

Rational Build Forge



AutoExpurge System

Version 7.1.2 and later

Note

Before using this information and the product it supports, read the information in "Notices," on page 11.

This edition applies to version 7.1.2 of Rational Build Forge and to all subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright IBM Corporation 2011.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Chapter 1. About the Autoexpurge system	1
Chapter 2. Managing jobs and purges by using classes in Rational Build Forge . .	3
Chapter 3. Setting up the AutoExpurge system	5

Chapter 4. Running AutoExpurge jobs. .	7
Chapter 5. Viewing job BOMs using the example XSL	9
Appendix. Notices	11
Trademarks	13

Chapter 1. About the Autoexpurge system

The AutoExpurge system provides an automated tool for exporting and removing files from archived jobs from the IBM® Rational Build Forge® database. It also provides the means to view the exported job BOM files in a web browser.

The AutoExpurge system is provided in a zip file for you to download:
<http://public.dhe.ibm.com/software/dw/demos/rautoexpurge/AutoExpurge.zip>

It consists of the following parts:

- AUTOEXPADAPT.xml. Import this file into Rational Build Forge to create a library and environment.
- AutoExpurgeAdaptor.xml.<http://public.dhe.ibm.com/software/dw/demos/rautoexpurge/AutoExpurge.zip> Create an adaptor and use this XML file to populate the Interface field.
- Viewing tools directory. Use the example.xsl and ibm_swoosh.jpg files to set up the exported job BOM XML files to be viewed in a web browser.
- Other executable files (.jar files) and associated material (AutoExpurgeDocs directory).

See Setting up the AutoExpurge system to learn how to set up the system.

Chapter 2. Managing jobs and purges by using classes in Rational Build Forge

Assigning jobs to classes allow you to specify how and when to purge completed jobs.

This section describes the typical job lifecycle and how it can be modified by using classes (to drive purge schedules) and by using AutoExpurge.

Job lifecycle

Normally jobs go through the following stages:

- Queued
- Running
- Completed

After a job is completed, the system takes no further action on it. the job will stay in the database forever. You can remove the completed jobs in two ways:

- Manually: you could move jobs to the Archived state or remove them manually.
- Purges: by assigning a job to a class and assigning the class to a schedule, you can set up the system to purge completed jobs.

Adding purges to the job lifecycle

A job class typically corresponds to a type of job. The actions associated with the class are set according to what information you need to keep. The schedule associated with the class depends on how often you want to perform those actions.

Depending on what a type of job does, you could have different needs for archiving and removing jobs. For example, consider three classes:

- `short`: This class could be used for short-lived utility jobs, such as those that send a status email or clean the files on build servers. These builds could be removed daily.
- `scratch`: This class could be used for daily builds of a software system. You would keep these jobs around for a few days in order to inspect them.
- `production`: This class could be used for jobs that are kept permanently, for example the production build of a software product. There would be no purge policy for this class. You might not assign a job to this class until it has already run. With Rational Build Forge, you can reassign the class for a job.

After the scratch jobs are moved to the archived state, you still need to determine how long to keep them and when to remove them. In this scenario, you need to remove them manually. In a larger environment, an automated way to work with multiple classes of job is needed.

Adding AutoExpurge to the job lifecycle

AutoExpurge is helpful because it exports and stores the job BOM and removes files from jobs that have already had some action performed on them through their class. Classes can have only one action and it is often desirable to archive jobs as

the action for a class but not delete the files. Leaving the files allows you to inspect them and compare successive runs of the job.

It is possible to use only the Rational Build Forge facilities to manage the short and production classes:

- **short:** Create a class and set its actions to remove all files. Schedule it to run daily.
- **production:** Create a class and assign no actions. This class is used to keep a shipped version of a product in the system. It has no purge actions or schedule. When you have run a scratch job that meets the release exit criteria, you manually reassign its class to production.

The scratch class needs more attention. You want to keep the jobs around for inspection for a few days, but you also want to remove them from the system. It might be valuable to keep a record of these jobs. In addition, in a production environment, there might be many different types of job that run on a daily basis, and these jobs might be segregated by product or product component. They might require different classes that are similar to scratch, both to keep the jobs separate and to reflect different levels of scrutiny that the jobs may require over time. A single scratch class is used in this document for simplicity.

