

functions.php – WordPress 6.2

Obwohl sie in Ihrem Design enthalten ist, ist die WordPress-Datei functions.php nicht darauf beschränkt, nur Anpassungen an Ihrem Design vorzunehmen.

Sie können an jedem Teil Ihrer Website Anpassungen vornehmen, ähnlich wie bei Plugins.

Hier sind einige gängige Arten von Anpassungen, die Sie mit der WordPress-Datei functions.php vornehmen können:

- Ändern Sie das Kernverhalten von WordPress, z. B. wie viele Beiträge auf der Suchergebnisseite angezeigt werden oder welche Inhalte in den RSS-Feed Ihrer Website aufgenommen werden sollen.
- Erstellen Sie Ihre eigenen benutzerdefinierten Shortcodes.
- Fügen Sie Ihrer Website neue Inhalte oder Skripte hinzu, indem Sie beispielsweise ein Skript von Ihrem Live-Chat-Dienst auf bestimmten Seiten einfügen oder die [Fußzeile Ihrer Website bearbeiten](#) .

Das kratzt nur an der Oberfläche...

[Da Sie mit der Datei functions.php statt statischem HTML benutzerdefinierte PHP-Codefragmente zu Ihrer Website hinzufügen können](#) , sind der Art der Änderungen, die Sie vornehmen können, keine Grenzen gesetzt.

Wo befindet sich die WordPress-functions.php-Datei?

Die WordPress-Datei functions.php befindet sich zusammen mit [anderen Theme-Dateien](#) im Ordner Ihres aktiven Themes . Um die Datei zu finden, kannst du dich über FTP oder ein Dateimanager-Tool mit deinem Server verbinden und zu `.../wp-`

`content/themes/[aktiver-Theme-Name]/functions.php` navigieren

Wenn Sie beispielsweise [das beliebte OceanWP-Design](#) verwenden , befindet sich die Datei `functions.php` unter `.../wp-content/themes/oceanwp/functions.php`

So arbeiten Sie sicher mit der Datei `functions.php`

Da das Arbeiten mit der WordPress-Datei `functions.php` das Hinzufügen von Code zu Ihrer Website beinhaltet, ist es wichtig, einige Best Practices zu befolgen, bevor Sie mit der Bearbeitung beginnen.

[Sogar etwas so Einfaches wie ein fehlendes Komma oder Apostroph kann Fehler auf Ihrer Website](#) auslösen , wie z. B. den [WordPress White Screen of Death](#) .

In neueren Versionen von WordPress ist WordPress besser darin geworden, diese Fehler eleganter zu behandeln, z. B. durch Anzeigen der [Meldung „Auf Ihrer Website ist ein kritischer Fehler aufgetreten“](#) oder durch Prüfen auf PHP-Fehler, bevor Ihre Änderungen gespeichert werden.

Aber trotzdem – um Probleme zu vermeiden, empfehlen wir immer, diese Best Practices zu befolgen...

Testen Sie Ihren `functions.php`-Code auf einer Staging-Site

Bevor Sie Code-Snippets von `functions.php` zu Ihrer Live-WordPress-Site hinzufügen, empfehlen wir immer, sie zuerst auf einer Staging-Version Ihrer Site zu testen. Auf diese Weise können Sie nach potenziellen Fehlern suchen und überprüfen, ob das Code-Snippet wie vorgesehen funktioniert.

Die Info

Eine „Staging-Site“ ist eine identische Kopie Ihrer Live-Website, die sich in einer sicheren, privaten Sandbox befindet. Damit können Sie Änderungen testen, ohne [den Wartungsmodus auf Ihrer Live-Site verwenden](#) zu müssen .

Verwenden Sie immer ein untergeordnetes Design, wenn Sie die Datei `functions.php` bearbeiten

[Da sich die WordPress-Datei `functions.php` im Code Ihres WordPress-Themes befindet](#) , überschreibt Ihr Theme die Datei `functions.php`, wenn [Sie Ihr Theme aktualisieren](#) .

Um zu vermeiden, dass die Änderungen, die Sie an der Datei `functions.php` Ihres Themes vorgenommen haben, überschrieben werden, sollten Sie immer ein untergeordnetes WordPress-Theme verwenden und Ihre Codeschnipsel zur Datei `functions.php` im untergeordneten Design *hinzufügen* .

Wenn Sie ein Child-Theme verwenden, können Sie das Parent-Theme bei Bedarf immer noch aktualisieren, aber alle Ihre `functions.php`-Anpassungen werden niemals überschrieben.

Um mehr zu erfahren, lesen Sie unsere vollständige [Anleitung zum Erstellen eines untergeordneten WordPress-Themes](#) .

2 `functions.php` Dateialternativen, die besser sein könnten

Während die WordPress-Datei `functions.php` eine einfache Möglichkeit bietet, Ihrer Website Code-Snippets hinzuzufügen, gibt es einige Alternativen zu `functions.php`, die in den meisten Situationen eine bessere Lösung bieten:

1. Verwenden eines Code-Manager-Plugins

2. Speichern von Codeschnipseln in einem benutzerdefinierten Plugin

Diese Alternativen können einige Vorteile gegenüber der Verwendung der Datei `functions.php` haben:

- **Nicht an Ihr Design gebunden** – das bedeutet, dass Ihre `functions.php`-Anpassungen immer noch vorhanden sind, wenn Sie jemals das Design ändern.
- **Bessere Organisation** – diese Alternativen erleichtern die Organisation Ihrer Code-Snippets, was hilfreich sein kann, wenn Sie planen, Ihrer Website viele Code-Snippets hinzuzufügen.
- **Mehr Kontrolle** – im Falle eines Code-Manager-Plug-ins erhalten Sie hilfreiche Optionen wie das Aktivieren/Deaktivieren eines Snippets mithilfe einer Umschaltfläche, das Ausführen von Snippets nur im Frontend oder Backend und mehr.

Verwenden Sie ein Code Manager-Plugin

Ein Code-Manager-Plug-in ist ein Plug-in, das Ihnen eine benutzerfreundliche Oberfläche zum Hinzufügen und Bearbeiten von Code-Snippets bietet, die sonst in der Datei `functions.php` landen würden.

