

Sign up on myMSC & Book Online

MSC

Sign Up

Beginners Guide to Cloudsim Project Structure

July 10, 2019 by [Anupinder Singh](#)

0
SHARES

f Share

🐦 Tweet

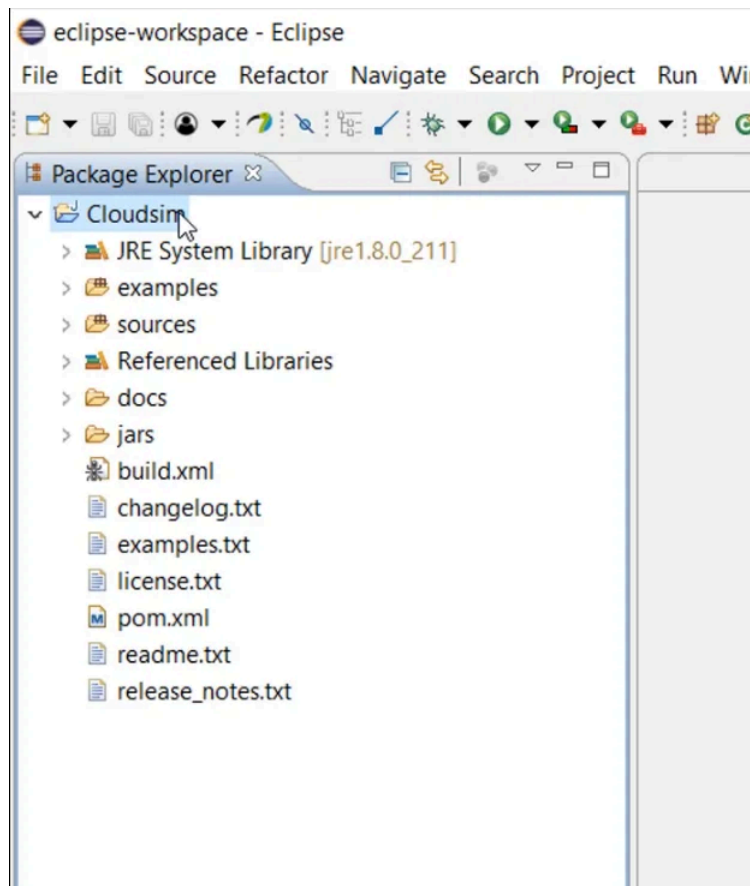
The Cloudsim Simulation Toolkit is an API that is written using the Java programming language and its classes are structured in a very specific way.

This article will serve as a guide to the Cloudsim project structure and will help you understand how the **Cloudsim architecture** is divided into different packages and their important classes that facilitate the cloud simulation using Cloudsim.

Let us first explore and understand the structure of this simulation project.

