Tpl_css_generator.php - Dokumentation (Joomla Template)

Diese Datei erzeugt auf Basis der Template-Parameter dynamisch CSS-Variablen für das Joomla-Template **joomla-tpl-prosoft**. Sie verarbeitet Light- und Dark-Mode-Werte, ergänzt berechnete Layout- und Designvariablen, minimiert das resultierende CSS und liefert dieses zusammen mit einem Hash zu weiterer Verwendung in der *index.php*.

1. Dateien-Überblick

tpl_css_generator.php

2. Initialisierung und Grundprinzip

In diesem Abschnitt werden der Joomla-Kontext und die Template-Parameter initialisiert sowie eine einheitliche Namenskonvention für CSS-Variablen definiert. Zusätzlich wird festgelegt, welche Parameter automatisch verarbeitet werden und welche über eine Ausschlussliste in der Datei *variables_skip.php* von der CSS-Generierung ausgenommen.

2.1 Initialisierung und Kontext

Zu Beginn der Datei wird der Joomla-Kontext hergestellt und auf die Template-Parameter zugegriffen:

- $this->params liefert alle im Backend gesetzten Template-Optionen
- Über Factory::getApplication() wird der aktuelle Joomla-Anwendungskontext geladen.
- Der Seitenname (sitename) wird für die spätere Verwendung bei Bild-Alt-Texten vorbereitet

Diese Initialisierung stellt sicher, dass alle folgenden Berechnungen auf den aktuellen Template-Einstellungen basieren.

2.2 Namenskonvention für CSS-Variablen

Die anonyme Funktion makeCSSName() dient der automatischen Umwandlung von Template-Parametern in gültige CSS-Variablennamen:

- CamelCase-Parameter werden in **kebab-case** überführt
- Jeder Variablenname wird mit dem Präfix **--tpl-** versehen (siteWidth → --tpl-site-width)

2.3 Ausschluss bestimmter Parameter

Über die Datei *variables_skip.php* wird eine Liste von Parametern eingebunden, die **nicht automatisch** in CSS-Variablen umgewandelt werden sollen.

Diese Auslagerung ermöglicht:

- Saubere Trennung zwischen automatischer und manueller CSS-Logik
- Bessere Wartbarkeit bei komplexen oder logisch abgeleiteten Parametern

2.4 Automatische Verarbeitung der Template-Parameter

Alle Template-Parameter werden iteriert und geprüft:

- Arrays, leere Werte und ausgeschlossene Parameter werden übersprungen
- Werte-Bereinigung (Zeilenumbrüche und CSS-Sonderzeichen entfernt)
- Zuordnung von Parametern mit dem Präfix *darkMode_* zum Dark-Mode
- Zuordnung übriger Parameter zum Light-Mode

Das Ergebnis sind zwei getrennte Variablenlisten:

- $lightVars für :root
- $darkVars für :root.dark-mode

3. Thematische Bereiche und berechnete Variablen

Nach der automatischen Verarbeitung der Template-Parameter werden in dieser Datei mehrere **thematische Abschnitte** definiert. In diesem Abschnitt werden **abhängige, zusammengesetzte oder logisch abgeleitete Werte** berechnet, die nicht direkt aus einem einzelnen Template-Parameter bestehen.

Diese Werte werden später als CSS Custom Properties ausgegeben.

3.1 Bereich „General"

Der Abschnitt *General* enthält grundlegende Layout- und Designwerte des Templates:

- Seitentitel und Fallback-Logik für Logos
- Logogrößen (links / rechts)
- Hintergrundbild oder Hintergrundfarbe des Body
- Globale Abstände (Padding, Margin)
- Schriftgrößen für Überschriften und Texte

- Border Radius
- Seitenbreite und vertikale Abstände
- Box-Shadows für:
    - Site-Container
    - Content-Bereiche
    - Header
    - Component-Positionen

Zusätzlich werden hier Zustände logisch ausgewertet und in konkrete CSS-Werte übersetzt:

- Sticky-Header
- Header-Animation
- Sidebar-Top-Abstand

3.2 Bereich „Content / Hero / Welcome"

Dieser Abschnitt steuert die visuelle Darstellung des Inhaltsbereichs:

- Hero-Bild
- Hero-Höhe und Fokuspunkt
- Welcome-Modul:
    - Position
    - Transparenz
    - Maximale Breite
- Breite des Haupt-Components

3.3 Bereich „Social Bar"

Der Abschnitt *Social Bar* wird sowohl die **Darstellung** als auch der **Inhalt** der Social-Media-Leiste vorbereitet:

- Positionierung (links / rechts)
- Offscreen-Positionen für Animationen
- Border-Radius abhängig von der Seitenposition
- Aufbau eines strukturierten Arrays ($socialItems) mit:
    - Typ (Instagram, Facebook, LinkedIn, YouTube, X, Mail)
    - Ziel-URL

3.4 Bereich „Menu"

Hier werden alle menübezogenen Layout-Werte berechnet:

- Hintergrundbild oder Hintergrundfarbe
- Breite des Menübereichs
- Schriftgröße der Menüs
- Hover-Farben
- Unterstreichungslogik inklusive Offset

3.5 Bereich „Sidebar"

Der Bereich *Sidebar* definiert:

- Relative Breite der Sidebars
- Sticky-Verhalten der Sidebars

3.6 Bereich „Scrollbar"

Hier werden Browser-spezifische Scrollbar-Werte vorbereitet:

- Scrollbar-Breite
- Scrollbar-Border-Radius

3.7 Bereich „Footer"

Der Footer-Abschnitt steuert:

- Hintergrundbild oder Hintergrundfarbe
- Gesamtbreite des Footers
- Spaltengewichtung der Footer-Module (A-D) → prozentuale Gewichtung und ermöglicht flexible Footer-Layouts

3.8 Bereich „Font Scheme"

Dieser Abschnitt behandelt die Schriftkonfiguration:

- Umschaltung zwischen System-Schriften und benutzerdefinierten Schriftdateien
- Erzeugung eines *@font-face*-Blocks bei Dateischriften
- Zuweisung der Schriftfamilien an:
    - Title

- Menü
- Welcome-Bereich
- Content (Haupt- und Unterüberschriften)
- Footer

3.9 Bereich „Popup"

Hier werden alle popupbezogenen Werte vorbereitet:

- Inhalt (Heading, Text, Button)
- Bilddatei
- Breite
- Border-Radius

3.10 Bereich „Scroll-to-Top & Mobile"

Abschließend werden Werte für zusätzliche Features gesetzt:

- Auswahl des Scroll-to-Top-Icons (Unicode-Symbole)
- Mobile-Navigation:
  - Title oder Logo
  - Abstände
  - Schriftgrößen für mobile Überschriften und Texte

4. CSS-Ausgabe, Minifizierung, Hash und Rückgabe

In diesem Abschnitt wird aus den zuvor ermittelten Variablen das finale CSS erzeugt. Die Light- und Dark-Mode-Variablen werden in *:root* bzw. *:root.dark-mode* ausgegeben, anschließend minifiziert, mit einem Hash versehen und zur weiteren Verarbeitung an die *index.php* zurückgegeben.

4.1 CSS-Aufbau via Output Buffer

Ab *ob_start()* wird der CSS-Text aufgebaut, ohne ihn direkt auszugeben. So kann der komplette CSS-Block am Ende als String weiterverarbeitet werden.

4.2 Ausgabe der Light-Mode Variablen in :root

Im Block *:root { … }* werden zwei Quellen zusammengeführt:

1. **Automatisch generierte Variablen** aus *$lightVars* (aus den Template-Parametern bereinigt und in *--tpl-** umgewandelt)
2. **Berechnete Variablen**, die aus mehreren Parametern bzw. Logik entstehen (z.B. *--tpl-body-file*, *--tpl-site-width*, *--tpl-header-box-shadow*, Logo-Größen, Footer-Gewichte, usw.)

4.3 Ausgabe der Dark-Mode Variablen in :root.dark-mode

Im Block *:root.dark-mode* werden ausschließlich die Variablen aus *$darkVars* ausgegeben. Diese stammen aus Parametern mit dem Präfix *darkMode_* und überschreiben im Dark Mode gezielt die entsprechenden Light-Mode-Werte.

4.4 Minifizierung der erzeugten CSS-Ausgabe

Die Funktion *$minifyingCSS()* reduziert die CSS-Ausgabe:

- Block-Kommentare und nnötige Whitespaces um Trennzeichen entfernt
- Mehrfach-Spaces reduziert
- Überflüssige Semikolons vor } entfernt

4.5 Hash-Bildung für Cache-/Update-Logik

Nach der Minifizierung wird ein SHA-256 Hash über den finalen CSS-String gebildet:

$hash = hash(‚sha256‘, $css);

Dieser Hash wird in der *index.php* genutzt, um festzustellen, ob *generated.css* neu geschrieben werden muss.

4.6 Return-Format

Am Ende liefert die Datei ein Array zurück:

- *css* enthält den finalen (minifizierten) CSS-String
- *hash* enthält den zugehörigen SHA-256 Hash

Damit kann die *index.php* den CSS-Inhalt schreiben und die Versionierung / Cache-Validierung steuern.