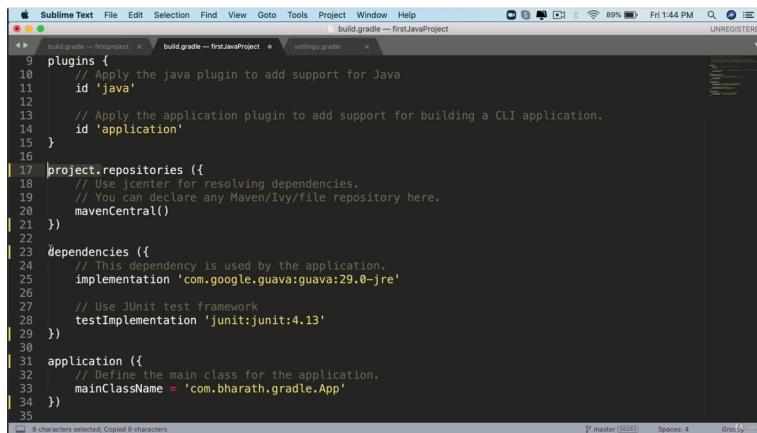


Everything that we saw is part of the project object. We do not need to call them explicitly and brackets are not required as well. Notice line number 17. Application method brackets are optional. Every method is part of the project object and it is not required to call them using project object explicitly. Every block is closer which is nothing but a lambda.

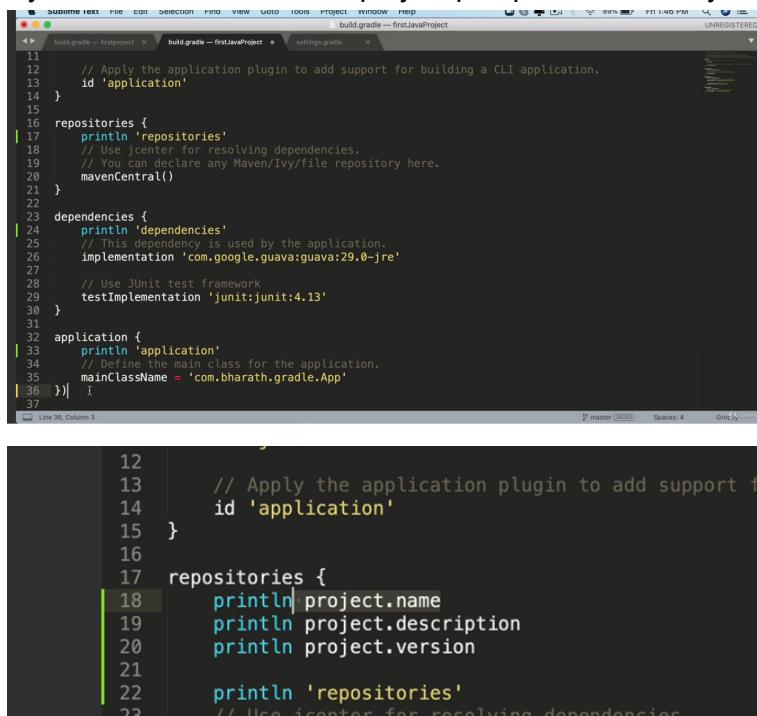


```

 9 plugins {
10     // Apply the java plugin to add support for Java
11     id 'java'
12
13     // Apply the application plugin to add support for building a CLI application.
14     id 'application'
15 }
16
17 project.repositories {
18     // Use JCenter for resolving dependencies.
19     // You can declare any Maven/Ivy/file repository here.
20     mavenCentral()
21 }
22
23 dependencies {
24     // This dependency is used by the application.
25     implementation 'com.google.guava:guava:29.0-jre'
26
27     // Use JUnit test framework
28     testImplementation 'junit:junit:4.13'
29 }
30
31 application {
32     // Define the main class for the application.
33     mainClassName = 'com.bharath.gradle.App'
34 }
35

```

If you notice this file from a project perspective it is very easy to understand.

