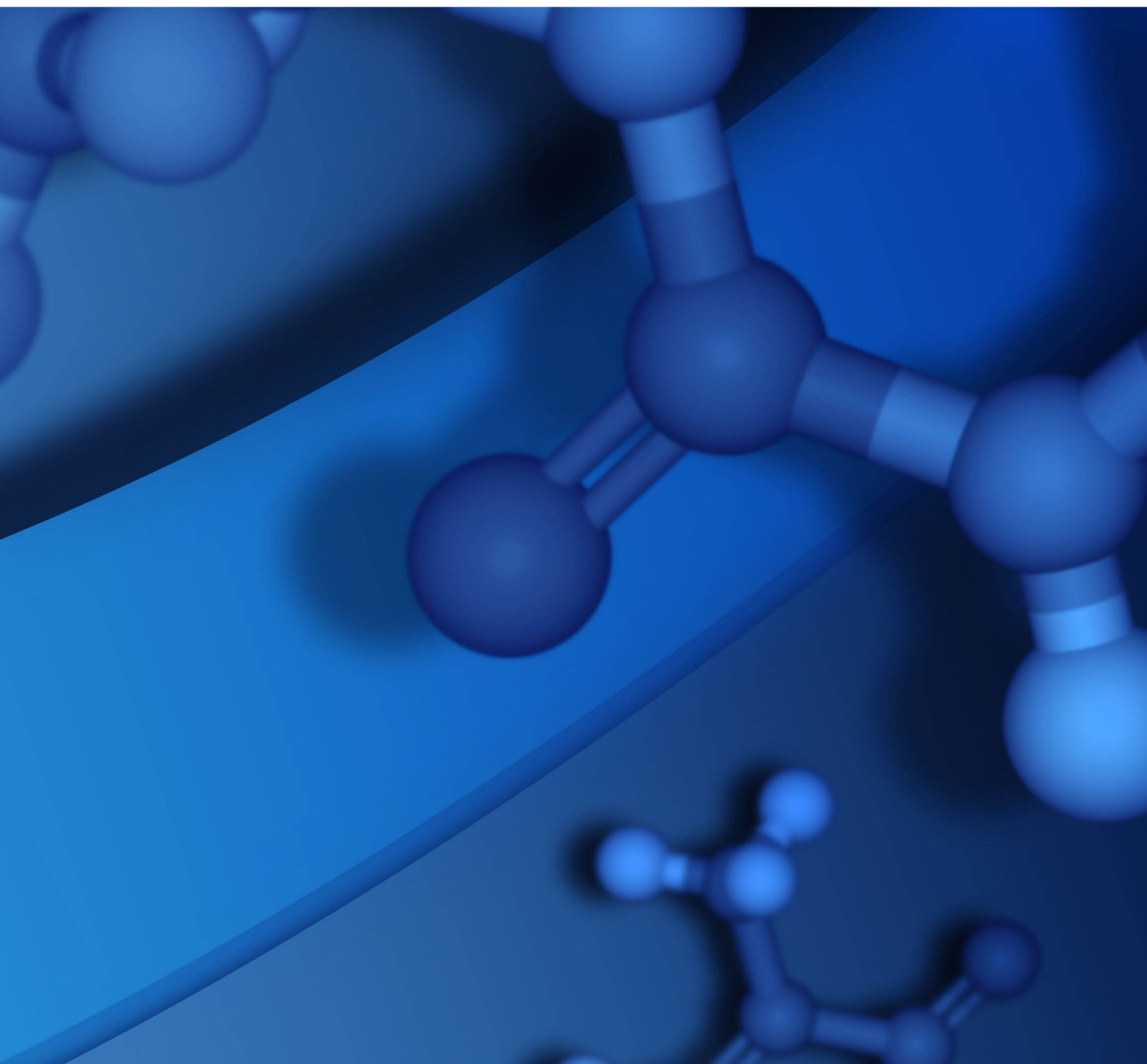


# ADMINISTRATION GUIDE

BIOVIA FOUNDATION HUB 2021 HF1



## Copyright Notice

©2021 Dassault Systèmes. All rights reserved. 3DEXPERIENCE, the Compass icon and the 3DS logo, CATIA, SOLIDWORKS, ENOVIA, DELMIA, SIMULIA, GEOVIA, EXALEAD, 3DVIA, 3DSWYM, BIOVIA, NETVIBES, IFWE and 3DEXCITE, are commercial trademarks or registered trademarks of Dassault Systèmes, a French "société européenne" (Versailles Commercial Register # B 322 306 440), or its subsidiaries in the U.S. and/or other countries. All other trademarks are owned by their respective owners. Use of any Dassault Systèmes or its subsidiaries trademarks is subject to their express written approval.

## Acknowledgments and References

To print photographs or files of computational results (figures and/or data) obtained by using Dassault Systèmes software, acknowledge the source in an appropriate format. For example:

"Computational results were obtained by using Dassault Systèmes BIOVIA software programs. BIOVIA Foundation Hub was used to perform the calculations and to generate the graphical results."

Dassault Systèmes may grant permission to republish or reprint its copyrighted materials. Requests should be submitted to Dassault Systèmes Customer Support, either by visiting <https://www.3ds.com/support/> and clicking **Call us** or **Submit a request**, or by writing to:

Dassault Systèmes Customer Support  
10, Rue Marcel Dassault  
78140 Vélizy-Villacoublay  
FRANCE

# Contents

<b>Chapter 1: Overview</b>	<b>1</b>
Getting Help	1
How Do I...	1
Best Practices for Administering Foundation Hub	2
What is BIOVIA Foundation?	3
What Is Foundation Hub?	3
Toolbar Links	3
REST API	3
<b>Chapter 2: Security Administration</b>	<b>4</b>
Managing Sessions	4
Managing Collaborative Spaces	5
Creating and Editing a Collaborative Space	6
Creating a Collaborative Space	6
Editing a Collaborative Space	6
Changing the Collaborative Space Type	8
Managing External Claims	8
Adding an External Claim	8
Deleting an External Claim	8
Example	9
Managing Groups	9
Adding a Group	9
Deleting a Group	9
Cloning a Group	10
Managing Roles	10
Adding a Custom Role	10
Cloning a Role	11
Default Roles	11
Managing Permissions	12
Foundation Hub Permissions	13
Pipeline Pilot Permissions	15
Compose and classic Capture Permissions	16
Experiment Permissions	17
Insight Permissions	17

Request Permissions .....	18
Workbook Permissions .....	19
Managing Security Events .....	21
Managing Trusted Certificates .....	21
Adding a Certificate .....	21
Troubleshooting .....	21
Setting Up User Authentication .....	22
Adding LDAP User Directories .....	22
Performance Considerations .....	24
Synchronizing LDAP Users .....	25
Configuring SPNEGO/Kerberos Authentication .....	25
Support .....	25
Prerequisites .....	26
Setup .....	26
Configuring Browsers for SPNEGO/Kerberos .....	26
Troubleshooting SPNEGO/Kerberos Setup .....	27
Setting Up User Authentication with 3DPassport .....	28
Configuring Foundation Hub to use 3DPassport Authentication .....	29
Configuring a 3DPassport user with Foundation Hub Admin Privileges .....	29
SSL Certificate for 3DPassport .....	29
Authentication Settings Matrix .....	30
Managing Users .....	31
Managing Users Directly in Foundation Hub .....	31
<b>Chapter 3: Lifecycle and Signature Policy Administration .....</b>	<b>34</b>
Lifecycle and Signature Policy Definitions .....	34
Example Lifecycle and Signature Policies .....	35
Understanding Signature Policies .....	35
Situations that Support Signature Policies .....	35
Signature Policy Signer Types and Basic Settings .....	36
Supported Signer Types .....	36
Signature Policies that Require More than One Signer .....	37
Signature Policies that Control Review and Approval Steps .....	37
Installed Signature Policies .....	38
Signature Policy for General Use .....	38
Signature Policies for Updating Compose Recipe States .....	38

Signature Policies for Updating Recipe Execution States .....	38
Signature Policies for Submitting Recipe Data .....	39
Signature Policy for Blocking an Action .....	40
Master Data Change Signature Policy .....	40
Lifecycle Actions and Default Signature Policy Assignments .....	42
Adding and Editing Signature Policies .....	46
Cloning Lifecycle Policies to Support Signature Policy Variations .....	48
Assigning Signature Policies to Lifecycle Actions .....	49
Requiring Signatures for Changes to Master Data .....	50
Managing Signature Policy Reason Codes .....	51
<b>Chapter 4: Vocabulary and Parameter Template Administration .....</b>	<b>52</b>
Vocabularies .....	52
Adding Vocabularies and Entries .....	52
Editing Vocabularies .....	53
Deleting Vocabularies .....	53
Cloning a Vocabulary .....	54
Adding Hierarchical Vocabularies .....	54
Viewing Vocabulary Entries .....	55
Viewing Vocabulary Aliases .....	55
Parameter Templates .....	56
Creating a Parameter Template .....	56
Editing a Parameter Template .....	57
Deleting a Parameter Template .....	58
Cloning a Parameter Template .....	58
Managing the Parameter Template Life Cycle .....	58
Reviewing and Approving or Rejecting a Parameter Template .....	58
Creating a New Version of an Approved Parameter Template .....	59
Withdrawing an Approved Parameter Template from Use .....	59
<b>Chapter 5: Activity and Activity Plan Administration .....</b>	<b>61</b>
Activities .....	63
Adding an Activity .....	63
Configuring an Activity .....	64
Rounding Rule Examples .....	67
Creating Parameterized Child Activities .....	68
Managing the Activity Lifecycle .....	69

Reviewing and Approving or Rejecting an Activity .....	69
Creating a New Version of an Approved Activity .....	70
Withdrawing an Approved Activity from Use .....	71
Making an Activity Visible Across Collaborative Spaces .....	71
Moving an Activity to Another Collaborative Space .....	71
Cloning an Activity .....	72
Deleting an Activity .....	72
Activity Plans .....	73
Adding an Activity Plan .....	73
Configuring Activities in an Activity Plan .....	75
Editing Activity Plans .....	77
Managing the Activity Plan Lifecycle .....	78
Reviewing and Approving or Rejecting an Activity Plan .....	78
Creating a New Version of an Approved Activity Plan .....	79
Withdrawing an Approved Activity Plan from Use .....	79
Making an Activity Plan Visible Across Collaborative Spaces .....	80
Moving an Activity Plan to Another Collaborative Space .....	80
Cloning an Activity Plan .....	81
Deleting an Activity Plan .....	81
<b>Chapter 6: Pipeline Pilot Protocol Configuration .....</b>	<b>82</b>
Adding a Pipeline Pilot Protocol .....	82
Deleting a Pipeline Pilot Protocol .....	83
Protocol Types .....	83
Activity Protocols .....	84
Activity Headless Protocols: Requirements .....	84
Activity Headless Protocols: Foundation Hub Configuration .....	84
Activity Headless Protocol: Example .....	85
Activity Report Protocols: Requirements .....	86
Activity Report Protocols: Foundation Hub Configuration .....	86
Activity Report Protocol: Example .....	87
Lifecycle Action Protocols .....	87
Lifecycle Action Headless Protocols: Requirements .....	88
Lifecycle Action Headless Protocols: Foundation Hub Configuration .....	88
Lifecycle Action Report Protocols: Requirements .....	89
Lifecycle Action Report Protocols: Foundation Hub Configuration .....	89

Task Planner Results Protocols .....	90
Task Planner Results Protocols: Requirements .....	90
Task Planner Results Protocols: Foundation Hub Configuration .....	90
Task Planner Results Protocols: Examples .....	91
Lifecycle Subscriber Protocols .....	91
Lifecycle Subscriber Protocols: Requirements .....	92
Lifecycle Subscriber Protocols: Foundation Hub Configuration .....	92
Lifecycle Subscriber Protocol: Example .....	93
Protocol Troubleshooting .....	94
<b>Chapter 7: Other Resource Data Administration .....</b>	<b>96</b>
Equipment Measurements Store .....	96
The Measurements Page .....	97
Fields in the Measurements Table .....	97
Viewing Full Measurement Details .....	97
Viewing Details of a Measurement's Equipment .....	97
Measurement Details .....	97
Viewing Measurement Details .....	97
Viewing Attachments .....	98
Locations, Organizations, Cost Centers, and Contacts .....	99
Add Locations .....	99
Add Organizations .....	100
Add Contacts .....	100
Add Cost Centers .....	100
Cloning a Location .....	100
Cloning an Organization .....	101
Cloning a Contact .....	101
Cloning a Cost Center .....	101
Projects .....	102
Adding a Project .....	102
Deleting a Project .....	102
Cloning a Project .....	102
Sequence Templates .....	103
Adding a Sequence Template .....	103
Deleting a Sequence Template .....	103
Unit Types .....	103

Adding a Unit Type .....	103
Editing a Unit Type .....	103
Deleting a Unit Type .....	104
Units .....	104
Adding Units .....	104
Editing a Unit .....	104
Deleting a Unit .....	105
Unit Categories .....	105
Adding a Unit Category .....	105
Editing a Unit Category .....	105
Deleting a Unit Category .....	105
Cloning a Unit Category .....	105
Exporting and Importing Resources .....	106
Data That You Can Export and Import .....	106
REST Clients .....	107
Permissions Required .....	107
Exporting Resources .....	108
JSON OData Parameters .....	110
Default Export Payload .....	113
Contents of an Exported ZIP File .....	113
manifest.json .....	114
Entity Type JSON Files .....	116
exportReport.txt .....	117
Importing Resources .....	118
Editing Data Before Import .....	119
Importing Resources from a ZIP File .....	119
Importing Using a Program, or a Pipeline Pilot Protocol .....	120
Backing Out Imported Resources .....	120
<b>Chapter 8: Additional Configuration Settings .....</b>	<b>121</b>
Installed Application Settings .....	121
Applications .....	121
Viewing Application Details .....	121
Editing the Application Label .....	121
Installations .....	122
Viewing Installation Details .....	122



Editing Installation Details .....	122
Application Settings .....	122
Foundation Hub Application Settings .....	122
General .....	122
Equipment .....	123
Security .....	123
Messages .....	124
File Service Settings .....	124
Samples .....	125
Tasks .....	125
Master Data .....	126
Auditing .....	126
Renaming and Hiding Life Cycle Buttons .....	126
Editing the Life Cycle Configuration Setting .....	126
JSON Code .....	127
Extended Properties .....	128
Adding a List of Extended Properties to an Object .....	128
Foundation Hub Configuration Settings .....	129
Restarting the Server .....	129
Editing Configuration Settings .....	129
Web Server Configuration Settings .....	129
Database Configuration Settings .....	130
Hazelcast Configuration Settings .....	130
Certificate Configuration Settings .....	131
Authentication Provider Settings .....	131
Application Monitoring Settings .....	131
Resetting Your Oracle Password .....	131
Inventory System Connections .....	132
Adding an Inventory System .....	132
Mapping Properties for Imported Materials and Containers .....	133
Activating and Inactivating Inventory Systems .....	134
Legal Notices (Personal Data Protection) .....	134
Software License Files .....	134
Supported Applications .....	134
Adding a License File .....	134

Deleting a License File .....	135
Viewing Existing Licenses .....	135
Updating an Expired License .....	135
Application Links .....	135
Adding Links .....	135
Log Files .....	136
Trusted Certificates .....	136
User Directories .....	136
Managing LDAP User Directories .....	136
Synchronizing LDAP Users .....	136
<b>Chapter 9: Managing the Foundation Hub Service .....</b>	<b>138</b>
Windows Services Console .....	138
Foundation Hub Command Line Tools .....	138
<b>Chapter 10: Uninstalling Foundation Hub .....</b>	<b>139</b>
Windows .....	139
Linux .....	139
<b>Chapter 11: Troubleshooting .....</b>	<b>140</b>
General Issues .....	140
My license expired and I cannot access the Admin and Settings page to update it .....	140
Load-balanced Pipeline Pilot nodes are no longer registered with Foundation Hub after upgrading Pipeline Pilot .....	140
Cannot start the Foundation Hub because the Oracle password has expired .....	141
Equipment and instrument features are not enabled as expected .....	141
The Hostname for the installation machine has changed .....	141
Foundation Hub does not start, no error or redirect .....	141
Issues manually editing configuration files .....	141
Timeout while attempting to restart the Foundation Hub Server .....	141
The log file was not updated or saved .....	141
3DPassport Issues .....	141
Single log out not working when using 3DPassport authentication .....	141
Directed to 3DPassport login page and cannot sign in .....	142
User cannot sign in .....	142
Foundation Hub Logging .....	142
Viewing Log Files .....	142
Editing Logging Settings .....	142

Tomcat Logging .....	143
Tomcat Http Access Logging .....	143
Java Garbage Collection Logs .....	143
Using PuTTY to Test Connectivity .....	143
Running PuTTY .....	143
Accessing PuTTY Documentation .....	143
Resolving an Invalid Database Connection String .....	143
Changing the Database Password .....	144
Reverting Foundation Hub to the Default Configuration .....	144
<b>Dassault Systèmes Support Resources .....</b>	<b>145</b>



# Chapter 1:

## Overview

---

This guide provides instructions for setting up and administering BIOVIA Foundation. You access the setup and administration functions from the Foundation Hub **Admin and Settings** page, which groups the functions into the following areas:

- **Security:** Set up groups and users, assign permissions, review sessions, configure collaborative spaces, manage external claims, and review login events.
- **Resources:** Set up and manage master data for use by applications, such as equipment resources, site information, activities and activity plans, parameter templates, and vocabularies.
- **Settings:** Manage settings for applications, extended properties, server configuration, and more.

### Tips:

- To quickly access entities that have recently changed, use the links under **Recent Changes** on the **Admin and Settings** page.
- To quickly access a specific Admin and Setting page, start typing its name in the **Find Pages** box at the top-right corner of the main page **Admin and Settings** page.
- To view the complete history of changes from inception through deletion for the entity or entities on the page you are viewing, click the **History** button.
  - For pages that lists all entities of a specific type, such as Vocabularies, a corresponding History page shows the history of all entities of that type.
  - For pages that provide the details for a specific entity, for example Vocabularies > Carrier Gas, a corresponding History page shows the details for just that entity.
  - You can filter, sort, and save your filter and sort options for each history page.

For more information about history, see "Viewing History Records" in the *Foundation Hub User Guide* or the online help, which mirrors the *User Guide*. For instructions for filtering and sorting, see "Personalizing Widgets and the Task Planner" in the same documentation.


To return from a History page to the page you were previously viewing, click your browser's **Back** button.

## Getting Help

See common problems and solutions in [Troubleshooting](#). If you do not find a solution there, visit <https://www.3ds.com/support/>.

## How Do I...

Open Admin and Settings from the Foundation Hub home page?

- Click  in the toolbar. You must be signed in with a user account that has administrator privileges.

Check on active user sessions?

- Open **Admin and Settings > Security > Active Sessions**. See [Managing Sessions](#).

Sync with an LDAP directory for user authentication?

- See [Adding LDAP User Directories](#).

Provision users directly in Foundation Hub that are not in an LDAP?

- Open **Admin and Settings > Security > Users**. See [Managing Users](#).

Set up Groups>

- Open **Admin and Settings > Security > Groups**. See [Managing Groups](#).

Set up location- and organization-related data?

- See [Locations, Organizations, Cost Centers, and Contacts](#).

Set up laboratory equipment?

- Use **Settings > Equipment**. See the *BIOVIA Foundation Hub Equipment Guide*, which is available with the installation files and at <https://www.3ds.com/support/>.

Change my Hub configuration?

- Open **Admin and Settings > Settings > Hub Configuration**. See [Additional Configuration Settings](#).

Extend existing data objects with custom properties?

- See [Extended Properties](#).

Manage my installed applications and application settings?

- Open **Admin and Settings > Settings > Applications**. See [Installed Application Settings](#).

Manage my Foundation Hub services?

- See [Managing the Foundation Hub Service](#).

Change my log settings or view logs?

- Open **Admin and Settings > Settings > Logging**. See [Log Files](#).

Add a link to the Application Links feature?

- Open **Admin and Settings > Settings > Links**. See [Application Links](#).

## Best Practices for Administering Foundation Hub

The following best practices are recommended Foundation Hub:

- Restart Foundation Hub servers every 30 days. In a load-balanced environment, this task should be staggered so that only one Foundation Hub server is restarted at a time. You can also restart manually. See [Foundation Hub Configuration Settings](#).
- Archive Foundation Hub logs every 30 days. See [Log Files](#).
- Restart the Foundation Hub Service periodically. See [Managing the Foundation Hub Service](#).
- Check disk fragmentation every 3 months.
- Review Windows Application and System logs every 60 days for errors and correct them.
- Integrate with an existing network management environment for advanced service monitoring.
- For Oracle, set up alerts for expiration of the schema owner password, regularly monitor the tablespace, and set up backups.
- Set alerts for when server certificate will expire so that you can take remediation steps before it expires.

**Tip:** In Windows you can use Windows scheduled tasks feature to automate tasks such as server and service restarts and log file archival.

## What is BIOVIA Foundation?

BIOVIA Foundation refers to the integrated scientific environment to which BIOVIA applications (including Pipeline Pilot) are connected. BIOVIA Foundation includes the Pipeline Pilot protocol execution engine (to support extension of BIOVIA applications that run Pipeline Pilot protocols), implements security and administration of users, and includes Foundation Hub, amongst other capabilities.

## What Is Foundation Hub?

Foundation Hub serves as the authentication provider for all connected foundation applications and enables session management, single sign-on, and single-sign out across BIOVIA applications.

- **Admin and Settings:** Manage security, resources such as equipment, organization, and location information, and settings for Foundation Hub and other BIOVIA applications.
- **Landing Page:** The BIOVIA Landing page provides a dashboard for managing tasks, viewing notifications, and monitoring equipment that requires calibration. For more information, see the *Foundation Hub User Guide*.
- **Quick Access to Applications:** Click the **Applications** link in the Foundation Hub toolbar to access BIOVIA applications and any additional links configured for your organization.

## Toolbar Links

The toolbar at the top of Foundation Hub contains the following tools:

## REST API

Foundation Hub supports REST APIs that allow you to interact with Foundation Hub using standard GET, POST, PUT, etc. calls in JSON and XML format. To access the Foundation Hub REST API help, go to the **Foundation Hub server page > Admin and Settings > ? (Help) > API Reference Guide**.

## Chapter 2:

# Security Administration

---

Foundation provides identity management services to BIOVIA applications. Use the **Admin and Settings > Security** pages to manage security for your deployment.

- **Active Sessions:** View a list of current sessions and check details. See [Managing Sessions](#).
- **Collaborative Spaces:** Create separate collaborative spaces in which different groups of users can create their own activities, activity plans, and task plans. Identify which users can access each space and specify their level of access (access role) to that space. See [Managing Collaborative Spaces](#).
- **External Claims:** Identity providers who can send claims along with the user's authentication. These claims can be used to assign group memberships, avoiding the need to explicitly assign users to groups within Foundation Hub. External claims that Foundation Hub will accept are managed here. See [Managing External Claims](#).
- **Groups:** Organize users who require the same permissions into groups and assign the permissions to the groups instead of to the individual users. This approach simplifies security administration and makes security assignments more visible. Some applications are installed with predefined sets of groups with commonly used permissions. These groups are prefaced with the application name. See [Managing Groups](#).
- **Roles:** Define collections of permissions necessary to perform various actions within Foundation Hub. See [Managing Roles](#).
- **Security Events:** View a list of security events such as logins, logouts, and login failures and check details. See [Managing Security Events](#).
- **Users:** Define users by manually entering them in Foundation Hub or by connecting Foundation Hub to an external user directory and synchronizing the users. See [Managing User Directories](#).

## Managing Sessions

You can view active and inactive user sessions from **Admin and Settings > Security > Active Sessions**:

- **Active:** The user has logged in and has been active within a period of time specified by **Session Activity Timeout** in **Settings > Hub Configuration** (30 minutes by default).
- **Inactive:** The user has been inactive for more than the time specified by the **Session Activity Timeout** Hub Configuration setting and will need to provide a password to resume. Note that a user will be completely logged out after the time specified by the **Session Global Timeout** in **Settings > Hub Configuration** (8 hours by default).

You can filter and sort the grid by column names or use the **Filter Active Sessions** tool to find specific sessions. Click a user name to view details.

**Note:** You can set the **Session Activity Timeout** and **Session Global Timeout** Hub Configuration settings in **Settings > Hub Configuration**.



## Managing Collaborative Spaces

Collaborative spaces are work spaces for segregating Foundation Hub content so that you can control who can create, access, and manage that content.

Collaborative spaces enable you to:

- Restrict different groups of users to different content areas.  
For example, you might give external contractors access to only one collaborative space, give internal analysts access to several collaborative spaces, and give lab managers access to all collaborative spaces.
- Control what each user or group can do in their collaborative spaces.  
For example, you might give a user read-only rights to content in one collaborative space, but give that same user authoring rights in another collaborative space.  
Users can change a content entity in a collaborative space in which they have author rights only if they also have appropriate rights for the specific entity.
- Identify the default [lifecycle policy](#) to apply to entities that are created in the collaborative space.
- Identify additional lifecycle policy choices to provide to users when they create activities, activity plans, parameter templates, task plans, and tasks in the space.

**Note:** Sample and material entities always use the default lifecycle policy defined for the space. Equipment does not belong to a collaborative space.

### Default Collaborative Space

Foundation Hub includes a predefined collaborative space called **Default**. Content resides in this collaborative space unless you define additional spaces and users switch to those spaces when they create content.

Content created before release 2019 SP1 (when collaborative spaces were introduced) resides in this collaborative space.

### Collaborative Space Types and Cross-Space Visibility

By default, the entities (activities, activity plans, parameter templates, task plans, and tasks) created in a collaborative space are *unshared*. This means that they are visible only to users who have a role higher than *Public reader*, and only while the users work in the same collaborative space as the content.

Collaborative spaces have a **Type** setting that controls whether content owners can *share* their content with other collaborative spaces (make it accessible from other spaces):

- Private. No users, regardless of role, can share content from a private space.

**Note:** The **Default** space is private and you cannot change its **Type** setting. Any collaborative spaces that you created before the **Type** setting was introduced are also private, but you can change their types.

- Protected. Users with a role of *leader* or higher can share content. Sharing is done one content object (activity, activity plan, parameter template, and task plan) at a time.
- Public. Users with a role of *author* or higher can share content.

For information about sharing an activity or activity plan, see [Making an Activity Visible Across Collaborative Spaces](#) and [Making an Activity Plan Visible Across Collaborative Spaces](#). For information about sharing task plans, see the *Foundation Hub User Guide*.

Activity and activity plan owners can also *move* their content entities, one at a time, from one

collaborative space to another. For more information, see [Moving an Activity to Another Collaborative Space](#) and [Moving an Activity Plan to Another Collaborative Space](#).

### IMPORTANT!

Task plan owners can also move their task plans, but *only* if the task plan content is not also used in another task plan, such as a personal task plan.

Consequently, when you first implement collaborative spaces or create new spaces, you might be unable to move in-progress task plans.

## Permissions Required to Manage Collaborative Spaces

Foundation Hub administrators with the *Foundation/Manage Collaborative Spaces* permission can perform the following tasks:

- Create collaborative spaces.
- Change a collaborative space from *private* to *protected* or to *public*, and from *protected* to *public*.
- Assign collaborative space members and their roles.
- Delete **empty** collaborative spaces; a collaborative space that has content cannot be deleted.

**Note:** Foundation Hub administrators can administer a collaborative space, but cannot view, create, edit, or delete content in a collaborative space unless they have the appropriate content access role for that space.

## Creating and Editing a Collaborative Space

### Creating a Collaborative Space

1. Open **Security > Collaborative Spaces**.
2. Click **Add Collaborative Space**.
3. Enter a **Name**.
4. Select the **Type**, which controls visibility of content from other collaborative spaces. For more information, see [Collaborative Space Types and Cross-Space Visibility](#).
5. Enter a **Description** and click **OK**.
6. If required, submit your e-signature.
7. Proceed to Editing a Collaborative Space.

The space is added to the list of collaborative spaces. You must now edit it to set up its members and basic rules such as which lifecycle policies are available to the entities created in the space.

### Editing a Collaborative Space

1. From Foundation Hub **Admin & Settings**, open **Security > Collaborative Spaces** and select the row of the collaborative space that you need to access.  
If you are currently working in that collaborative space, you can also access it by clicking the **Administer Collaborative Space** (small gear) icon next to the collaborative space name in the Main toolbar.
2. Click **Edit**.
3. If necessary, change the **Type**. For more information, see [Changing the Collaborative Space Type](#).
4. Under **Life Cycle Policies**, ensure that the selected default policy is appropriate and that all policies needed for this collaborative space are identified:

- a. Use the **Assign** function to identify the policies supported in this collaborative space. If you assign more than one policy, users can choose between them when they create new entities.
- b. Select the appropriate default policy to use for new entities.
- c. If you need to prevent the **Default LifeCycle Policy** (or a policy that you mistakenly added) from being applied to new entities, select its name and click its **Unlink** icon:

Life Cycle Policies:

Default LifeCycle Policy



HOV R&D Life Cycle Policy

GMP LCP

**Tip:** To unlink policies, you might need to **Submit** changes that you have already made.

5. If the **Use Latest Activity** option should be selected for tasks created in this collaborative space, select this check box.

**Note:** Users can change this setting at the individual task level for all tasks *except* those based on activities that use the **Lock Activity's Settings in Tasks** option.

6. **Assign** authorized **Users**.
7. **Assign** authorized **Groups**.
8. For each user and group you selected, click the **Access Role** cell and assign the appropriate access role:

Role	Capabilities
Public reader	Access content that has been explicitly shared. Cannot access unshared content, create content, or edit content.
Reader	Public reader rights, plus access unshared content.
Contributor	Currently the same as Reader, but might change in a future release.
Author	Reader rights, plus create task plans. If the space is public or protected, control whether to share your task plans so that they can be accessed from other spaces.
Leader	Author rights, plus create activities and activity plans. If the space is public or protected, control whether to share your activities and activity plans so that they can be accessed from other spaces. Change the owner of a task plan, but only if it specifies a collaborator group and you are a member of that group.
Owner	Leader rights, plus the ability to control who can use the space and their roles within the space.

**Notes:**

- To change an entity in a collaborative space, a user must have rights to that entity, as well as an appropriate role in the space.
- Users get the most powerful access role available to them. For example, a user with reader rights who belongs to a group with author rights gets author rights.

9. Click **Save**.
10. If required, submit your e-signature.

## Changing the Collaborative Space Type

### Notes:

- You can change the **Type** setting for a collaborative space to a less restrictive setting, but not to a more restrictive setting.
- **Keep in mind that if you change a *private* collaborative space to *public* or to *protected*, all of the space's content becomes shareable, and the space can never be changed back to private.** If you change a private space to protected, it can be changed to public, but not back to private.
- Changing a collaborative space type does *not* make any of its activities, activity plans, or task plans shared—it simply makes it possible for the owners of these entities and leaders of the space to share them, which is done one entity at a time, as needed.

### To change the Type setting for a collaborative space:

1. Navigate to **Security > Collaborative Spaces**.
2. Click the row of the collaborative space to open it.
3. Click **Edit**.
4. Select the **Type**.
  - If the space is *private* and you need to allow leaders and authors to share its content, change its type to *public*. To allow only leaders to share its content, change its type to *protected*.
  - If the space is *protected* and you need to allow authors, as well as leaders, to share its content, change its type to *public*.
5. Click **Save**.
6. If required, submit your e-signature.

## Managing External Claims

Foundation Hub can accept security information (external claims) from trusted security providers (for example, Active Directory). This allows Foundation Hub to compute the user's security permissions without the administrator having to explicitly set the user's permissions within Foundation Hub Admin and Settings. External claims can be helpful in reducing the amount of administration required for your Foundation Hub deployment.

### Adding an External Claim

1. Set up a User Directory in **Settings > User Directories**. See [Managing User Directories](#).
2. Navigate to **Security > External Claims**.
3. Click **Add External Claim**.
4. Set the following:
  - **Value:** Common name of the Active Directory group.
  - **User Directory:** Choose the user directory that this claim belongs to.
  - **Groups:** Assign the external claim to one or more groups. See [Managing Groups](#).

### Deleting an External Claim

1. Navigate to **Security > External Claims**.
2. Select the row for the external claim you want to delete, and then click the **Delete External Claim** (minus) icon.

## Example

Active Directory can provide the list of Active Directory groups to which the user is assigned. Assume that your company has a group named “Registration Users” in Active Directory. You could create an external claim of the same name in Foundation and add that claim to the group that grants access to the BIOVIA Chemical Registration application. With this in place, any users logging in that belong to this Active Directory group will be granted permissions in Foundation Hub. If you remove a user from the Active Directory group, that user will lose the permission in Foundation Hub.

## Managing Groups

Groups allow you to manage users to be assigned to roles or external claims. Groups can also be members of other groups, providing flexibility in assignment of roles at various levels.

### Adding a Group

1. Open **Admin and Settings > Security > Groups**.
2. Click the **Add Group** (plus) icon.
3. Define the following:
  - **Name:** Give the Group a unique and descriptive name.
  - **Email Alias:** Email contact for the group.
  - **Description:** Additional information about the group.
  - **External Claims:** Choose an External Claim to assign to the group and click Assign. You can assign more than one to the group. See [Managing External Claims](#).
  - **Member Groups:** Assign one or more existing groups to belong to this group. Users that belong to those groups will belong to this group implicitly.
  - **Member Of:** Assign one or more existing groups to this group to be a member of. Users that belong to this group will belong to those groups implicitly.
  - **Roles:** Assign one or more Roles to this group. See [Managing Roles](#).
  - **Users:** Assign one or more Users to this group. See [Managing Users](#).
  - **User Roles:** Assign one or more User Roles for this group. These govern workflow permissions for recipes in applications such as Compose and Capture.
  - **Collaborative Spaces:** Select the collaborative spaces in which the group is allowed to work and identify the group's access role for each space you select. For details, see [Managing Collaborative Spaces](#).
  - **Permissions:** Assign one or more [Permissions](#) for this group.

**Note:** Use Roles to manage permissions whenever possible instead of assigning permissions directly to groups.


4. Click **OK**.

### Deleting a Group

**Note:** You cannot delete a group that is associated with an application. Most applications delete their groups automatically when they unregister.

1. From the **Admin and Settings** page, click **Groups**.
2. Select the check box for the group you want to delete. You can select multiple groups.
3. Click the **Delete Selected** (minus) icon.
4. In the **Confirm Delete** dialog box, click **Yes**.

### Cloning a Group

1. From the **Admin and Settings** page, click **Groups**.
2. Select the check box for the group that you want to clone.
3. Click the **Clone Selected** icon .
4. In the **Confirm Clone** dialog box, click **Yes**.

A clone of the group is created, and an editor for the new group opens.

By default, the name of the new group is that of the source group, with a numeric suffix to make it unique. For example, if you clone the group Equipment Administrators, the first cloned group is called Equipment Administrators(2), the second Equipment Administrators(3), and so on.

When you clone a group, associations with external claims, member groups, parent groups, roles, users, user roles, and permissions are cloned with it. Note that the associated entities themselves are not cloned.

5. Edit the details of the cloned group.
6. Click **Save**.

### Managing Roles

A Role is defined by a set of permissions that allow users or groups of users to use specific Foundation features to perform their jobs. The available roles are based on applications that are part of your deployment. See [Default Roles](#).

You can assign individual users to Roles, but you may want to manage users better by organizing users into Groups, which you can assign to Roles. See [Managing Groups](#).

### Adding a Custom Role

If one of the predefined Roles does not meet your organization's needs you can create a custom Role and assign [permissions](#) to it as follows:

1. Open **Admin and Settings > Security > Roles**.
2. Click the **Add Role** (plus) icon.
3. Define the following:
  - **Name:** Give the Role a unique and descriptive name.
  - **Description:** Additional information about the role.
  - **Users:** Assign one or more Users to this role. See [Managing Users](#).
  - **Groups:** Assign one or more groups to belong to this role. Users that belong to those groups will belong to this role implicitly. See [Managing Groups](#).
  - **Permissions:** Assign one or more [Permissions](#) to the role. The list of permissions comes from the applications that are part of the deployment. When a new application is added, its permissions are added to the list. You cannot add or remove permissions manually.

