NSSpellChecker

Inherits From:	NSObject
Conforms To:	NSObject (NSObject)
Declared In:	AppKit/NSSpellChecker.h

Class Description

The NSSpellChecker class gives any application an interface to the OpenStep spell-checking service. To handle all its spell checking, an application needs only one instance of NSSpellChecker. It provides a panel in which the user can specify decisions about words that are suspect. To check the spelling of a piece of text, the application:

- Includes in its user interface a menu item (or a button or command) by which the user will request spell checking.
- Makes the text available by way of an NSString object.
- Creates an instance of the NSSpellChecker class and sends it a **checkSpellingOfString:startingAt:** message.

For example, you might use the following statement to create a spell checker:

```
range = [[NSSpellChecker sharedSpellChecker] checkSpellingOfString:aString
startingAt:0];
```

The **checkSpellingOfString:startingAt:** method checks the spelling of the words in the specified string beginning at the specified offset (this example uses 0 to start at the beginning of the string) until it finds a word that is misspelled. Then it returns an NSRange to indicate the location of the misspelled word.

In a graphical application, whenever a misspelled word is found, you'll probably want to highlight the word in the document, using the NSRange that **checkSpellingOfString:startingAt:** returned to determine the text to highlight. Then you should show the misspelled word in the Spelling panel's misspelled-word field by calling **updateSpellingPanelWithMisspelledWord:**. If **checkSpellingOfString:startingAt:** does not find a misspelled word, you should call **updateSpellingPanelWithMisspelledWord:** with the empty string. This causes the system to beep, letting the user know that the spell check is complete and no misspelled words were found. None of these steps is required, but if you do one, you should do them all.

The object that provides the string being checked should adopt the following protocols:

NSChangeSpelling	A message in this protocol (**changeSpelling:**) is sent down the responder chain when the user presses the Correct button.
NSIgnoreMisspelledWords	When the object being checked responds to this protocol, the spell server keeps a list of words that are acceptable in the document and enables the Ignore button in the Spelling panel.

The application may choose to split a document's text into segments and check them separately. This will be necessary when the text has segments in different languages. Spell checking is invoked for one language at a time, so a document that contains portions in three languages will require at least three checks.

Dictionaries and Word Lists

The process of checking spelling makes use of three references:

- A dictionary registered with the system's spell-checking service. When the Spelling panel first appears, by default it shows the dictionary for the user's preferred language. The user may select a different dictionary from the list in the Spelling panel.
- The user's "learn" list of correctly-spelled words in the current language. The NSSpellChecker updates the list when the user presses the Learn or Forget buttons in the Spelling panel.
- The document's list of words to be ignored while checking it (if the first responder conforms to the NSIgnoreMisspelledWords protocol). The NSSpellChecker updates its copy of this list when the user presses the Ignore button in the Spelling panel.

A word is considered to be misspelled if none of these three accepts it.

Matching a List of Ignored Words with the Document It Belongs To

The NSString being checked isn't the same as the document. In the course of processing a document, an application might run several checks based on different parts or different versions of the text. But they'd all belong to the same document. The NSSpellChecker keeps a separate "ignored words" list for each document that it checks. To help match "ignored words" lists to documents, you should call **uniqueSpellDocumentTag** once for each document. This method returns a unique arbitrary integer that will serve to distinguish one document from the others being checked and to match each "ignored words" list to a document. When searching for misspelled words, pass the tag as the fourth argument of **checkSpellingOfString:startingAt:language:wrap:inSpellDocumentWithTag:wordCount:**. (The convenience method **checkSpellingOfString:startingAt:** takes no tag. This method is suitable when the first responder does not conform to the NSIgnoreMisspelledWords protocol.)

When the application saves a document, it may choose to retrieve the "ignored words" list and save it along with the document. To get back the right list, it must send the NSSpellChecker an **ignoredWordsInSpellDocumentWithTag:** message. When the application has closed a document, it should notify the NSSpellChecker that the document's "ignored words" list can now be discarded, by

sending it a **closeSpellDocumentWithTag:** message. When the application reopens the document, it should restore the "ignored words" list with the message **setIgnoredWords:inSpellDocumentWithTag:**.

Method Types

Getting the spell checker	+ sharedSpellChecker + sharedSpellCheckerExists
Managing the spelling panel	– setAccessoryView: – accessoryView – spellingPanel
Checking spelling	– countWordsInString:language: – checkSpellingOfString:startingAt: – checkSpellingOfString:startingAt:language:wrap:inSpellDocumentWithTag:wordCount:
Setting the language	– setLanguage: – language
Managing the Spelling Process	+ uniqueSpellDocumentTag – closeSpellDocumentWithTag: – ignoreWord:inSpellDocumentWithTag: – setIgnoredWordsInSpellDocumentWithTag: – ignoredWordsInSpellDocumentWithTag: – setWordFieldStringValue: – updateSpellingPanelWithMisspelledWord:

Class Methods

sharedSpellChecker

+ (NSSpellChecker *)**sharedSpellChecker**

Returns the NSSpellChecker (one per application).

