appearance state. The optional **A** entry in the outline item dictionary (see "Table 151: Entries in an outline item dictionary") and the dictionaries of some annotation types (see "Table 176: Additional entries specific to a link annotation", "Table 190: Additional entries specific to a screen annotation", "Table 191: Additional entries specific to a widget annotation" and "Table 249: Entries in an FDF field dictionary") specifies an action performed when the annotation or outline item is activated; in PDF 1.2, a variety of other circumstances may trigger an action as well (see 12.6.3, "Trigger events"). In addition, the optional **OpenAction** entry in a document's catalog dictionary (7.7.2, "Document catalog dictionary") may specify an action that shall be performed when the document is opened. PDF includes a wide variety of standard action types, described in detail in 12.6.4, "Action types".

12.6.2 Action dictionaries

An *action dictionary* defines the characteristics and behaviour of an action. "Table 196: Entries common to all action dictionaries" shows the required and optional entries that are common to all action dictionaries. The dictionary may contain additional entries specific to a particular action type; see the descriptions of individual action types in 12.6.4, "Action types" for details.

Table 196: Entries common to all action dictionaries

Key	Type	Value
Type	name	*(Optional)* The type of PDF object that this dictionary describes; if present, shall be *Action* for an action dictionary.
S	name	*(Required)* The type of action that this dictionary describes; see "Table 201: Action types" for specific values.
Next	dictionary or array	*(Optional; PDF 1.2)* The next action or sequence of actions that shall be performed after the action represented by this dictionary. The value is either a single action dictionary or an array of action dictionaries that shall be performed in order; see Note 1 for further discussion.

NOTE 1 The action dictionary's **Next** entry (*PDF 1.2*) allows sequences of actions to be chained together. For example, the effect of clicking a link annotation with the mouse might be to play a sound, jump to a new page, and start up a movie. Note that the **Next** entry is not restricted to a single action but may contain an array of actions, each of which in turn may have a **Next** entry of its own. The actions may thus form a tree instead of a simple linked list. Actions within each **Next** array are executed in order, each followed in turn by any actions specified in *its* **Next** entry, and so on recursively. It is recommended that interactive PDF processors attempt to provide reasonable behaviour in anomalous situations. For example, self-referential actions ought not be executed more than once, and actions that close the document or otherwise render the next action impossible ought to terminate the execution sequence. Applications need also provide some mechanism for the user to interrupt and manually terminate a sequence of actions.

PDF 1.5 introduces transition actions, which allow the control of drawing during a sequence of actions; see 12.6.4.15, "Transition actions".

NOTE 2 It is recommended that no action modify its own action dictionary or any other in the action tree in which it resides. The effect of such modification on subsequent execution of actions in the tree is undefined.

12.6.3 Trigger events

An annotation, page object, or (beginning with PDF 1.3) interactive form field may include an entry named **AA** that specifies an *additional-actions dictionary* (*PDF 1.2*) that extends the set of events that can trigger the execution of an action. In PDF 1.4, the document catalog dictionary (see 7.7.2, "Document catalog dictionary") may also contain an **AA** entry for trigger events affecting the document as a whole. "Table 197: Entries in an annotation's additional-actions dictionary" to "Table 200: Entries in the

document catalog's additional-actions dictionary" show the contents of this type of dictionary.

PDF 1.5 introduces four trigger events in annotation's additional-actions dictionary to support multimedia presentations:

- The **PO** and **PC** entries have a similar function to the **O** and **C** entries in the page object's additional-actions dictionary (see "Table 197: Entries in an annotation's additional-actions dictionary"). However, associating these triggers with annotations allows annotation objects to be self-contained.

EXAMPLE Annotations containing such actions can be copied or moved between pages without requiring page open/close actions to be changed.

- The **PV** and **PI** entries allow a distinction between pages that are open and pages that are visible. At any one time, while more than one page may be visible, depending on the page layout.

NOTE 1 For these trigger events, the values of the flags specified by the annotation's **F** entry (see 12.5.3, "Annotation flags") have no bearing on whether a given trigger event occurs.

Table 197: Entries in an annotation's additional-actions dictionary

Key	Type	Value
E	dictionary	*(Optional; PDF 1.2)* An action that shall be performed when the cursor enters the annotation's active area.
X	dictionary	*(Optional; PDF 1.2)* An action that shall be performed when the cursor exits the annotation's active area.
D	dictionary	*(Optional; PDF 1.2)* An action that shall be performed when the mouse button is pressed inside the annotation's active area.
U	dictionary	*(Optional; PDF 1.2)* An action that shall be performed when the mouse button is released inside the annotation's active area. For backward compatibility, the **A** entry in an annotation dictionary, if present, takes precedence over this entry (see "Table 170: Entries in an appearance dictionary").
Fo	dictionary	*(Optional; PDF 1.2; widget annotations only)* An action that shall be performed when the annotation receives the input focus.
Bl	dictionary	*(Optional; PDF 1.2; widget annotations only)* (Uppercase B, lowercase L) An action that shall be performed when the annotation loses the input focus.
PO	dictionary	*(Optional; PDF 1.5)* An action that shall be performed when the page containing the annotation is opened. EXAMPLE 1 When the user navigates to it from the next or previous page or by means of a link annotation or outline item. The action shall be executed after the **O** action in the page's additional-actions dictionary (see "Table 198: Entries in a page object's additional-actions dictionary") and the **OpenAction** entry in the document **Catalog** (see "Table 29: Entries in the catalog dictionary"), if such actions are present.

