19 Arbeiten mit Child-Themes

In diesem Kapitel erfahren Sie ...

- ... wie Sie ein Child-Theme erstellen,
- ... wie Sie ein Child-Theme aktivieren,
- ... wie Sie das Standard-Theme Twenty Nineteen mit einem Child-Theme anpassen.

Manchmal möchte man an einem Theme weitere Anpassungen vornehmen, als jene mittels Customizer. Wenn Sie Änderungen in Original-Theme-Dateien durchführen, werden diese beim nächsten Theme-Update überschrieben. Um dies zu verhindern, werden Änderungen in sogenannte *Child-Themes* geschrieben. Wenn ein Child-Theme vorhanden ist, schaut WordPress zuerst in der Child-Theme-Datei nach und überschreibt mit den dort gefundenen Angaben die Original-Datei. Anders bei der `functions.php` im Child Theme. Diese Datei wird unmittelbar vor dem Parent-Theme `functions.php` geladen, die Funktionen etc. werden dem Parent-Theme `functions.php` hinzugefügt.

Für Änderungen in einzelnen Template-Dateien kopiert man die jeweilige Datei in den Child-Theme-Ordner und führt dort die Änderung durch. Bei einem Update bleiben die Child-Theme-Dateien unberührt. Dies hat allerdings auch den Nachteil, dass Korrekturen von etwa Fehlern oder Sicherheitslücken mit dem nächsten Update im Parent-Theme korrigiert und behoben werden können, nicht jedoch automatisch auch im Child-Theme.

19.1 Child-Theme einrichten

Für ein *Child-Theme* benötigen Sie im Ordner `wp-content/themes` einen neuen Ordner mit dem Namen des Original-Themes gefolgt von `-child`. Achten Sie darauf, dass sich keine Leerzeichen im Namen befinden, sonst erhalten Sie Fehlermeldungen. Im Beispiel sollen Änderungen an *Twenty Nineteen* vorgenommen werden, somit heißt der Child-Ordner `twentynineteen-child`. In diesem Ordner benötigen Sie zwei neue leere Dateien, `functions.php` und `style.css` (siehe Bild 19.1).

Anmerkung: Sie können den Child-Theme-Ordner theoretisch beliebig benennen, solange Sie nur Kleinbuchstaben und keine Sonderzeichen verwenden. Praktisch empfehle ich besonders am Anfang, das Parent-Theme im Namen beizubehalten.

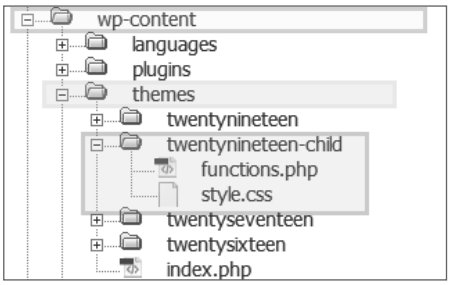

Bild 19.1
Ordnerstruktur bei Verwendung eines Child-Themes

19.1.1 Die style.css im Child-Theme

Im CSS-Header, d. h. am Anfang der Datei `style.css`, müssen zumindest die Angaben *Theme Name* und *Template* enthalten sein. Was bei *Theme Name* angegeben wird, erscheint in der Theme-Verwaltung als Name Ihres neuen Themes. Neben *Template* muss der Name des Eltern-Themes angegeben werden und zwar in der korrekten Schreibweise. *Author*, also Ihren Namen, die *Version* und *Description*, eine kurze Beschreibung, würde ich auch jedenfalls angeben. Demnach sieht der CSS-Header meiner Beispiel-Datei folgendermaßen aus (siehe Listing 19.1):

Listing 19.1 Der CSS-Header des Child-Themes

```
1 /*
2 Theme Name:   Mein neues Twenty Nineteen Child-Theme
3 Description:  WP5-Buch 2019 Twenty Nineteen Child-Theme
4 Author:       Jola Belik
5 Template:     twentynineteen
6 Version:      1.0.0
7 */
```

Im Folgenden finden Sie eine Auflistung aller Angaben, die erwartet werden, wenn Sie Ihr Theme anderen zur Verfügung stellen möchten. Die Zahlen in Klammern zeigen die Position der Angaben in der Themes-Verwaltung in Bild 19.3 und Bild 19.4:

- *Theme name* (1) – Der Name Ihres Child-Themes, Angabe zwingend erforderlich.
- *Theme URI* – Angabe einer Website, in der das Theme beschrieben wird bzw. Beispiel-Anwendungs-Seite, diese Angabe oder die Angabe der Website des Autors bzw. der Autorin ist erforderlich, damit Ihr Theme in der WordPress directory akzeptiert wird.
- *Description* (2) – Kurze Beschreibung des Themes, wird angezeigt, wenn Sie in der Themes-Verwaltung auf **Theme Details** klicken.
- *Author* (3) – Name des Autors bzw. der Autorin, also Ihr Name.
- *Author URI* – Die Adresse Ihrer Website (siehe Theme URI).

