IBM® Workplace Forms™

Version 2.7

Overview of IBM Workplace Forms

S229-1507-01

First Edition (February 2007)

This edition applies to version 2.7 of IBM® Workplace Forms™.

Contents

Introduction

The IBM® Workplace Forms™ suite includes the following products:

- IBM Workplace Forms Viewer
- IBM Workplace Forms Designer
- IBM Workplace Forms Server
  - API
  - Services Platform
  - Webform Server
  - Deployment Server

Together, these tools allow you to create, fill, and submit forms, and integrate those forms with your back-end processes.

This document explains each component of the Workplace Forms from a technical perspective to help you understand what the products are, how they work, and how they integrate with other products.

Audience

This document is intended for technical personnel who will need to work with Workplace Forms.

About IBM Workplace Forms

The IBM Workplace Forms suite of products is based around an open, XML-based, document-centric architecture. Workplace Forms documents are written in XFDL – the Extensible Forms Description Language, which is an XML-based markup language that describes forms and form components. Uncompressed, these files are human-readable.

All the components of a Workplace Form are contained in a single XFDL file. All the information the form provides and collects is encapsulated into one file, with data and presentation together. Workplace Forms are viewed and filled using Workplace Forms Viewer, or rendered into HTML via the Workplace Forms Webform Server. There are no templates, and no data files.

XFDL files can be created using the drag-and-drop interface of the Workplace Forms Designer, or if desired, in a text editor. They can be manipulated programmatically by servlets, portlets, or other applications that have been created using the Workplace Forms Server – API or the Workplace Forms Server – Services Platform.

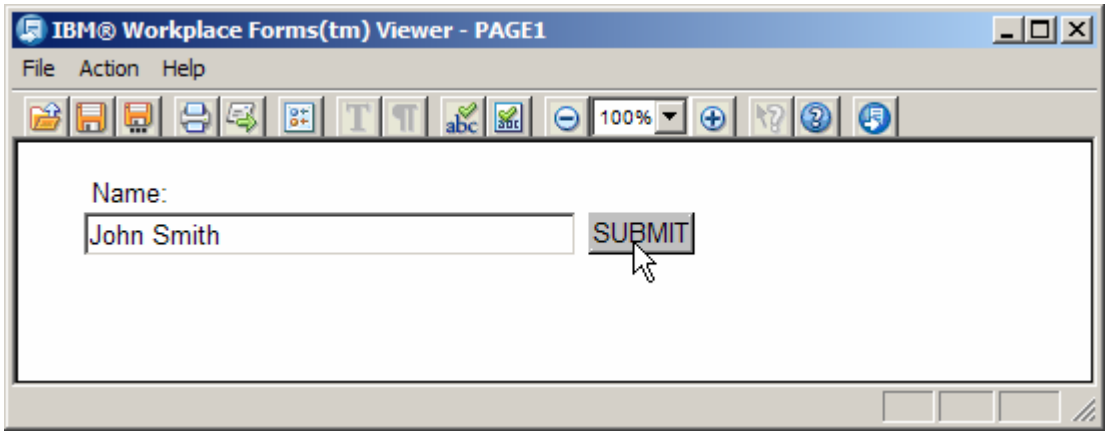
Extracting the data from a Workplace Form is straightforward, but always requires some custom application development that takes into account the type of data, and its intended purpose.

The primary purpose of Workplace Forms is to collect data, but they can be extended to do much more. They can act as the “front end” to any data-driven application, including workflows. This capability is enhanced by the ability to use webservices to transfer information to and from the form at runtime, the ability to run in a portlet environment, and the ability to extend the functionality of the Viewer with an IFX – a “plugin” to the Viewer that can interface with other applications, either on the user’s computer or over a network connection. (See page 5 for more information on IFXs.)

About XFDL

The Extensible Forms Description Language (XFDL) provides markup describing the most common form widgets: fields, text and image labels, buttons, lists (popup and static), combo field/popups, radio buttons, and check boxes. Additionally, with the assimilation of the XForms standards into XFDL, it provides tables, sliders, and grouping constructs including panes, checkgroups and radiogroups. These widgets depend on the form containing an xformsmodel. The xformsmodel is optional, but recommended, so the examples in this document will include one.

Here is an example of a basic form.