Eine der beliebtesten Optionen ist das kostenlose [Code Snippets-Plugin](#), aber Sie können auch andere Plugins finden, die ähnliche Funktionen bieten.

Mit Code-Snippets können Sie Ihre `functions.php`-Code-Snippets aus dem WordPress-Dashboard hinzufügen, komplett mit Optionen, um ...

- Fügen Sie einen Titel und eine Beschreibung hinzu.
- Organisieren Sie Ihre Snippets mithilfe von Tags.

- Führen Sie das Snippet nur auf einem bestimmten Teil Ihrer Website aus.

Dashboard Kinsta Cache Posts Media Pages Comments Astra Appearance Plugins Snippets All Snippets Edit Snippet Add New Import Settings Users Tools Settings Collapse menu

Edit Snippet [Add New](#) Help ▾

Disable admin bar

Code PHP [Save Changes](#) [Activate](#)

```
<?php
1 add_action( 'wp', function () {
2     if ( ! current_user_can( 'manage_options' ) ) {
3         show_admin_bar( false );
4     }
5 } );
```

Run snippet everywhere Only run in administration area Only run on site front-end Only run once Priority

Description Visual

B I U “ ABC ☰ ☷ ☹ ☺ ☻ ☼ ↶ ↷ ↸ ↹ ↺ ↻ 🔗 ✕

Turns off the WordPress admin bar for everyone except administrators.

This is a sample snippet. Feel free to use it, edit it, or remove it.

Tags

[Save Changes and Activate](#) [Save Changes](#) [Download](#) [Export](#) [Delete](#)

Hinzufügen eines Code-Snippets im Code-Snippets-Plugin.
Sie können dann alle Ihre Snippets in einer Liste sehen und sie nach Bedarf einfach aktivieren/deaktivieren.

So verwalten Sie Snippets in Code Snippets.

Da es sich bei Code Snippets um ein Plugin handelt, sind alle Ihre Code-Snippets auch dann noch vorhanden, wenn [Sie das WordPress-Theme ändern](#) .

Erstellen Sie ein Plug-in für benutzerdefinierte Funktionen

Eine weitere Alternative zur WordPress-Datei `functions.php` besteht darin, einfach Ihr eigenes benutzerdefiniertes Plugin zu erstellen, um die Code-Snippets aufzunehmen.

Auch wenn dies kompliziert klingen mag, ist es tatsächlich viel einfacher, als es scheint.

So funktioniert das:

1. Erstellen Sie einen Ordner auf Ihrem lokalen Computer für Ihr Plugin.
2. Erstellen Sie eine einzelne `.php`-Datei in diesem Ordner und bearbeiten Sie sie mit Ihrem bevorzugten Texteditor.

3. Fügen Sie die folgende Codevorlage zur Datei hinzu.
4. Fügen Sie Ihre functions.php-Code-Snippets zur Datei hinzu

```
<?php
```

```
/**
```

```
* Plugin Name: My Custom Code Snippets  
* Description: This is a custom plugin to house code snippets.  
* Author: Kinsta  
* Version: 1.0  
*/
```

```
/* Add the first code snippet below this comment - use the  
comment to explain what it does. */
```

```
[CODE]
```

```
/* Add the second code snippet below this comment. */
```

```
[CODE]
```

```
/* Continue as needed. */
```

```
[CODE]
```

```
?>
```

Dann müssen Sie nur noch das Plugin in WordPress installieren und aktivieren. Hier sind 2 Möglichkeiten, dies zu tun:

1. [Verbinde dich über FTP/SFTP](#) mit deinem Server und lade den benutzerdefinierten Plugin-Ordner in den Ordner **wp-content/plugins** hoch . Gehen Sie dann zur **Plugin-** Liste und aktivieren Sie das Plugin.
2. Verwenden Sie Ihren Computer, um eine ZIP-Datei des Ordners zu erstellen. Gehen Sie dann zu **Plugins → Neu hinzufügen** und laden Sie die .zip-Datei hoch, um sie wie jedes andere Plugin zu installieren.

So greifen Sie auf die WordPress-Datei `functions.php` zu: 2 Optionen

Es gibt verschiedene Möglichkeiten, auf die WordPress-Datei `functions.php` zuzugreifen und sie zu bearbeiten. Hier sind 2 der vielseitigsten Methoden, die auf jedem Host funktionieren:

1. WordPress In-Dashboard-Design-Code-Editor.
2. SFTP und Ihren eigenen Code-Editor.

1. Verwenden Sie den WordPress In-Dashboard Theme Code Editor

Standardmäßig können Sie mit WordPress den gesamten Code Ihres Themes von Ihrem WordPress-Dashboard aus bearbeiten, einschließlich der Datei `functions.php`:

1. Öffnen Sie Ihr WordPress-Dashboard.
2. Gehen Sie zu **Darstellung** → **Designdatei-Editor** .
3. Wählen Sie die Datei **Designfunktionen (functions.php)** in der Liste **Designdateien** auf der rechten Seite aus.
4. Fügen Sie Ihre Änderungen im Code-Editor hinzu.
5. Klicken Sie auf die Schaltfläche **Datei aktualisieren** , um Ihre Änderungen zu speichern.

Dashboard

Kinsta Cache

Posts

Media

Pages

Comments

Astra

Appearance

Themes

Customize

Widgets

Menus

Theme File Editor

Plugins

Users

Tools

Settings

Collapse menu

Edit Themes

Kinsta Astra Child Theme: Theme Functions (functions.php)

Select theme to edit: Kinsta Astra Child T

Selected file content:

```
1 <?php
2 /**
3  * Kinsta Astra Child Theme Theme functions and definitions
4  *
5  * @link https://developer.wordpress.org/themes/basics/theme-functions/
6  *
7  * @package Kinsta Astra Child Theme
8  * @since 1.0.0
9  */
10
11 /**
12  * Define Constants
13  */
14 define( 'CHILD_THEME_KINSTA_ASTRA_CHILD_THEME_VERSION', '1.0.0' );
15
16 /**
17  * Enqueue styles
18  */
19 function child_enqueue_styles() {
20
21     wp_enqueue_style( 'kinsta-astra-child-theme-theme-css',
22         get_stylesheet_directory_uri() . '/style.css', array( 'astra-theme-css' ),
23         CHILD_THEME_KINSTA_ASTRA_CHILD_THEME_VERSION, 'all' );
24 }
25 add_action( 'wp_enqueue_scripts', 'child_enqueue_styles', 15 );
```

Theme Files

This child theme inherits templates from a parent theme, Astra.

