




# BLACKDUCK

Hub Plugins for Maven  
2.0.0, Gradle 4.0.0, SBT  
1.0.0

Version 1.0.0



This edition of the *Hub Plugins for Maven 2.0.0, Gradle 4.0.0, SBT 1.0.0* refers to version 1.0.0 of the Black Duck Hub Plugin for Maven, Gradle, and SBT.

This document created or updated on Tuesday, March 28, 2017.

**Please send your comments and suggestions to:**

Black Duck Software, Incorporated  
800 District Avenue  
Suite 221  
Burlington, MA 01803 USA.

Copyright © 2017 by **Black Duck Software, Inc.**

All rights reserved. All use of this documentation is subject to the license agreement between Black Duck Software, Inc. and the licensee. No part of the contents of this document may be reproduced or transmitted in any form or by any means without the prior written permission of Black Duck Software, Inc.

Black Duck, Know Your Code, and the Black Duck logo are registered trademarks of Black Duck Software, Inc. in the United States and other jurisdictions. Black Duck Code Center, Black Duck Code Sight, Black Duck Export, Black Duck Hub, Black Duck Protex , and Black Duck Suite are trademarks of Black Duck Software, Inc. All other trademarks or registered trademarks are the sole property of their respective owners.

<b>Chapter 1: Introduction to Hub Plugins for Maven, Gradle, and SBT</b>	<b>5</b>
<b>Chapter 2: Hub Maven Plugin</b>	<b>6</b>
2.1 Hub Maven Plugin Overview	6
2.1.1 Hub Maven Plugin Requirements	6
2.2 Installation Overview	7
2.2.1 Installation Prerequisites	7
2.2.2 Downloading and Installing the Hub Maven Plugin	7
2.3 Configuring the Hub Maven Plugin	7
2.3.1 Configuration Prerequisites for the Hub Maven Plugin	8
2.4 Configuring the Hub Maven POM File	8
2.4.1 Configuring the Hub Maven Plugin in the POM File	8
2.4.2 Specifying the Hub Maven Configuration	8
2.4.3 Executing the Maven Goal	10
2.5 Hub Maven Plugin Release Notes	11
<b>Chapter 3: Hub Gradle Plugin</b>	<b>12</b>
3.1 Hub Gradle Plugin Overview	12
3.1.1 Hub Gradle Plugin Requirements	12
3.2 Installation Overview	13
3.2.1 Installation Prerequisites	13
3.2.2 Downloading and Installing the Hub Gradle Plugin	13
3.3 Hub Gradle Quick Start Guide	14
3.3.1 Configuring the Gradle Build Script	14
3.3.2 Configuring the Hub Gradle Plugin Task	14
3.3.3 Running the Hub Gradle Plugin Task	15
3.4 build.gradle with Buildsript Sections	15
3.4.1 Ensuring the Correct Repositories are Defined	15
3.4.2 Ensuring the Hub Gradle Plugin is Listed as a Dependency	15
3.4.3 Applying and Configuring the Hub Gradle Plugin	15
3.4.4 Gradle Task Properties	16
3.4.5 Running the Hub Gradle Plugin	17
3.5 Hub Gradle Plugin Release Notes	17
<b>Chapter 4: Hub SBT Plugin</b>	<b>19</b>
4.1 Hub SBT Plugin Overview	19

---

4.1.1 SBT Plugin Requirements .....	19
4.2 Hub SBT Plugin Installation Overview .....	20
4.2.1 Installing the SBT Plugin .....	20
4.3 Configuring the SBT Plugin .....	20
4.3.1 SBT Task Properties .....	20
4.4 Using the Hub SBT Plugin .....	21
4.4.1 Hub SBT Plugin Operation .....	21
4.5 SBT Plugin Release Notes .....	22
<b>Chapter 5: Troubleshooting Tips .....</b>	<b>23</b>
5.1 Development and Testing Errors .....	23
5.2 Error: Project Does Not Exist on the Hub .....	23
<b>Chapter 6: Black Duck Support .....</b>	<b>25</b>
6.1 Training .....	25
6.2 Services .....	26

# Chapter 1: Introduction to Hub Plugins for Maven, Gradle, and SBT

This guide combines the documentation for Black Duck Hub Maven, Gradle, and SBT plugins. This documentation assumes that you have a working knowledge of Maven, Gradle, or SBT.

Each plugin has its own section.