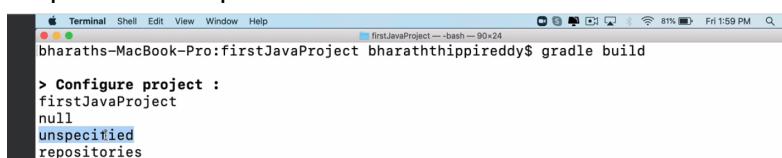


```

11
12     // Apply the application plugin to add support for building a CLI application.
13     id 'application'
14 }
15
16 repositories {
17     println 'repositories'
18     // Use JCenter for resolving dependencies.
19     // You can declare any Maven/Ivy/file repository here.
20     mavenCentral()
21 }
22
23 dependencies {
24     println 'dependencies'
25     // This dependency is used by the application.
26     implementation 'com.google.guava:guava:29.0-jre'
27
28     // Use JUnit test framework
29     testImplementation 'junit:junit:4.13'
30 }
31
32 application {
33     println 'application'
34     // Define the main class for the application.
35     mainClassName = 'com.bharath.gradle.App'
36 }|| I
37

```

Output of above print statement

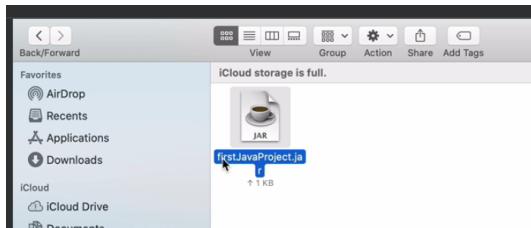


```

Terminal Shell Edit View Window Help
bharaths-MacBook-Pro:firstJavaProject bharathreddy$ gradle build
> Configure project :
firstJavaProject
null
unspecified
repositories

```

Jar file it is creating without version.



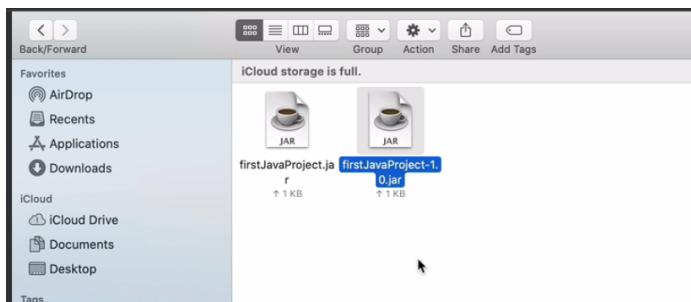
Project name comes from the settings file. Specify other values as specified below

```
8
9 plugins {
10   // Apply the java plugin to add support for
11   id 'java'
12
13   // Apply the application plugin to add support
14   id 'application'
15 }
16
17 project.description="First Java Project"
18 project.version=1.0
19
20 repositories {
21   println project.name
22   println project.description
23   println project.version
```

```
BUILD SUCCESSFUL in 976ms
7 actionable tasks: 4 executed, 3 up-to-date
bharaths-MacBook-Pro:firstJavaProject bharaththippireddy$ gradle build

> Configure project :
firstJavaProject
First Java Project
1.0
repositories
dependencies
application
```

Jar created with the version we provided above.



All info about this project object specified below

```
public interface Project
extends Comparable<Project>, ExtensionAware, PluginAware
```

This interface is the main API you use to interact with Gradle from your build file. From a `Project`, you have programmatic access to the build environment.

Project properties

Name	Type	Default Value
project	Project	The Project instance
name	String	The name of the project directory.
path	String	The absolute path of the project.
description	String	A description for the project.
projectDir	File	The directory containing the build script.
buildDir	File	projectDir/build
group	Object	unspecified
version	Object	unspecified
ant	AntBuilder	An AntBuilder instance

Specify some of the properties

```
12 // Apply the application plugin to add su
13     id 'application'
14 }
15
16
17 project.description="First Java Project"
18 project.version=1.0
19
20 repositories {
21     println project.name
22     println project.description
23     println project.version
24     println project.path
25     println project.projectDir
26 }
```

```
lrunable tasks. 0 executed
aths-MacBook-Pro:firstJavaProject bharaththippireddy$ gradle clean build

nfigure project :
JavaProject
t Java Project

rs/bharaththippireddy/Documents/gradle/firstJavaProject
itories
```

