Text Input and Output

Cocoa > Text & Fonts

2004-02-10

Apple Inc.
© 1997, 2004 Apple Computer, Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Apple, the Apple logo, and Cocoa are trademarks of Apple Inc., registered in the United States and other countries.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

Introduction to Text Input and Output 7

Who Should Read This Document 7
Organization of This Document 7
See Also 7

Text Input and Output Formats 9

Using Text Input and Output 11

Reading Text From a File 11
Writing To a Text File 12

Document Revision History 15

Index 17

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

Tables and Listings

Text Input and Output Formats 9

Table 1 Text system input and output formats 9

Using Text Input and Output 11

Listing 1 Determining the text format 11
Listing 2 Reading the text from a file 11
Listing 3 Writing text to a file 12

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

Introduction to Text Input and Output

Text Input and Output describes how to use the Cocoa's text system's built-in support for storing text in files.

Who Should Read This Document

You should read this document if you need to understand how to read and write text files from Cocoa text objects.

Organization of This Document

This document includes the following articles:

- "Text Input and Output Formats" (page 9) describes the file formats that the text system can read and write.
- "Using Text Input and Output" (page 11) presents code fragments that illustrate how to read and write text files from an NSTextView object.

See Also

For more information, refer to the following documents:

- *Attributed Strings Programming Guide* describes the NSAttributedString objects that manage sets of text attributes, such as font and kerning, associated with character strings in rich text.
- *Text Attributes* describes the text-related attributes maintained by the Cocoa text system.
- *Text Attachment Programming Topics for Cocoa* discusses the way the text system handles text attachments stored in the RTFD file format.

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

See Also

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

Text Input and Output Formats

The text system provides a convenient interface to the file system that enables you to read, display, and write files in the text formats described in Table 1.

Table 1 Text system input and output formats

Format	Description
Plain Text	Characters unaccompanied by attribute information.
Rich Text Format (RTF)	Character and attribute information expressed in Rich Text Format (RTF). See the *Rich Text Format (RTF) Specification* by Microsoft Corporation for more information.
Rich Text Format Directory (RTFD)	Character and attribute information expressed in Rich Text Format and stored in a directory along with the images and other attachments that are embedded in the text.

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

Using Text Input and Output

This article shows you how to read text from a file into an NSTextView and how to write text to a file from an NSTextView.

Reading Text From a File

To read text from a file, you first must determine the format of the text. To illustrate how this is done, consider an object of the custom class Controller. A Controller object is responsible for opening and closing files. It stores an NSTextView object and declares a variable that records the format of the text that it reads in. Here's the interface declaration:

Listing 1 Determining the text format

```
#import <AppKit/AppKit.h>
typedef enum _dataFormat {
    Unknown = 0,
    PlainText = 1,
    RichText = 2,
    RTFD = 3,
} DataFormat;

@interface Controller : NSObject
{
    DataFormat theFormat;
    NSTextView *theTextView;
}
- (void)openFile:(id)sender;
- (void)saveFile:(id)sender;
@end
```

Now, the Controller object's `openFile:` method can be implemented like this:

Listing 2 Reading the text from a file

```
- (void)openFile:(id)sender
{
    NSOpenPanel *panel = [NSOpenPanel openPanel];
    if ([panel runModal] == NSOKButton) {
        NSString *fileName = [panel filename];
        if ([[fileName pathExtension] isEqualToString:@"rtfd"]) {
            [theTextView readRTFDFromFile:fileName];
            theFormat = RTFD;}
        else if([[[fileName pathExtension]isEqualToString:@"rtf"]) {
            NSData *rtfData = [NSData dataWithContentsOfFile:fileName];
            [theTextView replaceRange:
            NSMakeRange(0, [[theTextView string] length]) withRTF:rtfData];
            theFormat = RichText;
```


2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

```
        } else {
            NSString *fileContents = [NSString
            stringWithContentsOfFile:fileName];
            [theTextView setString:fileContents range:NSMakeRange(0,
                [[theTextView string] length])];
            theFormat = PlainText;
        }
    }

    return;
}
```

The `openFile:` method checks the file name returned by the Open panel for the extensions "`rtfd`" or "`rtf`" and uses the appropriate means of loading data for each type. Files having any other extension are loaded as plain text. Note that the Controller object records the format of the loaded data in its `theFormat` variable. This information is used to determine how the file should be saved, as discussed in the next section.

Writing To a Text File

Depending on the format of an NSTextView object's text, you use slightly different approaches to write the text to a file. For plain text, you extract the contents of the NSTextView as an NSString object and use the NSString method `writeToFile:atomically:` to write the data to disk. RTF text is treated similarly, except that the contents is extracted as an NSData object. Easiest of all is RTFD data, which the NSTextView itself knows how to write to a file:

Listing 3 Writing text to a file

```
- (void)saveFile:(id)sender
{
    NSSavePanel *panel = [NSSavePanel savePanel];
    switch (theFormat) {
        case PlainText:
            [panel setRequiredFileType:@""];
            if ([panel runModal] == NSOKButton) {
                [[theTextView string] writeToFile:[panel filename]
                atomically:YES];
            }
            break;
        case RichText:
            [panel setRequiredFileType:@"rtf"];
            if ([panel runModal] == NSOKButton) {
                [[theTextView RTFFromRange:
                    NSMakeRange(0, [[theTextView string] length])]
                    writeToFile:[panel filename] atomically:YES];
            }
            break;
        case RTFD:
            [panel setRequiredFileType:@"rtfd"];
            if ([panel runModal] == NSOKButton) {
                [theTextView writeRTFDToFile:[panel filename]
                    atomically:YES];
            }
            break;
        default:
```


2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

```
            NSRunAlertPanel(@"Save Error",
                @"Couldn't save file (unknown data format).\n",
                nil, nil, nil);
        break;
    }
    return;
}
```


2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

Document Revision History

This table describes the changes to *Text Input and Output*.

Date	Notes
2004-02-10	Renamed topic from *Text I/O* to *Text Input and Output*. Revised introduction and added an index.
2002-11-12	Revision history was added to existing topic. It will be used to record changes to the content of the topic.

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.

Index

A

attachments, text 9

D

`dataWithContentsOfFile:` method 11

F

file formats
  determining 11
  understood by the text system 9

N

NSData class 12
NSString class 12
NSTextView class
  reading and writing files 11

P

plain text 9, 12

R

rich text format (RTF) 9, 12
rich text format directory (RTFD) 9, 12
RTF. *See* rich text format
RTFD. *See* rich text format directory

T

text
  attachments 9
  plain 9, 12
  reading from a file 11
  rich 9, 12
  rich, with attachments 9
  writing to a file 12

W

`writeToFile:atomically:` method 12

2004-02-10 | © 1997, 2004 Apple Computer, Inc. All Rights Reserved.