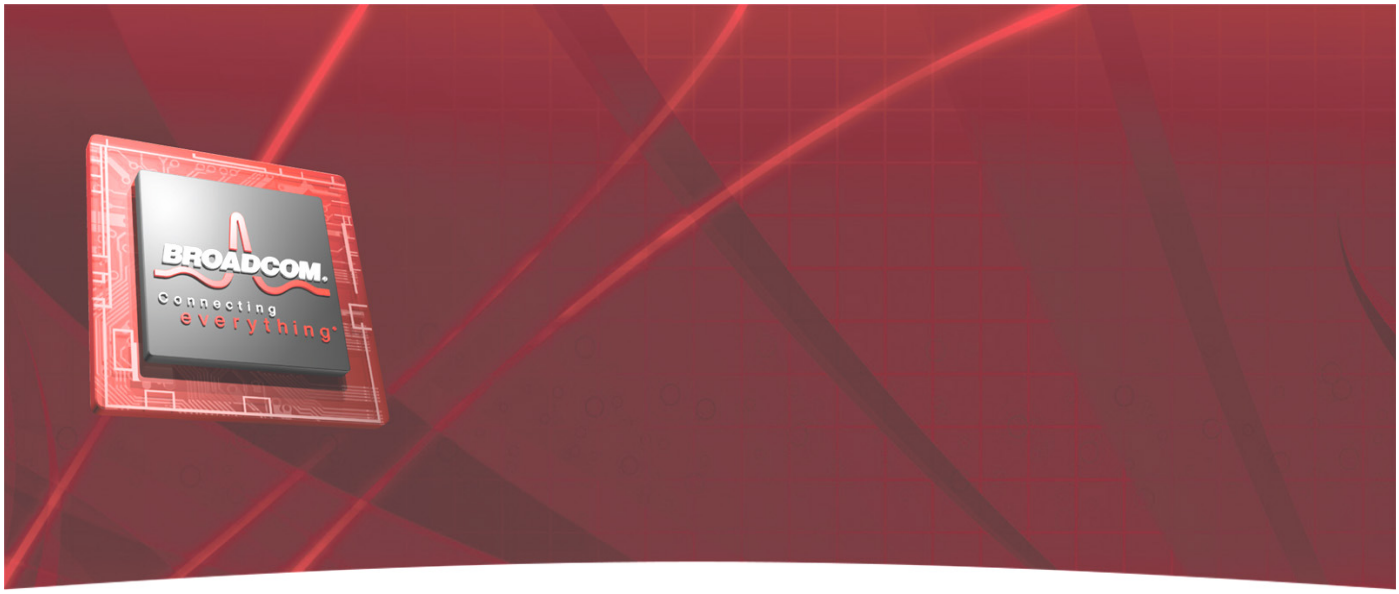

OSGi/Java Virtual Machine Development Guide

Broadcom Corporation
5300 California Avenue
Irvine, California, USA 92677
Phone: 949-926-5000
Fax: 949-926-5203

Web: www.broadcom.com

Revision History

Revision	Date	Change Description
0.1	11/08/2010	Initial draft
0.2	3/29/2011	Add configuration of R/W access of rootfs
0.3	8/10/2011	Add Heap and stack size configuration
0.4	9/15/2011	Add MAVEN_OPTS to increase size of memory in javac
0.5	2/19/2013	Update for 4.14L.01 release

Table of Contents

Introduction

This document describes the software tools required to build the OSGi/Jvm feature released in Release 4.10 and beyond. Detail instructions for installation of such tools will be specified. A developer of OSGi bundle must use the specified tools and libraries. There are many versions of software tools available in the opensource community. At the time of development of this feature, Broadcom picked the latest version of each tool and software implementation as specified in this document. Developers can choose to install different versions of such open source implementation. Broadcom will not guarantee these different versions would work as in our software release. Broadcom will not be able to experiment the different versions nor help debug such issues. Often times, new release versions' features or bug fixes could break existing features if users do not update all the tools all together. When necessary, Broadcom will perform software upgrade of such tools in future release.

Before reading this document, please read document ***BCA Modular Software Download*** (CPE-AN100-R) from DocSafe. This is a higher level document that provides more context and background information about Modular Software, Execution Environments, and build instructions in the latest BCA CPE Reference Software 4.14L.01. This document contains only OSGi specific information not covered by ***BCA Modular Software Download***.

Build Requirement

To build OSGI/Jvm, one needs to enable this feature in build profile using make menuconfig. On host machines, Java compiler and Maven tools need to be retrieved from opensource distribution and installed. Detailed make menuconfig instructions are contained in the BCA Modular Software Download document.

