**[How to Make a Drupal 8 Theme](http://www.wikihow.com/Make-a-Drupal-8-Theme)**

Drupal is one of the top 3 Content Management Systems (CMS's) in the world, along with Wordpress and Joomla. Drupal 8 (not yet released) will be the latest version of Drupal when it is released. Drupal is customised using themes, which are a powerful way to enhance a Drupal-powered website by setting out the structure and appearance (and some additional functionality) of the site. This article will work through creating a Drupal 8 theme from start to finish.

Please note that although this article guides you through the process of making a theme *from scratch*, it is often better to start your work with a base theme, such as the popular Zen, Omega or Tao themes. If you're new to this, making a theme from scratch first will be less confusing and teach you more!

Ad

**Before You Begin**

To be able to follow and complete the tutorial effectively, you should understand the basics of:

* The purpose of websites
* HTML and CSS: how they are used to make websites
* PHP: what it is, and how it is used to enhance
* CMS's: what they are, and how they are used to power websites
* Drupal: How to set up a simple website using it

Don't be put off by this list, however. With enough enthusiasm, you can learn what you need to while following this tutorial.  
  
In order to develop a theme quickly and smoothly (without internet-connection-related issues), you should set up a developing environment (a code editor program and a testing server) on your computer. The most common way of doing this is to install a WAMP / LAMP / MAMP stack (W: windows; L: Linux; M: Mac, A: Apache, the server software; M: MySQL, the database software; P: PHP, the server-side script) and download an open-source code editor.

**1**

**Design the website.**

* + Begin by sketching the layout of the home page of your website using a pencil and a clean sheet of paper; you need to feel free to make mistakes and changes. Draw all the parts of a basic web-page, including the header, navigation menu (which may be just below the header or in a sidebar), the page body (for the content), and the footer. You may have other areas in your design.
  + When you’re satisfied with your design, recreate your sketched design a graphics program that allows you to design web pages, such as Adobe Photoshop, or even better, Adobe Fireworks.
  + Then make sure you add all the other visual elements; consider the colours and dark-light contrasts, the typography and how it adds to the user’s experience, the layout and empty spaces (whitespace is important!), and how content will fit in.

**2**

**Develop the website using [HTML](http://www.wikihow.com/Learn-HTML" \o "Learn HTML) and CSS.**

* + Just like in the website design step, it is probably most efficient to only create the web pages of the site that have their own unique layouts, such as the Home page and Contact Us page. Keep the layout as simple as possible, and use modern practices and conventions where possible (such as HTML5 and CSS3). At this point, don't add any server-side scripting (such as PHP), and keep client-side scripting (such as Javascript) to a minimum, in order to not interfere with Drupal's code; this can be added later.

**3**

**[Set up Drupal](http://www.wikihow.com/Set-up-Drupal" \o "Set up Drupal) to begin theming.** [Download Drupal](https://drupal.org/project/drupal) and move the files to the server you will use to develop your theme (your testing server - recommended - or an online server).

* + Place the downloaded files into the *root* folder (if you're using a testing server, it's the *www* or *localhost* folder). Use a web browser to navigate to the root folder (e.g.<http://www/>) and follow the installation process. You should then install any modules that will be useful for theme development.

**4**

**Understand Drupal’s web page structure and convert your static pages to it.**

* + HTML web pages are known as ‘static’ – which means they aren’t designed to have content that regularly changes – and are built using block-type tags such as ‘divs’ (and ‘header’, ‘footer’, etc. in HTML5).
  + Drupal web pages are a combination of static parts (HTML) and dynamic parts – parts that contain content that is expected to be changed regularly. These dynamic parts are called ‘**regions**’ in Drupal. An example of a region is a sidebar, where you might want to insert navigational links, a search box, social media buttons, and so on. See the diagram below for a visual representation:  
      
    The things that you can put into regions are called ‘**blocks**’ in Drupal. For example, a search box or a menu are commonly-used blocks. You can insert blocks into regions, re-arrange the blocks, and remove blocks from regions when you log into Drupal on the *Structure* > *Blocks* page. Many of the available blocks in a typical Drupal website are provided by the installed modules, but you can create your own blocks on the Blocks page.  
      
    The content of each web page (the body text, headings, images, and so on) are also placed into a region: the ‘Content’ region, which is required for all Drupal themes. The content of each web page (or blog post or custom content type) is called a ‘**node**’, and is packaged into the *Main page content* block. In other words, the content of the *About Us* page is saved into node (with a node type of ‘page’, of course), and is output to the *Main page content* block, which is placed in the*Content* region.  
      
    Note that sometimes multiple nodes are displayed on one web page through the*Main page content* block; an example is the front page, which by default shows several of the website’s most recent articles (blog posts).  
      
    Your theme will control the regions, but not what the website administrators put into them. As the themer, you have inform Drupal which regions your theme will have, and register them in files called templates, which are covered in the next section.

**Part2**

**Creating the Theme Files**

**1**

**Set up the theme folder.** In this folder create a .info file which contains the theme's purpose and gives information about how it is used. List what goes into it. In the folder add screenshots. and a CSS folder.

**2**

**Create the template files.**  
  
Drupal ‘**template**’ files (that have a file name ending in ‘.html.twig’) are used by Drupal to structure the output of each web page. Standard templates include:

* + html.html.twig
  + page.html.twig
  + block.html.twig
  + node.html.twig
  + custom template pages  
      
    Each web page that Drupal outputs (the ‘Home page’ for example) will use the *html*template once, the *page* template once, and the *region*, *block* and *node* templates at least once each.  
      
    If you'd like to see the typical PHP code and HTML markup that goes into each of these templates, you can look at the template files in the directory of the core Drupal themes in the *drupal* > *themes* folder (not the *drupal* > *sites* > *all* > *themes*folder).

**3**

**Insert the Process and Pre-process functions.**  
  
To understand these functions, you first need to understand how drupal saves and outputs content and HTML markup. As you should have seen in the templates in the previous step, all Drupal templates have several variables available to them that they can output using functions like *{{ page['content'] }}*. In that example, the variable *content* (which is stored in the *page* array rather than as a variable by itself; its just how Drupal does it) stores all the text that the user typed into the web page's 'body' when the user was logged into Drupal.  
  
Basically, process and pre-process functions allow the themer to modify any of the variables available to the templates before Drupal makes them available for use by the templates. These functions are all written in the *template.php* file (note that this is *not* a '.html.twig' file).

**Part3**

**Install the Theme and Finishing Touches**

**1**

**Install the theme in Drupal, set up the required views and add the required nodes and regions.**

**2**

**Install additional useful modules which enhance appearance, security and the functionality.**  
  
The following list of modules are very useful to most Drupal websites, and was written by Steve Floyd in his article [Top 25 Drupal 7 Modules](http://www.axzm.com/top-25-drupal-7-modules) (**NB.** This list will likely change for Drupal 8):

* + **Views**. Vital for themers, Views allows you to gather almost any content from your website (such as ‘the 10 most recent articles’) and present it as a page or block. Particularly useful for creating a dynamic home page.
  + Views Slideshow
  + Pathauto
  + Mollom
  + Google Analytics
  + WYSIWYG
  + XMLsitemap
  + Backup and Migrate
  + Workbench
  + Media
  + Drupal Commerce

**3**

**Add website content to complete the site.**