T3SBOOTSTRAP · RELEASE-NOTES

Update auf Version 5.3.51

Der Editor kann jetzt Spalten, Abstände, Badges und farbige Code-Blöcke – und du bekommst 18 Einstellungen, um genau das wieder loszuwerden. Dazu das Brand-Logo endlich an einer Stelle statt an zweien, vier Abstürze weniger im Frontend, eine geschlossene XSS-Lücke und CLI-Kommandos, die dir bei einem Netzwerkfehler nichts mehr wegräumen.

VERSION	TYPO3	PHP
5.3.51	^14.3	8.2 – 8.5

i Auf einen Blick

Neu im RTE	Spalten, ein Dropdown für die Abstände `mt-1` … `mb-5`, ein Dropdown für Badges und Code-Blöcke mit 12 Sprachen und Syntaxfärbung.
18 Einstellungen	Jede RTE-Funktion einzeln abschaltbar, pro Seitenbaum übersteuerbar.
Zweites Preset	`t3sbootstrap_extended` lässt CSS-Klassen auf Textelementen durch.
Brand-Logo	Maße und Alt-Text stehen jetzt im Konfigurations-Datensatz statt nur in den Site-Settings. **Braucht eine Datenbankanalyse.**
Speaking ID	Neue Option, die das Feld „Anchor“ im Backend einblendet. Am Rendern ändert sich nichts.
icon-link	Im Link-Wizard wählbar – und die Hover-Animation läuft endlich auch mit Iconpack.
Behoben	Vier Abstürze im Frontend, ein Seitenverhältnis auf dem Kopf, die Navbar mit Plus-Icon, eine XSS-Lücke im Seitenmodul.
Achtung	Die Lightbox zeigt ab jetzt den Bildzuschnitt – bisher öffnete sie immer das volle Bild.

1 Spalten, Abstände, Badges und Code im RTE

Vier neue Buttons in der Toolbar:

- **Columns** – zwei, drei oder vier Bootstrap-Spalten. Raus kommt `<div class="row">` mit `col-md-*`, die Spalten stapeln sich also unterhalb von md. Genau wie überall sonst in der Extension.
- **Margin** – ein Dropdown für `mt-1` bis `mt-5` und `mb-1` bis `mb-5` auf dem aktuellen Block. Oben und unten kannst du getrennt wählen.
- **Badge** – die acht Bootstrap-Farben, jede als normales Badge und als Pill, alles in einem Dropdown. Diese 16 Einträge standen bisher in der Styles-Auswahl; die schrumpft dadurch von 57 auf 41 Einträge.
- **Code-Block** – `<pre><code>` mit Sprachauswahl: Plain text, TypoScript, PHP, YAML, JSON, XML, HTML, CSS, SCSS, JavaScript, Bash und SQL. Davon getrennt macht **Code** ein einzelnes Wort im Satz zu Inline-`<code>`.

Und ein kleines Ärgernis ist weg: Alert- und Spalten-Blöcke waren bisher eine Sackgasse. Steht der Cursor jetzt in der letzten leeren Zeile, kommst du mit **Enter** wieder raus – in der ersten leeren Zeile bringt dich **Backspace** nach oben.

Farbe in den Code-Block

Der Editor schreibt nur `<pre><code class="language-php">` – darin steht ein einziger Textknoten, den kein CSS einfärben kann. Die Token-Ebene entsteht erst im Frontend, und dafür ist **Prism.js** zuständig. Beides steuert eine einzige Einstellung:

```
// Extension-Konfiguration > RTE > Code block
0               // kein Code-Block
1               // Code-Block, aber ohne Färbung
Default | Dark | Coy | Funky | Okaidia | SolarizedLight | TomorrowNight | Twilight
```

Sobald ein Theme gewählt ist – und nur dann – werden `Resources/Public/Prism/<Theme>.css` und die gemeinsame `prism.js` geladen. Die alte Option `advanced.prism` entfällt damit. Die Bundles liegen ab jetzt in der Extension – vorher zeigte die Einstellung auf Verzeichnisse, die niemand hatte.