You can find each section here:

- [Hub Maven Plugin on page 6](#)
- [Hub Gradle Plugin on page 12](#)
- [Hub SBT Plugin on page 19](#)

## 2.1 Hub Maven Plugin Overview

Black Duck Hub is a new risk management tool designed to help you manage the logistics of using open source software in your organization.

The Hub Maven plugin provides the ability to automatically build, deploy, and scan a project in Black Duck Hub. The plugin accomplishes this by generating a Black Duck I/O (BDIO) formatted file containing the dependency information gathered from the Maven project. The file is generated in the target folder of the project. The Hub Maven plugin also has the ability to upload the Black Duck I/O file up to the Hub to create a code location in the Hub. To generate the file and upload the contents to the Hub, the Maven POM file must have a section for this plugin, and execute goals specific to this plugin. A developer or release engineer must update the projects `pom.xml` files to utilize the Hub Maven plugin.

You can run the Hub Maven plugin using either of the two following ways:

- Enter the full command on the CLI.
- Enter the full command in the plugin section of the POM file, using `group.artifact.ID`.

**Note:** You must be familiar with Maven, POM files, Maven settings, and Maven profiles to use this plugin.

**Tip:** You can run Maven by entering the GAV on the command line. Append this to your Maven invocation line. For more information, refer to the topic *Executing the Maven Goal*.

For more information about Maven, refer to:

<https://maven.apache.org/guides/mini/guide-configuring-plugins.html>

### 2.1.1 Hub Maven Plugin Requirements

#### Software Requirements

The installation instructions in this document assume that you have the following installed and configured on your system:

- Black Duck Hub 2.4 or higher
- Maven Plugin SDK for development purposes; the latest LTS release is recommended
- Java SE 7

The Maven plugin is supported on the same operating systems and browsers as Black Duck Hub.

## Network Requirements

The Hub Maven plugin requires internet connectivity. The machine that hosts your Maven server must be able to connect to the Hub server.

## 2.2 Installation Overview

### 2.2.1 Installation Prerequisites

Before you install the Hub Maven plugin, ensure that:

- Your Maven instance is up-to-date and fully patched.
- You know the host name and port for the Hub server.
- You have a user account with administrator privileges on the Hub system that you can use for the integration.
- You have connectivity to the internet. The machine that hosts your Maven server must be able to connect to the Hub server.

### 2.2.2 Downloading and Installing the Hub Maven Plugin

The Hub Maven plugin has no manual download step. The plugin automatically downloads and installs when you have altered your `pom.xml` file, which is described as follows.

#### \* To install the Hub Maven plugin:

1. In your Maven Projects folder, locate the `pom.xml` file, and open that file in a text editor.
2. At the top of the `pom.xml` file, add the following:

```
<build>
  <plugins>
    ...
    <plugin>
      <groupId>com.blackducksoftware.integration</groupId>
      <artifactId>hub-maven-plugin</artifactId>
      <version>1.4.1-SNAPSHOT</version>
    </plugin>
  </plugins>
</build>
```

3. Save and close the file.

## 2.3 Configuring the Hub Maven Plugin

The `build-bom` command is for Hub Maven versions 2.0.0 and higher. It creates and deploys the BDIO by default, but its behavior can be user-controlled.

## 2.3.1 Configuration Prerequisites for the Hub Maven Plugin

To use this plugin, you must perform the following:

1. Update the `pom.xml` file of the project to utilize the Maven plugin for generating the output file.
2. Build your Maven project. Ensure the package phase of the build is executed.

## 2.4 Configuring the Hub Maven POM File

This section provides examples for invoking the Hub Maven plugin.

**Note:** In the following examples, `ver` = Your version of choice.

### 2.4.1 Configuring the Hub Maven Plugin in the POM File

To configure the Hub Maven plugin in the POM file, use the following process.

#### Repositories

Include the following repositories:

- JCenter (<http://jcenter.bintray.com>)
- Restlet (<http://maven.restlet.com>)

#### Plugin

In the `plugins` portion of the POM file, add the `hub-maven-plugin` configuration:

```
<build>
  <plugins>
    <plugin>
      <groupId>com.blackducksoftware.integration</groupId>
      <artifactId>hub-maven-plugin</artifactId>
      <version>(latest version of the plugin)</version>
      <inherited>>false</inherited>
    </plugin>
  </plugins>
</build>
```

### 2.4.2 Specifying the Hub Maven Configuration

In execution, you can use the `build-bom` command and specify configuration items the same as you can in properties.