We can also create user defined properties (xyz) and how can we access shown below. No need to use ext while retrieving values.

```
15 }
16
17 project.description="First Java Project"
18 project.version=1.0
19 project.ext.xyz="abc"
20
21 repositories {
22     println project.name
23     println project.description
24     println project.version
25     println project.path
26     println project.projectDir
27     println project.xyz
28     println project.property("xyz")
29
30     println 'repositories'
```

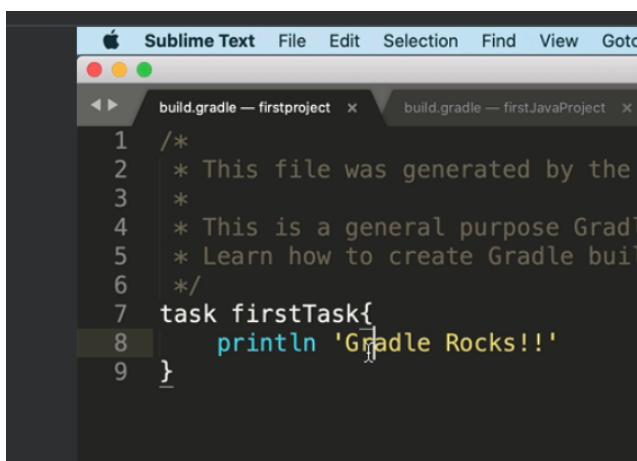
```
bharaths-MacBook-Pro:firstJavaProject bharaththippireddy$ gradle clean build

> Configure project :
firstJavaProject
First Java Project
1.0
:
/Users/bharaththippireddy/Documents/gradle/firstJavaProject
abc
abc
repositories
```

So far we have seen project objects. Now let us see the task.

Gradlew compile will create all the tasks required and run them implicitly.

```
10
11 Task task = new compile| x
12
13    */
14
15    project.addTask(task)
16
```



The above one equal to below one.

```
5 * Learn how to create Gradle bui
6 */
7 task(firstTask){
8     println 'Gradle Rocks!!'
9 }
```

There are so many overloaded methods

```
task

Task task(Map<String,?> args,
          String name,
          Closure configureClosure)

Creates a Task with the given name and adds it to this project. Before the task is returned, t
can be passed to this method to control how the task is created. See task(java.util.Map,
After the task is added to the project, it is made available as a property of the project, so tha
details
```

We can define variables and use them as follow. When variable name have conflict with project object properties, call them using project.abc.

```
6 */
7 task(firstTask){
8     def abc=123
9     println abc
10    println 'Gradle Rocks!!'
11 }
12 }
```

For example we gralde jar (jar is task - internally it run sequence of job like compile , run and install)

```
BUILD SUCCESSFUL in 600ms
bharaths-MacBook-Pro:firstproject bharaththippireddy$ gradle jar
```

```
3 *
4 * This is a general purpose Gradle build.
5 * Learn how to create Gradle builds at http
6 */
7 task(firstTask){
8     def abc=123
9     println abc
10    println 'Gradle Rocks!!'
11 }
12
13 task deployToStage{
14     doLast(){
15         println "Deployed to stage"
16     }
17 }
```

```
BUILD SUCCESSFUL in 621ms
bharaths-MacBook-Pro:firstproject bharaththippireddy$ gradle dTS

> Configure project :
123
Gradle Rocks!!

> Task :deployToStage
Deployed to stage
```

Notice how sequence is working from line number 25

```
6  */
7 task (firstTask){
8     def abc=123
9     println abc
10    println 'Gradle Rocks!!'
11 }
12
13 task deployToStage{
14     doLast(){
15         println "Deployed to stage"
16     }
17 }
18
19 task deployToProd{
20     doLast(){
21         println "Deployed to prod"
22     }
23 }
24
25 deployToProd.dependsOn deployToStage
```

Notice the sequence of execution below.

```
BUILD SUCCESSFUL in 600ms
1 actionable task: 1 executed
bharaths-MacBook-Pro:firstproject bharaththippireddy$ gradle dTP

> Configure project :
123
Gradle Rocks!!

> Task :deployToStage
Deployed to stage

> Task :deployToProd
Deployed to prod

BUILD SUCCESSFUL in 604ms
2 actionable tasks: 2 executed

bharaths-MacBook-Pro:firstproject bharaththippireddy$ gradle dTP

> Configure project :
123
Gradle Rocks!!

> Task :deployToStage
Deployed to stage

> Task :deployToProd
Deployed to prod

> Task :cleanupFiles
Cleaned Up Files

BUILD SUCCESSFUL in 617ms
3 actionable tasks: 3 executed
```

