Company custom dictionary

1. Overview

The custom dictionary is a special mechanism that allows creating company-wide dictionaries that are intended to extend the vocabulary of the standard dictionary with custom words specific to your industry, domain, etc.

2. Dictionary format

The application supports two types of custom dictionary formats: textual (*.txt) and native compiled (*.clx). Note that the second, compiled format, is supported for English by the WebSpellChecker versions less than 5.24.0.0.

- A **textual dictionary** is stored in text lexicon files **(*.txt)**. It has a simple structure where each line contains only one word.

⚠ The last line in such a file must be empty, otherwise the last term in the list won't be recognized by the engine.

- A **native dictionary** is a compiled version of a textual dictionary (***.clx**). Compiling a custom dictionary into the native format will significantly improve the performance of the application, especially when the number of terms in the dictionary is over several hundreds.

3. Create custom dictionary

3.1. Create custom dictionary for default language (for the WebSpellChecker version less than 5.24.0.0)

Option A. Plain textual dictionary

- Open any text editor that is available on your system.
- Create a new wordlist in *.txt format with a so-called “ANSI” encoding which also might refer to the ISO 8859-1 standard, Latin alphabet No. 1 (ISO Latin 1) or Windows-1252 code page. Each new word in such a wordlist has to start from a new line. Also, it is recommended to sort the wordlist in alphabetical order.

⚠ There is a minor issue in non-compiled dictionaries: the last word in the list won't be recognized unless a new empty line (the end of a line character) is added at the very end of the wordlist. Thus, when creating a dictionary, make sure you have **an empty line added at the end of the document**.

- Place the newly created wordlist in <WebSpellChecker_Installation_Path>/AppServer/CustomDictionaries.
- Open the **CustDictConfig.xml** file for editing.
- Inside the **<CustomerDb> </CustomerDb>** tag, add a new Dictionary tag for your new dictionary:
  - **Dictionary DicId** – a custom dictionary ID. It must be a unique digital number (e.g. “1”). Also, make sure that the CustDictConfig. xml file doesn't contain the same ID assigned for another custom dictionary.
  - **FileName** – a custom dictionary file name (e.g. custom_dictionary_name.txt).
  - **LangShortName** – a special short code for a default language (e.g. the language short code for the American English language is e n_US). The full list of the default languages with their short codes is available here.
  - **Description** – a short custom dictionary description.

Here is an example of CustDictConfig.xml with two global custom dictionaries. One for American English and the other for British English.

CustDictConfig.xml

```
<CustomerDb>
        <Dictionary DicId="1">
        <FileName>custom_dictionary_name.txt</FileName>
        <LangShortName>en_US</LangShortName>
        <Description>An example of a plain textual custom dictionary for the American English language
(en_US)</Description>
        </Dictionary>

        <Dictionary DicId="2">
        <FileName>custom_dictionary_name2.txt</FileName>
        <LangShortName>en_GB</LangShortName>
        <Description>An example of a plain textual custom dictionary for the British English language (en_GB)
</Description>
        </Dictionary>

</CustomerDb>
```

Option B. Compiled dictionary

Also, you may choose to create a compiled custom dictionary instead of a plain textual dictionary to improve overall performance of the application when working with huge custom dictionaries.

- Open any text editor that is available on your system.
- Create a new wordlist in *.txt format with a so-called “ANSI” encoding which also might refer to the ISO 8859-1 standard, Latin alphabet No. 1 (ISO Latin 1) or Windows-1252 code page. Each new word in such a wordlist has to start from a new line. Also, it is recommended to sort the wordlist in alphabetical order.
- Place the newly created wordlist in <WebSpellChecker_Installation_Path>/AppServer.
- Compile a previously created wordlist using the **compileCustDict** command with appropriate values for **FileName** and **LangShortName** para meters as shown below . A new compiled custom dictionary will be placed into <WebSpellChecker_Installation_Path>/AppServer /CustomDictionaries/.

Compile Custom Dictionary on Windows

- Open Command Prompt –> Switch to <WebSpellChecker_Installation_Path>/AppServer/
- Execute the next command:

```
AppServerX.exe -compileCustDict CustomDictionaries/custom_dictionary_name.txt en_US
```

Compile Custom Dictionary on Linux

- Open command line –> Switch to <WebSpellChecker_Installation_Path>/AppServer/
- Run the next command:

```
./AppServerX -compileCustDict CustomDictionaries/custom_dictionary_name.txt en_US
```

- Go to the CustDictConfig.xml configuration file to verify if a new custom dictionary record has been written there. It will be added automatically by compileCustDict.