**Note:** You can use properties or configuration, but configuration takes priority.

The following are the properties and values for Maven configuration.



**Note:** The property `hub.scan.timeout` and the configuration `<hubScanTimeout>` both default to 300. The scan timeout is the amount of time in seconds to wait for the Hub to process the BDIO file, and how long to wait for a report to be generated.

```
<plugin>
  <groupId>com.blackducksoftware.integration</groupId>
  <artifactId>hub-maven-plugin</artifactId>
  <version>2.0.0</version>
  <inherited>false</inherited>
  <executions>
    <configuration>
      <hubIgnoreFailure>true</hubIgnoreFailure>
      <hubProjectName>OverrideMavenProjectName</hubProjectName>
      <hubVersionName>OverrideMavenProjectVersion</hubVersionName>
      <hubUrl>http://hubUrl:8080</hubUrl>
      <hubUsername>admin</hubUsername>
      <hubPassword>password</hubPassword>
      <hubTimeout>120</hubTimeout>
      <hubProxyHost>ProxyHost</hubProxyHost>
      <hubProxyPort>3128</hubProxyPort>
      <hubNoProxyHosts>.*google.*</hubNoProxyHosts>
      <hubProxyUsername>proxyUser</hubProxyUsername>
      <hubProxyPassword>proxypassword</hubProxyPassword>
      <createFlatDependencyList>false</createFlatDependencyList>
      <createHubBdio>true</createHubBdio>
      <deployHubBdio>false</deployHubBdio>
      <checkPolicies>false</checkPolicies>
      <createHubReport>false</createHubReport>
      <outputDirectory>./blackduck/output</outputDirectory>
      <hubScanTimeout>300</hubScanTimeout>
      <includedScopes>compile, runtime</includedScopes>
      <excludedModules>subModule</excludedModules>
      <hubCodeLocationName>OverrideCodeLocationName</hubCodeLocat
ionName>
    </configuration>
  </executions>
  <properties>
```

```

        <hub.output.directory>/directory/where/output/is/stored</hub.output.
directory>
        <hub.code.location.name>OverrideCodeLocationName</hub.code.location.
name>
        <hub.project.name>overrideDefaultMavenProjectName</hub.project.name>
        <hub.version.name>overrideDefaultMavenVersionName</hub.version.name>
        <hub.url>http://some.hub.server.com:8080</hub.url>
        <hub.username>username</hub.username>
        <hub.password>password</hub.password>
        <hub.timeout>120</hub.timeout>
        <hub.proxy.host>some.proxy.uri</hub.proxy.host>
        <hub.proxy.port>1234</hub.proxy.port>
        <hub.proxy.no.hosts></hub.proxy.no.hosts>
        <hub.proxy.username>proxyuser</hub.proxy.username>
        <hub.proxy.password>proxypass</hub.proxy.password>
        <hub.scan.timeout>300</hub.scan.timeout>
        <hub.ignore.failure>false</hub.ignore.failure>
        <hub.create.flat.list>false</hub.create.flat.list>
        <hub.create.bdio>true</hub.create.bdio>
        <hub.deploy.bdio>true</hub.deploy.bdio>
        <hub.check.policies>false</hub.check.policies>
        <included.scopes>compile</included.scopes>
        <excluded.modules>subModule</excluded.modules>
    </properties>
    </execution>
</executions>
</plugin>

```

### 2.4.3 Executing the Maven Goal

Using the Hub Maven plugin, you can execute the goal without attaching to a Maven lifecycle phase by using the following command:

```
mvn com.blackducksoftware.integration:hub-maven-plugin:ver:build-bom
```

This command uses the properties and the configuration from the `.pom` file and active profiles. This automatically places your project into Black Duck Hub's **Code Locations** page, and automatically begins a scan of your project.

## 2.5 Hub Maven Plugin Release Notes

### Changes in Release 2.0.0

- Added new properties for Hub information.
- Introduced the `build-bom` command to simplify configuration and execution.
- Addressed a mapping error occurring in Hub versions 3.5 and higher.

### Changes in Release 1.0.6

- Added backwards compatibility for old properties.

### Changes in Release 1.0.5

- Added a flag to continue the build in the event of Black Duck-related failures.
- New `deployHubOutputAndCheckPolicies` is available, and fails the build if policy violations exist in the Hub.
- Project name/version can be passed in through the command line.
- Changes to all Black Duck setting names to be consistent with the Hub Gradle Plugin.

