

JNI Development on Spring

Innova Lee(이상훈)
gcccompil3r@gmail.com





The image shows a code editor window with several tabs: 'hello.c', 'app-context.xml', 'application.yaml', 'DemoApplication.java', and 'build.gradle'. The 'build.gradle' tab is active, displaying the following Gradle configuration:

```
1 plugins {  
2     id 'org.springframework.boot' version '2.2.1.RELEASE'  
3     id 'io.spring.dependency-management' version '1.0.8.RELEASE'  
4     id 'java'  
5 }  
6  
7 apply plugin: 'application'  
8 apply plugin: 'c'  
9  
10 group = 'com.example'  
11 version = '0.0.1-SNAPSHOT'  
12 sourceCompatibility = '1.8'  
13  
14 mainClassName = 'DemoApplication'  
15  
16 configurations {  
17     compileOnly {  
18         extendsFrom annotationProcessor  
19     }  
20 }  
21  
22 repositories {  
23     mavenCentral()  
24 }  
25
```

```

25
26 ▶ dependencies {
27     implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
28     implementation 'org.springframework.boot:spring-boot-starter-data-redis'
29     implementation 'org.springframework.boot:spring-boot-starter-web'
30     compileOnly 'org.projectlombok:lombok'
31     runtimeOnly 'com.h2database:h2'
32     runtimeOnly 'org.postgresql:postgresql'
33     annotationProcessor 'org.projectlombok:lombok'
34     testImplementation('org.springframework.boot:spring-boot-starter-test') {
35         exclude group: 'org.junit.vintage', module: 'junit-vintage-engine'
36     }
37 }
38
39 ▶ test {
40     useJUnitPlatform()
41 }
42
43 ▶ test {
44     systemProperty "java.library.path", file("${buildDir}/libs/hello/shared").absolutePath
45 }
46

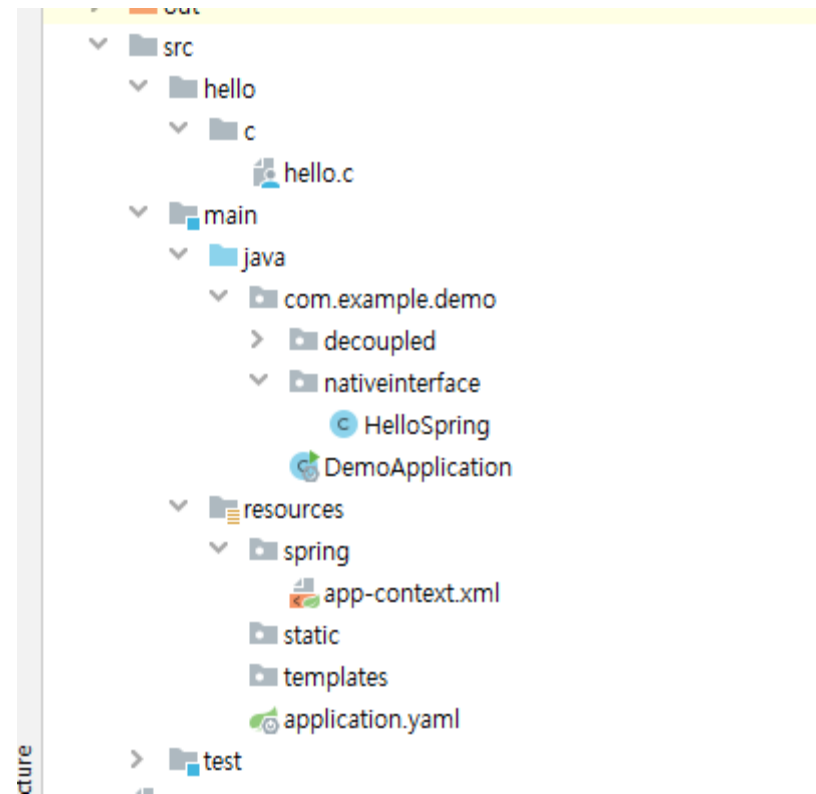
```

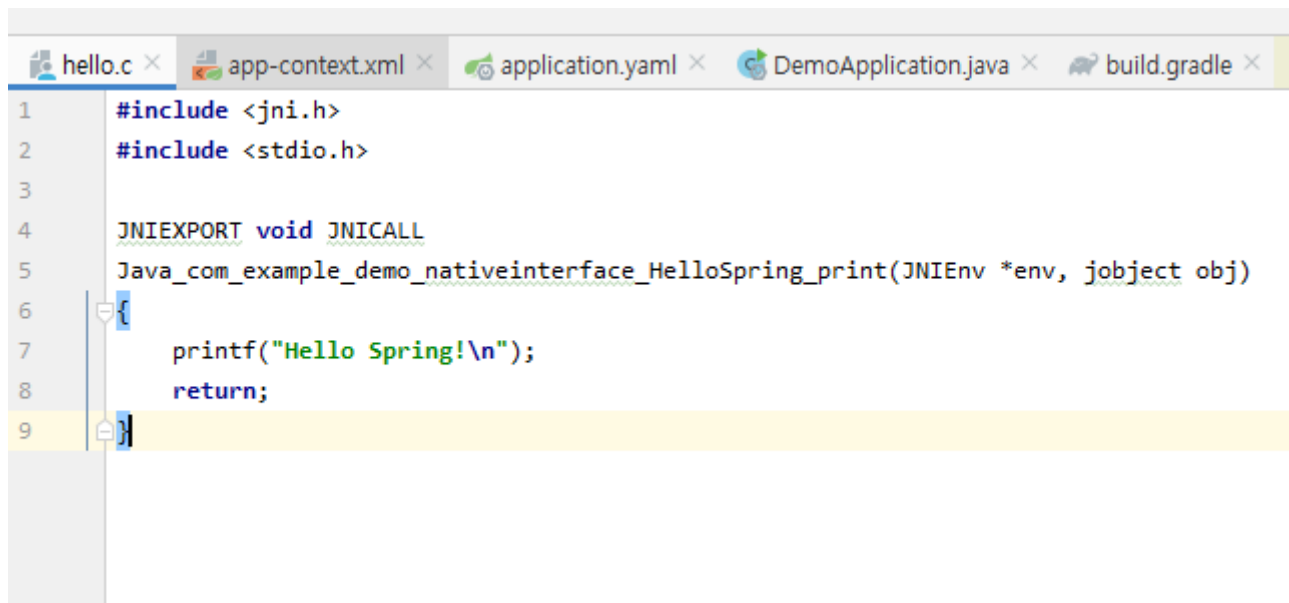
```
46 |
47 ► model {
48 |     platforms {
49 |         x64 {
50 |             architecture "x86_64"
51 |         }
52 |     }
53 |     toolChains {
54 |         gcc(Gcc) {
55 |             target("x86_64") {
56 |                 cCompiler.executable = "D:/MinGW_x86_64/bin/gcc"
57 |                 linker.executable = "D:/MinGW_x86_64/bin/ld"
58 |             }
59 |         }
60 |     }
```

```

61 components {
62     hello(NativeLibrarySpec) {
63         sources {
64             c {
65                 source {
66                     srcDir 'src/hello/c'
67                     include "**/*.c"
68                 }
69             }
70         }
71         buildTypes {
72             debug
73             release
74         }
75         binaries.all {
76             cCompiler.args "-c"
77             cCompiler.args "-m64"
78             cCompiler.args "-I${org.gradle.internal.jvm.Jvm.current().javaHome}/include"
79             cCompiler.args "-I${org.gradle.internal.jvm.Jvm.current().javaHome}/include/win32"
80             cCompiler.args "-LD:\\MinGW_x86_64\\mingw64\\lib"
81             linker.args "-m64"
82             linker.args "-LD:\\MinGW_x86_64\\mingw64\\lib"
83             linker.args "-shared"
84         }
85     }
86 }
87
88
89 test.dependsOn 'helloReleaseSharedLibrary'

```





The image shows a code editor window with several tabs: 'hello.c', 'app-context.xml', 'application.yaml', 'DemoApplication.java', and 'build.gradle'. The 'hello.c' tab is active, displaying the following C code:

```
1  #include <jni.h>
2  #include <stdio.h>
3
4  JNIEXPORT void JNICALL
5  Java_com_example_demo_nativeinterface_HelloSpring_print(JNIEnv *env, jobject obj)
6  {
7      printf("Hello Spring!\n");
8      return;
9  }
```