- *Template* – Hier steht der Name des Parent-Themes und zwar der Name des Ordners. Diese Angabe ist zwingend erforderlich. Achten Sie auf die korrekte Schreibweise, diese Angabe ist case-sensitiv. Ist dieser Wert falsch angegeben, erhalten Sie Fehlermeldungen statt eines funktionierenden Child-Themes.
- *Version* (4) – Die Versionsnummer beginnt üblicherweise mit 1.0.0.
- *License* – Üblicherweise werden Themes in der WordPress directory unter einer GPL license veröffentlicht. Am besten ist es, Sie übernehmen die Lizenz des Parent-Themes (z. B. `GNU General Public License v2 or later`).
- *License URI* – Die Adresse, unter der eine Beschreibung der verwendeten Lizenz zu finden ist. Auch hier sollten Sie sich an die Angabe des Parent-Themes halten (z. B. `http://www.gnu.org/licenses/gpl-2.0.html`).
- *Tags* – Stichwörter, sogenannte Tags, erleichtern das Finden von passenden Themes in der WordPress directory. Geben Sie hier aussagekräftige Stichwörter an, die Ihr Thema beschreiben (z. B. `one-column`, `two-columns`, `responsive`, `blau`, `custom-header` etc.).
- *Text domain* – Die Angabe des Theme-Slugs wird für die Internationalisierung verwendet, damit Ihr Theme in andere Sprachen übersetzt werden kann (für das Beispiel-Child-Theme wäre es `twentynineteen`).

Fügen Sie den Code aus Listing 19.1 – mit Ihren individuellen Angaben versehen – am Beginn der `style.css` im Ordner `twentynineteen-child` ein und speichern Sie die Datei.

19.1.2 Die functions.php im Child-Theme

Damit ein Child-Theme funktionieren kann, müssen die Stylesheets im Child-Theme und im Parent-Theme miteinander verknüpft werden. Dies wurde in früheren Versionen mit `@import` in der Child-CSS gemacht. Diese Vorgangsweise ist *nicht* mehr State of the Art, im WordPress Codex wird stattdessen die Verwendung von `wp_enqueue_script()` und `wp_enqueue_style()` im Child-Theme `functions.php` empfohlen.

Fügen Sie deshalb folgenden Code in die `functions.php` im Child-Theme-Ordner ein (siehe Listing 19.2): Achten Sie darauf, dass sich vor dem öffnenden PHP-Tag `<?php` weder eine Leerzeile noch ein Leerzeichen befindet!

Listing 19.2 Verknüpfen von Child- und Parent-Stylesheets

```
01 <?php
02 function jwp5buch_enqueue_styles() {
03    $parent_style = 'parent-style';
04    wp_enqueue_style( $parent_style, get_template_directory_uri()
   . '/style.css' );
05    wp_enqueue_style( 'child-style',
06        get_stylesheet_directory_uri() . '/style.css',
07        array( $parent_style ),
08        wp_get_theme()->get('Version')
09    );
10 }
11
12 add_action( 'wp_enqueue_scripts', 'jwp5buch_enqueue_styles' );
```

19.1.3 Das Child-Theme aktivieren

Das Child-Theme ist nun bereit, aktiviert zu werden. Es ist war nicht unbedingt erforderlich, jedoch empfehlenswert und praktischer, wenn Sie – wie für jedes Theme üblich – eine `screenshot.png`-Datei erstellen und in den Child-Theme-Ordner geben (siehe Bild 19.2, rechts). Mit dem Vorschaubild Ihres Child-Themes soll das Theme auf den ersten Blick erkennbar sein. Die erforderliche Mindestgröße, in der diese Vorschaudatei in der Theme-Verwaltung in der Themes-Übersicht angezeigt wird, ist 387 × 290 Pixel. Allerdings wird die Datei bei der Ansicht der Theme-Details abhängig von der Screen-Größe des Ausgabegeräts deutlich größer angezeigt. Bis vor etwa zwei Jahren war die Empfehlung deshalb 880 × 660 Pixel. Damit die Vorschau auch auf hochauflösenden Ausgabegeräten gut aussieht, wird aktuell im Codex eine Größe von 1200 × 900 Pixel empfohlen. Da im oberen und im unteren Teil der Grafik Teile in der Theme-Verwaltung abgedeckt sind, ist es ratsam, wichtige Logos etc. auf dem Screenshot weiter mittig zu platzieren, dass sie sichtbar bleiben. Sie können dafür eine beliebige PNG-Datei oder beispielsweise einen Screenshot der bisher angepassten Seite *Twenty Nineteen* erstellen. Ich verwende einen Screenshot in der Größe von 880 × 660 Pixel (siehe Bild 19.2, links).

