Handling PHP Files in Zurb Template

I have some sites that consist primarily of html files but have a few php files associated with the validation of form data. I used the Zurb template to build the sites. The Zurb template was designed to handle html web pages. The following describes the steps I take to handle php files with the Zurb template.

1. Files containing any php in src/pages, src/partials, and src/layouts are given the extension
   .php.

2. To handle php files in src/partials and src/layouts make the change (in red) to the lines

   var partials = utils.loadFiles(dir, '/*.{html,php,hbs,handlebars}');**

   in the file loadPartials.js and

   var layouts = utils.loadFiles(dir, '/*.{html,php,hbs,handlebars}');**

   in the file loadLayouts.js.

   These files are in the directory node_modules/panini/lib.

3. Pure php files that are included in other files using the php commands include or require are stored in src/assets/php. You will need to create this folder. They will be transferred unmodified to dist/assets/php. A typical php call would be

   require '{{root}}assets/php/file.php';

4. I use a Wamp server to render php files on my Windows PC. For each of my web sites I create a virtual host (site1, site2, etc.). These virtual hosts point to the /dist folder associated with the site. The sites are accessed by calling http://site1, http://site2, etc. In the appendix I show how to set up a virtual host on Windows using the Wamp server.

5. Assuming that my virtual host is named site1, I make the following changes to gulpfile.babel.js that is located in the main folder of the web site (changes in red):

```
...

// Copy page templates into finished HTML files
function pages() {
  return gulp.src('src/pages/**/*.{html,php,hbs,handlebars}')
    .pipe(panini({
      root: 'src/pages/',
      layouts: 'src/layouts/',
      partials: 'src/partials/',
      data: 'src/data/',
      helpers: 'src/helpers/'
    }))
    .pipe(gulp.dest(PATHS.dist));
}
```

...

```
  // Start a server with BrowserSync to preview the site in
  function server(done) {
    browser.init({
      // server: PATHS.dist, port: PORT
      proxy: "http://site1"
    }, done);
  }
```

...

```
  // Watch for changes to static assets, pages, Sass, and JavaScript
  function watch() {
    gulp.watch(PATHS.assets, copy);
    gulp.watch('src/pages/**/*.{html,php}').on('all', gulp.series(pages,
    browser.reload));
    gulp.watch('src/{layouts,partials}/**/*.{html,php}').on('all',
    gulp.series(resetPages, pages, browser.reload));
```

...

6. The {{#ifpage }} handlebar helper is very handy, but it only works for html files. To test if the current page is file1.php I set a variable **file1: true** in the front matter of file1.php and use the test

   {{#if file1}} **...** {/if}}

   It is not necessary to set a variable **file1: false** in the other pages as undefined is treated as false.

7. If you want to use the same layout for all files in a subdirectory of pages, you can add a pageLayouts option to the pages function in gulpfile.babel.js. For example, if you wanted to use layout1.html for all files in subdirectory1 and layout2.html for all files in subdirectory 2, you could write

```
function pages() {
return gulp.src('src/pages/**/*.{php,html,hbs,handlebars}')
  .pipe(panini({
     root: 'src/pages/',
     layouts: 'src/layouts/',
     pageLayouts: {
      'subdirectory1': 'layout1',
      'subdirectory2': 'layout2'
    },
    partials: 'src/partials/',

...
```

Appendix: How to set up a virtual host associated with a Wamp server

You can download the installer for a Wamp server from **https://sourceforge.net/projects/wampserver/.** Run the installer and follow instructions to set up the Wamp server. To start the server run the executable C:\wamp64\wampmanager.exe. An icon will appear in the task area. It will be red at first, then yellow, and finally green. Green signifies that the server is ready. To create a virtual host left click on the Wamp icon. In the pop up menu select **Your VirtualHosts** and then **VirtualHost Management**. A dialog box will appear. In the first input box type the name you want to call the virtual host. In the second input box type the path to the /dist folder associated with the desired site. Next press the button **Start the creation of the VirualHost (May take a while ...)**. When the process is completed it will be necessary to restart the server for the changes to take place. You should also restart the DNS server by right clicking on the wamp icon and selecting **Tools** followed by **Restart DNS**. If you go back to the VirtualHost Mangement dialog, it will list all the virtual hosts that have been created.

The site associated with the virtual host can be accessed with the URL http://HostName (HostName being replaced by the name you selected for the virtual host).

What the above procedure really does is modify two text files. Before looking at these files make sure that the lines

LoadModule vhost_alias_module modules/mod_vhost_alias.so

and

Include conf/extra/httpd-vhosts.conf

are uncommented in the file
C:\wamp64\bin\apache\apachex.x.xx\conf\httpd.conf (x.x.xx is the apache version number).

If the virtual host name is siteName that points to C:\mySite\dist, then wamp64 adds the lines

```
<VirtualHost *:80>
  ServerName siteName
  DocumentRoot " C:/mySite/dist "
  <Directory  " C:/mySite/dist/">
    Options +Indexes +Includes +FollowSymLinks +MultiViews
    AllowOverride All
    Require local
  </Directory>
</VirtualHost>
```

to the file
C:\wamp64\bin\apache\apachex.x.xx\conf\extra\httpd-vhosts.conf
(x.x.xx is the apache version number). In most systems the Options under Directory are not needed, but they won't cause any harm. It also adds the lines

```
127.0.0.1      siteName
::1      siteName
```

to the file **C:\Windows\System32\drivers\etc\hosts** .

The user could add these lines manually, but it is easier to allow wamp64 to make the changes. Note that the virtual host **localhost** is always required, i.e.,

```
<VirtualHost *:80>
  ServerName localhost
  ServerAlias localhost
  DocumentRoot "${INSTALL_DIR}/www"
  <Directory "${INSTALL_DIR}/www/">
    Options +Indexes +Includes +FollowSymLinks +MultiViews
    AllowOverride All
    Require local
  </Directory>
  </VirtualHost>
```

as is the entry

```
127.0.0.1 localhost
::1 localhost
```