The code is a JNI implementation of a Java method. It includes the necessary headers, defines a JNIEXPORT function, and uses printf to output "Hello Spring!". The function signature matches the Java method in the 'DemoApplication.java' tab.


```
hello.c x app-context.xml x application.yaml x HelloSpring.java x DemoApplication.java x build.gradle x 192.7142.3
1 package com.example.demo.nativeinterface;
2
3 import org.slf4j.Logger;
4 import org.slf4j.LoggerFactory;
5
6 public class HelloSpring {
7     private final static Logger logger = LoggerFactory.getLogger(HelloSpring.class);
8
9     public native void print();
10
11     static {
12         logger.info("*****");
13         logger.info("*** Native library initialization sequence beginning. ");
14         logger.info("*** java.library.path: " + System.getProperty("java.library.path"));
15
16         try {
17             System.loadLibrary("hello");
18         } catch (Exception e) {
19             logger.error("*****");
20             logger.error("*** Failed to load JNI bridge library: ", e);
21             logger.error("*****");
22         }
23     }
24 }
```



```
package com.example.demo;

import ...

@SpringBootApplication
public class DemoApplication {
    public static void main(String[] args) {

        HelloSpring hs = new HelloSpring();
        hs.print();

        SpringApplication.run(DemoApplication.class, args);
    }
}
```

sourceforge.net/projects/mingw-w64/files/Toolchains%20targetting%20Win32/Personal%20Builds/mingw-builds/inst



SOURCEFORGE

Open Source Software

SOFTWARE FOR DIGITAL MARKET

JOIN NOW FOR FREE!

[Home](#) / [Browse](#) / [Development](#) / [Compilers](#) / MinGW-w64 - for 32 and 64 bit Windows



MinGW-w64 - for 32 and 64 bit Windows

A complete runtime environment for gcc

Brought to you by: [jon_y](#), [ktietz70](#), [nightstrike](#)

Your download will start shortly...