## Cloning a Role

1. From the **Admin and Settings** page, click **Roles**.
2. Select the check box for the role that you want to clone.

3. Click the **Clone Selected** icon .

4. In the **Confirm Clone** dialog box, click **Yes**.

A clone of the role is created, and an editor for the new role opens.

By default, the name of the new role is that of the source role, with a numeric suffix to make it unique. For example, if you clone the role Equipment Administrator, the first cloned role is called Equipment Administrator(2), the second Equipment Administrator(3), and so on.

When you clone a role, associations with users, groups, and permissions are cloned with it. Note that the associated entities themselves are not cloned.

5. Edit the details of the cloned role.
6. Click **Save**.

## Default Roles

Foundation Hub provides a standard set of predefined Roles that are commonly found in a lab environment. Each Role is configured with a set of specific permissions that apply to the groups and users assigned to that Role. You can edit the permissions on any of these Roles as necessary. You can also create custom Roles to reflect the specific structure of your organization.

Role	Description
Application Developer	Builds and deploys extensions to the system.
Equipment Administrator	Defines the characteristics of equipment for the system (classes, types, properties, measurements, etc.)
Equipment Manager	Maintains the equipment inventory for the system. Installs instruments, retires instruments, and manages instrument status.
Inventory Administrator	Performs administrative functions, creates and modifies new chemical records, creates new containers/barcodes, updates/disposes existing inventory, and runs reports.
Inventory Receiver	Creates and modifies new chemical records, creates new containers/barcodes, updates/disposes existing inventory, and runs reports.
Lab Manager	Reviews specifications, procedures, and sample plans as well as the work done in Certificate of Analysis workflows and approves work performed in the lab.
Lab Scientist	Creates and uses content throughout the system.
Metrologist	Uses system capabilities to maintain equipment so that it can be used in lab activities. Includes verifications, calibrations, cleanings, and preventive maintenance.

Role	Description
Procedure Administrator	Exports and imports libraries, configures procedures (Compose recipes) for the system.
Procedure Library Manager	Builds libraries of operations, stages, and procedures for the system. Maintains vocabulary and parameter content.
Reference Data Administrator	Defines the reference data configuration for the system. Defines vocabularies, relationships between vocabularies, links to external reference data systems.
Reference Data Manager	Manages reference data content for the system (content of vocabularies and other reference data).
Registration Administrator	Administers registration-specific settings including the ability to work with retired data.
Registration Curator	Reviews submitted registrations, accepts or rejects registrations, and retires registrations.
Registration Moderator	Maintains registration data submitted by others.
Security Manager	Manages users, groups, and security assignments for the system.
Specification Author	Defines specifications for materials for use in Certificate of Analysis workflows.
System Administrator	Performs overall system configuration including database connections, server topologies, etc.

## Managing Permissions

Foundation Hub provides a set of permissions to control which functions of BIOVIA Foundation Hub and the integrated BIOVIA applications are accessible by users and system administrators.

Instead of assigning permissions to individual users, you assign them to roles and to groups. Roles identify users who perform the same job function, and groups identify users in related organizations. Each user inherits the permissions of each role and group to which they belong.

You configure Roles, Groups, and Permissions using **Admin and Settings** pages in Foundation Hub.

The following topics describe the set of permissions provided for controlling access to functions in each of the BIOVIA applications.

- [Foundation Hub](#)
- [Pipeline Pilot](#)
- [Compose and classic Capture](#)
- [Insight](#)
- [Experiment](#)
- [Request](#)
- [Workbook](#)



## Foundation Hub Permissions

The following permissions apply to the functionality in BIOVIA Foundation Hub.

Permission	Description
Foundation/Administration/Logon	<ul style="list-style-type: none"> <li>Can log on to Foundation Hub and can access the <b>Admin and Settings</b> page. This permission is expected to be used in conjunction with additional Foundation Hub permissions in order to grant access to specific administration functions. If this permission is used alone, the Admin and Settings page is empty with no access to the security, resources, and system settings.</li> <li>For example, a user who is only managing license files needs: <ul style="list-style-type: none"> <li>Foundation/Administration/Logon</li> <li>Foundation/Manage Licenses</li> </ul> </li> </ul>
Foundation/Logon	Can log on to Foundation Hub. The <b>Admin and Settings</b> page is not accessible.
Foundation/Manage Activities	Can create and edit activity related resources.
Foundation/Manage Any Object	Can view and manage any data object in Foundation Hub.
Foundation/Manage Any Taskplan	<p>Can edit the <b>Collaborator Group</b> information panel field (but not Owner) of <i>any</i> task plan in <i>any</i> collaborative space.</p> <p><b>Note:</b> Grant this permission sparingly. Its intended use is to reassign responsibility for stranded task plans. A task plan can become stranded if it has no collaborator group and its owner becomes unavailable. For more information, see "Transferring Task Plan Responsibilities" in the <i>Foundation Hub User Guide</i> or user help.</p>
Foundation/Manage Collaborative Spaces	Can create and delete collaborative spaces and control which users can access them and with which roles.
Foundation/Manage Equipment	Can change data packets and can add, remove, and update equipment classes, equipment types, and equipment instances (devices).
Foundation/Manage Master Data	<p>Can view and manage the locations, organizations, protocols, projects, sequence templates, vocabularies, and parameter templates that are configured in Foundation Hub <b>Admin &amp; Settings</b> pages.</p> <p>Can also create and manage public views and filters for Foundation pages that list tasks, equipment, samples, materials, and so on.</p>

Permission	Description
Foundation/Manage Security	Can view and manage: <ul style="list-style-type: none"> <li>■ Active sessions</li> <li>■ External claims</li> <li>■ Roles</li> <li>■ Groups</li> <li>■ Security events</li> <li>■ Trusted certificates</li> <li>■ Users</li> </ul>
Foundation/Manage Settings	Can view and manage: <ul style="list-style-type: none"> <li>■ Applications</li> <li>■ Extended Properties</li> <li>■ Hub Configuration</li> <li>■ License Files</li> <li>■ Life Cycle Policies</li> <li>■ Links</li> <li>■ Signature Policies</li> <li>■ User Directories</li> </ul>
Foundation/Manage System	Can manage configuration related to the system setup.
Foundation/Manage Units	Can create and edit units.
Foundation/Perform Metrology	Can perform equipment verification, calibration, cleaning, and preventative maintenance.
Foundation/View Activities	Can view activity related resources.
Foundation/View All Notifications	Can view all notifications.
Foundation/View Any Object	Can view any object in Foundation Hub.
Foundation/View Equipment	Can view equipment related resources.
Foundation/View Master Data	Can view all master data in Foundation Hub.
Foundation/View Metrology	Can view equipment.
Foundation/View Security	Can view: <ul style="list-style-type: none"> <li>■ Active Sessions</li> <li>■ External Claims</li> <li>■ Roles</li> <li>■ Groups</li> <li>■ Security Events</li> <li>■ Trusted Certificates</li> </ul>

Permission	Description
	<ul style="list-style-type: none"> <li>■ Users</li> </ul>
Foundation/View Settings	Can view: <ul style="list-style-type: none"> <li>■ Applications</li> <li>■ Extended Properties</li> <li>■ Hub Configuration</li> <li>■ License Files</li> <li>■ Life Cycle Policies</li> <li>■ Links</li> <li>■ Signature Policies</li> <li>■ User Directories</li> </ul>
Foundation/View System	Can view configuration related to the system setup.
Foundation/View Units	Can view units.

## Pipeline Pilot Permissions

The following permissions apply to the functionality in BIOVIA Pipeline Pilot.

Permission	Description
Platform/Administration/CreatePackage	Can use the package web service to create a new package.
Platform/Administration/Logon	Can log on to Pipeline Pilot server as administrator.
Platform/Catalog/DeveloperLogon	Can access Solr Admin UI at /appcatalog/admin.
Platform/Catalog/SeeAllIndex	Can see all protocols returned from Admin Search, regardless of owner. Queries must also contain 'globalsearch=true' to be shown.
Platform/Catalog/UpdateIndex	Can insert/remove Solr documents with /appcatalog/update.
Platform/Logon	Can log on to the Pipeline Pilot Server.
Platform/PipelinePilot/Administer	Can administer the Pipeline Pilot application from the Professional Client.
Platform/PipelinePilot/Logon	Can log using the Pipeline Pilot client.
Platform/RunProtocol	Can run Pipeline Pilot protocols from the RunProtocol command line.
Platform/ScheduleJobs	Can schedule jobs.
Platform/WebPort/Logon	Can log on to Web Port.

## Compose and Classic Capture Permissions

The following permissions apply to the functionality in Compose and in classic Capture. For Capture Hub, a user's access to a recipe execution is determined by their access to the task and their role within the collaborative space of the task plan. For more information, see *Working in Capture Hub* in the *BIOVIA Capture Hub User Guide* or the *BIOVIA Foundation Hub User Guide*.

Permission	Description
Capture/Application/Logon	Can log on to the classic Capture application.
Compose/Administration/Import and Export	Can import and export Compose data files.
Compose/Administration/Logon	Can log on to Compose as an administrator.
Compose/Administration/Manage Parameters	Can manage Compose recipe parameters.
Compose/Administration/Manage Parameter Templates	Can view and manage Parameter Templates in Foundation Hub.
Compose/Administration/Manage Process Library	Can manage the Compose Process Library.
Compose/Administration/Manage Recipe Library	Can manage the Compose Recipe Library.
Compose/Administration/Manage Reports	Can manage Compose reports.
Compose/Administration/Manage Workflows	Can create and manage Compose recipes.
Compose/API/Export Data	Can import/export Compose data files through the Compose API.
Compose/API/Import Data	Can import Compose data files through the Compose API.
Compose/API/Manage Equipment	Can manage Compose recipe equipment through the Compose API.
Compose/API/Manage Extended Properties	Can manage extended properties of Compose data objects through the Compose API.
Compose/API/Manage Libraries	Can manage Compose libraries through the Compose API.
Compose/API/Manage Materials	Can manage Compose recipe materials through the Compose API.
Compose/API/Manage Parameters	Can manage Compose recipe parameters through the Compose API.
Compose/API/Manage Parameter Templates	Can view and manage Parameter Templates in Foundation Hub through the Compose API.
Compose/API/Manage Process	Can manage Compose recipe process elements through the

Permission	Description
Elements	Compose API.
Compose/API/Manage Recipes Assignments	Can manage Compose recipe assignments through the Compose API.
Compose/API/Manage Recipes	Can manage Compose recipes through the Compose API.
Compose/API/Manage Reports	Can manage Compose reports through the Compose API.
Compose/API/Manage Values	Can manage planned and actual values in a Compose recipe through the Compose API.
Compose/Application/Logon	Can log on to the Compose application as a non-administrative user.
Design/Application/Logon	Users can manage and execute external recipes in Compose and classic Capture. An external recipe is one that is created in a different BIOVIA application, such as Workbook, Notebook, or Experiment.

## Experiment Permissions

The following permissions apply to the functionality in BIOVIA Experiment.

Permission	Description
Experiment/Administer	Can log in to Experiment and perform administration functions. This permission is included when the Experiment application is installed.
EKB/Administrator	Can log in to Experiment and perform administration functions. This permission is included when the EKB package is installed. It is the equivalent to the Experiment/Administer permission.
EKB/Package Developer	This is a Service permission that is not assigned to end-users.

## Insight Permissions

The following permissions apply to the functionality in BIOVIA Insight.

Permission	Description
Insight/Users	Can log into the Insight web application. By default, all users belong to this group.
Insight/Administrators	Can access to certain Insight administration functions. By default, Foundation Hub administrators belong to this group.
Insight/PluginAuthors	Can use the Pipeline Pilot client to create plugins for Insight.

Permission	Description
Insight/ProjectLeaders	Can share project data with others. By default, all users belong to this group.
Insight/ContentCreators	Can create new projects or save data into other user's projects. By default, all users belong to this group.
Insight/PackageDevelopers	Can use Pipeline Pilot client to save protocols directly into the package.

**Note:** All the Insight groups and permissions are available whether you are using Insight or BIOVIA Search without the Analyze Visualize application.

## Request Permissions

The following permissions apply to the functionality in BIOVIA Request (not the Request functionality in the Foundation Hub Task Planner).

Permission	Description
WorkRequest/AcceptRequest	Can accept a completed request.
WorkRequest/AcceptTask	Can accept tasks, provides access to MyTasks page.
WorkRequest/Admin	Administrative user, provides access to the admin pages.
WorkRequest/CreateRequest	Can create new requests.
WorkRequest/Decline	Can decline a task.
WorkRequest/DoNotAcceptRequest	Can refuse to accept a completed request.
WorkRequest/ManageRequestQueue	Provides access to the manage work page for assigning tasks.
WorkRequest/PrintSampleBarcode	Can print sample barcodes.
WorkRequest/PrintTaskBarcode	Can print task barcodes.
WorkRequest/Review	Can review task, provides access to the review page.
WorkRequest/Schedule Task	Can schedule tasks.
WorkRequest/User	Has base permission to use Request system.

## Workbook Permissions

The following permissions apply to the functionality in BIOVIA Workbook.

Permission	Description
Vault/AcceptRequest	Can accept requests in BIOVIA Request.
Vault/AcceptTask	Can accept tasks in BIOVIA Request.
Vault/Administrator	Can log on to the Vault Administration Console and perform Administrator functions.
Vault/CreateRequest	Can create requests in BIOVIA Request.
Vault/Decline	Can decline requests in BIOVIA Request.
Vault/DoNotAcceptRequest	Can mark Requests as invalid (Request requestors need this permission)
Vault/DefaultScriptHandlerService	This is a Service permission that is not assigned to end-users.
Vault/External Data Conversion Service	This is a Service permission that is not assigned to end-users.
Vault/Folder.Administrator	Can create and rename Workbook Folders.
Vault/Forms.Editor	Can create and edit Workbook Forms.
Vault/GroupProfile.Administrator	Can manage Groups.
Vault/Logon	Can log on to the Workbook application.
Vault/ManageRequestQueue	Can assign tasks, etc. as a Request Lab Manager.
Vault/MaterialInfoManager	This is a Service permission that is not assigned to end-users.
Vault/NotebookConfiguration	This is a Service permission that is not assigned to end-users.
Vault/NotebookExplorerContextualViewerConfiguration	This is a Service permission that is not assigned to end-users.
Vault/Operation.Editor	Can edit operations for Workbook Recipes section This is an obsolete permission used only for Workbook versions 6.4 and earlier.
Vault/PrintSampleBarcode	Can print barcodes for samples.
Vault/PrintTaskBarcode	Can create Request Task barcodes.
Vault/PropertySetEditor	Can create and edit Property Sets (admin Permission).

Permission	Description
Vault/Recipe.Editor	Can edit recipes for Workbook Recipes section. This is an obsolete permission that was previously used for Workbook versions 6.4 and earlier.
Vault/RecipeUser	Can use recipes in Workbook Recipes section. This is an obsolete permission that was previously used for Workbook versions 6.4 and earlier.
Vault/Report.Editor	Can create and edit fixed Workbook Reports.
Vault/Review	Can review Request tasks.
Vault/RunAnalysis	Can use Pipeline Pilot protocols through Workbook.
Vault/RunProtocol	This is a Service permission that is not assigned to end-users.
Vault/ScheduleTask	Can schedule tasks in Request (this is a Lab Manager permission).
Vault/ScriptDeveloper	
Vault/SearchExtensions	This is a Service permission that is not assigned to end-users.
Vault/SectionTemplate.Editor	Can create and edit Workbook Section Templates.
Vault/Symyx Laboperation Workrequest Reporting	This is a Service permission that is not assigned to end-users.
Vault/Symyx LabOperations WorkRequest	This is a Service permission that is not assigned to end-users.
Vault/SymyxRegistration	This is a Service permission that is not assigned to end-users.
Vault/Template.Editor	Can create and edit Workbook Templates.
Vault/TemplateManagementTools	
Vault/TransferTemplates	Can import/export Templates from/to Workbook environments.
Vault/UndoCheckout	Can undo checked out of a Workbook document.
Vault/Widget.Administrator	Can create, edit and publish Workbook Homepage widgets.
Vault/Workflow.Administrator	Can create, edit and publish Vault Document Workflows (Workbook).



## Managing Security Events

You can view security events including successful and failed logins, logouts, and session expirations from **Admin and Settings > Security > Security Events**. From there, you can filter and sort the **Security Events** table by the column names, or use the **Filter Security Events** tool.

### Tips:

- You can view the list of active and inactive sessions in **Admin and Settings > Security > Active Sessions**. See [Managing Sessions](#).
- If the event involved a valid user, the **Performed By** column will contain a link to view the details of that user account.

## Managing Trusted Certificates

You can establish trusted relationships between Foundation Hub and other applications and services by including SSL certificates for them in the Foundation Hub trust store.

**Note:** Trusted certificates are used by Foundation Hub to enable 3DPassport security.

## Adding a Certificate

1. Open **Admin and Settings > Security > Trusted Certificates**.
2. Click the **Add Trust Store Certificate** (plus) icon.
3. Add the certificate:
  - Paste in the certificate text.
  - Paste a URL that points to the SSL Certificate on the server.
4. Click OK.

**Note:** You do not need to restart the Foundation Hub Service when making changes to the trust store.

## Troubleshooting

Check the Foundation Hub log file for details about failed trust relationships. If Foundation Hub fails to make outbound requests to other connected services or applications it could be the result of a certificate error. Check the log file for errors to determine if there are certificate problems:

1. Open **Admin and Settings > Settings > Logging**.
2. Click **Download Current Log File**.
3. Look for the following:
  - `Error | 2017-07-19 14:46:47,958 [main] ERROR ssl.HubSSLService - Error while establishing connection to: [https://www.yahoo.com] Message: sun.security.validator.ValidatorException: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target`

- javax.net.ssl.SSLHandshakeException:

```
Error | 2017-08-02 20:51:41,685 [I/O dispatcher 4] ERROR
hub.PushNotificationWorkerCallback - Dispatch failed: DELETE
https://servername:8443/foundation/hub/logout/notify/93fa63d3-27a2-
4e41-ab13-6348f1311be7 HTTP/1.1 ->
javax.net.ssl.SSLHandshakeException: General SSLEngine problem :
com.accelrys.platform.core.Message :
1480949b-90ad-4fe0-9d6d-fe7888286c1f
```

## Setting Up User Authentication

You can configure user authentication using LDAP, SPNEGO/Kerberos, or 3DPassport.


- For SPNEGO/Kerberos authentication, see [Configuring SPNEGO/Kerberos Authentication](#).
- For LDAP authentication, configure one or more User Directories. See [Adding LDAP User Directories](#).
- 3DPassport is Dassault Systèmes' user authentication server. To set up 3DPassport authentication, see [Setting Up User Authentication with 3DPassport](#).

**IMPORTANT!** After setting up user authentication, change the password for the scitegicadmin user.

**Note:** Foundation Hub does not enforce unique usernames for multiple domains. If a user is not able to sign in, he or she may be using a username that is duplicated in a different domain. Ask the user to sign in while specifying the domain name. For example <domain>\username.

## Adding LDAP User Directories

**Note:** Lightweight active directories where the service is not connected directly to the domain controller such as Active Directory Application Mode (ADAM)/Active Directory Lightweight Directory Services (AD LDS) are not supported.

1. Navigate to the Foundation Hub Landing Page, for example:  
`https://<hubserver>:9953/foundation/hub`
2. Click .
3. Open **Settings > User Directories**.
4. Click the **Add User Directory** (plus) icon.
5. Complete the **User Directory** information:
  - **Name:** Add the name of the user directory.
  - **User Directory is active:** Choose whether this directory is being used currently for user authentication.
  - **Authentication Order:** If there are multiple user directories defined, this field determines the order used to evaluate credentials when no domain is provided during sign in. Lower values take higher precedence.

The servers can be unrelated or they can be different sub-domains of a root domain. They should not be a list of replicated LDAP servers that have the same user base. When a user directory authenticates a user, the user entry in Foundation Hub is distinct for that user, so in the case where the same user name exists in multiple directories, they are treated as distinct users in Foundation Hub.

- **Update User Account Data on Sign In:** If there have been changes to the user's account data in the User Directory (first name, last name, full name, email address, etc.), update the user's account when they sign in. See [Authentication Settings Matrix](#) for details about how authentication settings affect sign in for different scenarios.

**Note:** This setting is affected by **Admin and Settings > Applications > Application Settings > Foundation Hub > Security > Update User Account Data on Sign In**.

6. Complete the **Server Configuration** information:

- **Server:** Add the full server name for the LDAP server. Use **Find LDAP Server** to find an LDAP server on the network. The address should begin with `ldap://` or `ldaps://`. We recommend that you use the LDAPS URL to secure communication between Foundation Hub and the LDAP server.
- **Search Base:** Search base from which to authenticate users. This must be a base distinguished name. See [Performance Considerations](#).
- **Principal and Principal Password:** Include the name of the principal account that will be used to look up users. This must be a distinguished name for a user that has rights to authenticate with and read from the user directory.
- **User Directory Type:** ActiveDirectory by default. GenericLDAP is also supported.
- **User Object Classes:** Comma separated list of attribute values of LDAP objectClass field. This is primarily used to determine if the object is a user (for example, user, person).
- **Follow Referrals:** See [Performance Considerations](#).
- **Kerberos Integration:** Choose this if you are using SPNEGO or Kerberos. See [Configuring SPNEGO/Kerberos Authentication](#).

7. Check the **Attribute Mappings** information. These map directory attribute data to Foundation Hub attributes. The default values are based on the **User Directory Type** that you have chosen for the user directory:

- Username
- Email
- External Id
- Full Name
- First Name
- Last Name
- Group Attribute

**Tip:** The settings in the **Attribute Mappings** have a significant impact on the day-to-day use of BIOVIA applications. Use an LDAP tool like Softera or JXplorer to inspect user attributes in your directory before completing this section.

8. Click **Test Connection**. If the connection test is successful, click OK.
9. Configure **Permitted Groups** and **Custom Query Filter** if desired. See [Permitted Groups and Custom Query Filter](#).
10. If you are configuring a load-balanced environment, restart the Foundation Hub server by navigating to **Settings > Hub Configuration** and clicking **Restart Server**.
11. Set up a synchronization schedule and synchronize the user data for the user directory. See [Synchronizing LDAP Users](#).

### Permitted Groups and Custom Query Filter

The Permitted Groups and Custom Query Filter fields are only available by opening and editing a user directory that you have created.

1. Open **Admin and Settings > Settings > User Directories**.
2. Click the user directory you just created to open it.
3. Click **Edit** and edit the following fields as desired:
  - **Permitted Groups:** List of fully qualified LDAP groups. The Field will search for entered group names within the previously defined Search Base. Users that are a member of ANY of the defined groups will be part of the user synchronization. The filter will be constructed based on the defined value of the Group Attribute in the User Directory. For example:  
`CN=Admins,OU=Groups,OU=OrganizationalUnit1,DC=example,DC=org`  
`CN=PowerUsers,OU=Groups,OU=OrganizationalUnit1,DC=example,DC=org`  
(Group Attribute = memberOf)
  - **Custom Query Filter:** Additional query that allows you to define a filter without predefined attribute mapping. For example: `(&(!objectClass=computer)(employeeID=*))`.
4. Click **Save**.

### Example Filter

You can use the **User Object Classes**, **Permitted Groups**, and **Custom Query** filter fields to filter the results when synchronizing users. In these instructions, the following were provided as examples:

- **User Object Classes:** user , person
- **Permitted Groups:**  
`CN=Admins,OU=Groups,OU=OrganizationalUnit1,DC=example,DC=org`  
`CN=PowerUsers,OU=Groups,OU=OrganizationalUnit1,DC=example,DC=org`  
(Group Attribute = memberOf)
- **Custom Query Filter:** `(&(!objectClass=computer)(employeeID=*))`.

These settings result in the following filter:

```
(&(sAMAccountName=*)(&(objectClass=user)(objectClass=person))
(|(memberOf=CN=Admins,OU=Groups,OU=OrganizationalUnit1,DC=example,DC=org)
(memberOf=CN=PowerUsers,OU=Groups,OU=OrganizationalUnit1,DC=example,DC=org))
(&(!objectClass=computer)
(employeeID=*))
```

### Performance Considerations

How you choose to configure your User Directories could have an impact on the performance of authentication requests and user synchronization. Consider the following when setting up user directories to improve performance:

- Restrict your search base to the deepest point in the directory tree that contains all of the users you want instead of targeting it at the root. Depending on where the majority of your users reside in the directory, you may want to consider setting up more than one user directory, each targeted at a different search base beneath your root. When a user authenticates, the directory must be searched in order to lookup the user. Narrowing the search bases can drastically reduce the amount of time it takes to find the user in the directory. A single directory targeted at a search base potentially containing thousands more users increases the time it takes to find the user.

- Set **User Object Classes** to filter out Groups, Computers, or other arbitrary non-user entries (for example, user, person).
- Avoid using **Follow Referrals**. When resolving referrals, transparent connections are created to other servers. Each of the referred servers is searched by root.

## Synchronizing LDAP Users

You must create the user directory and then edit it to set up a user synchronization schedule.

**Note:** **Applications > Application Settings > Foundation Hub > Allow Automatic User Provisioning** must be enabled in order for the synchronization to be able to add new user accounts.

### Setting the Synchronization Schedule

1. Open **Admin and Settings > Settings > User Directories**.
2. Click the user directory you just created to open it.
3. Click **Edit**.
4. Choose whether to **Enable User Synchronization on Schedule**.
5. Set the **Schedule**:
  - **Days of the Week**: Choose which days of the week to synchronize.
  - **Hours**: Specify which hours of the day to synchronize. The time should be whole numbers based on a 24-hour clock. Enter a comma-delimited list to specify multiple hours. For example, 2, 14 would schedule a synchronization for 2 AM and 2 PM.
  - **Minutes Past Hour**: Specify the minutes past the hour you want to run the synchronization. Keep this set to 0 to schedule it to run at the beginning of the hours specified. Use a comma-delimited list to specify multiple runs per hour.
6. Choose **Delete Missing Users** to delete users that are in the Foundation Hub database, but no longer in the external user directory.
7. Set the **Page Size** to the batch size of users to synchronize per request. See your user directory help to view the request limit.
8. Click **Save**.
9. Manually synchronize the users for the first time.

### Manually Synchronizing the Users

Once you have set the synchronization schedule, you can manually synchronize for the first time and any time thereafter as needed. Note that the **Synchronize Users Now** button is not exposed until you have set the synchronization schedule.

1. Open **Admin and Settings > Settings > User Directories**.
2. Click the user directory you just created to open it.
3. In User Synchronization, click **Synchronize Users Now**.

## Configuring SPNEGO/Kerberos Authentication

If you are planning to authenticate with SPNEGO/Kerberos, review these instructions before configuring Foundation Hub.

### Support

- SPNEGO/Kerberos is supported for Foundation Hub installed on Windows.
- SPNEGO/Kerberos is only supported for Windows Active Directory.

### Prerequisites

You will need to gather the following if you need to set up SPNEGO/Kerberos authentication for Foundation Hub:

1. Obtain a Kerberos key tab file and copy it to the local disk of the Foundation Hub server.
2. Obtain the name of the Service Principal which includes the service type prefix (for example, HTTP/) and the server's fully qualified domain name (for example, HTTP/<server FQDN>). Consider setting up your own Active Directory environment to work with Hub Kerberos Authentication.
3. Obtain the name of the Kerberos realm. This is usually the server domain in upper-case.

### Setup

1. Configure the Foundation Hub and choose to **Enable Kerberos Authentication**. You will need to provide **Service Principal**, **Realm Name**, and **Key Tab Location**.
2. Set up one or more User Directories and enable Kerberos Integration. See [Adding LDAP User Directories](#).
3. When finished, if you are configuring a load-balanced environment, restart the Foundation Hub server by navigating to **Settings > Hub Configuration** and clicking **Restart Server**.
4. Configure the browsers. See [Configuring Browsers for SPNEGO/Kerberos](#).

### Configuring Browsers for SPNEGO/Kerberos

#### Firefox

1. Type `about:config` in the address field.
2. In filter/search, type `negotiate`.
3. Set `network.negotiate-auth.trusted-uris` to the full server address. Use https for all communication (recommended).

#### Internet Explorer

Open **Tools > Internet Options > Security tab** and add the Foundation Hub server to the list in **Local intranet**.

#### Chrome

- **Windows:** Open **Tools > Internet Options > Security tab** and add the Foundation Hub server to the **Local intranet** list.

- **Linux:**

- Use the following command-line parameters:

```
--auth-server-whitelist="*.example.com"  
--auth-negotiate-delegate-whitelist="*.example.com"
```

- Use https for all communication (recommended).
  - Type the following to check the policy: `//policy/`

#### Using Chrome Policy Files (Linux Only)

You can configure Chrome on Linux to read JSON format policy files from the following directory: `/etc/opt/chrome/policies/managed`

```
{  
  "AuthServerWhitelist" : "/*.example.org",  
  "AuthNegotiateDelegateWhitelist" : "/*.example.org",  
}
```

```
"DisableAuthNegotiateCnameLookup" : true,
"EnableAuthNegotiatePort" : true
}
```

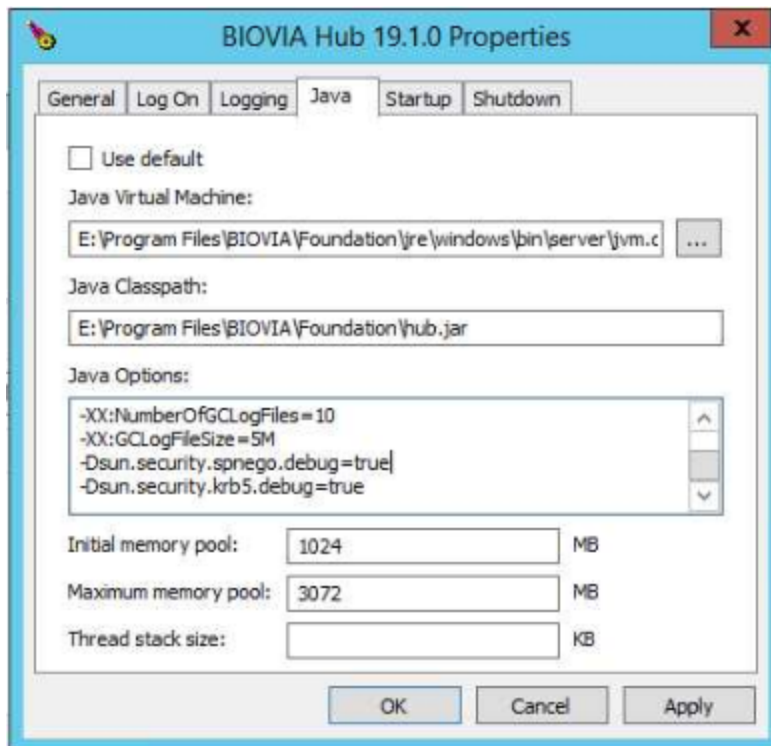
## Troubleshooting SPNEGO/Kerberos Setup

You can start investigating issues you are having by turning on the JVM debug options.

### Set the Java Virtual Machine Debug Options

1. Start the service manager: <hub\_install>\bin\manage.bat
2. In the **Java** tab, set the following **Java Options**:

```
-Dsun.security.spnego.debug=true
-Dsun.security.krb5.debug=true
```



3. Restart the Foundation Hub server:
  - a. Open **Settings > Hub Configuration**.
  - b. Click **Restart Server**.
4. Review <hub\_install>\logs\stdout.

Error Messages:

### LDAP User Search Failure

If you notice LDAP failures in the log file, check your User Directory configuration. You may not be in the same user directory's domain scope.

### Check the KeyTab File

The following Java command shows the keytab file entry.

**Note:** The Principal must exactly match with your Principal Name + "@" + Realm Name of your Foundation Hub configuration (they are case-sensitive).

```
% <Java home>\bin\ktab -k <keytab path> -l -e -t

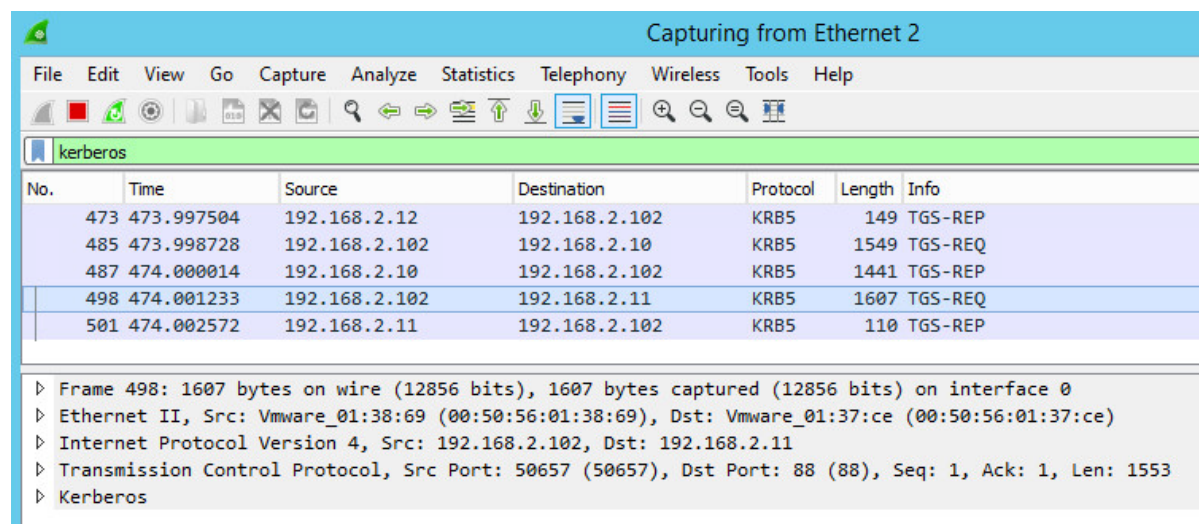
Keytab name: <keytab path>
KVNO Timestamp Principal
-----
12 12/31/69 4:00 PM HTTP/server1.west.biovia.com@WEST.BIOVIA.COM (1:DES CBC mode with CRC-32)
12 12/31/69 4:00 PM HTTP/server1.west.biovia.com@WEST.BIOVIA.COM (3:DES CBC mode with MD5)
12 12/31/69 4:00 PM HTTP/server1.west.biovia.com@WEST.BIOVIA.COM (23:RC4 with HMAC)
12 12/31/69 4:00 PM HTTP/server1.west.biovia.com@WEST.BIOVIA.COM (18:AES256 CTS mode with HMAC
SHA1-96)
12 12/31/69 4:00 PM HTTP/server1.west.biovia.com@WEST.BIOVIA.COM (17:AES128 CTS mode with HMAC
SHA1-96)
```

### Browser URL Mismatch

The hostname part of your Service Principal must exactly match your Browser's URL.

### Sniff Network Packets

Use a network sniffer such as Wireshark on the client machine that you use for your browser to diagnose issues with packets.



### Strange Hub Service Account Name

This error can occur if there is a mismatch between the Domain userPrincipalName and the user logon name for the Foundation Hub Service. Using the SetSPN command can overwrite the Domain userPrincipalName, creating the mismatch. Ensure that the user login name is correct.

**To ensure that the user logon name is correct:**

1. From the Windows Services console, right-click BIOVIA Hub Server <version> and click **Properties**.
2. Under the **Account** tab, ensure that the **User logon name** matches with the Domain **userPrincipalName**. If there is a mismatch, request your IT department to fix this attribute.

## Setting Up User Authentication with 3DPassport

3DPassport is Dassault Systèmes' user authentication server for the 3DS Platform. Setting up Foundation Hub for 3DPassport Authentication delegates the authentication to a 3DPassport server.



Roles, Groups, and Permissions for BIOVIA applications are still managed in Foundation Hub. Local Foundation Hub users (for example, scitegicadmin) cannot authenticate with 3DPassport. It is therefore recommended to assign the Hub System Administrator and Security Manager Role to a user account able to sign in using 3DPassport. During the sign in, 3DPassport and Foundation Hub exchange user information and session data which requires a SSL trust relation between the servers both ways.

## Configuring Foundation Hub to use 3DPassport Authentication

Set Foundation Hub to authenticate users using 3DPassport from the Hub Configuration Page. If you did not set Foundation Hub to use 3DPassport during the configuration steps, do the following:

1. Navigate to **Admin and Settings > Settings > Hub Configuration**.
2. Click **Edit**.
3. In the **Authentication Provider** fields, set **Authentication** to **Passport**.
4. Set **Server Url** to point to the 3DPassport server.
5. Choose whether to use the **Primary Authenticator** option:
  - If you choose the **Primary Authenticator** option, users are directed to the 3DPassport sign in dialog box.
  - If you leave **Primary Authenticator** unselected, users are directed to a sign in dialog box with an option to authenticate using Foundation Hub or 3DPassport.
6. Click **Save** and then **Restart Server**.

