

Check ommand

The **check** command is available since the release of WebSpellChecker 5.5.4x in October 2019. It combines all available check types (spelling and grammar) of text in a single command.

 **Command name:** check

Here is a list of all possible parameters and values that can be used with the **check** command.

 The list of parameters can be used and available only when **spelling check is enabled**. These parameters are marked with yellow color.

#	Parameter	Possible Values	Default Value	Description
1	format	- json - xml	json	The response format for output data.
2	callback	callback function name		A callback function name that will be used to manipulate with the JSON data received from the server. Such approach enables sharing of data bypassing same-origin policy. It can be used only along with "format=json".
3	text	plain text		A piece of text which will be sent for check. The text has to be in the UTF-8 encoding. Any found tags in the text will be interpreted as plain text as well.  Avoid using # and & symbols in the text.
4	tokens	Array of strings, e.g. ["This is a sentence number 1.", "This is a sentence number 2."]		A piece of text separated in tokens that will be sent for check. The text should be presented as an array of strings. Right now each string is a token which equals one sentence.  You can use either tokens or text at a time in a request. However, using text is more common.
5	lang	Supported languages (e.g. ar_SA)	en_US	A short code of a language which will be used for check.
6	disable_spelling	- true - false	false	Disable the check text for spelling errors.
7	disable_grammar	- true - false	false	Disable the check text for grammar and style problems.
8	user_dictionary	user dictionary name (e.g. testdict)		A user dictionary name which will be used during spell checking.
9	user_wordlist	additional wordlist		The list of additional comma-separated words which will be used for spell checking.
10	custom_dictionary	custom dictionary IDs (e.g. 100694)		Global custom dictionary ID(s) which can be used during spell checking.  Each new Dictionary on the creation obtains its unique Dictionary ID. Depending on the type of the version of product you are using, refer to Cloud or Server guides respectively.
11	ignore_all_caps	0 – Do not ignore all words written in capital letters (e.g. UPPERCASE). 1 – Ignore all words written in capital letters.	0	Ignore capitalized words.

12	ignore_words_with_numbers	0 – Do not ignore words that contain numbers (e.g. Number1). 1 – Ignore words that contain numbers.	0	Ignore words containing numbers.
13	ignore_mixed_case	0 – Do not ignore words with mixed case letters (e.g. MixedCase). 1 – Ignore words with mixed case letters.	0	Ignore words written with mixed case letters.
14	ignore_domain_names	0 – Do not ignore web addresses that start with either "www", "http:" or "https:" and end with a domain name. 1 – Ignore web addresses and domain names.	0	Ignore domain names, web addresses.
15	min_word_length	minimal number of letters in a word to be checked	3	The minimal number of letters in the word which will be checked for spelling. E.g. if 3 is specified, the words with 2 letters and less will be ignored.
16	custom_punctuation	string of chars (e.g. "-")	-	A list of characters that should be considered as delimiters during spelling check.
17	short_answer	- true - false	false	Shorten every static string JSON key name, like messages or type down to its first character, for example: <i>m</i> - matches, message <i>o</i> - offset <i>l</i> - length <i>t</i> - type <i>r</i> - rule <i>s</i> - suggestions
18	customerid	your-service-id value		A special service ID value (activation key) that has to be passed to a request query. It's obtained upon subscription to the Cloud services (paid or trial).  Applicable only for the Cloud version.
19	auto_lang_priorities	{"en":"en_US", "es":"es_ES"}		Priority of language dialect for auto-detected language code. For example, if auto-detect returns "en", then American English will be used as a language for check.
20	disable_style_guide	- true - false	false	Disabling style guide functionality starting WebSpellChecker v.5.29.0.0
21	disabled_rules	array	[]	Disabling specific grammar rules IDs starting WebSpellChecker v.5.29.0.0
22	disabled_categories	array	[]	Disabling specific grammar rules categories starting WebSpellChecker v.5.29.0.0
23	enforce_ai	- true - false	false	To replace the classic algorithmic engines with an AI-powered engine starting WebSpellChecker v.5.25.0.0. It only works along with American, British, Canadian and Australian English.