The markup describing this form looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Namespace declarations

```
<XFDL xmlns:custom="http://www.ibm.com/xmlns/prod/XFDL/Custom"
xmlns:designer="http://www.ibm.com/xmlns/prod/workplace/forms/designer/2.6"
xmlns:ev="http://www.w3.org/2001/xml-events"
xmlns:xfdl="http://www.ibm.com/xmlns/prod/XFDL/7.1"
xmlns:xforms="http://www.w3.org/2002/xforms"
xmlns="http://www.ibm.com/xmlns/prod/XFDL/7.1"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
   <globalpage sid="global">
      <global sid="global">
         <formid>
            <title></title>
            <serialnumber>7CFA0DB8C86DBF3C:-37747915:1109EC5DBAF:-8000</serialnumber>
            <version>1.0.0</version>
         </formid>
         <xformsmodels>
            <xforms:model>
               <xforms:instance id="INSTANCE" xmlns="">
                  <data>
                     <name labeltext="Name:"></name>
                  </data>
               </xforms:instance>
               <xforms:submission action="http://myserver.com/myServlet" id="SUBMISSION1"
instance="INSTANCE" method="post"></xforms:submission>
            </xforms:model>
         </xformsmodels>
         <designer:version>2.7.0.41</designer:version>
      </global>
   </globalpage>
   <page sid="PAGE1">
      <global sid="global">
         <label>PAGE1</label>
      </global>
      <field sid="nameField">
         <xforms:input ref="instance('INSTANCE')/name">
            <xforms:label></xforms:label>
         </xforms:input>
         <itemlocation>
            <x>29</x>
            <y>30</y>
         </itemlocation>
```

Global page – global item. This holds data and settings that affect the entire form, including the xformsmodel.

Xformsmodel. This holds data from the form, calculations to perform checks on the data, and submission descriptions.

First page of the form.

Field markup

```
            <scrollhoriz>wordwrap</scrollhoriz>
         </field>
         <label sid="nameLabel">
            <xforms:output>
               <xforms:label ref="instance('INSTANCE')/name/@labeltext"></xforms:label>
            </xforms:output>
            <itemlocation>
               <x>30</x>
               <y>9</y>
            </itemlocation>
         </label>
         <button sid="BUTTON1">
            <xforms:submit submission="SUBMISSION1">
               <xforms:label>SUBMIT</xforms:label>
            </xforms:submit>
            <itemlocation>
               <x>280</x>
               <y>29</y>
            </itemlocation>
         </button>
         <spacer sid="SPACER1">
            <itemlocation>
               <x>32</x>
               <y>85</y>
            </itemlocation>
         </spacer>
      </page>
   </XFDL>
```

Label markup

Button markup

Spacer markup. A spacer is a non-visible object used to size the form, or to provide whitespace between objects that are positioned relative to one another.

While it is possible to create XFDL forms in a text editor, it is a labour-intensive process and most people prefer to use Workplace Forms Designer.

Workplace Forms Viewer

The Workplace Forms Viewer renders XFDL. It can run as a standalone application, as a browser plugin, or as an embedded object in HTML. All instances use the same executable, with the plugin or embedded object using an ActiveX container for the executable.

The Viewer has a toolbelt that appears at the top of its window that provides the capability to open, save, print, and e-mail forms, check spelling, format text (in fields that are configured to support this), zoom in on the form,

