# Script for Building a Container Image for Fedora on RISC-V (RV64G)

Following are the commands that are used for generating a container image

```
#! /bin/bash


echo "Start of Script"

scratchcontainer=$(buildah from scratch)

scratchmnt=$(buildah mount $scratchcontainer)

dnf install --installroot $scratchmnt -y --releasever 38 @buildsys-build --setopt
install_weak_deps=false

buildah rename $scratchcontainer fedora38-riscv64g

buildah commit fedora38-riscv64g  fedora38-riscv64g-image

buildah unmount fedora38-riscv64g

podman run -it fedora38-riscv64g-image bash
```

## Description of each command within the script

```
buildah unshare
```

> *This Command is run before running the script. **`unshare`** launches a process (by default, `$SHELL`) in a new user namespace. The user namespace is configured so that the invoking user's UID and primary GID appear to be UID 0 and GID 0, respectively.*

**1) First command**

```
#! /bin/bash
```

> *Starting command for a script file. It is a special line at the beginning of a script that tells the operating system which interpreter to use when executing the script.*

- #! is known as Shebang or hashbang

**2) Second command**

```
scratchcontainer=$(buildah from scratch)
```

> ***`from`** instruction within buildah specifies the `parent image` from which the container is built. Here since the container being built is empty therefore parent image is set as `scratch`.*

**3) Third command**

```
scratchmnt=$(buildah mount $scratchcontainer)
```

> ***`mount`*** *instruction mounts the specified container's root file system in a location which can be accessed from the host, and returns its location.*

> *When running in rootless mode, `mount` runs in a different namespace so that the mounted volume might not be accessible from the host when using a driver different than vfs. To be able to access the file system mounted, we need to create the mount namespace separately as part of `buildah unshare`. In the environment created with `buildah unshare` we then use `buildah mount` and have access to the mounted file system.*

**4) Fourth command**

```
dnf install --installroot $scratchmnt -y --releasever 38 @buildsys-build --setopt
install_weak_deps=false
```

**Command used in dnf**

**- install**

dnf [options] install ...

> ***`install`*** *command makes sure that the given packages and their dependencies are installed on the system. Each can be either a , or a @, or a @. In this instance a `group-spec` has been provided by the name of `buildsys-build`.*

**options used in dnf**

**- -installroot=**

> *Specifies an alternative installroot, relative to where all packages will be installed, i.e instead of installing in the current `root` directory package installation takes place in a separate root directory that is mentioned after this option . This like doing `chroot <root> dnf`, except using `--installroot` allows dnf to work before the `chroot` is created. It requires absolute path.*

> *In this instance the root location for the empty container is mentioned.*

**-y or - -assumeyes**

> *Automatically answer yes for all questions.*

**- -releasever=**

> *Configure DNF as if the distribution release was . This can affect cache paths, values in configuration files and mirrorlist URLs.*

**- -setopt =**
=

--setopt install_weak_deps=false

> *Override a configuration option from the configuration file. Weak dependencies allow smaller minimal installations while keeping the default installation feature*

> *rich. They are add-ons on core functionality.*

**5) Fifth command**

```
buildah rename $scratchcontainer fedora38-riscv64g
```

> *`rename` command changes the container name from the default name i.e working-container to name specified.*

**6) Sixth command**

```
buildah unmount fedora38-riscv64g
```

> *`unmount` command unmounts the root file system on the specified working containers. in this instance it unmounts `fedora38-riscv64g` container.*

**7) Seventh command**

```
podman run -it fedora38-riscv64g-image bash
```

> *`run` command runs a process in a new container. It starts a process with its own file system, its own networking, and its own isolated process tree.*

**-i or - -interactive**

> *When set to true, keep stdin open even if not attached. The default is false.*

**--tty, -t**

> *Allocate a pseudo-TTY. The default is false.*

> *When set to true, Podman allocates a pseudo-tty and attach to the standard input of the container. This can be used, for example, to run a throwaway interactive shell.*