

JSpell IFrame Pro Spell Checker

Version 7.5

Copyright © 1996-2007 The Solution Cafe Inc.
All Rights Reserved

Table of Contents

License Agreement	3
Introduction	6
Requirements	7
Quick Start.....	7
Deploying JSpell to Your Web Server.....	7
Microsoft Windows with IIS.....	7
J2EE Compatible Server	8
Java JDK 2 Compatible Server (non J2EE).....	8
Adding JSpell to Your Web Page	8
Adding JavaScript Code to Your Web Page	8
Adding a Spell Check Button	8
Customizing the Spell Checker Appearance	10
Using a Style Sheet	10
The Learn Button	11
Customizing the Spell Checker Behavior	11
language.....	12
forceUpperCase	12
ignoreIrregularCaps.....	12
ignoreFirstCaps.....	12
ignoreNumbers	12
ignoreUpper	13
ignoreDouble	13
confirmAfterLearn	13
confirmAfterReplace	13
supplementalDictionary.....	13
spellCheckURL	14
styleSheetURL	14
Using Language Features	14
Word Filtering.....	14
Appendix A – Tomcat 3.x, 4.x and 5.x.....	16
Appendix B – JRun	17

License Agreement

License Agreement for JSpell IFrame Spell Checker

The enclosed software (the "Software") is owned by The Solution Cafe, Inc. and is protected by copyright law. Upon acceptance of the terms of this License Agreement, your rights and obligations with respect to use of the Software are as follows. These rights are in supplement of US copyright law and the requirements of local laws and treaties.

Under this license, you may:

You are permitted to run one instance of the Software per server and per domain.

You can have an unlimited number of clients connected to that server process.

You can use the Software on any number of web pages within your domain, provided that you do not execute the Software on more than one server, and that the server does not serve multiple virtual or non-virtual domains.

You may make backups of the Software in a manner consistent with industry practice.

You may not:

Redistribute the source code in any form without the prior written and signed consent of The Solution Cafe, Inc.

Redistribute the server portion of the Software in any manner, either inside or outside the purchasing organization without having the express written agreement from The Solution Cafe, Inc. and without having obtained a JSpell Server redistribution license.

Execute the server portion of the Software on more than one machine.

De-compile, disassemble, or reverse-engineer the Software

Create derivative works based on the source code for distribution as software components.

Use, copy, distribute or transfer the Software except as provided for in this License Agreement.

Embed the Software in any type of reusable software component, or web page design service or process.

Transfer ownership of the Software, License Agreement or Documentation to another party.

Licensee acknowledges and agrees to these terms and any breach of these terms by Licensee shall be considered a material breach of this Agreement resulting in irreparable harm to The Solution Cafe, Inc. for purposes of obtaining a temporary or permanent injunction against Licensee, in addition to which The Solution Cafe, Inc. may seek all related damages.

The source code for the Software (all files with a file name suffix of ".java" or ".jsp") is the confidential information of The Solution Cafe, Inc. A source code Licensee shall not disclose or distribute such source code in whole or in part under any circumstances without the express written agreement of The Solution Cafe, Inc. Licensee acknowledges and agrees that any disclosure of source code by Licensee shall be considered a material breach of this Agreement resulting in irreparable harm to The Solution Cafe, Inc. for purposes of obtaining a temporary or permanent injunction against Licensee, in addition to which The Solution Cafe, Inc. may seek all related damages.

Limited Warranty and Warranty Disclaimer. The Solution Cafe, Inc. warrants that the media on which the Software is furnished will be free from defects for three (3) months from your receipt of the product. Your sole remedy in the event of breach of the foregoing warranty is replacement of the media. THE SOLUTION CAFE, INC. DISCLAIMS ALL OTHER WARRANTIES EXPRESS OR IMPLIED, INCLUDING THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR ANY LOCAL ANALOGUES THERETO. You may have other rights to the extent required of licensors under local law.