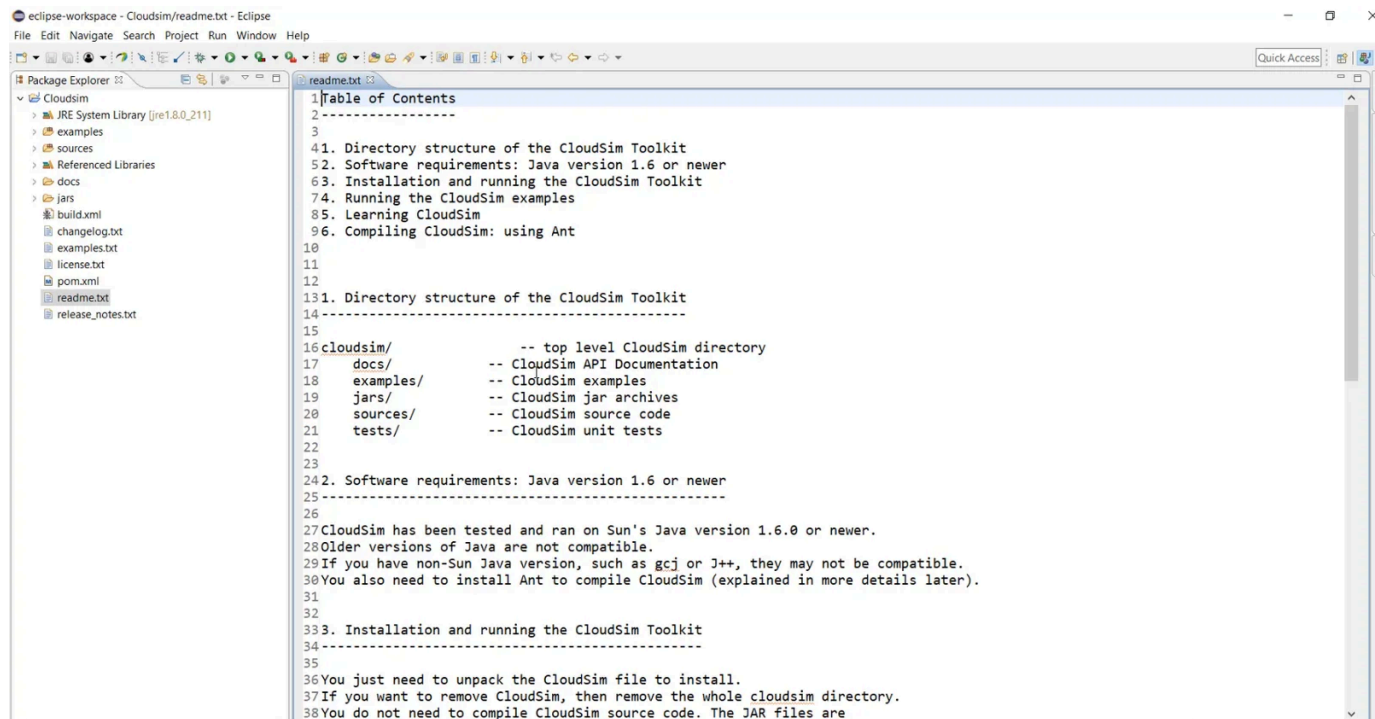
I assume that you have already set up the cloudsim using Eclipse if not you may follow my previous article on "[Cloudsim Setup using Eclipse](#)".

Let proceed further and your eclipse project explorer should contain 6 folders and 7 files.



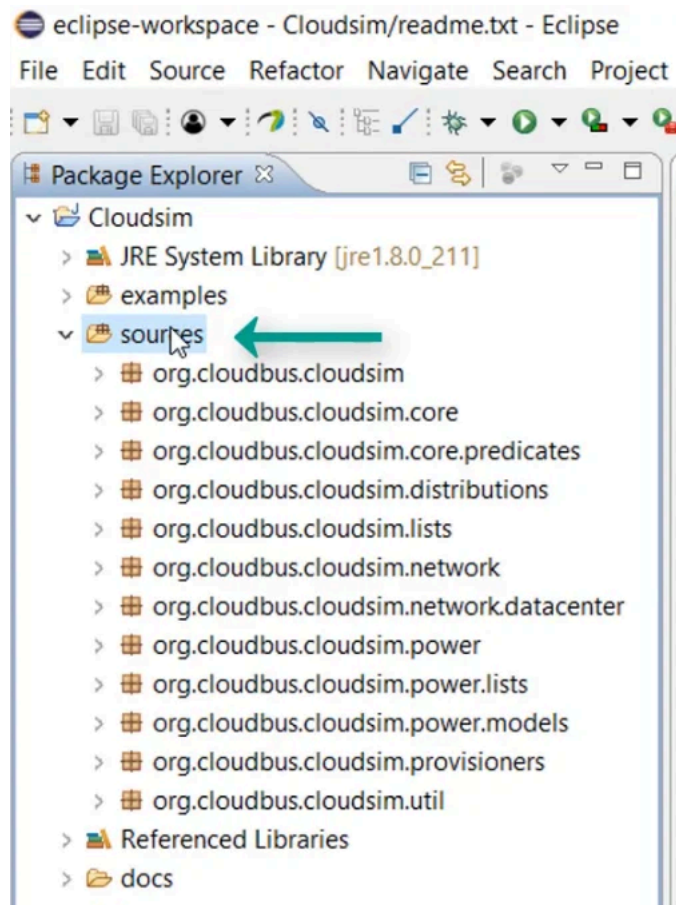
CloudSim Project Structure

The readme.txt file contains the necessary information about this project, like how to install and run the cloudsim, what is the basic directory structure, if you are not using any IDE then how to build the project source code.



