Red Hat Enterprise Linux 8.0 Beta

Using Application Stream

An introduction to Application Stream in Red Hat Enterprise Linux 8.0 Beta

Last Updated: 2018-12-06

Red Hat Enterprise Linux 8.0 Beta Using Application Stream

An introduction to Application Stream in Red Hat Enterprise Linux 8.0 Beta

Legal Notice

Abstract

This document describes searching, discovering, installing, and using content in the Application Stream in Red Hat Enterprise Linux 8.0 Beta. This includes a description of how to use modules, streams and profiles.

Table of Contents

THIS IS A BETA VERSION!

Thank you for your interest in Red Hat Enterprise Linux 8.0 Beta. Be aware that:

- Beta code should not be used with production data or on production systems.
- Beta does not include a guarantee of support.
- Feedback and bug reports are welcome. Discussions with your account representative, partner contact, and Technical Account Manager (TAM) are also welcome.
- Upgrades to or from a Beta are not supported or recommended.

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Please let us know how we could make it better. To do so:

- For simple comments on specific passages, make sure you are viewing the documentation in the Multi-page HTML format. Highlight the part of text that you want to comment on. Then, click the **Add Feedback** pop-up that appears below the highlighted text, and follow the displayed instructions.
- For submitting more complex feedback, create a Bugzilla ticket:
    1. Go to the Bugzilla website.
    2. As the Component, use **Documentation**.
    3. Fill in the **Description** field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.
    4. Click **Submit Bug**.

CHAPTER 1. USING APPLICATION STREAM

The following sections provide an overview of the concepts related to Application Stream in Red Hat Enterprise Linux 8.

- Section 1.1, “Distribution of content in RHEL 8” describes how content in Red Hat Enterprise Linux 8 is split into BaseOS and Application Stream.
- Section 1.2, “Types of content in Application Stream” describes the types of content provided by Application Stream.
- Section 1.3, “Modules, streams, and profiles in Application Stream” provides an overview of the new modular features in Application Stream in Red Hat Enterprise Linux 8.
- Section 1.4, “YUM usage” describes how the **YUM** package manager provided in Red Hat Enterprise Linux 8 combines the traditional and modular features.

1.1. DISTRIBUTION OF CONTENT IN RHEL 8

RHEL 8 content is distributed through the two main repositories: **BaseOS** and **Application Stream** (AppStream).

BaseOS

The BaseOS repository provides the core set of the underlying OS content in the form of traditional RPM packages. BaseOS components have a life cycle identical to that of content in previous Red Hat Enterprise Linux releases.

Application Stream

The Application Stream repository provides content with varying life cycles as both modules and traditional packages. Application Stream contains necessary parts of the system, as well as a wide range of applications previously available as a part of Red Hat Software Collections and other products and programs.

IMPORTANT

Both BaseOS and AppStream are a necessary part of a Red Hat Enterprise Linux system.

1.2. TYPES OF CONTENT IN APPLICATION STREAM

Application Stream contains two types of content:

Modules

A module describes a set of RPM packages that belong together. Modules can contain several *streams* to make multiple versions of applications available for installation. *Enabling* a module *stream* gives the system access to the RPM packages within that module stream.

Traditional RPM packages

Traditional RPM packages available for immediate installation.

The traditional methods of package management and installation are transparently supported for all content. The appropriate combination of modules and streams is automatically used to enable installation of packages that depend on modular features.

1.3. MODULES, STREAMS, AND PROFILES IN APPLICATION STREAM

Application Stream contains *modules*. A module is a set of RPM packages that can or must be installed together. A typical module can contain packages with an application, packages with the application's specific dependency libraries, packages with documentation for the application, and packages with helper utilities.

Module streams

Each module can have one or more *streams*, which hold different versions of the content. Each of the streams receives updates independently.

For each module, only one of its streams can be enabled and provide its packages, allowing installation of the respective version of content.

Usually, the stream with the latest version is marked as default. This stream is used when operations do not specify a particular stream and a different stream has not been previously enabled.

For simplicity, you can also think of module streams as virtual repositories in the Application Stream physical repository.

> **Example 1.1. postgresql module streams**
>
> The **postgresql** module provides the **PostgreSQL** database versions 9.6.10 and 10.3 in streams **9.6** and **10**, respectively. **10** is currently the default stream.