Limitation of Liability. Unless otherwise required by applicable law, THE SOLUTION CAFE, INC. SHALL NOT BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL OR SIMILAR DAMAGES (IRRESPECTIVE OF WHETHER THE SOLUTION CAFE, INC. HAS ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES) AND THE SOLUTION CAFE, Inc.'s TOTAL LIABILITY HEREUNDER (INCLUDING BUT NOT LIMITED TO ANY POSSIBLE LIABILITY FOR INDEMNITY, DEFENSE OR HOLD HARMLESS OBLIGATIONS) SHALL NOT EXCEED THE TOTAL AMOUNT PAID FOR THE SOFTWARE. THIS LIMITATION SHALL APPLY EVEN IN THE EVENT OF BREACH OF CONTRACT OR ACTIVE OR PASSIVE NEGLIGENCE AND REGARDLESS OF WHETHER ANY REMEDY FAILS OF ITS ESSENTIAL PURPOSE. YOU AGREE TO INDEMNIFY, DEFEND AND HOLD THE SOLUTION CAFE, INC. HARMLESS FROM AND AGAINST ANY THIRD-PARTY CLAIM IN EXCESS OF THIS LIMITATION.

General. This Agreement shall be governed by the laws of the State of Florida, without regard to conflict of laws principles. This Agreement may only be

modified by a written document signed by the party or parties to be bound. Except as may be specified in such a written agreement, this is the entire agreement between you and The Solution Cafe, Inc. and all other terms are rejected. You agree not to use, ship, or export the Software in violation of law.

US Government Restricted Rights. If provided to the US Government, the Software is provided with RESTRICTED RIGHTS. Use, duplication or disclosure by the US Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of the Commercial Computer Software Restricted Rights clause at 48 CFR 52.227-19, as applicable and as modified herein.

Introduction

The [JSpell IFrame Spell Checker](#) is designed to let you spell check the contents of contentEditable IFrame controls. This type of IFrame control is specific to Internet Explorer 5.0+ on the Microsoft Windows platform.

The JSpell IFrame Spell Checker package includes a copy of the htmlArea 2.03 WYSIWYG editor which has been customized to work with the JSpell IFrame Spell Checker. You may also use the JSpell IFrame Spell Checker with IFrame controls from other vendors or with IFrame controls that you have written yourself.

You can use the Forms Spell Checker with Microsoft IIS and ASP pages, Java™ JSP pages, CGI, PHP and more. The JSpell IFrame Spell Checker is compatible with virtually all servers since it relies upon the platform independent Java™ language to provide spell checking services.

Please see the next section for the installation and usage requirements.

We are sure you will greatly enjoy using this product. If you have any suggestions or feedback, please visit our support page at: <http://www.thesolutioncafe.com/jspellsupport.html>.

Now, let's get started!

Requirements

Client Side (Browser) – JSpell Evolution is designed for compatibility with Internet Explorer 5.5 through 7.0 on the Microsoft Windows platform.

Server Side - If you are using a web server other than Microsoft IIS, you will need a Java™ Servlet Engine running on your web server. We recommend [Tomcat](#) from the Apache organization. Tomcat is a free open source plug-in that operates with most web servers including Apache and Netscape. JSpell is also compatible with any J2EE compatible web server or any web server that supports JSP pages. It is even possible to use JSpell in a proxied mode where spell check requests are handled by a second server, as long as the first server can handle JSP, Servlet or PHP requests.

Quick Start

The JSpell IFrame Spell Checker is a client server application. As such, there are two main steps that you have to complete, **Deploying JSpell to Your Web Server** and **Adding JSpell to Your Web Page**. From start to finish, the installation of JSpell usually takes about fifteen minutes to one hour, including the time it takes to download files.

The JSpell IFrame Spell Checker is distributed in two formats: if you are using Microsoft Windows with IIS then you received an executable installer named **jspelliframe2k4.exe**; if you are using any other web server then you received a file in the J2EE compatible WAR file format, **jspelliframe2k4.war**.

In addition, you may have received dictionaries for languages other than U.S. English with the following naming convention, `lex_enGB.jdx` "English (GB)", or `lex_frFR.jdx` "French (FR)". These files will need to be placed in a common directory accessible to JSpell. If you are using the Windows installer version of JSpell IFrame then this directory defaults to "C:\Program Files\JSpell\lexicons". Otherwise, the directory is specified in the file `web.xml` using the J2EE configuration parameter `jspellLexicons`.

Deploying JSpell to Your Web Server

Microsoft Windows with IIS

Execute the setup program, `jspelliframe2k4.exe`. This program will install JSpell IFrame into your Program Files directory, configure and start an NT service for JSpell and launch a test page to verify the correct installation of the product.

