**Usage**

**Installing an SDK**

**Latest Stable**

Install the **latest stable** version of your SDK of choice (say, Java JDK) by running the following command:

$ sdk install java

You will see something like the following output:

Downloading: java 8u111

In progress...

######################################################################## 100.0%

Installing: java 8u111

Done installing!

Now you will be prompted if you want this version to be set as **default.**

Do you want java 8u111 to be set as default? (Y/n):

Answering *yes* (or *hitting enter*) will ensure that all subsequent shells opened will have this version of the SDK in use by default.

Setting java 8u111 as default.

**Specific Version**

Need a **specific** version of an SDK? Simply qualify the version you require:

$ sdk install scala 2.12.1

All subsequent steps same as above.

**Install Local Version(s)**

Need a snapshot? Already have a local installation? Setup a local version:

$ sdk install groovy 3.0.0-SNAPSHOT /path/to/groovy-3.0.0-SNAPSHOT

**Remove Version**

Remove an installed version.

$ sdk uninstall scala 2.11.6

**List Candidates**

To get a listing of available Candidates:

$ sdk list

This will render a searchable alphabetic list with name, current stable default version, website URL, description and easy install command for each Candidate. The output is piped to less so standard keyboard shortcuts may be used with q to exit.

================================================================================

Available Candidates

================================================================================

q-quit /-search down

j-down ?-search up

k-up h-help

--------------------------------------------------------------------------------

Groovy (2.4.5) http://www.groovy-lang.org/

Groovy is a powerful, optionally typed and dynamic language, with static-typing

and static compilation capabilities, for the Java platform aimed at multiplying

developers’ productivity thanks to a concise, familiar and easy to learn syntax.

It integrates smoothly with any Java program, and immediately delivers to your

application powerful features, including scripting capabilities, Domain-Specific

Language authoring, runtime and compile-time meta-programming and functional

programming.

$ sdk install groovy

--------------------------------------------------------------------------------

Scala (2.11.7) http://www.scala-lang.org/

...

**List Versions**

To get a listing of Candidate Versions:

$ sdk list groovy

This will result in a list view showing the available, local, installed and current versions of the SDK.

================================================================================

Available Groovy Versions

================================================================================

> \* 2.4.4 2.3.1 2.0.8 1.8.3

2.4.3 2.3.0 2.0.7 1.8.2

2.4.2 2.2.2 2.0.6 1.8.1

2.4.1 2.2.1 2.0.5 1.8.0

2.4.0 2.2.0 2.0.4 1.7.9

2.3.9 2.1.9 2.0.3 1.7.8

2.3.8 2.1.8 2.0.2 1.7.7

2.3.7 2.1.7 2.0.1 1.7.6

2.3.6 2.1.6 2.0.0 1.7.5

2.3.5 2.1.5 1.8.9 1.7.4

2.3.4 2.1.4 1.8.8 1.7.3

2.3.3 2.1.3 1.8.7 1.7.2

2.3.2 2.1.2 1.8.6 1.7.11

2.3.11 2.1.1 1.8.5 1.7.10

2.3.10 2.1.0 1.8.4 1.7.1

================================================================================

+ - local version

\* - installed

> - currently in use

================================================================================

**Use Version**

Choose to use a given version in the current terminal:

$ sdk use scala 2.12.1

It is important to realise that this will switch the candidate version **for the current shell only**. To make this change permanent, use the [default](http://sdkman.io/usage.html#default) command instead.

**Default Version**

Chose to make a given version the default:

$ sdk default scala 2.11.6

This will ensure that all subsequent shells will start with version 2.11.6 in use.