Stylesheet (style.css)

Theme Functions (functions.php)

Documentation:

So bearbeiten Sie die Datei functions.php von Ihrem WordPress-Dashboard aus.

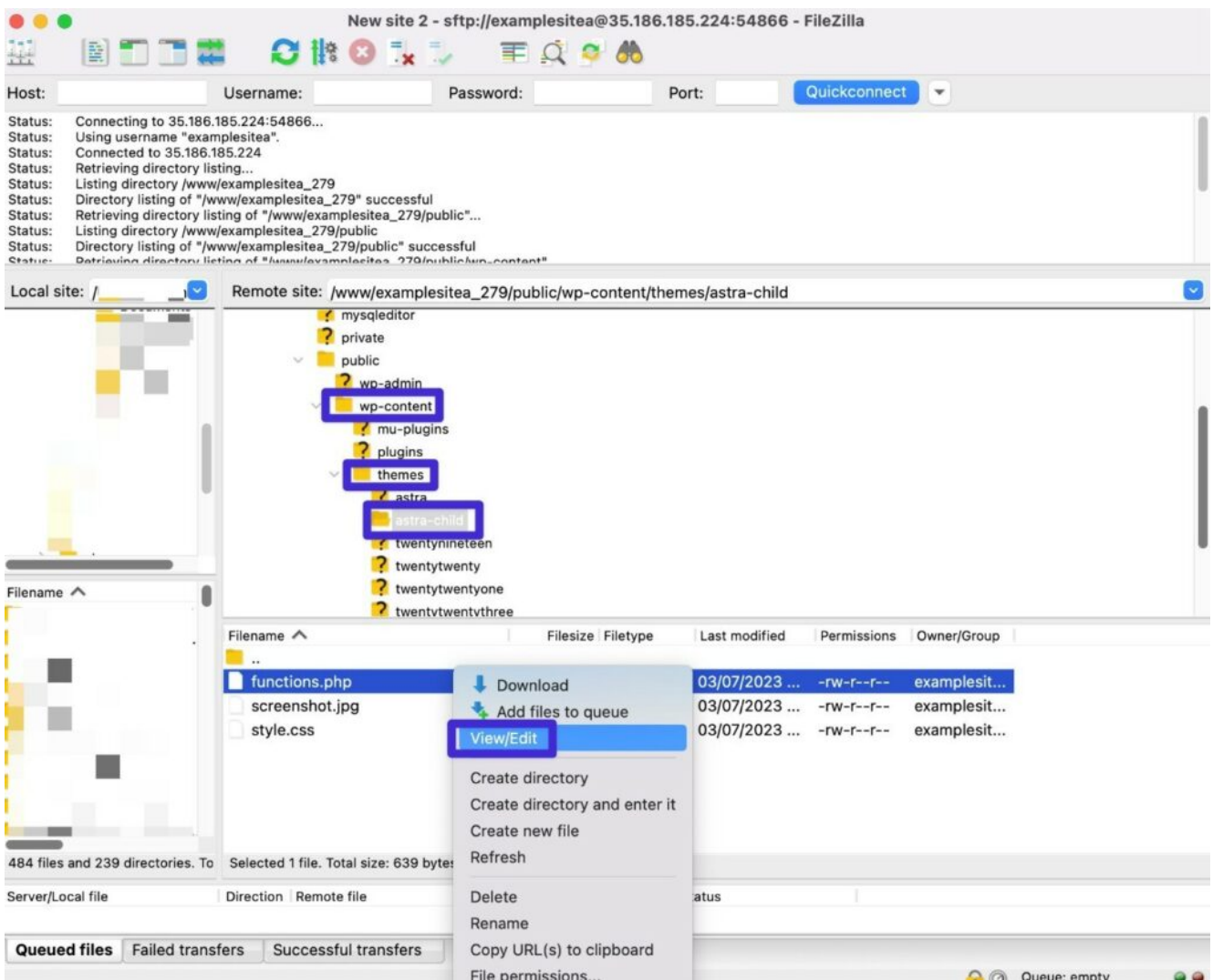
Abgesehen davon möchten viele Leute die [WordPress-Dateibearbeitung im Dashboard deaktivieren, um die Sicherheit zu verbessern](#) . In diesem Fall können Sie die nächste Methode verwenden.

2. Verwenden Sie SFTP + Ihren eigenen Code-Editor

Für eine andere Möglichkeit, die WordPress-Functions.php-Datei zu bearbeiten, kannst du dich [über FTP/SFTP mit deinem Server verbinden](#) und die Datei auf diese Weise bearbeiten.