Die Engine liegt bewusst nur einmal da statt achtmal: sie hängt an der Sprachauswahl, nicht am Theme, und war in allen acht Paketen des Downloaders byte-gleich. Wie sich die Bundles neu bauen lassen, steht in `Resources/Public/Prism/README.md`.

> Der Bundle-Baukasten auf prismjs.com liefert per Vorgabe nur vier Sprachen. Wer sein Bundle selbst zusammenstellt, muss alle zwölf ankreuzen – und `php` braucht zusätzlich `markup-templating`, sonst bleibt PHP ungefärbt.

2 18 Einstellungen für den RTE

Nicht jede Redaktion braucht jeden Button. Du kannst jetzt jede Funktion einzeln abschalten, ohne dafür das YAML-Preset zu forken – 17 Schalter und die Theme-Auswahl für den Code-Block. Die Vorgabe machst du in der Extension-Konfiguration unter **RTE**; wenn du es pro Seitenbaum anders willst, gewinnt das Page-TSconfig:

```
// Page-TSconfig
RTE.t3sbootstrap.features {
    columns   = 0
    badge     = 0
    codeBlock = 0
}
```

> ⚠ **AUSBLENDEN HEISST NICHT WEGWERFEN**
>
> Ein abgeschalteter Button verschwindet nur aus der Toolbar. Das CKEditor-Plugin bleibt geladen, damit deine bestehenden Alerts, Spalten und Code-Blöcke weiter geparst, bearbeitet und unverändert gespeichert werden. Würde der Schalter stattdessen das Modul rauswerfen, kennte CKEditor das Element nicht mehr – und General HTML Support würde es beim nächsten Speichern klammheimlich entsorgen.

Eine umbenannte Einstellung

Weil die Badges aus der Styles-Auswahl in einen eigenen Toolbar-Eintrag gezogen sind, heißt `rteStyleBadges` jetzt `rteBadge` (im TSconfig: `styleBadges` → `badge`). Hast du den alten Schlüssel gesetzt, **entscheidet er weiter** – und zwar mit Absicht: TYPO3 trägt beim Update jeden neuen Schlüssel mit seinem Vorgabewert in die gespeicherte Konfiguration ein. Nach dem Update stünde dort also `rteBadge = 1`, ohne dass es jemand entschieden hätte. Gewönne der neue Schlüssel, wäre deine Entscheidung still überschrieben.

Aufgeräumt wird das vom Upgrade-Wizard *„Rename the RTE setting rteStyleBadges to rteBadge“* – oder von selbst, sobald du das Formular der Extension-Konfiguration einmal speicherst.

Und wenn du Klassen durchreichen willst

Dafür gibt es ein zweites Preset: `t3sbootstrap_extended`. Gleicher Editor, eine Ergänzung – die GHS-Whitelist behält CSS-Klassen auf den üblichen Textelementen, statt sie auf das einzudampfen, was `style.definitions` kennt. Bewusst optional, denn dieselbe Whitelist ist auch dein Paste-Filter. Einschalten so:

```
RTE.default.preset = t3sbootstrap_extended
```

3 Brand-Logo an einer Stelle statt an zweien

Der Pfad zum Navbar-Logo konnte im Konfigurations-Datensatz stehen (`navbar_image`) oder in den Site-Settings (`bootstrap.navbar.image.defaultPath`). Breite, Höhe und Alt-Text gab es dagegen *nur* in den Site-Settings. Diese Asymmetrie hatte eine unangenehme Folge: Wer im Datensatz ein abweichendes Logo eintrug, bekam trotzdem die Maße und den Alt-Text des Site-Logos verpasst.

Der Tab **Brand (Logo)** hat deshalb drei neue Felder – Bildbreite, Bildhöhe und Alternativtext. Für alle vier Werte gilt jetzt dieselbe Rangfolge wie zuvor schon für den Pfad:

Datensatz gefüllt	Dieser Wert gilt. Das Site-Setting ist wirkungslos.
Datensatz leer bzw. 0	Es greift das Site-Setting `bootstrap.navbar.image.*` .

