ELO Notify

Stand: 12.05.2014

Die ELOas Libray notify hilft beim automatischen Versand von Emails aus einem Workflow oder zu einem Workflow. Weiterhin können auch automatisch Feed Einträge erzeugt werden.

Inhalt

1 Vorbereitungen

Das Modul notify benötigt relativ neue Funktion aus den Modulen elo, ix und mail. Deshalb kann man es nicht einzeln installieren sondern es wird sinnvollerweise ein Update auf die neuste JavaScript Libray Version des ELOas durchgeführt. Diese bringt dann das Modul notify automatisch mit.

Es ist wegen der Feed Unterstützung nur für ELOprofessional 9 und ELOenterprise 9 einsetzbar.

2 Konfiguration

Das Modul Notify führt drei unterschiedliche Aufgaben aus.

1. Anwender können eine automatische Mail Benachrichtigung „bestellen" wenn Sie Termine in Ihrer Aufgabenansicht besitzen.
2. Ein Workflow kann einen Knoten zum Mailversand ohne Skriptaufwand einrichten.
3. Ein Workflow kann einen Knoten zum Eintragen eines Feed Kommentars ohne Skriptaufwand einrichten.

Damit diese Funktionen aktiv werden, muss jeweils eine kurze Rule im Rules Verzeichnis hinterlegt werden. Die Aufgabe der Rule ist lediglich, eine Intervall-Triggerung der Notify Funktion auszulösen.

3 Mail Benachrichtigungen für Termine

Das Modul Notify bietet die Möglichkeit für bestimmte Anwender oder alle Anwender eine Benachrichtigung zu versenden wenn dieser aktive Aufgaben besitzt. Gibt es mehrere aktive Aufgaben, werden diese alle in eine gemeinsame Mail zusammengefasst. Dabei kann jeder Anwender für sich selber bestimmen, ob er die Mail haben möchte oder nicht. Weiterhin kann der Anwender bestimmen, ob er eine Benachrichtigung zu einem Workflow nur einmal oder in jeder Sammelmail erhält.

Diese Funktion kann über eine Rule zu bestimmten Zeitpunkten gestartet werden, z.B. jeden Morgen um 6:00 Uhr. Dann hat jeder Anwender zu Dienstbeginn eine entsprechende Benachrichtigung in seinem Postfach.

Der erste Schritt besteht also aus der Erstellung einer passenden Rule. Hier wird festgelegt, wann die Benachrichtigung erfolgen soll und welche Personen Zugriff auf diese Funktion haben sollen.

```
<ruleset>
  <base>
```

```
    <name>NotifyWf</name>
    <search>
      <name>"DIRECT"</name>
      <value>"1"</value>
      <mask>0</mask>
      <max>200</max>
    </search>
    <interval>6:00</interval>
  </base>
<rule>
  <name>CheckSendMail</name>
  <condition></condition>
  <script>
    mail.setSmtpHost("MyMailServer");
    var sender = "Inhouse-Support@elo.com";
    var subject = "Workflowbenachrichtigung"
    notify.processAllUsers(sender, subject, true, true, true);
  </script>
</rule>
<rule>
  <name>Global Error Rule</name>
  <condition>OnError</condition>
  <script></script>
</rule>
</ruleset>
```

In dieser Form wird der Mail Versand jeden Morgen um 6:00 Uhr gestartet und die Aufgabenlisten aller Anwender überprüft. Wenn man die Funktion notify.processUserItems statt notify.processAllUsers aufruft, kann man den Versand auf bestimmte Anwender beschränken.

Eine weitere wichtige Aufgabe der Rule besteht darin, dass Sie den Namen des Mailservers mittels mail.setSmptHost bekannt geben müssen. An dieser Stelle können Sie bei Bedarf auch die Anmeldedaten hinterlegen. Beachten Sie bitte, dass der Mailserver auch so konfiguriert ist, dass er von dem ELOas Server aus SMTP Mails entgegen nimmt. Das ist bei Exchange nicht automatisch der Fall.

Der Administrator hat die volle Kontrolle darüber, wie die Mail aussieht, die verschickt wird. Hierfür muss im Ordner „Administration\ELOas Base\Misc" eine html Datei mit dem Namen wfreminder hinterlegt werden. Diese Datei enthält den Mailkörper und muss bestimmte Markierungen für Variablen enthalten, die zur Laufzeit gegen die jeweiligen Indexwerte ausgetauscht werden.