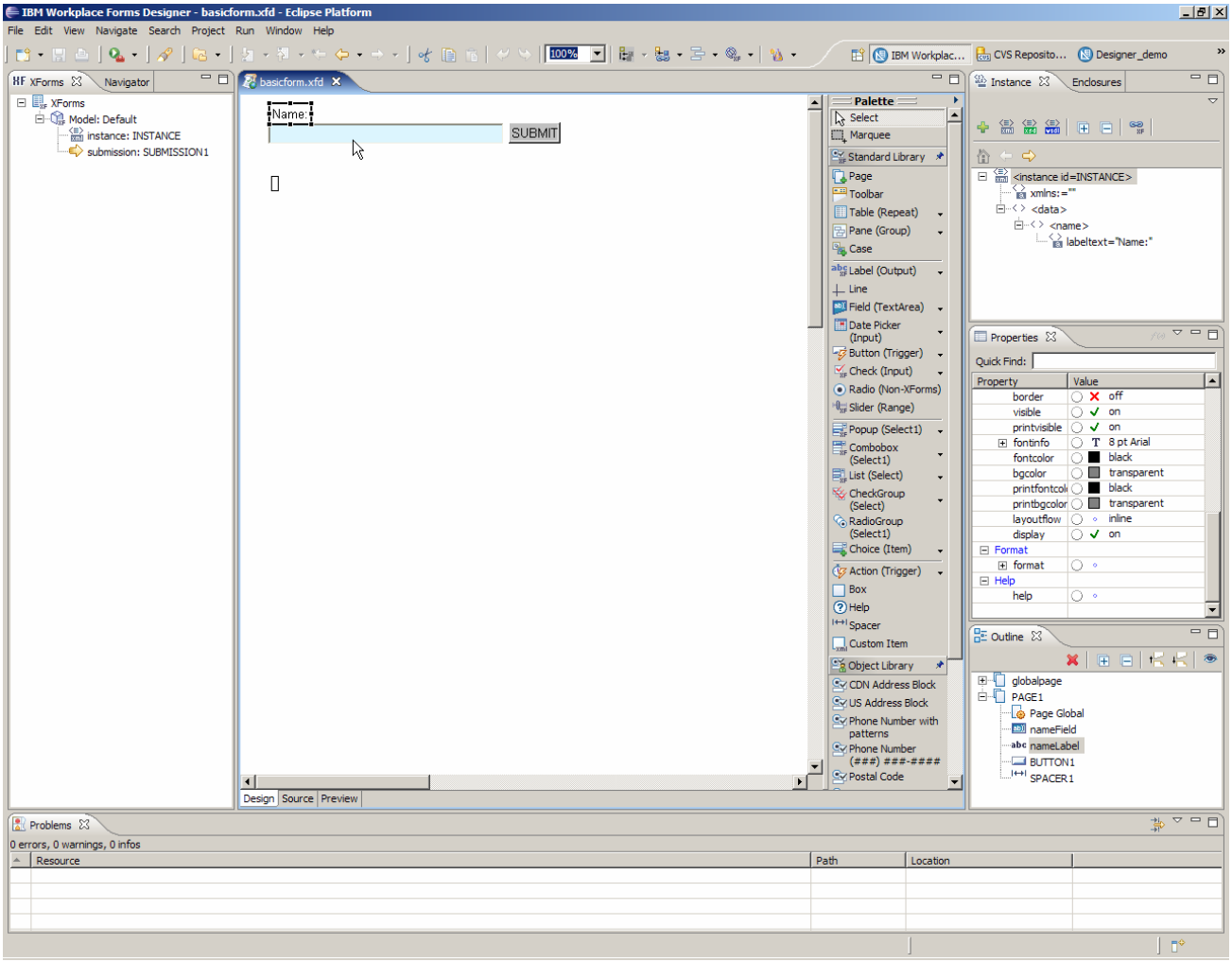
toggle in and out of accessibility mode, and configure preferences. However, these toolbelt buttons and the toolbelt itself can be deactivated or hidden entirely by settings in individual forms.

The Viewer provides a two libraries of functions that are available to forms. These libraries include functions for mathematical operations, string manipulation, and various utilities. Developers can also extend the Viewer's built-in functionality by creating forms extensions (IFX) that can be either embedded in a single form or deployed with the Viewer for use in all forms. An IFX consists of a package of functions that can perform tasks required by the form but not provided in the standard function package. Typically, if the form requires information from another process or application, an IFX is employed.

Functions within IFXs that are deployed with the Viewer are available to forms designers through the Designer's compute wizard interface.

Workplace Forms Designer

The Designer is an Eclipse plug-in that provides a WYSIWYG interface for placing items on the form, and various interfaces for setting item properties, setting up calculations in the form, creating submissions, links to Webservices, and so on. It provides several "views" into a form.



Design Panel and Tools Palette

This is the main interface, where you can select a widget from the Palette and place it on the “canvas”. Once placed, you can resize or reposition the widget as you wish. The Design Panel also has “snap to grid” and “quick-align” features that allow you to position widgets with pixel-precision accuracy.

You can also use the arrow keys on the keyboard for precise layout and sizing.

Code View Tab

This tab on the main design panel shows the XFDL source code of the form. It provides a good XML editor that has auto-complete functionality to assist with tags and attributes.

Preview Tab

This tab shows what the form will look like in the Viewer. It actually runs Workplace Forms Viewer embedded in the Designer context in Eclipse. The form is fully functional except for e-mail functionality.

Xformsmodel View

This view shows the elements of the xformsmodel including binds and submits. You can add and edit both binds and submits in this view.

XForms Instance View

This view shows all the xforms:instances in a form in an expandable tree view. It allows you to copy references into other views, cut and paste nodes, add nodes, and set attribute values.

Properties View

This view shows all the relevant properties for any selected widget(s). If more than one widget is selected, it shows all properties that the selected widgets have in common. Properties, including xforms options, can be edited and multiple choices for properties are available through drop-downs.