Die vier Site-Settings bleiben also bestehen – sie heißen im Settings-Editor nur noch `DEPRECATED (now in T3SB Config) – …` . Sie zu entfernen würde jede Installation, die dort einen Logo-Pfad stehen hat, beim Update auf das Bootstrap-Logo zurückwerfen. **An deiner Ausgabe ändert sich durch das Update nichts**, solange du nichts tust.

Willst du aufräumen, holt der Upgrade-Wizard *„Move the navbar brand image settings into the configuration record“* alle vier Werte je Siteroot in den Datensatz – nur dort, wo das Feld noch leer ist.

> ⚠ **DIESMAL MIT DATENBANKANALYSE**
>
> Anders als bei den bisherigen Versionen bringt dieses Update **drei neue Spalten** in `tx_t3sbootstrap_domain_model_config` mit: `navbar_image_width` , `navbar_image_height` und `navbar_image_alt` . Ohne die Analyse lässt sich der Konfigurations-Datensatz nicht mehr speichern.

4 Speaking ID und icon-link

Speaking ID

Das Feld `tx_t3sbootstrap_anchor` gibt es schon lange, es stand nur immer im Formular. Die neue Option blendet es ein oder aus – mehr nicht. Am Frontend ändert sie nichts: der Anchor-`<span>` entsteht weiterhin für jedes Element, das im Section-Index steht und einen Wert im Feld hat. Das muss so sein, denn die Section-Menü-Links der Navbar zeigen genau auf diese IDs – hinge das Rendern an einer Option, die auf *aus* steht, wäre auf jedem bestehenden One-Page-Layout die Navigation tot.

Dazu kam ein Layout-Problem: in einer `.row` hat `row-cols-*` den Span als Spalte mitgezählt, und alles verrutschte um eins. Er liegt jetzt `position: absolute` , belegt damit keinen Grid-Slot mehr und bleibt trotzdem genau da, wohin der Anker springen soll.

icon-link

`icon-link` und `icon-link icon-link-hover` kannst du jetzt im Link-Wizard auswählen. Bootstrap hängt den Hover-Effekt allerdings an seine eigene Icon-Schrift (`.icon-link > .bi`) – und Iconpack liefert `<i>` , `<svg>` , `<span>` oder `<img>` . Deshalb gelten dieselben Regeln jetzt auch für diese Elemente, `prefers-reduced-motion` inklusive.

Kleinkram nebenbei: die Extension-Konfiguration hat endlich lesbare Tab-Beschriftungen statt kleingeschriebener Kategorienamen, und `flexformDir` sitzt jetzt unter „FlexForm“ statt unter „Basic“.