CustDictConfig.xml

```
<CustomerDb>
        <Dictionary DicId="2">
        <FileName>custom_dictionary_name.clx</FileName>
        <LangShortName>en_US</LangShortName>
        <Description>An example of a compiled custom dictionary for the American English language (en_US)<
/Description>
        </Dictionary>
</CustomerDb>
```

3.2. Create custom dictionary for Hunspell-based language (for the WebSpellChecker version 5.24.0.0+ for all languages)

- Open any text editor that is available on your system.
- Create a new wordlist in *.txt format with UTF-8 encoding. Each new word in such a wordlist has to start from a new line. Also, it is recommended to sort the wordlist in alphabetical order.

⚠ There is a minor issue in non-compiled dictionaries: the last word in the list won't be recognized unless a new empty line (the end of a line character) is added at the very end of the wordlist. Thus, when creating a dictionary, make sure you have **an empty line added at the end of the document**.

- Place the newly created wordlist in <WebSpellChecker_Installation_Path>/AppServer/CustomDictionaries directory.
- Open the <WebSpellChecker_Installation_Path>/AppServer/CustDictConfig.xml file.
- Open the **CustDictConfig.xml** file for editing.
- Inside the **<CustomerDb> </CustomerDb>** tag add a new Dictionary tag for your new dictionary:
    - **Dictionary DicId** – a custom dictionary ID. It must be an unique digital number (e.g. “3”). Also, make sure that the CustDictConfig. xml file doesn't contain the same ID assigned for another custom dictionary.
    - **FileName** – a custom dictionary file name (e.g. custom_dictionary_name.txt).
    - **Description** – a short custom dictionary description.
    - **LangShortName** – a special short code for an additional language (e.g. the language short code for the Arabic language is ar_SA). The full list of the additional languages with their short codes is available here.

CustDictConfig.xml

```
<CustomerDb>
        <Dictionary DicId="3">
        <FileName>custom_dictionary_name.txt</FileName>
        <LangShortName>ar_SA</LangShortName>
        <Description>Custom dictionary for the Arabic language</Description>
        </Dictionary>
</CustomerDb>
```

4. Enable custom dictionary in web app

Depending on the product and where it is integrated, pass an appropriate custom dictionary parameter with a required dictionary ID (DicId) as a value on your web-page.

Product integration	Custom dictionary parameter
SCAYT plugin for CKEditor 4	scayt_customDictionaryIds: 'DicId'
WSC Dialog plugin for CKEditor 4 (for v. less than 5.18.0.0)	wsc_customDictionaryIds: 'DicId'
WProofreader add-on for RTEs and HTML controls	customDictionaryIds: 'DicID'
WebSpellChecker API	custom_dictionary=DicID

5. Create shared custom dictionaries

If you have or plan to run many copies of WebSpellChecker Server in your infrastructure, and your infrastructure is dynamic (for example, you need to add or terminate the servers depending on the workload), you may want to create a centralized folder with custom dictionaries to store them in a permanent location. There is no default functionality available for this. However, we can share the approach we have already implemented in the Cloud version. Follow these steps:

1. Create a Git repository for shared custom dictionaries.
2. Upload your custom dictionary files and **CustDictConfig.xml** file with the dictionaries configuration into this repository. For details on creating Custom dictionaries, refer to steps described in sections above.
3. Clone the repository to all servers that will use shared custom dictionaries. For this sample setup, let's assume that we have cloned the repository to the **dictionaries_path** folder.
4. Configure your **<WebSpellChecker_Installation_Path/AppServer/AppServerX.xml** file on each WebSpellChecker Server:
    - Replace a path in **<CustDictConfig>** xml tag with **dictionaries_path/CustDictConfig.xml**.
    - Also replace path in **<CustDictDir>** xml tag with **dictionaries_path**.
      Now each WebSpellChecker application located on different servers use the same Git repository for global custom dictionaries.
5. Synchronize the servers with the Git repository. To do so, configure a cron task for Linux or Task scheduler for Windows and add task which will execute the '**git pull**' command for the repository once in several minutes.
6. To update you shared custom dictionaries, make appropriate updates into your Git repository.
   All your servers will now execute the '**git pull**' command and will use updated Custom Global dictionaries.