Das 'sopra-changelog'-Paket

Dokumentation für das 'sopra-changelog'-Paket | Version v1.0.02

2. Dezember 2019

Florian Sihler (florian.sihler@uni-ulm.de)

1 Allgemeines

1.1 Warum, wieso, weshalb?

Dieses LaTeX 2_ε-Paket wurde im Rahmen des Sopras im Wintersemester 2019 und Sommersemester 2020 verfasst und dient zur automatischen generierung von Changelogs für das *Teams 20*. Diese Dokumentation wurde zusammen mit der `sopra-base.cls` sowie dem Paket `sopra-documentation.sty` kreiert.

Zum Visualisieren der einzelnen Code-Ausschnitte wird das `sopra-listings`-Paket verwendet. Das zugehörige Paket sollte ebenfalls in dieses Dokument eingebettet sein:  . Date Paket wurde binnen drei Stunden entwickelt und programmiert, es befindet sich aktuell in der Entwicklung.

1.2 Abhängigkeiten

Dieses Paket bindet die folgenden Paketen mit ein:

- `environ`
- `fontawesome`
- `marginfix`
- `hyperref`

All diese Pakete sollten Teil der gängigen LaTeX-Distribution sein.

1.3 Die Installation

Das Paket wird nicht als `.dtx` ausgeliefert, weswegen sich die folgenden Möglichkeiten ergeben:

- Das Paket kann in dasselbe Verzeichnis wie das Dokument gesetzt werden. In diesem Fall lautet die Einbindungsanweisung:

  ```
  \usepackage{sopra-changelog}
  ```

- Das Paket kann in ein Unterverzeichnis/in ein mit dem Dokument ausgeliefertes Verzeichnis gelegt werden. In diesem Fall erfolgt die Angabe durch den (relativen-) Pfad:

  ```
  \usepackage{./Mein/Pfad/zu/sopra-changelog}
  ```

- Man kann das Paket (mittels eines Symlinks oder ähnlichem) in einen eigenen *texmf*-Baum ablegen. So kann zum Beispiel auf Linux unter der Verwendung von texlive das Paket hier abgelegt werden: `~/texmf/tex/latex/`. Das Verzeichnis kann erstellt und anschließend mittels `texhash ~/texmf` aktualisiert werden. Nun kann das Paket wie jede andere installierte Paket verwendet werden:

  ```
  \usepackage{sopra-changelog}
  ```

1.4 Weitere Besonderheiten

In Version v1.0.02 (`\thesocversion` → p. 3) gibt es keine weiteren Besonderheiten.

2 Paket-Konfiguration

2.1 Akzeptierte Parameter

In Version v1.0.02 werden keine Parameter akzeptiert.

3 Befehle- und Umgebungen

Es gilt zu beachten, dass das Präfix env@ nur auf die Natur einer Umgebung hinweist und nicht zum eigentlichen Bezeichner zuzuordnen ist!

Weiter gilt: Damit die Änderungen richtig aufgelöst werden können, muss das Dokument in der Regel zwei mal kompiliert werden um eine korrekte Anzeige zu erzeugen.

3.1 Versionsänderungen angeben

▷ `env@version`*[changes]*`{Introduced Version}{Title}`

Die Hauptumgebung des Paketes und wird für die Versionierung verwendet. Ein wichtiger Hinweis vorne weg: *Der Inhalt der Umgebung wird nur dann gesetzt, wenn `\setTargetVersion` → p. 3 mindestens auf `Introduced Version` gesetzt ist! Sonst gibt es die Änderung ja noch nicht.*

Das Feld `Introduced Version` gibt an mit welcher Versionsnummer (von 1 an gezählt) die in der Umgebung eingeschlossenen Komponenten zur Verfügung stehen. Identifiziert werden diese durch den *eindeutigen* Titel in `Title`.

Das wichtigste Feld ist allerdings changes. Hierbei kann eine kommaseparierte Liste an Feldern übergeben die entweder der Syntax „`changed in: ⟨VerID⟩ (⟨Description⟩)`“ oder der Syntax „`removed in: ⟨VerID⟩ (⟨Description⟩)`“ gehorchen müssen. Ein Beispiel:

```
\begin{version}[%
   changed in: 2 (Added Some funny stuff),%
   changed in: 3 (Removed stupid sentences),%
   removed in: 4 (Rendered to be stupid!)
]{1}{Koordinaten des Spielfeldes}
   Hier kann nun alles stehen was man sich so vorstellen kann. Auch normaler \LaTeX-Code!
\end{version}
```

Dies erzeugt (Hinweis: Dieses Dokument wurde mit `\setTargetVersion{5}` kompiliert):
Hier kann nun alles stehen was man sich so vorstellen kann. Auch normaler LaTeX-Code!

V. 1-4

▷ `\simpleversion`*[changes]*`{Introduced Version}{Title}{Content}`

Einfach nur ein kleiner Wrapper, der `env@version` → p. 2 mit `Content` aufruft.

▷ `\setVersionData{Version ID}{Version Date}{Version Description}`

Speichert für `\thechangelog` → p. 3 Metadaten zur Version mit Versionsnummer `Version ID`. Das `Version-Date` ist in `YYYY-MM-DD` anzugeben!

3.2 Versionänderungen setzen

▷ `\thechangelog`

Setzt den Verlauf über alle Änderungen. Für die Gliederung wird `\section` und `\subsection` verwendet. Ein Beispiel findet sich am Ende des Dokuments.

3.3 Weitere Daten zuweisen

▷ `\setTargetVersion{VersionID}`

Sorgt dafür, dass alle Änderungen bis zur Version `VersionID` ausgegeben werden (bezüglich des Changelogs). Dieser Wert sollte mit jedem Änderungsdurchlauf inkrementiert werden. *Bisher ist keine automatisierte Lösung vorgehesehen!*

▷ `\setTargetVersionDate{Date}`

Bisher ohne Bedeutung!

3.4 Hilfreiche Befehle

▷ `\thesocversion`

Gibt die aktuelle Version der sopra–paper-Dokumentklasse aus. Hinweis: über `\value{socversion}` lässt sich die Version als 4-stellige Nummer erhalten: 1002.

4 Changelog

4.1 Changelog für Version 1

- Für diese Version liegen keine Informationen vor.
- *Koordinaten des Spielfeldes*: Neues Element!

4.2 Changelog für Version 2

- Für diese Version liegen keine Informationen vor.
  - *Koordinaten des Spielfeldes*: Added Some funny stuff

4.3 Changelog für Version 3

- Für diese Version liegen keine Informationen vor.
  - *Koordinaten des Spielfeldes*: Removed stupid sentences

4.4 Changelog für Version 4 (2. Dezember 2019)

- Hier war alles voll supi und so. Diese Version fügt Affen hinzu!
- *Koordinaten des Spielfeldes*: Rendered to be stupid!