WRITING APPLE EVENTS SCRIPTS FOR QUARKXPRESS

QuarkXPress™

The choice for publishing software worldwide.

IN NO EVENT WILL APPLE, ITS DIRECTORS, OFFICERS, EMPLOYEES OR AGENTS BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, INCIDENTAL OR INDIRECT DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, AND THE LIKE) ARISING OUT OF THE USE OR INABILITY TO USE THE APPLE SOFTWARE EVEN IF APPLE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE MAY NOT APPLY TO YOU. APPLE'S LIABILITY TO YOU FOR ACTUAL DAMAGES FROM ANY CAUSE WHATSOEVER, AND REGARDLESS OF THE FORM OF THE ACTION (WHETHER IN CONTRACT, TORT (INCLUDING NEGLIGENCE), PRODUCT LIABILITY OR OTHERWISE), WILL BE LIMITED TO $50.

Table of Contents

Introduction

This Guide provides information about Apple events scripting with QuarkXPress™, QuarkXPress Passport™, and single-language versions of QuarkXPress (Japanese, Korean, Simplified Chinese, and Traditional Chinese). Throughout, the concepts discussed apply to all versions of QuarkXPress; Apple event properties that are specific to single-language versions are identified as such.

Scripting Overview

This chapter provides an overview of Apple events scripting with QuarkXPress. First, it introduces the concepts and terminology involved, including: the Object Model, Objects, Object Hierarchy, Object References, Reference Forms, Insertion Points in the Hierarchy, Events, Suites, and the QuarkXPress Object Hierarchy. You should understand these concepts and terms before you attempt to write scripts for QuarkXPress.

The second part of this chapter notes the primary syntax differences between UserLand Frontier and AppleScript, and provides information on optimizing the performance of scripts.

About this Guide

This document is for people who are ready to create scripts that communicate with QuarkXPress 4.1 or later. If you need assistance writing or debugging scripts, consult the documentation provided with your scripting application and the scripting forums on the online services. Quark also provides additional online scripting support (forum: http://www.quark.com/forums, and e-mail: scriptsupport@quark.com).

To learn more about Apple events scripting, you may want to purchase a third-party book such as *The Tao of AppleScript, AppleScript for Dummies, or The Complete AppleScript Book*. Many other third-party books exist, and of those, many include scripting software.

What You Need

To write scripts, you need the following:

- Mac OS 7.1 or greater.
- A scripting application such as UserLand Frontier or AppleScript.
- The documentation (included with your scripting application) that teaches you the scripting language. You should familiarize yourself with the scripting language before attempting to write scripts for QuarkXPress.
- A basic understanding of programming (including concepts such as loops, conditional processing, if-then-else constructs, and variables) gained through writing HyperTalk scripts or macros, or working in programming languages such as C, BASIC, or Pascal.

A basic understanding of programming is optional. AppleScript is a relatively easy language to learn. You can begin by writing basic scripts, and add to them when your understanding of the language has developed to a more advanced level.

What this Guide Provides

This guide provides background information on Apple events, an analysis of a sample script, and specific information about writing scripts for QuarkXPress. If you are unfamiliar with Apple events terminology, read the chapters sequentially and refer to the Glossary as necessary.

Introduction to Apple Events

Apple events is a Mac OS feature that allows interapplication communication on a local system or across a network. Applications communicate through standard Apple events messages that give instructions, respond to instructions, and send or receive data. The terminology for Apple events messages is listed in the *Apple event Registry* for each application, which is maintained by Apple.

Scripts

Apple events can be generated by scripts, which are a series of statements sent to applications that tell them to do a series of tasks. The scripting language is provided by scripting software such as UserLand Frontier or AppleScript. Scripts combine the scripting language syntax with the standard Apple events terminology defined in the *Apple event Registry.*

System-level Scripting

Scripting software is developed specifically for script writing. It is more powerful than scripting systems built into applications because it allows you to use one scripting language to write scripts for any application that supports Apple events. You can even write scripts to link Apple events-aware applications.

You can do everything from simple formatting tasks to complex database publishing with scripts. For example, you might have a script that alphabetizes paragraphs or formats lead paragraphs. Or, you can write a script to merge addresses from a database into a QuarkXPress letter template. It's even possible to produce an entire catalog automatically by linking QuarkXPress to a database of pictures and text.

The Object Model

The Apple events Object Model is a message protocol that allows Mac OS applications to communicate. Messages built according to the Object Model consist of events, objects and, potentially, data. Objects are distinct items in an application, such as a text box. Events are the actions objects that are capable of performing.

If you're familiar with QuarkXPress, you understand that an application is composed of objects. QuarkXPress documents contain pages, pages contain text boxes, text boxes contain text, and text has various styles associated with it. Each object has specific capabilities. For example, a text box can be moved, resized, copied, and linked to other boxes. A text box has item specifications that can be changed (such as background color, number of columns, and text inset) and it can contain formatted text.

Objects

An object is a distinct item in an application that can be manipulated by an Apple event. QuarkXPress users are familiar with objects such as documents, pages, text boxes, picture boxes, and lines. Objects are defined according to their class, properties, elements, and the events they can respond to.

- **Object Class:** Objects that share specific characteristics are categorized into object classes. For example, all documents belong to the "document" object class.
- **Properties:** Properties are the characteristics shared by objects in the same object class. For example, the object class for documents has properties such as file path, name, print setup, and version.
- **Elements:** Elements are the objects directly accessible from within another object. For example, a page is an element of a document.
- **Events:** Events are the actions an object is capable of performing. Objects in the same object class respond to the same events. For example, the "set" event can be used to change the tool mode of all documents.

Object Hierarchy

The Apple events Object Hierarchy is based on the simple concept of placing things inside other things. An application's object hierarchy usually consists of objects such as windows, documents, boxes, and contents. A specific hierarchy in QuarkXPress might include a document that contains a page. The page contains a text box and the text box contains a story. The story contains paragraphs, and the paragraphs contain lines. The lines contain words and the words contain characters. Characters are at the end of the hierarchy because they can't contain anything.

Objects that enclose other objects are referred to as *containers*. Objects that are enclosed by other objects are referred to as *elements*. For example, a document is a container for a page; the page is an element of the document.

Object References

An Apple event message must identify a specific object in an application to communicate. Objects are identified by a *reference*. For example, the message might reference the second text box on the first page. The reference first identifies the container (the page) enclosing the object (the text box) you're specifying. It then uses a reference form to separate a specific object (the second text box) from all possible objects in the container.

Reference Forms

Objects in QuarkXPress can be referred to by five reference forms: index, name, range, relative position, or test. See the "Apple Events Terminology" portion of the "Reference Materials" section of this document for an example of how to use each reference form.

• **Index:** Used to identify an ordered element in a container with an integer number (for example, the first text box on a page).

Windows, documents, text boxes, and picture boxes are numbered from front to back. The active window or document is always number [1]; the frontmost picture box or text box in the document is always number [1]. (The frontmost picture box or text box may change as users manipulate and create other boxes.) Pages are numbered according to their absolute page numbers rather than section page numbers.

As you create and insert objects in the hierarchy, the index reference form for existing objects may change. For example, if you insert a text box before "text box 1," then "text box 1" becomes "text box 2."

• **Name:** Used to identify objects that are named with a text string (for example, a document named "Ad Layout" by a user).

• **Range:** Used to identify a range of objects (for example, text boxes three through five).

• **Relative Position:** Used to identify objects that are before or after other objects (for example, the text box before the last text box on the page).

• **Test:** Used to identify objects that meet certain conditions, (for example, the first text box with a red background).

Insertion Points in the Hierarchy

An insertion point specifies where to place an object within the container hierarchy. See the "Apple Events Terminology" portion of the "Reference Materials" section of this document for an example of how to use each insertion point.

• **Beginning:** Used to insert or create an object at the beginning of the specified container (for example, to create a text box at the beginning of page one).

• **Ending:** Used to insert or create an object at the end of the specified container (for example, to create a page at the end of a document).

• **After:** Used to insert or create an object after a specified object (for example, to move the first page of a document after the fourth page).

• **Before:** Used to insert or create an object before the specified object (for example, to move the last page of a document before the first page).

• **Replace:** Used to replace the specified object with a new object (for example, to replace one text box with another text box).

Events

Events are the actions an object is capable of performing. In an English sentence, an event is comparable to a verb and an object is comparable to a noun. Events are used to tell objects what to do. QuarkXPress uses most of the standard events defined by Apple.

Suites

Groups of events and objects that relate to a similar purpose are arranged into Suites. The Required Suite, Standard Suite, and Miscellaneous Suite include the events and objects that most Macintosh applications support. In addition, events and objects specific to QuarkXPress are defined in the QuarkXPress Suite, Text Suite, and the QuarkXPress Ancillary Objects suite.

QuarkXPress supports the events and objects in the Required, Standard, Miscellaneous, and QuarkXPress Suites, as well as objects in the Text Suite and the QuarkXPress Ancillary Objects suite. An object can respond to events from a variety of suites, and events can apply to objects from a variety of suites. For example, objects in the QuarkXPress Suite are generally manipulated using events in the Standard Suite.

Required Suite

- **Events:** All of the required events are handled by events in the Standard suite.
- **Objects:** The Required Suite does not define any objects.

Standard Suite

- **Events:** The Standard Suite events are common to most applications: “clone/duplicate,” “close,” “count,” “create/make,” “data size,” “delete,” “exists,” “get,” “get as,” “move,” “open,” “print,” “save,” “set,” and “quit.”
- **Objects:** The Standard Suite objects are common to most applications, for example: “application,” “document,” “file,” “insertion point,” “selection,” and “window.”

Text Suite

- **Events:** The Text Suite does not define any events.
- **Objects:** The Text Suite objects are the text-related objects common to most applications, for example: “character,” “line,” “paragraph,” “story,” “text,” “text style range,” and “word.”

Miscellaneous Suite

- **Events:** The Miscellaneous Suite events are related to the clipboard and other menu-related functions: “copy,” “cut,” “do script,” “paste,” “revert,” “show,” and “uniform.”
- **Objects:** The only objects in the Miscellaneous Suite are those related to menus, for example: “menu” and “menu item.”

QuarkXPress Suite

• **Events:** The QuarkXPress Suite includes two events: “coerce,” and “do updates.”

• **Objects:** The QuarkXPress Suite objects are specific to the application, for example: “character spec,” “color spec,”“graphic box,” “group box,” “horizontal guide,” “image,” “line box,” “master document,” “page,” “picture box,” “spread,” “style spec,” “text box,” “text style range,” “user box,” and “vertical guide.”

QuarkXPress Ancillary Objects

• **Events:** The QuarkXPress Ancillary Objects suite does not include any events.

• **Objects:** The QuarkXPress Ancillary Objects suite contains objects that are used for inheritance, returned as records, or data types, for example: “base class,” “box properties,” “character attributes,” “containing box properties,” “fixed point,” “fixed rectangle,” “font record,” “non containing box properties,” “paragraph properties,” “print setup record,” “rule record,” “tab record,” “text path properties.”

The “Definitions and Examples” section of this document provides definitions and sample syntax for each event QuarkXPress supports. The AppleScript Dictionary defines each event and object QuarkXPress supports. The following objects in the hierarchy are not familiar QuarkXPress terms. Their properties are defined fully in the AppleScript Dictionary.

• Color systems: The color models that QuarkXPress supports.

• Contour: A continuous path. Shape paths, Clipping paths, and Runaround paths are made up of contours. For instance, if you had a bezier element in the shape of a donut, you would have two contours. One contour for the outside shape and one contour for the inside shape.

• Default document: The object that contains all default document settings including colors, style sheets, H&Js, document settings specified in the New Document dialog box, and all document related preferences.

• Delimit item: Each character has an associated delimit item that QuarkXPress uses to determine whether a character should be part of a word.

• Delimit table: A container for 256 ‘delimit items.’

• Generic box: Any type of box on a page. Use the ‘generic box’ if you want to change the properties of a box in a specific location, regardless of whether it is a picture box, text box, line box, etc.

• Group box: An item that consists of a group of boxes. A group box can be either a ‘true’ group, i.e., boxes that have been grouped, or it can be a selection of multiple boxes.

• Master document: A container for master pages. The ‘master document’ allows access to master pages and master page objects.

• Path: A path is a bezier element, and can be either a Shape path, Clipping path, or a Runaround path.

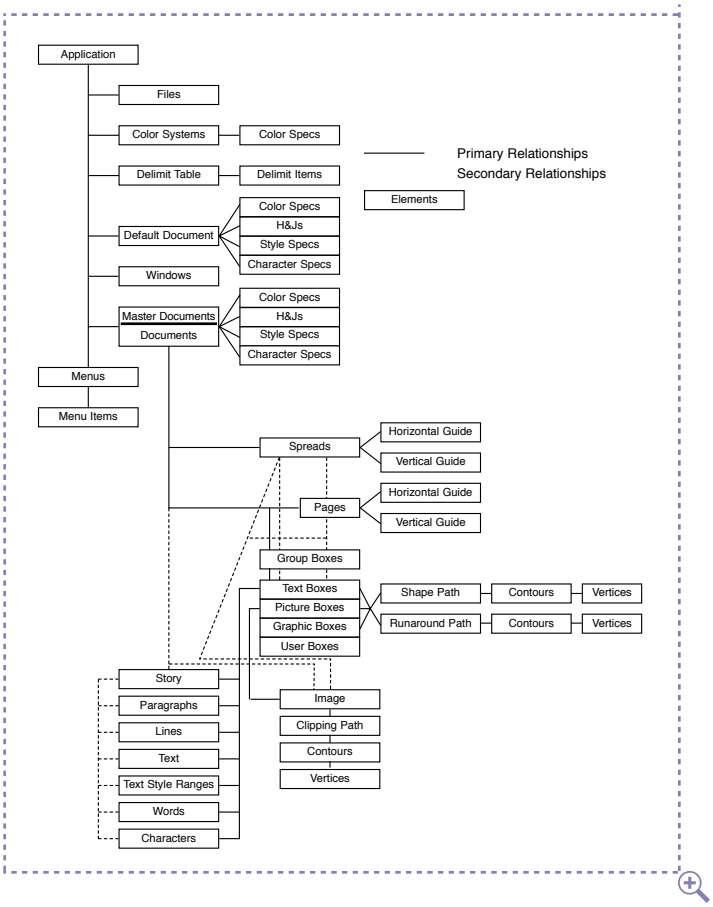
• Text style range: A range of text with a single set of styles specified. Use ‘text style range’ for functions such as Find/Change.

• User box: A user box is a box that is created by an XTension to serve a specific purpose.

• Vertex: A vertex is a point on a bezier line. A vertex is defined by its position, and can be modified by its anchor point or handles.

QuarkXPress Object Hierarchy

When you create a document in QuarkXPress, you are working within the QuarkXPress object hierarchy. From the application level, you set defaults and create documents. At the document level, you create pages, spreads, style sheets, colors, and H&Js. You then add picture boxes and images, text boxes and text, and line boxes at the page level. The QuarkXPress object hierarchy is structured according to the diagram shown below.



QuarkXPress 4.1 Scripting Containment Hierarchy.

Object Limitations

The Apple events implementation in QuarkXPress does not currently support the following:

- Long-document features: books, libraries, lists, indexing, and section page numbers
- Text: editing auxiliary dictionaries and hyphenation exceptions
- Pictures: clipping paths
- Items: Merge/split functions, box creation defaults, and anchored boxes
- Color and printing: trapping, Multi-Ink colors, Hexachrome colors, print styles, and the PPD Manager
- XTensions Manager and most QuarkXTensions™ software distributed by Quark

Apple events are supported by some third-party XTensions™ software, including XData, BeyondPress, and TableWorks.

Future versions of QuarkXPress may support the objects listed above. This document will be updated and distributed with new versions of QuarkXPress as necessary.

Script Writing Syntax

To write a script, first you need to learn the scripting language of the scripting application you purchase. Scripts combine events, objects, properties, and data in the order specified by the scripting language. UserTalk, UserLand Frontier's language, is similar to traditional programming languages such as C or Pascal. AppleScript is more like the English language.

Sample Syntax

UserTalk

Event	Object	Property	Data
set	(textBox[1].	color,	"Red")
set	(pictureBox[5].	runaround,	manual)

AppleScript

Event	Property	Object	Data
set	the color of	text box 1 to	"Red"
set	the runaround of	picture box 5 to	manual

Spaces

In UserTalk, the names of all objects and properties that contain spaces must be used without spaces. For example, you would write "text box"as "textBox" and "auto text box" as "autoTextBox." Although UserLand Frontier is not case sensitive, it may be easier to identify objects if you capitalize the first letter of a new word.

In AppleScript, you can use the names of all objects and properties as shown in the AppleScript Dictionary. AppleScript is not case sensitive.

Quotation Marks

In both UserTalk and AppleScript, you should enclose data in straight quotes and use typographer's quotation marks as foot and inch marks. For example, to specify six inches use "6”" rather than “6"” or "6"". You can also use a back slash to precede a straight quote, when indicating feet or inches. For example, to specify six inches you can use "6\"". The back slash can be used anytime you want to tell AppleScript that the following character is not to be used as a control character.

Optimizing the Performance of Scripts

Follow these suggestions, and any techniques advocated by your scripting software, to write more efficient scripts.

Process in QuarkXPress

Allow QuarkXPress to do all the calculations using its own built-in functions, and minimize context switches between QuarkXPress and the scripting application. For example, to set the color of all generic boxes to red:

Do

UserTalk

```
set(genericBox[all].color,"Red")
```

AppleScript

```
set the color of every generic box to "Red"
```

Don't

UserTalk

```
numberOfBoxes = count(it,genericBox)
  for i = 1 to numberOfBoxes
    set(genericBox[i].color, "Red")
```

AppleScript

```
set numberOfBoxes to count of generic boxes
  repeat with i from 1 to numberOfBoxes
    set color of generic box i to "Red"
  end repeat
```

Or, to change the color of all green generic boxes to red:

Do

UserTalk

```
set(genericBox[color.name=="Green"].color,"Red")
```

AppleScript

```
set color of every generic box whose color equals "Green" ¬
to "Red"
```

Don't

UserTalk

```
numberOfBoxes = count(it,genericBox)
    for i = 1 to numberOfBoxes
      if(get(genericBox[i].color.name)=="Green") then
      set(genericBox[i].color,"Red")
```

AppleScript

```
set numberOfBoxes to count of generic boxes
  repeat with i from 1 to numberOfBoxes
    if name of color of generic box i equals "Green" then set color of
    generic box i to "Red"
  end repeat
```

Specify Initial Properties when Performing a Create/Make Event

Use the "create/make" event to specify initial properties rather than using subsequent "set" events. You can also set multiple properties. In UserTalk, you do this by making a list of properties and using it as an argument in the "create" statement. In AppleScript, you specify the initial properties within the "make" statement.

Do

UserTalk

```
paramlist = 0
  putAppleListItem(90,rotation,@paramlist)
  putAppleListItem("Cyan",color,@paramlist)
  putAppleListItem(80,shade,@paramlist)
  create(pictureBox,0,paramlist,beginningof(page[last]))
```

AppleScript

```
make picture box at beginning of last page with properties ¬
{rotation:90,color: "Cyan",shade:80}
```

Or, set multiple “properties”:

Do

UserTalk

```
paramlist = 0
putAppleListItem(90,rotation,@paramlist)
putAppleListItem("Cyan",color,@paramlist)
putAppleListItem(80,shade,@paramlist)
set(pictureBox[1].properties,paramlist)
```

AppleScript

```
tell document 1
  set properties of picture box 1 to {rotation:90;color: ¬
  "Cyan";shade:80}
end tell
```

Don’t

UserTalk

```
with document[1].page[1]
  with pictureBox[1]
    set(rotation,90)
    set(color,"Cyan")
    set(shade,80)
```

AppleScript

```
tell page 1 of document 1
  tell picture box 1
    set rotation to 90
    set color to "Cyan"
    set shade to 80
  end tell
end tell
```

Compile Scripts

Using the “do script” event sends a compiled script directly to QuarkXPress where it is processed completely within the application before anything appears on-screen.

Add the following lines to AppleScripts to have them automatically compile and execute within QuarkXPress:

```
script theRealScript
  tell application "QuarkXPress"
    --the script goes here
  end tell
end script

tell application "QuarkXPress"
  do script {theRealScript}
end tell
```

Script Writing Sample

To illustrate how scripts interact with QuarkXPress objects, we have dissected a script that uses a representative sample of the objects that QuarkXPress supports. The breakdown discusses creating objects, specifying initial properties, and changing properties. In addition, the various aspects of the scripting environment, including suites, events, the object model, the object hierarchy, and object references, are discussed in the context of the objects in this script. You can apply similar syntactical constructs to other objects in other scripts.

The sample script, "Document Construction," illustrates how Apple events control objects within QuarkXPress. The script sets guides, creates text and picture boxes, imports text and images, then specifies the properties of these objects and their elements to produce a final layout.

The sample script was written in UserLand Frontier's scripting language, UserTalk, and in AppleScript. Although this discussion occasionally refers to specific commands in UserLand Frontier, the concepts outlined apply to AppleScript as well. The syntax for the "Document Construction" script is shown in both UserTalk and AppleScript.

The Document Construction Script

Before reading this section, run the "Document Construction" AppleScript script to familiarize yourself with its actions. The script and the files it requires are in the Sample Documents folder (**QuarkXPress folder ➝ Documents ➝ Apple Events Scripting**).

