content of* ***mtprov\_drop.html*** *will be inserted in the* ***div*** *tag that has the id “****mtprov.”***

# **Bibliography**

Pimenov, S. (2012-2015). *Metro UI CSS 3.0*. Retrieved from Metro UI CSS 3.0: http://metroui.org.ua/