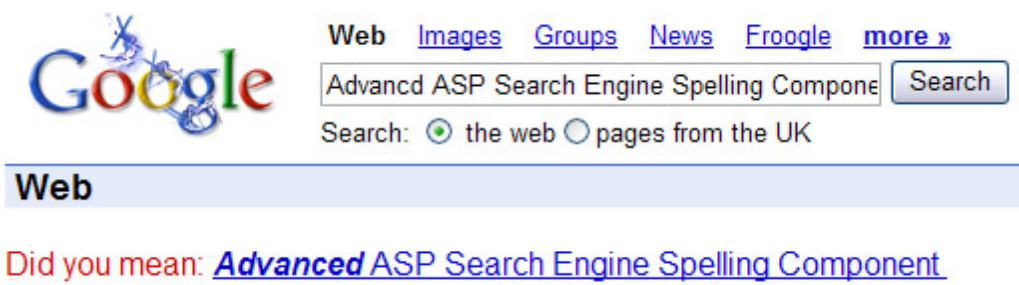


About ASPDidYouMean

ASPDidYouMean is the search string spell-check component for Microsoft® ASP 3 VBSCRIPT web applications. Its functionality resembles the "Did You Mean.." feature seen on Google and other major search engines.



ASPDidYouMean is an advanced script based component; there are no Active-X controls or DLLs to install on the server. In addition, no database is required. This means that it can be [installed](#) on almost all modern Microsoft® IIS web hosting environments - including Virtual/Shared Hosting.

ASPDidYouMean can be used in almost any ASP Web Application. It will work in a W3C and Accessible Web-site environment, and is [compatible](#) with all known web browsers.

ASPDidYouMean has been designed to work with modern Microsoft® web servers. The minimum requirement is an NT4 Server - although Windows Server 2000 or newer is advised.

ASPDidYouMean has [dictionaries](#) in many international languages including:

- English (Australia)
- English (Canada)
- English (International)
- English (UK)
- English (USA)
- French
- German
- Italian
- Dutch
- Portuguese
- Spanish
- Swedish
- Danish

Getting Started

Installation Basics

In brief, ASPDidYouMean is installed by copying the downloaded directory "ASPDidYouMean" to the root of your website.

Usage Basics

You can spell-check a search string and return the most likely result in only a few lines of code using the [DidYouMean Class](#) .

Its basic usage is shown below:

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%
    dim myDidYouMean, strSuggest
    Set myDidYouMean= new DidYouMean
    strSuggest= myDidYouMean.suggestionString(strQuery)
Set myDidYouMean = nothing %>
```

The variable `strSuggest` now contains the returned suggested search string. If there are no suggestions, then `strSuggest` will match the input string `strQuery` .

Dictionary Basics

You can [install additional language dictionaries](#) such as French, Spanish, German, Dutch, Portuguese and many Specialized English dictionaries. The dictionaries have the file extension .dic.

To install these dictionaries - first download them from www.ASPDidYouMean.com and place them in the "Dictionaries" directory within your ASPDidYouMean Directory.

The ASPDidYouMean Installation Process

ASPDidYouMean installation is simple. There are no DLLS, ActiveX controls or installers. This makes it very easy to use, even on most virtual hosting packages.

Installing the component into your web application onto your web server

ASPDidYouMean is installed by copying the entire ASPDidYouMean directory (in the ZIP archive you downloaded) into a directory of your website. The ASPDidYouMean folder should be copied into the root level of your website wherever possible. If you do not work with IIS sites, then the root level of your website is likely to be **c:\inetpub\wwwroot**

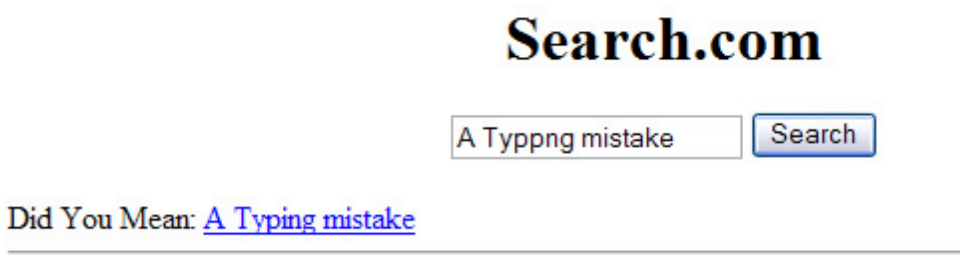
Normally, the best position for this is in the root level of your website, as this makes for a clean and simple usage.

If you decide to place the ASPDidYouMean component elsewhere, please update the **Installation_Path** variable in the [settings](#) file.

Testing your ASPSpellCheck Installation

Using your web browser, view the script **sample.asp** within the **ASPDidYouMean** directory. For example: <http://www.mysite.com/ASPDidYouMean/sample.asp>

You should be able to see a mock search engine dialog.
The "Did You Mean" Engine will take a few seconds to set its self up the first time you use it.



Compatibility

Browser Compatibility:

ASPDidYouMean is purely server-side. It has no known browser compatibility issues. It has been tested extensively, and sits perfectly in a W3C HTML or XHTML environment. It can be used in mobile applications; accessible websites...anywhere that ASP can be used.

Server Compatibility:

ASPDidYouMean has been designed to work with modern Microsoft® web servers. The minimum requirement is for NT4 Server - although Windows Server 2000 or newer is advised.

No special DLLS or EXEs or ACTIVEX controls need to be installed. In addition no database is required. This makes ASPSpellCheck ideal for many "Virtual Hosting" (shared hosting) environments.

To run the server must have IIS installed. The Application will need access to the following standard objects:

- *The Application object*
- *Scripting.FileSystemObject*
- *Scripting.Dictionary*
- *Msxml2.ServerXMLHTTP* .

These environmental factors are available in the majority of modern Microsoft hosting environments, without any special installations being performed.

Custom Dictionary

The Custom dictionary allows you to add a list of custom words to your spell checker's vocabulary.

This is useful for adding words specific to a business, application or website.

The Spell-Checker will look for updates to the custom dictionary every time a new user session is opened.

Modifying the Custom Dictionary

The custom dictionary file is by default "custom.txt" within the **ASPDidYouMean/Dictionaryes** folder within your website. This can be changed in the settings file.

To add words to your custom dictionary, simply place a list of up to 5000 words, each on separate lines into this file. Each word should only consist of ANSI characters.

For significantly improved performance, please sort these words in ANSI code order. If you do not have a tool to do this, then [EditPlus](#) is an excellent choice. For performance issues, it is best to remove any spaces from the file.

Example contents of custom.txt:

```
Aaron
Jacob
James
Julie
Steve
cybercom
intcom
marcoms
```

Setting the custom dictionary from a database

You can also build your custom dictionary from an ASP script.

The custom dictionary will have to be changed to in the settings file to specify a URL (e.g. "**custom.asp**").

Your ASP script should follow the same rules as per editing custom.txt by hand : up to 5000 words, each on separate lines, no spaces, in ANSI code order.

Installing Dictionaries for ASPDidYouMean

You can install additional language dictionaries to ASPDidYouMean such as:

- English (Australia)
- English (Canada)
- English (International)
- English (UK)
- English (USA)
- French
- German
- Italian
- Dutch
- Portuguese
- Spanish
- Swedish
- Danish

[Working with multiple languages](#) is easy using the DidYouMean Class.

You can download them from www.aspdidyoumean.com .

To install these dictionaries simply place them in the "**ASPDidYouMean/Dictionaries** " directory within your site.

You can also add a [custom dictionary](#) of words specific to your website using the **custom.txt** file

Banned Words List

The "Banned Words List" allows you to forbid ASPDidYouMean from returning a list of words. This allows you to forbid profanity, or annoying words that are similar to important key words.

Modifying the Banned Words List

The Banned Words List file is by default "banned.txt" within the **ASPDidYouMean/Dictionaryes** folder within your website. This can be changed in the settings file.

To add words to your custom dictionary, simply place a list of up to 5000 words, each on separate lines into this file. Each word should only consist of ANSI characters. The words can be in any order.

Example contents of banned.txt:

Rudeness
Impoliteness
AND
Bad
Sugestions

Setting the Banned Words List from a database

You can also build your custom dictionary from an ASP script. The Banned Words List will have to be changed to in the settings file to specify a URL (e.g. "**banned.asp**").

Your ASP script should output up to 5000 words, each on separate lines, with no spaces. In ASP VBScript, you can create a new line using the constants **vbNewLine** or **vbCrLf**.

The Settings File

ASPDidYouMean/settings.asp