Module profiles

Each module can have one or more *profiles*. A profile is a list of certain packages to be installed together for a particular use-case such as for a server, client, development, minimal install, or other. At the same time, profiles are also a recommendation by the application packagers and experts.

Installing packages by using a module's profile is a one-time action. It does not prevent installing or uninstalling any of the packages provided by the module. This also means that it is possible to install packages by using multiple profiles of the same module without any further preparatory steps.

The package list of a module can contain packages outside the module stream, usually from BaseOS or stream's dependencies.

Modules in Application Stream always have a *default profile* which is used for installing when no other profile is explicitly specified.

> **Example 1.2. httpd module profiles**
>
> The **httpd** module providing the **Apache** web server offers the following profiles for installation:
>
> - `default` - a hardened production-ready deployment
> - `devel` - the packages necessary for making modifications to `httpd`
> - `minimal` - the smallest set of packages that will provide a running webserver

1.4. YUM USAGE

The **YUM** package management tool has been updated and adds support for the new modular features of Application Stream based on the DNF technology.

There are no changes to established uses and commands of the **yum** tool. Where required, the new modular functionality is transparently used to achieve the same functionality as previously available. For example, installing a package from a default module stream enables the stream in order to receive updates from it.

For handling the modular content, the **yum module** command has been added. See the following chapters for additional details.

CHAPTER 2. DISCOVERING CONTENT IN APPLICATION STREAM

The following sections describe how to discover content in Application Stream in Red Hat Enterprise Linux 8.

- Section 2.1, “Searching for a package” describes how to search for packages providing desired content.
- Section 2.2, “Listing available modules” describes how to list available modules and find out details about them.
- Section 2.3, “Example: Finding out details about a module” contains an example of steps needed to examine a module in more detail.
- Section 2.4, “Commands for listing content” provides a reference of the commands useful for inspecting content in Application Stream.

2.1. SEARCHING FOR A PACKAGE

This section describes steps needed for finding a package providing a particular application or other content.

Prerequisites

- Name of the desired application or content must be known

Procedure

1. Search for a package with a text string, such as application name:

```
$ yum search "text string"
```

2. View details about a package:

```
$ yum info package
```

2.2. LISTING AVAILABLE MODULES

This section describes steps needed for finding what modules are available and what their details are.

Procedure

1. List module streams available to your system:

```
$ yum module list
```

The output of this command lists module streams with name, stream, profiles, and summary on a separate line.

2. Display details about a module, including a description, a list of all profiles, and a list of all provided packages:

```
$ yum module info module-name
```

3. Optional: If desired, display details about packages installed by each of module's profiles:

```
$ yum module info --profile module-name
```

4. Display the current status of a module, including enabled streams and installed profiles:

```
$ yum module list module-name
```

Additional resources

- Section 1.3, "Modules, streams, and profiles in Application Stream"

2.3. EXAMPLE: FINDING OUT DETAILS ABOUT A MODULE

This example shows how to locate a module in the Application Stream and how to find out more about its contents.

Procedure

1. List available modules:

```
$ yum module list
Name      Stream     Profiles                    Summary
(...)
php       7.1        devel, minimal,  PHP scripting language
                     default [d]
php       7.2 [d]    devel, minimal,  PHP scripting language
                     default [d]
(...)

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

2. Examine details of the **php** module:

```
$ yum module info php
Name             : php
Stream           : 7.2 [d]
Version          : 20181010120239
Context          : 76554e01
Profiles         : devel, minimal, default [d]
Default profiles : default
Repo             : appstream-8
Summary          : PHP scripting language
Description      : php 7.2 module
Artifacts        : apcu-panel-0:5.1.12-1.el8+1544+98b86041.noarch
                 : libzip-0:1.5.1-1.el8+1544+98b86041.x86_64
                 : libzip-devel-0:1.5.1-1.el8+1544+98b86041.x86_64
                 : libzip-tools-0:1.5.1-1.el8+1544+98b86041.x86_64
                 : php-0:7.2.11-1.el8+2002+9409c40c.x86_64
                 : php-bcmath-0:7.2.11-1.el8+2002+9409c40c.x86_64
(...)
Name             : php
```

