Cornershop INTRANET

(https://intranet2.cshp.co.ddev.site/)The best way to find stuff…

Search

Search

Menu

Documentation: Cornershop Plugin Tracker

Favorite ☆

Page Contents

Overview

The Cornershop Plugin Tracker is a WordPress plugin created to help keep track of the plugins and themes that a client's website is using. While we have traditionally tracked most of the code in a client's website using Git, that Git repo would track a lot of third-party code, including WordPress core, third-party plugins, and third-party themes. These third-party tools have greatly increased the size of the Git repos and makes it harder to know what custom code that we have developed for a client's website. When we update WordPress, update a plugin, or update a theme, all of those files are then pushed it Git.

Cornershop Plugin Tracker was developed as a tool to help keep track of the third-party tools that a client's site relies on without having to keep of the actual code of these tools.

Technical Details

The plugin works by creating two files: `composer.json` file and a `README.md` file that helps toto keep track of what version of WordPress core, themes, and plugins are installed on the website. It also tries to determine if a plugin or theme is available on wordpress.org and what themes and plugins are "premium". The plugin also backs up the installed premium plugins to AWS S3.

> **Note:** Premium **does not** mean that a plugin needs to be paid plugin or theme, it just means that the plugin and theme is not available on wordpress.org according to the WordPress API. Since we try to verify if a plugin is available on wordpress.org according to the plugin folder name, if a previously tracked "premium" theme and/or plugin has the same name and same version as a plugin or theme on wordpress.org, then the wordpress.org version of the theme and/or plugin will override the premium theme and/or plugin.

COMPOSER.JSON

The `composer.json` file generated tracks public plugins and wordpress.org plugins using https://wpackagist.org/ (https://wpackagist.org/) as the composer repo. Premium plugins are identified with the "premium-plugin/insert-plugin-slug" as the key in the `composer.json file`.

JavaScript

```
{
    "name": "movingtraditions/wordpress",
    "description": "Installed plugins and themes for the WordPress install https://www.movingtraditions.org",
    "type": "project",
    "repositories": [
        {
            "type": "composer",
            "url": "https://wpackagist.org",
            "only": [
                "wpackagist-muplugin/*",
                "wpackagist-plugin/*",
                "wpackagist-theme/*"
            ]
        },
        {
            "type": "composer",
            "url": "https://www.movingtraditions.org/wp-json/cshp-plugin-tracker/plugin/download",
            "only": [
                "premium-plugin/*",
                "premium-theme/*",
                "core/wp"
            ]
        }
    ],
    "require": {
        "core/wp": "6.2.2",
        "premium-plugin/advanced-custom-fields-pro": "6.1.7",
        "premium-plugin/blocksy-companion-pro": "1.8.97",
        "premium-plugin/cshp-plugin-tracker": "1.0.11",
        "premium-plugin/custom-wp-login": "1.2.5",
        "premium-plugin/events-calendar-pro": "6.1.1",
        "premium-plugin/gravityforms": "2.7.10",
        "premium-plugin/searchwp": "4.3.2",
        "premium-plugin/user-activity-log-pro": "2.3.2",
        "premium-plugin/wp-rocket": "3.14.2.1",
        "premium-plugin/wp-rocket-no-cache": "*",
        "premium-theme/blocksy-child": "1.0.0",
        "wpackagist-plugin/block-navigation": "4.1.1",
        "wpackagist-plugin/blockmeister": "3.1.10",
        "wpackagist-plugin/duplicate-post": "4.5",
        "wpackagist-plugin/editor-block-outline": "1.3.0",
        "wpackagist-plugin/google-analytics-for-wordpress": "8.18",
        "wpackagist-plugin/imagify": "2.1.1",
        "wpackagist-plugin/iwp-client": "1.12.3",
        "wpackagist-plugin/log-emails": "1.4.0",
        "wpackagist-plugin/printfriendly": "5.5.0",
        "wpackagist-plugin/redirection": "5.3.10",
        "wpackagist-plugin/simpletoc": "6.0.4",
        "wpackagist-plugin/the-events-calendar": "6.1.3",
        "wpackagist-plugin/timeline-block": "1.4",
        "wpackagist-plugin/user-switching": "1.7.0",
        "wpackagist-plugin/wordfence": "7.10.3",
        "wpackagist-plugin/wordpress-seo": "20.12",
        "wpackagist-plugin/wp-external-links": "2.58",
        "wpackagist-plugin/wp-mail-smtp": "3.8.2",
        "wpackagist-plugin/wpfront-notification-bar": "3.3.2",
        "wpackagist-theme/blocksy": "1.8.97",
        "wpackagist-theme/twentytwentythree": "1.1"
    },
    "extra": {
        "installer-paths": {
            "wp-content/mu-plugins/{$name}/": [
                "type:wordpress-muplugin"
            ],
            "wp-content/plugins/{$name}/": [
                "type:wordpress-plugin"
            ],
            "wp-content/themes/{$name}/": [
                "type:wordpress-theme"
            ]
        }
    }
}
```

README.MD

The `README.md` file tracks the installed and activated versions of the plugins and themes.

WordPress Version

- 6.2.2

WP-CLI Command to Install WordPress Core

```
wp core download --skip-content --version="6.2.2" --force --path=.
```

Themes Installed

Wordpress.org Themes

Status	Name	Folder	Version	Theme URL/Author
Inactive	Blocksy	blocksy	1.8.97	Link
Inactive	Twenty Twenty-Three	twentytwentythree	1.1	Link

Premium Themes

Status	Name	Folder	Version	Theme URL/Author
Active	Blocksy Child	blocksy-child	1.0.0	Link