UserLand Frontier Syntax: Document Construction Script

```
on ConstructDocument()
  with QXP,QXP.defs,objectModel
    sys.bringAppToFront(file.fileFromPath(QXP.appInfo.path))
    «
    local(oldHeight,oldWidth,oldAutoTextBox,oldGuidesShowing,
    xDocMeasure,yDocMeasure)
    local(thepath,howmany,lastFieldNumber,filepath,count)
    local(frect,bkgrndclr)
    thepath = "Hard Drive:Applications:QuarkXPress:Documents:Apple
    Events Scripting:Sample Documents"
    «
    with defaultdocument[1]
      oldHeight = get(pageHeight)
      oldWidth = get(pageWidth)
      oldAutoTextBox = get(autoTextBox)
```

```
  oldGuidesShowing = get(guidesShowing)
  xDocMeasure = get(horizontalMeasure)
  yDocMeasure = get(verticalMeasure)
  set(pageHeight,"30 cm")
  set(pageWidth,"34 cm")
  set(autoTextBox,false)
  set(guidesShowing,true)
  set(guidesInFront,true)
  set(horizontalMeasure,centimeters)
  set(verticalMeasure,centimeters)
create(document,0,0,beginningof(it))
«
«SET THE VIEW SCALE TO FIT IN WINDOW
with document[1]
  set(viewscale,fitpageinwindow)
«
«CREATE GUIDES TO LAYOUT ELEMENTS ON THE PAGE
with document[1]
  paramlist = 0
  putAppleListItem("4.218 cm",position,@paramlist)
  create(horizontalGuide,0,paramlist,beginningof(page[1]))
  paramlist = 0
  putAppleListItem("8.477 cm",position,@paramlist)
  create(horizontalGuide,0,paramlist,endof(page[1]))
  paramlist = 0
  putAppleListItem("27.152 cm",position,@paramlist)
  create(horizontalGuide,0,paramlist,endof(page[1]))
  paramlist = 0
  putAppleListItem("2 cm",position,@paramlist)
  create(verticalGuide,0,paramlist,endof(page[1]))
  paramlist = 0
  putAppleListItem("4.962 cm",position,@paramlist)
  create(verticalGuide,0,paramlist,endof(page[1]))
  paramlist = 0
  putAppleListItem("18.742 cm",position,@paramlist)
  create(verticalGuide,0,paramlist,endof(page[1]))
  paramlist = 0
  putAppleListItem("32 cm",position,@paramlist)
  create(verticalGuide,0,paramlist,endof(page[1]))
«
«CREATE FIRST TEXT BOX.
with document[1].page[1]
  paramlist = 0
```

```
    frect = textrect("2 cm","5 cm","8 cm","19 cm")
    putAppleListItem(frect,bounds,@paramlist)
    create(textbox,0,paramlist,beginningof(it))
    with textbox[1]
      set(VerticalJustification,bottom)
      set(color,"none")
    with textbox[1].story[1]
      set(it,"Biking Gear")
      set(font,"Times")
      set(word[1].size,30)
      set(word[1].textstyle,allcaps)
      set(word[1].baseshift,60)
      set(word[1].track,50)
      set(word[1].character[last].kern,-100)
      set(word[2].size,120)
      set(word[2].color,"Blue")
      set(word[2].textstyle,Italic)
      set(word[2].character[1].kern,-5)
      set(word[2].character[2].kern,-5)
      «
  «CREATE SECOND TEXT BOX.
  with document[1].page[1]
    paramlist = 0
    frect = textrect("8.5 cm","5 cm","29.959 cm","18.472 cm")
    putAppleListItem(frect,bounds,@paramlist)
    create(textbox,0,paramlist,endof(it))
    with textbox[2]
      try
        set(story[1],alias(thepath+"ASB Text"))
      else
        if (file.getFileDialog("Open the file named 'ASB
        Text'.",@filepath,'TEXT'))
          set(story[1],alias(filepath))
          «
          «COUNT THE NUMBER OF SEPARATE STRINGS SEPARATED BY ':'.
          lastFieldNumber = string.countFields(filepath,":")

          «RESET THE PATH TO THE TEXT.
          thepath = ""
          for count = 1 to lastFieldNumber-1
            «
            «RECONSTRUCT THE PATH.
```

```
            thepath =thepath+string.nthField(filepath,":",
            count)+":"
      set(story[1].size,11)
      set(story[1].leading,43)
      set(story[1].justification,full)
      set(story[1].font,"Times")
    with textbox[2].story[1].paragraph[1]
      set(dropcapchars,1)
      set(dropcaplines,3)
      set(word[1].character[1].color,"Blue")
    with textbox[2].story[1].paragraph[last]
      set(word[1].character[1].color,"Blue")
      set(word[1].character[1].size,28)
      set(ruleabove.ruleon,true)
      set(ruleabove.textlength,false)
      set(ruleabove.width,0.5)
      set(ruleabove.position,"1 cm")
      set(ruleabove.color,"Cyan")
      set(ruleabove.shade,100)
«
«CREATE FIRST PICTURE BOX.
with document[1].page[1]
  paramlist = 0
  frect = textrect("10.386 cm","20.758 cm","27.636 cm","33.508 cm")
  bkgrndclr = "None"
  putAppleListItem(frect,bounds,@paramlist)
  putAppleListItem(bkgrndclr,color,@paramlist)
  create(picturebox,0,paramlist,beginningof(it))
  with picturebox[1]
    set(rotation,-25)
    try
      set(image[1],alias(thepath+"Shirts.tiff"))
    else
      if (file.getFileDialog("Open the image named
      'Shirts.tiff'.",@filepath,'TIFF'))
        set(image[1],alias(filepath))
     with image[1]
      set(scale,textpoint("115","115"))
«
«CREATE SECOND PICTURE BOX.
with document[1].page[1]
  paramlist = 0
  frect = textrect("8.471 cm","2 cm","9.971 cm","3.5 cm")
```

```
  bkgrndclr = "None"
  putAppleListItem(frect,bounds,@paramlist)
  putAppleListItem(bkgrndclr,color,@paramlist)
  create(picturebox,0,paramlist,endof(it))
  with picturebox[2]
    try
      set(image[1],alias(thepath+"Glove.tiff"))
    else
      if (file.getFileDialog("Open the image named
      'Glove.tiff'.",@filepath,'TIFF'))
        set(image[1],alias(filepath))
    set(image[1].bounds,exactfit)
  clone(picturebox[2],after(picturebox[2]))
  with picturebox[3]
    set(bounds,textrect("12.471 cm","2 cm","13.971 cm","3.5 cm"))
  clone(picturebox[2],after(picturebox[3]))
  with picturebox[4]
    set(bounds,textrect("16.471 cm","2 cm","17.971 cm","3.5 cm"))
  clone(picturebox[2],after(picturebox[4]))
  with picturebox[5]
    set(bounds,textrect("20.471 cm","2 cm","21.971 cm","3.5 cm"))
«
«CREATE THIRD PICTURE BOX.
with document[1].page[1]
  create(picturebox,0,0,endof(it))
  with picturebox[6]
    set(bounds,textrect("6.875 cm","18.425 cm","12.729 cm","26.4
    cm"))
    set(color,"None")
    try
      set(image[1],alias(thepath+"Helmet.tiff"))
    else
      if (file.getFileDialog("Open the image named
      'Helmet.tiff'.",@filepath,'EPSF'))
        set(image[1],alias(filepath))
    set(image[1].scale,textpoint("70","70"))
    set(image[1].offset,textpoint("0.557 cm","1.254 cm"))
«CREATE LINES.
with document[1].page[1]
  create(linebox,0,0,beginningof(it))
  with linebox[1]
    set(leftpoint,textpoint("0 cm","21.406 cm"))
    set(rightpoint,textpoint("8 cm","21.406 cm"))
```

```
        paramlist = 0
        putAppleListItem("Magenta",color,@paramlist)
        putAppleListItem(3,Width,@paramlist)
        putAppleListItem(dotted,Style,@paramlist)
        set(Properties,paramlist)
    with document[1].page[1]
      create(linebox,0,0,endof(it))
      with linebox[2]
        paramlist = 0
        set(leftpoint,textpoint("8 cm","2 cm"))
        set(rightpoint,textpoint("8 cm","32 cm"))
        set(width,0.5 )
    «
    set(document[1].guidesShowing,false)
    saveDocument(document[1],thePath+"Constructed Document")
    with defaultdocument[1]
      set(pageHeight,oldHeight)
      set(pageWidth,oldWidth)
      set(autoTextBox,oldAutoTextBox)
      set(guidesShowing,oldGuidesShowing)
      set(horizontalMeasure,xDocMeasure)
      set(verticalMeasure,yDocMeasure)
ConstructDocument()
```

AppleScript Syntax: Document Construction Script

```
tell application "QuarkXPress"
  activate
  set thepath to (Choose folder with prompt "Select the Sample " &¬
  "Documents folder inside the Apple Events Scripting folder of "¬
  "your QuarkXPress folder:") as text

  tell default document 1
    set oldHeight to page height
    set oldWidth to page width
    set oldAutoTextBox to automatic text box
    set oldGuidesShowing to guides showing
    set xDocMeasure to horizontal measure
    set yDocMeasure to vertical measure
    set page height to "30 cm"
    set page width to "34 cm"
    set automatic text box to false
    set guides showing to true
    set guides in front to true
```

```
  set horizontal measure to centimeters
  set vertical measure to centimeters
end tell
make document at beginning

tell document 1
  set view scale to fit page in window
end tell

--CREATE GUIDES TO LAYOUT ELEMENTS ON THE PAGE
tell page 1 of document 1
  make horizontal guide at beginning with properties {position:¬
  "4.218 cm"}
  make horizontal guide at end with properties {position:"8.447 cm"}
  make horizontal guide at beginning with properties {position:¬
  "27.152 cm"}
  make vertical guide at end with properties {position:"2 cm"}
  make vertical guide at end with properties {position:"4.962 cm"}
  make vertical guide at end with properties {position:"18.742 cm"}
  make vertical guide at end with properties {position:"32 cm"}
end tell

--CREATE FIRST TEXT BOX.
tell page 1 of document 1
  make text box at beginning with properties {bounds:{"2 cm", ¬
  "5 cm", "8 cm", "19 cm"}}
  tell text box 1
    set vertical justification to bottom
    set color to "none"
  end tell
end tell

tell story 1 of text box 1 of page 1 of document 1
  set contents of it to "Biking Gear"
  set font to "Times"
  set size of word 1 to 30
  set style of word 1 to all caps
  set base shift of word 1 to 60
  set track of word 1 to 50
  set kern of last character of word 1 to -100
  set size of word 2 to 120
  set color of word 2 to "Blue"
  set style of word 2 to italic
  set kern of character 1 of word 2 to -5
```

```
  set kern of character 2 of word 2 to -5
end tell

--CREATE SECOND TEXT BOX.
tell page 1 of document 1
  make text box at end with properties {bounds:{"8.5 cm", ¬
  "5 cm", "29.959 cm", "18.472 cm"}}
  tell text box 2
    try
      set story 1 to alias (thepath & "ASB Text")
    on error
      set story 1 to (choose file with prompt "Please select the file ¬
      \"ASB Text.\"" of type {"Text"})
    end try
    set size of story 1 to 11
    set leading of story 1 to 43
    set justification of story 1 to full
    set font of story 1 to "Times"
  end tell
  tell paragraph 1 of story 1 of text box 2
    set drop cap characters to 1
    set drop cap lines to 3
    set color of character 1 of word 1 to "Blue"
  end tell
  tell last paragraph of story 1 of text box 2
    set color of character 1 of word 1 to "Blue"
    set size of character 1 of word 1 to 28
    set rule on of rule above to true
    set text length of rule above to false
    set width of rule above to 0.5
    set position of rule above to "1 cm"
    set color of rule above to "Cyan"
    set shade of rule above to 100
  end tell
end tell

--CREATE FIRST PICTURE BOX.
tell page 1 of document 1
  make picture box at beginning with properties ¬
  {bounds:{"10.386 cm", "20.758 cm", "27.636 cm", ¬
  "33.508 cm"}, color:"None"}
  tell picture box 1
    set rotation to -25
    try
```

```
      set image 1 to alias (thepath & "Shirts.TIFF")
    on error
      set image 1 to (choose file with prompt "Please select the file ¬
      \"Shirts.TIFF.\"" of type {"TIFF"})
    end try
    tell image 1
      set scale to {"115", "115"}
    end tell
  end tell
end tell

--CREATE SECOND PICTURE BOX.
tell page 1 of document 1
  make picture box at end with properties {bounds:¬
  {"8.471 cm", "2 cm", "9.971 cm", "3.5 cm"}, color:"None"}
  tell picture box 2
    try
      set image 1 to alias (thepath & "Glove.TIFF")
    on error
      set image 1 to (choose file with prompt "Please select the file ¬
      \"Glove.TIFF.\"" of type {"TIFF"})
    end try
    set bounds of image 1 to exact fit
  end tell
  duplicate picture box 2 to after picture box 2
  tell picture box 3
    set bounds to {"12.471 cm", "2 cm", "13.971 cm", "3.5 cm"}
  end tell
  duplicate picture box 2 to after picture box 3
  tell picture box 4
    set bounds to {"16.471 cm", "2 cm", "17.971 cm", "3.5 cm"}
  end tell
  duplicate picture box 2 to after picture box 4
  tell picture box 5
    set bounds to {"20.471 cm", "2 cm", "21.971 cm", "3.5 cm"}
  end tell
end tell

--CREATE THIRD PICTURE BOX.
tell page 1 of document 1
  make picture box at end with properties {bounds:{"6.875 cm", ¬
  "18.425 cm", "12.729 cm", "26.4 cm"}, color:"None"}
  tell picture box 6
    try
```

```
        set image 1 to alias (thepath & "Helmet.TIFF")
      on error
        set image 1 to (choose file with prompt "Please select "&¬
        "the file \"Helmet.TIFF\"" of type {"TIFF"})
      end try
      tell image 1
        set scale to {"70", "70"}
        set offset to {"0.557 cm", "1.254 cm"}
      end tell
    end tell
  end tell

  --CREATE LINES
  tell page 1 of document 1
    make line box at end with properties {left point:¬
    {"0 cm", "21.406 cm"}, right point:{"8 cm", "21.406 cm"}}
    tell line box 1
      set color to "Magenta"
      set width to 3
      set style to dotted
    end tell
    make line box at end
    tell line box 2
      set left point to {"8 cm", "2 cm"}
      set right point to {"8 cm", "32 cm"}
      set width to 0.5
    end tell
  end tell

  set guides showing of document 1 to false
  save document 1 in (thepath & "Constructed Document")

  tell default document 1
    set page height to oldHeight
    set page width to oldWidth
    set automatic text box to oldAutoTextBox
    set guides showing to oldGuidesShowing
    set horizontal measure to xDocMeasure
    set vertical measure to yDocMeasure
  end tell
end tell
```

About the Script Breakdown

This section first discusses how to direct a script to QuarkXPress. The script is then divided into the steps a user would perform when constructing a document. The steps include creating a new document, creating a text box, importing text, formatting the text, etc. The script syntax is then displayed in Courier font. Following the syntax is a concept line that translates the scripting language into actions in QuarkXPress. The events, objects, and properties set in the script are then analyzed line by line. The script breakdown follows this format:

A step in the document construction process

UserTalk

```
code
```

AppleScript

```
code
```

Concepts: The code above is described in terms of actions in QuarkXPress.

Each event, object, and/or property is discussed line by line.

Breakdown of the Document Construction Script

Locate the terminology for QuarkXPress objects and events

UserTalk

```
with QXP,QXP.defs,objectModel
```

AppleScript

```
tell application "QuarkXPress"
```

Concepts: This statement specifies the location of QuarkXPress terminology.

• Always put this "with" statement around all UserLand Frontier code written for QuarkXPress. In UserLand Frontier, "QXP" is a table and "QXP.defs" is a subtable of QXP. The "objectModel" indicates that the script uses Object Model syntax. To look at the table, click the **Object DB** button in UserLand Frontier. This opens the Frontier.root, which contains tables, scripts, and other items. Double-click the triangle next to a table name to open it. Open the "system" table. From the "system" table, open the "verbs" table. Continue in this fashion for the "apps" table, then the "QXP" table. Open the "defs" table from the "QXP" table.

• In the remainder of this section, the following format will be used to reference the location of items in UserLand Frontier: root.system.verbs.apps.QXP.examples.

Declare the variables

UserTalk

```
local(oldHeight,oldWidth,oldAutoTextBox,oldGuides Showing,
xDocMeasure,yDocMeasure)
local(thepath,howmany,lastFieldNumber,filepath,count)
local(frect,bkgrndclr)
```

AppleScript

```
no equivalent required
```

Concepts: This statement declares local variables for the script.

Although it is not essential to declare local variables, it makes scripts much safer. Making variables local ensures that UserLand Frontier or QuarkXPress table entries will not be altered inadvertently if they have the same name as a variable used in a script.

Activate QuarkXPress

UserTalk

```
sys.bringAppToFront(file.fileFromPath(QXP.appInfo. path))
```

AppleScript

```
activate
```

Concepts: This statement is similar to choosing QuarkXPress from the **Finder** menu.

• This statement ensures that QuarkXPress is the active application. Rather than including the actual code, the script calls another UserLand Frontier script, “bringAppToFront.”

• The “bringAppToFront” script is located at: root.system.verbs.builtins.sys. Another script, “fileFromPath,” is located at “root.system.verbs.builtins.file.” The “path” provides the information for UserLand Frontier to locate QuarkXPress when executing the script. Open these scripts and look at them if you wish.

Establish the path

UserTalk

```
thepath = "Hard Drive:QuarkXPress:Documents:Apple Events Scripting:Sam-
ple Documents"
```

AppleScript

```
set thepath to (Choose folder with prompt "Select the "¬
"Sample Documents folder inside the Apple Events "¬
"Scripting folder of your QuarkXPress folder:") as text
```

Concepts: This statement establishes a path for sample text and image files.

• This statement gives the variable "thepath" a string value that is the path to the location of the text and image files.

• In this example, "Hard Drive" is the name of the local hard drive. The path assumes that the QuarkXPress folder is at the root level of the hard drive. If you want to access something on the desktop, reference "Desktop Folder" in the path name after the name of the local hard drive.

Save current document default specifications

UserTalk

```
with defaultDocument[1]
oldHeight = get(pageHeight)
oldWidth = get(pageWidth)
oldAutoTextBox = get(autoTextBox)
oldGuidesShowing = get(guidesShowing)
xDocMeasure = get(horizontalMeasure)
yDocMeasure = get(verticalMeasure)
```

AppleScript

```
tell default document 1
  set oldHeight to page height
  set oldWidth to page width
  set oldAutoTextBox to automatic text box
  set oldGuidesShowing to guides showing
  set xDocMeasure to horizontal measure
  set yDocMeasure to vertical measure
```

Concepts: The "get" statements above determine your current document default specifications. When the document construction is complete, the script replaces the new default specifications with these original specifications.

• The "with" statement references the current "defaultDocument" by index [1]. (The "defaultDocument" is the object that contains all default document settings

including colors, style sheets, H&Js, document settings specified in the **New Document** dialog box, and all preferences.)

• The results of the "get" statements are assigned to local variables in the following six lines. The "get" statements determine the current page size, automatic text box setting, whether the guides are showing, and the default measurement system.

Set default specifications for a new document

UserTalk

```
set(pageHeight,"30 cm")
set(pageWidth,"34 cm")
set(autoTextBox,false)
set(guidesShowing,true)
set(guidesInFront,true)
set(horizontalMeasure,centimeters)
set(verticalMeasure,centimeters)
```

AppleScript

```
  set page height to "30 cm"
  set page width to "34 cm"
  set automatic text box to false
  set guides showing to true
  set guides in front to true
  set horizontal measure to centimeters
  set vertical measure to centimeters
end tell
```

Concepts: The first three "set" statements are similar to setting default specifications in the **New Document** dialog box. The next "set" statement is similar to choosing **Show Guides** from the **View** menu. The last three statements are settings in the **General** tab of the **Document Preferences** dialog box.

• The first two "set" events specify the "pageHeight" and "pageWidth" properties.

• The third "set" event determines whether the document will have an "autoTextBox" depending on the Boolean operator. If the Boolean operator is "false," the document will not have an automatic text box. If the Boolean operator is "true," it will.

• The fourth “set” event determines whether the document will have “guidesShowing” depending on the Boolean operator. If the Boolean operator is “true,” all guides will show. If the Boolean operator is “false,” all guides will be hidden.

• The fifth “set” event determines whether the guides will be displayed “inFront” of the page elements. The “true” Boolean operator indicates that the guides will display in front.

• The last two “set” events specify the default horizontal and vertical measurement system as centimeters.

Create a new document with default specifications

UserTalk

```
create(document,0,0,beginningOf(it))
```

AppleScript

```
make document at beginning
```

Concepts: This “create” event is similar to clicking **OK** in the **New Document** dialog box.

• The “create” event is used to make the object specified by the four parameters.

• The first parameter, “document,” refers to the object that will be created.

• The second parameter “0” specifies the initial data or initial contents of the object to be created. The “0” means no initial contents. Specifying initial contents or data usually only applies to text or image containers.

• The third parameter, “0,” specifies the initial properties of the created object. The “0” means the document will have no initial properties set and will be created according to information in the “defaultDocument” object. To set properties using the third parameter, you must put the desired properties in a list and pass this list as the third parameter (see the “Create Guides” portion of the “Script Writing Sample“ section in this document).

• The fourth parameter, “beginningOf(it)” specifies where you want the object to be created. (The “it” refers to the last object referenced in the “with” statement.) You can create an object at any insertion point: beginning, ending, after, before, or replace.

Set the view scale

UserTalk

```
with document[1]
set(viewScale,fitPageInWindow)
```

AppleScript

```
tell document 1
  set view scale to fit page in window
end tell
```

Concepts: The lines above are similar to choosing **Fit in Window** from the **View** menu for the active document.

• The "with" statement references the active "document" by index [1].

• The "set" event changes the "viewScale" property to the data "fitPageInWindow." The "viewScale" property can be a percentage or specific view. For example, to specify 100% view, use "100" for the second parameter. To specify thumbnails, use "thumbnails" for the second parameter.

Create guides

UserTalk

```
with document[1]
paramlist = 0
putAppleListItem("4.218 cm",position,@paramlist)
create(horizontalGuide,0,paramlist,beginningof(page[1]))
paramlist = 0
putAppleListItem("8.477 cm",position,@paramlist)
create(horizontalGuide,0,paramlist,endof(page[1]))
paramlist = 0
putAppleListItem("27.152 cm",position,@paramlist)
create(horizontalGuide,0,paramlist,endof(page[1]))
paramlist = 0
putAppleListItem("2 cm",position,@paramlist)
create(verticalGuide,0,paramlist,endof(page[1]))
paramlist = 0
putAppleListItem("4.962 cm",position,@paramlist)
create(verticalGuide,0,paramlist,endof(page[1]))
paramlist = 0
putAppleListItem("18.742 cm",position,@paramlist)
create(verticalGuide,0,paramlist,endof(page[1]))
paramlist = 0
putAppleListItem("32 cm",position,@paramlist)
create(verticalGuide,0,paramlist,endof(page[1]))
```

AppleScript

```
tell page 1 of document 1
  make horizontal guide at beginning with properties ¬
  {position:"4.218 cm"}
  make horizontal guide at end with properties ¬
  {position:"8.447 cm"}
  make horizontal guide at beginning with properties ¬
  {position:"27.152 cm"}
  make vertical guide at end with properties ¬
  {position:"2 cm"}
  make vertical guide at end with properties ¬
  {position:"4.962 cm"}
  make vertical guide at end with properties ¬
  {position:"18.742 cm"}
  make vertical guide at end with properties ¬
  {position:"32 cm"}
end tell
```

Concepts: The “create” events above simulate clicking the horizontal and vertical rulers to create guides, and then dragging the guides into position.

• The “with” statement references the active “document” by index [1]. The index value [1] refers to the frontmost document.

• The “paramlist” is a list containing multiple properties that can be passed as the third parameter in a “create” event. The “paramlist = 0” statement clears any information currently in the “paramlist.”

• The “putAppleListItem” statements add a property and associated data value to the given “paramlist” (in this case, the “position” property is specified in centimeters). The “paramlist = 0” statement is repeated prior to each “putAppleListItem” statement to clear the list.

• Each “create” event makes a “horizontalGuide” or “verticalGuide.” The guides are created with the properties specified in the “paramlist” (“position” in centimeters).

• The first guide is created at the “beginningOf(page[1])” in the object hierarchy according to the fourth parameter. Subsequent guides are created at “endOf(page[1]).”

Create the first text box

UserTalk

```
with document[1].page[1]
  paramlist = 0
  frect = textRect("2 cm","5 cm","8 cm","19 cm")
  putAppleListItem(frect,bounds,@paramlist)
  create(textBox,0,paramlist,beginningOf(it))
```

AppleScript

```
tell page 1 of document 1
  make text box at beginning with properties ¬
  {bounds:{"2 cm", "5 cm", "8 cm", "19 cm"}}
```

Concepts: The lines above are similar to creating a text box with the Text Box tool, and then sizing and positioning it from the **Measurements** palette.

• The "with" statement references the first "page" of the active "document"; both are referenced by index [1].

• The "paramlist" is a list containing multiple properties that can be passed as the third parameter in a "create" event. The "paramlist = 0" statement clears any information currently in the "paramlist."

• A local variable, "frect," is assigned a value. This value is the result of calling the "textRect" script (located in Frontier's "QXP.defs" table), which sets numerical values for the text box boundaries. The four parameters indicate how far from the top left corner of the document page the top, left, bottom, and right sides of the box will be positioned.

• The "putAppleListItem" statement adds the "frect" values to the given "paramlist." The "bounds" property is assigned the value of the variable "frect."

• The "create" event makes a "textBox" with default specifications (indicated by the "0" in the second parameter) and the specifications determined by the paramlist (indicated in the third parameter). The text box is placed according to the fourth parameter, "beginningOf(it)."

• If you want to see an object after it is created while the script is running, add the line "show(it)." This places the current object in the upper left corner of the document window.

Enter the headline into a text box

UserTalk

```
with textBox[1]
set(verticalJustification,bottomJustified)
  set(color,"None")
with textBox[1].story[1]
  set(it,"Biking Gear")
```

AppleScript

```
tell text box 1
      set vertical justification to bottom justified
      set color to "none"
    end tell
  end tell

  tell story 1 of text box 1 of page 1 of document 1 set contents of ¬
  it to "Biking Gear"
```

Concepts: The statements above are similar to specifying a Vertical Alignment and Background Color in the **Text** tab of the **Modify** dialog box, and then typing "Biking Gear" into the text box.