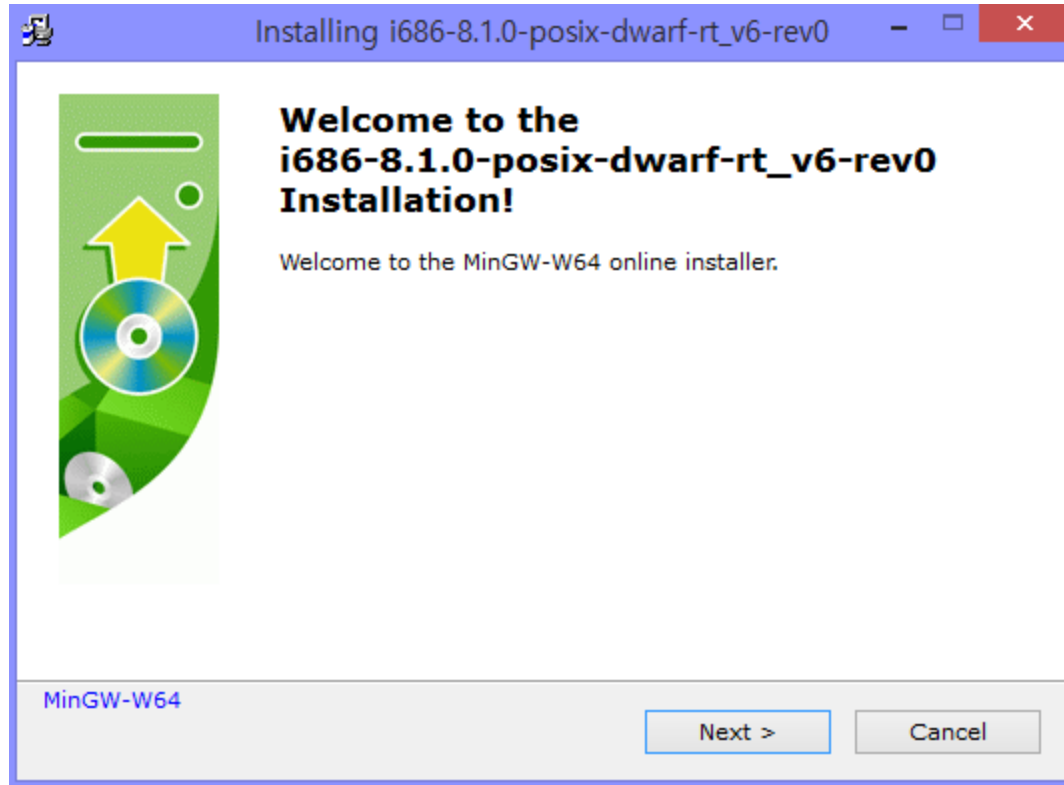
Get Updates

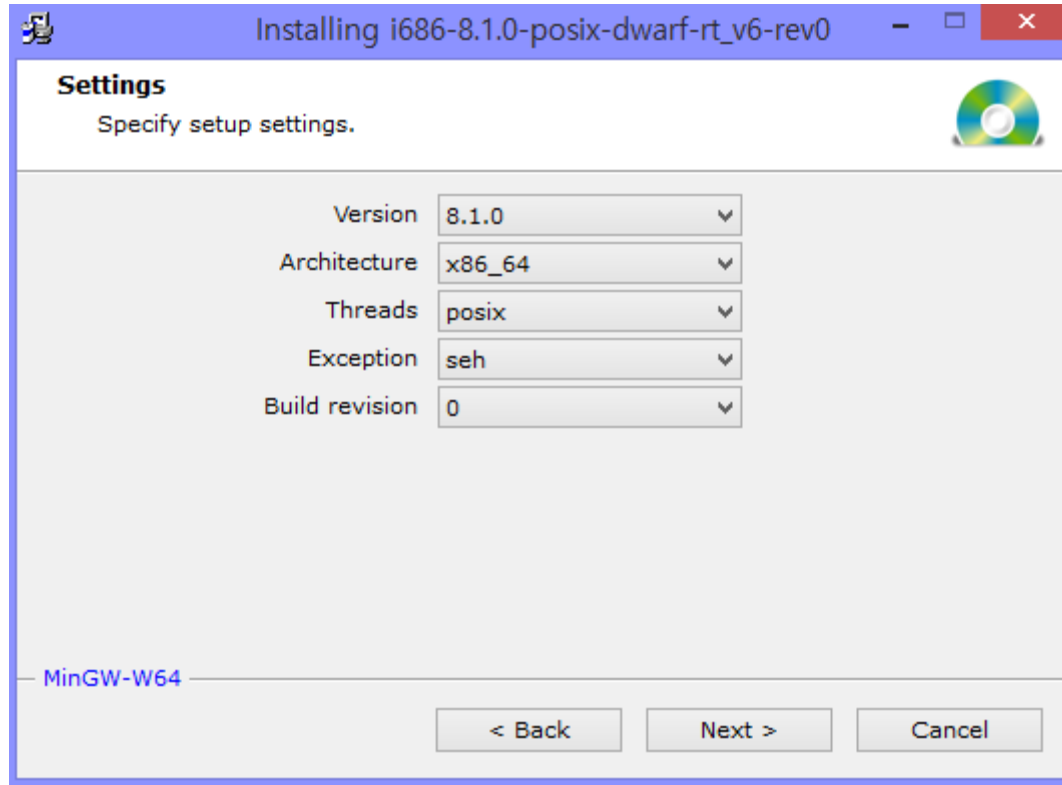
Share This

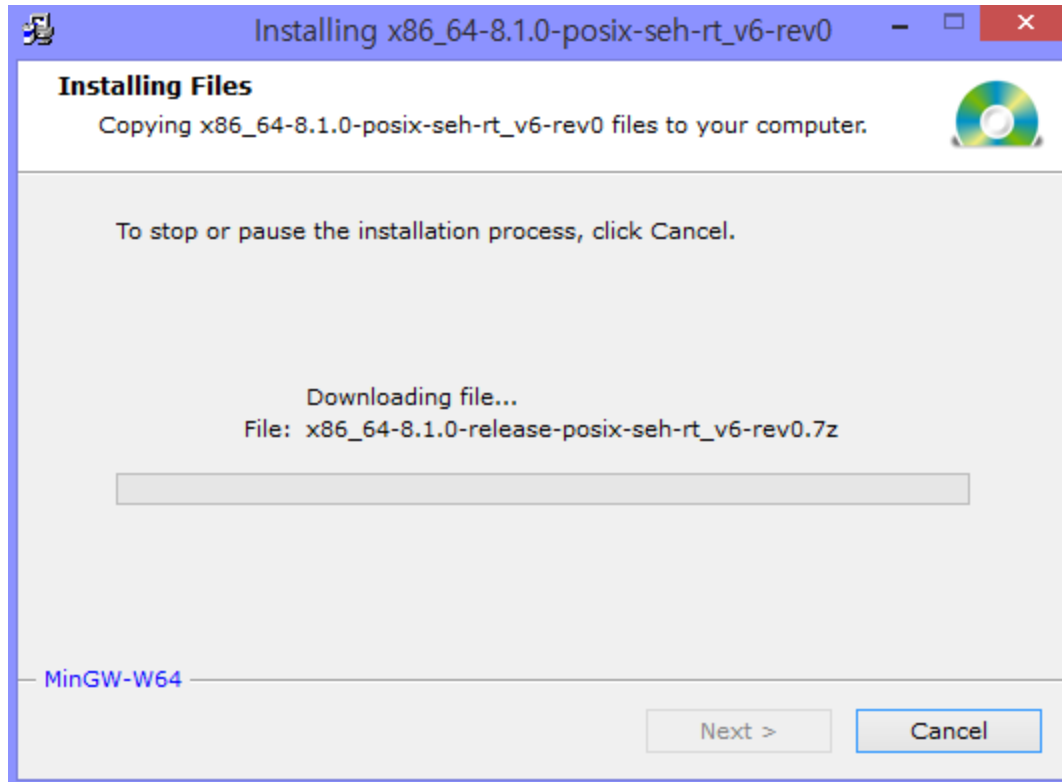
Problems Downloading?

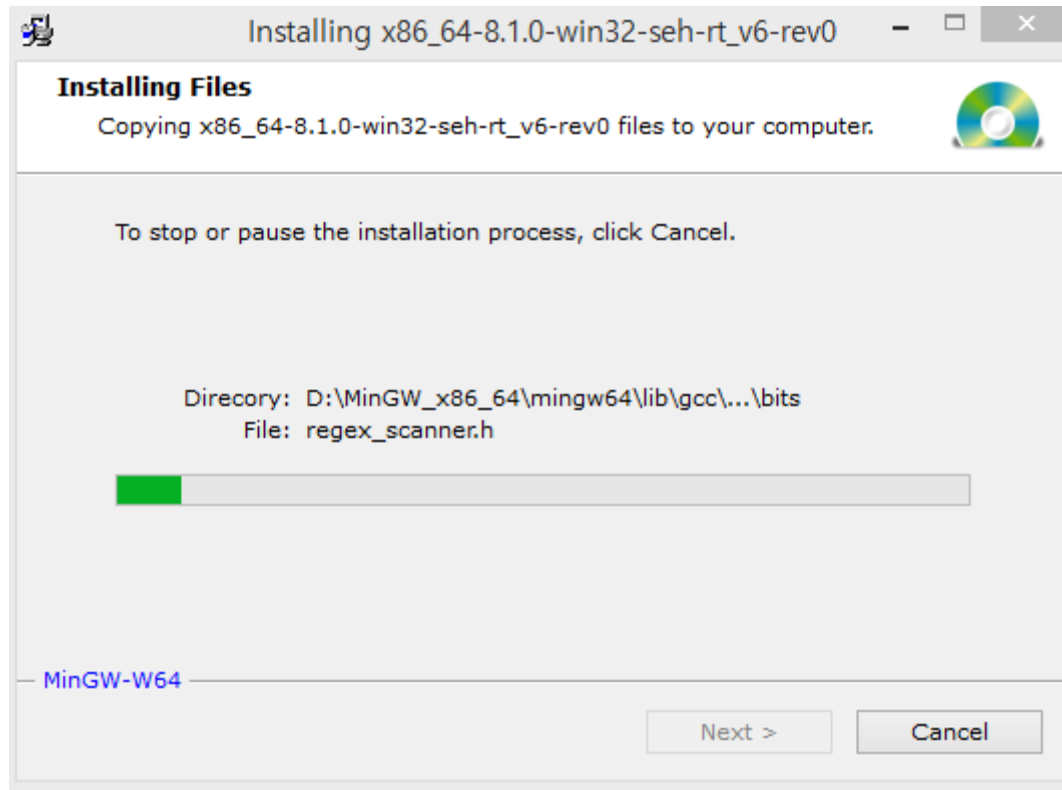
mingw-w64-install.exe | Scanned by: **Bitdefender**

