IBM® Workplace Forms™

Version 2.6.1

Setting Up Item Relationships

S325-2607-00

IBM® Workplace Forms™

Version 2.6.1

Setting Up Item Relationships

S325-2607-00

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

Setting Up Item Relationships in XFDL Forms

Item relationships are logical links among groups of items in a form that control the active and mandatory status of each item in the group. For example, a group of fields that accepts phone numbers, only one of which is entirely necessary. These links enable a form designer to require a user to fill in all, some, or specific items within a group of items.

Who Should Read this Document

This document is aimed at form designers who have some experience using Workplace Forms™ Designer or hand-coding forms. Form designers should have some familiarity with XFDL syntax, but need not be proficient with advanced form logic.

How to Use this Document

This document has two sets of instructions for each relationship type: one for form designers who are hand-coding their forms, and one for form designers who are using Workplace Forms Designer to create their forms. Both sets of instructions use the same examples.

There are five types of item relationships:

- **Required —** In a group of items, at least one of the group must be filled in.
- **Exclusion —** In a group of items, only one of the group can be filled in.
- **Paired —** In a group of items, if one item is filled in then all other items in the group must also be filled in.
- **Conditional —** A group of items must all be filled in if another form item is set to a certain value.
- **Conditional Required —** In a group of items, at least one of the group must be filled in if another form item is set to a certain value.

Why Use Item Relationships?

The ability to define relationships amongst groups of items in a form is a valuable tool for designing user-friendly forms. An item can be made mandatory or inactive depending on the status of a related item. Take, for example, an order form requesting item description and quantity. Each field is optional, but when a description is filled in, the related quantity field must also be filled in. This can be ensured by setting the quantity field to become mandatory.

By using relationships among groups of items, you can ensure that you get the information you require and reduce the possibility of receiving incorrectly completed forms. Your users are presented with forms that are easy to understand and easy to complete.

Item relationships are created by using logical operators to set the active option and the mandatory flag in the format option for each item in the relationship. For more information on creating item relationships, see “Creating Item Relationships” on page 3.

Using Logical Operators

Logical operators are used to evaluate mathematical or logical statements. These statements are either true or false, and are used in ″If/Then″ formulas to provide form functionality based on user input or events.

For example, if you wanted FIELD3 to become active when a user selected a certain check box, you could add a compute to the active option of the field. This compute would set the value of active to on if the value of the check box was on.

Logical operators are used to create each statement. The logical operators used in XFDL are:

Operator	Definition	Behavior
==	is equal to	evaluates the statements on both sides of the operator; true if they are equal, false if not. Example: **′5′ == ′11-6′** would evaluate to **true** **′4′ == ′3+2′** would evaluate to **false**
!=	is not equal to	evaluates the statements on both sides of the operator; true if they are not equal, false if they are equal. Example: **′5′ != ′11-4′** would evaluate to **true** **′4′ != ′2+2′** would evaluate to **false**
<	less than	evaluates the statements on both sides of the operator; true if left statement is less than right, otherwise false. Example: **′5′ < ′11-4′** would evaluate to **true** **′4′ < ′2+2′** would evaluate to **false**
>	is greater than	evaluates the statements on both sides of the operator; true if left statement is greater than right, otherwise false. Example: **′5′ > ′6-2′** would evaluate to **true** **′4′ > ′3+2′** would evaluate to **false**
<=	is less than or equal to	evaluates the statements on both sides of the operator; true if left statement is less than or equal to right, otherwise false. Example: **′4′ <= ′2+2′** would evaluate to **true** **′5′ <= ′2+2′** would evaluate to **false**
>=	is greater than or equal to	evaluates the statements on both sides of the operator; true if left statement is greater than or equal to right, otherwise false. Example: **′5′ >= ′6-1′** would evaluate to **true** **′4′ >= ′3+2′** would evaluate to **false**
&& *or* and	logical AND	evaluates the statements on both sides of the operator; true if and only if the statements on both sides of the operator are true. Example: **(′5+3′==′8′) and (′4+2′>′5′)** would evaluate to **true** **(′4+2′==′7′) and (′5+1′<′10′)** would evaluate to **false**

Operator	Definition	Behavior
|| *or* or	logical OR	evaluates the statements on both sides of the operator; true if one or both statements on either side of the operator are true. Example: **(’5+3’==’8’) or (’4+2’>’5’)** would be **true** **(’4+2’==’7’) or (’5+1’<’10’)** would be **true** **(’4-2’>’3’) or (’8+4’==’11’)** would be **false**

Example

A form contains two fields, FIELD1 and FIELD4, and a check box, CHECK2. If a user enters a value greater than 10 in FIELD1, and selects CHECK2, then FIELD4 will be active. If either the value of FIELD1 is less than ten or CHECK2 is not selected, or both, FIELD4 will be inactive.