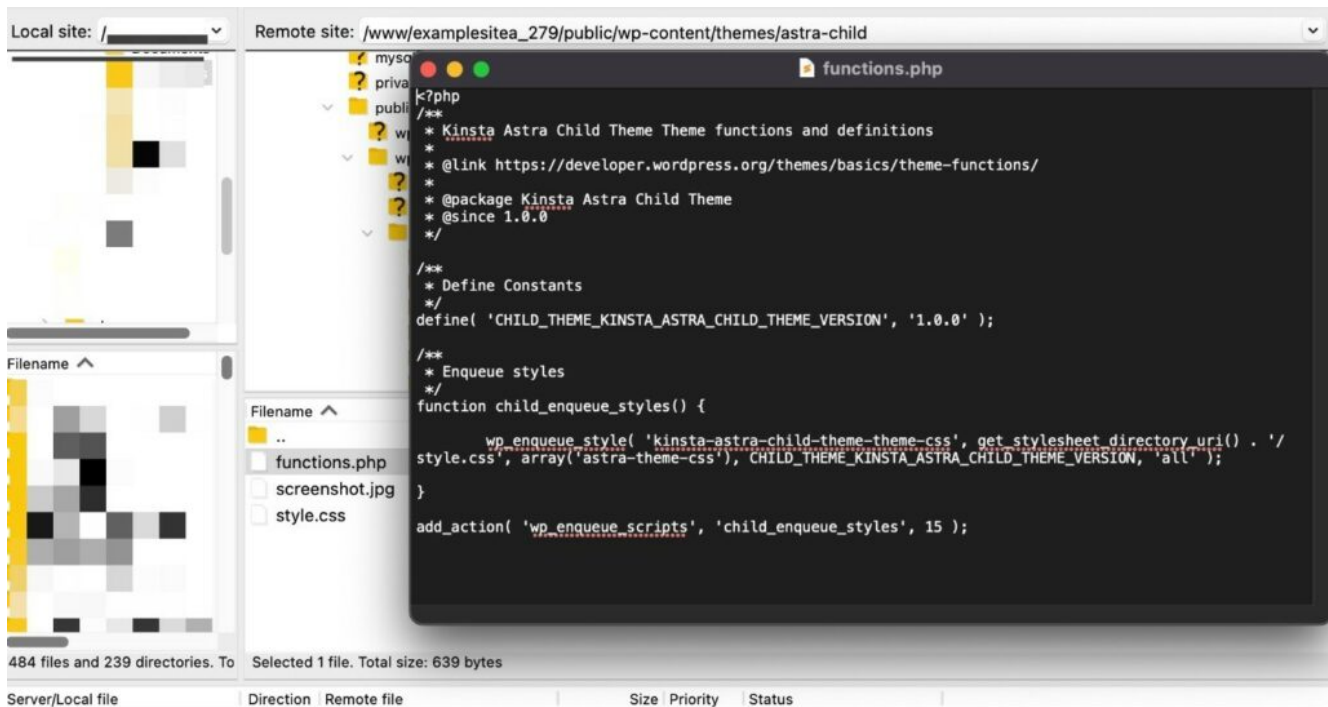
Hier ist wie:

1. Laden Sie [einen FTP-Client](#) herunter , falls Sie dies noch nicht getan haben – wir verwenden [den kostenlosen FileZilla-Client](#) für die Screenshots.
2. Stellen Sie mit Ihren FTP-Anmeldeinformationen eine Verbindung zu Ihrem Server her – so [verwenden Sie SFTP für die Verbindung](#) .
3. Navigieren Sie über die Ordnerstruktur zu **.../wp-content/themes/[Ihr-aktives-Child-Theme]**
4. Klicken Sie mit der rechten Maustaste auf die Datei **functions.php** und wählen Sie **Bearbeiten** .



So bearbeiten Sie die Datei `functions.php` mit FileZilla. Dadurch sollte die Datei auf Ihren lokalen Computer heruntergeladen und automatisch in einem Texteditor geöffnet

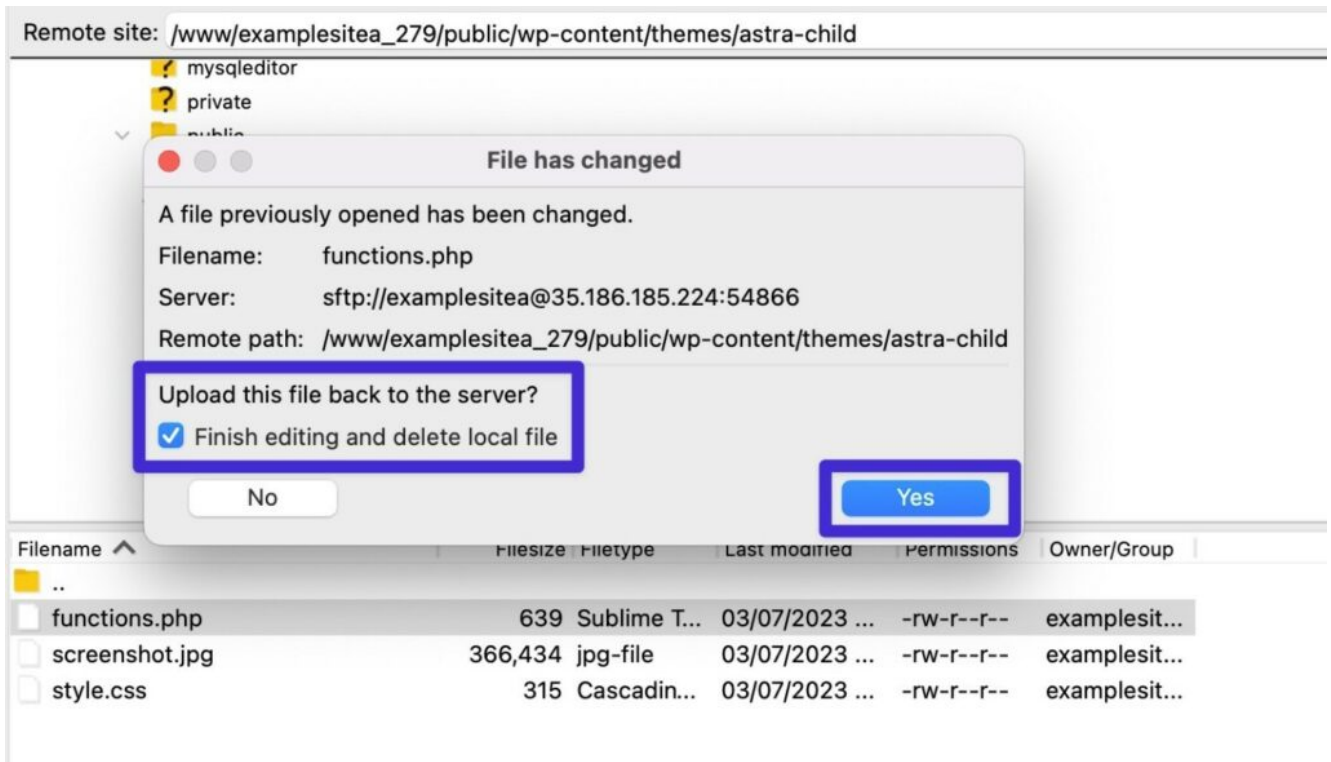
werden. Sie können der Datei jetzt Ihre Code-Snippets hinzufügen.



