



Destroying Form Nodes

Note

Before using this information and the product it supports, read the information in "Notices," on page 7.

First Edition (September 2006)

This edition applies to version 1, release 2.6.1 of Workplace Forms and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces version 1, release 2.6 of Workplace Forms.

© Copyright International Business Machines Corporation 2003, 2006. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Destroying Form Nodes	1
Why Delete Form Nodes?	1
When is a Node in Use?	1
Deleting Nodes That Are in Use	2
Permitted Cases	2
Forbidden Cases	2
Additional Cases	3
Setting the dirtyflag	3

Deleting Nodes in XForms.	3
Destroy Function Quick Reference	4
Quick Reference Table	5
Appendix. Notices	7
Trademarks.	8

Destroying Form Nodes

This guide provides information on the use of the *destroy* function. The purpose of this function is to enable the dynamic deletion of nodes from an XFDL form, even while the form is being displayed by an application.

Although some background information is provided, this guide assumes that you have working knowledge of the Extensible Forms Description Language (XFDL) and the XFDL compute system (including function calls). If you are new to XFDL, refer to the *XFDL Specification* for more information.

You should also have some understanding of how the components of an XFDL form are organized into a hierarchy of interrelated nodes. Refer to the *Workplace Forms Server — API User's Manual* for more information about the node structure.

More experienced users may wish to skip ahead to the "Destroy Function Quick Reference" on page 4 section.

Why Delete Form Nodes?

Although *destroy* can be used to delete many different types of nodes, this function is primarily intended to enable the dynamic removal of specific graphical user interface items such as buttons, fields, and labels. This powerful capability enables the developer to create forms that are:

- Faster to fill out.
- Less likely to be filled out incorrectly.
- More compact.
- Faster to save, print, and transmit.

You can use the *destroy* function to remove fields that are no longer relevant, based on information entered elsewhere on the form. In effect, this permits you to dynamically customize the form for the user. This reduces the possibility of users entering inaccurate or unnecessary information and makes filling out the form more enjoyable.

For example, a form could include a check box indicating whether a person's mailing address is the same as the home address. When the user selects this check box, the *destroy* function could be called to dynamically delete all items pertaining to the mailing address.

When is a Node in Use?

In general, a node is only considered to be "in use" when it is currently displayed by an application, such as the Viewer. However, a page that is not on display may contain nodes that are being referenced by a page that is on display. In that case, the first page is said to be in "partial" use, and the second page is regarded as being in "full" use.

For example, PAGE1, which is not currently on display, contains a list item and several cells. PAGE2, which is being displayed in the Viewer, contains a popup item that is populated with the cells from PAGE1. In this case, PAGE1 is in partial use, and PAGE2 is in full use.

This important distinction applies only to page nodes. In the section called “Permitted Cases,” you will see that a toolbar can be deleted from a page that is in partial use, but not from one that is in full use. The page node itself can only be destroyed if it is neither in full nor partial use. In other words, a page can only be deleted if it is not currently being displayed and no other page that refers to it is being displayed.

Deleting Nodes That Are in Use

It is possible to destroy a node from almost any level of the form structure. However, this is usually only true if the node in question is not in use at the time. For example, you are not allowed to destroy a page node that is currently on display in an application, because it is in use.

In keeping with the intended purpose of the function, you are allowed to destroy certain types of nodes even when they are in use. In particular, nodes that contain graphical user interface items may be destroyed while being displayed by an application.

To see when a particular type of node may be destroyed, see the Quick Reference Table.

Permitted Cases

The default behavior of the *destroy* function only allows the destruction of nodes that are not in use. However, as stated previously, there are several exceptions to this rule. In particular, the following nodes can be destroyed even when they are in use:

- The graphical user interface items: box, button, check, combobox, field, label, line, list, popup, radio, and spacer.
- Cell items.
- Data items.
- Toolbar items (provided the page is only in partial use).
- Any portion of an XML data model.
- Any elements in a non-XFDL namespace.
- Custom Options.
- Options on inactive pages.

Forbidden Cases

The ability to dynamically destroy the elements of a form provides the developer with a great deal of control. However, the destruction of certain types of nodes could produce undesirable results. For this reason, you should not attempt to destroy:

- The form itself.
- The form global or any of its descendants (excluding the XML data model).
- Any page that is currently in full or partial use.
- The page global (or any descendants) of a page that is in full or partial use.
- A toolbar item belonging to a page that is in full use.
- An action item belonging to a page that is in full use.
- A help item.
- An array element.

- Any kind of option, including:
 - An event option such as *focused*, *activated*, *mouseover*, or *keypress*.
 - A relative option such as *itemfirst*, *itemlast*, *itemprevious*, *itemnext*, *pagefirst*, *pagelast*, *pageprevious*, or *pagenext*.

Attempting to delete any of these nodes causes the form to display an error message.

Additional Cases

There are two other specific situations in which the *destroy* function cannot be used. First, an item cannot destroy itself. In other words, the call to the *destroy* function should not occur within the code for the item you wish to destroy.

