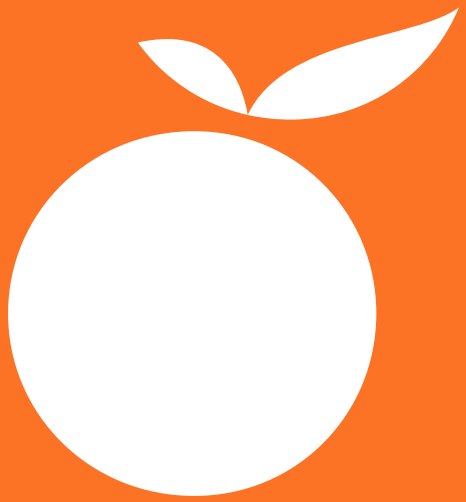
tt performance suite

# Configuration Guidelines



CUSTOMIZING WEB CONTEXT RECOGNITION

Imprint

Customizing web context recognition
29. February 2016 – Version 2.00

# Content

# 1 Introduction

tt performance suite allows you to customize the context recognition for web applications. This can be used to fine-tune context detection at recording time, as well as to tailor the context-oriented delivery of content to the end-users (via QuickAccess).

## 1.1 Overview

Customization is done using a configuration file in xml format. This file is used by both tt knowledge force (recording of simulations and documentation) and tt guide (recording of Guides, delivery of content via QuickAccess). The configuration file is distributed and updated via the central tt performance suite server.
The tt guide client checks for an update of the configuration file on every application start and downloads it automatically if needed.
The tt knowledge force client checks for updates on every server login and also downloads the file automatically if a newer version is available (no restart required).
For more information regarding the creation and distribution of the configuration file please refer to the documentation “Configuration distribution” which is available at the same location as this document.

# 2 Configuration

Currently, there are three different ways of identifying a web application and its individual contexts:

- Based on its URL, or parts of the URL
- Based on its page title, or parts of the page title
- Based on parts of the HTML source code.

## 2.1 Configuration file

Example

```
<context>
      <id>identifier</id>
      <uiname>plain name</uiname>
      <match_patterns>
            <pattern>
                  <where></where>
                  <matches></matches>
            </pattern>
      </match_patterns>
</context>
```

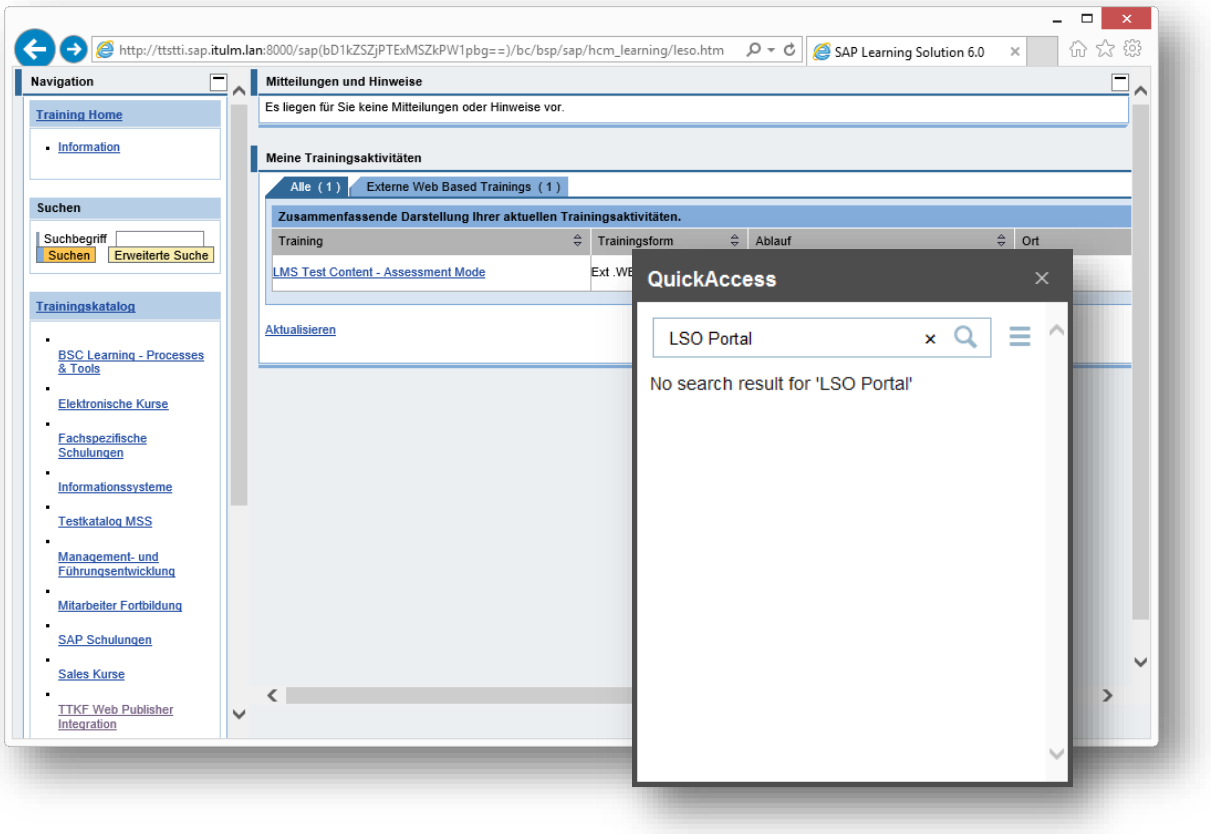
Each application-specific context configuration is represented by a `<context>` node within the context configuration file. This node contains at least three child nodes: `<id>`, `<uiname>`, and `<match_patterns>`. The `<id>` is a technical identifier which can be arbitrarily chosen but must not contain any blanks or special characters besides the period (“.”), the underscore (“_”), and the hyphen “-”. The `<uiname>` represents the application name as it is stored in the context field during recording of Guides or TT documents; this name is also used as a mandatory search term within QuickAccess. The `<match_patterns>` define how the context is determined.