FileZilla öffnet die Datei in einem lokalen Texteditor – zB Sublime Text.

Wenn Sie fertig sind, speichern Sie Ihre Änderungen und schließen Sie die Datei.

FileZilla sollte dann eine Aufforderung anzeigen, die bearbeitete Version erneut auf Ihren Server hochzuladen.



Stellen Sie sicher, dass Sie die Datei functions.php erneut hochladen, sobald Sie Ihre Änderungen vorgenommen haben.

Die wichtigsten nützlichen Funktionen, die Sie Ihrer WordPress-Datei functions.php hinzufügen können

Nachdem Sie nun wissen, wie Sie die WordPress-Datei functions.php bearbeiten, gehen wir einige der nützlichsten Code-Snippets von functions.php durch, die Sie Ihrer Website hinzufügen können.

Datum der letzten Änderung in Blogbeiträgen anzeigen

Standardmäßig zeigen die meisten Themen das Datum an, an dem Sie einen Blogbeitrag veröffentlicht haben. Wenn Sie jedoch regelmäßig alte Inhalte aktualisieren und aktualisieren, möchten Sie möglicherweise auch das Datum der letzten Änderung anzeigen (oder das Veröffentlichungsdatum durch das Datum der

letzten Änderung ersetzen).

Dies lässt nicht nur Ihre menschlichen Besucher wissen, dass Sie den Inhalt noch aktualisieren, sondern zeigt Google auch, dass Ihr Inhalt aktuell ist.

```
function show_last_updated( $content ) {
    $u_time = get_the_time('U');
    $u_modified_time = get_the_modified_time('U');
    if ($u_modified_time >= $u_time + 86400) {
        $updated_date = get_the_modified_time('F jS, Y');
        $updated_time = get_the_modified_time('h:i a');
        $custom_content .= '<p class="last-updated-date">Recently
updated on '. $updated_date . ' at '. $updated_time .'</p>';
    }
    $custom_content .= $content;
    return $custom_content;
}
add_filter( 'the_content', 'show_last_updated' );
```

Weitere Einzelheiten finden Sie in unserem vollständigen Beitrag, [der das Datum der letzten Änderung in WordPress zeigt](#)

Deaktivieren Sie die WordPress-Symbolleiste für bestimmte Benutzer

Standardmäßig zeigt WordPress die WordPress-Symbolleiste für alle angemeldeten Benutzer an, die die integrierten [WordPress-Benutzerrollen](#) verwenden . *Dies gilt nicht unbedingt für benutzerdefinierte Rollen, wie z. B. die Benutzerrolle „Kunde“ für [WooCommerce-Shops](#) .*

Wenn Sie dies ändern möchten, können Sie die WordPress-Symbolleiste für bestimmte Benutzerrollen mithilfe eines Code-Schnipsels `functions.php` ausblenden.

Dieses Beispiel würde die Admin-Symbolleiste für Benutzer mit der Benutzerrolle „Autor“ einstellen.

```
add_filter( 'show_admin_bar', function( $show ) {
if ( current_user_can( 'author' ) ) {
return false;
}
return $show;
} );
```

Verzögern Sie das Erscheinen von Beiträgen im RSS-Feed

Standardmäßig werden Inhalte in Ihrem RSS-Feed angezeigt, sobald Sie sie veröffentlichen. Dies kann [Content-Scrapern](#) die Möglichkeit geben, [Ihre Inhalte zu scrapen](#) und [in Google zu indizieren](#), bevor Google Ihren ursprünglichen Beitrag überhaupt indiziert.

Um zu vermeiden, dass jemand anderes die Anerkennung für Ihre Inhalte erhält, können Sie das Erscheinen von Beiträgen in Ihrem RSS-Feed verzögern.

Dies wird Content Scraper zwar nicht vollständig stoppen, bedeutet aber, dass Sie Google Zeit geben können, Ihre Inhalte zuerst zu indizieren.

Hier ist das Code-Snippet – Sie können die Zahl – in diesem Beispiel „30“ – in die Anzahl der Minuten ändern, die die Verzögerung betragen soll:

```
function kinsta_delay_feed_content($where) {

global $wpdb;

if ( is_feed() ) {
// timestamp in WP-format
$now = gmdate('Y-m-d H:i:s');

// value for wait; + device
$wait = '30'; // integer

//
```

http://dev.mysql.com/doc/refman/5.0/en/date-and-time-functions.html#function_timestampdiff

```
$device = 'MINUTE'; //MINUTE, HOUR, DAY, WEEK, MONTH, YEAR
```

```
// add SQL-syntax to default $where  
$where .= " AND TIMESTAMPDIFF($device,  
$wpdb->posts.post_date_gmt, '$now') > $wait ";  
}  
return $where;  
}
```

```
add_filter('posts_where', 'kinsta_delay_feed_content');
```

Bestimmte Kategorien vom RSS-Feed ausschließen

Zusätzlich zum Verzögern des Erscheinens von Inhalten im RSS-Feed kann es auch Situationen geben, in denen Sie ganze Kategorien von Blogbeiträgen ausschließen möchten, damit sie nicht im RSS-Feed Ihrer Website erscheinen.

Um dies zu erreichen, können Sie das folgende Code-Snippet `functions.php` verwenden:

```
function kinsta_exclude_category_rss($query) {  
if ($query->is_feed) {  
$query->set('cat', '-38');  
}  
return $query;  
}  
add_filter('pre_get_posts', 'kinsta_exclude_category_rss');
```

Stellen Sie sicher, dass Sie die Beispiel-Kategorie-ID – „38“ – durch die tatsächlichen Kategorie-IDs ersetzen, die Sie ausschließen möchten (und entfernen Sie nicht das Minuszeichen).