• The "with" statement references the first "textBox" by index [1].

• The next two "set" events change the "verticalJustification" to "bottom" and the background "color" to "None."

• The "with" statement references the "story" in the active "textBox"; both are referenced by index [1]. (Only one "story" is possible per text box or chain of linked text boxes.)

• The "set" event specifies "it" ("it" refers to the "story," the last object referenced in the "with" statement). The text "Biking Gear" is entered into the text box. It is then formatted with properties defined in the Normal style sheet for the "defaultDocument."

Format the headline

UserTalk

```
set(font,"Times")
set(word[1].size,30)
set(word[1].textStyle,allcaps)
set(word[1].baseShift,60)
set(word[1].track,50)
set(word[1].character[last].kern,-100)
set(word[2].size,120)
set(word[2].color,"Blue")
set(word[2].textStyle,Italic)
set(word[2].character[1].kern,-5)
set(word[2].character[2].kern,-5)
```

AppleScript

```
  set font to "Times"
  set size of word 1 to 30
  set style of word 1 to all caps
  set base shift of word 1 to 60
  set track of word 1 to 50
  set kern of last character of word 1 to -100
  set size of word 2 to 120
  set color of word 2 to "Blue"
  set style of word 2 to italic
  set kern of character 1 of word 2 to -5
  set kern of character 2 of word 2 to -5
end tell
```

Concepts: The “set” statements above are comparable to the **Font**, **Size**, **Type Style**, **Color**, **Baseline Shift**, **Track**, and **Kern** commands in the **Style** menu.

• The first “set” event changes the font for the “story” to “Times.”

• The next four “set” events reference the first “word” by index [1]. The “size,” “textStyle,” “baseShift,” and “track” properties of the word “Biking” are changed.

• The next “set” event references the last “character” of the first “word”; the “character” is referenced by relative position. The “kern” property is changed to “–100.” (To kern the space between two words, reference the last character of the first word.)

• The next three “set” events reference the second “word” by index [2]. The size, color, and type style properties of the word “Gear” are changed.

• The last two “set” events reference the first and second “character” of the second “word”; all are referenced by index. The “kern” property of each “character” is changed to “-5.” (To kern a pair of characters, you only need to reference the first character.)

Create the second text box

UserTalk

```
with document[1].page[1]
paramlist = 0
frect = textRect("8.5 cm","5 cm","29.959 cm","18.472 cm")
putAppleListItem(frect,bounds,@paramlist)
create(textBox,0,paramlist,endOf(it))
```

AppleScript

```
tell page 1 of document 1
  make text box at end with properties ¬
  {bounds:{"8.5 cm", "5 cm", "29.959 cm", ¬
  "18.472 cm"}}
```

Concepts: The lines above are similar to creating a text box with the Text Box tool, and then sizing and positioning it from the **Measurements** palette.

• The “with” statement references the first “page” of the active “document”; both are referenced by index [1].

• The “paramlist” is a list containing multiple properties that can be passed as the third parameter in a “create” event. The “paramlist = 0” statement clears any information currently in the “paramlist.”

• A local variable, “frect,” is assigned a value. This value is the result of calling the “textRect” script (located in Frontier’s “QXP.defs” table), which sets numerical values for the text box boundaries. The four parameters indicate how far from the top left corner of the document page the top, left, bottom, and right sides of the box will be positioned.

• The “putAppleListItem” statement adds the “frect” values to the given “paramlist.” The “bounds” property is assigned the value of the variable “frect.”

• The “create” event makes a “textBox” with default specifications (indicated by the “0” in the second parameter) and the specifications determined by the paramlist (indicated in the third parameter). The text box is placed according to the fourth parameter, “endOf(it).”

Locate and import a text file

UserTalk

```
with textbox[2]
try
  set(story[1],alias(thepath+"ASB Text"))
  else
    if (file.getFileDialog("Open the file named 'ASB
    Text'.",@filepath,'TEXT'))
      set(story[1],alias(filepath))
```

AppleScript

```
tell text box 2
  try
    set story 1 to alias (thepath & "ASB Text")
  on error
    set story 1 to (choose file with prompt ¬
    "Please select the file \"ASB Text\"" of type¬
{"TIFF"})
  end try
```

Concepts: The statements above are similar to locating and importing a text file in the **Get Text** dialog box.

• The "try" statement looks for the "ASB Text" file in the location previously defined by "thepath" (see the "Establish the path" portion of the "Breakdown of the Document Construction Script" section of this document). If the file exists in this location, the "set" event imports the "ASB Text" file, replacing the "story" in the text box.

• If "ASB Text" does not exist in the location defined by "thepath," the script will continue with the "if" statement. (The file will only be located via "thepath" if your hard drive and folders are named the same as those defined in "thepath.")

• The statement in the "if" construct calls a Frontier script, "getFileDialog," located at "root.system.verbs.builtins.file." The "getFileDialog" script brings up an **Open** dialog box.

• The first parameter is a message to the user shown at the bottom of the dialog box, "Open the file named 'ASB Text.'" The second parameter stores the path to the text file in an address — this path is used to import the text file. The third parameter is the signature for a text file (file type). Once the user locates the text file and clicks OK, the "set" event imports the text.

• If you want to open a QuarkXPress document using the "getFileDialog" script, the signature would be "XDOC." This simply limits the displayed files to QuarkXPress documents. This is an optional parameter.

Establish the new path

UserTalk

```
lastFieldNumber = string.countFields(filepath,":")
thepath = ""
for count = 1 to lastFieldNumber-1
thepath =thepath+string.nthField(filepath,":",count)+":"
```

AppleScript

```
no equivalent required
```

Concepts: The path to the text file is saved and used when the script attempts to locate the image files.

• In the statements above, the previous path (established in the "Establish the path" portion of the "Breakdown of the Document Construction Script" section of this document) is defined as "thepath"; the new path is defined as "filepath."

• The first statement assigns the variable "lastFieldNumber" to the results of another Frontier script, "string.countFields." The "string.countFields" counts the number of strings in the path separated by colons.

• The second statement clears "thepath" by setting it to a null string.

• The third statement starts a loop that counts the total number of separate strings separated by a colon in the "filepath."

• The fourth statement reconstructs "thepath" from the location established in the "filepath." The location of the text file is now defined as "thepath."

Format the body copy

UserTalk

```
set(story[1].font,"Times")
set(story[1].size,11)
set(story[1].leading,43)
set(story[1].justification,full)
```

AppleScript

```
  set size of story 1 to 11
  set leading of story 1 to 43
  set justification of story 1 to full
  set font of story 1 to "Times"
end tell
```

Concepts: The "set" statements above are comparable to choosing **Font**, **Size**, **Leading**, and **Alignment** from the **Style** menu.

The four set events reference the entire "story" by index [1]. The "font," "size," "leading," and "justification" properties of the story are set.

Create a colored drop cap

UserTalk

```
with textBox[2].story[1].paragraph[1]
  set(dropCapChars,1)
  set(dropCapLines,3)
  set(word[1].character[1].color,"Blue")
```

AppleScript

```
tell paragraph 1 of story 1 of text box 2
  set drop cap characters to 1
  set drop cap lines to 3
  set color of character 1 of word 1 to "Blue"
end tell
```

Concepts: The statements above are similar to specifying a **Drop Cap** in the **Formats** tab of the **Paragraph Attributes** dialog box. The color "Blue" is then applied to the drop cap character.

• The "with" statement references the first "paragraph" of the "story" in the second "textBox"; the objects are all referenced by index.

• The first "set" event specifies that the first character will be a drop cap. The second "set" event specifies that it will be a three-line drop cap.

• The third "set" event references the drop cap, which is the first "character" of the first "word"; both are referenced by index [1]. The "color" property is changed to "Blue."

Create an initial cap

UserTalk

```
with textBox[2].story[1].paragraph[last]
set(word[1].character[1].color,"Blue")
set(word[1].character[1].size,28)
```

AppleScript

```
tell last paragraph of story 1 of text box 2
  set color of character 1 of word 1 to "Blue"
  set size of character 1 of word 1 to 28
```

Concepts: The lines above are similar to creating a decorative initial cap with local formatting.

• The "with" statement references the last "paragraph" of "story" in the second "textBox." The "story" and "textBox" are referenced by index, the "paragraph" is referenced by relative position.

• The two “set” statements reference the first “character” of the first “word”; they are referenced by index. The “color” property is changed to “Blue” and the “size” property is changed to “28.”

Specify a Rule Above

UserTalk

```
set(ruleAbove.ruleOn,true)
set(ruleAbove.textLength,false)
set(ruleAbove.width,0.5)
set(ruleAbove.position,"1 cm")
set(ruleAbove.color,"Cyan")
set(ruleAbove.shade,100)
```

AppleScript

```
    set rule on of rule above to true
    set text length of rule above to false
    set width of rule above to 0.5
    set position of rule above to "1 cm"
    set color of rule above to "Cyan"
    set shade of rule above to 100
  end tell
end tell
```

Concepts: The “set” events above are comparable to settings in the expanded Rules tab of the Paragraph Attributes dialog box.

• The first “set” event uses a Boolean operator to determine if the paragraph’s “ruleAbove” will be turned on (“ruleOn”). The “true” Boolean operator indicates that the paragraph will have a rule above it.

• The second “set” event uses a Boolean operator to determine if the rule will match the “textLength.” The “false” Boolean operator indicates that it will not match the length of the text. The rule will extend the width of the text box (minus any defined text inset).

• The last four “set” events specify the “width,” “position,” “color,” and “shade” properties of the “ruleAbove.”

Create the first picture box

UserTalk

```
with document[1].page[1]
paramlist = 0
frect = textrect("10.386 cm","20.758 cm","27.636 cm","33.508 cm")
bkgrndclr = "None"
putAppleListItem(frect,bounds,@paramlist)
putAppleListItem(bkgrndclr,color,@paramlist)
create(picturebox,0,paramlist,beginningof(it))
```

AppleScript

```
tell page 1 of document 1
  make picture box at beginning with properties ¬
  {bounds:{"10.386 cm", "20.758 cm", "27.636 cm", ¬
  "33.508 cm"}, color:"None"}
```

Concepts: The lines above are similar to creating a picture box, sizing and positioning it, and then specifying a background color as you would in the **Picture** tab of the **Modify** dialog box.

• The “with” statement references the first “page” of the active “document”; both are referenced by index [1].

• The “paramlist” is a list containing multiple properties that can be passed as the third parameter in a “create” event. The “paramlist = 0” statement clears any information currently in the “paramlist.”

• A local variable, “frect,” is assigned a value. This value is the result of calling the “textRect” script (located in Frontier’s “QXP.defs” table), which sets numerical values for the picture box boundaries. The four parameters indicate how far from the top left corner of the document page the top, left, bottom, and right sides of the box will be positioned.

• A local variable, “bkgrndclr” is assigned a value. In this case, the background color of the picture box will be “None.”

• The “putAppleListItem” statements add the “frect” and “bkgrndclr” values to the given “paramlist.” The “bounds” property is assigned the value of the variable “frect” and the “color” property is assigned the value of the variable “bkgrndclr.”

• The “create” event makes a “pictureBox” with default specifications (indicated by the “0” in the second parameter) and the specifications determined by the paramlist (indicated in the third parameter). The picture box is placed according to the fourth parameter, “beginningOf(it).”

Import the first picture

UserTalk

```
with pictureBox[1]
  set(rotation,-25)
  try
    set(image[1],alias(thepath+"Shirts.tiff"))
  else
    if (file.getFileDialog("Open the image named
    'Shirts.tiff'.",@filepath,'TIFF'))
      set(image[1],alias(filepath))
  with image[1]
    set(scale,textPoint("115","115"))
```

AppleScript

```
  tell picture box 1
    set rotation to -25
    try
      set image 1 to alias (thepath & ¬
      "Shirts.TIFF")
    on error
      set image 1 to (choose file with prompt ¬
      "Please select the file \"Shirts.TIFF\"."¬
      of type {"TIFF"})
    end try
    tell image 1
      set scale to {"115", "115"}
    end tell
  end tell
end tell
```

Concepts: The statement above are similar to locating and importing an image file in the Get Picture dialog box (File menu).

- The “with” statement references the first “pictureBox” by index [1]
- The first “set” event specifies the “rotation” property of the “pictureBox.”
- The “try” statement looks for the “Shirts.tiff” file in the location previously defined by “thepath.” If the file exists in this location, the “set” event specifies “Shirts.tiff” as the “image” in the picture box. (A picture box can only have one image.)
- If “Shirts.tiff” does not exist in the location defined by “thepath,” the script will continue with the “if” statement. The statement in the “if” construct calls a Frontier script, “getFileDialog,” located at “root.system.verbs.builtins.file.” The “getFileDialog” script brings up an **Open** dialog box.

• The first parameter is a message to the user shown at the bottom of the dialog box, “Open the image named ‘Shirts.tiff.’” Once the user locates the image file and clicks **OK**, the “set” event imports the image.

• The second “with” statement references the “image” by index [1].

• The “set” event specifies the “scale” property of the “image” by calling the “textPoint” script (located in Frontier’s “QXP.defs” table). The “textPoint” script passes two values to be coerced into a property-specific data type by QuarkXPress. In this case, the values for “scale” are coerced into percentage types.

Create the second picture box and import a picture

UserTalk

```
with document[1].page[1]
  paramlist = 0
  frect = textRect("8.471 cm","2 cm","9.971 cm","3.5 cm")
  bkgrndclr = "None"
  putAppleListItem(frect,bounds,@paramlist)
  putAppleListItem(bkgrndclr,color,@paramlist)
  create(pictureBox,0,paramlist,endOf(it))
  with pictureBox[2]
    try
      set(image[1],alias(thepath+"Glove.tiff"))
    else
      if (file.getFileDialog("Open the image named
      'Glove.tiff'.",@filepath,'TIFF'))
        set(image[1],alias(filepath))
    set(image[1].bounds,exactFit)
```

AppleScript

```
tell page 1 of document 1
  make picture box at end with properties ¬
  {bounds:{"8.471 cm", "2 cm", "9.971 cm", ¬
  "3.5 cm"}, color:"None"}
  tell picture box 2
    try
      set image 1 to alias (thepath & "Glove.TIFF")
    on error
      set image 1 to (choose file with prompt ¬
      "Please select the file \"Glove.TIFF\"."¬
      of type {"TIFF"})
    end try
    set bounds of image 1 to exact fit
  end tell
```

Concepts: The first seven lines above are similar to creating a picture box, sizing and positioning it, and then specifying a background color as you would in the **Picture** tab of the **Modify** dialog box. The last six statements are similar to locating and importing an image file in the **Get Picture** dialog box.

• The "with" statement references the first "page" of the active "document"; both are referenced by index [1].

• The "paramlist" is a list containing multiple properties that can be passed as the third parameter in a "create" event. The "paramlist = 0" statement clears any information currently in the "paramlist."

• A local variable, "frect," is assigned a value. This value is the result of calling the "textRect" script (located in Frontier's "QXP.defs" table), which sets numerical values for the picture box boundaries. The four parameters indicate how far from the top left corner of the document page the top, left, bottom, and right sides of the box will be positioned.

• A local variable, "bkgrndclr" is assigned a value. In this case, the background color of the picture box will be "none."

• The "putAppleListItem" statements add the "frect" and "bkgrndclr" values to the given "paramlist." The "bounds" property is assigned the value of the variable "frect" and the "color" property is assigned the value of the variable "bkgrndclr."

• The "create" event makes a "pictureBox" with default specifications (indicated by the "0" in the second parameter) and the specifications determined by the paramlist (indicated in the third parameter). The picture box is placed according to the fourth parameter, "beginningOf(it)."

• The "with" statement references the second "pictureBox" by index [2].

• The "try" statement looks for the "Glove.tiff" file in the location previously defined by "thepath." If the file exists in this location, the "set" event specifies "Glove.tiff" as the "image" in the second picture box.

• If "Glove.tiff" does not exist in the location defined by "thepath," the script will continue with the "if" statement. The statement in the "if" construct calls a Frontier script, "getFileDialog," located at "root.system.verbs.builtins.file." The "getFileDialog" script brings up an **Open** dialog box.

• The first parameter is a message to the user shown at the bottom of the dialog box, "Open the image named 'Glove.tiff.'" Once the user locates the image file and clicks **OK**, the "set" event imports the image.

• The last "set" event references the "image" by index [1]. The "bounds" property of the image is set to "exactFit."

Create and position copies of the picture box

UserTalk

```
clone(pictureBox[2],after(pictureBox[2]))
with pictureBox[3]
set(bounds,textrect("12.471 cm","2 cm","13.971 cm","3.5 cm"))
clone(pictureBox[2],after(pictureBox[3]))
with pictureBox[4]
set(bounds,textrect("16.471 cm","2 cm","17.971 cm","3.5 cm"))
clone(pictureBox[2],after(pictureBox[4]))
with pictureBox[5]
set(bounds,textrect("20.471 cm","2 cm","21.971 cm","3.5 cm"))
```

AppleScript

```
  duplicate picture box 2 to after picture box 2
  tell picture box 3
    set bounds to {"12.471 cm", "2 cm", "13.971 cm", ¬
    "3.5 cm"}
  end tell
  duplicate picture box 2 to after picture box 3
  tell picture box 4
    set bounds to {"16.471 cm", "2 cm", "17.971 cm", ¬
"3.5 cm"}
  end tell
  duplicate picture box 2 to after picture box 4
  tell picture box 5
    set bounds to {"20.471 cm", "2 cm", "21.971 cm", ¬
"3.5 cm"}
  end tell
end tell
```

Concepts: The “clone” and “set” events above are similar to using the Step and Repeat feature.

• The first “clone” event references the second “pictureBox” by index [2]. A copy of the “pictureBox” is placed after the second “pictureBox.”

• The first “with” statement references the new “pictureBox” by index [3].

• The first “set” statement calls the “textRect” script, which sets numerical values for the picture box boundaries. The four parameters indicate how far from the top left corner of the document page the top, left, bottom, and right sides of the box will be positioned.

• The following lines use the object hierarchy to clone “pictureBox[2]” after “pictureBox[3],” then after “pictureBox[4].” Each new “pictureBox” is referenced by index and positioned.

Create the third picture box and import a picture

UserTalk

```
with document[1].page[1]
create(pictureBox,0,0,endOf(it))
with pictureBox[6]
  set(bounds,textRect("6.875 cm","18.425 cm","12.729 cm",
  "26.4 cm"))
  set(color,"None")
  try
    set(image[1],alias(thepath+"Helmet.tiff"))
  else
    if (file.getFileDialog("Open the image named
    'Helmet.tiff'.",@filepath,'TIFF'))
      set(image[1],alias(filepath))
  set(image[1].scale,textpoint("70","70"))
  set(image[1].offset,textpoint("0.557 cm","1.254 cm"))
```

AppleScript

```
tell page 1 of document 1
    make picture box at end with properties ¬
    {bounds:{"6.875 cm", "18.425 cm", "12.729 cm", ¬
    "26.4 cm"}, color:"None"}
  tell picture box 6
    try
      set image 1 to alias (thepath & ¬
      "Helmet.TIFF")
    on error
      set image 1 to (choose file with prompt ¬
      "Please select the file \"Helmet.TIFF\"."¬
      of type {"TIFF"})
    end try
    tell image 1
      set scale to {"70", "70"}
      set offset to {"0.557 cm", "1.254 cm"}
    end tell
  end tell
end tell
```

Concepts: The statements above are similar to creating a picture box and importing a picture. The properties are specified with “set” events rather than by using a “paramlist” as the third parameter in a “create” event.

• The “with” statement references the first “page” of the active “document”; both are referenced by index [1].

• The “create” event makes a “pictureBox” with default specifications (indicated by the “0” in the second and third parameters). The picture box is placed according to the fourth parameter “endOf(it).”

• The second “with” statement references the third “pictureBox” by index [3].

• The first “set” event calls the “textRect script” (located in Frontier’s “QXP.defs” table) to set numerical values for the picture box boundaries.

• The four parameters indicate how far from the top left corner of the document page the top, left, bottom, and right sides of the box will be positioned.

• The second “set” events sets the background “color” property of the “pictureBox” to “None.”

• The “try” statement looks for the “Helmet.tiff” file in the location previously defined by “thepath.” If the file exists in this location, the “set” event specifies “Helmet.tiff” as the “image” in the second picture box.

• If “Glove.tiff” does not exist in the location defined by “thepath,” the script will continue with the “if” statement. The statement in the “if” construct calls a Frontier script, “getFileDialog,” located at “root.system.verbs.builtins.file.” The “getFileDialog” script brings up an Open dialog box.

• The first parameter is a message to the user shown at the bottom of the dialog box, “Open the image named ‘Helmet.tiff.’” Once the user locates the image file and clicks OK, the “set” event imports the image.

• The fourth “set” event sets the “scale” property of the “image” to 70%. The last “set” event sets the “offset” property of the “image” in centimeters. In both cases, the “textPoint” script passes two values to be coerced into a property-specific data type by QuarkXPress.

Create a vertical line

UserTalk

```
with document[1].page[1]
create(lineBox,0,0,beginningOf(it))
with lineBox[1]
  set(leftPoint,textPoint("0 cm","21.406 cm"))
  set(rightPoint,textPoint("8 cm","21.406 cm"))
  paramlist = 0
  putAppleListItem("Magenta",color,@paramlist)
  putAppleListItem(3,Width,@paramlist)
  putAppleListItem(dotted,Style,@paramlist)
  set(Properties,paramlist)
```

AppleScript

```
tell page 1 of document 1
  make line box at end with properties {left ¬
  point:{"0 cm", "21.406 cm"}, right point:{"8 cm", ¬
  "21.406 cm"}}
  tell line box 1
    set color to "Magenta"
    set width to 3
    set style to dotted
  end tell
```

Concepts: The statements above are similar to creating and positioning a vertical line with the Line tool and choosing a **Color**, **Width**, and **Line Style** from the **Style** menu.

• The “with” statement references the first “page” of the active “document”; both are referenced by index [1].

• The “create” event makes a “lineBox” with default specifications (indicated by the “0” in the second and third parameters). The line is placed according to the fourth parameter “beginningOf(it).”

• The second “with” statement references the “lineBox” by index [1].

• The next two “set” events specify the “leftPoint” and “rightPoint” of the line in centimeters. The “textPoint” script passes two values to be coerced into a property-specific data type by QuarkXPress. In this case, the values are coerced into centimeters.

• The “paramlist = 0” statement clears any information currently in the “paramlist.”

• The “putAppleListItem” statements add specifications for the “color,” “lineWidth,” and “lineStyle” properties to the given “paramlist.”

• The last “set” event sets the “properties” of the “lineBox” according to the “paramlist.”

Create a horizontal line

UserTalk

```
with document[1].page[1]
create(linebox,0,0,endof(it))
with linebox[2]
  set(leftpoint,textpoint("8 cm","2 cm"))
  set(rightpoint,textpoint("8 cm","32 cm"))
  set(width,0.5 )
```

AppleScript

```
  make line box at end
  tell line box 2
    set left point to {"8 cm", "2 cm"}
    set right point to {"8 cm", "32 cm"}
    set width to 0.5
  end tell
end tell
```

Concepts: The statements above are similar to creating and positioning a horizontal line with the Line tool and choosing a **Width** from the **Style** menu.

• The “with” statement references the first “page” of the active “document”; both are referenced by index [1].

• The “create” event makes a “lineBox” with default specifications (indicated by the “0” in the second and third parameters). The line is placed according to the fourth parameter “endOf(it).”

• The second “with” statement references the second “lineBox” by index [2].

• The next two “set” events specify the “leftPoint” and “rightPoint” of the line in centimeters. The “textPoint” script passes two values to be coerced into a property-specific data type by QuarkXPress. In this case, the values are coerced into centimeters.

• The last “set” event sets the “lineWidth” property of the “lineBox” to “0.5.” Line widths are set according to points, the default line measurement system in QuarkXPress.

