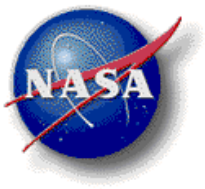
# Core Flight System
# Command and Data Dictionary Tool
# Installation Guide

## Engineering Directorate
## Software, Robotics, and Simulation Division

Version 2.0.24
November 2020






National Aeronautics and Space Administration
Lyndon B. Johnson Space Center
Houston, Texas 77058-3696

## Table of Contents

# 1.0 Overview

The Core Flight System Command and Data Dictionary (CCDD) is a software tool for managing the command and telemetry data for CFS and CFS applications. CCDD is written in Java™ and interacts with a PostgreSQL database, so it can be used on any operating system that supports the Java Runtime Environment (JRE) and PostgreSQL. CCDD is released as open source software under the NASA Open Source Software Agreement, version 1.3, and is hosted on GitHub.

This document describes the installation of the CCDD and its software dependencies.

Questions or comments concerning this document or the CCDD application can be addressed to:

Johnson Space Center
Software, Robotics, and Simulation Division
Spacecraft Software Engineering Branch, Mail Code ER6
Houston, TX 77058

# 2.0 Software Dependencies

The following open source software components are used by the CCDD tool, and hence, must be installed in order to run the CCDD.

## 2.1 Java

To install Java, go to www.java.com and locate the installation instructions appropriate for the operating system on which the application is to be run.

## 2.2 PostgreSQL

The PostgreSQL relational database management system is available for download from www.postgresql.org. The format appropriate for the target operating system must be used.

Once installed, PostgreSQL must be configured prior to use by the application. Configuration includes setting up the PostgreSQL server as a background service, creating database users and roles within the PostgreSQL server, and setting the desired level of password authentication. Extensive information on configuring PostgreSQL is available from www.postgresql.org.

## 2.3 Scripting Languages

CCDD supports the use of JVM-based scripting languages. At least one of these languages must be installed for the application to make use of CCDD's project-data-to-script-language interface. Only the scripting language(s) intended for use with the application need to be installed.

The tool was tested with five of the available languages: JavaScript, Python, Ruby, Groovy, and Scala. Details are provided in this in subsequent sections of this guide on the use of these five scripting languages; installation and use of other languages should be similar. The CCDD repository provides examples of common scripts in the JavaScript, Python, Ruby, and Groovy languages.

The scripting languages are not part of the CCDD package and must be installed separately on the platform from which the CCDD application is launched. The following links can be used to find further information on downloading and installing the scripting languages.

- *JavaScript*® is part of the JRE download and installation from www.java.com, so no further installation is necessary to use this scripting language. More information on JavaScript can be found at developer.mozilla.org.
- *Python*™ scripting is implemented using *Jython*, the Python implementation for Java. Jython can be downloaded from www.jython.org.
- *Ruby* scripting is implemented using *JRuby*, which implements Ruby in Java. JRuby is available for download from jruby.org.
- *Groovy* can be downloaded from www.groovy-lang.org.
- *Scala* can be downloaded from www.scala-lang.org.

# 3.0 Installation & Setup

## 3.1 PostgreSQL

This section describes the installation and setup of PostgreSQL relational database.

### 3.1.1 Installation

Execute the following commands as super user:

1. Pull the lattest versions of all installed components

```
$ sudo yum update
```

2. Install the required PostgreSQL packages

```
$ sudo yum install postgresql-server postgresql-contrib
```

3. Initialize PostgreSQL

```
$ sudo postgresql-setup initdb
```

4. Start PostgreSQL service

```
$ sudo systemctl start postgresql.service
```

5. Have PostgreSQL service starts at boot

```
$ sudo systemctl enable postgresql.service
```

### 3.1.2 Setup

Follow the steps below to setup new PostgreSQL user and password.

1. Execute

```
$ sudo passwd postgres
```

   and enter a new password for the "*postgres*" user.