Readme.txt file

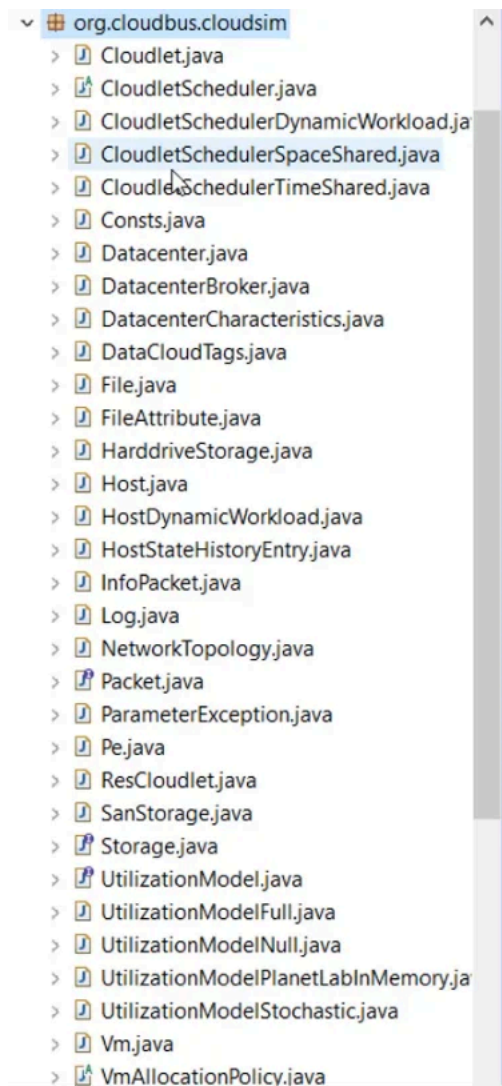
Now, let us look at the directory structure in detail and understand the various namespaces available in the "Source" folder



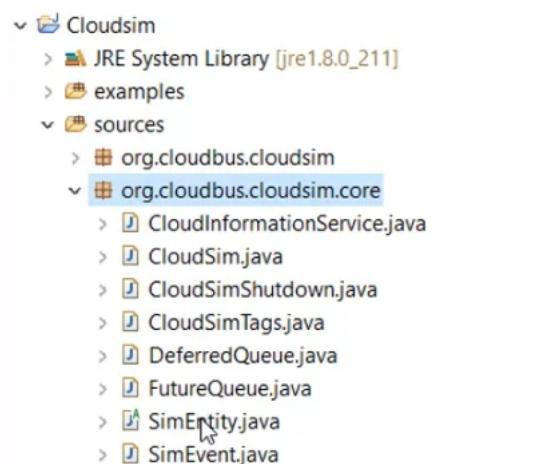
Cloudsim Source folder namespace/package view

There exist 12 namespaces; each namespace has a set of classes with specific correlated functions as discussed below:

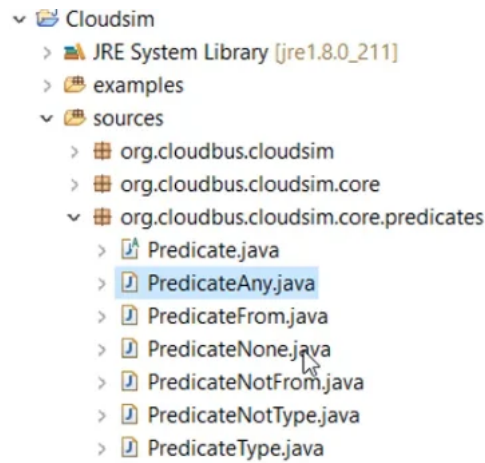
- **Org.cloudbus.cloudsim**: It contains the model classes of various basic hardware components, their groups, and the allocation/utilization methods, etc. Here, the model means that the classes contain the implementation of the various attributes and behaviors of the real-life cloud computing hardware component as a collection of Java methods. Once these classes initiate during the simulation, they are going to simulate the behavior of the real-life cloud-based system component.



