Neue Funktionen in Virthos 3.2

Stand: 03.02.2012

Automatische Bildskalierung

Der `vtMedia`-Platzhalter erlaubt ab der Virthos-Version 3.2 einen zweiten Parameter, der die Größe bestimmt, in der das jeweilige Bild ausgegeben wird:

```
<img src="{{vtMedia: Bild1, parameter}}">
```

Der Parameter kann folgende Werte erhalten:

Schema	*Beispiel*	*Wirkung*
n	`300`	Das Bild wird so verkleinert, dass keine Seite länger ist als dem angegebenen Wert *n* entspricht (in Pixeln gemessen). Dies ist gleichbedeutend mit der Angabe `wnhn`.
`wn`	`w300`	Das Bild wird so verkleinert, dass die Breite dem angegebenen Wert *n* entspricht.
`hn`	`h300`	Das Bild wird so verkleinert, dass die Breite dem angegebenen Wert *n* entspricht.
`wnhm`	`w300h200`	Das Bild wird so verkleinert, dass es nicht breiter als *n* und nicht höher als *m* Pixel ist.
`wnhmf`	`w300h200f`	Das Bild wird so verkleinert **und ggf. beschnitten**, dass es genau *n* mal *m* Pixel groß ist. Das „f“ steht für force (erzwingen).

Grundsätzlich werden Bilder, die kleiner sind als die vorgegebene Größe, nicht skaliert. Ein zusätzliches *x* (eXpand = erweitern) sorgt dafür, dass solche Bilder mittels Interpolation vergrößert werden, um das angegebene Format auszufüllen, zum Beispiel:

```
<img src="{{vtMedia: Bild1, w300h200fx}}" width="300" height="200">
```

Die Bildskalierungsfunktion lässt sich auch für Bilder verwenden, die unmittelbar (also nicht per `vtMedia`-Platzhalter) im Template referenziert sind. Dazu muss die Bild-URL folgendermaßen aufgebaut werden:

```
http://.../system/resources/imager.php/pfad-zum-bild?parameter
```

Wenn der `pfad-zum-bild` mit der Zeichenkette `-/` beginnt, wird er relativ zum Virthos-Verzeichnis interpretiert, ansonsten relativ zum Verzeichnis `data/media` bzw. zu dem Verzeichnis, das im Konfigurationsparameter `imagerBasePath` festgelegt ist (s. u.)

Die Bildskalierung funktioniert nur bei Bildern im JPEG-, GIF- und PNG-Format und nur, wenn in PHP die GD-Library zur Verfügung steht. Sollen sehr große Bilddateien skaliert werden, muss u. U. die PHP-Speicherzuteilung (memory_limit) erhöht werden.

Die skalierten Bilder werden stets im JPEG-Format erzeugt und in einem Cache-Verzeichnis gespeichert. Dieses Verzeichnis wird regelmäßig aufgeräumt, damit die Zahl der Dateien oder der verwendete Speicherplatz begrenzt bleiben.

Die Skalierungsfunktion lässt sich über eine Reihe von Konfigurationsparametern beeinflussen, die in der folgenden Tabelle beschrieben sind:

Parameter	*Beschreibung*
`imagerJpegQuality`	Qualitätsstufe für die JPEG-Komprimierung (Vorgabe: `90`)
`imagerBasePath`	Basisverzeichnis für relative Bildpfade (Vorgabe: `data/media`)
`imagerCachePath`	Verzeichnis, in dem die Cache-Bilder gespeichert werden (Vorgabe: `data/cache/images`)
`imagerCacheMaxSize`	maximale Größe für den Bilder-Cache in Bytes (`0` = unbegrenzt, Vorgabe: `1000000`)
`imagerCacheMaxFiles`	maximale Anzahl an Bildern, die im Cache gespeichert werden (Vorgabe: `0` = unbegrenzt)
`imagerCacheMaxAge`	maximale Vorhaltezeit (Tage) für Bilder im Cache (Vorgabe: `0` = unbegrenzt)
`imagerCleanupInterval`	Aufräumintervall für den Bilder-Cache in Sekunden (Vorgabe: `3600`)

Captcha-Funktion

Ab Version 3.2 stellt Virthos Master eine Funktion zur Verfügung, um Formulare mit Hilfe von Captcha-Bildern abzusichern. Dadurch lässt sich verhindern, dass Formulareingaben, die von Robotern stammen, weiterverarbeitet werden.