## Configuring a 3DPassport user with Foundation Hub Admin Privileges

1. Check if you can sign in to Foundation Hub using 3DPassport user credentials. Navigate to:
 

`https://<hub server>:9943/foundation/hub`
2. Sign in with your 3DPassport account.
  - If login fails, auto-provisioning is turned off. You must create the user manually in Foundation Hub as described later.
  - If login is successful, you are redirected back to the BIOVIA Landing Page. Click the **Applications** icon and choose **Admin and Settings** to check if your account has admin privileges (existing domain users). If you see the error "This account is not authorized...", you will need to configure the 3DPassport credentials with admin privileges.
3. To create the user or assign admin privileges:
  - a. Use the following URL for local Foundation Hub authentication and sign in using the default scitegicadmin/scitegic credentials.
 

`https://<hub server>:9953/foundation/hub/security/auth?local=true`
  - b. If the 3DPassport user does not exist, create it in Foundation Hub from **Admin and Settings > Security > Users**.
  - c. Navigate to **Admin and Settings > Security > Roles** and add your 3DPassport user to the **Hub System Administrator** and **Security Manager** roles.
  - d. Log out and then sign in to Foundation Hub normally using your 3DPassport credentials that you added to the Hub System Administrator and Security Manager roles. Check that you can now access the Admin and Settings area.

## SSL Certificate for 3DPassport

The 3DPassport server and the Foundation Hub server must have a trust relationship set up between them. Include the SSL certificate for the Foundation Hub server in the 3DPassport server's Truststore,

and include the SSL certificate for the 3DPassport server in the Foundation Hub server's Truststore.

- **Foundation Hub truststore:** From Foundation Hub **Admin and Settings**, set **Settings > Applications > Application Settings > Foundation Hub > SSL Truststore Policy** to **Auto import certificates during setup** to automatically import the 3DPassport server certificate to the Foundation Hub truststore.
- **3DPassport truststore:** See the 3DPassport server documentation.

## Authentication Settings Matrix

The following table describes the expected results of a sign in attempt based on these authentication settings and whether the directory administrator has moved the user to a different domain. Note that these results are the same regardless of whether the user signs in with just their username or signs in to a specific domain in the format DOMAIN/Username.

- **Allow Automatic User Provisioning:** Determines whether authenticated users are automatically added from the IP. This option is available in **Admin and Settings > Settings > Applications > Application Settings > Foundation Hub**.
- **Update User Account Data on Sign In:** Determines whether users are updated at sign in with the user information provided by the authentication provider (LDAP or 3D Passport). This option is available in **Admin and Settings > Settings > Applications > Application Settings > Foundation Hub**.
- **New Domain in Same User Directory:** Determines whether the user directory administrator has moved the user to a different domain within that directory.

Allow Automatic User Provisioning	Update User Account Data on Sign In	New Domain in Same User Directory	Result	Details
Off	Off	Yes	Sign in fails	Authorization succeeds but the domain name in the user record cannot be matched or updated.
Off	On	Yes	Sign in succeeds	Authorization succeeds and the user record is updated with the new domain value.
Off	Off	No	Sign in fails	Authorization succeeds but cannot match the user record to the domain.
Off	On	No	Sign in fails	Authorization succeeds but cannot update the account to the new user directory.
On	Off	Yes	Sign in fails	Authorization succeeds but cannot match or update the domain name in the user record.
On	On	Yes	Sign in succeeds	Authorization succeeds and user record is updated with the new

Allow Automatic User Provisioning	Update User Account Data on Sign In	New Domain in Same User Directory	Result	Details
				domain value.
On	Off	No	A new account is created	Auto-provision creates a new account because the username does not match an existing user record in the new User Directory in Foundation Hub.
On	On	No	A new account is created	Auto-provision creates a new account is because the username does not match an existing user record in the new user directory in Foundation Hub.

## Managing Users

Foundation Hub allows you to manage users for BIOVIA applications by synchronizing with one or more external directories. You can also manage individual users directly in the Foundation Hub **Admin and Settings** area.

- **Managing Users Using an External Directory:** Use the **Admin and Settings > Settings > User Directories** to use one or more external user directories to manage user authentication. See [Managing User Directories](#).
- **Adding Users to Groups:** After creating a user, you can add them to groups to manage permissions. See [Managing Groups](#).
- **Managing Users in Foundation Hub:** Use the **Admin and Settings > Security > Users** to manage the user accounts in your deployment. See [Managing Users Directly in Foundation Hub](#).

### Tips:

- You can view the list of active and inactive sessions in **Admin and Settings > Security > Active Sessions**. See [Managing Sessions](#).
- You can view security events including successful and failed logins, logouts and session expirations from **Admin and Settings > Security > Security Events**. See [Managing Security Events](#).

## Managing Users Directly in Foundation Hub

A user record has basic metadata about the user and credential information that will be used when signing on.

### Adding a New User

1. Open **Admin and Settings > Security > Users**, and then click **Add User** (icon).
2. Define the following:
  - **First Name, Last Name, and Email.**
  - **Preferred location:** Location for the user. See [Adding Locations and Organizations](#).

### ■ Account Type:

- **Database:** Accounts are stored in Foundation Hub. Use Database for accounts authenticated and managed within Foundation. Requires a **Username** and **Password**. The user cannot change the password.
- **Directory:** Typically an Active Directory account. Provide a **User Directory** (required) and a **Domain** (optional).
- **ServiceAccount:** Reserved for Foundation applications.

- **Roles:** Assign one or more Roles to this user. The user will inherit all of these roles' permissions. See [Managing Roles](#).

- **Group Memberships:** Assign this user to one or more groups. See [Managing Groups](#).

### Editing a User and Setting Additional Properties

After you create a new user, you can open it for editing to access additional settings for related groups and users, and to control whether an account is active, is flagged for automatic updates, or includes a reference ID:

1. Open **Admin and Settings > Security > Users**.
2. Click the row of the user you want to edit to view the details, and then click **Edit**.
3. From **Admin and Settings** page, click **Users**.
4. Select the user and click **Edit**.
5. Select values from the additional fields and edit original settings as needed:
  - **Related Groups/Users:** Associate groups that are related to the user, but to which the user does not belong, or choose related users.
  - **Collaborative Spaces:** Select the collaborative spaces in which the user is allowed to work and identify the user's access role for each space you select. For details, see [Managing Collaborative Spaces](#).
  - **Account is active:** Controls whether the user can log in.
  - **Include account in automated updates:** Flag that can be used by an external system to determine if the user account can be updated automatically. If the account comes from a user directory, the user's attributes will be updated to match what is in the directory when Foundation Hub user database is synchronized with the external directory (at login or per the synchronization schedule). See [Synchronizing Users](#).
  - **External Id:** Reference to an external system that contains additional information for the user.

### Deleting a User

If you created a user directly in Foundation Hub via Admin and Settings, you can delete it as follows:

1. Open **Settings > Security > Users**.
2. Choose the user you want to delete.
3. Click the **Delete User** (minus) icon.

**Note:** If the user is from a user directory, the account will be added back when Foundation Hub user database is synchronized with the external directory (at login or per the synchronization schedule). See [Synchronizing Users](#).

To block access for a user:

1. Open **Settings > Security > Users**.
2. Choose the user you want to block, and then **Edit**.
3. Uncheck **Account is Active**, and then **Save**.

## Chapter 3:

# Lifecycle and Signature Policy Administration

---

You can use lifecycle and signature policies to assign varying levels of signature requirements and other access controls to user actions. When a user attempts an action that has a signature requirement, the system displays a signature window and does not process the action until it collects the signature.

When it collects a signature, the system creates a signature event record in its audit history. It creates this record regardless of whether the signature meets the requirements of the signature policy—it logs failed, as well as successful signature submissions.

## Lifecycle and Signature Policy Definitions

This chapter discusses the following:

- *Lifecycles* identify the states for specific entity types, and lifecycle actions available for each state. For example, the Task lifecycle identifies task states and actions, whereas the Sample lifecycle identifies sample states and actions. The following table identifies the lifecycles and the entities that follow each lifecycle:

Lifecycle	Entities
Equipment	Equipment
Sample	Samples
Review Task	Review tasks
Publish	Activities and parameter templates
Specification	Activity plans
PEX	Process executions in Capture Hub. The following entities use the PEX lifecycle: <ul style="list-style-type: none"><li>■ Data acquisition tasks</li><li>■ Procedure tasks whose recipes require execution in Capture Hub</li></ul>
Task	Protocol tasks

- *Lifecycle actions* identify important changes that can impact an entity. Such changes include transitions from one state to another, deletion, ownership updates, and (for task and sample lifecycles), data-related situations like manual edits and out-of-limit results.
- *Signature policies* define the requirements of a signature window. Each signature policy indicates whether the window requires a signature. If it requires signatures, it also identifies the authorized signers. Each policy also indicates how to handle the signature window's Reason field and optional Comment field.
- A *lifecycle policy* lists all lifecycle actions and transitions that can impact each lifecycle entity, in each of its lifecycle states. It also provides a column so you can assign a signature policy to actions and transitions that require a signature. The configuration for each collaborative space identifies a default lifecycle policy for use in that space. It also identifies any alternative lifecycle policies that you allow in that space.

## Example Lifecycle and Signature Policies

Foundation Hub includes a set of signature policies and one default lifecycle policy. In the default lifecycle policy, each action that commonly requires a signature has an assigned signature policy. You can change these policies to meet your requirements, and you can create additional policies.

For example, you can:

- **Modify the installed signature policies**  
Create your own new signature policies.
- **Add, remove, or change the signature policy assignments in the default lifecycle policy.**
- **Clone the default lifecycle policy and change signature policy assignments in the clone, if the same lifecycle actions require different signature policies in different situations or collaborative spaces.**

For example, you could create a GxP collaborative space that identifies a very rigorous GxP lifecycle policy. When you set up that collaborative space, you would identify this rigorous policy, and not allow any alternative, less rigorous policies in that collaborative space. You could also create a Development collaborative space that identifies one or more less rigorous lifecycle policies, and that allows users to choose the most appropriate policy for their requirements.

**Note:** When you upgrade to a version of Foundation Hub that includes Lifecycle Policy enhancements, your Default Lifecycle Policy and any cloned lifecycle policies inherit the enhancements.

## Understanding Signature Policies

Ensure that you understand these concepts before you configure signature policies.

### Situations that Support Signature Policies

Signature policies can apply to a variety of situations, including:

- **Actions that impact lifecycle-driven entities.** You can create as many signature policies as you need, and map the appropriate signature policy to each action in your lifecycle policy. If the same action requires different signature policies for different environments, you can create a separate lifecycle policy and set of signature mappings for each environment. For details, see [Assigning Signature Policies to Lifecycle Actions](#) and [Cloning Lifecycle Policies to Support Signature Policy Variations](#).
- **Changes to master data entities in Admin and Settings.** Lifecycle-driven master data entities (activities, activity plans, and parameter templates) use the signature policies mapped in a lifecycle policy. You can implement a special signature policy, Master Data Signature, if you require signatures for changes to other master data entities. For details, see [Requiring Signatures for Changes to Master Data](#).
- **Data entry conditions during recipe execution.** You can require signature policies for a variety of data entry conditions that can occur during recipe execution. For each recipe, the Compose recipe author chooses which, if any, policy to apply to each condition. For details, see [Signature Policies for Submitting Recipe Data](#).

## Signature Policy Signer Types and Basic Settings

A signature policy specifies the requirements for a signature window.

Consider the following when you edit or create a signature policy:

- Do the actions for which you are creating the policy require an official e-signature? For some actions, you might want to create a signature policy that solicits a reason and free-form comment text, but that does **not** require a signature.
- Who can sign for the actions? To identify authorized signers such as user groups, you add one or more [Signer Types](#) to a signature policy.
- Do the groups that you authorized include anyone that you want to exclude from signing? If so, you can add a special "excluded" signer type to identify these exceptions. For example, if all members of a group except those who contributed to an entity can sign for changes to that entity, you can add two signer rules:
  - One with a signer type of Group that identifies the authorized group.
  - One with a signer type of Contributor excludes contributors to the entity from signing for changes, even if they are members of the group authorized by the other signer rule.
- What reasons must appear in the Reason selection list, and is a reason selection required? Is free-form reason entry allowed? Are comments allowed?
- Do the actions require more than one signer? Some actions support [signature policies that require multiple sign-offs](#), but others do not.

### Supported Signer Types

Signer Type	Meaning
<b>Any</b>	Any Foundation Hub user. <ul style="list-style-type: none"> <li>■ Use an <b>Any</b> rule and no other rules to allow anyone to submit the signature window.</li> <li>■ Use an <b>Any</b> rule and an <b>Exclude</b> rule to allow anyone except people in the <b>Exclude</b> rule to submit the signature window.</li> </ul>
<b>Current User</b>	Only the person who attempted the action (the logged-in user).
<b>Owner</b>	Only the current owner. <ul style="list-style-type: none"> <li>■ For entity types without an <b>Owner</b> property, the creator is the owner.</li> <li>■ For versioned entities, the owner is whoever created the version.</li> <li>■ Equipment is not versioned—the creator of the equipment also owns the equipment.</li> </ul> The Sample and Runset (Task Plan) lifecycles do not support this option.
<b>Contributor</b>	The owner, creator, or anyone who edited the entity. <ul style="list-style-type: none"> <li>■ For versioned entities, everyone who edited the affected version is a contributor.</li> <li>■ For equipment, everyone who edited the equipment after its last activation date is a contributor. The contributors list empties each time the equipment transitions from active to upgrading.</li> </ul> The Sample and Runset (Task Plan) lifecycles do not support this option.
<b>User</b>	If the <b>Signer Type</b> is <b>User</b> , select the user. Otherwise, leave this field empty.
<b>Group</b>	If the <b>Signer Type</b> is <b>Group</b> , select the group. Otherwise, leave this field empty.



## Signature Policies that Require More than One Signer

You can assign a signature policy that requires multiple signers to any action. However, only a subset of actions supports collecting more than one signature. These actions have a value of "yes" in the **Multi-Signer** column of the lifecycle actions table in [Lifecycle Actions and Default Signature Policy Assignments](#).

When a user attempts such an action, Foundation Hub redisplay the signature window until it collects a different signature for each signer rule. When the same signers could potentially satisfy more than one signer rule, an **On Behalf Of** field enables the system to prevent them from signing more than once. For other caveats about using this see the notes in [Adding and Editing Signature Policies](#).

If you apply a policy that requires more than one signature to an action that supports only one signature, Foundation Hub stops displaying the signature window as soon as it collects one authorized signature.

## Signature Policies that Control Review and Approval Steps

The **Publish** and **Specification** lifecycles control the activity, parameter template, and activity plan entities. These lifecycles support a series of up to four review and approval steps. You use signature policy assignments to control not only who can sign for a step, but also to control which steps you require.

**IMPORTANT!** Assign a signature policy *only* to the review and approval steps that you require. The system displays options to perform a review or approval step only if you assign a policy to the step. This behavior applies only to review and approval steps, not to any other types of actions or transitions.

The following table illustrates the approval process when all four steps have signature policies:

Required Lifecycle Step	0. Submit	1. Mark as Reviewed	2. Mark as Reviewed Step 2	3. Approve	4. Approve Step 2.
Resulting State	Submitted	Review in Progress	Reviewed	Approve in Progress	Approved

If you apply a signature policy to both review steps, the Lifecycle menu continues to display the **Mark as Reviewed** option until two signers successfully submit this option. The policy assigned to "Mark as Reviewed" controls who must sign for the first review. The policy assigned to the "Mark as Reviewed Step 2" action controls who must sign for the second review. The two approval steps work the same way as the two review steps.

As installed, *only Steps 1 and 3 have a signature policy*. Consequently, the lifecycle requires only one review step and only one approval step. The lifecycle state goes from *Submitted* to *Reviewed* to *Approved*, with no intermediate *In Progress* states:

Required Lifecycle Step	0. Submit	1. Mark as Reviewed	2. Approve
Resulting State	Submitted	Reviewed	Approved

If you remove signature policies from all steps, an entity's lifecycle state transitions directly to *Approved* as soon as a user saves it.

**Note:** Foundation Hub does not prevent the same user from signing for more than one step in a publish or specification lifecycle. Design your processes to ensure that you collect unique signatures for each step. Alternatively, create separate signature policies with separate user groups for each step, and ensure that the groups have no overlapping members.

### Installed Signature Policies

As installed, Foundation Hub provides the following types of signature policies:

- [Signature Policy for General Use](#)
- [Signature Policies for Updating Compose Recipe States](#)
- [Signature Policies for Updating Recipe Execution States](#)
- [Signature Policies for Submitting Recipe Data](#)
- [Signature Policy for Blocking an Action](#)
- [Master Data Change Signature Policy](#)

This section describes these signature policies and their default settings. For information about editing them and about creating additional signature policies, see [Adding and Editing Signature Policies](#).

#### Signature Policy for General Use

Foundation Hub provides a default signature policy called **Lifecycle Default Signature Policy**. In a default installation, this signature policy applies to actions that typically require a signature, as shown in [Lifecycle Actions and Default Signature Policy Assignments](#).

You can modify the settings in this signature policy and control which lifecycle actions require it.

As installed, this signature policy defines a signature window that accepts any credentials and that requires the signer to either select a built-in reason for signing or enter a free-form reason. By default, the window does not display the **Comment** field.

#### Signature Policies for Updating Compose Recipe States

Foundation Hub provides three signature policies that define signature requirements for approving, rejecting, and withdrawing a Compose recipe.

You can edit the settings in these policies, but you change signature policy assignments for Compose recipe state transitions. The following table describes the recipe state signature policies.

Policy Name	Default Required Signer	Action that Triggers Policy
<b>Recipe Approval</b>	Any member of Compose/Reviewers except a contributor	Compose recipe reviewer approves a recipe so that it can be published.
<b>Recipe Rejection</b>	Any member of Compose/Reviewers except a contributor	Compose reviewer rejects a recipe that requires corrections.
<b>Control Recipe Withdraw</b>	Any member of Capture/Withdrawers	Compose user withdraws a completed recipe so that it cannot be executed.

#### Signature Policies for Updating Recipe Execution States

Foundation Hub provides predefined signature policies for competing, approving, and rejecting classic Capture and Capture Hub recipe executions, including system-generated executions for data acquisition tasks.

You can edit these signature policies to change the authorized signers.

Recipes that execute in classic Capture always require these signature policies. Recipes that execute in Capture Hub always apply the predefined policy for recipe approval, but you can control what policy they apply, if any, for recipe completion and for recipe rejection.

The following table describes the predefined recipe execution signature policies.

Policy Name	Default Required Signer	Action that Triggers Policy
<b>Control Recipe Complete</b>	Current User	<p>Capture user completes a recipe execution.</p> <ul style="list-style-type: none"> <li>■ Classic Capture uses this signature policy for every recipe. You can edit the policy to change the required signer.</li> <li>■ Capture Hub uses the signature policy assigned to the PEX <b>Complete Execution</b> action in the lifecycle followed by the task. You can assign <i>Control Recipe Complete</i> or any other signature policy to this action. By default, this action has no signature policy assignment.</li> </ul>
<b>Control Recipe Approval</b>	Any member of Capture/Reviewers except a contributor	<p>Capture reviewer approves a completed recipe execution.</p> <ul style="list-style-type: none"> <li>■ Both classic Capture and Capture Hub always use this signature policy. You can edit the policy to change the signer rules, but you cannot assign a different signature policy to the approval action.</li> </ul>
<b>Control Recipe Redo</b>	Any member of Capture/Reviewers except a contributor	<p>Capture reviewer requests corrections to a completed recipe execution.</p> <ul style="list-style-type: none"> <li>■ Classic Capture uses this signature policy for every recipe. You can edit this policy to change the required signer.</li> <li>■ Capture Hub uses the signature policy identified in the PEX <b>Reopen Completed Execution</b> action in the lifecycle followed by the task. You can assign Control Recipe Redo or any other signature policy to this action. As installed, this action has no signature policy assignment.</li> </ul>

## Signature Policies for Submitting Recipe Data

Foundation Hub provides predefined signature policies for completing data entry, verifying data entry, and allowing submission of data when various irregular conditions occur.

You can edit the settings in these signature policies as required. The way you use them depends on task type. For data acquisition tasks, you can assign them (or *any* other signature policy) in the PEX section of your lifecycle policies. For procedure tasks, Compose authors assign them at the recipe level.

**Note:** Compose limits the signature policy selection list for each data entry condition to the following:

- The signature policy whose name matches the data entry condition.
- The predefined [Full Block](#) policy.
- Customer-defined (custom) signature policies.

The following table describes the predefined data entry signature policies.

Policy Name	Default Behavior (If assigned to the condition, and if you do not change the as-installed settings)
<b>Recipe Data Entry</b>	After a Capture or Capture Hub operator collects sample or step data, the operator must sign in order to continue to the next step or to complete the recipe.
<b>Recipe Data Verification</b>	After a Capture or Capture Hub operator collects sample or step data, a different user must sign before the operator can continue to the next step or complete the recipe.
<b>Recipe Data Modification</b>	If a Capture or Capture Hub operator modifies previously submitted data, the operator must sign to submit the modified data.
<b>Data Out of Limits</b>	If a Capture or Capture Hub operator enters data that is outside of planned limits, the operator must sign to submit the out-of-limit data.
<b>Expired Materials</b>	If a Capture or Capture Hub operator enters or scans an expired material, a different user must sign to allow the use of the material.

Policy Name	Default Behavior (If assigned to the condition, and if you do not change the as-installed settings)
* <b>Procedure Modification During Execution</b>	If a Capture Hub operator abandons a step in the recipe, a different user must sign to allow the step to be abandoned.
* <b>Material Substitution</b>	If a Capture Hub operator enters a material other than the planned material, the operator or any other user must sign to allow the substitution.
* <b>Metrology Exception</b>	If a Capture Hub operator collects data from equipment that has a metrology warning, a different user must sign to allow that data to be submitted.
* <b>GMP Equipment</b>	If a Capture Hub operator collects data for a GMP task and uses non-GMP equipment or inactive GMP equipment, a different user must sign to allow that data to be submitted.
* <b>Manual Override</b>	If a Capture Hub operator manually edits data that was previously collected from equipment, the operator must sign in order to continue.
* <b>Confirm Override</b>	If a Capture Hub operator deviates from the recipe in any of the following ways, a different user must sign to allow the submission of the data: <ul style="list-style-type: none"> <li>■ Overrides data collected from equipment by manually editing that data</li> <li>■ Manually enters the time instead of accepting the current time for a "Current Time" parameter.</li> </ul>

\* Recipes that execute in classic Capture do not use these policies.

### Signature Policy for Blocking an Action

Foundation Hub provides a special signature policy called **Full Block** that you can assign to prevent users from:

- Performing a lifecycle action or state transition that you cannot allow. You can assign the policy to any lifecycle action or state transition except **PEX** state transitions. (If you assign it to a **PEX** state transition, Capture Hub ignores it.)
- Submitting unacceptable data. You can prevent such submissions for data acquisition tasks by assigning the policy in the **PEX** section of your lifecycle policies. Compose authors can prevent such submissions at the individual recipe level.

One typical use for this policy is to prevent analysts in GMP environments from using inappropriate materials. If a Compose recipe author selects this policy for a Capture Hub recipe and an analyst scans material that does not match recipe requirements, Capture Hub displays a signature icon. However, when the analyst clicks the icon, Capture Hub displays an error message. The analyst must correct the situation before proceeding.

By default, all settings in this signature policy are either **No** or to **None**. If you change them, the policy might not continue to work as intended.

### Master Data Change Signature Policy

Foundation Hub provides a special unimplemented signature policy called **Master Data Change**. You can implement this policy if you require signatures for changes to master data entities, which are entities that you configure in Foundation Hub **Admin and Settings**.

By default, this rule requires the user who adds, edits, clones, changes, or deletes a master data entity to sign for their action. It supports, but does not require Reason entry, and it allows free-form reason entry.

To implement this policy, see [Requiring Signatures for Changes to Master Data](#).

## Entities Subject to the Master Data Change Signature Policy

The built-in Master Data Change signature policy enables you to require a signature whenever anyone adds, edits, clones, or deletes any of the following master data entities:

- |                             |                           |                                       |
|-----------------------------|---------------------------|---------------------------------------|
| ■ Collaborative Spaces      | ■ Inventory Systems       | ■ Data Packets                        |
| ■ External Claims           | ■ Legal Notices           | ■ Lifecycle Policies                  |
| ■ Groups                    | ■ License Files           | ■ Locations                           |
| ■ Roles                     | ■ Licenses                | ■ Organizations                       |
| ■ Equipment Classes & Types | ■ Links                   | ■ Pipeline Pilot Protocols            |
| ■ Equipment Adapters        | ■ Lifecycle Policies      | ■ Extended Properties, Properties     |
| ■ Projects                  | ■ *Logging                | ■ Readings                            |
| ■ Unit Categories           | ■ *Trusted Certificates   | ■ Reason Codes                        |
| ■ Unit Types                | ■ User Directories        | ■ Sequence Templates                  |
| ■ Applications              | ■ Domains                 | ■ Signature Policies                  |
| ■ Application Settings      | ■ Class Readings          | ■ Vocabularies and vocabulary entries |
| ■ Installations             | ■ Data Fields and Packets |                                       |
| ■ *Hub Configuration        |                           |                                       |

\*You can view signature and other history records for these entities only through the API. Use the History option on the Foundation Hub pages that list the other entity types to view their history. For more information, see the *Foundation Hub User Guide*.

**Note:** Although activities, parameter templates, and activity plans are master data entities, they have lifecycles. Consequently, they use signature policies from lifecycle policies, instead of using the **Master Data Change** policy.

## Lifecycle Actions and Default Signature Policy Assignments

The following table lists all lifecycle actions that can require signature policies, and identifies the predefined assignments in the predefined **Default LifeCycle Policy**. To add, change, or remove signature policy assignments, see [Assigning Signature Policies to Lifecycle Actions](#).

In the following table:

- Actions with **Default** in the **As Installed Signature Policy Assignment** column require the [LifeCycle Default Signature Policy](#).
- Actions with **yes** in the **Multi-Signer** column support signature policies that require more than one signer. Those with **no** can proceed after collection of only one required signature.

Lifecycle (Entities)	State	Available Action	As Installed Signature Policy Assignment	Multi-Signer
<b>Equipment</b> (Equipment)	Draft	Activate	Default	yes
	Upgrading	Activate	Default	yes
	Inactive	Activate	Default	yes
	Active	Inactivate	Default	yes
<b>Sample</b> (Samples)	Planned	Collect	—	no
	Collection Failed	Collect	—	no
	Planned	Delete Sample	—	yes
	Collection Failed	Delete Sample	—	yes
	Planned	Delete Task	—	yes
	Collection Failed	Delete Task	—	yes
	Collected	Dispose Sample	—	yes
	Available	Dispose Sample	—	yes
	Ready For Dispose	Dispose Sample	—	yes
	Planned	Make Manual Edits <sup>1</sup>	Default	no
	Collection Failed	Make Manual Edits <sup>1</sup>	Default	no
	Collected	Make Manual Edits <sup>1</sup>	Default	no
	Available	Make Manual Edits <sup>1</sup>	Default	no
	Ready For Dispose	Make Manual Edits <sup>1</sup>	Default	no
	Disposed	Make Manual Edits <sup>1</sup>	Default	no
	<sup>1</sup> Manual edits are edits made in the grids on the Samples widget and the Task Planner Samples tab.			

Lifecycle (Entities)	State	Available Action	As Installed Signature Policy Assignment	Multi-Signer
<b>Publish</b> (Activities and parameter templates)	Submitted	Mark as Reviewed	Default	yes
	Review in Progress	Mark as Reviewed Step 2	—	no
	Reviewed	Approve	Default	no
	Approval in Progress	Approve Step 2	—	no
	Approved	Reopen for Edit	Default	yes
	Approved	Withdraw	Default	yes
	Withdrawn	Reopen for Edit	Default	yes
	Superseded	Reopen for Edit	Default	yes
<b>Specification</b> (Activity plans)	Submitted	Mark as Reviewed	Default	yes
	Review in Progress	Mark as Reviewed Step 2	—	no
	Reviewed	Approve	Default	yes
	Approval in Progress	Approve Step 2	—	no
	Approved	Reopen for Edit	Default	yes
	Approved	Withdraw	Default	no
	Withdrawn	Reopen for Edit	Default	yes
	Superseded	Reopen for Edit	Default	yes
<b>Runset</b> (Task plans)	Draft	Change Owner	—	no
	Released	Change Owner	—	no
	Withdrawn	Change Owner	—	no
	Draft	Release Task Plan	—	yes
	Released	Reopen Task Plan	—	yes
	Withdrawn	Reopen Task Plan	—	yes
	Draft	Withdraw Task Plan	—	yes

Lifecycle (Entities)	State	Available Action	As Installed Signature Policy Assignment	Multi-Signer
Task <sup>1</sup>	Draft	Abandon Task	—	yes
	Submitted	Abandon Task	—	yes
	Assigned	Abandon Task	—	yes
	In Progress	Abandon Task	—	no
	Suspended	Abandon Task	—	no
	In Progress	Complete Task	—	yes
	Draft	Delete Task	—	yes
	Draft	Make Manual Edits <sup>2</sup>	Default	no
	Submitted	Make Manual Edits <sup>2</sup>	Default	yes
	Assigned	Make Manual Edits <sup>2</sup>	Default	yes
	In Progress	Make Manual Edits <sup>2</sup>	Default	yes
	Completed	Release Task	—	no
	Completed	Reopen Completed Task	Default	no
	Released	Reopen Released Task	Default	no
	Suspended	Resume Task	—	yes
	In Progress	Retest Task	—	no
	Completed	Retest Task	—	no
	Released	Retest Task	—	yes
	Draft	Suspend Task	—	yes
	Submitted	Suspend Task	—	yes
	Assigned	Suspend Task	—	yes
<sup>1</sup> If you require a signature for a Task state transition, you must also require one for the corresponding PEX state transition. Otherwise, the task and recipe lifecycle states can become out of sync. The policies you assign can be different, but one policy cannot require a signer that the other policy prohibits. <sup>2</sup> As of Foundation Hub 2021, signature policies for <b>Make Manual Edits</b> apply only to edits made using the Task Planner's <b>Edit Task</b> window, and only if the edited tasks use the <b>Use Latest Activity</b> setting. Collaborative space configuration controls the default for this setting, and an activity-level setting, <b>Lock Activity's Settings in Tasks</b> , controls whether users can override the default.				




Lifecycle (Entities)	State	Available Action	As Installed Signature Policy Assignment	Multi-Signer
PEX (Data acquisition tasks and procedures that use Capture Hub)	Draft	Abandon Execution	—	no
	Completed	Abandon Execution	—	no
	In Progress	Abandon Execution	—	no
	In Progress	Complete Execution	—	no
	Completed	Reopen Completed Execution	Default	no
	Released	Reopen Released Execution	Default	no
	Completed	Release Execution	—	no
	In Progress <sup>1</sup>	Data Entry	—	no
		Data Verification	—	no
		Data Modification	—	no
		Out Of Limits	—	no
		Expired Materials	—	no
		Material Substitution	—	no
		Procedure Modification	—	no
		Metrology Exception <sup>2</sup>	—	no
		GMP Equipment <sup>2</sup>	—	no
		Manual Override	—	no
		Confirm Override	—	no
	<p><sup>1</sup>The "actions" in this section are data entry conditions that can arise during task execution. Signature policies in this section apply only to data acquisition tasks. For procedure tasks, the recipe author controls signature policy assignments for data entry conditions. For information about built-in signature policies that you can assign to these conditions, see <a href="#">Signature Policies for Submitting Recipe Data</a>.</p> <p><sup>2</sup>Collection from non-GMP instruments or from inactive equipment types triggers the signature policy selected for GMP Equipment. All other metrology exceptions trigger the policy selected for the <b>Metrology Exception</b> action.</p>			
ReviewTask (Reviews)	Submitted	Abandon Task	—	yes
	Assigned	Abandon Task	—	yes

### Adding and Editing Signature Policies

A signature policy defines the behavior for a signature window. You can edit the default signature policy and create your own signature policies. After you create a signature policy, you can associate it with any action on the [Life Cycle Action Policies](#) page.

Before you add or edit signature policies, ensure that you understand the concepts in [Understanding Signature Policies](#).

#### To add or edit a signature policy:

1. Open **Settings > Signature Policies** and do either of the following:
  - To edit an existing policy, click its row, and then click **Edit**.
  - To add a new policy, click the **Add** (plus sign) icon, and then enter a name and description for the policy. Alternatively, select an existing policy to use as the starting point, and then click **Clone Selected** .
2. Under **General**, select or change the **Category**. Foundation Hub provides two built-in choices for categorizing signature policies:
  - **e-record** – Use for signature policies that you intend to require for actions that affect entities and their lifecycle states.
  - **data entry** – Use for signature policies that you intend to require for data results that differ from expectations, or that were collected under conditions that differ from expectations.

You can add other categories to the built-in Foundation Hub vocabulary called **Signature Policy Category**.

3. Under **Authorization Requirements**, select the following check boxes, if appropriate:
  - **Credentials Required** – Indicates that the policy requires entry of a username and password.
  - **View Content Required** – Indicates that users must review content before they sign. This option does not impact system behavior.
4. Add and edit rows in the **Signature Policy Signers** table to identify authorized and excluded signers:
  - To add a rule, click **Add** and proceed to the next step.

**IMPORTANT!** Add an authorized signer rule even if you did not select **Credentials Required**. If you do not need credentials, add a signer rule that identifies **Any User**. If you do not add a rule, users cannot perform actions that use the signature policy. When they attempt to do so, the system displays an error instead of the signature window. The error instructs the user to ask an administrator to correct the policy configuration.

- To edit a rule, click its name, click **Edit** in the **Definition** window, and enter your changes.
  - To remove a rule, hover over its name and click the **Unlink** icon in the third column.
5. Use the **Add Signature Policy Signer** window to add at least one signer rule.

**Notes:**

- To require a member of more than one group to sign, add a signer rule for each group. If the groups have overlapping members, be sure that everyone who can attempt actions that require this signature policy is also an authorized signer. Otherwise, if a user who belongs to more than one of the groups is the first person to attempt to sign, the system cannot obtain the groups for the **On Behalf Of** field.
- To identify actions that support multiple signatures, see the table in [Lifecycle Actions and Default Signature Policy Assignments](#).

- a. In the **Name** field, enter a name to identify your signer rule. Examples:
  - **Any User**
  - **Any User Except Current User**
  - **Any GroupX Member**
- b. If this signer rule identifies users who you do **not** allow to sign, select **Excluded**. Exclusion rules take precedence over other signer rules.

**Note:** Policies that include a rule with this option must also include a rule that does not use this option. Do not create policies whose only rules are exception rules.

- c. Select the **Signer Type** and, if relevant, select the user or the group.

Signer Type	Meaning
<b>Any</b>	Any Foundation Hub user. <ul style="list-style-type: none"> <li>■ Use an <b>Any</b> rule and no other rules to allow anyone to submit the signature window.</li> <li>■ Use an <b>Any</b> rule and an <b>Exclude</b> rule to allow anyone except people in the <b>Exclude</b> rule to submit the signature window.</li> </ul>
<b>Current User</b>	Only the person who attempted the action (the logged-in user).
<b>Owner</b>	Only the current owner. <ul style="list-style-type: none"> <li>■ For entity types without an <b>Owner</b> property, the creator is the owner.</li> <li>■ For versioned entities, the owner is whoever created the version.</li> <li>■ Equipment is not versioned—the creator of the equipment also owns the equipment.</li> </ul> The Sample and Runset (Task Plan) lifecycles do not support this option.
<b>Contributor</b>	The owner, creator, or anyone who edited the entity. <ul style="list-style-type: none"> <li>■ For versioned entities, everyone who edited the affected version is a contributor.</li> <li>■ For equipment, everyone who edited the equipment after its last activation date is a contributor. The contributors list empties each time the equipment transitions from active to upgrading.</li> </ul> The Sample and Runset (Task Plan) lifecycles do not support this option.
<b>User</b>	If the <b>Signer Type</b> is <b>User</b> , select the user. Otherwise, leave this field empty.
<b>Group</b>	If the <b>Signer Type</b> is <b>Group</b> , select the group. Otherwise, leave this field empty.

