Configuring WProofreader Cloud for CKEditor 5 up to v11.2.0

This guide outlines the main steps that you need to follow in order to integrate and configure WProofreader with CKEditor 5. All the described steps are provided for the Cloud version of WProofreader.

> **Good to know:**
>
> - Unlike CKEditor 4, where two WebSpellChecker plugins, SCAYT and WSC Dialog, are integrated by default and available under ABC button, there is no separate plugin for spelling and grammar check available in CKEditor 5.
> - **WProofreader** is a new ultimate multi-language proofreading tool. It provides both instant and on-click proofreading modes in a new convenient UI. It can be integrated with a wide range of various rich text editors, including CKEditor 5.

Step 1. Obtain service ID

Sign up for the trial version to get started with WProofreader Cloud. Once you have subscribed, you will receive an email with your service ID, an activation key, required for the WProofreader service activation.

> If you already have one, you can **skip** this step.

Step 2. Initialize CKEditor 5

Before integrating and configuring WProofreader, CKEditor should be properly configured in your web application. If you haven't done it yet, refer to Quick Start of the CKEditor 5 documentation pages for details.

Step 3. Grant WProofreader access to CKEditor 5 instance

Adjust initialization of CKEditor 5 to grant WProofreader access to it. You need to add a link to CKEditor 5 instance object as shown in the example below.

```
<textarea name="content" id="editor">This is a sample text.</textarea>

<script>
        ClassicEditor
            .create(document.querySelector('#editor'))
            .then(editor => {
                        editor.element.querySelector('.ck-editor__editable').editor = editor;
            })
</script>
```

Step 4. Initialize WProofreader

Now you need to enable WProofreader with the **autoSearch** feature. You can initialize WProofreader in your web app: using the **config variable** or **inline data attributes**.

Option A. Initializing using config variable

1. Add config with options

Define the required configuration options in WEBSPELLCHECKER_CONFIG.

> - The configuration options can be added as a script directly on the page or loaded from *.js file (e.g. wscbundle_config.js) where you define these options.
> - It can be added to any location on a web page before or after the wscbundle.js script. However, if you add wscbundle.js **asynchronously**, the configuration options must be loaded **before** the script.

Required options for the WProofreader Cloud:

- Specify your **serviceId**. It is a required option for the service activation.
- Enable autoSearch functionality using the **autoSearch** parameter.

This is an example of the defining WEBSPELLCHECKER_CONFIG in a separate script directly on a web page.

```
<script>
        window.WEBSPELLCHECKER_CONFIG = {
                autoSearch: true,
                lang: 'auto', // set the default language
        serviceId: 'your-service-ID' // the activation key for the Cloud-based version
   };
</script>
```

Alternatively, you can create a *.js file (e.g. wscbundle_config.js) with WEBSPELLCHECKER_CONFIG on your end and then load it from the file on your web page as shown in example below.

This is an example of the **wscbundle_config.js** file with WEBSPELLCHECKER_CONFIG.

wscbundle_config.js

```
window.WEBSPELLCHECKER_CONFIG = {
        autoSearch: true,
        lang: 'en_US',
    serviceId: 'your-service-ID'
};
```

Here is an example of the script that you need to add on your web page with the path to **wscbundle_config.js**.

```
<script type="text/javascript" src="[path_to_config]/wscbundle_config.js"></script>
```

2. Add WProofreader script

Add the **wscbundle.js** script on your web page.

```
<script type="text/javascript" src="https://svc.webspellchecker.net/spellcheck31/wscbundle/wscbundle.js"><
/script>
```

Option B. Initializing using inline data attributes

In general, the initialization of WProofreader using inline attributes is a good option if you want to have a single script with the base options. In order to define an additional option as an inline data attribute, use '**date-wsc-option_name**', e.g. 'data-wsc-lang'.

You can find the full list of options available here.

```
<script
   data-wsc-serviceid="your-service-ID"
   data-wsc-autosearch="true"
   data-wsc-lang="en_US"
   src="https://svc.webspellchecker.net/spellcheck31/wscbundle/wscbundle.js">
 </script>
```

There is a limitation which must be taken into account when using inline data attributes for the WProofreader options. As for now, only options which have **boolean** or **string** types can be used as data attributes. The options with **array** or **number** type **are not supported** (e.g. **actionItems** , **suggestionsCount**, **moreSuggestionsCount).**

Step 5. WProofreader for CKEditor 5 in action

Check this short video to see how WProofreader works in a bundle with CKEditor 5.

Your browser does not support the HTML5 video element

Still have a question?

Please feel free to contact our technical team if you are having any difficulties with the configuration.