FinalizedBy

```
build.gradle — firstproject * build.gradle — firstJavaProject x settings.gradle x
8     def abc=123
9         println abc
10        println 'Gradle Rocks!!'
11    }
12
13 task deployToStage{
14     doLast(){
15         println "Deployed to stage"
16     }
17 }
18
19 task deployToProd{
20     doLast(){
21         println "Deployed to prod"
22     }
23 }
24
25 task cleanupFiles{
26     doLast(){
27         println 'Cleaned Up Files'
28     }
29 }
30
31 deployToProd.dependsOn deployToStage
32 deployToProd.finalizedBy cleanupFiles
33
34
```

Line 34, Column 1

```
bharaths-MacBook-Pro:firstproject bharaththippireddy$ gradle dTP
> Configure project :
123
Gradle Rocks!!

> Task :deployToStage
Deployed to stage

> Task :deployToProd
Deployed to prod

> Task :cleanupFiles
Cleaned Up Files

BUILD SUCCESSFUL in 617ms
3 actionable tasks: 3 executed
```

```
28
29 }
30
31 deployToProd.dependsOn deployToStage
32 deployToProd.finalizedBy cleanupFiles
33 defaultTasks "deployToStage"
34
```

Line 33, Column 28, Saved ~/Documents/gradle/firstproject/build.gradle (UTF-8)

Just pass gradle - notice deploy to stage executes as it is the default task as we coded above.

```
bharaths-MacBook-Pro:firstproject bharaththippireddy$ gradle

> Configure project :
123
Gradle Rocks!!

> Task :deployToStage
Deployed to stage

BUILD SUCCESSFUL in 624ms
1 actionable task: 1 executed
```

Good job!

Question 1:
Every build.gradle has an object of which of the following Gradle api classes

Build
 Task
 Settings
 Project

Good job!

Question 2:
Which of the following should be used to add custom property called abc on the gradle project object

project.props.abc
 project.abc
 projectabc.ext
 projectext.abc



Question 3:
Which of the following method in a task will be run during the execution phase

execute
 doFirst
 doLast
 last

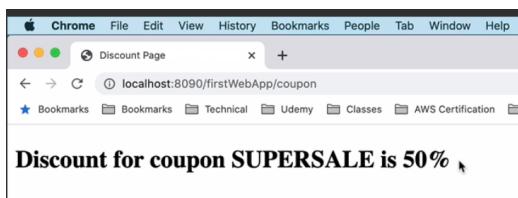
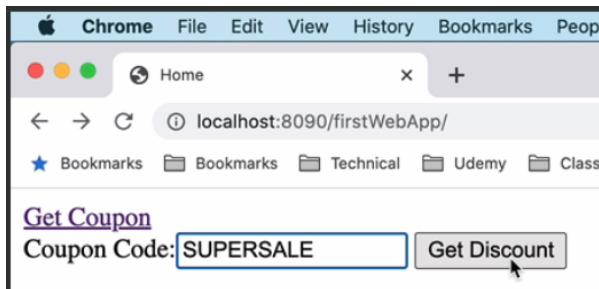
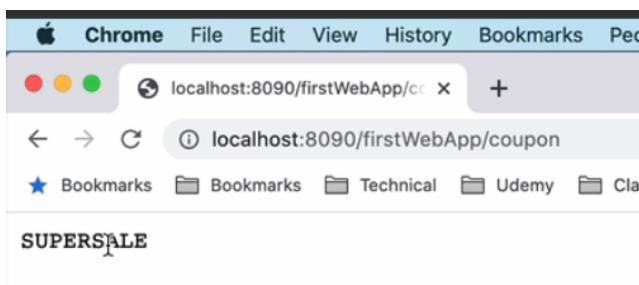
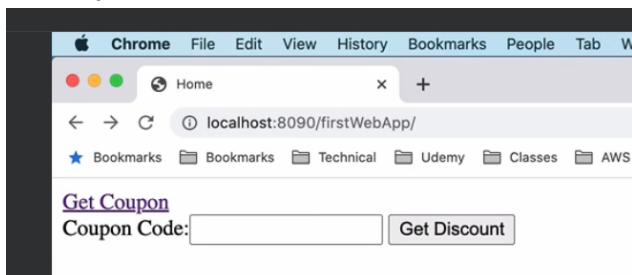


