maxx PORTAL 2.0 Template Engine

Mit entsprechender Lizenz können Sie als Portalbetreiber die Erscheinung des Portals für Ihre Kunden völlig individuell gestalten. Sie können die vorgefertigten System-Templates sowie eigene Templates mit Hilfe einer einfachen Scriptsprache auch funktionell nach Ihren Vorstellungen ändern bzw. entwerfen. In dieser Dokumentation finden Sie eine kurze Einweisung sowie eine Übersicht der Elemente dieser Scriptsprache.

Grundlegendes

Die Scriptsprache können Sie ähnlich wie PHP Code überall im HTML Code einbetten. Die Kommandos werden serverseitig interpretiert, Ihre Kunden sehen die Elemente der Scriptsprache nicht.

Die Sprache umfasst einfache Funktionen, mit etwas Kreativität können Sie sogar eigene Funktionen schreiben. Kombiniert mit clientseitigem JavaScript sind Ihren Ideen fast keine Grenzen gesetzt.

Nutzen Sie frei definierbare Variablen sowie Umgebungsvariablen, um Ihre Templates noch dynamischer zu gestalten.

Die Syntax

Alle Scriptelemente werden in geschweiften Klammern dargestellt. Ein Funktionsaufruf erfolgt durch ein kaufmännisches UND – Beispiel:

```
{&name}
```

Variablen werden in der Regel wie folgt angesprochen:

```
{name}
```

Variablen innerhalb von Funktionen werden wie folgt übergeben:

```
{&funktion,#variable1}
```

Nicht allen Funktionen können Variablen als Parameter übergeben werden. Näheres hierzu erfahren Sie in den folgenden Kapiteln.

Verschachtelte Scriptelemente sind nicht möglich – Negativbeispiel:

```
{&funktion1,{&funktion2}}
```

Scriptelemente

Link erzeugen: {&link,[site],[param],[name]}

Über die URL, die von Ihren Links angesteuert werden soll, brauchen Sie sich innerhalb des Portals keine Gedanken zu machen. Diese Funktion erzeugt einen vollwertigen HTML Link oder eine URL.

[site]
Der Name des Templates, welches angezeigt werden soll.

[param]
Optional zusätzliche Parameter in der Form „name1=wert1&name2=wert2&… “.

[name]
Optionaler HTML Code, der innerhalb des a-HTML-Tags platziert werden soll. Ignorieren Sie diesen Parameter, um lediglich eine URL zu generieren.

Grafik anzeigen: {&grafik,[name],[html]}

Ihre hochgeladenen Grafiken können Sie anhand des vergebenen Namens mit dieser Funktion ansprechen.

[name]
Der Name der Grafik.

[html]
Soll HTML generiert werden? Der Wert ist 0 (nein) oder 1 (ja).

Portal Logo anzeigen: {&logo}

Mit dieser Funktion erhalten Sie den HTML Code Ihres Portal Logos.

Template includieren: {&include,[template]}

Diese Funktion lädt zur Laufzeit ein anderes Template und fügt den Inhalt an der Stelle des Funktionsaufrufs ein.

[template]
Der Name des Templates.

Farbe holen: {&color,[name]}

Diese Funktion liefert Ihnen die vordefinierte HTML Farbe.

[name]
Der Name der vordefinierten Farbe.

Datum / Uhrzeit formatiert ausgeben: {&date,[format],[epochensekunden]}

Diese Funktion zeigt das aktuelle Datum bzw. die aktuelle Uhrzeit oder Datum oder Uhrzeit der angegebenen Epochensekunden.

[format]
Die Angabe zur Formatierung können Sie in der PHP Dokumentation nachlesen. Geben Sie kein Format an, wird das Datum in der Form „TT.MM.JJJJ“ dargestellt.

[epochensekunden]
Die Angabe eines Zeitstempels ist optional. Geben Sie keinen Zeitstempel an, wird der aktuelle Zeitpunkt verwendet.

Navigations-Tree erzeugen: {&tree,[width],[height]}

Diese Funktion erzeugt den Navigations-Baum. Wichtig: Die Funktion kann nur einmal aufgerufen werden. Bei einem zweiten Aufruf erzeugt die Funktion eine Fehlermeldung.

[width]
Die CSS Breitenengabe (optional).

[height]
Die CSS Höhenangabe (optional).

