

- ▶ **Verifying Your Portal Installation**
- ▶ **Understanding the Portal Environment**
- ▶ **Implementing Security**
 - ▶ **Security Concepts**
 - ▶ **Adding Users**
 - ▶ **Adding Groups**
 - ▶ **Controlling Access**
- ▶ **Adding SAS Content**
 - ▶ **Tables**
 - ▶ **Reports (Stored Processes)**
 - ▶ **Publication Channels**
 - ▶ **Warehouse Data**
 - ▶ **webEIS Documents**
- ▶ **Adding Links**
- ▶ **Adding Content Channels**
- ▶ **Adding Documents**
- ▶ **Adding Applications and Widgets**
- ▶ **Localizing the Portal**
- ▶ **Customizing the Portal's Appearance**

Implementation and Administration Guide for the SAS[®] Information Delivery Portal

This Implementation and Administration Guide provides instructions for fully implementing the SAS Information Delivery Portal to meet the unique needs of your organization. The guide tells how to:

- Set up portal security, including registration of individual users, dividing users into groups, and placing access controls on portal content.
- Make your organization's SAS-generated content available from the portal, including SAS tables, ad hoc reports, publication channels, warehouse data, and dynamic webEIS documents.
- Add other types of content to the portal, including links to internal or external web addresses, syndicated content from information providers, and documents.
- Create your own presentations of data to display in the portal through the use of widgets.
- Develop custom applications that leverage the portal's services, content, and security features.
- Localize the portal to use the appropriate language for the user audience.
- Change the appearance of the portal, including color schemes, logos, images, and other cosmetic features.

This guide will be expanded and modified on an ongoing basis to reflect new information gained from the implementation experiences of SAS customers.

Verifying Your Portal Installation

The procedures in this manual assume that you have successfully installed the demo version of the SAS Information Delivery Portal. Detailed installation instructions are contained in the `readme.html` file that is provided with the portal software.

To verify and ensure correct installation, refer to the following sources:

- **Installation Checklist.** Use the Installation Checklist on this page to help you verify that all of the steps in the installation process have been successfully completed. The checklist includes important technical details that are critical to the operation of the portal.
- **Touring the Demo Portal.** This document introduces you to the portal's user interface and shows you how to view the sample content that is part of the demonstration portal. By working through this demonstration, you can further verify that the installation has been completed correctly.
- **Troubleshooting Guide.** This document will assist you in resolving error messages and other problem situations that could occur when you first attempt to operate the portal.

When all requirements on the Installation Checklist have been met and the demo portal is operating successfully, this means the basic components of the SAS Information Delivery Portal environment are in place. You can modify the portal and add your own content to meet the unique needs of your organization. Before proceeding, you may wish to review the descriptive information in *Understanding the Portal Environment*.

Installation Checklist

The following checklist will help you verify your installation. If you detect problems as you go through this checklist, refer to the appropriate section of the installation instructions.

| Tier | Installation Component | Checklist Question | Yes/No |
|------|------------------------|--------------------|--------|
| | | | |

| | | | |
|------------|---|--|--|
| Web Server | Servlet Container | Have you installed a servlet container that supports the Java Servlet Specification, Version 2.2 or higher? | |
| | | Does the servlet container support JavaServer Pages Specification, Version 1.1 or higher? | |
| | | Is the servlet container configured to use Java 2 SDK, Standard Edition, Version 1.3.0_01 or higher? | |
| | | Were you able to run the examples that were provided with the servlet container? | |
| | | Do you have the necessary user rights and permissions to copy files into the servlet container's root directory? | |
| | | Does the servlet container's cache need to be cleaned out or updated? | |
| | | Java Platform | Have you installed Java 2 SDK, version 1.3.0_01 or higher as the Java environment for the servlet container? |
| | Is the Java 2 SDK bin directory in your path? | | |
| | Do you have the necessary user rights and permissions to copy files into the Java extensions directory? | | |
| | Have the portal's JAR files been placed in the Java extensions directory? | | |
| | | | |
| | | | |
| | | | |

| | | | |
|------------|--------------------|---|--|
| | Portal Application | Have you installed the portal files in the servlet container, either manually or using the WAR file? | |
| | | Are the portal.properties and portalregistry.properties files free of typographical errors? | |
| | | Have you copied the contents of the ServletDemos subdirectory to a location that is accessible to your servlet container? Does this match the location specified in the \$SERVLET_DEMO_DIR\$ property of the Portal.properties file? | |
| | | Does the \$servlet_url\$ property in the portal.properties file contain the correct URL for the portal application? | |
| | | | |
| SAS Server | SAS Software | Is base SAS Version 8.2 installed on the host where you are running the SAS server? | |
| | | <p>If you wish to enable viewing of MDDBs and webEIS documents:</p> <ul style="list-style-type: none"> ● Has either SAS/EIS Software or SAS/MDDB Server Software been installed on the host where you are running the SAS server? ● Have you updated the SAS catalogs to enable MDDB viewing? | |
| | | Have you copied the contents of the SASDemos subdirectory to a location that is accessible to your SAS server? Does this match the location specified in the \$SAS_DEMO_DIR\$ property of the Portal.properties file? | |

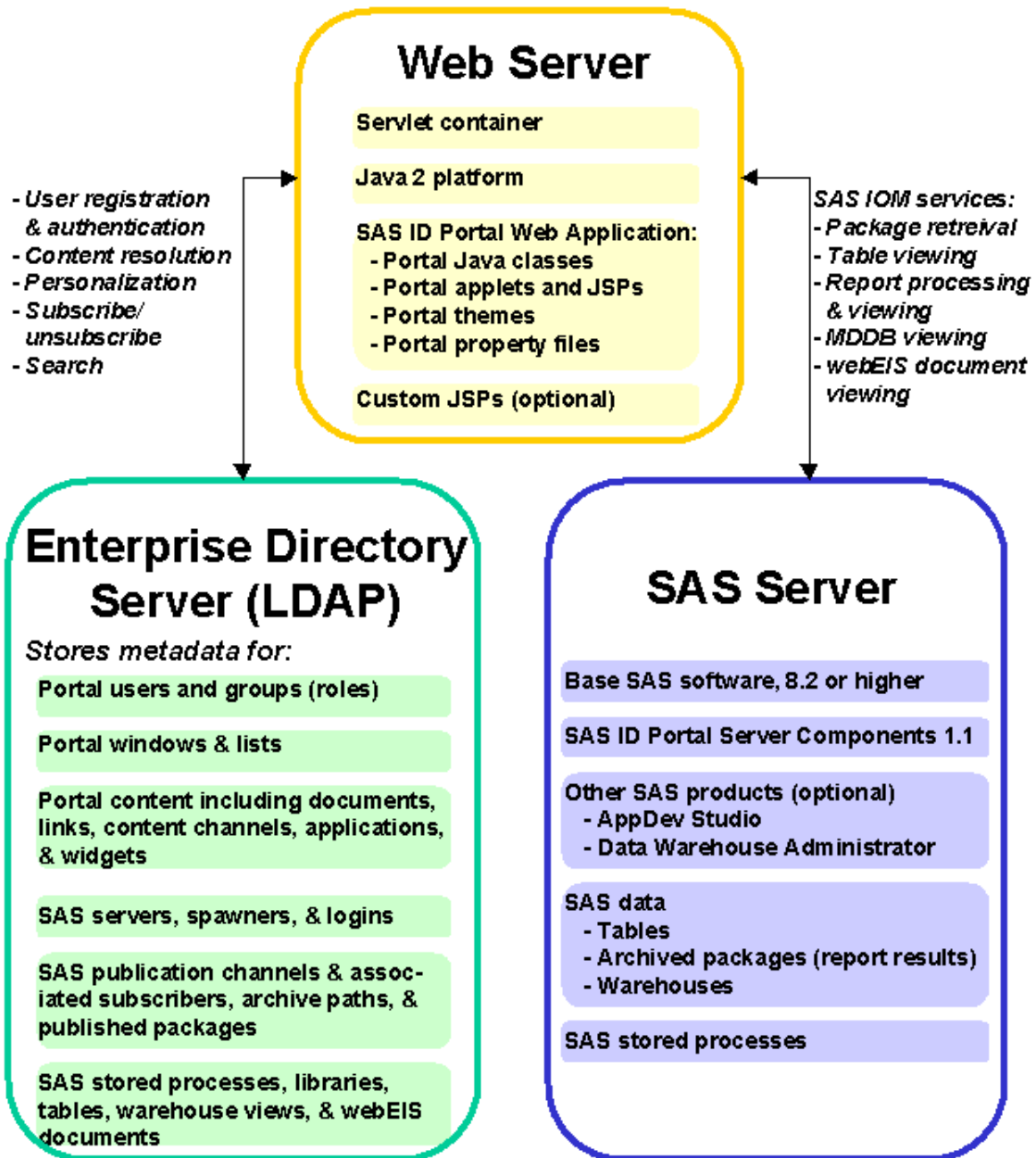
| | | | |
|------------------|----------------------|--|--|
| | SAS IT Administrator | Have you installed IT Administrator, Version 1.5? | |
| | | Do the setServer and setBase settings on the IT Administrator's configuration file (site.cfg) refer to the correct LDAP host and directory tree? | |
| | SAS IOM Spawner | Does the spawner's configuration file specify the correct sasspawnerCN (e.g., "Portal Demo Spawner" for the demo portal)? | |
| | | Does the user who is starting the spawner have the appropriate user rights and permissions on the SAS server machine? | |
| | | Has the spawner been successfully installed and started on the SAS server machine? | |
| | | | |
| Directory Server | LDAP Software | Has the appropriate LDAP software been installed? | |
| | LDAP Metadata | Have you imported the containers.ldif and portal.ldif files generated by the PortalConfigure tool? | |
| | | Are access controls set as required for the Portal User, Portal Guest User, and Portal Demo User identities? | |
| | | Does the sasLogin definition contain a user ID and password that are valid on the SAS server machine? | |

Understanding the Portal Environment

When you have completed the installation procedures, verified your installation, and ensured that the

demo portal is operating successfully, this means that all of the basic components of the SAS Information Delivery Portal environment are in place. You can now begin the implementation tasks that are necessary to meet your organization’s specific information delivery requirements.

Before you begin these tasks, it is important to understand the overall structure of the portal and the components that make it work. The following diagram provides a high-level, conceptual view of the portal’s main components and how they interact with one another:



The architecture of the SAS Information Delivery Portal gives you the flexibility to distribute these components as required. For small implementations, the Web server, enterprise directory server, and

SAS server may all run on the same machine. In contrast, a large enterprise may have multiple compute and data servers and an enterprise directory that is distributed across multiple platforms.

Note: For a more technically oriented view of the portal, refer to the Portal Architecture document.

Web Server

The Web server is the platform that supports the operation of the SAS Information Delivery Portal Web Application. This server contains the following components:

- **Servlet container.** The servlet container provides the platform, or engine, on which to run the portal's servlets and Java Server Pages (JSPs). For more information, see the Java Servlet Technology page on the Sun Microsystems Web site.
- **Java 2 platform.** The Java 2 Software Development Kit (SDK), Standard Edition, provides the software development language and runtime environment for the portal Web application. For information, see the Products and APIs page on the Sun Microsystems Web site.
- **Portal Java classes.** The foundation of the portal Web application consists of Java classes contained in two System Development Kits (SDKs): the Enterprise Directory SDK and the SAS Portal SDK. Refer to the API Reference for complete documentation of the Java classes included in these SDKs. If you wish, you can use these classes to develop your own custom JSPs for deployment in the portal. For details, see Adding Applications and Widgets to the Portal.
- **Portal applets and JSPs.** The portal applets and JSPs are the active components of the portal Web application. Using the portal Java classes, these applets and JSPs interact with the enterprise directory server and the SAS server to deliver portal functionality and content to users.
- **Portal themes.** The portal themes control the appearance of the portal's user interface. The themes consist of Cascading Style Sheets (CSSs) and graphical elements, including the portal's banner, background image, and logo. To modify the themes, refer to Customizing the Portal's Appearance in this manual.
- **Property files.** The following property files contain parameters which control the operation of the portal: Portal.properties, Mime.properties, NewUser.properties, and PortalRegistry.properties. The Portal.properties file contains your default locale setting, information about your enterprise directory, and a number of debug flags that you can set to "true" in order to display more detailed information in the servlet container log.
- **Custom JSPs.** You can develop your own custom JSPs that take advantage of the portal's content, metadata, and security services. Three types of custom JSPs can be deployed in the portal: widgets, applications, and webEIS documents. To deploy custom JSPs, place them in the appropriate path in the portal's setup directory; rerun the PortalConfigure program to create a new WAR file; deploy the new WAR file in the servlet container; and use the Portal's personalization tool to add metadata to the enterprise directory. For more information, see Adding SAS webEIS Documents to the Portal and Adding Applications and Widgets to the Portal.

Enterprise Directory Server

The enterprise directory server is a central repository for the portal's user, resource, and security information. This server does not contain actual content for the portal; instead, it contains *metadata*, or data about the content. The portal's enterprise directory is implemented using the Lightweight Directory Access Protocol (LDAP). For more information about LDAP, see the Directory Services topic on the SAS Integration Technologies Web site. The portal application uses the Java Naming and Directory Interface (JNDI) to access the enterprise directory server.

The portal's installation process builds an initial set of enterprise directory entries that form the basis for a demonstration portal. Building on the base set of entries, you can add and change metadata as needed to implement a portal that meets the needs of your organization. Procedures for updating the metadata are detailed throughout this Implementation and Administration Guide. In these procedures, you will find that there are several ways to update the metadata:

- The portal's **login** tool creates new user entries.
- The portal's **personalization** tool creates entries for portal windows, portal lists, and certain types of portal content.
- The **SAS Integration Technologies (IT) Administrator** application creates entries for the SAS server configuration and SAS data.
- The **SAS Data Warehouse Administrator** exports metadata about selected warehouse views.
- The **enterprise directory console** is used to update some types of metadata. (Authorized users can use this method to update any type of entry on the directory).

The following table provides a quick reference to the various categories of metadata that reside on the portal's enterprise directory. For each metadata category, the table shows the portal component that the metadata describes, the initial metadata entries that are loaded during the installation process, and the method that can be used to update the metadata. For detailed documentation, click on the entries in the "Component" column.

| Metadata in the Portal Enterprise Directory | | | |
|---|-----------|-----------------|---------------|
| Category | Component | Initial Entries | How to Update |
| | | | |

| | | | |
|--------------------------|---|---|--|
| Users and groups | Portal users | Portal User Portal Admin Portal Demo Portal Guest DW Admin | Enterprise directory console, or self-registration via portal log-in screen |
| | Portal Groups (Roles) | Public Portal Demo Group Portal Admin Group DW Admin Group | Enterprise directory console |
| SAS configuration | SAS servers, spawners, logins, and logical names | Portal Demo Server Portal Demo Spawner Portal Demo Login Portal Demo Logical | IT Administrator |
| Content | SAS Tables | Product Sales Summary (in sashelp library) | IT Administrator or export from Data Warehouse Administrator |
| | SAS Reports (Stored Processes) | Product Sales Demo | IT Administrator |
| | SAS Publication Channels | Channels: Demo Channels 1 and 2 Subscribers: Portal Demo Subscriber Published packages: s90c0700, s90c0701, s90c0702 | Channels: IT Administrator Subscriptions: Portal personalization Published packages: IT Publish Framework or Enterprise Guide |
| | SAS Data Warehouse views | N/A | Export from Data Warehouse Administrator |

| | | | |
|-----------------------|---------------------------------|---|------------------------|
| | SAS webEIS documents | Demo.jsp | Portal personalization |
| | Links | SAS Home Page SAS Integration Technologies CNN CNNSI | Portal personalization |
| | Documents | Demo Documents 1, 2, and 3 | Portal personalization |
| | Content channels | Top Finance Stories Top News Stories Top Sports Stories Top Tech Stories | Portal personalization |
| | Widgets and applications | Widget Sample 1 Application samples 1, 2, 3, and 4 | Portal personalization |
| User interface | Portal windows and lists | Various | Portal personalization |

SAS Server

The SAS server is the platform that allows the portal to exploit SAS analytic and reporting functions in order to deliver information to the desktops of authorized portal users. The SAS server enables portal users to perform the following functions from client machines that have only a Web browser installed: view SAS tables, run SAS reports (known as "stored processes"), view SAS data warehouses, view webEIS documents, subscribe to SAS publication channels, and view packages that are published to these channels.

The SAS server contains the following components:

- **Base SAS software.** Base SAS software performs the data access, management, analysis and presentation tasks that form the basis for all other SAS information delivery applications.
- **SAS Information Delivery Portal Server Components.** These components enable the operation

of the SAS Integrated Object Model (IOM) server on the SAS server machine. The IOM server contains distributed object interfaces that allow programs (such as the portal Web application) on client machines to execute the base SAS software features. An Object Spawner running on the server validates user and application requests for workspaces and instantiates workspaces as needed. For more information, refer to Integrated Object Model topic on the SAS Integration Technologies Web site.

- **Other SAS products.** Other SAS products such as AppDev Studio, Data Warehouse Administrator, and Enterprise Guide can also be installed on the SAS server. By leveraging the capabilities of these products, you can provide enhanced information delivery capabilities to portal users.
 - **SAS data.** Your organization's SAS data can be stored either on the SAS server or on a machine that is accessible to the server.
 - **SAS stored processes.** A stored process is a SAS program that resides on a SAS server and is available to be executed on a request basis. For more information, see the Stored Processes topic on the SAS Integration Technologies Web site, and see Adding Stored Processes to the Portal in this manual.
-

Implementing Security

Each implementation of the SAS Information Delivery Portal will have different security requirements. In determining how to implement portal security, you should consider your organization's internal security policies, the security mechanisms that are in place in your environment, the types of users who will need to access the portal, and the types of content that will be made available. You can then implement your security requirements using the following mechanisms:

- **User definitions.** The SAS Information Delivery Portal provides two methods for adding individual users: registration through the portal log-in screen, and registration by an administrator. Definitions of registered users are stored in the portal's enterprise directory. A user who has been registered on the system can personalize the portal by adding or modifying his or her own windows, lists, and content items. In addition, registered users can be granted access to specific portal content. See Defining Portal Users for details on setting up users.
- **Group definitions.** For efficient management of portal security, it is recommended that you organize registered users into groups on the enterprise directory. You can then grant access to portal content to the appropriate groups based on the sensitivity of the data and the users' needs for information. See Defining Portal Groups (Roles) for details on setting up groups.
- **User authentication.** To authenticate users, the SAS Information Delivery Portal can either require users to log on using the portal's own log-on procedure, or it can perform authentication based on Web server log-on information.
 - **Authentication using portal log-on.** By default, the portal uses its own log-on procedure to

authenticate users. When a user first brings up the SAS Information Delivery Portal, the public areas of the portal application become available for searching and viewing. To establish a specific user identity, the user chooses the **Log On** task from the toolbar. On the log-on page, the user enters a user name and password, which the portal uses to perform a bind operation to authenticate the user. During authentication, the portal searches the enterprise directory for a person entry whose "uid" and password attributes match the user name and password that were entered.

- **Authentication via Web server sign-on.** Enhancements provided with Version 1.1.1 allow you to implement user authentication based on the user's identity as it was provided to the Web server. When the user brings up the SAS Information Delivery Portal, the portal uses the `request.getRemoteUser()` method to obtain the user's identity from the Web server. The portal then searches the enterprise directory for a person entry whose "uid" attribute matches the user id provided by the Web server.

To implement this method, enter the following setting in the `Portal.properties` file:

```
Servlet.Security.TrustWebServerLogin=true
```

- **Access control.** You can use several different methods to allow or restrict access to portal content. The methods are as follows:
 - **SAS Login definitions.** SAS Login definitions on the enterprise directory ensure that only authorized portal users obtain access to SAS data and processes. Each SAS server that you define to the portal must have a login definition that specifies which users or groups of users can access the server.
 - **Personal and role-based content.** Through the portal's Personalization feature, you can create role-based or personal webEIS documents, links, applications, widgets, content channels, windows, and lists. Role-based content is available only to a particular group of users, and personal content is available only to user who adds it to the portal. The portal uses ACI rules (described in the next bullet) to impose these restrictions.
 - **Access Control Information (ACI) rules.** The portal also uses Access Control Information (ACI) rules in the enterprise directory to determine which portal components or content can be accessed by the user. For a given object or group of objects, an ACI rule can explicitly allow or disallow specific types of access to individual registered users or groups of users. An authorized administrator can manually update the enterprise directory with ACI rules to control access to any object on the directory. Use of these rules provides virtually unlimited flexibility in controlling access to portal content.

See [Controlling Access to Portal Content](#) for details on setting up access controls.

The installation process for the SAS Information Delivery Portal creates a default security structure in your enterprise directory that controls security on both the Web server (middle tier) and SAS server (back tier). The structure includes a default set of permissions that enable the portal to operate on a demonstration basis. For a description of this structure, see [Security Concepts and Policies](#).

Security Concepts and Policies

The installation process for the SAS Information Delivery Portal creates a security structure in your enterprise directory that controls access on both the Web server (middle tier) and the SAS server (back tier). The structure includes a default set of permissions that enable the portal to operate on a demonstration basis. This structure provides a starting point that you can build upon as needed to meet your organization's specific security requirements.

Web Server (Middle Tier) Security

The security structure for the Web server (middle tier) controls access to system components that reside within the portal application, including:

- Portal services
- Some types of portal content, including applications, widgets, content channels, and documents
- Portal components such as lists, links, and windows

The security structure for the middle tier includes the following levels of access:

- **Access by Privileged User.** The SAS Information Delivery Portal application uses a privileged identity to perform specific tasks on behalf of users who are logged on to the application. These tasks include creating new profiles, subscribing and unsubscribing to publication and content channels, and accessing credentials for SAS servers. In the default portal installation, the privileged identity belongs to a user called the Portal User. This definition must remain intact in order for the portal to operate correctly in its default form. (If you want to use another name for the privileged identity, you must change the portal configuration files and the enterprise directory accordingly.)
- **Public Access (Access by All Users).** Certain content items in the default installation of the portal are set up to allow read, search, and compare access for all users. These include anonymous users who have not logged on to the portal and have not been defined in the enterprise directory. Metadata for this type of content is stored in "Public" containers within each content type. To add, modify, or delete public content, an authorized user can log in as the public content administrator, choose the Personalize task, and choose **Public** as the role to personalize. In the default portal installation, the Portal.properties file assigns the public content administrator the user ID "portaladmin."
- **Access by Group Members.** When group-specific content (referred to as "role-based" content) is added to the portal, the portal automatically grants read, search, and compare access to all users who belong to the group. Metadata for this type of content is stored in group containers within each content type. The default portal installation does not include any role-based content. To add, modify, or delete role-based content, the user who has been designated as owner of the group can log on to the portal, choose the Personalize feature, and choose the group's name as the role to personalize.

- **Access by Registered User.** When individual users add content (referred to as "personal content") to the portal, the portal automatically grants all access rights to the user who created the content and no access to anyone else. Metadata for this type of content is stored in content type containers beneath each portal user container. The default portal installation includes personal content for a sample user called Portal Demo. To add, modify, or delete personal content, any registered portal user can log on to the portal, choose the Personalize feature, and then choose "Personal" as the role to personalize.

Back Tier (SAS Server) Security

The security structure for the SAS server (back tier) controls access to SAS system components including SAS tables, reports (stored processes), multidimensional databases (MDDBs), and archived packages. SAS Login definitions on the enterprise directory are the primary mechanism for controlling access to these objects. Each SAS server that you define to the portal must have a login definition that specifies which users or groups of users can access SAS objects on the server. If no users or groups are specified in the login definition, no users will be able to access these SAS objects.

The security structure for the back tier includes the following levels of access:

- **Access by Privileged User.** When a user requests SAS data or a SAS process via the SAS Information Delivery Portal, the workspace manager uses a privileged identity to create a workspace factory on behalf of the user. The workspace factory is a prerequisite to performing any type of process on the SAS server. The privileged identity, which is hidden in the workspace manager, belongs to a user called Portal User.
- **Access by Registered User.** After the privileged user creates the workspace factory, a workspace is requested from the spawner that runs on the SAS server. First, the portal determines whether the requesting user's distinguished name (or the distinguished name of a group to which the user belongs) is present in an associated SAS Login definition. If a match occurs, the portal uses the identity specified in the SAS Login to obtain a workspace on the SAS server.

Note: In the default portal installation's SAS Login definition, all predefined users, including Portal User, Portal Guest, Portal Admin, and Portal Demo, are specified as allowed users. When additional portal users are registered, they will not be able to access SAS data or processes unless their user names (or group names) are added to the SAS Login definition.

Implementing Security for Your Environment

Each implementation of the SAS Information Delivery Portal will have different security requirements. In determining how to implement portal security, you should consider your organization's internal security policies, the security mechanisms that are in place in your environment, the types of users who will need to access the portal, and the type of content that will be made available. Then you can modify the default portal security structure to meet your requirements

Defining Portal Users

Definitions for users of the SAS Information Delivery Portal are stored in the enterprise directory. In order to run, the SAS Information Delivery Portal requires definitions for two specific users at a minimum: `portaluser` and `portalguest`. These users definitions are created automatically during the installation process. Three additional default users are also created: `portaldemo`, `portaladmin`, and `dwadmin`.

In addition to these users, individual person entries should be created for each portal user. Once the user's person entry has been created, the following capabilities become available:

- The user can add and organize portal content to meet his or her unique needs. For details, see *Personalizing Your Portal in the User's Guide*.
- The user can be added to a group, giving the user access to role-based content and other content that is restricted to particular groups. For details, see *Defining Portal Groups*.
- The user's person DN can be used to give the user access to portal content other than that which is available publicly or to groups. For example, the person DN can be added to a SAS Login definition so that the user will have access to data sets and processes on a SAS server. For details, see *Controlling Access to Portal Content*.

The SAS Information Delivery Portal provides two methods for adding individual users: registration through the log-in screen, and registration by an administrator. If you do not have advance information about your users, registration through the log-in screen may be the best way to add users to the portal. If your potential users have differing information needs and differing rights to view content, then you should consider having an administrator register users. This section provides detailed information about both registration methods.

How Users are Defined

Definitions for users of the SAS Information Delivery Portal are stored in the enterprise directory in the `ou=People` organizational unit. When a user attempts to log on, the portal looks in that organizational unit to find the user's definition. It uses the definition information, along with associated group information and access information, to determine which portal content the user is allowed to see.

The LDAP directory contains three entries for each portal user:

- **Person.** The person entry in `ou=People` holds information about the individual, including the user name and password which are used to authenticate the user.
- **Sassubscriber.** The `sassubscriber` entry resides under `cn=sassubscribers`, `sascomponent=sasPublishSubscribe`, `cn=SAS`. This entry is used to store information about the user's subscriptions to SAS publication channels.
- **Sasportalprofile.** The `sasportalprofile` entry resides under `cn=sasPortalUsers`,

sascomponent=sasPortal, cn=SAS. This entry is used to store the user's portal application information, including information about the user's personal links, lists, and windows.

Initial Users: portaluser, portalguest, portaldemo, portaladmin, and dwadmin

In order to run, the SAS Information Delivery Portal requires definitions for two specific users at a minimum: `portaluser` and `portalguest`. Three other generic users are also defined during the installation process: `portaldemo`, `portaladmin`, and `dwadmin`. Each of these users is described below:

- **portaluser.** The user id `portaluser` is the portal's privileged identity. This identity allows the portal application to perform specific tasks on behalf of users who are logged on. These tasks include creating new profiles, subscribing and unsubscribing to publication and content channels, and accessing credentials for SAS servers. To perform these tasks, the `portaluser` must have the following permissions:
 - All access to the SAS application hierarchy (from `cn=SAS, $SAS_CONTEXT`).
 - All access to the areas where person entries and group entries are stored.
 - For SecureWay servers, the `portaluser` should also be defined as the `entryOwner` for entries under `sascomponent=sasPortal`. This will allow the `portaluser` to set access controls for new profiles and other content.
- **portalguest.** The user id `portalguest` is the portal's guest account. It is used to display public content to users who have not logged on.
- **portaldemo.** The user id `portaldemo` is provided for demonstration purposes. It allows you to test your portal implementation and learn about the portal's features. For a step-by-step guide to the demo, see [Stepping Through the Demo Portal](#).
- **portaladmin.** The user id `portaladmin` has the authority to administer the portal's public content. Public content is available to all users, whether or not they have logged on to the portal. The `portaladmin` user should not have any special permissions in the directory.
- **dwadmin.** The user id `dwadmin` has the authority to administer data warehouse content in the portal.

For proper operation of the portal, appropriate access controls *must* be set for the `portaluser`, `portalguest`, `portaldemo`, `portaladmin`, and `dwadmin` user identities. Refer to [Setting Proper Access Controls for User Identities](#) for detailed information about the settings that are required.

Adding Individual Users

In addition to the five initial users, individual person entries should be created for each portal user. The SAS Information Delivery Portal provides two methods for adding individual users: registration through

the log-in screen, and registration by an administrator.

Registration Through the Log-in Screen

If you do not have advance information about your users, registration through the log-in screen may be the best way to add users to the portal. In this scenario, new visitors to the portal use the registration feature on the portal log-in screen to create a user name and password. The portal uses the entered information to create a person entry, a sassubscriber entry, and a sasportalprofile entry in the enterprise directory. These entries give the user access to a default set of content that is available publicly.

If the user needs access to role-based content or to content that is restricted to only certain users, the administrator will need to grant the necessary access after the user has registered. This could involve adding the user's identity to a SAS Login definition; or it could involve adding the user to the appropriate group or groups, as described in Defining Portal Groups.