Key	Type	Value
PC	dictionary	*(Optional; PDF 1.5)* An action that shall be performed when the page containing the annotation is closed. EXAMPLE 2 When the user navigates to the next or previous page, or follows a link annotation or outline item. The action shall be executed before the **C** action in the page's additional-actions dictionary (see "Table 198: Entries in a page object's additional-actions dictionary"), if present.
PV	dictionary	*(Optional; PDF 1.5)* An action that shall be performed when the page containing the annotation becomes visible.
PI	dictionary	*(Optional; PDF 1.5)* An action that shall be performed when the page containing the annotation is no longer visible in the interactive PDF processor's user interface.

Table 198: Entries in a page object's additional-actions dictionary

Key	Type	Value
O	dictionary	*(Optional; PDF 1.2)* An action that shall be performed when the page is opened (for example, when the user navigates to it from the next or previous page or by means of a link annotation or outline item). This action is independent of any that may be defined by the **OpenAction** entry in the document catalog dictionary (see 7.7.2, "Document catalog dictionary") and shall be executed after such an action.
C	dictionary	*(Optional; PDF 1.2)* An action that shall be performed when the page is closed (for example, when the user navigates to the next or previous page or follows a link annotation or an outline item). This action applies to the page being closed and shall be executed before any other page is opened.

Table 199: Entries in a form field's additional-actions dictionary

Key	Type	Value
K	dictionary	*(Optional; PDF 1.3)* An ECMAScript action that shall be performed when the user modifies a character in a text field or combo box or modifies the selection in a scrollable list box. This action may check the added text for validity and reject or modify it.
F	dictionary	*(Optional; PDF 1.3)* An ECMAScript action that shall be performed before the field is formatted to display its value. This action may modify the field's value before formatting.
V	dictionary	*(Optional; PDF 1.3)* An ECMAScript action that shall be performed when the field's value is changed. This action may check the new value for validity. (The name **V** stands for "validate.")

Key	Type	Value
C	dictionary	*(Optional; PDF 1.3)* An ECMAScript action that shall be performed to recalculate the value of this field when that of another field changes. (The name **C** stands for "calculate.") The order in which the document's fields are recalculated shall be defined by the **CO** entry in the interactive form dictionary (see 12.7.3, "Interactive form dictionary").

Table 200: Entries in the document catalog's additional-actions dictionary

Key	Type	Value
WC	dictionary	*(Optional; PDF 1.4)* An ECMAScript action that shall be performed before closing a document. (The name **WC** stands for "will close.")
WS	dictionary	*(Optional; PDF 1.4)* An ECMAScript action that shall be performed before saving a document. (The name **WS** stands for "will save.")
DS	dictionary	*(Optional; PDF 1.4)* An ECMAScript action that shall be performed after saving a document. (The name **DS** stands for "did save.")
WP	dictionary	*(Optional; PDF 1.4)* An ECMAScript action that shall be performed before printing a document. (The name **WP** stands for "will print.")
DP	dictionary	*(Optional; PDF 1.4)* An ECMAScript action that shall be performed after printing a document. (The name **DP** stands for "did print.")

Interactive PDF processors shall ensure the presence of such a device, or equivalent controls for simulating one, for the corresponding actions to be executed correctly. Mouse-related trigger events are subject to the following constraints:

- An **E** (enter) event may occur only when the mouse button is up.
- An **X** (exit) event may not occur without a preceding **E** event.
- A **U** (up) event may not occur without preceding **E** and **D** events.
- In the case of overlapping or nested annotations, entering a second annotation's active area causes an **X** event to occur for the first annotation.

NOTE 2 The field-related trigger events **K** (keystroke), **F** (format), V (validate), and **C** (calculate) are not defined for button fields (see 12.7.5.2, "Button fields"). The effects of an action triggered by one of these events are limited only by the action itself and can occur outside the described scope of the event. For example, even though the **F** event is used to trigger actions that format field values prior to display, it is possible for an action triggered by this event to perform a calculation or make any other modification to the document.

These field-related trigger events can occur either through user interaction or programmatically, such as in response to the **NeedAppearances** entry (deprecated in PDF 2.0) in the interactive form dictionary (see 12.7.3, "Interactive form dictionary"), importation of **FDF** data (12.7.8, "Forms data format"), or **ECMAScript** actions (12.6.4.17, "ECMAScript actions"). For example, the user's modifying a field value can trigger a cascade of calculations and further formatting and validation for other fields in the document.

12.6.4 Action types

12.6.4.1 General

PDF supports the standard action types listed in "Table 201: Action types". The following subclauses describe each of these types in detail.