J2EE Compatible Server

Deploy the jspelliframe2k4.war file to your web server using the deployment techniques standard to your web server. For example, on Tomcat, this involves copying the jspelliframe2k4.war file into the Tomcat webapps directory and restarting the server. Copy the lex_enUS.jdx file to a directory on your server. The default location for this file is the /jspell/ directory, however, you may modify the location by editing the jspellLexicons parameter in the WEB-INF/web.xml file.

Java JDK 2 Compatible Server (non J2EE)

Extract the contents of the jspelliframe2k4.war file into your web application directory. Place the file jspelliframe2k4.jar into the classpath for your web application. Place the JSpell.jsp file, the HTML files and the images files into your web application directory. Copy the lex_enUS.jdx file to a directory on your server. The default location for this file is the /jspell/ directory, however, you may modify the location by editing the jspellLexicons parameter in the WEB-INF/web.xml file.

Test the spell checker: <http://localhost/jspelliframe2k4/test.html>.

Adding JSpell to Your Web Page

To use JSpell on your web page you must copy the JSpell JavaScript code onto each web page that will require spell checking services.

Adding JavaScript Code to Your Web Page

The JavaScript code is contained within the <SCRIPT> </SCRIPT> block on the test.html web page. You should always refer to the latest JavaScript code contained within the test.html file to make sure you're using the latest updates.

The items in **RED** are items that you may need, or want to modify, in order to use JSpell in your environment.

```
<script language="JavaScript" src="/jspelliframe2k4/jspelliframe.js"></script>
```

Adding a Spell Check Button

The test.html page that ships with JSpell IFrame uses the contentEditable IFrame control htmlArea 2.03 by Interactive Tools. To add the spell checker functionality to an IFrame control from another vendor or to an IFrame control that you have written, then you must add a button to your IFrame control or your web page that triggers the spell check functionality.

For example, in the IFrame control that ships with JSpell IFrame the following JavaScript command triggers the spell check:

```
parent.ActionSpellCheck(document.all[_editor].contentWindow);
```

The function ActionSpellCheck requires the DOM path to the contentWindow of your IFrame editor control. Also, if you are adding the spell checker to the toolbar of your editor you may use the included spellcheck18.gif or spellcheck24.gif files in the images directory for your spell check button.

Customizing the Spell Checker Appearance

The Spell Checker can be customized in a number of ways. For example, you can change the appearance of the spell checker using style sheets; you can control what gets treated as an error by the spell checker and more.

Using a Style Sheet

JSpell comes with two style sheets that you may experiment with. The first style sheet is located in the file 'jspell.css', the other style sheet is in the file 'jspell_alt.css'. The style sheet is specified using the styleSheetURL parameter contained within the jspelliframe.js file. Examples of these style sheets running in Internet Explorer are shown in the following figures.

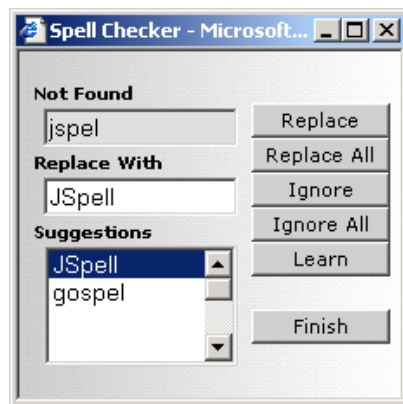


Figure 1 - jspell.css

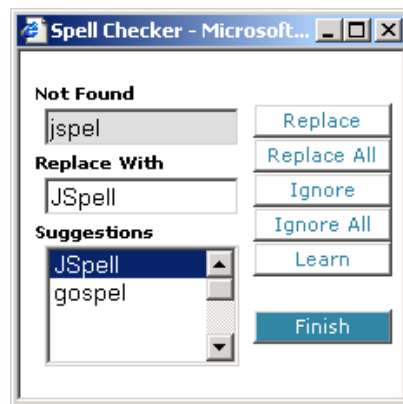


Figure 2 - jspell_alt.css

You can create your own style quite easily. At the time this document was written you can change the appearance of the buttons and the background. You can modify the JSpellIFrame.jsp source to use additional styles if necessary. However, if you modify JSpellIFrame.jsp, you will have to incorporate those changes into any updates that you receive from us. Of course, if you think you have some great modifications, let us know and we may just incorporate them into the standard release as well.

The Learn Button

The JSpell IFrame Spell Checker has the ability to semi-permanently ignore certain words. This feature is very similar to the Learn feature of standard word processor spell checkers. In the JSpell Spell Checker the learning of words is accomplished using client side cookies.