## 2.2 Match Patterns

The following example illustrates a basic context definition for the learning portal of the SAP Learning Solution (LSO), using one simple context for the entire application:

```
<context>
      <id>LSO.Portal</id>
      <uiname>LSO Portal</uiname>
      <match_patterns>
            <pattern>
                  <where>URL</where>
                  <matches>.*/sap/hcm_learning/leso\.htm.*</matches>
            </pattern>
      </match_patterns>
</context>
```

The application should be identified by a substring within the URL (`<where>`). The match pattern specified in `<matches>` is a regular expression following the syntax described in www.rexegg.com/regex-quickstart.html#ref. Please note that it is not case sensitive. In our example, it shall match the substring `/sap/hcm_learning/leso.htm` of the URL. For a generic solution, the protocol (http, https) and the domain name are not used as part of the match pattern, and any parameters following the page name "leso.htm" are also disregarded. Hence, the wildcard expression `".*"` is used at the beginning and the end of the URL pattern, as it matches any number of characters.



## 2.3 Multiple Match Patterns

To improve the accuracy of the context recognition, multiple patterns can be defined. Especially for applications where the individual application-specific patterns are not unique among web applications, the combination of two or even more match patterns can ensure a correct context recognition. For instance, within the SAP Netweaver Portal, the only identifying part within the URL is the string "`/irj/`", which might as well occur in other applications. Therefore, the string "`- sap netweaver portal`" from the page title is used as an additional pattern which has to be matched:

```
<context>
      <id>Web.SAP.Portal</id>
      <uiname>SAP Netweaver Portal</uiname>
      <match_patterns>
            <pattern>
                  <where>TITLE</where>
                  <matches>.* - sap netweaver portal</matches>
            </pattern>
            <pattern>
                  <where>URL</where>
                  <matches>.*/irj/.*</matches>
            </pattern>
      </match_patterns>
</context>
```

By default, all patterns are mandatory, i.e. they all have to match. Please see the next section on how to use optional patterns.

