

BlueUpCode Guide for HTML Template

Get starting

1. First, you must have [Node.js](#) installed on your computer.
2. Extract the downloaded package from the marketplace.
3. Start terminal and go to `/template/` directory from the package that was extracted.
4. This template uses **Gulp** as a task runner that helps you automate build tasks in the development workflow. To install Gulp globally, you can run `npm install --global gulp-cli`.
5. Install all dependencies needed by running `npm install`.
6. Run `gulp --version` to verify that Gulp is successfully installed, and the version of installed Gulp will appear and make sure you have installed Gulp 4.x version.
7. This template requires Firebase for authentication, you must set the configuration. Create new project on [Firebase console page](#) and get your Firebase configuration on **Project Settings -> General -> Your Apps** and copy the configuration into `/src/pages/components/variables.njk`
8. Run `gulp` on the terminal to build all assets and start the development server..
9. The server will automatically build assets and reload the browser if the source codes are changed. Press `CTRL+C` to stop the server.

Directory structure

This template comes with a simple and organized directory structure for easy to understand and maintainability. Look at the treeview below to find the basic template directory structure.

Template directory `/template/`

- **dist/** - contains compiled code and assets
 - **assets/** - contains all assets for supporting the pages
 - **app/** - contains compiled app scripts
 - **build/** - contains all compiled library assets
 - **scripts/** - contains compiled library JS files
 - **styles/** - contains compiled library CSS files
 - **fonts/** - contains fonts
 - **images/** - contains images
 - ***.html** - HTML pages from compiled Nunjucks
- **src/** - contains source code and assets
 - **app/** - contains source app scripts
 - **assets/** - contains all additional assets
 - **build/** - contains buildable library assets
 - **core/** - contains core component source codes
 - **vendors/** - contains custom 3rd party library source codes
 - **pages/** - contains source Nunjucks codes
- **tool/** - contains build tool scripts
- **config.json** - build configuration
- **gulpfile.js** - Gulp main script
- **package.json** - Package information

Build tools

This template uses Gulp to power the build tools. The build tool provides an easy way to organize, compile, and bundle assets. Below are all the build tool commands.

Command	Description
<code>gulp clean</code>	Delete <code>/template/dist/</code> directory
<code>gulp assets</code>	Copy all additional assets such as images, fonts that linked on <code>config.json</code> or inside <code>/template/src/assets/</code> into <code>/template/dist/assets/</code> directory
<code>gulp buildStyleCore</code>	Compile and bundle Bootstrap and core SCSS files that linked under <code>build.core.styles</code> object in <code>config.json</code> to <code>(ltr rtl)-core.css</code>
<code>gulp buildStyleVendor</code>	Compile and bundle 3rd party library SCSS files that linked under <code>build.vendors.optional.[package-name].styles</code> object in <code>config.json</code> to <code>(ltr rtl)-vendor.css</code>
<code>gulp buildStyle</code>	Run all SCSS build tasks: <code>gulp (buildStyleCore buildStyleVendor)</code>
<code>gulp buildScriptCore</code>	Bundle Bootstrap and core JS files that linked under <code>build.core.scripts</code> object in <code>config.json</code> to single file <code>core.js</code>
<code>gulp buildScriptMandatory</code>	Bundle mandatory library JS files that linked under <code>build.vendors.mandatory.[package-name].scripts</code> object in <code>config.json</code> to single file <code>mandatory.js</code>
<code>gulp buildScriptVendor</code>	Bundle 3rd party library JS files that linked under <code>build.vendors.optional.[package-name].scripts</code> object in <code>config.json</code> to single file <code>vendor.js</code>
<code>gulp buildScriptApp</code>	Transpile all JS files in <code>/template/src/app/</code> into <code>/template/dist/assets/app/</code> directory
<code>gulp buildScript</code>	Run all JS build tasks: <code>gulp (buildScriptCore buildScriptMandatory buildScriptVendor buildScriptApp)</code>
<code>gulp buildPage</code>	Compile all Nunjucks page codes in <code>/template/src/pages/public/</code> into <code>/template/dist/</code> directory. Page direction will be inherited from <code>config.direction</code> in <code>config.json</code>
<code>gulp buildPageLTR</code>	Compile all Nunjucks page codes with LTR direction
<code>gulp buildPageRTL</code>	Compile all Nunjucks page codes with RTL direction
<code>gulp build</code>	Run all build tasks <code>gulp (buildStyle buildScript buildPage assets)</code> parallely
<code>gulp serve</code>	Start the development server and watch source codes for live reloading. Press <code>CTRL+C</code> to stop the server

Configuration

All build tool configurations are located at `config.json` and you can customize the configuration.