## 3.1 Hub Gradle Plugin Overview

Black Duck Hub is a risk management tool designed to help you manage the logistics of using open source software in your organization.

The Hub Gradle plugin provides the ability to generate a Black Duck I/O formatted file containing the dependency information gathered from the Gradle project. The file is generated in either the specified folder, or defaults to the root of the project. This plugin also has the ability to upload the Black Duck I/O file up to the Hub to create a code location in the Hub. To generate the file and upload the contents to the Hub, the `build.gradle` file must have a section for this plugin, and execute tasks specific to this plugin.

Gradle is a continuous integration tool that monitors executions of repeated jobs, such as building a software project or cron jobs. Gradle focuses on building and testing software projects continuously and monitoring execution of externally run jobs.

As a Hub and Gradle user, the Hub Gradle plugin enables you to configure your environment to connect to the Hub server.

**Note:** This documentation assumes that you have familiarity with Gradle and how to use it.

For more information about Gradle, refer to:

<https://docs.gradle.org/current/userguide/plugins.html>

### 3.1.1 Hub Gradle Plugin Requirements

#### Software Requirements

The installation instructions in this document assume that you have the following installed and configured on your system:

- Black Duck Hub 2.4 or higher
- Gradle Plugin SDK for development purposes; the latest LTS version is recommended
- Java SE 7

The Gradle plugin is supported on the same operating systems and browsers as Black Duck Hub.

#### Network Requirements

The Hub Gradle plugin requires internet connectivity. The machine that hosts your Gradle server must be able to connect to the Hub server.

## 3.2 Installation Overview

### 3.2.1 Installation Prerequisites

Before you install the Hub Gradle plugin, ensure that:

- Your Gradle instance is up-to-date and fully patched.
- You know the host name and port for the Hub server.
- You have a user account with administrator privileges on the Hub system that you can use for the integration.
- You have connectivity to the internet. The machine that hosts your Gradle server must be able to connect to the Hub server.

### 3.2.2 Downloading and Installing the Hub Gradle Plugin

The Hub Gradle plugin has no manual download step. The plugin automatically downloads and installs when you have altered your `build.gradle` file, which is described as follows.

#### \* To install the Hub Gradle plugin:

1. In your Gradle Projects folder, locate the `build.gradle` file, and open that file in a text editor.
2. At the top of the `build.gradle` file, add the following line:

```
buildscript {  
    repositories {  
        mavenCentral()  
        maven { url "http://jcenter.bintray.com" }  
    }  
  
    dependencies {  
        classpath group: 'com.blackducksoftware.integration', name:  
        'hub-gradle-plugin', version: 'x.x.x'  
    }  
}  
  
apply plugin: 'com.blackducksoftware.hub'  
  
buildBom;  
  
    hubUrl = 'http://localhost:8080'  
    hubUsername = 'username'  
    hubPassword = 'password'  
}
```

3. Save and close the file.

## 3.3 Hub Gradle Quick Start Guide

This section provides a simple configuration that can be used to get the Hub Gradle plugin up and running with the fewest number of steps. The steps include:

1. Configuring the Hub Gradle build script.
2. Configuring the Hub Gradle plugin task.
3. Run the Hub Gradle plugin task.

### 3.3.1 Configuring the Gradle Build Script

For Hub Gradle plugin versions 4.0 and higher, you can configure your Gradle build script using the following process.

#### \* To configure the Gradle build script:

1. Add the following to the top of your `build.gradle` file.

```
buildscript {
    repositories {
        mavenCentral()
        maven { url "http://jcenter.bintray.com" }
    }

    dependencies {
        classpath group: 'com.blackducksoftware.integration', name:
'hub-gradle-plugin', version: 'x.x.x'
    }
}

apply plugin: 'com.blackducksoftware.hub'

buildBom;

hubUrl = 'http://localhost:8080'
hubUsername = 'username'
hubPassword = 'password'
}
```

### 3.3.2 Configuring the Hub Gradle Plugin Task

For Hub Gradle versions 4.0 and higher, you can configure your Hub Gradle task using the following process.

### ✳ To configure the Hub Gradle task:

1. Update the plugin version number from x.x.x to the version to be used.
2. Change the properties in the `buildBom` task to match the Hub server and credentials to be used.