- d. If you require only one signer, click **OK**. Otherwise, select **Add another**, click **OK**, and repeat

these steps. After you finish adding the remaining signers, clear the **Add another** check box, and then click **Cancel** to close the window.


6. Under **Signing Reason Requirements**, identify how you want the signature window to handle reasons and comments:
  - a. From **Reason Code Policy**:
    - To exclude the **Reason** field from the signature window, select **No**.
    - To include the field, but make selection of a reason optional, select **Optional**.
    - To include the field and make an entry mandatory, select **Yes**.
  - b. If you selected **Optional** or **Yes**, consider the following:
    - To provide a list of valid reasons, click **Add**, and then enter the first reason.  
Select the **Add another check box** before you click **OK** to keep entering reason codes until you have entered them all, and then click **Cancel**.  
Alternatively, use the [Signature Policies > Reason Codes](#) page to create and manage reason codes for all signature policies.
    - To allow signers to enter a custom reason in the **Reason** field, select **Free-form Reason Allowed**.
    - To provide a **Comment** block so that signers can enter comments, select **Free-form Comment Allowed**.
7. **Save** your settings.
8. To assign the signature policy to the actions that require it, see [Assigning Signature Policies to Lifecycle Actions](#).

## Cloning Lifecycle Policies to Support Signature Policy Variations

If your signature requirements vary from one development phase or environment to another, you can create a separate lifecycle policy for each environment.

For example, suppose you allow the initiator of most task-related actions to sign for those actions in one environment, but you require the initiator *and* a supervisor to sign for them in another environment. You can accommodate this scenario by cloning lifecycle policies and varying the signature requirements you assign to the actions in each cloned policy.

### To clone a lifecycle policy:

1. Open **Admin & Settings > Life Cycle Policies**.
2. Select the policy to clone, and then click **Clone** .
3. Edit the clone to give it a meaningful name and description.
4. In your clone's **Life Cycle Action Policies** table, update the Signature Policy column values as required. For details, see [Assigning Signature Policies to Lifecycle Actions](#).

**Tip:** To remove a signature policy assignment, select the blank (first) entry in the Signature Policy list.

5. Assign all **Collaborative Spaces** that must support this policy.  
To remove an assignment, click its name, and then click its **Unlink this item** icon.
6. Click **Save**.

## Assigning Signature Policies to Lifecycle Actions

A signature policy controls the requirements and behavior of a signature window. You use the **Life Cycle Action Policies** table to assign signature policies to lifecycle actions that require signatures. Before doing so, ensure that you are familiar with the concepts in [Understanding Signature Policies](#).

Each row in the Life Cycle Action Policies table identifies an action or state transition to which you can assign a signature policy.

**IMPORTANT!** Test your policies carefully to ensure that they work as intended. It is possible to unintentionally block actions and to prevent recovery from data entry mistakes.

**To assign a signature policy to a lifecycle action:**

1. Open **Admin & Settings > Life Cycle Policies**.
2. Click the lifecycle policy to change, and then click **Edit**.
3. In the **Life Cycle Action Policies** table, find the action that requires a signature policy:
  - a. Find the set of rows that identify the **Life Cycle** for your action. For example, if the action affects samples, find the rows that list **Sample** in their **Life Cycle** column.  
Some lifecycles apply to more than one entity. For a table of the lifecycles and the entities that follow them, see [Lifecycle and Signature Policy Definitions](#).
  - b. Within the set of rows from 3a, find those that identify the **Name** of the action.  
If more than one row identifies the action, note the values in the **State** column. You can assign different signature policies for each lifecycle state that supports the action.
  - c. To assign a signature policy to an action identified by a row, click in the **Signature Policy** cell, and then select the required policy.

**Tip:** When you assign a policy to a PEX lifecycle action that changes an *entity's state*, the policy applies to procedure tasks and to data acquisition tasks. However, when you assign a policy to a PEX lifecycle action that describes a *data entry condition*, the policy applies only to data acquisition tasks. Each individual recipe, not the PEX lifecycles, control data entry signature requirements for procedure tasks. For more information, see [Lifecycle Actions and Default Signature Policy Assignments](#).

4. Click **Save**.

**To remove an action's signature policy association:**

1. Perform Steps 1–3 above.
2. Click the row that contains the action.
3. Click **Edit**.
4. Select the blank entry at the top of the first page of the **Signature Policy** list.
5. Click **Save**.

### Requiring Signatures for Changes to Master Data

If you require a signature whenever anyone adds, clones, edits, or deletes a master data entity, you must:

- Edit the default settings in the [Master Data Change Signature Policy](#), if they do not meet your needs.

By default, this policy requires the user who changes master data to sign for their own changes, provides a single reason code, and allows free-form reason entry. You can change the policy to specify a different signer, but you cannot require more than one signer for changes to master data.

- Change the value of a [global system flag](#) to make the system enforce the policy.

#### To edit the default settings of the Master Data Change Signature Policy:

1. Open **Admin and Settings > Signature Policies**.
2. Click the row that contains **Master Data Change**, and then click **Edit**.
3. If required, change the **Signing Reason Requirements**:
  - a. From **Reason Code Policy**, select one of the following:
    - **Optional** – Users can select a reason, but you do not require them to.
    - **No** – Users cannot select a reason. The window does not display the **Reason** field.
    - **Yes** – Users must select a reason.
  - b. If you allow or require a reason and need to enter specific reasons to allow, click **Add**, enter the first reason, select **Add another**, and enter the remaining reasons. Then click **Cancel** when you finish.
  - c. To allow users to enter their own reasons, select **Free-form Reason Allowed**.
  - d. To provide a Comment field into which users can enter comments, select **Free-form Comment Allowed**.

#### To change the global flag:

1. Open **Foundation Hub Admin & Settings > Applications > Application Settings > Foundation Hub** and click **Edit**.
2. Navigate to the **Master Data** section.
3. Change the default value of **Signature Level** from **None** to **Signature**, and then click **Save**.

## Managing Signature Policy Reason Codes

Each signature policy has its own reason codes. To make two different signature policies offer the same new reason code, you must define the new reason code twice—once for each signature policy. You cannot associate the same reason code with more than one signature policy.

You can add reason codes to a signature policy when you create or edit the policy. For details, see [Adding and Editing Signature Policies](#). Alternatively, you can add reason codes by using the **Reason Codes** page, as described below. This page lists all reason codes for all signature policies.

**To add, edit, or delete a Reason Code:**

1. Open **Settings > Signature Policies > Related Items > Reason Codes**.
2. To add a reason code, click **Add**, and then enter a name and select the signature policy that uses it.
3. To edit a reason code, click its name, and then click **Edit**. You can change the name. You can also change the Signature Policy that uses it.
4. To delete a reason code, select its check box, and then click **Delete**.

# Chapter 4:

## Vocabulary and Parameter Template Administration

---

In Foundation Hub:

- A vocabulary identifies a subject and provides a list of entries that describe different attributes of that subject. Foundation Hub includes a suite of predefined vocabularies. You can expand the entries in many of these vocabularies. You can also define your own vocabularies.
- A parameter template defines a data parameter that you can use in Foundation Hub activities and activity plans, in Compose and Capture process elements, and in other integrated applications.

### Vocabularies

A vocabulary identifies a subject and provides a list of entries that describe different attributes of that subject. Foundation Hub includes a suite of predefined vocabularies. You can expand the entries in many of these vocabularies. You can also define your own vocabularies. A vocabulary can have a relationship with another vocabulary and each entry in such a vocabulary can be required to identify a corresponding entry in the related vocabulary.

Foundation Hub and other BIOVIA applications use vocabularies to define and control laboratory processes in numerous ways.

The following are just a few examples of the predefined vocabularies:


- **Activity Category:** Categories used to describe an activity.
- **Location Type:** Site, Building, Floor, Room, Lab.
- **Material Class:** Material classification, such as Chemical Solution or Biologics Protein. You can define *extended properties* based on the material class.
- **Organization Type:** Vendor, Customer, Organization, Division, Department.
- **Sample Event Type:** Types of events that can be performed during the lifetime of a sample.
- **Sample State:** Lifecycle states for samples.
- **Signature Policy Category:** Organizational categories for signature policies.
- **User Relationship Type:** Type of relationship of a user or group to the parent user.

**Tip:** To see all built-in vocabularies, sort the list in **Admin & Settings > Resources > Vocabularies** by the **Locked** column.

### Adding Vocabularies and Entries

**IMPORTANT!** Some applications require the entries in a vocabulary to be arranged in a hierarchy. You cannot create a hierarchical vocabulary in the Foundation Hub web interface. Instead, you must use the Foundation Hub REST API. For instructions, see [Adding Hierarchical Vocabularies](#).



1. Open **Resources > Vocabularies**.
2. Click **Add Vocabulary**  and enter appropriate information about your new vocabulary:
  - **Name:** Vocabulary name.
  - **Description:** Vocabulary description.
  - **Category:** Category to which your vocabulary belongs, such as the name of the application that will use it.
  - **Related Vocabulary:** Another vocabulary to use in conjunction with this vocabulary, if needed.
  - **Require Related Entry:** Indicates that each entry in this vocabulary must identify a corresponding entry in the selected **Related Vocabulary**.
3. Click **Save**.
4. Edit the vocabulary to add entries and, if relevant, identify its External ID and the Related Vocabulary:
  - a. Open **Resources > Vocabularies**.
  - b. Click the row of the vocabulary to view its details.
  - c. Click **Edit**.
  - d. Add and manage the vocabulary entries:
    - **Add:** Click **Add** and provide a **Name** and optional **Description**. If the vocabulary has **Require Related Entry** set to true, choose a **Related Entry**.
    - **Edit:** Double-click the entry.
    - **Delete:** Hover over an entry and click the **Remove** icon (entries provided by BIOVIA cannot be removed).
    - **Change the order:** Highlight the entry and use the **Up** and **Down** buttons.

**Notes:**


- You can also manage vocabulary entries from the **Vocabulary Entries** page. See [Viewing Vocabulary Entries](#).
- Vocabulary entries can have aliases. You can manage these by editing a vocabulary entry (see [Viewing Vocabulary Entries](#)) or from the **Vocabulary Aliases** page (see [Viewing Vocabulary Aliases](#)).

- e. Click **Save**.


## Editing Vocabularies

1. Open **Resources > Vocabularies**.
2. Click the row of the vocabulary to view the details.
3. Click **Edit**.

## Deleting Vocabularies

1. Open **Resources > Vocabularies**.
2. Select the check box and click the **Delete Vocabulary** button . This option is not available for vocabularies provided by BIOVIA.

## Cloning a Vocabulary

1. From the **Admin and Settings** page, click **Vocabularies**.
2. Select the check box for the vocabulary that you want to clone.
3. Click the **Clone Selected** icon .
4. In the **Confirm Clone** dialog box, click **Yes**.

A clone of the vocabulary is created, and an editor for the new vocabulary opens.

By default, the name of the new vocabulary is that of the source vocabulary, with a numeric suffix to make it unique. For example, if you clone the vocabulary Vessel Lining, the first cloned vocabulary is called Vessel Lining(2), the second Vessel Lining(3), and so on.

When you clone a vocabulary, related vocabulary entries (and their aliases) are cloned with it.

5. Edit the details of the cloned vocabulary.
6. Click **Save**.

## Adding Hierarchical Vocabularies

Some applications require the entries in a vocabulary to be arranged in a hierarchy. For example, the default Location Type hierarchy is **Site > Building > Floor > Room > Lab**.

You cannot create a vocabulary hierarchy in the Foundation Hub **Admin and Settings > Vocabularies** pages. Instead, you must use the REST API to create each lower-level entry in the vocabulary, specifying its parent entry.

### To create a hierarchical vocabulary:

1. Ensure that the vocabulary to which you want to add entries exists in the **Admin and Settings > Vocabularies** list.
2. Click the vocabulary to see its details. For example, click **Location Type** if you will be adding a new location type.
3. In your browser's address bar, copy the ID of the vocabulary. This is the part of the address after the final slash:

```
https://mybioviaserver:9953/foundation/hub/admin#/setting/Vocabularies/34cac102-8ae6-40a7-8dc2-38841c4e235f
```

Paste the vocabulary ID into a text editor. You will need it later.

4. In the **Entries** list, click the entry that will be the parent of the new entry. For example, if you will be creating an entry **Suite** under **Floor**, click **Floor**.
5. In your browser's address bar, copy the ID of the vocabulary entry:

```
https://mybioviaserver:9953/foundation/hub/admin#/setting/Vocabulary Entries/a5ddd3fb-1224-4e2f-b7d1-ec4204cbc548
```

Paste the vocabulary entry ID into a text editor for later use.

6. Using a REST client application (for example, a browser plugin), send a POST request to the `/foundation/hub/api/v1/vocabularyentries` endpoint on the Foundation Hub host computer. Specify `application/json` in the Content-Type header. In the request body, supply the new vocabulary using the following JSON structure:

```
{
  "description": "A suite on a floor.",
  "displayOrder": 3,
  "name": "Suite",
  "parent": "34cac102-8ae6-40a7-8dc2-38841c4e235f"
}
```

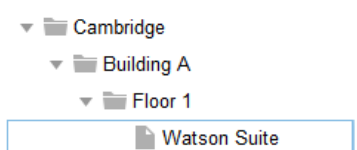
optional description.  
Position of the new vocabulary entry in the UI list.  
Name of the new entry.

```

"relatedEntry":{
  "id":"a5ddd3fb-1224-4e2f-b7d1-ec4204cbc548"  ID of the parent entry. This is what you copied in Step 5.
},
"vocabulary":{
  "id":"34cac102-8ae6-40a7-8dc2-38841c4e235f", ID of the vocabulary. This is what you copied in Step 3.
  "name":"Location Type"                      Name of the vocabulary.
}
}

```




7. Verify that the vocabulary entry was submitted correctly:
  - a. In your REST client application, check the response from the Foundation Hub server. This should be a 201 "Created" response with the newly created vocabulary entry in the body.
  - b. In the Foundation Hub **Admin and Settings** pages, go to the page for the resource type that uses the hierarchical vocabulary, and check that the hierarchy works as expected.  
 For example, if you added the **Suite** location type described above, select **Admin and Settings > Locations**. Add a top-level location, then add sublocations, and verify that you can add a **Suite** as the child of a **Floor**:



8. Repeat Steps 4 to 7 for each new entry in the vocabulary.

## Viewing Vocabulary Entries

To view a list of all vocabulary entries:

1. From the **Admin and Settings** page, click **Vocabularies**.
2. Click any vocabulary.
3. In the vocabulary's page, click any item in its **Entries** list.  
 A page opens, containing details of the vocabulary entry.
4. In the hierarchy path near the top of the page, click the link **Vocabulary Entries**.
5. A list of all vocabulary entries opens. In this list, you can:
  - Add an entry by clicking the **Add Vocabulary Entry** button . Provide a **Name** and optional **Description**. If the vocabulary has **Require Related Entry** set to true, choose a **Related Entry**.
  - Edit an entry by double-clicking it.
  - Delete an entry by selecting its check box and then clicking the **Delete Selected** button .
  - Clone an entry by selecting its check box, clicking the **Clone Selected** button , and supplying a name and a description. Click **Save** to save the cloned vocabulary entry.

## Viewing Vocabulary Aliases

Vocabulary aliases are alternative terms for vocabulary entries.




To view a list of all vocabulary aliases:

1. From the **Admin and Settings** page, click **Vocabularies**.
2. Click any vocabulary.
3. In the vocabulary's page, click any item in its **Entries** list.

A page opens, containing details of the vocabulary entry.

4. In the hierarchy path near the top of the page, click the link **Vocabulary Entries**.
5. Click the row of a vocabulary entry that has a non-zero value in the **Alias Count** column.
6. In the vocabulary entry's page, click any item in its **Aliases** list.

A page opens, containing details of the vocabulary alias.

7. In the hierarchy path, click the link **Vocabulary Aliases**.
8. A list of all vocabulary aliases opens. In this list, you can:
  - Add an alias by clicking the **Add Vocabulary Alias** button , specifying a name, description, and parent vocabulary entry. Click **OK** to save the vocabulary alias.
  - Delete an alias by selecting its check box and then clicking the **Delete Selected** button .
  - Clone an alias by selecting its check box, clicking the **Clone Selected** button , and supplying a name, vocabulary entry, and optional description. Click **Save** to save the cloned vocabulary alias.

## Parameter Templates


A parameter template defines a data parameter that you can use in Foundation Hub activities and activity plans, in Compose and Capture process elements, and in other integrated applications.

Foundation Hub comes with a set of predefined parameter templates and you can also create your own. Predefined parameter templates have a Type of **System**, whereas user-defined parameter templates have a Type of **Process**. You cannot change the Type.

### Permissions:

- Foundation/View Master Data is required to view parameter templates.
- Foundation/Manage Master Data is required to create, edit, delete, clone, and manage the life cycle of parameter templates.

## Creating a Parameter Template

1. Open **Admin and Settings > Resources > Parameter Templates**.
2. Click **Add Parameter Template** .
3. Define the template:
  - **Name:** Name of the parameter.
  - **Description:** Description of the template.
  - **Version:** Identifies the major and minor version of the parameter template. For new parameter templates, the version is **1.0** by default, but you can change it.


The number that precedes the decimal point indicates the major release, and the number that follows it indicates the minor release. A minor release is an update to a major release. For example, 2.10 indicates the tenth update to version 2. For more information, see [Creating a New Version of an Approved Activity](#).
  - **Life Cycle Policy:** [Lifecycle policy](#) that identifies the approval requirements for this parameter template. If the [collaborative space](#) supports more than one lifecycle policy, you can choose which one to use.

- **Reference Ontology:** A predefined reference ontology, if appropriate. Examples include:
    - Preparation
    - Metrology
    - Qualification
    - Characterization
4. To create another parameter template, select the **Add another** check box; otherwise click **OK**.
  5. Proceed to Editing a Parameter Template to see and edit additional fields, including **Value Type**, which defaults to **Text**, and **Version**, which defaults to 1.0.



## Editing a Parameter Template

**Note:** You cannot edit an approved parameter template, but you can [create a new version](#) that, when approved, supersedes the previous version.


When editing a System template, do *not* change the *Value Type* or any actual and planned value settings.

1. Open **Resources > Parameter Templates**.
2. Double-click the parameter template.
3. In the parameter template view, click **Edit** .
4. Make any required changes to the following:
  - **Name, Display Name, Description, and Version.**
  - **Actual Value Exists, Actual Value Required, Planned Values Exists, and Planned Value Required.**
  - **Value Type:** Type of value required for parameters based on this parameter template. Defaults to **Text**, but you can change it to any of the following:
    - A standard data type such as **Boolean, DateTime, Integer, LongText, or Numeric.**
    - **File** (Compose and Capture does not support **File** parameter templates.)
    - **Vocabulary** (A vocabulary is available for use in a parameter template only if its Category is set to Compose.)
    - **Quantity**
    - **ReferenceList**
    - **MaterialMapping**
    - **SampleMapping**
  - **Tag**
  - **External ID**
  - **Reference Ontology**
5. If your parameter template's *Value Type* is **Vocabulary**, select a vocabulary and the specific entries within that vocabulary to make available to users of this parameter template.




**Note:** The only vocabularies you can use for parameter templates are those with a **Category** of **Compose**.

6. If your parameter template's *Value Type* is **Quantity**:
  - a. Select the appropriate unit types and at least one unit within each unit type.
  - b. Select the **Default Unit** to apply when users select this parameter template.
7. Click **Submit** .
8. To start the approval process, select **Submit** from the **Lifecycle Actions**  menu. Your parameter template cannot be used until it is approved. For more information, see [Managing the Parameter Template Life Cycle](#).

### Deleting a Parameter Template

1. Open **Resources > Parameter Templates**.
2. Check the check box for the parameter template to remove.
3. Click **Delete Selected** .

### Cloning a Parameter Template

1. Open **Resources > Parameter Templates**.
2. Select the check box for the parameter template to clone.
3. Click **Clone Selected** .
4. In the **Confirm Clone** message, click **Yes**.
5. In the parameter template view, click **Edit** .
6. Edit the details of the clone.
7. Click **Submit** .


### Managing the Parameter Template Life Cycle

Each parameter template identifies a [lifecycle policy](#). Settings in the publish section of the policy control the following:

- Review and approval steps required to get a parameter template approved
- Signature policy for each required review and approval step
- Signature policy for withdrawing an approved parameter template to prevent further use
- Signature policy for reopening an approved parameter template to create an updated version

### Reviewing and Approving or Rejecting a Parameter Template

Perform this procedure as many times as required to advance the lifecycle state from *draft* to *approved*:

1. Open **Admin and Settings > Resources > Parameter Template**.
2. Click the row of the parameter template.
3. Open the **Lifecycle Actions** menu  and click the option required to advance the parameter template to its next state. Alternatively, click **Reject** to return it to its *draft* state for changes.

The following table lists the menu options for each possible review and approval step. Default installations exclude the Step 3 and Step 5 current states and menu options. A different person

should perform each required step.

Step	Current State	Menu Option
1.	Draft	Submit
2.	Submitted	Mark as Reviewed (or Reject)
3.	Review in Progress	Mark as Reviewed (or Reject)
4.	Reviewed	Approve (or Reject)
5.	Approval in Progress	Approve (or Reject)

4. Complete the signature policy window.



**Note:** If the updated **Life Cycle State** is unexpected, refresh the page. States that correspond to steps you skip are displayed temporarily after you submit the signature window, but the correct states are saved.

## Creating a New Version of an Approved Parameter Template

If changes are required to an approved parameter template, use **Reopen for edit** to create a new version. This action clones the approved version to create a new *draft* version and sets the draft's minor version (number that follows the period) to its previous value, plus **1**. It does not change the major version (number that precedes the period).

For example, reopening version **1.9** creates a draft for version **1.10**, and reopening version **1.10** creates a draft for version **1.11**. You can manually increment the major version when necessary, as well as change the minor version to suit your version numbering conventions.

**To create a new version of a parameter template:**


1. Open **Resources > Parameter Templates**.
2. Open the parameter template.
3. On the **Lifecycle Actions**  menu, click **Reopen for edit**.
4. If required, complete the signature window.
5. In the breadcrumb trail, click **Parameter Templates**.
6. On the Parameter Templates page, find and open the new draft version of the parameter template.
7. [Edit the new version as needed](#).
8. On the **Lifecycle Actions**  menu, click **Submit** to start the review and approval process.

## Withdrawing an Approved Parameter Template from Use

When you no longer want users to be able to select an approved parameter template, you can withdraw it. When you withdraw it, its lifecycle state changes from *approved* to *withdrawn*.

**To withdraw an approved activity:**

1. Open **Admin and Settings > Resources > Parameter Templates**.
2. Click the row of the parameter template.

3. On the **Lifecycle Actions** menu , click **Withdraw**.
4. Complete the signature window.



# Chapter 5:

## Activity and Activity Plan Administration

---

In Foundation Hub, activities and activity plans are templates for tasks in task plans:

- An *activity* identifies the input and output parameters that define a general type of *task*, as well as instructions and any relevant limits and conditions.
- A *task* is a specific instance of an activity that a scientist or other user creates when they need to perform the activity. Unless the activity prohibits such changes, the user configures the task as needed. Common configuration changes include refining or adding defaults for parameter values and excluding any unnecessary output parameters .
- A *task plan* is a set of related tasks that are created and managed using *Task Planner*. Task Planner provides an interface for generating tasks for activities, controlling assignment and execution of those tasks, and viewing task results.

You can group related activities into *activity plans*. Activity plans enable Task Planner users to quickly generate a set of related tasks. They can also specify the order of the tasks and several other characteristics that cannot be specified in individual activities.

### Activity Types

Foundation Hub categorizes activities by **activity type**. The way an activity is created and managed and the way tasks for an activity are handled vary based on activity type:

- **Procedure.** Procedures are Compose recipes that execute in classic Capture or Capture Hub. When a Compose recipe is published, a corresponding draft procedure activity is created. The draft procedure activity is configured in Foundation Hub, as needed. For example, planned values for samples and materials are entered, and any necessary task-level parameters are added. The draft is then routed through an approval process. A procedure activity must be approved before it can be used for production. Until it is approved, its owner is the only one who can see it in Task Planner and use it to generate tasks. Although other users cannot create tasks for unapproved activities, they can execute and test tasks created by the activity owner.
- **Data Acquisition.** Data acquisition activities identify data elements to gather from equipment and enter in Task Planner. The data can be entered using automated or manual methods. For automated methods, users execute a data acquisition task to view and validate the data. For manual methods, they execute the task to upload or manually enter data. Data acquisition activities are created, as well as configured, directly in Foundation Hub. Each data acquisition activity identifies what data elements to collect and what equipment class to use to collect them.
- **Protocol.** Protocol activities identify a Pipeline Pilot protocol that can be added to a task plan and executed directly from the plan. Protocols typically post-process results gathered through other activities and generate reports. Protocols can use task input parameters and can also generate results for task output parameters.

Protocol activities are created and configured directly in Foundation Hub.

**IMPORTANT!** Foundation Hub includes only one out-of-the-box activity: **Review**. Do not edit it, move it, or remove it. Doing so can cause problems with review tasks.

### Templating Techniques

It is important to configure an appropriate suite of activities and activity plans. Plan them carefully to support re-use when appropriate and to avoid creating more than you really need. A well-designed set of activities and activity plans can reduce maintenance and increase usability.

Using templating techniques is one way to maintain control. There are several ways to do this:

- Create [parameterized child activities](#).

If your organization performs numerous activities that all collect the same data parameters, but that require different planned values, limits, and precision settings, you can use a parent-child approach.

For this approach, you identify the required parameters in one activity (the parent) and then generate each child activity. In each child, you edit the parameters' planned values, limits, and precision settings to reflect the needs of one product or material. You also edit basic child activity settings such as name, description, lifecycle policy, and version.

For example, suppose 70 products require an Oven Temperature and a Baking Duration input parameter and a Loss on Drying output parameter. For this scenario, you can create a parent activity that identifies these parameters and then generate and edit 70 child activities, one for each product. The activity for each product specifies its own input values for Oven Temperature and Baking Duration, and its own passing value and limits for the Loss on Drying output parameter.

**Note:** If you edit or reversion a parent activity, you must recreate each of its required child activities, approve them, and then withdraw the old version of each child. For assistance deploying a Pipeline Pilot protocol to automate this process, contact Dassault Systèmes Customer Support.

- Create a Compose master recipe with process input parameters and conditions whose values vary by product, and handle the variations through activity plans.

After publishing the recipe to Foundation Hub as an activity, add the activity to a separate activity plan for each product, and enter the appropriate process input values for each particular product in each activity plan the same way you would for child activities.

- Clone a superset activity and remove unnecessary parameters from the clones.

When you need a set of activities that have many, but not all, of the same parameters, you can create an activity that identifies all of the parameters, clone it once for each individual activity you need, and then edit each clone to remove its unnecessary parameters and adjust other settings. There is no linkage or master-parent relationship when you clone an activity.

- Clone a superset activity plan.

As with superset activities, when you need several (or numerous) activity plans that require many of the same activities, you can create a template activity plan that contains each of the possible activities. Then clone the template activity plan and refine each clone by removing unnecessary activities and adjusting their settings.

Keep in mind that Task Planner generates one task per activity in an activity plan, but Task Planner users can remove any unnecessary tasks. Task Planner users can also omit any activity parameters that they do not need, unless the activity is configured to prevent such edits. Consequently, even your lowest-level activities can identify more parameters than are always needed, and your lowest-level activity plans can identify more activities and activity characteristics than are always needed.

## Permissions Required

To view, add, and edit activities and activity plans, you or a group to which you belong must have the following Foundation Hub permissions:

- Foundation/Administration/Logon (to access Admin and Settings)
- Foundation/View Activities (to view the activity pages)
- Foundation/Manage Activities (to edit activities)

## Activities


An activity specifies the following:

- Input and output parameters required for a particular type of task.
- Lifecycle policy that identifies sign-offs for the activity; the collaborative space configuration controls whether there is a choice.
- Optional settings, such as whether the parameter set can be modified at the task level.
- Information-only details such as project and department.

## Adding an Activity

**Note:** To add a procedure activity, publish a recipe from Compose. You can use Foundation Hub to configure procedure activities published from Compose, but not to create them.

Perform these steps to enter general information (metadata) for a new data acquisition or protocol activity:

1. Switch to the collaborative space in which to store the activity. For more information, see [Managing Collaborative Spaces](#).
2. Open **Admin and Settings > Resources > Activities**.
3. Click **Add Activity** .
4. Enter appropriate values for the following:
  - **Name:** Identifies the activity.
  - **Activity Type:** Indicates the purpose of the activity:
    - **Protocol:** Execute a Pipeline Pilot protocol. See [Managing Pipeline Pilot Protocols](#).
    - **Data Acquisition:** Identify parameters to collect from equipment.
  - **Category:** Categorizes the activity so that users can find it by filtering and sorting on Category values. You can select an existing category or add a new category. If you add a new one, it is included in the Category selection list for all activities.

**Note:** Do not change Category values for procedure activities.

- **Project:** Associates the activity with a project so that users can find it by filtering on Project values.

**Note:** Compose does not include Project when publishing activities to Foundation Hub.

- **Version:** Identifies the major and minor version of the activity. For new activities, the version is **1.0** by default, but you can change it.

The number that precedes the decimal point indicates the major release, and the number that follows it indicates the minor release. A minor release is an update to a major release. For example, 2.10 indicates the tenth update to version 2. For more information, see [Creating a New Version of an Approved Activity](#).

5. Click **OK** and then proceed to [Configuring an Activity](#).

### Configuring an Activity


Use the **Edit Activity** page to configure an activity for use:

- For a *protocol* activity, identify the protocol name and any required input and output parameters.
- For a *data acquisition* activity, identify the equipment and the output parameters you need to obtain from it.
- For a *procedure* activity, you might need to edit the input, output, and sample input parameters.
- For all activity types, if the collaborative space supports more than one lifecycle policy, you might also need to change the activity's lifecycle policy. The lifecycle policy for all activities, including those published from Compose, is initially set to the default specified for the collaborative space.

After you submit a *draft* activity, it changes to a read-only lifecycle state that is controlled by its lifecycle policy. You cannot edit the activity again unless a reviewer or approver rejects it, which changes it back to its *draft* state. If it is a data acquisition or protocol activity, users can generate tasks from it regardless of its state. However, if it is a procedure activity, it must be approved before anyone other than its owner can generate tasks for it.

To configure a child activity, see [Creating Parameterized Child Activities](#).

#### To configure an activity:

1. In **Admin and Settings > Resources > Activities**, click the activity.
2. On the toolbar, click **Edit** .
3. In the activity card, select the following:
  - **Templates:** For protocol activities, select a protocol template of usage type Activity Headless or Activity Report.
  - **Equipment Type/Class:** For data acquisition activities, identify the equipment that will collect the data.

**Note:** If the data must be entered manually, select **Manual Data Entry**. For many types of equipment, collection is automated. For more information, see the *Foundation Hub Equipment Guide*.

- **Is Non Destructive Test:** For procedure activities, select this check box if the samples can be used again after the procedure is completed.
- **Project:** For all activity types, identify the default project that tasks based on the activity should use. Project values can be useful for sorting and filtering.

**Note:** For procedure activities, enter a value that matches the value assigned in the Compose recipe. Project assignments in Compose recipes are not reflected in the corresponding activities.

- **Lock Activity's Settings in Tasks:** For all activity types, select this check box if users are not allowed to edit the task settings that their tasks inherit from this activity.

**Notes:**

This option also controls whether users can change the task-level **Use Latest Activity** setting, whose default is inherited from [collaborative space configuration](#).

4. (Optional) Update the remaining options on the activity card, if needed:

- **Life Cycle Policy:** [Lifecycle policy](#) that identifies the approval requirements for this activity. If the [collaborative space](#) supports more than one lifecycle policy, you can choose which one to use.

**Notes:**

You can change the lifecycle policy for a procedure activity only if the activity uses the Compose activity-driven workflow. All other types of Compose activities must use the default lifecycle specified in the collaboration space; neither the Compose author nor someone editing the published activity can change it.

Prior to Foundation Hub 2021, an activity's lifecycle policy was passed down to tasks, but tasks now inherit their lifecycle policy from the collaborative space.

- **Owner:** Initially, the person who creates the activity is both its author and its owner. You can specify a different owner at any time, but the author (creator) is stored separately and is always read-only. The author and owner have the same rights to the activity.

**Tip:** For procedure activities, the Owner is automatically set to *Compose Service Account*. Consider changing it to a person's account.

- **Method Type:** Used for sorting and filtering activities.

- **Version:** Indicates the major and minor release.

The number that precedes the decimal point indicates the major release, and the number that follows it indicates the minor release. A minor release is an update to a major release. For example, 2.10 indicates the tenth update to version 2. For more information, see [Creating a New Version of an Approved Activity](#).

- **Method ID and Method Version:** For data acquisition and protocol activities. Users who create tasks based on the activity can change these values in their tasks, unless you select **Lock Activity's Settings in Tasks**. Users can filter activities based on **Method ID** when they create tasks.

5. In the **Activity Parameters** table, add, edit, and remove the following, as needed:

- Input parameters that identify the materials and conditions for performing the task
- Output parameters that identify expected test results or material to be created as a result of performing the task
- Task-level parameters for passing executed data to Requestor

**Note:** When you edit the activity parameters table for a child activity, you can change the planned values, limits, precision, and formatting settings as needed for a specific product or material specification. However, you cannot change other properties of the parameters, and you cannot add or remove parameters. For more information, see [Creating Parameterized Child Activities](#).

- a. To add a parameter, click **Add** and then select the parameter template that identifies the parameter; to remove one, click it and then click **Remove**.



Add all parameters that are likely to be needed for tasks based on this activity. Users can exclude any that they do not need when they create tasks, unless you select **Lock Activity's Settings in Tasks**.

- b. Add or edit values for a parameter's properties by clicking the property cell and then typing or selecting the value. The Activity Parameters table provides a column for each of the following parameter properties, although some properties are read-only and others are relevant only for specific data types:
  - **Role:** Indicates whether the parameter provides *input* needed to perform the activity, or *output* obtained by performing the activity.
  - **Display Name:** Name used to identify the parameter to Task Planner and Capture users.  
If two parameters in the activity have the same name, it is important to edit them so that users can distinguish between them.  
You can also edit it for *procedure* activities, but your edits do *not* carry through to Capture when a user executes the procedure activity—Capture displays the original Compose Display Name identified in the recipe.
  - **Type:** Read-only indicator of the parameter value's data type—Sample, Text, LongText, Numeric, Integer, Date/Time, Quantity, Boolean, or Vocabulary.
  - **Sample Level:** Indicates whether the parameter's value is collected once for the activity as a whole (**False**), or collected each time each sample is subjected to the activity (**True**).
  - **Name:** Read-only name of the parameter. For parameters based on a Foundation Hub parameter template, this is the name defined in the template configuration.
  - **Display Order:** Order in which to display the parameter columns in the Task Planner results tab. If the activity is for manual data acquisition, this property also controls the column order in the manual result input grid and in downloaded CSV data entry files.
  - **Planned Value:** Default expected value for the parameter, if relevant.
    - If planned value and units were specified in Compose, you cannot change the unit type.
    - If you change units, values for lower and upper limits are converted to reflect your change.
  - **Lower Limit/Upper Limit:** Lowest /Highest acceptable value, if relevant.
  - **Unit:** Unit of measurement for the parameter, if its type is *quantity*.
  - **Equipment Reading Mapping:** For output parameters for data acquisition activities, identifies the equipment-specific parameter that maps to this activity parameter.  
The available values are based on the selected **Equipment Type/Class**. If *specific* equipment was selected, only values configured for that specific equipment can be selected. However, if *Manual Data Entry* was selected, any value for any equipment can be selected.
  - **Format Method:** For parameters of type *numeric* and *quantity*, identifies whether to display the **Raw** value, a **Decimal** value, or a **SignificantFigures** value.
  - **Rounding Rule and Reported Value Precision:** For parameters whose Format Method is Decimal or Significant Figures, controls how to adjust the raw value for reporting purposes.  
For examples of how Format Method, Rounding Rule, and Reported Value Precision work together, see [Rounding Rule Examples](#).
  - **Recorded Value Precision:** For output parameters of type *numeric* and *quantity* in data acquisition, protocol, and child procedure activities, indicates the number of significant digits

to record.