This allows the user to prevent certain words from being flagged as errors. However, because this feature is implemented using client side cookies, if the user changes computers or erases the cookies from their machine then the words will no longer be saved.

This approach has advantages over other client server solutions, in that the central dictionary does not get corrupted when a user 'learns' an incorrect word. However, you may find that you want to disable this feature.

To do so, set the **disableLearn** parameter to true. The disableLearn parameter is found in the file `jspelliframe.js`.

When you set the disableLearn parameter to true, the Learn button will be removed from the spell checker dialog.

Customizing the Spell Checker Behavior

Most spell checkers provide a way to customize the behavior of the spell check process, for example, if you want to make the spell checker flag repeated words as an error, or if you want words that are in UPPER CASE to be treated as correctly spelled you can normally set these as parameters of the spell checker.

JSpell provides the same capability via a number of JavaScript parameters. The parameters are set in the file `jspelliframe.js` contained in your JSpell installation directory. You can rely on hard coded values in your HTML or you can provide a way for users to set and retrieve their preferences and dynamically generate the JavaScript parameters for your users (this is not discussed here). The parameters that you can modify are shown in the next few sections.

language

The language JavaScript variable is a new feature of JSpell IFrame which lets you specify the language in which to perform your spell checks. The language variable defaults to "English (US)". If you have purchased and installed additional dictionaries from The Solution Café then you may set this variable to the correct value for your dictionary file. For example, if you want to perform a spell check in Spanish then you would set the language parameter to: "Spanish (ES)". See the section titled "Using Language Features" for more information.

forceUpperCase

With the forceUpperCase parameter set to true, all text that is spell checked will be returned as UPPER CASE. For example, if you spell check the following: "the quick brown fox JUMPS OVER THE LZY DOG", then the following words will be flagged as errors: the, quick, brown, fox, LZY and all of the suggested replacements will be in UPPER CASE.

ignoreIrregularCaps

The ignoreIrregularCaps parameter is used to detect improper capitalization of text that is being spell checked. For example, if you spell check the following with ignoreIrregularCaps set to false: "This is ok. this is not" then the second 'this' will be flagged as an error and the suggested replacement will be 'This'. JSpell defaults to 'false' for this parameter.

ignoreFirstCaps

When ignoreFirstCaps is set to false, the spell checker will flag the first word in the following sentence fragment as an error: "the quick brown fox" and it will suggest 'The' as a replacement for 'the'. If you are spell checking fields where it is not important that the first letter be capitalized then you may want to set this value to true. JSpell defaults to 'false' for this parameter.

ignoreNumbers

Normally, a number that is part of a word could be considered a typo, as in 'there9ore' instead of 'therefore'. When ignoreNumbers is set to false JSpell will treat that word as an error. However, there may be times that you want to ignore words that contain numbers especially in scientific settings, for example, if you set ignoreNumbers to true and spell check 'C6H12O6' then that will not be flagged as an error. JSpell defaults to 'false' for this parameter.

ignoreUpper

If ignoreUpper is false, then the following sentence fragment: "Thinking well is WIZE" will result in the word WIZE being flagged as an error with the suggested replacement being 'WISE'. If ignoreUpper is true, then the spell checker will not flag that word as an error. JSpell defaults to 'false' for this parameter.

ignoreDouble

If ignoreDouble is false, then the second word of a repeated word sequence will be flagged as an error. For example if ignoreDouble is false then in the following sentence "I went to to the park.", the second "to" will be flagged as an error. If ignoreDouble is true then that error will not be raised. JSpell defaults to 'false' for this parameter.

confirmAfterLearn

If disableLearn=false and confirmAfterLearn=true then the spell checker will present a dialog to the user asking them to confirm that they really want to add the word to their personal dictionary. JSpell defaults to 'false' for this parameter.

confirmAfterReplace

When a user encounters an error in their text and selects a replacement word from the suggested words list the spell checker will move on to the next error. However, if the user manually types a word as a replacement and that word is not in the suggested words list it could be an incorrect word. In order to trap this possibility you have the confirmAfterReplace parameter. If confirmAfterReplace is true, and the word that the user enters is not in the suggested words list, then a confirmation dialog will appear asking them to verify their intentions. If confirmAfterReplace is false then no such dialog is presented. Note, that JSpell will NOT revalidate the text that the user enters by going to the server unless the user explicitly performs another spell check. That is the reason for this parameter. JSpell defaults to 'true' for this parameter.