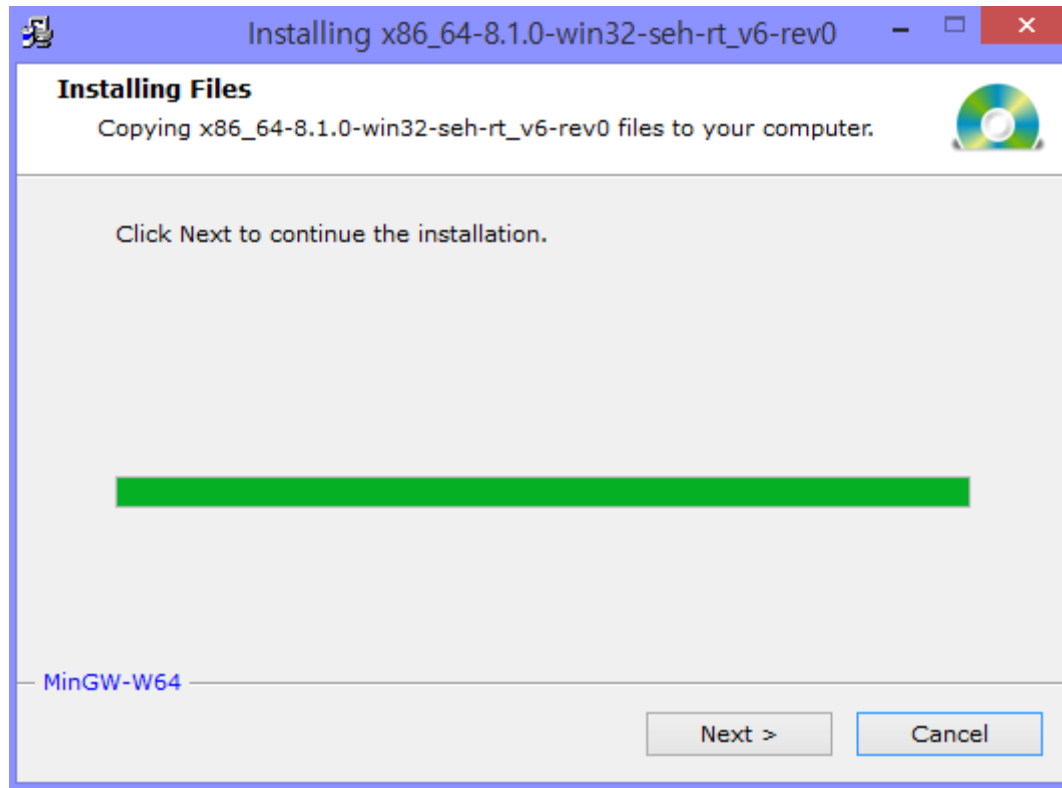
Other Useful Business Software

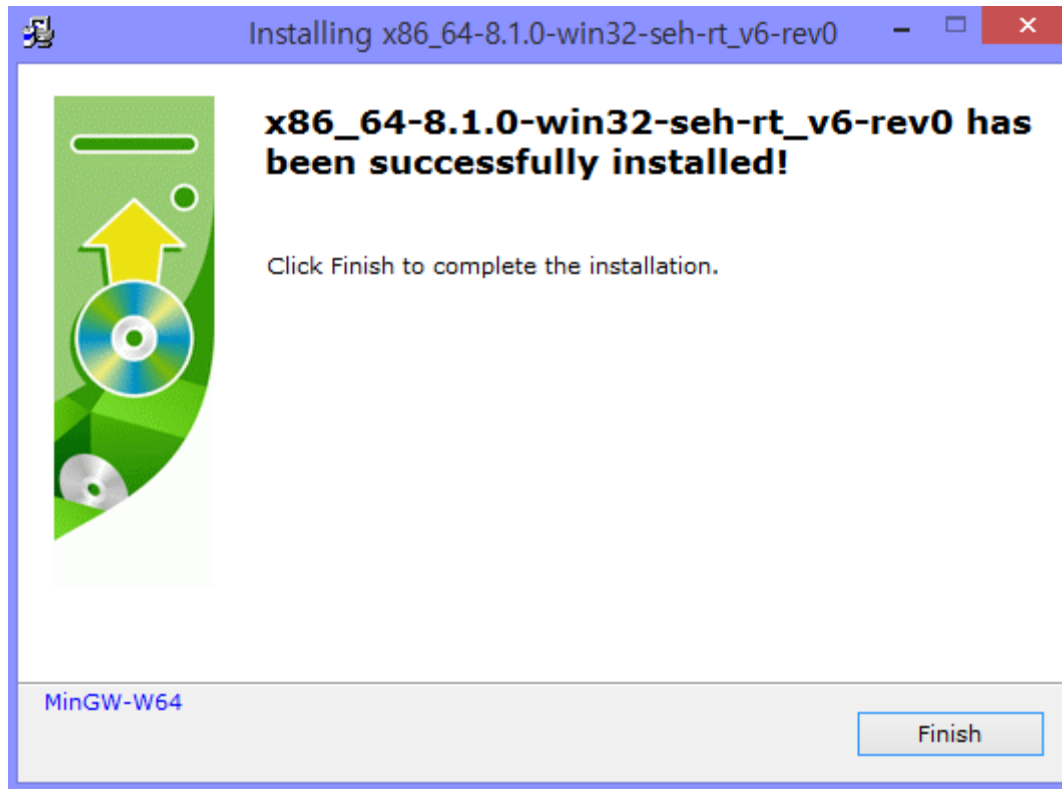


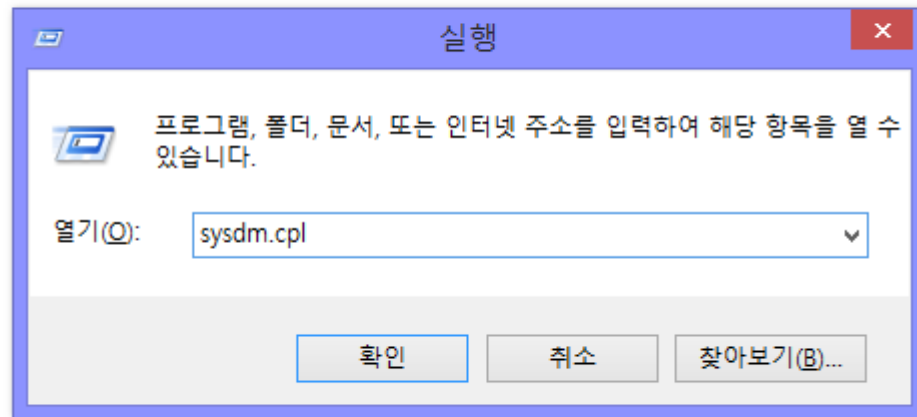


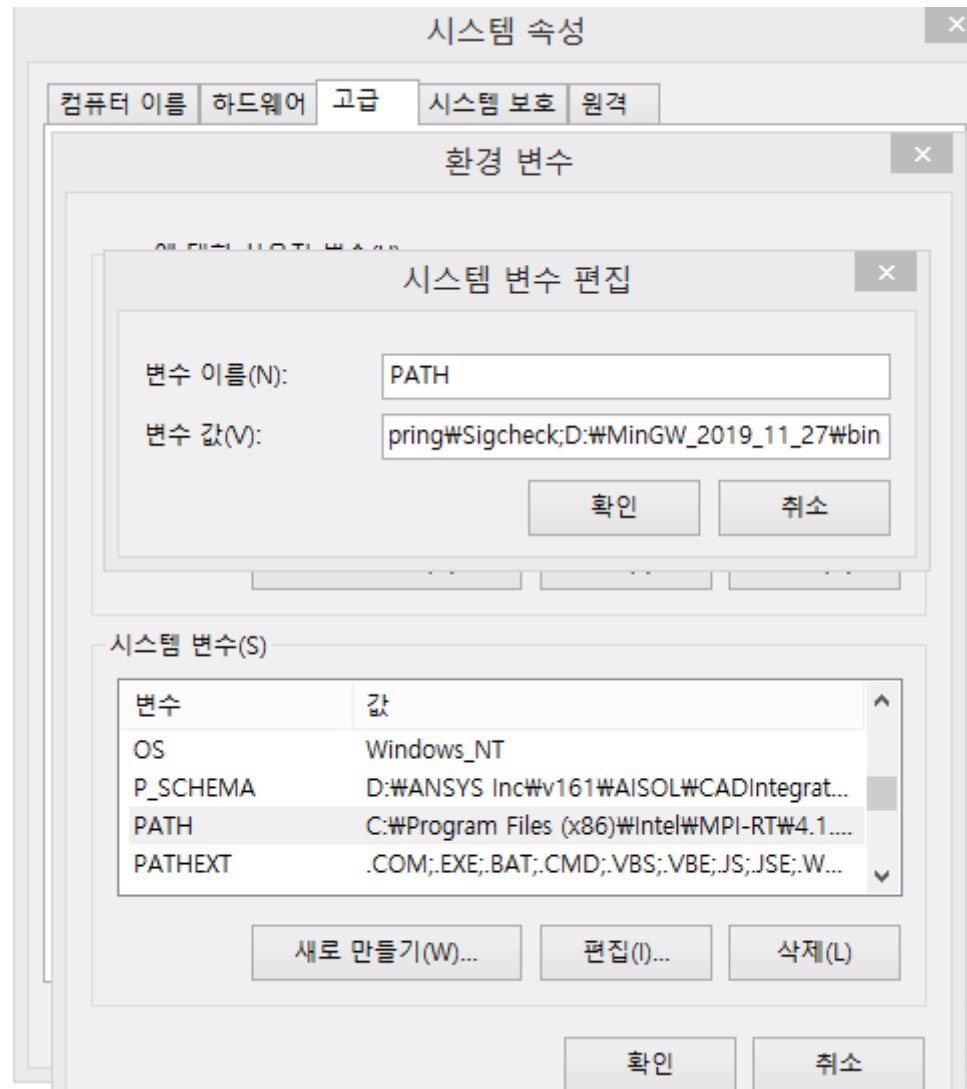


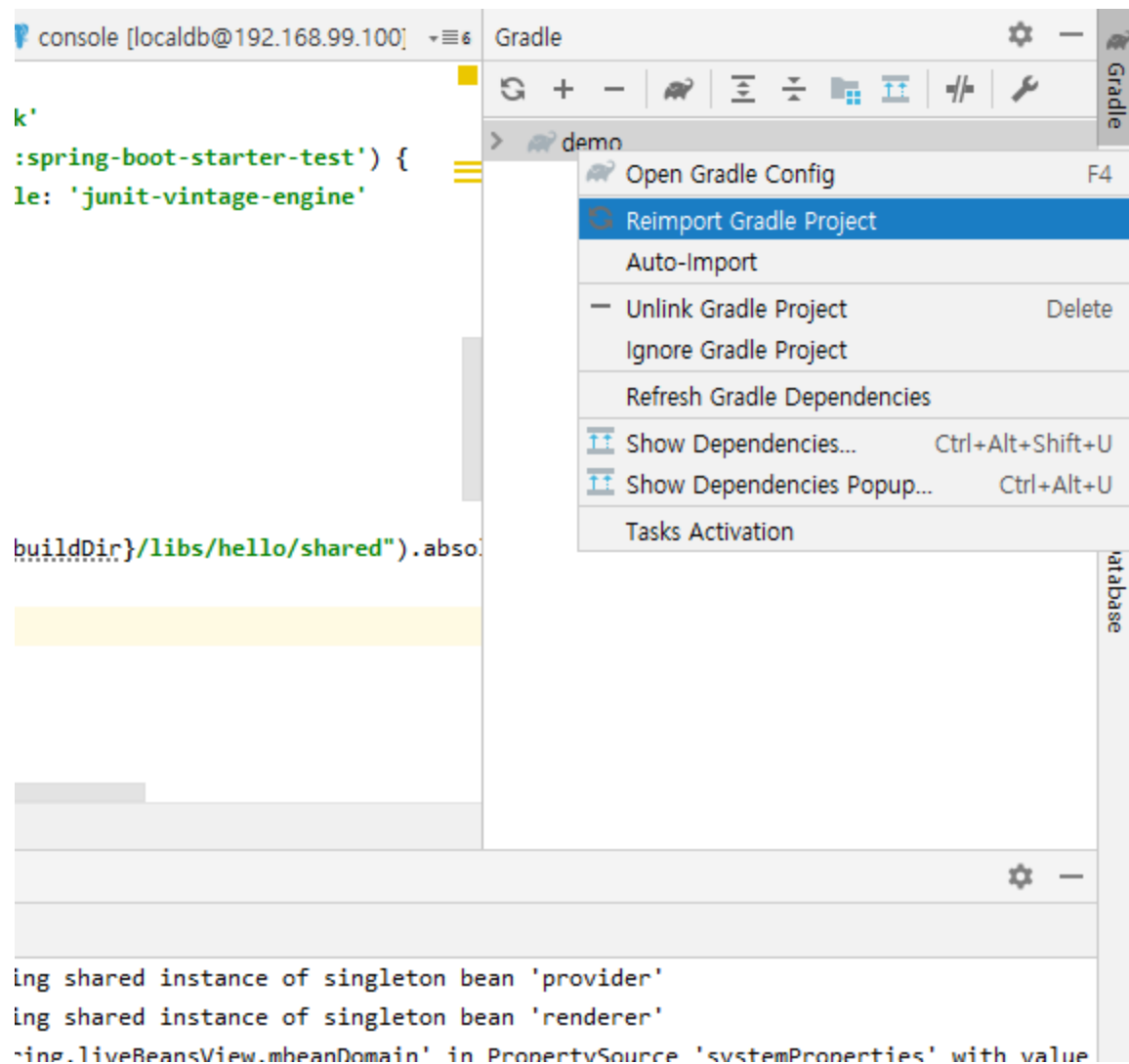













 Filter by title

Home

> Learn

▼ Downloads

Downloads

▼ File and Disk Utilities