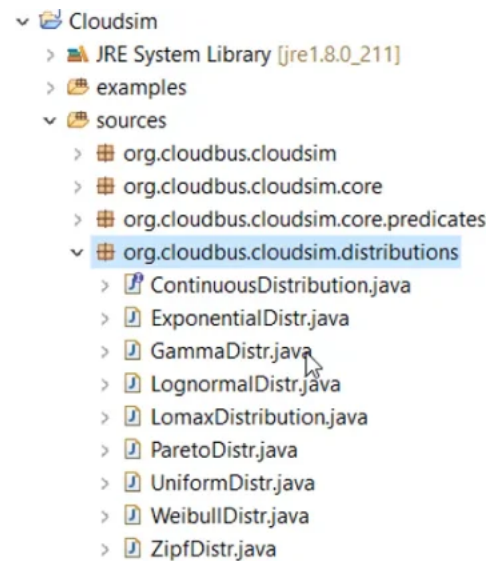
- **Org.cloudbus.cloudsim.core:** This namespace contains the implementation of the simulation engine where the `cloudsim.java` class is the main class and is responsible for the start and stop of the simulation process. Similarly, the `simentity.java` class is for maintaining the state of the simulated cloud components. `Simevent.java`, `futurequeue.java`, and `deferredqueue.java` are responsible for maintaining the inter-entity(cloud component) related event call during the simulation process.



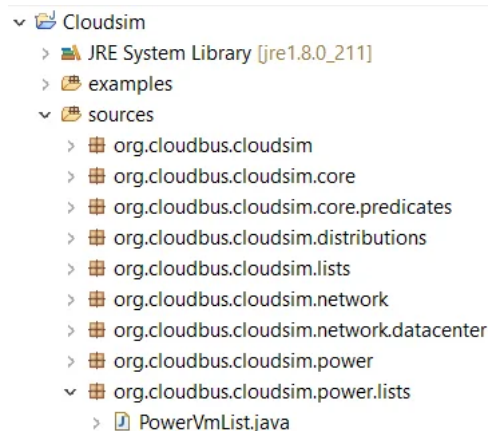
- **Org.cloudbus.cloudsim.core.predicates:** contains the classes related to the selection of events from the `deferredqueue.java` to be processed during the simulation event processing.



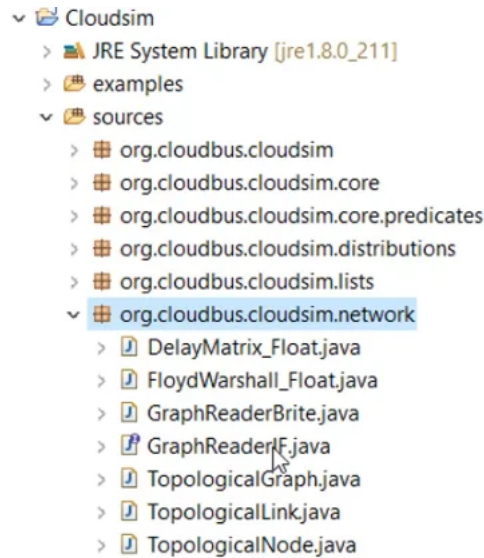
- **Org.cloudbus.cloudsim.distribution:** It contains the implementation of the various distribution function but is not used anywhere in the cloudsim. These classes might be for future purposes.



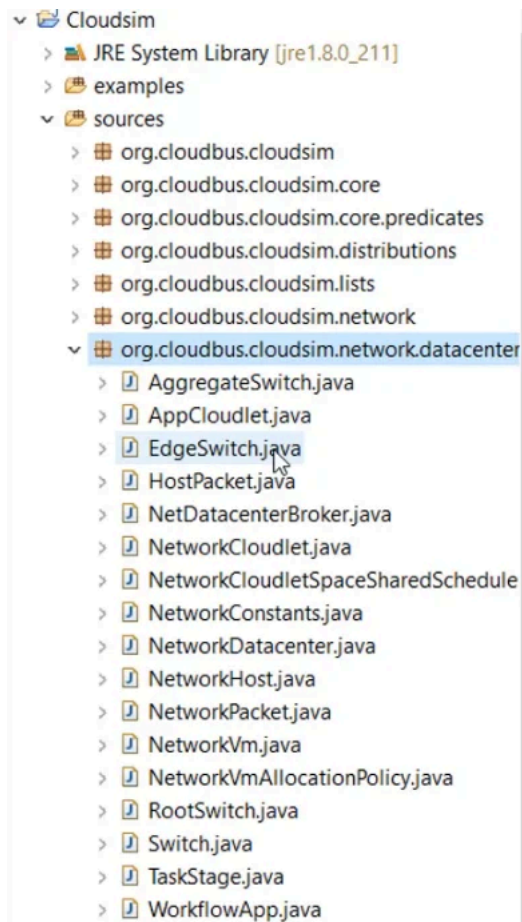
- **Org.cloudbus.cloudsim.lists:** It contains the implementation of lists to be used globally during the simulation process. These are the specialized set of classes, where sorting and comparison operation implemented. There exist the list class for the host, processing element, cloudlet, virtual machine, and resources.



