**Wordpress workflow:**

**Setup Tasks:**

* Your content
* Buy hosting
* Buy a domain
* Install Wordpress

**Tips for choosing a Domain:**

* Stick with .com.
* Use keywords in your domain name search.
* Keep your domain name short.
* Make it easy to pronounce and spell.
* Keep it unique and brandable.
* Avoid **hyphens** in domain name.
* Avoid doubled letters.
* Leave room to expand.

[www.hover.com](http://www.hover.com) is a good place to buy domains.

There are some tools available to generate domain names e.g. [www.namemesh.com](http://www.namemesh.com)

**Hosting Companies:**

Some good hosting companies which are expert in wordpress hosting are:

* Pantheon
* Flywheel
* **SiteGround (Recommended)**
* Liquid web
* Bluehost

**Features to look for a Hosting company:**

* They have a strong reputation in the industry.
* They promise at least 99% uptime.
* They offer a variety of plans.
* They provide 24/7 customer service.
* The price is right.
* Their account interface is intuitive.
* They provide ample resources for getting started.
* Are there automatic backups.
* Will they make your site secure?
* Are there restrictions.
* Are they compatible with Wordpress

**Evaluating Themes and workflows:**

* Determine main goal of your website.
* Find designs you like to help you achieve that goal
* Test a few to see which you think works best

[www.studiopress.com](http://www.studiopress.com) suggested for wordpress themes.

**The Customizer:**

The Customizer is where you will go to take your site’s Theme to the next level. From this section you can change your Site Title and Tagline, add Widgets to a Sidebar or Footer, create Custom Menus, change your Homepage Settings and more.

**Launching:**

* Test browsers for all screen sizes.
* Check all the forms.
* Check all the links.
* Turn on search engine visibility.

**Child Theme:**

A **child theme** in **WordPress** is a sub **theme** that inherits all the functionality, features, and style of its parent **theme**. **Child themes** are a safe way to modify a **WordPress theme** without actually making any changes to the parent **theme's** files.

Setup a local development environment so that you can do experiments without breaking actual site. [www.localbyflywheel.com](http://www.localbyflywheel.com) is recommended as an local WordPress development tool.

* Download localbyflywheel from their site.
* Follow all the steps.
* Make your account.
* Click on CREATE A NEW SITE.
* Give wordpress user id and pwd.
* Follow the on screen instructions.
* Now you are in local mode of your site.

To get the files of the site, Right click on the name of site and click on “Reveal in Finder”.

Go to app 🡪public🡪 wp-content

Now you are ready to start developing a child theme on your computer.

**Defining our child theme:**

* Open the wp-content in local editor. (themes folder)
* In the theme folder, create a new directory called twentytwenty-child.
* In that directory create a file called style.css.
* In start of that file, give some comments of basic information e.g Theme name, theme url, description, Author, Author url, template name, version.

/\*

 Theme Name:   Child Example

 Theme URI:    http://example.com/child-exxample/

 Description:  Child Example Twenty Fifteen Child Theme

 Author:       Your Name

 Author URI:   http://example.com

 Template:     twentyfifteen

 Version:      1.0.0

 License:      GNU General Public License v2 or later

 License URI:  http://www.gnu.org/licenses/gpl-2.0.html

 Tags:         Tags: black, blue, gray, pink, purple, white, yellow, dark, light, two-columns, left-sidebar, fixed-layout, responsive-layout, accessibility-ready, custom-background, custom-colors, custom-header, custom-menu, editor-style, featured-images, microformats, post-formats, rtl-language-support, sticky-post, threaded-comments, translation-ready

 Text Domain:  twenty-fifteen-child

\*/

 This may look like a lot, but there are really only two that are necessary, which are Theme Name and Template. The Theme Name tells WordPress the name of the child theme. The Template tells WordPress which theme it should consider as the parent theme, which is very important.

* Create one more file called function.php.
* To make sure you load the parent file’s stylesheet, we will need to enqueue it in the child theme. You will need a functions.php file in the child theme to put the following snippet in.

<?php

add\_action( 'wp\_enqueue\_scripts', 'enqueue\_parent\_styles' );

function enqueue\_parent\_styles() {

    wp\_enqueue\_style( 'parent-style', get\_template\_directory\_uri().'/style.css' );

}

After that in

body{

}

You can add your styles. like background or any other functions.

Now in wordpress dashboard you can see your child theme and activate it.

