Red Hat Developer Tools 1

Using Go 1.18.4 Toolset

Installing and using Go 1.18.4 Toolset

Last Updated: 2022-11-22

Red Hat Developer Tools 1 Using Go 1.18.4 Toolset

Installing and using Go 1.18.4 Toolset

Jacob Valdez
jvaldez@redhat.com

Eva-Lotte Gebhardt
egebhard@redhat.com

Peter Macko

Kevin Owen

Vladimir Slavik

Zuzana Zoubkova

Legal Notice

Copyright © 2022 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

Go Toolset is a Red Hat offering for developers on the Red Hat Enterprise Linux (RHEL) operating system. Use this guide for an overview of Go Toolset, to learn how to invoke and use different versions of Go tools, and to find resources with more in-depth information.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.

CHAPTER 1. GO TOOLSET

Go Toolset is a Red Hat offering for developers on Red Hat Enterprise Linux (RHEL). It provides the Go programming language tools and libraries. Note that Go is alternatively known as golang.

Go Toolset is distributed as a part of Red Hat Developer Tools for Red Hat Enterprise Linux 7. Go Toolset is available as a module for Red Hat Enterprise Linux 8. Go Toolset is available as packages for Red Hat Enterprise Linux 9.

1.1. GO TOOLSET COMPONENTS

The following components are available as a part of Go Toolset:

Name	Version	Description
golang	RHEL 7 – 1.18.4, RHEL 8 – 1.18.4, RHEL 9 – 1.18.4	A Go compiler.
delve	RHEL 7 – 1.8.0, RHEL 8 – 1.8.0, RHEL 9 – 1.8.0	A Go debugger.

1.2. GO TOOLSET COMPATIBILITY

Go Toolset is available for Red Hat Enterprise Linux 7 and Red Hat Enterprise Linux 8 and Red Hat Enterprise Linux 9 on the following architectures:

- AMD and Intel 64-bit
- 64-bit ARM (RHEL 8 and RHEL 9)
- IBM Power Systems, Little Endian
- 64-bit IBM Z

1.3. GETTING ACCESS TO GO TOOLSET ON RED HAT ENTERPRISE LINUX 7

To be able to install Go Toolset on Red Hat Enterprise Linux 7, you must access and enable Red Hat Developer Tools and Red Hat Software Collections repositories.
If these repositories are already attached to your system, see Installing Go Toolset.

Procedure

1. Install **Wget** by running:

```
# yum install wget
```

2. Download the latest subscription data by running:

```
# subscription-manager refresh
```

3. Register your system by running:

```
# subscription-manager register
```

To register your system using a graphical user interface (GUI), follow the Registering and Unregistering a System guide.

4. Display a list of all available subscriptions and identify the pool ID by running:

```
# subscription-manager list --available
```

5. Find the pool ID on the line beginning with **Pool ID**.

6. Attach the subscription that provides access to the **Red Hat Developer Tools** repository to your system by running:

```
# subscription-manager attach --pool=<pool ID from the subscription>
```

- Replace ***<pool ID from the subscription>*** with the pool ID you identified in the previous step.

7. Verify which subscriptions are attached to your system by running:

```
# sudo subscription-manager list --consumed
```

8. Enable the **rhel-7-*variant*-devtools-rpms** repository by running:

```
# subscription-manager repos --enable rhel-7-<variant>-devtools-rpms
```

- Replace ***<variant>*** with your Red Hat Enterprise Linux system variant: **server** or **workstation**.
  Use **server** to access the widest range of development tools.

9. Enable the **rhel-*variant*-rhscl-7-rpms** repository by running:

```
# subscription-manager repos --enable rhel-<variant>-rhscl-7-rpms
```

- Replace ***<variant>*** with your Red Hat Enterprise Linux system variant: **server** or **workstation**.

10. Add the Red Hat Developer Tools GPG key to your system by running:

```
# cd /etc/pki/rpm-gpg
# wget -O RPM-GPG-KEY-redhat-devel https://www.redhat.com/security/data/a5787476.txt
# rpm --import RPM-GPG-KEY-redhat-devel
```

Additional resources

- For more information on registering your system and associating it with subscriptions, see the Red Hat Subscription Management guides collection.

1.4. INSTALLING GO TOOLSET

Complete the following steps to install Go Toolset, including all dependent packages.