Meldung anzeigen: {&message,[text]}

Fehlermeldungen und Erfolgsmeldungen oder Hinweise werden während der Portal-Funktions-Verarbeitung gesammelt und können mit dieser Funktion angezeigt werden.

[text]
Hängt optional die angegebene Nachricht an die bereits vorhandenen Meldungen und zeigt diese an. Der Meldungs-Buffer wird dadurch geleert.

Bedingungen mit IF und ELSE

Einfache Fallentscheidungen können mit Hilfe von IF/ELSE realisiert werden. Beispiel:

```
{&if,1 eq 1}
<p>1 entspricht 1</p>
{&else}
<p>1 entspricht NICHT 1</p>
{&endif}

{&set,zahl,2}
{&if,#zahl eq 1}
<p>{zahl} entspricht 1</p>
{&endif}

{&if,#zahl gt 1}
<p>{zahl} ist größer als 1</p>
{&endif}

{&if,#zahl ne}
<p>Variable zahl ist nicht leer</p>
{&endif}

{&set,string,a}
{&if,#zahl ne #string}
<p>zahl entspicht nicht string</p>
{&endif}
```

Ausgabe:

```
1 entspricht 1

2 ist größer als 1

Variable zahl ist nicht leer

zahl entspricht nicht string
```

Als Vergleichsoperatoren gelten:

- eq → Equal → ist gleich
- ne → Not equal → ist ungleich
- lt → Lighter than → ist kleiner als
- gt → Greater than → ist größer als

Die Vergleichsoperatoren vergleichen den vor- und nachstehenden Wert oder Variable miteinander, sodass der Block innerhalb von IF und ENDIF oder IF und ELSE nur dann ausgeführt wird, wenn die Bedingung zutrifft. Dabei ist darauf zu achten, dass eine Variable als Vergleichsoperator mit einer Raute zu notieren ist.

Variablen setzen: {&set,[name],[wert],[anhang]}

Setzt den Wert einer Variablen, bzw. verknüpft ggf. zwei Strings. Wert und Anhang können eine Variable sein (Kennzeichnung durch vorangestellte Raute).

[name]

Name der Variable.

[wert]

Wert der Variablen (kann ebenfalls eine Variable sein, dann aber mit vorangestellter Raute).

[anhang]

String, der an den Wert angehängt werden soll.

Beispiele

Script:

```
{&set,test,Hallo!}
{test}
```

Ausgabe:

```
Hallo!
```

Script:

```
{&set,test,Hallo }
{&set,name,Werner}
{&set,test,#test,#name}
{&set,test,#test,!}
{test}
```

Ausgabe:

```
Hallo Werner!
```

Text2HTML: {&2html,[text]}

Wandelt Text in HTML Code um.

[text]

Der umzuwandelnde Text.

Übersetzten Text laden: {&text,[name]}

Lädt einen Text in der aktuellen Sprache.

[name]

Name des Textbausteins.

Interpreter abbrechen: {&exit}

Diese Funktion beendet die Interpretation des Templates sofort.

Sprache setzen: {&language}

Diese Funktion setzt die Sprache des aktuellen Besuchers in Ihrem Portal. Damit die Sprache gesetzt werden kann, muss in der Variable „lid“ die ID der Sprache stehen.

http Protokoll ermitteln: {&httpprotocol}

Diese Funktion ermittelt das aktuelle http Protokoll (http oder https).

Hostname ermitteln: {&host}

Diese Funktion ermittelt den FQDN des Portalservers.

Ein anderes Template interpretieren: {&parse,[name],[echo]}

Diese Funktion weist den Interpreter an ein weiteres Template zu parsen.

[name]

Der Name des Templates.

[echo]

Soll das interpretierte Template ausgegeben werden? Der Wert ist 0 (nein) oder 1 (ja).

Eine Zeichenfolge interpretieren: {&parsestring,[string],[echo]}

Diese Funktion parst eine Zeichenfolge.

[string]

Die zu parsende Zeichenfolge.

[echo]

Soll die interpretierte Zeichenfolge ausgegeben werden? Der Wert ist 0 (nein) oder 1 (ja)

Benutzerinformationen: {&userinfo}

Zeigt die Benutzerrechte als Icons an.

ToolTips: {&tooltip,[text]}

Gibt den Code für einen ToolTip zurück – Beispiel:

```
{&tooltip,Hinweistext}
```

Erzeugt diesen Code:

```
onmousemove=“…“ onmouseout=“…“
```

[text]

Der Text des ToolTips.

