R-1
Deployment Platform

RepliWeb

# R-1
# Deployment Platform

# Web Service API Guide

**Software Version 5.0**

**For Windows/UNIX operating systems**

**May 11, 2011**

---

RepliWeb, Inc., 6441 Lyons Road, Coconut Creek, FL 33073
**Tel:** (954) 946-2274, **Fax:** (954) 337-6424
E-mail: info@repliweb.com, Support: http://support.repliweb.com

**Please direct correspondence or inquiries to:**
RepliWeb, Inc.
6441 Lyons Road
Coconut Creek, Florida 33073
USA

Telephone: (954) 946-2274
Fax: (954) 337-6424

| | |
|---|---|
| Sales & General Information: | info@repliweb.com |
| Documentation: | docs@repliweb.com |
| Technical Support: | http://support.repliweb.com |
| Website: | http://www.repliweb.com |

# Table of Contents

# 1. Overview

The R-1 Deployment Platform (R-1) is a cost effective, feature-rich web content and application deployment automation platform. Hundreds of enterprises and government organizations use R-1 to meet their mission-critical deployment needs. R-1 ensures that local and distributed servers, as well as complete datacenters, are synchronized with the most up-to-date content (files, web content, IIS settings, .NET assemblies, and COM+ components) and code.

R-1 Web Service enables organizations to incorporate R-1 functionality into in-house and third party products. *In future version, additional options will be available, like submitting jobs and making Center Management changes.*

This document assumes that the reader is familiar with the basic Web service concepts, including SOAP, WSDL, XML, and so on.

Web Service/WSDL terms such as *request*, *response*, and others are used frequently throughout this document. Syntax and sample code are provided in both C# and Java. With basic Web Service knowledge, you can use the sample code as a foundation for other programming languages.

**NOTE:** A good introduction to Web Service concepts can be found at:

www.oreillynet.com/lpt/a/webservices/2002/02/12/webservicefaqs.html

## Compatibility

For compatibility reasons, the Web Service identifies the Center version, that is, the WSDL version used to compile this Center (the version is included in the WSDL file).

If the Center version is higher than the Web Service version, the Web Service will deny access to the Web Service. This will prevent cases in which data sent by the Center is not processed due to an older server version. The opposite situation (new server, old Center) should be handled properly. In addition, all fields added to the new version (as well as optional fields) should be defined as optional.

There should be a clear distinction between changing a value and ignoring it. If a client does not want the server to modify a field, it should set the value of the field to NULL (or not set it at all). If, on the other hand, the client wants to empty a certain field, it should send an empty value indicator (empty string, empty array) and not NULL.

# Architecture

R-1 Web Service consists of a single SOAP Web Service called *R1WebService*.

The Web Service provides various 'data objects', such as `BasicCommand` and `JobInformation`. They only contain the data, which is accessed as properties.

*Currently, these properties enable you to retrieve the Center's audit information. In the future, additional R-1 functionality will become available.*

To query the data object, the Web Service provides the `AuditManager` class that is implemented as WSDL `PortTypes`.

The manager objects are as follows:

| WSDL PortType | Purpose |
|---|---|
| `SessionManager` | Logging in, authentication and logging out. |
| `AuditManager` | Retrieving audit information (changes made to Center Management settings and job commands, or R-1 system-initiated Scheduler settings). |

# 2. Getting Started

## Web Services Access

You can install R-1 Web Service and enable the client to access the Center by accessing the web service using predefined port 2837 with R-1 running the web service as a DLL. This is a fast connection; however it requires port 2837 to be opened in the firewall.

To use this connection in Secured mode, set the desired Center to connect using SSL through port 2837. This can be done via R-1 CLI's **`Set`** command or through GUI. For more information, refer to the R-1 User Guide.

## Developing Client Interfaces

Install Web Services Enhancements 3.0 for Microsoft .NET (WSE).
See http://msdn2.microsoft.com/en-us/webservices/Aa740663.aspx.

### Visual Studio 2005

**To add the R-1 Web Service reference to your Visual Studio 2005 solution:**

1. Open **Web References** → **Add Web reference**.
2. Enter the following URL:

   http://<server>:2837/R1?wsdl

   Where <**server**> is the R-1 Server host name. This host is only used to download the WSDL file. Once the client is built, it can access other R-1 Servers.

### Visual Studio 2008

**To add the R-1 Web Service reference to your Visual Studio 2008 solution:**

1. Open **Add Service References**.
2. Enter the following URL:

   http://<server>:2837/R1?wsdl

   Where <**server**> is the R-1 Server host name. This host is only used to download the WSDL file. Once the client is built, it can access other R-1 Servers.

# Eclipse

**To add the R-1 Web Service reference to your JAVA IDE:**

1. Make sure your IDE should support web service (WSDL) consumption. If necessary, download the relevant plug-in. See: http://www.eclipse.org/webtools/

2. In the package explorer, right-click your project and select **New → Other → Web Services → Web Service Client**

3. Enter the following URL:

   http://<server>:2837/R1?wsdl

   Where <**server**> is the R-1 Server host name. This host is only used to download the WSDL file. Once the client is built, it can access other R-1 Servers.

# 3. SessionManager Class

The `SessionManager` class establishes connection between a client application and the Center using the given credentials.

A class of type “session” which contains the session token (GUID) is returned by the Login method and should be passed as a parameter to be used when calling various R-1 Web Service methods such as a `GetAuditRecords` request.

A client can connect to multiple Centers simultaneously and call each Center separately. On the contrary, multiple clients can connect to the same Center. A client should maintain several session objects (one for each Center) and use the session according to the server it is currently connected to.

A **session** is created by the `Login` method, specified by the enum type `CredentialsType`:

- **Credentials** – The client should specify the user name, password and optional domain of the Center. Values entered are treated as plain text.

  **NOTE:** The Center hostname/IP address is specified in the URL.

- **ScrambledPassword** – The password specified is a scrambled password and will be decrypted internally.

## Preparing the SessionManager

The client should set the following properties of the `SessionManager` object:

- Set the URL property to the Center hostname or IP address, for example:

  ```
  SessionMgr.Url = "http://r1host:2837/R1";
  ```

  The URL defines the Center and how the Web Service will be accessed. For an explanation about accessing the Web Service, refer to Web Service Access above.

  **IMPORTANT:** The common name of the certificate used to establish a secure connection to the R-1 RDS service must be the same as the machine name on which it resides.

**NOTE**: The URL and Proxy (if used) properties should be set in *every* manager object that you create i.e. `SessionManager` or `AuditManager`.

# Types

## Credentials

### Properties

| Property | Description |
|---|---|
| `Domain` | For Windows users only. |
| `Password` | Windows password or virtual R-1 password. The password is transmitted to the Web Service in plain text. For encryption, the Web Service should be secured by SSL. |
| `Username` | Windows user name or virtual R-1 user name. |

## Session

This type is created by the `Login()` method. It represents an R-1 user session. The session's properties are read-only and should never be modified by clients.

### Properties

| Property | Description |
|---|---|
| `Token` | Identifies a Web Service session. |

# Methods

## Session.Login

Call the `Login` method to log in to a given remote Center. A session ID token (GUID) is returned. Provide this GUID when querying Center audit information.

*In future versions, additional options will be available, like submitting jobs and making Center Management changes.*

### Syntax (C#):

```
Session Login(CredentialsType, Credentials, ClientInfo)
```

### Request Parameters

| Parameter | Type | Description |
|---|---|---|
| `CredentialsType` | `CredentialsType Enum` | `CredentialsType.Credentials` or `CredentialsType.Scrambledpassword` |
| `Credentials` | `Credentials Enum` | For `CredentialsType.Credentials`, this parameter should contain the user name, password, and optional domain.<br>For `CredentialsType.ScrambledPassword`, this parameter should contain the R-1 credentials, with the encrypted password. The web service will internally decrypt the scrambled password on the server side.<br><br>**NOTE:** To use `CredentialsType=ScrambledPassword`, first encrypt the password using R-1 CLI's scramble command. Then, enter the encrypted password in the Password field. The encrypted password is transmitted to the Web Service as a scrambled password and decrypted on the server side.<br><br>The other credentials (Domain, Password and Username) are passed as plain text. |

| Parameter | Type | Description |
|---|---|---|
| `clientInfo` | `ClientInfo` | This parameter should contain the version of the WSDL file used to build the client application. The version is provided in the WSDL file, for example:<br>`<xs:element name="Version" type="xs:string" fixed="2010/09/29"/>`<br><br>**Note:** In C#, the `ClientInfo` parameter is filled automatically with the version defined in the `fixed` attribute, so you can simply pass `new ClientInfo()`. In Java, you should explicitly set the version property. |

## Response

| Parameter | Type | Description |
|---|---|---|
| `Session` | `Session` | A session object that should be used later with the `GetAuditRecords` request. The application should not directly get or set any of the session's properties, since they are for internal use only. |

# KeepAlive

Preserves a session created by the `Login` method. The default session timeout is 30 minutes.

## Syntax (C#)

```
void KeepAlive(Session s);
```

## Request Parameters

| Parameter | Type | Description |
|---|---|---|
| `Session` | `Session` | A session object created by `Login`. |

## Response

None

# Logout

Call `Logout` to terminate a session created by `Login`, thereby rendering it invalid for future access and freeing the resources held by the Web Service.

## Syntax (C#)

```
void Logout(Session s);
```

## Request Parameters

| Parameter | Type | Description |
|---|---|---|
| `Session` | `Session` | A session object created by `Login`. |

## Response

None

# Login Samples

The samples below demonstrate how to perform `Login` to a Center located on machine **r1comp**.

The client application should assign the Web Service's URL to the required host. For example, if the Web Service is installed on machine 10.0.33.44, then the URL should be:

**http://10.0.33.44:2837/R1**

## C#

```
using System;
using WebServiceSamples.R1WebService;

try
{
        //SessionManager and AuditManager should be disposed after usage.
        using (SessionManager mySessionManager = new SessionManager())
        {
                string webSvcUrl = "http://r1server:2837/R1";
                mySessionManager.Url = webSvcUrl;

                Credentials myCredentials = new Credentials();
                myCredentials.Username = "User";
                myCredentials.Password = "7KRVS4F7Q54D1";
                myCredentials.Domain = "Domain";

                //Establish a session with R1 Web Service.
                Session mySession = mySessionManager.Login(CredentialsType.ScrambledPassword,
                                                           myCredentials,
                                                           new ClientInfo());

                try
                {
                        //....
                }
                finally
                {
                        //Disconnect the session:
                        if (mySessionManager != null && mySession != null)
                                mySessionManager.Logout(mySession);
                }
        }
}
catch (Exception exp)
{
        Console.WriteLine();
        Console.WriteLine("An Exception of type: {0} Was Thrown.", exp.GetType().FullName);
        Console.WriteLine(exp.Message);
        Console.WriteLine(exp.StackTrace);
    }
```

# Java

```
SessionManagerProxy smp = new SessionManagerProxy();

    //Set Session Manager webservice url
    String url = "http://localhost:2837/R1";
smp.setEndpoint(url);
Session mySession = new Session();

try {
    Credentials myCred = new Credentials();
    myCred.setUsername("administrator");
    myCred.setDomain("");
    myCred.setPassword("123456");

        ClientInfo myClientInfo = new ClientInfo();
    myClientInfo.setVersion("2010/07/19");    // See IMPORTANT note below.

    mySession = smp.login(CredentialsType.Credentials, myCred, myClientInfo);
} catch (Exception e) {
    e.printStackTrace();
}
finally{
    //logout
    try {
        smp.logout(mySession);
        System.out.println("logout done");
      } catch (RemoteException e) {
        e.printStackTrace();
      }
}
```

**IMPORTANT:** The version date in `clientInfo.setVersion("2010/07/19");` changes from time to time. To find the current version date, browse to **http://<IP of machine>:2837/R1?WSDL** (the WSDL file) and locate the `ClientInfo` element:

```
<xs:element name="ClientInfo">
<xs:complexType>
    <xs:sequence>
        <xs:element name="Version" type="xs:string" fixed="2010/07/19"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

The value of the `fixed` attribute is the current version.

# 4. AuditManager Class

The `AuditManager` object has the GetAuditRecords method. Use this method to query a Center's audit information.

**NOTE:** Since a Center's Audit information can take up a large amount of space, `GetAuditRecords` is designed to query "chunks" of Audit information based on filters rather than to return all available information.

Audit information can be queried in one session using one or more `GetAuditRecords` calls, or over multiple calls called over different sessions.

When querying information over multiple calls, it is recommended to specify the QueryContext member in order to "remember" what information has been queried in previous call so as to prevent duplication.

For a graphical display of the class hierarchy, refer to the A: Class Hierarchy appendix.

# Set Types

The following classes are included in the GetAuditRecords request. Classes and enums which are specific to certain elements are documented together with their respective elements.

## Class Hierarchy

The following classes are used to set audit requests in the `AuditManager` object. For a graphical display of the hierarchy, refer to the Set Class Hierarchy appendix.

| Class | Description |
|---|---|
| `AuditFilter[]` | Class array specifying the audit filtering criteria. |
| `AuditFilter` | Included in `AuditFilter[]`. Defines which command objects to query. |
| `IssuerInformation` | Included in `AuditFilter`. Defines command issuer's information. Used to filter commands based on the issuer's details. |
| `CommandFilter` | Included in `AuditFilter`. Defines the `CommandType` (Jobs, Management, or Scheduler), and specific commands to query. |
| `JobInformation` | Included in `AuditFilter`. Defines the details of the job the command was performed on. |

## AuditFilter Class

This class is included in the GetAuditRecords method and enables you to filter the information you wish to query. All `AuditFilter` fields are optional, simply set to **NULL** any fields you wish to leave unfiltered.

Use the CommandFilter class, which is specified in `AuditFilter`, to specify the commands (and command types) you wish to query.

### Example

Specifying two filters:

```
/* Defining two filters */
AuditFilter[] filterArray = new AuditFilter[2];

/*Filter 1 – job command */
AuditFilter filter = new AuditFilter(1);
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Jobs;
filter.CommandFilter.JobCommands = new JobCommands[1];
filter.CommandFilter.JobCommands[0] = JobCommands.SubmitEdgeOverridesCommand;
```

```
/*Filter 2 – Center command */
AuditFilter filter = new AuditFilter(2);
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Management;
filter.CommandFilter.CenterCommands = new CenterCommands[1];
filter.CommandFilter.CenterCommands[0] = CenterCommands.UserDefinition;

/* setting a time range filter */
filter.FromTimeRange = new DateTime(2010, 5, 4, 7, 0, 0);
filter.FromTimeRangeSpecified = true;
filter.UntilTimeRange = DateTime.Now;
filter.UntilTimeRangeSpecified = true;
```

### Request Parameters

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| `IssuerInformation` | `IssuerInformation Object` | Specify the `IssuerInformation` details. | |
| `FromTimeRange` | `dateTime` | Specify the earliest (start) date to use for filtering commands. | |
| `UntilTimeRange` | `dateTime` | Specify the latest (end) date to use for filtering commands. | |
| `CommandFilter` | `CommandFilter Object` | Specify the command type and specific commands to return. | |
| `Status` | `Status Enum` | (Flag) Specify the status of commands to return. | **Success**<br>**Failure** |
| `JobInformation` | `JobInformation Object` | Specify the `JobInformation` details. | |

## IssuerInformation Class

This class defines the connection credentials of the user who was connected to the Center at the time of the change. It can be used to filter commands according to issuer details, and it is returned in each AuditRecord object, providing the issuer details.

All `IssuerInformation` fields are optional. When used to filter, simply set to **NULL** any fields you wish to leave unfiltered.

### Example

Specifying issuer details for a job command filter:

```
AuditFilter filter = new AuditFilter();
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Jobs;
filter.CommandFilter.JobCommands = new JobCommands[1];
```

```
filter.CommandFilter.JobCommands[0] = JobCommands.AllJobCommands;

filter.IssuerInformation = new IssuerInformation();
filter.IssuerInformation.Username = "replace with username";
filter.IssuerInformation.SourceIP = "replace with ip address";
filter.IssuerInformation.SourceMachine = "replace with machine name";
filter.IssuerInformation.Domain = "replace with domain name";
filter.IssuerInformation.Interface = Interface.Console;
```

### Request Parameters

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| `Interface` | `Interface Enum` | (Optional) (Flag) The interface from which the command was initiated. | **Console**<br>**CLI**<br>**API**<br>**Scheduler**<br>**Other** |
| `Domain` | `String` | (Optional) The connected user's domain. | |
| `Username` | `String` | (Optional) The connected user's user name. | |
| `SourceIP` | `String` | (Optional) The Center machine's source IP. | |
| `SourceMachine` | `String` | (Optional) The Center machine's hostname. | |

## CommandFilter Class

This class is specified in the `AuditFilter` class. It enables you to specify which commands to return.

### Example

Specifying a Confirm job command filter:

```
AuditFilter[] filterArray = new AuditFilter[1];

AuditFilter filter = new AuditFilter(0);
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Jobs;
filter.CommandFilter.JobCommands = new JobCommands[1];
filter.CommandFilter.JobCommands[0] = JobCommands.ConfirmCommand;
```

### Request Parameters

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| `CommandType` | `CommandType Enum` | (Flag) Specify the command type of the command(s) you wish to return. | **Jobs**<br>**Management**<br>**Scheduler** |

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| JobCommands[] | String | (Optional) Specify an array of the job commands whose information you wish to retrieve. | |
| CenterCommmands[] | String | (Optional) Specify an array of the Center commands you wish to retrieve. | |
| SchedulerCommands[] | String | (Optional) Specify an array of the Scheduler actions you wish to retrieve. | |

# Enum CommandType

This enum specifies which type of command(s) to query (Job, Center, and Scheduler commands). This section describes each command type and the command values each command accepts.

**NOTE:** CommandType accepts flags, in other words you can specify more than one command type for the same query. For example, **Jobs** and **Scheduler**.

## JobCommands Enums

Use these enums to query information relating to user-initiated commands executed on jobs (Submit, Modify, Delete, etc.).

### Example

```
AuditFilter[] filterArray = new AuditFilter[1];

AuditFilter filter = new AuditFilter();
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Jobs;
filter.CommandFilter.JobCommands = new JobCommands[1];
filter.CommandFilter.JobCommands[0] = JobCommands.AbortCommand;
```

### Request Parameters

**NOTE:** For an explanation of each enum's response, click the type's link in the **Response** column.

| Parameter | Description | Response |
|---|---|---|
| AllJobCommands | Specifies all job commands. | Relevant objects are returned, one for each job command, in the AuditRecord array. For more information, refer to the "get" AllJobCommands section. |

| Parameter | Description | Response |
|---|---|---|
| Abort | Specifies job Abort commands. | AbortCommand |
| Commit | Specifies Commit commands performed on Transactional Update jobs. | CommitCommand |
| Confirm | Specifies Confirm commands performed on Preview jobs. | ConfirmCommand |
| Delete | Specifies job Delete commands. | DeleteCommand |
| DemandSubmit | Specifies Demand Submit commands performed on On Demand jobs. | DemandSubmitCommand |
| Hold | Specifies job Hold commands. | HoldCommand |
| ModifyDistribution | Specifies Modify commands performed on scheduled Distribution jobs. | ModifyDistributionCommand |
| ModifyReplication | Specifies Modify commands performed on scheduled Replication jobs. | ModifyReplicationCommand |
| ModifyMsiReplication | (Relevant for R-1) Specifies Modify commands performed on scheduled MSI Deployment Replication jobs. | ModifyMsiReplicationCommand |
| ModifyMsiDistribution | (Relevant for R-1) Specifies Modify commands performed on scheduled MSI Deployment Distribution jobs. | ModifyMsiDistributionCommand |

| Parameter | Description | Response |
|---|---|---|
| ModifySdp | (Relevant for ROSS) Specifies Modify commands performed on scheduled Solution Deployment Replication jobs. | ModifySdpCommand |
| ModifySdpDistribution | (Relevant for ROSS) Specifies Modify commands performed on scheduled Solution Deployment Distribution jobs. | ModifySdpDistributionCommand |
| ModifyDistributionOverrides | Specifies Modify commands performed on scheduled Distribution jobs. | ModifyDistributionOverridesCommand |
| Reinitialize | Specifies Reinitialize commands performed on Continuous Update jobs. | ReinitializeCommand |
| Resume | Specifies Continue commands performed on held jobs. | ResumeCommand |
| ResubmitDistribution | Specifies Resubmit commands performed on scheduled Distribution jobs. | ResubmitDistributionCommand |
| ResubmitReplication | Specifies Resubmit commands performed on scheduled Replication jobs. | ResubmitReplicationCommand |
| ResubmitMsiReplication | (Relevant for R-1) Specifies Resubmit commands performed on scheduled MSI Deployment Replication jobs. | ResubmitMsiReplicationCommand |
| ResubmitMsiDisribution | (Relevant for R-1) Specifies Resubmit commands performed on scheduled MSI Deployment Distribution jobs. | ResubmitMsiDisributionCommand |

| Parameter | Description | Response |
|---|---|---|
| ResubmitSdp | (Relevant for ROSS) Specifies Resubmit commands performed on scheduled Solution Deployment Replication jobs. | ResubmitSdpCommand |
| ResubmitSdpDistribution | (Relevant for ROSS) Specifies Resubmit commands performed on scheduled Solution Deployment Distribution jobs. | ResubmitSdpDistributionCommand |
| RollbackDistribution | Specifies Rollback commands performed on Distribution jobs. | RollbackDistributionCommand |
| RollbackReplication | Specifies Rollback commands performed on Replication jobs. | RollbackReplicationCommand |
| RollbackDistributionOverrides | Specifies Rollback commands performed on the Edge Override settings of Distribution jobs. | RollbackDistributionOverridesCommand |
| SubmitDistribution | Specifies Submit commands performed on the Edge Override settings of scheduled Distribution jobs. | SubmitDistributionCommand |
| SubmitReplication | Specifies Submit commands performed on Replication jobs. | SubmitReplicationCommand |
| SubmitMsiReplication | (Relevant for R-1) Specifies Submit commands performed on MSI Deployment Replication jobs. | SubmitMsiReplicationCommand |
| SubmitMsiDistribution | (Relevant for R-1) Specifies Submit commands performed on MSI Deployment Distribution jobs. | SubmitMsiDistributionCommand |

| Parameter | Description | Response |
|---|---|---|
| SubmitSdp | (Relevant for ROSS) Specifies Submit commands performed on Solution Deployment Replication jobs. | SubmitSdpCommand |
| SubmitSdpDistribution | (Relevant for ROSS) Specifies Submit commands performed on Solution Deployment Distribution jobs. | SubmitSdpDistributionCommand |
| SubmitEdgeOverrides | Specifies Submit commands performed on the Edge Override settings of scheduled Distribution jobs. | SubmitEdgeOverridesCommand |

## CenterCommands Enums

Use these enums to query changes made by users to a Center's Center Management configuration settings including configuration changes made to Backup & Restore, Security Rules, Scheduler, and Email.

### Example

```
AuditFilter[] filterArray = new AuditFilter[1];

AuditFilter filter = new AuditFilter();
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Management;
filter.CommandFilter.JobCommands = new CenterCommands[1];
filter.CommandFilter.JobCommands[0] = CenterCommands.ManageChangePasswordCommand;
```

### Request Parameters

**NOTE:** For an explanation of each enum's response, click the type's link in the **Response** column.

| Parameter | Description | Response |
|---|---|---|
| AllCenterCommands | Specifies all Center commands. | Relevant objects are returned, one for each job command, in the AuditRecord array. For more information, refer to the "get" AllCenterCommands section. |
| ManageAudit | Specifies changes made to Audit Trail settings. | ManageAuditCommand |

| Parameter | Description | Response |
|---|---|---|
| ManageBackupConfiguration | Specifies changes made to Backup Configuration settings. | ManageBackupConfigurationCommand |
| ManageBackupStart | Specifies settings of Backup executions. | ManageBackupStartCommand |
| ManageBackupAbort | Specifies Abort commands performed on running Backup processes. | ManageBackupAbortCommand |
| ManageChangePassword | Specifies user password changes. | ManageChangePasswordCommand |
| ManageDiskSpace | Specifies changes made to Disk Space settings. | ManageDiskSpaceCommand |
| ManageEraserHold | Specifies Hold commands performed on running Eraser processes. | ManageEraserHoldCommand |
| ManageEraserContinue | Specifies Continue commands performed on held Eraser processes. | ManageEraserContinueCommand |
| ManageEraserModify | Specifies changes made to the Eraser's SCAN FREQUENCY and Report Cleanup rate. | ManageEraserModifyCommand |
| ManageFailover | Specifies changes made to Failover settings. | ManageFailoverCommand |
| ManageGlobalExclude | Specifies changes made to the Center's Exclude Filter definitions. | ManageExcludeFilterCommand |
| ManageHosts | Specifies changes made to the Center's Hosts list. | ManageHostsCommand |
| ManageLicense | Specifies changes made to the Center's License. | ManageLicenseCommand |
| ManageLicenseRemoval | Specifies removal events of the Center's License. | ManageLicenseRemovalCommand |

| Parameter | Description | Response |
|---|---|---|
| ManageMail | Specifies changes made to the Center's Email configuration. | ManageMailCommand |
| ManageNotification | Specifies changes made to the default job notification settings. | ManageNotificationCommand |
| ManageRestoreAbort | Specifies Abort commands performed on running Restore processes. | ManageRestoreAbortCommand |
| ManageRestoreStart | Specifies settings of Restore processes. | ManageRestoreStartCommand |
| ManageRun | Specifies script settings that were executed using the Center's Run Command feature. | ManageRunCommand |
| ManageScheduler | Specifies changes made to the Scheduler settings. | ManageSchedulerCommand |
| ManageSSL | Specifies changes made to the Center's SSL authentication and encryption settings. | ManageSSLCommand |
| ManageUserRules | Specifies changes made to the Center's security rules. | ManageUserRulesCommand |
| ManageUserDefinitions | Specifies changes made to the Center's User definitions. | ManageUserDefinitionsCommand |

## SchedulerCommands Enums

Use these enums to query commands that were initiated by R-1 Scheduler.

R-1 Scheduler executes each command, action and operation that takes place on the Center. This includes internal R-1 operations such as putting a running job in recovery in the event of an error, purging archived jobs, or sending an On Completion notification email when a job completes successfully, as well as the Scheduler's execution of user-initiated commands like modifying Preferences, aborting running jobs, or making changes to Center Management configurations.

### Example