```
<%
' .....
'           ASPDidYouMean Settings
' .....
Default_Language="ALL"
E.g. "English(International)" or "Deutch,Espagnol" or "ALL"
Installation_Path="/ASPDidYouMean" ' Absolute Path of the
ASPDidYouMean directory within the website
DictionaryPath="Dictionaries/" ' Relative Path to
Dictionaries
Custom_Word_List="custom.txt" ' List of custom words, each
on a new line, in Ascii order.. May be .txt or .asp
Banned_Word_List= "banned.txt" ' List of banned words, each
on a new line, May be .txt or .asp
Case_Sensitive=False ' Corrects Case Mistakes
' .....
'           End Of ASPDidYouMean Settings
' .....
%>
```

The settings file can be found in the ASPDidYouMean directory. It is called settings.inc. The settings file can be used to change the default settings for your search engine spell checker.

Default_Language

Default_Language sets the default dictionary which the DidYouMean engine will use.

- If you want it to use all of the dictionaries (.dic files) in your ASPDidYouMean/Dictionaries directory - set the value to **"ALL"**
- To choose a specific default language, choose a dictionary name. This will be the same as the name of any dictionary you have installed in the ASPDidYouMean/Dictionaries (do not use the ".dic" and the end though). **E.g. "English (USA)" etc.**
- To choose multiple dictionaries - use a comma separated list of dictionary names. **E.g. "English (USA),Svenska" etc.**

Instalation_Path

This is the absolute path to the ASPDidYouMean directory within your website. This is normally */ASPDidYouMean*. It is essential that this setting is correct.

DictionaryPath

DictionaryPath sets the path where the application looks for installed dictionaries. This path is relative to the ASPDidYouMean **instalation_Path** .

Custom_Word_List

Custom_Word_List sets the name of the [Custom Dictionary File](#) relative to the **DictionaryPath** . This is normally a text file, but can also be an ASP script to retrieve words from a database.

Banned_Word_List

Banned_Word_List sets the name of the [list of banned words](#) . This is normally a text file, but can also be an ASP script to retrieve words from a database.

Case_Sensitive

Case_Sensitive sets whether the ASPDidYouMean spellcheck engine will correct CaSe mistakes. Possible values are true or false.

Usage Basics

The most fundamental step to using the DidYouMean Class is to include the file: **/ASPDidYouMean/Include.inc** to your ASP Script.

Here are 2 Examples:

```
<!--#include file= "ASPSpellCheck/ASPSpellInclude.inc"-->
```

This uses a relative path to the file.

Or

```
<!--#include virtual= "/ASPSpellCheck/ASPSpellInclude.inc"-->
```

This uses an absolute path to the file, relative to the root directory of your website.

If you are unfamiliar with ASP includes, there is a good reference online at [w3schools](http://w3schools.com).

The *DidYouMean* Class

The *DidYouMean* Class allows you to easily return probable spellings of search strings. You can also set up the spell checker's language settings.

The file */ASPDidYouMean/Include.inc* must be included into your ASP script to use the *DidYouMean* Class.

Using *DidYouMean*

The code below shows an example the ASPSpellLink Class in action .

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%
    Set myDidYouMean= new DidYouMean
    strSuggest= myDidYouMean.suggestionString(strQuery)
    Set myDidYouMean = nothing
%>
```

DidYouMean.languages="strLang"

The ***languages*** property is used to decide which dictionary language(s) the search engine spell-check will use.

strLang represents the dictionary to be added. This will be the same as the name of any dictionary you have installed in the Dictionaries directory within your ASPSpellCheck Directory (do not use the ".dic" and the end though). You may use a comma separated list for multiple dictionaries, or "ALL" to add all installed dictionaries.

