Leitfäden

Best Practices Für Die User Exit Entwicklung

Dieser Leitfaden dokumentiert Coding-Richtlinien, deren Einhaltung zu fehlerfreien und updatefähigen User Exits führen sollen.

Motivation hinter dem Leitfaden ist, dass in der Vergangenheit häufig User Exits programmiert wurden, welche bspw. auf Variablen oder Methoden zugeriffen haben, die eigentlich nicht für den Gebrauch in User Exits entworfen wurden. Das wiederum erhöht die Wahrscheinlichkeit für Fehler bei der Ausführung von User Exits, besonders nach Updates von SQUEEZE. Aus dieser Motivation wird perspektivisch eine Scripting API entwickelt und bereitgestellt.

Best Practices

Methoden vs. Klassen-Eigenschaften

> Grundsätzlich ist die Verwendung von Methoden dem direkten Zugriff auf Klassen-Eigenschaften vorzuziehen.

```
$creditor = $xDoc->getFieldByName("Creditor");

// Bad
$creditor->value->value = "New Creditor";

// Good
$creditor->getValue()->setValue("New Creditor");
```

Der direkte Zugriff auf Eigenschaften, insbesondere bei Verkettung (`$creditor->value->value`) kann zu Exceptions führen, falls eine Eigenschaft null ist. Außerdem kann so nicht gewährleistet werden, dass die Werte, die gesetzt werden sollen, normalisiert werden sollen.

Deprecations - Veraltete APIs nicht

verwenden

Wenn Sie ihre IDE korrekt konfiguriert haben, werden Ihnen veraltete Methoden, Felder und Klassen durchgestrichen dargestellt. Das ist im folgenden Screenshot zu sehen:

Hier ist bspw. das Feld `value` der Klasse `xDocValue` veraltet. Im Tooltip ist zu lesen:

> Please use the setter, getter or adder to modify this field. This field will not be removed but could eventually be made private.

Sie sollten hier also versuchen Getter und Setter zu verwenden:

```php
// Good
$val = $tax3->getValue();
$val->setValue("0.00");
$val->setText("0.00");

// Bad
$tax3->value->value = '0.00';
$tax3->value->text = '0.00';
$tax3->state = "OK";
$fieldList->updateField($tax3);
}
```

Deprecations werden phasenweise eingeführt als frühzeitige Warnungen, dass der Standard sich zukünftig verändern **kann**.

Solche Veränderungen werden in den Changelogs als Breaking Changes festgehalten, damit Sie bei einem Update prüfen können, ob es Handlungsbedarf gibt.

Fehlende Funktionalitäten

Sollten Funktionalitäten in Script-Libraries oder der Scripting API fehlen, ist es sinnvoll die fehlenden Funktionalitäten bspw. im Forum zu diskutieren. So können Funktionalitäten, die häufiger benötigt werden in den Standard aufgenommen werden.

Wiederverwendbare Methoden

Es bietet sich an einzelne Funktionen, die in User Exits umgesetzt werden müssen, als seperate, wiederverwendbare Methoden umzusetzen. Der Vorteil davon ist, dass die Methoden einfacher kopiert und bei anderen Systemen erneut eingesetzt werden können, vorausgesetzt die Produktversionen sind kompatibel.

Die Zusammenfassung von gleichartigen Methoden in Klassen ist zu empfehlen.