## 3.3.3 Running the Hub Gradle Plugin Task

For Hub Gradle plugin versions 4.0 and higher, you can run your Hub Gradle plugin task using the following command:

```
gradle buildBom
```

## 3.4 build.gradle with Buildscript Sections

If your `build.gradle` file contains a *buildscript* section, then the steps are similar to the previous section. This section describes the changes in this scenario. The revised steps include:

1. Ensure the correct repositories are defined.
2. Ensure the plugin is listed as a dependency.
3. Apply and configure the plugin.
4. Run the plugin.

### 3.4.1 Ensuring the Correct Repositories are Defined

In the *buildscript* section of your `build.gradle` file, verify that the *repositories* section is present and includes the following:

```
repositories {  
    mavenCentral()  
    maven { url "http://jcenter.bintray.com" }  
}
```

### 3.4.2 Ensuring the Hub Gradle Plugin is Listed as a Dependency

In the *buildscript* section of the `build.gradle` file, verify that the *dependencies* section is present and includes the following:

```
dependencies {  
    classpath group: 'com.blackducksoftware.integration', name: 'hub-gradle-  
plugin', version: 'x.x.x'  
}
```

### 3.4.3 Applying and Configuring the Hub Gradle Plugin

For Hub Gradle plugin versions 4.0 and higher, you can apply and configure your Hub Gradle plugin using the following process.

### ✱ To apply and configure your Hub Gradle plugin:

1. Add the following to your `build.gradle` file after the `buildscripts` section.

```
buildscript {  
    .  
    .  
    .  
}  
apply plugin: 'com.blackducksoftware.hub'  
buildBom {  
    hubUrl = 'http://localhost:8080'  
    hubUsername = 'username'  
    hubPassword = 'password'  
}
```

2. Update the plugin version number from x.x.x to the version to be used.
3. Change the properties in the `buildBom` task to match the Hub server and credentials to be used.

## 3.4.4 Gradle Task Properties

Add the following task properties to your `build.gradle` file.

```
hubIgnoreFailure = true  
hubCodeLocationName = OverrideCodeLocationName  
hubProjectName = OverrideMavenProjectName  
hubVersionName = OverrideMavenProjectVersion  
hubUrl = http://YourHubUrl:8080  
hubUsername = admin  
hubPassword = password  
hubTimeout = 120  
hubProxyHost = ProxyHost  
hubProxyPort = 3128  
hubNoProxyHosts = .*google.*  
hubProxyUsername = proxyUser  
hubProxyPassword = proxypassword  
createFlatDependencyList = false  
createHubBdio = true  
deployHubBdio = false
```



```
checkPolicies = false
createHubReport = false
outputDirectory = ./blackduck/output
hubScanTimeout = 300
includedScopes = compile, runtime
excludedModules = submodule
```

### 3.4.5 Running the Hub Gradle Plugin

You can now run the Hub Gradle plugin using the following command:

```
gradle buildBOM
```

## 3.5 Hub Gradle Plugin Release Notes

### Changes in Release 4.0.0

- Provides the ability to generate a BDIO file.
- Can now upload the BDIO file to the Hub server.
- The new `buildBom` command replaces the previous configuration commands.
- Addressed a mapping error occurring in Hub versions 3.5 and higher.

### Changes in Release 3.4.1

- Resolves *StringUtils class not found* from package `org/apache/commons/lang`.

### Changes in Release 3.4.0

- Now, `blackDuckHubProjectName.txt` and `blackDuckProjectVersionName.txt` files are created with the output.

### Changes in Release 3.3.0

- Excludes certain modules from dependency gathering. Use `excludedModules` as a comma-separated list of module names to exclude from the output.
- Catch `Exception` instead of `GradleException` for all failures.
- Now, the `deployHubOutput` property `hubTimeout` defaults to a reasonable value (120 seconds).

### Changes in Release 3.2.0

- Resolved an issue which allows the `hubProjectName` configuration property to contain spaces and `/` characters.
- Support for the `includedConfiguration` task property to define which configurations should be used to retrieve the list of dependencies.

**Changes in Release 3.1.0**

- When `deployHubOutput` fails, build still returns as successful.
- Ability to add a Gradle task to expose the Risk report.

**Changes in Release 2.0.7**

- Added support for subprojects and removed duplicate components.

## 4.1 Hub SBT Plugin Overview

Black Duck® Hub™ provides a comprehensive environment in which to search, request, discuss, approve, and manage internally- and externally-developed components used in software development. Hub is designed to help software developers quickly and efficiently re-use software components in their applications.