**Notes:**

- Prior to Foundation Hub 2021, recorded value precision was set implicitly based off the planned value.
- The precision impacts the how the Recorded Value column in Task Plan Activity Results is displayed. It is not used in Compose and Capture, in any limit checks, or on the Review page.

6. In the **Activity Samples** table, edit the following parameters for each sample, as needed. You cannot add sample rows, but you can delete them. For data acquisition activities, a single sample row is automatically shown. For procedure activities, one row is provided for each sample placeholder specified in the Compose procedure. The Sample Parameters table provides a column for each of the following:
  - **Display Name:** Name used to identify the sample parameter to Task Planner and Capture users.
  - **Role:** Read-only, always **Input**.
  - **Type:** Read-only, always **Text**.
  - **Name:** Read-only name of the sample parameter.
  - **Sample Level:** Read-only, always true.
  - **Sample Size:** Physical amount of sample required for the activity.
  - **Sample Size Unit:** Unit of measurement in which the sample size is expressed.
7. In the **Activity Costs** table, identify departments and expected turn-around times (TATs) for standard and expedited tasks, if needed.  
Procedure activities inherit these values from Compose, but you can change them.
8. On the toolbar, click **Save** .
9. When the configuration is complete, click the **Lifecycle Actions** menu  and choose **Submit** to start the approval process. For more information, see [Reviewing and Approving or Rejecting an Activity](#).

## Rounding Rule Examples

The following table provides rounding rule examples.

Format Method	Rounding Method	Reported Value Precision	Raw Value	Formatted Value
Decimal	Ceiling	1	3.1415	3.2
		2	3.1415	3.15
	Floor	2	3.1415	3.14
		3	3.1415	3.141
	HalfUp	2	3.1415	3.14
		3	3.1415	3.142
		3	1.8	1.800
Raw	HalfUp	3	1.8	1.8
		3	3.1415	3.1415
		3	1.8	1.8

Format Method	Rounding Method	Reported Value Precision	Raw Value	Formatted Value
SignificantFigures	Ceiling	1	3.1415	4
		2	3.1415	3.2
		3	3.1415	3.15
	Floor	1	3.1415	3
		3	3.1415	3.14
		4	3.1415	3.141
	HalfUp	3	3.1415	3.14
		4	3.1415	3.142
		3	1.8	1.80
	HalfUpNoPadding	3	1.8	1.8


## Creating Parameterized Child Activities

After an activity is approved, you can use it as a template to create child activities. You can edit each child activity to reflect a different product or material by updating the following elements, but you cannot add or change parameters:

- Activity Name, Description, and Version
- Project, Method Type Category, and Activity Cost settings
- Planned Value, Upper Limit, Lower Limit, and Recorded Value Precision settings for each parameter
- Life Cycle Policy

Child activities behave like other activities. You can combine them with regular activities and with other child and parent activities into activity plans, and Task Planner users can generate tasks for a combination of regular activities, child activities, and activity plans.

### To create a parameterized child activity:

1. Open **Admin and Settings > Resources > Activities**.
2. Select the check box of the template activity on which to base the child activity.
3. On the toolbar, click **Create Child Activity** .
4. Open the child activity, click **Edit**, and update the following properties, as needed:
  - Activity metadata: Name, Description, Project, Method Type, Category, Activity Costs
  - Parameter properties: Planned Value, Upper Limit, Lower Limit, and Recorded Value Precision

#### Notes:

- **Recorded Value Precision** applies only to output parameters of type *numeric* and *quantity*, and only to data acquisition, protocol, and child procedure activities. It indicates the number of significant digits to record.
- Prior to Foundation Hub 2021, Recorded Value Precision was set implicitly based off the Planned Value.

- Life Cycle Policy
  - Version
5. Test the child activity before you approve it to ensure that it works as expected.



- After verifying that the child works, repeat steps 2 and 3 to create each of the other child activities you need.

Alternatively, create the other child activities by [cloning](#) the one that you just created. When you clone a child activity, the new activity is associated with the same parent as the child that you cloned.

- Submit each child activity to start the [approval process](#). The approval process is the same as for regular activities.

Like other activities, you cannot edit an approved child activity, but you can [re-open it for edit](#), which results in a new version of the activity.


## Managing the Activity Lifecycle

Each activity identifies a [lifecycle policy](#). Settings in the [Publish](#) section of the policy control the following:

- Review and approval steps required to [get an activity approved](#)
- Signature policy for each required review and approval step
- Signature policy for [reopening an approved activity](#) to create an updated version
- Signature policy for [withdrawing an approved activity](#) so that it can no longer be used to create new tasks

## Reviewing and Approving or Rejecting an Activity

Perform this procedure as many times as required to advance the lifecycle state from *draft* to *approved*:

- Open **Admin and Settings > Resources > Activities**.
- Click the row of the activity.
- On the **Lifecycle Actions** menu , click the option required to advance the activity to its next state. Alternatively, click **Reject** to return it to its *draft* state for changes.

The following table lists the menu options for each possible review and approval step. Default installations exclude the Step 3 and Step 5 current states and menu options. A different person should perform each required step.

Step	Current State	Menu Option
1.	Draft	Submit
2.	Submitted	Mark as Reviewed (or Reject)
3.	Review in Progress	Mark as Reviewed (or Reject)
4.	Reviewed	Approve (or Reject)
5.	Approval in Progress	Approve (or Reject)

- Complete the signature policy window.

**Note:** If the updated **Life Cycle State** is unexpected, refresh the page. States that correspond to steps you skip are displayed temporarily after you submit the signature window, but the correct states are saved.

### Creating a New Version of an Approved Activity



If changes are required to an approved activity, use **Reopen for edit** to create a new version. This action clones the approved version to create a new *draft* version and sets the draft's minor version (number that follows the period) to its previous value, plus **1**. It does not change the major version (number that precedes the period).

For example, reopening version **1.9** creates a draft for version **1.10**, and reopening version **1.10** creates a draft for version **1.11**. You can manually increment the major version when necessary, as well as change the minor version to suit your version numbering conventions.

After the new version is *approved*, the old version is automatically *superseded* so that it cannot be used to create new tasks.

**Note:** You can use the Foundation Hub **Reopen for edit** activity lifecycle action to create new versions of procedure activities only if you created them using the Compose activity-driven workflow. To update activities based on other Compose workflows, use the **Reopen for edit** feature in Compose. For more information see the Compose and Capture help.

#### To create a new version of an approved activity:

1. Open **Resources > Activity**.
2. Open the activity.
3. On the **Lifecycle Actions**  menu, click **Reopen for edit**.
4. If required, complete the signature window.
5. In the breadcrumb trail, click **Activities**.
6. On the **Activities** page, find and open the new draft version of the activity.
7. [Edit the new version as needed](#).
8. On the **Lifecycle Actions**  menu, click **Submit** to start the [activity review and approval process](#) for your new version.
9. If the activity that you reverted had child activities, [recreate the child activities](#) and then [withdraw the previous versions](#). When you reopen and edit a parent activity, its child activities do **not** inherit your changes and they are **not** reverted.

#### Tips:

- To identify the child activities, apply two column filters to your **Activities** page. On the **Parent Activity** column, apply a filter that finds all children of the activity. Then apply an additional filter on the **Parent Activity Version** column to narrow the results to children of the previous version of your newly reverted parent activity.
- For assistance deploying a Pipeline Pilot protocol to automate this step, contact Dassault Systèmes Customer Support.

10. Update each activity plan that uses your reverted activity or any of its reverted child activities:
  - a. Open the activity plan.
  - b. If the activity plan is approved, use **Reopen for Edit** to generate a new draft version and then open the new draft. If it is in a submitted state, reject it to return it to its draft state.
  - c. Remove outdated activities by clicking the **Delete** icon in their title bars.
  - d. Add the updated activities by clicking **Add Activities** at the top of the page.
  - e. Submit the draft to start it through your approval process.

**Tip:** Draft tasks can be automatically updated after you create a new version of the activity on which they are based. Two settings control whether this is possible: **Lock Activity's Settings in Tasks** and **Use Latest Activity**. For more information, see the *Foundation Hub User Guide*.


## Withdrawing an Approved Activity from Use

When an approved activity is no longer needed, you can use the **Withdraw** lifecycle action to prevent users from creating new tasks based on that activity.

When you withdraw an activity, its lifecycle state transitions from *approved* to *withdrawn*.

Currently existing tasks that are based on the withdrawn activity continue to use it.

**To withdraw an approved activity:**

1. Open **Admin and Settings > Resources > Activities**.
2. Click the row of the activity.
3. Open the **Lifecycle Actions** menu  and click **Withdraw**.
4. Complete the signature window.



**IMPORTANT!** If the activity has any child activities, withdraw them. Withdrawing a parent activity does not withdraw its child activities. If you do not withdraw them, they remain available for use in task plans.

## Making an Activity Visible Across Collaborative Spaces

When you initially create an activity, it is unshared. To access it, you must be working in the collaborative space in which you created it (its home space). To make your activity accessible from other spaces, you can share it. After you share it, you can access it from any of your collaborative spaces. So can other users, if they have the right to work with it and also have a role of reader or higher in its home space.

- To share an activity in a public collaborative space, you must be its *creator*, current *owner*, or have a role of *leader* or higher.
- To share an activity in a protected collaborative space, you must have a role of *leader* or higher.
- You cannot share an activity that resides in a private collaborative space.

**To share an activity across collaborative spaces:**

1. Open the activity.
2. On the activity title bar, click the  **Share** icon. The icon changes to  **Unshare**.  
If you need to return it to unshared, you can click the icon again.

## Moving an Activity to Another Collaborative Space

You can move an activity from one collaborative space to another if you are the creator or current owner of the activity *and* you have at least an author role for both collaborative spaces.

### Notes:

- Moving an entity can result in broken links between that entity and other entities for users who do not have access to the entity's new **and** previous collaborative space.
- If you move an activity to a private collaborative space and any draft tasks based on that activity require the use of the latest activity, Task Planner will be unable to find the activity or update the tasks. When it attempts to do so, it will display a message that informs the user of the situation.

### To move an activity:

1. On the activity's card, click the **Move to a Different Collaborative Space** icon.




2. From the **Select Collaborative Space** field, choose the destination collaborative space.
3. Click **Move**.
4. If appropriate, [share the activity](#) so that it can be accessed by authorized users from other collaborative spaces.

## Cloning an Activity

When you need a new activity that is similar to an existing one, you can clone the existing one and then edit your clone.

- You can clone regular activities, parent activities, and child activities.
- You can clone any data acquisition or protocol activity.
- To clone procedure activities, use the Compose cloning feature, not Foundation Hub. You can use the link in the **Templates** field on the Activity page to open the corresponding Compose recipe so that you can clone it. For more information about cloning Compose recipes, see the Compose and Capture help.

### To clone an activity:

1. Open **Admin and Settings > Resources > Activities**.
2. Select the check box for the activity to clone.
3. Click the **Clone Selected**  icon.
4. In the **Confirm Clone** window, click **Yes**.
5. [Configure the new version as needed](#), and then submit your changes to start the review and approval steps.

## Deleting an Activity

1. Open **Admin and Settings > Resources > Activities**.
2. Select the check box of each activity you need to delete.
3. Click the **Delete Selected** (minus) icon.
4. In the **Confirm Delete** dialog box, click **Yes**.

## Activity Plans

Foundation Hub distinguishes between two types of activity plans, *standard activity plan* and *material specification*.

**Note:** All fields and options except **Reference** are the same for both types of plans. Material specifications identify QC activities required to test and certify a specific lot of material against a specification that you identify in a **Reference** field. This field is added to the page only if you set the activity plan **Type** field to **Material Specification**.

All activity plans consist of one or more individual lab activities and provide details about the input and output parameters for each activity. For example, an activity might specify *color* and *density* as parameters to collect, but after you add that activity to an activity plan, you can identify a *specific* color of interest and a *passing range* of density values for that particular plan.


When an activity plan contains more than one activity, you can also specify the order in which the activities must be performed and control whether each activity requires approval.

When a Task Planner user chooses an activity plan for which to generate tasks, the system creates one task for each activity in the activity plan. The user can then remove tasks for any unnecessary activities and adjust or remove parameters for the remaining tasks as needed.

**Tip:** To minimize the number of activity plans you need, it is a good practice to include all activities that can be required, keeping in mind that Task Planner users can easily remove tasks that they do not need without affecting the base activity plan in Foundation Hub.

## Adding an Activity Plan

Perform the following steps to create an activity plan that identifies a set of activities:

1. Switch to the collaborative space in which to store the activity plan. For more information, see [Managing Collaborative Spaces](#).
2. Open **Admin and Settings > Resources > Activity Plans**.
3. Click **Add Activity Plan** .
4. Enter general information about the activity plan in the following fields:
  - **Name:** Enter a recognizable name.
  - **Type:** Select **Activity Plan**, unless this activity plan is for a material lot workflow that involves BIOVIA CISPro, in which case select **Material Specification**. This selection causes the system to add a **Reference** field to the form.
  - **Authoring Group:** Select the group whose members are authorized to edit, review, and approve this activity plan.

**IMPORTANT!** Select a group to which you belong. Otherwise, you cannot continue.

- **Reference:** If the **Type** is **Material Specification**, enter the CISPro material reference.
- **ID:** If appropriate, enter your own identifier for the activity plan. By default, the system generates a unique identifier *after* you submit the plan. If you manually enter an ID, the system ensures that your entry is unique.

- **Sub Type:** If appropriate, select a subtype. The system includes three predefined Sub Types, *qualification*, *preparation*, and *characterization*. Any additional values come from your Vocabularies configuration. Sub types are useful for sorting and filtering.
- **External ID:** If applicable, enter the ID that an external application uses for this activity plan.
- **Phase:** Select the developmental stage for which this activity plan is relevant. The system includes three predefined values: *pre-pivotal*, *pivotal*, *clinical*. Phases are useful for sorting and filtering tasks.
- **Version:** If needed, change the version, which is **1.0** for new plans.  
The number that precedes the decimal point indicates the major version, and the number that follows it indicates the minor version. A minor version is an update to a major version. For example, 2.10 indicates the tenth update to version 2. For more information, see [Creating a New Version of an Approved Activity Plan](#).

5. Click **OK** to save the plan. You are returned to the **Activity Plans** list.
6. Click your new activity plan to open it and determine if you need to edit the values of the following fields, which are not available on the **Add Activity** form:
  - **Owner:** Initially, the person who creates the plan is both its author and its owner. You can specify a different owner at any time, but the author (creator) is stored separately and is always read-only. The author and owner have the same rights to the plan.
  - **Life Cycle Policy:** [Lifecycle policy](#) that identifies the appropriate approval requirements for this activity plan. The default value and any other available values are specified in the [collaborative space configuration](#). If the configuration allows more than one lifecycle policy, you can select which one to use.
7. To edit values on the activity card, click **Edit** on the card's toolbar, enter your changes, and then click **Submit** on the toolbar.

**Note:** If you have configured [extended properties](#) for activity plans, they appear at the bottom of the card as read-only fields. To provide or edit values for them, you must use an API application.

8. Click **Add Activities** and select the activities to include in the plan:
  - Use the Filters panel and column sorting feature as needed to find activities.
  - To add multiple instances of the same activity or selected set of activities:
    - a. Select the **Enable Duplicate Selection** check box.
    - b. Select the activities from the top panel.
    - c. Click the panel's **Add (+)** icon once for each instance you require. For example, to add three instances of the same selection of activities, click the icon three times.
  - To add a single instance of one or more activities, select their check boxes, then click **OK**.
9. [Configure each activity](#) to meet the needs of this activity plan.

Changes you make in the activity or activity plan configuration are passed on to tasks, and changes made during task configuration in Task Planner are passed on to Capture when a task is executed.


## Configuring Activities in an Activity Plan

After you add an activity to an activity plan, you can add, override, and clone the settings to meet the needs of the activity plan. This is true for all types of activities, including those based on Compose recipe templates.

Changes to the base activity or to an activity plan's copy of an activity are passed on to tasks. Task Planner users can change them at the task level, unless you select **Lock Activity's Settings in Tasks** when configuring the activity.

**To configure an activity in an activity plan:**

**Notes:** Settings and properties marked **‡** are not displayed or used in the Task Planner or Capture interfaces. You can, however, develop applications that use these properties through the API.

1. Open the activity plan and find the activity card.
2. Expand the activity card, and then click **Edit**  in its title bar.
3. In the information panel, add or update these settings as needed:
  - **Fixed Sample Count:** If this activity must be performed on more than one than one sample, enter the number of samples required. Defaults to **1**, but you can change it to **0** for tasks that do not involve samples.
  - **‡Automatic Aliquot:** If samples on which this activity is performed should be aliquoted automatically, select this check box.
  - **Sample Size and Sample Size Unit:** If needed, add values or override values inherited from the activity.
  - **‡Approval Period, Approval Period Unit:** If this activity plan should be periodically reviewed and approved, type a number in Approval Period and select time period from Approval Period Unit. Defaults are **1** and **year** to indicate an annual approval cycle.
  - **‡Required for Approval:** Indicates whether characteristics identified in this activity must be within limits in order for a corresponding Certificate of Analysis (COA) to be approved.
  - **Predecessor:** If another activity in this activity plan must precede this activity, select the name of the other activity.

**Note:** Your global Foundation Hub [Application Setting for Predecessor Status](#) controls whether predecessors must be *completed* or must be *started* before their successors can start. The default is setting is **Completed**.

- **Description:** If appropriate, override the default description.
4. If the **Input Conditions** table lists parameters, you can adjust the following properties:

**Tip:** To make some cells editable, you must click them twice.

- **Planned Value:** Value that is expected to be used when performing the activity.

**IMPORTANT!** If **Material Name** appears as a **Parameter Name**, enter the name of the required material in the **Planned Value** column. Compose master recipes use **Material Name** as a proxy placeholder.

- **Lower Limit and Upper Limit:** Lowest and highest value that can be used to perform the activity.
- **Unit:** Units used for the values.

**Note:** If you change the units, the system recalculates the values.

- **Reported Value Precision:** Precision with which to report the value. For activities published from Compose, the affect of this setting depends on the parameter's **Display Method** setting in Compose.

For more information about rounding methods and rules, see the *Compose and Capture Help*.

- **Recorded Value Precision:** For output parameters of type *numeric* and *quantity* in data acquisition, protocol, and child procedure activities, indicates the number of significant digits to record.

**Notes:**

- Prior to Foundation Hub 2021, recorded value precision was set implicitly based off the planned value.
- The precision impacts the how the Recorded Value column in Task Plan Activity Results is displayed. It is not used in Compose and Capture, in any limit checks, or on the Review page.

5. If the **Qualitative Characteristics** table lists parameters, update, clone, and remove them as needed.

**Note:** Qualitative, task-level characteristics of an activity are *not* displayed or accessible from activity plans. Their planned values and other attributes are managed at the individual task level, when tasks are generated off either the activity or the activity plan.

- a. To change the set of parameters, use the **Clone** and **Remove** icons.
- b. To update a parameter property value, click in its cell. You might have to click twice, depending on value type.
  - **†Component:** For clones, system-generated name that enables users to distinguish a cloned parameter from the parameter from which it was cloned. You can enter a component value for a parameter that does not have one, as well as override system-generated values.
  - **Approved Values:** Passing values.
  - **†Allow Qualified C of A:** Indicates whether the parameter *can* be reported on a *qualified* Certificate of Analysis (CoA).
  - **†Report Only:** Indicates whether the parameter can be included in other reports, but is not for a qualified CoA.
  - **Display Order:** Indicates the order in which to display this parameter relative to the other parameters in Task Planner and, for procedure tasks, in Capture.

6. If the activity's **Quantitative Characteristics** table lists any parameters, update, clone, and remove them as needed:
  - a. Use the parameter **Clone** and **Remove** icons as needed to refine the set of quantitative parameters available for tasks based on this activity. Task Planner lists all parameters you select, but allows its users to remove parameters that are unnecessary when they generate tasks.
  - b. Update inherited values for the following properties of each parameter as needed. Task Planner users can override the values that you set here if necessary.

**Tip:** You must click most cells twice in order to add or edit content. Cells populated by vocabulary entries are an exception.





- **‡Component:** For clones, system-generated name that enables users to distinguish a cloned parameter from the parameter from which it was cloned. You can enter a component value for a parameter that does not have one, as well as override system-generated values.
  - **Unit:** Units to use for all values collected for quantity parameters, including raw, recorded, reported, and limit values. If you change a Unit value, the system recalculates associated quantity values accordingly.
  - **Target/CL:** Ideal output value.
  - **Reported Value Precision:** Precision to use for reported values. In Capture, planned values are identified to users above the data entry fields. The planned values are displayed with the precision specified in the activity plan. For quantity parameters, the planned values also indicate the units specified in the activity plan.
  - **Lower Limit / Upper Limit:** Limits within which the recorded output value must fall to be considered a passing value. Recorded values that are *not* with these limits are considered failing values and are flagged and reported.
  - **LoQ:** Number that identifies the threshold for detecting or quantifying the value.
  - **LoQ Display:** Expression that starts with a mathematical operator that indicates whether the threshold indicated in **LoQ** is the *highest* or the *lowest* quantifiable or detectable value.
    - <LOQ - the default, which indicates that anything beneath the LOQ value cannot be detected.
    - >LOQ - indicates that anything above the LoQ value cannot be detected.
    - <=LOQ - indicates that anything less than or equal to the LoQ value cannot be detected.
    - >=LOQ - indicates that anything equal to or greater than the LoQ value cannot be detected.
  - **‡Report Only:** Indicates whether to include the parameter and its value in reports.
7. When you finish editing the activity, click the **Submit** icon on its toolbar.
  8. If the entire activity plan configuration is now complete, click the Lifecycle Actions menu on the activity plan card and choose **Submit**.

## Editing Activity Plans

You can add and change general information about a *draft* activity plan, as well as add, remove, and reconfigure its lineup of activities. After you approve an activity plan, you cannot edit it. You must instead [create a new version of the plan](#).




### To edit an activity plan:

1. Open **Admin and Settings > Resources > Activity Plans**.
2. Click the row of the activity plan to edit.
3. To edit or add general settings, click **Edit**  in the title bar of the activity plan card, enter your changes, and then click **Submit** .

**Note:** The only fields on this page that are not on the Add Activity page are **Owner** and **Life Cycle Policy**. The [Lifecycle policy](#) identifies the approval requirements for the activity plan. The default value and any other available values come from the [collaborative space configuration](#). If the configuration allows more than one lifecycle policy, you can select which one to use.

4. To add activities, click **Add Activities** above the activity plan card, and then select the activities to

add.

5. To [configure the settings](#) for an activity in the plan, click the **Edit** icon  in its title bar.
6. To remove an activity from the plan, click the **Delete** icon in its title bar.
7. To save your work, click the **Submit** icon  on the toolbar.
8. After the activity plan is complete, click the **Lifecycle Actions** menu  and choose **Submit** to start the [review and approval process](#).


### Managing the Activity Plan Lifecycle

Each activity plan identifies a [lifecycle policy](#). Settings in the [Specification](#) section of the policy control the following:

- Review and approval steps required to [get an activity plan approved](#)
- Signature policy for each required review and approval step
- Signature policy for [reopening an approved activity](#) to create an updated version
- Signature policy for [withdrawing an approved activity plan](#) so that it can no longer be used to create new tasks

### Reviewing and Approving or Rejecting an Activity Plan

Perform this procedure as many times as required to advance the lifecycle state from *draft* to *approved*:

1. Open **Admin and Settings > Resources > Activity Plans**.
2. Click the row of the activity plan.
3. From the activity's **Lifecycle Actions** menu , select the option required to advance the activity plan to its next state. Alternatively, click **Reject** to return it to its *draft* state for changes.

The following table lists the menu options for each possible review and approval step. Default installations exclude the Step 3 and Step 5 current states and menu options. A different person should perform each required step.

Step	Current State	Menu Option
1.	Draft	Submit
2.	Submitted	Mark as Reviewed (or Reject)
3.	Review in Progress	Mark as Reviewed (or Reject)
4.	Reviewed	Approve (or Reject)
5.	Approval in Progress	Approve (or Reject)

4. Complete the signature policy window.

**Note:** If the updated **Life Cycle State** is unexpected, refresh the page. States that correspond to steps you skip are displayed temporarily after you submit the signature window, but the correct states are saved.



## Creating a New Version of an Approved Activity Plan

If changes are required to an approved activity plan, use **Reopen for edit** to create a new version. This action clones the approved version to create a new *draft* version and sets the draft's minor version (number that follows the period) to its previous value, plus **1**. It does not change the major version (number that precedes the period).

For example, reopening version **1.9** creates a draft for version **1.10**, and reopening version **1.10** creates a draft for version **1.11**. You can manually increment the major version when necessary, as well as change the minor version to suit your version numbering conventions.

After the new version is *approved*, the old version becomes *superseded* and is no longer available for creating new tasks.


**To create a new version of an approved activity plan:**

1. Open **Resources > Activity Plan**.
2. Open the activity plan.
3. On the **Lifecycle Actions**  menu, click **Reopen for edit**.
4. Complete the signature window.
5. In the breadcrumbs trail, click **Activity Plans**.
6. On the **Activity Plans** page, find and open the new draft version of the activity plan.
7. [Edit the new version](#) as needed.
8. On the **Lifecycle Actions**  menu, click **Submit** to start the [activity plan review and approval process](#) for your new version.

## Withdrawing an Approved Activity Plan from Use

When you no longer want users to be able to select an approved activity plan, you can withdraw it. When you withdraw it, its lifecycle state changes from *approved* to *withdrawn*.

**To withdraw an approved activity plan:**



1. Open **Admin and Settings > Resources > Activity Plans**.
2. Click the row of the activity plan.
3. On the **Lifecycle Actions** menu , click **Withdraw**.

### Making an Activity Plan Visible Across Collaborative Spaces

When you initially create an activity plan, it is unshared. To access it, you must be working in the collaborative space in which you created it (its home space). To make your activity plan accessible from other spaces, you can share it. After you share it, you can access it from any of your collaborative spaces. So can other users, if they have the right to work with it and also have a role of reader or higher in its home space.

- To share an activity plan in a public collaborative space, you must be its *creator*, current *owner*, or have a role of *leader* or higher.
- To share an activity plan in a protected collaborative space, you must have a role of *leader* or higher.
- You cannot share an activity plan that resides in a private collaborative space.

#### To share an activity plan across collaborative spaces:

1. Open the activity plan.
2. On the activity plan title bar, click the  **Share** icon. The icon changes to  **Unshare**.  
If you need to return it to unshared, you can click the icon again.

### Moving an Activity Plan to Another Collaborative Space

You can move an activity plan from one collaborative space to another if you are the plan's author, owner, or a member of its designated authoring group *and* you have at least an author role in both collaborative spaces.

**Note:** Moving entities can result in broken links for users who do not have access to both spaces. For example, if a user can access the target space but cannot access the source space, that user cannot open any links from the target space that go to entities in the source space.

#### To move an activity plan:

1. On the plan's card, click the **Move to a Different Collaborative Space** icon.




2. From the **Select Collaborative Space** field, choose the destination collaborative space.
3. Click **Move**.

**Note:** If you move an activity plan, be aware that the stand-alone versions of the activities used in the plan are **not** automatically moved with the activity plan.

## Cloning an Activity Plan

It is often more efficient to clone an existing activity plan than to start from scratch. Maintaining "template" activity plans specifically for such purposes is a best practice. For more information, see [Templating Techniques](#).




### To clone an activity plan:

1. From the **Admin and Settings** page, click **Activity Plans**.
2. Select the check box for the activity plan that you want to clone.
3. Click the **Clone Selected** icon .
4. In the **Confirm Clone** dialog box, click **Yes**.

A clone of the activity plan is created and shown in the Activity Plan view.

By default, the name of the new activity plan is that of the source activity plan, with a numeric suffix to make it unique. For example, if you clone the activity plan Make Cake, the first cloned activity plan is called Make Cake(2), the second Make Cake(3), and so on.

When you clone an activity plan, related activities, activity conditions, and characteristics are cloned with it.

5. In the Activity Plan view, click the **Edit** button .
6. Edit the details of the cloned activity plan. For information about the fields, see [Adding an Activity Plan](#).
7. Click the **Submit** button .
- Click **Cancel**  to discard the changes.
8. Click **Add Activities**, and follow the instructions in [Configuring Activities in an Activity Plan](#).

## Deleting an Activity Plan

1. Open **Admin and Settings > Resources > Activity Plans**.
2. Select the check box of each activity plan to delete.
3. Click the **Delete Selected** (minus) icon.
4. In the **Confirm Delete** dialog box, click **Yes**.

# Chapter 6:

## Pipeline Pilot Protocol Configuration


---

The **Resources > Pipeline Pilot Protocols** page allows you to manage Pipeline Pilot protocols used by Foundation Hub for adding custom activities, implementing custom lifecycle actions, generating reports from the Task Planner **Results** tab, and responding to task lifecycle changes.

Foundation Hub protocols can read and write data using the Foundation Hub REST API. For more information, see the *Foundation Hub API Reference Guide*.

For details of how each type of protocol works, see [Protocol Types](#).

### Adding a Pipeline Pilot Protocol

1. Write the protocol, and save it in Pipeline Pilot Server's protocols folder. For guidelines on writing Foundation Hub protocols, see [Protocol Types](#) and the subsequent topics.
2. Open **Admin and Settings > Resources > Pipeline Pilot Protocols**.
3. Click the **Add Protocol** button .
4. Complete the following fields:
  - **Name:** This name appears wherever the protocol is exposed in the user interface. For example, it identifies the protocol in the **Templates** list when you edit an Activity. (See [Configuring an Activity](#).)
  - **Application:** Name of the Pipeline Pilot server that hosts the protocol. Select the server from the list.
  - **Path:** Path of the Pipeline Pilot protocol. The path must begin with the string `/Protocols`. The initial slash is mandatory.

Example:

`/Protocols/Examples/Integration/Integrating  
Applications/Hub/Activities/Foundation pH Activity`

- **Usage Type:**

Type	Description
Activity Headless	Implements a protocol-based Activity that creates task results without generating an intermediate report.
Activity Report	Implements a protocol-based Activity that generates a report for tasks. The report can provide the means for users to select or edit data before task results are generated.
Lifecycle Action Headless	Implements a custom lifecycle action without generating a report. The user triggers the life cycle action by clicking a button in the relevant page of the Foundation Hub user interface (for example, the Task Planner).

Type	Description
Lifecycle Action Report	Implements a custom lifecycle action that generates a report. The user triggers the lifecycle action by clicking a button in the relevant page of the Foundation Hub user interface (for example, the Task Planner).
Task Planner Results	Adds a custom button to the <b>Results</b> tab in Task Planner that generates a report.
Lifecycle Subscriber	Performs a custom action in response to Task lifecycle changes. The lifecycle changes that trigger running of the protocol are specified in the <b>Subscriber JSON</b> field.

- **Life Cycle Entity:** Entity to which the life cycle change applies. This option applies only to Lifecycle Action protocol types.  
The value must be a valid Foundation Hub entity, for example Task for a task, or Runset for a Task Plan.
- **Life Cycle Action:** The label of a custom lifecycle button. This option applies only to Lifecycle Action protocol types.
- **Subscriber JSON:** JSON definition of the lifecycle changes that trigger running of the protocol. This option only applies to Lifecycle Subscriber protocol types. For details of the Subscriber JSON format, see [Lifecycle Subscriber Protocols](#).

**Note:** For guidelines on configuring different protocol types, see [Protocol Types](#) and the subsequent topics.

5. Click **OK**.

## Deleting a Pipeline Pilot Protocol

1. Open **Admin and Settings > Resources > Pipeline Pilot Protocols**.
2. Select the check box for the **Pipeline Pilot Protocol** that you want to delete. You can select multiple protocols.
3. Click the **Delete Selected** (minus) icon.
4. In the **Confirm Delete** dialog box, click **Yes**.

## Protocol Types

There are four general categories of Foundation Hub protocols:

- [Activity Protocols](#)
- [Lifecycle Action protocols](#)
- [Task Planner Results protocols](#)
- [Lifecycle Subscriber protocols](#)

Activity and Lifecycle Action protocols are further divided into two subcategories:

- **Headless protocols** operate in the background, without displaying a user interface.
- **Report protocols** generate reports. These reports can be interactive, enabling the user to select items

for further processing.

In total, then, there are six protocol types:

- Activity Headless
- Activity Report
- Lifecycle Action Headless
- Lifecycle Action Report
- Task Planner Results
- Lifecycle Subscriber

Example protocols are provided of the types Activity Headless, Activity Report, Task Planner Results, and Lifecycle Subscriber.

## Activity Protocols

**Subcategories:** Activity Headless, Activity Report

An Activity protocol implements a Foundation Hub Activity. The protocol is run when a user executes a task (or tasks) in the Task Planner or the **My Tasks** page. The Activity on which the task is based must have an **Activity Type** of Protocol.

### Activity Headless Protocols: Requirements

An Activity Headless protocol should be configured as described below.

Item	Description
Input Parameters	<ul style="list-style-type: none"><li>■ <i>Task ID.</i> This is passed to the protocol by Foundation Hub, and is accessible within the protocol as a global parameter.</li></ul>
Output File	N/A
Actions Performed	<p>An Activity Headless protocol should generate task results by writing values to children of the taskparameters REST endpoint. For example, if the protocol stores a task parameter in the <b>@parameter_id</b> global property, it should send a PUT request to the following endpoint: <code>api/v1/taskparameters/\${parameter_id}</code></p> <p>The body of the PUT request should contain the task parameter value in JSON format. For more information, see the <i>Foundation Hub API Reference Guide</i>.</p>
Results	<p>The protocol must generate a response property as a JSON string of the following form: <code>{JobResponseCode: 200, errorMessage: '', Results: ''}</code></p>

### Activity Headless Protocols: Foundation Hub Configuration

To set up an Activity Headless protocol in Foundation Hub:

1. Create an Activity Headless protocol, conforming to the description in [Activity Headless Protocols: Requirements](#).



2. In **Admin and Settings > Resources > Pipeline Pilot Protocols**, create a protocol entry that includes the following settings:
  - **Usage Type:** Activity Headless
  - **Path:** The path of the Pipeline Pilot protocol that you created in Step 1.

**Note:** For complete instructions for adding protocols, see Foundation Hub, see [Adding a Pipeline Pilot Protocol](#).

3. In **Admin and Settings > Resources > Activities**, create an activity that includes the following setting:

- **Activity Type:** Protocol

For full instructions on creating an activity, see [Adding an Activity](#).

4. Edit the activity that you created in Step 3, adding the following settings:

- **Templates:** The protocol that you added to Foundation Hub in Step 2. Select the protocol from the list.
- **Activity Parameters:** The activity parameters that are processed and generated by the protocol. These should include the following:
  - **Sample.** This input parameter is created by default in protocol activities.
  - **Output parameters.** These must match the type of the task parameters that are set by the protocol. (See [Activity Headless Protocols: Requirements](#).) For example, if the protocol sets a task parameter with a pH value, the activity must have an output parameter of type pH.

For instructions for editing an activity, see [Editing Activity Plans](#).

## Activity Headless Protocol: Example

An example Activity Headless protocol is supplied with the Foundation Hub Component Collection at the following location:

*/Protocols/Examples/Integration/Integrating Applications/Hub/Activities/Foundation pH Activity*

This protocol generates a random pH value for each task. This simulates manual or machine-generated entry of pH numbers in a laboratory.