Plugins Installed

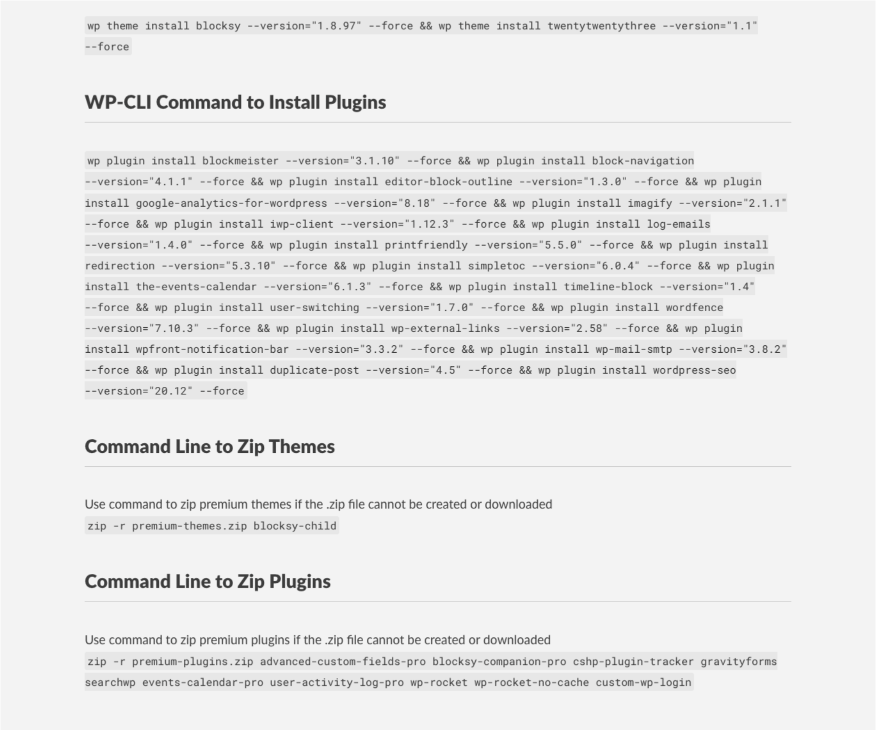
Wordpress.org Plugins

Status	Name	Folder	Version	Plugin URL/Author
Active	Block Navigation	block-navigation	4.1.1	Link
Active	BlockMeister	blockmeister	3.1.10	Link
Active	Yoast Duplicate Post	duplicate-post	4.5	Link
Active	Editor Block Outline	editor-block-outline	1.3.0	Link
Active	Google Analytics for WordPress by MonsterInsights	google-analytics-for-wordpress	8.18	Link
Active	Imagify	imagify	2.1.1	Link
Active	InfiniteWP - Client	iwp-client	1.12.3	Link
Active	Log Emails	log-emails	1.4.0	Link
Active	Print, PDF & Email by PrintFriendly	printfriendly	5.5.0	Link
Active	Redirection	redirection	5.3.10	Link
Active	SimpleTOC - Table of Contents Block	simpletoc	6.0.4	Link
Active	The Events Calendar	the-events-calendar	6.1.3	Link
Active	Timeline Block	timeline-block	1.4	Link
Active	User Switching	user-switching	1.7.0	Link
Active	Wordfence Security	wordfence	7.10.3	Link
Active	Yoast SEO	wordpress-seo	20.12	Link
Active	WP External Links	wp-external-links	2.58	Link
Active	WP Mail SMTP	wp-mail-smtp	3.8.2	Link
Active	WPFront Notification Bar	wpfront-notification-bar	3.3.2	Link

Premium Plugins

Status	Name	Folder	Version	Plugin URL/Author
Active	Advanced Custom Fields PRO	advanced-custom-fields-pro	6.1.7	Link
Active	Blocksy Companion (Premium)	blocksy-companion-pro	1.8.97	Link
Active	Cornershop Plugin Tracker	cshp-plugin-tracker	1.0.11	Link
Active	YIKES Custom Login	custom-wp-login	1.2.5	Link
Active	The Events Calendar Pro	events-calendar-pro	6.1.1	Link
Active	Gravity Forms	gravityforms	2.7.10	Link
Active	SearchWP	searchwp	4.3.2	Link
Active	User Activity Log Pro	user-activity-log-pro	2.3.2	Link
Active	WP Rocket	wp-rocket	3.14.2.1	Link
Active	WP Rocket	Disable Page Caching	wp-rocket-no-cache	

WP-CLI Command to Install Themes



```
wp theme install blocksy --version="1.8.97" --force && wp theme install twentytwentythree --version="1.1"
--force
```

WP-CLI Command to Install Plugins

```
wp plugin install blockmeister --version="3.1.10" --force && wp plugin install block-navigation
--version="4.1.1" --force && wp plugin install editor-block-outline --version="1.3.0" --force && wp plugin
install google-analytics-for-wordpress --version="8.18" --force && wp plugin install imagify --version="2.1.1"
--force && wp plugin install iwp-client --version="1.12.3" --force && wp plugin install log-emails
--version="1.4.0" --force && wp plugin install printfriendly --version="5.5.0" --force && wp plugin install
redirection --version="5.3.10" --force && wp plugin install simpletoc --version="6.0.4" --force && wp plugin
install the-events-calendar --version="6.1.3" --force && wp plugin install timeline-block --version="1.4"
--force && wp plugin install user-switching --version="1.7.0" --force && wp plugin install wordfence
--version="7.10.3" --force && wp plugin install wp-external-links --version="2.58" --force && wp plugin
install wpfront-notification-bar --version="3.3.2" --force && wp plugin install wp-mail-smtp --version="3.8.2"
--force && wp plugin install duplicate-post --version="4.5" --force && wp plugin install wordpress-seo
--version="20.12" --force
```

Command Line to Zip Themes

Use command to zip premium themes if the .zip file cannot be created or downloaded

```
zip -r premium-themes.zip blocksy-child
```

Command Line to Zip Plugins

Use command to zip premium plugins if the .zip file cannot be created or downloaded

```
zip -r premium-plugins.zip advanced-custom-fields-pro blocksy-companion-pro cshp-plugin-tracker gravityforms
searchwp events-calendar-pro user-activity-log-pro wp-rocket wp-rocket-no-cache custom-wp-login
```

(https://intranet2.cshp.co.ddev.site/wp-content/uploads/2023/08/cshp-plugin-tracker-readme-md.png)
Cornershop Plugin Tracker generated README.md file

BACKUP PREMIUM PLUGINS TO CORNERSHOP PLUGIN RECOVERY (CPR) WEBSITE

The plugin does daily backups of premium plugins that are installed on a client's website. These backups are sent to the Cornershop Plugin Recovery website https://plugins.cornershopcreative.com (https://plugins.cornershopcreative.com) . CPR is primarily used for restoring previous versions of premium plugins that were installed on a client's site. Some premium plugins don't have an easy way to download previous versions of plugins such as Gravity Forms.

CPR is meant to help restore these plugins when the Support team does weekly plugin updates.

Downloading the plugin

> Note: The plugin should **only** be used on **active Maintenance only, Maintenance + Support, and Retainer client sites**. If a client is not a Maintenance client or is offboarding from Maintenance, remove the plugin

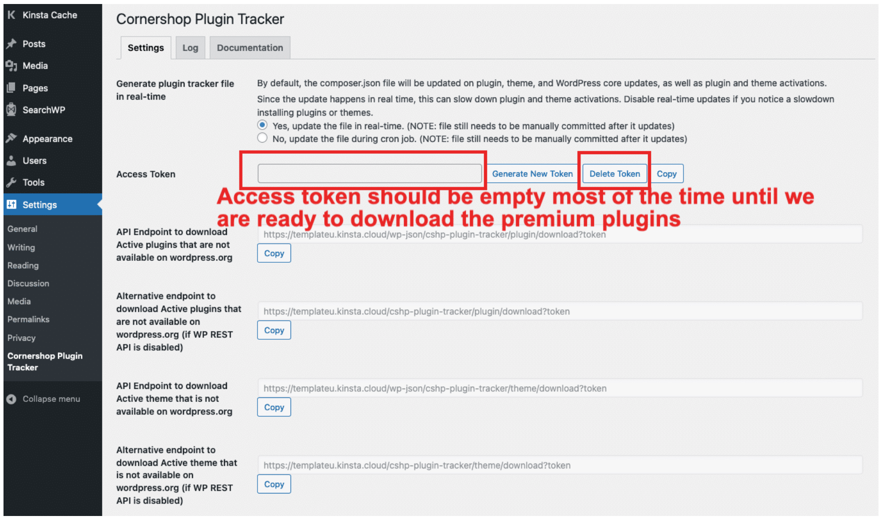
If you are need to upload the plugin to a site, it can be downloaded from 1Password (https://start.1password.com/open/i?a=7IFO7QAJ5ZCLROZSQTVWNGM6HY&v=d2ssky7umxkif5vqgqbovtuaey&i=5klf5fhxc3fvu33tu5jf3re4bm&h=cornershopcreative.1password.com) .