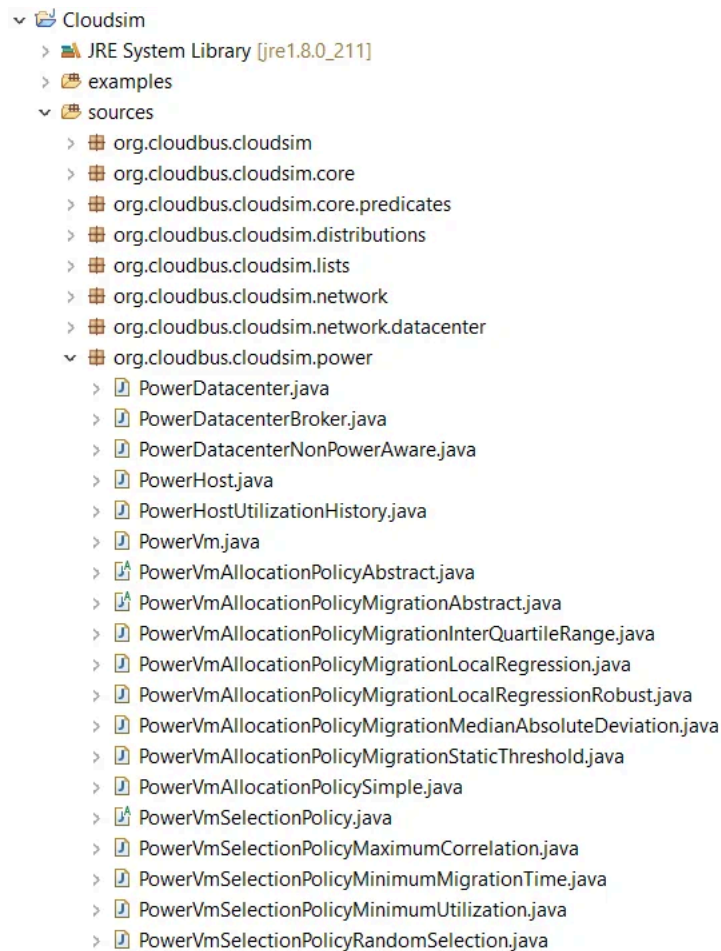
- **Org.cloudbus.cloudsim.network:** it contains the implementation for the network related simulations purpose, and if you are working on a research work based on network optimization techniques then this is the best set of classes to extend for simulation.



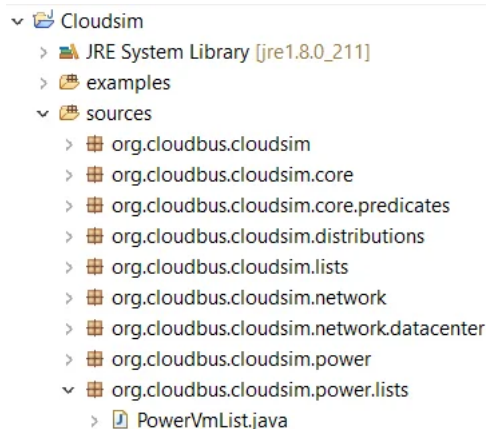
- **Org.cloudbus.cloudsim.network.datacenter:** This namespace contains the extended implementation of the basic cloud hardware components which can support the simulation of federated cloud functions distributed across the regions.



