IBM® Workplace Forms™

Version 2.6.1

The XFDL Event Model

S325-2602-00

Note

Before using this information and the product it supports, read the information in “Notices,” on page 13.

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.

Contents

The XFDL Event Model

This document explains how to use the XFDL Event Model, which relies on the following XFDL options: *activated*, *dirtyflag*, *focused*, *focuseditem*, *keypress*, *mouseover*, and *printing*.

This document explains how to work with each option, and provides instructions for designing forms that use the Event Model, either by writing XFDL code in a text editor or by designing forms in Workplace Forms™ Designer. Before reading this document, you should be familiar with both the XFDL language and Workplace Forms Designer.

What is the XFDL Event Model?

The XFDL Event Model tracks user actions, such as mouse movements and keyboard input. These actions are called *events*, and you can design your forms so that these events are captured and trigger specific responses. For example, using the Event Model you could:

- Create a button that changes color when the user moves the mouse over it.
- Create a form that submits when the user types F2 on the keyboard.
- Create a ″help″ label that displays a different message depending on which field the user is filling out.
- Create links to websites or other documents that change color once they have been clicked.

About the Event Model Options

The Event Model relies on the following XFDL options:

- **activated** — Detects whether or not an item, page, or form has been activated by the user. For example, a button becomes activated when the user clicks it.
- **dirtyflag** — Records whether the form has been updated since the last save or submission.
- **focused** — Detects whether an item, page, or form currently has the input focus. For example, a field has the focus when the cursor is in it.
- **focuseditem** — Specifies which item in the page currently has the focus.
- **keypress** — Stores the last keystroke made by the user.
- **mouseover** — Detects whether the mouse pointer is currently positioned over an item or page.
- **printing** — Indicates whether the form is currently printing.

These are special options in the XFDL language that are never written out in XFDL forms. Instead, these options are created in memory only, and can be thought of as *virtual options* — they never appear in the XFDL text for a form, but are always created and maintained when the form is run.

This means that you never have to add these options to your form. You can simply assume that they will exist when the form is in use. Furthermore, each option is automatically included in every item that supports it. For example, every button item will include an *activated* option.

Using activated

The *activated* option specifies whether an item is active, and stores one of the following values:

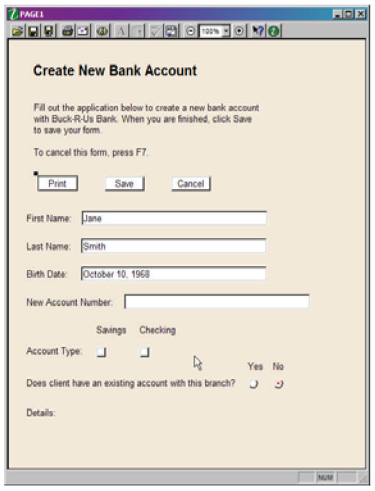
- **on** — The item is active. For example, a button is active when the user clicks and releases it.
- **maybe** — The item is in an undetermined state. This state is specific to buttons, and occurs when the button has been clicked but has not yet been released.
- **off** — The item is not active. For example, an action that is not firing is not active.

The following tables lists the items that support the activated option, and provides more information about when status of the *activated* option changes:

Item	When *activated* status changes
action	*activated* changes to **on** when the action fires, and back to **off** when the action is complete.
button	*activated* changes to **on** when the button is clicked, switches to **maybe** while the button is held down, and changes back to **off** when the button is released.
cell	*activated* changes to **on** when the cell's action fires, and back to **off** when the cell's action is complete.
combobox	*activated* changes to **on** when the list of choices is displayed, and back to **off** when the list is closed.
popup	*activated* changes to **on** when the list of choices is displayed, and back to **off** when the list is closed.
page global	*activated* changes to **on** when the page is displayed, and back to **off** when a different page is displayed or when the form is closed.
form global	*activated* changes to **on** when the form is open, and back to **off** when the form is closed.

Example

The following diagram shows how an activated Save button changes color (from white to grey) while the save action occurs. The button's color returns to white when the save is complete.