Table 201: Action types

Action type	Description	Discussed in subclause
GoTo	Go to a destination in the current document.	12.6.4.2, "Go-To actions"
GoToR	("Go-to remote") Go to a destination in another document.	12.6.4.3, "Remote Go-To actions"
GoToE	("Go-to embedded"; *PDF 1.6*) Go to a destination in an embedded file.	12.6.4.4, "Embedded Go-To actions"
GoToDPart	("Go-to document part"; PDF 2.0) Go to a specified DPart the current document.	12.6.4.5, "GoToDPart action"
Launch	Launch an application, usually to open a file.	12.6.4.6, "Launch actions"
Thread	Begin reading an article thread.	12.6.4.7, "Thread actions"
URI	Resolve a uniform resource identifier.	12.6.4.8, "URI actions"
Sound	*(PDF 1.2; deprecated in PDF 2.0)* Play a sound.	12.6.4.9, "Sound actions"
Movie	*(PDF 1.2; deprecated in PDF 2.0)* Play a movie.	12.6.4.10, "Movie actions"
Hide	*(PDF 1.2)* Set an annotation's Hidden flag.	12.6.4.11, "Hide actions"
Named	*(PDF 1.2)* Execute a predefined action.	12.6.4.12, "Named actions"
SubmitForm	*(PDF 1.2)* Send data to a uniform resource locator.	12.7.6.2, "Submit-form action"
ResetForm	*(PDF 1.2)* Set fields to their default values.	12.7.6.3, "Reset-form action"
ImportData	*(PDF 1.2)* Import field values from a file.	12.7.6.4, "Import-data action"
SetOCGState	*(PDF 1.5)* Set the states of optional content groups.	12.6.4.13, "Set-OCG-state actions"
Rendition	*(PDF 1.5)* Controls the playing of multimedia content.	12.6.4.14, "Rendition actions"
Trans	*(PDF 1.5)* Updates the display of a document, using a transition dictionary.	12.6.4.15, "Transition actions"
GoTo3DView	*(PDF 1.6)* Set the current view of a 3D annotation	12.6.4.16, "Go-To-3D-View actions"
JavaScript	*(PDF 1.3)* Execute a ECMAScript script.	12.6.4.17, "ECMAScript actions"
RichMediaExecute	*(PDF 2.0; RichMedia annotation only)* Specifies a command to be sent to the annotation's handler.	12.6.4.18, "Rich-Media-Execute actions"

12.6.4.2 Go-To actions

A *go-to action* changes the view to a specified destination (page, location, and magnification factor). "Table 202: Additional entries specific to a go-to action" shows the action dictionary entries specific to this type of action.

Table 202: Additional entries specific to a go-to action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *GoTo* for a go-to action.
D	name, byte string, or array	*(Required)* The destination to jump to (see 12.3.2, "Destinations").
SD	array	*(Optional; PDF 2.0)* The structure destination to jump to (see 12.3.2.3, "Structure destinations"). If present, the structure destination should take precedence over destination in the **D** entry.

NOTE Specifying a go-to action in the **A** entry of a link annotation or outline item (see "Table 176: Additional entries specific to a link annotation" and "Table 151: Entries in an outline item dictionary") has the same effect as specifying the destination directly with the **Dest** entry. For example, the link annotation shown in the Example in 12.6.4.13, "Set-OCG-state actions" which uses a go-to action, has the same effect as the one in the following Example, which specifies the destination directly. However, the go-to action is less compact and is not compatible with PDF 1.0; therefore, using a direct destination is preferable.

EXAMPLE

```
93 0 obj
      <</Type /Annot
         /Subtype /Link
         /Rect [71 717 190 734]
         /Border [16 16 1]
         /A <</Type /Action
               /S /GoTo
               /D [3 0 R /FitR -4 399 199 533]
            >>
      >>
endobj
```

12.6.4.3 Remote Go-To actions

A remote go-to action is similar to an ordinary go-to action but jumps to a destination in another PDF file instead of the current file. "Table 203: Additional entries specific to a remote go-to action" shows the action dictionary entries specific to this type of action.

NOTE Remote go-to actions cannot be used with embedded files; see 12.6.4.4, "Embedded Go-To actions"

Table 203: Additional entries specific to a remote go-to action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *GoToR* for a remote go-to action.
F	file specification	*(Required)* The file in which the destination shall be located.

Key	Type	Value
D	name, byte string, or array	*(Required)* The destination to jump to (see 12.3.2, "Destinations"). If the value is an array defining an explicit destination (as described under 12.3.2.2, "Explicit destinations"), its first element shall be a page number within the remote document rather than an indirect reference to a page object in the current document. The first page shall be numbered 0.
SD	array	*(Optional; PDF 2.0)* The structure destination to jump to (see 12.3.2.3, "Structure destinations"). The first element in the array shall be a byte string representing a structure element **ID** in the remote document, instead of an indirect reference to a structure element dictionary. If present, the structure destination should take precedence over destination in the **D** entry.
NewWindow	boolean	*(Optional; PDF 1.2)* A flag specifying whether to open the destination document in a new window. If this flag is *false*, the destination document replaces the current document in the same window. If this entry is absent, the interactive PDF processor should behave in accordance with its preference.

12.6.4.4 Embedded Go-To actions

An embedded go-to action (PDF 1.6) is similar to a remote go-to action but allows jumping to or from a PDF file that is embedded in another PDF file (see 7.11.4, "Embedded file streams"). Embedded files may be associated with file attachment annotations (see 12.5.6.15, "File attachment annotations") or with entries in the **EmbeddedFiles** name tree (see 7.7.4, "Name dictionary"). Embedded files may in turn contain embedded files. "Table 204: Additional entries specific to an embedded go-to action" shows the action dictionary entries specific to embedded go-to actions.

NOTE Embedded go-to actions work only for files of Type PDF.

Embedded go-to actions provide a complete facility for linking between a file in a hierarchy of nested embedded files and another file in the same or different hierarchy. The following terminology shall be used:

- The *source* is the document containing the embedded go-to action.
- The *target* is the document in which the destination lives.
- The **T** entry in the action dictionary is a target dictionary that locates the target in relation to the source, in much the same way that a relative path describes the physical relationship between two files in a file system. Target dictionaries may be nested recursively to specify one or more intermediate targets before reaching the final one. As the hierarchy is navigated, each intermediate target shall be referred to as the *current* document. Initially, the source is the current document.

NOTE It is an error for a target dictionary to have an infinite cycle (for example, one where a target dictionary refers to itself). Interactive PDF processors need to attempt to detect such cases and refuse to execute the action if one is found.