Group assignment is not necessary for users who only need to view publicly available content. For example, you may decide to make a set of standard, non-restricted portal content available to anyone who visits your organization's internet site.

Registration by an Administrator

Alternatively, a user with administrative permissions can manually create person entries on the enterprise directory. The administrator only needs to create a person entry for the user. The first time the user logs into the portal, the portal automatically creates a sassubscriber entry and a sasportalprofile entry.

If your potential users have differing information needs and differing rights to view content, then you should consider using this registration method. This method allows you to plan user groups and security structures before users are added to the portal. For example, if you are implementing the portal on an organization's intranet or internal network, you may find it beneficial to analyze potential users, create groups based on users' roles and needs, and set up security controls before users begin logging on to the portal.

If you have a user base that includes specific groups of users as well as general users, you may decide to use a combination of user registration and administrator registration.

Creating Person Entries

Each directory entry in the ou=People organizational unit should look like the following. The bold items are those that are different for each user.

```
dn: cn=username, $PERSON_CONTEXT$  
cn: username  
description: user description  
mail: user email address  
objectclass: inetorgperson  
objectclass: person  
sn: short name of the user  
uid: user's portal login ID
```

userpassword: login password

Creating an entry in the directory manually for each portal user can be time consuming. Creating and importing an LDIF file simplifies the process and also provides a backup file of portal users.

Setting Proper Access Controls for User Identities

It is important that you properly set the access controls for the portaluser, portalguest, portaldemo, portaladmin, and dwadmin user identities. The following list should help you create the proper permission settings. **These settings are required for proper operation of the portal.**

The portaluser identity

- must be able to read, write, search, compare, delete, and add entries to the `cn=SAS,$SAS_CONTEXT$` entry and its children
- must be able to read, write, search, compare, delete, and add entries to the `$PERSON_CONTEXT$` entry
- must be able to read, search, and compare the `$GROUP_CONTEXT$`

The portalguest user identity

- must be able to read, search, and compare its own person entry
- must be able to read, search, and compare the group context
- must be able to read, search, and compare entries to the `cn=PortalGuest,cn=sasPortalUsers,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$` entry and its children
- must NOT be able to access any other `cn=sasPortalUser` entries in `cn=sasPortalUsers,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
- must be able to read, search, and compare entries in
 - `cn=Public,cn=sasPortalWindows,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
 - `cn=Public,cn=sasPortalLists,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
 - `cn=Public,cn=sasPortalLinks,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
 - `cn=sasPortalThemes,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
- should be able to read, search, and compare entries in the following contexts that are defined to be publicly available:
 - `cn=sasSubscribers,sascomponent=sasPublishSubscribe,cn=SAS,$SAS_CONTEXT$`
 - `cn=sasChannels,sascomponent=sasPublishSubscribe,cn=SAS,$SAS_CONTEXT$`
 - `sascomponent=sasApplications,cn=SAS,$SAS_CONTEXT$`
 - `cn=sasLibraries,sascomponent=sasDataSources,cn=SAS,$SAS_CONTEXT$`
 - `cn=sasTables,sascomponent=sasDataSources,$SAS_CONTEXT$`
 - `cn=sasInfomarts,sascomponent=sasDataSources,cn=SAS,$SAS_CONTEXT$`

The portaldemo user identity

- must be able to read, write, search, and compare its own person entry
- must be able to read, search, and compare the group context
- must be able to read, write, search, compare, delete, and add entries to the `cn=sasPortalUserCn=Portal` entry and its children
- must NOT be able to access any other `cn=sasPortalUser` entries in `cn=sasPortalUsers, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
- must be able to read, search, and compare entries in
 - `cn=Public, cn=sasPortalWindows, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
 - `cn=Public, cn=sasPortalLists, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
 - `cn=Public, cn=sasPortalLinks, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
 - `cn=sasPortalThemes, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
- should be able to read, search, and compare entries in (**Note:** Disallowing access to any of the following entries will result in content not being shown as documented in the demo.)
 - `cn=sassubscribers, sascomponent=sasPublishSubscribe, cn=SAS, $SAS_CONTEXT$`
 - `cn=saschannels, sascomponent=sasPublishSubscribe, cn=SAS, $SAS_CONTEXT$`
 - `sascomponent=sasApplications, cn=SAS, $SAS_CONTEXT$`
 - `cn=sasLibraries, sascomponent=sasDataSources, cn=SAS, $SAS_CONTEXT$`
 - `cn=sasTables, sascomponent=sasDataSources, $SAS_CONTEXT$`
 - `cn=sasInfomarts, sascomponent=sasDataSources, cn=SAS, $SAS_CONTEXT$`

The portaladmin user identity

- must be able to read, write, search, and compare its own person entry
- must be able to read, search, and compare the group context
- must be able to read, write, search, compare, delete, and add entries to the `cn=sasPortalUserCn=Portal` entry and its children
- must NOT be able to access any other `cn=sasPortalUser` entries in `cn=sasPortalUsers, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
- must be able to read, write, search, and compare entries in
 - `cn=Public, cn=sasPortalWindows, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
 - `cn=Public, cn=sasPortalLists, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
 - `cn=Public, cn=sasPortalLinks, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
 - `cn=sasPortalThemes, sascomponent=sasPortal, cn=SAS, $SAS_CONTEXT$`
- should be able to read, search, and compare entries in
 - `cn=sassubscribers, sascomponent=sasPublishSubscribe, cn=SAS, $SAS_CONTEXT$`
 - `cn=saschannels, sascomponent=sasPublishSubscribe, cn=SAS, $SAS_CONTEXT$`
 - `sascomponent=sasApplications, cn=SAS, $SAS_CONTEXT$`
 - `cn=sasLibraries, sascomponent=sasDataSources, cn=SAS, $SAS_CONTEXT$`
 - `cn=sasTables, sascomponent=sasDataSources, $SAS_CONTEXT$`
 - `cn=sasInfomarts, sascomponent=sasDataSources, cn=SAS, $SAS_CONTEXT$`

The dwadmin user identity

- must be able to read, write, search, and compare its own person entry
- must be able to read, search, and compare the group context
- must be able to read, write, search, compare, delete, and add entries to the `cn=DWAdmin,cn=sasPortalUsers,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$` entry and its children
- must NOT be able to access any other `cn=sasPortalUser` entries in `cn=sasPortalUsers,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
- must be able to read, search, and compare entries in
 - `cn=Public,cn=sasPortalWindows,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
 - `cn=Public,cn=sasPortalLists,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
 - `cn=Public,cn=sasPortalLinks,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
 - `cn=sasPortalThemes,sascomponent=sasPortal,cn=SAS,$SAS_CONTEXT$`
- should be able to read, search and compare entries in
 - `cn=sasSubscribers,sascomponent=sasPublishSubscribe,cn=SAS,$SAS_CONTEXT$`
 - `cn=sasChannels,sascomponent=sasPublishSubscribe,cn=SAS,$SAS_CONTEXT$`
 - `sascomponent=sasApplications,cn=SAS,$SAS_CONTEXT$`
 - `cn=sasLibraries,sascomponent=sasDataSources,cn=SAS,$SAS_CONTEXT$`
 - `cn=sasTables,sascomponent=sasDataSources,$SAS_CONTEXT$`
 - `cn=sasInfomarts,sascomponent=sasDataSources,cn=SAS,$SAS_CONTEXT$`

Defining Portal Groups (Roles)

When setting up user entries for the SAS Information Delivery Portal, it is recommended that you organize the portal users into groups. You can then grant access to portal content to the appropriate groups based on the sensitivity of the data and the users' needs for information. The use of groups is particularly important if the users have differing information needs and differing rights to view content.

The use of groups simplifies the process of administering and maintaining portal security and reduces the chance for errors. For example:

- As new content is added to the portal, you can make it available to the appropriate groups based on the type of information and its level of sensitivity. This process is much simpler than giving access to a long list of individual users. For details, see [Controlling Access to Portal Content](#).
- As new users are added, you can assign them to the appropriate groups and they will automatically have access to the appropriate content. For details about adding users, see [Defining Portal Users](#).
- Using the portal's Personalization feature, users who are authorized as group owners can make "role-based" changes to the portal which become available to all members of the group. For details, see [Personalizing Your Portal](#) in the SAS Information Delivery Portal User's Guide.

Guidelines for Dividing Users Into Groups

Step 1: Analyze Content

The first step in setting up groups is to analyze the content that is planned for the SAS Information Delivery Portal. For each category of content, determine whether access restrictions are needed. If restrictions are needed, identify the types of users that should and should not have access.

Step 2: Analyze and Group Users

After analyzing the content, you can identify groups of users. These user groups may be based on your organization's structure; however, it is more important to group users that have similar data access needs. Start by identifying large groups of users. You can then subdivide those large groups into smaller groups if necessary. For example, you could create an Accounting user group that needs access to financial data through the portal. Within that group, you could identify a subgroup of users who need access to salary information that should not be accessed by the rest of the group.

You may find that the access needs of a group of users are not necessarily identical. In these cases, you can assign a user to more than one group to accommodate unique needs.

The goal is to organize the user base in a way that reduces the number of cases in which specific users must be granted access to specific data. By keeping exception situations to a minimum, you will simplify maintenance tasks and reduce the chance for errors.

Step 3: Assign Group Owners

When you set up a group, you should identify one user in each group as the group owner. The portal gives the group owner authorization to create role-based windows, lists, and links that can be accessed by all members of the group. In addition, the group owner can add role-based content including applications, widgets, content channels, and documents to the portal. This content can then be accessed by all members of the group.

Defining Group Entries

Group entries are typically stored in the enterprise directory under the ou=Groups container. The format of a sample LDAP entry is as follows. The highlighted items are those that are different for each group.

```
dn: cn=group name, $GROUP_CONTEXT$
objectclass: groupofUniqueNames
cn: group name
owner: cn=group administrator, $PERSON_CONTEXT$
uniqueMember: username1, $PERSON_CONTEXT$
uniqueMember: username2, $PERSON_CONTEXT$
.
.
(add as many uniqueMember entries as needed)
```

Notes:

- The groupOfNames object class can also be used for groups. This object class uses the member attribute rather than uniqueMember attribute.

- SecureWay uses the AccessGroup class for groups that represent security groups.

Depending on the size of your organization and the number of groups required, creating group definitions can be time consuming. In some cases, you may be able to reduce the time required for this task:

- If you already store user information in the LDAP directory and you have organized the users into some sort of group structure, it is possible to automate the process of grouping by extracting the LDAP entries based on organizational data.
- Another method is to use definitions in /etc/passwd and /etc/group to create an LDIF file. This file can then be imported into the directory using command-line tools, a directory console utility, or a custom application.

If these methods are not feasible, it will be necessary to create each group definition manually. However, the time you invest in creating the groups will be repaid in lower directory maintenance time.

Granting Groups Access to Portal Content

Once your user groups have been defined, you can use various methods to give the groups access to portal content. For details, see Controlling Access to Portal Content.

Controlling Access to Portal Content

When implementing or administering the SAS Information Delivery Portal, you can use several different methods to allow or restrict access to the portal content. The level of security you apply will depend on your user base and on the sensitivity of the content that you make available through the portal.

The methods for controlling access vary depending on the type of content. The following table summarizes the available methods and their applicability to each content type:

| Content Category | Content Type | How to Control Access | | |
|------------------|--------------|--|---|------------------|
| | | Specify User or Group DN in SAS Login Definition | Use Personalization to Create Personal, Role-based, or Public Content | Specify ACI Rule |

| | | | | |
|------------------|---|----------|----------|----------|
| SAS | Publication Channel | | | x |
| | Package Published from Publication Channel | x | | x |
| | Report (Stored Process) | x | | x |
| | Warehouse View | | | x |
| | Table | x | | x |
| | Table Column | x | | x |
| | MDDB | x | | x |
| | webEIS Document | x | x | x |
| Non-SAS | Link | | x | x |
| | Application | | x | x |
| | Widget | | x | x |
| | Content Channel | | x | x |
| | Document | | | x |
| Portal Component | Window | | x | x |
| | List | | x | x |

Before using any of these methods, it is generally helpful to first organize the potential users of the portal into groups. Each group should contain users who have similar job functions and/or similar information needs. A user can be assigned to more than one group. For details on creating groups, see Defining Portal Groups (Roles).

Specifying User or Group DNs in SAS Login Definitions

To Control Access to SAS Data and Processes

For SAS data and processes, SAS Login definitions are the most efficient way to ensure that only authorized users obtain access through the portal. An added benefit of SAS Login definitions is that they also prevent unauthorized access from outside of the SAS Information Delivery Portal.

When a user attempts to view SAS content via the portal, the portal first attempts to create a workspace on the SAS server that contains the content. The workspace is created *only* if the SAS Login definition associated with the server contains the Distinguished Name (DN) of either the individual user or a group to which the user belongs. If the user's name or group is not specified in the SAS Login definition, an error message appears.

For information about creating a SAS Login, see Adding a SAS Login on the SAS Integration Technologies Web site.

To Control Access to SAS webEIS Documents and MDDBs

For SAS webEIS documents and MDDBs, SAS login definitions can also be used to provide credentials to the EIS access control system. Alternatively, you can prompt the user to enter EIS credentials. For implementation details, refer to the access control instructions in the topic Adding SAS webEIS Documents to the Portal.

Using Personalization to Create Personal, Role-based, or Public Content

For Portal content other than SAS data and processes, the most efficient way to control access is to use the portal's Personalization feature to create *personal*, *role-based*, or *public* content.

Personal content is content that can be accessed only by the user who added it to the portal. Through the portal's Personalization feature, any user can add personal webEIS documents, links, applications, widgets, content channels, windows, and lists to the portal.

Role-based content is content that can be accessed only by users who belong to a specific group. Role-based content can be added only by the user who is authorized as the owner of the group. Through the portal's Personalization feature, the group owner can add role-based webEIS documents, links, applications, widgets, content channels, windows, and lists to the portal.

Public content is content that can be accessed by any user, even a user who has not registered. In the default installation of the portal, public content can be added only by the public content administrator. In the default portal installation, the Portal.properties file assigns the public content administrator the user ID "portaladmin." Through the portal's Personalization feature, the public content administrator can add public webEIS documents, links, applications, widgets, content channels, windows, and lists to the portal.

For details on creating personal and role-based content, see Personalizing Your Portal in the SAS Information Delivery Portal User's Guide.

Specifying Access Control Information (ACI) Rules

The SAS Login and the portal's personalization feature will provide adequate security for many implementations of the SAS Information Delivery Portal. However, in some cases security may need to be implemented at a greater level of detail. By entering Access Control Information (ACI) rules into the metadata on the portal's enterprise directory, you can control security at virtually any level of granularity.

What Are ACI Rules?

ACI rules are statements that apply to specific objects on an LDAP server. Any number of rules can be entered for a given object. Each rule specifies:


- whether permission is allowed or denied;
- the specific rights that are allowed or denied (read, write, add, delete, search, compare, and/or selfwrite);
- the DN's of the users or groups of users to whom the rule applies.

If you are using the Sun ONE Directory Server or the Netscape Directory Server, you can refer to Netscape Directory Server Access Control Overview on the SAS Integration Technologies Web site for more information about ACI rules. If you are using the IBM SecureWay server, you can refer to SecureWay Directory Server Access Control Overview.

How to Enter ACI Rules

To enter ACI rules for SAS processes and data, you can use either the SAS Integration Technologies (IT) Administrator or the enterprise directory console. To enter ACI rules for other types of content, you must use the enterprise directory console.

To enter ACI rules in IT Administrator:

1. Display the tree view of the appropriate directory.
2. Select the folder or object for which you want to create a rule. (If you select a folder, the rule will affect all of the objects in the folder.)
3. Select the Set Access Permissions tool () in the toolbar.
4. Enter the rule information in the dialog boxes provided.

For more information, see Integration Technologies Administrator and Setting Access Permissions for an Object in the SAS Integration Technologies Web site.

IT Administrator allows you to enter ACI rules for any of the following objects:

- Components of the SAS Publish Framework, including:
 - Publication channels
 - Packages that have been published through a publication channel
 - Archive paths in which published packages are stored
- SAS configuration components, including:
 - SAS servers
 - SAS spawners

- SAS logins
- SAS stored processes, including:
 - Stored processes themselves
 - Paths in which stored processes reside
 - Packages produced through stored processes
- SAS data sources, including
 - SAS tables (including tables that are part of a warehouse view)
 - Columns within a SAS table
 - SAS Multidimensional databases (MDDBs), including MDDBs that are part of a warehouse view
 - SAS libraries (which can contain multiple tables or MDDBs)

To enter ACI rules for other types of portal content, you must use the enterprise directory console.

Access Control Overview for the SecureWay Directory Server

When Lightweight Directory Access Protocol (LDAP) was first developed, it was only a protocol for accessing data in an X500 directory server. Therefore, many specifics of how the server itself was supposed to work were left out. Even when the new standard was written for LDAP version 3 (RFC2251), some important issues were left out simply because the people involved had already picked an implementation, and didn't want a new standard to force them to redesign and reimplement large portions of their servers. Access control was one of those important issues. That is why when discussing access control, it is important to remember that each vendor has a different mechanism, and very little is portable from one server to another.

Access control starts with authentication. There are several mechanisms to accomplish the authentication, but all must eventually resolve to a distinguished name (DN) that exists in the directory. This distinguished name is then used to determine the access that is granted to a user. The process of associating a distinguished name with a user is called binding. A user can bind to a server using the DN and a password, or they can bind anonymously, providing no credentials.

Authentication is accomplished by using permissions for objects in the IBM Secureway directory. Using these permissions, you can allow or deny access to objects or groups of objects by users or classes of users. For information on using the Integration Technologies Administrator to set access permissions, see [Setting Access Control for an Object](#) on the SAS Integration Technologies Web site.

SecureWay Syntax

There are four attribute types which determine the access that is allowed on an entry:

| Attribute | Definition |
|----------------|--|
| AclEntry | A multivalue attribute that describes access to attributes of the associated LDAP object, as well as permissions on the object itself. |
| AclPropagate | A "true" or "false" flag that indicates if this particular ACL should be propagated down the directory hierarchy. |
| EntryOwner | The owner of this particular directory object. The entryOwner receives complete access to all attributes of the object. |
| OwnerPropagate | A "true" or "false" flag that indicates if this owner should be propagated down the directory hierarchy. |

These attributes can only be modified by the entry owner, or the directory administrator. There are two other attributes which are maintained by the server, and are not user modifiable, but are available to read for informational purposes:

| Attribute | Definition |
|-------------|---|
| AclSource | An attribute which identifies the directory object from which the ACL information is inherited. |
| OwnerSource | An attribute which identified the directory object from which the owner information is inherited. |

AclEntry

The aclEntry attribute describes the access granted to the entry object. It describes who has rights (the subject), what rights they have to the object itself, and what rights they have to the attributes of the object. The format of the aclEntry attribute is:

```
<subject-type>:<subjectDN>:<object-access>:<attribute-access>
```

| | |
|------------------|---|
| Subject-type | one of access-id, group, or role. If access-id, then subjectDN should be the DN of a user entry. If group, subjectDN should be the DN of an AccessGroup entry, and role should point to an AccessRole entry. |
| SubjectDn | The Dn of the subject for the aclEntry. |
| Object-access | The access rights granted to the object itself. Valid permissions are a to allow the subject to add children under the entry, and d to allow delete permission. The format for object access is "object:permissions". |
| Attribute-access | Specifies the permissions granted to the entry attributes. There are three levels of attribute access: normal, sensitive, and critical. The security level for an attribute is defined in the schema. |

AclPropagate

This attribute will have a value of "true" or "false". The default is "true", and indicates that the aclEntry values for this entry should propagate down the hierarchy to apply to any entries below it which don't have their own aclEntry attribute(s).

EntryOwner

Like the subject clause of the aclEntry, the entryOwner has a subject type and a subjectDN. The subject type can be access-id, group, or role. The subjectDN should be the distinguished name of an entity that represents the correct type of entry (person, accessGroup, or accessRole).

Example: access-id:cn=Portal User,ou=People,o=SAS Institute,c=US

OwnerPropagate

A value of "true" or "false" that determines whether the entryOwner value will propagate to down the hierarchy to apply to entries below it.

ACI Rule Considerations

AclEntry attributes in SecureWay propagate down (assuming aclPropagate is true) to all its subordinates until the aclEntry is overridden. Any aclEntry will override all the values of the previous aclSource. That is to say, if you want to add access by another individual or group at a given point in the tree, while retaining the access controls specified higher up, you must copy the aclEntry attribute values that the entry is already inheriting, and add the new values. Just creating an aclEntry with the new value will revoke the access provided by the previous aclSource.

Access Control Examples

AclEntry: access-id:cn=Portal User,ou=People,o=SAS Institute,c=US:object:ad:normal:r

This example allows the user with a DN of `cn=Portal User,ou=People,o=SAS Institute,c=US` all access to everything in the directory. This level of access is unusual, but acceptable for this example because the Portal User identity is used by an application that performs very specific operations in the directory.

```
AclEntry: group:cn=Sales,ou=Groups,o=SAS Institute,c=US:object:a:normal:rsc
```

This example allows access to the group: `cn=Sales,ou=Groups,o=SAS Institute,c=US`. The permissions allow read, search, compare, and add. The add permission is important because it allows members of these groups can create archive entries under the channel Orders for Manufacturing Materials. If the administrator wished, the add permission could have been moved down to the archivepath entry. This would have restricted who could publish archives to be stored under the channel.

SAS application requirements

There are several places in the SAS hierarchy that do not require any special access control. There are other places, however, that require careful thought when applying the ACI rule. The following are the places in the hierarchy that require special attention.

Cn=SAS

This is the top of the SAS application tree. Special permissions for an administrative user, or application identity (like Portal User) can go here, if desired.

Cn=sasSubscribers,sascomponent=sasPublishSubscribe

No sensitive data is contained in this part of the tree, so allowing public read access is acceptable. It's also good to provide a way for users to update their own subscriber entries. The only way to do this is to make the personDn the entryOwner for each entry, or put an aclEntry on each subscriber granting write access to the person.

Cn=saschannels,sascomponent=sasPublishSubscribe

The level of control you implement here depends on how you want to secure channels, archive paths and archives. The way publishing works, if a user can read a channel entry, they can publish TO_SUBSCRIBERS, which sends email to all of the users subscribed to a channel. This is allowed even if the publishing user is not subscribed to the channel and therefore cannot write to the channel. This situation can be covered by only allowing read access to specific groups. The other consideration is how archive paths are secured. Allowing add access to an archive path means a user can create archives under that path (assuming they have permission on the physical path).

Cn=sasArchivePaths,sascomponent=Archiving

This area has the same considerations as the channels. To define the ACI rule, you must decide how you want to protect the global archive paths.

Cn=saslogins,sascomponent=sasserver

This requires careful consideration, because the saslogins below this container contain user names and passwords. These attributes are defined as being critical, but the appropriate user needs to be able to read those attributes in order to access their data. The best answer is to make the entryOwner of individual logins the individual they belong to, and for group logins, set the aclEntry to allow access to critical attributes by the appropriate group, and set the owner to a group administrator.

Cn=sasStoredProcessPaths,sascomponent=sasApplications

It is likely that each storedprocesspath entry will have its own ACI rule set. This is because the stored processes will generate information that will be intended for a certain audience. The stored processes should be grouped under a sasstoredprocesspath according to the group that needs access to them.

Sascomponent=sasPortal

The SAS Business Information Portal Installation Guide contains guidelines for setting the access controls on portal entries.

Sascomponent=sasDataSources

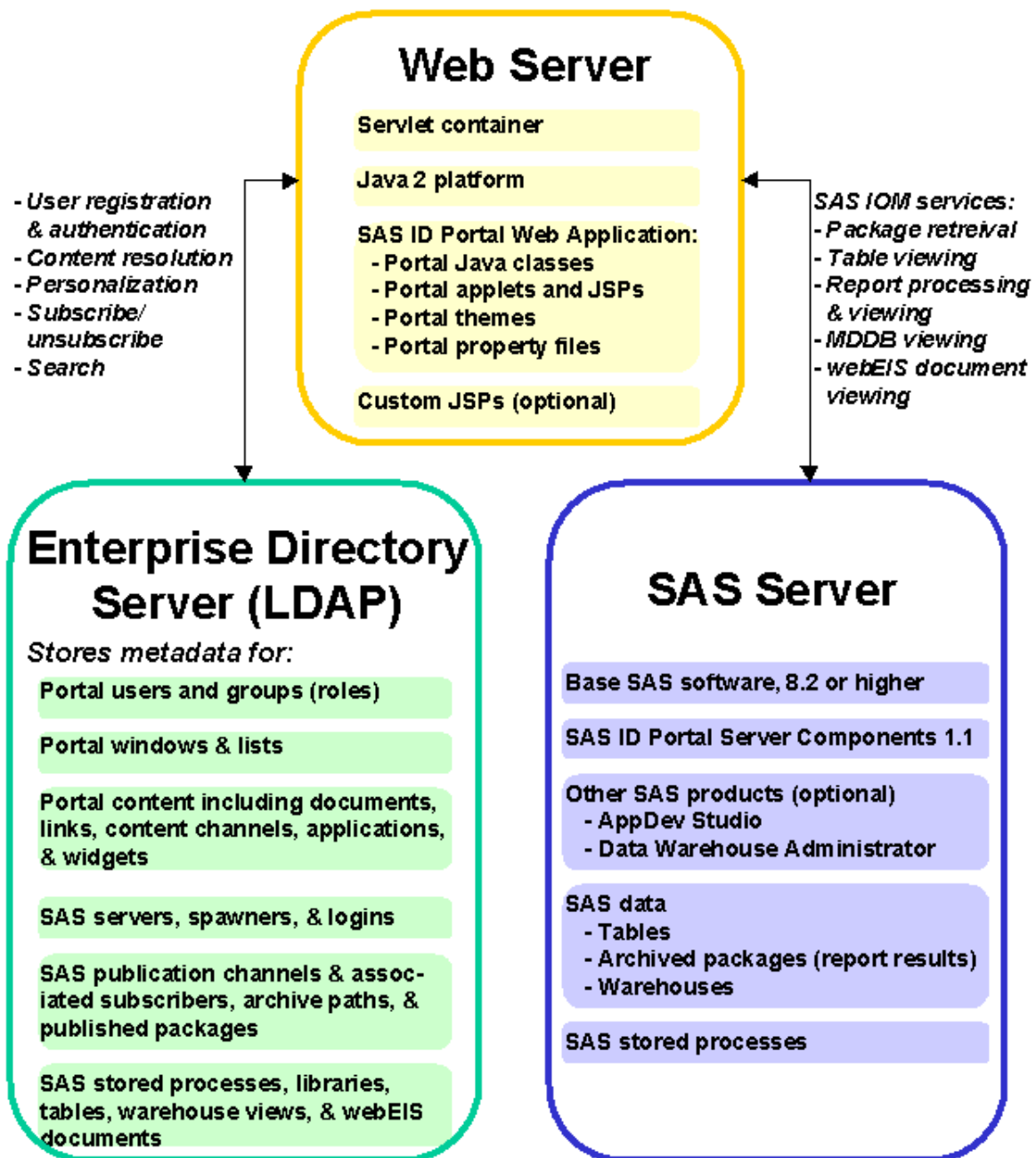
This is another location that requires careful consideration. Libraries, tables, infomarts and other data sources may all have individual security requirements.

Adding SAS[®] Content

The SAS Information Delivery Portal exploits the analytical and reporting powers of SAS by allowing you to deliver SAS data to the desktop of portal users. Through the portal, authorized users can perform the following functions from client machines that have only a Web browser installed:

- View SAS tables
- Run SAS reports, known as "stored processes"
- Subscribe to SAS publication channels, and view packages that are published to these channels
- View SAS data warehouses
- View SAS webEIS documents

The following diagram shows how the SAS Information Delivery Portal web application, residing on a web server, interacts with the enterprise directory server and the SAS IOM server to provide this functionality for multiple users:



Before you can add SAS data and functionality to the SAS Information Delivery Portal, the following prerequisites must be in place:

1. **The following software must be installed on the SAS server:**

- Base SAS software, Release 8.2 or higher
- SAS Information Delivery Portal Server Components
- Any other SAS products needed to support content types that you wish to host in the portal

(for example, AppDev Studio for webEIS documents and Warehouse Administrator for warehouse views).

2. **The following metadata must be present on the enterprise directory server.** To manage these definitions and to create new ones, you can use the Integration Technologies Administrator application. For details, refer to the SAS Integration Technologies Administrator page on the SAS Integration Technologies Web site. The required definitions are as follows:

- **SAS server definition.** A Portal Demo Server is automatically defined when you follow the portal's installation process. The metadata for a SAS server includes:

- Server name
- Machine name
- Logical name. The portal uses the logical name to associate the servers to related objects on the enterprise directory. Be sure to use the same logical name when you define spawners, logins, and libraries for this server.
- Connection information, including domain name and protocol
- Other information including encryption, commands, and workspace pooling, as required.

For an example of the process for creating a server definition, refer to the SAS Server Metadata Example.

- **SAS spawner definition.** The IOM object spawner is a daemon that listens for incoming client requests for IOM services. When the daemon receives a request from a new client, it launches an instance of an IOM server to fulfill the request. A Portal Demo Spawner is automatically defined when you follow the portal's installation process. The spawner definition should include the following:

- Spawner name
- Machine name
- Logical name (this must match the logical name of the associated server)

Instructions for administering and starting an object spawner are available from the Starting and Administering an Object Spawner page on the SAS Integration Technologies Web site. For an example of the process for creating a spawner definition, refer to the SAS Spawner Metadata Example.