Der Bereich für die Termine kann Platzhalter enthalten, die aus den aktuellen Termindaten gefüllt werden. Diese Platzhalter beinhalten Workflownamen oder Datum sowie auch Indexzeilen. Zudem können eigene Termine von Gruppenterminen sowie eskalierten Terminen farblich getrennt werden. Dafür die es Platzhalter für CSS Formatierungen

$$className$$, welcher die Werte normal, urgent und group annehmen kann. In Ihrem html Template können Sie dann in der CSS Definition die gewünschten Farben oder Schriftgrößen festlegen.

Die vollständige Liste aller Platzhalter finden Sie im Anhang B.

Workflowübersicht

Für Sie sind die folgenden Workflowaufgaben aktiv:

>

Name	Anwender/ Gruppe	Startdatum	Lieferant
$$nodeName$$	$$userName$$	$$activateDate$$	$$ixkey_0$$

Sie können diese Aufgaben im ELO Client bearbeiten.

Das Template muss eine Tabelle enthalten, welche für jede Aufgabe eine Zeile bekommt. Deshalb müssen also nicht nur die variablen Texte sondern auch der Bereich einer kompletten Aufgabenzeile markiert werden. Ein Auszug aus dem Template für die Anzeige von oben sieht so aus:

```
<body>
  <h1>Workflowübersicht</h1>
  Für Sie sind die folgenden Workflowaufgaben aktiv:
  <table border="0">
  <tr><td class="header">Name</td>
      <td class="header">Anwender/ Gruppe</td>
      <td class="header">Startdatum</td>
      <td class="header">Lieferant</td>>
  </tr>
  <!--ListStart-->
  <tr><td class="$$className$$">$$nodeName$$</td>
      <td class="$$className$$">$$userName$$</td>
      <td class="$$className$$">$$activateDate$$</td>
      <td class="$$className$$">$$ixkey_0$$</td>
  </tr>
  <!--ListEnd-->
  </table>
  Sie können diese Aufgaben im ELO Client bearbeiten.
</body>
```

Den vollständigen Dateiinhalt finden Sie im Anhang A.

Beim Mailversand läuft das Notify Modul nun über alle Aufgaben des Anwenders und erzeugt für jeden Termin eine Kopie des Abschnitts zwischen <!—ListStart--> und <!—

ListEnd-->. In dieser Kopie werden dann die Platzhalter, wie z.B. $$userName$$, gegen die entsprechenden Werte des Termins ausgetauscht.

4 Mail Benachrichtigung als Workflow Knoten

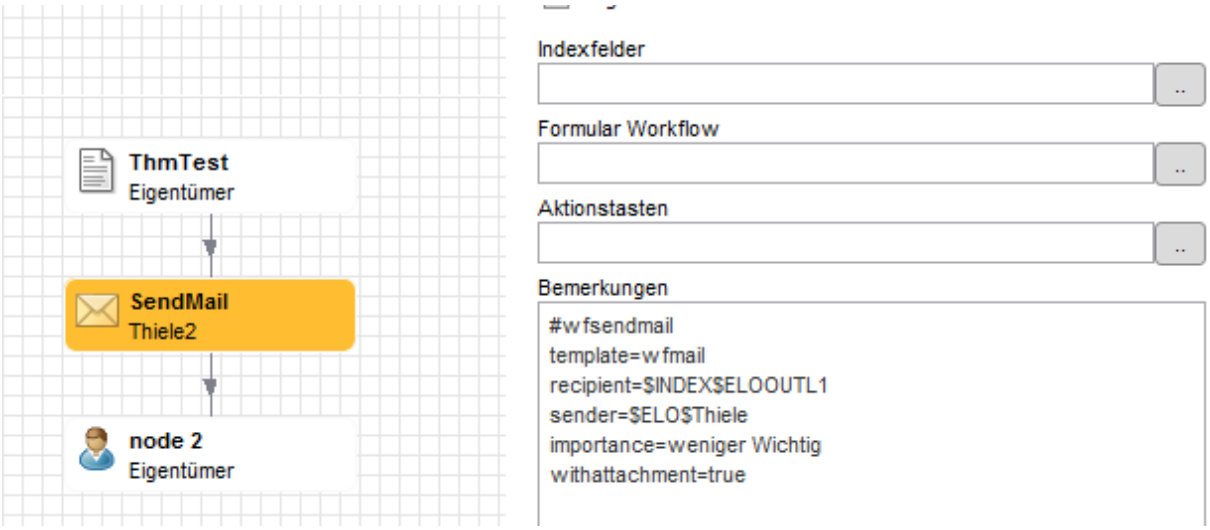
Ab der Version 9 erlaubt die Notify Library auch den Versand von Emails als Workflow Knoten, ohne dass zusätzliche Skripte nötig sind. Es muss dafür lediglich eine Rule zur regelmäßigen Kontrolle der Workflowtermine des ELOas Anwenders hinterlegt werden.

```
<ruleset>
  <base>
    <name>SendWfMail</name>
    <search>
      <name>"WORKFLOW"</name>
      <value>"1"</value>
      <mask>0</mask>
      <max>200</max>
    </search>
    <interval>1M</interval>
    <onstart>
      EM_ALLOWALLMASKS = true;
    </onstart>
  </base>
<rule>
  <name>Send</name>
  <condition></condition>
  <script>
    mail.setSmtpHost("MyMailServer");
    notify.checkAddFeed();
    notify.checkSendMail();
  </script>
</rule>
<rule>
  <name>Global Error Rule</name>
  <condition>OnError</condition>
  <script></script>
</rule>
</ruleset>
```