Hide guides and save the document

UserTalk

```
set(document[1].guidesShowing,false)
with document[1]
saveDocument(document[1],thepath+"Constructed Document")
```

AppleScript

```
set guides showing of document 1 to false
save document 1 in (thepath & "Constructed Document")
```

Concepts: The lines above simulate choosing **Hide Guides** from the **View** menu and **Save As** from the **File** menu.

- The first "set" event determines whether the document will have "guidesShowing." The Boolean operator "false" indicates that guides will not be showing.
- The "with" statement references the active "document" by index [1].
- The "save" event saves the document in "thepath" with the name "Constructed Document."

Reset default specifications for future documents

UserTalk

```
  with defaultdocument[1]
    set(pageHeight,oldHeight)
    set(pageWidth,oldWidth)
    set(autoTextBox,oldAutoTextBox)
    set(guidesShowing,oldGuidesShowing)
    set(horizontalMeasure,xDocMeasure)
    set(verticalMeasure,yDocMeasure)
ConstructDocument()
```

AppleScript

```
tell default document 1
    set page height to oldHeight
    set page width to oldWidth
    set automatic text box to oldAutoTextBox
    set guides showing to oldGuidesShowing
    set horizontal measure to xDocMeasure
    set vertical measure to yDocMeasure
  end tell
end tell
```

Concepts: The “set” statements above replace the document default specifications with your original specifications.

• The “with” statement references the current “defaultDocument” by index [1].

• Each “set” statement specifies a property of the “defaultDocument” according to the local variable. For example, the “pageHeight” property is specified as the variable “oldHeight.” The original page size, automatic text box setting, whether the guides are showing, and the default measurement system are reset.

Definitions and Examples — Apple Events Terminology

This section provides definitions and examples (in UserTalk and AppleScript) for object references, insertion points, and each event that QuarkXPress supports.

Once you are familiar with the scripting language's syntax, you should be able to write scripts for QuarkXPress by referring to the information in Chapter 3 and the AppleScript Dictionary.

This section also includes definitions of object reference forms and insertion points, including descriptions of their usage and examples in both UserTalk and AppleScript. The examples are taken from various scripts and are shown out of context.

Format

Each event is listed with a description of its usage, a prototype in UserTalk and AppleScript, and any applicable possible values and results. The terms and events are shown in the following format:

Term or Event

Usage: Description of when to use this term or event.

UserTalk

```
UserTalk prototype with parameters in italics
```

AppleScript

```
AppleScript prototype with parameters in italics
```

Possible Values: variable: options in Times font

Result: Result in Times font

Object Reference Forms

An Apple event message must reference a specific object in an application to communicate. The reference first identifies the container enclosing the object you're specifying. It then uses a reference form to separate a specific object from all possible objects in the container. The reference form can be defined by index, name, range, relative position, or test.

Index

Usage: To identify ordered elements in a container with an integer number.

UserTalk

```
set(word[2].character[1].kern,-14)
```

AppleScript

```
set the kern of character 1 of word 2 to -14
```

Name

Usage: To identify named objects with a text string.

UserTalk

```
set(pictureBox["Robert"].runaround,manualRunaround)
```

AppleScript

```
set runaround of picture box "Robert" to manual runaround
```

Range

Usage: To identify a range of objects.

UserTalk

```
set(word[2to5]).color,"cyan")
```

AppleScript

```
set color of word 2 through word 5 to "cyan"
```

Relative Position

Usage: To identify objects that are before or after other objects.

UserTalk

```
set(textBox[2].pictureBox[next].rotation,45)
```

AppleScript

```
set the rotation of the picture box after text box 2 to 45
```

Test

Usage: To identify objects that meet certain conditions.

UserTalk

```
set (textStyleRange[color.name=="red"],"blue")
```

AppleScript

```
set the color of (every word whose color = "red") to "blue"
```

Insertion Points in the Hierarchy

An insertion point specifies where you want to place an object within the container hierarchy. As you create and insert objects in the hierarchy, the index reference form for existing objects may change.

After

Usage: To insert or create an object after a specified object (the specified object will not be the container). For example, use "after" to move the first paragraph in a story to after the seventh paragraph.

UserTalk

```
move(paragraph[1],after(paragraph[7]))
```

AppleScript

```
move paragraph 1 to after paragraph 7
```

Before

Usage: To insert or create an object before the specified object (the specified object will not be the container). For example, use "before" to paste a copy of the fifth word before the first word in a sentence.

UserTalk

```
duplicate(word[5],before(word[1]))
```

AppleScript

```
duplicate word 5 to before word 1
```

Beginning

Usage: To insert or create an object at the beginning of the specified container. For example, use "beginning" to insert a word as the first word of a paragraph.

UserTalk

```
move(word[1],beginningOf(paragraph[1]))
```

AppleScript

```
move word 1 to beginning of paragraph 1
```

Ending

Usage: To insert or create an object at the end of the specified container. For example, use “ending” to create a text box that is the last text box on the document page.

UserTalk

```
create(textBox,0,0,endOf(it))
```

AppleScript

```
make text box at end
```

Replace

Usage: To replace the specified object with a new object. For example, use “replace” to substitute a new text box for the third text box in a document. AppleScript uses “make” to create an object in the same location, rather than “replace.”

UserTalk

```
create(textBox,0,0,replace(textBox[3]))
```

AppleScript

```
make text box at text box 3
```

Definitions and Examples — Events Supported by QuarkXPress

This section provides definitions and examples (in UserTalk and AppleScript) for object references, insertion points, and each event that QuarkXPress supports.

Once you are familiar with the scripting language's syntax, you should be able to write scripts for QuarkXPress by referring to the information in Chapter 3 and the AppleScript Dictionary.

This section also covers the events in the five Suites supported by QuarkXPress: the Required, Standard, Miscellaneous, Text, and QuarkXPress Suites.

Standard Suite

The Standard Suite consists of basic events most applications use to communicate.

Close

Usage: To close a specified object and determine whether to save it. "Close" is usually used for a window or document.

UserTalk

```
Close(reference,save options,alias)
```

AppleScript

```
Close reference saving save options saving in alias
```

Possible Values: *saving:* yes, no, ask *saving in:* file alias

Clone/Duplicate

Usage: To copy the data and properties of a specified object and create a new object with the same data and properties. You can specify an insertion point for the new object. (If you don't specify a new insertion point, the new object is placed in the same container as the initial object, at the end of the container's elements.) "Duplicate" is similar to "create."

UserTalk

```
duplicate(reference,insertion location)
```

AppleScript

```
duplicate reference to insertion location
```

Possible Values: *to:* location reference (See the "Apple Events Terminology" section of this document for insertion point information.)

Result: Reference (to the new object)

Count

Usage: To determine how many objects in a specified class are contained in a specified object.

UserTalk

```
Count(reference,object class)
```

AppleScript

```
Count of object class in reference
```

Possible Values: *each:* type class (any object class)

Result: Integer

Create/Make

Usage: To create a new element of an object. You can specify the type of object you want to create, set properties in the new object, and specify an insertion point.

UserTalk

```
create(object type,data,properties,insertion location)
```

AppleScript

```
make object type at insertion location with data data with
properties properties
```

Possible Values:

new: type class (any object class)

at: location reference (See the "Apple Events Terminology" section of this document for insertion point information.)

with data: anything (the initial data for the element)

with properties: record

Results: Reference (to the new object)

Data Size
Usage: To obtain an object's size in bytes.

UserTalk

```
dataSize(reference,type)
```

AppleScript

```
data size reference as type
```

Possible Values: *as:* type class

Result: Integer

Delete
Usage: To remove a specified object from an object or application.

UserTalk

```
delete(reference)
```

AppleScript

```
delete reference
```

Exists
Usage: To check for the existence of a specified object.

UserTalk

```
exists(reference)
```

AppleScript

```
exists reference
```

Result: Boolean

Get
Usage: To determine the data structure for an object. "Get" and "set" are usually used to read and write an object's internal data and properties, rather than the whole object.

UserTalk

```
Get(reference)
```

AppleScript

```
Get reference
```

Result: The properties of the object you reference

Get As

Usage: To determine the data structure for an object in a specific data type.

UserTalk

```
GetAs(reference.property,type)
```

AppleScript

```
Get property of reference as type
```

Possible Values: *as:* data type (See the "Data Coercion Chart" later in this section.)

Result: The properties of the object you reference in the data type you specify

Move

Usage: To change an object's position in an application's container hierarchy. The specified object is moved from its current location to the specified insertion point. "Move" is not used to change the physical location of an object. To change the physical location, you would use "set" to change its properties.

UserTalk

```
Move(reference,insertion location)
```

AppleScript

```
Move reference to insertion location
```

Possible Values: *to:* location reference (See the "Apple Events Terminology" section of this document for insertion point information.)

Result: Reference (to the object in its new location)

Open

Usage: To open an object or file.

UserTalk

```
Open(reference,useDocPrefs,remapFonts,doAutoPictureImport)
```

AppleScript

```
Open reference, use doc prefs use doc prefs remap fonts remap fonts do
auto picture import do auto picture import)
```

Possible Values:

use doc prefs: `yes, no, ask`

remap fonts: `no, ask`

do auto picture import: `yes, no, ask`

Print

Usage: To print a specified object.

UserTalk

```
print(reference,copies,OPI,cover page,paper source to alias,list
of plates)
```

AppleScript

```
print reference, copies copies OPI OPI cover page cover page paper
source paper source to alias plates list of plates
```

Possible Values:

copies: small integer

cover page: `no, first page, last page`

OPI: `omit TIFF, omit TIFF and EPS, include images`

paper source: `paper cassette, manual feed`

plates: a list of strings (names of process/spot color specs)

PostScript file: alias (a file path)

Quit

Usage: To quit QuarkXPress and close all open documents.

UserTalk

```
quit
```

AppleScript

```
quit
```

Possible Values: *saving:* yes, no, ask

Save

Usage: To save a specified object to a specified file on disk.

UserTalk

```
Save(reference,alias,filetype,EPS Format,EPS Data,EPS OPI)
```

AppleScript

```
Save reference in alias as file type EPS format EPS format EPS data
EPS data OPI OPI
```

Possible Values:

in: alias

as: type class

template: Boolean

include preview: Boolean

EPS Format: Mac black and white, Mac color, Mac DCS, Mac DCS2, PC black and white, PC color, PC DCS, PC DCS2

EPS Data: ASCII EPS, binary EPS, clean EPS

OPI: omit TIFF, omit TIFF and EPS, include images

bleed: vertical measurement

scale: percent

version: vers 33, vers 40, vers current

Set

Usage: To change an object. "Get" and "set" are usually used to read and write an object's internal data and properties, rather than the whole object.

UserTalk

```
Set(reference,data,replacing)
```

AppleScript

```
Set reference to data replacing
```

Possible Values:

data: object specific data

replacing:* `ask/ignore/replace/rename`

*Replacing is used for importing text with style sheets in any text file format supported by QuarkXPress.

Miscellaneous Suite

The Miscellaneous Suite consists of functions related to the clipboard and other menu-driven functions.

Copy

Usage: To place a copy of the selected object on the clipboard.

UserTalk

```
copy
```

AppleScript

```
copy
```

Cut

Usage: To place the selected object on the clipboard.

UserTalk

```
cut
```

AppleScript

```
cut
```

Do Script

Usage: To execute a script entirely before showing the results.

UserTalk

```
doScript(data,type)
```

AppleScript

```
do script data script type type
```

Possible Values: *script type:* type class

Result: Anything (result of the script execution)

Paste

Usage: To place the data on the clipboard into a designated/selected container.

UserTalk

```
paste
```

AppleScript

```
paste
```

Revert

Usage: To restore an object to its last saved state.

UserTalk

```
revert(reference)
```

AppleScript

```
revert reference
```

Show

Usage: To bring an object into view; also changes the object's index reference form.

UserTalk

```
show(reference)
```

AppleScript

```
show reference
```

Uniform

Usage: To confirm that all objects in a group have the same value for a specified property.

UserTalk

```
uniform(reference, property)
```

AppleScript

```
uniform reference in property
```

Possible Values: *in:* type class

Result: Boolean

QuarkXPress Suite

The Miscellaneous Suite consists of one event related to redraw.

Coerce

Usage: To change data from one type to another type.

UserTalk

```
coerce (reference.property,type)
```

AppleScript

```
coerce property of reference to type
```

Possible Values: *to:* type class (see the "Data Coercion Chart" later in this section)

Result: Anything (result of script execution)

Do Updates

Usage: To redraw the screen after the execution of a script.

UserTalk

```
doUpdates
```

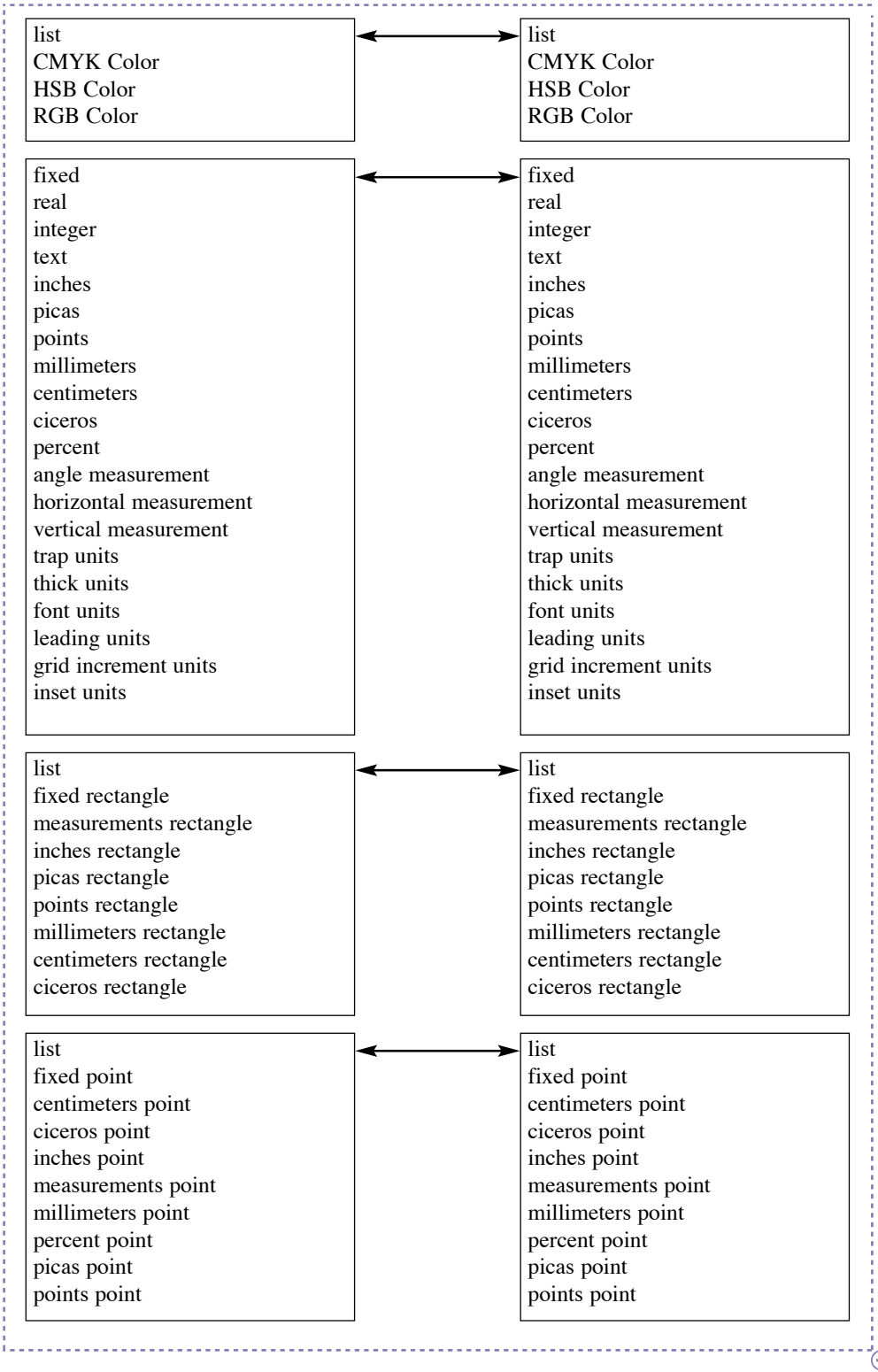
AppleScript

```
do updates
```

Data Coercion Chart

This chart lists the possible data structure you can request with a "Get As" event. The ⇔ indicates that both data types can be coerced into each other. The left-facing arrow ⇐ indicates that the data types on the right can be coerced into the data types on the left.

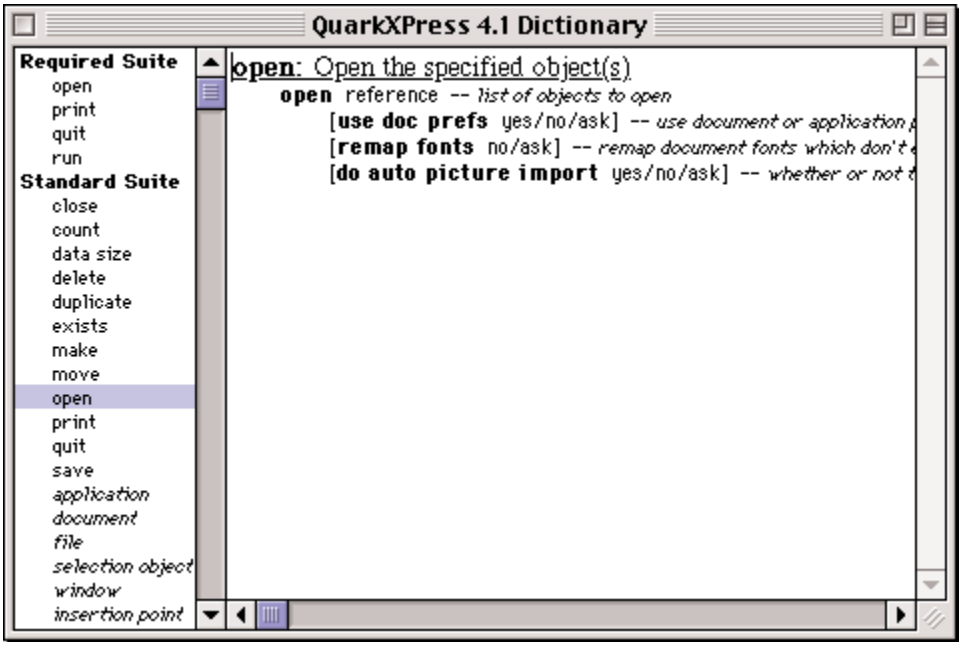
text	⇔	alias
integer	⇔	Boolean text
color spec	⇐	string integer list RGB Color
style spec	⇐	string integer
h and j spec	⇐	string integer



Data Coersion Chart

Using the AppleScript Dictionary

Every scriptable file has an internal dictionary that defines the Apple events it can respond to, as well as the acceptable options or required parameters for those commands. These dictionaries can be accessed by any script-editing application, including Apple's Script Editor. Using Script Editor, choose **Open Dictionary** from the **File** menu and navigate to the intended application. If an application displays in the directory dialog box, it has a scripting dictionary, and can be considered scriptable. When you open the dictionary, a two-part window displays:



The AppleScript dictionary window

Although Apple events definitions are built into an application itself, the dictionary interface is provided by the specific script-editing application. Third-party script editors may have additional capabilities above and beyond those of Apple's Script Editor program. If your scripting language of choice is Userland Frontier, the Doc Server utility from Userland provides much the same functionality as a dictionary.

Events, objects, and parameters

The left column of the dictionary displays a list of events and the objects on which those events can operate. Events display in Roman typeface, while objects appear in italics. (Bold typeface indicates words or phrases that have special meaning to the application.)

When one or more items are selected in the left column, their definitions display in the right column. In the window shown above, the Open event in the Standard Suite is selected. The right column displays the various parts of the event as well as the types of information the event expects. An appropriate open event might be:

```
Open alias(˝Hard Drive:Desktop Folder: my Document˝) use doc prefs yes
remap fonts no
```

In this case, the script uses the parameters "use doc prefs" and "remap fonts", but does not use "do auto picture import." The square brackets ([ ]) in the dictionary indicate that the "do auto picture import" parameter is optional.

Elements and properties

When viewing an entry for objects, you may see additional subheadings in the right column called "Elements and Properties."

In simple terms, elements can be thought of as objects that "belong" to the selected object in the hierarchy. For example, a page can hold generic, text, picture, and line boxes, as well as images, and horizontal and vertical guides, so you will see these listed as *elements* of the page. *Properties,* on the other hand, are characteristics of the object itself. Using *page* as an example again, you will find properties such as margin sizes, column counts, and gutter widths. These are not objects themselves, but do describe how a page appears and behaves.

Inherited properties

You may also notice an entry under properties labeled *<inheritance>*. This indicates which other objects contribute to the appearance and behavior of the selected object. For example, select *text box* in your QuarkXPress dictionary. Under properties, these three entries display:

```
<inheritance> generic box -- All properties and elements of the given
class are inherited by this class.
<inheritance> text path properties -- All properties and elements of
the given class are inherited by this class.
<inheritance> text container properties -- All properties and elements
of the given class are inherited by this class.
```

This means that a text box, while having certain specific properties of its own, also has all the properties shown in the dictionary entries for generic box, text path properties, text container properties, and for *box properties* and *containing box properties which are inherited by generic box*. In other words, all boxes have a certain set of common properties that define how they behave as boxes. Of those, some boxes have additional properties that enable them to contain other information. Out of those, some boxes are even more specialized to hold only text and therefore have properties appropriate to perform that function.

In practice, this simply means that if you want to change the color of a text box you would write:

```
set color of text box 1 to "green"
```

Color is not a property of the text box class, but we can use it as such, because it has been inherited from *box properties, which is inherited by generic box*. Likewise, if you want to change the shape to rounded corner, you can write:

```
set skew of text box 1 to 20
```

Once again, skew is not defined in the text box entry, but is picked up through inheritance from *containing box properties*. Consequently, when using an Apple events dictionary, you may want to think of <*inheritance*> as a scripting equivalent of *see also* in an ordinary dictionary.

