Get started with WProofreader Server (autoSearch)

WProofreader add-on for rich text editors enables grammar and spell check, combining spelling and grammar suggestions while you type or work with your text in a floating dialog.

Its auto-searching feature enables detecting new editable fields on the page and proofreading the text they contain automatically on hover selection. No additional actions or plugins are required for enabling WProofreader in a specific WYSIWYG editor or HTML editable element. A single configuration applies to almost all editors and controls. Here is the list of WProofreader supported integrations.

This guide outlines the main steps to integrate and configure WProofreader with auto-searching functionality (**autoSearch** option). All the steps described are provided for the **Server version of WProofreader** that is hosted and deployed on your server.

1. Initialize WProofreader

There are two approaches for initializing WProofreader with the **autoSearch** feature in your web app: using the **config variable** or **inline data attributes**. Their descriptions follow.

Option A. Initializing using config variable

1. Add config with options

Define the required options in WEBSPELLCHECKER_CONFIG. 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**, this CONFIG must be added **before** the script.

- Enable autoSearch functionality using the **autoSearch** parameter.
- Specify **serviceProtocol**, **serviceHost**, **servicePort** and **servicePath** to access the service.

In case of the direct connection to WebSpellChecker AppServer (AppServer is an entry point), you need to configure the SSL connection. Follow the steps described on the Enabling SSL connection with AppServer page (if you missed that step during the installation).

```
<script>
        window.WEBSPELLCHECKER_CONFIG = {
                autoSearch: true,
                lang: 'en_US', // set the default language
        serviceProtocol: 'https',
        serviceHost: 'your_host_name',
        servicePort: '443',
        servicePath: 'virtual_directory/api' // by default the virtual_directory is wscservice
   };
</script>
```

If you are using the older version of the deployment where **SSRV.CGI component** is an entry point, please refer to the sample below.

```
<script>
        window.WEBSPELLCHECKER_CONFIG = {
                autoSearch: true,
                lang: 'auto', // set the default language
        serviceProtocol: 'https',
        serviceHost: 'your_host_name',
        servicePort: '443',
        servicePath: 'virtual_directory/script/ssrv.cgi'
   };
</script>
```

You also have an option to load your CONFIG option on your web page from the file. Just create a *.js file (e.g. wscbundle_config.js) with CONFIG. This is an example of the **wscbundle_config.js** file.

wscbundle_config.js

```
window.WEBSPELLCHECKER_CONFIG = {
        autoSearch: true,
        ...
};
```

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="http(s)://your_host_name/wscservice/wscbundle/wscbundle.js"></script>
```

Option B. Initializing using inline data attributes

In general, initialization of WProofreader using inline attributes is a good option if you want to have a single script with the base options. 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 here.

```
<script
        data-wsc-autosearch="true"
        data-wsc-lang="auto"
        data-wsc-enableGrammar="true"
        data-wsc-serviceProtocol="https"
        data-wsc-serviceHost="your_host_name"
        data-wsc-servicePort="443"
        data-wsc-servicePath="virtual_directory/api"
        src="https://your_host_name/virtual_directory/wscbundle/wscbundle.js">
</sctipt>
```

⚠ Consider the following limitation 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).

2. Adjust default settings

You can customize the default options using WProofreader API.

ⓘ You can check the demos of WProofreader integrations with various rich-text editors on our website.