File and Disk Utilities

AccessChk

AccessEnum

CacheSet

Contig

Disk2vhd

DiskExt

DiskMon

Sigcheck v2.73

05/22/2017 • 2 minutes to read •    

By Mark Russinovich

Published: September 05, 2019

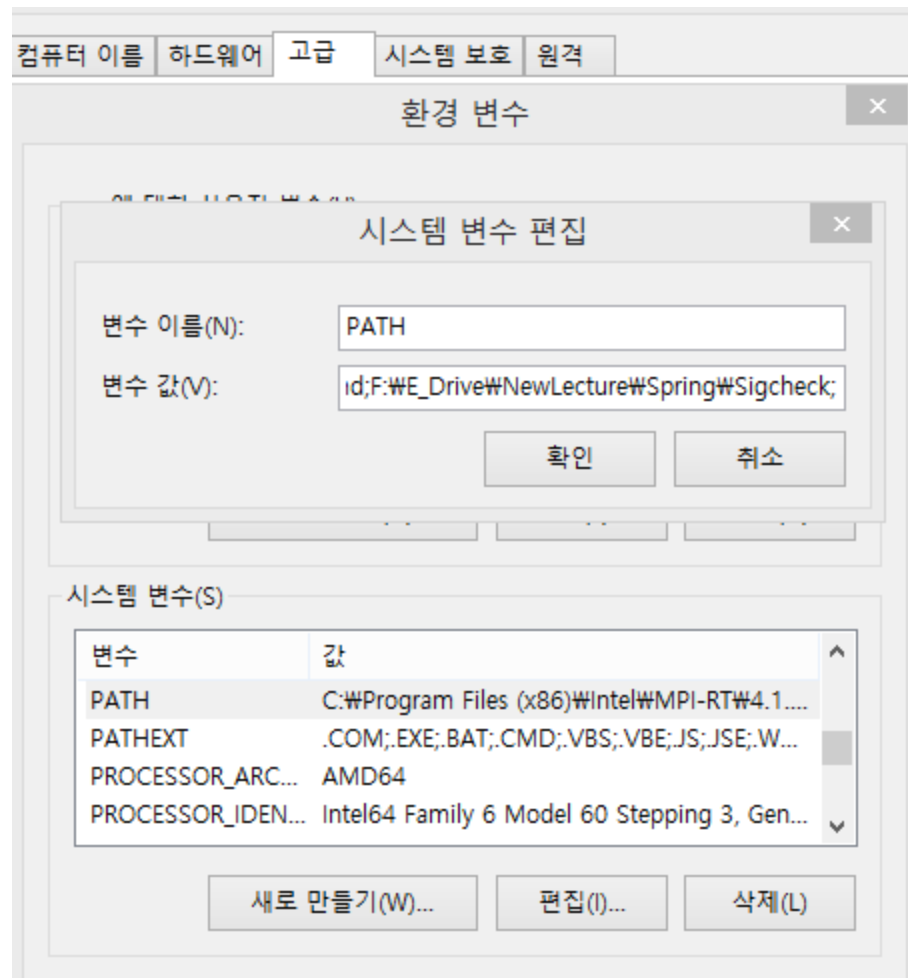


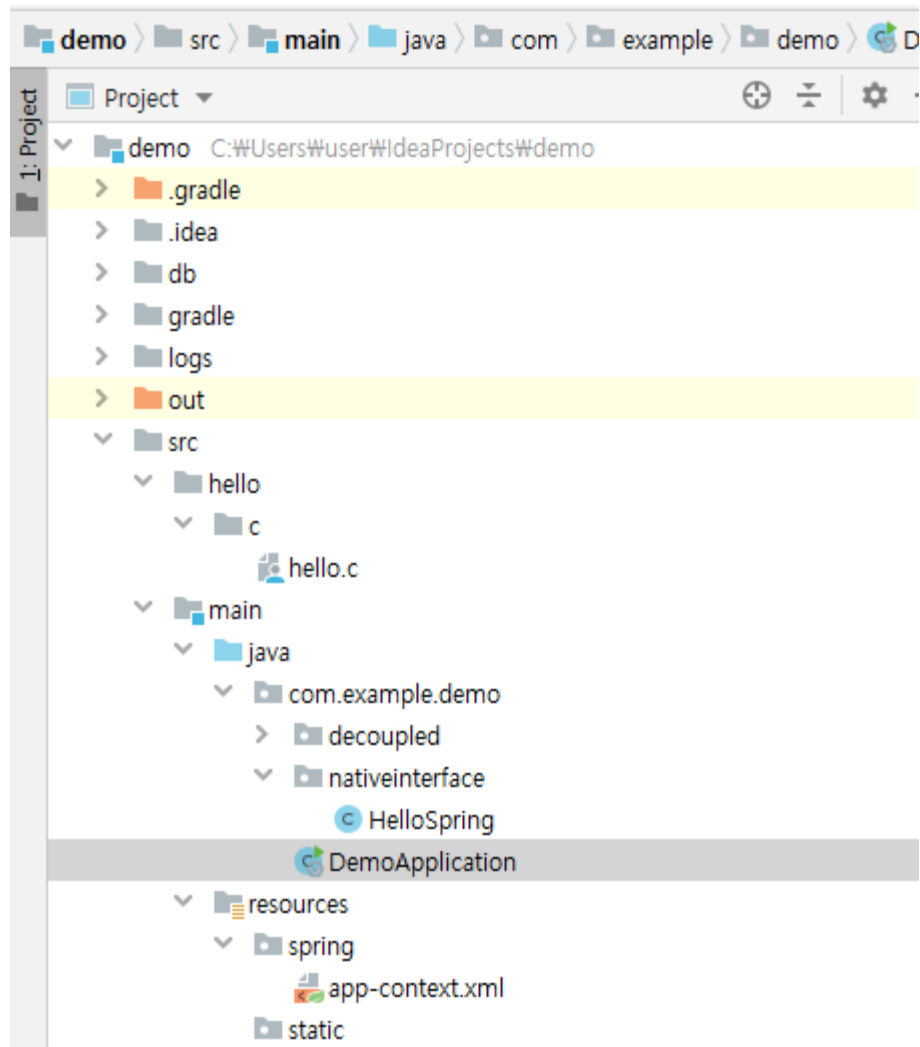
[Download Sigcheck](#) (799 KB)