Response Structure

The **result** is an array of objects which contains matches, where **matches** is also an array of objects consisting of attribute-value pairs.

The table below represents the following attribute-value pairs:

Attribute	Type	Value	Description
type	string	- spelling - grammar	Type of the problem found.
offset	number		Start position of a problem found in a sentence/text; start position value here equals '0'.

length	number		The length of offset from the beginning of the error; offset here is the beginning of error related to sentence/text plus the length of the error.
ud	boolean	• true	True if a misspelled word is present in a user dictionary. This attribute-value pair is used to indicate the application not to underline the word in UI.
suggestions	array of strings		Suggested corrections for spelling, grammar or style problem.
rule	string		A short description of the problem by rule; available only for type 'grammar'.
message	string		Description of the problem; available only for type 'grammar'.

Type: Spelling

```
{
  "result": [
    {
      "matches": [
        {
          "type": "spelling",
          "offset": X1,
          "length": Y1,
          "ud": true,
          "suggestions": ["..."]
        }
      ]
    }
  ]
}
```

Type: Grammar

```
{
  "result": [
    {
      "matches": [
        {
          "type": "grammar",
          "offset": X2,
          "length": Y2,
          "rule": "...",
          "message": "...",
          "suggestions": ["..."]
        }
      ]
    }
  ]
}
```

Example 1.1 [GET]: Check request for American English text with all available check types (output in JSON)

Request URL (GET):

http(s)://svc.webspellchecker.net/api?cmd=check&text=this sampl text demonstrates the work of the Web API service.&lang=en_US&format=json&customerid=[your-service-id]

Parameters:

- Command: *check*
- Text: *this sampl text demonstrates the work of the Web API service.*
- Language: *en_US*
- Format: *json*

Request response:

```
{
  "result": [
    {
      "matches": [
        {
          "type": "spelling",
          "offset": 5,
          "length": 5,
          "suggestions": [
            "sample",
            "sampled",
            "sampler",
            "samples",
            "ample",
            "amply",
            "scamp",
            "stamp"
          ]
        },
        {
          "type": "grammar",
          "offset": 0,
          "length": 4,
          "rule": "UPPERCASE_SENTENCE_START",
          "message": "This sentence does not start with an uppercase letter",
          "suggestions": [
            "This"
          ]
        }
      ]
    }
  ]
}
```

Example 1.2 [GET]: Check request for American English text with all available check types (output in XML)

Request URL (GET):

http(s)://svc.webspellchecker.net/api?cmd=check&text=this sampl text demonstrates the work of the Web API service.&lang=en_US&format=xml&customerid=[your-service-id]

Parameters:

- Command: *check*
- Text: *this sampl text demonstrates the work of the Web API service.*
- Language: *en_US*
- Format: *xml*

Request response:

```

<result>
  <result>
    <matches>
      <matches>
        <type>spelling</type>
        <offset>5</offset>
        <length>5</length>
        <suggestions>
          <suggestions>sample</suggestions>
          <suggestions>sampld</suggestions>
          <suggestions>sampler</suggestions>
          <suggestions>samples</suggestions>
          <suggestions>ample</suggestions>
          <suggestions>amply</suggestions>
          <suggestions>scamp</suggestions>
          <suggestions>stamp</suggestions>
        </suggestions>
      </matches>
      <type>grammar</type>
      <offset>0</offset>
      <length>4</length>
      <rule>UPPERCASE_SENTENCE_START</rule>
      <message>This sentence does not start with an uppercase letter</message>
      <suggestions>
        <suggestions>This</suggestions>
      </suggestions>
    </matches>
  </result>
</result>

```

Example 1.3 [GET]: Check request for American English text as two tokens with all available check types (output in JSON)

Request URL (GET):