The plugin can be installed by uploading the plugin to the website. You don't need to configure the plugin or change any settings for the plugin for it to work.

SETTINGS

There is only one configurable setting on the plugins's Settings page. That is the Access Token setting. The Access Token is **only** needed if you are trying to download the premium plugins or themes directly from the website.

This Access Token should be empty otherwise.



Cornershop Plugin Tracker

Generate plugin tracker file in real-time

By default, the composer.json file will be updated on plugin, theme, and WordPress core updates, as well as plugin and theme activations.
Since the update happens in real time, this can slow down plugin and theme activations. Disable real-time updates if you notice a slowdown installing plugins or themes.

● Yes, update the file in real-time. (NOTE: file still needs to be manually committed after it updates)
○ No, update the file during cron job. (NOTE: file still needs to be manually committed after it updates)

Access Token

Access token should be empty most of the time until we are ready to download the premium plugins

API Endpoint to download Active plugins that are not available on wordpress.org: https://templateu.kinsta.cloud/wp-json/cshp-plugin-tracker/plugin/download?token

Alternative endpoint to download Active plugins that are not available on wordpress.org (if WP REST API is disabled): https://templateu.kinsta.cloud/cshp-plugin-tracker/plugin/download?token

API Endpoint to download Active theme that is not available on wordpress.org: https://templateu.kinsta.cloud/wp-json/cshp-plugin-tracker/theme/download?token

Alternative endpoint to download Active theme that is not available on wordpress.org (if WP REST API is disabled): https://templateu.kinsta.cloud/cshp-plugin-tracker/theme/download?token

(https://intranet2.cshp.co.ddev.site/wp-content/uploads/2023/08/cshp-plugin-tracker-access-token.png)

Access Token should only be used when downloading the premium plugins from the live website

Update Git to stop tracking core, plugins, and themes

> Note: These steps only relate to removing the currently tracked plugins from the Git repo. This only needs to be done one time. The plugin does **not** depend on this process and will work just fine without doing these.
>
> You should follow these steps when you are updating Git.

In order to not track WP core, plugins, and themes in Git, the `.gitignore (https://bitbucket.org/cornershopcreative/kinstatemplate/src/main/.gitignore)` file for the project repo needs to be updated.

If the plugin or theme is already being tracked in the Git repo, there are steps you will need to take remove those.

COMMANDS TO REMOVE PLUGINS AND THEMES FROM EXISTING REPO

1. **Commit all untracked changes**

   Before removing third-party code, you should commit all uncommitted code so we can roll the codebase back to a previous version in case something goes wrong.

   Git

   ```
   git add -u && git add wp-includes/ wp-admin/ wp-content/mu-plugins wp-content/plugins wp-content/themes && git commit -m "Commit uncommitted code changes for this site

   - Baseline commit of whatever code has been modified for this website before removing all third-party plugins and themes from being tracked"
   ```

2. **Update the `.gitignore` file.**

   Update the `.gitignore` file to stop tracking WordPress core, third-party plugins, and third-party themes.

   Note: Update the references to `wp-content` if the wp-content/ folder has been renamed.

```
rm .gitignore;  cat > .gitignore  << EOL
# WordPress core files
/wp-admin/*
/wp-includes/*
/index.php
/license.txt
/readme.html
/wp-*.php
/xmlrpc.php
/wp-content/upgrade/*
!/wp-config.php

# Ignore all files and folders in the /wp-content folder except for the plugins, themes,and uploads folders. We ignore them later
/wp-content/*
!/wp-content/plugins
!/wp-content/themes
!/wp-content/mu-plugins
!/wp-content/uploads

# Ignore WordPress uploads so we can track certain files later on since we want to track some Cornershop plugins that write to the upload folder
/wp-content/uploads/*

# WordPress plugin, themes, and custom files
/wp-content/plugins/*
!/wp-content/plugins/index.php
/wp-content/themes/*
!/wp-content/themes/index.php
!/wp-content/themes/blocksy-child
!/wp-content/themes/crate
!/wp-content/themes/crate-child
!/wp-content/themes/minerva
!/wp-content/themes/jupiter-child
!/wp-content/themes/jupiterx-child

# Custom plugins that we want to track (more than likely just Cornershop plugins)
# Don't ignore mu-plugins
!/wp-content/mu-plugins
/wp-content/mu-plugins/cshp*/node_modules

# Ignore Kinsta mu-plugins
/wp-content/mu-plugins/kinsta-mu-plugins
/wp-content/mu-plugins/kinsta-mu-plugins.php
# Ignore Pantheon mu-plugins
/wp-content/mu-plugins/pantheon-mu-plugin
/wp-content/mu-plugins/pantheon-mu-plugin.php

# Don't ignore plugins that start with cshp, for custom Cornershop plugins
!/wp-content/plugins/cshp*
/wp-content/plugins/cshp*/node_modules
/wp-content/plugins/cshp-support
/wp-content/plugins/cshp-plugin-tracker
/wp-content/plugins/cshp-kinsta

# Files in the Uploads directory to track (more than likely files created by Cornershop plugins)
# Unignore and ignore the subfolders for the Cornershop Plugin Tracker plugin except for the composer.json and the README.md file
!/wp-content/uploads/cshp-plugin-tracker
/wp-content/uploads/cshp-plugin-tracker/*
!/wp-content/uploads/**/cshp-plugin-tracker
/wp-content/uploads/**/cshp-plugin-tracker/*
!/wp-content/uploads/cshp-plugin-tracker/composer.json
!/wp-content/uploads/cshp-plugin-tracker/README.md
!/wp-content/uploads/cshp-plugin-tracker/.htaccess
!/wp-content/uploads/**/cshp-plugin-tracker/composer.json
!/wp-content/uploads/**/cshp-plugin-tracker/README.md
!/wp-content/uploads/**/cshp-plugin-tracker/.htaccess


# Unignore and ignore the subfolders for the Cornershp Customizer CSS plugin except for the CSS files in that folder
!/wp-content/uploads/cshp-customizer-css
!/wp-content/uploads/**/cshp-customizer-css
/wp-content/uploads/cshp-customizer-css/*
/wp-content/uploads/**/cshp-customizer-css/*
!/wp-content/uploads/cshp-customizer-css/*.css
!/wp-content/uploads/**/cshp-customizer-css/*.css