- **SAS login definition.** SAS logins are a convenient method for providing the credentials necessary for a spawner to start a SAS session. Using logins, the spawner can create SAS sessions without having to keep track of the specifics of a user's ID and permissions. A Portal Demo Login is automatically defined when you follow the portal's installation

process. The login definition should include:

- Login name
- Person reference, including the distinguished names of individual portal users or groups of users who will be allowed to access the SAS server through the portal
- Login, consisting of the SAS login and password needed to access to SAS server
- Domain name
- Logical name (this must match the logical name of the associated server)

For an example of the process for creating a login definition, refer to the SAS Logins Metadata Example.

- **The IOM Object Spawner must be running on the SAS server.** Instructions for administering and starting an object spawner are available from the Starting and Administering an Object Spawner page on the SAS Integration Technologies Web site. The following is an example of a spawner command that works with the SAS System, Release 8.2 and runs the Portal Demo on the Windows platform:

```
del /q c:\temp\obj.log
"c:\program files\sas institute\sas\v8\inttech\sasexe\objspawn"
  -sasverbose -saslogfile c:\temp\obj.log
  -ldaphost ldapsrvr.ace.com -ldapport 389
  -ldapbase "sascomponent=sasServer,cn=SAS,o=ACE Industries,c=US"
  -ldap_binddn "cn=Portal User,ou=People,o=ACE Industries,c=US"
  -ldap_bindpw portall -sasspawnercn "Portal Demo"
```

The following is an example of a spawner command that works with the SAS System, Release 8.2 and runs the Portal Demo on the UNIX platform:

```
rm /tmp/obj.log
/sas/inttech/sasexe/objspawn
  -sasverbose -saslogfile /tmp/obj.log
  -ldaphost ldapsrvr.ace.com -ldapport 389
  -ldapbase "sascomponent=sasServer,cn=SAS,o=ACE Industries,c=US"
  -ldap_binddn "cn=Portal User,ou=People,o=ACE Industries,c=US"
  -ldap_bindpw portall -sasspawnercn "Portal Demo"
```

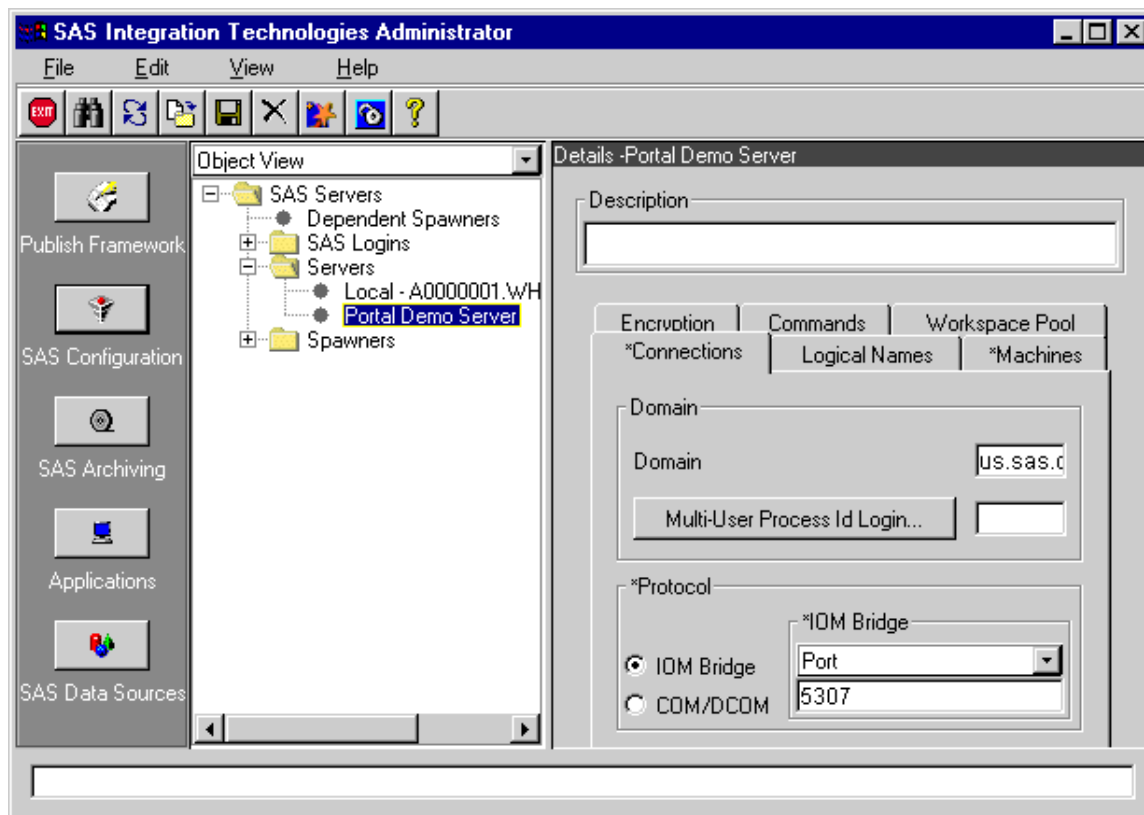
When the above prerequisites are in place, you can proceed to add SAS content as described in the following topics:

- Adding SAS Tables
 - Adding Stored Processes
 - Adding SAS Publication Channels
 - Adding Warehouse Data
 - Adding webEIS Documents
-

SAS[®] Server Metadata Example

This page provides an example of the metadata for a SAS server.

To view the demo server's metadata, open the IT Administrator interface, select the **SAS Configuration** button, and select the **Server** entry, as shown below.



To add an additional server, select **File, New, Server** from the menu bar and make entries in the following dialog box:

The screenshot shows a 'New Server' dialog box with the following fields and controls:

- *Name:** An empty text input field.
- Description:** An empty text input field.
- Encryption:** A tabbed section with three sub-sections:
 - *Connections:** A sub-section with a 'Multi-User Process Id Login...' button and an empty text input field.
 - Commands:** A sub-section with a 'Logical Names' label.
 - Workspace Pool:** A sub-section with a '*Machines' label.
- Domain:** A text input field containing 'us.sas.com'.
- *Protocol:** A section with two radio buttons:
 - IOM Bridge: Includes a dropdown menu showing 'Port' and a text input field containing '5307'.
 - COM/DCOM
- Buttons:** 'OK', 'Cancel', and 'Help' buttons at the bottom.

Domain:

The server's domain. In order for a spawner to work with the server, the spawner must be defined for the same domain (in addition to having matching logical names).

Protocol:

IOM Bridge or COM/DCOM protocols, as well applicable service or port IDs.

Logical name:

All logical names under which this server can operate.

Machine:

All machines on which the server can run.

Encryption:

Client and server algorithms, what content to encrypt.

SAS logins:

The SAS logins available to start a SAS session.

Commands:

The command to start the SAS session on the server.

Maximum workspaces per pool:

The maximum number of workspaces that will be available for any workspace pool that is established with the server.

For more information on each input field, select the **Help** button on the dialog box.

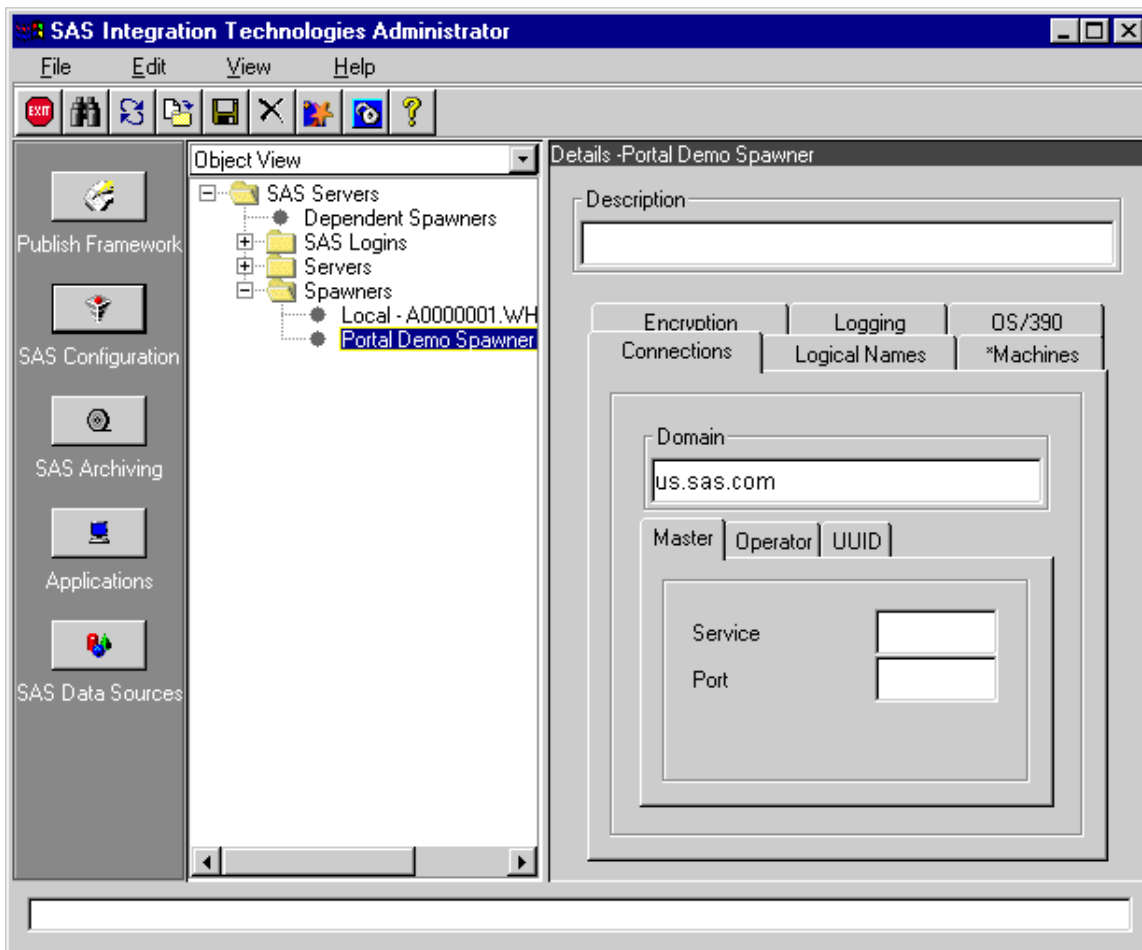
Required Fields

Server name, machine name, connections(domain and protocol), and logical name are required fields. Completion of the description field is optional but strongly recommended.

SAS[®] Spawner Metadata Example

This page provides an example of the metadata for a SAS spawner.

To view the demo server's metadata, open the IT Administrator interface, select the **SAS Configuration** button, and select the **Spawner** entry, as shown below.



To add an additional spawner, select **File, New, Spawner** from the menu bar and make entries in the

following dialog box:

The image shows a 'New Spawner' dialog box with the following fields and tabs:

- *Name
- Description
- *Machines
- Encryption
- Logging
- OS/390
- Connections
- Logical Names
- Domain (us.sas.com)
- Master
- Operator
- UUID
- Service
- Port
- OK
- Cancel
- Help

Domain:

The spawner's domain. In order for a spawner to work with the server, the spawner must be defined for the same domain (in addition to having matching logical names).

Protocol:

IOM Bridge or COM/DCOM protocols, as well applicable service or port IDs.

Logical name:

All logical names under which this spawner can operate.

Connection information:

Service, port and passwords for master, operator, and UUID connections

Machines:

All machines on which the spawner can run.

Encryption:

The modules path and key length for encryption

Logging:

The path to the log file and whether to use verbose logging

OS/390:

The OS/390 logical unit name

For more information on each input field, select the **Help** button on the dialog box.

Required Fields

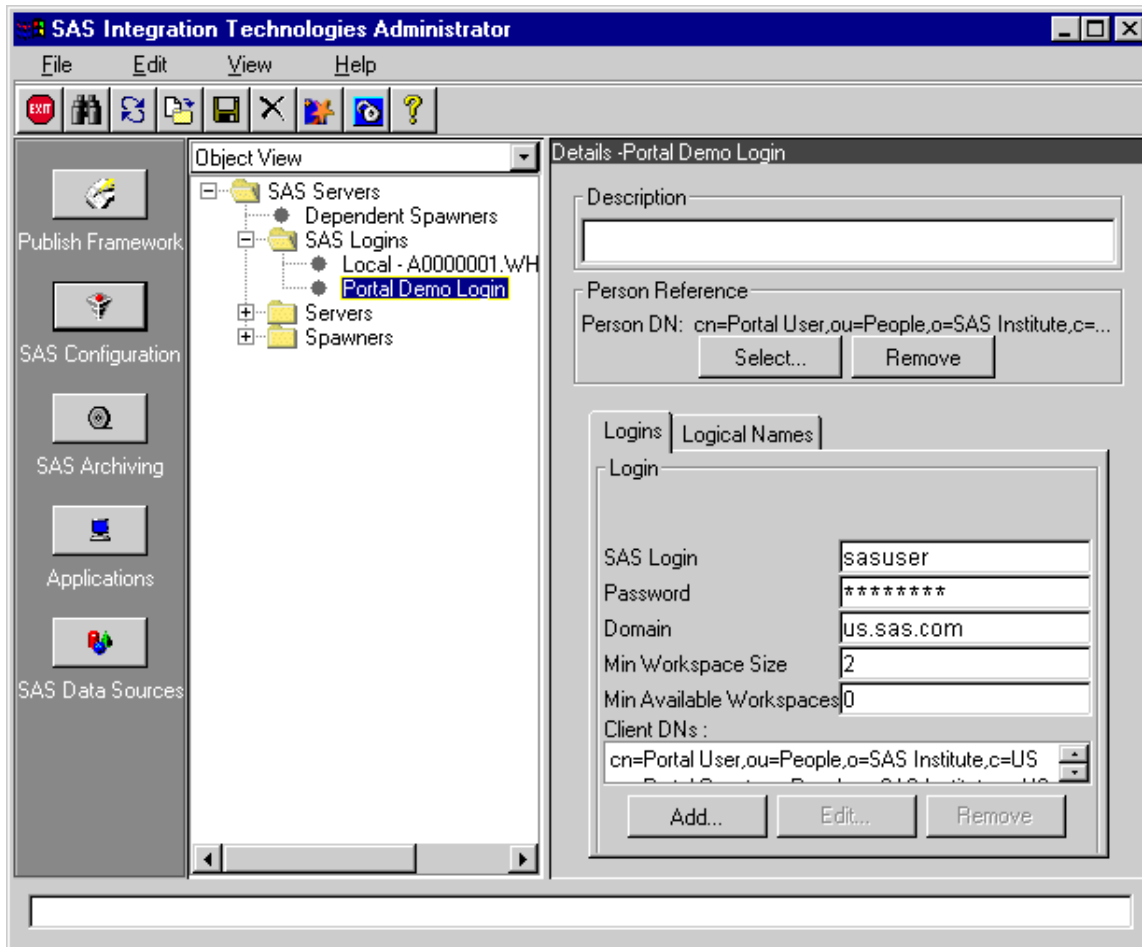
Spawner name, machine name, and logical names are the only fields that are required. Completion of the description field is optional but strongly recommended.

SAS[®] Login Metadata Example

This page provides examples of metadata for a SAS login. The examples show the metadata for Portal Demo Login, which is installed automatically in the demo portal. The Portal Demo Login contains the information needed to gain access to the portal demo.

To view the demo server's metadata, open the IT Administrator interface, select the **SAS Configuration** button, and select the **SAS Login** entry, as shown below.

When you install the demo version of the portal, **Portal Demo Login** is automatically defined in the directory. The metadata for the archive path includes the complete path relative to the root of the SAS server where published packages are to be stored. The portal's installation process initializes the path to the value you provide in the the \$SAS_DEMO_DIR\$: parameter of the Install.properties file.



To change the SAS login metadata or to add additional SAS logins, select **File, New, SAS Login** from the menu bar and make entries in the following dialog box:

Name:

The name of the user.

Description:

A description for the user (optional).

Person Reference:

The person reference in the LDAP directory for the user. Select the Select button to display the Person Index window, which contains a tabbed list of all the person references in the LDAP directory. Select the appropriate reference and select OK to include the selected person reference in the user definition. Press the Ctrl key and click the mouse button to deselect an item.

SAS Login:

The user's login ID. The login ID must be valid for the server on which you are going to establish a SAS session.

Password:

The user's login password for starting a SAS session. The password must be valid for the server on which you are going to establish a SAS session.

Domain:

The domain where the login resides. The domain name specified in this window must exactly match the domain name of the server on which you are going to establish a SAS session.

Logical Name:

If you are setting up a workspace pool, specify the logical name of the SAS server with which this

login is associated for workspace pooling.

Client DN:

The distinguished name of a user or a group entry that will be used to allow entry to a workspace pool.

Min Workspace Size:

Specify the number of workspaces currently serving or waiting to service a request.

Min Available Workspaces:

The number of workspaces waiting to service a request.

For more information on each input field, select the **Help** button on the dialog box.

Required Fields

All fields except description are required. It is strongly recommended that you fill in the description field

Adding SAS[®] Tables to the Portal

Users of the SAS Information Delivery Portal can view SAS tables using the Portal Table Viewer. The Table Viewer presents an input form that allows the user to specify which table columns to display and the order in which they are to appear. In addition, the user can enter a Standard Query Language (SQL) WHERE expression to specify which rows of the table to display. The portal application then renders the table based on these criteria.

To set up a table in the portal, you must:

1. Create the table and place it on a SAS server that is accessible to the portal users who will access it.
2. Specify the necessary format catalog libref(s) in the SAS configuration file if any of the columns in the table has a user-defined format.
3. Update the portal application's enterprise directory with metadata about the table and its columns.
4. Use the Personalize tool to make the table appear as a selection on the portal.

Step 1: Create the Table and Place it on a SAS Server

First, use SAS to create the table. A table is any SAS data set that is made up of rows and columns. Place the table on a server that the portal application can access. The server must be defined on the portal application's enterprise directory.

Step 2: Specify Format Catalog Librefs in the SAS Configuration File

If any of the columns in the table has a user-defined format, you must specify the necessary format catalogs on the SAS server as follows:

- Set environment variables to specify the librefs that contain the format catalogs. To set the environment variables on a Unix or Windows NT server, use the -SET option in the SAS configuration file. On OS/390 hosts, use a DD statement in your job control language (JCL). On VMS hosts, use a DCL command. For more information, see the SAS documentation for your operating environment.

Here is an example of a statement to specify a format catalog environment variable on Windows NT:

```
-set userfmts 'c:\mysaslibs\formats'
```

Note: Do not set the environment variables in an AUTOEXEC file, since the IOM server does not process the AUTOEXEC.

- After you have set the environment variables, place a FMTSEARCH= option in the SAS configuration file. This option should list the librefs that you specified in the environment variables. The portal will use this list to locate the format catalogs. The FMTSEARCH= option has the following syntax:

```
FMTSEARCH= (libref-1 libref-2... libref-n)
```

The value of *libref* can be either a libref or a libref.catalog. If only the libref is given, SAS assumes that FORMATS is the catalog name. The order of librefs in the list determines the order in which the catalogs will be searched. The WORK.FORMATS catalog is always searched first, unless it appears in the FMTSEARCH= list. The LIBRARY.FORMATS catalog is searched after WORK.FORMATS and before anything else in the FMTSEARCH= list, unless it appears in the FMTSEARCH= list.

Here is an example of a FMTSEARCH= statement that lists the format catalog specified in the preceding example:

```
-fmtsearch=(userfmts)
```

For more information about the FMTSEARCH= option, see "Using SAS System Options" in the *SAS Language Reference: Dictionary*. For more information about creating SAS configuration files, see the SAS documentation for your operating environment.

Be sure to keep the SAS configuration file updated with environment variables and FMTSEARCH= statements to define the necessary format librefs for all tables in the portal.

Step 3: Update the Enterprise Directory with Metadata about the Table and its Columns

Use the SAS Integration Technologies (IT) Administrator interface to update your enterprise directory

with metadata about the table and each of the table's columns. For detailed instructions on the use of this interface, refer to the IT Administrator section of the SAS Integration Technologies Web site.

Perform the following tasks using IT Administrator. Refer to the Table Metadata Example for an annotated illustration of these tasks. The links below point to specific portions of the example.

- Define the SAS library in the enterprise directory (if it is not already defined), as shown in this example. The library information will enable the portal application to locate the table when it is invoked by the user. The logical name that you assign to the library identifies the SAS server on which the table is located.
- Define the table in the enterprise directory, as shown in this example. The member name and library name that you enter will enable the portal application to locate the table when it is invoked by the user.
- Define each of the table's columns in the enterprise directory, as shown in this example. The Portal Table Viewer will use the column information to populate the default user input form so that the user can choose which columns he or she wishes to display.
- Define access rules for the table and/or its individual columns. For more information, refer to [Controlling Access to Portal Content](#)

Step 4: Use the Personalize Tool to Make the Table Appear as a Selection on the Portal

Depending on who has access permission to the stored process, you can use one of several methods to make a table appear as a selection on the portal:

- If the table is accessible to all portal users, the public content administrator can use the Personalize feature to add the table to a public list. When the administrator performs this function, the portal application displays a list box containing all tables in the enterprise directory that are accessible to all portal users. From the list box, the public content administrator selects the tables that are to appear on the public list.
 - Group owners can use the Personalize feature to add the table to a role-based list. When the group owner performs this function, the portal application displays a list box containing all tables in the enterprise directory that the group is authorized to access. From the list box, the group owner selects the tables that are to appear on the role-based list.
 - Individual users can use the Personalize feature to add the table to a personal list. When the user performs this function, the portal application displays a list box containing all tables in the enterprise directory that the user is authorized to access. From the list box, the user selects the tables that are to appear on his or her personal list.
-

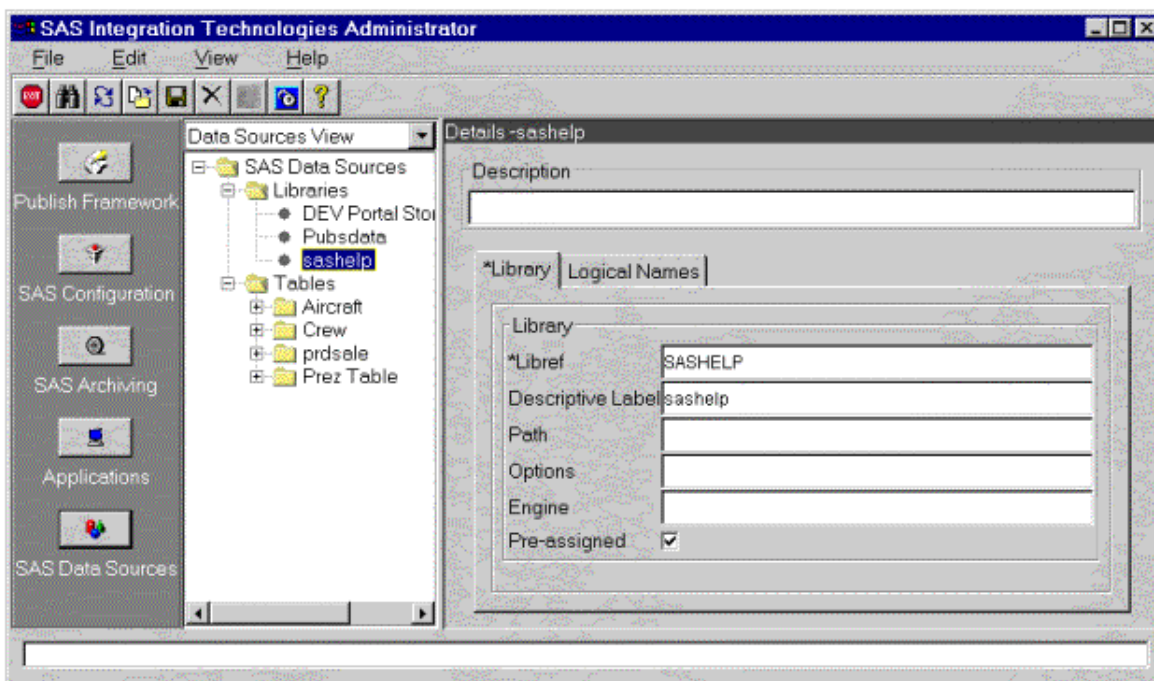
Table Metadata Example

This page provides examples of metadata that has been defined for a SAS table in the SAS Information Delivery Portal. The example metadata describes the table prdsale, which is installed automatically in the demo portal. You can find the data set containing this table in the "SASDemos" directory within the portal's installation directory. When you log on to the portal with the user name "portaldemo," the table appears in the Data Sources window as "Product Sales Summary."

Defining the Library

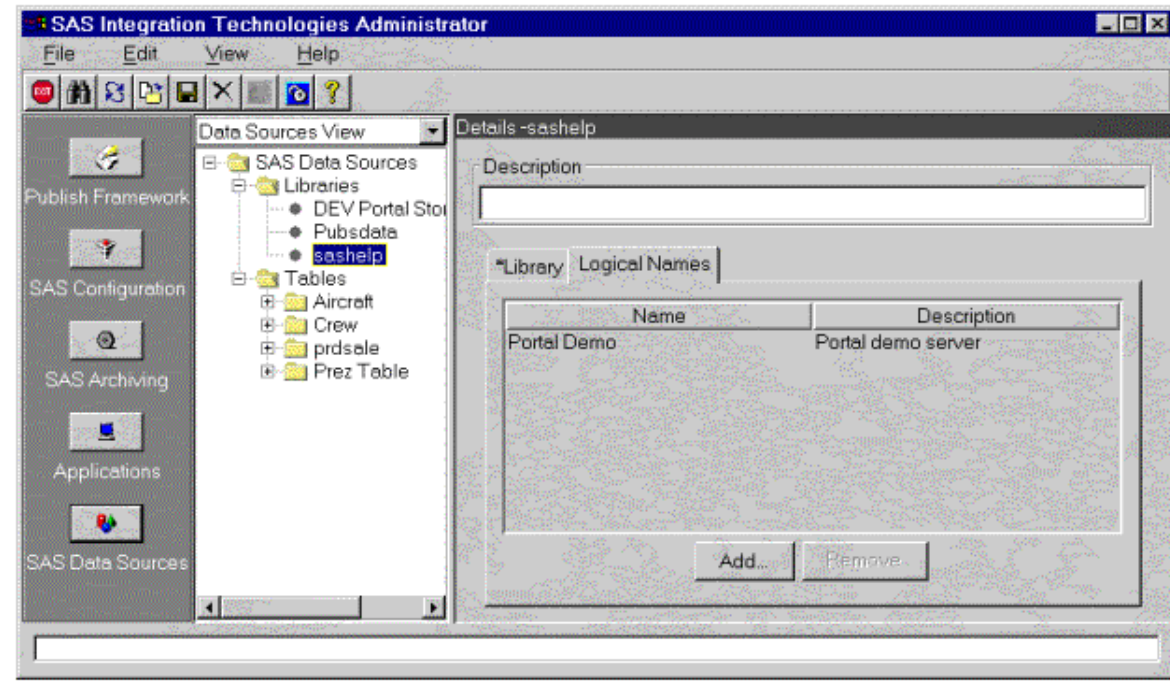
To define a table, you must first update the enterprise directory with metadata identifying the SAS library where the table resides. When you install the demo version of the portal, a library called "sashelp" is automatically defined.

To view the metadata for the sashelp library, open the IT Administrator interface, select the **SAS Data Sources** button, and select the **sashelp** entry under Libraries, as shown below.



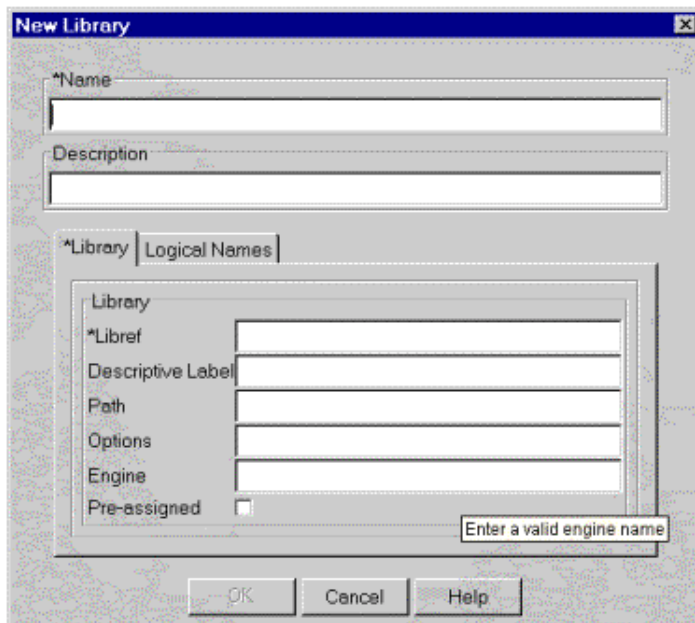
In this case, the library is predefined in the SAS server. If the library is not predefined, you must enter a path relative to the root of the SAS server on which the library resides.

To see the logical name for the sashelp library, click on the Logical Names tab:



The logical name identifies the server on which the library resides. (For information about the SAS servers defined in your enterprise directory, select the SAS Configuration button.)