Um de Captcha-Schutz zu verwenden, müssen in das betreffende Formular zwei zusätzliche Elemente eingebaut werden, ein Bild und ein Eingabefeld:

```
<form action="{{vtLink}}" method="post">
  ...
  <img src="{{vtCaptcha}}" width="140" height="40" alt="captcha">
  ...
  <input name="captcha_code" type="text" size="8">
  ...
</form>
```

Der Name des Eingabefeldes (im obigen Beispiel `captcha_code`) kann frei gewählt werden. Bei der Verarbeitung des Formulares wird der eingegebene Wert mit dem Code verglichen, den der Captcha-Generator erzeugt hat:

```
<!--{{vtIf: {vtPostValue:captcha_code} .neq. {vtCaptchaSecret} }}-->
  <p>Der eingegebene Captcha-Code war nicht korrekt.</p>
<!--{{vtElse}}-->
  ...
<!--{{vtEndIf}}-->
```

Der von Virthos bereitgestellte Captcha-Generator liegt unter *system/resources/captcha/captcha.php*. Damit er funktioniert, muss PHP mit der GD-Bibliothek und FreeType-Support konfiguriert sein. Es ist auch möglich, einen eigenen Captcha-Generator zu verwenden. Dazu stehen zwei Konfigurationsparameter zur Verfügung:

Parameter	*Beschreibung*
`captchaUrl`	Vollständige oder relative URL des Skriptes, das als Captcha-Generator verwendet werden soll (Vorgabe: `system/resources/captcha/captcha.php`)
`captchaSessionVar`	Name der PHP-Sessionvariablen, in der der Captcha-Code gespeichert wird (Vorgabe: `vtcaptcha`)

Suchmaschinenfreundliche URLs

Bei umfangreicheren Webprojekten kann es leicht passieren, dass ein und dieselbe Webseite über verschiedene URLs erreichbar ist. Dies gilt insbesondere dann, wenn Sie Funktionen nutzen, die zum Start einer Session führen. Normalerweise werden die Sessioninformationen in Form von Cookies transportiert, wenn der Besucher jedoch keine Cookies unterstützt – wie es bei Suchmaschinenrobotern der Fall ist –, bindet Virthos die Sessioninformationen in die URLs ein, was dazu führt, dass sich diese dann quasi von Seitenaufruf zu Seitenaufruf ändern.

Google setzt inzwischen ausgefeilte Methoden ein, um solche Sessionparameter aus den URLs wieder herauszufiltern, aber es gibt immer wieder Fälle, in denen diese Methoden versagen. Das kann dann im schlimmsten Fall dazu führen, dass der Google-Roboter einen Webauftritt überhaupt nicht mehr vollständig indizieren kann, weil er beim Verfolgen der Links in eine Endlosschleife gerät. Eine andere Auswirkung kann darin bestehen, dass dieselben Seiten mehrfach in den Google-Suchergebnissen auftauchen, was mit der Gefahr verbunden ist, durch ein schlechtes Ranking abgestraft zu werden. Google mag keinen ‚duplicate content'.

Um diesem Problem zu begegnen, hat Google einen Weg definiert, auf dem man Suchmaschinenrobotern die „richtige" URL für eine Webseite mitteilen kann, unabhängig davon, unter welcher URL sie gerade aufgerufen wurde. Diese „Canonical URL", wie Google sie nennt, lässt sich einfach über ein Meta-Tag in den HTML-Kopf einer Seite einbinden, zum Beispiel so:

```
<link rel="canonical" href="http://www.meinedomain.de/impressum.html">
```

Virthos 3.2 macht es Ihnen leicht, diese Möglichkeit zu nutzen. Der `vtLink`-Platzhalter verfügt dazu über einen neuen Parameter `-urltype`, der den Typ der erzeugten URL angibt. Das könnte dann so aussehen:

```
<link rel="canonical" href="{{vtLink:-urltype='readable'}}">
```

Der Wert `readable` bedeutet, dass eine gut lesbare und suchmaschinenfreundliche URL erzeugt wird, die den vollständigen Seitenpfad und -namen enthält. Alternativ wäre auch der Wert `short` möglich, bei dem eine sehr kurze URL auf Basis der Seitennummer erzeugt wird.

Wenn Sie einen dieser beiden URL-Typen verwenden, werden außer `-pg` alle anderen Parameter im `vtLink`-Platzhalter ignoriert. Sie können also beispielsweise nicht auf spezielle Methoden oder auf andere Sprachen verlinken.

Wichtig zu beachten: Die erzeugten URLs sind **nicht** sessionsicher. Wenn ein Besucher ohne Cookie-Unterstützung einem solchen Link folgt, startet Virthos eine neue Session, und die alten Sessiondaten gehen verloren. Der neue Parameter sollte daher nur in Sonderfällen benutzt werden, zum Beispiel zur Angabe der „Canonical URL" oder um Hyperlinks in E-Mails einzubinden.