Reference Material for QuarkXPress Objects

application Events and Examples

Verb	Frontier Example	AppleScript Example
count	count(document,application[“QuarkXPress”])	count of each document in application “QuarkXPress”
data size	datasize(application[“QuarkXPress”].name,inttype)	data size of name of application “QuarkXPress” as integer
get	get(application[1].autosave)	get auto save of application 1
get as	getas(application[“QuarkXPress”].name,stringtype)	get name of application “QuarkXPress” as string
set	set(application[1].doclayoutshowing,true)	set doc layout showing of QuarkXPress to true

application Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
application	•	•			
color system	•	•			
default document	•				
delimit table	•				
document	•	•	•	•	•
file		•			
menu	•	•			
window	•	•	•	•	•

application Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	about showing	Boolean	If TRUE, then the about is showing.
	auto backup	Boolean	If TRUE, then the auto backup option is on.
	auto lib save	Boolean	If TRUE, saves changes to a library automatically whenever an entry is added.
	auto save	Boolean	If TRUE, auto save is on.
	auto save interval	fixed	Amount of time (in minutes) between each auto save
	backup destination	alias	Destination folder of backup files.
•	best type	type class	The best descriptor type.
•	class	type class	The class.
	clipboard showing	Boolean	If TRUE, then the clipboard is showing.
	color TIFF resolution	use 8 bit/ use 16 bit/ use 32 bit	Resolution at which to display color TIFF images.
	colors showing	Boolean	If TRUE, then the Colors palette is showing.
	convert quotes	Boolean	If TRUE, then convert standard quotes to typographer’s quotes when importing text.
	current box	reference	Currently selected box.
•	default type	type class	The default descriptor type.

application Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	doc layout showing	Boolean	If TRUE, then the Document Layout palette is showing.
•	font list	a list of font record	List of fonts available to application.
•	frontmost	Boolean	Is this the frontmost application?
	full resolution import	Boolean	If TRUE, then import TIFFs at 72 dpi
	gray TIFF resolution	use 16 levels/ use 256 levels	Resolution at which to display gray scale TIFF images.
	grid guide color	RGB color	Color of grid guides.
	import styles	Boolean	If TRUE, then import style sheets when importing text.
•	language	English/Japanese/ Traditional Chinese/ Simplified Chinese/ Korean	Program language (this property is r/o for single language versions of QuarkXPress only).
	live scroll	Boolean	If TRUE, then perform live scrolling.
	margin guide color	RGB color	Color of margin guides.
	maximize document bounds	Boolean	If TRUE, maximize document boundary when zooming tiling or stacking.
	measurements showing	Boolean	If TRUE, then the measurements palette is showing.
•	name	plain text (string)	Name of this application.
•	object reference	reference	An object reference for this object.
	offscreen draw	Boolean	If TRUE, then off screen draw is on.
	open document preference	keep document settings/use application preferences/ask user	Settings to use when opening a document.
	page grabber	Boolean	If TRUE, then page grabbing function is on.
	pasteboard width	percent	Width of the pasteboard (in percent).
	properties	record	Property that allows setting of a list of properties.
	quote types	small integer	Type of quotes to use for smart quotes.
	registration marks offset	fixed	Registration marks offset.
	ruler guide color	RGB color	Color of ruler guides.
	scroll speed	small integer	Speed of scrolling.
	selection	selection object	The selection visible to the user.
	smart quotes	Boolean	If TRUE, then converts standard quotes to typographer's quotes.
	style sheets showing	Boolean	If TRUE, then the style sheets palette is showing.
	tile to multiple monitors	Boolean	If TRUE, tiles documents to multiple monitors.
	tools showing	Boolean	If TRUE, then the Tools palette is showing.
	total backups	small integer	Total number of backups to keep.
	trap information showing	Boolean	If TRUE, then the Trap Information palette is showing.
•	version	version	The version of the application.

document Events and Examples

Verb	Frontier Example	AppleScript Example
close	close(document[“Kristy”])	close document “Kristy”
count	count(story,document[1])	count of each story in document 1
create	create(document,0,0,beginningof(it))	make document at beginning
data size	data size(document[1].name,inttype)	data size of name of document 1 as integer
get	get(document[1].name)	get name of document 1
get as	getas(document[2].filepath,stringtype)	get file path of document 2 as string
open	open(document[“Test”],yes,ask,ask)	open document “Test” use doc prefs yes remap fonts ask do auto picture import ask
print	print(document[all])	print every document
save	save(document[1],alias(“Constructed Document”))	save document 1 in “Constructed Document”
set	set(document[1].keepmasterpageitems,true)	set keep master page items of document 1 to true
show	show(document[middle])	show middle document

document Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character spec	•	•	•	•	•
color spec	•	•	•	•	•
fontset spec (East Asian only)	•	•	•	•	•
generic box	•	•	•	•	•
graphic box		•	•	•	•
h and j spec	•	•	•	•	•
image	•	•	•	•	•
line box	•	•	•	•	•
page	•	•	•	•	•
picture box	•	•	•	•	•
spread	•	•	•	•	•
story	•	•	•	•	•
style spec	•	•	•	•	•
text box	•	•	•	•	•

document Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	auto constrain	Boolean	If TRUE, then automatically constrain limits of items within boxes.
	auto kern	Boolean	TRUE if auto kerning should apply.
	auto leading	percent	Specifies the value to use for auto leading.
	auto page insertion location	no auto page insertion/end of story/ end of section/ end of document	Specifies where new pages are inserted.
	auto picture import	auto import off/ auto import on/ auto import verify	Automatically updates modified pictures since last opened.
	automatic text box	Boolean	If TRUE, create an automatic text box for each new page.
	automatic trap amount	trap units	Specifies auto trap amount.
	auxiliary dictionary path	alias	The path of the auxiliary dictionary file of this document.
	baseline grid increment	grid increment units	Specifies the baseline grid interval.
	baseline grid showing	Boolean	If TRUE, baseline grid is showing.
	baseline grid start	vertical measurement	Specifies baseline grid start.
•	best type	type class	The best descriptor type.
	bottom margin	vertical measurement	The location of the bottom margin of a page in this document.
	ciceros per centimeter	fixed	Specifies number of ciceros per centimeter.
•	class	type class	The class.
	column count	small integer	Number of columns in this document.
	current box	reference	Currently selected box.
	current page	page	Current page displayed to user.
	current spread	spread	Current spread displayed to user.
•	default spread count	integer	Default Spread Count.
	default tool mode	drag mode/contents mode rotate mode/view mode/text mode/ rounded rect mode/oval mode/poly mode/ pic mode/rounded rect pic mode/oval pic mode/ poly pic mode/ orthogonal line mode/ line mode	Default tool mode.
•	default type	type class	The default descriptor type.
	enhanced hyphenation method	Boolean	If TRUE, enhanced hyphenation mode is active.
	facing pages	Boolean	If TRUE, creates double-sided pages.
•	file path	alias	File path of this document.
	flex space width	percent	Specifies value for custom width space.
•	flow version	fixed	Document flow version.
•	font list	a list of font record	List of fonts used in this document.
	fractional character widths	Boolean	If TRUE, print characters using fractional widths (default). If FALSE, print character widths using integral widths.

document Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	frame inside	Boolean	If TRUE, then places frames inside text or picture boxes.
	greek below	font units	Specifies text size below which to display text as gray lines.
	greek pictures	Boolean	If TRUE, then displays pictures as grayed images.
	guides in front	Boolean	If TRUE, then places guides in front of all boxes.
	guides showing	Boolean	If TRUE, then guides are showing.
	gutter width	horizontal measurement	The width of the default text box's gutter in this document.
	horizontal measure	inches/inches decimal/ picas/points/ millimeters/centimeters ciceros/agates/ Qs (East Asian only)	Horizontal measurement units.
	ignore white	Boolean	If TRUE, then specifies that an object color in front of multiple backgrounds that include white not take white into account when trapping.
	indeterminate trap amount	trap units	Specifies value for trapping to indeterminate background color.
•	index	integer	Index of object.
	inside margin	horizontal measurement	The location of the inside margin of a page in this document (with facing pages TRUE).
	invisibles showing	Boolean	If TRUE, then invisible characters are showing.
	item spread coords	Boolean	If TRUE, then displays items in spread coordinates.
	keep master page items	Boolean	If TRUE, then specifies whether modified master items are kept or removed when they are modified on the master.
	kern above	font units	Size of text above which auto kerning should apply.
	knockout limit	percent	point at which an object color knocks out of the background color.
	left margin	horizontal measurement	The location of the left margin of a page in this document.
	ligatures on	Boolean	If TRUE, combine certain characters into a single character (ligatures).
	lock guides	Boolean	If TRUE, then lock guides.
	low quality blends	Boolean	If TRUE then display banded blends (faster).
	maintain leading	Boolean	If TRUE then specifies that the baseline of a line that falls immediately below an obstruction is placed according to its applied leading value.
	maximum ligature track	fixed	The maximum amount that ligatures can be tracked or kerned apart before they break into separate characters.
	maximum view scale	percent	Specifies the largest document view using the zoom tool.
	minimum view scale	percent	Specifies the smallest document view using the zoom tool.
•	modified	Boolean	Has the document been modified since the last save?
•	name	plain text (string)	Name of this document.
•	object reference	reference	An object reference for this object.

document Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	outside margin	horizontal measurement	The location of the outside margin of a page in this document (with facing pages TRUE).
	overprint limit	percent	Specifies the value for shade of color below which it will not overprint.
	page height	vertical measurement	The height of a page in this document.
	page rule origin	measurements point	Location of the page's ruler origin
	page width	horizontal measurement	The width of a page in this document.
	points per inch	fixed	Specifies number of points per inch.
	print setup	print setup record	Settings used when printing this document.
	process trap	Boolean	If TRUE, process trapping is on.
	properties	record	Property that allows setting of a list of properties.
	Q measurement	Boolean	If TRUE, use Q for measurements (East Asian only)
	right margin	horizontal measurement	The location of the right margin of a page in this document.
	Roman Extra	percent	percent of space between Roman and Japanese characters (East Asian only)
	rulers showing	Boolean	If TRUE, then rulers are showing.
	small caps horizontal scale	percent	Value of horizontal scale for small cap characters.
	small caps vertical scale	percent	Value of vertical scale for small cap characters.
	snap distance	small integer	Specifies distance within which items snap to guides.
	spread height	vertical measurement	The height of a spread (including pasteboard) in this document.
	spread rule origin	measurements point	Location of the spread's ruler origin.
	spread width	horizontal measurement	The width of a spread (including pasteboard) in this document.
	subscript horizontal scale	percent	Value of horizontal scale for subscript characters.
	subscript offset	percent	Value of offset for subscript characters.
	subscript vertical scale	percent	Value of vertical scale for subscript characters.
	superscript horizontal scale	percent	Value of horizontal scale for superscript characters.
	superscript offset	percent	Value of offset for superscript characters.
	superscript vertical scale	percent	Value of vertical scale for superscript characters.
	superior horizontal scale	percent	The horizontal scale for superior characters
	superior vertical scale	percent	The vertical scale for superior characters
	tool mode	drag mode/contents mode/rotate mode/view mode/text mode/ rounded rect mode/oval mode/poly mode/pic mode/rounded rect pic mode/oval pic mode/ poly pic mode/ orthogonal line mode/ line mode	The tool that is currently selected.
	top margin	vertical measurement	The location of the top margin of a page in this document.

document Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	trapping method	absolute trap/ proportional trap/ knock all	The trapping method to be used.
	typesetting leading mode	Boolean	If TRUE, specifies value of leading upward from the baseline of one line of text to the baseline of the line above it. If FALSE, specifies Word Processing mode, which measures leading downward from the top of the ascent on the line below it.
•	version	vers 33/vers 40/ vers current/vers other	The flow version of the document.
	vertical measure	inches/inches decimal/ picas/points/millimeters/ centimeters/ciceros/ agates/ Qs (East Asian only)	Vertical measurement units.
	view scale	fit page in window/ fit spread in window/ thumbnails, or percent	The current view scale of this document.
	view scale increment	percent	Specifies the percent of change in view for each mouse click using the zoom tool.
	vStory direction	Boolean	If TRUE, then the story will be vertical (East Asian only)

window Events and Examples

Verb	Frontier Example	AppleScript Example
close	close(window[1])	close window 1
data size	datasize(window[1].name,inttype)	data size of name of window 1 as integer
exists	exists(window[3])	exists window 3
get	get(window[1].name)	get name of window 1
get as	getas(window[1].bounds,inttype)	get bounds of window 1 as integer
set	set(window[1].index,2)	set index of window 1 to 2
show	show(window[2])	show window 2

window Elements and Reference Forms

None

window Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	best type	type class	The best descriptor type.
	bounds	rectangle	The boundary rectangle for the window.
•	class	type class	The class.
•	closeable	Boolean	Does the window have a close box?
•	default type	type class	The default descriptor type.
•	floating	Boolean	Does the window float?
•	index	integer	The numbered order of the window.
•	modal	Boolean	Is the window modal?
•	name	plain text (string)	Window name (title).
•	object reference	reference	An object reference for this object.
	properties	record	Property that allows setting of a list of properties.
•	resizable	Boolean	Is the window resizable?
•	tltled	Boolean	Does the window have a title bar?
	visible	Boolean	Is the window visible?
•	zoomable	Boolean	Does the window have a zoom box?
	zoomed	Boolean	Is the window zoomed?

character Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(character[1]after(charcter[2]))	duplicate character 1 after character 2
count	count(character,word[1])	count of each character in word 1
create	create(character,"s",0,endOf(word[1]))	create character "s" at end of word 1
data size	datasize(character[1].style,inttype)	data size of style of character 1
delete	delete(character[1])	delete character 1
exists	exists(character[1])	exists(character 1)
get	get(character[4].baseshift)	get base shift of character 4
get as	getas(character[4].color,stringtype)	get color of character 4 as string
move	move(character[1],after(character[3]))	move character 1 to after character 3
set	set(character[1].dropcaplines,4)	set drop cap lines of character 1 to 4
show	show(character[last])	show last character

character Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character	•		•	•	•
line	•		•	•	•
paragraph	•		•	•	•
story	•	•	•	•	•
text	•		•	•	•
text style range	•		•	•	•
word	•		•	•	•

character Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	ascent	font units	The maximum ascent of any character in this text.
	base shift	base units	Baseline shift of the first character of this text.
•	baseline	vertical measurement	Vertical offset (from the top of the containing text box) of the baseline of the first character of this text.
•	best type	type class	The best descriptor type.
	character style	character spec	The character spec applied to this text.
•	character type	no type/one byte/ two byte/many types	The type of the character. (East Asian only)
•	class	type class	The class.
	color	color spec	The color of the first character of this text.
	contents	plain text (string)	Contents of this text.
•	default type	type class	The default descriptor type.
•	descent	font units	The maximum descent of any character in this text.
	font	plain text (string)	The name of the font of the first character in this text.
	grouped character	Boolean	If TRUE, then the text is grouped. (East Asian only)
•	height	font units	Height of this text.
•	horizontal offset	horizontal measurement	Horizontal offset (from the left side of the containing text box) of the first character of this text.
	horizontal scale	percent	Horizontal scale of the first character of this text.
	kern	fixed	Kerning of the first character of this text.
•	length	integer	number of characters in this text object
•	object reference	reference	An object reference for this object.
•	offset	integer	Offset (character index) of the first character of this text object within the containing story.
	PinYin	plain text (string)	The PinYin for the text. (Simplified Chinese only)
	properties	record	Property that allows setting of a list of properties.
	rubi	plain text (string)	The rubi for the text. (Japanese and Korean only)
	sending	horizontal measurement	The sending of this text. (East Asian only)
	shade	percent	Shade of the first character of this text.
	size	fixed	The size of the first character of this text in points.
	style	text style info	The text styles applied to this text.
	track	fixed	Tracking of the first character of this text.

character Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	trap text	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Trapping specification for this character.
•	uniform styles	text style info	Text styles that are uniformly applied to this text.
	vertical scale	percent	Vertical scale of the first character of this text.
•	width	horizontal measurement	Width of the first character of this text.
	ZhuYin	plain text (string)	The ZhuYin for the text. (Traditional Chinese only)

line Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(line[1],after(line[3])	duplicate line 1 to after line 3
count	count(character,line[2])	count of each character in line 2
create	create(line,0,0,beginningof(it))	make line at beginning
data size	datasize(line[1].track,inttype)	data size of track of line 1 as integer
delete	delete(line[1])	delete line 1
get	get(line[1].track)	get track of line 1
get as	getas(line[1].justification,stringtype)	get justification of line 1 as string
move	move(line[1],after(line[3]))	move line 1 after line 3
save	saveAS(line[1],alias("Hard Disk:TextFile"),"TEXT")	save line 1 as "TEXT" in file "Hard Disk:TextFile"

line Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character	•		•	•	•
line	•		•	•	•
paragraph	•		•	•	•
story	•	•	•	•	•
text	•		•	•	•
text style range	•		•	•	•
word	•		•	•	•

line Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	ascent	font units	The maximum ascent of any character in this text.
	base shift	base units	Baseline shift of the first character of this text.
•	baseline	vertical measurement	Vertical offset (from the top of the containing text box) of the baseline of the first character of this text.
•	best type	type class	The best descriptor type.
	character style	character spec	The character spec applied to this text.
•	character type	no type/one byte/ two byte/many types	The type of character. (East Asian only)
•	class	type class	The class.
	color	color spec	The color of the first character of this text.
	contents	plain text (string)	Contents of this text.
•	default type	type class	The default descriptor type.
•	descent	font units	The maximum descent of any character in this text.
	font	plain text (string)	The name of the font of the first character in this text.
	grouped character	Boolean	If TRUE, then the text is grouped. (East Asian only)
•	height	font units	Height of this text.
•	horizontal offset	horizontal measurement	Horizontal offset (from the left side of the containing text box) of the first character of this text.
	horizontal scale	percent	Horizontal scale of the first character of this text.
•	justification	left justified/ right justified/centered/ fully justified/force justified	The justification of the text.
	kern	fixed	Kerning of the first character of this text.
•	length	integer	number of characters in this text object
•	object reference	reference	An object reference for this object.
•	offset	integer	Offset (character index) of the first character of this text object within the containing story.
	PinYin	plain text (string)	The PinYin for the text. (Simplified Chinese only)
	properties	record	Property that allows setting of a list of properties.
	rubi	plain text (string)	The rubi for the text. (Japanese and Korean only)
	sending	horizontal measurement	The sending for this text. (East Asian only)
	shade	percent	Shade of the first character of this text.
	size	fixed	The size of the first character of this text in points.
	style	text style info	The text styles applied to this text.
	track	fixed	Tracking of the first character of this text.
	trap text	default/overprint/ knockout/spread auto/ amount/choke auto amount, or fixed	Trapping specification for the first character of this text.
•	uniform styles	text style info	Text styles that are uniformly applied to this text.
	vertical scale	percent	Vertical scale of the first character of this text.
•	width	horizontal measurement	Width of the first character of this text.
	ZhuYin	plain text (string)	The ZhuYin for the text. (Traditional Chinese only)

paragraph Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(paragraph[4],before(paragraph[1]))	duplicate paragraph 4 to before paragraph 1
count	count(word,paragraph[1])	count of each word in paragraph 1
create	create(paragraph,0,0,endof(it))	make paragraph at end
data size	datasize(paragraph[3].leading,inttype)	data size of leading of paragraph 3 as integer
delete	delete(paragraph[1])	delete paragraph 1
get	get(paragraph[4].height)	get height of paragraph 4
get as	getas(paragraph[1].font,stringtype)	get font of paragraph 1 as string
move	move(paragraph[1],after(paragraph[3]))	move paragraph 1 to after paragraph 3
save	save(paragraph[3],"TEXT")	save paragraph 3 as "TEXT"
set	set(paragraph[1].justification,center)	set justification of paragraph 1 to center
show	show(paragraph[1])	show paragraph 1

paragraph Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character	•		•	•	•
line	•		•	•	•
paragraph	•		•	•	•
story	•	•	•	•	•
text	•		•	•	•
text style range	•		•	•	•
word	•		•	•	•

paragraph Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	ascent	font units	The maximum ascent of any character in this text.
	base shift	base units	Baseline shift of the first character of this text.
•	baseline	vertical measurement	Vertical offset (from the top of the containing text box) of the baseline of the first character of this text.
•	best type	type class	The best descriptor type.
	character alignment	top align/center align/ baseline align/ bottom align	The alignment for characters. (East Asian only)
	character style	character spec	The character spec that is applied to this text.
•	character type	no type/one byte/ two byte/many types	The type of this character. (East Asian only)
•	class	type class	The class.
	color	color spec	The color of the first character of this text.
	contents	plain text (string)	Contents of this text.
•	default type	type class	The default descriptor type.
•	descent	font units	The maximum descent of any character in this text.
	drop cap characters	small integer	Specifies the number of drop characters.
	drop cap lines	small integer	Specifies the number of lines the enlarged character(s) drop.
	first indent	horizontal measurement	Specifies the first line indent value.

paragraph Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	font	plain text (string)	The name of the font of the first character in this text.
	grid lock	Boolean	If TRUE, then lock paragraph to baseline grid.
	grouped character	Boolean	If TRUE, then this text is grouped. (East Asian only)
	h and j set	h and j spec	The hyphenation and justification specification applied to this paragraph.
•	height	font units	Height of this text.
•	horizontal offset	horizontal measurement	Horizontal offset (from the left side of the containing text box) of the first character of this text.
	horizontal scale	percent	Horizontal scale of the first character of this text.
	justification	left justified/ right justified/centered/ fully justified/force	Justification for lines of text.
	keep all	Boolean	If TRUE, and keep together is on, then keeps all lines together.
	keep together	Boolean	If TRUE, then keep together is on.
	keep together end	small integer	Specifies number of lines to keep together at end of paragraph.
	keep together start	small integer	Specifies number of lines to keep together at start of paragraph.
	keep with next	Boolean	If TRUE, will not separate paragraph from next paragraph.
	kern	fixed	Kerning of the first character of this text.
•	language	small integer	Language of the paragraph.(This property is R/O for single language versions of QuarkXPress only)
	leading	leading units	Specifies the vertical spacing between lines of text in the paragraph.
	left indent	horizontal measurement	Specifies the left indent value.
•	length	integer	The number of characters in this text object.
•	object reference	reference	An object reference for this object.
•	offset	integer	Offset (character index) of the first character of this text object within the containing story.
	PinYin	plain text (string)	The PinYin for the text. (Simplified Chinese only)
	properties	record	Property that allows setting of a list of properties.
	punct indent	horizontal measurement	The value of Ten Maru gutter for the paragraph. (East Asian only)
	relative leading	Boolean	If TRUE, then leading is relative to largest font on each line.
	right indent	horizontal measurement	Specifies the right indent value.
	rubi	plain text (string)	The rubi for the text. (Japanese and Korean only)
	rule above	rule record	Rule above properties.
	rule below	rule record	Rule below properties.
	sending	horizontal measurement	The sending of this text. (East Asian only)
	shade	percent	Shade of the first character of this text.
	size	fixed	The size of the first character of this text in points.
	space after	vertical measurement	Specifies the value of the space below the last line of the paragraph.
	space before	vertical measurement	Specifies the value of the space above the paragraph.

paragraph Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	style	text style info	The text styles applied to this text.
	style sheet	style spec	Specifies the name or reference of the style spec applied to the paragraph.
	tab list	a list of tab record	A list of the tabs in the paragraph.
	track	fixed	Tracking of the first character of this text.
	trap text	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Trapping specification for the first character of this text.
•	uniform styles	text style info	Text styles that are uniformly applied to this text.
	vertical scale	percent	Vertical scale of the first character of this text.
•	width	horizontal measurement	Width of the first character of this text.
	ZhuYin	plain text (string)	The ZhuYin for the text. (Traditional Chinese only)

story Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(story[1],after(story[2]))	duplicate story 1 to after story 2
count	count(paragraph,story[1])	count of each paragraph in story 1
create	create(story,0,0,beginningOf(it))	make story at beginning
data size	datasize(story[4].name,inttype)	data size of name of story 4 as integer
delete	delete(story[1])	delete story 1
get	get(story[1].contents)	get contents of story1
get as	getas(story[1].color,stringtype)	get color of story 1 as string
move	move(story[1],after(story[2]))	move story 1 to after story 2
save	save(story[1],"TEXT")	save story 1 as "TEXT"
set	set(story[1].font,"Times")	set font of story 1 to "Times"
show	show(story[1])	show story 1

story Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After
Another Element					
character	•		•	•	•
line	•		•	•	•
paragraph	•		•	•	•
story	•	•	•	•	•
text	•		•	•	•
text style range	•		•	•	•
word	•		•	•	•

story Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	ascent	font units	The maximum ascent of any character in this text.
	base shift	base units	Baseline shift of the first character of this text.
•	baseline	vertical measurement	Vertical offset (from the top of the containing text box) of the baseline of the first character of this text.
•	best type	type class	The best descriptor type.
	character style	character spec	The character spec applied to this text.
•	character type	no type/one byte/ two byte/many types	The type of the character. (East Asian only)
•	class	type class	The class.
	color	color spec	The color of the first character of this text.
	contents	plain text (string)	Contents of this text.
•	default type	type class	The default descriptor type.
•	descent	font units	The maximum descent of any character in this text.
	font	plain text (string)	The name of the font of the first character in this text.
	grouped character	Boolean	If TRUE, then the text is grouped. (East Asian only)
•	height	font units	Height of this text.
•	horizontal offset	horizontal measurement	Horizontal offset (from the left side of the containing text box) of the first character of this text.
	horizontal scale	percent	Horizontal scale of the first character of this text.
	kern	fixed	Kerning of the first character of this text.
•	length	integer	number of characters in this text object
	name	plain text (string)	The name of the story.
•	object reference	reference	An object reference for this object.
•	offset	integer	Offset (character index) of the first character of this text object within the containing story.
	PinYin	plain text (string)	The PinYin for the text. (Simplified Chinese only)
	properties	record	Property that allows setting of a list of properties.
	rubi	plain text (string)	The rubi for the text. (Japanese and Korean only)
	sending	horizontal measurement	The sending of this text. (East Asian only)
	shade	percent	Shade of the first character of this text.
	size	fixed	The size of the first character of this text in points.
	style	text style info	The text styles applied to this text.
	track	fixed	Tracking of the first character of this text.
	trap text	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Trapping specification for the first character of this text.
•	uniform styles	text style info	Text styles that are uniformly applied to this text.
	vertical scale	percent	Vertical scale of the first character of this text.
•	vertical story direction	Boolean	If TRUE, then the story is oriented vertically. (East Asian only)
•	width	horizontal measurement	Width of the first character of this text.
	ZhuYin	plain text (string)	The ZhuYin for the text. (Traditional Chinese only)

text Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(text[1],after(text[2]))	duplicate text 1 to after text 2
count	count(words,text[1]))	count of each word in text 1
create	create(text,0,0,endof(it))	make text at end
data size	datasize(text[2].height,inttype)	data size of height of text 2 as integer
delete	delete(text[1])	delete text 1
get	get(text[all].leading)	get leading of text all
get as	getas(text[2].height,inttype)	get height of text 2 as integer
save	save(text[3],"TEXT")	save text 3 as "TEXT"
set	set(text[1].justification,center)	set justification of text 1 to center
show	show(text[1])	show text 1

text Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character	•		•	•	•
line	•		•	•	•
paragraph	•		•	•	•
story	•	•	•	•	•
text	•		•	•	•
text style range	•		•	•	•
word	•		•	•	•

text Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	ascent	font units	The maximum ascent of any character in this text.
	base shift	base units	Baseline shift of the first character of this text.
•	baseline	vertical measurement	Vertical offset (from the top of the containing text box) of the baseline of the first character of this text.
•	best type	type class	The best descriptor type.
	character style	character spec	The character spec applied to this text.
•	character type	no type/one byte/ two byte/many types	The type of the character. (East Asian only)
•	class	type class	The class.
	color	color spec	The color of the first character of this text.
	contents	plain text (string)	Contents of this text.
•	default type	type class	The default descriptor type.
•	descent	font units	The maximum descent of any character in this text.
	font	plain text (string)	The name of the font of the first character in this text.
	grouped character	Boolean	If TRUE, then the text is grouped. (East Asian only)
•	height	font units	Height of this text.
•	horizontal offset	horizontal measurement	Horizontal offset (from the left side of the containing text box) of the first character of this text.
	horizontal scale	percent	Horizontal scale of the first character of this text.
	kern	fixed	Kerning of the first character of this text.

text Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
•	length	integer	number of characters in this text object
•	object reference	reference	An object reference for this object.
•	offset	integer	Offset (character index) of the first character of this text object within the containing story.
	PinYin	plain text (string)	The PinYin for the text. (Simplified Chinese only)
	properties	record	Property that allows setting of a list of properties.
	rubi	plain text (string)	The rubi for the text. (Japanese and Korean only)
	sending	horizontal measurement	The sending of this text. (East Asian only)
	shade	percent	Shade of the first character of this text.
	size	fixed	The size of the first character of this text in points.
	style	text style info	The text styles applied to this text.
	track	fixed	Tracking of the first character of this text.
	trap text	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Trapping specification for the first character of this text.
•	uniform styles	text style info	Text styles that are uniformly applied to this text.
	vertical scale	percent	Vertical scale of the first character of this text.
•	width	horizontal measurement	Width of the first character of this text.
	ZhuYin	plain text (string)	The ZhuYin for the text. (Traditional Chinese only)

word Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(word[“Son of”],before(word[“Bob”]))	duplicate word “Son of” to before word “Bob”
count	count(character,word[1])	count of each character in word 1
create	create(word,”Blue”,0,beginningof(it))	make word at beginning with data “Blue”
data size	datasize(word[1].font,inttype)	data size of font of word 1 as integer
delete	delete(word[1])	delete word 1
get	get(word[4].baseshift)	get base shift of word 4
get as	getas(word[“QuarkXPress”].baseshift,inttype)	get base shift of word “QuarkXPress” as integer
move	move(word[1],after(word[3]))	move word 1 to after word 3
save	save(word[“Blue”],”TEXT”)	save word “Blue” as “TEXT”
set	set(word[1].horizontalscale,30)	set horizontal scale of word 1 to 30
show	show(word[5])	show word 5

word Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character	•		•	•	•
line	•		•	•	•
paragraph	•		•	•	•
story	•	•	•	•	•
text	•		•	•	•
text style range	•		•	•	•
word	•		•	•	•

word Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	ascent	font units	The maximum ascent of any character in this text.
	base shift	base units	Baseline shift of the first character of this text.
•	baseline	vertical measurement	Vertical offset (from the top of the containing text box) of the baseline of the first character of this text.
•	best type	type class	The best descriptor type.
	character style	character spec	The character spec applied to this text.
•	character type	no type/one byte/ two byte/many types	The type of the character. (East Asian only)
•	class	type class	The class.
	color	color spec	The color of the first character of this text.
	contents	plain text (string)	Contents of this text.
•	default type	type class	The default descriptor type.
•	descent	font units	The maximum descent of any character in this text.
	font	plain text (string)	The name of the font of the first character in this text.
	grouped character	Boolean	If TRUE, then the text is grouped. (East Asian only)
•	height	font units	Height of this text.
•	horizontal offset	horizontal measurement	Horizontal offset (from the left side of the containing text box) of the first character of this text.
	horizontal scale	percent	Horizontal scale of the first character of this text.
	kern	fixed	Kerning of the first character of this text.
•	length	integer	number of characters in this text object
•	object reference	reference	An object reference for this object.
•	offset	integer	Offset (character index) of the first character of this text object within the containing story.
	PinYin	plain text (string)	The PinYin for the text. (Simplified Chinese only)
	properties	record	Property that allows setting of a list of properties.
	rubi	plain text (string)	The rubi for the text. (Japanese and Korean only)
	sending	horizontal measurement	The sending of this text. (East Asian only)
	shade	percent	Shade of the first character of this text.
	size	fixed	The size of the first character of this text in points.
	style	text style info	The text styles applied to this text.
	track	fixed	Tracking of the first character of this text.

word Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	trap text	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Trapping specification for the first character of this text.
•	uniform styles	text style info	Text styles that are uniformly applied to this text.
	vertical scale	percent	Vertical scale of the first character of this text.
•	width	horizontal measurement	Width of the first character of this text.
	ZhuYin	plain text (string)	The ZhuYin for the text. (Traditional Chinese only)

menu Events and Examples

Verb	Frontier Example	AppleScript Example
select	select(menu[“File”].menuitem[“Append”])	select menu item “Append” of menu “File”
get	get(menu[1].name)	get name of menu 1

menu Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
menu item	•	•			

menu Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	ID	small integer	The id of this menu.
•	index	integer	The index of this menu.
•	name	international text	The name of this menu.

character spec Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(characterspec[“Enhanced”], after(characterspec[3]))	duplicate character spec “Enhanced” to after character spec 3
count	count(characterspec[All])	count every character spec
create	create(characterspec,0,0,beginningof(it))	make character spec at beginning
data size	datasize(characterspec[“Enhanced”]. name,inttype)	data size of name of character spec “Enhanced” as integer
delete	delete(characterspec[“BodyCopy”])	delete character spec “BodyCopy”
get	get(characterspec[“Normal”].color.name)	get name of color of character spec “Normal”
get as	getas(characterspec[“Enhanced”].keycharacter, stringtype)	get key character of character spec “Enhanced” as string
move	move(characterspec[“Enhanced”], before(characterspec[1]))	move character spec “Enhanced” to before character spec 1
set	set(characterspec[1].name,”Header”)	set name of character spec 1 to “Header”

character spec Elements and Reference Forms

None

character spec Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	base shift	base units	Baseline shift of the first character of this text.
	base style	character spec (reference)	The character spec that this character spec is based on.
•	best type	type class	The best descriptor type.
•	class	type class	The class descriptor type.
	color	color spec	The text color for this character spec.
•	default type	type class	The default descriptor type.
	font	plain text (string)	The font to be used with this character spec.
	horizontal scale	percent	The horizontal scale to be used with this character spec.
•	index	integer	The index of this character spec
	key character	plain text (character)	The hot key used to invoke this style.
	key modifier	a list of command/ shift/option/control	Modifier keys, to use in conjunction with the key character (may use more than one).
	name	plain text (string)	The name of this character spec.
•	object reference	reference	The object reference for this object.
	properties	record	Property that allows setting of a list of properties.
	shade	percent	The shade of the color for this character spec.
	size	fixed	The size of the text to be used with this character spec.
	style	text style info	The text style applied with this character spec.
	track	fixed	The track amount to be used with this character spec.
	vertical scale	percent	The vertical scale to be used with this character spec.

color spec Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(colorspec[“Red”],after (colorspec[“Blue”]))	duplicate color spec “Red” to after color spec “Blue”
create	create(colorspec,0,0,beginningof(it))	make color spec at beginning
data size	datasize(colorspec[“Mountain Purple”]. name, inttype)	data size of name of color spec “Mountain Purple” as integer
delete	delete(colorspec[“New Color”])	delete color spec “New Color”
get	get(colorspec[“NewCMYK”].separation)	get separation of color spec “NewCMYK”
get as	getas(colorspec[2].name,stringtype)	get name of color spec 2 as string
move	move(colorspec[“Red”], after(colorspec[“Blue”]))	move color spec “Red” to after color spec “Blue”
set	set(colorspec[“Elizabeth”].name, ”Mountain Purple”)	set name of color spec “Elizabeth” to “Mountain Purple”

color spec Elements and Reference Forms

None

color spec Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	CMYK color value	CMYK color	Representation of color in CMYK space.
	HSB color value	HSB color	Representation of color in HSB space.
	RGB color value	RGB color	Representation of color in RGB space.
	angle	small integer	Screen angle.
•	best type	type class	The best descriptor type.
•	class	type class	The class.
	color flag	RGB color info/CMYK color info/LAB color info	The basic color information type associated with this color.
	color type	HSB type/RGB type/ CMYK type	Type of color.
•	default type	type class	The default descriptor type.
•	index	integer	Index of the object.
•	locked	Boolean	If TRUE, then the color cannot be modified.
•	long name	plain text (string)	The long-form name, if applicable.
	name	plain text (string)	Name of the color.
•	object reference	reference	An object reference for this object.
	properties	record	Property that allows setting of a list of properties.
•	registration color	Boolean	If TRUE, then this color is the registration color.
	separation	Boolean	If TRUE, then separate into process color component.
•	short name	plain text (string)	The short-form name, if applicable.

color system Events and Examples

Verb	Frontier Example	AppleScript Example
data size	datasize(colorsystem[“Panton Coated”].name, inttype)	data size of name of color system “Panton Coated” as integer
get	get(colorsystem[1].colorspec[1].name)	get name of color spec 1 of color system 1
get as	getas(colorsystem[3].name,stringtype)	get name of color system 3 as string

color system Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
color spec	•	•	•	•	•

color system Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	best type	type class	The best descriptor type.
•	class	type class	The class descriptor type.
•	copyright	plain text (string)	The copyright notice, if any.
•	default type	type class	The default descriptor type.
•	name	plain text (string)	The name of this color system.
•	object reference	reference	An object reference for this color system.
	properties	record	Property that allows setting of a list of properties.

contour Events and Examples

Verb	Frontier Example	AppleScript Example
count	count(contour,shapepath[1])	count every contour of shape path 1
data size	datasize(contour[1].bounds,inttype)	data size of bounds of contour 1 as integer
get	get(contour[1].inverted)	get inverted of contour 1

contour Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
vertex	•				•

contour Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	bounds	measurements rectangle	The bounds of this contour.
•	inverted	Boolean	If TRUE, then the contour moves counter clockwise, usually indicating a whole in a shape.

default document Events and Examples

Verb	Frontier Example	AppleScript Example
count	count(colorspec,defaultdocument[1])	count of each color spec in default document 1
data size	datasize(defaultdocument[1].viewscale,inttype)	data size of view scale of default document as integer
get	get(defaultdocument[1].pagewidth)	get page width of default document 1
get as	getas(defaultdocument[1].lockguides,stringtype)	get lock guides of default document 1 as string
set	set(defaultdocument[1].autoconstrain,true)	set auto constrain of default document 1 to true

default document Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character spec	•	•	•	•	•
color spec	•	•	•	•	•
fontset spec (East Asian only)	•	•	•	•	•
h and j spec	•	•	•	•	•
style spec	•	•	•	•	•

default document Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	auto constrain	Boolean	If TRUE, then automatically constrain limits of items within boxes.
	auto kern	Boolean	TRUE if auto kerning should apply.
	auto leading	percent	Specifies the value to use for auto leading.
	auto page insertion location	no auto page insertion/ end of story/end of section/end of document	Automatic page insertion location.
	auto picture import	auto import off/ auto import on/auto import verify	Automatically updates modified pictures since last opened. Selection dependent.
	automatic text box	Boolean	If TRUE, create an automatic text box for each new page.
	automatic trap amount	trap units	Specifies auto trap amount.
	auxiliary dictionary path	alias	The path of the auxiliary dictionary file of this document.
	baseline grid increment	grid increments unit	Specifies the baseline grid interval.
	baseline grid showing	Boolean	If TRUE, baseline grid is showing.
	baseline grid start	vertical measurement	Specifies baseline grid start.
•	best type	type class	The best descriptor type.
	bottom margin	vertical measurement	The location of the bottom margin of a page in this document.
	ciceros per centimeter	fixed	Specifies number of ciceros per centimeter.
•	class	type class	The class.
	column count	integer	Number of columns in this document.
•	default spread count	small integer	Default Spread Count.
•	default type	type class	The default descriptor type.
	enhanced hyphenation method	Boolean	If TRUE, enhanced hyphenation mode is active.
	facing pages	Boolean	If TRUE, creates double-sided pages.
	flex space width	percent	Specifies value for custom width space.
	fractional character widths	Boolean	If TRUE, print characters using fractional widths (default). If FALSE, print character widths using integral widths.
	frame inside	Boolean	If TRUE, then places frames inside text or picture boxes.
	greek below	font units	Specifies text size below which to display text as gray lines.
	greek pictures	Boolean	If TRUE, then displays pictures as grayed images.
	guides in front	Boolean	If TRUE, then places guides in front of all boxes.
	guides showing	Boolean	If TRUE, then guides are showing.

default document Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	gutter width	horizontal measurement	The width of the default text box's gutter in this document.
	horizontal measure	inches/inches decimal/ picas/points/ millimeters/centimeters/ ciceros/agates/ Qs (East Asian only)	Horizontal measurement units.
	ignore white	Boolean	If TRUE, then specifies that an object color in front of multiple backgrounds that include white not take white into account when trapping.
	indeterminate trap amount	trap units	Specifies value for trapping to indeterminate background color.
•	index	integer	Index of object.
	inside margin	horizontal measurement	The location of the inside margin of a page in this document (with facing pages TRUE).
	invisibles showing	Boolean	If TRUE, then invisible characters are showing.
	item spread coords	Boolean	If TRUE, then displays items in spread coordinates.
	keep master page items	Boolean	Specifies whether modified master items are kept or removed when they are modified on the master.
	kern above	font units	Size of text above which auto kerning should apply.
	knockout limit	percent	Point at which an object color knocks out of a background color.
	left margin	horizontal measurement	The location of the left margin of a page in this document.
	ligatures on	Boolean	If TRUE, combine certain characters into a single character (ligatures).
	lock guides	Boolean	If TRUE, then lock guides.
	low quality blends	Boolean	If TRUE then display banded blends (faster).
	maintain leading	Boolean	If TRUE then specifies that the baseline of a line that falls immediately below an obstruction is placed according to its applied leading value.
	maximum ligature track	fixed	The maximum amount that ligatures can be tracked or kerned apart before they break into separate characters.
	maximum view scale	percent	Specifies the largest document view using the zoom tool.
	minimum view scale	percent	Specifies the smallest document view using the zoom tool.
•	object reference	reference	An object reference for this object.
	outside margin	horizontal measurement	The location of the outside margin of a page in this document (with facing pages TRUE).
	overprint limit	percent	Specifies the value for shade of color below which it will not overprint.
	page height	vertical measurement	The height of a page in this document.
	page width	horizontal measurement	The width of a page in this document.
	points per inch	fixed	Specifies number of points per inch.
	process trap	Boolean	If TRUE, process trapping is on.
	properties	record	Property that allows setting of a list of properties.

default document Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	Q measurement	Boolean	If TRUE, then use Q for measurements. (East Asian only)
	right margin	horizontal measurement	The location of the right margin of a page in this document.
	Roman Extra	percent	The percent of the space between Roman and Japanese characters. (East Asian only)
	rulers showing	Boolean	If TRUE, then rulers are showing.
	small caps horizontal scale	percent	Value of horizontal scale for small cap characters.
	small caps vertical scale	percent	Value of vertical scale for small cap characters.
	snap distance	small integer	Specifies distance within which items snap to guides.
	spread height	vertical measurement	The height of a spread (including pasteboard) in this document.
	spread width	horizontal measurement	The width of a spread (including pasteboard) in this document.
	subscript horizontal scale	percent	Value of horizontal scale for subscript characters.
	subscript offset	percent	Value of offset for subscript characters.
	subscript vertical scale	percent	Value of vertical scale for subscript characters.
	superscript horizontal scale	percent	Value of horizontal scale for superscript characters.
	superscript offset	fixed	Value of offset for superscript characters.
	superscript vertical scale	percent	Value of vertical scale for superscript characters.
	superior horizontal scale	percent	The horizontal scale for superior characters.
	superior vertical scale	percent	The vertical scale for superior characters
	top margin	vertical measurement	The location of the top margin of a page in this document.
	trapping method	absolute trap/ proportional trap/ knock all	The default trapping method.
	typesetting leading mode	Boolean	If TRUE, specifies value of leading upward from the baseline of one line of text to the baseline of the line above it. If FALSE, specifies Word Processing mode which measures leading downward from the top of the ascent on the line below it.
	vertical measure	inches/inches decimal/ picas/points/ millimeters/centimeters/ ciceros/agates/ Qs (East Asian only)	Vertical measurement units.
	view scale	fit page in window/ fit spread in window/ thumbnails, or percent.	The current view scale of this document.
	view scale increment	percent	Specifies the percent of change in view for each mouse click using the zoom tool.
	Story direction	Boolean	If TRUE, then the story will be vertical. (East Asian only)

delimit item Events and Examples

Verb	Frontier Example	AppleScript Example
set	set(delimitTable[1].delimitItem[":"],0)	set delimit item ":" of delimit table 1 to can start or end word
get	get(delimitTable[1].delimitItem[":"])	get delimit item ":" of delimit table 1

delimit item Elements and Reference Forms

None

delimit item Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	index	integer	The index of the delimit item.
	delimit	not word member/ can start or end word/ can be contained in word/can start or end or be contained in word	The delimit method for this character.

delimit table Events and Examples

Verb	Frontier Example	AppleScript Example
set	set(delimitTable[1].delimitItem[":"],0)	set delimit item ":" of delimit table 1 to can start or end word
get	get(delimitTable[1].delimitItem[":"])	get delimit item ":" of delimit table 1

delimit table Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
delimit item	•				

delimit table Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	best type	type class	The best descriptor type.
•	class	type class	The class descriptor type.
•	default type	type class	The default descriptor type.
•	object reference	reference	An object reference for this object.
	properties	record	Property that allows setting of a list of properties.

fontset spec Events and Examples (East Asian only)

Verb	Frontier Example	AppleScript Example
count	count(fontsetspec[all])	count every fontset spec
data size	datasize(fontsetspec[1].alphabet,stringtype)	data size of alphabet of fontset spec 1 as string
get	get(fontsetspec[2].pictogram)	get pictogram of fontset spec 2
get as	getas(fontsetspec[all].phoneme,stringtype)	get phoneme of every fontset spec as string
set	set(fontsetspec[2].name,"Head")	set name of fontset spec 2 to "Head"

fontset spec Elements and Reference Forms (East Asian only)

fontset spec Properties, Data Types, and Descriptions (East Asian only)

R/O	Property Name	Type	Description
	alphabet	plain text (string)	The name of the font for the alphabetic text.
•	best type	type class	The best descriptor type.
•	class	type class	The class descriptor type.
•	default type	type class	The default descriptor type.
•	index	integer	The index of this fontset spec.
	name	plain text (string)	The name of the fontset spec.
	numerical	plain text (string)	The name of the font for the numerical text.
•	object reference	reference	The object reference for this fontset spec.
	phoneme	plain text (string)	The name of the font for the phoneme text.
	pictogram	plain text (string)	The name of the font for the pictogram text.
	properties	record	Property that allows setting of a list of properties.
	symbol	plain text (string)	The name of the font for the symbolic text.

generic box Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(genericbox[10],before(genericbox[1]))	duplicate generic box 10 to before generic box 1
count	count(word,genericbox[1])	count of each word in generic box 1
data size	datasize(genericbox[1].rotation,inttype)	data size of rotation of generic box 1 as integer
delete	delete(genericbox[2])	delete generic box 2
get	get(genericbox[2].bounds)	get bounds of generic box 2
get as	getas(genericbox[all].framecolor,stringtype)	get frame color of every generic box as string
move	move(genericbox["Linda"],after(genericbox[last]))	move generic box "Linda" to after last generic box
set	set(genericbox["Sid"].color,"Blue")	set color of generic box "Sid" to "Blue"
show	show(genericbox["Kelly"])	show generic box "Kelly"

generic box Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character	•		•	•	•
image	•	•		•	
line	•		•	•	•
paragraph	•		•	•	•
runaround path	•				
shape path	•				
story	•	•	•	•	•
text	•		•	•	•
text style range	•		•	•	•
word	•		•	•	•

generic box Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	anchored	Boolean	If TRUE, then the box is anchored in text.
	background trap	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Specifies amount to trap background.
•	best type	type class	The best descriptor type.
	blend	blend record	The blend properties of this box.
	bounds	measurements rectangle	The bounds of the box.
	box shape	rectangular/rounded corner/bevel corner/ concave corner/ovular/ polygonal/line shape/ orthogonal line/ spline line	The shape of the box.
	box type	picture box type/text box type/graphic box type/line box type/ xtension box type/ group box type	Type of this box.
•	class	type class	The class.
	color	color spec	Color of the box.
	content	picture content/text content/none content	The content type of this box.
	corner radius	horizontal measurement	The radius of the corners of the box.
•	default type	type class	The default descriptor type.

generic box Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	end caps	plain line/left arrow/ right arrow/left feathered arrow/right feathered arrow/ double arrow	The arrowheads and tail feathers for the line.
	flipped horizontal	Boolean	Whether the contents are flipped left to right.
	flipped vertical	Boolean	Whether the contents are flipped top to bottom.
	frame	frame record	The frame properties of the box.
	gap color	color spec	The color of the line gaps.
	gap shade	percent	The shade of the line gaps.
•	index	integer	Index of this box on its containing spread.
	locked	Boolean	Whether the box can be moved or resized.
	name	plain text (string)	Name of the box.
•	object reference	reference	An object reference for this object.
	polygon points	polygon points list	A list of the vertices for the shape path of the box.
	properties	record	Property that allows setting of a list of properties.
	rotation	angle measurement	Rotation of this box.
	runaround	none runaround/ item runaround/ auto runaround/ manual runaround/ embedded runaround/ alpha runaround/ non white runaround/ clipping runaround/ pic bounds runaround/ custom runaround	Specifies control of the way text flows with respect to this box.
	selected	Boolean	If TRUE, then this box is selected.
	shade	percent	Shade of the box.
	skew	angle measurement	The angle at which the box is skewed.
	style	solid line/sparsely dashed line/densely dashed line/dashed line/dotted line/ double line/thin thick line/thick thin line/thin thick thin line/thick thin thick line/thin thin thin line	The style of the line.
	suppress printing	Boolean	If TRUE, then this box is suppressed on printout.
	width	thick units	The line thickness.
•	uniqueID	integer	A unique id that is good for the life of the document.

graphic box Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(graphicbox[10],before(graphicbox[1]))	duplicate graphic box 10 to before graphic box 1
count	count(graphicbox[1].document[1])	count every graphic box 1 of document 1
data size	datasize(graphicbox[1].rotation,inttype)	data size of rotation of graphic box 1 as integer
delete	delete(graphicbox[2])	delete graphic box 2
get	get(graphicbox[2].bounds)	get bounds of graphic box 2
get as	getas(graphicbox[all].framecolor,stringtype)	get frame color of every graphic box as string
move	move(graphicbox[“Linda”],after(graphicbox[last]))	move graphic box “Linda” to after last graphic box
set	set(graphicbox[“Sid”].color,”Blue”)	set color of graphic box “Sid” to “Blue”
show	show(graphicbox[“Kelly”])	show graphic box “Kelly”

graphic box Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
runaround path	•				
shape path	•				

graphic box Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	anchored	Boolean	If TRUE, then the box is anchored in text.
	background trap	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Specifies amount to trap background.
•	best type	type class	The best descriptor type.
	blend	blend record	The blend properties of this box.
	bounds	measurements rectangle	The bounds of the box.
	box shape	rectangular/rounded corner/bevel corner/ concave corner/ovular/ polygonal/line shape/ orthogonal line/ spline line	The shape of the box.
	box type	picture box type/text box type/graphic box type/line box type/ xtension box type/ group box type	Type of this box.
•	class	type class	The class.
	color	color spec	Color of the box.
	content	picture content/text content/none content	The content type of this box.
	corner radius	horizontal measurement	The radius of the corners of the box.

graphic box Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
•	default type	type class	The default descriptor type.
	end caps	plain line/left arrow/ right arrow/left feathered arrow/right feathered arrow/ double arrow	The arrowheads and tail feathers for the line.
	flipped horizontal	Boolean	Whether the contents are flipped left to right.
	flipped vertical	Boolean	Whether the contents are flipped top to bottom.
	frame	frame record	The frame properties of the box.
	gap color	color spec	The color of the line gaps.
	gap shade	percent	The shade of the line gaps.
•	index	integer	Index of this box on its containing spread.
	locked	Boolean	Whether the box can be moved or resized.
	name	plain text (string)	Name of the box.
•	object reference	reference	An object reference for this object.
	polygon points	polygon points list	A list of the vertices for the shape path of the box.
	properties	record	Property that allows setting of a list of properties.
	rotation	angle measurement	Rotation of this box.
	runaround	none runaround/ item runaround/ auto runaround/ manual runaround/ embedded runaround/ alpha runaround/ non white runaround/ clipping runaround/ pic bounds runaround/ custom runaround	Specifies control of the way text flows with respect to this box.
	selected	Boolean	If TRUE, then this box is selected.
	shade	percent	Shade of the box.
	skew	angle measurement	The angle at which the box is skewed.
	style	solid line/sparsely dashed line/densely dashed line/dashed line/dotted line/double line/thin thick line/ thick thin line/thin thick thin line/thick thin thick line/thin thin thin line	The style of the line.
	suppress printing	Boolean	If TRUE, then this box is suppressed on printout.
	width	thick units	The line thickness.
•	uniqueID	integer	A unique id that is good for the life of the document.