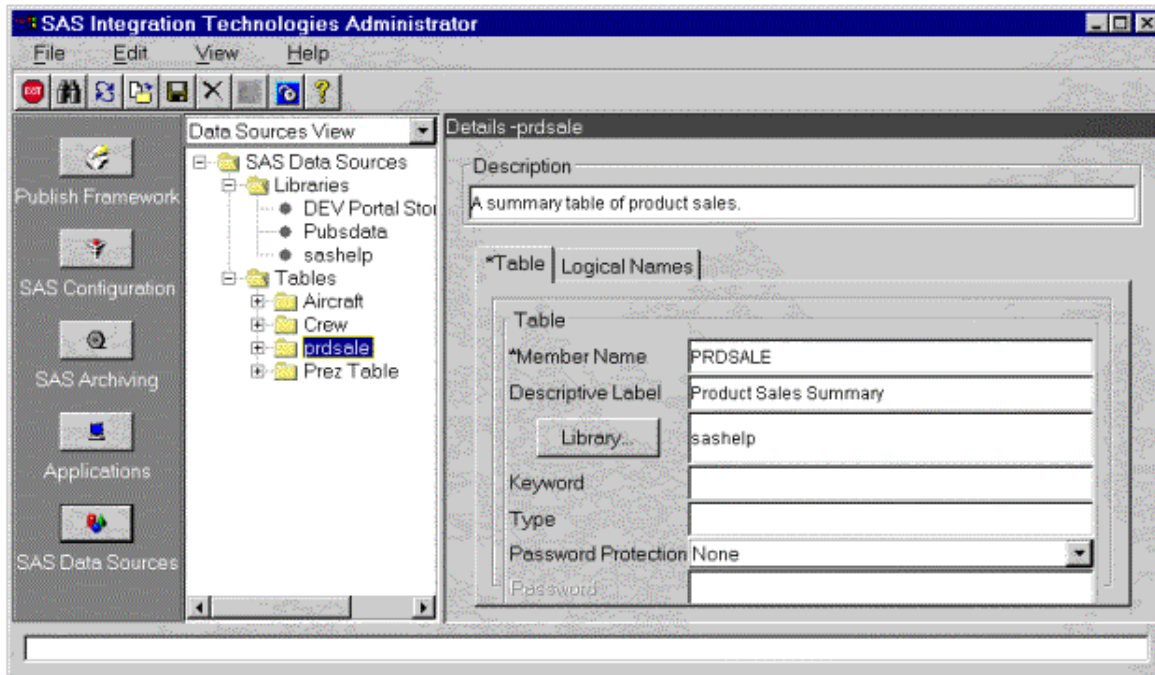
If you want to define another library, choose **File, New, Library** from the menu bar and enter the library information in the following dialog box:



For a complete description of the input fields, select the **Help** button on the dialog box.

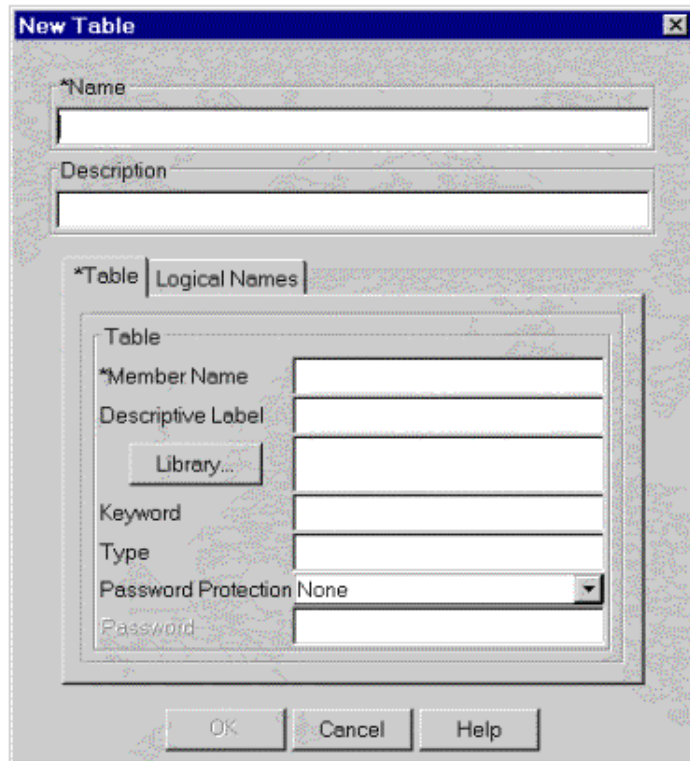
Defining the Table

Next, you must update the enterprise directory with metadata to define the table itself. In IT Administrator, you can view the metadata for the table **prdsale** by selecting **prdsale** in the tree under **Tables**, as shown below.



The metadata for the library includes a description, the member name (the data set name *without* the extension), a descriptive label to identify the table on lists in the portal, and the library in which the table resides. You can optionally enter keywords, a table type, and password protection criteria.

If you want to define another table, choose **File, New, Table** from the menu bar. Then enter the table information in the following dialog box:

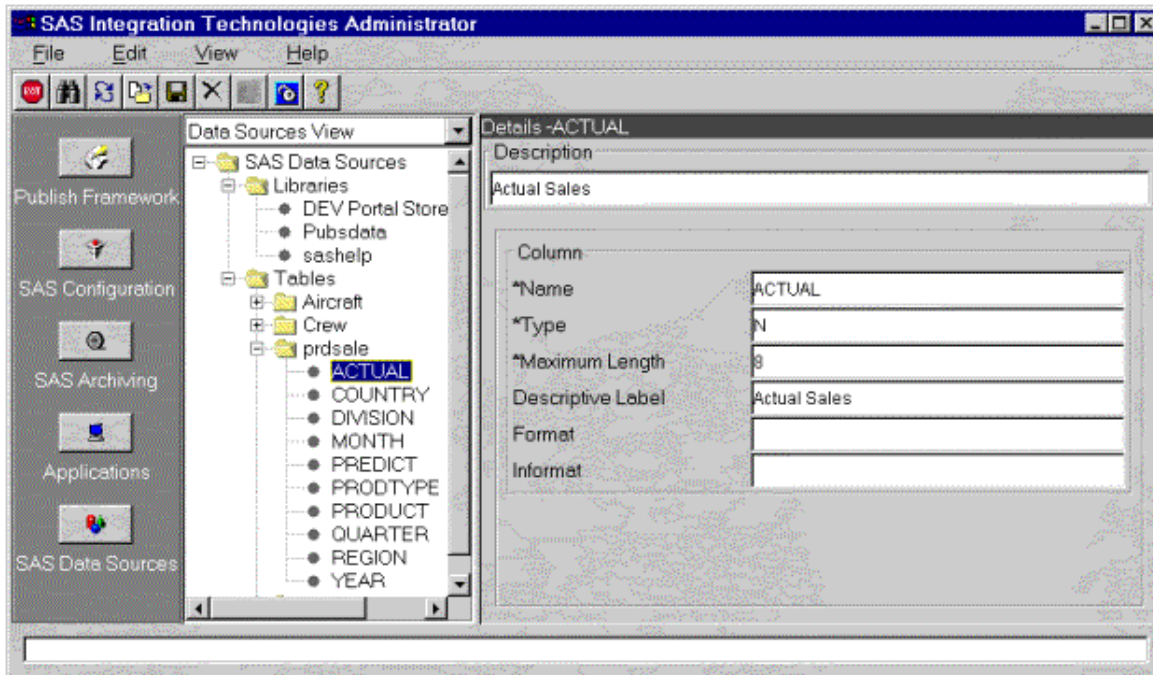


For a complete description of the input fields, select the **Help** button on the dialog box.

Defining the Columns

After defining the table, you must define the columns that the table contains. The Portal Table Viewer will use the column information to populate the default user input form so that the user can choose which columns he or she wishes to display.

In IT Administrator, you can view the metadata for the columns in the table `prdsale` by viewing each entry in the tree under the **prdsale** table definition.



The metadata for each column includes the column name (as specified in the SAS program that created the table), the data type, the maximum length, and a descriptive label to appear in the Portal Table Viewer.

To add new column definitions for a table, select the table entry in the tree and choose **File, New, Column**. Then enter the column information in the following dialog box:

The image shows a 'New Column' dialog box. It has a title bar with the text 'New Column' and a close button (X). Below the title bar are three main sections: a '*Name' text field, a 'Description' text area, and a 'Column' section. The 'Column' section contains six fields: '*Name', '*Type', '*Maximum Length', 'Descriptive Label', 'Format', and 'Informat'. At the bottom of the dialog are three buttons: 'OK', 'Cancel', and 'Help'.

For a complete description of the input fields, select the **Help** button on the dialog box.

Adding Stored Processes to the Portal

SAS stored processes give users of the SAS Information Delivery Portal the ability to run SAS reports dynamically using current data residing on enterprise databases.

A stored process is a SAS program that resides on a server and is available to be executed on a request basis. The benefits of stored processes include centralized code management, increased security, and ad hoc reporting capabilities. For more information, see the Stored Processes topic on the SAS Integration Technologies Web site.

When a user runs a stored process from the portal, the Portal Report Runner displays the associated input form, allowing the user to filter the report contents. The user can have the report runner display the report results immediately, or the results can be stored for later viewing by the requesting user.

To set up a stored process in the portal, you must:

1. Develop and test the SAS code.

2. Place the code and data on a SAS server that is accessible to the portal users who will access it.
3. Update the portal application's enterprise directory with metadata about the stored process and its application parameters.
4. Use the Personalize tool to make the report appear as a selection on the portal.

Step 1: Develop and Test the SAS Code

First, determine what type of report you would like to make available through the stored process. Then, either locate an existing SAS program that produces the report or develop the program from scratch. Test the program on a standalone basis to be sure that it operates without errors and that it produces the desired output.

You must then add the following elements to the SAS program so that it will function as a stored process. Refer to the example stored process for an annotated illustration of these requirements. The links below point to specific portions of the example.

- Add a prologue to the beginning of the program to declare the variables that are to be used as execution parameters. The program can then use these variables as filters.
- Add a *ProcessBody line to delineate the prologue from the body of the program.
- Create the program's output so that it can be presented within the portal.
- Use the Publish Framework to place the report results in a package and publish the package to the portal.

Step 2: Place the Stored Process on a SAS Server

Place the stored process on a server that the portal application can access. The server must be defined on the portal application's enterprise directory.

Step 3: Update the Enterprise Directory with Metadata about the Stored Process

Use the SAS Integration Technologies (IT) Administrator interface to update your enterprise directory with metadata about the stored process and its execution parameters. For detailed instructions on the use of this interface, refer to the IT Administrator section of the SAS Integration Technologies Web site.

Perform the following tasks using IT Administrator. Refer to the example stored process definition for an annotated illustration of these tasks. The links below point to specific portions of the example.

- Define a path for the stored process, if one is not already defined. The path information will enable the portal application to locate the stored process when it is invoked by the user. The logical name that you assign to the path identifies the SAS server on which the stored process is located.
- Define the stored process. The stored process definition includes:

- A descriptive label, which will appear on lists in the portal application.
- The name of the SAS program that contains the stored process.

Note: If you use an LDIF file or other LDAP tool to enter your metadata, you must append the program name to the value of the descriptive label attribute, described above, followed by an underscore character, before assigning it to the `sasstoredprocessvalue` attribute. For example, if the program file name is **prdsale1.sas** and the descriptive label is **Product Sales Demo**, you would place the value **Product Sales Demo_prdsale1.sas** in the metadata.

If you use IT Administrator, the required concatenation occurs automatically.

- The path to the JSP file that contains the input form to be displayed in the portal for execution parameter selection. The JSP must be located within the ServletContext of the portal application.

For convenience, the portal application includes a default input form (`DefaultInputForm.jsp`) which automatically renders your stored process parameters as list boxes in the portal. You can choose to use this JSP, or you can create a JSP that renders your own customized input form.

- The parameters that are passed to the SAS program upon execution. These are the same parameters that you defined in the prologue section of the SAS program.

For each parameter, you must define a name, a label (to appear on the user input form), and the valid values. You can define the valid values in either of two ways:

- by providing a comma-separated list of values; or
- by providing the path to a data source that contains the valid values. If you use this method, you must also define a report query to extract the valid values from the data source.

When portal displays the default input form in the Portal Report Runner window, it will render the possible values in list boxes. You can also provide a description, a default value, and a flag indicating whether the parameter is required.

- Define access rules for the stored process and/or its path. For more information, refer to [Controlling Access to Portal Content](#).

Step 4: Use the Personalize Tool to Make the Report Appear as a Selection on the Portal

Depending on who has access permission to the stored process, you can use one of several methods to make a stored process appear as a report selection on the portal:

- If the stored process (report) is accessible to all portal users, the public content administrator can use the Personalize feature to add the report to a public list. When the administrator performs this function, the portal application displays a list box containing all reports in the enterprise directory that are accessible to all portal users. From the list box, the public content administrator selects the

reports that are to appear on the public list.

- Group owners can use the Personalize feature to add the report to a role-based list. When the group owner performs this function, the portal application displays a list box containing all reports in the enterprise directory that the group is authorized to access. From the list box, the group owner selects the reports that are to appear on the role-based list.
- Individual users can use the Personalize feature to add the reports to a personal list. When the user performs this function, the portal application displays a list box containing all reports in the enterprise directory that the user is authorized to access. From the list box, the user selects the reports that are to appear on his or her personal list.

SAS[®] Stored Process Coding Example

The following SAS program is an example of a stored process. This program, called `prdsale1.sas`, is delivered with the SAS Information Delivery Portal application and is part of the demo portal. You can find the source file in the "SASDemos" directory within the portal's installation directory. When you log on to the portal with the user name "portaldemo," the stored process appears in the Reports window as "Product Sales Demo."

The first section of `prdsale1.sas`, called the *prologue*, uses the SAS Macro Language to create and initialize a variable for each of the stored process's execution parameters. In this example, the prologue creates the variables `pCountry` and `pYear` and initializes them to null values.

```
%let pCountry=;  
%let pYear=;
```

The next line of code, `*ProcessBody`, is a marker. This statement is required in order to delineate the boundary between the prologue and the source body.

```
*ProcessBody;
```

The next section of code, the data step, reads the input file `sashelp.prdsale`. The stored process filters the records based on the user-supplied values for the execution parameters `pCountry` and `pYear`.

```
data filter;  
set sashelp.prdsale;  
if UPCASE(country) = UPCASE(symget("pCountry"));  
if year = symget("pYear");  
run;
```

The proc steps sort and tabulate the data. The SAS Output Delivery System (ODS) is used to place the output in an HTML format.

```
proc sort data=filter;
by country;
run;
ods html close;
filename b temp;
ods html body=b (url="body.htm")
      style=BarrettsBlue;
proc tabulate data=filter;
table country*division,
      actual;
class country division;
var actual;
run;
ods html close;
```

The final section of code uses call routines from the Publish Framework section of SAS Integration Technologies to publish the report results to the portal. First, the `package_begin` statement creates the new package. The `insert_html` statement inserts the HTML file into the package. The `package_publish` statement sends the package to the requesting application, which in this case is the SAS Information Delivery Portal. The `package_end` statement then releases system resources from the Publish framework.

For detailed information and additional examples of the use of these call routines, refer to the Publish Package Interface section of the SAS Integration Technologies Web site.

```
call package_begin(pid, description, nameValue, rc);
if rc ne 0 then do;
  msg = sysmsg();
  put msg;
end;
body="fileref:b";
bodyUrl="body.htm";
frame="";
frameUrl="";
contents="";
contentsUrl="";
page="";
pageUrl="";
nameValue="";
description="Sales Report: Tabular";
call insert_html(pid, body, bodyUrl,
                frame, frameUrl,
                contents, contentsUrl,
                page, pageUrl,
                description, nameValue, rc);
if rc ne 0 then do;
  msg = sysmsg();
  put msg;
end;
```

```

call package_publish(pid, "TO_REQUESTER", rc, "", "");
if rc ne 0 then do;
  msg = sysmsg();
  put msg;
end;
call package_end(pid, rc);
run;

```

The program in its entirety is displayed below.

```

%let pCountry=;
%let pYear=;
*ProcessBody;
title "&pYear Sales Report for &pCountry";
data filter;
set sashelp.prdsale;
if UPCASE(country) = UPCASE(symget("pCountry"));
if year = symget("pYear");
run;
proc sort data=filter;
by country;
run;
ods html close;
filename b temp;
ods html body=b (url="body.htm")
  style=BarrettsBlue;
proc tabulate data=filter;
table country*division,
  actual;
class country division;
var actual;
run;
ods html close;
data _null_;
rc = 0;
length description $100 nameValue $100
  body $64 bodyUrl $64 frame $64 frameUrl $64
  contents $64 contentsUrl $64 page $64 pageUrl $64;
NameValue="category=sales context=product";
description="&pYear Sales Report for &pCountry";
call package_begin(pid, description, nameValue, rc);
if rc ne 0 then do;
  msg = sysmsg();
  put msg;
end;
body="fileref:b";
bodyUrl="body.htm";
frame="";
frameUrl="";
contents="";
contentsUrl="";
page="";
pageUrl="";
nameValue="";
description="Sales Report: Tabular";
call insert_html(pid, body, bodyUrl,
  frame, frameUrl,
  contents, contentsUrl,

```

```
        page, pageUrl,  
        description, nameValue, rc);  
if rc ne 0 then do;  
    msg = sysmsg();  
    put msg;  
end;  
call package_publish(pid, "TO_REQUESTER", rc, "", "");  
if rc ne 0 then do;  
    msg = sysmsg();  
    put msg;  
end;  
call package_end(pid, rc);  
run;
```

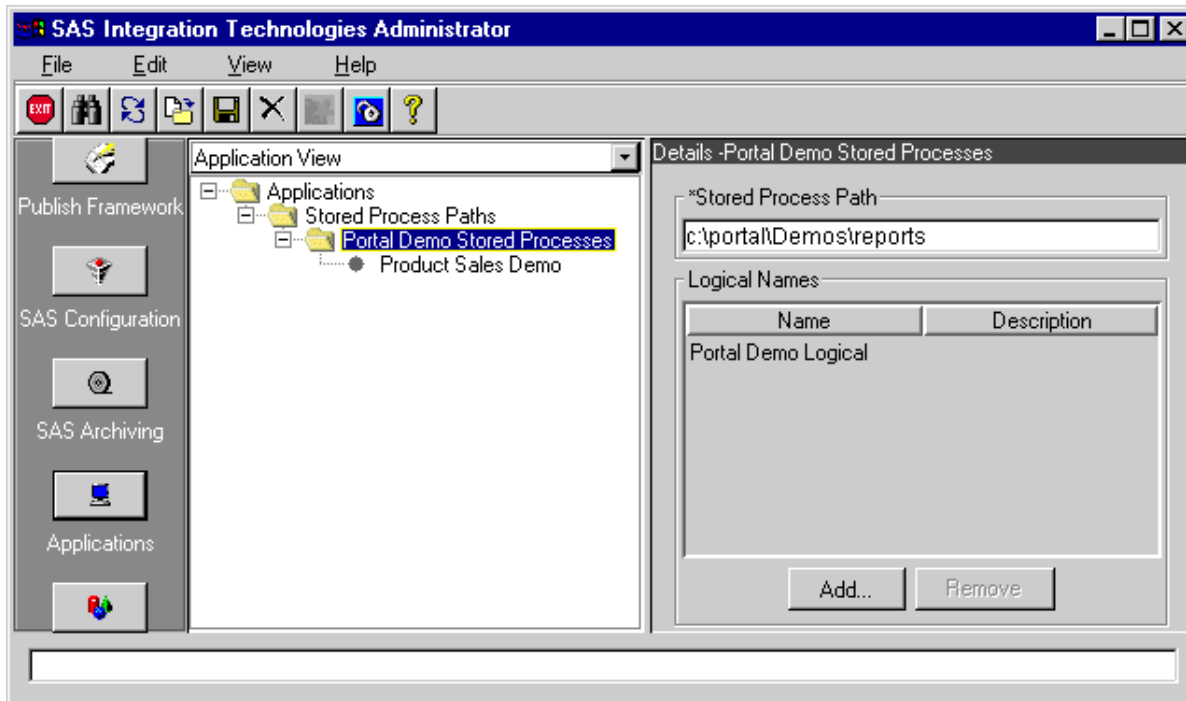
Stored Process Metadata Example

This page provides examples of metadata for stored processes. The examples are based on the stored process `prdsale1.sas`, which is installed automatically in the demo portal. You can find the source file for this stored process in the "SASDemos" directory within the portal's installation directory. When you log on to the portal with the user name "portaldemo," the stored process appears in the Reports window as "Product Sales Demo."

Defining the Stored Process Path

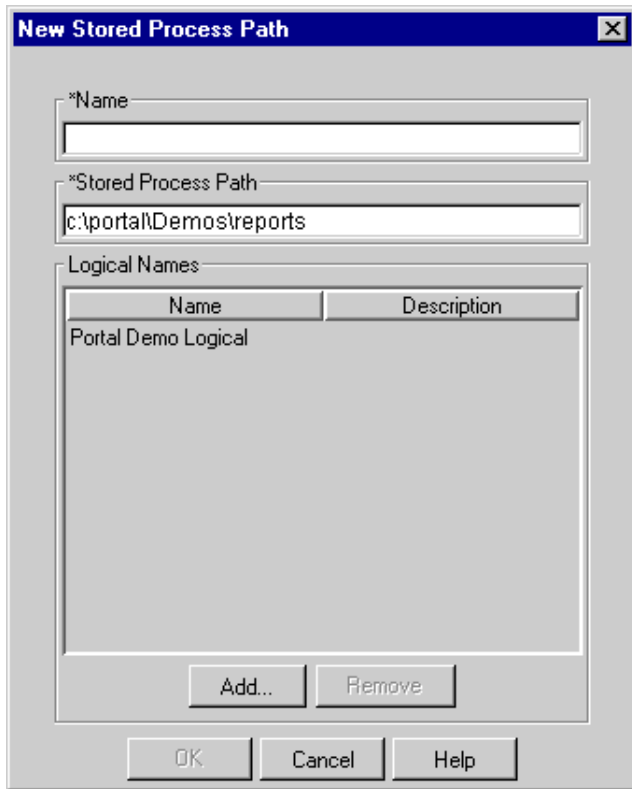
First, you must update the enterprise directory with metadata identifying the path where your stored processes reside. When you install the demo version of the portal, a stored process path called Portal Demo Stored Processes is automatically defined using the value you provide in the the `$$SAS_DEMO_DIR$:` parameter of the `Install.properties` file.

To view this metadata, open the IT Administrator interface, select the Applications button, and select the Portal Demo Stored Processes entry under Stored Process Paths, as shown below.



The stored process path is the complete path to the file on which the stored processes reside. The logical name identifies the server. (For information about the SAS servers defined in your enterprise directory, select the SAS Configuration button.)

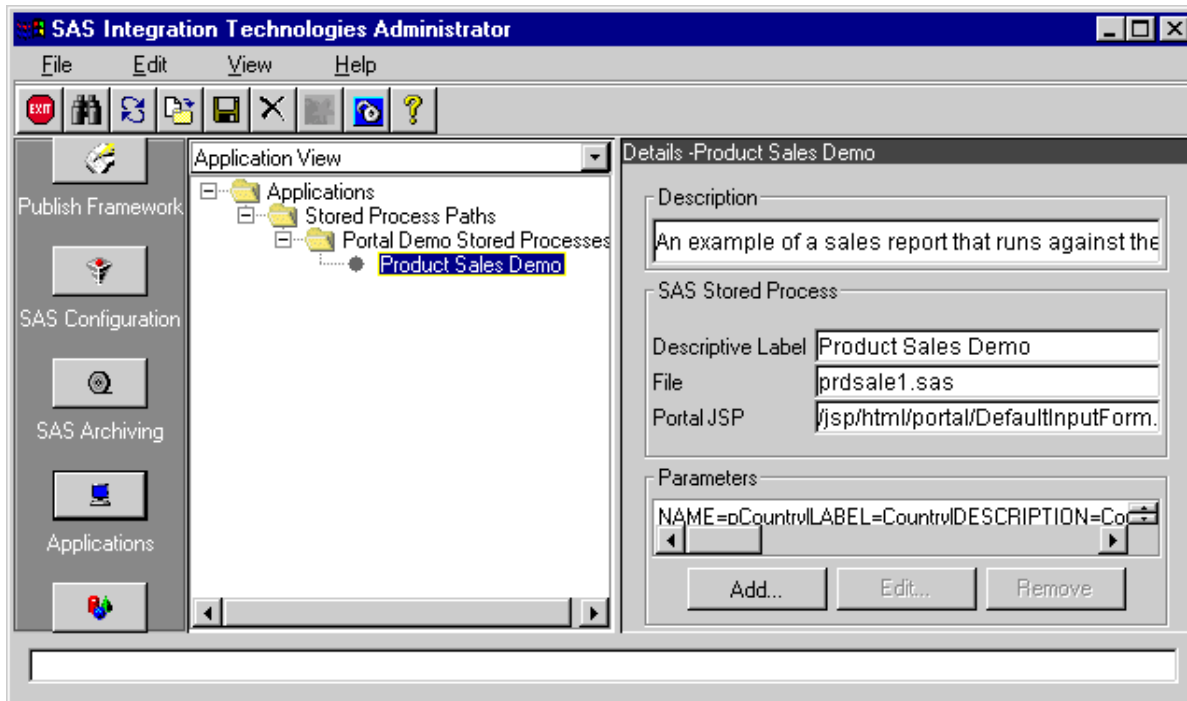
If you want to define another stored process path, choose **File, New** from the menu bar and enter the path information in the following dialog box:



Defining the Stored Process

Next, you must update the enterprise directory with metadata to define the stored process and its execution parameters.

In IT Administrator, you can view the metadata for the stored process prdsale1.sas by selecting **Product Sales Demo** in the tree under the Portal Demo Stored Processes path, as shown below.



The metadata for the stored process includes:

Description

An optional Description, for documentation purposes

Descriptive Label

A label to identify the stored process on lists in the portal.

File

The SAS program file containing the stored process

Note: Before placing the file name in the enterprise directory, IT Administrator appends it to the value of the descriptive label attribute, described above, followed by an underscore character.

In this example, IT Administrator appends the program file name **prdsale1.sas** to the descriptive label **Product Sales Demo**. The resulting value, **Product Sales Demo_prdsale1.sas** is assigned to the `sasstoredprocessvalue` attribute on the enterprise directory. If you are entering your metadata directly into the enterprise directory, it is important to use this format for the `sasstoredprocessvalue` attribute.

Portal JSP

The path to the JSP file that contains the input form to be displayed in the portal for execution parameter selection. The JSP must be located within the ServletContext of the portal application. This example uses the portal application's default input form, which automatically renders your stored process parameters as list boxes in the portal.

Parameters

The parameters that are passed to the SAS program upon execution. The parameters for the stored process prdsale1.sas are as follows.

Note: Your DATA_SOURCE value will be different, since it will reflect the structure of your enterprise directory.

```
NAME=pCountry | LABEL=Country | DESCRIPTION=Country |  
  REQUIRED=true | DEFAULT= |  
  DATA_SOURCE=sasuniqueName=prdsale,cn=sasTables,  
  sascomponent=sasDataSources,cn=SAS,  
  ou=Pubs,o=SAS Institute,c=US |  
  REPORT_QUERY=select distinct country from sashelp.prdsale
```

```
NAME=pYear | LABEL=Year | DESCRIPTION=Year |  
  REQUIRED=true | DEFAULT= |  
  DATA_SOURCE=sasuniqueName=prdsale,cn=sasTables,  
  sascomponent=sasDataSources,cn=SAS,  
  ou=Pubs,o=SAS Institute,c=US |  
  REPORT_QUERY=select distinct year from sashelp.prdsale
```

The parameters pCountry and pYear have the same names as the variables that were defined in the prologue section of the SAS program prdsale1.sas.

The DATA_SOURCE is the the fully qualified distinguished name of the table that contains valid values for the parameter. In this case, the data source is a table. (You can see the metadata that have been defined for the table by selecting the SAS Data Sources button in the IT Administrator interface.)

The REPORT_QUERY contains an SQL statement to extract the valid values for pCountry (or pYear) from the table. When the stored process is invoked from the portal, these values will be placed in list boxes on the portal's default user input form.

Instead of defining a data source and report query for a parameter's valid values, you can hard-code the valid values into the parameter definition. This method may be appropriate when it is unlikely that the valid values will change. For example, here is the pCountry parameter definition with hard-coded valid values:

```
NAME=pCountry | LABEL=Country | DESCRIPTION=Country |  
  REQUIRED=true | DEFAULT= | VALUES=CANADA,GERMANY,U.S.A.
```

If you want to define another stored process within a path, select the path and choose **File, New** from the menu bar. Then enter the stored process information in the following dialog box:

New Stored Process

Name

Description
An example of a sales report that runs against the product sales data set.

SAS Stored Process

| | |
|-------------------|---------------------------------------|
| Descriptive Label | Product Sales Demo |
| File | prdsale1.sas |
| Portal JSP | /jsp/html/portal/DefaultInputForm.jsp |

Parameters

NAME=pCountry|LABEL=Country|DESCRIPTION=Country|REQUIRED=true|DEFAULT=|DATA_SOURCE=sasuni
NAME=pYear|LABEL=Year|DESCRIPTION=Year|REQUIRED=true|DEFAULT=|DATA_SOURCE=sasunique

Add... Edit... Remove

OK Cancel Help

Adding SAS[®] Publication Channels to the Portal

The SAS Information Delivery Portal provides an interface through which users can subscribe to SAS publication channels. After subscribing to a channel, users can use the portal to view archived content that is published through the channel. This portal feature relies on the SAS Publishing Framework software, which is part of SAS Integration Technologies. For detailed documentation of this software, see the Publishing Framework section of the SAS Integration Technologies Web site.