- A *child* document shall be one that is embedded within another PDF file.
- The document in which a file is embedded shall be its *parent*.
- A *root document* is one that is not embedded in another PDF file. The target and source may be contained in root documents or embedded documents.

Table 204: Additional entries specific to an embedded go-to action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *GoToE* for an embedded go-to action.
F	file specification	*(Optional)* The root document of the target relative to the root document of the source. If this entry is absent, the source and target share the same root document.
D	name, byte string, or array	*(Required)* The destination in the target to jump to (see 12.3.2, "Destinations").
NewWindow	boolean	*(Optional)* If *true*, the destination document should be opened in a new window; if *false*, the destination document should replace the current document in the same window. If this entry is absent, the interactive PDF processor should act according to its preference.
T	dictionary	*(Optional if F is present; otherwise required)* A *target dictionary* (see "Table 205: Entries specific to a target dictionary") specifying path information to the target document. Each target dictionary specifies one element in the full path to the target and may have nested target dictionaries specifying additional elements.

Table 205: Entries specific to a target dictionary

Key	Type	Value
R	name	*(Required)* Specifies the relationship between the current document and the target (which may be an intermediate target). Valid values are *P* (the target is the parent of the current document) and *C* (the target is a child of the current document).
N	byte string	(*Required if the value of **R** is C and the target is located in the* **EmbeddedFiles** *name tree; otherwise, it shall be absent*) The name of the file in the **EmbeddedFiles** name tree.
P	integer or byte string	*(Required if the value of **R** is C and the target is associated with a file attachment annotation; otherwise, it shall be absent)* If the value is an integer, it specifies the page number (zero-based) in the current document containing the file attachment annotation. If the value is a string, it specifies a named destination in the current document that provides the page number of the file attachment annotation.
A	integer or text string	*(Required if the value of **R** is C and the target is associated with a file attachment annotation; otherwise, it shall be absent)* If the value is an integer, it specifies the index (zero-based) of the annotation in the **Annots** array (see "Table 31: Entries in a page object") of the page specified by **P**. If the value is a text string, it specifies the value of **NM** in the annotation dictionary (see "Table 166: Entries common to all annotation dictionaries").
T	dictionary	*(Optional)* A target dictionary specifying additional path information to the target document. If this entry is absent, the current document is the target file containing the destination.

EXAMPLE The following example illustrates several possible relationships between source and target. Each object shown is an action dictionary for an embedded go-to action.

```
1 0 ob                                  %Link to a child
      <</Type /Action
         /S /GoToE
         /D (Chapter 1)
         /T  <</R /C
               /N (Embedded document)>>
      >>
endobj

2 0 obj                                 %Link to the parent
      <</Type /Action
         /S /GoToE
         /D (Chapter 1)
         /T  <</R /P>>
      >>
endobj

3 0 obj                                 %Link to a sibling
      <</Type /Action
         /S /GoToE
         /D (Chapter 1)
         /T <</R /P
               /T <</R /C
                     /N (Another embedded document)>>
            >>
      >>
endobj

4 0 obj                                 %Link to an embedded file in an external document
      <</Type /Action
         /S /GoToE
         /D (Chapter 1)
         /F (someFile.pdf)
         /T <</R /C
               /N (Embedded document)>>
   >>
endobj

5 0 obj                                 %Link from an embedded file to a normal file
      <</Type /Action
         /S /GoToE
         /D (Chapter 1)
         /F (someFile.pdf)
      >>
endobj

      6 0 obj                           %Link to a grandchild
         <</Type /Action
            /S /GoToE
            /D (Chapter 1)
            /T <</R /C
                  /N (Embedded document)
                  /T <</R /C
                        /P (A destination name)
                        /A (annotName)
                     >>

               >>
         >>
endobj

7 0 obj                                 %Link to a niece/nephew through the source's parent
      <</Type /Action
```

```
            /S /GoToE
            /D (destination)
            /T <</R /P
                  /T <</R /C
                        /N (Embedded document)
                        /T <</R /C
                              /P 3
                              /A (annotName)
                           >>
                     >>
               >>
         >>
endobj
```

12.6.4.5 GoToDPart action

A *GoToDPart action* changes the view to the **Start** page of a specified DPart (see "Table 409: Entries in a DPart dictionary"). "Table 206: Entries in a GoToDPart dictionary" shows the action dictionary entries specific to this type of action.

Table 206: Entries in a GoToDPart dictionary

Key	Type	Value
S	name	*(Required; PDF 2.0)* The type of action that this dictionary describes; shall be *GoToDp* for a go-to document part action.
Dp	dictionary	*(Required; PDF 2.0)* The indirect reference to a DPart dictionary to go to.

EXAMPLE 1 Using a GoToDp action reference

```
92 0 obj
   <<
      /Parent 6 0 R
      /Start 100 0 R
      /End 101 0 R
      /Properties <</PartType (Cover)>>
   >>
endobj

93 0 obj
   <<
      /Type /Annot
      /Subtype /Link
      /Rect [71 717 190 734]
      /Border [16 16 1]
      /A
      <<
         /Type /Action
         /S /GoToDp
         /Dp 92 0 R
      >>
   >>
endobj
```

12.6.4.6 Launch actions

A *launch action* launches an application or opens or prints a document. "Table 207: Additional entries specific to a launch action" shows the action dictionary entries specific to this type of action.

Previously, **Win**, **Mac**, and **Unix** entries allowed the action dictionary to include platform-specific parameters for launching the designated application, however, they are now deprecated with PDF 2.0. The **F** entry determines the file specification platform to be launched.