```
AuditFilter[] filterArray = new AuditFilter[1];
```

```
AuditFilter filter = new AuditFilter();
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Scheduler;
filter.CommandFilter.JobCommands = new SchedulerCommands[1];
filter.CommandFilter.JobCommands[0] = SchedulerCommands.SchedulerCommitCommand;
```

## Request Parameters

**NOTE:** For an explanation of each enum's response, click the type's link in the **Response** column.

| Parameter | Description | Response |
|---|---|---|
| `AllSchedulerCommands` | Specifies information related to all commands performed by R-1 Scheduler. | Relevant objects are returned, one for each job command, in the `AuditRecord` array. For more information, refer to the "get" `AllSchedulerCommands` section. |
| `SchedulerAbort` | Specifies information related to job Abort commands performed by R-1 Scheduler. | `SchedulerAbortCommand` |
| `SchedulerAutoRollbackDistribution` | Specifies information related to Rollback commands performed on Distribution jobs that were set to be automatically rolled back. | `SchedulerAutoRollbackDistributionCommand` |
| `SchedulerAutoRollbackReplication` | Specifies information related to Rollback commands performed on Replication jobs that were set to be automatically rolled back. | `SchedulerAutoRollbackReplicationCommand` |
| `SchedulerChainJobs` | Specifies information related to the chaining of jobs to the Exit status of completed jobs by R-1 Scheduler. | `SchedulerChainJobsCommand` |
| `SchedulerCommit` | Specifies information related to Commit commands performed on Transactional Updates jobs by R-1 Scheduler. | `SchedulerCommitCommand` |

| Parameter | Description | Response |
|---|---|---|
| SchedulerCompletionExitProcedure | Specifies information related to Exit Procedure commands performed on completed jobs by R-1 Scheduler. | SchedulerCompletionExitProcedureCommand |
| SchedulerDeleteJob | Specifies information related to Delete commands performed on jobs by R-1 Scheduler. | SchedulerDeleteJobCommand |
| SchedulerDemandSubmit | Specifies information related to Demand Submit commands performed on jobs by R-1 Scheduler. | SchedulerDemandSubmitCommand |
| SchedulerJobCompleted | Specifies information related to Exit status details of completed jobs. | SchedulerJobCompletedCommand |
| SchedulerJobNotification | Specifies information related to job On Completion/Early Warnings notifications sent by R-1 Scheduler. | SchedulerJobNotificationCommand |
| SchedulerHold | Specifies information related to Hold commands performed on jobs by R-1 Scheduler. | SchedulerHoldCommand |
| SchedulerModify | Specifies information related to Modify commands performed on jobs by R-1 Scheduler. | SchedulerModifyCommand |
| SchedulerContinue | Specifies information related to Continue commands performed on held jobs by R-1 Scheduler. | SchedulerContinueCommand |

| Parameter | Description | Response |
|---|---|---|
| SchedulerRollback | Specifies information related to Rollback commands performed on jobs by R-1 Scheduler. | SchedulerRollbackCommand |
| SchedulerSubmit | Specifies information related to Submit commands performed on jobs by R-1 Scheduler. | SchedulerSubmitCommand |
| SchedulerReinitialize | Specifies information related to Reinitialize commands performed on Continuous Update jobs. The Reinitialize command spawns a new instance of a Continuous Update job. | SchedulerReinitializeCommand |
| SchedulerTriggeredDemandSubmit | Specifies information related to the triggering of chained jobs by R-1 Scheduler. | SchedulerTriggeredDemandSubmitCommand |

# JobInformation Class

This class defines job details for Job and Scheduler command audit requests. All JobInformation fields are optional, simply set to **NULL** any fields you wish to leave unfiltered.

## Response

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| Application | Application Enum | The job's RepliWeb application. | **R1**<br>**ROSS** |
| Id | String | The job ID. | |
| Name | String | The job name, if specified. | |
| ParentId | String | For Scheduled job instances, the job ID of the instance's parent job. | |
| GroupId | String | For chained jobs, the chained job's group ID. | |
| EdgeName | String | The job's Edge(s). | |
| SourceDir | String | The job's source directory. | |

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| `TargetDir` | `String` | The job's target directorys. | |

## Example

Specifying a job command filter's job details:

```
AuditFilter[] filterArray = new AuditFilter[1];

AuditFilter filter = new AuditFilter();
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Jobs;
filter.CommandFilter.JobCommands = new JobCommands[1];
filter.CommandFilter.JobCommands[0] = JobCommands.AllJobCommands;

/* setting JobInformation filter settings */
filter.JobInformation = new JobInformation();
filter.JobInformation.Application = Application.R1;
filter.JobInformation.GroupId = "NULL";
filter.JobInformation.Id = "NULL";
filter.JobInformation.Name = "Weekly Updates";
filter.JobInformation.ParentId = "replace with job parent id";
```

# Get Types

`GetAuditRecords` returns a single `AuditInformation` class, containing the requested information. For a graphical display, refer to Get Class Hierarchy.

## Class Hierarchy

The following classes are used to get audit requests in the `AuditManager` object. For a graphical display of the hierarchy, refer to the Get Class Hierarchy appendix.

| Class | Description |
|---|---|
| `AuditInformation` | Class containing all get classes as well as the `QueryContext` string to be used in the next `GetAuditRecords` request to return the next chunk of queried audit information. |
| `AuditRecord[]` | Included in `AuditInformation`. Returns objects of the queried commands. |
| `AuditRecord` | Included in `AuditRecord[]`. Returns each command object's settings, command name, and general job and issuer details. |
| `BasicCommand` | Included in `AuditRecord`. Any of the commands defined as inheriting from Job/Center/Management commands can appear instead of `BasicCommand` and the user needs to query the type of object in order to properly read the fields. |
| `IssuerInformation` | Included in `AuditRecord`. Returns the issuer of the command (user and machine details). |
| `JobInformation` | Included for `JobCommand` and `SchedulerCommand` requests. Returns general information of the job on which the command was performed. |
| `CommandType` | Included in `AuditRecord`. Specifies the type of command (Job, Center Management and Scheduler). |

## AuditInformation Class

This class is returned as a response to a GetAuditRecords query and contains all the audit information that was requested by `GetAuditRecords`.

### Response

| Parameter | Type | Description |
|---|---|---|
| AuditRecord | AuditRecord[] | An array of the command objects returned, according to the filter criteria specified. |
| QueryContext | String | A string of the last audit record returned, to be used in the following GetAuditRecords request to return the next chunk of audit records. |

## AuditRecord Class

This class defines the command's details (command type, name, and issuer information).

### Response

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| CommandType | CommandType Enum | The command's type. | **Jobs**<br>**Management**<br>**Scheduler** |
| Command | BasicCommand Object | Object defining the command and general details of the audit record request and its display format. | |
| IssuerInformation | IssuerInformation Object | Specifies the connection credentials of the user who was connected to the Center at the time of the change. | |

# Command Types

Audit Trail Web Service API uses three classes for specifying which commands to request audit information for, JobCommand, CenterCommand, and SchedulerCommand. This chapter explains each command, the objects each command accepts, and provides examples.

**NOTE:** `CommandType` accepts flags, in other words you can specify more than one command types for the same query. For example: **Jobs** and **Scheduler**.

## Syntax

For job commands:

```
AuditFilter[] filterArray = new AuditFilter[1];

AuditFilter filter = new AuditFilter();
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Jobs;
```

For Center Management settings:

```
AuditFilter[] filterArray = new AuditFilter[1]; /* we can define more filters and fill in the array if we want – the example use one filter */

AuditFilter filter = new AuditFilter();
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Management;
```

For Scheduler commands:

```
AuditFilter[] filterArray = new AuditFilter[1]; /* we can define more filters and fill in the array if we want – the example use one filter */

AuditFilter filter = new AuditFilter();
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Scheduler;
```

## JobCommand Classes

These classes contain details of the queried job commands.

The following job commands can be queried:

- All Job Commands
- Abort
- Commit
- Confirm
- Continue
- Delete
- Demand Submit
- Hold
- Modify – Distribution
- Modify – Replication
- Modify – MSI Distribution
- Modify – MSI Replication
- Modify – Solution Distribution
- Modify – Solution Replication
- Modify – Distribution Edge Overrides
- Reinitialize
- Resubmit – Distribution
- Resubmit – Replication
- Resubmit – MSI Distribution
- Resubmit – MSI Replication
- Resubmit – Solution Distribution
- Resubmit – Solution Replication
- Rollback – Distribution
- Rollback – Replication
- Rollback – Distribution Edge Overrides
- Submit – Distribution
- Submit – Replication
- Submit – MSI Distribution
- Submit – MSI Replication
- Submit – Solution Distribution
- Submit – Solution Replication
- Submit – Distribution Edge Overrides

**NOTE:** These commands reflect R-1 job actions. Some commands do not return any parameters, such as Commit, Delete, and Continue.

### Response

An `AuditRecord` array is returned, containing the following parameters along with the requested details of the job commands themselves. For information about a command's relevant parameters, refer to the desired object below.

Additionally, BasicCommand, IssuerInformation and JobInformation objects are returned for each job command query.

| Parameter | Type | Description |
|---|---|---|
| `Command` | `JobCommand` | The command performed by the user. |
| `JobInformation` | `JobInformation` | Details about the job such as application (R-1/ ROSS), Group ID, job ID, name and parent ID. |

## Types

The following child classes contain information of job commands that were performed on Replication and Distribution jobs.

### All Job Command Classes

Contains information related to all changes made to the Center's jobs.

#### Syntax

```
AllJobCommands : JobCommand
```

#### Response

The `auditFilterArray` object is returned, containing objects of each user-initiated Job command. Each job command object includes the command's respective parameters, as follows:

### Abort Class

Contains information related to Abort commands performed on replication and distribution jobs.

#### Syntax

```
AbortCommand : JobCommand
```

#### Response

| Parameter | Type | Description |
|---|---|---|
| `Reason` | `String` | The reason for aborting the job, if specified. |

### Commit Class

Contains information related to Commit commands performed on Transactional Update jobs (replication and distribution).

#### Syntax

```
CommitCommand : JobCommand
```

#### Response

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| `IsUserOverride` | `Boolean` | Specifies if the job was committed by the user instead of the Commit condition. | **True**<br>**False** |

## Confirm Class

Contains information related to Confirm commands performed on replication and distribution Preview jobs via R-1 Console or Web UI.

### Syntax

```
ComfirmCommand : JobCommand
```

### Response

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| SourceFile | String | The source XML file. | |
| TargetFile | String | The target XML file. | |
| IsConfirmed | Boolean | The Preview job's current state. | **0 – Pending**<br>**1 – Confirmed** |

## Delete Class

Contains information related to Delete commands performed on replication and distribution jobs.

### Syntax

```
DeleteCommand : JobCommand
```

## Demand Submit Class

Contains information related to Demand Submit commands performed on replication and distribution jobs.

### Syntax

```
DemandSubmitCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| HoldUntil | dateTime | (Optional) The date and time at which the scheduled On Demand job's instance will be released from hold and continue its replication process. |

## Hold Class

Contains information related to Hold commands performed on replication and distribution jobs.

### Syntax

```
HoldCommand : JobCommand
```

### Response

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| HoldUntil | dateTime | (Optional) The date and time at which the job will be released from hold and continue its replication process. | |
| Mode | HoldMode Enum | The type of hold command. | **AdministrativeHold**<br>**HoldingPriority**<br>**WaitingForCommitCondition**<br>**HeldByUserCommand**<br>**OutOfExecutionTimeframe** |

## Reinitialize Class

Contains information related to reinitialized instances of Continuous Update jobs (replication and distribution).

### Syntax

```
ReinitializeCommand : JobCommand
```

## Resume Class

Contains information related to Resume commands performed on held replication and distribution jobs.

### Syntax

```
ResumeCommand : JobCommand
```

## Modify - Distribution Class

Contains information related to Modify commands performed on scheduled and Continuous Update Distribution jobs.

### Syntax

```
ModifyDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution job's settings. |

| Parameter | Type | Description |
|---|---|---|
| CoreConfiguration | String | The Edge Default settings. |
| PreDistributionConfiguration | String | The Pre-distribution Download job's settings. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Modify - Replication Class

Contains information related to Modify commands performed on Scheduled and Continuous Update jobs.

### Syntax

```
ModifyReplicationCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The job's settings. |

## Modify - MSI Deployment Distribution Class

(Relevant for R-1) Contains information related to Modify commands performed on scheduled MSI Deployment Distribution jobs.

### Syntax

```
ModifyMsiDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution job's settings. |
| CoreConfiguration | String | The Distribution job's Edge Default settings. |
| PreDistributionConfiguration | String | The Pre-distribution Download job's settings, such as Edge user credentials, target directories and job properties. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Modify - MSI Deployment Replication Class

(Relevant for R-1) Contains information related to Modify commands performed on scheduled MSI Deployment Replication jobs.

### Syntax

```
ModifyMsiReplicationCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The job's settings. |

## Modify - Solution Deployment Distribution Class

(Relevant for ROSS) Contains information related to Modify commands performed on scheduled Solution Deployment Distribution jobs.