To set up a channel in the portal, you must update the portal application's enterprise directory with metadata about the channel. You must then use the Personalize tool to make the channel appear as a selection on the portal. Instructions for both of these steps are provided below.

Once the channel is created, you can publish information to the channel and its subscribers.

Step 1: Update the Enterprise Directory with Metadata about the Channel

Use the SAS Integration Technologies (IT) Administrator interface to perform the following updates to metadata in your portal's enterprise directory. Refer to the SAS Channel Metadata Example for an annotated illustration of these tasks. The links below point to specific portions of the example.

- Define the channel in the enterprise directory, as shown in this example. For more information, refer to the [Creating Channels](#) topic in the SAS Integration Technologies Web site.
- Define one or more archive paths for the channel in the enterprise directory, as shown in this example. For more information, refer to [Creating an Archive Path](#) in the SAS Integration Technologies Web site.
- Define access rules for the channel. For more information, refer to [Controlling Access to Portal Content](#).

Step 2: Use the Personalize Tool to Make the Channel Appear as a Selection on the Portal

Depending on who has access permission to the channel, you can use one of several methods to make a channel appear as a selection on the portal:

- If the channel is accessible to all portal users, the public content administrator can use the Personalize feature to add the channel to a public list. When the administrator performs this function, the portal application displays a list box containing all channels in the enterprise directory that are accessible to all portal users. From the list box, the public content administrator selects the channels that are to appear on the public list.
- Group owners can use the Personalize feature to add the channel to a role-based list. When the group owner performs this function, the portal application displays a list box containing all channels in the enterprise directory that the group is authorized to access. From the list box, the group owner selects the channels that are to appear on the role-based list.
- Individual users can use the Personalize feature to add the channel to a personal list. When the user performs this function, the portal application displays a list box containing all channels in the enterprise directory that the user is authorized to access. From the list box, the user selects the channels that are to appear on his or her personal list.

SAS[®] Channel Metadata Example

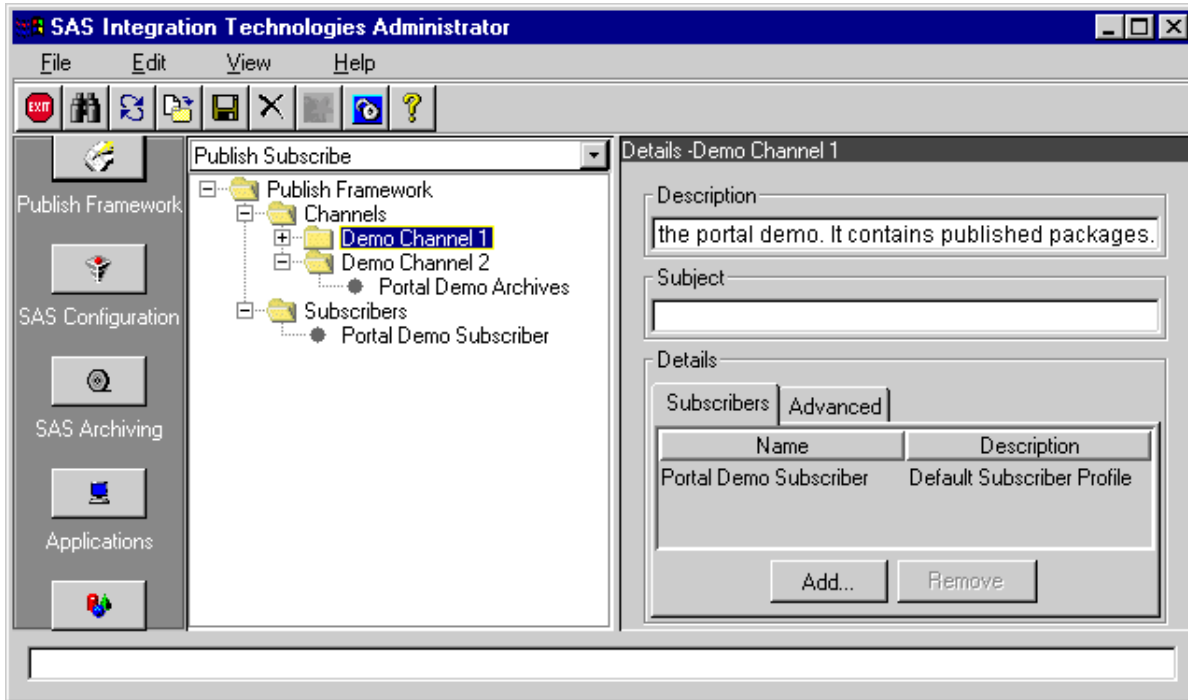
This page provides examples of metadata for a SAS publication channel. The examples show the metadata for Demo Channel 1, which is installed automatically in the demo portal. When you log on to the portal with the user name "portaldemo," this channel appears in the Channels window.

Defining the Channel

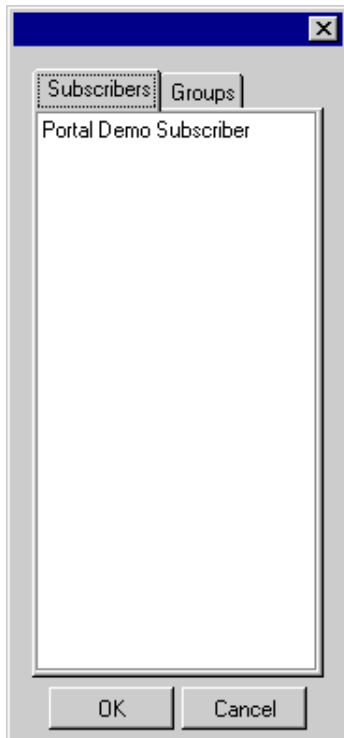
To create a channel, you must first update the enterprise directory with metadata describing the channel. When you install the demo version of the portal, Demo Channel 1 is automatically defined in the

directory.

To view the demo channel's metadata, open the IT Administrator interface, select the **Publish Framework** button, and select the **Demo Channel 1** entry under **Channels**, as shown below.

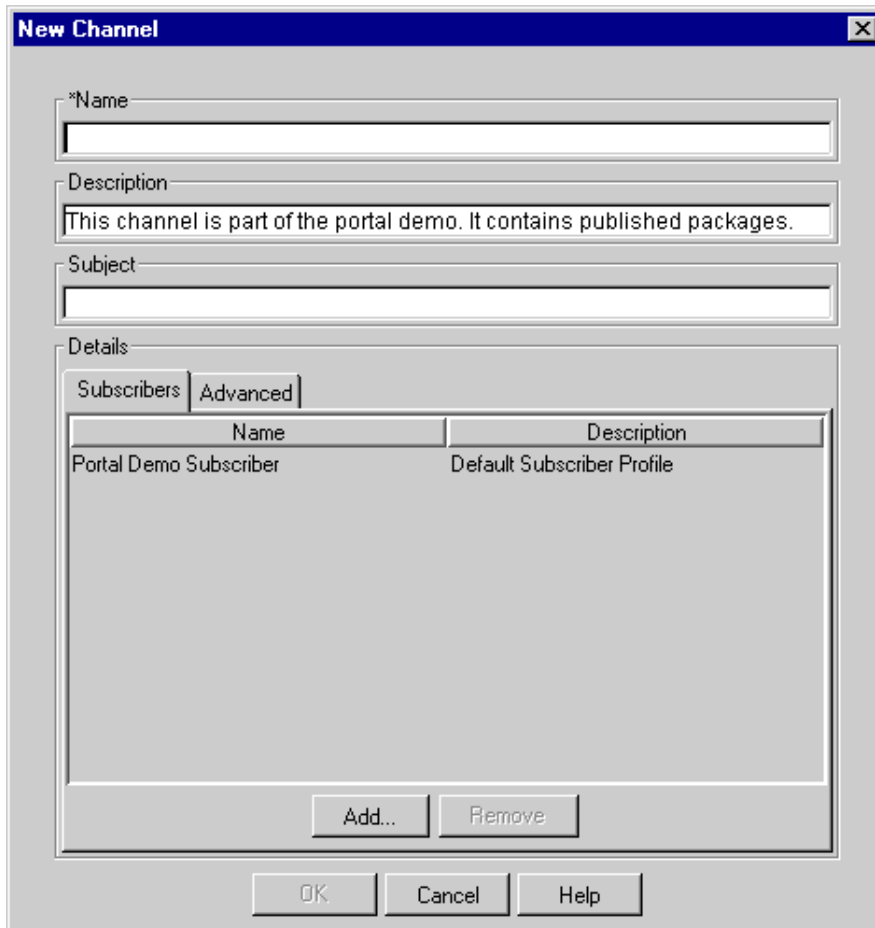


The metadata for this channel includes just a description and a subscriber, the Portal Demo user. To add more subscribers, you can select the Add button. A list of all users who are registered to the portal is displayed:



You can select additional users to subscribe to the channel, or you can click on the Group tab and subscribe an entire group of users. Alternatively, individual portal users can subscribe to channels themselves using the portal's Personalize feature, or by clicking the subscribe icon next to a channel that is listed in a portal window.

If you want to define another channel, select **File, New, Channel** from the menu bar and make entries in the following dialog box:

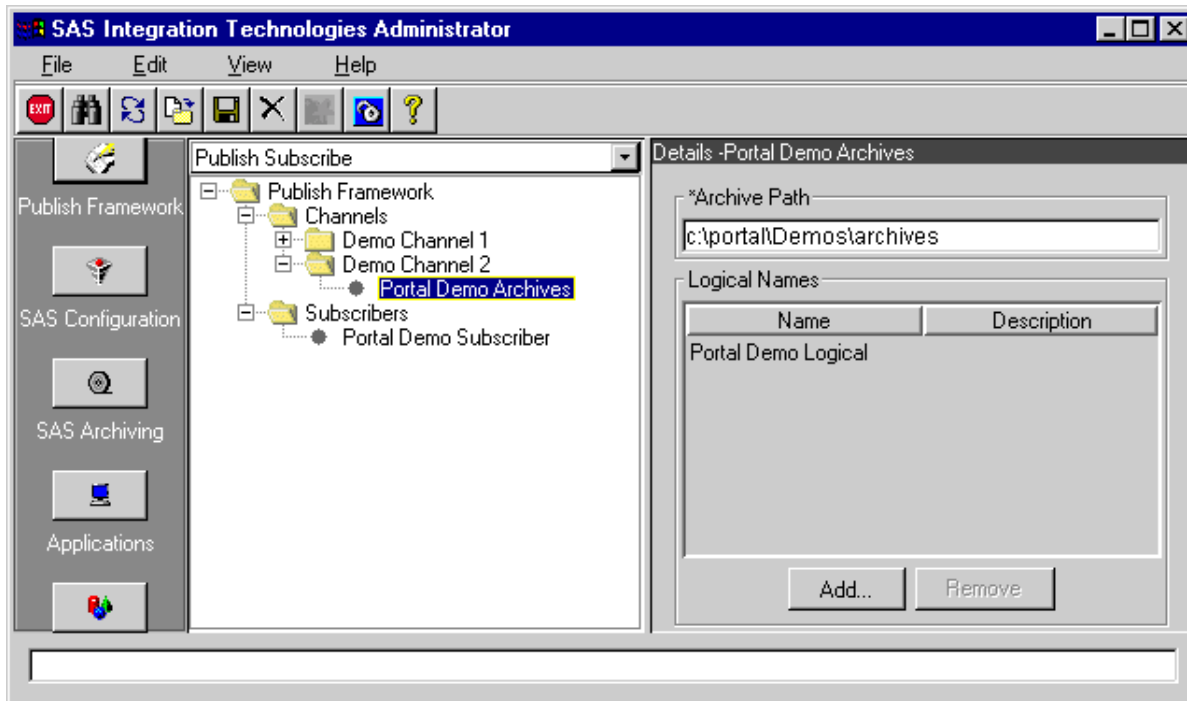


For a detailed description of each input field, select the **Help** button on the dialog box.

Defining an Archive Path

In order for SAS publication channel content to be made available on the SAS Information Delivery Portal, the content must be published to the channel using the "archive" transport method. In order for this method to work, you must update the enterprise directory with metadata to define the archive path. The archive path identifies the location in the portal's file system where published packages for the channel will be stored.

For Demo Channel 1, the portal installation process automatically creates an archive path called Portal Demo Archives using the value you provide in the the `$$SAS_DEMO_DIR$:` parameter of the `Install.properties` file. You can see this metadata in the first entry in the tree under Demo Channel 1, as shown below:



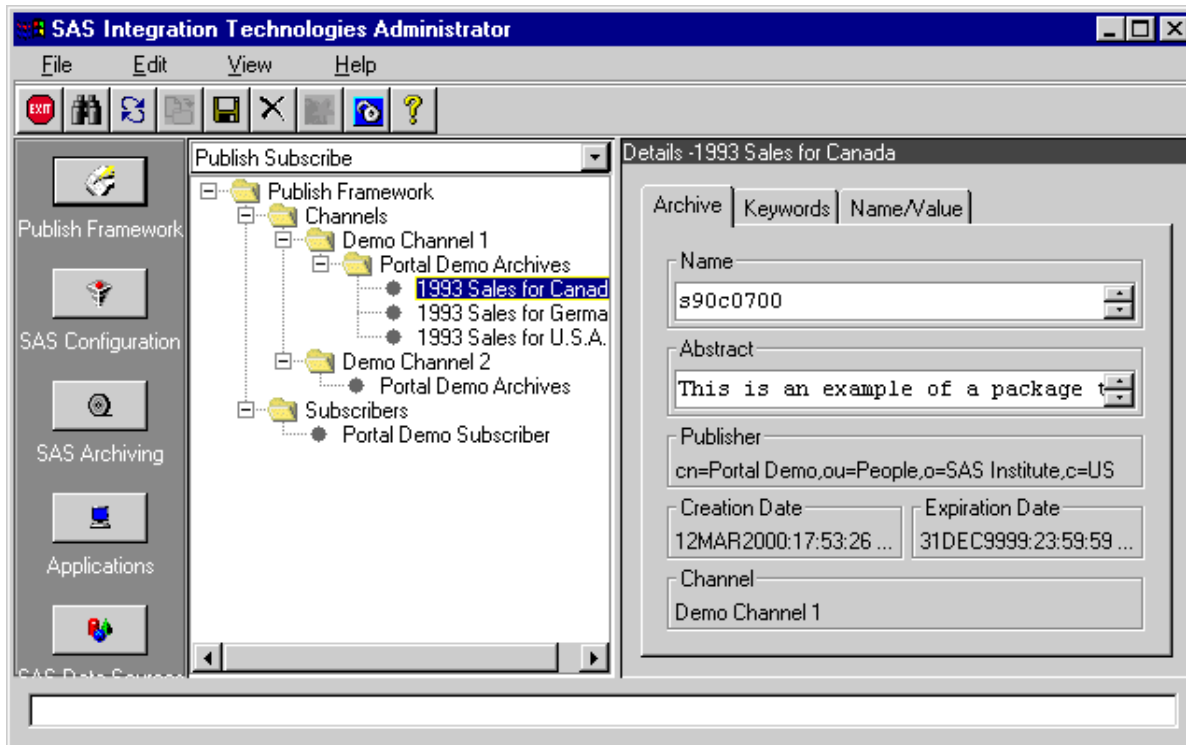
The metadata for the archive path includes the complete path relative to the root of the SAS server where published packages are to be stored.

The logical name identifies the SAS server that houses the archive path. (For information about the SAS servers defined in your enterprise directory, select the SAS Configuration button.)

Viewing Metadata about Published Packages

Each time a package is published to a channel via the Publishing Framework, information about the published package (referred to as an *archive*) is automatically stored in the enterprise directory. The portal application uses this information to render channel content to portal users.

You can use the IT Administrator interface to view metadata on published packages. For example, to see metadata for packages published to Demo Channel 1, view the tree under the **Portal Demo Archives** path as shown below:



Adding Data Warehouse Views to the Portal

If you have implemented a data warehouse using the SAS/Warehouse Administrator, you can export metadata for selected warehouse views from the Warehouse Administrator's enterprise directory to the portal's enterprise directory. Authorized portal users can then view the most current warehouse data directly from the portal. For more information about Warehouse Administrator, see the SAS/Warehouse Administrator section of the SAS Web site.

In the portal, warehouse content is displayed in a manner similar to channels. The user opens the warehouse entry to see the list of tables, multidimensional databases (MDDBs), and InfoMarts that it contains. When the user selects a particular content item, the portal displays it using the appropriate viewer (for example, the portal table viewer, MDDB viewer, or document viewer).

The warehouse export capability is an efficient alternative to the manual entry of metadata for tables and MDDBs in the portal. After the warehouse export process is completed, you must make several adjustments to the resulting metadata so that the various SAS components will work together correctly. These adjustments are described in detail in the steps below.

To add a warehouse view to the portal, follow these steps:

1. Make note of the logical name attributes currently assigned to the portal's SAS server and spawner definitions.

2. Export the appropriate metadata from Warehouse Administrator to the portal's enterprise directory.
3. Update logical name attributes as necessary in the SAS server and SAS spawner definitions.
4. Update the domain name attribute in the SAS server definition.
5. Ensure that fully qualified paths are provided for the SAS libraries that contain the warehouse.
6. Ensure that a SAS login has been defined to allow access to the SAS server that contains the warehouse.
7. Verify that MDDDB viewing is enabled on the SAS server.
8. Use the Personalize tool to make each warehouse view or InfoMart appear as a selection on the portal.

Step 1: Make Note of Logical Name Attributes

Before you perform the export from Warehouse Administrator, use either the SAS Integration Technologies (IT) Administrator or the enterprise directory console to examine any SAS spawner and SAS server definitions that are already present in the portal's enterprise directory. Make note of the logical name attributes that are present in these definitions, since you may need to use them in a later step.

For detailed instructions on using IT Administrator, refer to the IT Administrator section of the SAS Integration Technologies Web site.

Step 2: Export the Appropriate Metadata from Warehouse Administrator

In the Warehouse Administrator application, select the specific warehouse view that you would like to make available to portal users. Then choose **Tools, LDAP Security** from the SAS menu bar. On the panels provided, enter the required information about the export operation. Then click the **Export** button. The SAS log will display results of the export operation and any error messages. For more detail, refer to the Data Warehouse Export Example.

The following metadata is exported from Warehouse Administrator:

- Metadata describing the selected warehouse view. This metadata is exported to the sasMetadataRepository component of the portal's enterprise directory.
- A SAS server definition and SAS spawner definition for the host on which the warehouse resides. This metadata is exported to the sasServer component of the portal's enterprise directory.

Note: If the portal's enterprise directory already contains a server definition with the appropriate

machine DNS, protocol, port, and command attributes, then the export procedure updates this definition with the logical name attributes for the warehouse server. If the portal's enterprise directory already contains a spawner definition with the appropriate machine DNS attribute, then the export procedure attempts to update this definition with the logical name attributes for the warehouse spawner. In these cases, you must manually inspect the logical names and adjust them as necessary so that server connections can be made successfully. For details, see Step 3.

- Metadata describing the data sources. This metadata, which is stored in the `sasDataSources` component of the enterprise directory, includes definitions for the following objects:
 - The SAS libraries that contain the warehouse view.
 - The SAS MDDBs and tables that are included in the view.
 - Each of the columns included in these MDDBs and tables.
 - Infomarts included in the view.

To view the exported metadata, you can use either the IT Administrator application or the enterprise directory console.

Step 3: Update Logical Name Attributes As Necessary

In the following cases, the export procedure makes changes to logical name attributes on the portal's enterprise directory. You must correct the logical names manually in order for the object spawner to run correctly and server connections to be made successfully.

- If the portal already has a SAS server definition with the appropriate machine DNS, protocol, port, and command attributes, then the export procedure updates this definition with the logical name attributes for the warehouse server.

When the export is complete, use either IT Administrator or the enterprise directory console to determine if the server definition's original logical name attributes are still present. If they are not, you must re-enter the original logical names that were overwritten. However, do not replace the logical name that was exported from Warehouse Administrator.

- If the portal already has a SAS spawner definition with the appropriate machine DNS attribute, then the export procedure attempts to update this definition with the logical name attributes for the warehouse spawner.

When the export is complete, use either IT Administrator or the enterprise directory console to determine if the spawner definition's original logical name attributes are still present. If they are not, you must re-enter the original logical names that were overwritten.

You must also determine if the spawner definition contains a logical name attribute that matches the logical name for the warehouse server. If this logical name is not present in the spawner definition, you must add it.

Step 4: Update the Server Domain Attribute

If the export operation creates a new SAS server definition on the portal's enterprise directory, you must add a domain attribute to this definition. The object spawner requires that the domain be identified so that the correct server can be located. You can make this change using either IT Administrator or the enterprise directory console.

Step 5: Ensure Fully Qualified Paths for SAS Libraries

The portal requires that SAS library definitions contain fully qualified path information. If relative paths are used in any of the library definitions that were exported from Warehouse Administrator, you must extend this information to include the entire path to the root of the server. You can use either IT Administrator or the enterprise directory console to perform this modification.

Step 6: Add a SAS Login Entry to the Portal's Enterprise Directory

Since the exported metadata does not include a SAS login definition, you must create one. The login definition should specify:

- The same logical name and the same domain as the associated SAS server and SAS spawner definitions.
- The SAS login id and password which are to be used as credentials to access the server that hosts the warehouse.
- Distinguished names (DNs) for the portal users and/or groups of users who are authorized to access the warehouse view.

To create the SAS login, you can use the IT Administrator application. For detailed instructions, see [Adding a SAS Login on the SAS Integration Technologies Web site](#).

Step 7: Verify That MDDB Viewing is Enabled on the SAS Server

Since warehouse views can include MDDBs, you must ensure that SAS has been updated to enable the portal's MDDB viewer to communicate with the SAS server. The steps for performing this update, which are included in the portal's installation instructions, are as follows:

1. Copy the transport file, `mddbview.cpo`, from the SASUpdates subdirectory of the Portal setup directory to the appropriate SAS server.
2. Start SAS on the server.
3. Enter and submit the following code:

```
filename updates 'path-to-transport-file';  
proc cimport library=sashelp infile=updates force; run;
```

4. Exit SAS.

Step 8: Use the Personalize Tool to Make the Warehouse Views Appear as Selections on the Portal

After you have completed steps 1 through 3 to add metadata for a warehouse view to the portal's metadata, you can use one of several methods to make the warehouse view appear as a selection on the portal:

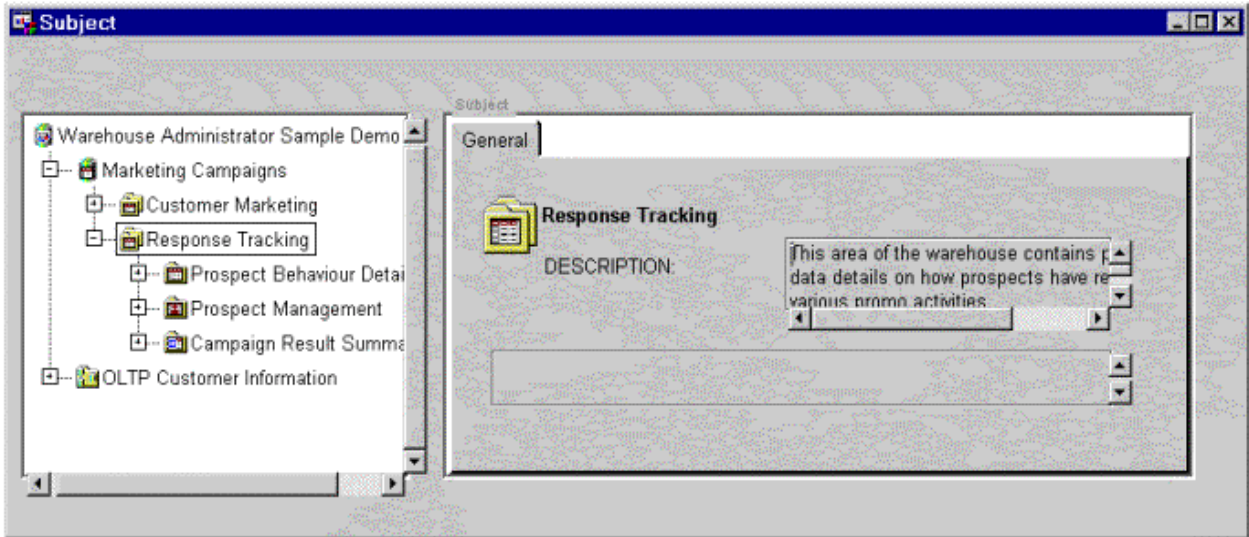
- If the warehouse view is accessible to all portal users, the public content administrator can use the Personalize feature to add the view to a public list. When the administrator performs this function, the portal application displays a list box containing all warehouse views in the enterprise directory that are accessible to all portal users. From the list box, the public content administrator selects the warehouse views that are to appear on the public list.
- Group owners can use the Personalize feature to add a warehouse view to a role-based list. When the group owner performs this function, the portal application displays a list box containing all warehouse views in the enterprise directory that the group is authorized to access. From the list box, the group owner selects the warehouse views that are to appear on the role-based list.
- Individual users can use the Personalize feature to add the warehouse view to a personal list. When the user performs this function, the portal application displays a list box containing all warehouse views in the enterprise directory that the user is authorized to access. From the list box, the user selects the warehouse views that are to appear on his or her personal list.

For more information, refer to the Personalization topic in the portal User's Guide.

Data Warehouse Export Example

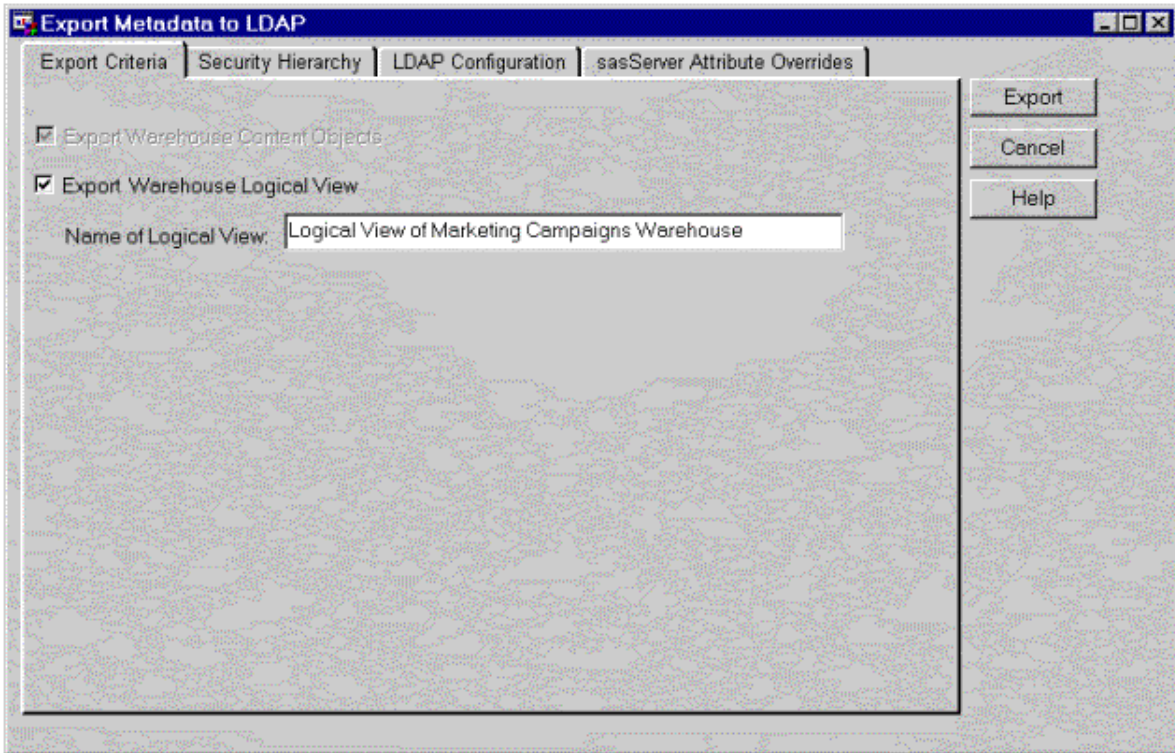
This section provides an example of the procedure to export metadata from SAS Warehouse Administrator to the SAS Information Delivery Portal. The example uses the Sample Demo warehouse data that is provided with the SAS Warehouse Administrator software.