See also: + **sharedSpellCheckerExists**

sharedSpellCheckerExists

+ (BOOL)**sharedSpellCheckerExists**

Returns whether the application's NSSpellChecker has already been created.

See also: + **sharedSpellChecker**

uniqueSpellDocumentTag

+ (int)**uniqueSpellDocumentTag**

Returns a guaranteed unique tag to use as the spell-document tag for a document. Use this method to generate tags to avoid collisions with other objects that can be spell-checked.

Instance Methods

accessoryView

– (NSView *)**accessoryView**

Returns the Spelling panel's accessory NSView object.

See also: – **setAccessoryView:**

checkSpellingOfString:startingAt:

– (NSRange)**checkSpellingOfString:**(NSString *)*stringToCheck*
startingAt:(int)*startingOffset*

Starts the search for a misspelled word in *stringToCheck* starting at *startingOffset* within the string object. Returns the range of the first misspelled word. Wrapping occurs but no ignored-words dictionary is used.

checkSpellingOfString:startingAt:language:wrap:inSpellDocumentWithTag:wordCount:

– (NSRange)**checkSpellingOfString:**(NSString *)*stringToCheck*
startingAt:(int)*startingOffset*
language:(NSString *)*language*
wrap:(BOOL)*wrapFlag*
inSpellDocumentWithTag:(int)*tag*
wordCount:(int *)*wordCount*

Starts the search for a misspelled word in *stringToCheck* starting at *startingOffset* within the string object. Returns the range of the first misspelled word and optionally the word count by reference. *tag* is an identifier unique within the application used to inform the spell check which document (actually, a dictionary) of ignored words to use. *wrapFlag* determines whether spell checking continues at the beginning of the string when the end is reached. *language* is the language used in the string. If *language* is the empty string, the current selection in the Spelling panel's pop-up menu is used.

closeSpellDocumentWithTag:

– (void)**closeSpellDocumentWithTag:**(int)*tag*

Notifies the spell checker that the user has finished with the ignored-word document identified by *tag*, causing it to throw that dictionary away.

countWordsInString:language:

– (int)**countWordsInString:**(NSString *)*stringToCount*
language:(NSString *)*language*

Returns the number of words in *string*. The *language* argument specifies the language used in the string. If *language* is the empty string, the current selection in the Spelling panel's pop-up menu is used.

ignoreWord:inSpellDocumentWithTag:

– (void)**ignoreWord:**(NSString *)*wordToIgnore* **inSpellDocumentWithTag:**(int)*tag*

Instructs the spell checker to ignore all future occurrences of *wordToIgnore* in the document identified by *tag*. You should call this method from within your implementation of the NSIgnoreMisspelledWords protocol's **ignoreSpelling:** method.

ignoredWordsInSpellDocumentWithTag:

– (NSArray *)**ignoredWordsInSpellDocumentWithTag:**(int)*tag*

Returns the array of ignored words for a document identified by *tag*. Invoke this before **closeSpellDocument:** if you want to store the ignored words.

See also: – **setIgnoredWords:inSpellDocumentWithTag:**

language

– (NSString *)**language**

Returns the current language used in spell-checking.

See also: – **setLanguage:**

setAccessoryView:

– (void)**setAccessoryView:**(NSView *)*aView*

Makes an NSView object an accessory of the Spelling panel by making it a subview of the panel's content view. This method posts the notification NSWindowDidResizeNotification with the Spelling panel object to the default notification center.

See also: – **accessoryView**

setIgnoredWords:inSpellDocumentWithTag:

– (void)**setIgnoredWords:**(NSArray *)*words* **inSpellDocumentWithTag:**(int)*tag*

Initializes the ignored-words document (i.e., dictionary identified by *tag* with *someWords)*, an array of words to ignore.

See also: – **ignoredWordsInSpellDocumentWithTag:**

setLanguage:

– (BOOL)**setLanguage:**(NSString *)*language*

Sets the language to use in spell-checking to *aLanguage*. Returns whether the Language pop-up list in the Spelling panel lists *aLanguage*.

See also: – **language**

setWordFieldStringValue:

– (void)**setWordFieldStringValue:**(NSString *)*aString*

Sets the string that appears in the misspelled word field, using the string object *aString*.

spellingPanel

– (NSPanel *)**spellingPanel**

Returns the spell checker's panel.

updateSpellingPanelWithMisspelledWord:

– (void)**updateSpellingPanelWithMisspelledWord:**(NSString *)*word*

Causes the spell checker to update the Spelling panel's misspelled-word field to reflect *word*. You are responsible for highlighting *word* in the document and for extracting it from the document using the range returned by the **checkSpelling:...** methods. Pass the empty string as *word* to have the system beep, indicating no misspelled words were found.