```
Stream           : 7.1
Version          : 820181025145012
Context          : 76554e01
Profiles         : devel, minimal, default [d]
Default profiles : default
Repo             : appstream-8
Summary          : PHP scripting language
Description      : php 7.1 module
Artifacts        : apcu-panel-0:5.1.11-1.el8+1543+e18ce76f.noarch
                 : libzip-0:1.5.1-1.el8+1543+e18ce76f.x86_64
                 : libzip-devel-0:1.5.1-1.el8+1543+e18ce76f.x86_64
                 : libzip-tools-0:1.5.1-1.el8+1543+e18ce76f.x86_64
                 : php-0:7.1.20-2.el8+1700+11d526eb.x86_64
                 : php-bcmath-0:7.1.20-2.el8+1700+11d526eb.x86_64
(...)
```

Because no stream is specified, all streams are used for the listing.

3. Examine profiles available in stream **7.2** of the **php** module:

```
$ yum module info --profile php:7.2
(...)
Name    : php:7.2:20181010120239:76554e01:x86_64
devel   : libzip
        : php-cli
        : php-common
        : php-devel
        : php-fpm
        : php-json
        : php-mbstring
        : php-pear
        : php-pecl-zip
        : php-process
        : php-xml
minimal : php-cli
        : php-common
default : php-cli
        : php-common
        : php-fpm
        : php-json
        : php-mbstring
        : php-xml
```

Each of the profiles installs a certain set of packages, including their dependencies.

4. Install the **php** module using the default stream **7.2** and profile **default**:

```
# yum install @php
Dependencies resolved.
================================================================================
===========
 Package            Arch     Version
Repository     Size
================================================================================
===========
Installing group/module packages:
```

```
 php-cli            x86_64   7.2.11-1.el8+2002+9409c40c
appstream-8   3.1 M
 php-fpm            x86_64   7.2.11-1.el8+2002+9409c40c
appstream-8   1.6 M
Installing dependencies:
 nginx-filesystem   noarch   1:1.14.0-3.el8+1631+ba902cf0
appstream-8    23 k
Installing module profiles:
 php/default
Enabling module streams:
 httpd                      2.4
 nginx                      1.14
 php                        7.2

Transaction Summary
================================================================================
===========
Install  3 Packages

Total download size: 4.7 M
Installed size: 15 M
Is this ok [y/N]: y
(...)
```

The stream **7.2** is enabled and packages in its profile **default** installed.

5. Inspect the current status of the **php** module:

```
$ yum module list php
Name    Stream        Profiles                           Summary
php     7.1           devel, minimal, default [d]        PHP
scripting language
php     7.2 [d][e]    devel, minimal, default [d] [i]    PHP
scripting language

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

The output shows that the default stream **7.2** is enabled and its profile **default** is installed.

2.4. COMMANDS FOR LISTING CONTENT

This section lists commonly used commands for finding content and its details in Application Stream.

Command list

List available packages

```
$ yum list available
```

Search for a package using arbitrary text string

```
$ yum search "text string"
```

Display details for a package

```
$ yum info package
```

Find out which modules provide a package

```
$ yum module provides package
```

If the package is available outside any modules, the output of this command is empty.

List available modules

```
$ yum module list
```

Display details of a module

```
$ yum module info module-name
```

List packages installed by profiles of a module using the default stream

```
$ yum module info --profile module-name
```

Display packages installed by profiles of a module using a specified stream

```
$ yum module info --profile module-name:stream
```

Display the current status of a module

```
$ yum module list module-name
```

CHAPTER 3. INSTALLING CONTENT FROM APPLICATION STREAM

The following sections describe how to install content from Application Stream in Red Hat Enterprise Linux 8.

- Section 3.1, “Installing a package” includes steps for installing a package.
- Section 3.2, “Installing a module” describes steps to install sets of packages provided by modules.
- Section 3.3, “Running installed content” provides details for running content installed from Application Stream.
- Section 3.4, “Example: Installing a non-default stream of an application” shows an example of steps needed to install a set of packages in a non-default version.
- Section 3.5, “Commands for installing content from Application Stream” provides a reference of commands useful for installing content from Application Stream.

3.1. INSTALLING A PACKAGE

This section describes how to install packages from Application Stream.