```

```
# Folders created by plugins in the wp-content folder
/wp-content/advanced-cache.php
/wp-content/object-cache.php
/wp-content/cache/*
/wp-content/cache/supercache/*
/wp-content/w3tc/*
/wp-content/wfcache/
/wp-content/wflogs/
/wp-content/infinitewp/
/wp-content/uploads/backupbuddy_temp/
/settings_backup-*.php
/wp-content/wp-rocket-config
/wp-content/mu-plugins/kinsta-mu-plugins*
/wp-content/envato-backups/
/wp-content/smush-webp/
/wp-content/backup-db/*
/wp-content/updraft/
/wp-content/backupwordpress-*/
/wp-content/managewp/backups/
/wp-content/backups/
/wp-content/a1wm-backups/
/wp-content/wp-cache-config.php
/wp-content/ngg
/wp-content/gallery
/wp-content/wp-rocket-config
/wp-content/wp-cloudflare-super-page-cache

# Imagify NGINX webp rewrite rules
/conf

# Log files
*.log
*.log-*
error_log

# Dotenv
.env
.env.*
!.env.example

# Composer (top-level files only)
/vendor
/composer.lock

# WP-CLI
wp-cli.local.yml

# Log files
*.log
*.log-*
error_log

# Cornershop themes asset compiling that we should not be tracking
/wp-content/themes/crate/node_modules
/wp-content/themes/blocksy-child/node_modules
/wp-content/themes/jupiter-child/node_modules
/wp-content/themes/jupiterx-child/node_modules
/wp-content/themes/minerva/node_modules
.sass-cache

# Wordfence things
/wp-content/plugins/wordfence/tmp/
wordfence-waf.php
wflogs/
.user.ini

# GoWP files
/sucuri-*
.sucuriquarantine

# Exported top level SQL files
/*.sql
/*.sql.gz
/wp-content/*.sql
/wp-content/*.sql.gz
/wp-content/themes/blocksy-child/**/*.sql
/wp-content/themes/crate/**/*.sql
/wp-content/themes/jupiter-child/**/*.sql

# Exported top level compressed files
```

```
/*.zip
/*.tar.gz
/wp-content/*.zip
/wp-content/*.tar.gz

# Project files
*.sublime-*
.idea/
.ideas/
sftp-config.json
*.un~
.*.swp
.vscode/

# Packages #
############
*.7z
*.dmg
*.gz
*.bz2
*.iso
*.jar
*.rar
*.tar
*.zip
*.tgz

# Misc
.DS_Store
**/.DS_Store
__MACOSX/
**/__MACOSX/
/google*.html
/newhost.txt
/index.html
/BingSiteAuth.xml
/.well-known
ehthumbs.db
Thumbs.db
/sitemap.xml
/sitemap.xml.gz
/sitemap.html
/settings_backup-*.php


# Temporary files generated by build/test tools
wp-content/themes/crate/inc/gulp-config.inc
wp-content/themes/crate/_backstop/backstop_data/
wp-content/themes/blocksy-child/inc/gulp-config.inc
wp-content/themes/blocksy-child/_backstop/backstop_data/

# Site specific ignores

EOL
```

3. **Update the `.gitignore` file to ignore additional files that are specific to this project.**

   Look at the list of files and directories in the site root folder and the files and directories in the `wp-content/` folder to see if there are other files that need to be ignored or other files that need to be tracked where exceptions need to be added to the `.gitignore` file.

   Files that should be tracked:

   - Custom themes.
   - Custom plugins.
   - Custom mu-plugins.
   - Custom drop-ins.
   - Other important files we want to track.
4. **Update the `.gitattributes` file.**

```
rm .gitattributes; cat > .gitattributes << EOL
# Don't bother showing diffs for Crate's or Blocksy Child compiled CSS,
# or for sourcemaps or minified files in general
wp-content/themes/crate/css/**/*.css -diff
wp-content/themes/crate/js/**/*.js -diff
wp-content/themes/blocksy-child/css/**/*.css -diff
wp-content/themes/blocksy-child/js/**/*.js -diff
*.map -diff
*.min.* -diff
*.sql -diff
EOL
```

5. **Commit the updated updated .gitignore and .gitattribute file.**

Git

```
git add .gitignore .gitattributes && git commit -m "
Update .gitignore file to ignore WP core, third-party plugins, and third-party themes when tracking in the repo

 - Updating git repo so we are only tracking the custom code that we develop"

```

6. **Remove all of the previously tracked code and recommit all the code we now want to track.**

   Reset Git's cache so we have to re-add all files that we want to now track

Git

```
git rm -r --cached . && git add . && git commit -m "Removing WP core, third-party plugins. and third-party themes from Git tracking"
```

Install and activate the Cornershop Plugin tracker plugin with WP CLI.

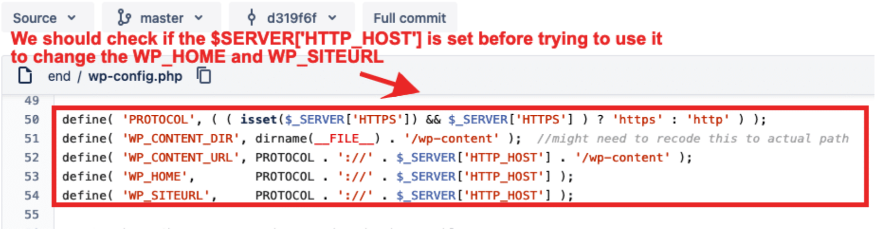
If the Cornershop Plugin Tracker plugin is not already installed, you can download the plugin using WP CLI or by downloading (http://plugins.cornershopcreative.com/plugin-releases/cshp-plugin-tracker.zip) and uploading the Zip file to the WordPress admin.

1. UPDATE THE `WP-CONFIG.PHP` FILE TO DETECT IF `$_SERVER['HTTP_HOST']` IS SET BEFORE USING IT TO DYNAMICALLY CHANGE THE SITE URL.

Note: If you don't do this before running the `wp cshp-pt generate` then the generated `composer.json` file will not log the correct domain URL.

```
if ( ( ! defined( 'WP_CLI' ) || ! WP_CLI ) && isset( $_SERVER['HTTP_HOST'] ) && ! empty( $_SERVER['HTTP_HOST'] ) )
{
```

This only applies if the `wp-config.php` file does not check if the $_SERVER['HTTP_HOST'] is set before trying to use it. A lot of our older `wp-config.php` files have this bug that needs to be manually fixed.



We should check if the $SERVER['HTTP_HOST'] is set before trying to use it to change the WP_HOME and WP_SITEURL

```
define( 'PROTOCOL', ( ( isset($_SERVER['HTTPS']) && $_SERVER['HTTPS'] ) ? 'https' : 'http' ) );
define( 'WP_CONTENT_DIR', dirname(__FILE__) . '/wp-content' );  //might need to recode this to actual path
define( 'WP_CONTENT_URL', PROTOCOL . '://' . $_SERVER['HTTP_HOST'] . '/wp-content' );
define( 'WP_HOME',        PROTOCOL . '://' . $_SERVER['HTTP_HOST'] );
define( 'WP_SITEURL',     PROTOCOL . '://' . $_SERVER['HTTP_HOST'] );
```

(https://intranet2.cshp.co.ddev.site/wp-content/uploads/2023/08/cshp-plugin-tracker-change-wp-config.png)

2. DOWNLOAD AND ACTIVATE THE PLUGIN

Bash

```
wp plugin install https://plugins.cornershopcreative.com/plugin-releases/cshp-plugin-tracker.zip --skip-plugins --skip-themes && wp plugin activate cshp-plugin-tracker --skip-plugins --skip-themes
```

3. GENERATE THE COMPOSER.JSON AND README.MD FILE

Bash

```
wp cshp-pt generate --skip-themes
```

4. CREATE A .HTACCESS FILE IN THE CSHP-PLUGIN-TRACKER UPLOADS FOLDER TO BLOCK REQUEST TO README.MD AND COMPOSER.JSON FILE.