supplementalDictionary

The supplementalDictionary parameter allows you to keep a list of words in an ASCII text file on your server that will be ignored for purposes of the spell check process. You can have a different supplementalDictionary per user, per server, etc. The words in this list are not added to the dictionary for the purposes of suggestions, however, any words in that list will be ignored for purposes of performing the spell check. JSpell defaults to "" for this

parameter. The supplementalDictionary should be edited using a simple text editor such as Notepad on Windows or vi on UNIX. The format of the file is one word per line. The file should be saved in the same directory as your JSpell lexicon files *.jdx. These files are located based on the setting of the jspellLexicons parameter in your web.xml file. If you have not configured your web.xml file for use with JSpell then the JSpell lexicon files will have been automatically extracted during the installation process to a system folder, in which case you should perform a file search on your system for the location of the *.jdx files and place your supplemental dictionary file in that location.

spellCheckURL

The spellCheckURL is the path to the JSpell JSP file. It is specified as a relative path in most circumstances. JSpell defaults to “/jspellhtml/JSpell.jsp” for this parameter.

styleSheetURL

The styleSheetURL is the path to the JSpell Style Sheet. It is specified as a relative path in most circumstances. JSpell defaults to “/jspellhtml/jspell.css” for this parameter.

Using Language Features

By using the ‘language’ parameter in your JavaScript code you can control the language used when performing the spell check operation. The language parameter follows the pattern of “language (COUNTRY)”. The language value is the English word representing the language, and the country value is the two letter ISO code for the country. For example, “Spanish (ES)”.

Word Filtering

The JSpell IFrame Spell Checker supports the concept of a word filter that is maintained at the server. This word filter allows you to create a block list which can be used as a ‘naughty word filter’ or for other specialized tasks in your application. The word filter may be used by placing a file called **block.lst** in your JSpell dictionary directory.

The file is a standard text file which can be edited with any text based editor such as notepad or vi. Each word that you want to block should appear on its own line with an optional replacement word next to it.

[word to block] [replacement word]

When the user types a word in the block list it will be flagged as an error and asterisks will be shown as a suggested replacement. They will also not be able to use the Ignore, Ignore All, or Learn buttons. If a replacement word is specified then these buttons will be enabled. You can use the word filter functionality to expand acronyms used within your company, or to change internal secret project names to another value. An example block list is shown below:

NASA National_Aeronautics_and_Space_Administration
starfire classified_project_name
dummy
jerk
nincompoop

If a user tries to use the acronym NASA the suggested replacement will be "National Aeronautics and Space Administration". Note that underscores are replaced by spaces.

If a user tries to use 'dummy', 'jerk', or 'nincompoop' then the suggested replacements will be a series of asterisks.

Appendix A – Tomcat 3.x, 4.x and 5.x

If you already have Tomcat installed then you can install the JSpell Spell Checker by following these simple instructions.

Required Software

JSpell IFrame WAR file **jspelliframe2k4.war**

Installation

Copy the jspelliframe2k4.war file into the Tomcat webapps directory. For example, if you used the default settings and installed Tomcat 4.x on the Windows platform this will be: \Program Files\Apache Tomcat 4.0\webapps.

You **MUST** have the Java SDK installed. The Java JRE is not sufficient. If you are installing Java and Tomcat for the first time make sure to install the Java SDK first.

Check Tomcat Installation

Restart Tomcat

Access the test URL <http://localhost:8080/jspelliframe2k4/test.html> and verify that you can successfully perform a spell check. After you've verified the spell checker is working properly you can continue with the Getting Started guide and Adding JSpell to Your Web Page.

Appendix B – JRun

The JRun platform is one of the easiest platforms on which to install. You should be up and running in minutes.

Required Software

JSpell IFrame WAR file **jspelliframe2k4.war**

Installation

Start the JRun Management Console and login.

Select the correct server, usually 'default'.

Click on the J2EE Components link.

Click the 'Add' button in the Web Applications section.

Specify the path to the jspelliframe2k4.war file and click 'Deploy'.

Check JRun Installation

Access the test URL <http://localhost:8100/jspelliframe2k4/test.html> and verify that you can successfully perform a spell check. After you've verified the spell checker is working properly you can continue with the Getting Started guide and Adding JSpell to Your Web Page.