Prerequisites

- On Red Hat Enterprise Linux 7, a subscription providing access to the Red Hat Developer Tools content set is attached to your system.
  To attach the subscription, see Getting access to Go Toolset on Red Hat Enterprise Linux 7 .
- All available Red Hat Enterprise Linux updates are installed.

Procedure

On Red Hat Enterprise Linux 7, install the **go-toolset-1.18** collection by running:

```
# yum install go-toolset-1.18
```

On Red Hat Enterprise Linux 8, install the **go-toolset** module by running:

```
# yum module install go-toolset
```

On Red Hat Enterprise Linux 9, install the **go-toolset** package by running:

```
# dnf install go-toolset
```

1.5. INSTALLING GO DOCUMENTATION

You can install documentation for the Go programming language on your local system.

Prerequisites

- Go Toolset is installed.
  For more information, see Installing Go Toolset.

Procedure

To install the **golang-docs** package, run the following command:

- On Red Hat Enterprise Linux 7:

  ```
  # yum install go-toolset-1.18-golang-docs
  ```

  You can find the documentation under the following path: **/opt/rh/go-toolset-1.18/root/usr/lib/go-toolset-1.18-golang/doc/docs.html**.

- On Red Hat Enterprise Linux 8:

  ```
  # yum install golang-docs
  ```

  You can find the documentation under the following path: **/usr/lib/golang/doc/effective_go.html**.

- On Red Hat Enterprise Linux 9:

  ```
  # dnf install golang-docs
  ```

You can find the documentation under the following path: **/usr/lib/golang/doc/go_spec.html**.

1.6. ADDITIONAL RESOURCES

- For more information on the Go programming language, tools, and libraries, see the official Go documentation.

CHAPTER 2. THE GO COMPILER

The Go compiler is a build tool and dependency manager for the Go programming language. It offers error checking and optimization of your code.

2.1. PREREQUISITES

- Go Toolset is installed.
  For more information, see Installing Go Toolset.

2.2. SETTING UP A GO WORKSPACE

To compile a Go program, you need to set up a Go workspace.

Procedure

1. Create a workspace directory as a subdirectory of **$GOPATH/src**.
   A common choice is **$HOME/go**.
2. Place your source files into your workspace directory.
3. Set the location of your workspace directory as an environment variable to the **$HOME/.bashrc** file by running:

   ```
   $ echo 'export GOPATH=<workspace_dir>' >> $HOME/.bashrc
   $ source $HOME/.bashrc
   ```

   Replace <*workspace_dir*> with the name of your workspace directory.

Additional resources

- The official Go workspaces documentation.

2.3. COMPILING A GO PROGRAM

You can compile your Go program using the Go compiler. The Go compiler creates an executable binary file as a result of compiling.

Prerequisites

- A set up Go workspace.
  For information on how to set up a workspace, see Setting up a Go workspace.

Procedure

In your project directory, run:

- On Red Hat Enterprise Linux 7:

  ```
  $ scl enable go-toolset-1.18 'go build <output_file>'
  ```

  - Replace ***<output_file<*** with the desired name of your output file and ***<go_main_package>*** with the name of your main package.

- On Red Hat Enterprise Linux 8:

```
$ go build <output_file>
```

  - Replace ***<output_file>*** with the desired name of your output file and ***<go_main_package>*** with the name of your main package.

- On Red Hat Enterprise Linux 9:

```
$ go build <output_file>
```

  - Replace ***<output_file>*** with the desired name of your output file and ***<go_main_package>*** with the name of your main package.

2.4. RUNNING A GO PROGRAM

The Go compiler creates an executable binary file as a result of compiling. Complete the following steps to execute this file and run your program.

Prerequisites

- Your program is compiled.
  For more information on how to compile your program, see Compiling a Go program.

Procedure

To run your program, run in the directory containing the executable file:

```
$ ./<file_name>
```

- Replace ***<file_name>*** with the name of your executable file.

2.5. INSTALLING COMPILED GO PROJECTS

You can install already compiled Go projects to use their executable files and libraries in further Go projects. After installation, the executable files and libraries of the project are copied to according directories in the Go workspace. Its dependencies are installed as well.

Prerequisites

- A Go workspace.
  For more information, see Setting up a Go workspace.