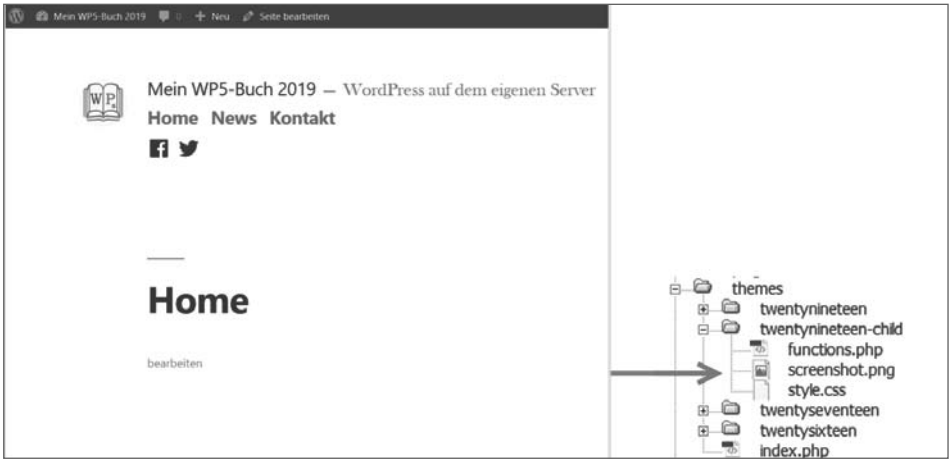

Bild 19.2 Die Datei screenshot.png im Child-Theme-Ordner (rechts)

Öffnen Sie nun die *Themes-Verwaltung* auf dem Dashboard über *Design/Themes*. An erster Stelle ist immer das aktuell aktivierte Theme zu sehen. Das neue Child-Theme *Mein neues Twentynineteen Child Theme* wird im Beispiel gleich daneben angezeigt (siehe Bild 19.3).

Bild 19.3 Das neue Child-Theme (rechts) in der Themes-Verwaltung

Fahren Sie mit der Maus über das Vorschaubild und klicken Sie auf **Theme-Details**. Sie sehen nun die Angaben, die im CSS-Header des Child-Themes eingefügt wurden: *Theme Name* (1), *Versionsnummer* (4), *Autorname* (3) und die *Beschreibung* (2). Außerdem erscheint der Hinweis, dass es sich um ein *Child-Theme von Twenty Sixteen* handelt (siehe Bild 19.4).

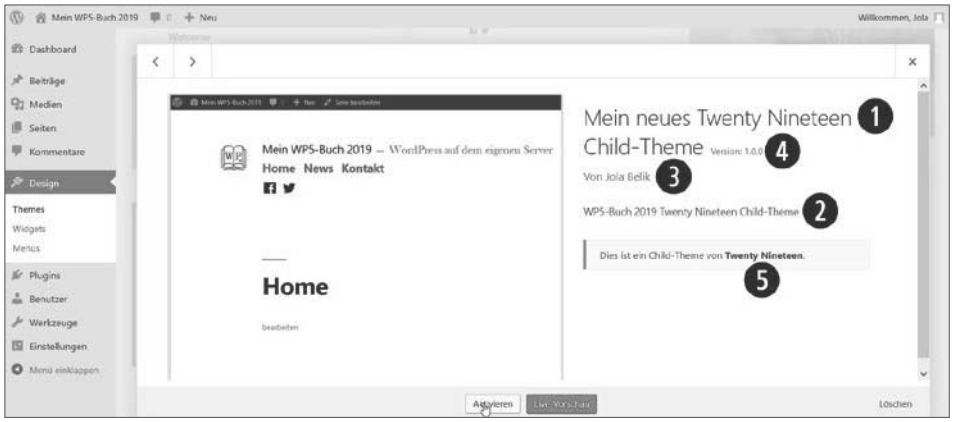

Bild 19.4 Aktivieren Sie das neue Child-Theme

Sie können nun das neue Child-Theme aktivieren. Klicken Sie unterhalb des Vorschaubilds auf **Aktivieren** (siehe Bild 19.4). Warten Sie, bis das neue Theme aktiviert wurde, das kann ein Weilchen dauern. Wenn alles erfolgreich durchgeführt werden konnte, erscheint oben auf der Seite der Themes-Verwaltung eine Erfolgsmeldung (siehe Bild 19.5).