If the website is on Apache or even if the site is not using an Apache server, you should create a `.htaccess` file to prevent someone from coming across the composer.json file and revealing the list of plugins installed on the site.

```
cat > wp-content/uploads/cshp-plugin-tracker/.htaccess << EOL
<IfModule mod_alias.c>
# Deny access to the composer.json and the README.md created by the Cornershop Plugin Tracker plugin using Alias Module
RedirectMatch 404 ^(.*/)?\.(json|md)+
</IfModule>
<IfModule !mod_alias.c>
<IfModule mod_rewrite.c>
# If Alias Module is not available but Rewrite Module is,
# use Rewrite Module to deny access to the composer.json and the README.md files
RewriteEngine On
RewriteBase /
RewriteRule ^(.*/)?\.(json|md)+ - [R=404,L]
</IfModule>
</IfModule>
EOL
```

5. ADD THE `COMPOSER.JSON`, `README.MD`, AND `.HTACCESS` FILES TO GIT

Git

```
git add wp-content/uploads/cshp-plugin-tracker && git commit -m "Add README.md and composer.json file that has the list of WordPress core, third-party themes, and third-party plugins installed" && git push -u origin
```

6. PUSH REPO CHANGES TO REMOTE GIT REPO.

Bash

```
CURRENT_GIT_BRANCH=$(git symbolic-ref --short HEAD) && git push --set-upstream origin $CURRENT_GIT_BRANCH
```

YOU DID IT

You should now be set with adding the plugin to an existing website and removing third-party code from Git.

Cloning a Cornershop Plugin Tracker website to D20

Unlike the previous process of cloning a site to D20 (https://intranet2.cshp.co.ddev.site/clone-a-kinsta-site-to-d20/#Set_up_the_site_on_D20) where all of the plugins and themes were tracked in Git, there are some additional steps that need to be taken.

1. MOVE TO THE USER'S SITES DIRECTORY

Bash

```
cd ~/sites
```

2. COPY/PASTE AND EXECUTE THE SSH CLONE COMMAND FROM THE BITBUCKET REPOSITORY PAGE.

Git

```
git clone git@bitbucket.org:cornershopcreative/pcjf.git
```

3. MOVE TO THE NEW SITE DIRECTORY.

Bash

```
cd pcjf
```

4. INSTALL WORDPRESS CORE WITH WP CLI.

Since WordPress core is no longer being tracked in Git, you will need to download WordPress.

Note: These commands should be run from the site root.

Use the version of WordPress core that was installed on the live site by reading the `composer.json` file:

Bash

```
wp_version=$(jq .require'."core/wp"' wp-content/uploads/cshp-plugin-tracker/composer.json) && wp core download --
skip-content --path=. --version=$wp_version --force
```

Update the path to the `composer.json` file if the `wp-content/` folder has been renamed to something else or if the WordPress uploads folder `wp-content/uploads` has been moved or has a custom name.

Examples:

- cms/assets
- assets/
- etc…

Or just install the **latest** version of WordPress core.

Bash

```
wp core download --skip-content --path=. --force
```

5. CREATE A DATABASE FOR THE SITE. RESPOND TO THE PROMPTS DURING EXECUTION.

Bash

```
createdb
```

6. COPY THE DB CONNECTION PHP FROM `CREATEDB` AND ADD THEM TO THE `WP-CONFIG.PHP` FILE. COMMENT OUT RATHER THAN DELETING THE KINSTA DB LINES.

Bash

```
vi wp-config.php
```

7. VERIFY DB CONNECTION AND FORCE GIT TO IGNORE THE CHANGED FILE. THIS CHANGE MUST NEVER REACH KINSTA!

Bash

```
wp db query 'select database(); show tables;' && git update-index --assume-unchanged wp-config.php
```

8. GET THE DB EXPORT FROM KINSTA AND IMPORT IT INTO THE NEWLY CREATED DB.

There are multiple ways of exporting the database from the live site:

- WP CLI database export (recommended approach)
- PHPMyAdmin export
- mySQL command line export
- Using a database export plugin in WordPress

There are multiple ways of downloading the database export from the server and pulling it onto D20:

- SFTP
- SCP
- Rsync
- Using Transmit to connect to connect to the live site, downloading the database to your local computer, uploading the database to D20

After uploading the database, use import the database with the `importdb` command.

Bash

```
importdb insert-name-of-export-file.sql
```

9. CHECK THAT THE TABLES ALL LOADED PROPERLY.

Bash

```
wp db query 'select database(); show tables;'
```

10. UPDATE THE ROOT `.HTACCESS` FILE TO REDIRECT REQUESTS FOR MISSING UPLOADED FILES TO THE KINSTA DOMAIN. MAKE SURE TO CHANGE THE HOSTNAME TO THE URL OF THE LIVE WEBSITE!

```
# Redirect missing files on Cornershop dev server to prod server
# NOTE: place this block BEFORE other blocks containing RewriteRule directives,
# especially WordPress's own!
<IfModule mod_rewrite.c>
    RewriteEngine On
    RewriteBase /
    RewriteCond %{HTTP_HOST} \.cshp\.co$ [NC]
    RewriteCond %{REQUEST_FILENAME} !-f
    RewriteCond %{REQUEST_FILENAME} !-d
    RewriteRule ^wp-content/uploads/(.*)$ https://DOMAIN.ORG/wp-content/uploads/$1 [L,R=302]
</IfModule>

```

11. FORCE GIT TO IGNORE THE CHANGED FILE. THIS CHANGE MUST NEVER REACH KINSTA!

Git

```
git update-index --assume-unchanged .htaccess
```

12. EXECUTE `FIXPERMS` TO CLEAN UP ANY FILESYSTEM PERMISSION ISSUES.

13. INSTALL THE CORNERSHOP PLUGIN TRACKER

Bash

```
wp plugin install https://plugins.cornershopcreative.com/plugin-releases/cshp-plugin-tracker.zip --activate --skip-plugins --skip-themes
```

14. BULK INSTALL THE WORDPRESS.ORG PLUGINS THAT WERE TRACKED IN THE CORNERSHOP PLUGIN TRACKER'S `COMPOSER.JSON` FILE.

Bash

```
wp cshp-pt plugin-install
```

15. BULK INSTALL THE WORDPRESS.ORG THEMES THAT WERE TRACKED IN THE CORNERSHOP PLUGIN TRACKER'S COMPOSER.JSON FILE

Bash

```
wp cshp-pt theme-install
```

16. BULK INSTALL THE PREMIUM PLUGINS AND PREMIUM THEME FROM THE LIVE SITE.

> Note: As of version 1.1.0 released on 2/27/24, if you are running these commands on D20, D24, or from a Kinsta website that is on the Cornershop Kinsta server, you do not have to generate a token to install the premium plugins. The IP address of these servers are whitelisted so they can bypass the need to generate a token.
>
> If you are working from some other server or a local website, you will still need to generate a token to download the premium plugins.

If working from D20, D24, or a Kinsta website on Cornershop Kinsta Server

1. **Run WP CLI command to install the premium plugins**

Bash

```
wp cshp-pt plugin-install https://joybound.org
```

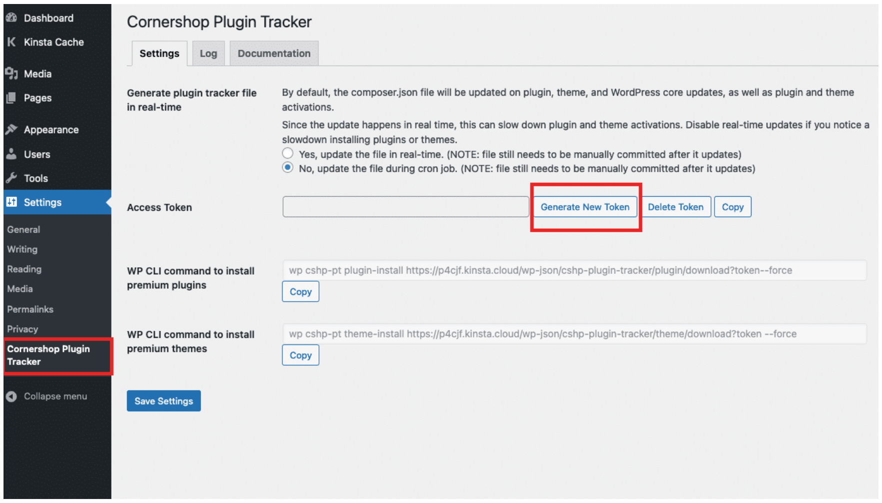
2. **Run WP CLI command to install the premium theme**

```bash
wp cshp-pt theme-install https://joybound.org
```

If NOT working from D20, D24, or a Kinsta website on Cornershop Kinsta Server

1. **Log into the live website and create a new token.**

   Save the settings page after generating the token. This will pre-fill the token in the generated commands.



