CMS-Schnittstellen

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Single-Sign-On (SSO)

1.1. Schritt 1: Login

Das Single-Sign-On beschreibt eine Schnittstelle, um einen Benutzer des CMS direkt in etracker einzuloggen und eine gültige Session ID[1] zurückzugeben. Für die Umsetzung zum direkten Login ohne Benutzereingabe gibt es zwei Schritte, wobei der zweite Schritt im Falle einer Clientseitigen[2] Umsetzung wegfallen kann.
Der erste Teil ist der Login über die Schnittstelle application.etracker.com/login.php. Diese muss mittels POST-Request[3] mit den Parametern username, password und cmsLogin angesprochen werden. Die Werte für username und password entsprechen den normalen Login-Parametern, der Parameter cmsLogin muss zwingend den Wert CMS enthalten, da ansonsten ein normaler Login durchgeführt wird.

1.1.1. Schnittstelle

Protokoll	Domain	Schnittstelle
https	application.etracker.com	login.php

1.1.2. Parameter

Parameter	Beschreibung	Inhalt
username	Name des Accounts	String
password	Passwort des Accounts	String
cmsLogin	Sicherheitsparameter	CMS

1.1.3. Rückgabewerte

Rückgabewert	Beschreibung	Inhalt
Session ID	Eine gültige etracker Session	String

[1] http://de.wikipedia.org/wiki/Session-ID
[2] http://de.wikipedia.org/wiki/Client-Server
[3] http://de.wikipedia.org/wiki/Hypertext_Transfer_Protocol

1.1.4. Beispiel eines Serverseitigen POST-Requests mit PHP

```php
/**
 * \brief Sends a post request to specified url
 *
 * \param    string $url         \brief send to this url
 * \param    string $data \brief the data to send
 * \return   string
 */
function postRequest($url, $data)
{
    $params = array
    (
       'http' => array
       (
            'method' => 'POST',
            'content' => $data
       )
    );

    $ctx = stream_context_create($params);
    $fp = @fopen($url, 'rb', false, $ctx);

    $response = @stream_get_contents($fp);

    return $response;
}

$username = '';
$password = '';

$loginInterface = 'https://application.etracker.com/login.php';

$data  = 'username=' . $username;
$data .= '&password=' . $password;
$data .= '&cmsLogin=CMS';

$session = postRequest($loginInterface, $data);

echo $session;
```

Abbildung 1: Beispiel eines Serverseitigen POST-Requests mit PHP

1.2. Schritt 2: Sicherheitscheck

Wenn der Login über diese Schnittstelle serverseitig durchgeführt wird, ist es zwingend erforderlich anschließend den zweiten Schritt durchzuführen. Der Client erhält darüber ein gültiges Sicherheitscookie[4]. Zusätzlich müssen weitere Sicherheitseinstellungen durchgeführt werden um ein Sessionhijacking[5] zu erschweren. Dazu muss die Schnittstelle application.etracker.com/cms_sc.php mit der aus dem ersten Schritt erhaltenen SessionId vom Client aufgerufen werden. Dies muss bis spätestens 10 Sekunden nach dem 1. Schritt geschehen, ein späterer Versuch wird abgewiesen. Wenn Sie die erste Schnittstelle Clientseitig durchführen, werden die Sicherheitseinstellungen schon durchgeführt inklusive des Sicherheitscookies. In diesem Fall muss die Schnittstelle application.etracker.com/cms_sc.php nicht aufgerufen werden.

[4] http://de.wikipedia.org/wiki/HTTP-Cookie
[5] http://de.wikipedia.org/wiki/Session_Hijacking

Die Schnittstelle application.etracker.com/cms_sc.php können Sie auf für ein Keep-Alive[6] der Session verwenden. Wenn die Sicherheitsüberprüfung erfolgreich war, aktualisiert jeder Folgeaufruf die Session.

1.2.1. Schnittstelle

Protokoll	Domain	Schnittstelle
https	application.etracker.com	cms_sc.php

1.2.2. Parameter

Parameter	Beschreibung	Inhalt
sid	Session ID	Ergebnis der Schnittstelle login.php

1.2.3. Rückgabewerte

Rückgabewert	Beschreibung	Inhalt
Cookie	Sicherheitscookie für den Clientbrowser zur Absicherung der Session	String
OK	Die übergebene Session ist gültig	OK
FAIL	Die übergebene Session ist nicht mehr gültig	FAIL

1.2.4. Beispiel eines Aufrufes mit einem iframe in PHP

Der Aufruf des Schnittstelle muss auf dem Client erfolgen, in diesem Fall wahrscheinlich ein Browser. Eine mögliche Verwendung ist ein Iframe[7], welcher den Aufruf enthält.

```
$session = postRequest($interface, $data);
?>
<iframe src="http://eleven/cms_sc.php?sid=<?=$session?>"></iframe>
```