So finden Sie die ID-Nummer:

1. Gehen Sie zu **Beiträge → Kategorien** , um eine Liste aller

Kategorien auf Ihrer Website anzuzeigen.

2. Klicken Sie unter der Kategorie, die Sie ausschließen möchten, auf **Bearbeiten** .
3. Sehen Sie sich die URL der Bearbeitungsseite in der Adressleiste Ihres Browsers an. Die Kategorie-ID ist die Zahl, die nach `?taxonomy=category&tag_ID=X` kommt .

Wenn die URL beispielsweise `https://yoursite.com/wp-admin/term.php?taxonomy=category&tag_ID=38&post_type=post` lautet , lautet die Kategorie-ID **38** .

Begrenzen Sie WordPress-Revisionen nach Beitragstyp

Um zu vermeiden, dass die Datenbank Ihrer Website aufgebläht wird, sollten Sie die Anzahl der zu speichernden Post-Revisionen begrenzen.

[Während eine Möglichkeit, WordPress-Beitragsrevisionen](#) seitenweit einzuschränken , darin besteht, [die wp-config.php-Datei deiner Website](#) zu bearbeiten , möchtest du vielleicht unterschiedliche Anzahlen von Revisionen für verschiedene Beitragstypen speichern – z. B. Blog-Beiträge im Vergleich zu WooCommerce-Produkten.

Mit diesem Code-Snippet von `functions.php` können Sie genau das tun. Um es zu Ihrem eigenen zu machen, ändern Sie den Beitragstyp – in diesem Beispiel „Beitrag“ – in den tatsächlichen Beitragstyp, den Sie steuern möchten, und die Zahl – in diesem Beispiel „5“ – in die Anzahl der Revisionen, die Sie speichern möchten.

```
function kinsta_post_revisions_by_type( $revisions, $post ) {  
  
if( 'post' == $post->post_type ) {  
$revisions = 5;  
}  
}
```

```
return $revisions;
```

```
}
```

```
add_filter( 'wp_revisions_to_keep',  
'kinsta_post_revisions_by_type', 10, 2 );
```

* Hinweis – Sie bearbeiten den „post“, der unmittelbar nach der „if“-Anweisung kommt. Zum Targeting von Produkten würde diese Zeile beispielsweise so aussehen:

```
if( 'product' == $post->post_type ) {
```

Deaktivieren Sie die WordPress-Suchfunktion

Wenn Ihre Website keine Suchfunktion benötigt, sollten Sie [die WordPress-Suchfunktion](#) deaktivieren , um potenziellen Missbrauch zu verhindern.

Selbst wenn Sie beispielsweise kein Suchfeld auf Ihrer Website anzeigen, können Bots dennoch die Suchfunktion verwenden, indem sie Ihrer URL `?s=[Suchbegriff]` hinzufügen.

Dazu können Sie das folgende Code-Snippet hinzufügen:

```
function kinsta_disable_wp_search( $query, $error = true ) {
```

```
if ( is_search() ) {  
$query->is_search = false;  
$query->query_vars[s] = false;  
$query->query[s] = false;
```

```
// to error  
if ( $error == true )  
$query->is_404 = true;  
}  
}
```

```
add_action( 'parse_query', 'kinsta_disable_wp_search' );  
add_filter( 'get_search_form', create_function( '$a', "return
```

```
null;" ) );
```

Wenn jemand versucht, die Suchfunktion direkt zu verwenden, gibt Ihre Website stattdessen eine 404-Seite zurück.

Erstellen Sie Ihre eigenen benutzerdefinierten Shortcodes (zB aktuelles Jahr)

[WordPress-Shortcodes](#) sind wirklich praktische Verknüpfungen, mit denen Sie dynamische Inhalte in Ihre Website einbetten können.

Während viele WordPress-Plugins auf Shortcodes angewiesen sind, um ihre Funktionalität zu verbessern, können Sie auch Ihre eigenen benutzerdefinierten Shortcodes erstellen, indem Sie die WordPress-Datei `functions.php` und [die Funktion `add_shortcode\(\)`](#) verwenden .

Ein häufiger Anwendungsfall ist beispielsweise die Erstellung eines Shortcodes, der das aktuelle Jahr anzeigt. Auf diese Weise können Sie automatisch das aktuelle Jahr in all Ihren Inhalten anzeigen, ohne jeden 1. Januar alles manuell aktualisieren zu müssen.

Um einen solchen Shortcode zu erstellen, können Sie den folgenden Code verwenden:

```
function year_shortcode() {
$year = date('Y');
return $year;
}
add_shortcode('currentyear', 'year_shortcode');
```

Wenn Sie dann den Shortcode `[aktuelles Jahr]` zu Ihren Inhalten hinzufügen, wird dieser automatisch durch das tatsächliche Jahr ersetzt – z. B. 2023.

Shortcodes in Beitragstiteln zulassen

Standardmäßig führt WordPress keine Shortcodes aus, die Sie im Titelfeld Ihres Beitrags platzieren. Wenn Sie Shortcodes in WordPress-Beitragstiteln verwenden möchten, können Sie diese Funktionalität aktivieren, indem Sie der WordPress-Datei `functions.php` ein einfaches Code-Snippet hinzufügen.

In Kombination mit dem vorherigen Code-Snippet können Sie beispielsweise automatisch das aktuelle Jahr in den Beitragstitel einfügen, indem Sie den Shortcode `[aktuelles Jahr]` hinzufügen.

```
add_filter( 'the_title', 'do_shortcode' );
```

Anmeldefehler ausblenden, um die Sicherheit zu verbessern

Standardmäßig zeigt WordPress bei fehlgeschlagenen Anmeldeversuchen eine erklärende Nachricht an. Diese Nachricht kann einige Informationen preisgeben, z. B. ob eine bestimmte E-Mail-Adresse auf Ihrer Website registriert ist oder nicht (selbst wenn das Passwort falsch ist).