The following logical statement and XFDL code both describe this If/Then statement.

Statement:

IF the value of FIELD1 **is greater than** ’10’ **and** the value of CHECK2 is equal to **on**, THEN the active option of FIELD4 should be **on**, ELSE it should be **off**.

Code:

```
<field sid="FIELD4">
   <value></value>
   <active compute="FIELD1.value > 10 and CHECK2.value=='on' &#xA;
      ? 'on' : 'off'">   </active>
</field>
```

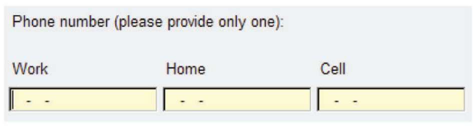
Creating Item Relationships

You can create item relationships by hand-coding computes in the form’s source code, or by adding them with the Designer. When hand-coding forms, you will be adding computes in the item’s *active* option and the **mandatory** element of its *format* option. In the Designer, you will set up formulas for an item’s *format* properties and *active* status. The following sections provide instructions for each of these methods.

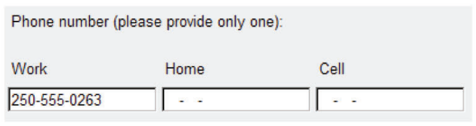
The ‘Required’ Relationship

This relationship forces the user to complete at least one item in a group of items.

For example, a form contains fields for a work phone number, a home phone number, a cellular phone number and a pager number, but only one is required. The section of the form might look like this:


Phone number (please provide only one):

Work	Home	Cell
- -	- -	- -


Phone number (please provide only one):

Work	Home	Cell
250-555-0263	- -	- -

When the form opens, all the phone number fields are mandatory, but if the user fills in one field and then tabs out of it, the remaining three become optional.

This behavior can be expressed through a simple logical statement:

IF none of the fields contain a value, THEN the format is **mandatory**, otherwise (ELSE) it is **optional**.

Setting Up Required Relationships in the Designer

To create a required relationship, you must set a formula for the “required” property for each field.

To do this:

1. Open the form in the Designer.
2. Select the first field (FIELD1 in this example).
3. Select **Source**.
   - The form’s source code appears in the editor window with the focus on the field you selected earlier.
4. Scroll down until you find the first field (FIELD1 in this example) in the form’s source code.
5. Add a *format* option with a compute on the **mandatory** element. Your code should look like this:

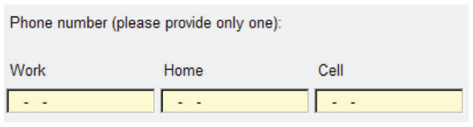
   ```
   <field sid="FIELD1">
         <active>on</active>
         <value></value>
         <format>
            <datatype>string</datatype>
        <constraints>
               <mandatory compute="FIELD2.value=='' and FIELD3.value=='' &#xA;
               ? 'on' : 'off'"></mandatory>
            </constraints>
         </format>
      </field>
   ```

   Note: You may have additional options in your field code.
6. Find the other two fields and add similar format options, adding the compute for the mandatory element to each.

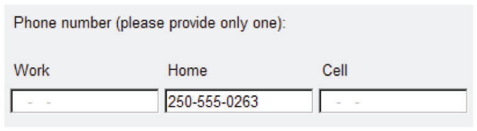
The ‘Exclusion’ Relationship

This relationship requires the user to complete one and only one item from a group of items.

The form in this example contains fields for a work phone number, a home phone number, a cellular phone number, and a pager number. In this case, the user must only complete one of the required items. Once it is filled, the user will not be able to enter data into any of the other fields. The section of the form might look like this:


Phone number (please provide only one):

Work	Home	Cell
- -	- -	- -


Phone number (please provide only one):

Work	Home	Cell
- -	250-555-0263	- -

If the user enters a phone number in one of the fields, the others will be deactivated to prevent the user from entering more data than is desired. All the fields are mandatory initially so that the user will be forced to enter data in one of them.

This behavior can be expressed by a simple logical statement:

IF none of the other fields contain a value, THEN the field's active is **on** and its *mandatory* element is **on**, otherwise (ELSE) the field's active is **off** and its *mandatory* element is **off**.

Setting Up Exclusion Relationships in the Designer

To set up an exclusion relationship in the Designer, you must set a formula for the *active* property and the **required** status for each field.