H&J spec Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(handjspec[1],after(handjspec[2])	duplicate h and j spec 1 to after h and j spec 2
count	count(handjspec.document[1])	count every h and j spec of document 1
create	create(handjspec,0,0,beginningof(it))	make h and j spec at beginning
data size	datasize(handjspec[“Standard”].name,inttype)	data size of name of h and j spec “Standard” as integer
delete	delete(handjspec[1])	delete h and j spec 1
get	get(handjspec[“Standard”].breakCapitalizedWords)	get break capitalized words of h and j spec “Standard”
get as	getas(handjspec[“Standard”].flushzone,inttype)	get flush zone of h and j spec “Standard” as integer
set	set(handjspec[“Standard”].minimumafter,3)	set minimum after of h and j spec “Standard” to 3

H&J spec Elements and Reference Forms

None

H and J spec Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	auto hyphenation	Boolean	If TRUE, then auto hyphenation is active.
•	best type	type class	The best descriptor type.
	break capitalized words	Boolean	If TRUE, then proper nouns and the first words of sentences are broken.
	character justification	justification record	Justification settings for non-space characters.
•	class	type class	The class.
•	default type	type class	The default descriptor type.
	flush zone	horizontal measurement	Controls whether the last line of text in a justified paragraph will automatically extend to the right indent.
	hyphenation zone	horizontal measurement	Specifies area within which hyphenation can occur (automatic or manual).
	hyphens in a row	small integer	Specifies the maximum number of consecutive lines that can end in manually or automatically hyphenated words.
•	index	integer	Index of the object.
	Japanese punctuation justification	justification record justification record characters. (East Asian only)	Justification settings for Japanese punctuation.
•	kinsokushori	plain text (string)	Kinsoku shori setting of h and j. (East Asian only)
	minimum after	small integer	Specifies the minimum number of characters that must follow an automatic hyphen.
	minimum before	small integer	Specifies the minimum number of characters that must precede an automatic hyphen.
	name	plain text (string)	Name of the h and j spec.

H and J spec Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
•	object reference	reference	An object reference for this object.
	phonetic justification	justification record	Justification settings for phonetic characters. (East Asian only)
	pictogram justification	justification record	Justification settings for pictogram characters. (East Asian only)
	properties	record	Property that allows setting of a list of properties.
	single word justify	Boolean	If TRUE, then words alone on line are justified.
	smallest word	small integer	Defines the minimum number of characters a word must contain to be hyphenated.
	space justification	justification record	Justification settings for space characters.

horizontal guide Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(horizontalguide[1], after(horizontalguide[10]))	duplicate horizontal guide 1 to after horizontal guide 10
count	count(horizontalguide.document[1])	count every horizontal guide of document 1
create	create(horizontalguide,0,0,beginningof(it))	make horizontal guide at beginning
data size	datasize(horizontalguide[2]position,inttype)	data size of position of horizontal guide 2 as integer
delete	delete(horizontalguide[1])	delete horizontal guide 1
get	get(horizontalguide[4].undeletable)	get undeletable of horizontal guide 4
get as	getas(horizontalguide[1].position,inttype)	get position of horizontal guide 1 as integer
move	move(horizontalguide[1], after(horizontalguide[3]))	move horizontal guide 1 to after horizontal guide 3
set	set(horizontalguide[1].position,"6 cm")	set position of horizontal guide 1 to "6 cm"

horizontal guide Elements and Reference Forms

None

horizontal guide Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	best type	type class	The best descriptor type.
•	class	type class	The class.
•	default type	type class	The default descriptor type.
•	from master	Boolean	If TRUE, this guide is from the master.
•	index	integer	Index of the object.
•	object reference	reference	An object reference for this object.
	position	horizontal measurement	Position of the guide.
	properties	record	Property that allows setting of a list of properties.
	scale	percent	Scale of the guide.
	undeletable	Boolean	If TRUE, guide can't be deleted.
	unmoveable	Boolean	If TRUE, guide can't be moved.

image Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(image[“Bag.Tiff”],replace(image[“Gloves”]))	duplicate image “Bag.Tiff” to replace image “Gloves”
data size	datasize(image[“Bag.Tiff”].imagescale,inttype)	data size of image scale of image “Bag.Tiff” as integer
delete	delete(image[1])	delete image 1
get	get(image[2].imageskew)	get image skew of image 2
get as	getas(image[“Bag.Tiff”].name,stringtype)	get name of image “Bag.Tiff” as string
set	set(image[1].suppressimageprint,true)	set suppress image print of image 1 to true
show	show(image[“Bag.Tiff”])	show image “Bag.Tiff”

image Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
clipping path	•				

image Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	angle	angle measurement	Value for rotating a picture around its center.
•	best type	type class	The best descriptor type.
	bounds	centered/exact fit/ proportional fit	Bounds of this image, in relation to the bounds of the picture box.
•	bounds	rectangle	Bounds of this image.
•	class	type class	The class.
	color	color spec	Color of this image.
	contents	picture	Pict representation of this image.
	contrast	normal contrast/high contrast/posterized contrast/other contrast	Specifies type of contrast.
•	default type	type class	The default descriptor type.
•	file path	alias	File path to the disk image for this picture (if any).
•	file type	type class	Type of the file from which this image was loaded (if any).
	greek pictures	Boolean	Display pictures as a gray box when inactive.
	image trap	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Trap override that is applied to image.
•	image type	unknown image/ line art image/ gray scale image/ color image image/ color 16 bit image/ color 32 bit image	Displays the bit depth the picture was saved as.
	invert runaround	Boolean	If TRUE, then flow text within the runaround.
•	missing	Boolean	Is the image missing from the saved location?

image Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	model	HSB model/RGB model/CMY model/ CMYK model	Model for contrast modification.
•	modified	Boolean	Has the picture been modified since last imported?
•	modification date	date	The modification date of the file when last imported.
•	name	plain text (string)	The name of this image.
	negative	Boolean	If TRUE, then display negative of this image.
•	object reference	reference	An object reference for this object.
	offset	measurements point	Specifies the distance between the origin of the picture box and the upper left corner of the picture.
	properties	record	Property that allows setting of a list of properties.
	scale	percent point	Specifies the scale of this image.
	screen	small integer	Modifies screen function.
	screen angle	angle measurement	Halftone screen angle of this image.
	screen frequency	fixed	Halftone screen frequency of this image.
	screen function	dot spot/line spot/ ellipse spot/square spot/ ordered dither/ tri dot spot	Halftone screen function applied to this image.
	shade	percent	Shade of this image.
	show half tone	Boolean	Show halftones on screen.
	skew	angle measurement	Value to slant the picture.
•	subscribed	Boolean	If TRUE, then this picture is from a subscription.
	suppress printing	Boolean	If TRUE, then this image is suppressed on printout.

line box Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(linebox[2],before(linebox[1]))	duplicate linebox 2 to before linebox 1
create	create(linebox,0,0,beginningof(it))	make line box at beginning
data size	datasize(linebox[1].linewidth,inttype)	data size of line width of linebox 1 as integer
delete	delete(linebox[1])	delete line box 1
get	get(linebox[4].color)	get color of line box 4
get as	getas(linebox[1].endcaps,intype)	get end caps of line box 1 as integer
move	move(linebox[“Deborah”],before(linebox[“Jay”]))	move line box “Deborah” to before line box “Jay”
set	set(linebox[1].linewidth,”6 pt”)	set line width of line box 1 to “6 pt”
show	show(linebox[“Fred”])	show line box “Fred”

line box Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
runaround path	•				
shape path	•				

line box Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	anchored	Boolean	If TRUE, then the box is anchored in text.
	background trap	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Specifies amount to trap background.
•	best type	type class	The best descriptor type
	box shape	rectangular/rounded corner/bevel corner/ concave corner/ovular/ polygonal/line shape/ orthogonal line/ spline line	The shape of this box.
	box type	picture box type/ text box type/graphic box type/ line box type/ xtension box type/ group box type	Type of this box.
•	class	type class	The class.
	color	color spec	Color of the box.
	content	picture content/ text content/ none content	The content type of this box.
•	default type	type class	The default descriptor type.
	end caps	plain line/left arrow/ right arrow/left feathered arrow/right feathered arrow/ double arrow	Specifies the arrowheads and tail feathers for line.
	gap color	color spec	The color of the line gaps.
	gap shade	percent	The shade of the line gaps.
•	index	integer	Index of this box on its containing spread.
	left point	measurements point	Specifies the left endpoint of the line.
	locked	Boolean	Whether the box can be moved or resized.
	name	plain text (string)	Name of the box.
•	object reference	reference	An object reference for this object.
	polygon points	polygon points list	A list of the vertices of the shape path.
	properties	record	Property that allows setting of a list of properties.
	right point	measurements point	Specifies the right endpoint of the line.
	rotation	angle measurement	Rotation of this box.

line box Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	runaround	none runaround/ item runaround/ auto runaround/ manual runaround/ embedded runaround/ alpha runaround/ non white runaround/ clipping runaround/ pic bounds runaround/ custom runaround	Specifies control of the way text flows with respect to this box.
	selected	Boolean	If TRUE, then this box is selected.
	shade	percent	Shade of the box.
	style	solid line/sparsely dashed line/densely dashed line/dashed line/dotted line/double line/thin thick line/thick thin line/thin thick thin line/thick thin thick line/thin thin thin line	Specifies the line style.
	suppress printing	Boolean	If TRUE, then this box is suppressed on printout.
	width	thick units	Specifies the line thickness.
•	uniqueID	integer	A unique id that is good for the life of the box.

master document Events and Examples

Verb	Frontier Example	AppleScript Example
close	close(masterDocument[“Kristy”])	close master document “Kristy”
count	count(story.masterDocument[1])	count of each story in master document 1
create	create(masterDocument,0,0,beginningof(it))	make master document at beginning
data size	data size(masterDocument[1].name,inttype)	data size of name of master document 1 as integer
get	get(masterDocument[1].name)	get name of master document 1
get as	getas(masterDocument[2].filepath,stringtype)	get file path of master document 2 as string
open	open(masterDocument[“Test”],yes,ask,ask)	open master document “Test” use doc prefs yes remap fonts ask do auto picture import ask
print	print(masterDocument[all])	print every master document
save	save(masterDocument[1],alias (“Constructed Document”)	save master document 1 in “Constructed Document”
set	set(masterDocument[1].keepmasterpageitems,true)	set keep master page items of master document 1 to true
show	show(masterDocument[middle])	show middle master document

master document Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character spec	•	•	•	•	•
color spec	•	•	•	•	•
fontset spec	•	•	•	•	•
generic box	•	•	•	•	•
graphic box	•	•	•	•	•
h and j spec	•	•	•	•	•
image	•	•	•	•	•
line box	•	•	•	•	•
page	•	•	•	•	•
picture box	•	•	•	•	•
spread	•	•	•	•	•
story	•	•	•	•	•
style spec	•	•	•	•	•
text box	•	•	•	•	•

master document Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	auto constrain	Boolean	If TRUE, then automatically constrain limits of items within boxes.
	auto kern	Boolean	TRUE if auto kerning should apply.
	auto leading	percent	Specifies the value to use for auto leading.
	auto page insertion location	no auto page insertion/ end of story/end of section/end of document	Automatic page insertion location.
	auto picture import	auto import off/auto import on/auto import verify	Automatically updates modified pictures since last opened.
	automatic text box	Boolean	If TRUE, create an automatic text box for each new page.
	automatic trap amount	trap units	Specifies auto trap amount.
	auxiliary dictionary path	alias	The path of the auxiliary dictionary file of this document.
	baseline grid increment	grid increment units	Specifies the baseline grid interval.
	baseline grid showing	Boolean	If TRUE, baseline grid is showing.
	baseline grid start	vertical measurement	Specifies baseline grid start.
•	best type	type class	The best descriptor type.
	bottom margin	vertical measurement	The location of the bottom margin of a page in this document.
	ciceros per centimeter	fixed	Specifies number of ciceros per centimeter.
•	class	type class	The class.
	column count	integer	Number of columns in this document.
	current box	reference	Currently selected box.
	current page	page	Current page displayed to user.

master document Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	current spread	spread	Current spread displayed to user.
•	default spread count	small integer	Default Spread Count.
•	default type	type class	The default descriptor type.
	enhanced hyphenation method	Boolean	If TRUE, enhanced hyphenation mode is active.
	facing pages	Boolean	If TRUE, creates double-sided pages.
•	file path	alias	File path of this document.
	flex space width	percent	Specifies value for custom width space.
•	flow version	fixed	Document flow version.
•	font list	a list of font record	List of fonts used in this document.
	fractional character widths	Boolean	If TRUE, print characters using fractional widths (default). If FALSE, print character widths using integral widths.
	frame inside	Boolean	If TRUE, then places frames inside text or picture boxes.
	greek below	font units	Specifies text size below which to display text as gray lines.
	greek pictures	Boolean	If TRUE, then displays pictures as grayed images.
	guides in front	Boolean	If TRUE, then places guides in front of all boxes.
	guides showing	Boolean	If TRUE, then guides are showing.
	gutter width	horizontal measurement	The width of the default text box's gutter in this document.
	horizontal measure	inches/inches decimal/ picas/points/ millimeters/centimeters/ ciceros/agates/ Qs (East Asian only)	Horizontal measurement units.
	ignore white	Boolean	If TRUE, then specifies that an object color in front of multiple backgrounds that include white not take white into account when trapping.
	indeterminate trap amount	trap units	Specifies value for trapping to indeterminate background color.
•	index	integer	Index of object.
	inside margin	horizontal measurement	The location of the inside margin of a page in this document (with facing pages TRUE).
	invisibles showing	Boolean	If TRUE, then invisible characters are showing.
	item spread coords	Boolean	If TRUE, then displays items in spread coordinates.
	keep master page items	Boolean	Specifies whether modified master items are kept or removed when they are modified on the master.
	kern above	font units	Size of text above which auto kerning should apply.

master document Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	knockout limit	percent	The point at which an object color knocks out a background color.
	left margin	horizontal measurement	The location of the left margin of a page in this document.
	ligatures on	Boolean	If TRUE, combine certain characters into a single character (ligatures).
	lock guides	Boolean	If TRUE, then lock guides.
	low quality blends	Boolean	If TRUE then display banded blends (faster).
	maintain leading	Boolean	If TRUE then specifies that the baseline of a line that falls immediately below an obstruction is placed according to its applied leading value.
	maximum ligature track	fixed	The maximum amount that ligatures can be tracked or kerned apart before they break into separate characters.
	maximum view scale	percent	Specifies the largest document view using the zoom tool.
	minimum view scale	percent	Specifies the smallest document view using the zoom tool.
•	modified	Boolean	Has the document been modified since the last save?
•	name	plain text (string)	Name of this document.
•	object reference	reference	An object reference for this object.
	outside margin	horizontal measurement	The location of the outside margin of a page in this document (with facing pages TRUE).
	overprint limit	percent	Specifies the value for shade of color below which it will not overprint.
	page height	vertical measurement	The height of a page in this document.
	page rule origin	measurements point	Location of the page's ruler origin.
	page width	horizontal measurement	The width of a page in this document.
	points per inch	fixed	Specifies number of points per inch.
	print setup	print setup record	Settings used when printing this document.
	process trap	Boolean	If TRUE, process trapping is on.
	properties	record	Property that allows setting of a list of properties.
	Q measurement	Boolean	If TRUE, then use Q for measurements. (East Asian only)
	right margin	horizontal measurement	The location of the right margin of a page in this document.
	Roman Extra	percent	The percent of space between Roman and Japanese characters. (East Asian only)
	rulers showing	Boolean	If TRUE, then rulers are showing.
	small caps horizontal scale	percent	Value of horizontal scale for small cap characters.
	small caps vertical scale	percent	Value of vertical scale for small cap characters.
	snap distance	small integer	Specifies distance within which items snap to guides.
	spread height	vertical measurement	The height of a spread (including pasteboard) in this document.
	spread rule origin	measurements point	Location of the spread's ruler origin.
	spread width	horizontal measurement	The width of a spread (including pasteboard) in this document.

master document Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	subscript horizontal scale	percent	Value of horizontal scale for subscript characters.
	subscript offset	percent	Value of offset for subscript characters.
	subscript vertical scale	percent	Value of vertical scale for subscript characters.
	superscript horizontal scale	percent	Value of horizontal scale for superscript characters.
	superscript offset	percent	Value of offset for superscript characters.
	superscript vertical scale	percent	Value of vertical scale for superscript characters.
	superior horizontal scale	percent	The horizontal scale for superior characters.
	superior vertical scale	percent	The vertical scale for superior characters.
	tool mode	drag mode/contents mode/rotate mode/ view mode/text mode/ rounded rect mode/oval mode/poly mode/pic mode/ rounded rect pic mode/ oval pic mode/poly pic mode/orthogonal line mode/line mode	The tool that is currently selected.
	top margin	vertical measurement	The location of the top margin of a page in this document.
	trapping method	absolute trap/ proportional trap/ knockout all	The default trapping method.
	typesetting leading mode	Boolean	If TRUE, specifies value of leading upward from the baseline of one line of text to the baseline of the line above it. If FALSE, specifies Word Processing mode which measures leading downward from the top of the ascent on the line below it.
•	version	vers 33/vers 40/ vers current/vers other	The flow version of the document.
	vertical measure	inches/inches decimal/ picas/points/ millimeters/centimeters/ ciceros/agates/ Qs (East Asian only)	Vertical measurement units.
	view scale	fit page in window/ fit spread in window/ thumbnails, or percent	The current view scale of this document.
	view scale increment	percent	Specifies the percent of change in view for each mouse click using the zoom tool.
	vStory direction	Boolean	If TRUE, then the story will be vertical. (East Asian only)

page Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(page[3],after(page[5]))	duplicate page 3 to after page 5
count	count(picturebox,page[1])	count of each picture box in page 1
create	create(page,0,0,beginningof(it))	make page at beginning
data size	datasize(page[2].topmargin,inttype)	data size of top margin of page 2 as integer
delete	delete(page[2])	delete page 2
get	get(page[5].gutterwidth)	get gutter width of page 5
get as	getas(page[4].pageprefix,stringtype)	get page prefix of page 4 as string
move	move(page[3],before(page[1]))	move page 3 to before page 1
save	save(page[1],alias("I"),"EPSF",macincolor, ASCIIEPS,0)	save page 1 in "I" as "EPSF" eps format mac in color eps data ASCIIEPS
set	set(page[last].columncount,3)	set column count of last page to 3
show	show(page[first])	show first page

page Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
generic box	•	•	•	•	•
graphic box	•	•	•	•	•
horizontal guide	•		•	•	•
image	•	•		•	
line box	•	•	•	•	•
picture box	•	•	•	•	•
text box	•	•	•	•	•
vertical guide	•		•	•	•

page Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	best type	type class	The best descriptor type.
	bottom margin	vertical measurement	The location of the bottom margin of this page.
•	bounds	measurements rectangle	The boundary rectangle of this page.
•	class	type class	The class.
	column count	small integer	Number of columns in this page.
•	default type	type class	The default descriptor type.
	gutter width	horizontal measurement	The width of the gutter in this page.
	left margin	horizontal measurement	The location of the left margin of this page.
	master spread	single sided blank master/double sided blank master,or spread	Specifies the master spread applied to this page.
•	name	plain text (string)	The name of the page (section number when using section starts).
•	object reference	reference	An object reference for this object.
•	page number	small integer	Page number of the page.
	properties	record	Property that allows setting of a list of properties.
	right margin	horizontal measurement	The location of the right margin of this page.
	top margin	vertical measurement	The location of the top margin of this page.

picture box Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(picturebox[4],before(picturebox[2]))	duplicate picture box 4 to before picture box 2
count	count(image,picturebox[1])	count of each image in picture box 1
create	create(picturebox,0,0,endof(it))	make picture box at end
data size	datasize(picturebox[1].cornerradius,inttype)	data size of corner radius of picture box 1 as integer
delete	delete(picturebox[“Dave”])	delete picture box “Dave”
get	get(picturebox[1].boxtype)	get box type of picture box 1
get as	getas(picturebox[1].runaround,stringtype)	get runaround of picture box 1 as string
move	move(picturebox[1],after(picturebox[“Tim”]))	move picture box 1 to after picture box “Tim”
set	set(picturebox[first].locked,true)	set locked of first picture box to true
show	show(picturebox[last])	show last picture box

picture box Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
image	•	•		•	
runaround path	•				
shape path	•				

picture box Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	anchored	Boolean	If TRUE, then the box is anchored in text.
	background trap	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Specifies amount to trap background.
•	best type	type class	The best descriptor type.
	blend	blend record	The blend properties of this box.
	bounds	measurements rectangle	Bounds of the box.
	box shape	rectangular/rounded corner/bevel corner/ concave corner/ovular/ polygonal/line shape/ orthogonal line/ spline line	Specifies the shape of a box.
	box type	picture box type/text box type/graphic box type/line box type/ xtension box type/ group box type	Type of this box.
•	class	type class	The class.
	color	color spec	Color of the box.
	content	picture content/text content/none content	The content type of this box.
	corner radius	horizontal measurement	Specifies the radius of the circles that form the corners of the picture box.