Field	Type	Description
<code>config.port</code>	<code>integer</code>	Local development server port
<code>config.production</code>	<code>boolean</code>	Enable/disable production mode. By default, this option is <code>false</code> to speed up live reload. If you want to build for deployment, you must enable this option
<code>config.direction</code>	<code>string</code>	Page direction, you can set this option with <code>ltr rtl</code>
<code>config.sass_output_style</code>	<code>string</code>	CSS output structure, you can set this option with <code>nested expanded compact compressed</code>
<code>config.html_beautify</code>	<code>boolean</code>	Beautify compiled HTML files. You can disable this option if you want to increase build tool performance. This option will be automatically disabled if you enable production mode
<code>config.sourcemaps</code>	<code>boolean</code>	Enable/disable sourcemap. This option will be automatically disabled if you enable production mode
<code>config.skip</code>	<code>array</code>	An array of 3rd party libraries to be skipped from being compiled, you can skip the libraries under <code>build.vendors.optional</code> . example: <code>["apexcharts", "autosize", "datepicker"]</code>
<code>config.path</code>	<code>object</code>	Collection of paths
<code>config.output</code>	<code>object</code>	An object of build output file names
<code>build.core</code>	<code>object</code>	The object specifies Bootstrap and core buildable assets
<code>build.vendors</code>	<code>object</code>	The object specifies 3rd party library buildable assets
<code>build.vendors.mandatory</code>	<code>object</code>	Assets list under this node is required, you can't skipped to be compiled
<code>build.vendors.optional</code>	<code>object</code>	Assets list under this node can be skipped to be compiled

Page customization

This template uses Nunjucks to generate HTML pages. Nunjucks provides many useful features like template inheritance, variables, modulation, and more to make your development easier.

You can find all Nunjucks source codes in `/template/src/pages/` directory, look at the tree view below to understand the directory structure.

Nunjucks source code directory `/template/src/pages/`

- **components/** - contains all supporting components
 - **[component]/** - contains specific page components
 - **template.njk** - base template for inheriting
 - **variables.njk** - Nunjucks general variables
- **public/** - contains main pages that will be compiled

If you run the `gulp buildPage` command, all Nunjucks files inside `/template/src/pages/public/` will be compiled into `/template/dist/` directory. You can set the page direction by changing `config.direction` option in `config.json`

Don't make your pages from sketches, just start by customizing the prebuilt pages.

Tips: To set dark theme as default theme, you can set `defaults.theme` variable in `variables.njk` with `dark` value.

Maybe your code editor doesn't support the `jinj` syntax highlighting of Nunjucks. There are plugins to support `jinj` syntax highlighting for some code editors.

- atom <https://github.com/alohaas/language-nunjucks>
- vim <https://github.com/niftylettuce/vim-jinja>
- brackets <https://github.com/axelboc/nunjucks-brackets>
- sublime <https://github.com/mogga/sublime-nunjucks/blob/master/Nunjucks.tmLanguage>
- emacs <http://web-mode.org>
- vscode <https://github.com/ronnidc/vscode-nunjucks>

Stylesheet customization

This template uses SCSS for handling the CSS efficiently. You can find all Bootstrap and core SCSS files in `/template/src/build/core/styles/`. We remove several Bootstrap components because we replace them with our custom components. Find the custom 3rd party library SCSS files in `/template/src/build/vendors/(package-name)/styles/`.

You can easily customize the color palette and other options by editing the variables in `variables.scss`.

SCSS source directory structure `/template/src/build/core/styles/`

- **components/** - contains general components
- **forms/** - contains form components
- **helpers/** - contains helper classes
- **layout/** - contains layout components
- **mixins/** - contains SCSS mixins
- **utilities/** - contains utilities
- **vendor/** - contains 3rd party library
- **widgets/** - contains widgets
- **_components.scss** - concatenate all general components
- **_forms.scss** - concatenate all form components
- **_functions.scss** - helper functions
- **_helpers.scss** - concatenate all helper classes
- **_layout.scss** - concatenate all layout components
- **_mixins.scss** - concatenate all SCSS mixins
- **_reboot.scss** - reset default page style
- **_root.scss** - contains root variables
- **_theme.scss** - contains theme variables
- **_utilities.scss** - concatenate all utilities
- **_variables.scss** - global variables
- **_widgets.scss** - concatenate all widgets
- **index.scss** - the main file to be compiled and concatenate all parts

Script customization

There are 2 types of the JS scripts: **Bundlable script** and **App script**. Bundlable script is all library scripts that will be bundled to a single file. App script is the script that is written for a specific function and will be transpiled with Babel.

Bundlable script

Bundlable script is all JS files that are linked under **build** object in `config.json`. Bundlable scripts include all core and 3rd party library scripts, all the scripts will be bundled into `/template/dist/assets/build/scripts/`.

Find all core component scripts in `/template/src/build/core/scripts/`. This template takes 3rd party library scripts from `node_modules` directory, but several library scripts have been customized. All the custom 3rd party library scripts are located inside `/template/src/build/vendors/(package-name)/scripts/`.

App script

All app scripts are located in `/template/src/app/`, all scripts inside this directory will be compiled with Babel into `/template/dist/assets/app/`. These scripts are written for a specific function or page. You can use ES6 syntax for these scripts.

Authentication

This template uses Firebase for authentication features, you must set the configuration. Create a new project on the [Firebase console page](#) and get your Firebase configuration on **Project Settings -> General -> Your Apps** and copy the configuration into `/src/pages/components/variables.njk`.

If you don't need Firebase authentication, you can disable this feature by setting `enable_firebase: false` on `src/pages/components/variables.njk`.

Analytics

This template also supports analytics features with Google Analytics. To enable this feature, you must set `enable_analytics: true` and copy Google Analytics tracking code to `google_analytics_id: G-XXXXXXXXXX` on `src/pages/components/variables.njk`.