- If you want it to use all of the dictionaries (.dic files) in your ASPDidYouMean/Dictionaries directory - set the value to **"ALL"**
- To choose a specific default language, choose a dictionary name. This will be the same as the name of any dictionary you have installed in the ASPDidYouMean/Dictionaries (do not use the ".dic" and the end though). **E.g. "English (USA)" etc.**
- To choose multiple dictionaries - use a comma separated list of dictionary names. **E.g. "English (Canada),Francais" etc.**

N.B. These dictionaries must, of course, be [installed](#) to be used.

You may use up to 10 dictionaries simultaneously!

Example

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%
    dim myDidYouMean, strSuggest
    Set myDidYouMean= new DidYouMean
    myDidYouMean.languages="English (International)"
    strSuggest= myDidYouMean.suggestionString(strQuery)
    Set myDidYouMean = nothing
%>
```

DidYouMean.deleteTypos = true

The **deleteTypos** property is used to turn off the spelling suggestion algorithm. **deleteTypos** is Boolean - its possible values are **True** and **False** .

If **deleteTypos** is set true, then all typos will be deleted from the suggestions produced by the [suggestionString](#) and [suggestionStrings](#) functions.

Example

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%
    dim myDidYouMean, strSuggest
    Set myDidYouMean= new DidYouMean
    myDidYouMean.deleteTypos =true
    strSuggest= myDidYouMean.suggestionString(strQuery)
    Set myDidYouMean = nothing
%>
```

DidYouMean.hasSuggestions

After using the **DidYouMean** class to look for spelling suggestions, *hasSuggestions* will return **true** only if there are any valid suggestions, or **false** otherwise. *hasSuggestions* is *Read Only* .

hasSuggestions is automatically set after the [suggestionString](#), [suggestionStrings](#) or [suggestionMatrix](#) functions are called.

The *hasSuggestions* property is very useful for developers.

Example

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%
    dim myDidYouMean, strSuggest
    Set myDidYouMean= new DidYouMean
    strSuggest= myDidYouMean.suggestionString(strQuery)

    if myDidYouMean.hasSuggestions then
        response.write "DidYouMean:"&strSuggest
    end if

    Set myDidYouMean = nothing
%>
```

DidYouMean.suggestionString(strString)

The *DidYouMean.suggestionString* function returns the most probable suggestion string based on the input *strString*.

You can spell-check a search string and return the most likely result in only a few lines of code.

Basic usage is shown below:

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%
    dim myDidYouMean, strSuggest
    Set myDidYouMean= new DidYouMean
    strSuggest= myDidYouMean.suggestionString(strQuery)
Set myDidYouMean = nothing %>
```

strQuery represents the string to be spell-checked.

The variable *strSuggest* now contains the returned suggested search string.

Example usage in a Search Engine:

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%
dim strQuery, strSuggest, myDidYouMean
strQuery=Trim(request.QueryString("search"))
strSuggest=""
%>

<form action="" method="get">
    <div align= "center">
        <h1>Search.com</h1>
        <input type= "text" value="<%=strQuery%>" name="search" />
        <input type="submit" value="Search" name="submit" />
    </div>
</form>

<%
if strQuery<>""then
    Set myDidYouMean= new DidYouMean
    strSuggest= myDidYouMean.suggestionString(strQuery)

if myDidYouMean.hasSuggestions then
    response.write"DidYouMean:<a href=' "&request.ServerVariables("SC
RIPT_NAME") &"?search="&server.URLEncode(strSuggest) &" '
>"&strSuggest&"<a>"
end if

    Set myDidYouMean=nothing
end if
%>

<hr />
```

DidYouMean.suggestionStrings(strString,intMax)

The ***DidYouMean.suggestionStrings*** function returns an array of the most probable suggestion strings based on the input ***strString***.

strString is the string to be spell-checked.

intMax is the maximum number of results you wish to be returned.

The output is an Array of strings. The most probable strings come first in the array.

Basic usage is shown below:

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%

dim strQuery
arrSuggest=array()
strQuery ="A String to Spel Check"

    dim myDidYouMean
    Set myDidYouMean= new DidYouMean
    arrSuggest= myDidYouMean.suggestionStrings(strQuery,3)

if myDidYouMean.hasSuggestions then
    for each strSuggest in arrSuggest
        response.write"<br>DidYouMean : "&strSuggest
    next
end if

    Set myDidYouMean=nothing

%>
```

strQuery represent the string to be spell-checked.