Create New Bank Account

Fill out the application below to create a new bank account with Buck-R-Us Bank. When you are finished, click Save to save your form.

To cancel this form, press F7.

First Name: Jane

Last Name: Smith

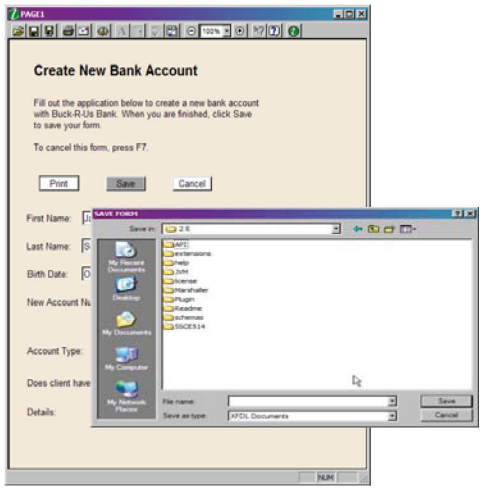
Birth Date: October 10, 1968

New Account Number:

Account Type:

Does client have an existing account with this branch? Yes [✓ No]

Details:



Create New Bank Account

Fill out the application below to create a new bank account with Buck-R-Us Bank. When you are finished, click Save to save your form.

To cancel this form, press F7.

First Name:

Last Name:

Birth Date:

New Account Nu

Account Type:

Does client have

Details:

Save in: 2.6

File name:

Save as type: XFDL Documents

Working with activated in the Designer

To change the background color of a button when it is activated, you must edit the XFDL source code for the button. This involves inserting an If/Then/Else statement into the bgcolor option of the button using the following logic:

IF activated is set to **on**, THEN the color of the button is *grey*. Otherwise (ELSE), it is *white*.

To set a button to change color when activated:

1. Open the form in the Designer and select the button you want to modify with the *activated* option.
2. Select **Source**.
   - The **Code View** window appears.
3. In theCode View window, edit the *bgcolor* option of the button so that it changes color based on the value of activated.

   For example:

   ```
   <button sid="saveButton">
      <type>saveform</type>
      <value>Save</value>
      <activated>off</activated>
      <bgcolor compute="activated=='on' ? ('white') : 'gray'"></bgcolor>
   </button>
   ```
4. Click **OK** when you are done editing.

Note: Because the *activated* option is created and destroyed by the form, you do not have to create this option to refer to it in the compute for the Save button. For more details about the activated option, refer to the XFDL Specification.

Using dirtyflag

The *dirtyflag* option records whether the form has been updated since the last save or submission, and stores one of the following values:

- **on** — The user made a change to the form.
- **off** — The user had not changed the form since it was last saved or submitted.

The *dirtyflag* option is valid for the *form global* only.

Note that *dirtyflag* is not set by computed changes to the form. For example, if the user clicks a button that triggers a compute, and that compute copies information to a field in the form, the *dirtyflag* would not be set. In these cases, the form should include additional computes that set the *dirtyflag*.

If necessary, the save prompt can be disabled by using a compute to set the *dirtyflag* to **off**.

This option is not saved or transmitted as part of the form. Instead, it is automatically created each time the form is read into memory, and is maintained only during display or processing.

Example

In the Create New Bank Account form, clicking the Save button activates a special function that bypasses the usual Save As dialog box and saves the form to a specific directory on the bank system. However, because the usual method of saving was bypassed, the Viewer will still prompt users to save when they close the form unless *dirtyflag* is turned off.

Working with dirtyflag in the Designer

To ensure that the Viewer does not prompt the user to save after using a custom function to save the form, you must set the dirtyflag event to off. To do this, you must complete the following steps:

- Create a custom option in the Save button.
- Create an If/Then/Else statement which specifies that when the function has been activated, it should trigger the custom save function and set the *dirtyflag* option to off. For example, to trigger the custom save function and turn off the dirtyflag event, you would use the following logic:

  IF the *toggle* function is activated (or the Save button is pressed), THEN the *MySaveFunction* function is triggered AND the *dirtyflag* option is set to off. Otherwise (ELSE), no action is specified.
- Use the *toggle* function to trigger the custom save function.
- Use your custom function to perform the save.
- Use the *set* function to set turn off the dirtyflag event.