Table 207: Additional entries specific to a launch action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *Launch* for a launch action.
F	file specification	*(Required if none of the entries **Win**, **Mac**, or **Unix** is present)* The application that shall be launched or the document that shall be opened or printed. If this entry is absent and the interactive PDF processor does not understand any of the alternative entries, it shall do nothing.
Win	dictionary	*(Optional; deprecated in PDF 2.0)* A dictionary containing Microsoft Windows™ specific launch parameters (see "Table 208: Entries in a Microsoft Windows™ launch parameter dictionary").
Mac	(undefined)	*(Optional; deprecated in PDF 2.0)* Mac OS–specific launch parameters; not yet defined.
Unix	(undefined)	*(Optional; deprecated in PDF 2.0)* UNIX-specific launch parameters; not yet defined.
NewWindow	boolean	*(Optional; PDF 1.2)* A flag specifying whether to open the destination document in a new window. If this flag is *false*, the destination document replaces the current document in the same window. If this entry is absent, the interactive PDF processor should behave in accordance with its current preference. This entry shall be ignored if the file designated by the **F** entry is not a PDF document.

Table 208: Entries in a Microsoft Windows™ launch parameter dictionary

Key	Type	Value
F	byte string	*(Required)* The file name of the application that shall be launched or the document that shall be opened or printed, in standard Microsoft Windows™ specific pathname format. If the name string includes a backslash character (\), the backslash shall itself be preceded by a backslash. This value shall be a simple string; it is not a file specification.
D	byte string	*(Optional)* A bye string specifying the default directory in standard DOS syntax.
O	ASCII string	*(Optional)* An ASCII string specifying the operation to perform: *open* Open a document. *print* Print a document. If the **F** entry designates an application instead of a document, this entry shall be ignored and the application shall be launched. Default value: *open*.

Key	Type	Value
P	byte string	*(Optional)* A parameter string that shall be passed to the application designated by the **F** entry. This entry shall be omitted if **F** designates a document.

12.6.4.7 Thread actions

A *thread action* jumps to a specified bead on an article thread (see 12.4.3, "Articles"), in either the current document or a different one. "Table 209: Additional entries specific to a thread action" shows the action dictionary entries specific to this type of action.

Table 209: Additional entries specific to a thread action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *Thread* for a thread action.
F	file specification	*(Optional)* The file containing the thread. If this entry is absent, the thread is in the current file.
D	dictionary, integer, or text string	*(Required)* The destination thread, specified in one of the following forms: An indirect reference to a thread dictionary (see 12.4.3, "Articles"). In this case, the thread shall be in the current file. The index of the thread within the **Threads** array of its document's catalog dictionary (see 7.7.2, "Document catalog dictionary"). The first thread in the array has index 0. The title of the thread as specified in its thread information dictionary (see "Table 162: Entries in a thread dictionary"). If two or more threads have the same title, the one appearing first in the document catalog's **Threads** array shall be used.
B	dictionary or integer	*(Optional)* The bead in the destination thread, specified in one of the following forms: An indirect reference to a bead dictionary (see 12.4.3, "Articles"). In this case, the thread shall be in the current file. The index of the bead within its thread. The first bead in a thread has index 0.

The title of the thread as specified in its thread information dictionary (see "Table 162: Entries in a thread dictionary"). If two or more threads have the same title, the one appearing first in the document catalog's **Threads** array shall be used.

12.6.4.8 URI actions

A **uniform resource identifier** (URI) is a string that identifies (resolves to) a resource on the Internet — typically a file that is the destination of a hypertext link, although it may also resolve to a query or other entity. (URIs are described in Internet RFC 3986, *Uniform Resource Identifiers (URI): Generic Syntax*.)

A *URI action* causes a URI to be resolved. "Table 210: Additional entries specific to a URI action" shows

the action dictionary entries specific to this type of action.

Table 210: Additional entries specific to a URI action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *URI* for a URI action.
URI	ASCII string	*(Required)* The uniform resource identifier to resolve, encoded in UTF8.
IsMap	boolean	*(Optional)* A flag specifying whether to track the mouse position when the URI is resolved (see the discussion following this Table). Default value: *false*. This entry applies only to actions triggered by the user's clicking an annotation; it shall be ignored for actions associated with outline items or with a document's **OpenAction** entry.

If the **IsMap** flag is *true* and the user has triggered the URI action by clicking an annotation, the coordinates of the mouse position at the time the action has been triggered shall be transformed from device space to user space and then offset relative to the upper-left corner of the annotation rectangle (that is, the value of the **Rect** entry in the annotation with which the URI action is associated).

EXAMPLE 1 If the mouse coordinates in user space are (x_m, y_m) and the annotation rectangle extends from (ll_x, ll_y) at the lower-left to (ur_x, ur_y) at the upper-right, the final coordinates (x_f, y_f) are as follows:

$$(x_f = x_m - ll_x)$$
$$y_f = ur_y - y_m$$

If the resulting coordinates (x_f, y_f) are fractional, they shall be rounded to the nearest integer values. They shall then be appended to the URI to be resolved, separated by COMMAS (2Ch) and preceded by a QUESTION MARK (3Fh), as shown in this example:

EXAMPLE 2 http://www.iso.org/intro?100,200

To support URI actions, a PDF document's catalog dictionary (see 7.7.2, "Document catalog dictionary") may include a **URI** entry whose value is a URI dictionary. Only one entry shall be defined for such a dictionary (see "Table 211: Entry in a URI dictionary").