### Syntax

```
ModifySdpDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution job's settings. |
| CoreConfiguration | String | The Distribution job's Edge Default settings. |
| PreDistributionConfiguration | String | The Pre-distribution Download job's settings, such as Edge user credentials, target directories and job properties. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Modify - Solution Deployment Replication Class

(Relevant for ROSS) Contains information related to Modify commands performed on scheduled Solution Deployment Replication jobs.

### Syntax

```
ModifySdpCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The job's settings. |

## Modify - Distribution Edge Overrides Class

Contains information related to Modify commands performed on the Edge Overrides settings of Scheduled and Continuous Update Distribution jobs.

### Syntax

```
ModifyDistributionOverridesCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Resubmit - Distribution Class

Contains information related to Resubmit commands that were performed on Distribution jobs.

### Syntax

```
ResubmitDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution job's settings. |
| CoreConfiguration | String | The Distribution job's Edge Default settings. |
| PreDistributionConfiguration | String | The Pre-distribution Download job's settings, such as Edge user credentials, source/target directories and job properties. |
| NewJobId | String | The resubmitted job's ID. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Resubmit - Replication Class

Contains information related to Resubmit commands performed on Replication jobs.

### Syntax

```
ResubmitReplicationCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The job's settings. |
| NewJobId | String | The resubmitted job's ID. |

## Resubmit - MSI Deployment Distribution Class

(Relevant for R-1) Contains information related to Resubmit commands that were performed on MSI Deployment Distribution jobs.

### Syntax

```
ResubmitMsiDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution job's settings. |
| CoreConfiguration | String | The Distribution job's Edge Default settings. |
| PreDistributionConfiguration | String | The Pre-distribution Download job's settings, such as Edge user credentials, target directories and job properties. |
| NewJobId | String | The resubmitted job's ID. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Resubmit - MSI Deployment Replication Class

(Relevant for R-1) Contains information related to Resubmit commands that were performed on Solution Deployment Replication jobs.

### Syntax

```
ResubmitMsiReplicationCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The job's settings. |
| NewJobId | String | The resubmitted job's ID. |

## Resubmit - Solution Deployment Distribution Class

(Relevant for ROSS) Contains information related to Resubmit commands that were performed on Solution Deployment Distribution jobs.

### Syntax

```
ResubmitSdpDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution job's settings. |
| CoreConfiguration | String | The Distribution job's Edge Default settings. |
| PreDistributionConfiguration | String | The Pre-distribution Download job's settings, such as Edge user credentials, target directories and job properties. |
| NewJobId | String | The resubmitted job's ID. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Resubmit - Solution Deployment Replication Class

(Relevant for ROSS) Contains information related to Resubmit commands that were performed on Solution Deployment Replication jobs.

### Syntax

```
ResubmitSdpCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The job's settings. |
| NewJobId | String | The resubmitted job's ID. |

## Rollback - Distribution Class

Contains information related to Rollback commands performed on Distribution jobs.

### Syntax

```
RollbackDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution Rollback job's settings. |
| ErbConfiguration | String | The Distribution Rollback job's Edge Default settings. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Rollback - Replication Class

Contains information related to Rollback commands that were performed on Replication jobs.

### Syntax

```
RollbackReplicationCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| RollbackConfiguration | String | The Rollback job's settings, such as user credentials, rollback policy and job properties like Run Options, On Exit procedure and Purge Policy. |

## Rollback - Distribution Edge Overrides Class

Contains information related to Rollback commands that were performed on the Edge Override settings of Distribution jobs.

### Syntax

```
RollbackDistributionOverridesCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Submit - Distribution Class

Contains information related to Submit commands that were performed on Distribution jobs.

### Syntax

```
SubmitDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution job's settings. |
| CoreConfiguration | String | The Distribution job's Edge Default settings. |
| PreDistributionConfiguration | String | The Pre-distribution Download job's settings, such as Edge user credentials, source/target directories and job properties. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Submit - MSI Deployment Distribution Class

(Relevant for R-1) Contains information related to Submit commands that were performed on MSI Deployment Distribution jobs.

### Syntax

```
SubmitMsiDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution job's settings. |
| CoreConfiguration | String | The Distribution job's Edge Default settings. |
| PreDistributionConfiguration | String | The Pre-distribution Download job's settings, such as Edge user credentials, source/target directories and job properties. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Submit - MSI Deployment Replication Class

(Relevant for R-1) Contains information related to Submit commands performed on Solution Deployment Replication jobs.

### Syntax

```
SubmitMsiReplicationCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The job's settings. |

## Submit - Solution Deployment Distribution Class

(Relevant for ROSS) Contains information related to Submit commands that were performed on Solution Deployment Distribution jobs.

### Syntax

```
SubmitSdpDistributionCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The Distribution job's settings. |
| CoreConfiguration | String | The Distribution job's Edge Default settings. |
| PreDistributionConfiguration | String | The Pre-distribution Download job's settings, such as Edge user credentials, source/target directories and job properties. |
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Submit - Solution Deployment Replication Class

(Relevant for ROSS) Contains information related to Submit commands performed on Solution Deployment Replication jobs.

### Syntax

```
SubmitSdpCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The job's settings. |

## Submit - Distribution Edge Overrides Class

Contains information related to Submit commands that were performed on the Edge Override settings of Distribution jobs.

### Syntax

```
SubmitEdgeOverridesCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| EdgeOverrides[] | EdgeOverride | An array of Edge Override objects containing the name and Edge Override settings of each overridden Edge. |

## Submit - Replication

Contains information related to Submit commands performed on Replication jobs.

### Syntax

```
SubmitReplicationCommand : JobCommand
```

### Response

| Parameter | Type | Description |
|---|---|---|
| JobConfiguration | String | The job's settings. |

## CenterCommand Classes

These classes contain details of the queried Center Management changes.

Changes made to the settings of the following Center Management features can be queried:

- All Center Commands
- Audit Trail
- Backup Configuration
- Backup – Abort
- Backup – Start
- Center Exclude Definitions
- Change Password
- Disk Space
- Email
- Eraser – Continue
- Eraser – Hold
- Eraser – Modify
- Failover
- Hosts
- License
- Notifications
- Restore – Abort
- Restore – Start
- Rules
- Run Command
- Scheduler
- SSL
- Users

**NOTE:** These commands reflect changes made to the Center's Center Management settings. Some commands do not return any parameters, such as Eraser Hold/Continue or Backup Start/Abort.

### Response

An `AuditRecord` array is returned, containing the following parameters along with the requested command's details of the job commands themselves. For information about a command's relevant parameters, refer to the desired object in the Types section below.

Additionally, BasicCommand and IssuerInformation objects are returned for each Center query.

| Parameter | Type | Description |
|---|---|---|
| `Command` | `CenterCommands Enum` | The change(s) made to the Center Management settings by the user. |

## Types

The following child classes contain information about changes made to Center Management settings.

### All Center Command Classes

Contains information related to all changes made to the Center's Center Management settings.

#### Syntax

```
AllCenterCommands : CenterCommand
```

#### Response

The `auditFilterArray` object is returned, containing objects of all user-initiated Center Management commands. Each returned Center Management object includes the command's respective parameters, as follows:

### Audit Trail Class

Contains information related to changes made to the Audit Trail settings.

#### Syntax

```
ManageAuditCommand : CenterCommand
```

#### Response

| Parameters | Type | Description |
|---|---|---|
| `AuditConfiguration` | `DiffObject` Object | Audit Trail settings. |

### Backup Abort Class

Contains information related to Abort commands performed on Center Backup jobs.

#### Syntax

```
ManageBackupAbortCommand : CenterCommand
```

### Backup Configuration Class

Contains information related to changes made to the Center's Backup settings.

#### Syntax

```
ManageBackupConfigurationCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| BackupConfiguration | String | The Center's Backup settings. |

## Backup Start Class

Contains information related to executions of Backup processes.

### Syntax

```
ManageBackupStartCommand : CenterCommand
```

## Center Exclude Filter Class

Contains information related to changes made to the Center's Exclude Filter specifications. The Exclude Filter enables users to define a list of files and directories to exclude from all future replication and distribution jobs.

### Syntax

```
ManageExcludeFilterCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| ExcludeConfiguration | DiffObject Object | Exclude Filter specification. |

## Change Password Class

Contains information related to changes made to the Center's Change Password settings.

### Syntax

```
ManageChangePasswordCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| ChangePasswordParameters | String | Change Password settings. |

## Disk Space Class

Contains information related to changes made to the Center's Disk Space settings and limits.

### Syntax

```
ManageDiskSpaceCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| Disk Configuration | DiffObject Object | The Disk Space limits and settings. |

## Email Class

Contains information related to changes made to the Center's Email configuration settings.

### Syntax

```
ManageMailCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| MailConfiguration | DiffObject Object | The Email configuration settings. |

## Eraser Continue Class

Contains information related to Continue commands performed on held Eraser processes.

### Syntax

```
ManageEraserContinueCommand : CenterCommand
```

## Eraser Hold Class

Contains information related to Hold commands performed on running Eraser processes.

### Syntax

```
ManageEraserHoldCommand : CenterCommand
```

## Eraser Modify Class

Contains information related to changes made to the Eraser's Scan Frequency and Report Cleanup rate.

### Syntax

```
ManageEraserModifyCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| EraserConfiguration | DiffObject Object | Eraser's Scan Frequency and Report Cleanup rate settings. |

## Failover Configuration Class

Contains information related to changes made to the Center's Failover settings.

### Syntax

```
ManageFailoverCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| FailoverConfiguration | DiffObject Object | Center's Failover settings. |

## Hosts Class

Contains information related to changes made to the Center's Hosts and their configuration settings.

### Syntax

```
ManageHostsCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| Hosts | DiffObject Object | Center's Hosts settings. |

## License Class

Contains information related to changes made to the RepliWeb application's license.

### Syntax

```
ManageLicenseCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| License | String | Application's license. |

## License Removal Class

Contains information related to this Center's license removals.

### Syntax

```
ManageLicenseRemovalCommand : JobCommand
```

## On Completion/Early Warnings Notifications Class

Contains information related to changes made to the Center's notification settings including the content and recipients of the email to send. On Completion refers to notifications sent upon job completion (Success/Abort/Error) while Early Warnings refers to notifications sent when jobs enter recovery for some reason.

### Syntax

```
ManageNotificationCommand : CenterCommand
```

### Response

| Parameters | Type | Description | Valid Values |
|---|---|---|---|
| Type | NotificationCommandType Enum | The notification type. | **OnCompletion**<br>**EarlyWarnings** |
| NotificationConfiguration | DiffObject Object | The relevant notification type's settings. | |

## Restore Abort Class

Contains information related to changes made to the Restore's Abort settings. The Restore option enables you to revert configuration changes made to the Center back to a specific Backup point.

### Syntax

```
ManageRestoreAbortCommand : CenterCommand
```

## Restore Start Class

Contains information related to changes made to the Center's Restore settings. The Restore option enables you to revert configuration changes made to the Center back to a specific Backup point.

### Syntax

```
ManageRestoreStartCommand : CenterCommand
```

### Return Parameters

| Parameters | Type | Description |
|---|---|---|
| BackupEntitiesXML | String | The RepliWeb entities which were backed up. For example, Scheduler settings, Security Rules, User definitions, jobs, templates, etc. |
| RestoreConfiguration | String | The Restore configuration settings. |

## Run Command Class

Contains information related to the command script and output of the commands executed using the Center's Run Command option.

### Syntax

```
ManageRunCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| CommandString | String | Run Command script. |
| Output | String | Command's output. |

## Scheduler Class

Contains information related to changes made to the Center's Scheduler settings for the Active, Archive and Submit servers, as well as administrative actions like holding all running jobs and allowing local administrators to view all jobs on the Center.

### Syntax

```
ManageSchedulerCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| SchedulerConfiguration | DiffObject Object | The Center's Scheduler and administrative action settings. |

## Security Rules Class

Contains information related to changes made to the Center's security rule settings.

### Syntax

```
ManageUserRulesCommand : CenterCommand
```

### Return Parameters

| Parameters | Type | Description |
|---|---|---|
| SecurityFile | DiffObject Object | Security Rule definitions. |
| SlLoginFile | DiffObject Object | Security Rule definitions. |
| SlSecurityFile | DiffObject Object | Security Rule definitions. |

## SSL Class

Contains information related to changes made to the Center's SSL authentication and encryption settings.

### Syntax

```
ManageSSLCommand : CenterCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| ControlFile | DiffObject Object | The SSL configuration between Console and Center. |
| DataFile | DiffObject Object | The SSL configurations between the Center and Edge(s). |

## Users Class

Contains information related to changes made to the Center's users and their configuration settings.

### Syntax

```
ManageUserDefinitionsCommand : CenterCommand
```

### Response

The UsersDefinition object is returned, it contains the following properties:

| Parameters | Type | Description | Valid Values |
|---|---|---|---|
| Domain | String | The user's domain, if specified. | |
| ErrorLog | String | Any errors related to the creation of this user. | |
| UserName | String | The user defined in the Center's User Definition list. | |

| Parameters | Type | Description | Valid Values |
|---|---|---|---|
| IsVirtualUser | Boolean | Specifies if the user specified in the UserName field is Virtual or Real. | **True**<br>**False** |

## SchedulerCommand Classes

These classes contain details of the queried Scheduler commands.