Prerequisites

- Name of the package must be known

Procedure

- Install the package:

```
# yum install package
```

  - If the package is not provided by any module, this procedure is identical to the procedure used on previous versions of Red Hat Enterprise Linux.
  - If the package is provided by a module stream marked as default, the **yum** tool automatically transparently enables that module stream before installing this package.
  - If the package is provided by a module stream not marked as default, it is not recognized until you manually enable the respective module stream.

Additional resources

- Section 3.2, “Installing a module”
- Section 1.4, “YUM usage”

3.2. INSTALLING A MODULE

This section describes using a module to install the recommended set of packages from that module.

Procedure

- Install a module with a selected stream and in a chosen profile:

```
# yum install @module-name:stream/profile
```

This installs the recommended set of packages for a given stream (version) and profile (purpose) of the module.

Omit */profile* to use the default profiles. Additionally, omit *:stream* to use the default stream.

Additional resources

- Section 1.3, “Modules, streams, and profiles in Application Stream”
- Section 3.5, “Commands for installing content from Application Stream”

3.3. RUNNING INSTALLED CONTENT

Usually, after you install content from the Application Stream, new commands will be enabled. If the commands originated from a RPM package or RPM packages enabled by a module the experience of using the command should be no different. To run the new commands use them directly:

```
$ command
```

3.4. EXAMPLE: INSTALLING A NON-DEFAULT STREAM OF AN APPLICATION

This example shows how to install an application from a non-default stream (version).

More specifically, this example shows how to install the **PostgreSQL** server (package **postgresql-server**) in version **9.6**, while the default stream provides version **10**.

Procedure

1. List modules that provide the **postgresql-server** package to see what streams are available:

```
$ yum module list postgresql
Name              Stream       Profiles                   Summary
postgresql        10 [d]       client, default [d]        postgresql
module
postgresql        9.6          client, default [d]        postgresql
module

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

The output shows that the **postgresql** module is available with streams **10** and **9.6**. The default stream is **10**.

2. Install the packages provided by the **postgresql** module in stream **9.6**:

```
# yum install @postgresql:9.6
Dependencies resolved.
================================================================================
```

```
============
 Package             Arch     Version
Repository     Size
================================================================================
============
Installing group/module packages:
 postgresql-server   x86_64   9.6.10-1.el8+1547+210b7007
appstream-8   5.0 M
Installing dependencies:
 libpq               x86_64   10.5-1.el8
appstream-8   188 k
 postgresql          x86_64   9.6.10-1.el8+1547+210b7007
appstream-8   1.4 M
Installing module profiles:
 postgresql/default
Enabling module streams:
 postgresql                   9.6

Transaction Summary
================================================================================
============
Install  3 Packages

Total download size: 6.6 M
Installed size: 27 M
Is this ok [y/N]: y
(...)
Complete!
```

Because the installation profile was not specified, the default profile was used.

3. Verify the installed version of **PostgreSQL**:

```
$ postgres --version
postgres (PostgreSQL) 9.6.8
```

3.5. COMMANDS FOR INSTALLING CONTENT FROM APPLICATION STREAM

This section lists commonly used commands for installing content from Application Stream.

Command list

Install a package

```
# yum install package
```

If the package is provided by a module stream, **yum** resolves the required module stream, and enables it automatically while installing this package. This happens recursively for all package dependencies, too. If more module streams satisfy the requirement, the default ones are used.

Enable a module using its default stream

```
# yum module enable module-name
```

- Enable the module when you wish to make the packages available to the system but do not, at this time, wish to install any of them.

  Some modules may not define default streams. In such case, you must explicitly specify the stream.

Enable a module using a specific stream

```
# yum module enable module-name:stream
```

If the module defines a default stream, you can omit the stream and colon.

Install a module using the default stream and profiles

```
# yum install @module-name
```

Alternatively:

```
# yum module install module-name
```

CAUTION

Some modules do not define default streams.

Install a module using a specific stream and default profiles

```
# yum install @module-name:stream
```

Alternatively:

```
# yum module install module-name:stream
```

Install a module using a specific stream and profile

```
# yum install @module-name:stream/profile
```

Alternatively:

```
# yum module install module-name:stream/profile
```

CHAPTER 4. MANAGING VERSIONS OF APPLICATION STREAM CONTENT

Content in Application Stream can be available in multiple versions, corresponding to module streams. This chapter describes the operations you need to perform when changing the enabled module streams in other ways than only enabling new streams.

4.1. MODULAR DEPENDENCIES AND STREAM CHANGES

Traditionally, packages providing content depend on further packages, and usually specify the desired dependency versions. For packages contained in modules, this mechanism applies as well, but the grouping of packages and their particular versions into modules and streams provides further constraints. Additionally, module streams can declare dependencies on streams of other modules, independent of the packages contained and provided by them.

After any operations with packages or modules, the whole dependency tree of all underlying installed packages must satisfy all the conditions the packages declare. Additionally, all module stream dependencies must be satisfied.

As a result:

- Enabling a module stream can require enabling streams of further modules.
- Installing a module stream profile or installing packages from a stream can require enabling streams of further modules and installing further packages.
- Disabling a stream of a module can require disabling other module streams. No packages will be removed automatically.
- Removing a package can require removing further packages. If these packages were provided by modules, the module streams remain enabled in preparation for further installation, even if no packages from these streams are installed any more. This mirrors the behavior of an unused yum repository.
- Switching the stream enabled for a module is equivalent to resetting the current stream and enabling the new stream. This action does not automatically change any installed packages. Removing the packages provided by the previous stream and any packages that depend on them, and installation of the packages in the new stream are explicit manual operations.
- Directly installing a different stream of a module than the currently installed one is not recommended, due to potential upgrade scripts run during the installation.

Because some of the operations may require careful consideration, changing the enabled module streams does not automatically manipulate packages, so that the user has a complete control over the changes. The **yum** tool always provides a summary of the actions to do.

4.2. INTERACTION OF MODULAR AND NON-MODULAR DEPENDENCIES

Modular dependencies are an extension of regular RPM dependencies: Apart from regular package dependencies, package availability can depend on the enablement of module streams, and module streams can depend on other module streams.

Dependence of non-modular packages on modular ones is used in Application Stream only when the modular package is provided by a module stream marked as default.

For modular packages depending on non-modular ones, the system will always retain the module and stream choices, unless explicitly instructed to change them. A modular package will receive updates contained in the currently enabled stream of the module that provides this package, but will not upgrade to a version contained in a different stream.

4.3. REMOVING INSTALLED MODULES

Removing a module removes all of the packages installed by profiles of the currently enabled module stream, and any further packages and modules that depend on these.

Packages installed from this module stream not listed in any of its profiles remain installed on the system and can be removed manually.

Prerequisites

- A module which you want removed must have already installed some profiles.
- You must understand modular dependency resolution.

Procedure

1. Remove the module:

   ```
   # yum module remove module-name
   ```

   Replace *module-name* with the name of the module.

   This removes all packages installed from this module. The **yum** tool will present a summary of the changes and ask for confirmation.

   The currently enabled module stream remains enabled.

2. Disable the module stream:

   ```
   # yum module disable module-name
   ```

   Replace *module-name* with the name of the module.

   The **yum** tool will present a summary of the changes and ask for confirmation.

3. Finally, remove manually any packages that you installed from the module stream:

   ```
   # yum remove package ...
   ```

   The **yum** tool will present a summary of the changes and ask for confirmation.

4.4. SWITCHING MODULE STREAMS

Switching to a different module stream usually means upgrading or downgrading the content to a different version than the installed version.

Prerequisites

- A module stream must be enabled, and another stream of the module must exist.
- You understand modular dependency resolution.

Procedure

1. Install profiles of a different stream of the module:

```
# yum install @module-name:stream
```

   Replace *module-name* with name of the module, and *stream* with the desired stream.

   The new stream wil lbe enabled and the current stream disabled.

   The **yum** tool will present a summary of the changes and ask for confirmation. Changes to further module streams and packages can be necessary.

2. Update or downgrade any packages installed from the previous module stream and not listed in the profiles installed in the previous step:

```
# yum distro-sync
```

   The **yum** tool will present a summary of the changes and ask for confirmation.

3. Finally, remove manually any packages that remained installed from the previous module stream:

```
# yum remove package ...
```

   The **yum** tool will present a summary of the changes and ask for confirmation.