To do this:

1. Open the form in the Designer.
2. Select the first field (FIELD1 in this example).
3. Select **Source**.
   - The form's source code appears in the editor window with the focus on the field you selected earlier.
4. Add an *active* option with a compute on the **value**. Your code should look like this:

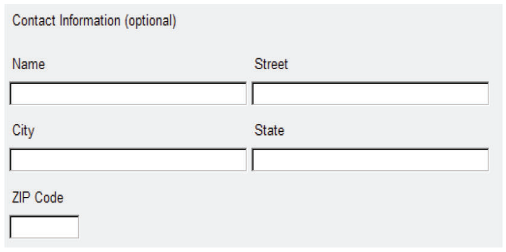
   ```
   <field sid="FIELD1">
      <active compute="FIELD2.value=='' and FIELD3.value=='' and &#xA;
         FIELD4.value=='' ? 'on' : 'off'"></active>
      <value></value>
   </field>
   ```

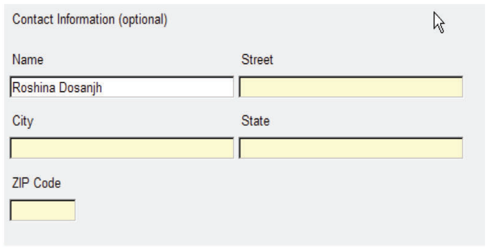
   Note: You may also have other options in your field.
5. Find the other three fields and add similar format options, including the compute on the mandatory element to each.

The 'Paired' Relationship

This relationship requires the user to either complete all the items in a group or to leave all the items blank.

For example, a survey form may have a "contact information" section, which the user may fill in or not. However, if one field in the contact information section is filled in, then all contact information must be filled in. This type of relationship helps form designers ensure that the information entered into the form is complete and useful. The following diagrams illustrate this relationship:





Roshina Dosanjh

The following logical statement can express this behavior:

IF none of the other fields in the group contains a value, THEN the field's format is **optional**, otherwise (ELSE) it is **mandatory**.

Setting Up Paired Relationships in the Designer

To set up a paired relationship in the Designer, you must set a formula for the required property for each field.

To do this:

1. Open the form in the Designer.
2. Select the first field (FIELD1 in this example).
3. Select **Source**.
   - The form's source code appears in the editor window with the focus on the field you selected earlier.
4. Add a *format* option with a compute on the **mandatory** element. Your code should look like this:

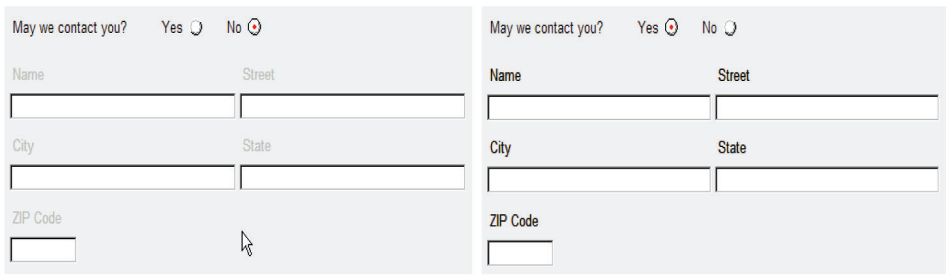
   ```
   <field sid="FIELD1">
      <active>on</active>
      <value></value>
      <format>
         <datatype>string</datatype>
         <mandatory compute="FIELD2.value=='' and FIELD3.value=='' &#xA;
            and    FIELD4.value=='' and FIELD5.value=='' &#xA;
            ? 'off' : 'on'"></mandatory>
      </format>
   </field>
   ```

   Note: You may also have other options in your field.
5. Find the other fields and add similar *format* options, adding the compute for the **mandatory** element to each.

The 'Conditional' Relationship

When this relationship is used, the user is required to complete an item (or group of items) if a specific item outside the group is completed.

For example, a consumer survey might have a radio button that asks if the user can be contacted. If the ″yes″ radio button is selected, fields asking for address information become mandatory. If the "no" radio button is selected, these fields remain inactive and optional. The following diagrams illustrate this relationship:



If the user selects "Yes", the *active* option and the *format* option are set to **on** and **mandatory** respectively for each field in the address group.

This behavior can be expressed through the following logical statement:

IF the appropriate radio button has a value of **on**, THEN the field's active status is **on** and its format is **mandatory**, ELSE the field's active status is **off** and its format is **optional**.

Setting Up Conditional Relationships in the Designer

To set up a conditional relationship in the Designer, you must set a formula for the *active* property and the **required** status for each field.