Um zu vermeiden, dass möglicherweise vertrauliche Informationen preisgegeben werden, können Sie diese Anmeldefehler ausblenden, indem Sie das folgende Code-Snippet zu Ihrer Datei `functions.php` hinzufügen.

```
function kinsta_hide_login_errors(){  
    return 'These are not valid credentials';  
}  
add_filter( 'login_errors', 'kinsta_hide_login_errors' );
```

Dieses Snippet ersetzt den standardmäßigen Anmeldefehlertext durch die folgende Meldung – **Dies sind keine gültigen Anmeldeinformationen** .

Sie können den Text nach Ihren Wünschen anpassen – vermeiden

Sie jedoch Bindestriche, Schrägstriche oder andere Zeichen, da dies zu Fehlern führen kann (es sei denn, Sie „escapen“ diese Zeichen).

Ändern Sie die Ausschnittlänge

Standardmäßig zeigt der WordPress-Auszug die ersten 55 Wörter eines Beitrags.

Wenn Sie dies anpassen möchten, können Sie das folgende Code-Snippet in die Datei `functions.php` einfügen.

```
function kinsta_change_excerpt_length($length) {  
    return 90;  
}  
add_filter('excerpt_length', 'kinsta_change_excerpt_length');
```

Das obige Code-Snippet ändert die Länge des Auszugs auf 90 Wörter. Wenn Sie eine andere Anzahl bevorzugen, können Sie „90“ durch die genaue Anzahl der Wörter ersetzen, die Sie verwenden möchten.

Entfernen Sie die WordPress-Versionsnummer

Um es böswilligen Akteuren zu erschweren, zu erkennen, welche Version von WordPress Sie verwenden, entfernen einige Benutzer gerne die WordPress-Versionsnummer aus dem Frontend-Code Ihrer Website.

Dazu können Sie Ihrer Datei `functions.php` die folgenden Codeschnipsel hinzufügen.

```
function kinsta_hide_version() {  
    return '';  
}  
add_filter('the_generator', 'kinsta_hide_version');
```

Richten Sie die WordPress-Paginierung ein

Wenn Sie mit der Funktionsweise der Paginierung in Ihrem Design nicht zufrieden sind, können Sie ein `functions.php`-Code-Snippet verwenden, um das Paginierungsverhalten anzupassen oder sogar Ihr eigenes Paginierungssystem zu erstellen.

Wenn Sie an dieser Art von Modifikation interessiert sind, haben wir [eine vollständige Anleitung zur WordPress-Paginierung](#), die einige Beispiele für `functions.php`-Code-Snippets für Sie enthält.

Ändern Sie, wie viele Ergebnisse auf der Suchlistenseite enthalten sein sollen

WordPress bietet Ihnen eine In-Dashboard-Option, um zu steuern, wie viele Beiträge auf Archivseiten aufgelistet werden sollen (**Einstellungen** → **Lesen**).

Dadurch ändert sich jedoch die Nummer für *alle* Archivseiten. Was ist, wenn Sie nur eine andere Nummer für die Suchergebnisseite verwenden möchten?

Dazu können Sie dieses Snippet zu Ihrer `functions.php`-Datei hinzufügen – stellen Sie sicher, dass Sie die Zahl („12“ in diesem Beispiel) auf die tatsächliche Anzahl der Ergebnisse ändern, die Sie anzeigen möchten, bevor Sie zusätzliche Ergebnisse paginieren.

```
function kinsta_search_results_list() {
    if ( is_search() )
        set_query_var('posts_per_archive_page', 12);
}

add_filter('pre_get_posts', 'kinsta_search_results_list');
```

Fügen Sie ausgewählte Bilder in den WordPress-RSS-Feed ein

[Wenn Sie die vorgestellten Bilder Ihrer Posts](#) in Ihren RSS-Feed-Inhalt aufnehmen möchten, können Sie das folgende Snippet zu Ihrer functions.php-Datei hinzufügen.

```
function kinsta_featured_image_rss($content) {
global $post;
if ( has_post_thumbnail( $post->ID ) ){
$content = '<div>' . get_the_post_thumbnail( $post->ID,
'full', array( 'style' => 'margin-bottom: 15px;' ) ) .
'</div>' . $content;
}
return $content;
}
```

```
add_filter('the_excerpt_rss', 'kinsta_featured_image_rss');
add_filter('the_content_feed', 'kinsta_featured_image_rss');
```

Wie geschrieben, wird das obige Code-Snippet das Bild in voller Größe einfügen. Um eine andere Thumbnail-Größe zu verwenden, können Sie „Full“ auf eine andere Thumbnail-Größe ändern – zB „Large“ oder „Medium“.

Upload-Unterstützung für Dateitypen hinzufügen (z. B. SVG)

Standardmäßig blockiert WordPress das Hochladen bestimmter Dateitypen, z. B. [SVG-Dateien](#).

Um die Unterstützung für diese blockierten Dateitypen zu aktivieren, können Sie das folgende Code-Snippet zur WordPress-Datei functions.php hinzufügen:

```
function kinsta_myme_types($mime_types){
$mime_types['svg'] = 'image/svg+xml';
return $mime_types;
}
add_filter('upload_mimes', 'kinsta_myme_types', 1, 1);
```

Dieses Code-Snippet ermöglicht nur SVG-Uploads, aber Sie können es ändern, um bei Bedarf Unterstützung für andere Dateitypen hinzuzufügen.

Ausblenden von WordPress-Update-Nörgeln für Nicht-Administratoren

Standardmäßig zeigt WordPress allen Benutzern mit Dashboard-Zugriff Dashboard-weite Nachrichten an, wenn [ein neues Update verfügbar](#) ist .

Selbst wenn die Rolle eines Benutzers es ihm nicht erlaubt, das Update anzuwenden, weist die Dashboard-Meldung diesen Benutzer an, sich an den Administrator zu wenden.

Um dies anzupassen, können Sie dieses Code-Snippet verwenden, um Update-Benachrichtigungen für alle Nicht-Administrator-Benutzer auszublenden:

```
function kinsta_hide_update_nag() {  
if ( ! current_user_can( 'update_core' ) ) {  
remove_action( 'admin_notices', 'update_nag', 3 );  
}  
}
```

```
add_action('admin_menu','kinsta_hide_update_nag');
```

Für einige andere Variationen sieh dir unsere [Anleitung zum Deaktivieren von WordPress-Update-Benachrichtigungen](#) an .

