DEINTEMPLATE? Was ist das Override System?

Der Aufwand für ein späteres Update reduziert sich ganz enorm, wenn man von Anfang an, das Override System nutzt.
Also nie irgendwelche Coredateien bei Änderungen einfach überspielen, sondern wo immer es möglich ist, für geänderte Dateien den jeweiligen Override-Ordner verwenden.
Ebenfalls ratsam: Wenn irgendwelche Zusatzmodule eingebaut werden, dann notieren, welche man eingebaut hat. Und auch Module immer nach dem Override-Prinzip einbauen, gute Module machen das ohnehin, es wimmelt von Ordnern namens DEINTEMPLATE…

Die Override-Ordner heißen immer so, wie das Template, das man im Shop aktiv hat, in diesem Beispiel nenn ichs mal classic.
Zen-Cart schaut immer erst, ob eine benötigte Datei im jeweiligen Override-Ordner vorhanden ist. Falls ja, wird diese verwendet. Falls nein, wird die entsprechende Originaldatei verwendet.

Dieses Override System steht für folgende Dinge zur Verfügung:

1) Sprachdateien
Die werden wohl am häufigsten angepasst und mit eigenen Texten versehen.
Ändere ich z.B. einen Text in der includes/languages/german.php, dann überschreibe ich nicht die

bestehende damit, sondern lege sie nach:
includes/languages/classic/german.php
Ändere ich einen Text in der includes/languages/german/header.php, dann überschreibe ich nicht die bestehende damit, sondern lege sie nach:
includes/languages/german/classic/header.php
Ändere ich einen Text in der includes/languages/german/extra_definitions/rl.vat_info.php, dann überschreibe ich nicht die bestehende damit, sondern lege sie nach:
includes/languages/german/extra_definitions/classic/rl.vat_info.php
Macht man später mal ein Update gehen keinerlei Änderungen in den Sprachfiles verloren, weil man ja jede Änderung im Overrideordner der Sprachfiles liegen hat und die werden beim Update nicht überschrieben.

2) Template
Die Originaltemplates liegen in includes/templates/template_default
Die Struktur, die man dort vorfindet (Unterordner common, images, css, jscript, sideboxes usw.) lässt sich komplett ins eigene Template übernehmen.
Der erste Schritt zu einem eigenen Template ist also, in includes/templates/classic exakt diese Struktur zu verwenden und geänderte Templatedateien nie in includes/templates/template_default zu spielen, sondern immer in includes/templates/classic
Macht man später mal ein Update gehen keinerlei Änderungen im

eigenen angepassten Template verloren, weil man ja jede Änderung im Overrideordner des Templates liegen hat und die werden beim Update nicht überschrieben.

3) includes/modules
Viele Module ändern Funktionalitäten in Dateien in includes/modules
Gut gemachte Module werden immer das Override System nutzen und geänderte Corefiles dafür in includes/modules/classic legen.
Auch hier wird bei einem Update dann nichts überschrieben

4) includes/extra_datafiles
Auch hier gehören geänderte Dateien nicht überschrieben, sondern nach:
includes/extra_datafiles/classic

5) includes/init_includes
Hier steht ein kompletter Override Ordner includes/init_includes/overrides zur Verfügung, in den geänderte Dateien aus includes/init_includes gelegt werden sollten.

Sollte man bisher immer Originaldateien verändert haben, dann wäre ein Update ein guter Zeitpunkt, den Shop VORHER wo immer es möglich ist, nach dem Override Prinzip zu organisieren.