The variable ***arrSuggest*** now contains an array of the suggested search string.

Example usage in a Search Engine:

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%
dim strQuery, arrSuggest, strSuggest, myDidYouMean
strQuery=Trim(request.QueryString("search"))
arrSuggest=array()
%>

<form action="" method="get">
    <div align= "center">
        <h1>Search.com</h1>
        <input type= "text" value="<%=strQuery%>" name="search" />
        <input type="submit" value="Search" name="submit" />
    </div>
</form>

<%

if strQuery<>" " then
    Set myDidYouMean= new DidYouMean
    arrSuggest= myDidYouMean.suggestionStrings(strQuery,3)

    if myDidYouMean.hasSuggestions then
        response.write "Did You Mean:<br>"
        for each strSuggest in arrSuggest
if strSuggest<>strQuery then
response.write"<a href=' "&request.ServerVariables("SCRIPT_NAME") &
```



```
"?search=" & server.URLEncode(strSuggest) & "' "& strSuggest & "<a><br>"
    end if
  next
end if

Set myDidYouMean=nothing
end if
```

```
%>
```

```
<hr />
```

DidYouMean.suggestionMatrix (strString,binFull)

The ***suggestionMatrix*** function returns a multi-dimensional array of the spelling suggestions for **each word** in a string.

This is useful for automating extra search results, and can also be used to suggest spelling mistakes in longer documents and AJAX applications.

strString is the string to be spell-checked.

binFull is a Boolean (true/false).

If **false**, the results do not contain suggestions for correctly spelled words.

If **true**, the results contain suggestions for all words in the string, even if spelled correctly.

The output is an irregular, multidimensional array (also called an "uneven set").

The first order of the array represents each word in search string. Note that punctuation and spaces also count as "words" (also known as tokens).

The second order of the array represents the suggestions for each word.

E.g.

suggestionMatrix (strString,binFull) (**0**) contains an array of suggestions for the **first** word

suggestionMatrix (strString,binFull) (**4**) contains an array of suggestions for the **fifth** word

suggestionMatrix (strString,binFull) (0) (**0**) contains the **first** suggestion for the **first word** as a string

suggestionMatrix (strString,binFull) (0) (**2**) contains the **third** suggestion for the **first word** as a string

Note that as you will not know the sizes of the returned array, we suggest you use **UBound** or **for each ... in** syntax to ensure that you do not attempt to access a non existent array member.

Example usage:

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->

<%
  Set myDidYouMean= new DidYouMean
  arrMatrix= myDidYouMean.SuggestionMatrix("a baddly spelt
stringg",true)
  Set myDidYouMean=nothing
%>
```

Example Usage in a Search Environment:

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->

<%
dim strQuery, binFull, arrSuggest, strSuggest,
arrMatrix,myDidYouMean

arrSuggest=array()
strQuery=Trim(request.QueryString("search"))
binFull=request.QueryString("binFull")<>" "
%>

<form action="" method="get">
  <div align="center">
    <h1>Search.com</h1>
    <input type="text" value="<%=strQuery%>" name="search" />
    <input type="submit" value="Search" name="submit" />
    <br />
    <label>Full Set of results
    <input type="checkbox" value="True" name="binFull" <% if
binFull then response.write "checked" %>/>
    </label>
  </div>
</form>
```

```

<%
if strQuery<>""then

Set myDidYouMean= new DidYouMean
arrMatrix= myDidYouMean.SuggestionMatrix(strQuery,binFull)

if myDidYouMean.hasSuggestions then
response.write "Possible Spellings of Words:<br>"
response.write "<table ><tr>"
for each arrSuggest in arrMatrix
response.write "<td valign=top>"
if ubound(arrSuggest)=0 then
response.write arrSuggest(0)
else
response.write "<select>"
for each strSuggest in arrSuggest
if strSuggest<>strQuery then
response.write "<option>"&strSuggest& "</option>"
end if
next
response.write "</select>"
end if
response.write "</td>"
next
response.write "</tr></table>"
end if