The Hub SBT plugin provides the ability to generate a Black Duck I/O formatted file containing the dependency information gathered from the SBT project. The file is generated in either the specified folder, or defaults to the root of the project. This plugin also has the ability to upload the Black Duck I/O file up to the Hub to create a code location in the Hub. To generate the file and upload the contents to the Hub, the `build.sbt` file must have a section for this plugin, and execute tasks specific to this plugin.

The Hub SBT plugin runs automatically with your builds, automatically pushing your build into the Hub and generating a Bill of Materials (BOM).

You must have a Hub user name to configure and use the Hub SBT plugin. Your user name must have either the application developer role or the application manager role. If your system administrator has configured custom roles, you must have the **Can add a component to an application** permission. For more information about user roles, refer to the Hub help or the *Configuring and Managing Hub* guide.

For detailed information about searching for components and making component requests, refer to the Hub Help or the *Requesting Component Use in Hub* guide.

For more information about SBT, refer to:

<http://www.scala-sbt.org/0.13/docs/Using-Plugins.html>

### 4.1.1 SBT Plugin Requirements

#### Software Requirements

Before installing the Hub SBT plugin, your system must meet the following prerequisites:

- Hub 6.7 or higher installed.
- Java JDK 6 or higher.

For Hub installation instructions and system requirements, refer to the *Hub Installation Guide*.

For additional information regarding browser versions supported by SBT, refer to:

<http://www.SBT.org/swt/faq.php#browserlinux>

and for additional SBT information , refer to:

<http://help.SBT.org/indigo/index.jsp?topic=%2Forg.SBT.platform.doc.user%2Freference%2Fref-42.htm>

## Network Requirements

The Hub SBT plugin requires internet connectivity. The machine hosting your SBT environment must be able to connect to the Hub server.

## 4.2 Hub SBT Plugin Installation Overview

The general process for installing the Hub SBT plugin is as follows:

1. Install the SBT plugin.
2. Configure the SBT plugin.

The following sections include detailed instructions.

### 4.2.1 Installing the SBT Plugin

**Note:** You must restart SBT after installing the SBT plugin.

#### \* To install the Hub SBT plugin:

1. In your SBT Projects folder, locate the `plugins.sbt` file, and open that file in a text editor.
2. At the top of the `plugins.sbt` file, add the following line:

```
addSbtPlugin("com.blackducksoftware.integration" % "sbt-hub-plugin" %  
"1.0.0-SNAPSHOT")
```

3. Save and close the file.

## 4.3 Configuring the SBT Plugin

**Note:** You must have a Hub user name and password to configure and use the Hub SBT plugin.

#### \* To configure the SBT plugin:

1. In your SBT Projects folder, locate the `plugins.sbt` file, and open that file in a text editor.
2. Type your server settings, user name, and password into the `plugins.sbt` file:

```
a. hubUrl := "http://your_server.com:9000"  
b.  hubUsername := "sysadmin"  
c.  hubPassword := "blackduck"
```

3. Save and close the file.
4. Restart your instance of SBT.

### 4.3.1 SBT Task Properties

Add the following task settings to your `plugins.sbt` file.

```
hubIgnoreFailure := true
```

```
hubCodeLocationName := OverrideCodeLocationName
hubProjectName := OverrideMavenProjectName
hubVersionName := OverrideMavenProjectVersion
hubUrl := http://hubUrl:8080
hubUsername := admin
hubPassword := password
hubTimeout := 120
hubProxyHost := ProxyHost
hubProxyPort := 3128
hubNoProxyHosts := .*google.*
hubProxyUsername := proxyUser
hubProxyPassword := proxypassword
createFlatDependencyList := false
createHubBdio := true
deployHubBdio := false
checkPolicies := false
createHubReport := false
outputDirectory := ./blackduck/output
hubScanTimeout := 300
includedScopes := compile, runtime
excludedModules := subModule
```

## 4.4 Using the Hub SBT Plugin

The following sections outline usage of the Hub SBT plugin.

For more information about searching for components and making component requests, refer to the Hub Help or the *Requesting Component Use in Hub* guide.

### 4.4.1 Hub SBT Plugin Operation

Using the Hub SBT plugin only requires a single command. The plugin is automatically triggered as part of your build.