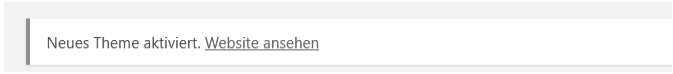

Bild 19.5 Das neue Child-Theme wurde erfolgreich aktiviert!

Gratulation, Sie können nun mit dem Child-Theme arbeiten und weitere Anpassungen vornehmen. Die Freude währt allerdings nur so lange, bis Sie sich das Frontend anschauen. Statt den bisher vorgenommenen Änderungen und Anpassungen erscheint der Widget-Titel wieder riesengroß, zudem ist das Website-Logo verschwunden (siehe Bild 19.6).

Bild 19.6 Das Frontend nach der Aktivierung des Child-Themes

Nein, es ist kein Fehler in Ihrem Code, der das verursacht. Es ist lediglich die übliche Problematik beim Aktivieren eines neuen Themes. Sie müssen in den meisten Fällen nicht nur die bereits erwähnten „verschwundenen“ Anpassungen wieder durchführen. Wenn im neuen Theme Menü- und Widgets-Positionen anders benannt sein sollten (was im Beispiel-Child-Theme nicht der Fall ist), müssen zusätzlich auch die Menüs neu zugewiesen und die Widgets neu hinzugefügt werden.

19.2 Änderungen im Child-Theme

Nachdem unser Child-Theme aktiviert wurde, kann das Theme nach Belieben angepasst werden. Als Beispiele in diesem Kapitel werden die Copyright-Zeile mit Anmelde-Link im Footer und RSS-Link samt RSS-Icon sowie diverse Änderungen bei Textformatierungen und Anzeige eines größeren Site-Logos gezeigt. In einigen anderen Kapiteln werden weitere Änderungen bzw. Erweiterungen des Themes beschrieben wie beispielsweise das Hinzufügen eines neuen Widgets-Bereichs (siehe Abschnitt 7.1) und Hinzufügen einer benutzerdefinierten Bildgröße in der Mediathek. Vorher wurden die im Customizer (Kapitel 6) gemachten Änderungen bezüglich Site-Logo und Farben nochmals durchgeführt – als primäre Farbe wurde nun Rot statt Blau gewählt. Das Site-Logo muss nicht nochmals hochgeladen werden, da es sich in der Mediathek befindet.

19.2.1 Neue Footer-Zeile

Als Erstes werden einige Änderungen im *Footer*, konkreter gesagt in der Datei `footer.php` im Child-Theme-Ordner, vorgenommen. Im Folgenden wird eine neue Copyright-Zeile mit diversen Links erstellt.

19.2.1.1 Neue ©-Zeile im Footer

Der Footer mit der Zeile *Mein WP5-Buch 2019, Stolz präsentiert von WordPress.* und der Verlinkung mit `wordpress.org` mag ja nett sein (siehe Bild 19.7, oben). Wenn Sie unter *Einstellungen/Datenschutz* die Seite für die Datenschutzerklärung festgelegt und diese Seite unter *Seiten* bearbeitet und veröffentlicht haben, so wird im Footer auch der Link zur *Datenschutzerklärung* angezeigt (siehe Bild 19.7, oben). Sinnvoller fände ich eine Fußzeile mit einem Copyright-Hinweis samt Namen und Jahreszahl und ohne Verlinkung zu *wordpress.org* sowie den Link zur Datenschutzerklärung und einen Link zum Anmelden (siehe Bild 19.7, unten).

Bild 19.7 Fußzeile in Twenty Nineteen ohne (oben) und mit veröffentlichter Seite Datenschutzerklärung (unten)

Link zur Datenschutzerklärung Ob der Link zur *Datenschutzerklärung* im Footer oder an einer anderen Stelle platziert wird, ist individuell zu entscheiden. Mit folgenden Code-Zeilen wird der Link automatisch an der gewünschten Stelle eingefügt. Zuerst wird überprüft, ob eine Seite für den Datenschutz definiert wurde (siehe Listing 19.3, Zeile 2). Falls ja, so wird der Link zur Datenschutzerklärung mit `the_privacy_policy_link( ' | ', ' | ' )` eingefügt. Die beiden Parameter definieren, was vor und nach dem Link eingefügt werden soll, im Beispiel jeweils ein Separator zum vorherigen bzw. nachfolgenden Inhalt. Beachten Sie, dass es sich hier jeweils um einzelne Hochkommas handelt.