**Current Version(s)**

To see what is currently in use for a Candidate:

$ sdk current java

Using java version 8u111

To see what is currently in use for **all** Candidates:

$ sdk current

Using:

groovy: 2.4.7

java: 8u111

scala: 2.12.1

**Outdated Version(s)**

To see what is currently out of date for a Candidate on your system:

$ sdk outdated springboot

Outdated:

springboot (1.2.4.RELEASE, 1.2.3.RELEASE < 1.2.5.RELEASE)

To see what is outdated for **all** Candidates:

$ sdk outdated

Outdated:

gradle (2.3, 1.11, 2.4, 2.5 < 2.6)

grails (2.5.1 < 3.0.4)

springboot (1.2.4.RELEASE, 1.2.3.RELEASE < 1.2.5.RELEASE)

**SDKMAN! Version**

Display the current version of SDKMAN!:

$ sdk version

**Broadcast Messages**

Get the latest SDK release notifications on the command line:

$ sdk broadcast

==== BROADCAST =================================================================

\* 06/12/16: Scala 2.12.1 released on SDKMAN! #scala

\* 23/11/16: Gradle 3.2.1 released on SDKMAN! #gradle

\* 22/11/16: Ceylon 1.3.1 released on SDKMAN! #ceylonlang

================================================================================

It is also worth mentioning that whenever an SDK version is released on SDKMAN!, a notification will appear when next using the CLI. Every new broadcast is also pushed to Twitter.

**Offline Mode**

Initially called *Aeroplane Mode*, this allows SDKMAN! to function when working offline. It has a parameter that can be passed to*enable* or *disable* the offline mode.

$ sdk offline enable

Forced offline mode enabled.

$ sdk offline disable

Online mode re-enabled!

When operating in **offline** mode, most commands will still work even though they will operate in a scaled down capacity. An example is the list command, which will only display the currently installed and active version(s):

$ sdk list

------------------------------------------------------------

Offline Mode: only showing installed groovy versions

------------------------------------------------------------

> 2.4.4

\* 2.4.3

------------------------------------------------------------

\* - installed

> - currently in use

------------------------------------------------------------

The offline mode will also be disabled/enabled automatically when the internet becomes available/unavailable. Naturally, commands that require internet connectivity will not function but give a warning.

**Self-Update**

Installs a new version of SDKMAN! if available.

$ sdk selfupdate

If no new version is available an appropriate message will be displayed. Re-installation may be forced by passing the force parameter to the command:

$ sdk selfupdate force

Automatic daily checks for new versions of SDKMAN! will also be performed on the behalf of the user.

**Flush**

From time to time it may be necessary to flush SDKMAN!'s local state.The flush command helps with this and allows for the following to be performed:

**Candidates**

$ sdk flush candidates

Clears out the Candidate list. Opening a new terminal will fetch and store the latest list.This is usually required when a new Candidate is made available on SDKMAN!.

**Broadcast**

$ sdk flush broadcast

Clears out the Broadcast cache, downloading the latest available news on next command invocation.

**Archives**

$ sdk flush archives

Cleans the cache containing all downloaded SDK binaries. This can take up a lot of space so is worth clearing out from time to time!

**Temporary Folder**

$ sdk flush temp

Clears out the staging work folder used when installing new versions of candidates and SDKMAN! itself.

**Help**

You can get basic help by running the following command:

$ sdk help

This should yield something like:

Usage: sdk [version]

sdk offline

commands:

install or i [version]

uninstall or rm

list or ls

use or u [version]

default or d [version]

current or c [candidate]

outdated or o [candidate]

version or v

broadcast or b

help or h

offline

selfupdate [force]

flush

candidate : ...

version : where optional, defaults to latest stable if not provided

eg: sdk install groovy

**Configuration**

Although configuration is limited, the list of configurable items will grow as required. Configuration can be found in the ~/.sdkman/etc/config file. Currently the following is configurable:

# make sdkman non-interactive, preferred for CI environments

sdkman\_auto\_answer=true|false

# perform automatic selfupdates

sdkman\_auto\_selfupdate=true|false

# disables SSL certificate verification

# https://github.com/sdkman/sdkman-cli/issues/327

# HERE BE DRAGONS....

sdkman\_insecure\_ssl=true|false

# disable GVM alias, for users of the Go Version Manager

sdkman\_disable\_gvm\_alias=true|false

# configure curl timeouts

sdkman\_curl\_connect\_timeout=5

sdkman\_curl\_max\_time=4

# subscribe to the beta channel

sdkman\_beta\_channel=true