Cornershop Plugin Tracker

Generate plugin tracker file in real-time: By default, the composer.json file will be updated on plugin, theme, and WordPress core updates, as well as plugin and theme activations.

Since the update happens in real time, this can slow down plugin and theme activations. Disable real-time updates if you notice a slowdown installing plugins or themes.

○ Yes, update the file in real-time. (NOTE: file still needs to be manually committed after it updates)
● No, update the file during cron job. (NOTE: file still needs to be manually committed after it updates)

Access Token:

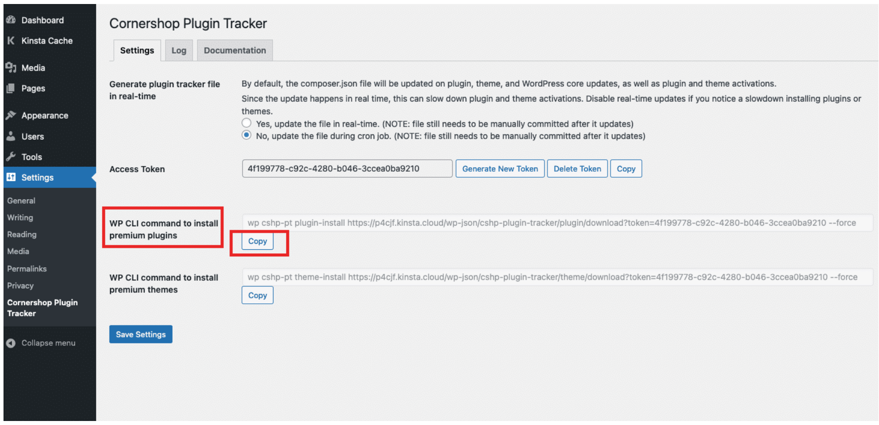
WP CLI command to install premium plugins: wp cshp-pt plugin-install https://p4cjf.kinsta.cloud/wp-json/cshp-plugin-tracker/plugin/download?token--force

WP CLI command to install premium themes: wp cshp-pt theme-install https://p4cjf.kinsta.cloud/wp-json/cshp-plugin-tracker/theme/download?token --force

(https://intranet2.cshp.co.ddev.site/wp-content/uploads/2023/08/cshp-plugin-tracker-generate-download-token-instructions.png)

Generate a download token

2. **Copy the REST API endpoint for downloading the premium plugins (if there are active premium plugins)**



Cornershop Plugin Tracker

Generate plugin tracker file in real-time: By default, the composer.json file will be updated on plugin, theme, and WordPress core updates, as well as plugin and theme activations.

Since the update happens in real time, this can slow down plugin and theme activations. Disable real-time updates if you notice a slowdown installing plugins or themes.

○ Yes, update the file in real-time. (NOTE: file still needs to be manually committed after it updates)
● No, update the file during cron job. (NOTE: file still needs to be manually committed after it updates)

Access Token: 4f199778-c92c-4280-b046-3ccea0ba9210

WP CLI command to install premium plugins: wp cshp-pt plugin-install https://p4cjf.kinsta.cloud/wp-json/cshp-plugin-tracker/plugin/download?token=4f199778-c92c-4280-b046-3ccea0ba9210 --force

WP CLI command to install premium themes: wp cshp-pt theme-install https://p4cjf.kinsta.cloud/wp-json/cshp-plugin-tracker/theme/download?token=4f199778-c92c-4280-b046-3ccea0ba9210 --force

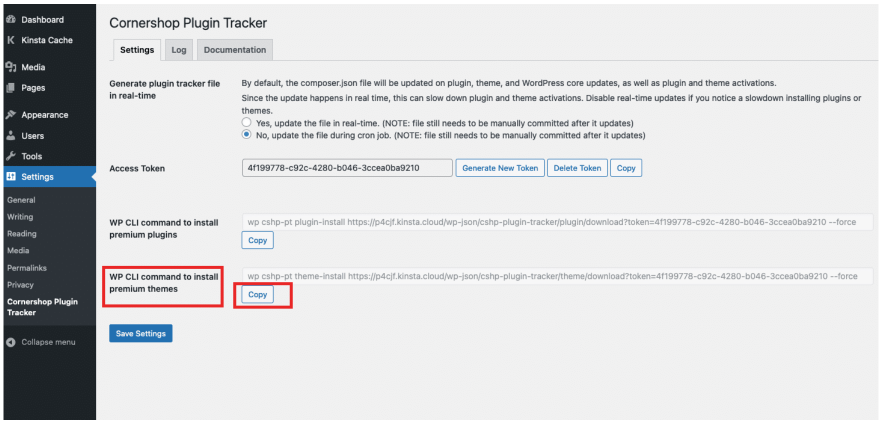
(https://intranet2.cshp.co.ddev.site/wp-content/uploads/2023/08/cshp-plugin-tracker-premium-plugins-wp-cli-instructions.png)

Copy WP CLI command to install premium plugins

3. **Run WP CLI command to install plugins**

```bash
wp cshp-pt plugin-install https://joybound.org/wp-json/cshp-plugin-tracker/plugin/download?token=4f199778-c92c-4280-b046-3ccea0ba9210 --force
```

4. **Copy the REST API endpoint for downloading the premium themes (if there is an active premium theme)**



Cornershop Plugin Tracker

Generate plugin tracker file in real-time: By default, the composer.json file will be updated on plugin, theme, and WordPress core updates, as well as plugin and theme activations. Since the update happens in real time, this can slow down plugin and theme activations. Disable real-time updates if you notice a slowdown installing plugins or themes.

○ Yes, update the file in real-time. (NOTE: file still needs to be manually committed after it updates)

● No, update the file during cron job. (NOTE: file still needs to be manually committed after it updates)

Access Token: 4f199778-c92c-4280-b046-3ccea0ba9210

WP CLI command to install premium plugins:

```
wp cshp-pt plugin-install https://p4cjf.kinsta.cloud/wp-json/cshp-plugin-tracker/plugin/download?token=4f199778-c92c-4280-b046-3ccea0ba9210 --force
```

WP CLI command to install premium themes:

```
wp cshp-pt theme-install https://p4cjf.kinsta.cloud/wp-json/cshp-plugin-tracker/theme/download?token=4f199778-c92c-4280-b046-3ccea0ba9210 --force
```

(https://intranet2.cshp.co.ddev.site/wp-content/uploads/2023/08/cshp-plugin-tracker-premium-themes-wp-cli-instructions.png)

Copy WP CLI command to install premium themes

5. **Run WP CLI command to install themes**

Bash

```
wp cshp-pt theme-install https://joybound.org/wp-json/cshp-plugin-tracker/theme/download?token=4f199778-c92c-4280-
b046-3ccea0ba9210 --force
```

17. YOU'RE DONE!

If the premium plugins and/or premium theme were download successfully, your work can continue.