Abbildung 2: Beispiel eines Aufrufes mit einem iframe in PHP

[6] http://de.wikipedia.org/wiki/Keepalive
[7] http://de.wikipedia.org/wiki/Iframe

2. Website-Overlays

2.1. Schnittstelle zur Seitenidentifikation

Für die Anzeige des Overlays wird die interne ID der jeweiligen Seite benötigt. Um diese ID von etracker zu erhalten, müssen Sie die Webservice-Schnittstelle[8] verwenden. Dieses Protokoll ermöglicht die Kommunikation über XML-Dokumente[9] und ist zudem plattformunabhängig.
Um die Schnittstelle verwenden zu können, benötigen Sie ein Entwickler Token. Dieses können Sie bei etracker beantragen. Damit sind Sie auch in der Lage, alle weiteren Funktionen der Webservice-Schnittstelle zu verwenden.

Der Aufruf zur Seitenidentifikation ist so aufgebaut, dass sie eine oder mehrere Seiten gleichzeitig abfragen können. Das folgende XML-Beispieldokument zeigt den Aufbau einer Anfrage. Zwingend notwendig ist der Header, der die Anmeldedaten enthält. Im Body sind dann die eigentlichen Abfragen beschrieben.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v2="http://ws.etracker.com/api/ws/v2">
   <soapenv:Header>
      <v2:accountId>ACCOUNT NAME</v2:accountId>
      <v2:developerToken>ENTWICKLER TOKEN</v2:developerToken>
      <v2:password>PASSWORT</v2:password>
      <v2:email>E-MAIL ADRESSE</v2:email>
   </soapenv:Header>
   <soapenv:Body>
      <v2:getPageId>
         <pagename>Seitenname 1</pagename>
         <pagename>Seitenname 2</pagename>
      </v2:getPageId>
   </soapenv:Body>
</soapenv:Envelope>
```

Abbildung 3: XML- Beispieldokument einer Anfrage

Das als Envelope bezeichnete Dokument muss an die Schnittstelle http://ws.etracker.com/api/ws/v2/CMSService?wsdl gesendet werden.

[8] http://de.wikipedia.org/wiki/Webservice
[9] http://de.wikipedia.org/wiki/XML

Das Resultat ist wiederum ein XML-Dokument.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns1="http://ws.etracker.com/api/ws/v2">
   <SOAP-ENV:Header>
      <ns1:requestId xsi:type="xsd:string">884</ns1:requestId>
      <ns1:operations xsi:type="xsd:long">1</ns1:operations>
      <ns1:responseTime xsi:type="xsd:long">2</ns1:responseTime>
   </SOAP-ENV:Header>
   <SOAP-ENV:Body>
      <ns1:getPageIdResponse>
         <pageIdRows>
            <pagename>Seitenname 1</pagename>
            <pageId>1</pageId>
         </pageIdRows>
         <pageIdRows>
            <pagename>Seitenname 2</pagename>
            <pageId>2</pageId>
         </pageIdRows>
      </ns1:getPageIdResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Abbildung 4: Beispielergebnis der Webserviceschnittstelle

Das Ergebnis enthält wieder einen Header mit Informationen und einen Body mit den eigentlichen Ergebnissen. Sollte ein Seitenname nicht existieren, wird keine ID zurück geliefert. Wird nur ein Seitenname abgefragt, erhalten Sie als Ergebnis ein Objekt mit Seitenname und Page ID. Wird gleichzeitig nach mehreren Seiten gefragt, ist das Ergebnis ein Array von Objekten. Darauf ist in der Anbindung zu achten.

2.1.1. Schnittstelle

Protokoll	Domain	Schnittstelle
http	ws.etracker.com	api/ws/v2/CMSService?wsdl

2.1.2. Parameter

Parameter	Beschreibung	Inhalt
accountId	Name des Accounts	String
password	Passwort des Accounts	String
developerToken	Das Entwickler Token	Das Token erhalten Sie von etracker
email	Die E-Mail-Adresse des Entwicklers	Die E-Mail-Adresse, mit der das Entwickler Token beantragt wurde
Pagename	Ein oder mehrere Seitennamen	Array von Strings

2.1.3. Rückgabewerte

Rückgabewert	Beschreibung	Inhalt
pageIdRows	Ein Objekt mit einem Tupel aus pagename und pageId oder ein Array von Objekten, wenn mehrere Seiten abgefragt wurden.	String

2.1.4. Beispiele

Für die Implementierung in ein CMS kann ein bestehender SOAP-Client[10] verwendet werden, der die XML-Dokumente erzeugt und die Ergebnisse in die für die verwendete Programmiersprache verwendbare Objekte umwandelt.

Auch diese Beispiele sind in PHP verfasst, als SOAP Client wurde die SoapClient Klasse[11] verwendet. Diese steht zur Verfügung, wenn PHP mit der Option –enable-soap konfiguriert wurde.

Um die Rückgabewerte in eigene Objekte umzuwandeln benötigt der SOAP-Client eine Mapping-Tabelle. Die Ergebnisse sind dann in diesen im PHP-Code definierten Objekten enthalten.

