Tutorial - Create a New Paper Lantern Interface

Introduction

This tutorial uses Template Toolkit to create a new cPanel interface for the Paper Lantern theme.

The examples below use the Paper Lantern theme's master template, which creates new interfaces that integrate seamlessly with the look and feel of the main cPanel interface.

> **Note:**
>
> For help to create a new cPanel interface in PHP, read our Create a New Paper Lantern Interface in PHP tutorial.

Create a new Paper Lantern interface

Create a Template Toolkit file with the Paper Lantern header and footer.

To begin, create a new Template Toolkit file. This file **must** use the `.html.tt` file extension.

Use Template Toolkit's `WRAPPER` directive to add Paper Lantern's header and footer to the template file.

```
[% WRAPPER '_assets/master.html.tt' -%]
[% END %]
```

Add the interface's header and image.

Add the `app_key` attribute to the `WRAPPER` directive. The `app_key` value **must** match the interface's `file` value in the `dynamicui.conf` file.

```
[% WRAPPER '_assets/master.html.tt'
    app_key = 'backup'               # The file value from the
dynamicui.conf file.
-%]
[% END %]
```

In this tutorial

This tutorial uses the following techniques, languages, tools, or concepts:

- Template Toolkit
- Perl
- JavaScript

Use this tutorial when you develop:

- cPanel Plugins

Related documentation

- UAPI - The Cpanel::Args Object — UAPI modules use the `Cpanel::Args` object to receive arguments (input parameters).
- UAPI - Custom UAPI Modules — You can add custom modules to cPanel & WHM and call them with UAPI.
- UAPI - Create Your Module — To create a custom module, you must first create the `Module.pm` file.
- UAPI - Custom Function Basics — Custom modules contain one or more individual functions.
- Guide to cPanel Interface Customization - Apply Styles — To display a style for a particular user in the cPanel interface, you **must** apply that style to the account.

Add the interface's title.

Add the `page_title` attribute to the `WRAPPER` directive. You can set any string as the `page_title` value.

```
[% WRAPPER '_assets/master.html.tt'
    app_key = 'backup'               # The file value from the
dynamicui.conf file.
    page_title = "Cloud Backup"      # The page title.
-%]
[% END %]
```

Add stylesheets.

Add the `page_stylesheets` attribute to the `WRAPPER` directive. Use this attribute to apply one or more CSS stylesheets to the interface.

Enter the CSS files' paths relative to the `/usr/local/cpanel/base/frontend/paper_lantern/` directory.

```
[% WRAPPER '_assets/master.html.tt'
    app_key = 'backup'               # The file value from the
dynamicui.conf file.
    page_title = "Cloud Backup"      # The page title.
    page_stylesheets = [             # CSS files.
        'backup/cloud_backup.css',
        'backup/controller_styles.css'
    ]
-%]
[% END %]
```

Add scripts.

Add the `page_scripts` attribute to the `WRAPPER` directive. Use this attribute to call one or more scripts that will run when the interface loads.

Enter the scripts' paths relative to the `/usr/local/cpanel/base/frontend/paper_lantern/` directory.

```
[% WRAPPER '_assets/master.html.tt'
    app_key = 'backup'               # The file value from the
dynamicui.conf file.
    page_title = "Cloud Backup"      # The page title.
    page_stylesheets = [             # CSS files.
        'backup/cloud_backup.css',
        'backup/controller_styles.css'
    ]
    page_scripts = [                 # Scripts.
        'custom/example.js',
    ]
-%]
[% END %]
```

Notes:

- When you include jQuery in an external file, you can call it normally via Template Toolkit's `page_scripts` attribute. For example, the following code calls the `example.min.js` file:

  ```
  page_scripts = [                 # Scripts.
      'custom/example.js',         # The system
  transforms this file to custom/example.min.js,
                                   # which requires
  jQuery in a delayed RequireJS block.
  ]
  ```

  For more information about how to use the `page_scripts` attribute, read our Create a New Paper Lantern Interface tutorial.
- If you use the `page_scripts` attribute to load exernal files, the external files **must** use the `.min.js` file extension. However, the filename that you specify to the `page_scripts` attribute will only use the `.js` file extension. cPanel's master template automatically transforms the `.js` extension in the `page_scripts` attribute to the `.min.js` extension.
- cPanel's Paper Lantern interface incorporates the RequireJS module loader. RequireJS may conflict with the webpack module loader. We **strongly** recommend that you do not use webpack with any cPanel plugin or custom interface that you develop.

Add styles.

Add the `page_styles` attribute to the `WRAPPER` directive. Use this attribute to add specific styles for use in your template.

In the example below:

- Line 11 adds the `css_code_block` style.
- Lines 14 through 20 add a `css_block` style.

```
[% WRAPPER '_assets/master.html.tt'
    app_key = 'backup'               # The file value from the
dynamicui.conf file.
    page_title = "Cloud Backup"      # The page title.
    page_stylesheets = [             # CSS files.
        'backup/cloud_backup.css',
        'backup/controller_styles.css'
    ]
    page_scripts = [                 # Scripts.
        'custom/example.js',
    ]
    page_styles = css_code_block     # Use the css_code_block
style.
-%]

[% BLOCK css_block %]
<style>
.help {
    padding: 10px;
}
</style>
[% END # css_block END %]
[% END # WRAPPER END %]
```

Add JavaScript.