[https://svc.webspellchecker.net/api?cmd=check&tokens=\["this sampl text.", " It demonstrate the work of the Web API service."\]&lang=en_US&customerid=\[your-service-id\]](https://svc.webspellchecker.net/api?cmd=check&tokens=[)

Parameters:

- Command: *check*
- Tokens: *["this sampl text.", " It demonstrate the work of the Web API service."]*
- Language: *en_US*
- Format: *json*

Request response:

```
{
  "result": [
    {
      "matches": [
        {
          "type": "spelling",
          "offset": 5,
          "length": 5,
          "suggestions": [
            "sample",
            "sampled",
            "sampler",
            "samples",
            "ample",
            "amply",
            "scamp",
            "stamp"
          ]
        },
        {
          "type": "grammar",
          "offset": 0,
          "length": 4,
          "rule": "UPPERCASE_SENTENCE_START",
          "message": "This sentence does not start with an uppercase letter",
          "suggestions": [
            "This"
          ]
        }
      ]
    },
    {
      "matches": [
        {
          "type": "grammar",
          "offset": 4,
          "length": 11,
          "rule": "IT_VBZ",
          "message": "Did you mean demonstrates?",
          "suggestions": [
            "demonstrates"
          ]
        }
      ]
    }
  ]
}
```

Example 1.4 [GET]: Check request for American English text as two tokens with all available check types and shortened response (output in JSON)

Request URL (GET):

`https://svc.webspellchecker.net/api?cmd=check&tokens=["this sampl text.", " It demonstrate the work of the Web API service."]&lang=en_US&short_answer=true&customerid=[your-service-id]`

Parameters:

- Command: *check*
- Tokens: *["this sampl text.", " It demonstrate the work of the Web API service."]*
- Language: *en_US*
- Format: *json*
- Short Answer: *true*

Request response:

```
{
  "r": [
    {
      "m": [
        {
          "t": "spelling",
          "o": 5,
          "l": 5,
          "s": [
            "sample",
            "sampled",
            "sampler",
            "samples",
            "ample",
            "amply",
            "scamp",
            "stamp"
          ]
        },
        {
          "t": "grammar",
          "o": 0,
          "l": 4,
          "r": "UPPERCASE_SENTENCE_START",
          "m": "This sentence does not start with an uppercase letter",
          "s": [
            "This"
          ]
        }
      ]
    },
    {
      "m": [
        {
          "t": "grammar",
          "o": 4,
          "l": 11,
          "r": "IT_VBZ",
          "m": "Did you mean demonstrates?",
          "s": [
            "demonstrates"
          ]
        }
      ]
    }
  ]
}
```

Example 1.4 [POST]: Check request for American English text with all available check types (output in JSON)

Here we use the same request and parameters as described in **example 1.1** but form it as a POST request.

Request URL (POST):

<https://svc.webspellchecker.net/api?>

Body (Raw):

cmd=check&text=this sampl text demonstrates the work of the Web API service.&lang=en_US&format=json&customerid=[your-service-id]

Request response:

```
{
  "result": [
    {
      "matches": [
        {
          "type": "spelling",
          "offset": 5,
          "length": 5,
          "suggestions": [
            "sample",
            "sampled",
            "sampler",
            "samples",
            "ample",
            "amply",
            "scamp",
            "stamp"
          ]
        },
        {
          "type": "grammar",
          "offset": 0,
          "length": 4,
          "rule": "UPPERCASE_SENTENCE_START",
          "message": "This sentence does not start with an uppercase letter",
          "suggestions": [
            "This"
          ]
        }
      ]
    }
  ]
}
```

Example 1.5 [POST]: Check request for text with the auto-detected language (output in JSON)

In this POST type request, we use "auto" as a value for language and define the priorities for the language dialects. If "en" is detected, then AI-based English will be used during check request.

Request URL (POST):

<https://svc.webspellchecker.net/api?>

Body (Raw):

cmd=check&text=this sampl text demonstrates the work of the Web API service.&lang=auto&format=json&customerid=[your-service-id]&auto_lang_priorities={"en":"en_AI"}