To do this:

1. Open the form in the Designer and select the Save button.
2. Select **Source**.
   - The **Code View** window appears.
3. In the Code View window, create the custom option and its conditional statement. For example:

```
<button sid="ApplicationSpecificSave">
   <value>Save</value>
   <type>cancel</type>
   <custom:save compute="toggle(activated, 'off', 'on')== '1' &#xA;
      ? (MySaveFunction() + set('global.global.dirtyflag', &#xA;
      'off')) : '' "></custom:save>
</button>
```

Note: Because the *dirtyflag* option is created and destroyed by the form, you do not have to include it in your XFDL code to use it in the compute. For more details about the toggle and set functions, refer to the XFDL Specification.

Using focused

The *focused* option specifies which type of form object currently has the input focus, and stores one of the following values:

- **on** — The item, page, or form has the input focus.
- **off** — The item, page, or form does not have the input focus.

The *focused* option is valid for the following items:

- button
- field
- radio

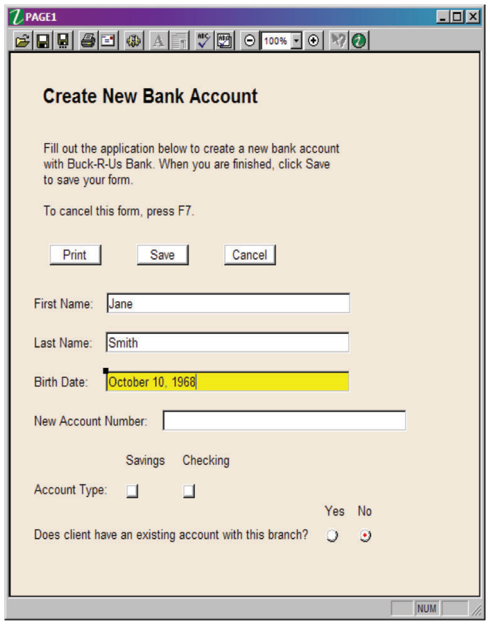
- check
- combobox
- list
- popup
- page global
- form global

An object's *focused* option is set to **on** when that item, page, or form receives the input focus, and is set to **off** when the focus moves to another item, page or form. However, in objects that are hierarchical, it is possible for more than one object to have the focus at one time. For example, a form, a page, and a field can all be focused at the same time.

Note: The item that has the focus does not change when the Viewer window is minimized or when the user is working in another window.

Example

The following diagram shows how the background color of an item changes (from white to yellow) depending on whether it has the focus. The item's color returns to white when the focus moves to a new item.



Create New Bank Account

Fill out the application below to create a new bank account with Buck-R-Us Bank. When you are finished, click Save to save your form.

To cancel this form, press F7.

First Name: Jane

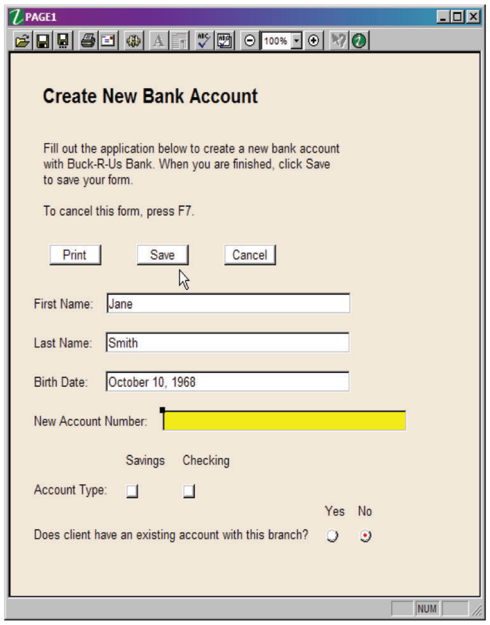
Last Name: Smith

Birth Date: October 10, 1968

New Account Number:

Account Type: Savings ☐ Checking ☐

Does client have an existing account with this branch? Yes [✓ No]



Create New Bank Account

Fill out the application below to create a new bank account with Buck-R-Us Bank. When you are finished, click Save to save your form.