Add the `page_js` attribute to the `WRAPPER` directive. Use this attribute to apply a JavaScript block to your template.

In the example below:

- Lines 1 and 2 process the `css_block` and `js_block` blocks.
- Line 14 adds the `js_block` script to the template.
- Lines 25 through 36 contain the `js_block` script.

> **Note:**
>
> This example includes code that uses jQuery and wraps it in a `define()` block. In cPanel & WHM version 54 and higher, you **must** wrap jQuery code in a `define()` block. For more information, read our Guide to cPanel Interface Customization - jQuery documentation.

```
[% css_code_block = PROCESS css_block %]
[% js_code_block = PROCESS js_block %]
[% WRAPPER '_assets/master.html.tt'
    app_key = 'backup'               # The file value from the
dynamicui.conf file.
    page_title = "Cloud Backup"      # The page title.
    page_stylesheets = [             # CSS files.
        'backup/cloud_backup.css',
        'backup/controller_styles.css'
    ]
    page_scripts = [                 # Scripts.
        'custom/example.js',
    ]
    page_styles = css_code_block     # Use the css_code_block
style.
    page_js = js_code_block          # Use the js_block script.
-%]

[% BLOCK css_block %]
<style>
.help {
    padding: 10px;
}
</style>
[% END # css_block END %]
[% BLOCK js_block %]
<script>
    define(
        [
            "jquery"
        ],
        function($) {
            // Code that uses jQuery goes here.
            $(document).ready(function() {
                $("#div_id").html("Text to display.");
            });
        });
</script>
[% END # js_block END %]

[% END # WRAPPER END %]
```

Disable legacy cPanel styles, scripts, and CJT scripts.

Set the `include_legacy_stylesheets`, `include_legacy_scripts`, and `include_cjt` attributes in the `WRAPPER` directive to `0` to exclude cPanel's legacy styles, scripts, and CJT scripts from the template.

This ensures that your template uses Paper Lantern's functionality, rather than styles and scripts from our older x3 theme.

```
[% css_code_block = PROCESS css_block %]
[% js_code_block = PROCESS js_block %]
[% WRAPPER '_assets/master.html.tt'
    app_key = 'backup'               # The file value from the
dynamicui.conf file.
    page_title = "Cloud Backup"      # The page title.
    page_stylesheets = [             # CSS files.
        'backup/cloud_backup.css',
        'backup/controller_styles.css'
    ]
    page_scripts = [                 # Scripts.
        'custom/example.js',
    ]
    page_styles = css_code_block     # Use the css_code_block
style.
    page_js = js_code_block          # Use the js_block script.
    include_legacy_stylesheets = 0   # Exclude legacy yui
stylesheets.
    include_legacy_scripts = 0       # Exclude yui utilities and
x3 optimized scripts.
    include_cjt = 0                  # Exclude legacy cjt
scripts.
-%]

[% BLOCK css_block %]
<style>
.help {
    padding: 10px;
}
</style>
[% END # css_block END %]
[% BLOCK js_block %]
[% BLOCK js_block %]
<script>
    define(
        [
            "jquery"
        ],
        function($) {
            // Code that uses jQuery goes here.
            $(document).ready(function() {
                $("#div_id").html("Text to display.");
            });
        });
</script>
[% END # js_block END %]

[% END # WRAPPER END %]
```

Set the directory prefix cPanel variable.

Set the `CPANEL.CPVAR.dprefix` variable. This ensures that links to other cPanel interfaces, search results from the cPanel interface's search menu, and certain menus function correctly in cPanel & WHM version 11.52 and later.

Line 20 sets the `CPANEL.CPVAR.dprefix` variable.

```
[% css_code_block = PROCESS css_block %]
[% js_code_block = PROCESS js_block %]
[% WRAPPER '_assets/master.html.tt'
    app_key = 'backup'               # The file value from the
dynamicui.conf file.
    page_title = "Cloud Backup"      # The page title.
    page_stylesheets = [             # CSS files.
        'backup/cloud_backup.css',
        'backup/controller_styles.css'
    ]
    page_scripts = [                 # Scripts.
        'custom/example.js',
    ]
    page_styles = css_code_block     # Use the css_code_block
style.
    page_js = js_code_block          # Use the js_block script.
    include_legacy_stylesheets = 0   # Exclude legacy yui
stylesheets.
    include_legacy_scripts = 0       # Exclude yui utilities and
x3 optimized scripts.
    include_cjt = 0                  # Exclude legacy cjt
scripts.
-%]

[% SET CPANEL.CPVAR.dprefix = "../" %]

[% BLOCK css_block %]
<style>
.help {
    padding: 10px;
}
</style>
[% END # css_block END %]
[% BLOCK js_block %]
<script>
    define(
        [
            "jquery"
        ],
        function($) {
            // Code that uses jQuery goes here.
            $(document).ready(function() {
                $("#div_id").html("Text to display.");
            });
        });
</script>
[% END # js_block END %]

[% END # WRAPPER END %]
```