## 2.4 Optional Match Patterns

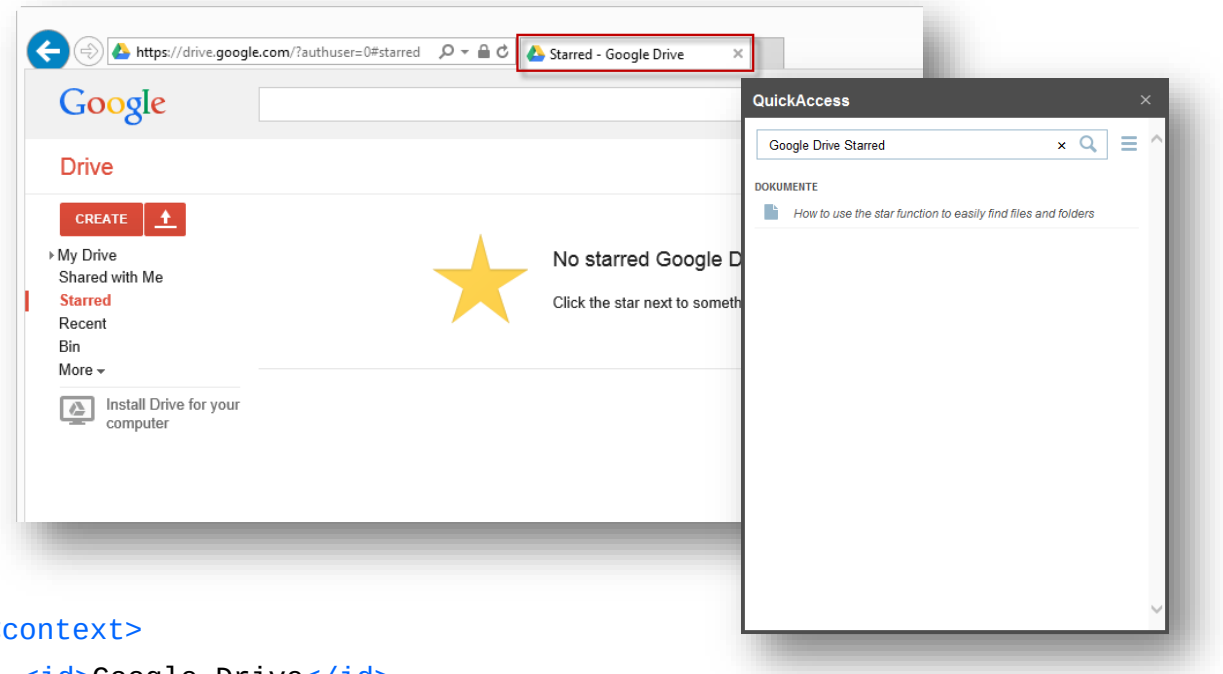
You might also want to define alternative match patterns (only one of two or more patterns have to match) to identify the application. This may be necessary if there are no match patterns within an application that stay the same throughout the entire application. In that case, the node `<optional>true</optional>` can be added to the respective match pattern:

```
<context>
  <id>Sample</id>
  <uiname>Sample Application</uiname>
  <match_patterns>
    <pattern>
      <where>URL</where>
      <matches>.*mainpage.*</matches>
      <optional>true</optional>
    </pattern>
    <pattern>
      <where>URL</where>
      <matches>.*shoppingcart.*</matches>
      <optional>true</optional>
    </pattern>
  </match_patterns>
</context>
```

In this example, the context "Sample Application" is recognized if either the string `mainpage` OR the string `shoppingcart` is a substring of the captured URL.

## 2.5 Context Patterns

While match patterns are used to detect the application itself, context patterns are used to define which context information shall be retrieved from the application. By default, only the application name defined in `<uiname>` will be used as context information for content filtering; but this can be extended by providing additional context patterns using a `<get_context_patterns>` node. In the following example for the Google Drive web application, a label displayed in the browser's title bar shall be used to extract additional context information:




```
<context>
  <id>Google.Drive</id>
  <uiname>Google Drive</uiname>
  <match_patterns>
    <pattern>
      <where>URL</where>
      <matches>.*drive.google.com.*</matches>
    </pattern>
  </match_patterns>
  <get_context_patterns>
    <pattern>
      <where>TITLE</where>
      <matches>(.*) - Google Drive</matches>
      <replace_with>$1</replace_with>
    </pattern>
  </get_context_patterns>
</context>
```

In the context pattern, the `TITLE` of the web page is defined as the place `<where>` to look for additional context information. If the title contains the substring " – Google Drive" (including the

blanks), it qualifies for context retrieval. Since the actual context information we are interested in is at the beginning of the title, the <matches> node starts with a so-called match group (.*), a wildcard placeholder enclosed in parentheses. The $1 in the <replace_with> node refers to this match group to extract its content as context information.
In the example, "Starred – Google Drive" will yield "Starred" as $1.