```
// soap server
$wsdl = 'http://ws.etracker.com/api/ws/v2/CMSService?wsdl';

// authentication
$email             = ''; // geben Sie bitte hier ihre E-Mailadresse an
$developerToken    = ''; // geben Sie bitte hier ihr developer token an
$accountId         = ''; // geben Sie bitte hier ihre Account Id an
$password          = ''; // geben Sie bitte hier das Account Passwort an

// init soap environment
ini_set('display_errors', 'on');
ini_set('soap.wsdl_cache_enabled', 0);
$namepace = 'http://ws.etracker.com/api/ws/v2';

// init soap client
$client = new SoapClient(
    $wsdl,
    array(
        'classmap' => array(
            // webservice type => php type
             'getPageId'         => 'PageId',
            'getPageIdResponse' => 'GetPageIdResponse',
             'PageIdResponseRow' => 'PageIdResponseRow'
        ),
    )
);

// add request headers
$headers = array();
$headers[] = new SoapHeader($namepace, 'accountId', $accountId);
$headers[] = new SoapHeader($namepace, 'developerToken', $developerToken);
$headers[] = new SoapHeader($namepace, 'email', $email);
$headers[] = new SoapHeader($namepace, 'password', $password);

$client->__setSoapHeaders($headers);

// call method
```

[10] http://de.wikipedia.org/wiki/SOAP#Implementierungen
[11] http://de2.php.net/manual/de/book.soap.php

```php
header('Content-Type: text/html; charset=utf8');
try
{
    // build request
    $request = new PageId();
    $request->pagename = array(
        utf8_encode("Seitenname 1"),
        utf8_encode("Seitenname 2")
    );
    $pageIds = $client->getPageId($request);
}
catch (Exception $e)
{
    echo '<pre>';
    print_r($e);
    echo '</pre>';
    exit();
}

/*
 * Declaration of mapping classes
 */

/**
 * \brief class for request of page ids
 */
class PageId
{
       /**
        * array \a $pagename
        */
    public $pagename;
}

/**
 * \brief container of page id rows
 */
class GetPageIdResponse
{
       /**
        * array \a $pageIdRows
        */
    public $pageIdRows;
}

/**
 * \brief contains a tuple of pagename and page id
 */
class PageIdResponseRow
{
       /**
        * string \a $pagename
        */
       public $pagename;

       /**
        * string \a $pageId
        */
       public $pageId;
}
```

Abbildung 5: Beispiel einer Webservice Abfrage in PHP

2.2. Anzeige des Overlays

Um ein Overlay zu starten wird lediglich ein Anchor-Tag[12] benötigt, der die entsprechende Schnittstelle mit den zuvor gesammelten Parametern aufruft.

2.2.1. Schnittstelle

Protokoll	Domain	Schnittstelle
https	application.etracker.com	roverlay.php

2.2.2. Parameter

Parameter	Beschreibung	Inhalt
sid	Session ID	Ergebnis der Schnittstelle login.php
pid	Seiten ID	Ergebnis der Webservice-Schnittstelle
overlay	Auswahl des Overlays	heatmap oder overlay
overlayStartDate	optionales Startdatum	Datum nach ISO 8601 (YYYY-MM-DD)
overlayEndDate	optionales Enddatum	Datum nach ISO 8601 (YYYY-MM-DD)

2.2.3. Rückgabewerte

Rückgabewert	Beschreibung	Inhalt
HTML-Seite	Die abgefragte Seite wird mit dem Overlay aufgerufen	HTML-Dokument

[12] http://de.wikipedia.org/wiki/Hyperlink

2.2.4. Beispiel

Das Beispiel enthält zwei Links für die Heatmap und Clickmap mit eingestelltem Datum und zwei Links ohne Datum. Die Session entstammt dem Single-Sign-On, die Page ID der Webservice-Schnittstelle.