Im Beispiel läuft die Rule jede Minute, es kann aber auch ein größeres Intervall gewählt werden. Zudem setzt es die Variable EM_ALLOWALLMASKS auf true damit es für alle Workflows verwendet werden kann und nicht auf eine Verschlagwortungsmaske beschränkt ist. Und auch hier sollte man die Initialisierung des Email Servers beachten.

Die Rule ruft für jeden Termin des ELOas Anwenders die Notify-Funktionen checkAddFeed und checkSendMail auf. Diese Funktionen prüfen, ob es sich bei dem Knoten um einen Mail-Versand oder Feed-Kommentar Knoten handelt. So ein Knoten ist ein normaler

Personenknoten welcher sich an den ELOas Anwender richtet und in der Bemerkung die Parameter für den Mail Versand und/ oder Feed Kommentar enthält.




```
#wfsendmail
template=wfmail
recipient=$INDEX$ELOOUTL1
sender=$ELO$Thiele
importance=weniger Wichtig
withattachment=true
```

Das Bemerkungsfeld muss für den Mailversand mit der Kennung #wfsendmail oder #wfmailandfeed in der ersten Zeile beginnen. Daran erkennt das Notify Modul, dass es sich um einen Mail Knoten handelt. Für den Feed-Kommentar muss es mit #wfaddfeed oder #wfmailandfeed anfangen. Danach folgen teilweise optionale Parameter, die jeweils in der name=wert Notation geschrieben sind.

Name	Wert	Optional
template	Enthält den Namen des HTML Mail Templates im ELOas Base\Misc Ordner. Dieses Template wird als HTML Vorlage für den Mailtext verwendet. Es kann die im Anhang B aufgelisteten Variablen enthalten. Wenn kein Name angegeben wurde, wird das Template mit dem Namen wfmail verwendet.	Ja
recipient	Empfänger der Email. Der Empfänger kann direkt als Mailadresse eingetragen werden: max.mustermann@musterfirma.de	Nein

	Er kann aus einer Indexzeile ausgelesen werden. In diesem Fall muss der Gruppenname der Indexzeile, mit einer $INDEX$ Sequenz angeführt, eingegeben werden: $INDEX$ELOOUTL1 Er kann aus dem Email Feld der ELO Anwenderdaten ausgelesen werden. In diesem Fall wird das Präfix ELO vorangestellt: ELOMustermann Weiterhin kann auch der ELO Username des Vorgängerknotens verwendet werden. In diesem Fall darf es aber auch nur genau einen Vorgänger geben und dieser muss ein Personenknoten sein. Als recipient wird dann ein $PARENT$ angegeben.	
sender	Absender der Mail, wird für die ReplyTo Funktion verwendet. Wenn kein sender Parameter eingetragen wird, bleibt das Mail Feld leer.	Ja
subject	Betreff der Mail. Wenn hier nichts eingetragen wird, wird die Kurzbezeichnung der Verschlagwortung als Betreff verwendet.	Ja
withattachment	true oder false – gibt an, ob die Dokumentendatei als Attachment an die Mail gehangen werden soll. Darf nur für Dokumente und nicht für Ordner verwendet werden.	Ja
*	Es können weitere Parameter gesetzt werden, die im HTML Template verwendet werden. Im Beispiel oben wird der Parameter „importance“ mit dem Wert „weniger Wichtig“ definiert. Wenn das HTML Template einen Platzhalter $$param.importance$$ enthält, wird dieser vor dem Mailversand durch den Text „weniger Wichtig“ ersetzt.	Ja