Listing 19.3 Code zum Einfügen des Links zur Datenschutzerklärung

```
1 <?php
2    if ( function_exists( 'the_privacy_policy_link' ) ) {
3       the_privacy_policy_link( '', ' | ' );
4    }
5 ?>
```

Änderungen in der Datei footer.php Die Änderungen im Footer werden nicht in der Originaldatei durchgeführt! Kopieren Sie die Datei `footer.php` aus dem Parent-Theme-Ordner in den Child-Theme-Ordner. In dieser Datei wird nun der Quellcode verändert. Nach dem Speichern der Datei (wenn Sie nicht auf XAMPP arbeiten, nach dem Hochladen der Datei in den Child-Theme-Ordner!) wird die Originaldatei im Parent-Theme beim Laden der Seite überschrieben.

1. Kopieren Sie die Datei `footer.php` aus dem Ordner `themes/twentynineteen` in den Child-Theme-Ordner `twentynineteen-child`.
2. Öffnen Sie die Datei `footer.php` und suchen Sie die Zeile mit `<div class="site-info">` (Zeile 20 in Bild 19.8). Darunter befinden sich die einzelnen Code-Blöcke für die Fußzeile und zwar:

   In Zeile 21 – 24 der *Name der Website* verlinkt mit der Startseite (1 in Bild 19.8) – *kann gelöscht werden*. Im Beispiel wird dies durch einen Copyright-Hinweis ersetzt.

   In Zeile 25 – 30 der Text *Stolz präsentiert von WordPress* mit einer Verlinkung zu WordPress.org und dem Theme Twenty Nineteen (2 in Bild 19.8) – *kann gelöscht werden*.

   In Zeile 31 – 35 der Link zur Seite mit der *Datenschutzerklärung*, falls diese Seite definiert und bereits veröffentlicht wurde (3 in Bild 19.8) – nur dann löschen, falls Sie die Datenschutzerklärung nicht im Footer verlinken möchten. Meine Empfehlung: *nicht löschen*.

   In Zeile 36 beginnt der Code für die *Menü-Position Footer* (4 in Bild 19.8) – *nicht löschen*!


```
17
18      <footer id="colophon" class="site-footer">
19          <?php get_template_part( 'template-parts/footer/footer', 'widgets' ); ?>
20          <div class="site-info">
21              <?php $blog_info = get_bloginfo( 'name' ); ?>
22              <?php if ( ! empty( $blog_info ) ) : ?>
23                  <a class="site-name" href="<?php echo esc_url( home_url( '/' ) ); ?>" rel="home"><?php
    bloginfo( 'name' ); ?></a>,
24              <?php endif; ?>
25              <a href="<?php echo esc_url( __( 'https://wordpress.org/', 'twentynineteen' ) ); ?>" class=
    "imprint">
26                  <?php
27                  /* translators: %s: WordPress. */
28                  printf( __( 'Proudly powered by %s.', 'twentynineteen' ), 'WordPress' );
29                  ?>
30              </a>
31              <?php
32              if ( function_exists( 'the_privacy_policy_link' ) ) {
33                  the_privacy_policy_link( '', '<span role="separator" aria-hidden="true"></span>' );
34              }
35              ?>
36              <?php if ( has_nav_menu( 'footer' ) ) : ?>
37                  <nav class="footer-navigation" aria-label="<?php esc_attr_e( 'Footer Menu',
    'twentynineteen' ); ?>">
38                      <?php
```

Bild 19.8 Website-Name (1), Link zu WordPress (2), Datenschutzerklärung (3) sowie Beginn der Menü-Position Footer (4) in der Datei footer.php

3. Da nur mehr eine benutzerdefinierte Zeile angezeigt werden soll, löschen Sie die Zeilen 21 bis 30, d. h. von `<?php $blog_info` in Zeile 21 bis inklusive `</a>` in Zeile 30. Übrig bleiben ein öffnendes `DIV` mit der Klasse `.site-info` in Zeile 20 und – mit einigen Leerzeilen dazwischen – der PHP-Block für die Datenschutzerklärung und darunter die Definition für die Menü-Position *Footer* (siehe Bild 19.9).

```
17
18      <footer id="colophon" class="site-footer">
19          <?php get_template_part( 'template-parts/footer/footer', 'widgets' ); ?>
20          <div class="site-info">
21
22
23              <?php
24              if ( function_exists( 'the_privacy_policy_link' ) ) {
25                  the_privacy_policy_link( '', '<span role="separator" aria-hidden="true"></span>' );
26              }
27              ?>
28              <?php if ( has_nav_menu( 'footer' ) ) : ?>
29                  <nav class="footer-navigation" aria-label="<?php esc_attr_e( 'Footer Menu',
```

Bild 19.9 Blog-Info und Link zu WordPress wurden gelöscht

4. Fügen Sie das Copyright-Symbol ©, Ihren *Namen* und die aktuelle *Jahreszahl* sowie *Alle Rechte vorbehalten* in die Zeile unterhalb des öffnenden DIV-Tags ein. Da das Copyright-Symbol ein Sonderzeichen ist, wird es maskiert und als Entity `&copy;` eingefügt. Die aktuelle Jahreszahl soll automatisch eingelesen werden, dafür verwenden Sie den PHP-Codeblock `<?php echo date('Y'); ?>`. Zudem ergänze ich noch die Jahreszahl, in der die Website (oder Firma) gestartet wurde. Meine Zeile lautet somit `&copy; J. Belik 2018-<?php echo date('Y'); ?> Alle Rechte vorbehalten.` (siehe Zeile 21 in Bild 19.10, oben). Das Ergebnis im Frontend zeigt Bild 19.10, unten.