picture box Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
•	default type	type class	The default descriptor type.
	flipped horizontal	Boolean	Whether the contents are flipped left to right.
	flipped vertical	Boolean	Whether the contents are flipped top to bottom.
	frame	frame record	Frame properties of this box.
•	index	integer	Index of this box on its containing spread.
	locked	Boolean	Whether the box can be moved or resized.
	name	plain text (string)	Name of the box.
•	object reference	reference	An object reference for this object.
	polygon points	polygon points list	A list of the vertices of the polygon if picture box type is a polygon.
	properties	record	Property that allows setting of a list of properties.
	rotation	angle measurement	Rotation of this box.
	runaround	none runaround/ item runaround/ auto runaround/ manual runaround/ embedded runaround/ alpha runaround/ non white runaround/ clipping runaround/ pic bounds runaround/ custom runaround	Specifies control of the way text flows with respect to this box.
	selected	Boolean	If TRUE, then this box is selected.
	shade	percent	Shade of the box.
	skew	angle measurement	Specifies angle that box is skewed.
	suppress printing	Boolean	If TRUE, then this box is suppressed on printout.
	text outset	outset units	Specifies the space between text and the outer edges of a picture box.
•	uniqueID	integer	A unique id that is good for the life of the box.

spread Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(spread[1],after(spread[6]))	duplicate spread 1 to after spread 6
count	count(textbox,spread[1])	count of each text box in spread 1
create	create(spread,0,0,endof(it))	make spread at end
data size	datasize(spread[1].numberofpages,inttype)	data size of number of pages of spread 1 as integer
delete	delete(spread[last])	delete last spread
get	get(spread[1].numberofpages)	get number of pages of spread 1
get as	getas(spread[1].numberofpages,inttype)	get number of pages of spread 1 as integer
move	move(spread[1], after(spread[2]))	move spread 1 to after spread 2
set	set(spread[1],name,”Cover”)	set name of spread 1 to “Cover”
show	show(spread[middle])	show middle spread

spread Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
generic box	•	•	•	•	•
graphic box	•	•	•	•	•
horizontal guide	•		•	•	•
image	•	•	•	•	
line box	•	•	•	•	•
page	•	•	•	•	•
picture box	•	•	•	•	•
text box	•	•	•	•	•
vertical guide	•		•	•	•

spread Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	best type	type class	The best descriptor type.
•	class	type class	The class.
•	default text box	Boolean	Only used when creating master spreads.
•	default type	type class	The default descriptor type.
•	double sided	Boolean	Only used when creating master spreads.
	name	plain text (string)	The name of the spread (if a master spread).
•	number of pages	small integer	Specifies number of pages.
•	object reference	reference	An object reference for this object.
	properties	record	Property that allows setting of a list of properties.

style spec Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(stylespec[“Enhanced”],after(stypespec[3]))	duplicate style spec “Enhanced” to after style spec 3
count	count(stylespec[All])	count every style spec
create	create(stylespec,0,0,beginningof(it))	make style spec at beginning
data size	datasize(stylespec[“Enhanced”].name,inttype)	data size of name of style spec “Enhanced” as integer
delete	delete(stylespec[“BodyCopy”])	delete style spec “BodyCopy”
get	get(stylespec[“Normal”].nextStyle)	get next style of style spec “Normal”
get as	getas(stylespec[“Enhanced”].keycharacter,) stringtype)	get key character of style spec “Enhanced” as string
move	move(stylespec[“Enhanced”],before(stylespec[1])	move style spec “Enhanced” to before style spec 1
set	set(stylespec[1].name,”Header”)	set name of style spec 1 to “Header”

style spec Elements and Reference Forms

None

style spec Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	base style	style spec	Style that this style is based on.
•	best type	type class	The best descriptor type.
	character style	character spec	The character attributes for this style spec.
•	class	type class	The class.
•	default type	type class	The default descriptor type.
•	index	integer	Index of the object.
	key character	plain text (character)	Hot key to evoke application of style.
	key modifiers	list of command/shift/option/control	Modifier keys, to use in conjunction with the key character (may user more than one).
	name	plain text (string)	Name of the style.
	next style	style spec	Style that paragraph created after this paragraph will have applied to it.
•	object reference	reference	An object reference for this object.
	paragraph attributes	paragraph properties	The paragraph properties for this style spec.
	properties	record	Property that allows setting of a list of properties.
	text and paragraph attributes	text and paragraph properties	Specifies the text and paragraph attributes for the style (for compatibility).

path Events and Examples

Verb	Frontier Example	AppleScript Example
count	count(contours,shapepath[1])	count of contours of shape path 1
delete	delete(clippingpath[1])	delete clipping path 1
get	get(runaroundpath[1].bounds)	get bounds of runaround path 1
get as	get(shapepath[1].bounds,listtype)	get bounds of shape path 1 as list

path Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
contours	•				

path Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	bounds	measurements rectangle	The bounds of this path.

text box Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(textbox[10],before (textbox[1]))	duplicate text box 10 to before text box 1
count	count(word,textbox[“Gene”])	count of each word in text box “Gene”
create	create(textbox,0,0,beginningof(page[2]))	make text box at beginning of spread 2
data size	datasize(textbox[“Gene”]textinset,inttype).	data size of text inset of text box “Gene” as integer
delete	delete(textbox[“Ralph”])	delete text box “Ralph”
get	get(textbox[first].firstbaseline	get first base line offset of first text box offset)
get as	getas(textbox[1].nexttextbox,stringtype)	get next text box of text box 1 as string
move	move(textbox[last],before(textbox[first]))	move last text box to before first text box
set	set(textbox[1].skew,25)	set skew of text box 1 to 25
show	show(textbox[“Rob”])	show text box “Rob”

text box Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character	•		•	•	•
line	•		•	•	•
paragraph	•		•	•	•
runaround path	•				
shape path	•				
story	•	•	•	•	•
text	•		•	•	•
text style range	•		•	•	•
word	•		•	•	•

text box Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	align text	ascent align/centered align/baseline align/ descent align	The text alignment on the path.
	align with line	top align/center align/ bottom align	The alignment of the text with relation to the width of the path.
•	anchored	Boolean	If TRUE, then the box is anchored in text.
	background trap	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Specifies amount to trap background.
•	best type	type class	The best descriptor type.
	blend	blend record	The blend properties of this box.
	bounds	measurements rectangle	Bounds of the box.
•	box overflows	Boolean	Whether the box overflow symbol is present in the box

text box Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	box shape	rectangular/rounded corner/bevel corner/ concave corner/ovular/ polygonal/line shape/ orthogonal line/ spline line	Specifies the shape of a box.
	box type	picture box/text box type/graphic box type/line box type/ xtension box type/ group box type	Type of this box.
•	box wraps	Boolean	Whether text in this box wraps to the next box
•	class	type class	The class.
	color	color spec	Color of the box.
	columns	small integer	Number of columns in the text box.
	content	picture content/text content/none content	The content type of this box.
	corner radius	horizontal measurement	The radius of the corners of this box.
•	default type	type class	The default descriptor type.
	end caps	plain line/left arrow/ right arrow/left feathered arrow/right feathered arrow/double arrow	Arrowheads and tail feathers for the line.
	first baseline minimum	cap height/cap plus accent/ascent baseline	Method for placing the first baseline of text.
	first baseline offset	vertical measurement	Specifies the distance between the first baseline and the top of a text box.
	flip	Boolean	If TRUE, then the text is flipped to the other side of the path.
	flipped horizontal	Boolean	If TRUE, the contents of the box are flipped left to right.
	flipped vertical	Boolean	If TRUE, the contents of the box are flipped top to bottom.
	frame	frame record	Frame properties of this box.
	gap color	color spec	The color of the line gaps.
	gap shade	percent	The shade of the line gaps.
	gutter	horizontal measurement	Specifies space between columns.
•	index	integer	Index of this box on its containing spread.
	inter para max	fixed	Specifies the maximum amount of space between paragraphs when vertical justification is selected.
	left point	measurements point	The start point of the line. (only applicable with text paths)
	locked	Boolean	Whether the box can be moved or resized.
	name	plain text (string)	Name of the box.
	next text box	text box (reference)	Specifies next text box in text box chain.
•	object reference	reference	An object reference for this object.

text box Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	orientation	normal/skewed/stair step/warped	The orientation of the text on the path.
	polygon points	polygon points list	A list of the vertices for the shape path.
	previous text box	text box (reference)	Specifies previous text box in text box chain.
	properties	record	Property that allows setting of a list of properties.
	right point	measurements point	The end point of the line. (only applicable with text paths)
	rotation	angle measurement	Rotation of this box.
	runaround	none runaround/ item runaround/ auto runaround/ manual runaround/ embedded runaround/ alpha runaround/ non white runaround/ clipping runaround/ pic bounds runaround/ custom runaround	Specifies control of the way text flows with respect to this box.
	runaround all sides	Boolean	If TRUE, then text will be flowed around all sides of obstructions.
	selected	Boolean	If TRUE, then this box is selected.
	shade	percent	Shade of the box.
	skew	angle measurement	Specifies angle that box is skewed.
	style	solid line/sparsely dashed line/densely dashed line/dashed line/dotted line/ double line/thin thick line/thick thin line/thin thick thin line/thick thin thick line/ thin thin thin line	The style of the frame for the box.
	suppress printing	Boolean	If TRUE, then this box is suppressed on printout.
	text angle	angle measurement	The angle of the text in the text box.
	text inset	measurements rectangle	Specifies the space between text and the inner edges of a text box.
	text skew	angle measurement	The skew of the text in the box.
•	uniqueID	integer	A unique ID that is good for the life of the box.
	vertical justification	top justified/centered/ bottom justified/full	Method for placing the first baseline of text (vertical alignment).
	width	thick units	The line or frame thickness.

text style range Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(textstylerange[1],after(textstylerange[2]))	duplicate text style range 1 to after text style range 2
count	count(paragraph,textstylerange[1])	count of each paragraph in text style range 1
create	create(textstylerange,0,0,endof(it))	make text style range at end
data size	datasize(textstylerange[1].font,inttype)	data size of font of text style range 1 as integer
delete	delete(textstylerange[1])	delete text style range 1
get	get(textstylerange[all].leading)	get leading of text style range all
get as	getas(textstylerange[2].descent,inttype)	get descent of text style range 1 as integer
move	move(textstylerange[1],after(textstylerange[3]))	move text style range 1 to after text style range 3
save	save(textstylerange[1],"TEXT")	save text style range 1 as "TEXT"
set	set(textstylerange[1].justification,center)	set justification of text style range 1 to center
show	show(textstylerange[1])	show text style range 1

text style range Elements and Reference Forms

Element Class	By Numeric Index	By Name	By Range	Satisfying a Test	Before/After Another Element
character	•		•	•	•
line	•		•	•	•
paragraph	•		•	•	•
story	•	•	•	•	•
text	•		•	•	•
word	•		•	•	•

text style range Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	ascent	font units	The maximum ascent of any character in this text.
	base shift	base units	Baseline shift of this text.
•	baseline	vertical measurement	Vertical offset (from the top of the containing text box) of the baseline of the first character of this text.
•	best type	type class	The best descriptor type.
	character style	character spec	The character spec applied to this text.0
•	class	type class	The class.
	color	color spec	The color of this text.
	contents	plain text (string)	Contents of this text.
•	default type	type class	The default descriptor type.
•	descent	font units	The maximum descent of any character in this text.
	font	plain text (string)	The name of the font of the first character in this text.
•	height	font units	Height of this text.
•	horizontal offset	horizontal measurement	Horizontal offset (from the left side of the containing text box) of the first character of this text.
	horizontal scale	percent	Horizontal scale of the first character of this text.
	kern	fixed	Kerning of the first character of this text.
•	length	integer	The number of characters in this text object.
•	object reference	reference	An object reference for this object.

text style range Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
•	offset	integer	Offset (character index) of the first character of this text object within the containing story.
	off styles	plain/bold/italic/ underline/outline/ shadow/superscript/ subscript/superior/ strikethrough/ all caps/small caps/ word underline/ comma emphasis (East Asian only)/ dot emphasis	The styles that are not used for the text.
	on styles	plain/bold/italic/ underline/outline/ shadow/superscript/ subscript/superior/ strikethrough/ all caps/small caps/ word underline/ comma emphasis (East Asian only)/ dot emphasis	The styles that are used for the text.
	properties	record	Property that allows setting of a list of properties.
	shade	percent	Shade of the first character of this text.
	size	fixed	The size of the first character of this text in points.
	style	text style info	The text styles applied to this text.
	track	fixed	Tracking of the first character of this text.
	trap text	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Trapping specification for the first character of this text.
•	uniform styles	text style info	Text styles that are uniformly applied to this text.
	vertical scale	percent	Vertical scale of the first character of this text.
•	width	horizontal measurement	Width of the first character of this text.

vertex Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(vertex[1],after(vertex[5]))	duplicate vertex 1 to after vertex 5
create	create(vertex,0,0,beginningof(it))	make vertex at beginning
data size	datasize(vertex[1].lefthandle,inttype)	datasize of left handle of vertex 1 as integer
delete	delete(vertex[1])	delete vertex 1
get	get(vertex[4].symmetry)	get symmetry of vertex 4
get as	getas(vertex[3].righthandle,listtype)	get right handle of vertex 3 as list
move	move(vertex[1],after(vertex[3]))	move vertex 1 to after vertex 3
set	set(vertex[1].anchor,"6 cm","10 cm")	set anchor of vertex 1 to {"6 cm", "10 cm"}

vertex Elements and Reference Forms

None

vertex Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	anchor	measurements point	The position of the vertex.
	left handle	measurements point	The position of the left control handle of the vertex.
	right handle	measurements point	The position of the right control handle of the vertex.
•	symmetry	corner/smooth/ symmetrical	The linking method for the control handles.

vertical guide Events and Examples

Verb	Frontier Example	AppleScript Example
clone	clone(verticalguide[1],after(verticalguide[5]))	duplicate vertical guide 1 to after vertical guide 5
create	create(verticalguide,0,0,beginningof(it))	make vertical guide at beginning
data size	datasize(verticalguide[1].scale,inttype)	datasize of scale of vertical guide 1 as integer
delete	delete(verticalguide[1])	delete vertical guide 1
get	get(verticalguide[4].undeletable)	get undeletable of vertical guide 4
get as	getas(verticalguide[3].position,inttype)	get position of vertical guide 3 as integer
move	move(verticalguide[1],after(verticalguide[3]))	move vertical guide 1 to after vertical guide 3
set	set(verticalguide[1].position,"6 cm")	set position of vertical guide 1 to "6 cm"

vertical guide Elements and Reference Forms

None

vertical guide Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	best type	type class	The best descriptor type.
•	class	type class	The class.
•	default type	type class	The default descriptor type.
•	from master	Boolean	If TRUE, this guide is from the master page.
•	index	integer	Index of the object.
•	object reference	reference	An object reference for this object.
	position	vertical measurement	Position of the guide.
	properties	record	Property that allows setting of a list of properties.
	scale	percent	Viewscale at which the guide will appear.
	undeletable	Boolean	If TRUE, guide can't be deleted.
	unmoveable	Boolean	If TRUE, guide can't be moved.

fixed point Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	horizontal	fixed	Horizontal component of point.
	left	fixed	Horizontal component of point.
	top	fixed	Vertical component of point.
	vertical	fixed	Vertical component of point.

fixed rectangle Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	bottom	fixed	Bottom side of fixed rectangle.
	bottom right	fixed point	Bottom right point of fixed rectangle.
	height	fixed	Height of fixed rectangle.
	left	fixed	Left side of fixed rectangle.
	origin	fixed point	Origin of fixed rectangle (changing offsets entire rectangle).
	right	fixed	Right side of fixed rectangle.
	top	fixed	Top side of fixed rectangle.
	top left	fixed point	Top left point of fixed rectangle.
	width	fixed	Width of fixed rectangle.

font record Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
•	ID	small integer	Font id.
•	name	plain text (string)	Font name.

frame record Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	color	color spec	Frame Color.
	gap color	color spec	The color of the line gaps.
	gap shade	percent	The shade of the line gaps.
	inside trap	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Trap specification for inside of frame.
	outside trap	default/overprint/ knockout/spread auto amount/choke auto amount, or fixed	Trap specification for outside of frame.
	shade	percent	Frame shade.
	style	solid line/sparsely dashed line/densely dashed line/dashed line/dotted line/double line/thin thick line/ thick thin line/thin thick thin line/thick thin thick line/thin thin thin line	Style of frame for box.
	width	thick units	Frame thickness.

justification record Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	maximum	percent	Maximum spacing expansion (as a percentage of width).
	minimum	percent	Minimum spacing expansion (as a percentage of width).
	optimum	percent	Optimum spacing expansion (as a percentage of width).

print setup record Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	adjust horizontal tile	Boolean	If TRUE, then allow auto tiling to increase the auto tile overlap to better fit tiles horizontally.
	auto tile overlap	horizontal measurement	The amount by which to overlap tiles when auto tiling is on.
	back to front	Boolean	If TRUE, then print pages in reverse order.
	collate	Boolean	If TRUE, then collate output pages.
	data format	ASCII data/binary data/clean data	Whether to send image data in ASCII or binary format to PostScript printers.
	fit in area	Boolean	If TRUE, the printed output will be sized to fit on the paper (printable area).
	flip horizontal	Boolean	If TRUE, flip output horizontally.
	flip vertical	Boolean	If TRUE, flip output vertically.
	halftone screen	fixed	Number of lines per inch at which to halftone pictures.
	include blank pages	Boolean	If TRUE, then include blank pages in printed output.
	invert image	Boolean	If TRUE, invert print image.
	orientation	portrait/landscape	Paper orientation.
	page gap	vertical measurement	Vertical gap between printed pages.
	page position	left position/center position/center horizontal/center vertical	The position of the page on the media.
	page sequence	all pages/odd pages/ even pages	Output page sequence.
	paper offset	vertical measurement	Offset from left edge of paper at which to begin printing.
	paper size	plain text (string)	Selected paper size.
•	paper size list	a list of plain text (strings)	List of available paper sizes.
	paper width	horizontal measurement	The width of paper in the printer.
	print colors as grays	Boolean	If TRUE, then print colors in representative shades of gray.
	print quality	normal/low resolution/ rough	Print quality.
	print spreads	Boolean	If TRUE, then print spreads.
	print thumbnails	Boolean	If TRUE, then print thumbnails.

print setup record Properties, Data Types, and Descriptions (continued)

R/O	Property Name	Type	Description
	printer type	plain text (string)	Selected printer type.
•	printer type list	a list of plain text (strings)	List of available/known printer types.
	reduce or enlarge	percent	The scale at which to print.
	registration marks	off/center/off center	Registration marks setting.
	resolution	small integer	Number of dots per inch at which to print the document.
	separation	Boolean	If TRUE, then separation is on.
	tiling	off/manual/automatic	If TRUE, then tiling is on.

rule record Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	color	color spec	Specifies the color of the rule.
	left indent	horizontal measurement	Specifies the distance between the left end of a rule and the left indent or the left end of a line of text above/below the paragraph.
	position	vertical measurement	Specifies the value of the offset of the rule away from the paragraph.
	right indent	horizontal measurement	Specifies the distance between the right end of a rule and the right indent or the right end of a line of text above/below the paragraph.
	rule on	Boolean	If TRUE, then paragraph rule is on.
	shade	percent	Specifies the shade of the color of the rule.
	style	solid line/sparsely dashed line/densely dashed line/dashed line/ dotted line/double line/ thin thick line/thick thin line/thin thick thin line/thick thin thick line/thin thin thin line	Style of the rule for box.
	text length	Boolean	If TRUE, then indents rule by the width of the first/last line of the paragraph.
	width	thick units	Specifies the width of the rule.

tab record Properties, Data Types, and Descriptions

R/O	Property Name	Type	Description
	align character	plain text (character)	Character to align on.
	fill character	plain text (character)	Characters to fill tab area with.
	justification	left justified/ right justified/ centered/align on	Tab justification.
	position	horizontal measurement	Horizontal offset of tab from left side of container.

Glossary

Apple Events
Messages sent from one Mac OS application or process to another that give instructions, respond to instructions, and send or receive data. Apple events are defined by Apple Computer, Inc. or other application developers and must conform to the Apple Event Interprocess Messaging Protocol.

Apple Event Registry: Standard Suites
A compilation of standard Apple events defined by Apple Computer, Inc. or other application developers including: Apple events, object classes, primitive object classes, descriptor types, key forms, and constants. The *Apple Event Registry: Standard Suites* is maintained by the Apple Event Developers' Association.

Apple Event Interprocess Messaging Protocol
The protocol for interapplication communication defined by Apple Computer, Inc. Interapplication messages must conform to this protocol to qualify as Apple events.

AppleScript
A system-wide scripting language developed by Apple Computer, Inc. AppleScript scripts can control the Mac OS operating system and applications that support Apple events.

Attribute
The component of an Apple event that identifies it and the tasks it can perform on the data specified in the parameters. Attributes consist of an event class and event ID.

Container
The object that contains the element specified by an Apple event.

Element
An object contained by another Apple events object. The element classes in the *Apple Event Registry: Standard Suites* define the types of objects each Apple events object can contain.

Event Class
The attribute of an Apple event that identifies which suite (group of related Apple events) it belongs to such as the Required Suite, Standard Suite, etc.

Event ID
The attribute of an Apple event that uniquely identifies it within its event class and defines the tasks it can perform.

Events
The part of an Apple events message that tells objects what to do (similar to a verb).

Functional-area Suites
Groups of objects and events that relate to similar functional areas, including: the Text Suite, the QuickDraw Graphics Suite, the Table Suite, and Miscellaneous Standards.

Insertion Points
A reference with a parameter that defines where in the container hierarchy an object should be placed.

Miscellaneous Suite
Basic Apple events, related to the clipboard and other menu-driven functions, that most applications supports. The events include: cut, copy, paste, undo, etc.

Object Class
A category for Apple event objects that share specific characteristics and capabilities.

Object Model
The Apple events Object Model is a standard message structure that allows Mac OS applications to communicate. Messages built according to the Object Model consist of events, objects, and potentially data.

Object
A distinct item in an application that can respond to an Apple event. Objects are usually items a user can identify and manipulate.

Object Hierarchy
The breakdown of an application into specific objects and object classes. To support the Standard and Functional-area Suites, an application must define an object hierarchy based on standard classifications in the *Apple Event Registry: Standard Suites*.

Parameter
A method for identifying the object an Apple event is sent to, the task it will perform, and the desired options for performing the task.

Property
Characteristics used to describe Apple events objects in the same object class.

QuarkXPress Suite
The Suite that defines all the events and objects (and their properties) specific to QuarkXPress.

Reference
The component of an Apple event that first identifies the container enclosing a specific object and then uses a reference form to separate a specific object from all possible objects in the container.

Reference Form
A parameter that identifies the specific object in a container that an Apple event is sent to. QuarkXPress objects can be referenced by index, name, relative position, or test.

Required Suite
Four Apple events, sent from the Finder, that all scriptable applications support: open application, open documents, print documents, quit application. In QuarkXPress all of the required items are handled by equivalent items in the Standard Suite.

Scripts
A series of statements in a scripting language that send Apple events to applications asking them to perform a series of tasks. Scripts combine the scripting language syntax with the standard Apple events vocabulary defined in the object model.

Standard Suite
The basic Apple events and objects that most applications use to communicate. The events include: get, set, create, duplicate, move, delete, count, close, save, print, open, data size, and exists. Objects include windows, documents, pages, etc.

Suite
A group of objects and events that relate to a common purpose.

Text Suite
The Functional-area Suite that defines all objects (and their properties) related to working with text in any application.

UserTalk
A system-wide scripting language developed by UserLand Software, Inc. The application that is used to create Frontier scripts is UserLand Frontier. Frontier scripts can control the Mac OS operating system and applications that support Apple events.