## 2.6 Alternative Context Patterns

Sometimes, the syntax of the string from which the context is retrieved varies within one application. In the following example for Microsoft Office 365 Online, the web applications for Word, Excel and PowerPoint can be identified in the page title as follows:

```
Document.docx – Microsoft Word Online
Book.xslx – Microsoft Excel Online
Presentation.pptx – Microsoft PowerPoint Online
```

In OneDrive, however, albeit using the same domain name, the title has a different structure and contains context information of interest at the beginning of the page title:

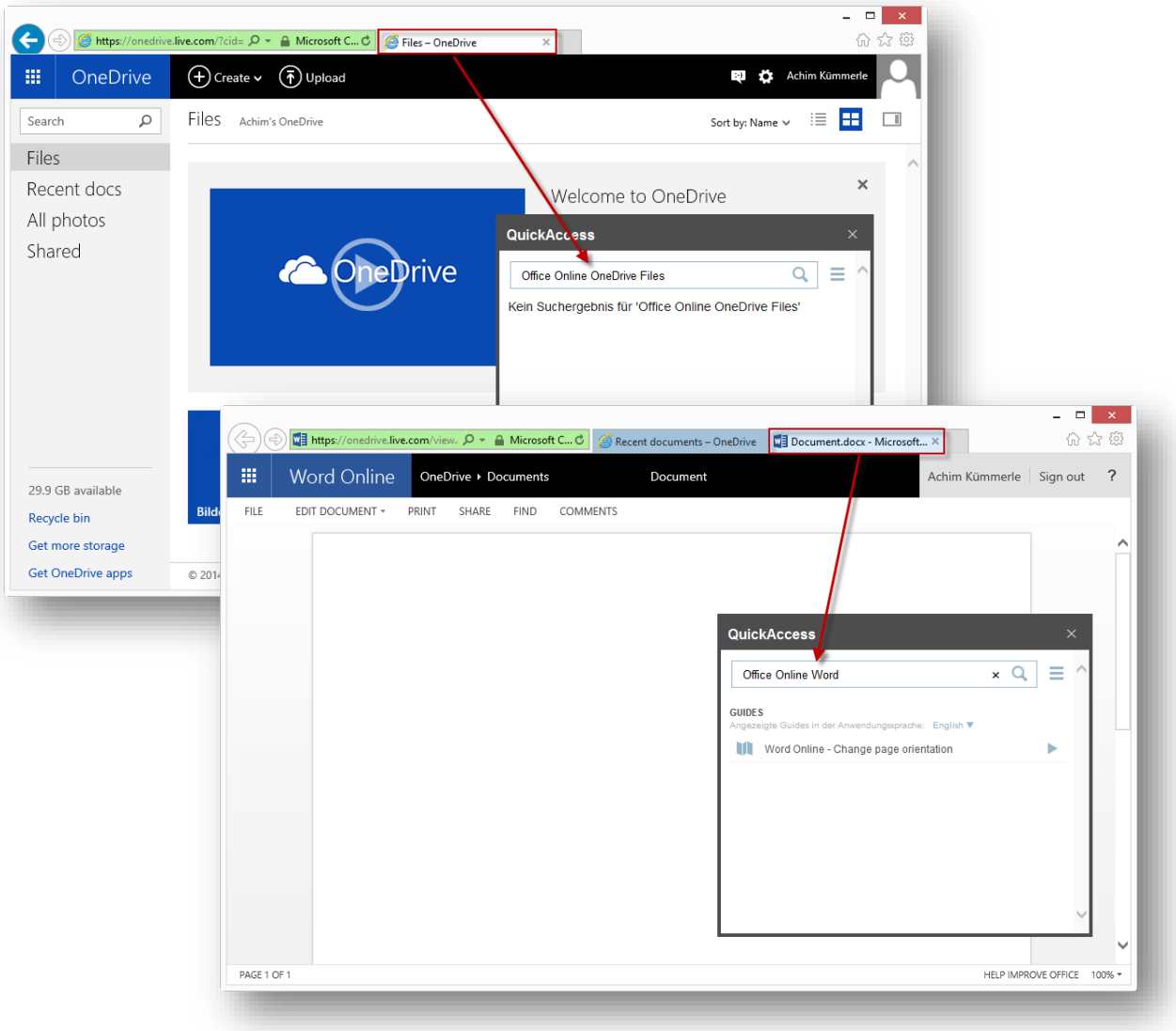
```
Files – OneDrive
Recent docs – OneDrive
Shared – OneDrive
```