A job can belong to only one class, and one class can have only one set of action definitions assigned to it. However, this example has two requirements:

- *Move the jobs from Completed to Archived daily.* This task can be accomplished through the class assigned to the job. The purge action consists of moving the job from Completed to Archived. Files are not removed. This action is scheduled daily.
- *Archive the job BOMs and delete all files according to a schedule.* After the job is in the Archived state, you cannot use the class assigned to the job to perform different actions. In this case, you use AutoExpurge to export the job BOMs and delete the files associated with the job. This action is scheduled at a longer interval, perhaps one to two weeks.

For the second case, you can use AutoExpurge to identify the classes to work on and to schedule the removal of files associated with those classes. The rest of this document describes how to set up and use AutoExpurge.

Assessing your environment to determine job removal policy

It is not possible to make general recommendations about what jobs to remove and what schedule to use. You must do the following assessments:

- Assess your needs for retaining completed jobs in Rational Build Forge
- Determine the space requirements for retaining these jobs and assess these requirements against the total space available and the database performance when the used space increases.

For methods of assessing your needs, see the white paper Rational Build Forge General Maintenance Strategy. To assess the database needs for running Rational Build Forge, use the Database Resource Usage Worksheet in the white paper.

Chapter 3. Setting up the AutoExpurge system

This task describes how to import the AutoExpurge project and environment definitions, and how to set up the AutoExpurgeAdaptor adaptor.

Procedure

1. Download and extract the contents of the `autoexpurge.zip` file. Create a directory home for the AutoExpurge system. This directory will be used by the system as your `TOOLS_DIR`. Download the archive file from the IBM demos site into that directory and extract its contents.
2. Create a properties directory and properties file.
 - a. Create a directory for the AutoExpurge properties file. The system refers to this directory through the `PROPERTIES_DIR` variable.
 - b. In the properties directory, create a text file. The `autoExpurgeProperties.txt` file is used as an example. The system refers to this file through the `PROPERTIES_FILE` variable.
3. Import the `AUTOEXPADAPT.xml` file into Rational Build Forge.
 - a. Click **Administration** > **Import** in the console.
 - b. Import the file. Browse to the location where you extracted the contents of the `AutoExpurge.zip` file. The `AUTOEXPADAPT.xml` file is in the root directory. Accept all of the defaults for the import action.

After importing the file, you have the following new objects in the Rational Build Forge system:

- `AutoExpurgeAdaptorRunner`, a library that performs the job export and removes archived jobs.
 - `AutoExpurgeEnvironment`, an environment used by the `AutoExpurgeAdaptorRunner` project when it runs.
4. Set the variables in the AutoExpurge environment.
 - a. Click **Environments** in the left menu of the console.
 - b. Click **AutoExpurgeEnvironment**. The list of variables in the environment is shown.
 - c. Assign values to the variables as follows.
 - `PASSWORD`: Password for `USER_NAME`.
 - `USER_NAME`: User name defined on the console where the AutoExpurge job will run. The user needs appropriate permissions to execute jobs, delete jobs, and export. The Build Engineer access group provided with the product has the necessary permissions.
 - `SERVER`: Name or IP address of the host where Rational Build Forge is running.
 - `BUILD_EXPORT_DIR`: Directory to store exported job BOM files.
 - `TOOLS_DIR`: Directory that you created to store the extracted AutoExpurge files.
 - `PROP_DIR`: Directory that you created to contain the properties file.
 - `PROP_FILE_NAME`: File that you created to store the properties that define how AutoExpurge runs. The file is blank now.

Note: Do not include the .txt extension in the file name you enter here. The example name was autoExpurgeProperties.txt. If you used it, enter autoExpurgeProperties. For information about how to populate AutoExpurge jobs before running them, see Chapter 4, “Running AutoExpurge jobs,” on page 7.

5. Create the AutoExpurge adaptor.
 - a. Click **Projects > Adaptors** in the console.
 - b. Click **Add Adaptor**.
 - c. Specify the name to be AutoExpurgeAdaptor. Leave the following fields set to their defaults: Type=Source, Template= – None –, and Access=Build Engineer.
 - d. In the Interface field, copy the entire contents of AutoExpurgeAdaptor.xml. It is in the root of the directory you created when you extracted the files from AutoExpurge.zip.