To cancel this form, press F7.

First Name: Jane

Last Name: Smith

Birth Date: October 10, 1968

New Account Number:

Account Type: Savings ☐ Checking ☐

Does client have an existing account with this branch? Yes [✓ No]

When the Birth Date field has the input focus, the *focused* options are set to:

- **on** for the Birth Date field.
- **off** for every other item in the form.
- **on** for the page that contains the Birth Date field.
- **on** for the form (the *form global* option) as it contains the Birth Date field.

Working with focused in the Designer

To set the value of another item in the form based on the *focused* option, you must:

- Create an If/Then/Else statement which specifies that when the *focused* option is ″on″ its background color should change accordingly. For example, to set the Birth Date field background color to change when it has the focus, you would use the following logic:

  IF the Birth Date field's *focused* option is set to ″on″, THEN its bgcolor is yellow. Otherwise (ELSE), the bgcolor is white.

To create a compute that uses the *focused* option to change the item's background color:

1. Open the form in the Designer and select the item that will have its content changed depending upon the position of the input focus.
2. Select **Source**.
   - The **Code View** window appears.
3. In the Code View window, create the compute in the *bgcolor* option. For example:

```
<bgcolor compute="Birth_Date.focused == 'on' ? ('yellow') &#xA;
       : 'white'"></bgcolor>
```

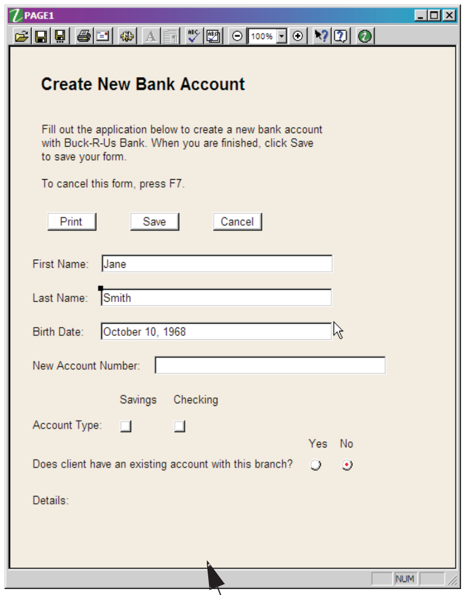
Note: Because the *focused* option is created and destroyed by the form, you do not have to include it in your XFDL code to use it in the compute. For more details on the focused option, refer to the XFDL Specification.

Using focuseditem

The *focuseditem* option specifies which item in the page currently has the input focus and stores its sid. It is valid for the page global only.

Example

The following shows how the dynamic Help messages at the bottom of the form changes depending upon the part of the form that has the focus. In diagram A, the focus is on the page, and there is no Help message. In diagram B, the focus is on the Birth Date field, and the Help message contains instructions for completing the field. When a user clicks the Birth Date input field, the input focus shifts to that field.



Create New Bank Account

Fill out the application below to create a new bank account with Buck-R-Us Bank. When you are finished, click Save to save your form.

To cancel this form, press F7.

First Name: Jane

Last Name: Smith

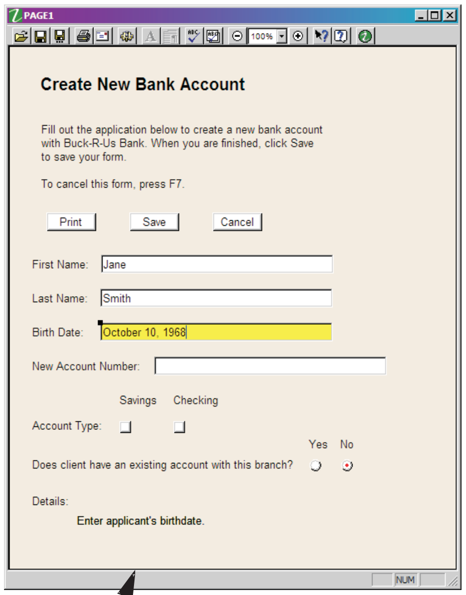
Birth Date: October 10, 1968

New Account Number:

Account Type: Savings Checking

Does client have an existing account with this branch? Yes [✓ No]

Details:



Create New Bank Account

Fill out the application below to create a new bank account with Buck-R-Us Bank. When you are finished, click Save to save your form.