Procedure

To install a Go project, run:

- On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'go install <go_project>'
```

  - Replace ***<go_project>*** with the name of the Go project you want to install.

- On Red Hat Enterprise Linux 8:

```
$ go install <go_project>
```

- Replace **<go_project>** with the name of the Go project you want to install.

- On Red Hat Enterprise Linux 9:

```
$ go install <go_project>
```

- Replace **<go_project>** with the name of the Go project you want to install.

2.6. DOWNLOADING AND INSTALLING GO PROJECTS

You can download and install third-party Go projects from online resources to use their executable files and libraries in further Go projects. After installation, the executable files and libraries of the project are copied to according directories in the Go workspace. Its dependencies are installed as well.

Prerequisites

- A Go workspace.
  For more information, see Setting up a Go workspace.

Procedure

- To download and install a Go project, run:
  - On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'go get <third_party_go_project>'
```

    - Replace **<third_party_go_project>** with the name of the project you want to download.
  - On Red Hat Enterprise Linux 8:

```
$ go get <third_party_go_project>
```

    - Replace **<third_party_go_project>** with the name of the project you want to download.
  - On Red Hat Enterprise Linux 9:

```
$ go get <third_party_go_project>
```

    - Replace **<third_party_go_project>** with the name of the project you want to download.
- For information on possible values of third-party projects, run:
  - On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'go help importpath'
```

  - On Red Hat Enterprise Linux 8:

```
$ go help importpath
```

- On Red Hat Enterprise Linux 9:

```
$ go help importpath
```

2.7. ADDITIONAL RESOURCES

- For more information on the Go compiler, see the official Go documentation.
- To display the **help** index included in Go Toolset, run:
  - On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'go help'
```

  - On Red Hat Enterprise Linux 8:

```
$ go help
```

  - On Red Hat Enterprise Linux 9:

```
$ go help
```

- To display documentation for specific Go packages, run:
  - On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'go doc <package_name>'
```

  - On Red Hat Enterprise Linux 8:

```
$ go doc <package_name>
```

  - On Red Hat Enterprise Linux 9:

```
$ go doc <package_name>
```

    See Go packages for an overview of Go packages.

CHAPTER 3. THE GOFMT FORMATTING TOOL

Instead of a style guide, the Go programming language uses the **gofmt** code formatting tool. **gofmt** automatically formats your code according to the Go layout rules.

3.1. PREREQUISITES

- Go Toolset is installed.
  For more information, see Installing Go Toolset.

3.2. FORMATTING CODE

You can use the **gofmt** formatting tool to format code in a given path. When the path leads to a single file, the changes apply only to the file. When the path leads to a directory, all **.go** files in the directory are processed.

Procedure

To format your code in a given path, run:

- On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'gofmt -w <code_path>'
```

  - Replace **<*code_path*>** with the path to the code you want to format.

- On Red Hat Enterprise Linux 8:

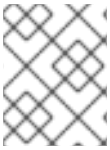
```
$ gofmt -w <code_path>
```

  - Replace **<*code_path*>** with the path to the code you want to format.

- On Red Hat Enterprise Linux 9:

```
$ gofmt -w <code_path>
```

  - Replace **<*code_path*>** with the path to the code you want to format.



NOTE

To print the formatted code to standard output instead of writing it to the original file, omit the **-w** option.

3.3. PREVIEWING CHANGES TO CODE

You can use the **gofmt** formatting tool to preview changes done by formatting code in a given path. The output in unified diff format is printed to standard output.

Procedure

To show differences in your code in a given path, run:

- On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'gofmt -d <code_path>'
```

- Replace **<code_path>** with the path to the code you want to compare.

- On Red Hat Enterprise Linux 8:

```
$ gofmt -d <code_path>
```

  - Replace **<code_path>** with the path to the code you want to compare.

- On Red Hat Enterprise Linux 9:

```
$ gofmt -d <code_path>
```

  - Replace **<code_path>** with the path to the code you want to compare.

3.4. SIMPLIFYING CODE

You can use the **gofmt** formatting tool to simplify your code.

Procedure

- To simplify code in a given path, run:
  - On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'gofmt -s <code_path>'
```

    - Replace **<code_path>** with the path to the code you want to simplify.
  - On Red Hat Enterprise Linux 8:

```
$ gofmt -s <code_path>
```

    - Replace **<code_path>** with the path to the code you want to simplify.
  - On Red Hat Enterprise Linux 9:

```
$ gofmt -s <code_path>
```

    - Replace **<code_path>** with the path to the code you want to simplify.