Table 211: Entry in a URI dictionary

Key	Type	Value
Base	ASCII string	*(Optional)* The *base URI* that shall be used in resolving relative URI references. URI actions within the document may specify URIs in partial form, to be interpreted relative to this base address. If no base URI is specified, such partial URIs shall be interpreted relative to the location of the document itself. The use of this entry is parallel to that of the body element <BASE>, as described in the *HTML 4.01 Specification*.

NOTE The **Base** entry allows the URI of the document to be recorded in situations in which the document is accessed out of context. For example, if a document has been moved to a new location but contains relative links to other documents that have not been moved, the **Base** entry could be used to refer such links to the true location of the other documents, rather than that of the moved document.

12.6.4.9 Sound actions

The features described in this subclause are deprecated with PDF 2.0. They are superseded by the general multimedia framework described in 13.2, "Multimedia".

A *sound action* (*PDF 1.2*) plays a sound through the computer's speakers. "Table 212: Additional entries specific to a sound action" shows the action dictionary entries specific to this type of action. Sounds are discussed in 13.3, "Sounds".

Table 212: Additional entries specific to a sound action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *Sound* for a sound action.
Sound	stream	*(Required)* A sound object defining the sound that shall be played (see 13.3, "Sounds").
Volume	number	*(Optional)* The volume at which to play the sound, in the range -1.0 to 1.0. Default value: 1.0.
Synchronous	boolean	*(Optional)* A flag specifying whether to play the sound synchronously or asynchronously. If this flag is *true*, the interactive PDF processor retains control, allowing no further user interaction other than cancelling the sound, until the sound has been completely played. Default value: *false*.
Repeat	boolean	*(Optional)* A flag specifying whether to repeat the sound indefinitely. If this entry is present, the **Synchronous** entry shall be ignored. Default value: *false*.
Mix	boolean	*(Optional)* A flag specifying whether to mix this sound with any other sound already playing. If this flag is *false*, any previously playing sound shall be stopped before starting this sound; this can be used to stop a repeating sound (see **Repeat**). Default value: *false*.

12.6.4.10 Movie actions

The features described in this subclause are deprecated with PDF 2.0. They are superseded by the general multimedia framework described in 13.2, "Multimedia".

A *movie action* (*PDF 1.2*) can be used to play a movie in a floating window or within the annotation rectangle of a movie annotation (see 12.5.6.17, "Movie annotations" and 13.4, "Movies"). The movie annotation shall be associated with the page that is the destination of the link annotation or outline item containing the movie action, or with the page object with which the action is associated.

NOTE A movie action by itself does not guarantee that the page the movie is on will be displayed before attempting to play the movie; such page change actions shall be done explicitly.

The contents of a movie action dictionary are identical to those of a movie activation dictionary (see "Table 307: Entries in a movie activation dictionary"), with the additional entries shown in "Table 213: Additional entries specific to a movie action". The contents of the activation dictionary associated with the movie annotation provide the default values. Any information specified in the movie action dictionary overrides these values.

Table 213: Additional entries specific to a movie action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *Movie* for a movie action.
Annotation	dictionary	*(Optional)* An indirect reference to a movie annotation identifying the movie that shall be played. The dictionary shall include either an Annotation or a T entry but not both.
T	text string	*(Optional)* The title of a movie annotation identifying the movie that shall be played. The dictionary shall include either an **Annotation** or a **T** entry but not both.
Operation	name	*(Optional)* The operation that shall be performed on the movie: *Play* Start playing the movie, using the play mode specified by the dictionary's **Mode** entry (see "Table 307: Entries in a movie activation dictionary"). If the movie is currently paused, it shall be repositioned to the beginning before playing (or to the starting point specified by the dictionary's **Start** entry, if present). *Stop* Stop playing the movie. *Pause* Pause a playing movie. *Resume* Resume a paused movie. Default value: *Play.*

12.6.4.11 Hide actions

A *hide action* (*PDF 1.2*) hides or shows one or more annotations on the screen by setting or clearing their Hidden flags (see 12.5.3, "Annotation flags"). This type of action can be used in combination with appearance streams and trigger events (see 12.5.5, "Appearance streams" and 12.6.3, "Trigger events") to display popup help information on the screen.

NOTE The **E** (enter) and **X** (exit) trigger events in an annotation's additional-actions dictionary can be used to show and hide the annotation when the user rolls the cursor in and out of its active area on the page. This can be used to pop up a help label, or tool tip, describing the effect of clicking at that location on the page.

"Table 214: Additional entries specific to a hide action" shows the action dictionary entries specific to this type of action.

Table 214: Additional entries specific to a hide action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *Hide* for a hide action.

Key	Type	Value
T	dictionary, text string, or array	*(Required)* The annotation or annotations to be hidden or shown, shall be specified in any of the following forms: • An indirect reference to an annotation dictionary • A text string giving the fully qualified field name of an interactive form field whose associated widget annotation or annotations are to be affected (see 12.7.4.2, "Field names") • An array of such dictionaries or text strings
H	boolean	*(Optional)* A flag indicating whether to hide the annotation (*true*) or show it (*false*). Default value: *true*.

12.6.4.12 Named actions

"Table 215: Named actions" lists several *named actions* (*PDF 1.2*) that interactive PDF processors shall support; further names may be added in the future.

Table 215: Named actions

Name	Action
NextPage	Go to the next page of the document.
PrevPage	Go to the previous page of the document.
FirstPage	Go to the first page of the document.
LastPage	Go to the last page of the document.

NOTE Interactive PDF processors may support additional, nonstandard named actions, but any document using them is not portable.

If the PDF processor encounters a named action that is inappropriate for a viewing platform, or if the viewer does not recognise the name, it shall take no action.

"Table 216: Additional entries specific to named actions" shows the action dictionary entries specific to named actions.