Introduction

Sigcheck is a command-line utility that sh certificate chains. It also includes an optio over 40 antivirus engines, and an option t

usage: sigcheck [-a][-h][-i][-e][-l][-n][[-





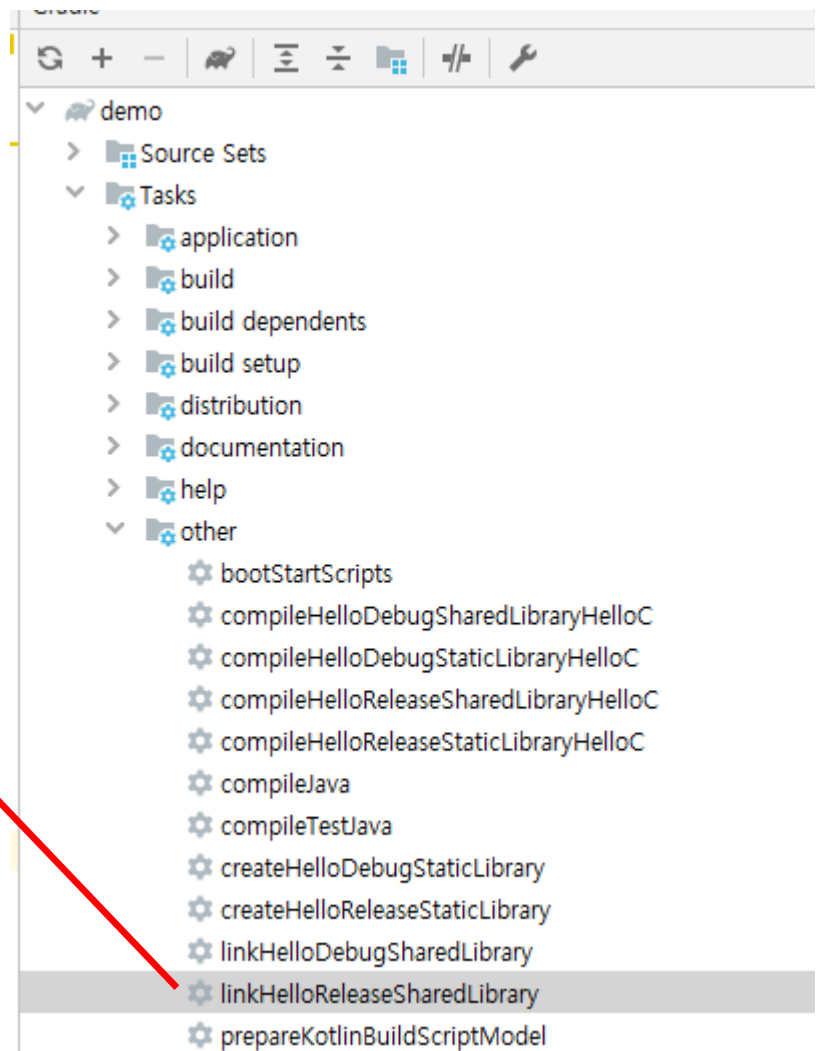
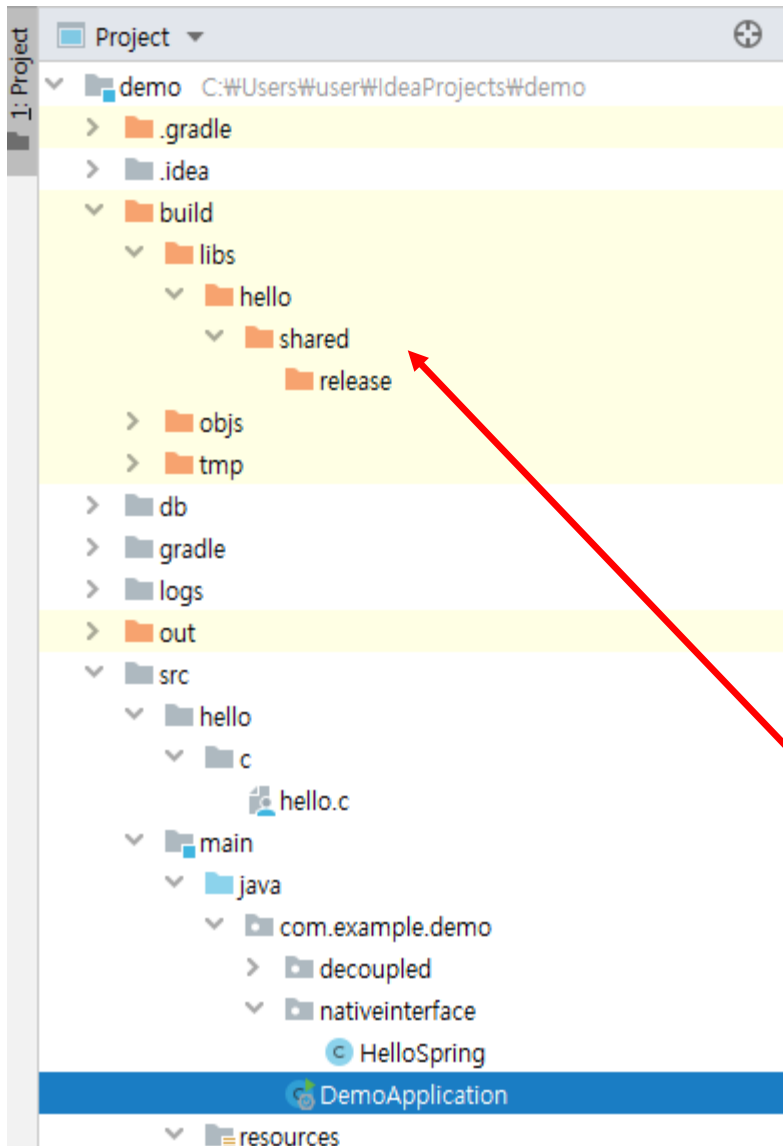
The image shows two panels from an IDE. The left panel, titled 'Project', displays a file tree for a project named 'demo'. The right panel, titled 'Gradle', displays a list of tasks for the same project. A red arrow originates from the 'objs' folder under the 'build' directory in the Project panel and points to the 'compileHelloReleaseSharedLibraryHelloC' task in the Gradle panel.

Project Panel Structure:

- demo (C:\Users\user\IdeaProjects\demo)
 - .gradle
 - .idea
 - build
 - objs
 - tmp
 - db
 - gradle
 - logs
 - out
 - src
 - hello
 - c
 - hello.c
 - main
 - java
 - com.example.demo
 - decoupled
 - nativeinterface
 - HelloSpring

Gradle Panel Structure:

- demo
 - Source Sets
 - Tasks
 - application
 - build
 - build dependents
 - build setup
 - distribution
 - documentation
 - help
 - other
 - bootStartScripts
 - compileHelloDebugSharedLibraryHelloC
 - compileHelloDebugStaticLibraryHelloC
 - compileHelloReleaseSharedLibraryHelloC**
 - compileHelloReleaseStaticLibraryHelloC
 - compileJava
 - compileTestJava
 - createHelloDebugStaticLibrary
 - createHelloReleaseStaticLibrary
 - linkHelloDebugSharedLibrary
 - linkHelloReleaseSharedLibrary
 - prepareKotlinBuildScriptModel
 - processResources
 - processTestResources
 - startScripts



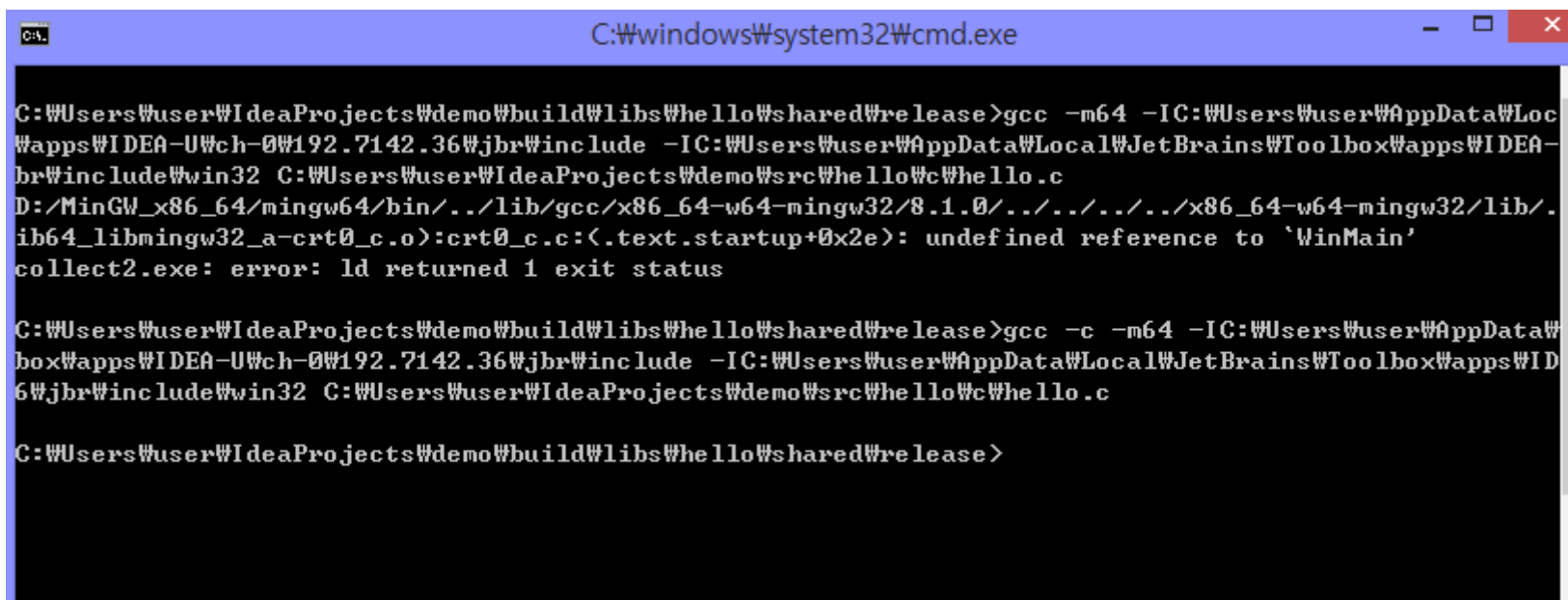
5.2. Tool chain support

Gradle offers the ability to execute the same build using different tool chains. When you build :
“Tool chains” for details.