- To apply the changes, run:
  - On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'gofmt -w <code_path>'
```

    - Replace **<code_path>** with the path to the code you want to format.
  - On Red Hat Enterprise Linux 8:

```
$ gofmt -w <code_path>
```

- Replace ***<code_path>*** with the path to the code you want to format.

- On Red Hat Enterprise Linux 9:

```
$ gofmt -w <code_path>
```

  - Replace ***<code_path>*** with the path to the code you want to format.

3.5. REFACTORING CODE

You can use the **gofmt** formatting tool to refactor your code by applying arbitrary substitutions.

Procedure

- To refactor your code in a given path, run:
  - On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'gofmt -r <rewrite_rule> <code_path>'
```

    - Replace ***<code_path>*** with the path to the code you want to refactor and ***<rewrite_rule>*** with the rule you want it to be rewritten by.

  - On Red Hat Enterprise Linux 8:

```
$ gofmt -r <rewrite_rule> <code_path>
```

    - Replace ***<code_path>*** with the path to the code you want to refactor and ***<rewrite_rule>*** with the rule you want it to be rewritten by.

  - On Red Hat Enterprise Linux 9:

```
$ gofmt -r <rewrite_rule> <code_path>
```

    - Replace ***<code_path>*** with the path to the code you want to refactor and ***<rewrite_rule>*** with the rule you want it to be rewritten by.

- To apply the changes, run:
  - On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'gofmt -w <code_path>'
```

    - Replace ***<code_path>*** with the path to the code you want to format.

  - On Red Hat Enterprise Linux 8:

```
$ gofmt -w <code_path>
```

    - Replace ***<code_path>*** with the path to the code you want to format.

  - On Red Hat Enterprise Linux 9:

```
$ gofmt -w <code_path>
```

- Replace ***<code_path>*** with the path to the code you want to format.

3.6. ADDITIONAL RESOURCES

- The official gofmt documentation.

CHAPTER 4. THE GO RACE DETECTOR

Go Toolset includes the Go race detector, which is a tool of the Go standard library for finding race conditions.
Note that the race detector has a significant runtime resource overhead.

4.1. PREREQUISITES

- Go Toolset is installed.
  For more information, see Installing Go Toolset.

4.2. USING THE GO RACE DETECTOR

Use the Go race detector to check your code for race conditions.

Procedure

To use the race detector, run:

- On Red Hat Enterprise Linux 7:

```
$ scl enable go-toolset-1.18 'go build -race -o <output_file> <go_main_package>'
```

  - Replace ***<output_file>*** with the name of your executable file and ***<go_main_package>*** with the name of the package you want to test.

- On Red Hat Enterprise Linux 8:

```
$ go build -race -o <output_file> <go_main_package>
```

  - Replace ***<output_file>*** with the name of your executable file and ***<go_main_package>*** with the name of the package you want to test.

- On Red Hat Enterprise Linux 9:

```
$ go build -race -o <output_file> <go_main_package>
```

  - Replace ***<output_file>*** with the name of your executable file and ***<go_main_package>*** with the name of the package you want to test.

4.3. ADDITIONAL RESOURCES

- The official Go race detector documentation.

CHAPTER 5. CONTAINER IMAGES WITH GO TOOLSET

You can build your own Go Toolset containers from either Red Hat Enterprise Linux container images or Red Hat Universal Base Images (UBI).

5.1. RED HAT ENTERPRISE LINUX GO TOOLSET CONTAINER IMAGES CONTENTS

The Red Hat Enterprise Linux 7, Red Hat Enterprise Linux 8 and Red Hat Enterprise Linux 9 container images of Go Toolset contain the following packages:

Component	Version	Package
Go	1.18	RHEL 7 – go-toolset-1.18.4 RHEL 8 – go-toolset-1.18.4 RHEL 9 – go-toolset-1.18.4

5.2. ACCESSING RED HAT ENTERPRISE LINUX CONTAINER IMAGES

Pull the container image from the Red Hat registry before running your container and performing actions.

Procedure

To pull the required image, run:

- For an image based on Red Hat Enterprise Linux 7:

```
# podman pull registry.redhat.io/devtools/go-toolset-rhel7
```

- For an image based on Red Hat Enterprise Linux 8:

```
# podman pull registry.redhat.io/rhel8/go-toolset
```

- For an image based on Red Hat Enterprise Linux 9:

```
# podman pull registry.redhat.io/rhel9/go-toolset
```

5.3. ACCESSING THE UBI GO TOOLSET CONTAINER IMAGE ON RHEL 8

On RHEL 8, install the UBI Go Toolset container image to access Go Toolset.
Alternatively, you can install Go Toolset to the RHEL 8 base UBI container image. For further information, see Accessing Go Toolset from the base UBI container image on RHEL 8 .

Procedure

To pull the UBI Go Toolset container image from the Red Hat registry, run:

```
# podman pull registry.access.redhat.com/ubi8/go-toolset
```

5.4. ACCESSING GO TOOLSET FROM THE BASE UBI CONTAINER IMAGE ON RHEL 8

On RHEL 8, Go Toolset packages are part of the Red Hat Universal Base Images (UBIs) repositories, which means you can install Go Toolset as an addition to the base UBI container image. To keep the container image size small, install only individual packages instead of the entire Go Toolset. Alternatively, you can install the UBI Go Toolset container image to access Go Toolset. For further information, see Accessing the UBI Go Toolset container image on RHEL 8 .

Prerequisites

- An existing Containerfile.
  For information on creating Containerfiles, see the Dockerfile reference page.

Procedure

- To create a container image containing Go Toolset, add the following lines to your Containerfile:

```
FROM registry.access.redhat.com/ubi8/ubi:latest

RUN yum module install -y go-toolset
```

- To create a container image containing an individual package only, add the following lines to your Containerfile:

  ```
  RUN yum install -y <package-name>
  ```

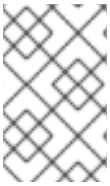
  - Replace ***<package-name>*** with the name of the package you want to install.

5.5. USING CONTAINER IMAGES AS SOURCE-TO-IMAGE BUILDER IMAGES ON RED HAT ENTERPRISE LINUX 7

You can use the Go Toolset container image as a Source-to-Image (S2I) builder image on Red Hat Enterprise Linux 7.

Procedure

1. Set the **IMPORT_URL** variable to a URL specifying the location of your code.
2. To build your S2I builder image, run the **s2i build** command.



NOTE

If the main package location is not identical with the location specified by the **IMPORT_URL** variable, set the **INSTALL_URL** variable to a URL that specifies the package location providing the application's main executable file when built.

Additional resources

- The OpenShift Container Platform Image Creation Guide .
- For more information on using Source-to-Image (S2I), see Using Red Hat Software Collections Container Images.

5.6. ADDITIONAL RESOURCES

- Go Toolset Container Images in the Red Hat Container Registry .
- Building Application Images Using Red Hat Software Collections Container Images .
- For more information on Red Hat UBI images, see Working with Container Images.
- For more information on Red Hat UBI repositories, see Universal Base Images (UBI): Images, repositories, packages, and source code.

CHAPTER 6. CHANGES IN GO TOOLSET

Go Toolset has been updated from version 1.17 to 1.18.4 on RHEL 7, RHEL 8, and RHEL 9.

Notable changes include:

- The introduction of generics while maintaining backwards compatibility with earlier versions of Go.
- A new fuzzing library .
- New **debug**/**buildinfo** and **net**/**netip** packages.
- The **go get** tool no longer builds or installs packages. Now, it only handles dependencies in **go.mod**.
- If the main module's **go.mod** file specifies **go 1.17** or higher, the **go mod download** command used without any additional arguments only downloads source code for the explicitly required modules in the main module's **go.mod** file. To also download source code for transitive dependencies, use the **go mod download all** command.
- The **go mod vendor** subcommand now supports a **-o** option to set the output directory.
- The **go mod tidy** command now retains additional checksums in the **go.sum** file for modules whose source code is required to verify that only one module in the build list provides each imported package. This change is not conditioned on the Go version in the main module's **go.mod** file.

For detailed information regarding the updates, see the upstream Go 1.18 Release Notes .