QFQ - Feature #5456

TWIG als Template Engine fuer Report Syntax

17.02.2018 11:31 - Carsten Rose

Status:	Closed	**Start date:**	17.02.2018
Priority:	Normal	**Due date:**	26.07.2019
Assignee:	Marc Egger	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	19.7.1	**Spent time:**	0.00 hour
Discuss:			

Description

History

#1 - 30.04.2018 18:10 - Carsten Rose

- Target version set to next

#2 - 14.06.2018 17:34 - Benjamin Baer

- Target version changed from next to 18.10.3

#3 - 02.07.2018 08:52 - Carsten Rose

- Assignee changed from Nicola Chiapolini to Elias Villiger

#4 - 02.07.2018 08:55 - Carsten Rose

- Priority changed from Normal to High

#5 - 02.07.2018 15:44 - Carsten Rose

- File qfq-record.txt added

- File result.png added

- File test.html.twig added

- File twig.patch added

Anyway, die Anhänge: * twig.patch: Die nötigen Änderungen am QFQ-Source * test.html.twig: Das verwendete Template file * qfq-record.txt: Der Inhalt des QFQ-Record auf der Test-Seite * result.png: Screenshot des Resultats

Zusätzliche Voraussetzung: Twig muss natürlich installiert sein (Paket `php-twig` in Debian stable).

Finde das sieht schon sehr vielversprechend aus... Was noch fehlt sind natürlich die UnitTests und die Doku

härzlichi Grüäss und es schöns Wuchenend
Nicola

#6 - 27.07.2018 18:04 - Elias Villiger

Bei Gebrauch von Twig würde ev. die Generalisierung von QFQ etwas verloren gehen:

- Die einzelnen Felder (row.firstname, row.name, row.link|qfqlink, ...) müssten pro Template jeweils einzeln aufgelistet werden (? - jedenfalls im Fall von speziellen Spalten wie AS _link)
- Spezielle Spalten wie _Pagee oder versteckte Spalten AS _hidden müssten vermutlich auch einzeln gehandhabt werden (wie qfqLink für AS _link) - je nach angebotener Funktionalität könnte das schnell kompliziert werden.

Die Idee des Templatings finde ich aber sehr attraktiv, da man so einiges an repetitivem Tippen einsparen könnte. Mein Vorschlag wäre, ein QFQ-eigenes Templating anzubieten:

```
10 {
  sql = SELECT * FROM person
  template = fileadmin/qfqTemplates/standardTable
}
```

Der User kann dann entsprechende Templates abspeichern, welche in QFQ-Notation gehalten sind, z.B.

```
head = <table class="table">
tail = </table>
rbeg = <tr>
rend = </tr>
fbeg = <td>
fend = </td>
```

Bequem wäre auch, wenn man einzelne Felder des Templates überschreiben könnte (Template wird zuerst eingelesen, dann mit zusätzlichen angegebenen Feldern überschrieben):

```
20 {
  sql = SELECT * FROM person
  template = fileadmin/qfqTemplates/standardTable
  head = <table class="table table-condensed"><tr><th>Id</th><th>Name</th><th>E-Mail</th></tr>
}
```

#7 - 27.07.2018 21:22 - Nicola Chiapolini

> Bei Gebrauch von Twig würde ev. die Generalisierung von QFQ etwas verloren gehen [...]

Korrekt. Aber das ist ja auch ein Stück weit die Idee. Ziel von Template-Engines ist üblicherweise Darstellung von Datenzugriff zu trennen. (d.h. das SQL-Query sollte keine Info zur Darstellung enthalten)

(Aber es wäre natürlich nicht besonders schwierig Templates zu erstellen, die die Darstellung vom Spaltennamen abhängig machen.)

> Mein Vorschlag wäre, ein QFQ-eigenes Templating anzubieten

Davon würde ich dringend abraten. Wenn immer möglich sollten wir auf bestehende und etablierte Tools zurückgreifen. (und im Dekanat werden definitiv weiter mit Twig arbeiten, das hat sich bei uns bereits sehr bewährt. Nötigenfalls werden wir das also auch weiterhin selbst reinpatchen) Falls du konkrete, komplexere Template-Beispiele möchtest, einfach melden.

> Bequem wäre auch, wenn man einzelne Felder des Templates überschreiben könnte

Twig bietet die Möglichkeit aus einem Template andere Teil-Templates zu laden. Das ist aber in meinem QFQ-Code noch nicht richtig integriert.

#8 - 27.07.2018 21:32 - Elias Villiger

- Assignee changed from Elias Villiger to Carsten Rose

#9 - 27.07.2018 22:03 - Elias Villiger

Macht Sinn. Ich dachte, es ginge vor allem darum, den Boilerplate von head/tail/rbeg/... zu reduzieren, aber wenn es um eine Trennung von Design und SQL-Code geht, macht Twig schon Sinn.

Das oben beschriebene "QFQ-eigene Templating" wäre auch nicht wirklich ein "Templating" in dem Sinne, sondern eher ein Auslagern häufiger Darstellungsoptionen zur Wiederverwendung.

Für eigenen meinen Gebrauch von QFQ (geolean) sehe ich keine grossen Anwendungen für Templating im Sinn von Twig; aber da es sich bei euch bewährt, würde mir ein Einblick in euren Umgang damit bestimmt weitere Perspektiven zeigen.

Ich habe das Ticket jetzt wieder Carsten zugewiesen, dann können wir nach seinen Ferien weiterschauen.

#10 - 30.07.2018 11:37 - Nicola Chiapolini

- File display_committee.txt added

- File display_state.txt added

- File screenshot.png added

Hier noch zwei komplexere Beispiele für den Einsatz von Twig. (display_committee.txt, display_state.txt und screenshot.png)

#11 - 27.10.2018 14:15 - Carsten Rose

- Target version changed from 18.10.3 to 18.12.1

#12 - 11.12.2018 10:20 - Carsten Rose

- Target version changed from 18.12.1 to 141

#13 - 05.06.2019 13:05 - Marc Egger

- Assignee changed from Carsten Rose to Marc Egger

- Priority changed from High to Normal

#14 - 05.06.2019 13:05 - Marc Egger

- Due date set to 19.06.2019

#15 - 19.06.2019 15:55 - Carsten Rose

- Target version changed from 141 to 20.12.0

#16 - 04.07.2019 11:00 - Carsten Rose

- Due date changed from 19.06.2019 to 26.07.2019

#17 - 17.07.2019 11:38 - Carsten Rose

- Target version changed from 20.12.0 to 19.7.1

#18 - 17.07.2019 11:44 - Carsten Rose

- Status changed from New to Closed

Files

qfq-record.txt	396 Bytes	02.07.2018	Carsten Rose
result.png	18.7 KB	02.07.2018	Carsten Rose
test.html.twig	112 Bytes	02.07.2018	Carsten Rose
twig.patch	4.86 KB	02.07.2018	Carsten Rose
display_committee.txt	1023 Bytes	30.07.2018	Nicola Chiapolini
display_state.txt	965 Bytes	30.07.2018	Nicola Chiapolini
screenshot.png	44.6 KB	30.07.2018	Nicola Chiapolini