The following tool chains are supported:

Operating System	Tool Chain	Notes
Linux	GCC	
Linux	Clang	
Mac OS X	XCode	Uses the Clang tool chain bundled with XCode.
Windows	Visual C++	Windows XP and later, Visual C++ 2010 and later.
Windows	GCC with Cywin 32	Windows XP and later.
Windows	GCC with MinGW	Windows XP and later. Mingw-w64 is also supported.

본인의 경로에 맞게 작성해야함



```
C:\windows\system32\cmd.exe

C:\Users\user\IdeaProjects\demo\build\libs\hello\shared\release>gcc -m64 -IC:\Users\user\AppData\Local\apps\IDEA-U\ch-0\192.7142.36\jbr\include -IC:\Users\user\AppData\Local\JetBrains\Toolbox\apps\IDEA-br\include\win32 C:\Users\user\IdeaProjects\demo\src\hello\c\hello.c
D:/MinGW_x86_64/mingw64/bin/./lib/gcc/x86_64-w64-mingw32/8.1.0/./././././x86_64-w64-mingw32/lib/.ib64_libmingw32_a-crt0_c.o): crt0_c.c:(.text.startup+0x2e): undefined reference to `WinMain'
collect2.exe: error: ld returned 1 exit status

C:\Users\user\IdeaProjects\demo\build\libs\hello\shared\release>gcc -c -m64 -IC:\Users\user\AppData\box\apps\IDEA-U\ch-0\192.7142.36\jbr\include -IC:\Users\user\AppData\Local\JetBrains\Toolbox\apps\ID6\jbr\include\win32 C:\Users\user\IdeaProjects\demo\src\hello\c\hello.c

C:\Users\user\IdeaProjects\demo\build\libs\hello\shared\release>
```

```
C:\Users\user\IdeaProjects\demo\build\libs\hello\shared\release>gcc -shared -o hello.dll hello.o
```

```
C:\Users\user\IdeaProjects\demo\build\libs\hello\shared\release>dir
```

c 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: EA43-F6F0

```
C:\Users\user\IdeaProjects\demo\build\libs\hello\shared\release 디렉터리
```

```
2019-11-27 오후 03:49 <DIR> .
2019-11-27 오후 03:49 <DIR> ..
2019-11-27 오후 03:49          48,055 hello.dll
2019-11-27 오후 03:47          1,070 hello.o
                2개 파일          49,125 바이트
                2개 디렉터리 16,677,482,496 바이트 남음
```

```
C:\Users\user\IdeaProjects\demo\build\libs\hello\shared\release>sigcheck.exe hello.dll
```

Sigcheck v2.73 - File version and signature viewer

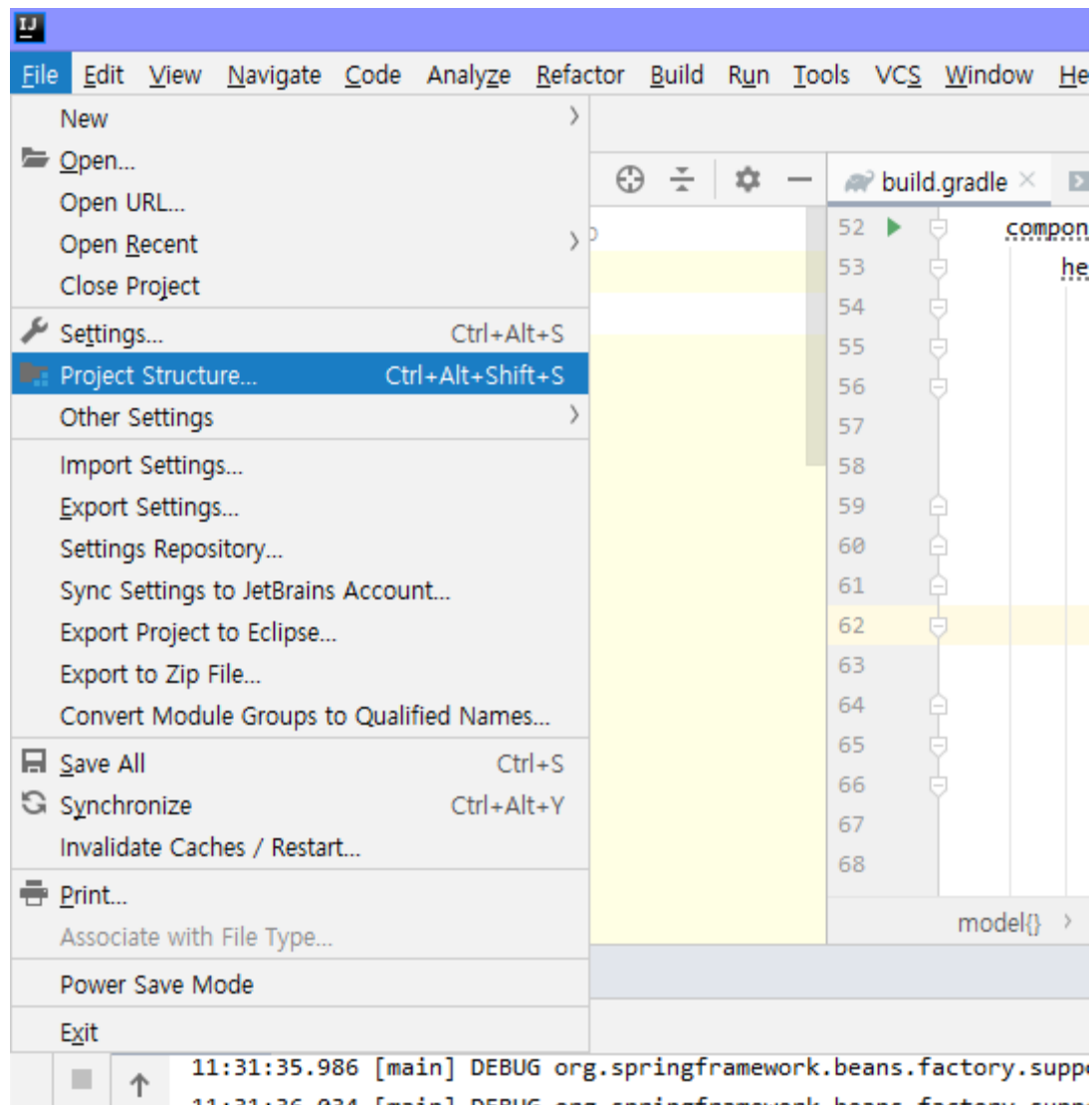
Copyright (C) 2004-2019 Mark Russinovich