1. In Warehouse Administrator, select the warehouse view whose metadata you would like to export, as in the following example:

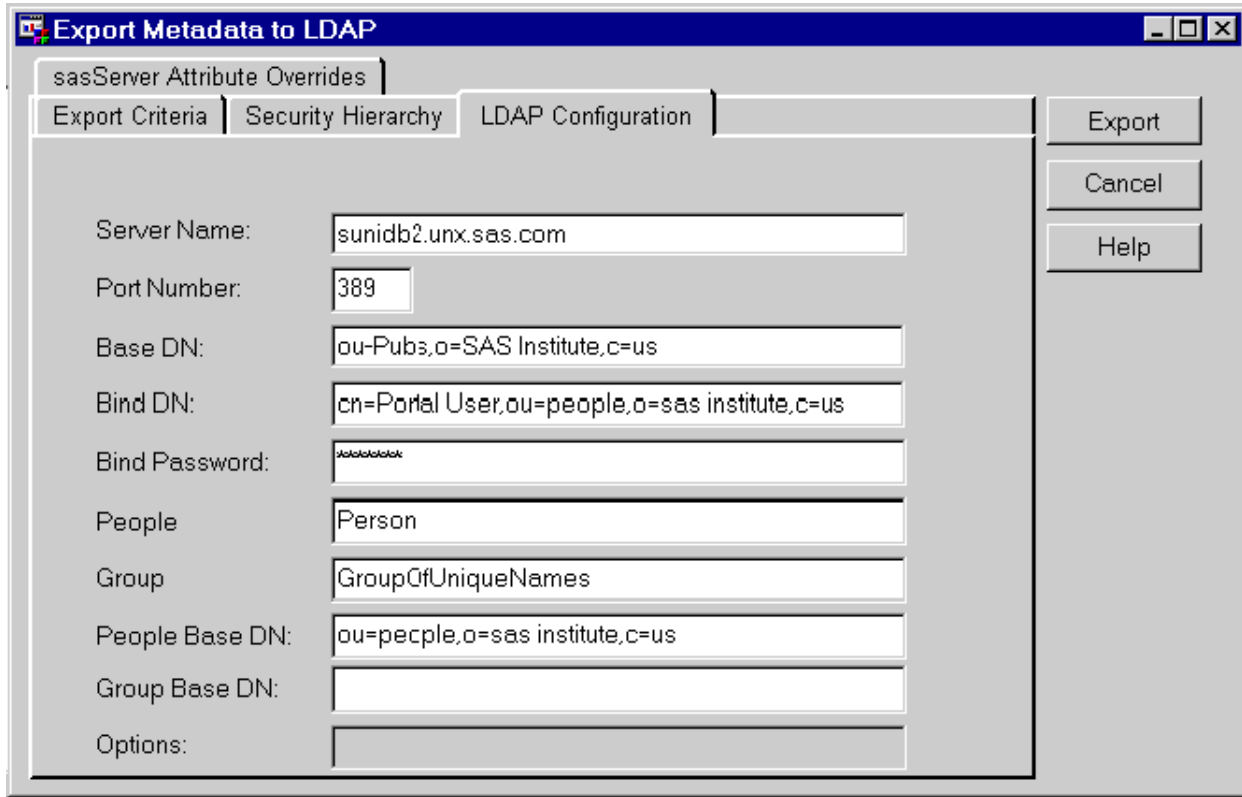


In this example, the Response Tracking view of the Sample Demo warehouse has been selected.

2. With the Subject window still active, choose **Tools, LDAP Security** from the SAS menu bar. The Export Metadata to LDAP window appears, as in the following example:



3. Select the **LDAP Configuration** tab. On this panel, enter information about the LDAP server and tree to which the metadata is to be exported, as in the following example:



4. Make any necessary entries on the Security Hierarchy and sasServer Attribute Overrides panels. For detailed instructions, you can click **Help** button on each of these panels.
5. Click the **Export** button to begin the export process.

Adding webEIS™ Documents to the Portal

The SAS Information Delivery Portal can provide access to interactive JSPs that were created using the webEIS thin-client reporting solution. These JSPs provide point-and-click access to live data from enterprise databases centrally located on SAS servers. The data is presented in the form of charts, tables, and images. Users can drill into the data, create subsets, highlight exceptions, and create new computations.

To add a webEIS document to the portal, you must:

- Save the webEIS document as a Java server page (JSP).
- Move the JSPs and their associated files to the appropriate directory on the Web server.
- Update URL and SAS server connection information in the JSP files.

- Verify That MDDB viewing is enabled on the SAS server.
- Use the Personalize tool to add the webEIS document to the portal's enterprise directory.
- Define access rules for the document.
- Use the Personalize tool to make the webEIS document appear as a selection on the portal.

Step 1: Save the webEIS Document as a Java Server Page (JSP)

Use webEIS (Version 2.0 or later) to create a webEIS document. Then save the document as a Java Server Page (JSP). The saved document will consist of a main JSP and one or more secondary JSPs. It may also include other support files such as cascading style sheet (CSS) files and graphics files.

Step 2: Move the JSPs and Their Associated Files to the Appropriate Directory on the Web Server

Move or copy all of the exported files--including the main JSP, any secondary JSPs, and all supporting files--to the Web server. The files must be placed in or under the directory that contains the JSP files for the portal application.

If you installed the demonstration portal, your Web server will contain an example folder called "WebEisDoc." This folder contains the files for a sample webEIS document.

Step 3: Update URL and SAS Server Connection Information in the JSP Files

You must make the following changes to the JSP files that comprise the webEIS document:

- In the main JSP, edit any URLs that point to secondary JSPs. The modified URLs should point to the JSPs at their new location within the portal application.
- Edit server connection information in each JSP. The server connections should use the appropriate SAS server definition on the portal's enterprise directory.

For more information, refer to the webEIS JSP Coding Example.

Step 4: Verify That MDDB Viewing is Enabled on the SAS Server

Since webEIS documents can include multidimensional databases (MDDBs), you must ensure that SAS has been updated to enable the portal's MDDB viewer to communicate with the SAS server. The steps for performing this update, which are included in the portal's installation instructions, are as follows:

1. Copy the transport file, mddbview.cpo, from the SASUpdates subdirectory of the Portal setup directory to the appropriate SAS server.
2. Start SAS on your server.
3. Enter and submit the following code:

```
filename updates 'path-to-transport-file';  
proc cimport library=sashelp infile=updates force; run;
```

4. Exit SAS.

Note: You must perform this update in order to view the sample webEIS document that is provided with the portal.

Step 5: Use the Personalize Tool to Add the webEIS Document to the Portal's Enterprise Directory

Use the portal's Personalize tool to add the webEIS document to the portal. This tool provides a graphical user interface for entering the document's metadata, including the path and file name of the main JSP. For detailed instructions, refer to Adding and Editing Personal webEIS Documents or Adding and Editing Role-Based webEIS Documents in the portal User's Guide.

Step 6: Define Access Rules for the Document

Take any necessary steps to control access to the webEIS document. For general information about access control, refer to Controlling Access to Portal Content.

If EIS access control is active on the SAS server, EIS credentials must be provided before a user can view the document through the portal. To support this authentication process, you can implement one or both of the following methods in the portal:

- **Use a SAS login definition to provide the required user credentials.** To implement this method, you must:
 - Create a SAS login definition, using IT Administrator. (For details, see Adding a SAS Login on the SAS Integration Technologies Web site.) The definition should include:
 - The distinguished names of individual portal users or groups of users who will be allowed to access the webEIS document through the portal.
 - The SAS login and password which are to be used as EIS credentials to access the document on the server.
 - In the webEIS document's metadata, enter a sasDomainName attribute that matches the sasDomainName attribute in the SAS login definition. To add this attribute to the webEIS document's metadata, you must use the enterprise directory console.

- **Present the user with a prompt to enter his or her EIS credentials.** To implement this method, you must use the enterprise directory console to add a sas-PromptForCredentials attribute to the webEIS document's metadata. The attribute should be given a value of "1."

When a user attempts to view the webEIS document through the portal, the portal performs EIS authentication as follows:

- First, the portal looks for a SAS login definition that has the same sasDomainName attribute as the webEIS document. If a match is found, the SAS login information is used to perform EIS authentication.
- If no matching login definition is found, or if the webEIS document does not have a sasDomainName attribute, the portal looks for a sas-PromptForCredentials attribute that has a value of "1." If this attribute and value are found, the portal prompts the user to enter EIS credentials, and the entered information is used to perform EIS authentication.

Step 7: Use the Personalize Tool to Make the webEIS Document Appear as a Selection on the Portal

Depending on who has access permission to the stored process, you can use one of several methods to make a webEIS document appear as a selection on the portal:

- If the webEIS document is accessible to all portal users, the public content administrator can use the Personalize feature to add it to a public list. When the administrator performs this function, the portal application displays a list box containing all webEIS documents in the enterprise directory that are accessible to all portal users. From the list box, the public content administrator selects the webEIS documents that are to appear on the public list.
- Group owners can use the Personalize feature to add the webEIS document to a role-based list. When the group owner performs this function, the portal application displays a list box containing all webEIS documents in the enterprise directory that the group is authorized to access. From the list box, the group owner selects the webEIS documents that are to appear on the role-based list.
- Individual users can use the Personalize feature to add the webEIS document to a personal list. When the user performs this function, the portal application displays a list box containing all webEIS documents in the enterprise directory that the user is authorized to access. From the list box, the user selects the webEIS documents that are to appear on his or her personal list.

For more information, refer to the Personalization topic in the portal User's Guide.

webEIS™ JSP Coding Example

Before you can access a webEIS document in the portal, you must make the following changes to the Java Server Page (JSP) files that comprise the document:

- In the main JSP, edit any URLs that point to secondary JSPs. The modified URLs should point to the JSPs at their new location within the portal application.
- Edit server connection information in each JSP. The server connections should use the IOM server and should refer to the appropriate SAS server definition on the Portal's enterprise directory.

The examples shown below are taken from the sample webEIS document that is delivered with the SAS Information Delivery Portal. If you follow the installation instructions supplied with the portal, this sample webEIS document is installed as part of the demonstration portal. You can find it on your Web server in a folder called "WebEisDoc."

Edit URLs that Point to Secondary JSPs

Generally, a webEIS document consists of a main JSP and one or more secondary JSPs. Path information stored on the enterprise directory provides the location of the main JSP. URLs in the main JSP then point to any secondary JSPs that are needed for the webEIS document. You must change these URLs so that they point to the secondary JSPs' new locations on the Web server. You must also change the mechanism used to display the secondary JSPs so that they will be displayed correctly when a Web browser has cookies disabled.

For example, suppose the webEIS document's main JSP originally contained a URL to the secondary JSP (called DemoSection1.jsp), as shown below:

```
<html>
  <head>
    <title>WebEIS Document Demo</title>
    <meta Http-equiv="refresh" Content="0; Url=DemoSection1.jsp">
  </head>
</html>
```

To enable the portal to access the secondary JSP, you must modify the URL to point to the JSP's new location on the Web server and to handle Web browsers with cookies disabled. The modifications are shown below:

```
<%@ page import="java.net.URL" %>
<%
  URL url = new URL("http", request.getServerName(), request.getServerPort(),
    application.getAttribute("DocBase") +
    "/jsp/html/samples/WebEisDoc/DemoSection1.jsp");
  response.sendRedirect(response.encodeRedirectURL(url.toString()));
%>
```

Note: In this example, the URL uses path information from the DocBase parameter of the Portal.properties file. The portal's install process builds this file using information that you provide in the Install.properties file. In addition, note that the sendDirect method is used in this example to ensure that the session ID will be encoded in the URL.

Edit Server Connection Information in Each JSP

Any of the JSPs that comprise a webEIS document can contain a definition for a connection to a SAS server. When you add the webEIS document to the portal, you must update these JSPs to use the IOM server and to refer to the appropriate SAS server definition on the Portal's enterprise directory.

To find connection information in a webEIS JSP, look for the following comments:

```
// Start connection definition
{
}
// End connection definition
```

In each JSP that contains a connection definition, you must add code similar to the following *immediately before the closing bracket*:

```
logicalName = "Portal Demo Logical";
com.sas.edir.webapp.portal.util.WebEisDocumentUtil.setConnection(request,
connectionInstance1xConnection, logicalName);
```

The `setConnection` method sets the connection to use IOM. The credentials of the portal user are set in the connection and will be used to create a workspace on the SAS server. The method includes the following parameters:

Request

The predefined `HttpServletRequest` variable.

connectionInstance1xConnection

The `com.sas.rmi.Connection` instance. The name of this variable will differ among JSPs and is based on the name of the webEIS document.

logicalName

The logical name that identifies the server and login definitions to be used when creating the connection to the server

When the webEIS document is invoked, an IOM connection is created and the portal renders the webEIS document in a separate browser window. The resources (including the connection) that are associated with the webEIS document are not cleaned up when the user closes this window. The Object Server terminates when the user logs off from the portal or the portal session times out.

Adding Links to the Portal

The SAS Information Delivery Portal allows you to create links to sites on the Web or on a local intranet. The links can then be included on lists and displayed in windows in the portal.

To add a link to the portal, follow these steps:

1. Use the Personalize tool to add the link to the portal's enterprise directory.
2. Define access rules for the link.
3. Use the Personalize tool to make the link appear as a selection on the portal.

Step 1: Use the Personalize Tool to Add the Link to the Portal's Enterprise Directory

The portal's Personalize tool provides a graphical user interface for adding the name, description, and URL address for a link to the enterprise directory. Individual users can use the Personalize tool to add personal links that are available to only that user. If you have authority as a group owner, you can use this tool to add a role-based link which will be available to all users in the group. If you have authority as a public content administrator, you can use this tool to add a public link which will be available to all portal users.

For instructions on using the Personalize tool to add a personal or role-based link, refer to *Creating and Editing Personal Links* and *Creating and Editing Role-based Links* in the portal User's Guide. To add a public link, the procedure is the same except that you must sign on with public content administrator privileges, and then choose **Public** as the role to personalize.

Step 2: Define Access Rules for the Link

Take any necessary steps to control access to the link. For general information about access control, refer to *Controlling Access to Portal Content*.

Step 3: Use the Personalize Tool to Make the Link Appear as a Selection on the Portal

Depending on who has access permission to the link, you can use one of the following methods to make it appear as a selection on the portal:

- The public content administrator can use the Personalize feature to add the link to a public list. When the public content administrator performs this function, the portal application displays a list box containing all public links. From the list box, the administrator selects the links that are to appear on the public list.
- Group owners can use the Personalize feature to add the link to a role-based list. When the group owner performs this function, the portal application displays a list box containing all role-based links that are available to the group. From the list box, the group owner selects the links that are to appear on the role-based list.
- Individual users can use the Personalize feature to add the link to a personal list. When the user

performs this function, the portal application displays a list box containing all links in the enterprise directory that the user is authorized to access. From the list box, the user selects the links that are to appear on his or her personal list.

Adding Content Channels to the Portal

The SAS Information Delivery Portal provides support for the emerging RSS (Rich Site Summary) standard, a lightweight XML format designed for sharing news headlines and other syndicated Web content. By incorporating RSS content into the portal, you can give users access to high-quality, continually updated news that is relevant to their roles in the organization. The BBC, CNET, CNN, Disney, Forbes, Motley Fool, Wired, Red Herring, Salon, Slashdot, and ZDNet channels are just a few examples of RSS channels that are available publicly.

RSS documents contain metadata, or summary information, about content that is available on the provider's Web site. Each content item consists of a title, a link, and a brief description. By clicking on a link, the user can display the full text for a content item.

To add a content channel to the portal, you must have authority either as a public content administrator or as a group owner. The steps for adding a content channel are as follows:

1. Use the Personalize tool to add the content channel to the portal's enterprise directory.
2. Define access rules for the content channel.
3. Use the Personalize tool to make the content channel appear as a selection on the portal.


Step 1: Use the Personalize Tool to Add the Content Channel to the Portal's Enterprise Directory

The portal's Personalize tool provides a graphical user interface for adding the name, description, and URL address for a content channel to the enterprise directory. If you have authority as a public content administrator, you can use this tool to add a public content channel which will be available to all portal users. If you have authority as a group owner, you can use the Personalize tool to add a role-based content channel which will be available to all users in the group.

The detailed steps for adding a content channel are as follows:

1. If you have not already done so, open the SAS Information Delivery Portal, log on with your user name and password, and click **Personalize** on the toolbar. The Personalize window appears.
2. At the prompt **Which role do you want to personalize?**, do the following:
 - If you are authorized as a public content administrator and would like to add a public content

channel, choose **Public**.

- If you are authorized as a group owner and would like to add a role-based content channel, choose the group name.
3. At the prompt **What do you want to personalize?**, choose **Content Channels** from the drop-down list. The Content Channels control panel appears, showing the existing public or role-based content channels.
 4. Opposite **Add a new <public or your group's name> content channel**, click  to add a new content channel. The New Content Channel control panel appears.
 5. Fill in the fields as follows:

Name

Enter a name for the channel. This name will identify the channel when it is displayed in the portal.

Url

Enter the complete URL for the channel. To ensure accuracy, you might want to use your browser to find the desired address, and then copy the URL from the browser's address line.

Description

Enter a short description for the channel. The description will appear after the channel's name when it is displayed on a list.

Keywords

Enter key words that will help users find this channel when using the search tool. This field is optional.

6. Click **Create Content Channel**. The Content Channels control panel reappears, showing the new channel you have created. You can now proceed by choosing another Personalize function, choosing another task, or opening another window.

Step 2: Define Access Rules for the Content Channel

Take any necessary steps to control access to the content channel. For general information about access control, refer to Controlling Access to Portal Content.

Step 3: Use the Personalize Tool to Make the Content Channel Appear as a Selection on the Portal

Depending on who has access permission to the content channel, you can use one of the following methods to make it appear as a selection on the portal:

- The public content administrator can use the Personalize feature to add the content channel to a

public list. When the public content administrator performs this function, the portal application displays a list box containing all public content channels. From the list box, the administrator selects the channels that are to appear on the public list.

- Group owners can use the Personalize feature to add the channel to a role-based list. When the group owner performs this function, the portal application displays a list box containing all role-based content channels that are available to the group. From the list box, the group owner selects the channels that are to appear on the role-based list.
- Individual users can use the Personalize feature to add the channel to a personal list. When the user performs this function, the portal application displays a list box containing all content channels in the enterprise directory that the user is authorized to access. From the list box, the user selects the channels that are to appear on his or her personal list.

For more information, refer to the Personalization topic in the portal User's Guide.

Adding Documents to the Portal

To add a document to the portal, follow these steps:

1. Add the Document to the Portal's Enterprise Directory
2. Place the document in a secure directory.
3. Define access rules for the document.
4. Use the Personalize tool to make the document appear as a selection on the portal.

When a portal user selects a document for viewing, the browser displays it using the appropriate software based on the MIME type that is assigned to the document.

Step 1: Add the Document to the Portal's Enterprise Directory

Because of potential security issues, users cannot add documents to the portal using the Personalization feature.

If you would like users at your site to be able to add documents to the portal, it is recommended that you develop a custom interface to provide this function. The interface should:

- Require documents to be placed in a secure directory. The directory should be accessible only to the user (or group of users) that owns the document.
- Create an object for the document under the appropriate portal user (or user group) on the enterprise directory. The object should have the following attributes:

objectclass

The object class identifier. For documents, this is always `sas-PortalDocument`.

sas-contentprovidertype

The type of portal content. For documents, this is always `file`.

saslabel

The name that will identify the document when it is displayed in the portal.

saspath

The complete path, including the filename, for the document.

description

A short description for the document. The description will appear after the name when the document is displayed on a list.

sas-contentmimetype

The MIME type for the document.

saskeyword

Key words that will help users find this document when they use the Search tool. This field is optional.

As an alternative to developing a custom interface, an authorized system administrator can use the enterprise directory console to add the metadata to the directory.

Once a document has been added to the portal, the user who owns the document can use the Personalize tool to edit the name, description, or key words. For more information, refer to *Creating and Editing Personal Documents* and *Creating and Editing Role-based Documents* in the portal User's Guide. To edit a public document, the procedure is the same except that you must sign on with public content administrator privileges, and then choose **Public** as the role to personalize.

Step 2: Place the Document in the Appropriate Directory

Place the document in the path that was specified in the metadata in step 1.

Note: To protect access to documents, you may want to create a separate document directory for each user and/or group, and make each directory accessible only to the portal Web application and the appropriate user or group.

Step 3: Define Access Rules for the Document

Take any necessary steps to control the portal's access to the document. For general information about access control, refer to *Controlling Access to Portal Content*.

Step 4: Use the Personalize Tool to Make the Document Appear as a Selection on the Portal

Depending on who has access permission to the document, you can use one of the following methods to make it appear as a selection on the portal:

- The public content administrator can use the Personalize feature to add the document to a public list. When the public content administrator performs this function, the portal application displays a list box containing all public documents. From the list box, the administrator selects the documents that are to appear on the public list.
- Group owners can use the Personalize feature to add the document to a role-based list. When the group owner performs this function, the portal application displays a list box containing all role-based documents that are available to the group. From the list box, the group owner selects the documents that are to appear on the role-based list.
- Individual users can use the Personalize feature to add the document to a personal list. When the user performs this function, the portal application displays a list box containing all documents in the enterprise directory that the user is authorized to access. From the list box, the user selects the documents that are to appear on his or her personal list.

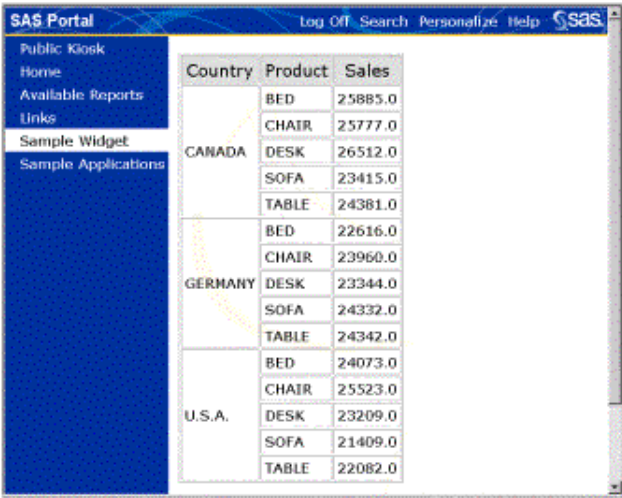
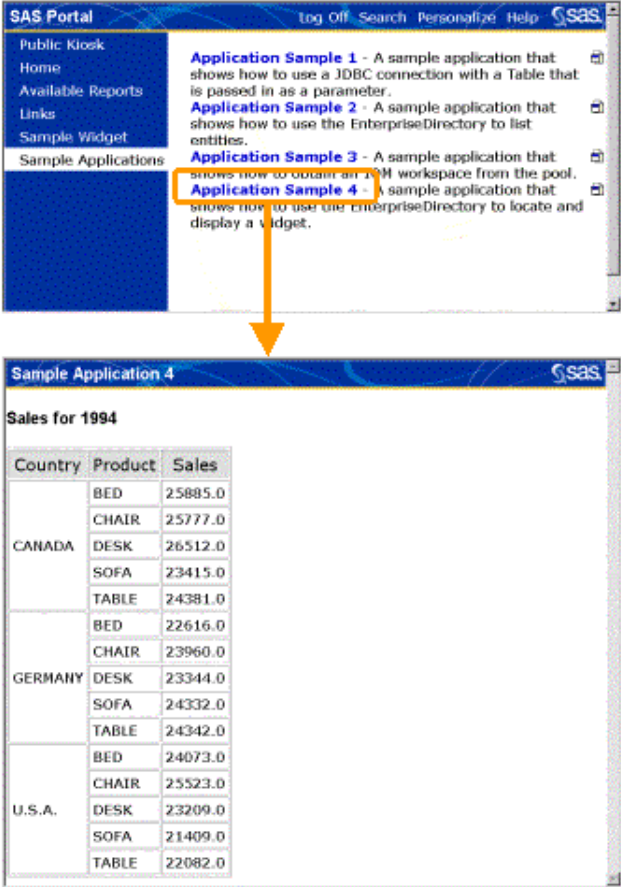
For more information, refer to the Personalization topic in the portal User's Guide.

Adding Applications and Widgets to the Portal

To expand the functionality of the SAS Information Delivery Portal, you can develop your own custom Java Server Pages (JSPs) that take advantage of the portal's content, metadata, and security services. These services are available through the portal's Application Programming Interfaces (APIs). Using these APIs, you can implement robust data analysis and reporting features with minimal programming effort.

Custom JSPs can be deployed in the portal either as widgets or as applications. A **widget** (also referred to as a **portlet**) is a graphical user interface (GUI) component that is produced by a JSP. A widget can be hosted within a portal window or within a custom application. An **application** is a Web application deployed in the Portal that is composed of JSPs and other resources. Applications are launched from within the portal, and they run outside of the portal container. In either scenario, the portal framework provides a secure and convenient platform on which to deploy custom-developed JSPs and make them available to the appropriate users.

Both applications and widgets leverage the portal's APIs for content, metadata, and security services. However, they differ from one another in several ways, as described in the following table:

| | |
|---|---|
| A widget... | An application... |
| ...generates an HTML fragment that is hosted by the portal (or by a custom application that is launched from the portal). | ...is a Web application that is launched from the portal. |
| ...is a generic presentation component driven by parameters. | ...can perform any type of function, and is sometimes driven by parameters. |
| ...is subject to the navigation features of the portal (or other host application). | ...has its own navigation features. |
| <p>...appears within a window on the portal (or within the host application's user interface).</p>  | <p>...is launched in a separate browser window.</p>  |

Note: If an application does not use the services of the portal, you should implement it as a link from the portal, not as an application.

To design and develop a custom widget or application for deployment in the portal, you should have a working knowledge of JSPs, Java servlets, and the Java programming language. Follow these steps to develop a custom JSP (widget or application) and add it to the portal:

1. Design the widget or application.

2. Code the widget or application.
3. Install the widget or application and its associated files in the portal web application.
4. Use the Personalize tool to add the widget or application to the portal's enterprise directory.
5. Add the data sources to the portal's enterprise directory.
6. Define access rules for the widget or application and for the data sources.
7. Use the Personalize tool to make the widget or application appear as a selection on the portal.

Step 1: Design the Widget or Application

As you design a widget or application to be implemented in the SAS Information Delivery Portal, you should consider the following questions:

- **What parameters will be required?** The portal provides a mechanism for passing any type of parameter, or multiple parameters, to a widget or application. Through the use of parameters, you can develop a single generic widget or application and use it for multiple purposes. For example, a parameter could specify which data source an application is to access; or it could specify the type of graph that is to be surfaced as a widget. To implement a parameter, you must define it using the Personalization feature, as described in step 4
- **What data source will be used?** Determine which SAS table, MDDB, or other data source will provide input for the application or widget. So that you can leverage the portal's content and security features, be sure to place metadata for the data source on the portal's enterprise directory, as described in step 5. You can then use methods from the Enterprise Directory SDK to access this data.
- **What components can be shared as widgets?** When developing an application, you may want to implement some features by developing a re-usable widget to be displayed inside this and other applications. For an example of application code that locates and displays a widget, see Application Sample 4.
- **Does any state need to be persisted between sessions?** Determine if any state information from application sessions will need to be persisted, or retained, after the session is closed. If so, the application can use methods from the Enterprise Directory SDK to add new objects to the portal's enterprise directory or to update existing objects.

Step 2: Code the Widget or Application

To design and develop a custom widget or application for deployment in the portal, you should have a working knowledge of JSPs, Java servlets, and the Java programming language. If you wish, you can develop the widget or application using the SAS webAF development platform. The webAF software is part of SAS AppDev Studio, which is provided in the SAS Information Delivery Portal package. For information about webAF, see the webAF Software section of the AppDev Studio Web site.

To assist you in the development process, several sample applications and widgets are provided with the SAS Information Delivery Portal. You can view the source code for these samples, along with brief explanatory comments, by accessing the following links:

- Application Sample 1: Create a JDBC Connection
- Application Sample 2: Locate an Entity
- Application Sample 3: Obtain a SAS Workspace
- Application Sample 4: Locate and Display a Widget
- Widget Sample

When coding an application or a widget for deployment in the portal, it is important to follow these basic guidelines:

- Refer to the Application Programming Interfaces (API) Reference for detailed documentation of the Java classes that make up the Enterprise Directory System Development Kit (SDK) and Portal SDK. To ensure that your applications and widgets are tightly coupled with the portal, you should make use of the services provided by these classes whenever possible. The convenience methods in the LogicBean class, together with the name/value pairs defined in the ModelId class, are especially useful for accessing portal data. You can augment these classes with additional data as needed to support your application,
- As an authorization check, include code that ensures that the user is accessing the application or widget from inside the portal, rather than from an external application. Operation of the application or widget should be halted if the user did not obtain access from within the portal. An example of the code for performing this authorization is provided in each of the sample applications.
- When performing SAS functions or accessing SAS data, be sure that your code uses the SAS workspace manager to obtain a workspace on the SAS server. For an example of this code, see Application Sample 3: Obtain a SAS Workspace. For additional information, refer to Using the Java Workspace Factory on the SAS Integration Technologies Web site.

Step 3: Install the Widget or Application and Its Associated Files in the Portal Web Application

To install the widget or application in the portal Web application, follow these steps:

1. Move or copy the JSP that invokes the application or widget, along with its supporting resource files, to the appropriate path in the portal's setup directory. This path must be in or under the directory that contains the JSP files for the portal application. The four sample applications reside in the path /jsp/html/samples. For effective management of the portal, you should create a separate directory structure, such as /jsp/custom, for your custom widgets and applications.
2. If the application or widget requires Java classes, place them in the path Portal/WEB-INF/classes. The subdirectory name should match the name of the package that contains the classes.

3. If the application or widget requires JAR files, place them in the path Portal/WEB-INF/lib.
4. Run the PortalConfigure program, and deploy the resulting WAR file in the servlet container. For detailed installation instructions, see the readme file that accompanies the portal package.

Step 4: Use the Personalize Tool to Add the Widget or Application to the Portal's Enterprise Directory

Use the portal's Personalize tool to add the application or widget to the portal. This tool provides a graphical user interface for entering the metadata for the application or widget. The metadata includes the path and file name of the main JSP, as well as information about parameters and their default values. For detailed instructions, refer to Adding and Editing Personal Applications, Adding and Editing Role-Based Applications, Adding and Editing Personal Widgets, or Adding and Editing Role-Based Widgets in the portal User's Guide.

Step 5: Add the Data Sources to the Portal's Enterprise Directory

If your data sources have already been defined in the enterprise directory, you can skip this step.