When a user executes a task that uses *Foundation pH Activity*, a result is generated with the random pH number as its value:

Results											
											View: Default
Sample ID	Sample Name	Task Id	Activity	Method Id	Project	Task Status	Characteristic	Attribute	Result	Formatted Result	Interpretation
S000005	p-5	T010	Headless Activity			Completed	pH		8.97	8.97	Valid

## Activity Report Protocols: Requirements

Item	Description
Input Parameters	<ul style="list-style-type: none"> <li>■ <b>Task ID.</b> This is passed to the protocol by Foundation Hub, and is accessible within the protocol as a global parameter.</li> </ul>
Output File	<code>\$(jobdir)/index.html</code>
Actions Performed	<p>An Activity Report protocol should generate a Pipeline Pilot HTML report. The report can be static or interactive. If the report is static, the user must perform a simple action to complete the tasks and generate results (for example, pressing a button). If the report is interactive, the user can select or edit items in the report before sending the task for completion.</p> <p>For both static and interactive reports, completion of the task is typically carried out by a separate <i>worker protocol</i> that is called when the user submits the HTML report form. If you use the <i>Input Form</i> component to generate the report form, supply the path of the worker protocol in the <i>Work Protocol</i> parameter.</p> <p>The worker protocol should generate the task results by writing values to children of the <code>taskparameters</code> REST endpoint. For example, if the protocol stores a task parameter in the <code>@parameter_id</code> global property, it should send a PUT request to the following endpoint:  <code>api/v1/taskparameters/\$(parameter_id)</code>  The body of the PUT request should contain the task parameter value in JSON format. For more information, see the <i>Foundation Hub API Reference Guide</i>.</p>
Results	<p>The protocol must generate a response property as a JSON string of the following form:</p> <pre>{JobResponseCode: 200, errorMessage: '', Results: ''}</pre>

## Activity Report Protocols: Foundation Hub Configuration

To set up an Activity Report protocol in Foundation Hub:

1. Create an Activity Report protocol, conforming to the description in [Activity Report Protocols: Requirements](#).
2. Create a worker protocol that processes the data submitted from the activity report. The worker protocol is invoked when the user submits the report form.
3. In **Admin and Settings > Resources > Pipeline Pilot Protocols**, create a protocol entry that includes the following settings:
  - **Usage Type:** Activity Report
  - **Path:** The path of the Pipeline Pilot protocol that you created in Step 1.

**Note:** For complete instructions for adding protocols, see Foundation Hub, see [Adding a Pipeline Pilot Protocol](#).

4. In **Admin and Settings > Resources > Activities**, create an activity that includes the following setting:
  - **Activity Type:** Protocol

For full instructions on creating an activity, see [Adding an Activity](#).

5. Edit the activity that you created in Step 4, adding the following settings:

- **Templates:** The protocol that you added to Foundation Hub in Step 2. Select the protocol from the list.
- **Activity Parameters:** The activity parameters that are processed and generated by the protocol. These should include the following:
  - **Sample.** This input parameter is created by default in protocol activities.
  - Output parameters. These must match the type of the task parameters that are set by the protocol. (See [Activity Headless Protocols: Requirements](#).) For example, if the protocol sets a task parameter with a pH value, the activity must have an output parameter of type pH.

For instructions for editing an activity, see [Editing Activity Plans](#).

## Activity Report Protocol: Example

An example Activity Report protocol is supplied with the Foundation Hub Component Collection at the following location:

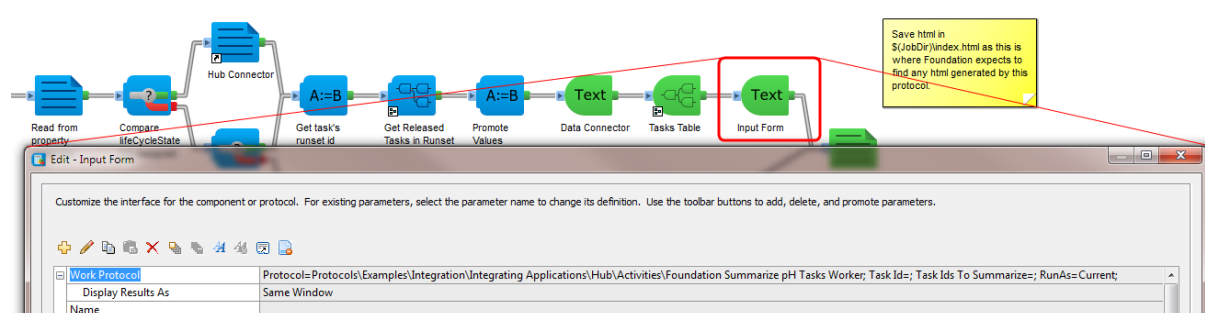
*/Protocols/Examples/Integration/Integrating Applications/Hub/Activities/Foundation Summarize pH Tasks*

There is an associated worker protocol:

*/Protocols/Examples/Integration/Integrating Applications/Hub/Activities/Foundation Summarize pH Tasks Worker*

The *Foundation Summarize pH Tasks* protocol generates a report that lists all the released tasks from the same Task Plan as the source task. Included in the list are the recorded pH values. The user selects tasks from the list, and submits the form. The worker protocol calculates the pH value of the source task from the mean of the pH values of the selected tasks. The new pH value is added to the Results tab of the Task Plan.

Note how *Foundation Summarize pH Tasks*, the worker protocol is invoked by the *Work Protocol* parameter of the *Input Form* component:



## Lifecycle Action Protocols

**Subcategories:** Lifecycle Action Headless, Lifecycle Action Report

You can create custom lifecycle action protocols for entities and Foundation Hub will display buttons to execute them on the relevant pages. For example, if you create a lifecycle action protocol for tasks plans, Foundation Hub adds a button to execute it to the **My Task Plans** list. Similarly, if you can create a lifecycle action protocol for tasks, a button to execute it appears on the **Tasks** tab of Task Planner, on the **My Tasks** widget, and on other widgets that list tasks.

**Note:** The Foundation Hub Component Collection does not include example lifecycle action protocols.

### Lifecycle Action Headless Protocols: Requirements

Configure a Lifecycle Action Headless protocol as described below.

Item	Description
Input Parameters	<ul style="list-style-type: none"><li>■ ID of the entity to be processed. The type of the entity is determined by the protocol's <b>Life Cycle Entity</b> field in Foundation Hub (for example, Task for a task, or Runset for a Task Plan).</li></ul> <p>The entity ID is passed to the protocol by Foundation Hub. The protocol should have an input parameter that captures the entity ID.</p>
Output File	N/A
Actions Performed	A Lifecycle Action Headless protocol can perform any action on the entity whose ID is supplied. The action would typically involve a change of the entity's lifecycle state, but this is not required.
Results	The protocol must generate a response property as a JSON string of the following form: <code>{JobResponseCode: 200, errorMessage: '', Results: ''}</code>

### Lifecycle Action Headless Protocols: Foundation Hub Configuration

To set up a Lifecycle Action Headless protocol in Foundation Hub:

1. Create a Lifecycle Action Headless protocol, conforming to the description in [Lifecycle Action Headless Protocols: Requirements](#).
2. In **Admin and Settings > Resources > Pipeline Pilot Protocols**, create a protocol entry that includes the following settings:
  - **Usage Type:** Lifecycle Action Headless
  - **Path:** The path of the Pipeline Pilot protocol that you created in Step 1.
  - **Life Cycle Entity:** The entity to which the lifecycle change applies.  
The value must be a valid Foundation Hub entity, for example Task for a task, or Runset for a Task Plan.
  - **Life Cycle Action:** The label of the custom lifecycle button.

**Note:** For complete instructions for adding protocols, see Foundation Hub, see [Adding a Pipeline Pilot Protocol](#).

A button to execute your custom lifecycle action is automatically added to each Foundation Hub page that lists entities to which it applies.

## Lifecycle Action Report Protocols: Requirements

A Lifecycle Action Report protocol should be configured as described below.

Item	Description
Input Parameters	<ul style="list-style-type: none"> <li>■ ID of the entity to be processed. The type of the entity is determined by the protocol's <b>Life Cycle Entity</b> field in Foundation Hub (for example, Task for a task, or Runset for a Task Plan).</li> </ul> <p>The entity ID is passed to the protocol by Foundation Hub. The protocol should have an input parameter that captures the entity ID.</p>
Output File	<code>\$(jobdir)/index.html</code>
Actions Performed	<p>A Lifecycle Action Report protocol can perform any action on the entity whose ID is supplied. The action would typically involve a change of the entity's lifecycle state, but this is not required.</p> <p>A Lifecycle Action Report protocol should generate an HTML report. This can be a static web page, or it can include a form with which the user interacts to trigger processing by a separate worker protocol. If you use the <i>Input Form</i> component to generate the report form, supply the path of the worker protocol in the <i>Work Protocol</i> parameter.</p>
Results	<p>The protocol must generate a response property as a JSON string of the following form:</p> <pre>{JobResponseCode: 200, errorMessage: '', Results: ''}</pre>

## Lifecycle Action Report Protocols: Foundation Hub Configuration

To set up a Lifecycle Action Report protocol in Foundation Hub:

1. Create a Lifecycle Action Report protocol, conforming to the description in [Lifecycle Action Report Protocols: Requirements](#).
2. If appropriate, create a worker protocol that processes the data submitted from the lifecycle action report. The worker protocol is invoked when the user submits the report form.
3. In **Admin and Settings > Resources > Pipeline Pilot Protocols**, create a protocol entry that includes the following settings:
  - **Usage Type:** Lifecycle Action Report
  - **Path:** The path of the Pipeline Pilot protocol that you created in Step 1.
  - **Life Cycle Entity:** The entity to which the lifecycle change applies.  
The value must be a valid Foundation Hub entity, for example Task for a task, or Runset for a Task Plan.
  - **Life Cycle Action:** The label of the custom lifecycle button.

**Note:** For complete instructions for adding protocols, see Foundation Hub, see [Adding a Pipeline Pilot Protocol](#).

A button that performs the custom lifecycle action will appear in the relevant Foundation Hub pages (for example, those with task lists or task plan lists).

## Task Planner Results Protocols

A Task Planner Results protocol adds a custom button to the **Results** tab of the Task Planner. The button generates an HTML report for the results. The report opens in a new tab in the Task Planner.

### Task Planner Results Protocols: Requirements

Item	Description
Input Parameters	<ul style="list-style-type: none"> <li>■ <b>Task Plan ID:</b> Passed to the protocol by Foundation Hub.</li> <li>■ <b>Filters:</b> A string of OData or facet parameters that are appended to the API call to the <code>runsetresults</code> endpoint. See "Actions Performed" below.</li> </ul>
Output File	<code>\$(jobdir)/index.html</code>
Actions Performed	<p>A Task Planner Results protocol should generate a Pipeline Pilot HTML report. To retrieve the results from the Task Plan, the protocol should send a GET REST call to the Foundation Hub API as follows:</p> <pre>api/v1/runsetresults?\$filter=\$(filterParam)</pre> <p>The filters are generated from the options supplied in the Filters panel of the tasks list. If no filters are supplied, all results are included in the report.</p> <p>For more information about the <code>runsetresults</code> endpoint, see the <i>Foundation Hub API Reference Guide</i>.</p>
Results	<p>The protocol must generate a response property as a JSON string of the following form:</p> <pre>{JobResponseCode: 200, errorMessage: '', Results: ''}</pre>

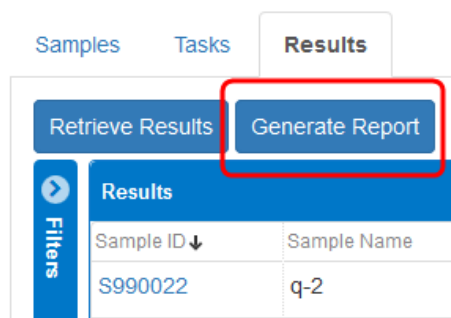
### Task Planner Results Protocols: Foundation Hub Configuration

To set up a Task Planner Results protocol in Foundation Hub:

1. Create a Task Planner Results protocol, conforming to the description in [Task Planner Results Protocols: Requirements](#).
2. In **Admin and Settings > Resources > Pipeline Pilot Protocols**, create a protocol entry that includes the following settings:
  - **Name:** The label of the button that will be added to the **Results** tab of Task Plans (for example, Generate Report).
  - **Usage Type:** Task Planner Results
  - **Path:** The path of the Pipeline Pilot protocol that you created in Step 1.

**Note:** For complete instructions for adding protocols, see Foundation Hub, see [Adding a Pipeline Pilot Protocol](#).

A button that generates the results report will appear in the Results tab of Task Plans:



## Task Planner Results Protocols: Examples

Two example Task Planner Results protocols are supplied with the Foundation Hub Component Collection:

- */Protocols/Examples/Integration/Integrating Applications/Hub/Task Planner Results/Hello World Task Planner Results*

This protocol generates a very simple "report" that shows Task Plan ID and the string "Hello World!!!". It is suitable for testing that Foundation Hub is correctly configured to generate Task Planner Results reports.

- */Protocols/Examples/Integration/Integrating Applications/Hub/Task Planner Results/Simple Task Planner Results*

This protocol generates a report containing a basic list of the Task Plan's results.

## Lifecycle Subscriber Protocols

A Lifecycle Subscriber protocol performs a custom action when triggered by Task lifecycle changes. For example, a protocol can be configured to send an email notification to an administrator whenever a Task entity enters the complete state.

Subscription to lifecycles of entities other than Tasks is not currently possible.

## Lifecycle Subscriber Protocols: Requirements

Item	Description
Input Parameters	<ul style="list-style-type: none"> <li>■ <i>publisher</i>: The URN of the Task whose lifecycle change triggered running of the protocol. Example: <code>urn:task:2fe08aa7-fe43-4d4e-9a85-d469a263a920</code></li> <li>■ <i>invokeAction</i>: The name of the lifecycle state change that triggered running of the protocol. Examples: <code>assign</code>, <code>inProgress</code></li> </ul>
Output File	Output is not mandatory, but many output formats are possible - see "Actions Performed".
Actions Performed	<p>A Lifecycle Subscriber protocol runs when triggered by particular Task lifecycle changes. The lifecycle changes are specified in the protocol's <b>Subscriber JSON</b> field in Foundation Hub. See <a href="#">Lifecycle Subscriber Protocols: Foundation Hub Configuration</a>.</p> <p>There are no special requirements for the operation of the protocol. Possible implementations include logging lifecycle changes in a file, and sending notifications of lifecycle changes to an administrator.</p>
Results	N/A

## Lifecycle Subscriber Protocols: Foundation Hub Configuration

To set up a Lifecycle Subscriber protocol in Foundation Hub:

1. Create a Lifecycle Subscriber protocol, conforming to the description in [Lifecycle Subscriber Protocols: Requirements](#).
2. In **Admin and Settings > Resources > Pipeline Pilot Protocols**, create a protocol entry that includes the following settings:
  - **Usage Type**: Lifecycle Subscriber
  - **Path**: The path of the Pipeline Pilot protocol that you created in Step 1.
  - **Subscriber JSON**: A JSON definition of the state changes that trigger the protocol. States are listed under the entity entry Task. For example:

```
{
  "task": {
    "states": [
      "draft",
      "submit",
      "assign",
      "inProgress",
      "complete",
      "released",
      "abandoned"
    ],
    "filters": {
      "activity/activityType": "Protocol"
    }
  },
}
```



**Notes:**

- The `states` field lists the states that trigger the protocol. If the `states` field is empty or nonexistent, the protocol is executed upon all state changes.
- The states have an internal name and a display name. The display name is shown in the user interface, but the internal name should be used when configuring the Subscriber JSON. The following table shows the Internal and display state names that can be used with Tasks:

Internal	Display
draft	Draft
submitted	Submitted
assigned	Assigned
inProgress	In Progress
completed	Completed
released	Released
abandoned	Abandoned
suspended	Suspended

- The `filters` field provides a mechanism to fine-tune the triggering of the protocol. In the example above, a Task state change only triggers a lifecycle subscription if the Activity Type associated with the Task is `Protocol`.

Filtering can be performed directly on the object (for example, the description of a Sample) or on children of the object, using a slash / as a separator (for example, the `activity/activityType` of a Task).

Field and entity names are case-sensitive. For information on entity structures, and proper casing of field names, see the *Foundation Hub API Reference Guide*.

You can specify multiple filters on a Task. Lifecycle subscriptions are only created for Tasks that match all the filter criteria. If no filters are specified, the subscription applies to all Tasks.

- Lifecycle subscriptions are only generated for Tasks that postdate the protocol entry. Lifecycle changes to Tasks that were created before the protocol entry do not trigger the protocol.

**Note:** For complete instructions for adding protocols, see Foundation Hub, see [Adding a Pipeline Pilot Protocol](#).

## Lifecycle Subscriber Protocol: Example

An example Lifecycle Subscriber protocol is supplied with the Foundation Hub Component Collection at the following location:

*/Protocols/Examples/Integration/Integrating Applications/Hub Lifecycle Subscription Example*

This protocol appends a message to an output file when an appropriate lifecycle action occurs. The message includes the date and time of the action, the URN of the publishing Task, and the name of the action. The file is output to the folder of the Foundation Hub user; for example:

C:\Program Files\BIOVIA\PPS95\public\users\svc\_foundation-hub\lifecycle.txt

To become familiar with how lifecycle subscriber protocols work, you can modify the protocol's **Subscriber JSON** in Foundation Hub, and see how this affects the generated messages. When verifying basic operation of the protocol, you might find it useful to avoid restrictive filters in the **Subscriber JSON**, so that more subscriptions are generated.

Example output:

```
wed Nov 07 06:02:01 2018 urn:task:9d6a9a8f-0523-43ba-929a-5eb8bfcd5849 invoked: assign
wed Nov 07 06:05:40 2018 urn:task:9d6a9a8f-0523-43ba-929a-5eb8bfcd5849 invoked:
inProgress
wed Nov 07 07:25:12 2018 urn:task:c05a176d-bf99-4a86-807e-aa557cb685c9 invoked: assign
Thu Nov 08 07:24:32 2018 urn:task:9268c3d5-4dac-49dd-ad4b-af5a554a5a25 invoked:
abandoned
```

## Protocol Troubleshooting

If a protocol fails to run correctly, perform the following checks:

- Consult the Foundation Hub and Pipeline Pilot Server log files. Either of these should provide clues to the nature of the error.
- Verify that Pipeline Pilot Server is registered with Foundation Hub.
- In **Admin and Settings > Pipeline Pilot Protocols**, open the protocol entry. Check that:
  - The **Usage Type** field specifies the correct protocol type.
  - The **Application** field specifies the correct Pipeline Pilot instance.
  - The **Path** field specifies a valid protocol path. Folders must be separated by single forward slashes (**not** backslashes). The path must begin with /Protocol. The initial slash is mandatory.
  - The **Life Cycle Entity** and the **Life Cycle Action** are correctly set if the protocol is a Lifecycle Action protocol. The Life Cycle Entity must be a valid Foundation Hub entity type, for example Task or Runset. The **Life Cycle Action** field cannot be empty: if it is, the protocol's button will not be displayed correctly in the user interface.
- In the case of an Activity protocol, check in **Admin and Settings > Activities** that the Foundation Hub Activity is correctly linked to its protocol. In particular:
  - Verify that the **Activity Type** field is set to Protocol.
  - Verify that the correct protocol is selected in the Activity's **Templates** field.
  - Verify in the **Activity Parameters** table that a suitable output parameter is configured for the results generated by the protocol. For example, if the protocol generates a pH value, the Activity must have a pH output parameter.
- In Pipeline Pilot client, check that the protocol is configured correctly. In particular:
  - Ensure that the protocol sets the *response* property properly.
  - Ensure that the protocol exports the response object.
- Use your browser's developer console to check for HTTP timeout (408) errors. These indicate that the protocol exceeds Foundation Hub's 30-second timeout limit for the protocol action to return a response. In this situation, you may need to refactor the protocol, by splitting it into two separate ones:
  1. A quick-running protocol that generates a placeholder page in an iframe, and launches a "worker" protocol during the onload event as a protocol function.
  2. The "worker" protocol that does the bulk of the processing.

When the "worker" protocol has finished running, the first protocol should load its content into the iframe, replacing the placeholder page.

## Chapter 7:

# Other Resource Data Administration

---

In addition to configuring [activities and activity plans](#), you use Foundation Hub **Admin and Settings** to configure other resource data that is used in Task Planner and in applications such as Compose and Capture.

- **Data Packets:** Map data from equipment to data in Foundation Hub. See the *BIOVIA Foundation Hub Equipment Guide*, which is available with the installation files and at <https://www.3ds.com/support/>.
- **Equipment:** Define data for managing lab equipment, such as equipment classes (for example, Balances), types (make and model), and individual device definitions. See the *BIOVIA Foundation Hub Equipment Guide*, which is available with the installation files and at <https://www.3ds.com/support/>.
- **Equipment Adapters:** Manage installed add-in software for equipment such as Empower and Chromeleon. See the *BIOVIA Foundation Hub Equipment Guide*, which is available with the installation files and at <https://www.3ds.com/support/>.
- **Locations:** Define physical locations for devices (for example, Site, Building, Lab). See [Adding Locations and Organizations](#).
- **Measurements:** See [Managing Measurements](#).
- **Organizations:** Define your company, vendors, and customers. An organization can be broken down into sub-organizations (such as departments and divisions). See [Adding Locations and Organizations](#).
- **Parameter Templates:** Define templates. See [Managing Parameter Templates](#).
- **Pipeline Pilot Protocols:** Manage Pipeline Pilot Protocols used by Foundation Hub for adding Activities, managing Life Cycle actions, and adding Task Planner Results buttons. See [Managing Pipeline Pilot Protocols](#).
- **Projects:** Project data required by some applications. Refer to the help for that application for details. See [Projects](#).
- **Sequence Templates:** Sequence Template data required by some applications. See [Sequence Templates](#).
- **Unit Types:** System of units based on the International System of Units. See [Managing Unit Types](#).
- **Vocabularies:** Lists of common terms used by BIOVIA applications. See [Managing Vocabularies](#).

## Equipment Measurements Store

A measurement is a package of data that Foundation Hub has generated from an equipment reading. The **Measurement Store** is collection of measurements from devices that are configured with direct-connect, advanced, file-based, or adapter connection types in Foundation Hub.

Measurements can be generated in several different ways, depending on how the device is connected. For example:

- Automatic execution of the *Parse* protocol on a file discovered by the File Crawler.
- Manual execution of the *Parse* protocol in the Pipeline Pilot client. .
- Automatic execution of the *Parse* protocol after manual upload of a data file..

For more information about how measurements are generated from equipment, see the *BIOVIA Foundation Hub Equipment Guide*.

## The Measurements Page

The **Measurements** page shows summary information about each measurement package in the Measurement Store. To view the Measurements page, open **Admin and Settings > Resources > Measurements**.

From the Measurements page, you can open a page showing full details of individual measurements. You can also view the details of the equipment that generated each measurement. For more information, see [Viewing Full Measurement Details](#) and [Viewing Details of a Measurement's Equipment](#).

## Fields in the Measurements Table

For each measurement package, the following information is shown:

- **Display Name:** The automatically-generated display name of the measurement package. This begins with the name of the output file of parsed data, and also includes the Barcode and Nickname of the equipment that generated the data.
- **Record Date:** The date and time when the raw data was last updated in the file system.
- **Equipment:** The Barcode of the equipment that produced the measurement.
- **Date Created:** The date and time when the measurement was uploaded to Foundation Hub. This is approximately when the raw data was parsed.
- **Last Updated:** The date and time when the data file was last modified in Foundation Hub.
- **Context:** A reference to the context of the measurement. For example, a Capture recipe execution or another task.

## Viewing Full Measurement Details

To view full details of a measurement:

1. Open **Admin and Settings > Resources > Measurements**.
2. Click the row of the measurement in the list of measurements.

**Note:** Clicking the hyperlink in the **Equipment** column opens the Equipment page, and not the Measurement Details page. See [Viewing Details of a Measurement's Equipment](#).

The **Measurement Details** page opens. There, you can view full information about the measurement, and open its attachments.

## Viewing Details of a Measurement's Equipment

To view the Equipment page for the instrument that generated a measurement:

1. Open **Admin and Settings > Resources > Measurements**.
2. Click the hyperlinked equipment name in the **Equipment** column of the measurement.

## Measurement Details

The **Measurement Details** page shows the data generated from an equipment reading. It can also show general header data about the measurement and attachment files associated with the measurement, if applicable.

## Viewing Measurement Details

By default, the Measurement Details page opens in Grid View, which lists the readings in the measurement file. You can toggle between Grid View and Header View, which shows general information about the measurement, if available.

To view details of a measurement:

1. Open **Admin and Settings > Resources > Measurements**.
2. Click the measurement you want to view.

The main panel shows the raw data string, if applicable, and parsed data in Grid View.

3. To toggle between views, click **Grid View**  or **Header View** .

### Raw Data

For direct-connected equipment, the raw data string that the device sent to Foundation Hub appears above any parsed data in Grid View or Header View.

### Grid View

The data listed in Grid View varies depending on the device configuration and how the data is parsed. However, these fields are always present:

- **Mapped Sample Id:** The sample ID to which the measurement reading is mapped in Foundation Hub. If the sample was mapped automatically when the measurement was taken, this is the same as the sample ID in the measurement reading. It can be different, however, if a user mapped the ID manually during execution of a data acquisition task. For more information, see "Executing Data Acquisition Tasks" in the *BIOVIA Foundation Hub Help*.
- **Instrument Sample Id:** The sample ID to which the reading applies. The **Is Sample Id** flag in the data packet determines which field this is in the source data. For an explanation of data packets and data fields in Foundation Hub, see the *BIOVIA Foundation Hub Equipment Guide*.
- **Row:** The record's row in the parsed data table. Rows are numbered from the first data record. For example, if the parsed data file is an Excel spreadsheet, and table data starts on Row 6 (with Rows 1 to 5 containing metadata and column headings), then Row 1 in the Grid View table corresponds to Row 6 in the Excel file.

### Header View

The Header View shows information from the parsed file. Header information is typically metadata that is not specific to any individual results (for example, the instrument operator or comments).

The fields shown in the Header View are specific to the format of the parsed data. This format is defined in the data packet details of the instrument's type. For an explanation of data packets and data fields in Foundation Hub, see the *BIOVIA Foundation Hub Equipment Guide*.

If the measurement does not have header data, the Header View is empty, with just the message, "No header data is available".

### Viewing Attachments

The **Attachments** panel contains links to files associated with the measurement. By default, it shows just the source file or files of the measurement. You can expand the list to include system files.

### Opening an Attachment

To open an attachment:

- In the **Attachments** panel, click the entry for the attachment that you want to open.  
The attachment opens in the associated application (for example, Microsoft Excel if the file is in XLSX or CSV format).

## Showing System File Attachments



By default, the **Attachments** panel shows only the final parsed data files. It does not show the intermediary files (the "system" files) that are generated during parsing. To show the system files as well:

- Select the **Display System Files** box.

The system files can be in various formats, such as CSV or JSON.

## Expanding and Collapsing the Attachments Panel

The Attachments panel is expanded by default.

- To collapse the panel, click the icon  in the title bar.
- To expand a collapsed panel, click the icon  in the title bar.

## Locations, Organizations, Cost Centers, and Contacts

Foundation Hub and integrated applications use organization and location information for functions such as registering equipment, identifying where samples are stored, and other important informational purposes. They use cost centers and contacts primarily for informational purposes and, in some cases, filtering.

Organizations and locations can be hierarchical. When you define a location and organization hierarchies, define them from the top down, so that you can select the correct parent entity when you define a child entity.

### Tips:

Contact is an attribute you can select when you define a location, so it is most efficient to define contacts before you define locations.

Location and cost center are attributes you can select when you define organizations, so it is most efficient to define them before you define organizations.

## Add Locations

Locations are arranged hierarchically: **Site > Building > Floor > Room > Lab**. Begin by adding a site. Then add child locations such as buildings, floors, rooms, and labs as needed, from the top down. You can skip unnecessary levels. For example, you can define sites that have no child locations, sites that have buildings and labs, but no floors, and so on. You can also use the REST interface to define custom locations types. For details, see [Adding Hierarchical Vocabularies](#).

1. Open **Resources > Locations**.
2. Click **Add Location**.
3. Specify a **Name**, **Location ID**, and **Description**.
4. To create a site:
  - a. From **Parent Location**, select the blank row.
  - b. From **Location Type**, choose **Site**.
  - c. From **Timezone**, select the timezone. The timezone is required if equipment events must be logged at this location. Child locations of a site inherit the site's timezone.
5. To create a child location, select a **Parent Location**. The child types available depend on the selected parent location.
6. If needed, select a **Primary Contact** and enter an **External ID**.

7. If you use CISPro, select the **Sync to Inventory** check box to sync inventory with CISPro.
8. If the location represents a site or building location, complete the **Address** fields.
9. Click **OK**.


### Add Organizations

You can define organizations to represent your vendors, customers, and your own company. You can then add lower-level organizations such as division and department. Organizations are often used to indicate ownership or responsibility for equipment and other resources. For hierarchical organizations, add the parents first so that you can select them when you add the child organizations.

1. Open **Resources > Organizations**.
2. Click **Add Organization**.
3. Set the **Name** and **Organization Type**, and optionally **Description**, **Parent Organization**, and **Primary Contact**.
4. Click **OK**.


**Note:** You can control the available list of organization types in **Resources > [Vocabularies](#)**.

### Add Contacts


1. Open **Resources > Organizations > Related Items > Contacts**.
2. Click the **Add Contact** button .
3. Set the **Name**, and optionally **Role**, **Email**, and **Phone Number**.
4. Click **OK**.

### Add Cost Centers

Cost centers identify the organization responsible for charges for activities.

1. Open **Resources > Organizations > Related Items > Cost Centers**.
2. Click the **Add Cost Center** button .
3. Set the **Name**, and optionally **Description**.
4. Click **OK**.

### Cloning a Location

1. From the **Admin and Settings** page, click **Locations**.
2. In the list of locations, move the pointer over the location that you want to clone.
3. Click the **Clone location** icon .
4. In the **Confirm Clone** dialog box, click **Yes**.

A clone of the location is created, and an editor for the new location opens.

By default, the name of the new location is that of the source location, with a numeric suffix to make it unique. For example, if you clone the location Building A, the first cloned location is called Building A(2), the second Building A(3), and so on.


When you clone a location, related descendant locations are cloned with it. The mapImage is not cloned.



The ID of the cloned location is a serialized, automatically-generated string. It has the form LSn, where n is a zero-padded integer, incremented by 1 for each new location (LS000002, LS000003, and so on).

5. Edit the details of the cloned location.
6. Click **Save**.

## Cloning an Organization

1. From the **Admin and Settings** page, click **Organizations**.
2. In the list of organizations, move the pointer over the organization that you want to clone.
3. Click the **Clone organization** icon .
4. In the **Confirm Clone** dialog box, click **Yes**.


A clone of the organization is created, and an editor for the new organization opens.

By default, the name of the new organization is that of the source organization, with a numeric suffix to make it unique. For example, if you clone the organization Research and Development, the first cloned organization is called Research and Development(2), the second Research and Development (3), and so on.

When you clone an organization, related descendent organizations are cloned with it.

5. Edit the details of the cloned organization.
6. Click **Save**.

## Cloning a Contact


1. From the **Admin and Settings** page, click **Organizations > Related Items > Contacts**.
2. Select the check box for the contact that you want to clone.
3. Click the **Clone Selected** icon .
4. In the **Confirm Clone** dialog box, click **Yes**.

A clone of the contact is created, and an editor for the new contact opens.

By default, the name of the new contact is that of the source contact, with a numeric suffix to make it unique. For example, if you clone the contact Jan Smits, the first cloned contact is called Jan Smits (2), the second Jan Smits(3), and so on.

5. Edit the details of the cloned contact.
6. Click **Save**.

## Cloning a Cost Center

1. From the **Admin and Settings** page, click **Organizations > Related Items > Cost Centers**.
2. Select the check box for the cost center that you want to clone.
3. Click the **Clone Selected** icon .
4. In the **Confirm Clone** dialog box, click **Yes**.

A clone of the cost center is created, and an editor for the new cost center opens.

By default, the name of the new cost center is that of the source cost center, with a numeric suffix to make it unique. For example, if you clone the cost center IT, the first cloned cost center is called IT(2), the second IT(3), and so on.

5. Edit the details of the cloned cost center.
6. Click **Save**.

## Projects


You can define projects that are available for various uses within BIOVIA Foundation and its integrated applications.

In Foundation Hub, users can associate a project with each activity, activity plan, and task plan that they create.

Users of Task Planner, the Tasks widget, and the Available Tasks widget can use the Project field to filter their task plans and tasks.

**Note:** Tasks inherit their project assignments from the task plan in which they are created, not from the activity on which they are based.


### Adding a Project

1. Open **Admin and Settings > Resources > Projects**.
2. Click the **Add Project** button .
3. Specify the **Name**, and optionally the **Description**.
4. Click **OK**.

### Deleting a Project

1. Open **Admin and Settings > Resources > Projects**.
2. Select the check box for the project that you want to delete. You can select multiple projects.
3. Click the **Delete Selected** (minus) icon.
4. In the **Confirm Delete** dialog box, click **Yes**.

### Cloning a Project

1. From the **Admin and Settings** page, click **Projects**.
2. In the list of projects, move the pointer over the project that you want to clone.
3. Click the **Clone project** icon .
4. In the **Confirm Clone** dialog box, click **Yes**.


A clone of the project is created, and an editor for the new project opens.

By default, the name of the new project is that of the source project, with a numeric suffix to make it unique. For example, if you clone the project Great Cakes, the first cloned project is called Great Cakes(2), the second Great Cakes(3), and so on.

5. Edit the details of the cloned project.
6. Click **Save**.

## Sequence Templates

### Adding a Sequence Template

1. Open **Admin and Settings > Resources > Sequence Template**.
2. Click the **Add Sequence Template** button .
3. Complete the **Add Sequence Template** form.
4. Click **OK**.


### Deleting a Sequence Template

1. Open **Admin and Settings > Resources > Sequence Template**.
2. Select the check box for the Sequence Template that you want to delete. You can select multiple projects.
3. Click the **Delete Selected** (minus) icon.
4. In the **Confirm Delete** dialog box, click **Yes**.

## Unit Types

Foundation Hub includes common Unit Types (for example, Area) and associated Units (for example, cm<sup>2</sup>) based on the International System of Units. From **Resources > Unit Types**, you can add a new Unit Type or add new Units to an existing Unit Type. You can also classify existing Units into Unit Categories to help you manage unit standards.

### Adding a Unit Type

1. Open **Resources > Unit Types**.
2. Click the **Add Unit Type** button .
3. Add a **Name** and optional **Description**.
4. Click **Add** to [define the Units](#) for the new Unit Type.
5. Check the **Add another** checkbox if you need to create further Unit Types.
6. Click **OK**.

Click **Cancel** to discard the new Unit Type.

### Editing a Unit Type


1. Open **Resources > Unit Types**.
2. Click name of Unit Type that you want to edit.

**Tip:** Use the filter above the Unit Types list to search for the Unit Type you require.

3. In the Unit Type view click the **Edit** button.
4. Make the required changes.
5. Click **Save**.

Click **Cancel** to discard the changes.

### Deleting a Unit Type


1. Open **Resources > Unit Types**.
2. Check the checkbox for the Unit Type that you want to remove.
3. Click the **Delete Selected** button .
4. In the confirmation dialog, click **Yes** to delete the Unit Type or **No** to cancel the deletion.

### Units

You can classify existing Units into Unit Categories to help you manage unit standards.

**Note:** If you are using BIOVIA Compose and Capture and you add new units, you must restart the BIOVIA Compose Service before they are available to your application.

### Adding Units

1. You can add Units by:
  - While defining a new Unit Type: Click **Add** in the Units section.
  - While editing an existing Unit Type: Click **Add** in the Units section.
  - Add a new unit:
    - a. Open **Resources > Unit Types**.
    - b. Click **Units** in the Related Items section.
    - c. Click the **Add Unit** button .
2. Add a **Name** for the Unit and a **Text Name**, and any of the other optional fields. Note that Text Name only accepts alpha-numeric characters.
3. If you are adding a new Unit as a related item, select the **Unit Type** to which the Unit should be assigned.
4. Set the conversion factor between the new unit and the base unit, using **Conversion Factor**, **Conversion Divisor**, and **Conversion Offset**: (input value x Conversion Factor / Conversion Divisor) + Conversion Offset. For the base unit itself, the factor and divisor will be 1 and the offset 0.
5. Check the **Add another** checkbox if you need to create further Unit Types.
6. Click **OK**.  
Click **Cancel** to discard the new Unit.
7. If you are adding a new Unit directly to a Unit Type, select a Unit and use the **Up** and **Down** buttons to define the order of the Units.


### Editing a Unit

1. Open **Resources > Unit Types** and click **Units** in the Related Items section.
2. Click name of Unit that you want to edit.

**Tip:** Use the filter above the Units list to search for the Unit you require.

3. In the Unit view click the **Edit** button.
4. Make the required changes.
5. Click **Save**.  
Click **Cancel** to discard the changes.