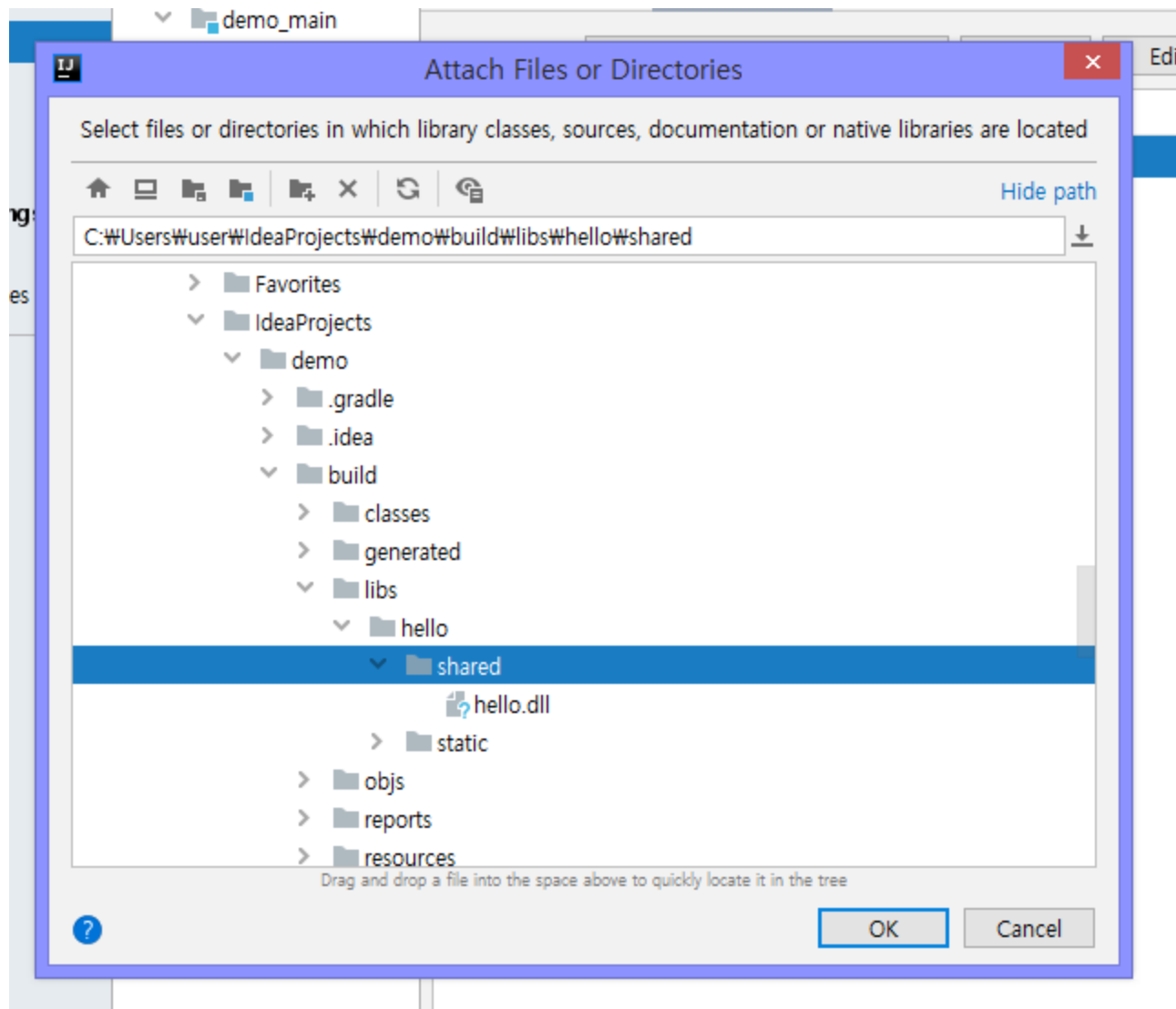
Sysinternals - www.sysinternals.com

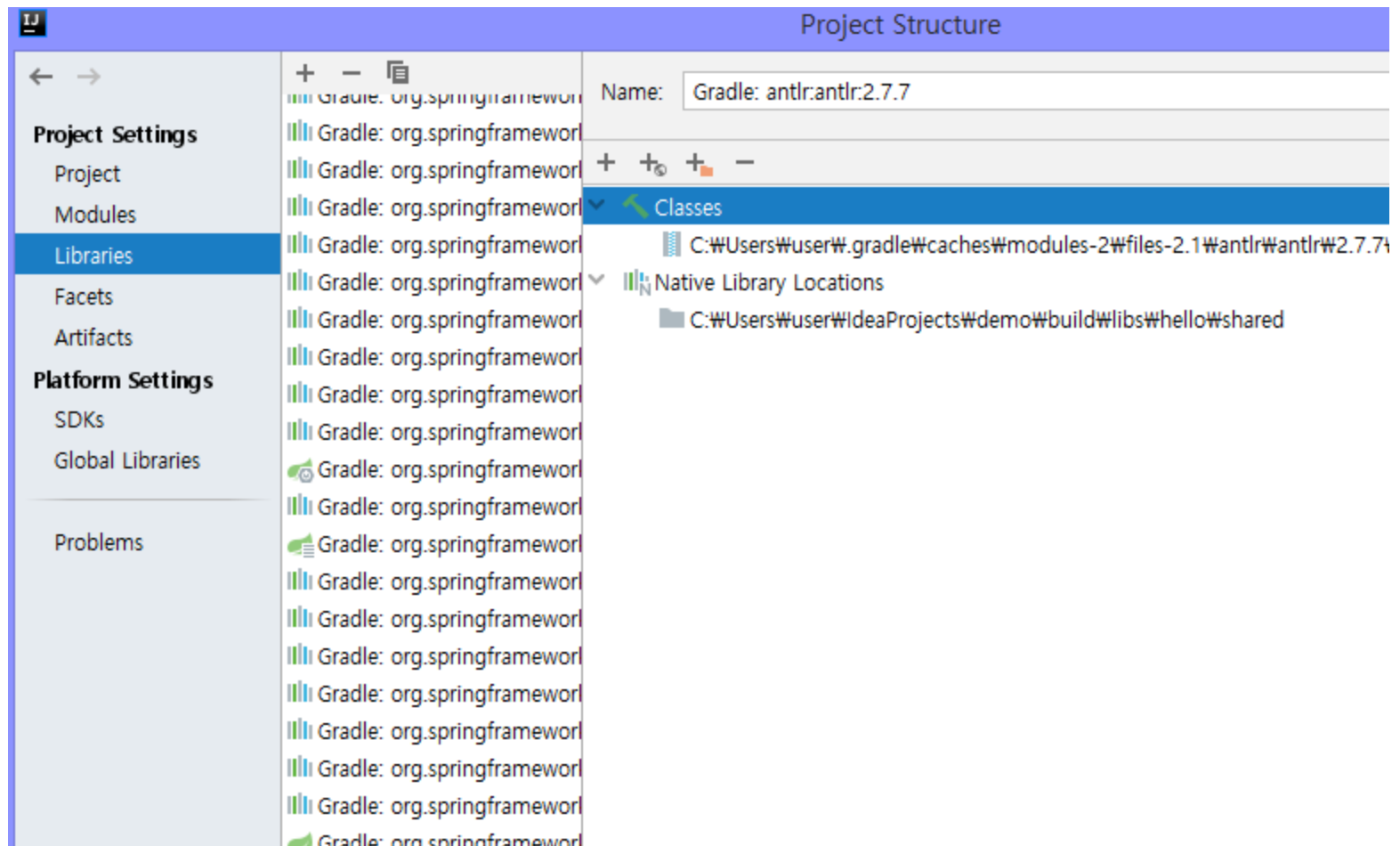
```
C:\Users\user\IdeaProjects\demo\build\libs\hello\shared\release\hello.dll:
```

Verified:	Unsigned
Link date:	?? 3:49 2019-11-27
Publisher:	n/a
Company:	n/a
Description:	n/a
Product:	n/a
Prod version:	n/a
File version:	n/a
MachineType:	64-bit

```
C:\Users\user\IdeaProjects\demo\build\libs\hello\shared\release>
```







hello.un

hello.o

> x64

> objs

> reports

> resources

> test-results

> tmp

> db

> gradle

> logs

98

99

100

101

102

103

104

105

106

107

108

ccompiler.args -1 , {org.gi

}

*/

}

}

/*

toolChains {

gcc(Gcc){

target("windows_x86_64") {

compilerExecutable = "D:/MinGW

model{} > components{} > hello{} > all{}

Run: DemoApplication x

Console

Endpoints

15:49:41.379 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared

15:49:41.382 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared

15:49:41.385 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared

15:49:41.387 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared

15:49:41.394 [main] DEBUG org.springframework.core.env.PropertySourcesPropertyResolver - Found key 'spring.liveBe

Hello Spring

15:49:41.406 [main] INFO com.example.demo.nativeinterface.HelloSpring - *****

15:49:41.406 [main] INFO com.example.demo.nativeinterface.HelloSpring - *** Native library initialization sequenc

15:49:41.406 [main] INFO com.example.demo.nativeinterface.HelloSpring - *** java.library.path: C:\Users\user\Idea

LOGBACK: No context given for c.q.l.core.rolling.SizeAndTimeBasedRollingPolicy@2078641942

.

/ \ / _ ' _ _ _ () _ _ _ _ \ \ \ \

(() \ _ | ' _ | ' _ | ' _ V _ | \ \ \ \

\ \ _ | |) | | | | | | (| |))))

' | _ | . _ | | | | | \ , | / / / /

===== | _ | ===== | _ / = / / / /

:: Spring Boot :: (v2.2.1.RELEASE)

32