2. Remain logged in as "*postgres*" user until step #6. Execute

```
$ su - postgres
```

   a. Change the user to "*postgres*" user.
   b. You should now be able to enter the PostgreSQL prompt by executing

```
$ psql postgres
```

   c. Exit by typing "\q" or "exit"

3. Create a new PostgreSQL user by executing

```
$ createuser --interactive -pwprompt
```

   a. Enter "<*new username*>"
   b. Enter the desired password, then answer "*Y*" to all the questions

4. Next, edit the *'PostgreSQL Client Authentication Configuration File*'.

   a. While still logged in as "*postgres*" user, execute

```
$ cd
```

   to make sure you are in the pgsql "home" directory, which should be "*/var/lib/pgsql*".

   b. Now edit the file, "*data/pg_hba.conf*"
   c. Scroll to the very bottom, where there are several lines starting with

"*# Type DATABASE USER ADDRESS METHOD*".

d. For the uncommented lines (There should be 3: 1 for local Unix sockets, 1 for IPv4, 1 for IPv6), replace the final column entry, i.e., "METHOD", with "*md5*".
e. Save and close the file.

5. Restart the postgresql service by executing

```
$ systemctl restart postgresql.service
```

6. Enter "<*Ctrl-D*>" to logout as "*postgres*" user
7. Add the new user created in step #3 to the "*postgres*" group by executing

```
$ sudo usermod -a -G postgres <new username>
```

PostgreSQL should now be fully configured for CCDD usage.

8. To verify that the new user created in step #3 has been properly setup,

a. Log out or restart your computer
b. Log back in and create a test database for the "<new username>" user by executing

```
$ createdb ccdd_test -O <new username>
```

c. Next, log into the psql shell by executing

```
$ psql ccdd_test
```

d. Enter "*du*" to verify that the "<*new username*>" user is configured with the attributes: *Superuser, Create role, and Create DB.*

## 3.2 CCDD

This section describe the installation and setup of the CCDD tool.

### 3.2.1 Installation

Execute the following steps to install the CCDD tool.

1. Clone a copy from NASA github repository

```
$ git clone https://github.com/nasa/ccdd.git
```

2. Go into the ccdd directory and checkout "CCDD-2" branch (highly recommended; version 1 is reaching end-of-life)

```
$ cd ccdd
$ git checkout CCDD-2
```

3. Launch CCDD

```
$ java -jar CCDD.jar
```

   If everything was configured correctly, you should get a login prompt. Enter the new username and password.

4. You should now be able to restore any compatible "*.dbu*" file through "*Project*->Restore" pulldown menu option.

### 3.2.2 Setup

To setup CCDD preferences,

1. From the CCDD pulldown menu at the top, select "*File->Preferences*", then the "*Path*" tab to set all of your paths.

   "*Script location*" and "*Project back-up*" should point to the proper subdirectories for scripts and databases. Set "*Script output*" should point to the directory for storing the CCDD generated data products.

   In the "*Other*" tab, you can set the "*Environment variable override*" to the "*home*" location of your CCDD databases and data products if you are managing them under the same directory.

## 4.0 Troubleshooting

You should not assume that CCDD.jar is fully self-contained: some versions of the jar are, and some are not. To be safe, always run CCDD from within its directory in the git repository, rather than copying just the jar file elsewhere. If you really feel the need to move the jar, then make sure the directory CCDD_lib moves with it.

# Appendix A. Acronyms

| | |
|---|---|
| CCDD | CFS Command & Data Dictionary |
| CCSDS | Consultative Committee for Space Data Systems |
| cFE | Core Flight Executive |
| CFS | Core Flight System |
| CPU | Central Processing Unit |
| CSV | Comma-Separated Values |
| DBU | Database Backup |
| EDS | Electronic Data Sheet |
| GUI | Graphical User Interface |
| HK | Housekeeping |
| I/O | Input/Output |
| ID | Identifier |
| ITOS | Integrated Test and Operations System |
| JAR | Java Archive |
| JDBC | Java DataBase Connectivity |
| JRE | Java Runtime Environment |
| JSON | JavaScript Object Notation |
| JVM | Java Virtual Machine |
| L&F | Look and Feel |
| OID | Object Identifier |
| OS | Operating System |
| PDF | Portable Document Format |
| PNG | Portable Network Graphics |
| SQL | Structured Query Language |
| SSL | Secure Sockets Layer |
| URL | Uniform Resource Locator |
| XML | Extensible Markup Language |
| XTCE | XML Telemetric and Command Exchange |