Enclosures View

This view displays all enclosures in the form, including images used in buttons and labels, attached files, WSDL definitions, XML Schema definitions, external XForms instances, and .jar files (Java-based forms extensions). This view also allows you to enclose any of the above file types.

Outline View

This view shows the heirarchical tree view of the form. You can cut and paste (and thus re-order) items in this view, or easily switch pages by clicking on the page you want to view in the outline.

Problems View

This view shows any problems with the form, including xforms references that cannot be resolved and syntax errors in the code view.

The Designer also provides several wizards to assist in developing table widgets, creating signature buttons, and developing dynamic form behaviour via “computes” – calculations that the form executes at runtime.

Workplace Forms Server Products

Application Programmer Interface

The API consists of two parts: an FCI (function call interface) used for developing forms extensions (IFXs), and a Form Library, for manipulating forms in the context of a custom application.

The API is the same codebase on which the Viewer, Designer, and all Workplace Forms products are based. It is not entirely exposed; much of it remains private due to maintenance concerns.

Function Call Interface

This toolkit enables you to create extensions to forms (IFXs). These are usually created because the desired functionality cannot be accomplished in XDFL, or because the desired functionality is prohibitively complex in XFDL and the functionality needs to be simplified to be reusable for less technical form designers. Extensions can also be used to extend the Viewer’s functionality to provide things like interactive popup windows or connections to other applications on the user’s computer.

Form Library

The form library provides a collection of function for manipulating workplace forms, including:

- Reading forms into memory
- Adding/removing/changing elements in forms
- Working with signatures
- Enclosing/Extracting data models (XForms or XMLModel)
- Writing forms to disk

Typically, the form library is used to create back-end processes that require form manipulation. While the Forms Services Platform (detailed in the next section) provides ready-made application pieces, some functionality in many applications will still need to be completed manually by a developer. Examples of this include:

- form manipulation within servlets and portlets

- back-end processes that strip extraneous information from forms before they are stored in databases
- processes that piece together a single form from multiple forms
- processes that validate signatures
- processes that extract attachments from forms

Forms Services Platform

The Forms Services Platform is a full-featured, drag-and-drop server-based application development framework. It utilizes IBM Websphere Transformation Extender to create reusable links to repositories, workflow applications, databases, and other applications like SAP. It provides a plugin for Workplace Forms Designer to enable application designer to do direct form-to-process mapping. The Services Platform is the basis of all the Workplace Forms integration offerings. It provides an architectural style and tool kit that allow you to define modules which you can connect together to quickly and simply integrate backend applications.

The Services Platform consists of a collection of Java programming libraries to help you develop applications. These libraries include:

- a form Object that represents the form in memory
- a mechanism to handle requests
- a mechanism to handle responses
- a set of reusable pipes
- a configuration system that controls pipeline execution at runtime
- a set of Services Platform classes

To create an application component under the Services Platform, you utilize “pipes” – small, task-oriented pieces of Java code with accompanying configuration information – and assemble them into “pipelines” to perform a more complex task. Pipes may be pre-existing or created by a programmer.

Deployment Server

The Deployment Server is a server-based product that allows organizations to deploy the Viewer to user desktops via a signed Java applet in a browser. The signed applet installs the contents of a package to the user’s computer.

Deployment Server has three key components:

- An applet that controls the installation of software on the end-user’s computer.