5 Skript Callback Funktionen

Prinzipiell können Sie die Notify Library beliebig an eigene Anforderungen anpassen, da diese im Source Code als Skriptdatei vorliegt. Solche Anpassungen haben aber den Nachteil, dass diese bei jedem Update der JavaScript Library verloren gehen. Aus diesem Grund gibt

es an Stellen die häufig angepasst werden müssen eine Callback Funktion. Diese tragen Sie in einem eigenständigen Modul „NotifyCallback“ ein. Da dieses Modul keinen ELO Funktionscode enthält, ist es von Updates nicht betroffen und Sie können Ihre eigene Version beibehalten.

Alle Callback Funktionen müssen in ein Objekt mit dem Namen „notifyCallback“ eingetragen werden. Sie sollten diese in einem Lib Modul „notifyCallback“ zusammenfassen.

```
function NotifyCallback() {
}

var notifyCallback = new NotifyCallback();

NotifyCallback.prototype.filterTask = function(task) {
  return true;
}

NotifyCallback.prototype.beforeSend = function(text) {
  return text;
}

NotifyCallback.prototype.getTableLine = function(task) {
  return null;
}

NotifyCallback.prototype.getMailUser = function(userName) {
  return null;
}

NotifyCallback.prototype.formatMessage = function(template, node, sord,
properties) {
  return null;
}

NotifyCallback.prototype.formatFeedMessage = function(template, node, sord,
properties) {
  return null;
}

NotifyCallback.prototype.getSubject = function(node, sord, properties) {
  return null;
}
```

5.1 Aufgabenliste filtern

In der E-Mail sollen nicht immer alle Termine aufgeführt werden. Unwichtige Termine sollen weggelassen werden. Hierfür gibt es eine Callback Funktion mit dem Namen filterTask. Diese wird für jeden Termin aufgerufen – wenn sie true zurückgibt, wird der Termin in die Mail aufgenommen, gibt sie false zurück, wird er nicht aufgenommen. Diese Funktion wird beim Sammelversand der Aufgabenübersicht für jeden einzelnen Termin aufgerufen. Als Parameter wird das jeweilige Aufgabenobjekt übergeben.

5.2 Vor dem Senden

Die Funktion beforeSend wird vor dem Senden einer Mail beim Sammelversand aufgerufen. Sie erhält als Parameter den Mail-Body und gibt den veränderten (oder unveränderten) Text als Returnwert zurück.

5.3 Füllen einer Aufgabenzeile

Die Funktion getTableLine erlaubt es Ihnen für eine Aufgabe einen eigenen HTML Text zurückzugeben. Wenn Sie hier im Returnwert einen Text ausgeben, wird dieser verwendet. Geben Sie einen null Wert zurück, wird der Standardtext aus dem Template verwendet.

5.4 Ermitteln der Mailadresse

Wenn die Standardwege zur Ermittlung des Empfängers aus einer direkten Mailadresse, aus einem Indexfeld oder aus den ELO Anwenderdaten nicht ausreichen, können Sie hier eine eigene Auswertungsroutine erstellen. Ein null Returnwert verwendet den Standard, ein Returnwert ungleich null wird als Absender- oder Empfängeradresse verwendet.

5.5 Eigenen Mailkörper erstellen

Die Callback Funktion formatMessage erlaubt es Ihnen einen komplett eigenen Mailkörper per Skript zu erstellen. Als Parameter erhalten Sie das eingestellte Template, den aktuellen Workflowknoten (WFCollectNode), die aktuelle Verschlagwortung (Sord) und die Parameter aus dem Workflowknoten (Properties). Ein null Returnwert verwendet die Standardauflösung des Mailkörpers aus dem Template und der Verschlagwortung.

5.6 Feed-Kommentar formatieren

Die Callback Funktion formatFeedMessage erlaubt es Ihnen den Kommentar per Skript zu erstellen. Als Parameter erhalten Sie das eingestellte Feed-Template, den aktuellen Workflowknoten, die Verschlagwortung und die Parameter. Ein null Returnwert verwendet die Standardauflösung des Feed-Kommentars aus dem Template und der Verschlagwortung.