To cancel this form, press F7.

First Name: Jane

Last Name: Smith

Birth Date: October 10, 1968

New Account Number:

Account Type: Savings Checking

Does client have an existing account with this branch? Yes [✓ No]

Details:

Enter applicant's birthdate.

This message changes dynamically, depending on the location of the input focus.

Working with focuseditem in the Designer

To create a *focuseditem* compute:

1. Open the form in the Designer and select the item that will display the form's help messages.
2. Select **Source**.
   - The **Code View** window appears.
3. In theCode View window, create the compute and its dereference. For example:

```
<value compute="global.focuseditem->help->value"></value>
```

Note: Because the focuseditem option is automatically created and destroyed by the form, you do not have to include this option in the form before referring to it in the compute. For more details about the focuseditem option, dereferences, or computes, refer to the XFDL Specification.

Using keypress

The *keypress* option stores the last keystroke made by the user, and is valid for the following items:

• button	• field	• radio
• check	• list	• page global
• combobox	• popup	• form global

For the form global, *keypress* stores the last keystroke made for any valid items on the form. For the page global, *keypress* stores the last keystroke made for any valid item on that page.

The *keypress* option is most useful for creating shortcut keys in your forms. Such keys make it faster for users to access frequently used features. For example, you might configure your form so that the F2 key submits it.

The *keypress* option only stores keystrokes that are invalid input for the current item. For example, if the focus were on a field and the user typed ″a″, an ″a″ would appear in the field and the *keypress* option would remain empty because ″a″ is valid input for a field. However, if the focus were on a radio button and the user typed ″a″, the *keypress* option would be set to ″a″ because the radio button cannot accept that character as input.

Once the *keypress* option is set for an item, that value trickles up to the page global and form global so that it can be accessed at any level of the form.

Example

In the Create New Bank Account form, users can press the F7 key at any time to cancel the form. F7 is a shortcut key, which activates the Cancel button when pressed. This occurs regardless of where the focus is at the time the user presses F7.

Working with keypress in the Designer

If you are using the Designer, you can only create shortcut keys for form items that are buttons. To create these shortcut keys:
- Select the button that will be the default action item in the form.
- Edit the source code for the default button to specify the desired shortcut key.

To do this:

1. Open the form in the Designer and select the button that you want to use as the default action item in the form. This is typically a button that performs an action such as printing, saving, or canceling the form.
2. Select **Source**.
3. In the Code View window, edit the global option *vfd_default* so that the desired shortcut key is attached to the default button.
   - The **Code View** window appears.
   - For example, if you want to cancel the form using the F7 key, then use F7 with the *toggle* and *set* functions:

     ```
     <vfd_default compute="toggle(keypress, '', 'F7') =='1' &#xA;
        ? (set('cancelButton.activated', 'on')) : ''"></vfd_default>
     ```

Note: The Designer only allows you to attach shortcut keys to buttons that are declared as default items. If you want to attach shortcut keys to other items, refer to the following section, called ″Working with keypress in a text editor″.

Using Mouseover

The *mouseover* option recognizes whether the mouse pointer is currently positioned over the item, and stores one of the following values:

- **on** — The mouse is over the item.
- **off —** The mouse is not over the item.