To deal with such differences, additional context patterns can be defined within the same <get_context_patterns> node, as illustrated in the following example:

```
<context>
  <id>Office.Online</id>
  <uiname>Office Online</uiname>
  <match_patterns>
    <pattern>
      <where>URL</where>
      <matches>.*onedrive.live.com.*</matches>
    </pattern>
  </match_patterns>
  <get_context_patterns>
    <pattern>
      <where>TITLE</where>
      <matches>.* - Microsoft (.*?) Online</matches>
      <replace_with>$1</replace_with>
      < stop_if_matches>1</ stop_if_matches>
    </pattern>
```

```
    <pattern>
      <where>TITLE</where>
      <matches>(.*) - OneDrive</matches>
      <replace_with>OneDrive $1</replace_with>
    </pattern>
  </get_context_patterns>
</context>
```

Matching is done from the top to the bottom of the configuration file. By default, the last matching pattern is used, unless you inserted a forced stop after a pattern that matches. If you define several patterns, starting with a specific one that only matches a small number of cases, followed by a generic pattern that matches in any case, you may want to insert `<stop_if_matches>1</stop_if_matches>` after the specific pattern to ensure that this pattern is used if it matches, instead of the generic pattern that comes last.



## 2.7 Object Recognition Optimization

For object recognition, TT Guide and QuickAccess rely heavily on ID's of objects in HTML pages. This yields reliable results with static ID's. In some web applications, however, dynamic ID's are used for objects inside the web pages. This causes object recognition problems during rerecording and playback of Guides. To deal with this, the configuration file also allows you to define criteria for HTML element ID's which should be used for (or excluded from) object recognition. This is done using the `<use_id_pattern>` node.
In the following example for the SAP Enterprise Portal, all element ID's starting with either "`WD`", "`Link`", "`FD`", or "`htmlb_`" should be ignored. Since the list of IDs is a positive list, a NOT operator ("!") has to be put in front of the ID pattern:

```
<context>
      <id>Web.SAP.Portal</id>
      <uiname>SAP Enterprise Portal</uiname>
      <use_id_pattern>(?!WD|Link|FD|htmlb_).*</use_id_pattern>
      <match_patterns>
            <pattern>
                  <where>TITLE</where>
                  <matches>.* - sap netweaver portal</matches>
            </pattern>
            <pattern>
                  <where>URL</where>
                  <matches>.*/irj/.*</matches>
            </pattern>
      </match_patterns>
      <get_context_patterns>
            <pattern>
                  <where>TITLE</where>
                  <matches>(.*) - sap netweaver portal</matches>
                  <replace_with>$1</replace_with>
            </pattern>
      </get_context_patterns>
</context>
```

# 3 Reference

The following table lists all items that may be contained in the *com.tts.at.configuration.custom.xml* configuration file:

| Item | Example | Explanation |
|---|---|---|
| context | - (empty) | Basic tree element |
| id | `Google.Drive` | Any alphanumeric character plus period (.) hyphen (-) underscore (_) |
| uiname | `Google Drive` | Represents the application name as it is stored in the context field during recording of Guides or TT documents |
| match_patterns | - (empty) | Holds the definition of a match pattern, consisting at least of <pattern>, <where> and <matches>. |
| get_context_patterns | - (empty) | Holds the definition of a context pattern, consisting at least of <pattern>, <where> and <matches>. |
| pattern | - (empty) | |
| where | `URL` | Defines the place where the match (see below) must occur. The following values may be used:<br>`TITLE` – looks for a match in the HTML page's title element (as displayed in the browser's title bar or the tab).<br>`URL` – looks for a match in the URL (web address) of the web page. |
| matches | `.*google.com/drive/.*` | For match_patterns and get_context_patterns: Contains a regular expression that describes all matching strings. |
| replace_with | `$1` | For context patterns: If the regular expression in <matches> contains a capturing group (in parentheses), the replace_with item can be used to refer to the content of the capturing group. |
| optional | `true` | If there are several patterns and only one needs to be matched, mark them as <optional>true</optional>. |
| stop_if_matches | `1` | If <get_context_patterns> contains more than one <pattern>, you can use this option to stop evaluation upon the first match. Otherwise, the last match will be used. |