## Deleting a Unit

1. Open **Resources > Unit Types** and click **Units** in the Related Items section.
2. Check the checkbox for the Unit that you want to remove.
3. Click the **Delete Selected** button .
4. In the confirmation dialog, click **Yes** to delete the Unit or **No** to cancel the deletion.

## Unit Categories

You can assign Units to Unit Categories to help manage the units. For example, you may want to group all of the mass units for the imperial system of measurement.

### Adding a Unit Category

1. Open **Resources > Unit Types** and click **Unit Categories** in the Related Items section.
2. Click the **Add Unit Category** button .
3. Add a **Name** and optional **Description**.
4. Select a Unit from the Units list and click **Assign** to associate it with this category.

**Tip:** Start typing the name of the Unit you require to filter the Units list.

5. Add further Units as required and use the **Up** and **Down** buttons to define their order.
6. Check the **Add another** checkbox if you need to create further Unit Categories.
7. Click **OK**.  
Click **Cancel** to discard the new Unit Category.


### Editing a Unit Category

1. Open **Resources > Unit Types** and click **Units Categories** in the Related Items section.
2. Click name of Unit Category that you want to edit.

**Tip:** Use the filter above the Unit Categories list to search for the Unit Category you require.


3. In the Unit Category view click the **Edit** button.
4. Make the required changes.
5. Click **Save**.  
Click **Cancel** to discard the changes.

### Deleting a Unit Category

1. Open **Resources > Unit Types** and click **Units Categories** in the Related Items section.
2. Check the checkbox for the Unit Category that you want to remove.
3. Click the **Delete Selected** button .
4. In the confirmation dialog, click **Yes** to delete the Unit Category or **No** to cancel the deletion.

### Cloning a Unit Category

1. From the **Admin and Settings** page, click **Unit Types > Related Items > Unit Categories**.
2. Select the check box for the unit category that you want to clone.

3. Click the **Clone Selected** icon .
4. In the **Confirm Clone** dialog box, click **Yes**.  
 A clone of the unit category is created, and an editor for the new unit category opens.  
 By default, the name of the new unit category is that of the source unit category, with a numeric suffix to make it unique. For example, if you clone the unit category Manufacturing, the first cloned unit category is called Manufacturing(2), the second Manufacturing(3), and so on.  
 When you clone a unit category, associations with units are cloned with it. Note that the units themselves are not cloned.
5. Edit the details of the cloned unit category.
6. Click **Save**.

## Exporting and Importing Resources

You can [export](#) data from Foundation Hub, and [import](#) data into it. This facilitates migration of data from one instance of Foundation Hub to another.

You export and import data using the BIOVIA Foundation REST interface. Note that there is no user interface for export and import.

## Data That You Can Export and Import

The table below shows the data that you can export from and import into Foundation Hub. When you export data, use appropriate values from the table in the body of the REST request. For instructions, see [Exporting Resources](#).

Columns in the table:

- **Entity type code name** shows the names of the top-level entity types that you can export and import.
- **Expandable entity subfields** shows which related entity types can be included in-line using the \$expand parameter when you export data.  
 If you do not use the \$expand parameter, you must export any required related entities as top-level entities. Which of the two methods you choose is a matter of preference: the new data will be the same in the target Foundation Hub database after import.  
 For guidelines on using the \$expand parameter to expand subfields, see [JSON OData Parameters](#).
- **Other expandable subfields** shows data structures that can be included in-line using the \$expand parameter, but that are not top-level entity types. The only way of including these data structures is to use the \$expand parameter.  
 For guidelines on using the \$expand parameter to expand subfields, see [JSON OData Parameters](#).

Entity type code name	Expandable entity subfields	Other expandable subfields
activity	–	activitycosts activitytemplates
contact	–	–
costcenter	–	–
datafield	–	–

Entity type code name	Expandable entity subfields	Other expandable subfields
datapacket	datafields	-
equipment	-	-
equipmentclass	-	-
equipmenttype	equipment	equipmenteventtypes
location	-	-
organization	-	-
parametertemplate	-	parameterconstraints
property	-	-
propertybag	-	-
reasoncode	-	-
signaturepolicy	reasoncodes	-
specification		specificationmethods
unit	-	-
unitcategory	-	-
unittype	units	-
vocabulary	vocabularyentries	-
vocabularyentry	-	vocabularyaliases

## REST Clients

You can use the following types of REST client applications to call the Foundation Hub export and import REST resources:

- A dedicated REST client application such as Postman, available from <https://www.getpostman.com/>.
- A BIOVIA Pipeline Pilot protocol that uses the *Hub Connector* or the *HTTP Connector* component for the REST calls.
- A program or script in a language with an HTTP client library such as C#, Java, Groovy, Perl, or Python.

## Permissions Required

To export and import Foundation Hub data, you must be a member of the Foundation/Administrators group.

## Exporting Resources

### Notes:

- If you do not specify resources to export in the REST request payload, a default set of resources is exported. See [Default Export Payload](#).
- You cannot export authentication entities (users, groups, and so on), or entities that rely on authentication entities (samples, tasks).
- You can only export Foundation Hub entity types, not entity types created by other applications.
- All export requests are added to the Foundation Hub audit table.

To export Foundation Hub data:

1. Log in to Foundation Hub by sending the following REST request:

### Request:

HTTP Method	POST
Resource	<code>https://&lt;server&gt;:&lt;port&gt;/foundation/hub/api/v1/security/login</code>
Header	Content-Type: application/json
Body Content	<p>Login credentials in JSON format:</p> <pre>{   "client_id"    : "foundation-hub",   "username"     : <i>Hub Administrator user name</i>,   "password"     : <i>Hub Administrator password</i> }</pre> <p>The <code>client_id</code> value must be <code>foundation-hub</code>.</p> <p>Example:</p> <pre>{   "client_id"    : "foundation-hub",   "username"     : "hubadmin",   "password"     : "Hu6p^55vv0r )" }</pre>

### Response:

An HTTP 200 "OK" response with a JSON payload as follows:

```
{
  "access_token": Bearer access token,
  "token_type": "bearer",
  "expires_in": Time to expiry
}
```

Example:

```
{
  "access_token": "328a66f5-5518-4aa2-8b8c-cd3193239d04",
  "token_type": "bearer",
  "expires_in": 28799
}
```



- Copy the access token from the response in Step 1. You will need it for authentication when calling the export/download resource.
- Export data from Foundation Hub by sending the following REST request:

**Request:**

HTTP Method	POST
Resource	<code>https://&lt;server&gt;:&lt;port&gt;/foundation/hub/api/v1/export/download</code>
Headers	<p>Content-Type: application/json</p> <p>Authorization: Bearer <i>Access token from Step 1</i></p> <p>The word Bearer must be followed by a single space.</p> <p>Example:</p> <p>Authorization: Bearer 328a66f5-5518-4aa2-8b8c-cd3193239d04</p>
Body Content	<p>JSON description of the data to export:</p> <pre>{ "filename"      : <i>Exported ZIP file name(optional)</i>,   "reason"       : <i>Reason for export(optional)</i>,   "timestamp"    : <i>Time stamp filter (optional; entities                     modified after this time are returned)</i>,   "entityTypes" : [     { "filename"   : <i>JSON file name(optional)</i>,       "entityType" : <i>Entity type</i>,       "params"     : <i>Parameter list (optional)</i>     }     ...   ] }</pre> <p>Example:</p> <pre>{ "filename"      : "hubData.zip",   "reason"       : "Location and Vocabulary data drop for new server",   "timestamp"    : "2017-03-28T11:01:46.234Z",   "entityTypes" : [     { "filename"   : "myVocabularies.json",       "entityType" : "vocabulary",       "params"     : {"\$filter": "locked eq false"}     },     { "entityType" : "vocabularyentry",       "params"     : {"\$expand": "vocabularyaliases" }     },     { "entityType" : "location"     }   ] }</pre>

### Notes:

- The order of entity types is important. Entities that are dependent on other entities must come after their dependencies. This is most commonly the case where entities have a parent-child relationship, with the parent coming first. For example, `equipmentClass` entities must precede `equipmentType` entities.  
If you do not export entities in the correct order, the order will be incorrect in the `manifest.json` file, and a subsequent import using this file will probably fail. (You can, however, correct the order of entities in the `manifest.json` file before using it for import.)
- If the top-level `filename` field is absent, the exported file is called `export.zip`.
- The body can be empty. If it is, a default payload is applied that exports data created since Foundation Hub was installed. See [Default Export Payload](#).
- All the top-level fields are optional, but if there is an `entityTypes` array field, it must contain at least one member, and that member must have a valid `entityType` field. The `filename` and `params` fields are optional.
- For a list of `entityType` values, see [Data That You Can Export and Import](#).
- If no `filename` field is supplied for an entity type, the archived file is named after the entity type, with the extension `.json`; for example, `vocabularies.json`, `unitcategories.json`.
- For an explanation of the optional `params` field, see [JSON OData Parameters](#).
- If an entity type has extended properties (usually accessible through the facet `$properties`), these are also exported.

### Response:

An HTTP 200 "OK" response with a streamed binary ZIP file payload. See [Contents of an Exported ZIP File](#).

**Tip:** Some REST client applications allow you to download the returned payload as a file on your computer, instead of displaying it (which is not very useful for a ZIP file). If this feature is available, it is recommended that you use it. For example, in Postman, click the down arrow on the **Send** button, and select **Send and Download**.

## JSON OData Parameters

Each entity type record in the submitted JSON can include a `params` parameter. This includes parameters that modify what data is retrieved.

The parameters are OData parameters, and support their standard OData syntax, described at <https://<Foundation Hub server>:<port>/foundation/hub/doc/odata>

Two parameters are available:

- `$filter`: Filters the entities according to the specified condition.

Standard OData functions and special variables can be included in the condition expression.

The following example body content retrieves units whose unit type is `Angle`:

```
{ "filename": "angleUnits.zip", "reason": "Angle units",
  entityTypes: [
    {
      "entityType": "unit",
      "params": { "$filter": "unitType/name eq 'Angle'" }
```

```
    }
  ]
}
```

- **\$expand**: Includes related data as inline JSON records. The related data can be an entity type (in the plural; for example `vocabularyentries`), or a child data type that does not have its own entity type (for example `vocabularyaliases`).

In the case of an entity type, as an alternative to using the `$expand` parameter, you can specify the entity type as a separate top-level entity. The effect when the data is imported to another Foundation Hub database is the same.

In the case of a child data type that does not have its own entity type, using the `$expand` parameter is the only way that you can transfer it between Foundation Hub databases.

For details of which expandable data fields are allowed with each top-level entity type, see [Data That You Can Export and Import](#).

The following example body content retrieves unit types. In each unit type record, the `units` field is expanded to include all units of the type:

```
{ "filename": "unitTypesAndUnits.zip",
  "reason": "Unit types with expanded units",
  "entityTypes": [
    {
      "entityType": "unittype",
      "params": { "$expand": "units" }
    }
  ]
}
```

Example returned unit type record:

```
{
  "id": "d96e0fd7-8ca1-4000-b21e-0326d2dcfd76",
  "description": null,
  "dimensionAmountOfSubstance": 0,
  "dimensionElectricCurrent": 0,
  "dimensionLength": 2,
  "dimensionLuminousIntensity": 0,
  "dimensionMass": 0,
  "dimensionThermodynamicTemperature": 0,
  "dimensionTime": -2,
  "externalId": "http://qudt.org/schema/qudt#AbsorbedDoseUnit",
  "lastUpdated": "2017-06-13T11:28:21.928Z",
  "locked": true,
  "name": "Absorbed Dose",
  "units": [
    {
      "id": "477f99e3-09ee-4b28-b5c0-1ab50eb0cf0f",
      "baseUnit": true,
      "conversionDivisor": 1,
      "conversionFactor": 1,
      "conversionOffset": 0,
      "description": "gray",
      "displayName": "Gy (gray)",
      "displayOrder": 0,

```

```
    "externalId": "http://qudt.org/vocab/unit#Gray",
    "lastUpdated": "2016-11-29T14:35:05.528Z",
    "locked": true,
    "name": "Gy",
    "source": null,
    "symbol": "Gy",
    "textName": "GRAY",
    "unitType": {
      "id": "d96e0fd7-8ca1-4000-b21e-0326d2dcfd76",
      "name": "Absorbed Dose"
    }
  },
  {
    "id": "a8309508-a193-44c7-924d-85c6ece915d1",
    "baseUnit": false,
    "conversionDivisor": 100,
    "conversionFactor": 1,
    "conversionOffset": 0,
    "description": "radiation doses",
    "displayName": "rad (radiation doses)",
    "displayOrder": 1,
    "externalId": "http://qudt.org/vocab/unit#Rad",
    "lastUpdated": "2016-11-29T14:35:05.543Z",
    "locked": true,
    "name": "rad",
    "source": null,
    "symbol": "rad",
    "textName": "RAD",
    "unitType": {
      "id": "d96e0fd7-8ca1-4000-b21e-0326d2dcfd76",
      "name": "Absorbed Dose"
    }
  }
]
}
```

Only one JSON entity type file is generated in the exported ZIP file, because there is only one top-level data type. By default, this file is called `unittypes.json`.

Note that, because units are also top-level entities, the same data can be retrieved without using the `$expand` parameter:

```
{
  "filename": "unitTypesAndUnits.zip",
  "reason": "Separate uUnit types and units",
  "timestamp": "2017-03-28T11:01:46.234Z",
  "entityTypes": [
    {
      "entityType": "unittype"
    },
    {
      "entityType": "unit"
    }
  ]
}
```

```
  ]
}
```

This generates two separate JSON entity type files in the ZIP file, which by default are called `unittypes.json` and `units.json`. Although the structure of the exported JSON is different from when `$expand` is used, the actual data is the same. Therefore, importing the ZIP file into a new Foundation Hub instance has the same effect in both cases.

## Default Export Payload

If you send an export REST request with an empty body payload, the following default export payload is used:

```
{ "entityTypes": [
  {"entityType": "propertybag"},
  {"entityType": "property"},
  {"entityType": "unittype", "params": {"$filter": "locked eq false"}},
  {"entityType": "unit", "params": {"$filter": "locked eq false"}},
  {"entityType": "unitcategory", "params": {"$expand": "units"}},
  {"entityType": "vocabulary", "params": {"$filter": "locked eq false"}},
  {"entityType": "vocabularyentry", "params": {"$expand":
"vocabularyaliases" }},
  {"entityType": "costcenter"},
  {"entityType": "contact"},
  {"entityType": "datapacket"},
  {"entityType": "datafield"},
  {"entityType": "location"},
  {"entityType": "organization"},
  {"entityType": "signaturepolicy", "params": {"$expand": "reasoncodes"}},
  {"entityType": "parametertemplate", "params": {"$expand":
"parameterconstraints"}},
  {"entityType": "activity", "params": {"$expand":
"activitycosts,activitytemplates"}},
  {"entityType": "specification", "params": {"$expand": "*"}},
  {"entityType": "equipmentclass"},
  {"entityType": "equipmenttype", "params": {"$expand":
"equipmenteventtypes"}},
  {"entityType": "equipment"}
]}
```

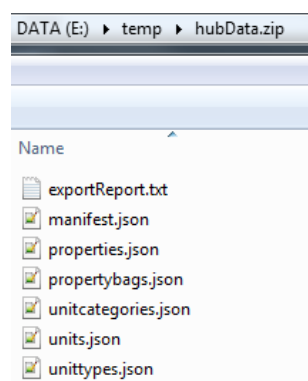
Note that this payload uses `locked eq false` filters to export only custom data that has been added since Foundation Hub was installed.

## Contents of an Exported ZIP File

The ZIP file that is generated when you export Foundation Hub resources contains the following files:

- [manifest.json](#): A JSON description of the exported data files.
- [A set of JSON files, one for each exported entity type](#), containing details of the entities.
- [exportReport.txt](#): A text file containing log information for the export.

Example:



**Note:** You can create your own ZIP file of data to import into Foundation Hub. If you do this, ensure that its component files conform to the descriptions that follow.

### manifest.json

The `manifest.json` file lists the entity type JSON files and provides summary information about them. It is required for data import (see [Importing Resources](#)).

Each exported entity type has a record in `manifest.json` in the following format:

```
{
  "filename": JSON file name,
  "entityType": Entity type code name,
  "mode": Synchronization mode. Allowed values: "all", "insertOnly",
"noDelete",
"deleteOnly",
  "$filter": OData filter to apply to the synchronization event used for
import,
  "matchOn": Field used to determine whether an imported entity already
exists in the
database, when the synchronization mode is not "all". For example,
{"matchOn": "name"} makes an appropriate change to an existing
database record,
if its name field matches the name field of the imported record
}
```

In an exported `manifest.json` file:

- "mode" is always "noDelete"
- "matchOn" is always "id"
- "\$filter" is absent

### Creating or Editing a manifest.json File Manually

If you want to customize how the entity type files will be imported, you can edit the `manifest.json` file, and then reinsert it into the ZIP file.

The order of entity types is important. Entities that are dependent on other entities must come after their dependencies. This is most commonly the case where entities have a parent-child relationship, with the parent coming first. For example, `equipmentclass` entities must precede `equipmenttype` entities.

## Further Reading

More information about the fields of the manifest.json file is available at the following locations:

- mode and matchOn:

`https://<Foundation Hub server>:<port>/foundation/hub/doc/facet`

- \$filter:

`https://<Foundation Hub server>:<port>/foundation/hub/doc/odata/doc/odata#filter`

## Example 1: manifest.json File from Export

```
[
  {
    "filename": "propertybags.json",
    "entityType": "propertybag",
    "mode": "noDelete",
    "matchOn": "id"
  },
  {
    "filename": "properties.json",
    "entityType": "property",
    "mode": "noDelete",
    "matchOn": "id"
  },
  {
    "filename": "unittypes.json",
    "entityType": "unittype",
    "mode": "noDelete",
    "matchOn": "id"
  },
  {
    "filename": "units.json",
    "entityType": "unit",
    "mode": "noDelete",
    "matchOn": "id"
  },
  {
    "filename": "unitcategories.json",
    "entityType": "unitcategory",
    "mode": "noDelete",
    "matchOn": "id"
  }
]
```

## Example 2: User-Created manifest.json File for Import

```
[
  { "filename": "Locations.json", "entityType": "location",
    "mode": "all", "$filter": "description eq 'Example Locations'"},
  { "filename": "OntologyVocab.json", "entityType": "vocabulary",
    "mode": "all", "$filter": "description eq 'TGA 500 Ontology Vocab'"},
  { "filename": "OntologyVocabEntries.json",
    "entityType": "vocabularyentry", "mode": "all",
    "$filter": "vocabulary/description eq 'TGA 500 Ontology Vocab'"},
]
```

```
{ "filename": "TGA500EquipmentReadings.json",
  "entityType": "equipmentreading", "mode": "all",
  "$filter": "name eq 'TGA Q500'",
  { "filename": "TGA500DataFields.json", "entityType": "datafield",
    "mode": "all", "$filter": "dataPacket/name eq 'TGA Q500'",
    { "filename": "EquipmentClass.json", "entityType":
      "equipmentclass", "mode": "all", "$filter": "name eq 'Thermal
Analysis'",
      { "filename": "EquipmentTypes.json", "entityType": "equipmenttype",
        "mode": "all", "$filter": "description eq 'Example File Based Equipment
Type'",
        { "filename": "Equipment.json", "entityType": "equipment", "mode": "all",
          "$filter": "description eq 'Example File Based Equipment Type'"}
      ]
    }
  }
```

## Entity Type JSON Files

A JSON file is exported for each entity type, containing details of the entities. For example, if you export the vocabulary and unit entities, two entity type JSON files are included (with the default names `vocabularies.json` and `units.json`).

If more than 1000 entities of a type are returned, they are split between multiple files. Files are numbered with a suffix; for example, `units01.json`, `units02.json`.

Example (content of a `locations.json` file):

```
[
  {
    "id": "23fa73cb-308d-4365-8167-27d0b8ce7cef",
    "addressCity": "Whelkham",
    "addressCountry": "United States",
    "addressState": "MA",
    "addressStreet": "Mangrove Blvd",
    "addressStreet2": null,
    "addressZipCode": null,
    "contact": null,
    "description": "Research site for cakes, pastries, and buns.",
    "externalId": null,
    "lastUpdated": "2017-06-28T10:26:42.792Z",
    "locationId": "001",
    "locationType": {
      "id": "9ecad79a-59e7-4bd7-b99d-2ffd03ed8577",
      "externalId": null,
      "name": "Site",
      "vocabulary": {
        "id": "34cac102-8ae6-40a7-8dc2-38841c4e235f"
      }
    },
    "name": "Food Research Kitchen",
    "parent": null,
    "path": "Food Research Kitchen",
    "timezone": {
      "id": "dc71cda5-7b70-4a70-945a-7c63ad4c7450",
      "externalId": null,
      "name": "(UTC-05:00) Eastern Time (US & Canada)",
    }
  },
  ...
]
```



```

    "vocabulary": {
      "id": "b48e908f-efa8-44b2-a0fe-7f99f680d794"
    },
    "x": null,
    "y": null
  },
  {
    "id": "b2943b92-a112-444c-b8f5-bce8e5143fa6",
    "addressCity": "Offhall",
    "addressCountry": "United Kingdom",
    "addressState": null,
    "addressStreet": "Strada Street",
    "addressStreet2": null,
    "addressZipCode": "000 000",
    "contact": null,
    "description": "Chemical research site",
    "externalId": null,
    "lastUpdated": "2017-06-28T09:56:10.940Z",
    "locationId": "002",
    "locationType": {
      "id": "9ecad79a-59e7-4bd7-b99d-2ffd03ed8577",
      "externalId": null,
      "name": "Site",
      "vocabulary": {
        "id": "34cac102-8ae6-40a7-8dc2-38841c4e235f"
      }
    },
    "name": "Smith Laboratory",
    "parent": null,
    "path": "Smith Laboratory",
    "timezone": {
      "id": "f3dd6686-8e20-4d67-83a2-08e0d6f439de",
      "externalId": null,
      "name": "(UTC) Dublin, Edinburgh, Lisbon, London",
      "vocabulary": {
        "id": "b48e908f-efa8-44b2-a0fe-7f99f680d794"
      }
    },
    "x": null,
    "y": null
  }
]

```

### exportReport.txt

This file contains log information for the export. Fields and values are tab-delimited to facilitate viewing in Pipeline Pilot client or Microsoft Excel.

This file is *not* required for data import.

### Log Lines

- **Export created:** The time stamp of the file's creation in standard UTC format.
- **Server:** The server from which the file was exported.

- **Software release:** The release version and build number of the Foundation Hub server from which the data was exported.
- **Exported by:** The user name of the Hub administrator who exported the files.
- **Reason:** A reason for the data export. Note that this field cannot contain tab characters.
- **Export criteria:** A table listing the following information for each exported entity type:
  - **Entity type**
  - **Selection criteria:** Any `$filter` or `$expand` criteria applied to the entity type, otherwise `All data`
  - **Count of exported entities:** This can be useful to verify quickly that the expected data was exported
  - **File name:** The name of the exported entity type's JSON file
  - **Export status:** The HTTP export response code. Check that no HTTP error codes are logged
  - **Errors:** Any errors for the entity type's export

Example content:

```
Export started: 2017-06-28T10:27:04.966Z
Server:        myserver.mydomain.net
Software release: 2018 - 18.1.0
Exported by:   scitegicadmin
Reason:       Vocabulary and location data drop for new server
Export criteria: Entity type Selection criteria
                Count of exported entities File name Export status Errors
Export criteria: vocabularies {"$filter":"( locked eq false ) and
lastUpdated
ge '2017-03-28T11:01:46.234Z'"} 2 myVocabularies.json 200 []
Export criteria: vocabularyentries {"$filter":"lastUpdated ge
'2017-03-28T11:01:46.234Z'", "$expand":"vocabularyaliases"} 27
vocabularyentries.json 200 []
Export criteria: locations {$filter=lastUpdated ge
'2017-03-28T11:01:46.234Z'} 2 locations.json 200 []
Export result: Export completed
```

## Importing Resources

Usually, the data you are importing into Foundation Hub will have been exported from another Foundation Hub instance. In this case, all the files that you need for import should already exist.

You import resources by including the ZIP file containing the resources in the body of a REST request. There are two ways that you can do this:

- Use a REST client (such as Postman) that allows you to attach the ZIP file to a POST request. The REST client inserts the file content into the HTTP body when it sends the request.
- Write a short program or Pipeline Pilot protocol that inserts the ZIP file as binary data into the body of a POST request.

### Notes:

- You cannot import authentication entities (users, groups, and so on), or entities that rely on authentication entities (samples, tasks).
- You can [back out](#) imported data.

## Editing Data Before Import

You may want to edit exported data before you import it into the target Foundation Hub instance, or even to create import data from scratch. If you do either of these things, ensure that the source JSON files conform to the formats described in [Contents of an Exported ZIP File](#). Then add them to the ZIP file to upload. Ensure that the ZIP file also contains a valid `manifest.json` file.

## Importing Resources from a ZIP File

To import Foundation Hub data from a ZIP file:

1. Export or create the ZIP file of resources to import, as described in [Exporting Resources](#) and [Contents of an Exported ZIP File](#).
2. Log in to Foundation Hub on the target computer as described in Step 1 of [Exporting Resources](#).

**Note:** If you are still logged in to Foundation Hub from a previous REST call, you can omit this step, and reuse the access token from the earlier login.

3. Copy the access token from the response in Step 2. You will need it for authentication when calling the `import/upload` resource.
4. Import data into Foundation Hub by sending the following REST request:

### Request:

HTTP Method	POST
Resource	<code>https://&lt;server&gt;:&lt;port&gt;/foundation/hub/api/v1/import/upload</code> Example: <code>https://myhubserver:9953/foundation/hub/api/v1/import/upload</code>
Header	Authorization: Bearer <i>Access token copied in Step 3</i> The word Bearer must be followed by a single space. Example: Authorization: Bearer 328a66f5-5518-4aa2-8b8c-cd3193239d04
Body Content	The binary content of the ZIP file that contains the resources to import. <div> <b>Tip:</b> If you are using Postman as your REST client, specify <b>binary</b> as the <b>Body</b> type, and attach the ZIP file from its location on your computer. Postman inserts the ZIP file into the body when it sends the request.           </div>

### Response:

An HTTP 200 "OK" response with a JSON payload summarizing the database changes for each entity type.

Example:

```
[
  {
    "inserted": 6,
    "updated": 0,
    "deleted": 0,
    "_status": "200",
    "filename": "contacts.json"
```

```

    },
    {
      "inserted": 6,
      "updated": 0,
      "deleted": 0,
      "_status": "200",
      "filename": "organizations.json"
    },
    {
      "inserted": 10,
      "updated": 0,
      "deleted": 0,
      "_status": "200",
      "filename": "locations.json"
    }
  ]

```

### Importing Using a Program, or a Pipeline Pilot Protocol

If you have knowledge of a programming language with an HTTP client library, or of how to write Pipeline Pilot protocols, you can create a client application that sends an import REST request, with the import data in its body payload.

If you write a Pipeline Pilot protocol, you can use the *Hub Connector* or the *HTTP Connector* component to communicate with Foundation Hub.

To import Foundation Hub data from the body of a REST request, write code or a protocol that:

1. Logs in to the source Foundation Hub instance, as described in Step 1 of [Exporting Resources](#).
2. Reads the ZIP file from the response body of a call to the export/download resource of the source Foundation Hub instance. See [Exporting Resources](#).
3. Logs in to the target Foundation Hub instance.
4. Inserts the ZIP file as a byte stream into the body of a call to the import/upload resource of the target Foundation Hub instance. For the formats of the import REST request and response, see [Importing Resources from a ZIP File](#).

Example Groovy code excerpt:

```

byte[] exported = (byte[])respExport.body      // From export request
response.
def respImport = restBuilderForAdminUser.post('import/upload') {
  body exported
}

```

### Backing Out Imported Resources

You can back out (uninstall) data imported from a ZIP file.

To back out imported data:

- Follow the instructions in [Importing Resources](#), but add an `uninstall=true` parameter to the import URL.

For example:

```


https://myhubserver:9953/foundation/hub/api/v1/import/upload?uninstall=true

```

# Chapter 8:

## Additional Configuration Settings

---

- **Applications:** View installed BIOVIA applications and configure available application settings including Foundation Hub settings. See [Managing Applications](#) and [Foundation Hub Settings](#).
- **Extended Properties:** Extend BIOVIA Foundation data objects by setting up extended properties. See [Using Extended Properties](#).
- **Hub Configuration:** Make changes to the Foundation Hub configuration settings, such as Web Server and Database settings. See [Managing Foundation Hub Configuration](#).
- **Inventory Systems:** Connect an external inventory system for material information. See [Inventory System Connections](#).
- **Legal Notices:** Manage a legal notice, such as Personal Data Protection policy. See [Managing Legal Notices \(Personal Data Protection\)](#).
- **License Files:** Add, delete, and view license files for your deployment. See [Managing License Files](#).
- **Links:** Foundation Hub includes a  feature that users can use to access BIOVIA applications, Admin and Settings, and links to other applications and websites that you can configure. See [Managing links](#).
- **Logging:** Download log files and control the level of logging for Foundation Hub. See [Managing Logging](#).
- **Trusted Certificates:** Manage public certificates for trusted sites in the Foundation Hub server trust store. See [Managing Trusted Certificates](#).
- **User Directories:** Manage external directories that are used for authentication. See [Managing User Directories](#).

## Installed Application Settings

You can use the **Settings > Applications** area to view and manage installed BIOVIA applications and platform installations registered with Foundation Hub. You can also access application specific settings, such as those for Foundation Hub from this area.

### Applications

You can view all applications registered with Foundation Hub and view details such as version, associated groups and permissions, installation URL, data objects managed by the application, etc.

#### Viewing Application Details

1. Open **Settings > Applications**.
2. Click a row to view the details.

#### Editing the Application Label

1. Open **Settings > Applications**.
2. Click a row to view the details.
3. Click **Edit** to make changes to the **Label**.
4. Click **Save**.

### Installations

You can view registered software platforms upon which applications are installed (for example, Pipeline Pilot) and view details such as Root URL, applications installed on the platform, URL aliases, and whether it is load balanced and the list of nodes if it is.

#### Viewing Installation Details

1. Open **Settings > Applications > Related Items > Installations**.
2. Click a row to view the details.

#### Editing Installation Details

You can add URL Aliases, load-balanced Nodes, and set the Location for the installation.

1. Open **Settings > Applications > Related Items > Installations**.
2. Click a row to view the details.
3. Click **Edit**.
  - **UrlAliases:** Click **Add**, type the **URL**, and then click **OK**.
  - **Nodes:** Click **Add**, type a **Root URL**, and optionally an **SSL Root URL**, and then click **OK**.
  - **Location:** Choose a location from the list.
4. Click **Save**.

### Application Settings

The Foundation Hub Settings page provides access to settings specific to Foundation Hub. To access them, open **Settings > Applications > Application Settings > Foundation Hub**. For details about Foundation Hub settings, see [Foundation Hub Settings](#).

#### Foundation Hub Application Settings

To access the application settings for Foundation Hub, open **Settings > Applications > Application Settings > Foundation Hub**.

**Note:** For server configuration settings, open **Settings > Hub Configuration**. See [Foundation Hub Configuration Settings](#).

#### General

- **Maximum Results From a REST call:** Prevents a client from requesting every entity. The default is 5000.
- **Maximum Task Plan Results for Pivot:** Maximum number of pivoted results that Task Planner can display in its Activity Results tab. Defaults to 25,000. If you have larger result datasets and a server that is capable of handling them, you can increase this value. If you do so, ensure that your setting does not negatively impact server performance and usability.
- **Cycle Configuration:** JSON code used to rename or hide lifecycle buttons. See [Renaming and Hiding Life Cycle Buttons](#).
- **Days to Preserve Unread Notifications:** Number of days to keep unread notifications before they are deleted automatically. The default is 7.
- **Days to Preserve Read Notifications:** Number of days to preserve read notifications before they are deleted automatically. The default is 1.

## Equipment

Paths to Pipeline Pilot protocols used for parsing file data from direct connection equipment, retrieving that parsed data, and controlling devices such as printers. Change these paths to point to your version of these protocols if you plan to customize them. When setting up Equipment Types that will interact with these protocols, you will need to configure them to pass parameters. See the *BIOVIA Foundation Hub Equipment Guide*, which is available with the installation files and at <https://www.3ds.com/support/>.

- **Control Instrument Protocol:** Pipeline Pilot protocol used to control instruments such as label printers or CDS systems.
- **Parse Instrument Protocol:** Pipeline Pilot protocol used to parse instrument data files.
- **Retrieve Measurement Protocol:** Pipeline Pilot protocol used to retrieve parsed instrument data.
- **Enable Equipment File Crawler:** Enables the File Crawler feature, which crawls file based equipment data.
- **Equipment File Crawler Schedule:** Schedule file crawling events.
- **Equipment File Crawler Protocol:** Pipeline Pilot protocol used to crawl file based equipment data.
- **Maximum Active Crawlers:** Limits the number of simultaneous active crawlers. Leave this blank for no limit.
- **Maximum File Age to be sent for Parsing:** Maximum age for files to be parsed. Leave blank for no limit.
- **Automatically Parse Equipment Work Items:** Whether to parse equipment work items automatically.
- **Number of Equipment Work Item Parse Jobs to run concurrently:** Maximum number of equipment work item jobs that can be parsed at the same time.
- **Number of slots reserved for Short Equipment Work Item Parse Jobs:** Number of slots reserved for short equipment work item parse jobs as defined by the field, Maximum Duration of Short Equipment Work Item Parse Jobs. Use this to maintain performance of the Pipeline Pilot server.
- **Maximum Duration of Short Equipment Work Item Parse Jobs:** Defines maximum length for a short equipment work item parse job.
- **Attempts on Equipment Work Item Parsing:** Number of parse job fails before the job goes into a FAILED state, which requires a manual trigger. The wait time for the first retry is 2 minutes, and increases exponentially to a maximum wait time of 32 minutes.
- **Pipeline Pilot Job Expiration:** Length of time job results are retained on the Pipeline Pilot server.

## Security

- **Allow Automatic User Provisioning:** Automatically add authenticated users from the IP. If you are deploying in a regulated environment (for example, GxP), it is recommended that you turn this option off.
- **Update User Account Data on Sign In:** Users are updated at sign in with the user information provided by the authentication provider (LDAP or 3D Passport). Otherwise, you can update them manually or by scheduled jobs. See [Managing Users](#) and [Synchronizing Users](#). See [Authentication Settings Matrix](#) for details about how authentication settings affect sign in for different scenarios.

**Note:** This is a system wide-setting. You can disable automatic updates at sign in for individual User Directories by leaving this option on and using the **Update User Account Data on Sign In** setting in **Admin and Settings > Settings > User Directories**. See [Managing User Directories](#).

- **Editable User Profile:** Allow users to edit their own profiles. See Deploying in a regulated environment.
- **Advanced Group Administration:** Allows permissions to be assigned directly to groups.

**Note:** Use Roles to manage permissions whenever possible instead of assigning permissions directly to groups.

- **Save use passwords for duration of session:** Store the users' passwords encrypted for use by other services for impersonation and delegation of credentials. Do not enable this option unless you need to delegate credentials.
- **SSL Truststore Policy:** Identifies how SSL certificates are managed.

### Messages

- **Maximum Job Workers:** Maximum number of concurrent job handlers allowed.
- **Quiet Period for Messages:** Time in milliseconds to wait before creating a new message for the same entity.
- **Expiration Time for Completed Messages:** Time in days to wait before removing completed messages.

### File Service Settings

- **Default Storage Type:** Choose how files should be saved when the Foundation Hub File Service API is used. The File Service API is used internally when measurements are created that have attachments of parsed data.
  - **Database:** Save files as an Oracle blob.
  - **File System:** Save files to a local network. If you choose this option, **File System File Path** is required.
  - **Amazon S3:** Save files to the Amazon S3 private cloud service. If you choose this option, **Amazon S3 Default Region**, **Amazon S3 Default Bucket**, **Amazon S3 Access Key**, and **Amazon S3 Private Key** are required.
  - **Minio:** Save files to a local private cloud powered by Minio. If you choose this option, **Amazon S3 Default Region**, **Amazon S3 Default Bucket**, **Amazon S3 Access Key**, **Amazon S3 Private Key**, and **Amazon S3 Endpoint** are required.
- **File System File Path:** The location where the files should be stored if **Default Storage Type** is set to *File System*.
- **Amazon S3 Default Region:** Required to connect to the bucket. Bucket names are unique per region but are not unique by themselves—a combination of region and bucket name to find the storage.