R-1 Scheduler executes each command, action and operation that takes place on the Center. This includes internal R-1 operations such as putting a running job in recovery in the event of an error, purging archived jobs, or sending an On Completion notification email when a job completes successfully, as well as the Scheduler's execution of user-initiated commands like modifying Preferences, aborting running jobs, or making changes to Center Management configurations.

The following Scheduler commands can be queried:

- All Scheduler Commands
- Abort
- Automatic Rollback - Distribution
- Automatic Rollback - Replication
- Commit
- Continue
- Delete
- Demand Submit
- Demand Submit Trigger
- Exit Procedure
- Hold
- Job Chain
- Job Completed
- Job Notifications
- Modify
- Reinitialize
- Rollback
- Submit

**NOTE:** These commands reflect changes made to the Center's Center Management settings. Some commands do not return any parameters, such as Scheduler Continue and Scheduler Modify.

### Response

An AuditRecord array is returned, containing the following parameters along with the requested command's details of the job commands themselves. For information about a command's relevant parameters, refer to the desired object in the Types section below.

Additionally, BasicCommand, IssuerInformation and JobInformation objects are returned for each Scheduler command query.

| Parameter | Type | Description |
|---|---|---|
| Command | SchedulerCommands Enum | The commands initiated by R-1 Scheduler. |

| Parameter | Type | Description |
|---|---|---|
| JobInformation | JobInformation Object | Details about the job such as application (R-1/ ROSS), Group ID, job ID, name and parent ID. |

## Types

The following child classes contain information of Scheduler commands.

### All Scheduler Command Classes

Contains information related to all commands performed by R-1 Scheduler.

#### Syntax

```
AllSchedulerCommands : SchedulerCommand
```

#### Response

The auditFilterArray object is returned, containing objects of all R-1 Scheduler-initiated commands. Each Scheduler command object includes the command's respective parameters, as follows:

### Scheduler Abort Class

Contains information related to Abort commands performed by R-1 Scheduler.

#### Syntax

```
SchedulerAbortCommand : SchedulerCommand
```

#### Response

| Parameters | Type | Description |
|---|---|---|
| Reason | String | The reason for the Abort operation. |

### Scheduler Auto Rollback - Distribution Class

Contains information related to R-1 Scheduler-initiated Rollback commands performed on replication jobs that were set to be automatically rolled back.

#### Syntax

```
SchedulerAutoRollbackDistributionCommand : SchedulerCommand
```

## Scheduler Auto Rollback - Replication Class

Contains information related to R-1 Scheduler-initiated Rollback commands performed on replication jobs that were set to be automatically rolled back.

### Syntax

```
SchedulerAutoRollbackReplicationCommand : SchedulerCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| RollbackConfiguration | String | Specifies the Rollback job's configuration settings. |

## Scheduler Commit Class

Contains information related to Commit commands performed on Transactional Updates jobs by R-1 Scheduler.

### Syntax

```
SchedulerCommitCommand : SchedulerCommand
```

### Response

| Parameters | Type | Description | Valid Values |
|---|---|---|---|
| IsUserOverride | Boolean | Specifies if the job was committed by the user instead of the Commit condition. | **True**<br>**False** |

## Scheduler Continue Class

Contains information related to Continue commands performed on held jobs by R-1 Scheduler.

### Syntax

```
SchedulerContinueCommand : SchedulerCommand
```

## Scheduler Delete Class

Contains information related to Delete commands performed on jobs by R-1 Scheduler.

### Syntax

```
SchedulerDeleteJobCommand : SchedulerCommand
```

## Scheduler Demand Submit Class

Contains information related to Demand Submit commands performed on jobs by R-1 Scheduler.

### Syntax

```
SchedulerDemandSubmitCommand : SchedulerCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| HoldUntil | dateTime | (Optional) The date and time at which the scheduled On Demand job's instance will be released from hold and continue its replication process. |
| ControlTrigger | JobControlTrigger object | Properties related to Job Chain-enabled jobs. |

## Scheduler Demand Submit Trigger Class

Contains information related to the triggering of chained jobs by R-1 Scheduler.

### Syntax

```
SchedulerTriggeredDemandSubmitCommand : SchedulerCommand
```

### Response

| Parameters | Type | Description | Valid Values |
|---|---|---|---|
| TriggeredJobs | String | Specifies the chained jobs triggered by R-1 Scheduler. | |
| JobsLocation | String | Specifies the jobs' locations. | **Center**<br>**Edge** |

## Scheduler Exit Procedure Class

Contains information related to Exit Procedure commands performed on completed jobs by R-1 Scheduler.

### Syntax

```
SchedulerCompletionExitProcedureCommand : SchedulerCommand
```

### Response

| Parameters | Type | Description |
|---|---|---|
| Image | String | The extension of the Exit Procedure file specified for execution . |
| Parameters | String | The Exit Procedure file's parameters. |

## Scheduler Hold Class

Contains information related to Hold commands performed on jobs by R-1 Scheduler.

### Syntax

```
SchedulerHoldCommand : SchedulerCommand
```

### Response

| Parameters | Type | Description | Valid Values |
|---|---|---|---|
| Mode | HoldMode | The type of hold command applied. | **AdministrativeHold**<br>**HoldingPriority**<br>**WaitingForCommitCondition**<br>**HeldByUserCommand**<br>**OutOfExecutionTimeframe** |
| HoldUntil | dateTime | (Optional) The date and time at which the scheduled On Demand job's instance will be released from hold and continue its replication process. | |

## Scheduler Job Chain Class

Contains information related to the chaining execution of jobs to the exit status of Job Chain-enabled jobs by R-1 Scheduler.

### Syntax

```
SchedulerChainJobsCommand : SchedulerCommand
```

### Response

| Parameters | Type | Description | Valid Values |
|---|---|---|---|
| TriggeredJobs | String | Chained jobs set to be triggered. | |
| JobsLocation | String | The jobs' location. | **Center**<br>**Edge** |

## Scheduler Job Completed Class

Contains information related to the Exit status of completed jobs.

### Syntax

```
SchedulerJobCompletedCommand : SchedulerCommand
```

### Response

| Parameters | Type | Description | Valid Values |
|---|---|---|---|
| ExitStatus | JobExitStatus | Job's exist status | **Success**<br>**Error**<br>**Warning**<br>**Aborted** |
| ExitMessage | String | Job's completion message. | |

## Scheduler Job Notifications Class

Contains information related to job On Completion/Early Warnings notifications sent by R-1 Scheduler.

### Syntax

```
SchedulerJobNotificationCommand : SchedulerCommand
```

### Response

| Parameters | Type | Description | Valid Values |
|---|---|---|---|
| NotificationType | Notification Type Enum | The type of notification sent. | **Mail** – notification email<br>**EventViewer** – report to Windows Event Viewer or UNIX syslog |
| NotificationStage | Notification Stage Enum | The job stage set to trigger the notification. | **Recovery**<br>**DiskSpace**<br>**OnExit** |

## Scheduler Modify Class

Contains information related to Modify commands performed on jobs by R-1 Scheduler.

### Syntax

```
SchedulerModifyCommand : SchedulerCommand
```

## Scheduler Rollback Class

Contains information related to Rollback commands performed on jobs by R-1 Scheduler.

### Syntax

```
SchedulerRollbackCommand : SchedulerCommand
```

## Scheduler Submit Class

Contains information related to Submit commands performed on jobs by R-1 Scheduler.

### Syntax

```
SchedulerSchedulerSubmitCommand : SchedulerCommand
```

## Scheduler Reinitialize Class

Contains information related to Reinitialize commands performed on Continuous Update jobs. The Reinitialize command spawns a new instance of a Continuous Update job.

### Syntax

```
SchedulerReinitializeCommand : SchedulerCommand
```

# BasicCommand Class

This class is included in the AuditRecord object and defines general details for the audit record request and the format to display it in.

Any of the commands defined as inheriting from Job/Center/Management commands can appear instead of `BasicCommand` and the user needs to query the type of object in order to properly read the fields.

`BasicCommand` is casted to the appropriate Command (JobCommand, CenterCommand or SchedulerCommand) and returns an object for each command specified in the GetAuditRecords request.

### Example

```
private void HandleJobCommand(object basicCMD)
{
            JobCommand job = basicCMD as JobCommand;
             log("job Command: " + job.Command);
             log("IssuedAt: " + job.IssuedAt);
            log("Status: " + job.Status);
            log("Message: " + job.Message);
```

### Response

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| `Issued At` | `dateTime` | The date and time at which the change was requested by the user. | |
| `Status` | `Status Enum` | The status of the change. | **Success**<br>**Failure** |
| `Message` | `String` | The message returned, depending on the status. | **NULL** for success/ error message for Failure |

# IssuerInformation Class

For information about this class, refer to the Set - IssuerInformation section.

# JobInformation Class

For information about this class, refer to the Set – JobInformation section.

# DiffObject Class

This class is returned in the AuditRecord object for CenterCommands queries. It helps you understand the changes that were made to the Center Management settings by displaying the details of the Center Management command you are querying.

### Response

| Parameter | Type | Description |
|---|---|---|
| `AfterUpdate` | `String` | The updated Center Management settings. |
| `BeforeUpdate` | `String` | The original Center Management settings. |

# EdgeOverride Class

This class defines the Edge Override settings of the Distribution job. The `EdgeOverride` object is returned when requesting audit information for Rollback Distribution Job Overrides, Modify Distribution Job Overrides, Resubmit Edge Overrides or Submit Edge Overrides job commands.

**NOTE:** Since all Edge-related operations follow a list of potential edge overrides, it is recommended to define a class of type `edge_override`.

### Response

| Parameter | Type | Description |
|---|---|---|
| `EdgeConfiguration Overrides` | `String` | Specifies the Edge's Edge Override configuration settings. |
| `EdgeName` | `String` | Specifies the Edge's name. |

# JobControlTrigger Class

This class defines job properties related to Job Chain-enabled jobs. It is returned when requesting audit information for Demand Submit and Scheduled Demand Submit commands.

### Response

| Parameter | Type | Description |
|---|---|---|
| `HeadJobId` | `String` | Specifies the job ID of the chain's initiating job. |
| `HeadJobName` | `String` | Specifies the job name of the chain's initiating job. |
| `HeadJobCenter` | `String` | Specifies the Center where the chain's initiating job is located. |
| `DirectJobId` | `String` | Specifies the job ID of the chained job. |

| Parameter | Type | Description |
|---|---|---|
| DirectJobName | String | Specifies the job name of the chained job. |
| DirectJobCenter | String | Specifies the Center where the chained job is located. |

# Method

## GetAuditRecords

Use `GetAuditRecords` to query the information relating to commands executed on the Center. An `AuditInformation` object is returned, containing the requested audit information and a `QueryContext` string, if specified.

You can query three types of commands:

- **Job commands** – Commands performed by Center users on jobs.
- **Center commands** – Changes made by users to Center Management settings.
- **Scheduler commands** – Commands performed by the R-1 Scheduler service on jobs.