```
<!-- mit Datum -->

<a href="https://application.etracker.com/roverlay.php?sid=<?=$session?>
&overlayStartDate=2009-05-20
&overlayEndDate=2009-06-02
&pid=<?=$pageIds->pageIdRows->pageId;?>
&overlay=heatmap" target="_blank">heatmap</a>

<a href="https://application.etracker.com/roverlay.php?sid=<?=$session?>
&overlayStartDate=2009-05-20
&overlayEndDate=2009-06-02
&pid=<?=$pageIds->pageIdRows->pageId;?>
&overlay=clickmap" target="_blank">clickmap</a>

<!-- ohne Datum -->

<a href="https://application.etracker.com/roverlay.php?sid=<?=$session?>
&pid=<?=$pageIds->pageIdRows->pageId;?>
&overlay=heatmap" target="_blank">heatmap</a>

<a href="https://application.etracker.com/roverlay.php?sid=<?=$session?>
&pid=<?=$pageIds->pageIdRows->pageId;?>
&overlay=clickmap" target="_blank">clickmap</a>
```

Abbildung 6: Beispiel des Overlay-Aufrufes mit PHP

3. Komplettbeispiel in PHP

Das folgende Beispiel enthält alle vorher definierten Funktionen und Klassen.

```php
<?
/**
* CMS Test
*
* Copyright (c) 2000 - 2009 etracker GmbH. All Rights Reserved
*/

/**
 * \brief Sends a post request to specified url
 *
 * \param    string $url         \brief send to this url
 * \param    string $data \brief the data to send
 * \return   string
 */
function postRequest($url, $data)
{
    $params = array
    (
       'http' => array
       (
             'method' => 'POST',
           'content' => $data
        )
       );

    $ctx = stream_context_create($params);
    $fp = @fopen($url, 'rb', false, $ctx);

    $response = @stream_get_contents($fp);

    return $response;
}

/**
 * Declaration of mapping classes
 */

/**
 * \brief class for request of page ids
 */
class PageId
{
       /**
        * array \a $pagename
        */
    public $pagename;
}

/**
 * \brief container of page id rows
 */
class GetPageIdResponse
{
       /**
        * array \a $pageIdRows
        */
    public $pageIdRows;
}
```

```php
/**
 * \brief contains a tuple of pagename and page id
 */
class PageIdResponseRow
{
	/**
	 * string \a $pagename
	 */
	public $pagename;

	/**
	 * string \a $pageId
	 */
	public $pageId;
}

// Initializing
$wsdl = 'http://ws.etracker.com/api/ws/v2/CMSService?wsdl';


// authentication
$email             = ''; // geben Sie bitte hier ihre E-Mailadresse an
$developerToken    = ''; // geben Sie bitte hier ihr developer token an
$accountId         = ''; // geben Sie bitte hier ihre Account Id an
$password          = ''; // geben Sie bitte hier das Account Passwort an


// init soap environment
ini_set('display_errors', 'on');
ini_set('soap.wsdl_cache_enabled', 0);
$namepace = 'http://ws.etracker.com/api/ws/v2';


$loginInterface = 'https://application.etracker.com/login.php';
$data  = 'username=' . $accountId;
$data .= '&password=' . $password;
$data .= '&cmsLogin=CMS';

$x = postRequest($loginInterface, $data);


// init soap client
$client = new SoapClient(
    $wsdl,
    array(
        'classmap' => array(
            // webservice tyoe => php type
             'getPageId'         => 'PageId',
             'getPageIdResponse' => 'GetPageIdResponse',
             'PageIdResponseRow' => 'PageIdResponseRow'
        ),
    )
);


// add request headers
$headers = array();
$headers[] = new SoapHeader($namepace, 'accountId', $accountId);
$headers[] = new SoapHeader($namepace, 'developerToken', $developerToken);
$headers[] = new SoapHeader($namepace, 'email', $email);
$headers[] = new SoapHeader($namepace, 'password', $password);

$client->__setSoapHeaders($headers);
```

```php
// call method
header('Content-Type: text/html; charset=utf8');
try
{
    // build request
    $request = new PageId();
    $request->pagename = array(utf8_encode("Seitenname 1"));
    $pageIds = $client->getPageId($request);
}
catch (Exception $e)
{
    echo '<pre>';
    print_r($e);
    echo '</pre>';
    exit();
}

?>
<!-- Getting security cookie -->
<iframe style="display:none"
src="https://application.etracker.com/cms_sc.php?sid=<?=$x?>"></iframe>
<br><br>

<!-- Building links to overlay -->
<a href="https://application.etracker.com/roverlay.php?sid=<?=$x?>
&pid=<?=$pageIds->pageIdRows->pageId;?>&overlay=heatmap"
target="_blank">heatmap</a>
<br><br>

<a href="https://application.etracker.com/roverlay.php?sid=<?=$x?>
&pid=<?=$pageIds->pageIdRows->pageId;?>&overlay=clickmap"
target="_blank">clickmap</a>
```

Abbildung 7: Komplettbeispiel eines Aufrufes in PHP