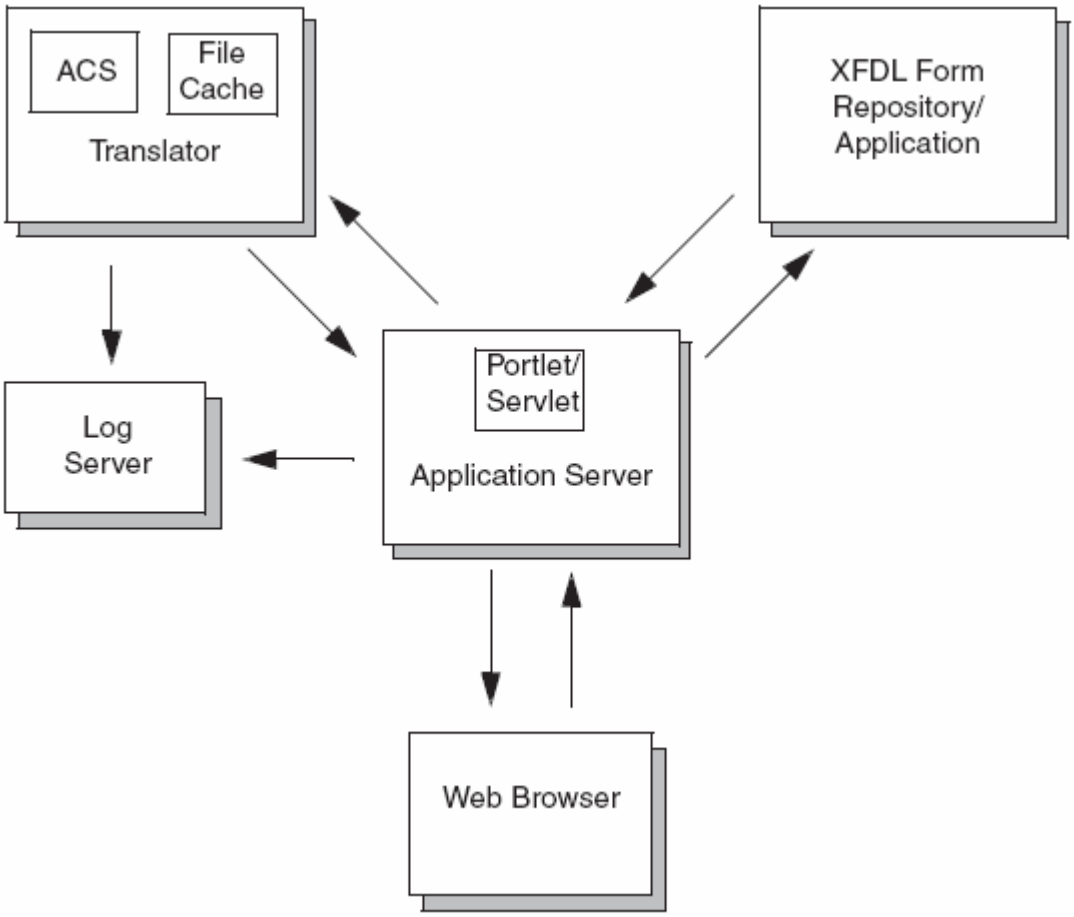
- A servlet that communicates with the applet and passes both instructions and files to the applet.
- A file system that stores all of the instructions and files necessary for deployment.

The file system stores two types of files, called manifests and packages. Each manifest is a set of instructions that the applet follows when deploying an application. Each package is a set of installation files that may install either a complete application or a portion of an application. The package can be configured to contain virtually any software, not just the Viewer, so forms extensions (IFXs) and other applications can be installed this way as well – the only consideration is download size. With this in mind, the Deployment Server also utilizes an extremely lightweight installer that is configured with the manifest file.

Webforms Server

This server product provides a zero-footprint method of viewing Workplace Forms in a client-side browser. It runs within Websphere Application Server or another server, intercepts form requests, and translates forms to HTML, returning the HTML-rendered form to the user's browser. It perfectly replicates the pixel-precise layout of Workplace Forms and even replicates much of the dynamic functionality using AJAX (asynchronous javascript and XML).

Webform Server relies on three central components: a portlet or servlet, a Translator, and a Log Server. The Translator also contains two sub-components: an Access Control Server (ACS) and a File Cache. The following diagram shows how these components are set up in a typical installation:



The portlet/servlet is installed on the application server. You can install the Translator and the Log Server on any computer that the application server can communicate with. Finally, you may also have a Form Repository or a Form Application from which the portlet or servlet retrieves XFDL forms.

The components perform the following functions:

- Portlet/Servlet — The portlet or servlet controls the forms application and processes all incoming and outgoing forms. It passes forms to the Translator for conversion, and may communicate with other applications or rely on a forms repository. This component is unique to each application, and must be written by you. Webform Server provides a Framework that includes methods for this purpose.
- Translator — The Translator converts forms between XFDL and HTML. When converting a form to HTML, the Translator stores the original XFDL in a file cache and saves some metadata about the form in access control list. When converting a form back to XFDL, the Translator retrieves the original form from the file cache and transfers the data from the submitted HTML form.
- Access Control Server – The ACS controls access to the Translator, ensuring that any resource requests come from current users/forms that require the given resource.
- Log Server — The Log Server logs activity for both the Translator and the portlet/servlet. The logs provide general transactional information, as well as error information.
- XFDL Form Repository/Application — This optional component may be a collection of XFDL forms that users may work with or an application that will return XFDL forms.

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' responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

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 " 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.

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.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Trademarks

IBM, the IBM logo, Workplace Forms, DB2, and WebSphere are trademarks of International Business Machines Corporation in the United States, other countries, or both:

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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