### Syntax

```
AuditInformation AuditMgr.GetAuditRecords (session,
AuditFilter[], QueryContext, MaxRecordsAtOnce, RecordsOrder)
```

### Request Parameters

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| `Session` | `token` | Session GUID returned by the Login method. | |
| `AuditFilter` | `auditFilter` | Properties to fine tune the command filtering call.<br><br>Enables filtering according to general properties like the R-1 user who executed the commands, time range, command status, and job ID, or command-based properties like the command type (Job, Scheduler, Center Management), as well as specifying the specific commands. | nillable |
| `QueryContext` | `String` | Specify to return a string ID of the last record to be returned in the response so that future queries will return the next “chunk” of audit records without duplicates. | nillable |
| `MaxRecordsAtOnce` | `Int` | Specify the maximum number of records to return.<br><br>**NOTE:** To not limit the number of records returned, specify “0”. | |

| Parameter | Type | Description | Valid Values |
|---|---|---|---|
| `RecordsOrder` | `Enum` | Specify how to sort the returned records. | **OldestToLatest**<br>**LatestToOldest** |

## Response

The AuditInformation object is returned, containing an array of AuditRecord objects, one for each command, containing the requested audit information. Additional parameters may be included, if specified in the GetAuditRecords request.

| Parameter | Type | Description |
|---|---|---|
| `AuditRecord[]` | `AuditRecord[]` | An array of the command objects returned, according to the filter criteria specified. |
| `QueryContext` | `String` | A string of the last audit record returned, to be used in subsequent `GetAuditRecords` calls to return the next chunk of audit records. |

# Example (C#)

The following example queries all job commands within a defined time range:

```
Session session = null;
SessionManager SessionMgr = null;
AuditManager AuditMgr = null;

/* first we need to login */

SessionMgr = new SessionManager();
try
{
        SessionMgr.Url = "http://" + txtServer.Text + ":2837/R1" ;
    Credentials cred = new Credentials();
    cred.Center = "Replace with Center name";
    cred.Username = "Replace with User name";
    cred.Password = "Replace with User password";
    cred.Domain = "Replace with Domain name, if needed";
     session = SessionMgr.Login(CredentialsType.Credentials, cred, new
ClientInfo());
}
catch (Exception exp)
{
        Log("User Login error: " + exp.Message);
}

    /* Filling in the filter array (can be left null if a filter is not needed).


AuditFilter[] filterArray = new AuditFilter[1]; /* You can define more filters and fill in the array if desired. This
example uses one filter. */

AuditFilter filter = new AuditFilter();
filter.CommandFilter = new CommandFilter();
filter.CommandFilter.CommandType = CommandType.Jobs;
filter.CommandFilter.JobCommands = new JobCommands[1];
filter.CommandFilter.JobCommands[0] = JobCommands.AllJobCommands; // “JobCommands.AllJobComm
ands” can be any of the job command types like JobCommands.DemandSubmit.

/* Setting time range filter (can be left null) */
filter.FromTimeRange = new DateTime(2010, 5, 4, 7, 0, 0); /* can be left null */
filter.FromTimeRangeSpecified = true; // Must be set, otherwise FromTimeRange will be ignored
filter.UntilTimeRange = DateTime.Now; /* can be left null */
filter.UntilTimeRangeSpecified = true; // Must be set, otherwise UntilTimeRange will be ignored.

/* Setting issuer information filter. Any combination of the following parameters can be used. Praemters can
be left null. */
filter.IssuerInformation = new IssuerInformation();
filter.IssuerInformation.Username = "replace with Username";
filter.IssuerInformation.SourceIP = "replace with source machine’s ip address";
filter.IssuerInformation.SourceMachine = "replace with source machine name";
filter.IssuerInformation.Domain = "replace with Domain name";
filter.IssuerInformation.Interface = Interface.Console;

/* Setting job information filter. Any combination of the following parameters can be used. Parameters can
be left null. */
filter.JobInformation = new JobInformation();
filter.JobInformation.Application = Application.R1;
filter.JobInformation.GroupId = "replace with job group id";
```

```
filter.JobInformation.Id = "replace with job id";
filter.JobInformation.Name = "replace with job name";
filter.JobInformation.ParentId = "replace with job parent id";

/* Setting the filter in the filters array */
filterArray[0] = filter;


AuditMgr = new AuditManager();
AuditMgr.Url = "http://" + txtServer.Text + ":2837/R1";

/* Main function - calling GetAuditRecords */
AuditInformation inf = AuditMgr.GetAuditRecords(session, filterArray, null,
                                                50, RecordsOrder.Ascending);

if (inf == null)
{
        log("Get audit returned NULL");
    return;
}

if ((inf.AuditRecord == null) || (inf.AuditRecord.Length <= 0))
{
        Log("Get audit returned empty array");
    return;
}

Log("QueryContext: " + inf.QueryContext);

for (int i = 0; i < inf.AuditRecord.Length; i++)
{
        Log("Record #" + i);
    Log("CommandType: " + inf.AuditRecord[i].CommandType.ToString());
    Log("Domain: " + inf.AuditRecord[i].IssuerInformation.Domain);
    Log("Interface: " + inf.AuditRecord[i].IssuerInformation.Interface);
    Log("SourceIP: " + inf.AuditRecord[i].IssuerInformation.SourceIP);
        Log("SourceMachine: " +                    inf.AuditRecord[i].IssuerInformation.SourceMachine);
    log("Username: " + inf.AuditRecord[i].IssuerInformation.Username);

        /* You can get basic information by casting to BasicCommand or inside each function by casting to
"JobCommand/CenterCommand/Scheduler comamnd. */
    BasicCommand basecmd = inf.AuditRecord[i].Command is BasicCommand;
    Log("Issued at: " + basecmd.IssuedAt);
    Log("Message: " + basecmd.Message);
    Log("Status: " + basecmd.Status);

    if (inf.AuditRecord[i].Command is JobCommand)
            HandleJobCommand(inf.AuditRecord[i].Command); /* Example below */
    else if (inf.AuditRecord[i].Command is CenterCommand)
       HandleCenterCommand(inf.AuditRecord[i].Command);  /* Same as job
with specific Center command casting. */
    else
       HandleSchedulerCommand(inf.AuditRecord[i].Command); /* Same as job
with specific Scheduler command casting. */

       }



       /* At the end, you need to logout. */

try
```

```
{
            SessionMgr.Logout(session);
    Log("User Logged out");
}
catch (Exception exp)
{
            Log("User Logout error: " + exp.Message);
}




private void HandleJobCommand(object basicCMD)
{
            /* BasicCommand can be taken outside like we showed above */
            JobCommand job = basicCMD as JobCommand;
    log("job Command: " + job.Command);
    log("IssuedAt: " + job.IssuedAt);
    log("Status: " + job.Status);
    log("Message: " + job.Message);

            /* get job information */
    if (job.JobInformation != null)
    {
            Log("Application: " + job.JobInformation.Application);
        Log("GroupId: " + job.JobInformation.GroupId);
        Log("Id: " + job.JobInformation.Id);
        log("Name: " + job.JobInformation.Name);
        Log("ParentId: " + job.JobInformation.ParentId);
    }

            /* Casting the command to the appropiate specific CommandType. */
    if (basicCMD is HoldCommand) /* or if(job.Command == HoldCommand) */
    {
              HoldCommand hold = basicCMD as HoldCommand;
       Log("HoldUntil: " + hold.HoldUntil);
       Log("HoldMode: " + hold.Mode);
    }
    else if (basicCMD is AbortCommand)
    {
       AbortCommand abort = basicCMD as AbortCommand;
       Log("Reason: " + abort.Reason);
    }
            else if… // Handling all specific job commands.
```

# 5. Using Audit Events Viewer

The **Audit Events Viewer** web page enables you to retrieve a subset of the Center's audit information (i.e., one or all commands of a certain type at a time). To retrieve multiple commands of one or more types at a time, please use the Web Service API, as documented in this guide.

## Requirements

Please make sure the following requirements are met before setting up and using the IIS web site.

- Appropriate IIS Server installation (6, 7 or 7.5)
- ASP.NET 2.0.xxx installed and enabled

## Setting up R-1 Web Site

Before login, you must create an IIS virtual directory as a web site. Through this web site, you can then retrieve the desired audit information.

### Configuring IIS 6 for R-1 Web Site

**To set up the IIS 6 web site:**

1. Start the Microsoft IIS Internet Services Manager by browsing to **Start** \ **Control Panel** \ **Administrative Tools**, and then **Internet Services Manager**. The **IIS Manager** dialog appears.

2. In **IIS Manager**'s left pane, right click **Default Web Site**, select **New** and then select either **Virtual Directory** or **Web Site**.

3. In the **Creation Wizard** that appears, define the new virtual directory or web site:

4. To define a virtual directory:

    a. In the **Virtual Directory Alias** screen, define the name of the virtual directory. For example, "**`R1Audit`**".

    b. In the **Web Site Content Directory** screen, define the virtual directory's content directory as follows:

        i. Before defining the content directory, open Windows Explorer, browse to `~\RepliWeb\RDS\WebServices` and unzip the

`samples.zip` folder.

ii. In the **Web Site Content Directory** screen, click the **Browse** button and browse to `~\RepliWeb\RDS\WebServices\samples\Samples\WebUI` as the content directory.

By default, the base installation directory is `C:\Program Files\RepliWeb`

c. In the **Virtual Directory Access Permissions** screen, select the **Read**, **Run Scripts** and **Write** checkboxes.

d. Click **Next** and then click **Finish**.

e. In **IIS Manager**, make sure the virtual directory you created is an application, indicated with a  icon. Otherwise, right click the virtual directory, select **Properties** and in the **Properties** dialog that appears, click **Create** and then click **OK**.

5. To define a web site:

a. In the **Web Site Description** screen, provide a description for the web site. For example, "**`R1Audit`**".

b. In the **IP Address and Port Settings** screen, specify the web site's IP address, port number and, if needed, host header.

c. In the **Web Site Home Directory** screen, define the web site's content directory as follows:

i. Before defining the content directory, open Windows Explorer, browse to `~\RepliWeb\RDS\WebServices` and unzip the `samples.zip` folder.

ii. In the **Web Site Content Directory** screen, click the **Browse** button and browse to `~\RepliWeb\RDS\WebServices\samples\Samples\WebUI` as the content directory.

By default, the base installation directory is `C:\Program Files\RepliWeb`

d. In the **Web Site Access Permissions** screen, select the **Read**, **Run Scripts** and **Write** checkboxes.

e. Click **Next** and then click **Finish**.

6. Configure the default content page:

a. In the **IIS Manager**'s left pane, right click the new web site/virtual directory and select **Properties**.

b. In the **Properties** dialog, select the **Documents** tab.

c. Make sure the **Enable** default content page checkbox is selected, click **Add** and specify "**Audit.aspx**" as a new default content page.

7. In the **IIS Manager**'s left pane, click **Web Service Extensions** and make sure the **ASP.NET v2.0.xxx** extension is set to **Allow**.

## Configuring IIS 7 / 7.5 for R-1 Web Site

**To set up the IIS 7 / 7.5 web site:**

1. Start the Microsoft IIS Internet Services Manager by browsing to **Start** \ **Control Panel** \ **Administrative Tools**, and then **Internet Services Manager**.

2. In the **IIS Manager**'s **Connections** pane, right click **Default Web Site** and select **Add Application**.

3. In the **Add Application** dialog that appears, define the IIS web site:

   a. In the **Alias** field, define the name of the virtual directory or web site. For example, "**`R1Audit`**".

   b. In the **Physical Path** field, define the web site's content directory as follows:

      i. Before defining the content directory, open Windows Explorer and browse to `~\RepliWeb\RDS\WebServices` and unzip the `samples.zip` folder.

      ii. In the **Physical Path** field, click the **Browse** button and browse to `~\RepliWeb\RDS\WebServices\samples\Samples\WebUI` as the content directory.

      By default, the base installation directory is `C:\Program Files\RepliWeb`

   c. Click **OK**.

4. Provide appropriate access permissions to the Audit Events Viewer's web application folder:

   a. In the **IIS Manager**'s **Connections** pane, right click the newly created web site and select **Edit Permissions**.

b. In the **Properties** dialog that appears, click the **Security** tab and either add the **IIS_WPG** user for Windows Server 2003 or the **IIS_IUSRS** user for Windows Server 2008, as follows:

   i. In the **Properties** dialog, click **Edit**.

   ii. In the **Permissions** dialog that appears, click **Add**.

   iii. In the **Select Users, Computers, Service Accounts, or Groups** dialog that appears, click **Advanced**.

   iv. Click **Find Now**, select the appropriate user name and click **OK**.

   v. In the **Permissions** dialog, check the **Read**, **Write** and **Modify** checkboxes.

   vi. Click **OK** again and click **Apply**.

c. Repeat steps **4\a** and **4\b** to configure the **Users** group.

5. Configure the default content page:

   a. In the **IIS Manager**'s **Connections** pane, click the new web site/virtual directory.

   b. In the **Features list**'s **IIS** section, double click the **Default Document** feature.

   c. In the **Actions** pane, click **Add** and specify "**Audit.aspx**" as a new default content document.

6. Make sure ASP.NET is set to allow the appropriate page viewing permissions:

   a. In the **IIS Manager**'s **Connections** pane, click the machine's name.

   b. In the **Features list**'s **IIS** section, double click the **ISAPI and CGI Restrictions** feature.

   a. In the **ISAPI and CGI Restrictions** view, make sure the **ASP.NET v2.0.xxx** extension is set to **Allow**.

# Logging In

After setting up the IIS web site (explained in step 3 of the web site configuration sections above), log into the Audit Events Viewer page as follows.