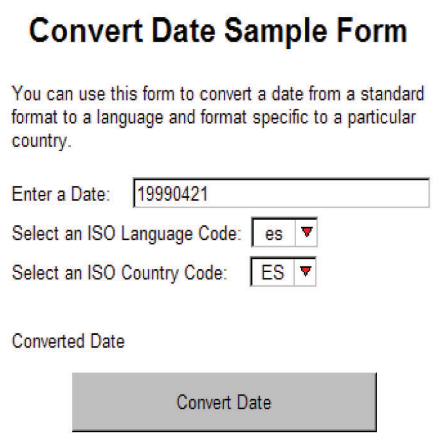
The *mouseover* option is valid for the following items:

• button	• combobox	• list	• toolbar
• check	• field	• popup	• page global

For the page global, *mouseover* stores whether the mouse is over that page. For example, if you were working with *Page1* of a form, the *mouseover* option for *Page2* would be **off**. Similarly, if you were working with *Page1* of a form, but the mouse was over a different window (such as your email client), the *mouseover* option for *Page1* would be **off**.

Example

The following diagrams show how the *mouseover* option can change the text of a button:



Convert Date Sample Form

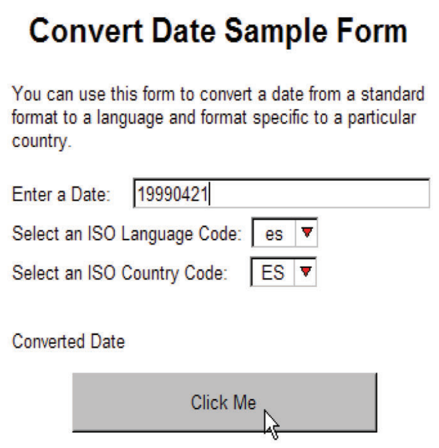
You can use this form to convert a date from a standard format to a language and format specific to a particular country.

Enter a Date: 19990421

Select an ISO Language Code: es

Select an ISO Country Code: ES

Converted Date



Convert Date Sample Form

You can use this form to convert a date from a standard format to a language and format specific to a particular country.

Enter a Date: 19990421

Select an ISO Language Code: es

Select an ISO Country Code: ES

Converted Date

Notice how the caption on the button changes from **Convert Date** to **Click Me**.

In the first diagram, the mouse pointer is positioned over the page, but not over any particular form item. Therefore, the *mouseover* options are set to:
- **off** for every item on the page.
- **on** for the page.

In the second diagram, the mouse pointer is positioned over the **Convert Date** button. In this case the mouseover options are set to:
- **on** for the **Convert Date** button.
- **off** for every other item on the page.
- **on** for the page, since it contains the **Convert Date** button.

Working with mouseover in the Designer

To change the value of a form item (in this case, a button) based on the *mouseover* option, you must insert an If/Then/Else statement into the value option of the button. For example:

IF mouseover is set to ″on″, THEN the button says ″Click Me″. Otherwise (ELSE), it reads ″Convert Date″.

To create a compute that uses the *mouseover* option to change a button's text:
1. Open the form in the Designer and select the button whose caption you want to change dynamically. For example, the **Convert Date** button.
2. Select **Source**.
   - The **Code View** window appears.
3. In theCode View window, create the compute in the *value* option. For example:
   ```
   <button sid="convertDateButton">
      <value compute="mouseover == 'on' ? 'Click Me' &#xA;
         : 'Convert Date'"></value>
      <type>select</type>
   </button>
   ```

Note: Because the mouseover option is automatically created and destroyed by the form, you do not have to include this option in the form before referring to it in the compute. For more details about the mouseover option, If/Then/Else statements, or computes, refer to the XFDL Specification.