To do this:

1. Open the form in the Designer.
2. Select the first field (FIELD1 in this example).
3. Select **Source**.
   - The form's source code appears in the editor window with the focus on the field you selected earlier.
4. Add an *active* option with a compute on the value.
   ```
   <field sid="FIELD1">
      <active compute='RADIO1.value'></active>
   </field>
   ```
5. Add a *format* option with a compute on the **mandatory** element:
   ```
   <format>
   <datatype>string</datatype>
   <mandatory compute="FIELD1.active=='on' &#xA;
    ? 'on' : 'off'"></mandatory>
  </format>
   ```

   Note: Because the *format* option is set using the same criteria as the *active* option, you can simply reference the *active* option to set the *format* option and avoid coding the same conditions over again.
6. Find the other fields and add similar *active* and *format* options to each.

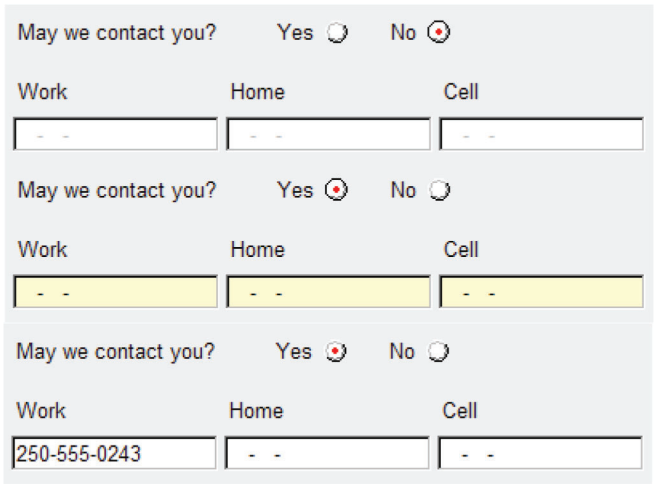
   Note: Remember that the option reference in the compute in the *format* option should be for the current item. For example, for FIELD2, the compute on the **mandatory** element should be:
   ```
   FIELD2.active=='on' ? 'on' : 'off'
   ```

The 'Conditional Required' Relationship

This relationship combines the 'Conditional' relationship and the 'Required' relationship.

As with the 'Conditional' relationship, all of the items in this group are inactive until a specific item outside the group is completed. When the items in the group become active, they behave like items in the 'Required' relationship. That is, at least one of them must be completed.

The example form from the 'Required' relationship contains fields for a work phone number, a home phone number, a cellular number, and pager number. This time, however, all of these fields are inactive unless a radio button indicating that the user wishes to be called is selected. The following diagrams illustrate this relationship:



May we contact you? Yes [✓ No]

Work | Home | Cell

May we contact you? [✓ Yes] No

Work | Home | Cell

May we contact you? [✓ Yes] No

Work: 250-555-0243 | Home | Cell

This behavior can be expressed with the following logical statement:

IF and only if the appropriate radio button is **on** and none of the other fields contain a value, THEN the field's active is **on** and its *mandatory* element is **on**. IF the appropriate radio button is **on** and one or more of the other fields contains a value, THEN the field's active is **on**, and its *mandatory* element is **off**.

Setting Up Conditional Required Relationships in the Designer

To set up a conditional relationship in the Designer, you must set formulas for the *active* property and the **required** status for each field.

To do this:

1. Open the form in the Designer.
2. Select the first field (FIELD1 in this example).
3. Select **Source**.
   - The form's source code appears in the editor window with the focus on the field you selected earlier.
4. Add an active option with a compute on the value.
   ```
   <field sid="FIELD1">
      <active compute="RADIO1.value"></active>
   </field>
   ```
5. Add a *format* option with a compute on the **mandatory** element.
   ```
   <format>
   <datatype>string</datatype>
   <constraints>
    <mandatory compute="active=='on' and FIELD2.value=='' &#xA;
     and  FIELD3.value=='' ? 'off' : 'off'"></mandatory>
   </constraints>
  </format>
   ```

   Note: You may also have other options in your field.
6. Find the other fields and add similar active and format options to each. While the same logic applies for each item, the computes will be slightly different. The logic states that if the OTHER fields are empty and the current field is active, then the current field is optional. Therefore, you must reference the OTHER fields in the compute. For example, to build the compute on the

mandatory element of the format option in FIELD2, you would replace the reference to FIELD2 in the formula shown in the code above with a reference to FIELD1:

```
active=='on' and FIELD1.value=='' and FIELD3.value=='' ? 'off' : 'on'
```

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

S325-2607-00