Table 216: Additional entries specific to named actions

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *Named* for a named action.
N	name	*(Required)* The name of the action that shall be performed (see "Table 215: Named actions").

12.6.4.13 Set-OCG-state actions

A *set-OCG-state action* (*PDF 1.5*) sets the state of one or more optional content groups (see 8.11,

"Optional content"). "Table 217: Additional entries specific to a set-OCG-state action" shows the action dictionary entries specific to this type of action.

Table 217: Additional entries specific to a set-OCG-state action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *SetOCGState* for a set-OCG-state action.
State	array	*(Required)* An array consisting of any number of sequences beginning with a name object (*ON*, *OFF*, or *Toggle*) followed by one or more optional content group dictionaries. The array elements shall be processed from left to right; each name shall be applied to the subsequent groups until the next name is encountered: *ON* sets the state of subsequent groups to *ON*. *OFF* sets the state of subsequent groups to *OFF.* *Toggle* reverses the state of subsequent groups.
PreserveRB	boolean	(Optional) If *true*, indicates that radio-button state relationships between optional content groups (as specified by the **RBGroups** entry in the current configuration dictionary; see "Table 99: Entries in an optional content configuration dictionary") should be preserved when the states in the **State** array are applied. That is, if a group is set to *ON* (either by *ON* or *Toggle*) during processing of the **State** array, any other groups belonging to the same radio-button group shall be turned *OFF*. If a group is set to *OFF*, there is no effect on other groups. If **PreserveRB** is *false*, radio-button state relationships, if any, shall be ignored. Default value: *true*.

When a set-OCG-state action is performed, the **State** array shall be processed from left to right. Each name shall be applied to subsequent groups in the array until the next name is encountered, as shown in the following example.

EXAMPLE 1

```
<</S /SetOCGState
   /State [/OFF 2 0 R 3 0 R /Toggle 16 0 R 19 0 R /ON 5 0 R]
>>
```

A group may appear more than once in the **State** array; its state shall be set each time it is encountered, based on the most recent name. *ON*, *OFF* and *Toggle* sequences have no required order. More than one sequence in the array may contain the same name.

EXAMPLE 2 If the array contained [/OFF 1 0 R /Toggle 1 0 R], the group's state would be *ON* after the action was performed.

NOTE While the specification allows a group to appear more than once in the State array, this is not intended to implement animation or any other sequential drawing operations. PDF processing applications are free to accumulate all state changes and apply only the net changes simultaneously to all affected groups before redrawing.

12.6.4.14 Rendition actions

A *rendition action* (*PDF 1.5*) controls the playing of multimedia content (see 13, "Multimedia features"). This action may be used in the following ways:

- To begin the playing of a rendition object (see 13.2.3, "Renditions"), associating it with a screen annotation (see 12.5.6.18, "Screen annotations"). The screen annotation specifies where the rendition shall be played unless otherwise specified.
- To stop, pause, or resume a playing rendition.
- To trigger the execution of a ECMAScript script that may perform custom operations.

"Table 218: Additional entries specific to a rendition action" lists the entries in a rendition action dictionary.

Table 218: Additional entries specific to a rendition action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *Rendition* for a rendition action.
R	dictionary	(*Required when **OP** is present with a value of 0 or 4; otherwise optional*) A rendition object (see 13.2.3, "Renditions").
AN	dictionary	(*Required if **OP** is present with a value of 0, 1, 2, 3 or 4; otherwise optional*) An indirect reference to a screen annotation (see 12.5.6.18, "Screen annotations").
OP	integer	(*Required if **JS** is not present; otherwise optional*) The operation to perform when the action is triggered. Valid values shall be: 0 If no rendition is associated with the annotation specified by **AN**, play the rendition specified by **R**, associating it with the annotation. If a rendition is already associated with the annotation, it shall be stopped, and the new rendition shall be associated with the annotation. 1 Stop any rendition being played in association with the annotation specified by **AN**, and remove the association. If no rendition is being played, there is no effect. 2 Pause any rendition being played in association with the annotation specified b *y* **AN**. If no rendition is being played, there is no effect. 3 Resume any rendition being played in association with the annotation specified by **AN**. If no rendition is being played or the rendition is not paused, there is no effect. 4 Play the rendition specified by **R**, associating it with the annotation specified by **AN**. If a rendition is already associated with the annotation, resume the rendition if it is paused; otherwise, do nothing.
JS	text string or stream	(*Required if* OP *is not present; otherwise optional*) A text string or stream containing a ECMAScript script that shall be executed when the action is triggered.

Either the **JS** entry or the **OP** entry shall be present. If both are present, **OP** is considered a fallback that shall be executed if the interactive PDF processor is unable to execute ECMAScripts. If **OP** has an unrecognised value and there is no **JS** entry, the action is invalid.

In some situations, a pause (**OP** value of 2) or resume (**OP** value of 3) operation may not make sense or the player may not support it. In such cases, the user should be notified of the failure to perform the operation.

EXAMPLE A JPEG image

Before a rendition action is executed, the interactive PDF processor shall make sure that the **P** entry of

the screen annotation dictionary references a valid page object and that the annotation is present in the page object's **Annots** array (see "Table 31: Entries in a page object").

A rendition may play in the rectangle occupied by a screen annotation, even if the annotation itself is not visible; for example, if its *Hidden* or *NoView* flags (see "Table 167: Annotation flags") are set. If a screen annotation is not visible because its location on the page is not being displayed by the viewer, the rendition is not visible. However, it may become visible if the view changes, such as by scrolling.