Set myDidYouMean=nothing
end if

%>
<hr />

```

Technical Support

We are happy to provide technical support to all users - even if you are only using the free trial.

We can be contacted at support@aspdidyoumean.com.

Also please read our FAQ at www.aspdidyoumean.com/support.asp .

Search.com

Did You Mean: [A Typing mistake](#)

Example usage in a Search Engine:

```
<!--#include virtual="/ASPDidYouMean/Include.inc"-->
<%
dim strQuery, strSuggest, myDidYouMean
strQuery=Trim(request.QueryString("search"))
strSuggest=""
%>

    <form action="" method= "get">
    <div align= "center">
        <h1>Search.com</h1>
        <input type= "text" value="<%=strQuery%>" name="search" />
        <input type="submit" value="Search" name="submit" />
    </div>
</form>

<%
if strQuery<>"" then
    Set myDidYouMean= new DidYouMean
    myDidYouMean.deleteTypos=false
    strSuggest= myDidYouMean.suggestionString(strQuery)
    if myDidYouMean.hasSuggestions then
response.write"DidYouMean:<a href=' "&request.ServerVariables("SCR
IPT_NAME") &"?search="&server.URLEncode(strSuggest) &" '
>"&strSuggest&"<a>"
Set myDidYouMean=nothing
end if

%>

<hr />
```

Notes:

The **suggestionString** function is used to acquire suggestions for the string **strQuery**.

Search.com

Did You Mean:

[bad typing today](#)

[bash typing today](#)

[bad topping today](#)

Example usage in a Search Engine:

```
<!--#include virtual= "/ASPDidYouMean/Include.inc"-->
<%
dim strQuery, arrSuggest, strSuggest, myDidYouMean
strQuery= Trim(request.QueryString("search"))
arrSuggest= array()
%>

<form action="" method="get">
  <div align="center">
    <h1>Search.com</h1>
    <input type= "text" value="<%=strQuery%>" name="search" />
    <input type="submit" value="Search" name = "submit"/>
  </div>
</form>

<%

if strQuery<>" " then
  Set myDidYouMean= new DidYouMean
  myDidYouMean.deleteTypos=false
  arrSuggest= myDidYouMean.suggestionStrings(strQuery,3)

  if myDidYouMean.hasSuggestions then
    response.write "Did You Mean:<br>"
    for each strSuggest in arrSuggest
      if strSuggest<>strQuery then
        response.write
"<a href=' "&request.ServerVariables("SCRIPT_NAME") &"?search="&ser
ver.URLEncode(strSuggest) &" '>"&strSuggest"<a><br>"
        end if
      next
    end if

    Set myDidYouMean=nothing

  end if

%>

<hr />
```

Notes:

In this example, the **suggestionStrings** function is used to acquire up to 3 suggestions for the string **strQuery**. The most probably strings come first in the array.



Example usage in a Search Engine:

```
<!--#include virtual= "/ASPDidYouMean/Include.inc"-->

<%
dim strQuery, strSuggest, myDidYouMean
strQuery=Trim(request.QueryString("search"))
strSuggest=""
%>

<form action="" method="get">
  <div align= "center">
    <h1>Search.com</h1>
    <input type= "text" value="<%=strQuery%>" name="search" />
    <label>
      <select name="strLang">

        <% if request.QueryString("strLang")<>" " then %>
        <% '''PersistentStatefortheLanguageSelectMenu ''' %>
        <option
value="<%=request.QueryString("strLang")%>"><%=request.QueryString("strLang")%></option>
        <option value="<%=request.QueryString("strLang")%>"
>-----</option>
        <% end if%>
        <option value="English (International)" >English
(International)</option>
        <option value="Deutch" >Deutch</option>
        <option value="Dansk" >Dansk</option>
        <option value="English (Canada)" >English (Canada)</option>
        <option value="English (International)" >English
(International)</option>
        <option value="English (Australia)" >English
(Australia)</option>
        <option value="English (UK)" >English (UK)</option>
        <option value="Espagnol" >Espagnol</option>
        <option value="Francais" >Francais</option>
        <option value="Nederlands" >Nederlands</option>
        <option value="Italiano" >Italiano</option>
        <option value="Portugues" >Portugues</option>
        <option value="Svenska" >Svenska</option>
      </select>