The second situation is somewhat more complicated. Consider a form containing a field item and a button. Within the XFDL code for the field item there is a *destroy* function that can be activated (or toggled) by the button. In this case, you would not be allowed to use the field's *destroy* function to delete the button. In other words, you are not permitted to destroy an item that activates the *destroy* function.

Notes

- Although you cannot specifically destroy a relative option such as *pagenext*, it may be possible to delete the entire node in which it occurs. For more information about relative options, refer to the *XFDL Specification*.
- For security reasons, you should not use *destroy* to delete signature items. The Workplace Forms™ API provides a specific method, called *UFLDeleteSignature*, for this purpose. For more information on this method, refer to the Java™, C, or COM Editions of the *IBM Workplace Forms API User's Manual*.

Setting the dirtyflag

Workplace Forms Viewer and other form-viewing applications are alerted to changes in a form by the status of the *dirtyflag*. For example, when the *dirtyflag* is **on**, the application is aware that some aspect of the form has been changed. Consequently, when the user closes the form, the Viewer displays a save prompt.

Deleting nodes with the *destroy* function does not automatically set the *dirtyflag* to **on**. This must be done manually by using the *set* function.

To set *dirtyflag* to **on**, insert the following line in your XFDL code where a *destroy* function is called:

```
set('global.global.dirtyflag', 'on')
```

Now the Viewer will be aware that a change to the form has taken place and will automatically display a save prompt when the form is closed.

Note: For more information on the *dirtyflag*, refer to the *XFDL Specification*.

Deleting Nodes in XForms

You cannot use the *destroy* function to delete item or option nodes that are bound to the XForms model. To ensure that bound nodes are correctly deleted, you must destroy the entire page node.

Destroy Function Quick Reference

The *destroy* function destroys (or deletes) a specified node from a form. Although there may be other uses, it is intended primarily to enable the deletion of graphical user interface items such as boxes, buttons, checks, comboboxes, fields, labels, lines, lists, popups, radios, and spacers. Under certain circumstances, it can be used to destroy entire pages.

Call

```
destroy("theReference", "theNodeType")
```

Parameters

Expression	Type	Description
<i>theReference</i>	String	A string that contains the reference to the node you wish to destroy. For instance, if you want to destroy FIELD1 on PAGE2, <i>theReference</i> would be PAGE2.FIELD1.
<i>theNodeType</i>	String	A string that contains the type of node identified in <i>theReference</i> . Valid values are form , page , item , or option . For instance, if <i>theReference</i> specifies a field, <i>theNodeType</i> would be item

Returns

Nothing if the operation was successful. An error message is displayed if the specified node cannot be destroyed.

Notes

- In general, a node that is being displayed in the Viewer is considered to be in use.
- The default behavior of *destroy* is to only allow the destruction of nodes that are not in use, although there are several exceptions. In particular, graphical user interface items may be deleted while they are being displayed in the Viewer.
- You are normally permitted to destroy any page that is not currently displayed in the Viewer. However, a page that is not in use may contain items that are being referenced by a page that is in use. In this case, neither page can be destroyed. Refer to the section called "When is a Node in Use?" for more information.
- The status of the *dirtyflag* must be manually set to "on" when the form is modified by the *destroy* function.
- If you destroy an entire page, the bindings to the XForms data model are automatically removed. However, this update does not occur if you only destroy an item or option within the page. This means that destroying items or options that are bound to the data model may cause your form to behave erratically.

Example

The following code creates a button that, when clicked by the user, deletes a field called mailStreet_FIELD from PAGE2 of the form. The *dirtyflag* is set to "on" to indicate that the form has changed

```
<button sid="deleteField_BUTTON">  
  <itemlocation>  
    <x>168</x>
```



```

        <y>273</y>
    </itemlocation>
    <value>Delete</value>
    <size>
        <width>6</width>
        <height>1</height>
    </size>
    <custom:destroyOpt xfdl:compute="toggle(activated, 'off', &#xA;
        'on') == '1' ? destroy('PAGE2.mailStreet_FIELD', 'item') &#xA;
        + set('global.global.dirtyflag', 'on') : ''"></custom:destroyOpt>
</button>

```

Quick Reference Table

x indicates that the specified node can be deleted with the *destroy* function.

	page is in full use	page is in partial use	page is not in use
action		x	x
array elements			
box	x	x	x
button	x	x	x
cell	x	x	x
check	x	x	x
combobox	x	x	x
data	x	x	x
field	x	x	x
form global (globalpage)			
form			
help			
label	x		x
line	x	x	x
list	x	x	x
non-XFDL namespace	x	x	x
options*			
page			x
page global			x
popup	x	x	x
radio	x	x	x
signature			
spacer	x	x	x
toolbar		x	x
XML data model	x	x	x

* options include: standard options, event options, and relative options

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Office 4360
One Rogers Street
Cambridge, MA 02142
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

AIX
IBM
Workplace
Workplace Forms

Other company, product, or service names may be trademarks or service marks of others.



Program Number:

Printed in USA

S325-2605-00