To enable the widget or application to leverage the portal's content and security features, you must add metadata to the portal's enterprise directory to define each SAS table, MDDB, or other data source that will provide input. The SAS servers, spawners, and logins associated with the data sources must also be defined. You can then use methods from the Enterprise Directory SDK to access these resources.

To create metadata for SAS data sources, you can use the Integration Technologies Administrator application. For details, refer to the SAS Integration Technologies Administrator page on the SAS Integration Technologies Web site. For an overview of metadata requirements for the portal, refer to Adding SAS Content and related sections in this manual.

Note: Although the metadata for application and widget data sources must be added to the enterprise directory, it is not necessary for these data sources to be surfaced on portal lists.

Step 6: Define Access Rules for the Widget or Application and for Its Data Sources

Take any necessary steps to control access to the application or widget and its data sources. For general information about access control, refer to Controlling Access to Portal Content.

Step 7: Use the Personalize Tool to Make the Widget or Application Appear as a Selection on the Portal

Depending on who has access permission to the application or widget, you can use one of several methods to make it appear as a selection on the portal:

- The public content administrator can use the Personalize feature to add an application to a public list, or to create a public window that is defined by a widget. When the administrator performs one of these functions, the portal displays a list box containing all applications or widgets in the enterprise directory that are accessible to all portal users. From the list box, the public content administrator selects the desired applications or the desired widget.
- Group owners can use the Personalize feature to add an application to a role-based list, or to create a role-based window that is defined by a widget. When the group owner performs one of these functions, the portal displays a list box containing all applications or widgets in the enterprise directory that the group is authorized to access. From the list box, the group owner selects the desired applications or the desired widget.
- Individual users can use the Personalize feature to add an application to a personal list, or to create a personal window that is defined by a widget. When the user performs one of these functions, the portal displays a list box containing all applications or widgets in the enterprise directory that the user is authorized to access. From the list box, the user selects the desired applications or the desired widget.

For more information, refer to the Personalization topic in the portal User's Guide.

Application Sample 1: Create a JDBC Connection

A Java Server Page (JSP) called AppSample1.jsp is shown below. This JSP is shipped with the SAS Information Delivery Portal, and it can be found in the `jsp/html/samples` directory in the portal Web application.

The application creates a Java Data Base Connectivity (JDBC) connection with a SAS table. The table is identified to the application through Global Unique Identifier (GUID) that is passed into the application as a parameter.

JDBC is a programming interface that allows Java applications to access a database using Standard Query Language (SQL) statements.

```
<!-- Copyright (c) 2000 by SAS Institute Inc., Cary, NC 27513 -->
<%@ page
  language="java"
  import="com.sas.edir.Debug,
         com.sas.edir.TrackedObject,
         com.sas.edir.datasource.Library,
         com.sas.edir.datasource.Table,
         com.sas.edir.util.StringOp,
         com.sas.edir.webapp.Application,
         com.sas.edir.webapp.portal.PortalEnterpriseDirectory,
         com.sas.edir.webapp.portal.PortalJdbcConnector,
```



```

        com.sas.edir.webapp.portal.LogicBean,
        com.sas.edir.webapp.portal.RequestId,
        com.sas.edir.webapp.portal.samples.RB,
        java.sql.Connection,
        java.sql.ResultSet,
        java.sql.Statement,
        java.util.Iterator,
        java.util.Locale,
        java.util.Map,
        java.util.Properties"
    errorPage="PortalError.jsp"
%>

```

The following code uses logicBean convenience methods to process the request that has been passed from the portal session. These methods use the information in the request to locate the portal user's enterprise directory and the requested application.

```

<%
LogicBean logicBean = new LogicBean();
PortalEnterpriseDirectory edir = logicBean.getEnterpriseDirectory(request);
Application app = logicBean.getApplication(request);

```

The following code checks for the existence of the EnterpriseDirectory and Application objects. If they exist, this means the user has accessed the application from within the portal.

```

// authorization check
if ((edir != null) && (app != null))
{

```

The following code uses logicBean convenience methods to obtain the active locale setting and the names of currently active style sheets. It also obtains the application's name attribute from the enterprise directory, and it obtains the application's title text from the the samples resource bundle.

```

    // user is authorized to view this content
    Locale locale = logicBean.getLocale(request);
    Map model = logicBean.getModel(request);
    String sasStyleSheet = logicBean.getSASStyleSheet(request);
    String styleSheet = logicBean.getStyleSheet(request);
    String name = app.getName();
    String title = RB.getString(locale,"appsample1.title.txt");

```

The following code uses a logicBean convenience method to obtain application parameters from the portal session. In this case, the parameter is a Global Unique Identifier (GUID) that identifies a SAS table. **Note:** The portal's Personalization process assigns GUIDs to content items when they are added to the portal. To determine a content item's GUID, display the item's Properties window.

```

// check for application parameters which may get passed in as a request

```

```
// parameter or attribute
String tableGuid = logicBean.getValueFromRequest("com.sas.portal.AppSample1.Tabl
```

The following code uses the PortalTitle.jsp file to display the title at the top of the page. The appropriate style sheets and the application's title text are incorporated into the page.

```
// display the title using PortalTitle.jsp
request.setAttribute(RequestId.PortalTitle, title);
request.setAttribute(RequestId.PortalTitleFragment, "true");

%>
<HTML>
  <HEAD>
    <TITLE><%= title%></TITLE>
    <LINK rel=stylesheet href="<%= sasStyleSheet %>" type="text/css">
    <LINK rel=stylesheet href="<%= styleSheet %>" type="text/css">
  </HEAD>
  <BODY>
    <jsp:include page="/jsp/html/portal/PortalTitle.jsp" flush="true" />
  <%
```

The following code uses the GUID, which was passed in as a parameter, to locate the SAS table.

```
// use the GUID passed in to locate the table
Table table = (Table)logicBean.getEntityFromGuid(tableGuid, request);
Library library = table.getLibrary();

if (table != null)
```

If the table is found, the following code creates a JDBC connection to the table. It then submits an SQL statement that selects all of the table's rows and columns. Results of the query are placed in the "rs" variable. In this example, the result set is not actually used in the output; instead, the table name is assigned to the xArgs variable.

```
{
  // create a JDBC connection
  PortalJdbcConnector connector = new PortalJdbcConnector(model, table);
  Connection con = connector.getConnection();
  Statement select = con.createStatement();

  // execute an SQL query
  StringBuffer query = new StringBuffer();
  query.append("select * from ");
  query.append(library.getLibref());
  query.append(".");
  query.append(table.getMemberName());
  ResultSet rs = select.executeQuery(query.toString());

  // This is where you use the result set
  Object xArgs[] = new Object[1];
  xArgs[0] = table.getName();
```

The following code writes a message to the page, using the table name as an argument. The message text is obtained from the samples resource bundle.

```
%>
  <p><%= RB.formatString(locale, "appsample1.message.txt", xArgs) %>
<%
    // cleanup
    select.close();
    con.close();
    connector.close();
  }
}
else
{
```

The following code writes an error message if the authorization test failed. The message text is obtained from the samples resource bundle.

```
    // user is not authorized to view this content
%>
<HTML>
  <HEAD>
  </HEAD>
  <BODY>
  <p align=center><%= RB.getString(logicBean.getLocale(request), "content.notauthorize
  <%
  }
  %>
  </BODY>
</HTML>
```

Application Sample 2: Locate an Entity

A Java Server Page (JSP) called AppSample2.jsp is shown below. This JSP is shipped with the SAS Information Delivery Portal, and it can be found in the jsp\html\samples directory in the portal Web application.

The application shows how to use the Enterprise Directory classes to list content entities that match specific filtering criteria.

```
<!-- Copyright (c) 2000 by SAS Institute Inc., Cary, NC 27513 -->
<%@ page
  language=" java"
  import="com.sas.edir.Debug,
         com.sas.edir.FilteredEntityCollection,
         com.sas.edir.TrackedObject,
         com.sas.edir.util.StringOp,
         com.sas.edir.webapp.Application,
```

```

        com.sas.edir.webapp.ContentDocument,
        com.sas.edir.webapp.ContentDocumentFilter,
        com.sas.edir.webapp.portal.PortalEnterpriseDirectory,
        com.sas.edir.webapp.portal.LogicBean,
        com.sas.edir.webapp.portal.ModelId,
        com.sas.edir.webapp.portal.RequestId,
        com.sas.edir.webapp.portal.samples.RB,
        java.util.Iterator,
        java.util.Locale,
        java.util.Map,
        java.util.Properties"
    errorPage="PortalError.jsp"
%>

```

The following code uses logicBean convenience methods to process the request that has been passed from the portal session. These methods use the information in the request to locate the portal user's enterprise directory and the requested application.

```

<%
LogicBean logicBean = new LogicBean();
PortalEnterpriseDirectory edir = logicBean.getEnterpriseDirectory(request);
Application app = logicBean.getApplication(request);

```

The following code checks for the existence of the EnterpriseDirectory and Application objects. If they exist, this means the user has accessed the application from within the portal.

```

// authorization check
if ((edir != null) && (app != null))

```

The following code uses logicBean convenience methods to obtain the active locale setting and the names of currently active style sheets. It also obtains the application's name attribute from the enterprise directory, and it obtains the application's title text from the samples resource bundle.

```

{
    Locale locale = logicBean.getLocale(request);
    String sasStyleSheet = logicBean.getSASStyleSheet(request);
    String styleSheet = logicBean.getStyleSheet(request);
    String name = app.getName();
    String title = RB.getString(locale, "appsample2.title.txt");

```

The following code uses the PortalTitle.jsp file to display the title at the top of the page. The appropriate style sheets and the application's title text are incorporated into the page.

```

// display the title using PortalTitle.jsp
request.setAttribute(RequestId.PortalTitle, title);
request.setAttribute(RequestId.PortalTitleFragment, "true");

```

```

%>

```

```

<HTML>
  <HEAD>
    <TITLE><%= title %></TITLE>
    <LINK rel=stylesheet href="<%= sasStyleSheet %>" type="text/css">
    <LINK rel=stylesheet href="<%= styleSheet %>" type="text/css">
  </HEAD>
  <BODY>
    <jsp:include page="/jsp/html/portal/PortalTitle.jsp" flush="true" />
  </BODY>
</HTML>

```

The following code searches the enterprise directory to find portal content items which are documents and which have names beginning with the characters "Demo Document."

```

//
// locate all the documents by creating a filtered collection
//
// the EnterpriseDirectory is a factory that uses reflection to create objects
// by looking up the object implementation in PortalRegistry.properties
String filterMapping = edir.CLASS_MAPPING + "com.sas.edir.webapp.ContentDocument
ContentDocumentFilter filter = (ContentDocumentFilter)edir.newObject(filterMappi
// search the entire cn=sas tree
filter.setContext(logicBean.getPortalContext(edir.APPLICATION_CONTEXT, request))
filter.setName("Demo Document *");

// the filtered entity collection allows us to iterate over all the entities
// that match the filter
String collectionMapping = edir.CLASS_MAPPING + "com.sas.edir.FilteredEntityColl
Object[] args = { edir, new String("") };
FilteredEntityCollection collection = (FilteredEntityCollection)edir
collection.setFilter(filter);

```

The following code iterates over the search results and obtains the name and description of each document that met the filter criteria. The names and descriptions are written to the output page.

```

// iterate over the results
ContentDocument doc = null;
Iterator itor = collection.listIterator();
while (itor.hasNext())
{
    doc = (ContentDocument)itor.next();
%>
    <p><b><%= doc.getName() %></b> - <%= doc.getDescription() %>
<%
}
}
else
{

```

The following code writes an error message if the authorization test failed. The message text is obtained from the samples resource bundle.

```

// user is not authorized to view this content
%>

```

```
<HTML>
  <HEAD>
  </HEAD>
  <BODY>
<p align=center><%= RB.getString(logicBean.getLocale(request), "content.notauthorize
<%
}
%>
  </BODY>
</HTML>
```

Application Sample 3: Obtain a SAS[®] Workspace

A Java Server Page (JSP) called AppSample3.jsp is shown below. This JSP is shipped with the SAS Information Delivery Portal, and it can be found in the jsp\html\samples directory in the portal Web application.

The application shows how to use the Workspace Factory to obtain an IOM workspace on a SAS server. For a complete description of the workspace factory, refer to Using the Java Workspace Factory on the SAS Integration Technologies Web site.

```
<!-- Copyright (c) 2000 by SAS Institute Inc., Cary, NC 27513 -->
<%@ page
  language="java"
  import="com.sas.edir.Debug,
        com.sas.edir.EnterpriseDirectory,
        com.sas.edir.PortalPool,
        com.sas.edir.TrackedObject,
        com.sas.edir.WorkspaceFactoryManager,
        com.sas.edir.util.StringOp,
        com.sas.edir.webapp.Application,
        com.sas.edir.webapp.portal.PortalEnterpriseDirectory,
        com.sas.edir.webapp.portal.LogicBean,
        com.sas.edir.webapp.portal.ModelId,
        com.sas.edir.webapp.portal.RequestId,
        com.sas.edir.webapp.portal.samples.RB,
        com.sas.iom.SAS.IWorkspace,
        com.sas.iom.WorkspaceConnector,
        java.util.Hashtable,
        java.util.Locale,
        java.util.Map,
        java.util.Properties"
  errorPage="PortalError.jsp"
%>
```

The following code uses logicBean convenience methods to process the request that has been passed from the portal session. These methods use the information in the request to locate the portal user's enterprise directory and the requested application.

```
<%  
LogicBean logicBean = new LogicBean();  
PortalEnterpriseDirectory edir = logicBean.getEnterpriseDirectory(request);  
Application app = logicBean.getApplication(request);
```

The following code checks for the existence of the EnterpriseDirectory and Application objects. If they exist, this means the user has accessed the application from within the portal.

```
// authorization check  
if ((edir != null) && (app != null))  
{
```

If the user is found to be authorized, the following code uses logicBean convenience methods to obtain the active locale setting and the names of currently active style sheets. It also obtains the application's name attribute from the enterprise directory, and it obtains the application's title text from the samples resource bundle.

```
Locale locale = logicBean.getLocale(request);  
Map model = logicBean.getModel(request);  
String sasStyleSheet = logicBean.getSASStyleSheet(request);  
String styleSheet = logicBean.getStyleSheet(request);  
String name = app.getName();  
String title = RB.getString(locale, "appsample3.title.txt");
```

The following code uses a logicBean convenience method to obtain application parameters from the portal session. In this case, the parameter is a logical name, which the portal uses to identify the specific SAS server on which a workspace is to be obtained.

```
// check for application parameters which may get passed in as a request  
// parameter or attribute  
String logicalName = logicBean.getValueFromRequest("com.sas.portal.AppSample3.Lo
```

The following code uses the PortalTitle.jsp file to display the title at the top of the page. The appropriate style sheets and the application's title text are incorporated into the page.

```
// display the title using PortalTitle.jsp  
request.setAttribute(RequestId.PortalTitle, title);  
request.setAttribute(RequestId.PortalTitleFragment, "true");
```

```
%>  
<HTML>
```

```

<HEAD>
  <TITLE><%= title %></TITLE>
  <LINK rel=stylesheet href="<%= sasStyleSheet %>" type="text/css">
  <LINK rel=stylesheet href="<%= styleSheet %>" type="text/css">
</HEAD>
<BODY>
  <jsp:include page="/jsp/html/portal/PortalTitle.jsp" flush="true" />
<%

```

The following code uses the workspace factory to connect to the SAS server and obtain a workspace on the server.

```

// use the logical name passed in to determine where to create a workspace
if (!StringOp.isEmpty(logicalName))
{
  WorkspaceFactoryManager workspaceManager = (WorkspaceFactoryManager)model.ge
  WorkspaceConnector connector = workspaceManager.getWorkspaceConnector(edir,
  IWorkspace workspace = connector.getWorkspace();

  // This is where you use the workspace
  Object[] xArgs = new Object[2];
  xArgs[0] = workspace;
  xArgs[1] = logicalName;

```

The following code writes a message to the output page if the workspace operation was successful. The message text is obtained from the samples resource bundle.

```

%>
  <p><%= RB.formatString(locale, "appsample3.message.txt", xArgs) %>

```

The following code returns the workspace to the factory so that the object can be closed and the supporting connection can be either reused or canceled.

```

<%
  // cleanup
  try
  {
    workspace.Close();
  }
  catch (Exception ex)
  {
    // not a whole lot we can do
  }

  try
  {
    connector.close();
  }
  catch (Exception ex)
  {

```



```

        // not a whole lot we can do
    }
}
else
{

```

The following code writes an error message if the authorization test failed. The message text is obtained from the samples resource bundle.

```

    // user is not authorized to view this content
%>
<HTML>
  <HEAD>
  </HEAD>
  <BODY>
<p align=center><%= RB.getString(logicBean.getLocale(request), "content.notauthorize
<%
}
%>
  </BODY>
</HTML>

```

Application Sample 4: Locate and Display a Widget

A Java Server Page (JSP) called AppSample4.jsp is shown below. This JSP is shipped with the SAS Information Delivery Portal, and it can be found in the `jsp/html/samples` directory in the portal Web application.

The application shows how to locate a widget and display it in the application's output.

```

<!-- Copyright (c) 2000 by SAS Institute Inc., Cary, NC 27513 -->
<%@ page
  language="java"
  import="com.sas.edir.Debug,
    com.sas.edir.FilteredEntityCollection,
    com.sas.edir.TrackedObject,
    com.sas.edir.delivery.Parameter,
    com.sas.edir.util.StringOp,
    com.sas.edir.webapp.Application,
    com.sas.edir.webapp.Widget,
    com.sas.edir.webapp.WidgetFilter,
    com.sas.edir.webapp.portal.AttributeId,
    com.sas.edir.webapp.portal.PortalEnterpriseDirectory,
    com.sas.edir.webapp.portal.LogicBean,
    com.sas.edir.webapp.portal.ModelId,
    com.sas.edir.webapp.portal.RequestId,
    com.sas.edir.webapp.portal.samples.RB,

```

```

        java.util.Iterator,
        java.util.Locale,
        java.util.Map,
        java.util.Properties"
    errorPage="PortalError.jsp"
%>

```

The following code uses logicBean convenience methods to process the request that has been passed from the portal session. These methods use the information in the request to locate the portal user's enterprise directory and the requested application.

```

<%
LogicBean logicBean = new LogicBean();
PortalEnterpriseDirectory edir = logicBean.getEnterpriseDirectory(request);
Application app = logicBean.getApplication(request);

```

The following code checks for the existence of the EnterpriseDirectory and Application objects. If they exist, this means the user has accessed the application from within the portal.

```

// authorization check
if ((edir != null) && (app != null))
{

```

If the user is found to be authorized, the following code uses logicBean convenience methods to obtain the active locale setting and the names of currently active style sheets. It also obtains the application's name attribute from the enterprise directory, and it obtains the application's title text from the samples resource bundle. Finally, it creates a variable called "year" and assigns it the value "1994."

```

    Locale locale = logicBean.getLocale(request);
    String sasStyleSheet = logicBean.getSASStyleSheet(request);
    String styleSheet = logicBean.getStyleSheet(request);
    String title = RB.getString(locale, "appsample4.title.txt");
    String year = "1994";

```

The following code uses a logicBean convenience method to obtain application parameters from the portal session. In this case, the parameter is a logical name, which identifies the SAS server on which the widget's data source resides.

```

    // check for application parameters which may get passed in as a request
    // parameter or attribute
    String logicalName = logicBean.getValueFromRequest("com.sas.portal.AppSample4.Lo

```

The following code uses the PortalTitle.jsp file to display the title at the top of the page. The appropriate style sheets and the application's title text are incorporated into the page.

```

// display the title using PortalTitle.jsp
request.setAttribute(RequestId.PortalTitle, title);
request.setAttribute(RequestId.PortalTitleFragment, "true");

Object[] xArgs = new Object[1];
xArgs[0] = year;

%>
<HTML>
  <HEAD>
    <TITLE><%= title %></TITLE>
    <LINK rel=stylesheet href="<%= sasStyleSheet %>" type="text/css">
    <LINK rel=stylesheet href="<%= styleSheet %>" type="text/css">
  </HEAD>
  <BODY onload="startup()" onunload="shutdown()">
  <SCRIPT LANGUAGE="JavaScript">
  <!--
    // This script is to load all object onLoad() functions
    function startup(){ }
    function shutdown(){ }
  //-->
  </SCRIPT>

  <jsp:include page="/jsp/html/portal/PortalTitle.jsp" flush="true" />

  <h2><%= RB.formatString(locale, "appsampl4.message.txt", xArgs) %></h2>
<%

```

| |
|---|
| <p>The following code searches the enterprise directory to find widget content items which have the name "Widget Sample 1."</p> |
|---|

```

//
// locate the widget
//
// the EnterpriseDirectory is a factory that uses reflection to create objects
// by looking up the object implementation in PortalRegistry.properties
String filterMapping = edir.CLASS_MAPPING + "com.sas.edir.webapp.WidgetFilter";
WidgetFilter filter = (WidgetFilter)edir.newObject(filterMapping);
// search the entire cn=sas tree
filter.setContext(logicBean.getPortalContext(edir.APPLICATION_CONTEXT, request))
filter.setName("Widget Sample 1");

// the filtered entity collection allows us to iterate over all the entities
// that match the filter
String collectionMapping = edir.CLASS_MAPPING + "com.sas.edir.FilteredEntityColl
Object[] args = { edir, new String("") };
FilteredEntityCollection collection = (FilteredEntityCollection)edi
collection.setFilter(filter);

Widget widget = null;
Iterator itor = collection.listIterator();
if (itor.hasNext())
    widget = (Widget)itor.next();

if (widget != null)
{

```

The following code passes parameters from the application to the widget. The parameters include the logical name that was passed into the application, and the year "1994" which is hard-coded in the application.

```
itor = widget.getParameterIterator();
while (itor.hasNext())
{
    Parameter param = (Parameter)itor.next();
    String name = (String)param.get(Parameter.NAME);
    String value = (String)param.get(Parameter.DEFAULT_VALUE);

    if (StringOp.equalsIgnoreCase(name, "com.sas.portal.WidgetSample1.Logica
        request.setAttribute(name, logicalName);
    else if (StringOp.equalsIgnoreCase(name, "com.sas.portal.WidgetSample1.Y
        request.setAttribute(name, year);
    else if (!StringOp.isEmpty(name) && !StringOp.isEmpty(value))
        request.setAttribute(name, value);
}
```

The following code includes the widget, causing the HTML fragment generated by the widget to be displayed on the application's output page.

```
// pass a handle to the widget into the widget jsp
request.setAttribute(AttributeId.Widget, widget);

String jsp = widget.getJsp();
%>
<jsp:include page="<%= jsp %>" flush="true" />
<%
}
}
else
{
```

The following code writes an error message if the authorization test failed. The message text is obtained from the samples resource bundle.

```
// user is not authorized to view this content
%>
<HTML>
<HEAD>
</HEAD>
<BODY>
<p align=center><%= RB.getString(logicBean.getLocale(request), "content.notauthorize
<%
}
%>
</BODY>
</HTML>
```

Widget Sample

A Java Server Page (JSP) called `WidgetSample1.jsp` is shown below. This JSP is shipped with the SAS Information Delivery Portal, and it can be found in the `jsp\html\samples` directory in the portal Web application.

This JSP executes an SQL query against a SAS table and writes the results to an HTML table.

```
<!-- Copyright (c) 2000 by SAS Institute Inc., Cary, NC 27513 -->
<%@ page
  language="java"
  import="com.sas.edir.Debug,
        com.sas.edir.EnterpriseDirectory,
        com.sas.edir.PortalPool,
        com.sas.edir.TrackedObject,
        com.sas.edir.WorkspaceFactoryManager,
        com.sas.edir.util.StringOp,
        com.sas.edir.webapp.Widget,
        com.sas.edir.webapp.portal.AttributeId,
        com.sas.edir.webapp.portal.LogicBean,
        com.sas.edir.webapp.portal.ModelId,
        com.sas.edir.webapp.portal.PortalEnterpriseDirectory,
        com.sas.edir.webapp.portal.RequestId,
        com.sas.edir.webapp.portal.samples.RB,
        com.sas.iom.SAS.ILanguageService,
        com.sas.iom.SAS.IWorkspace,
        com.sas.iom.WorkspaceConnector,
        com.sas.rio.MVACConnection,
        java.sql.Connection,
        java.sql.ResultSet,
        java.sql.Statement,
        java.util.ArrayList,
        java.util.Hashtable,
        java.util.Locale,
        java.util.Map,
        java.util.Properties"
  errorPage="PortalError.jsp"
%>
```

The following code uses `logicBean` convenience methods to process the request that has been passed from the portal session (or calling application). These methods use the information in the request to locate the portal user's enterprise directory and the requested widget.

```
<%
LogicBean logicBean = new LogicBean();
PortalEnterpriseDirectory edir = logicBean.getEnterpriseDirectory(request);
Widget widget = logicBean.getWidget(request);
```

The following code checks for the existence of the `EnterpriseDirectory` and `Widget` objects. If they exist, this means the user has accessed the widget from within the portal.

```
// authorization check
if ((edir != null) && (widget != null))
{
    // user is authorized to view this content
```

If the user is found to be authorized, the following code uses logicBean convenience methods to obtain information about the current session as well as the active locale setting.

```
Map model = logicBean.getModel(request);
Locale locale = logicBean.getLocale(request);
```

The following code uses a logicBean convenience method to obtain widget parameters from the portal session (or the calling application). The parameters include a year value and a logical name which identifies the SAS server on which the widget's data source resides.

```
// use widget parameters passed in as a request parameters or attributes
String logicalName = logicBean.getValueFromRequest("com.sas.portal.WidgetSample1
String year = logicBean.getValueFromRequest("com.sas.portal.WidgetSample1.Year",

if (!StringOp.isEmpty(logicalName))
```

If a logical name parameter is found, the following code uses the workspace factory to connect to the appropriate SAS server and to obtain a workspace on the server.

```
{
    WorkspaceFactoryManager workspaceManager = null;
    WorkspaceConnector connector = null;
    IWorkspace workspace = null;
    Connection con = null;
    Statement select = null;

    try
    {
        workspaceManager = (WorkspaceFactoryManager)model.get(ModelId.PortalWork
        connector = workspaceManager.getWorkspaceConnector(edir, logicalName);
        workspace = connector.getWorkspace();
```

The following code uses the Language Service interface to construct a SAS procedure step and submit it for processing. The procedure step reads data from the table "prdsale" and creates a temporary table called "wdgt1."/td>

```
ILanguageService lang = workspace.LanguageService();

// todo - need a unique output dataset name

StringBuffer submit = new StringBuffer();
submit.append("proc means data=sashelp.prdsale noprint SUM; ");
submit.append("output out=wdgt1 sum=actual; ");
submit.append("class country product year; ");
```

```

submit.append("var actual; ");
submit.append("run; ");
lang.Submit(submit.toString());

```

The following code creates a JDBC connection to the temporary table "wdgt1." It then submits an SQL statement that selects all rows in which the `_type_` column is equal to "7" and the year column is equal to the year parameter that was passed into the widget. The query results are returned to a variable called "rs."

```

// create a JDBC connection
con = new MVACConnection(workspace.DataService(), new Properties());
select = con.createStatement();

// execute an SQL query
StringBuffer query = new StringBuffer();
query.append("select country,product,actual from work.wdgt1 where _type_");
query.append(year);
ResultSet rs = select.executeQuery(query.toString());

```

The following code parses the result string and constructs an HTML table to present the data. This resulting HTML fragment can then be incorporated into the page that is output by the portal (or other calling application).

```

ArrayList country = new ArrayList();
ArrayList product = new ArrayList();
ArrayList sales = new ArrayList();

int count = 0;
while (rs.next())
{
    country.add(rs.getObject(1).toString().trim());
    product.add(rs.getObject(2).toString().trim());
    sales.add(rs.getObject(3).toString().trim());
    count++;
}

%>
<p>
<table border="1" cellpadding="4">
<tr>
    <th>Country</th>
    <th>Product</th>
    <th>Sales</th>
</tr>
<%

int nextCountry = 0;
for (int i = 0; i < count; i++)
{

%>
<tr>
<%

String countryS = (String)country.get(i);
String productS = (String)product.get(i);
String salesS = (String)sales.get(i);

```

```

        if (i == nextCountry)
        {
            int j;
            for (j = i; j < count; j++)
            {
                if (!StringOp.equals(countryS, (String)country.get(j)))
                {
                    nextCountry = j;
                    break;
                }
            }
        }
    }
}

<td rowspan="<%= j - i %>"><%= countryS %></td>
<%=
    }
}
<td><%= productS %></td>
<td><%= salesS %></td>
</tr>
<%=
    }
}
}
</table>
<%=
    }
    catch (Exception ex)
    {
        ex.printStackTrace();
    }
    finally
    {

```

The following code returns the workspace to the factory so that the object can be closed and the supporting connection can be either reused or canceled.

```

        // make sure and cleanup so the workspace can be released
        try { select.close(); }
        catch (Exception ex) { /* not a whole lot we can do */ }

        try { con.close(); }
        catch (Exception ex) { /* not a whole lot we can do */ }

        try { workspace.Close(); }
        catch (Exception ex) { /* not a whole lot we can do */ }

        try { connector.close(); }
        catch (Exception ex) { /* not a whole lot we can do */ }
    }
}
else

```

The following code writes an error message if the authorization test failed. The message text is obtained from the samples resource bundle.


```
{
    // user is not authorized to view this content
%>
<p align=center><%= RB.getString(logicBean.getLocale(request), "content.notauthorize
<%
}>
```

Localizing the Portal

The SAS Information Delivery Portal can be localized to meet the language requirements of a particular country or region. To localize the portal, follow these steps:

1. If necessary, create a resource bundle for the needed language(s). The resource bundle is a file containing the fixed text that appears on the Portal's graphical user interface, including text for menus, buttons, field names, and messages. The portal application provides resource bundles in the following languages: U.S. English, Chinese, German, Italian, Japanese, Korean, and Polish.
2. Translate the portal's metadata. The metadata, which is stored in the enterprise directory, includes the text associated with Portal windows, subwindows, lists, and content items (such as link names, report names, channel names, etc.). The portal supports only one set of metadata per installation.
3. Customize the portal's appearance, if needed, to incorporate the appropriate colors, type style, etc.