If the premium plugins or themes were not downloaded and installed, you will need to SSH into the live site and make an archive or the premium plugins and/or theme. The `zip` command to do this on the command line should be in the `README.md` that the plugin generates.

Git pull latest changes from live site to D20

> Note: You will ONLY need to run this command one time. You ONLY need to run this command right after doing the FIRST git pull from a production website.
>
> If there is no existing site on D20 and you are cloning from Kinsta to D20, then use the Cloning a Cornershop Plugin Tracker website to D20 (https://intranet2.cshp.co.ddev.site/documentation-cornershop-plugin-tracker/#Cloning_a_Cornershop_Plugin_Tracker_website_to_D20) instructions.

If there is already an existing copy of a website on D20 and you notice that the live site has had the Cornershop Plugin Tracker plugin added to the site, when you run the command `git pull origin main` or `git pull origin production` , the D20 website will be missing the WordPress core, plugins, and themes code. This is due to the Git repo and the `.gitignore` file being updated to ignore this third-party code.

IF YOU ARE WORKING ON A DIFFERENT BRANCH OR HAVE MADE CHANGES TO THE LIVE BRANCH

In cases where you have made code changes on a different branch or made changes to the live branch.

1. Commit uncommitted changes to the staging branch.
2. Preserve the `wp-config.php` file. It will be different than the one on prod. Either stash it or copy it elsewhere.

RESTORE THE DELETED FILES (I.E. WORDPRESS CORE, THEMES, AND PLUGINS)

To restore WordPress core, plugins and themes, you will need run some commands to recover the deleted files

Bash

```
git pull origin main
git log --oneline
COMMIT_HASH_RIGHT_BEFORE_FILES_WERE_DELETED="INSERT_THE_BEFORE_HASH"
COMMIT_HASH_WHERE_FILES_WERE_DELETED="INSERT_THE_DELETE_HASH"
git checkout $COMMIT_HASH_RIGHT_BEFORE_FILES_WERE_DELETED
git log --diff-filter D --pretty="oneline" --name-only --format= -l 0 $COMMIT_HASH_RIGHT_BEFORE_FILES_WERE_DELETED..
$COMMIT_HASH_WHERE_FILES_WERE_DELETED | sed '/^$/d' | zip deleted-files.zip -@
git checkout main
unzip deleted-files.zip
git reset -- `git diff --diff-filter=D --name-only $COMMIT_HASH_WHERE_FILES_WERE_DELETED`
wp plugin install https://plugins.cornershopcreative.com/plugin-releases/cshp-plugin-tracker.zip --activate --skip-
plugins --skip-themes
```

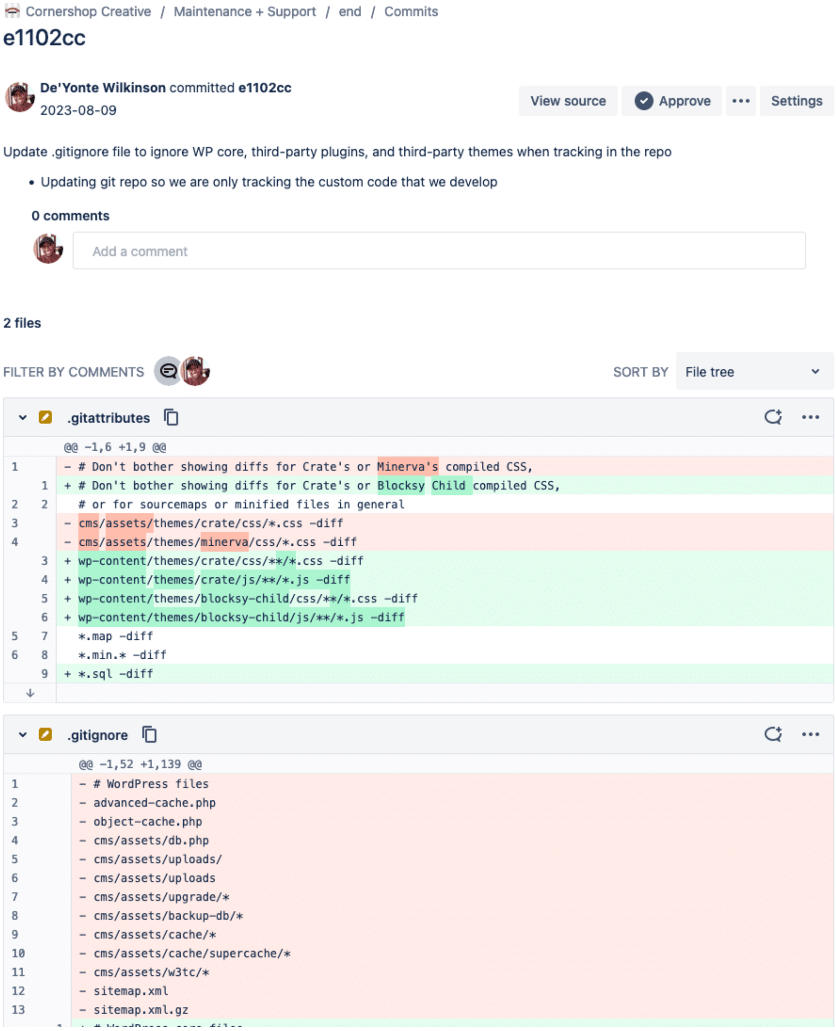
This command:

1. Pull from the latest version of the branch (e.g. `git pull origin main`)

2. Run `git log --oneline` to grab the correct commit hashes.

- The BEFORE commit should have a commit message “Update .gitignore file to ignore WP core, third-party plugins, and third-party themes when tracking in the repo”

Bash



Cornershop Creative / Maintenance + Support / end / Commits

e1102cc

De'Yonte Wilkinson committed e1102cc
2023-08-09

Update .gitignore file to ignore WP core, third-party plugins, and third-party themes when tracking in the repo

- Updating git repo so we are only tracking the custom code that we develop

0 comments

2 files

.gitattributes

```
@@ -1,6 +1,9 @@
- # Don't bother showing diffs for Crate's or Minerva's compiled CSS,
+ # Don't bother showing diffs for Crate's or Blocksy Child compiled CSS,
  # or for sourcemaps or minified files in general
- cms/assets/themes/crate/css/*.css -diff
- cms/assets/themes/minerva/css/*.css -diff
+ wp-content/themes/crate/css/**/*.css -diff
+ wp-content/themes/crate/js/**/*.js -diff
+ wp-content/themes/blocksy-child/css/**/*.css -diff
+ wp-content/themes/blocksy-child/js/**/*.js -diff
  *.map -diff
  *.min.* -diff
+ *.sql -diff
```