5 Was im Frontend nicht mehr abstürzt

WO ES KNALLTE	WARUM – UND WAS JETZT PASSIERT
Background-Wrapper mit lokalem Video Fatal Error	Das ganze `localvideo`-Sheet hängt an `displayCond → isLocalVideo`, und die Bedingung schaut auf die schon gespeicherte assets-Relation. Beim ersten Speichern wurden die Felder deshalb nie angezeigt und nie geschrieben – und im Frontend gab es `trim(null)`. Alle Feldzugriffe sind jetzt mit den FlexForm-Vorgaben abgesichert.
Galerie aus Dateisammlung oder Ordner Fatal Error	Die Bildbreite wurde nur innerhalb von `instanceof FileReference` gesetzt – Dateisammlungen und Ordner liefern aber `File`. Geprüft wird jetzt auf `FileInterface`.
Masonry-Wrapper ohne FlexForm Fatal Error	`strpos(null, …)` beim Lesen von `colclass` – trifft dich bei Datensätzen aus einem Import oder einer Kopie, die nie gespeichert wurden.
Card-Wrapper und Modal ohne FlexForm Fatal Error	Ein Container, der im FlexForm nie geöffnet wurde, hat `NULL` gespeichert – die Konfiguration kommt als leeres Array an. Unter PHP 8 wird jeder Zugriff auf einen fehlenden Schlüssel zur Warnung, und der TYPO3-ErrorHandler macht daraus eine Exception, die das ganze Frontend mitnimmt. Alle Zugriffe in `Classes/` sind jetzt abgesichert; gefunden wurden sie mit zwei Token-Scannern, die den Code auf ungeprüfte Zugriffe durchsuchen.
Seitenverhältnis stand kopf Falsche Darstellung	Der Value Picker speichert die Höhe zuerst („37by9“ liegt als `9/37` in der Datenbank), gerechnet wurde aber Breite durch Höhe. Aus 24,3 % wurden 411 % – bei 1300 px Breite also eine 5300 px hohe Section. Alle Schreibweisen laufen jetzt durch dieselbe Normalisierung wie im übrigen Code.
Navbar mit Plus-Icon Falsche Darstellung	Der Plus-Button rutschte unter den Menüpunkt, blieb ungestylt und trug zusätzlich den Bootstrap-Caret. Die Medienabfrage im Partial suchte den Breakpoint unter seinem *Pixelwert* statt unter seinem Schlüssel – `settings.navbar.992` statt `settings.navbar.lg`. Heraus kam `@media (min-width: px)`, eine ungültige At-Regel, die der Browser samt ihrem *gesamten Inhalt* verwirft. Damit fiel die komplette Desktop-Darstellung des Plus-Icons weg.
Lightbox ignorierte den Zuschnitt Falsche Darstellung	In der Popup-Konfiguration stand `crop =` – eine gesetzte, aber leere Eigenschaft. TYPO3 prüft mit `isset()`, ob TypoScript den Zuschnitt vorgibt, nicht ob dort etwas drinsteht. Das leere `crop =` hat also gereicht, um den Zuschnitt der Dateireferenz zu überspringen. Jetzt steht dort `crop.data = file:current:crop`. **Siehe den Kasten dazu.**
Lokales Video im Background-Wrapper Falsche Darstellung	Auf dem iPhone ging das Video beim Öffnen der Seite sofort in den Vollbildmodus – TYPO3 schreibt `autoplay`, aber nie `playsinline`. Dazu blieb die Option „Höhe (mobil)“ ohne Wirkung, und über dem Video stand ein grauer Streifen.

> ⚠ **GEÄNDERTES VERHALTEN: LIGHTBOX UND BILDZUSCHNITT**
>
> Die Lightbox hat bisher *immer* das ungeschnittene Bild geöffnet – auch dann, wenn im Bildeditor ein Ausschnitt festgelegt war. Ab jetzt zeigt sie denselben Ausschnitt wie das Bild auf der Seite. Willst du weiterhin das volle Bild, aktivierst du das je Element mit **„Originalbild für Lightbox verwenden“** unter *Erscheinungsbild → Verhalten*. Genau dafür war die Option gedacht – sie war nur ohne Wirkung, weil der Zuschnitt ohnehin für alle übersprungen wurde.

6 Zwei Sicherheitslücken

XSS im Seitenmodul

Der Info-ViewHelper gibt seine Ausgabe mit `$escapeOutput = false` und `f:format.raw()` aus – „Extra Class“ und „Header Extra Class“ sind aber freie Eingabefelder. Wer auf `tt_content` schreiben darf, konnte dort Markup hinterlegen, das im Seitenmodul in der Session jedes Admins lief. Alles, was aus einem Datensatz kommt, geht jetzt durch `htmlspecialchars()`.

Export und Import ohne Rechteprüfung

Das Modul läuft mit `access => user`, die Seiten-ID kam ungeprüft aus der URL, und der Export hat alle Restrictions abgeräumt. Damit ließ sich die Konfiguration fremder Sites samt `custom_scss` herunterladen. Jetzt entscheidet `doesUserHaveAccess()` – `PAGE_SHOW` zum Exportieren, `PAGE_EDIT` zum Importieren. Ein Export ohne Seiten-ID bleibt Admins vorbehalten, und die Auswahlliste zeigt nur noch Root-Seiten, die der Benutzer auch sehen darf.

> ⚠ **KURZ GESAGT**
>
> Wenn bei dir Redakteure am Seitenmodul oder am T3SB-Config-Modul sitzen, solltest du dieses Update nicht liegen lassen. Reine Solo-Installationen können sich Zeit lassen.