5.7 Kurzbezeichnung ermitteln

Normalerweise wird die Kurzbezeichnung des Verschlagwortungsobjekts als Betreff für die zu versendende Mail verwendet. Optional können Sie auch einen Betreff in den Parametern

des Workflowknotens festlegen. Wenn diese beiden Möglichkeiten nicht ausreichen, können Sie per Skript über diese Callback Funktion eine beliebige Kurzbezeichnung erzeugen. Als Parameter stehen der Funktion der aktuelle Workflowknoten (WFCollectNode), die Verschlagwortung (Sord) und die Parameter (Properties) zur Verfügung.

6 Weitere html Optionen für den Sammelversand

Wenn man unterschiedliche Workflows mit unterschiedlichen Verschlagwortungsmasken hat, möchte man in der Mail oft auch unterschiedliche Informationen zum jeweiligen Termin mitteilen. Aus diesem Grund kann man den Teil der html Templates, der für die Terminausgabe verwendet wird, auch Maskenabhängig hinterlegen. Hierfür legen Sie im „Misc" Ordner weitere Dokumente mit dem Namen „wfreminder_<Maskenname>" an. Für die Maske „Rechnung" also „wfreminder_Rechnung". Dieses Dokument darf dann nur den Teil zwischen <!--ListStart--> und <!--ListEnd--> enthalten.

Das Feed-Template wird im Prinzip genauso behandelt. Nur darf es keinen HTML Text enthalten.

Beispiel:

```
<tr>
  <td class="$$className$$">$$nodeName$$</td>
  <td class="$$className$$">$$userName$$</td>
  <td class="$$className$$">$$activateDate$$</td>
  <td class="$$className$$">$$ixkey_0$$</td>
  <td class="$$className$$">$$ixgroup_LIEFERNR$$</td>
</tr>
```

Wenn Sie unterschiedliche Ausgaben in Abhängigkeit von der Verschlagwortung oder anderen Parametern benötigen, können Sie die Formatierung auch per Skript Callback Funktion bestimmen.

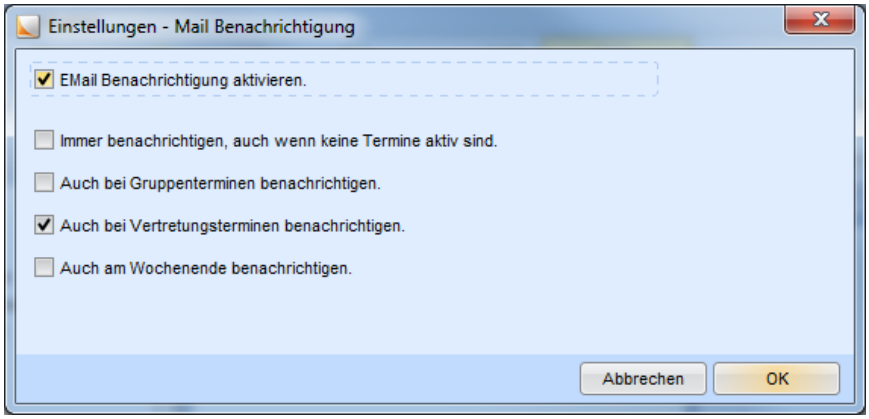
```
NotifyCallback.prototype.getTableLine = function(task) {
  if (task.wfNode && (task.wfNode.nodeName == "Test3")) {
    return '<tr><td class="$$className$$"
colspan="5"><p><b>$$nodeName$$</b></p><p>$$userName$$ at
$$activateDate$$</p></td></tr>';
  }

  return null;
}
```

7 Einstellung der Anwenderoptionen

Die anwenderbezogenen Einstellungen für den Sammelversand werden in der Profileopts Tabelle hinterlegt. Hier liegen alle Optionseinstellungen für Anwender, Optionengruppen und auch die globalen Einstellungen.