Templateeditor: {&quickedit}

Diese Funktion erzeugt einen Link zum Bearbeiten des aktuellen Templates.

Eigene „Funktionen“ schreiben

Sie können Templates innerhalb von Templates definieren und somit sozusagen eine eigene Funktion realisieren, die Sie an beliebiger Stelle in einem Template aufrufen können. Die Definition Ihrer Funktionen sollten Sie in einem separaten Template vornehmen. Nennen Sie dieses Template beispielsweise „funktionen“.
Ein solches Template, in dem ausschließlich Funktionen definiert werden, verwendet maxx PORTAL 2.0 für Tabellen im System-Template „tabellen“. Die dort definierten Funktionen werden allerdings lediglich intern verwendet.

Dieses Beispiel-Template enthält die Funktionen „fnc1“ und „fnc2“:

```
{&begintmpl,fnc1}<p>Testfunktion 1 [parameter]{&endtmpl}
{&begintmpl,fnc2}[&if,#parameter ne]<p>Testfunktion 2
[parameter]</p>[&else]<p>Kein Parameter für Testfunktion
2</p>[&endif]{&endtmpl}
```

In einem anderen Template verwenden Sie diese Funktionen dann wie folgt:

```
{&parse,funktionen,0}
{&set,parameter,test}
{fnc1}
{fnc2}
{&set,parameter,}
{fnc2}
```

Ausgabe dieses Templates:

```
Testfunktion 1 test

Testfunktion 2 test

Kein Parameter für Testfunktion 2
```

Durch das Parsen des Templates „funktionen“ ohne Ausgabe, werden die Funktionen in den Variablen „fnc1“ und „fnc2“ zur Verwendung vorbereitet. Innerhalb des aufrufenden Template werden diese Variablen ausgegeben, um die jeweilige Funktion auszuführen.

Innerhalb der Definition der Funktionen werden Scriptelemente verwendet, die erst beim Aufruf der Funktion ausgeführt, nicht jedoch bereits beim Vorbereiten der Funktionen durch „parse“ interpretiert werden sollen, in eckigen Klammern dargestellt. Achtung: Die Verwendung von Scriptelementen in geschweiften Klammern innerhalb der Funktionsdefinitionen kann nicht interpretiert werden!

Umgebungsvariablen

Je nach Template sind unterschiedliche Umgebungsvariablen verfügbar. Eine Liste der verfügbaren Variablen erhalten Sie durch diese Funktion:

{&env}

Diese Variablen sind immer verfügbar:

Name	Wert	Info
sid	$	Session ID
$	$	Alle an das Portal übergebenen Parameter sind als Umgebungsvariablen verfügbar
portal	#	ID des aktuellen Portals
biller	0|1	Ist der Besucher Portalbetreiber?
signer	0|1	Hat der Besucher das Recht Belege zu signieren?
sender	0|1	Hat der Besucher das Recht Belege zu veröffentlichen?
dataprov	0|1	Hat der Besucher das Recht XML/CSV bereitzustellen?
customer	0|1	Ist der Besucher Kunde des Portals?
profil.*	$	„*“ steht für das Objekt des Profils des aktuellen Benutzers
profil.s.*	$	„*“ steht für das Objekt der Einstellungen zum Profil des aktuellen Benutzers
portal.*	$	„*“ steht für das Objekt der Portaleigenschaften des aktuellen Portals
portal.s.*	$	„*“ steht für das Objekt der Einstellungen des aktuellen Portals

Freie Variablen

Sie können wie bereits erwähnt freie Variable definieren. Die Variablennamen dürfen aus diesen Zeichen bestehen:

- Buchstaben a-z und A-Z
- . (Punkt)
- Unterstrich

Verwenden Sie ungültige Zeichen, kann die Variable nicht interpretiert werden.

Fehlermeldungen

Der Interpreter gibt Fehlermeldungen direkt im HTML Code gut sichtbar und im Klartext aus, sodass Script-Fehler recht einfach erkannt werden können. Sehen Sie sich die vordefinierten System-Templates an, um zu lernen, was mit dem maxx PORTAL 2.0 Script-Interpreter möglich ist, und um lauffähige und praxisbezogene Beispiele zum Einsatz der Funktionen zu erhalten.