#### ✳ To use the Hub SBT plugin:

1. Issue the command `buildBOM`. This automatically triggers your build, pushes that build into the Hub, and generates a Bill of Materials (BOM).

## 4.5 SBT Plugin Release Notes

### Hub SBT Plugin Release 1.0.0

- Addressed a mapping error occurring in Hub versions 3.5 and higher.

Refer to the following sections should issues arise during use of your Hub plugin instance.

**Tip:** After major releases of Hub, check for updated versions of your Black Duck plugins and their installation prerequisites. Changes to the APIs, schema, and SDK versions may require updated versions of the integration plugins.

## 5.1 Development and Testing Errors

If an error message is generated that states *During development and testing the following errors were encountered*, use the following solutions:

- If you try to use Java 6 instead of Java 7, instead of getting an *Unsupported major:minor version error* message, the plugin sometimes throws a false *java.lang.OutOfMemoryError: Java heap space* message instead.
- If you get a message that reads *Service Unavailable*, either the Hub server can't be reached, or the request to the server is invalid. Contact your Hub server administrator.
- If you get a *Precondition failed* error message, then the request to the server is invalid. Verify that your global configuration is correct, and verify that the job configuration is correct. If you are still getting this message after you have checked your configuration, contact your Black Duck technical account manager.
- If you get a *Not Found (404) - Not Found* error message, then the request to the server is invalid. Contact your Black Duck technical account manager.

## 5.2 Error: Project Does Not Exist on the Hub

- If you try to use the Hub plugin integration, and you configure a job with a project and version, and that project already exists, but the current Hub user is not assigned to it, then the following errors display:
  - In the job configuration **Project Name** field, a notification displays *This project does not exist on the Hub Server*. Clicking **Create project/version** displays a message reading *This version may already exist.com.blackducksoftware.integration.hub.exception.BDRestException: There was a problem creating this Hub project. Error Code: 412.*
  - If you run the build, the following displays:

*Status : 412*

```
Response : {"errorMessage":"project name already exists","arguments":
{"fieldName":"name"},"errors":[{"errorMessage":"project name already
exists","arguments":{"fieldName":"name"},"errorCode":"{central.constraint_
violation.project_name_duplicate_not_allowed}"},"errorCode":}
```

```
{central.constraint_violation.project_name_duplicate_not_allowed}"}
```

*Problem creating the project.*

**Solution:**

Assigning the current user to the existing project with this name resolves the issue.



# Chapter 6: Black Duck Support

If you have questions or find issues, contact Black Duck Software.

For the latest in web-based support, access the Black Duck Software Customer Support Web Site:  
<https://www.blackducksoftware.com/support/contact-support>

To access a range of informational resources, services and support, as well as access to Black Duck experts, visit the Black Duck Customer Success portal at:  
<https://www2.blackducksoftware.com/support/customer-success>

You can also contact Black Duck Support in the following ways:

- **Email:** [support@blackducksoftware.com](mailto:support@blackducksoftware.com)
- **Phone:** +1 781.891.5100, ext. 5
- **Fax:** +1 781.891.5145
- **Standard working hours:** Monday through Friday 8:00 AM to 8:00 PM EST

**Note:** Customers on the **Enhanced Customer Support Plan** are able to contact customer support 24 hours a day, 7 days a week to obtain Tier 1 support.

If you are reporting an issue, please include the following information to help us investigate your issue:

- Name and version of the plugin.
- Black Duck product name and version number.
- Third-party integrated product and version; for example, Artifactory, Eclipse, Jenkins, Maven, and others. For Black Duck Hub, only Jenkins, TeamCity, and Bamboo is supported.
- Java version.
- Black Duck KnowledgeBase version, where applicable.
- Operating system and version.
- Source control management system and version.
- If possible, the log files, configuration files, and Project Object Model (POM) XML files.

## 6.1 Training

Black Duck training courses are available for purchase. Learn more at  
<https://www.blackducksoftware.com/services/training>.

View the full catalog of our online offerings: <https://www.blackducksoftware.com/academy-catalog>.

When you are ready to learn, you can log in or sign up for an account:  
<https://www.blackducksoftware.com/academy>.

## 6.2 Services

If you would like someone to perform Black Duck Software tasks for you, please contact the Black Duck Services group. They offer a full range of services, from planning, to implementation, to analysis. They also offer a variety of training options on all Black Duck products. Refer to <https://www.blackducksoftware.com/services/> for more information.