Ändern Sie die Qualität der automatischen JPEG-Optimierung

Obwohl nicht jeder von dieser Funktion weiß, [komprimiert WordPress automatisch JPEG-Bilder](#) , wenn [Miniaturansichten für Ihre Website erstellt werden](#), nachdem Sie Bilder hochgeladen haben.

Seit WordPress 4.5 hat WordPress die Standardqualitätsstufe

auf 82 eingestellt (wobei 100 eine Null-Komprimierung ist).

Wenn Sie diese Qualitätsstufe höher oder niedriger machen möchten, können Sie das folgende Code-Snippet zu Ihrer Datei `functions.php` hinzufügen – stellen Sie sicher, dass Sie die Zahl („90“ in diesem Beispiel) auf die tatsächliche Qualitätsstufe ändern, die Sie verwenden möchten :

```
add_filter( 'jpeg_quality', create_function( '', 'return 90;' ) );
```

Best Practices für die Organisation Ihrer WordPress-functions.php-Datei

Wenn Sie nur die WordPress-Datei `functions.php` verwenden, um ein paar Codeschnipsel hinzuzufügen, müssen Sie sich wahrscheinlich keine Gedanken über die Organisation machen.

Wenn Sie Ihrer Website jedoch viele Code-Snippets hinzufügen, kann dies schnell unhandlich und komplex werden, wenn Sie einige Best Practices für die Organisation von `functions.php`-Dateien nicht befolgen.

Hier sind ein paar Best Practices, um Ihre Code-Snippets zu organisieren...

Fügen Sie Codekommentare hinzu, um alles zu erklären

Wenn Sie Ihrer `functions.php`-Datei zum ersten Mal ein Code-Snippet hinzufügen, wissen Sie genau, was das Snippet bewirkt und warum Sie es hinzugefügt haben. Aber wenn Sie ein Jahr später auf die Datei `functions.php` zurückblicken, sind die Dinge vielleicht nicht so klar.

Um dies zu vermeiden, sollten Sie jedem Snippet Codekommentare hinzufügen, die erklären, was das Snippet tut und warum Sie es hinzugefügt haben.

Code-Kommentare sind Text, der nicht von WordPress ausgeführt wird, aber den menschliche Benutzer lesen können, um zu verstehen, was das Code-Snippet tut.

Um einen einzeiligen Codekommentar hinzuzufügen, können Sie dieses Format verwenden:

```
// This is a single-line code comment
```

Um einen mehrzeiligen Codekommentar hinzuzufügen, können Sie dieses Format verwenden:

```
/**  
 * This is a code comment that spans multiple lines  
 * This text is also in the same comment  
 * This is the last line in the code comment  
 */
```

Hier ist ein Beispiel dafür, wie Sie Codekommentare verwenden könnten – Sie können sehen, dass der Kommentar vor jedem Snippet erklärt, was das Snippet tut.

Kinsta Astra Child Theme: Stylesheet (style.css)

Select theme to edit: Kinsta Astra Child T

Selected file content:

```
1 /**
2 Theme Name: Kinsta Astra Child Theme
3 Author: Kinsta
4 Author URI: https://kinsta.com
5 Description: A Kinsta child theme for Astra.
6 Version: 1.0.0
7 License: GNU General Public License v2 or later
8 License URI: http://www.gnu.org/licenses/gpl-2.0.html
9 Text Domain: kinsta-astra-child-theme
10 Template: astra
11 */
12
13 // Hides update notices for non-administrators
14
15 function kinsta_hide_update_nag() {
16 if ( ! current_user_can( 'update_core' ) ) {
17 remove_action( 'admin_notices', 'update_nag', 3 );
18 }
19 }
20
21 add_action( 'admin_menu', 'kinsta_hide_update_nag' );
22
23 // Adds the last modified date to the single posts page
24
25 function show_last_updated( $content ) {
26 $u_time = get_the_time( 'U' );
27 $u_modified_time = get_the_modified_time( 'U' );
28 if ( $u_modified_time >= $u_time + 86400 ) {
29 $updated_date = get_the_modified_time( 'F jS, Y' );
30 $updated_time = get_the_modified_time( 'h:i a' );
31 $custom_content .= '<p class="last-updated-date">Recently updated on '. $updated_date . ' at '.
$updated_time . '</p>';
32 }
33 $custom_content .= $content;
34 return $custom_content;
35 }
36 add_filter( 'the_content', 'show_last_updated' );
37
38
39
40
```

Theme Files

This child theme inherits templates from a parent theme, Astra.

Stylesheet
(style.css)

Theme Functions
(functions.php)

Ein Beispiel für die Verwendung von Codekommentaren zum Dokumentieren von Snippets in der Datei functions.php.

Verwenden Sie Dateien einschließen, um Snippets in verschiedene Dateien aufzuteilen

Wenn Sie wirklich viele Codeschnipsel haben, können Sie sie in separaten Dateien speichern, anstatt nur die Datei functions.php zu verwenden.

Anschließend können Sie diese Codeschnipsel mithilfe von include oder require in die Datei functions.php einfügen. [Mike Schinkel hat ein gutes Beispiel](#) dafür, wie Sie dies auf StackExchange tun könnten.

Betrachten Sie eine der `functions.php`-Alternativen von oben

Wenn Sie feststellen, dass Ihre Datei `functions.php` etwas unübersichtlich wird, bevorzugen Sie vielleicht eine der `Funktionen.php`-Alternativen, die wir zuvor beschrieben haben.

[Das kostenlose Code-Snippets-Plug-in](#) macht es beispielsweise sehr einfach, alles zu organisieren, da Sie jedes Code-Snippet separat hinzufügen können, komplett mit seinem eigenen Titel und seiner eigenen Beschreibung. Sie können auch Tags verwenden, um Ihre Snippets zu organisieren.