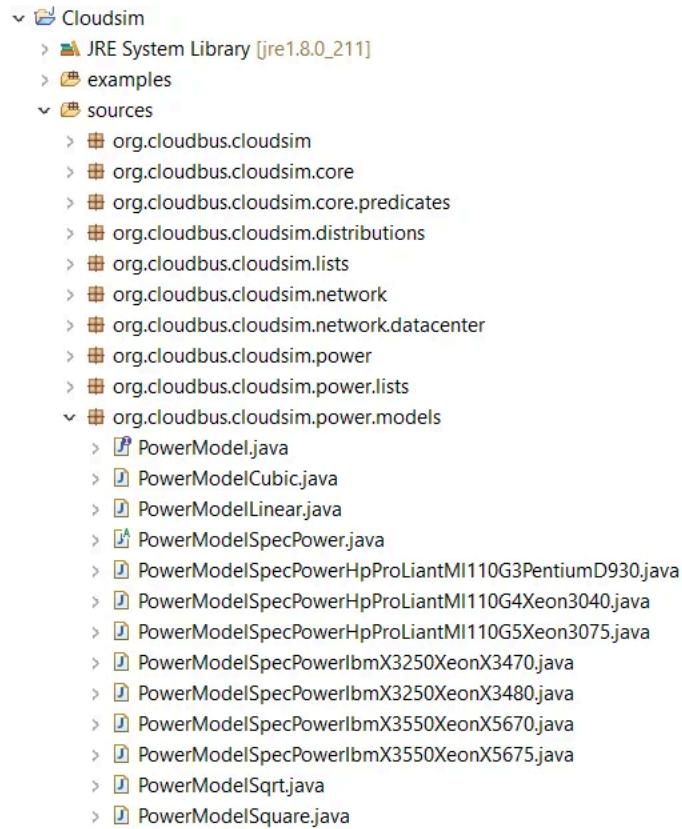
- **Org.cloudbus.cloudsim.power:** This package contains the extended implementation of cloud components, which can simulate the green or power-aware computing scenarios. This namespace implements the VM migration related classes where extensive algorithm codes for virtual machine selection and allocation related to migrations are available.



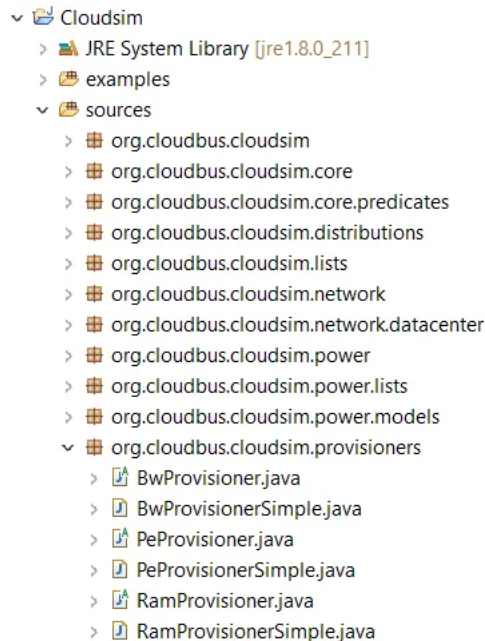
- **Org.cloudbus.cloudsim.power.lists:** This namespace contains only the implementation of powervmList.java, which is an extended version of the VmList.java class of org.cloudbus.cloudsim.list namespace.



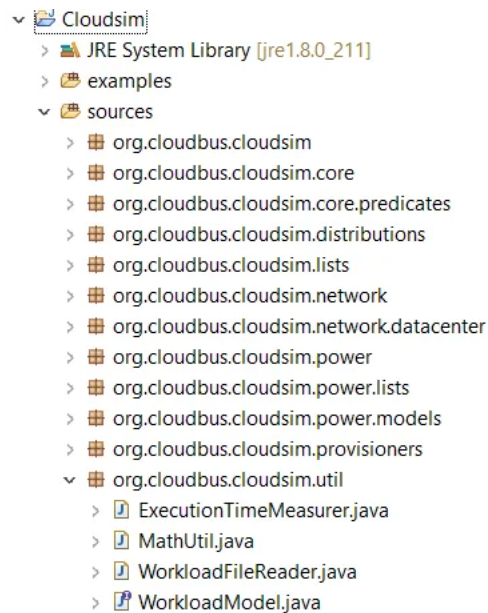
- **Org.cloudbus.cloudsim.power.models:** It contains the set of classes that specifies the power configurations of servers as model classes. These classes imitate the actual working of the various server machine brands available in the market and help in determining the power consumption by such machines during the simulation process.