    </label>
    <input type="submit" value="Search" name="submit" />
  </div>
</form>
```



```

<%

if strQuery<>" " then
    Set myDidYouMean= new DidYouMean
    myDidYouMean.Languages=request.QueryString("strLang")
    strSuggest= myDidYouMean.suggestionString(strQuery)

    if myDidYouMean.hasSuggestions then

        response.write request.QueryString("strLang") &"-DidYouMean:<a href=' "&request.ServerVariables("SCRIPT_NAME") &"?search="
        &strSuggest&"'>"&strSuggest&"<a>"

    end if

    Set myDidYouMean= nothing

end if

%>

<hr />

```

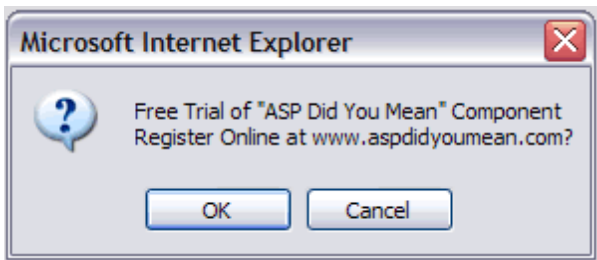
Notes:

myDidYouMean.Languages values match the names of the downloaded dictionaries. You must [install](#) the relevant dictionaries first.

Free Trial

There is a freely downloadable trial edition of ASPDidYouMean which you may use to evaluate the software.
It is licensed for evaluation only, and not for commercial or public use.

The trial edition presents a "Registration Reminder" from time to time. This may occur within your returned suggestion results - or in a pop up dialog.



This is removed by purchasing and installing a full version of ASPDidYouMean.

Registering

The trail is not licensed for commercial or public use!

To purchase a suitable license for ASPDidYouMean please visit
<http://www.aspdidyoumean.com/purchase.asp>.

License upgrades are also available at: <http://www.aspdidyoumean.com/upgrade.asp>.

License

ASPDidYouMean.com hereby grants you a non-exclusive license to the ASPDidYouMean Software Component (the Software). By downloading or using the Software, the Licensee agrees not to utilize the software in a manner which is disparaging to ASPDidYouMean.com, and not to rent, lease or otherwise transfer rights to the Software.

The Licensee agrees that no attempt will be made by the Licensee or associated parties to translate, reverse engineer, modify, decompile, disassemble or distribute the Software.

License terms are offered on the following terms, as purchased. If no purchase or insufficient purchase has been made then the Free Trial License terms apply.

Free Trial License - Allows the use of the TRIAL VERSION of the software for private evaluation purposes only. The software should not be used for public or commercial use of any kind.

Single Website License - Allows the use of the software on any number of web / intranet pages belonging to a single domain.

Single Server License - This license allows the usage of the software on any number of web / intranet pages for any number of domains located on a single server.

Enterprise License - This license allows the usage of the software on any number of web / intranet pages for any number of domains located on any number of servers belonging to one company or organization.

UPGRADES

If a new release of the software is produced within 12 months from the date of purchase then you will be entitled to a free upgrade. This license does not grant you any right to any enhancement or update beyond the initial 12 month period, commencing from the date of purchase.

COPYRIGHT

Title, ownership rights, and intellectual property rights in and to the Software shall remain with ASPDidYouMean.com. The Software is protected by the international copyright laws. Title, ownership rights, and intellectual property rights in and to the content accessed through the Software is the property of the applicable content owner and may be protected by applicable copyright or other law. This License gives you no rights to such content.

LIMITATION OF LIABILITY.

THIS SOFTWARE IS PROVIDED "AS IS," WITHOUT A WARRANTY OF ANY KIND. ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. ASPDidYouMean.COM AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THE SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL ASPDidYouMean.COM OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF

OR INABILITY TO USE SOFTWARE, EVEN IF ASPDidYouMean.COM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

MISCELLANEOUS

This software is not designed or intended for use in on-line control of aircraft, air traffic, aircraft navigation or aircraft communications; or in the design, construction, operation or maintenance of any nuclear facility. Licensee represents and warrants that it will not use or redistribute the Software for such purposes.