Chapter 4. Running AutoExpurge jobs

An AutoExpurge job exports the job BOMs and removes all files for archived jobs that meet the purge policy that is defined in the properties file.

About this task

This procedure assumes that you have done the following tasks:

- Defined the classes to use (including purge policy) in Rational Build Forge
- Defined projects that use those classes
- Run the projects as jobs
- Allowed the purge schedules that are associated with the classes to run and produce archived jobs for each class

You can also manually move completed jobs for each class to the Archived state. The jobs must use the classes that you use to define AutoExpurge policy.

Procedure

1. Define the AutoExpurge policy in the properties file you created. The example file was called `autoPurgeProperties.txt`.

Each line in the file associates the name of a class with the number of days between the AutoExpurge purges. For example, using the earlier class examples of `short`, `scratch` and `production`, the file would contain their names and the purge intervals (in days) as follows:

```
short=7  
scratch=14
```

The `production` class does not have a purge policy because it is set up to be retained permanently.

2. Make the AutoExpurgeRunner library a project. Open the library and use the Selector field to define the selector to use when running the project. The selector must already exist. When the selector is defined, AutoExpurgeRunner is displayed in the projects list rather than the libraries list.
3. Run the AutoExpurgeRunner project. When it runs, it looks for jobs in the Archived state that are older than the days specified in the policy. For each qualifying archived job, it exports the job BOM and removes all files for the job.

Chapter 5. Viewing job BOMs using the example XSL

After the AutoExpurgeRunner job has run, you can view the exported job BOMs in a web browser. You must first modify them to use an XSL stylesheet.

About this task

An XSL transform file named `example.xsl` is provided in the `viewingTools` directory. It is in the root directory where you extracted the contents of `AutoExpurge.zip`. To use the provided stylesheet to view exported job BOM XML files, do the following:

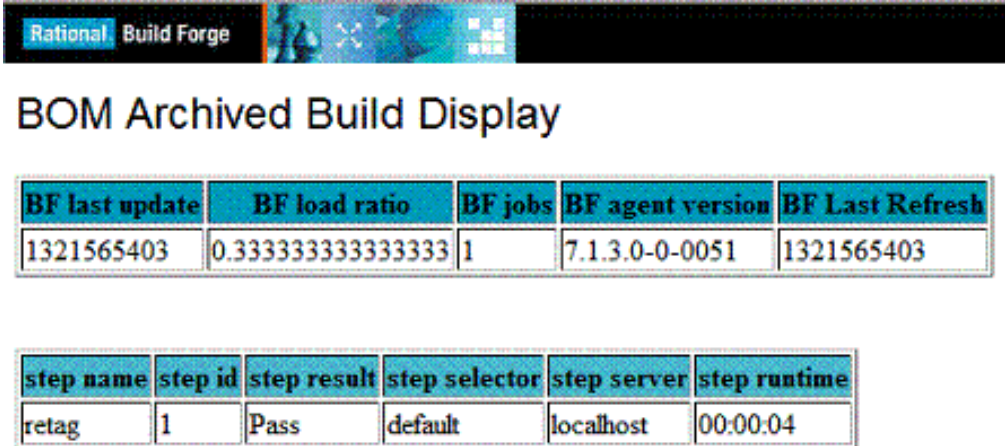
Procedure

1. Copy `example.xsl` and `ibm_swoosh.jpg` from the `viewingTools` directory into the directory where `AutoExpurge` exported the job BOMs.
2. Add a stylesheet specification to the top of each XML file you want to view. The stylesheet specification is as follows:

```
<?xml-stylesheet type="text/xsl" href="example.xsl"?>
```
3. Use your web browser to open the XML files you want to view.

Example

The following image is an example of what is seen when viewing a job BOM.



BOM Archived Build Display

BF last update	BF load ratio	BF jobs	BF agent version	BF Last Refresh
1321565403	0.3333333333333333	1	7.1.3.0-0-0051	1321565403

step name	step id	step result	step selector	step server	step runtime
retag	1	Pass	default	localhost	00:00:04

You can customize the `example.xsl` according to your requirements.

Appendix. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created

programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for Rational Software
IBM Corporation
5 Technology Park Drive
Westford, MA 01886
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

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

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

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

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2011.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at www.ibm.com/legal/copytrade.html.

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