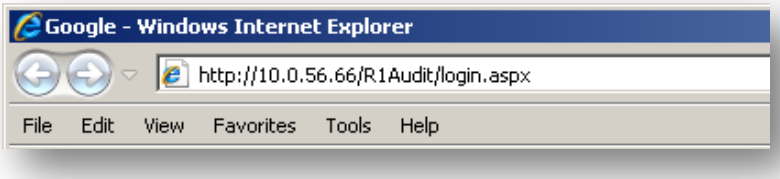
**To login to the Audit Events Viewer web page:**

1. Open your Internet browser.

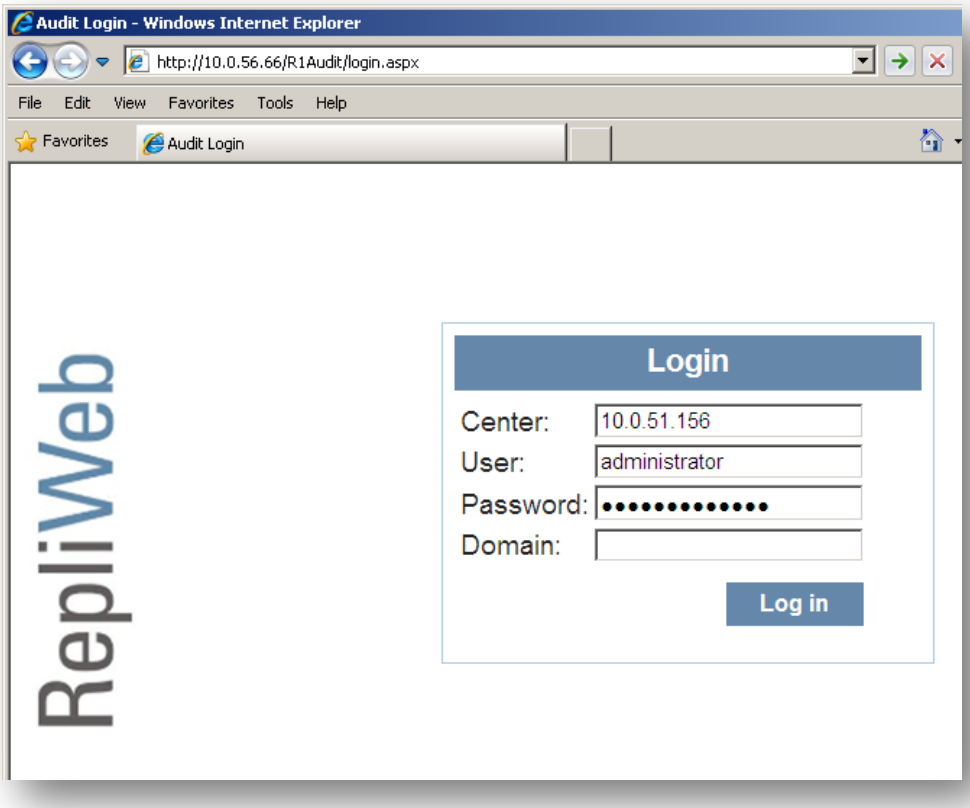
2. In the URL field, enter http://server/website/Audit.aspx

   Where <**server**> is the R-1 Server host name and <**website**> is the name of the IIS web site configured in the previous section.

   For example:




   The **Web Service login** page appears.



3. Enter the Center connection credentials:

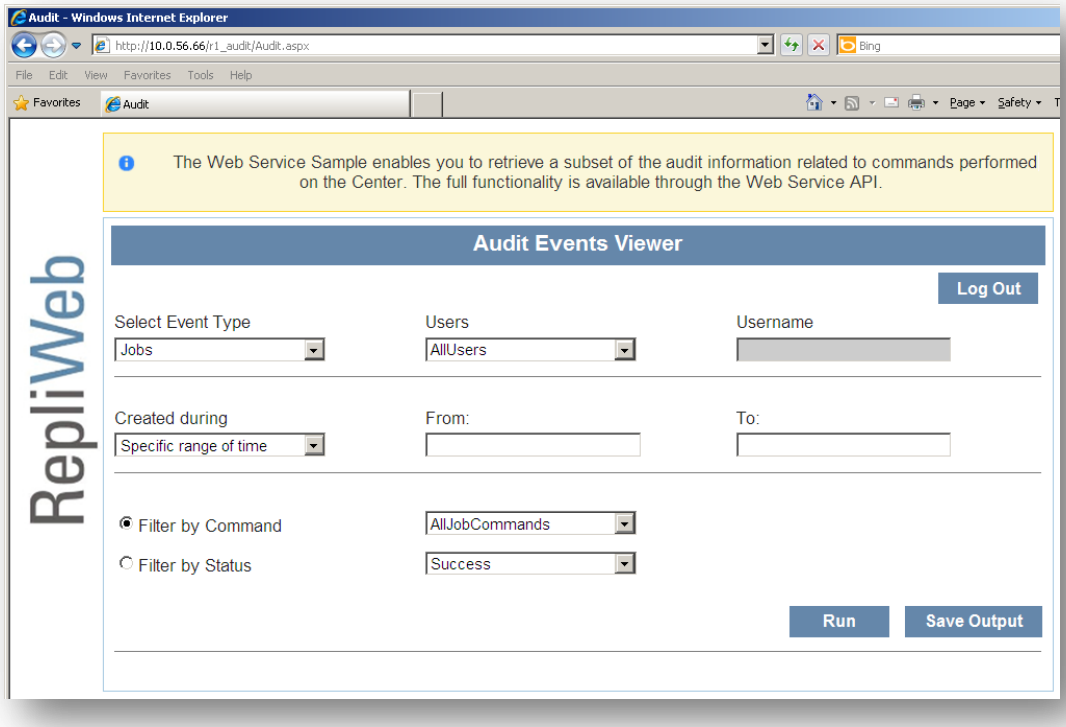
    a. In the **User** field, specify the user name that all jobs on the Center will use to perform the synchronization.

    b. Specify the **Password** for the account you specified in the **User** field.

    c. Specify the **Domain** if the user is part of a domain. Otherwise, leave this field blank.

4. Once you have filled in all the necessary fields, click **Login**. Login logs in to the Center specified.

    The **Audit Events Viewer** web page appears.



# Retrieving Records

**To retrieve a record from the Center:**

1. Log into the **Audit Events Viewer** web page, as explained in the previous section.

2. From the **Select Event Type** drop-down list, select the type of command to retrieve:
   - **Jobs** – to query commands performed by users on jobs.
   - **Management** – to query changes made by users to Center Management settings.
   - **Scheduler** – to query commands initiated by the Scheduler service.

3. From the **Users** drop-down list, select:
   - To retrieve records related to any user, select **AllUsers**.
   - To retrieve records related to a specific user, select **UserSpecific** and enter the username in the Username field.

4. From the **Created during** drop-down list, select the command's creation time:
   - **Specific range of time** – to retrieve records created during a specific period, click anywhere in the **From**/**To** fields and select the start and end time from the Date-Time selection window.
   - **Last 24 hours** – to retrieve records created during the last 24 hours.
   - **Last 7 days** – to retrieve records created during the last 7 days.

5. Select how to filter the records:
   - To retrieve records according to commands, select the **Filter by Command** option and select the desired command from the drop-down list to the right.
     You retrieve records for all the commands by selecting the **AllCommands** option from the drop-down list.

     **NOTE:** The available commands depend on the setting you selected in the **Select Event Type** drop-down list.
   - To retrieve records according to the command's status, select the **Filter by Status** option and select the desired status from the drop-down list to the right.

6. Click **Run** to retrieve records based on these settings. A list of the retrieved records appears.

7. In the Items per page field, specify how many records to display per page. Click **Next** at the bottom of the records table to display more records.

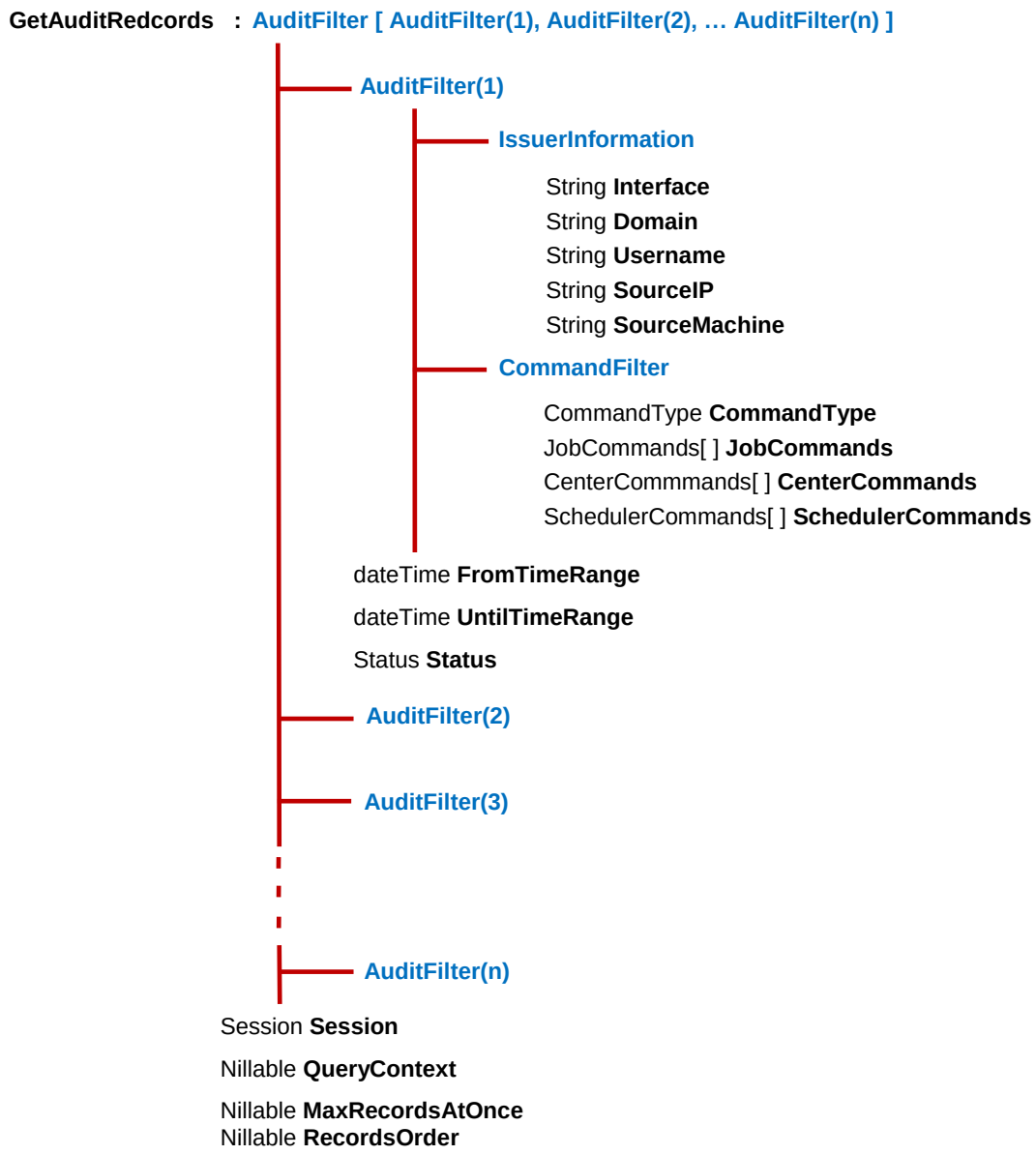
8. To sort the records from latest to oldest or oldest to latest, click the arrow that appears in the title of the list's Issued at column.

9. To save the records retrieved in a .CSV file, click **Save Output**.

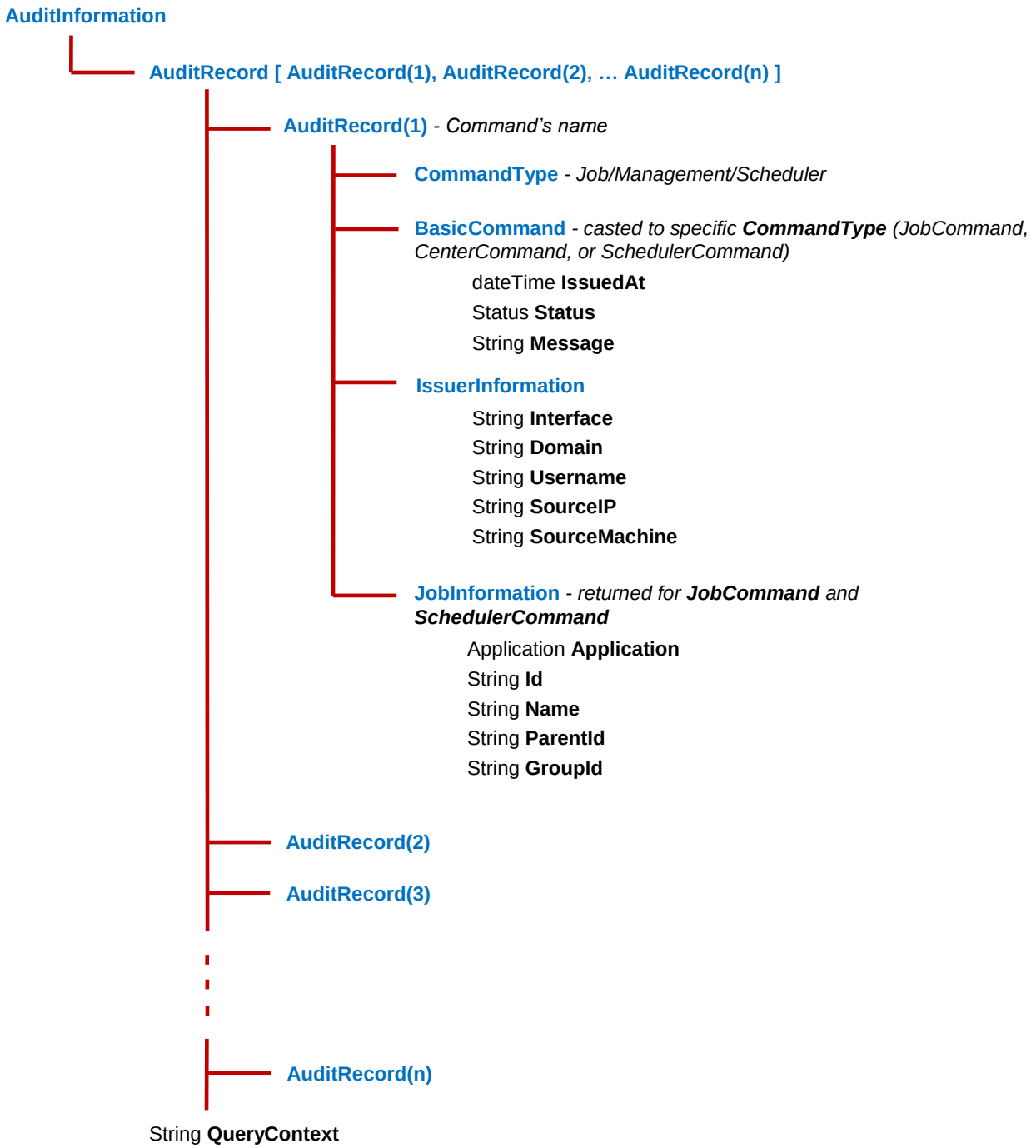
# A. Class Hierarchies

## Set Class Hierarchy

**AuditInformation** GetAuditRecords ( session,
AuditFilter[],
QueryContext,
MaxRecordsAtOnce,
RecordsOrder )



# Get Class Hierarchy



# B. Workflow Sample Code

This appendix provides samples of the R-1 Web Service workflow in C# and Java.