The following provides information about the specific settings:

1. Make menuconfig

    i. Initial Heap Size of JVM in kbytes unit (default is 4000 which is 4M)

    ii. Maximum Heap Size of JVM in Kbytes unit (default is 8000 which is 8M)

    iii. Stack Size per thread in Kbytes unit (default 256 which is 256K)

    Java heap space is a part of memory allocated to JVM by the operating system. The JVM's heap is where all objects and shared types created by an executing Java program are stored. Garbage collection is the process that frees objects that are no longer referenced by a program. Garbage collection also has to deal with heap

fragmentation where old blocks of memories are claimed leaving memory fragmented in between the allocated live objects. Garbage collection creates overhead although it relieves programmers' job of memory management. The initial heap size is the amount of memory allocated at the start of JVM. The maximum heap size is the maximum size OS would give to JVM. Garbage collection occurs when heap memory reaches a certain threshold (such as when heap space falls less than 25%). Setting initial heap and max heap size to be the same is generally not a good idea because this delays the start of garbage collection process. Garbage collection at this time will take longer since it takes more time to de-fragment the heap. However, performance is also impacted if garbage collection happens too often though the process takes less time in defragmentation. Depending on the types of applications run on the JVM and available memory in a platform, the heap sizes can be fine-tuned for optimal performance.

Java stack size is the approximate number of bytes of address space that a JVM will allocate to a thread's stack. The stack size determines how many threads one can have. OutOfMemoryError with message such as "cannot create native thread" means that the program is creating too many threads and the thread size is too large. If StackOverflowError is seen, this means that the stack size is too small. However, most often this problem is caused by an infinite recursive loop rather than the stack being too small.

- Enable https support in WGET (under Major Features, Busybox)

- Enable R/W access of ROOTFS (under Root File System Selection menu)

2.Install Java SE Development—Java JDK 1.6 required

3. Install Maven

The following sub-sections describes in detail of the installation of Java and Maven on host machines.

OSGi Implementation

The open source OSGi R4 implementation included in the release is Apache Felix Framework Distribution version 3.0.1. The full distribution can be retrieved from http://felix.apache.org/site/downloads.cgi. In our code release, some directories are being stripped to reduce the size of the tarball. This document will not cover detail of Felix implementation of Felix. More info can be obtained from Felix's web site http://felix.apache.org/site/index.html.

Maven

As for any distribution of OSGi implementation, Apache Felix framework requires tools to compile its software. Apache Maven needs to be installed to build Felix framework. Maven is a tool to build and manage a Java-based project. More detail of Apache Maven can be found at this website: http://maven.apache.org/. Some other popular development using Maven include Eclipse, Netbeans, IntelliJ and Jbuilder.

Maven dynamically downloads Java libraries and Maven plug-ins from one or more repositories. Therefore, during the very first run of Maven software included in the release, the host machine needs to have internet access so that Maven can go to the repositories to download the necessary libraries to the host machine for this project (in this case for compiling Felix). The whole process will take total of about 10 minutes.

Maven Installation to Host Machine

Maven-2.2.1 (Apache-Maven 2.2.1-bin.tar.bz2) can be downloaded from http://maven.apache.org/download.html.

The following installation instruction of Maven is assuming that users choose to install Maven-2.2.1 under /usr/local/ directory.

1. Extract the Maven distribution archive:

   tar –xfvz apache-maven2.2.1-bin.tar.bz2

2. Add apache-maven path to system.

   Example –add the following lines to /etc/profile file:

```
#set Java environment
JAVA_HOME=/usr/java/jdk1.6.0_18
PATH=$JAVA_HOME/bin:$PATH
#set Maven environment
M2_HOME=/usr/local/apache-maven-2.2.1
M2=$M2_HOME/bin
PATH=$M2:$PATH
export JAVA_HOME M2 M2_HOME
```

export MAVEN_OPTS="-Xms512m -Xmx960m -XX:PermSize=128m -XX:MaxPermSize=300m"

3. Dynamic libraries will be downloaded when Felix is compiled the first time; the total time would take approximately 10 minutes. The following is a screen shot of the ending result of such process when building Felix. This will be done only once. The second time Felix compiles, it will be a lot faster not having to download dynamic libraries.

run "mvn -Dpackaging=plugins install"
For the very first run, this will take about 5 minutes.

```
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO] ------------------------------------------------------------------------
[INFO] OBR Maven Plugin ..... SUCCESS [2:59.007s]
[INFO] Maven Bundle Plugin ..... SUCCESS [49.062s]
[INFO] Annotations for SCR ..... SUCCESS [0.900s]
[INFO] Maven SCR Plugin ..... SUCCESS [18.201s]
[INFO] Maven OSGi Plugin ..... SUCCESS [10.198s]
[INFO] Apache Felix ..... SUCCESS [39.420s]
[INFO] ------------------------------------------------------------------------
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 5 minutes 1 second
[INFO] Finished at: Wed Jun 23 21:01:39 PDT 2010
[INFO] Final Memory: 19M/76M
[INFO] ------------------------------------------------------------------------
```