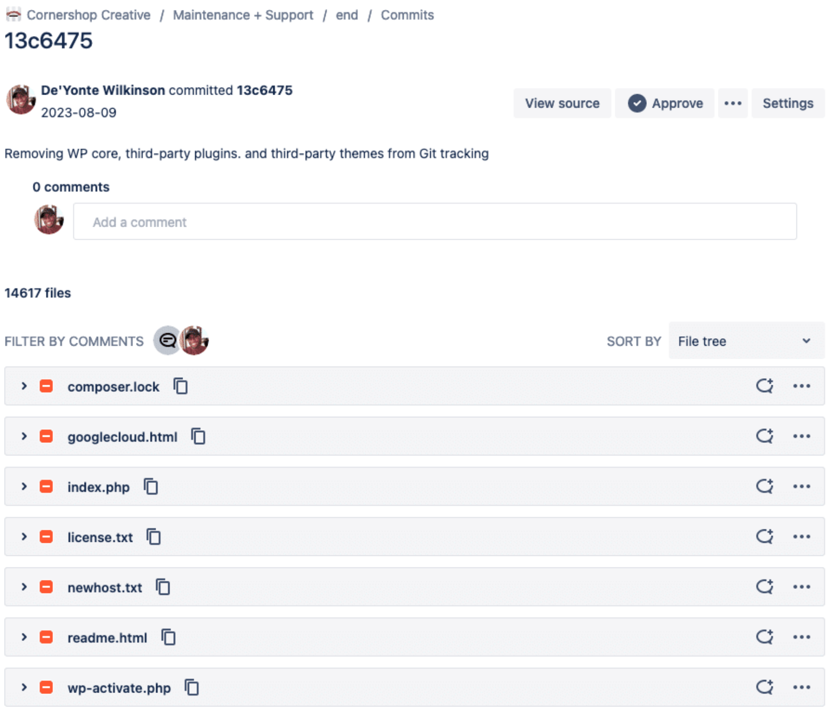
.gitignore

```
@@ -1,52 +1,139 @@
- # WordPress files
- advanced-cache.php
- object-cache.php
- cms/assets/db.php
- cms/assets/uploads/
- cms/assets/uploads
- cms/assets/upgrade/*
- cms/assets/backup-db/*
- cms/assets/cache/*
- cms/assets/cache/supercache/*
- cms/assets/w3tc/*
- sitemap.xml
- sitemap.xml.gz
```

(https://intranet2.cshp.co.ddev.site/wp-content/uploads/2023/08/cshp-plugin-tracker-commit-before.png)

Before commit log as seen in Bitbucket

- The DELETE commit should have a commit message “Removing WP core, third-party plugins. and third-party themes from Git tracking”



13c6475

De'Yonte Wilkinson committed 13c6475
2023-08-09

Removing WP core, third-party plugins. and third-party themes from Git tracking

0 comments

Add a comment

14617 files

- composer.lock
- googlecloud.html
- index.php
- license.txt
- newhost.txt
- readme.html
- wp-activate.php

(https://intranet2.cshp.co.ddev.site/wp-content/uploads/2023/08/cshp-plugin-tracker-commit-after.png)

Delete commit log as seen in Bitbucket

3. Add the correct commit hashes to the temporary, environment varibles

   ◦ COMMIT_HASH_RIGHT_BEFORE_FILES_WERE_DELETED
   ◦ COMMIT_HASH_WHERE_FILES_WERE_DELETED

   Bash

   ```
   COMMIT_HASH_RIGHT_BEFORE_FILES_WERE_DELETED="12345"
   COMMIT_HASH_WHERE_FILES_WERE_DELETED="54321"
   ```

4. Git checkout the commit **right** before all of the files were removed from git (e.g. `commit_hash_right_before_files_were_deleted`)

5. Get the difference between the list of files in the current commit vs the list of files that were deleted in the commit where all the files were deleted (e.g `git log --diff-filter D --pretty="oneline" --name-only --format= -l 0 commit_hash_right_before_files_were_deleted..commit_hash_where_where_files_were_deleted`)

6. Convert that list into the `sed` command to remove empty lines (e.g. `sed '/^$/d'`)

7. Pipe that list to the Zip command to zip all of those deleted files up into zip file (e.g. `sed '/^$/d' | zip deleted-files.zip -@` . The `sed` command removes blank empty lines from the output.)

8. Switch back to the branch (e.g. `git checkout main`)

9. Unzip the deleted files zip file (e.g. `unzip deleted-files.zip`)

10. Reset the list of files to remove it from the local index to avoid committing them when you commit changes to the branch in the future (e.g. `git reset -- git diff --diff-filter=D --name-only commit_hash_where_files_were_deleted`)

    1. Pull from the latest version of the branch (e.g. `git pull origin main`)
    2. Run `git log --oneline` to grab the correct commit hashes.
       ◦ The BEFORE commit should have a commit message “Update .gitignore file to ignore WP core, third-party plugins, and

    third-party themes when tracking in the repo”

   - The AFTER commit should have a commit message “Removing WP core, third-party plugins. and third-party themes from Git tracking”

3. Git checkout the commit **right** before all of the files were removed from git (e.g. `commit_hash_right_before_files_were_deleted`)
4. Get the difference between the list of files in the current commit vs the list of files that were deleted in the commit where all the files were deleted (e.g `git log --diff-filter D --pretty="oneline" --name-only --format= -l 0 commit_hash_right_before_files_were_deleted..commit_hash_where_where_files_were_deleted`)
5. Convert that list into the `sed` command to remove empty lines (e.g. `sed '/^$/d'`)
6. Pipe that list to the Zip command to zip all of those deleted files up into zip file (e.g. `sed '/^$/d' | zip deleted-files.zip -@` . The `sed` command removes blank empty lines from the output.)
7. Switch back to the branch (e.g. `git checkout main`)
8. Unzip the deleted files zip file (e.g. `unzip deleted-files.zip`)
9. Reset the list of files to remove it from the local index to avoid committing them when you commit changes to the branch in the future (e.g. `git reset -- git diff --diff-filter=D --name-only commit_hash_where_files_were_deleted`)
10. Install the Cornershop Plugin Tracker plugin (if not already installed)

WP CLI commands

CSHP-PT GENERATE

Create the installed plugins and themes composer.json and README.md tracker files.

Bash

```
wp cshp-pt generate
```

CSHP-PT PLUGIN-INSTALL

Install the wordpress.org plugins that are activated on this site.

Bash

```
wp cshp-pt plugin-install
```

CSHP-PT PLUGIN-INSTALL <ZIP_PATH|PREMIUM_PLUGIN_DOWNLOAD_URL>

Install the premium plugins.

Bash

```
wp cshp-pt plugin-install https://example.org/wp-json/cshp-plugin-tracker/plugin/download?token=insert-token
```

CSHP-PT THEME-INSTALL

Install the wordpress.org plugins that are activated on this site.

Bash

```
wp cshp-pt theme-install
```

CSHP-PT THEME-INSTALL <ZIP_PATH|PREMIUM_THEME_DOWNLOAD_URL>

Install the premium theme.

Bash

```
wp cshp-pt theme-install https://example.org/wp-json/cshp-plugin-tracker/theme/download?token=insert-token
```

CSHP-PT PLUGIN-ZIP

Create the zip of the installed premium plugins.

Bash

```
wp cshp-pt plugin-zip
```

CSHP-PT THEME-ZIP

Create the zip of the installed premium themes.

Bash

```
wp cshp-pt theme-zip
```

CSHP-PT BACKUP-PLUGIN

Mmanually back up the premium plugins to the Cornershop Plugin Recovery (CPR) website.

Bash

```
wp cshp-pt backup-plugin
```