**Migrating a site from local to live:**

There are two ways for that:

* **WordPress’s import/export function.**
	1. Tools—export—all content—download(as an xml file)
	2. Then In your loca siteTools—import—WordPress—installnow—runimoporter
	3. Run the importer and upload the exported xml file by clicking upload file and import.
	4. Follow the instructions.
* **WP Migrate DB plug-in:**

Each migration takes shape in the form of an easy to follow three-step process.

1. Find and replace the data you want directly within the WP Migrate DB interface.
2. Export the SQL.
3. Import into your new database using a tool such as phpMyAdmin

**Testing our theme with extreme content:**

**Step 1: Install WordPress’ theme unit test data on your staging website**

Download Wordpress theme unit test data

WordPress’ Theme Unit Test Data enables you to simulate the look and feel of a site that’s been around for a while. Once you import the data, WordPress will automatically generate a bunch of pages, different types of posts, menu items, sidebars, and more. This lets you conduct a much more realistic test of your potential theme.

To get started, first download the latest version of the Theme Unit Test Data, then access your testing site’s dashboard. Next, navigate to the Tools → Import tab, and look for the WordPress option at the bottom. (same steps listed above)

### Step 2: Set up and test a WordPress theme

### At this point, you can **install and activate your new theme as normal**, then finally get down to business. The testing process is simple, although it may take some time since you need to be as thorough as possible.

### Install monster widget plugin and activate it then go to appearance-widget and add it to footer area.

### Moving your child theme live:

### There are 2 ways to do that:

###  WP Dashboard:

### From the local environment make a zipped folder of child theme. Then go to live site, in the dash board go to appearance🡪themes🡪addnew🡪uploadtheme🡪zippedfolder🡪installnow. Now you can take live preview and activate/publish.

### FTP:

### Filezilla can be used to ftp. ftp and then same process as above.

### Building Custom sites with Wordpress:

### A project relies on clear communication, and even clear deliverables and deadlines.

### Pre -project Setup:

### Setting up the site.

### Installing your local development environment.

### Creating a version control repository

### Building a staging site

### What you need from the client:

### Content

### Designs

### Milestone approval

### Developing

### Testing

### Deploying

### Creating Multiple environment:

### In bigger projects, its important to have multiple environments

### Dev

### This is where developers can experiment. Client will not see it. You can do development experiments on your local environment.

### Staging

### Every client project should have a staging site. This is private duplicate of live project.

###  Dev🡪>staging🡪live

### Since you are working on multiple environment synchronization between environments are important. WP Migrate DB is plugin for that.

### Using version control:

### GIT is best where you can work on different branches. You can then merge them with master branch or revert the changes. Then you can push them to staging server.

### Testing the site:

### All testing should be done on staging site. Check your site on different browsers and operating systems.

### Taking the site live:

### Click on every menu item.

### Read every page

### Make sure all images and other media load

### Follow every link and test the site against all available devices.

###  Then make your site live.

### Advanced Tools and WorkFlows:

### Grunt and other build tools

### You install different dependencies in start of projects and tools e.g Sass, Linters and Js Libraries. Root of different task runners is node.js. Gulp and Grunt are leaders of pack.

### Automated testing

### Tests your code without pushing it to the server.

### Prevent unknown bugs and crashes.

### Find bigger issues before users.

### Tests that your code works with the rest of the project.

### Unit Tests:

### Take a specific component of code to make sure it works on its own .e.g adding 2 numbers

### Integration Tests:

### Scripts that are written to check for a specific chain of events. e.g our added numbers are properly passed to a shopping cart.

### UI Regression Testing:

### Simulates user interaction on a website to make sure the UI or major components of the website do not beak. E.g. When the “Add to Cart” button is clicked, an item is added to the cart and the total updated.

### React and Gutenberg

### The new wordpress block editor is developed in React js so its essential to understand react. Before learning React you should have understanding of html,css, javascript specifically ES6, Node.js/npm

### Static site generators:

### Cons of CMS-Driven Sites

### Can be slow

### Feature bloated

### Can’t rely on caching in certain situations

### Have become complex

###  To overcome these problems Static site Generator come in.

###  Convert cms-driven sites into html, CSS and JavaScript only.

###  Pro and Cons of Static site Generators:

|  |  |
| --- | --- |
| Pros | Cons |
| Generally faster. | Not great for interactive sites. |
| More secure. | Does not support always changing content well. |
| Version control for content. |  |

### Jekyll, Gatsby, VuePress are few site generated tools.