Im Java Client kann man einen Button einbinden, der dem Anwender einen Dialog zur Einstellung seiner Benachrichtigungen einblendet.




```
// JavaScript Dokument


function getScriptButton50Name() {
  return "Mail Benachrichtigung";
}

function getScriptButtonPositions() {
  return "50,home,view";
}

function eloScriptButton50Start(){
  var actOpt = archive.getUserOption("ELOas.SendWfAsMail", "");
  actOpt = selectOptions(actOpt);
  if (actOpt >= 0) {
    archive.setUserOption("ELOas.SendWfAsMail", actOpt);
  }
}

function selectOptions(actOptions) {
  var dlg = workspace.createGridDialog("Einstellungen - Mail
Benachrichtigung", 2, 6);
  var panel = dlg.gridPanel;

  actOptions = Number(actOptions);

  var ckMail = panel.addCheckBox(1, 1, 1, "EMail Benachrichtigung
aktivieren.", (actOptions & 1) != 0);
```

```
  var ckAlways = panel.addCheckBox(1, 3, 1, "Immer benachrichtigen, auch
wenn keine Termine aktiv sind.", (actOptions & 2) != 0);
  var ckGroup = panel.addCheckBox(1, 4, 1, "Auch bei Gruppenterminen
benachrichtigen.", (actOptions & 4) != 0);
  var ckDeputy = panel.addCheckBox(1, 5, 1, "Auch bei Vertretungsterminen
benachrichtigen.", (actOptions & 8) != 0);
  var ckWeekend = panel.addCheckBox(1, 6, 1, "Auch am Wochenende
benachrichtigen.", (actOptions & 16) != 0);

  var result = -1;
  if (dlg.show()) {
    result = 0;

    if (ckMail.isChecked()) {
      result |= 1;
    }

    if (ckAlways.isChecked()) {
      result |= 2;
    }

    if (ckGroup.isChecked()) {
      result |= 4;
    }

    if (ckDeputy.isChecked()) {
      result |= 8;
    }

    if (ckWeekend.isChecked()) {
      result |= 16;
    }
  }

  return result;
}
```

8 Anhang A – ein Muster für den Mailversand

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Workflowübersicht</title>
    <style type="text/css" >
      body {
        font-family: Verdana,Arial;
```

```
        font-size: 14px;
      }

      table {
        margin-top: 20px;
        margin-bottom: 20px;
        border: 1px silver solid;
        border-collapse: collapse;
      }

      td {
        border-bottom:1px silver dotted;
        padding: 5px;
      }

      .header {
        background-color:#f0f2ff;
      }

      .urgent {
        background-color:#ffd0d0;
      }

      .group {
        background-color:#d0ffd0;
      }

    </style>
  </head>
  <body>
    <h1>Workflowübersicht</h1>
    Für Sie sind die folgenden Workflowaufgaben aktiv:
    <table border="0">
    <tr><td class="header">Name</td><td class="header">Anwender/
Gruppe</td>
                                <td class="header">Startdatum</td><td
class="header">Lieferant</td>></tr>
    <!--ListStart-->
    <tr><td class="$$className$$">$$nodeName$$</td><td
class="$$className$$">$$userName$$</td><td
class="$$className$$">$$activateDate$$</td>
    <td class="$$className$$">$$ixkey_0$$</td></tr>
    <!--ListEnd-->
    </table>
    Sie können diese Aufgaben im ELO Client bearbeiten.
  </body>
</html>
```

9 Anhang B – Verfügbare Variablen im Mail und Feed Template

Erlaubte Platzhalter (jeweils in $$ am Anfang und Ende einfügen):

activateDate	Datum und Uhrzeit wann der Workflow Termin zugestellt wurde.
className	CSS Klasse für die Formatierung der Termindaten: normal, group, urgent.
flowName	Name des Workflows
flowStatus	Statusfeld des Workflows
ixgroup_	Indexzeilenfeld aus der Verschlagwortung zum Workflow. Dieser Platzhalter muss um den Gruppennamen ergänzt werden, z.B. für die Indexzeile RENR wird $$ixgroup_RENR$$ verwendet. Falls dieses Feld im aktuellen Workflow nicht vorhanden ist, wird ein Leerstring verwendet, es wird keine Fehlermeldung angezeigt.
ixkey_	Indexzeilenfeld aus der Verschlagwortung zum Workflow. Dieser Platzhalter muss um die Indexzeilennummer ergänzt werden, z.B. für die Indexzeile 0 wird $$ixkey_0$$ verwendet. Falls dieses Feld im aktuellen Workflow nicht vorhanden ist, wird ein Leerstring verwendet, es wird keine Fehlermeldung angezeigt.
maskName	Name der Verschlagwortungsmaske zum Termin.
nodeName	Knotenname des aktuellen Workflowtermins
objName	Kurzbezeichnung des verbundenen Ordners oder Dokuments.
timeLimit	Eskalationsdatum für den aktuellen Termin
userName	Name des Knoteneigentümers. Diese Angabe ist insbesondere für Gruppen- oder Vertretungstermine sinnvoll.