Details for each step are provided below.

Note: The webAF JAR files shipped with the portal are translated into the following languages: Czech, German, Spanish, French, Hungarian, Italian, Polish, and Turkish. Therefore, the portal's viewers for MDDBs and webEIS documents will display native text only in those languages.

Step 1: Create a Resource Bundle for the Needed Language(s)

The resource bundle is a file containing the fixed text that appears on the portal's graphical user interface, including text for menus, buttons, field names, and messages. The default (U.S. English) resource bundle provided with the portal is called Resources.properties. It can be found in the portal setup directory in the following path: Portal1.1\WEB-INF\classes\com\sas\edir\webapp\portal\res. This path also contains resource bundles for the following languages: Chinese, German, Italian, Japanese, Korean, and Polish.

If necessary, you can create additional resource bundles to support your user base. To create a resource bundle in a different language:

- a. Copy the Resources.properties file into a new file.

- b. Name the new file to designate the appropriate language or language and country. For example:
- A German-language resource bundle would be named `Resources_de.properties`.
 - A German-language resource bundle for the country of Germany would be named `Resources_de_DE.properties`.
 - A German-language resource bundle for the country of Austria would be named `Resources_de_AT.properties`.

For a list of standard locale codes, refer to the Open Group Locale standards.

- c. Translate the text strings in the new file into the desired language.

Note: When translating the resource bundle into languages that use native-encoded characters (characters that are non-Latin 1 and non-Unicode), you must convert those characters into Unicode escape sequences. To perform the conversion, you can use a tool called `native2ascii`. For information about this tool, scroll to the bottom of the Java Internationalization page on the Sun Microsystems Web site.

- d. Save the new file and place it in the same path as the original file.

Most browser software allows users to specify their language preferences. For example, users of Release 5 of Microsoft Internet Explorer can choose **Tools -> Internet Options -> Languages** and then select one or more languages in the order of preference. Windows users can also use the **Internet Options** icon on the Windows Control Panel to specify language preferences.

When a user accesses the portal, the portal application queries the user's browser software and operating system software to determine the user's preferred language. The portal then loads the resource bundle whose name matches the user's language specification. If the portal application does not have a resource bundle for the user's preferred language(s), the `Resources.properties` file is loaded.

The user's language preference affects *only* the portal's fixed text such as menus, buttons, field names and messages. The user-defined text that appears in the portal (including the names of portal windows, subwindows, lists, link names, report names, and channel names) is defined by metadata stored in the enterprise directory. The portal supports only one set of metadata per installation, as described in Step 2. In addition, content that is supplied by a SAS server will be in the language specified for that SAS server.

Note: You will find properties for default locale and supported locales in the portal's `Installation.properties` and `Portal.properties` files. However, these properties are not used in Release 1.1 of the portal.

Step 2: Translate the Metadata

The metadata, which is stored in the enterprise directory, includes the text associated with Portal windows, subwindows, lists, and content items (such as link names, report names, channel names, etc.). The portal product includes a demonstration set of metadata in U.S. English. If your primary user base has a different language preference, you can translate the starting metadata into that language.

The portal supports only one set of metadata per installation. Therefore, the user's language preference does not affect the metadata that is displayed on portal screens.

Step 3: Customize the Portal's Appearance

To give the portal a look and feel appropriate to your locale, you can modify the portal's banner, background image, and company logo. For instructions, refer to the Illustrated Guide to Portal Graphics. In addition, you can modify the cascading style sheets that control the portal's text and background colors. For instructions, see the Illustrated Guide to Portal Styles. The customizations apply uniformly to all users of a given portal, regardless of their language preference.

Customizing the Portal's Appearance

When implementing the SAS Information Delivery Portal, you can easily customize the portal's appearance to meet the requirements of your organization. You can customize the appearance in the following ways:

- Modify the portal's cascading style sheets to incorporate different colors of background and text. Cascading style sheets are a standard mechanism for defining consistent and reusable formatting instructions for Web-based content.
- Modify or replace the portal's main graphical elements, including the company logo, banner, and background image.

Note: You can also modify the language or wording used for menu items, messages, and other text in the portal. For detailed instructions, see [Localizing the Portal](#).

To customize the portal's appearance, follow these steps:

1. Locate the themes directory and files.

The **themes\default** directory contains the files that control the appearance of the SAS Information Delivery Portal. When you first download and unzip the portal software, you can find this directory in the portal setup directory. When you are finished installing the portal, you can find this directory within the portal application on the Web server. The directory contains the following files:

- Portal.css
- sasStyle.css
- sasads.css
- Banner_primary.gif
- Logo.gif
- Background.gif

2. Back up the default theme.

Before starting to customize the portal's appearance, you should create a backup copy of the themes\default directory. You can place the copy in the themes directory and give it a name such as default_bak.

3. Determine which styles and graphics to modify.

Determine which aspects of the portal's appearance you would like to change. To determine which style sheets and graphics files you need to modify in order to implement these changes, refer to the following:

- Illustrated Guide to Portal Graphics
- Illustrated Guide to Portal Styles
- Reference Guide to Portal Styles

4. Modify the theme.

Make the desired changes to the files in the themes\default directory.

Invoking the Theme

The portal invokes style sheets based on settings in the portal's **resources.properties** file. You can find this file in the path \WEB-INF\classes\com\sas\edir\portal\res within the portal setup directory and application directory. The resources file contains the following settings:

```
# theme
theme.sasstylesheet={0}/themes/{1}/sasStyle.css
theme.stylesheet={0}/themes/{1}/Portal.css
theme.sasadsstylesheet={0}/themes/{1}/sasads.css
theme.imagedir={0}/themes/{1}/
```

These settings must contain the correct path and file name for each style sheet.

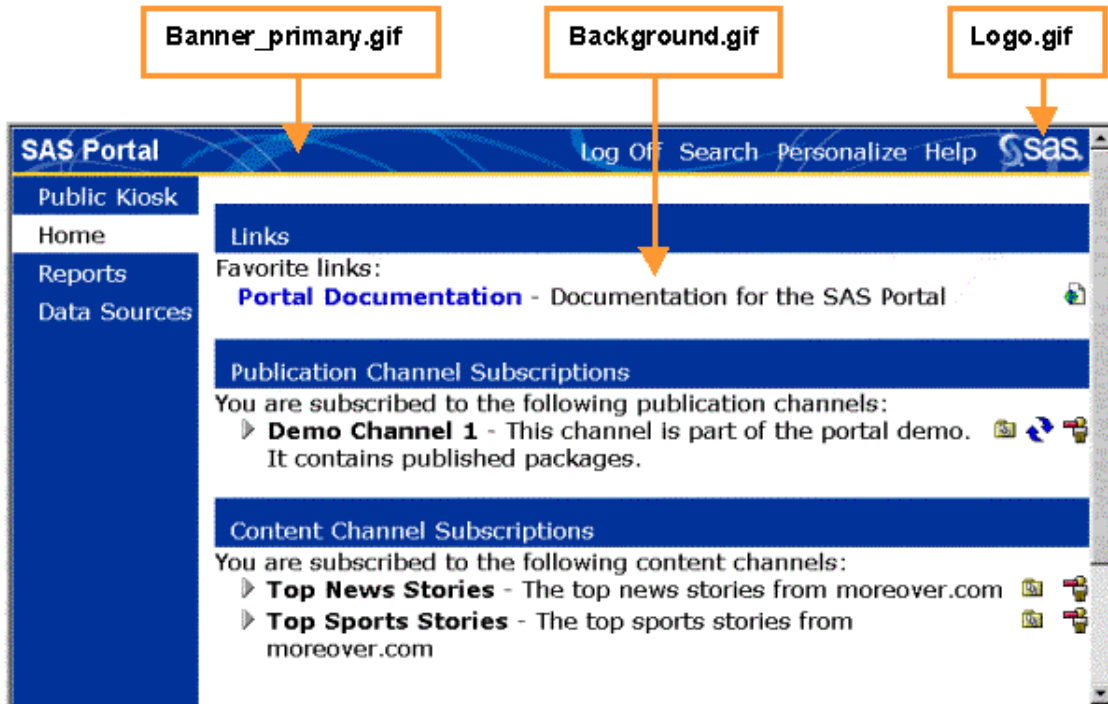
Restoring the Default Theme

At a later time, if you decide to revert to the original theme, you can assign a different name to the default directory (for example, my_default), and then rename the backup directory to "default."

Illustrated Guide to Portal Graphics

You can easily customize the appearance of the SAS Information Delivery Portal by modifying or replacing the main graphical elements. The images that can be modified or replaced are the banner, the background image, and the company logo. These images are stored as Graphical Interchange Format (GIF) files in the themes\default directory of the portal application.

The default graphics are shown in the following illustration:



- **Banner_primary.gif.** This image is stretched across the top of the portal's interface. You can use any of these methods to modify the banner:
 - Use a graphics program to modify the Banner_primary.gif file.
 - Create or obtain a new image, and save it in the default directory using the name Banner_primary.gif. (The new file should replace the original one.)
 - Use a solid color in the banner area instead of an image. To do so, you must modify the bannerBackground class in the Portal.css style sheet.
- **Background.gif.** This background image is tiled in the main, or working, area of the portal's interface. You can use any of these methods to modify the background:
 - Use a graphics program to modify the Background.gif file.
 - Create or obtain a new image, and save it in the default directory using the name Background.gif. (The new file should replace the original one.)
 - Use a solid color in the working area instead of an image. To do so, you must modify the workarea class in the Portal.css style sheet.
- **Logo.gif.** This file contains the SAS logo, which has a transparent background. To replace it, you

can move your organization's logo to the default directory and rename it to logo.gif. The new file should replace the original one.

Illustrated Guide to Portal Styles

You can easily customize the appearance of the SAS Information Delivery Portal by modifying the cascading style sheets that control the portal's text and background colors. Cascading style sheets are a standard mechanism for defining consistent and reusable formatting instructions for Web-based content.

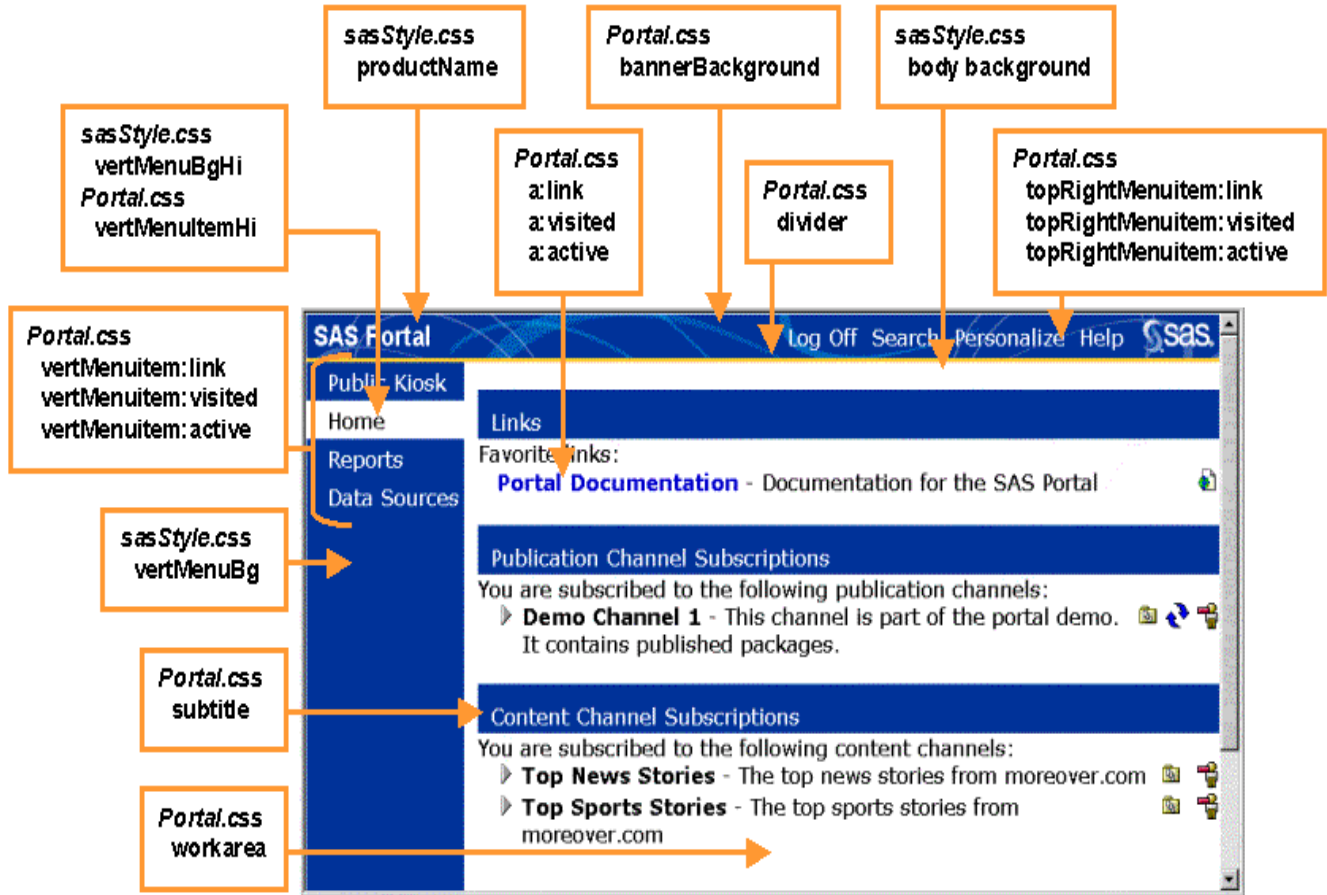
The portal makes use of the following style sheets: **Portal.css**, **sasStyle.css**, and **sasads.css**. These style sheets are stored in the themes\default directory of the portal application. The following illustrations show how the styles in Portal.css and sasStyle.css styles affect the appearance of the various Portal screens:

- Main portal screen
- Portal Package Viewer screen
- Portal Table Viewer screen
- Portal Personalization screen

For more information, you can also refer to the alphabetical Reference Guide to Portal Styles. For information about the sasads.css style sheet, which affects the display of MDDBs and webEIS documents in the portal, refer to the TransformationBean Style Sheet Reference on the AppDev Studio Web Site.

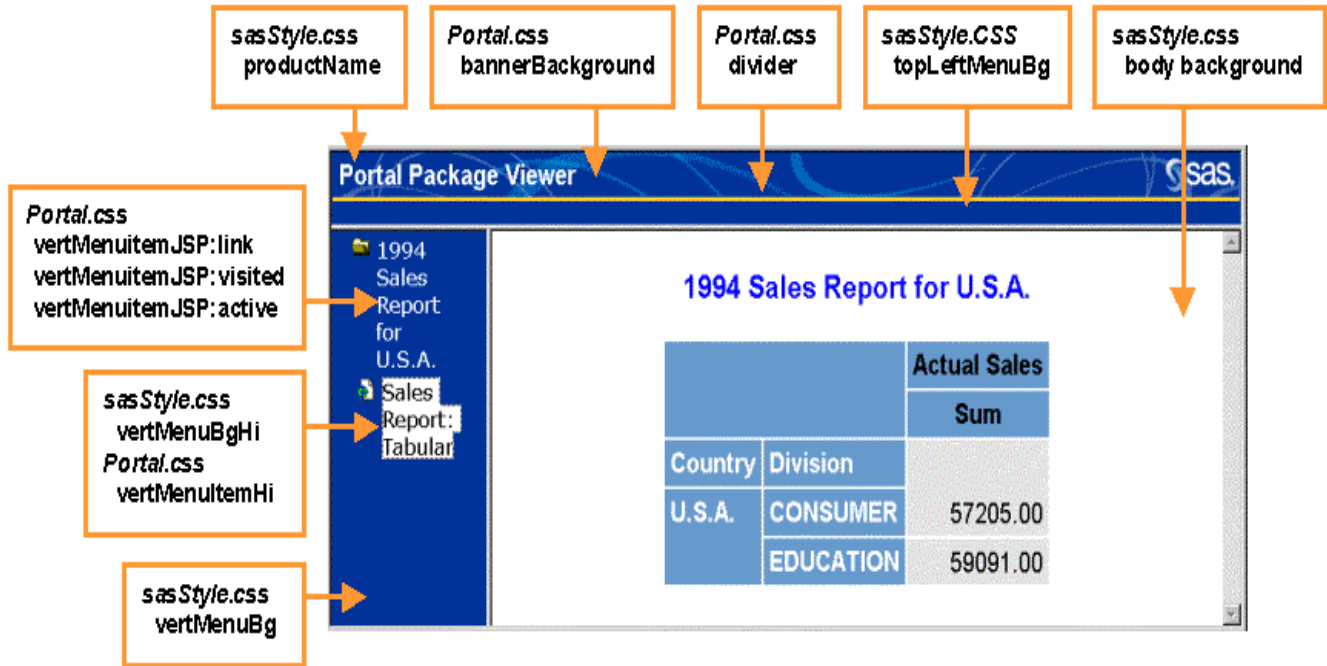
Styles on the Main Portal Screen

The following illustration shows which areas of the main portal screen are affected by the styles in Portal.css and sasStyle.css. To display more information about a style, click on the style name.



Styles on the Portal Package Viewer Screen

The following illustration shows which areas of the Portal Package Reader screen (and similar screens) are affected by the styles in Portal.css and sasStyle.css. To display more information about a style, click on the style name.



Styles on the Portal Table Viewer Screen

The following illustration shows which areas of the Portal Table Reader screen (and similar screens) are affected by the styles in `Portal.css` and `sasStyle.css`. To display more information about a style, click on the style name.

The screenshot displays the 'Portal Table Viewer' interface. At the top, a blue banner contains the text 'Portal Table Viewer' and the SAS logo. Below the banner is the title 'Product Sales Summary' and a table with the following data:

| Year | Month | Product Type | Region | Actual Sales |
|------|------------|--------------|--------|--------------|
| 1993 | 1993-01-01 | FURNITURE | EAST | 925.0 |
| 1993 | 1993-02-01 | FURNITURE | EAST | 999.0 |
| 1993 | 1993-03-01 | FURNITURE | EAST | 608.0 |
| 1993 | 1993-04-01 | FURNITURE | EAST | 642.0 |
| 1993 | 1993-05-01 | FURNITURE | EAST | 656.0 |

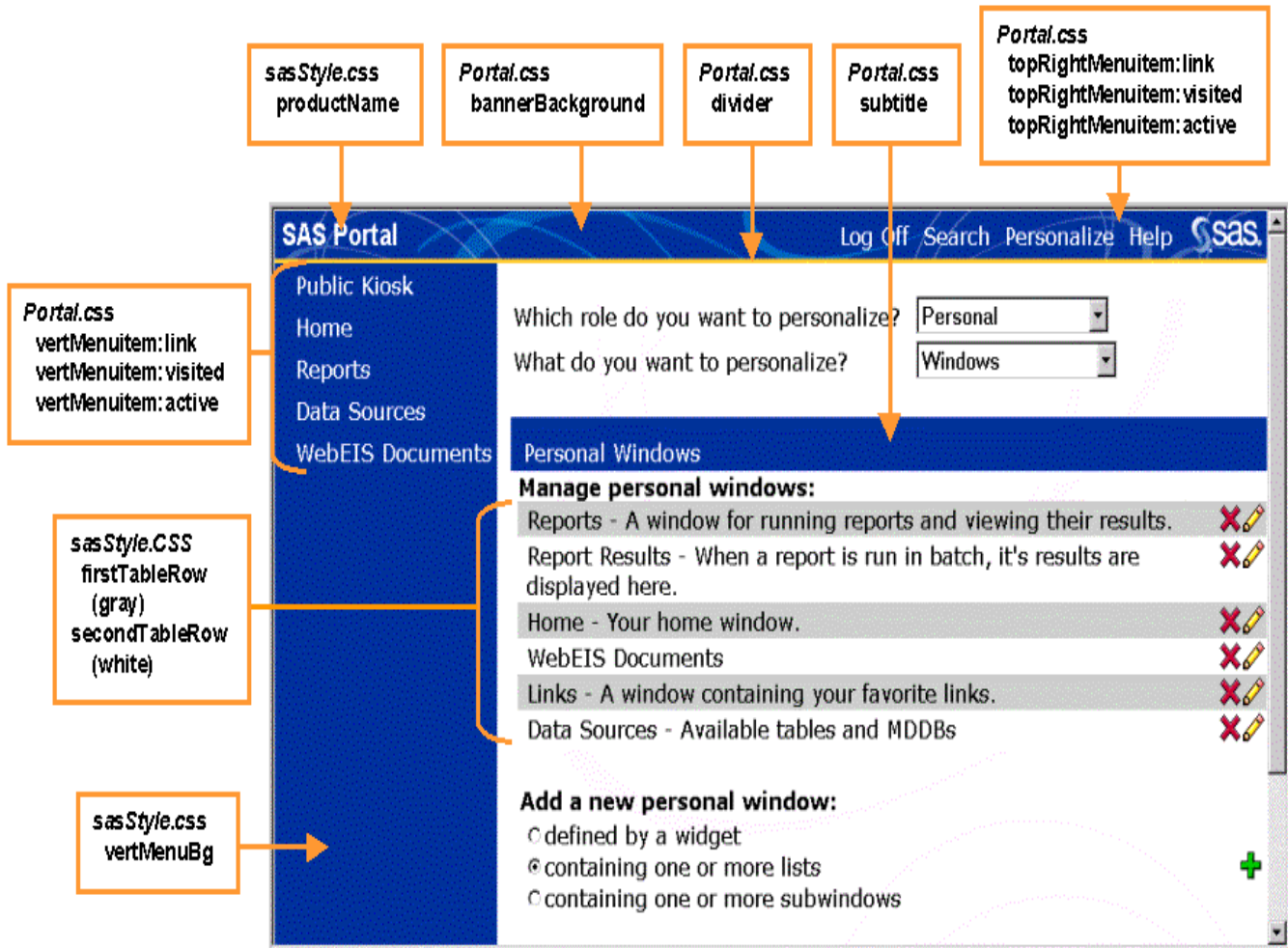
Below the table are navigation links: '< Previous Page' and 'Next Page >'. To the right, it says 'Showing rows 1 - 5'. Below the table is a 'Filtering Criteria' section with a 'Show Columns:' label and a list box containing: Division - [DIVISION], Country - [COUNTRY], Quarter - [QUARTER], Year - [YEAR], Month - [MONTH], Product Type - [PRODTYPE], Region - [REGION], and Actual Sales - [ACTUAL]. Below the list box are 'Move Up' and 'Move Down' buttons. There is a 'Where:' label followed by an empty text input field. Below that is 'Maximum Rows:' followed by a text input field containing the number '5'. At the bottom right of the filtering section is a 'Filter' button. The browser's address bar at the bottom shows 'Local intranet'.

Callout boxes identify the following styles:

- `sasStyle.css productName` points to the 'Portal Table Viewer' header.
- `Portal.css bannerBackground` points to the blue banner background.
- `Portal.css divider` points to the vertical line separating the banner from the content.
- `sasStyle.css body background` points to the overall page background.
- `sasStyle.css table.data th` points to the table header row.
- `sasStyle.css table.data td` points to the table data rows.
- `sasStyle.css button` points to the 'Filter' button.

Styles on the Personalization Screen

The following illustration shows which areas of the Portal Personalization screen are affected by the styles in Portal.css and sasStyle.css. To display more information about a style, click on the style name.



For more information refer to the alphabetical Reference Guide to Portal Styles.

Reference Guide to Portal Styles

You can easily customize the appearance of the SAS Information Delivery Portal by modifying the cascading style sheets that control the portal's text and background colors. Cascading style sheets are a standard mechanism for defining consistent and reusable formatting instructions for Web-based content. The portal makes use of the following style sheets: **Portal.css**, **sasStyle.css**, and **sasads.css**. These style sheets are stored in the themes\default directory of the portal application.

The styles contained in the Portal.css and sasStyle.css style sheets are documented below. For a visual representation of the areas of the portal that are affected by these styles, refer to the Illustrated Guide to Portal Styles.

For information about the sasads.css style sheet, which affects the display of MDDBs and webEIS

documents in the portal, refer to the TransformationBean Style Sheet Reference on the AppDev Studio Web Site.

Styles in Portal.css

Portal.css is a style sheet that was created specifically for the SAS Information Delivery Portal. It contains the following styles, listed here in alphabetical order:

| CSS File | Class | Presentation Component Affected | Default |
|------------|---------------------------------|--|--|
| Portal.css | a:link a:visited a:active | Text color for Web links that appear in the main area of the portal. You can set different colors for links that have been visited during the current session and the link that is active (just selected). See illustration. | <i>Link, visited, and active:</i> #0000CC (blue) |
| Portal.css | bannerBackground | Image and background color for the banner area. See illustration. | <i>Image:</i> Banner_primary.gif <i>Color:</i> #003399 (dark blue) |
| Portal.css | Body background | Background color for areas that are not defined by an image or which are not set explicitly in other stylesheets. | #ffffff (white) |
| Portal.css | divider | Height and color of the divider that appears below the banner. To hide the divider, set the height to 0. See illustration. | <i>Color:</i> #ffcc33 (yellow) <i>Height:</i> 3px |
| Portal.css | firstTableRow secondTableRow | Alternating background colors for items on lists that appear on personalization panels. See illustration. | <i>First row:</i> #ffffff (white) <i>Second row:</i> #CCCCCC (silver) <i>...and so on.</i> |

| | | | |
|------------|--|--|--|
| Portal.css | subtitle | Text color and background color for subwindow titles. These are the title bars that appear in the main portal window. See illustration. | <i>Text:</i> #ffffff (white); <i>Background:</i> #003399 (dark blue) |
| Portal.css | topRightMenuItem:link topRightMenuItem:visited topRightMenuItem:active | Text color for links in the task bar (top right menu). You can set different colors for links that have been visited during the current session and the link that is active (just selected). See illustration. | <i>Link, visited, and active:</i> #FFFFFF (white) |
| Portal.css | vertMenuItem:link vertMenuItem:visited vertMenuItem:active | Text color for links (window names) in the navigation bar, or top left menu. You can set different colors for links that have been visited during the current session and the link that is active (just selected). See illustration. | <i>Link, visited, and active:</i> #FFFFFF (white) |
| Portal.css | vertMenuItemHi | Background of currently open window name in the navigation bar (top left menu). See illustration. | <i>Background-color:</i> #FFFFFF (white) |
| Portal.css | vertMenuItemJSP:link vertMenuItemJSP:visited vertMenuItemJSP:active | Text color for links in the top left menu of the Portal Package Viewer screen. You can set different colors for links that have been visited during the current session and the link that is active (just selected). See illustration. | <i>Link (text):</i> #FFFFFF (white) <i>Visited (text)</i> #FFFFFF (white) <i>Active (text):</i> #FFFFFF (white) <i>Active (background):</i> #000000 (black) |

| | | | |
|------------|----------|--|---|
| Portal.css | workarea | Background color and tiled background image for the work area on the main portal screen. See illustration. | <i>Image:</i> background.gif <i>Color:</i> #ffffff (white) |
|------------|----------|--|---|

Styles in sasStyle.css

The style sheet sasStyle.css provides a common look and feel for a variety of SAS products that are Web-based. The following styles from this style sheet are used in the SAS Information Delivery Portal:

| CSS File | Class | Presentation Component Affected | Default |
|-----------------|---------------|---|--|
| sasStyle.css | A:hover | Text style to be used when the cursor is placed over a link. | <i>Text-decoration:</i> underline |
| sasStyle.css | button | Background and font for buttons that invoke input actions. See illustration. | <i>background-color:</i> #FFFFE0 <i>font-family:</i> Arial, Helvetica |
| sasStyle.css | productName | Text font and color for the name that appears on the left side of the banner on all portal screens. See illustration. | <i>font-family:</i> Arial, Helvetica, sans-serif <i>font-size:</i> small <i>font-weight:</i> bold <i>color:</i> #FFFFFF(white) |
| sasStyle.css | table.data th | Column headings in Portal Table Viewer. See illustration. | <i>background-color:</i> #DCDCDC <i>border:</i> 1px solid #C0C0C0 <i>font-weight:</i> normal <i>font-size:</i> x-small |
| sasStyle.css | table.data td | Table cells (other than column headings) in Portal Table Viewer. See illustration. | <i>border:</i> 1px solid #C0C0C0 |

| | | | |
|--------------|---------------|--|---|
| sasStyle.css | topLeftMenuBg | Background color for the area under the divider on the Portal Package Viewer screen. See illustration. | #003399 (dark blue) |
| sasStyle.css | vertMenuBg | Background color of navigation bar (left menu). See illustration. | <i>Background color:</i> #003399 (dark blue) |
| sasStyle.css | vertMenuBgHi | Background of currently open window name in the navigation bar (top left menu). See illustration. | <i>Background-color:</i> #FFFFFF (white) |