Using printing

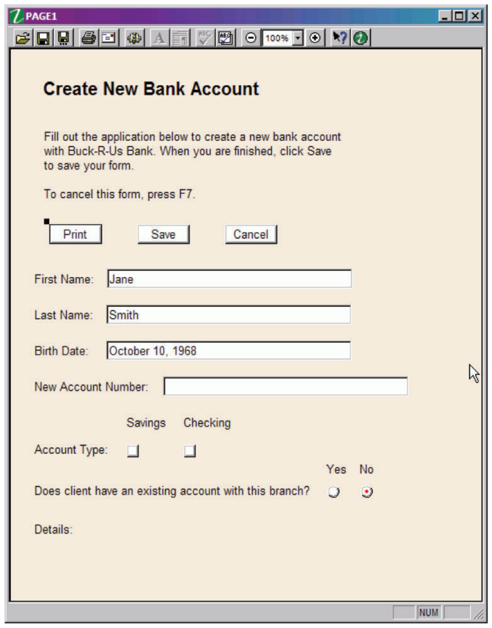
The *printing* option indicates whether the form is currently printing. This value toggles from **off** to **on** just before printing. Any computes that rely on this option are updated before the form prints. This allows you to make computed changes to the form just before it is printed.

The *printing* option is valid for the *form global* only.

Example

As many users do not have a color printer, we would like the background color of the Create New Bank Account form to change to white when users print it. Triggering this change off of the printing event ensures that the form's background color is changed, no matter what method users employ to print the form, be it a short-cut key, the Print button on the toolbar, or the Print button in the form.

The following diagrams show how the *printing* option can trigger a change to the background color of the form:



Create New Bank Account

Fill out the application below to create a new bank account with Buck-R-Us Bank. When you are finished, click Save to save your form.

To cancel this form, press F7.

First Name: Jane

Last Name: Smith

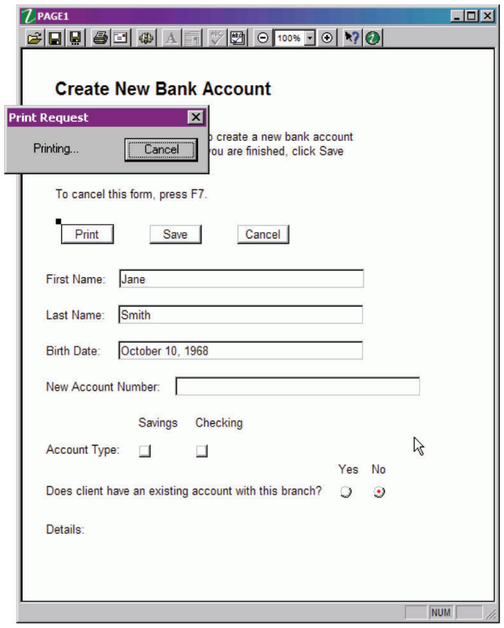
Birth Date: October 10, 1968

New Account Number:

Account Type: Savings ☐ Checking ☐

Does client have an existing account with this branch? Yes [✓ No]

Details:



Create New Bank Account

o create a new bank account
ou are finished, click Save

To cancel this form, press F7.

First Name: Jane

Last Name: Smith

Birth Date: October 10, 1968

New Account Number:

Account Type: Savings ☐ Checking ☐

Does client have an existing account with this branch? Yes [✓ No]

Details:

Working with printing in the Designer

To change the background color of the form while the form is printing, you must:

- Create an If/Then/Else statement which specifies that when the *printing* option is "on" its background color should change accordingly.

For example, to set the form background color while the form is printing, you would use the following logic:

IF the *printing* option is set to "on", THEN its bgcolor is white. Otherwise (ELSE), the bgcolor is navajo white.

To create a compute that uses the *printing* option to change the form's background color:

1. Open the form in the Designer.

2. Select **Source**.
   - The **Code View** window appears.
3. In theCode View window, create the compute in the form global's *bgcolor* option. For example:

```
<bgcolor compute="global.global.printing == 'on' &#xA;
   ? ('white') : 'navajo white'"></bgcolor>
```

Note: Because the printing option is created and destroyed by the form, you do not have to include it in your XFDL code to use it in the compute. For more details about the printing option, refer to the XFDL Specification.

Where to Find More Help

- To find out more about XFDL and related syntax, refer to the *XFDL Specification*.
- To find out more about creating forms using Workplace Forms Designer, refer to the *IBM® Workplace Forms Designer Getting Started Manual* and the Designer Help.

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.

Trademarks

Program Number:

Printed in USA

S325-2602-00