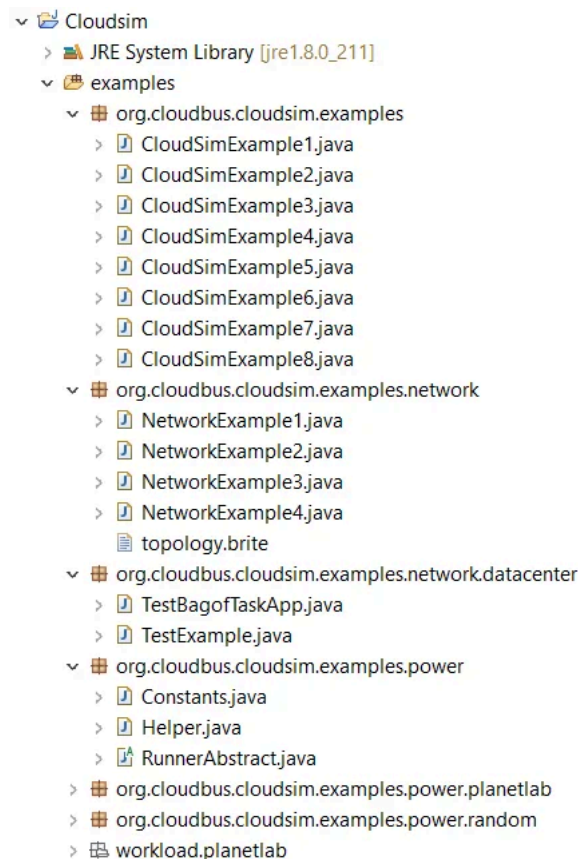
- **Org.cloudbus.cloudsim.provisioners:** This namespace contains the behavior implementation regarding how the specified cloud component provisioned to requesting virtual resources.



- **Org.cloudbus.cloudsim.util:** this namespace contains the implementation of essential calculation functions and a general utility set.



Now let's look at the "examples" folder. It contains 7 namespaces, and each namespace contains a particular type of scenario implementations.

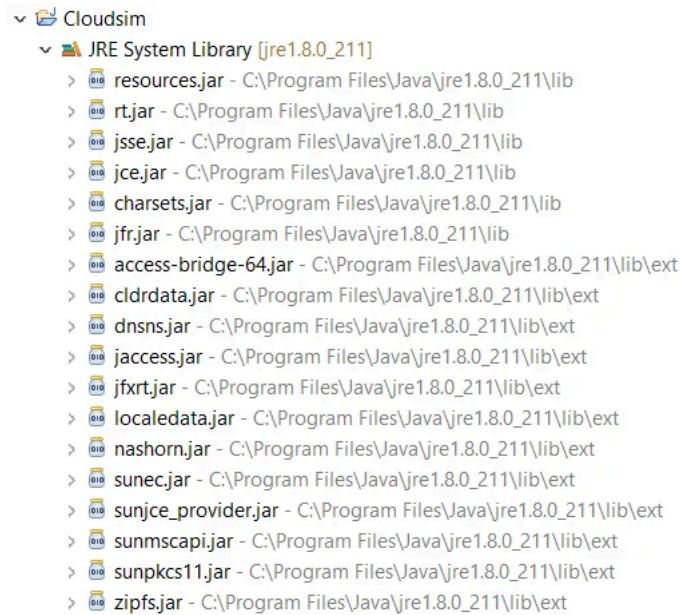


Examples.txt gives you an overview of various basic example scenarios that are available in this folder. The file contains the instructions for the compile and execution of example classes.