## C# Sample

```
using System;
using WebServiceSamples.R1WebService;

namespace WebServiceSamples
{
   class Samples
   {
      /// <summary>
      /// Initialize Login Credentials
      /// </summary>
      /// <param name="UserName">User name to use.</param>
      /// <param name="Password">Password of the user.</param>
      /// <param name="Domain">Domain</param>
      /// <returns>Credentials</returns>
      private static Credentials InitializeCredentials(string UserName, string Password, string Domain)
      {
         //Initialize connection credentials
         Credentials res = new Credentials();
         res.Username = UserName;
         res.Domain = Domain ;
         res.Password = Password;
         return res;
      }

      /// <summary>
      /// Initialize an array of Audit Filters
      /// </summary>
      /// <returns>Audit Filter Array, which include all Audit events from the last half hour. </returns>
      private static AuditFilter[] InitializeAuditFilters()
      {

            //Initialize Audit filter:
            AuditFilter myFilter = new AuditFilter();
            myFilter.JobInformation = new JobInformation();
            myFilter.CommandFilter = new CommandFilter();

            //Define command types to filter:
            myFilter.CommandFilter.CenterCommands = new CenterCommands[] { CenterCommands.AllCenterCommands };
            myFilter.CommandFilter.JobCommands = new JobCommands[] { JobCommands.AllJobCommands };
            myFilter.CommandFilter.SchedulerCommands = new SchedulerCommands[] {
SchedulerCommands.AllSchedulerCommands };

            //Set From Time Range to Half an Hour Ago:
            myFilter.FromTimeRange = DateTime.Now.Subtract(new TimeSpan(0, 30, 0));
            myFilter.FromTimeRangeSpecified = true;
            myFilter.UntilTimeRange = DateTime.Now;
            myFilter.UntilTimeRangeSpecified = true;

            //Specify only successful actions:
            myFilter.Status = Status.Success;
```

```
            myFilter.StatusSpecified = true;

            return new AuditFilter[] { myFilter };
        }

        /// <summary>
        /// Display a single AuditRecord in standard output.
        /// </summary>
        /// <param name="Record">The record to display.</param>
        private static void DisplayAuditRecord(AuditRecord Record)
        {
            Console.WriteLine("{0}{1}: {2}\\{3} has performed a {4}, Status:{5}. ",
                              Environment.NewLine,
                              Record.Command.IssuedAt.ToString("HH:mm:ss"),
                              Record.IssuerInformation.Domain,
                              Record.IssuerInformation.Username,
                              Record.Command.GetType().Name,
                              Record.Command.Status);

            if (Record.CommandType == CommandType.Jobs || Record.CommandType == CommandType.Scheduler)
            {
                JobInformation jobInfo = ((Record.CommandType == CommandType.Jobs) ?
                                          ((JobCommand)Record.Command).JobInformation  :
                                          ((SchedulerCommand)Record.Command).JobInformation);
                Console.WriteLine("\t  Job Name: \"{0}\", Id: {1}", jobInfo.Name , jobInfo.Id);
            }
        }

        /// <summary>
        /// Display an Audit Information results in standard output.
        /// </summary>
        /// <param name="AuditInfo">The Audit Information to display.</param>
        private static void DisplayAuditInformation(AuditInformation AuditInfo)
        {
            if (AuditInfo != null && AuditInfo.AuditRecord != null)
            {
                foreach (AuditRecord aRecord in AuditInfo.AuditRecord)
                {
                    DisplayAuditRecord(aRecord);
                }
            }
        }

        /// <summary>
        /// The following method queries the web service for successful audit events from the last
        /// half hour, and displays them in the console window. Job Id and Job name will be displayed for Job and Scheduled
       events.
        /// </summary>
        public static void DisplayAllJobs()
        {
            try
            {
                //SessionManager and AuditManager should be discarded after use.
                using (SessionManager mySessionManager = new SessionManager())
                {
                    string webSvcUrl = "http://localhost:2837/R1";
                    mySessionManager.Url = webSvcUrl;
                    Credentials myCredentials = InitializeCredentials("User","7KRVS4F7Q54D1", "Domain");

                    //Establish a session with R1 Web Service.
                    Session mySession = mySessionManager.Login(CredentialsType.Credentials, myCredentials, new ClientInfo());
```

```
            try
            {
                using (AuditManager myAudit = new AuditManager())
                {
                    //Set web service url for AuditManager.
                    myAudit.Url = webSvcUrl;
                    string strQueryContext = null;

                    //Initialize the audit filter
                    AuditFilter[] myAuditFilters = InitializeAuditFilters();
                    AuditInformation myResult = null;

                    do
                    {
                        Console.WriteLine("==================================");

                        //Get the Audit records 10 at a time:
                        myResult = myAudit.GetAuditRecords(mySession,
                                                myAuditFilters,
                                                strQueryContext,
                                                10,
                                                RecordsOrder.LatestToOldest);

                        //Display the query result:
                        DisplayAuditInformation(myResult);

                        //Update the query context for the next iteration.
                        strQueryContext = myResult.QueryContext;

                    } while (myResult != null && myResult.AuditRecord != null && myResult.AuditRecord.Length > 0);
                }
            }
            finally
            {
                //Disconnect the session:
                if (mySessionManager != null && mySession != null)
                    mySessionManager.Logout(mySession);
            }
          }
        }
        catch (Exception exp)
        {
            Console.WriteLine();
            Console.WriteLine("An Exception of type: {0} Was Thrown.", exp.GetType().FullName);
            Console.WriteLine(exp.Message);
            Console.WriteLine(exp.StackTrace);
        }
    }
  }
}
```

# Java Sample

```
import java.text.SimpleDateFormat;
import java.util.Date;
import com.repliweb.rw.*;

public class WebServiceSamples {

	public static void main(String args[]) {
		Credentials myCred = InitializeCredentials("administrator", "123456","", "localhost");
		CredentialsType myCredType = CredentialsType.Credentials;
		AuditFilter[] myAuditFilterArray = InitializeAuditFilters();
		String url = "http://localhost:2837/R1";
		AuditInformation myAuditInfo = GetAuditInfo(myCred, myCredType,myAuditFilterArray, url);
		PrintAuditInfo(myAuditInfo);
	}

	public static Credentials InitializeCredentials(String UserName, String Password, String Domain, String Center) {
		// Initialize connection credentials
		Credentials res = new Credentials();
		res.setUsername(UserName);
		res.setDomain(Domain);
		res.setPassword(Password);
		return res;
	}

	public static AuditFilter[] InitializeAuditFilters() {
		// Create an Audit Filter
		AuditFilter myAuditFilter = new AuditFilter();

		// Create a command filter and set the filter to All center commands
		CommandFilter myCommandFilter = new CommandFilter();

		myCommandFilter.setCenterCommands(new CenterCommands[] { CenterCommands.AllCenterCommands
});
		myCommandFilter.setJobCommands(new JobCommands[] { JobCommands.AllJobCommands });
		myCommandFilter.setSchedulerCommands(new SchedulerCommands[] {
SchedulerCommands.AllSchedulerCommands });

		// Set the Audit Filter's Command filter to Management & Scheduler & Jobs
		myAuditFilter.setCommandFilter(myCommandFilter);
		String[] myCommandType = {"Management","Jobs","Scheduler" };
		myCommandFilter.setCommandType(myCommandType);
		myAuditFilter.setCommandFilter(myCommandFilter);

		// Create a filter array and put MyFilter in the filter array
		AuditFilter[] myFilterArray = new AuditFilter[1];
		myFilterArray[0] = myAuditFilter;

		return myFilterArray;
	}

	public static AuditInformation GetAuditInfo(Credentials myCredentials, CredentialsType myCredentialsType,
AuditFilter[] myAuditFilterArray, String url) {
		ClientInfo ci = new ClientInfo();
		ci.setVersion("2010/07/19");
		Session mySession = new Session();
		SessionManagerProxy smp = new SessionManagerProxy();
```

```
		// Set Session webservice url
		smp.setEndpoint(url);

		// clientInfo holds the wsdl file version - now used with null
		try {
			mySession = smp.login(myCredentialsType, myCredentials, ci);
		} catch (Exception e) {
			e.printStackTrace();
		}
		// getting the information
		try {
			AuditManagerProxy myAudit = new AuditManagerProxy();
			// Set AuditManager webservice url
			myAudit.setEndpoint(url);
			// Get the results
			AuditInformation myResult = myAudit.getAuditRecords(mySession, myAuditFilterArray, null, 0,
RecordsOrder.LatestToOldest);
			if (myResult.getAuditRecord() != null) {
				return myResult;
			} else
				System.out.println("Audit Info DB does not exist");
		} catch (Exception e) {
			e.printStackTrace();
		} finally {
			try {
				smp.logout(mySession);
			} catch (Exception e) {
				e.printStackTrace();
			}
		}
		return null;
	}
	public static void PrintAuditInfo(AuditInformation myAuditInfo) {
		if (myAuditInfo.getAuditRecord() != null) {
			try {
				AuditRecord[] auditRecords = myAuditInfo.getAuditRecord();
				for (int i = 0; i < auditRecords.length; i++) {
					// print row number
					System.out.print(i + 1 + ". ");
					// check if the current audit record is of management type
					if (auditRecords[i].getCommand() instanceof CenterCommand) {
						System.out.print("Management");
						// This is a place holder for more Management commands
					}

					if (auditRecords[i].getCommand() instanceof SchedulerCommand) {
						System.out.print("Scheduler");
						SchedulerCommand myTempSchedCommand = new
SchedulerCommand();
						if
(myTempSchedCommand.getCommand()==SchedulerCommands.CompletionExitProc) {
							myTempSchedCommand.getStatus();
						}
						// This is a place holder for more Scheduler commands
					}
					if (auditRecords[i].getCommand() instanceof JobCommand) {
						System.out.print("Jobs - ");
						JobCommand myTempJobCommand = new JobCommand();
						myTempJobCommand = (JobCommand)
auditRecords[i].getCommand();

						if (myTempJobCommand.getCommand() == JobCommands.Abort) {
```

```
                                                AbortCommand myTempAbortCommand = (AbortCommand)
auditRecords[i].getCommand();
                                                System.out.print(", Message: "+
myTempAbortCommand.getMessage());
                                                System.out.print(" Aborted ,");
                                                System.out.print(myTempAbortCommand.getReason());

                                        }
                                        if (myTempJobCommand.getCommand() == JobCommands.Delete) {
                                                System.out.print(" Deleted");
                                        }
                                        if (myTempJobCommand.getCommand() ==
JobCommands.SubmitReplication) {
                                                System.out.print(" Replication Submit");
                                        } else {
                                        }
                                        // print job info (Job Name & Job Id)
                                        JobInformation myJobInfo = myTempJobCommand.getJobInformation();
                                        System.out.print(" ,Job Info: " + " Job Name:"+ myJobInfo.getName() + "
Job Id:"+ myJobInfo.getId());
                                }

                                if (auditRecords[i].getCommand() instanceof SchedulerJobCompletedCommand)
{
                                        SchedulerJobCompletedCommand myTemp1 =
(SchedulerJobCompletedCommand) auditRecords[i].getCommand();
                                        System.out.print(", Exit Status:"+myTemp1.getExitStatus());
                                }

                                BasicCommand myBasicCommand = auditRecords[i].getCommand();

                                // print session time & date
                                long myTime = myBasicCommand.getIssuedAt().getTimeInMillis();
                                SimpleDateFormat formatter1 = new SimpleDateFormat("dd/MM/yyyy
HH:mm:ss");
                                Date regDate1 = new Date(myTime);
                                System.out.print(" , Time: " + formatter1.format(regDate1));
                                // print job status
                                System.out.print(", Status: "+ myBasicCommand.getStatus()[0]);
                                // print user info
                                IssuerInformation myIssuerInfo = auditRecords[i].getIssuerInformation();
                                if (myIssuerInfo.getSourceMachine() != null)
                                        System.out.print(" ,Issuer Info: " + " Source Machine:"+
myIssuerInfo.getSourceMachine());
                                System.out.println();
                        }

                } catch (Exception e) {
                        e.printStackTrace();
                }
        } else
                System.out.println("Audit Information object is null");
    }
}
```