**Tip:** For a list of supported regions, see *Amazon Simple Storage Service (Amazon S3)* in <https://docs.aws.amazon.com/general/latest/gr/rande.html>.

- **Default Bucket:** Files saved to a cloud service are saved to this bucket.
- **Access Key:** Access key used to authenticate when saving files using a cloud service.
- **Private Key:** Private key used to authenticate when saving files using a cloud service.
- **Endpoint:** Combination of region and bucket used by Minio to access the storage.



## Samples

- **Round up decrements from source samples to nearest each:** There are two options for specifying how to round the decremented quantity of a source sample after a child sample has been collected or split from it. This only applies to a source sample whose Unit Type is “each.” This setting does not apply to any other Unit Type (for example, “g”). If Yes, the decremented quantity of the source sample will be rounded up to the next whole unit. If No, the decremented quantity of the source sample will be specified as a fraction of the unit.

Rounding Enabled	Parent Sample Quantity	Child Sample Quantity	Decrement Quantity of Source Sample
Yes	10 Box	1.5 Box	8 Box
No	10 Box	1.5 Box	8.5 Box
Yes	10 Box (1g/Box)	1.5 g	8 Box
No	10 Box (1g/Box)	1.5 g	8.5 Box
Yes	10 g	1.5 g	8.5 g
No	10 g	1.5 g	8.5 g

- **Hide Create Sample Button:** Disables the **Create Sample** button in the user interface.

## Tasks

- **Required Task Action Fields:** Identifies which fields are required in the form used to create a regular task request, and in the form used to create a review task. The following fields are provided in these forms:

- |                          |                                   |
|--------------------------|-----------------------------------|
| ■ request.requestedGroup | ■ createReviewTask.requestedGroup |
| ■ request.assignee       | ■ createReviewTask.assignee       |
| ■ request.dueDate        | ■ createReviewTask.dueDate        |
| ■ request.priority       | ■ createReviewTask.priority       |
| ■ request.instructions   |                                   |

Enter the names of the required fields, separated by a comma. For example, to require a value in both the **Group** and **Assignee** fields for both types of task actions, enter the following:

request.requestedGroup,request.assignee,  
createReviewTask.requestedGroup,createReviewTask.assignee.

BIOVIA recommends always making **Group** a required field, and requiring the same fields for both versions of the form.

- **Task Predecessor Lifecycle State:** Specifies the lifecycle state that a task must reach before any task that specifies it as a **Predecessor** can be started (executed):
  - **Completed/Released:** A task cannot start until its predecessor task has been either *Completed* or *Released*.
  - **In Progress:** A task can start as soon as its predecessor has started (is *In Progress*, *In Progress*, *Completed*, or *Released*.)

### Master Data

- **Signature Level:** Controls whether changes (including addition, cloning, edits, and deletion) of master data entities requires a signature. Off (**None**) by default. To enable it, select **Signature** and configure the [Master Data Change Signature Policy](#) to meet your needs.

### Auditing

- **Entity Access Logging:** Lists entities for which user access events must be logged. By default, no user access events are logged.

Viewing a tab or widget that lists entities does not result in access events for the listed entities. An access event occurs only when the user *selects* a specific entity. For example, viewing a list of tasks or samples in a widget or in a Task Plan tab does not result in access events for any of the listed tasks or samples, but selecting specific tasks or samples does result in access events for the selected entities.

If required, you can log access events for entities such as activities, tasks, samples, and task plans.

To do so, enter the names of each entity for which you require access logging, separated by commas and no spaces. Use lower-case for the entity names, and (usually) use the singular form of the name.

The following are some examples:

- **task**
- **sample**
- **sample,task**
- **runset**
- **runset,sample,task**
- **activities** (use the plural form)

#### IMPORTANT!

Be aware of the following if you choose to set up entity access logging:

- Logging user access events can negatively impact system performance. Consequently, logging access for an entity is advised *only* if you must do so to comply with legal requirements such as those defined by HIPAA.
- Although Foundation Hub can currently log user access events, you can view the logs only through the API or a Pipeline Pilot protocol. The history user interface does not yet include access log records.
- If you use Compose and Capture, you must use the Foundation Hub Application Settings for Capture to set up logging for that application to be consistent with how you have set it up for Foundation Hub.

### Renaming and Hiding Life Cycle Buttons

You can rename or hide the life cycle buttons for samples, tasks, activities, and other entities that support life cycle actions. To do so, you insert JSON format code in the Life Cycle Configuration field on the Foundation Hub Settings page.

### Editing the Life Cycle Configuration Setting

1. Open **Settings > Applications > Application Settings > Foundation Hub**.
2. Insert JSON code using the syntax shown below into the **Life Cycle Configuration** field:
  - Replace `<entity>` with the name of the affected entity. See below for the names of the lifecycle-driven entities.

- Replace `<button>` with name of the button to change.
- To change the button label, include the `displayName` property, replacing `<language code>` with the locale, for example `en` for English, and replacing `<text>` with the button label that you want to display.
- To hide the button, include the `hide` element, replacing `<True or False>` with `True`.

## JSON Code

### Supported Properties

The following properties are supported:

- `displayName`: Text to display for the button. The format is `"<language code>" : "<text>"`, where `<language code>` specifies that the text applies to that locale (for example, `en` indicates English).
- `hide`: Whether to hide the button. Value is `true` or `false`.

### Lifecycle-Driven Entities

All lifecycle-driven entities are supported:

- Activity
- Certificate
- CertificateResult
- Equipment
- EquipmentType
- ParameterTemplate
- Runset (Task Plan)
- Sample
- Specification (Activity Plan)
- Task

### Syntax

```
{
  "<entity>": [
    {
      "name": "<button>",
      "displayName": {
        "<language code>": "<text>"
      }
    },
    {
      "name": "<button>",
      "hide": "<true or false>"
    }
  ]
}
```

### Example

The following example shows two modifications to the life cycle buttons for Activity. The submit button is renamed "Send" for English locales, and the reopenforedit button is hidden.

```
{
  "Activity": [
    {
      "name": "submit",
      "displayName": {
        "en": "Send"
      }
    },
    {
      "name": "reopenforedit",
      "hide": true
    }
  ]
}
```

## Extended Properties

You can add properties to many Foundation Hub data objects by using the extended properties feature. Extended properties are organized into lists associated with the kind of data object they extend.

In addition to creating extended properties that apply to all instances of a data object, you can create conditional properties that apply to only a specific class of the data object. For example, when you define extended properties for Equipment Type, you can define properties to apply only when the Equipment Type has a class of Balance.

### Adding a List of Extended Properties to an Object

Extended properties are organized into lists ("bags") based on the data object that you associate them with. To create a new list of extended properties for a data object:

1. Open **Settings > Extended Properties**.
2. Click the **Add Property Bag** (plus) icon.
3. Type a **Name** and **Description** for this list of extended properties.
4. Select the **Associated Data Object** to extend.
5. If this extended property list is valid only for a specific vocabulary entry or class of the associated data object, choose that entry or class from the **Conditional Value** field. This field is available only for objects with classes, such as Equipment Type, which has Equipment Class, and objects with vocabulary entries, such as Material.

**Note:** You cannot associate the same data object with more than one conditional list of extended properties and one regular list of extended properties.

6. Under Properties, click **Add**, and then define each extended property:
  - Type a **Name**.
  - Select the **Property Data Type**. When you select some data types, additional related fields are added to the form. For example, if you select Vocabulary, a Vocabulary field is added. If you select Measurement or Quantity, the Unit Type, Unit Category, and Unit fields are added.

**Note:** Avoid selecting **Boolean** if you need users to be able to change the value to null. A Boolean can start with a null default, but after a user changes the value, nobody can change it back to null. If this could be an issue, create a **List** with three values, instead of creating a **Boolean**.

- To display this extended property to users with a name other than the one in the Name field, enter a **Display Label**.
  - To display hover text when users move the mouse pointer over this extended property, enter a **Description**.
  - If the data type is **List**, enter the values to use to populate the list.
  - If appropriate, select the check boxes for **Required**, **Read Only**, and **Allow Null**.
  - If appropriate, specify a **Default Value**.
  - Fill in any fields added to support the data type you selected.
7. To add more values, select the **Add Another** check box and then click **OK**. After adding all values, clear the check box and click **OK**.
  8. Use **Up** and **Down** to control the order of the list. Click the **Remove** icon to delete it, or double-click it to edit.

**Note:** The pages that list the objects of a specific type, such as Equipment Types, does not provide columns for extended properties. To view the extended properties, open one of the individual objects.

## Foundation Hub Configuration Settings

The Foundation Hub Configuration page provides access to server, database, certificates, and SPNEGO/Kerberos configuration.

Foundation Hub is configured as part of the installation process. If you are installing for the first time, see the *Foundation Hub Installation and Configuration Guide*.

**Note:** Additional Foundation Hub settings are available in **Settings > Applications > Application Settings > Foundation Hub**. See [Foundation Hub Application Settings](#).

### Restarting the Server

1. Open **Settings > Hub Configuration**.
2. Click **Restart Server**.

### Editing Configuration Settings

1. Open **Settings > Hub Configuration**.
2. Click **Edit**.
3. Make changes as needed.

### Web Server Configuration Settings

- **Host Name:** Provide the name of the host. The default is the fully qualified server host name.

**Note:** If the host name changes, you can edit this setting in the `app-config.groovy` file. See [Troubleshooting](#).

- **HTTP, HTTPS, and Shutdown Ports:** Change the ports if needed. The defaults are:
  - HTTP: 9954
  - HTTPS: 9953
  - Shutdown: 9955
- **Reverse Proxy HTTP and HTTPS URLs:** Provide the URL of the reverse proxy server that fronts this Foundation Hub application.

### Notes:

- For a load-balanced server, the URL contains the HTTP and HTTPS ports. For example:  
`<hub_lb_server>.mycompany.com:9953`  
`<hub_lb_server>.mycompany.com:9954`
- If you are configuring the load balancer to use HTTPS only, include the HTTPS URL in both the HTTPS URL and HTTP URL fields.
- Only specify if the Foundation Hub is installed behind a reverse proxy.

- **Session Inactivity Timeout:** Set the length of inactivity before a user session becomes locked and requires a password. Specify the time in days (d), hours (h), or minutes (m). The default is 30m.
- **Session Global Timeout:** Set the length of a session before the user must log in again regardless of inactivity. Specify the time in days (d), hours (h), or minutes (m). The default is 8h.
- **Validate Login Return Url:** Select this check box to allow redirects only to known endpoints after login. This is selected by default.
- **Environment Label:** Enter text to display in the toolbar to identify the environment (for example, Test or Production).
- **3DDashboard Hosted:** To use a 3DX Platform application header and color scheme for this installation of Foundation Hub, select this check box. This check box is unselected by default, and is intended for use only by Dassault Systèmes support personnel.

## Database Configuration Settings

- **Data Source URL:** Provide the JDBC URL to your Oracle database. On restart, Foundation Hub checks the schema status and creates the necessary schema and data modifications to prepare the database for use. If you are connecting to a previously used schema, only necessary modifications are made. The following is an example of an Oracle JDBC connection string: `jdbc:oracle:thin:@//<oracle server name>:<port>/<service name>`
- **Data Source Username and Password:** Provide the Oracle schema username and password in which Foundation Hub data is stored.

You must keep the Oracle database password up to date in Foundation Hub. It is recommended that you configure the Oracle database user so that the password does not expire. If you need to reset the Oracle password, see [Resetting Your Oracle Password](#).

**Note:** If the password expires or has been changed without following these instructions, you will need to manually update it in the configuration file. See [Chapter 11: Troubleshooting](#).

## Hazelcast Configuration Settings

Hazelcast is *not* enabled by default. Do not change the **Hazelcast Configuration** settings unless instructed by Support.

## Certificate Configuration Settings

- **Hub SSL Certificate Keystore Path:** Provide the path to the JKS format keystore file.
- **Keystore Password:** Include the password for the keystore or the path to the truststore JKS certificate file, which must contain the load balancer's public key.  
Leave this blank if the Foundation Hub server has a valid signed SSL certificate issued by a recognized Certificate Authority (CA).
- **Keystore Alias:** Provide the keystore alias. This is required if the specified keystore file has multiple certificate entries.

## Authentication Provider Settings

- Foundation Hub authentication: Set **Authentication** to **Hub**. For more information, see [Setting Up User Authentication](#).
- 3DPassport authentication: Set **Authentication** to **Passport**, set **Server Url** to the 3DPassport server, and then choose whether to use the **Primary Authenticator** option:
  - If you choose the **Primary Authenticator** option, users are directed to the 3DPassport - Login page.
  - If you leave **Primary Authenticator** unselected, users are directed to a sign in dialog box with an option to authenticate using Foundation Hub or 3DPassport.
 See [Setting Up User Authentication with 3DPassport](#).
- SPENGO/Kerberos authentication: Set **Authentication** to **Hub**, and choose **Enable Kerberos Authentication**, see [Configuring SPNEGO/Kerberos Authentication](#) and complete the following information:
  - **Service Principal:** Provide the full name of the service principal including the service type prefix (for example, HTTP/) and the server's fully qualified domain name (for example, HTTP/<server FQDN>).
  - **Realm Name:** Provide the name of the Kerberos realm. This is usually the server domain in upper-case.
  - **Key Tab Location:** Include the path to the keytab file you copied to the server.
 See "Managing User Directories" in the *Foundation Hub Admin Guide*.

## Application Monitoring Settings

Leave **Application Monitoring** settings disabled unless instructed by Dassault Systèmes Customer Support to turn them on.

- **Enable Melody:** Select the checkbox to diagnose issues such as performance and system load.
- **Enable Tomcat Http Access Logging:** Select the checkbox to log the remote host/IP address URL and response codes. Logs are created in <hub\_install>/logs. See [Foundation Hub Logging](#).

## Resetting Your Oracle Password

You must keep the Oracle database password up to date in Foundation Hub. It is recommended that you configure the Oracle database user so that the password does not expire. If you need to reset the Oracle password, follow these instructions.

**Note:** If the password expires or has been changed without following these instructions, you will need to manually update it in the configuration file. See [Chapter 11: Troubleshooting](#).




1. Open **Admin and Settings > Hub Configuration** and click **Edit**.
2. Set the **Data Source Password** field to the *new* Oracle schema password.
3. Click **Save**. **Do not** close the browser and **do not** click **Restart Server**.
4. In Oracle, set the Foundation Hub schema owner's *new* password and commit.
5. On the **Admin and Settings > Hub Configuration** page, click **Restart Server**.

## Inventory System Connections

You can connect Foundation Hub with an inventory system so that users can import material and container information from it. You can also allow Capture Hub users to register batches of materials in inventory. To connect an inventory system, you must first add the inventory system definition and then map data fields in the system to data fields in Foundation Hub.

### Adding an Inventory System

Foundation Hub supports active connections to one or more CISPro inventory systems. However, only one active connection can be added as a registered application; additional active connections must be configured as external inventory systems.

1. Determine whether the application is registered with this Foundation Hub.
  - Open **Settings > Applications**, and verify the application information. For more information, see [Installed Application Settings](#).
  - If you need to register your CISPro inventory system, see the *CISPro Installation Guide*.
2. Open **Settings > Inventory Systems**.
3. Click  **Add Inventory System**.
4. Enter a unique **Name**.
5. If the inventory system is registered with this Foundation Hub, select the **Application**.
6. If the inventory system is registered with a different Foundation Hub or is not registered with Foundation Hub, select the **External Inventory** check box. Then, enter the **Ssl Root Url** in this format:  
*<inventory system URL>:<Foundation Hub URL where it is registered>*  
Example (registered with a different Foundation Hub):  
`https://my-cispro2.com/CISPro:https://my-foundationhub2.com/foundation/hub`  
Example (not registered with Foundation Hub):  
`https://my-cispro2.com/CISPro`
7. Click **OK**.
8. Click the row to open the inventory system definition.
9. Click  **Edit** and edit the properties, as needed. Click  **Submit** to save your changes.
  - If the inventory system is registered with a different Foundation Hub, you must enter the **Service Account User** and **Service Account Password** used to access the system.
  - To allow Capture Hub users to register material batches during recipe executions, clear the **Read Only** check box.




**Note:** Material registration from Capture Hub is supported only for the CISPro inventory system that is registered with this Foundation Hub. The **Can Register** column in the property map indicates which properties a Compose author can use in a recipe to send values to the inventory system.

- If you plan to connect more than one active inventory system, consider setting the **Search Order**. This property determines priority of search results when importing materials and the search order when scanning container barcodes. A lower value takes priority (for example, 1 is searched before 2).
10. Complete the property map to define how data is imported. See [Mapping Properties for Imported Materials and Containers](#).
  11. Make sure the inventory system is active. See [Activating and Inactivating Inventory Systems](#).

## Mapping Properties for Imported Materials and Containers

The property map defines how data is imported from an inventory system (the *source*) into Foundation Hub (the *target*). When you connect a CISPro inventory system, a standard property mapping is added by default. You can modify the default mappings and specify additional mappings (for example, for *extended properties*).

**To map properties for imported materials and containers:**

1. Open the inventory system definition.
2. Click  **Edit**.
3. In the **Property Map** grid, add or edit a row for each field you want to map, keeping in mind the following:
  - The **Target Entity** determines whether the property is mapped to a material or to a container in Foundation Hub.
  - To map an extended property field for materials, you must first select a **Material Class**.
  - The **Target Field** represents the field name in Foundation Hub.
  - The **Source Entity** and **Source Field** represent the object and field names, respectively, in the inventory system.

**Tip:** To map a custom extended property in CISPro, use the following format for the source field name: `customProperties.propertyName`.

- Only one source field can be mapped to a given target field. If you want to change an existing mapping, change the source entity and field.
- To set a default value for the target field, use either the **Default Value** or the **Default Unit** field. For example, ensure a required target field has a value even if the source field does not contain data.
- You must have mappings defined for these required fields.
  - For materials: **Inventory Lot ID, Inventory Material ID, Inventory Material Name, Material Part Number, CAS Number**.
  - For containers: **Barcode, Inventory Lot ID**.



**Note:** An error message appears if you try to save a property mapping that does not include the required fields.

4. When you are finished mapping properties, click  **Submit**.

### Activating and Inactivating Inventory Systems

An inventory system must be active for users to search it and import material and container information from it. When you add a CISPro inventory system as a registered application, it is active by default. You might want to inactivate an inventory system to prevent users from importing materials while still maintaining a record of the inventory system definition and property mappings. You can also activate a system that has previously been inactivated.

**To activate or inactivate an inventory system:**

1. Open the inventory system definition.
2. Click  **Edit**.
3. Clear the **Active** check box to inactivate it, or select the **Active** check box to activate it.
4. Click  **Submit**.

### Legal Notices (Personal Data Protection)

The **Settings > Legal Notices** page allows you to define and manage a legal notice, such as Personal Data Protection policy, that you require your users to accept before signing in. For details about using this feature, see the *BIOVIA Personal Data Protection Administration Guide*, which is included with the documentation available with the Foundation Hub installation files.

### Software License Files

Licenses are made available when the software is purchased. If you need assistance with licenses, contact Dassault Systèmes Customer Support.

You can add, delete, and view license files for your deployment from **Admin and Settings > Settings > License Files**. Although you are prompted for license files when installing your deployment, your organization may purchase new licenses to activate additional functionality, or you may need to update expired licenses.

From this area, you can also view properties of existing license files such as whether it is valid or when it expires. You can also inspect the license file itself.

**Note:** Do not use this feature to add licenses for Pipeline Pilot applications such as Insight or the registration products. See the documentation for those applications for more information.

### Supported Applications

Foundation Hub license management is supported for the following applications and features:

- Foundation Hub
- Equipment and Instrument features
- Compose
- Capture
- Workbook
- Experiment

### Adding a License File

Licenses are cumulative. For example, if you have a Capture license that expires 31-Dec-2020 and then add another license file that has a Workbook license that expires 31-Dec-2021, Foundation Hub will have

current licenses for both Workbook and Capture. If you add another Capture license that expires 31-Dec-2023, you still have licenses for Workbook and Capture but the Capture license is extended by three years. If you then add a license for Capture that expires 30-Jun-2020 there is no effect because there is an existing Capture license with a later expiration.

1. Open **Admin and Settings > Settings > License Files**.
2. Click the **Add License File** (plus) icon.
3. Click **Browse...** and navigate to and choose the License File.

## Deleting a License File

1. Open **Admin and Settings > Settings > License Files**.
2. Check the box for the License File you want to delete.
3. Click the **Delete License File** (minus) icon.

## Viewing Existing Licenses


1. Open **Admin and Settings > Settings > License Files**.
2. Under **Related Items**, click **Licenses**.

## Updating an Expired License

A valid license is required to access the Admin and Settings page. If your license is expired, you cannot access the **License Files** page to add an updated license file. You must manually copy the license file to the installation directory and restart the Foundation Hub service.

1. Navigate to the installation folder for Foundation Hub. For example: C:\Program Files\BIOVIA\Foundation\hub.
2. Remove the expired license file (.lic) from the installation folder and from the imported\_licenses folder if it exists.
3. Copy the updated license file to the installation folder.
4. Restart the *BIOVIA Hub Server <version>* service. See [Managing the Foundation Hub Service](#).
5. Repeat for each Foundation Hub server node if you are working with a load-balanced environment.

## Application Links

Foundation Hub includes a  feature that users can use to access BIOVIA applications, Admin and Settings, and any other applications or websites you want to provide links to.

### Tips:

- You can click the links in the **Application** column to test them.
- The links in the **Permission Required** open the detailed view where you can also edit them.

## Adding Links

1. Open **Settings > Links**.
2. Click the **Add Link** (plus) icon.

3. Set the following:
  - **Name:** Link text
  - **URI:** URI for the link. This is usually a full path relative to the application. For external applications, provide a full URL.
  - **Icon:** Base64 encoded icon for the link. Use the following format:  
data:image/png;base64,<encoded icon image>
  - **Permission Required:** Users with this Permission will have access to the link.
  - **Status URI:** URI to display a small status with the icon such as a count.

## Log Files

Open **Settings > Logging**. The logging page allows you to download log files and control the level of logging for Foundation Hub. See [Foundation Hub Logging](#).

## Trusted Certificates

You can manage public certificates for trusted sites in your Foundation Hub server trust store using the Trusted Certificates page.

**To add a trust store certificate:**

1. Open **Admin and Settings > Settings > Trusted Certificates**.
2. Click the **Add Trust Store Certificate** (plus) icon.
3. Paste the certificate text of the trusted site including the -----BEGIN CERTIFICATE----- and -----END CERTIFICATE----- lines and the text between them. You can also specify an HTTPS URL that points to a trusted site.
4. Click **OK**.

## User Directories

You can configure user authentication using LDAP, SPNEGO/Kerberos, or 3DS Passport. See the Foundation Hub Installation and Configuration Guide for details.

### Managing LDAP User Directories

You can set up user directories to sync with one or more external directories to manage user authentication.

**To edit a User Directory:**

1. Open **Admin and Settings > Settings > User Directories**.
2. Open the user directory to open it.
3. Click **Edit**.

### Synchronizing LDAP Users

You must create the user directory and then edit it to set up a user synchronization schedule.

**Note:** **Applications > Application Settings > Foundation Hub > Allow Automatic User Provisioning** must be enabled in order for the synchronization to be able to add new user accounts.

## Setting the Synchronization Schedule

1. Open **Admin and Settings > Settings > User Directories**.
2. Click the user directory you just created to open it.
3. Click **Edit**.
4. Choose whether to **Enable User Synchronization on Schedule**.
5. Set the **Schedule**:
  - **Days of the Week**: Choose which days of the week to synchronize.
  - **Hours**: Specify which hours of the day to synchronize. The time should be whole numbers based on a 24-hour clock. Enter a comma-delimited list to specify multiple hours. For example, 2, 14 would schedule a synchronization for 2 AM and 2 PM.
  - **Minutes Past Hour**: Specify the minutes past the hour you want to run the synchronization. Keep this set to 0 to schedule it to run at the beginning of the hours specified. Use a comma-delimited list to specify multiple runs per hour.
6. Choose **Delete Missing Users** to delete users that are in the Foundation Hub database, but no longer in the external user directory.
7. Set the **Page Size** to the batch size of users to synchronize per request. See your user directory help to view the request limit.
8. Click **Save**.
9. Manually synchronize the users for the first time.

## Manually Synchronizing the Users

Once you have set the synchronization schedule, you can manually synchronize for the first time and any time thereafter as needed. Note that the **Synchronize Users Now** button is not exposed until you have set the synchronization schedule.

1. Open **Admin and Settings > Settings > User Directories**.
2. Click the user directory you just created to open it.
3. In User Synchronization, click **Synchronize Users Now**.

## Chapter 9:

# Managing the Foundation Hub Service

---

You can manage the Foundation Hub service by using the Windows Services console or Foundation Hub command-line tools.

### Windows Services Console

You can use the Windows Services console to view services that run as Windows services, to identify the logon account to use for each such service, and to start, stop, and restart services. Foundation Hub includes a service named *BIOVIA Hub Server <version>*.

### Foundation Hub Command Line Tools

Command-line tools provide the same features as the Windows Services Console. However, the command line also simplifies the process of customizing the service memory settings and start-up parameters.

#### ■ Start the service

- **Windows:** <hub\_install>\bin\startService.bat
- **Linux:** sudo bin/daemon.sh start

#### ■ Stop the service

- **Windows:** <hub\_install>\bin\stopService.bat
- **Linux:** sudo bin/daemon.sh stop

#### ■ Restart the service

- **Windows:** <hub\_install>\bin\restartService.bat
- **Linux:** sudo bin/daemon.sh restart

# Chapter 10:

## Uninstalling Foundation Hub

---

**Note:** If you are using Foundation Hub as your authentication server in Pipeline Pilot, be sure to change your authentication settings in the Pipeline Pilot Admin Portal on the **Security > Authentication** page.

### Windows

Navigate to the Foundation Hub installation directory and run the uninstaller:

```
uninst.exe
```

The uninstaller will stop the Foundation Hub services, but will leave the configuration and log files.

### Linux

1. Stop the service:

```
daemon.sh stop
```

2. Uninstall the service as a root user:

```
sudo daemon.sh uninstall
```

3. Remove the installation directory:

```
rm -rf installDir
```

4. Remove the following:

```
rm /etc/init.d/Hub  
rm /etc/rc.d/rc*.d/Hub
```

# Chapter 11:

## Troubleshooting

---

Foundation Hub generates standard log files that you can use to troubleshoot issues with the installation. See the following for tools that are available with Foundation Hub for troubleshooting purposes:

- [Foundation Hub Logging](#)
- [Using PuTTY to Test Connectivity](#)

### General Issues

#### My license expired and I cannot access the Admin and Settings page to update it

**Problem:** A valid license is required to access the Admin and Settings page. If your license is expired, you cannot access the **License Files** page to add an updated license file.

**Solution:** You must manually copy the license file to the installation directory and restart the Foundation Hub service.

Licenses are made available when the software is purchased. If you need assistance with licenses, contact Dassault Systèmes Customer Support.

1. Navigate to the installation folder for Foundation Hub. For example: C:\Program Files\BIOVIA\Foundation\hub.
2. Remove the expired license file (.lic) from the installation folder and from the imported\_licenses folder if it exists.
3. Copy the updated license file to the installation folder.
4. Restart the *BIOVIA Hub Server <version>* service. See [Managing the Foundation Hub Service](#).
5. Repeat for each Foundation Hub server node if you are working with a load-balanced environment.

#### Load-balanced Pipeline Pilot nodes are no longer registered with Foundation Hub after upgrading Pipeline Pilot

Check for missing load-balanced Pipeline Pilot nodes. If you are upgrading from 2017 to 2018 and have load-balanced Pipeline Pilot servers, there is a known issue that may have caused existing nodes registered with the installation to be removed:

1. Navigate to **Admin and Settings > Settings > Applications**.
2. Find the row for **Pipeline Pilot**, click the link in the **Installation** column, and check for a list of your load-balanced Pipeline Pilot nodes.
3. If they are missing, click **Edit**, click **Add** under **Nodes**, and then add the fully qualified URL (HTTP) and fully qualified secure URL (HTTPS) for each Pipeline Pilot node.



## Cannot start the Foundation Hub because the Oracle password has expired

1. Make a backup of the Foundation Hub configuration file:  
`<hub_install>\conf\app-config.groovy`
2. Make the following changes to the original configuration file:
  - Change the Data Source Password to the new Oracle Password.
  - Comment out the line that references `.passwordEncryptionCodec`.
3. Save the file and restart Foundation Hub.
4. In Foundation Hub, open **Admin and Settings > Hub Configuration** and click **Edit**.
5. Set the **Data Source Password** field to the new Oracle schema password again and **Save**. Saving will re-enable encryption.

Also If you know in advance that you will be resetting your Oracle schema password, see [Resetting Your Oracle Password](#).

## Equipment and instrument features are not enabled as expected

Check that you have a license for equipment and instrument functionality in **Settings > License Files**. If you add a license file, you *do not* need to stop or restart the Foundation Hub service.

Licenses are made available when the software is purchased. If you need assistance with licenses, contact Dassault Systèmes Customer Support.

## The Hostname for the installation machine has changed

You need to update the `rootUrl` and `sslRootUrl` settings in the following file:

`<hub_install>/conf/app-config.groovy`

## Foundation Hub does not start, no error or redirect

There may be no access to the database because the credentials or URL changed, or there may be a port conflict. Check `hub.log` to confirm. See [Foundation Hub Logging](#).

- To resolve a database connection string issue, see [Resolving an Invalid Database Connection String](#).

## Issues manually editing configuration files

See [Reverting Foundation Hub to the Default Configuration](#).

## Timeout while attempting to restart the Foundation Hub Server

Restart the server manually. See [Chapter 9: Managing the Foundation Hub Service](#).

## The log file was not updated or saved

Check that there is enough disk space. See *Foundation Hub System Requirements*.

## 3DPassport Issues

### Single log out not working when using 3DPassport authentication

The 3DPassport server and the Foundation Hub server must have a trust relationship set up between them. Include the SSL certificate for the Foundation Hub server in the 3DPassport server's Truststore, and include the SSL certificate for the 3DPassport server in the Foundation Hub server's Truststore.

### Directed to 3DPassport login page and cannot sign in

Authentication was set to Passport on the Hub Configuration page.

If you do *not* plan to use 3DPassport for authentication, update the Hub Configuration to not use Passport for Authentication:

1. Use the following URL to use local authentication instead of 3DPassport:

```
https://<hub_server>:9953/foundation/hub/security/auth?local=true
```

2. Navigate to **Admin and Settings > Settings > Hub Configuration** and set **Authentication** to **Hub**.

If you do plan to use 3DPassport for authentication, you need to create the user and/or set up permissions for the user account. See the Foundation Hub Installation Guide.

### User cannot sign in

Foundation Hub does not enforce unique usernames for multiple domains. If a user is not able to sign in, he or she may be using a username that is duplicated in a different domain. Ask the user to sign in while specifying the domain name. For example <domain>\username.

## Foundation Hub Logging

**Note:** Each node in a load-balanced deployment has its own log file. To figure out if an error occurred you must check all log files.

### Viewing Log Files

1. Open **Settings > Logging**.
2. Click **Download Current Log File** or **Download All Log Files**.
3. See <hub\_install>/logs/hub.log.

### Editing Logging Settings

To manage Foundation Hub application logging, navigate to **Admin and Settings > Settings > Logging** and click **Edit**:

- **Application Logging Level:** Choose from **Debug**, **Info**, and **Warn** to set the level of verbosity. It is recommended that you leave the level set to **Warn**.
- **Log SQL Statements:** Logs a line for each SQL statement executed against the database. This option is verbose. It is not recommended for normal operation.
- **Log SQL Statement Parameters:** Logs a line for each parameter of each SQL statement executed against the database. This option is extremely verbose. It is not recommended for normal operation.
- **Download Current Log File:** Retrieves the active hub.log file from the server.
- **Download All Log Files:** Retrieves the current hub.log file, all archived log files (up to a maximum of ten) as well as the most recent stderr and stdout log files.
- **Output Elapsed Times:** Includes the time elapsed for various operations such as GET, PUT, and POST. This provides profiling information.
- **Include System Information:** Includes the system information (for example, host operating system and Java system information).

## Tomcat Logging

Apache Tomcat logs additional information in its own log files:

- stdout (standard output)
- stderr (standard error)

## Tomcat Http Access Logging

You can turn on Tomcat Http Access Logging in **Settings > Hub Configuration > Enable Tomcat Http Access Logging**.

Foundation Hub will record all requests processed by the server. With this enabled the following information is written to an access\_log file in the log folder:

- Remote host name (or IP address if enableLookups for the connector is false)
- Date and time, in Common Log Format
- First line of the request (method and request URI)
- HTTP status code of the response
- User session ID
- Time taken to process the request, in milliseconds
- Time taken to commit the response, in milliseconds
- Current request thread name

**Note:** These log files are not cleaned up automatically. If you plan to use this feature, it is recommended that you include regular cleanup as part of your business process.

## Java Garbage Collection Logs

Timestamped logs for Java garbage collection are created in <hub\_install>/logs. These log files are not cleaned up automatically. It is recommended that you include regular cleanup as part of your business process.

## Using PuTTY to Test Connectivity

Installations of Foundation Hub include the PuTTY terminal emulator, which you can use to test network connectivity of equipment and devices.

## Running PuTTY

Access PuTTY here: <hub\_install>\util\putty\win64.

## Accessing PuTTY Documentation

See the PuTTY documentation: <https://www.chiark.greenend.org.uk/~sgtatham/putty/docs.html>.

## Resolving an Invalid Database Connection String

The Foundation Hub validates the Database Configuration during the initial save of the server configuration. Issues with the connection string (Datasource URL, Username, and Password) are reported to the UI and logged to <hub\_install>/logs/hub.log.

For example:

```
2015-02-10 10:14:58,203 [tomcat-exec-2] ERROR application.Foundation
HubConfigSingletonService - An error occurred establishing a database
connection. ORA-01017: invalid username/password; logon denied
```

After configuration, if the database credentials or the URL has changed and the Foundation Hub can no longer access the database, an error is reported to `hub.log`.

For example:

```
2015-02-10 11:45:09,313 [localhost-startStop-1] ERROR pool.ConnectionPool -
Unable to create initial connections of pool.

java.sql.SQLException: ORA-01017: invalid username/password; logon denied
```

This causes the *BIOVIA Hub Server <version>* service to fail to start.

You can change the username and URL values directly in `<hub_install>\conf\app-config.groovy`. However, the password is encrypted during the configuration step and additional steps are required.

### Changing the Database Password

1. Stop the *BIOVIA Hub Server <version>* service.
2. Edit `<hub_install>\conf\app-config.groovy` and comment out or remove the following line:  
`passwordEncryptionCodec='com.accelrys.platform.utils.DataSourceCodec'`
3. Change the encrypted **Data Source password** string to the actual plaintext password for the **Data Source** user.
4. Start the Foundation Hub and navigate to the **Administration > Foundation Hub Configuration** page.
5. Edit the configuration and re-enter the new password in the **Data Source Password** field.
6. Click **Save** and **Restart Server**.

The Foundation Hub server re-enables password encryption and writes the database password as an encrypted string to the `app-config.groovy` file.

### Reverting Foundation Hub to the Default Configuration

Run the following command:

```
<hub_install>\bin\resetConfiguration.bat
```

Or:

```
<hub_install>/bin/resetConfiguration.sh
```

The command performs the following steps:

1. Copy the current configuration files to `<hub_install>/conf/backup`.
2. Overwrite pre-existing backups.
3. Delete the existing configuration files, and replace them with template versions.

**Note:** After restarting, the browser is redirected to the Foundation Hub Configuration Page.

## Dassault Systèmes Support Resources

---

For additional resources or to contact Dassault Systèmes Customer Support, visit the Support portal:

<https://www.3ds.com/support/>

From this portal, you can:

- Call or email Dassault Systèmes Customer Support
- Submit a request
- Download installers
- Access hardware and software requirements
- Access Knowledge Base
- Access Communities and Twitter feeds