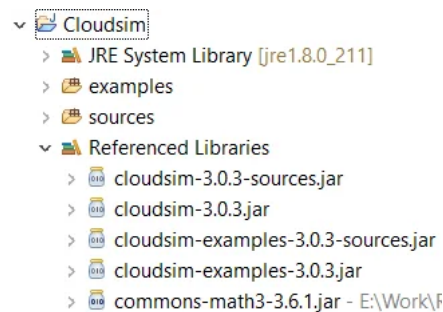
Workload.planetlab: It contains a real dataset containing a set of CPU utilization traces from PlanetLab VMs collected during the 10 random days in March & April of 2011. The PlanetLab dataset is used as a benchmark dataset for the validation of various research scenarios.

Now let's move to the other folders:

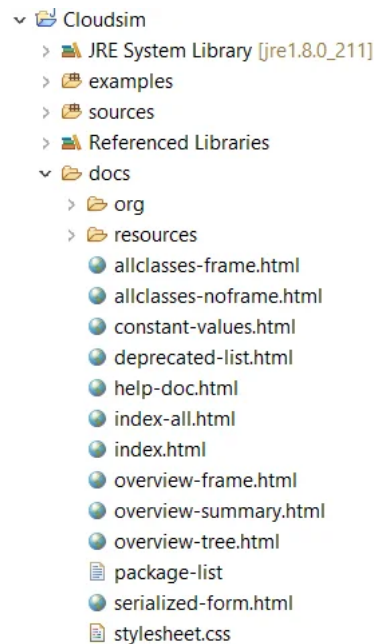
- **JRE system library:** It is automatically created during the first build process of the CloudSim by the eclipse.



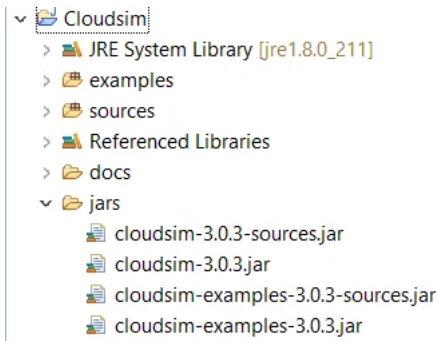
- **Reference Libraries:** It contains the various dependency jars again automatically added by the eclipse during the first build process.



- **Docs:** Contains the minimalist documentation of the cloudsim class API and is automatically generated from the inline comments.



- **Jars:** This folder is supplied with the source zip file and contains the pre-compiled build of the cloudsim 3.0.3. It can be used for creating custom project scenarios whenever required. We will work on this in further blog posts.



- **Build.xml** is an ANT based XML file used during the console based builds.
- **Changelog.txt** contains the history of the changes done to cloudsim during various version releases.
- **License.xml** contains GNU-GPL notification.
- **Pom.xml** is a maven configuration file used during the setup of cloudsim using eclipse.
- **Releasenotes.txt** contains the note from the cloudsim development team.

You may also follow the detailed step by step demo to "[Cloudsim Tutorial: Introduction to Package Intro \(Part 2\)](#)"

Cloudsim Tutorials: Understanding Cloudsim Namespace



Learn More

Also, To quickly get started with the Cloudsim Simulation Toolkit, Feel free to join our learners community for an online self-paced course named "[Essential Cloudsim Tutorials](#)" I would be interested in interacting with you. for further discussion.

📁 [Uncategorized](#)

🔖 [cloudsim introduction](#)

< [How to do Virtual machine and Task Scheduling in CloudSim](#)

> [Guide to CloudsimExample1.java simulation workflow](#)

Leave a Comment



[About](#)[Cloudsim Training](#)

Archives

[April 2023](#)[May 2022](#)[January 2021](#)[December 2020](#)[April 2020](#)[January 2020](#)[December 2019](#)[October 2019](#)[September 2019](#)[July 2019](#)[June 2019](#)[May 2019](#)

Quick Links

[About](#)[Cloudsim Training](#)

Recent Posts

[Edge Computing Research Trends Newsletter -April 2023](#)[Cloud Computing Research Trends- May 2022](#)[Simulating Virtualization in Cloudsim](#)[iFogsim Entities](#)[iFogsim Project Structure: A Beginners Guide](#)

Cloudsim Courses

[Cloudsim One to One Coaching Session](#) [Cloudsim and iFogsim Course Bundle](#)

© 2024 Cloudsim Tutorials • Built with [GeneratePress](#)