Bild 19.10 Die neue Fußzeile im Child-Theme

19.2.1.2 Anmelde-Link im Footer

Ich möchte statt des Widgets *Meta* einen Anmelde-Link mit *Anmelden* in die Fußzeile geben. Wenn ich eingeloggt bin, soll *Abmelden* angezeigt werden. Ist man angemeldet, so wird im Widget *Meta* neben *Abmelden* auch ein Link *Website-Administration* angezeigt. Mit Klick auf diesen Link gelangt man direkt aufs Dashboard. Dies soll auch im Footer angezeigt werden.

1. Damit man sich über einen Link in der Fußzeile einloggen kann, wird `<?php wp_loginout(); ?>` in Zeile 27 direkt oberhalb des Code-Blocks für die *Menü-Position Footer* hinzugefügt (siehe Bild 19.11, oben). Damit erscheint im Frontend *Anmelden* bzw. *Abmelden* in der Fußzeile (siehe Bild 19.11, Mitte und unten).

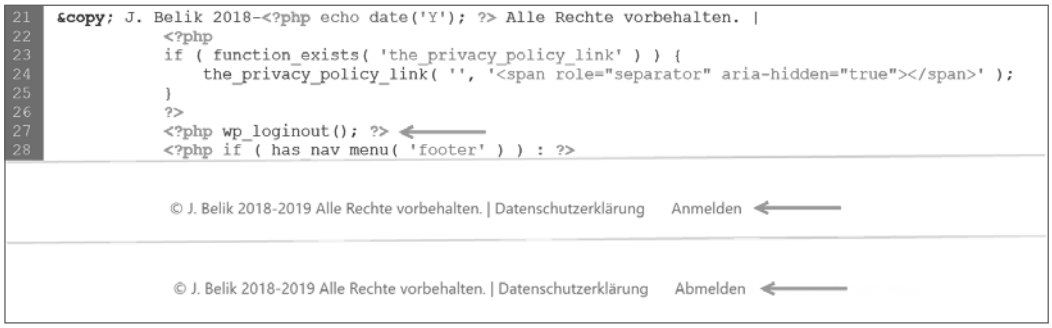

Bild 19.11 Anmelden und Abmelden im Footer

2. Zum einfacheren Administrieren wäre es praktischer, wenn nicht nur der Link *Abmelden*, sondern auch *Website-Administration*, wie sie im Widget *Meta* angezeigt wird, erscheint, sobald man angemeldet ist. Dazu wird `<?php wp_register(' | ','');?>` verwendet. Beachten Sie, dass die Schaltfläche *Registrieren* nur dann verfügbar ist, wenn die Option *Jeder darf sich registrieren* aktiviert wurde. Ist man eingeloggt, so erscheint der Direkt-Link *Website-Administration* zum Dashboard unabhängig davon, ob man sich registrieren darf oder nicht.

   Defaultmäßig ist der erste Parameter in der Klammer ein öffnender `<li>`-Tag, der zweite Parameter ein schließender `</li>`-Tag. Die Schaltfläche soll jedoch nicht als Listenelement (mit einem Listenpunkt davor) in der Fußzeile angezeigt werden. Außerdem soll sie zwecks besserer Optik durch einen senkrechten Strich | von Abmelden getrennt werden. Deshalb wird der erste Parameter als senkrechter Strich und ein Leerzeichen `' | '` davor und dahinter definiert und der zweite Parameter als leeres Element `''` angegeben. Beachten Sie, dass es sich jeweils um einfache Hochkommas handelt. Nach dem Speichern der Datei (und Hochladen per FTP, wenn Sie nicht auf XAMPP arbeiten)

kann man sich das Ergebnis im Frontend anschauen. Nach dem Anmelden erscheinen die Links *Abmelden* und *Website-Administration* (siehe Bild 19.12, unten).