7 CLI-Kommandos, die nichts mehr wegräumen

Beide Kommandos haben ihre Zieldateien gelöscht, *bevor* klar war, ob der Download überhaupt klappt. Ein 404 oder ein kurz nicht erreichbarer Server – und die Dateien waren weg. Gemeldet hat das Kommando trotzdem Erfolg.

KOMMANDO	WAS SICH ÄNDERT
`cdnToLocal`	Erst laden, dann über eine temporäre Datei atomar tauschen. Geht ein Download schief, bleibt deine vorhandene Datei liegen, du erfährst welche – und am Ende kommt `Command::FAILURE`, im Scheduler also sichtbar statt grün. Fehlt `ext-zip`, werden deine lokalen Google Fonts nicht mehr gelöscht.
`cdnToLocal` Google Fonts	Ein Gewicht, das es für eine Familie gar nicht gibt – Metrophobic etwa gibt es nur als `regular` – hat bisher den kompletten Font-Lauf abgeräumt und das Verzeichnis leer zurückgelassen. Fehlende Gewichte werden jetzt übersprungen und gemeldet, in `googlefonts.css` landet nur noch, was es wirklich gibt. Und die Zuordnung stimmt: bei mehreren Fonts bekam der zweite bisher die Dateien des ersten.

KOMMANDO	WAS SICH ÄNDERT
`customScss`	Die Bootstrap-Quellen landen erst in einem Staging-Verzeichnis und werden nur bei vollem Erfolg getauscht. Eine vertippte Bootstrap-Version im Site Set nimmt dir damit nicht mehr das Frontend mit einem fehlenden `@import` mit. Dazu Null-Prüfungen auf den Konfigurations-Datensatz und den Bootswatch-Download.
Iconpack-Wizards	Erkannten sie keinen Icon-Namen, schrieben sie ein nacktes `fa7:` und leerten das Quellfeld – dein Wert war weg. Gesucht wird jetzt der erste `fa-*`-Token, der kein Style, Modifier oder Größenkürzel ist; ohne Treffer bleibt der Datensatz, wie er ist. Ein bereits gesetztes Icon wird nicht mehr überschrieben.

8 Checkliste

- **Pflicht**

- [ ] **Datenbankanalyse ausführen** – drei neue Spalten in `tx_t3sbootstrap_domain_model_config`. Install-Tool → *Analyze Database Structure* oder `vendor/bin/typo3 extension:setup`. Ohne diesen Schritt lässt sich der Konfigurations-Datensatz nicht speichern
- [ ] **Caches leeren** – es gibt neue Event-Listener und geänderte Signaturen, der kompilierte DI-Container muss neu gebaut werden

- **Nachsehen**

- [ ] **Extension-Konfiguration, Tab *RTE*** – dort liegen die 18 Einstellungen. Willst du farbige Code-Blöcke, wähle bei *Code block* ein Prism-Theme; die Vorgabe ist „kein Code-Block“
- [ ] **Speaking ID** – neu und auf aus. Bestehende Anker funktionieren weiter; einschalten musst du sie nur, wenn Redakteure das Feld sehen sollen
- [ ] **Lightbox mit zugeschnittenen Bildern** – sie zeigt jetzt den Zuschnitt. Willst du das volle Bild, setz je Element „Originalbild für Lightbox verwenden“

- **Wenn du magst**

- [ ] Die beiden **Upgrade-Wizards** laufen lassen – sie holen die Logo-Werte aus den Site-Settings in den Datensatz und benennen `rteStyleBadges` um. Beide sind Bequemlichkeit, keine Voraussetzung
- [ ] Brauchst du durchgereichte CSS-Klassen im RTE, setz `RTE.default.preset = t3sbootstrap_extended`
- [ ] `t3sbootstrap:cdnToLocal` einmal ausführen

> ✓ **FERTIG!**
>
> Wenn das Frontend sauber rendert und im Editor die neuen Buttons auftauchen, bist du durch. An deinen Inhalten ändert sich nichts: die neuen Logo-Felder sind leer und fallen auf die Site-Settings zurück, Badges behalten ihr Markup, und die Upgrade-Wizards räumen nur auf. Einzige sichtbare Änderung ohne dein Zutun ist der Bildzuschnitt in der Lightbox.