12.6.4.15 Transition actions

A *transition action* (*PDF 1.5*) may be used to control drawing during a sequence of actions. As discussed in 12.6.2, "Action dictionaries" the **Next** entry in an action dictionary may specify a sequence of actions. interactive PDF processors shall normally suspend drawing when such a sequence begins and resume drawing when it ends. If a transition action is present during a sequence, the interactive PDF processor shall render the state of the page viewing area as it exists after completion of the previous action and display it using a transition specified in the action dictionary (see "Table 219: Additional entries specific to a transition action"). Once this transition completes, drawing shall be suspended again.

Table 219: Additional entries specific to a transition action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *Trans* for a transition action.
Trans	dictionary	*(Required)* The transition to use for the update of the display (see "Table 164: Entries in a transition dictionary").

12.6.4.16 Go-To-3D-View actions

A *go-to-3D-view action* (*PDF 1.6*) identifies a 3D annotation and specifies a view for the annotation to use (see 13.6, "3D Artwork"). "Table 220: Additional entries specific to a go-to-3D-view action" shows the entries in a go-to-3D-view action dictionary.

Table 220: Additional entries specific to a go-to-3D-view action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *GoTo3DView* for a transition action.
TA	dictionary	*(Required)* The target annotation for which to set the view.

Key	Type	Value
V	(various)	*(Required)* The view to use. It may be one of the following types: • A 3D view dictionary (see 13.6.4, "3D views"). • An integer specifying an index into the **VA** array in the 3D stream (see "Table 311: Entries in a 3D stream dictionary"). • A text string matching the **IN** entry in one of the views in the **VA** array (see "Table 315: Entries in a 3D view dictionary"). • A name that indicates the first (**F**), last (**L**), next (**N**), previous (**P**), or default (**D**) entries in the **VA** array; see discussion following this Table.

The **V** entry selects the view to apply to the annotation specified by TA. This view may be one of the predefined views specified by the **VA** entry of the 3D stream (see "Table 311: Entries in a 3D stream dictionary") or a unique view specified here.

If the predefined view is specified by the names **N** (next) or **P** (previous), it should be interpreted in the following way:

- When the last view applied was specified by means of the **VA** array, **N** and **P** indicate the next and previous entries, respectively, in the **VA** array (wrapping around if necessary).
- When the last view was not specified by means of **VA**, using **N** or **P** should result in reverting to the default view.

12.6.4.17 ECMAScript actions

Upon invocation of an ECMAScript action, a PDF processor shall execute a script that is written in the ECMAScript programming language. That script may include the extensions described in ISO 22537:2006. Depending on the nature of the script, various interactive form fields in the document may update their values or change their visual appearances. ISO/IEC 16262:2011 and the *Adobe JavaScript for Acrobat API Reference* give details on the contents and effects of ECMAScript scripts. "Table 221: Additional entries specific to an ECMAScript action" shows the action dictionary entries specific to this type of action.

Table 221: Additional entries specific to an ECMAScript action

Key	Type	Value
S	name	*(Required)* The type of action that this dictionary describes; shall be *JavaScript* for an ECMAScript action.
JS	text string or text stream	*(Required)* A text string or text stream containing the ECMAScript script to be executed.

To support the use of parameterized function calls in ECMAScript scripts, the *JavaScript* entry in a PDF document's name dictionary (see 7.7.4, "Name dictionary") may contain a name tree that maps name strings to document-level ECMAScript actions. When the document is opened, all of the actions in this name tree shall be executed, defining ECMAScript functions for use by other scripts in the document.

NOTE The name strings associated with individual ECMAScript actions in the name dictionary serve merely as a convenient means for organizing and packaging scripts. The names are arbitrary and need not bear any relation to the ECMAScript name space.

12.6.4.18 Rich-Media-Execute actions

A *rich-media-execute action* identifies a rich media annotation and specifies a command to be sent to that annotation's handler (see Clause 13.7, "Rich media". "Table 222: Additional entries specific to a rich-media-execute action" shows the entries in a rich-media-execute action dictionary.

Table 222: Additional entries specific to a rich-media-execute action

Key	Type	Value
S	name	*(Required; PDF 2.0)* The type of action that this dictionary describes; shall be *RichMediaExecute* for a rich-media-execute action.
TA	dictionary	*(Required; PDF 2.0)* An indirect object reference to an annotation dictionary (of Subtype *RichMedia*) upon which to execute the command.
TI	dictionary	*(Optional; PDF 2.0)* A dictionary that shall be an indirect object reference to a *RichMediaInstance dictionary* that is also present in the Instances array of the annotation.
CMD	dictionary	*(Required; PDF 2.0)* A *RichMediaCommand dictionary* containing the command name and arguments to be executed when the rich-media-execute action is invoked. See "Table 223: Entries in a RichMediaCommand dictionary".

The RichMediaCommand dictionary contains a command name and optional arguments to be passed to the annotation handler specific to the target instance specified by the **TI** key in the parent rich-media-execute action dictionary.

Table 223: Entries in a RichMediaCommand dictionary

Key	Type	Value
Type	name	*(Optional; PDF 2.0)* The type of PDF object that this dictionary describes; shall be *RichMediaCommand*.
C	text string	*(Required; PDF 2.0)* A text string specifying the script command (a primitive ECMAScript function name). If the target instance is a 3D model, the call shall be made in the global context of the annotation's instance of the 3D ECMAScript engine.
A	various	*(Optional; PDF 2.0)* An object that specifies the arguments to the command. The object may either be a single typed value or an array of typed values, each an argument. Valid arguments are objects of type text string, integer, real, or boolean. Default value: no arguments.