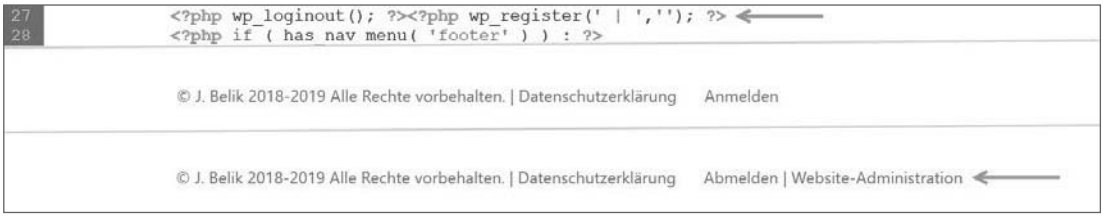

Bild 19.12 Ergänzter Code für Link Anmelden und Registrieren (oben), Anmelden (Mitte) sowie Abmelden und Website-Administration (unten)

19.2.1.3 RSS-Link im Footer

Wenn Sie häufig Beiträge veröffentlichen, ist es sinnvoll und üblich, einen *RSS-Link* zu Beiträgen in die Sidebar oder in die Fußzeile zu geben. Da das *Twenty Nineteen* Theme keine Sidebar hat, kommt der Link in den Footer. Die Adresse des Beiträge-Feeds hat die Form `www.meinedomain.xyz/feed/`. Sie könnten einen Link natürlich auch hardcoded mit `<a href="http://www.meinedomain.xyz/feed/" title="RSS Feed">Beitrags-Feed (RSS)</a>` einfügen. Falls Sie später von HTTP auf HTTPS umsteigen, haben Sie einen unsicheren Link auf Ihren Seiten und Sie müssen nachträglich den Link anpassen (siehe Kapitel 3). Daher ist es empfehlenswert, den Link zum Feed mit `<?php bloginfo('rss2_url'); ?>` automatisch ausgeben zu lassen. Somit lautet der Link `<a href="<?php bloginfo('rss2_url'); ?>" title="RSS Feed"> Beitrags-Feed (RSS)</a>`. Bild 19.13, oben, zeigt das Ergebnis im Frontend.

Bild 19.13 RSS-Beitrags-Link ohne (oben) und mit RSS-Icon (unten)

Ansprechender und professioneller wirkt der RSS-Link mit dem passenden Icon (siehe Bild 19.13, unten). Unter der Adresse *http://www.feedicons.com/* können Sie die offiziellen *RSS-Icons* in diversen Grafikformaten und Farben downloaden. Ich verwende das *Standard Icon Bundle* mit zwei PNG-Dateien in der Größe 28 × 28px und 14 × 14px, die beiden Dateien heißen `feed-icon-14x14.png` und `feed-icon-28x28.png`. Diese wurden in den `images`-Ordner im Child-Theme-Ordner entpackt.

1. Laden Sie die gewünschten RSS-Icons auf Ihren Rechner.
2. Falls noch nicht vorhanden, erstellen Sie einen Ordner mit dem Namen `images` in Ihrem Child-Theme-Ordner.
3. Entpacken Sie die RSS-Icons in den Ordner `images` im Child-Theme-Ordner.

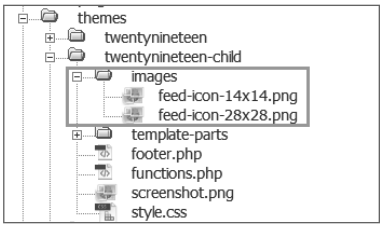

Bild 19.14
Die RSS-Icons im images-Ordner im Child-Theme-Ordner

4. Die Grafik soll vor dem Link angezeigt werden, dies wird mit einer Klasse in der Datei `style.css` im Child-Theme-Ordner definiert. Die Klasse bekommt im Beispiel den Namen `.rss`, die Grafik wird als Hintergrundgrafik festgelegt (siehe Zeile 5 in Listing 19.4). Öffnen Sie die Datei `style.css` im Child-Theme-Ordner und fügen Sie die CSS-Regel aus Listing 19.4 ein. Speichern Sie die Datei.

Listing 19.4 Klasse .rss in der style.css des Child-Themes

```
1 /* RSS-Icon */
2 .rss {
3     margin-left: 2em;
4     padding: 0 0 0 20px;
5     background: url("images/feed-icon-14x14.png") no-repeat 0 50%;
6 }
```

5. Wechseln Sie in die Datei `footer.php` in Ihrem Child-Theme-Ordner. Um den Link herum wird ein `<span>`-Tag mit der Klasse `.rss` gelegt. So lautet der Link mit dem RSS-Icon im Footer nun `<span class="rss"><a href="<?php bloginfo('rss2_url'); ?>" title="RSS Feed">RSS-Beiträge</a></span>`. Fügen Sie diesen Link nach dem PHP-Block mit `wp_register` ein (siehe Zeile 28 in Bild 19.15). Das Ergebnis im Frontend zeigt Bild 19.13, unten.

```
27              ?>
28              <?php wp_loginout(); ?><?php wp_register(' | ',''); ?> <span class="rss"><a
    href="<?php bloginfo('rss2_url'); ?>" title="RSS Feed">Beitrags-Feed (RSS)</a></span>
29
30              <?php if ( has_nav_menu( 'footer' ) ) : ?>
```