run "mvn -Dpackaging=bundle install"
For the very first run, this will take approximately 5 minutes.

```
[INFO] Installing /home/yentran/myviews/yentran_laptop_4.06jvm/CommEngine/userspace/gpl/apps/Trunk/pom.xml to
/home/yentran/.m2/repository/org/apache/felix/reactor/1/reactor-1.pom
[INFO]
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO] ------------------------------------------------------------------------
[INFO] OSGi R4 Foundation EE ..... SUCCESS [4.091s]
[INFO] Servlet 2.1 API ..... SUCCESS [1.219s]
[INFO] OSGi R4 Core Bundle ..... SUCCESS [1.911s]
[INFO] OSGi R4 Compendium Bundle ..... SUCCESS [1:39.485s]
[INFO] Apache Felix Framework ..... SUCCESS [8.973s]
[INFO] Apache Felix Main ..... SUCCESS [1:53.746s]
[INFO] Apache Felix Gogo ..... SUCCESS [0.757s]
[INFO] Apache Felix Gogo Runtime ..... SUCCESS [6.289s]
[INFO] Apache Felix Gogo Shell ..... SUCCESS [10.273s]
[INFO] Apache Felix Gogo Command ..... SUCCESS [5.908s]
[INFO] Apache Felix Gogo ..... SUCCESS [0.007s]
[INFO] Apache Felix ..... SUCCESS [0.005s]
[INFO] ------------------------------------------------------------------------
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 4 minutes 35 seconds
[INFO] Finished at: Wed Jun 23 21:27:30 PDT 2010
[INFO] Final Memory: 39M/94M
[INFO] ------------------------------------------------------------------------
```

Maven install Notes

On a recent download of Maven (Oct 2012), two required libraries were missing. The following were used instead and seems to work.

- org.apache.felix.webconsole-4.0.0.jar
- javax.servlet-1.0.0.jar

How to Include library in Maven Depository

There are many Java libraries not located in Maven depository. Sometimes there are private custom libraries as well. The following is a guide to add a library into Maven.

1. Assume the private library is located on c:\privateLib-1.0.jar
2. Install privateLib.jar to Maven by using the following command:

```
mvn install:install-file -Dfile=c:\privateLib-1.0.jar -
DgroupId=com.broadcom.code
-DartifactId=private -Dversion=1.0 -Dpackaging=jar
```

Result :

```
D:\>mvn install:install-file -Dfile=c:\privateLib-1.0.jar -
DgroupId=com.broadcom.code
-DartifactId=private -Dversion=1.0 -Dpackaging=jar
[INFO] Scanning for projects...
[INFO] Searching repository for plugin with prefix: 'install'.
[INFO] ------------------------------------------------------------------------
--
[INFO] Building Maven Default Project
[INFO]    task-segment: [install:install-file] (aggregator-style)
[INFO] ------------------------------------------------------------------------
--
[INFO] [install:install-file]
[INFO] Installing c:\privateLib-1.0.jar to
D:\maven_repo\com\broadcom\code\private\1.0\privateLib-1.0.jar
[INFO] ------------------------------------------------------------------------
--
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
--
[INFO] Total time: < 1 second
[INFO] Finished at: Tue August 2 13:41:42 PT 2011
[INFO] Final Memory: 3M/6M
[INFO] ------------------------------------------------------------------------
--
```

Now, privateLib jar library is included in the local Maven depository.

3.Modify pom.xml file

```
<dependency>
    <groupId>com.broadcom.code</groupId>
    <artifactId>private</artifactId>
    <version>1.0</version>
</dependency>
```

More reading:

http://maven.apache.org/plugins/maven-install-plugin/index.html

Additional Development Information

This web site contains useful information for developing an OSGi bundles using Eclipse.

http://felix.apache.org/site/integrating-felix-with-eclipse.html

JAVA Virtual Machine

The open source implementation of Java Virtual Machine is Jamvm version 1.5.4. The software distribution and details of Jamvm can be retrieved from this web site http://jamvm.sourceforge.net/. Jamvm requires GNU classpath libraries version 0.98. Website http://www.gnu.org/software/classpath/announce/20090205.html contains distribution and full documentation of the software.

Java JDK Installation

Java JDK version 1.5 was used for Data Model Designer. However, this version is broken for compiling Jamvm code. Java JDK 1.6 or newer is needed. This version is also working with Data Model Designer.

Install Java SE Development Kit 6u18 (1.6.0.18) for Linux. Download JDK at http://java.sun.com/products/archive/j2se/6u18/index.html.

See installation instruction at http://java.sun.com/javase/6/webnotes/install/jdk/install-linux-RPM.html.

Updating OSGi/JVM Software Modules (Bundles)

OSGi Software Modules are called Bundles or Deployment Units (TR-157 term). OSGi Software Modules/Bundles/Deployment Units can be downloaded, installed, updated, and/or deleted from the system independent of the Base/Primary Firmware image. The instructions for doing this are in the ***BCA Modular Software Download*** document.