Bild 19.15 Fügen Sie den RSS-Link mit RSS-Icon in Zeile 28 ein

19.2.2 Site-Logo größer anzeigen

Das *Site-Logo* wird im Theme *Twenty Nineteen* links oben neben Titel und Navigationsleiste in einem runden Ausschnitt angezeigt. Bis zu einer Browserfenster-Breite von mindestens 768 Pixel hat es eine Größe von 50 × 50px, auf schmäleren Monitoren 64 × 64px. Dies ist meiner Meinung nach besonders auf großen Monitoren viel zu klein (siehe Bild 19.16, links). Deshalb möchte ich das Site-Logo doppelt so groß anzeigen lassen (siehe Bild 19.16, rechts).

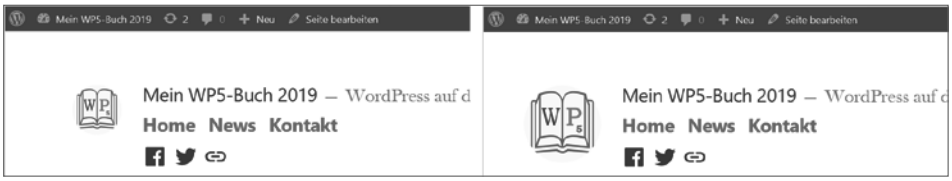

Bild 19.16 Ursprüngliche Größe des Site-Logos (links) und neue Größe (rechts)

Dazu wird eine bestehende CSS-Regel aus dem Parent-Theme im Child-Theme neu definiert und zwar die Klassen `.site-logo` und `.custom-logo-link`. Am einfachsten ist es, die bestehende Regel aus der Datei `style.css` im Parent-Theme-Ordner in die Datei `style.css` im Child-Theme-Ordner zu kopieren und dort wunschgemäß zu bearbeiten.

Mit Hilfe von *Media Queries* können beim Design von responsiven Layouts CSS-Regeln für unterschiedliche Ausgabemedien definiert werden. Im Beispiel soll die CSS-Regel für das größere Site-Icon nur auf dem Monitor und nur ab einer Mindestbreite von 768 Pixel gelten.

Listing 19.5 Neue Größe 100 × 100px für Site-Logo für Screens ab 768px Breite

```
1 /* Site-Logo 100x100px */
2 @media only screen and (min-width: 768px) {
3    .site-logo .custom-logo-link {
4       width: 100px;
5       height: 100px;
6    }
7 }
```

Da die „große“ Version des Site-Logos nur auf Screens ab mindestens 768px Breite angezeigt werden soll, wird ein *Media Query* in der Form `@media only screen and (min-width: 768px) { ... }` um die CSS-Regel herum gelegt (siehe Zeile 2 und 7 in Listing 19.5). Die CSS-Regel wird zwischen die geschwungenen Klammern { } gesetzt. Beachten Sie, dass es nun zwei geschwungenen Klammern } gibt, die erste von der CSS-Regel, die zweite von der Media Query. Fügen Sie die CSS-Regel samt Media Query aus Listing 19.5 in die `style.css` im Child-Theme-Ordner ein (siehe Bild 19.17).

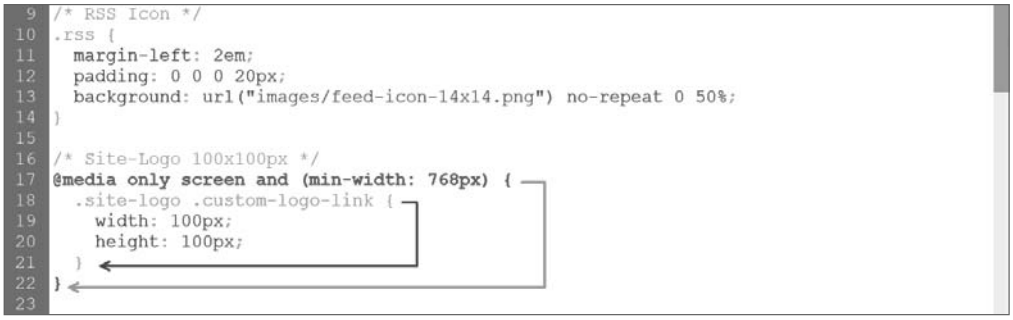

Bild 19.17 RSS-Icon und neue Größe für Site-Logo in der style.css