This was discussed in Lecture 29: Task variables and methods >

Question 4:
Using which of the below can we always run a task after another task runs

dependsOn
 finalizedBy

Create java web application



```
<!DOCTYPE html>
<html>
<head>
    <title>Home</title>
</head>
<body>
    <a href="coupon">Get Coupon</a>
    <form method="post" action="coupon">
        Coupon Code:<input type="text" name="coupon">
        <input type="submit" value="Get Discount">
    </form>
</body>
</html>
```

```
package com.bharath.gradle;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/coupon")
public class CouponServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.getWriter().print("SUPERSALE");
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String coupon = request.getParameter("coupon");
        request.setAttribute("discount", "Discount for coupon " + coupon + " is 50%");
        request.getRequestDispatcher("response.jsp").forward(request, response);
    }
}
```

```
<%@ page contentType="text/html;charset=UTF-8" language="java"%>
<!DOCTYPE html>
<html>
<head>
    <title>Discount Page</title>
</head>
<body>
    <h2>${discount}</h2>
</body>
</html>
```

Notice the plugin 'war'

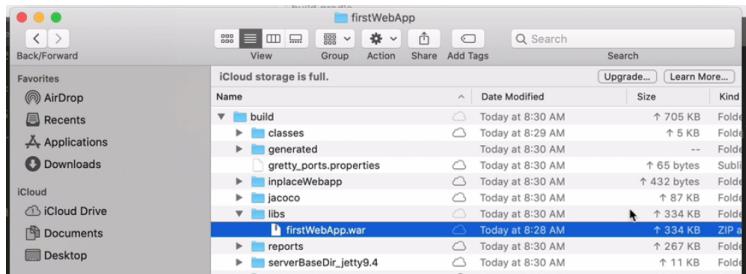
```
* This file was generated by the Gradle 'init' task.
*
* This generated file contains a sample Java project to get you started.
* For more details take a look at the Java Quickstart chapter in the Gradle
* User Manual available at https://docs.gradle.org/6.5.1/userguide/tutorial_java_projects.

plugins {
    // Apply the java plugin to add support for Java
    id 'war'
    id 'org.gretty' version '3.0.3'
}

repositories {
    // Use jcenter for resolving dependencies.
    // You can declare any Maven/Ivy/file repository here.
    jcenter()
}

dependencies {
    providedCompile 'javax.servlet:javax.servlet-api:3.1.0'
    compile group: 'com.fasterxml.jackson.core', name: 'jackson-core', version: '2.11.2'
}
gretty.httpPort=8090
```

Gradlew build generate the war file so we can deploy in tomcat server.



How to overcome this.

Or use the gretty plugin- this helps us to run the application as part of gradle build.

```
/*
8
9 plugins {
10     // Apply the java plugin to add support for Java
11     id 'war'
12     id 'org.gretty' version '3.0.3'
13 }
14
```

```
Terminal Shell Edit View Window Help
firstWebApp - doctor@minikube ~ -- java Xmx64m Xms64m -Dorg.gradle.appname=gradle -classpath /usr/local/Cellar/gradle/7.3.3/libexec/gradle-launcher-7.3.3.jar org.gradle.launcher.GradleMain inst --bg 2
bharaththippireddy@bharaths-MacBook-Pro-2 gradle % cd firstWebApp
bharaththippireddy@bharaths-MacBook-Pro-2 firstWebApp % gradle init

Select type of project to generate:
1: basic
2: application
3: library
4: Gradle plugin
Enter selection (default: basic) [1..4] 2

Select implementation language:
1: C++
2: Groovy
3: Java
4: Kotlin
5: Scala
6: Swift
Enter selection (default: Java) [1..6] 3

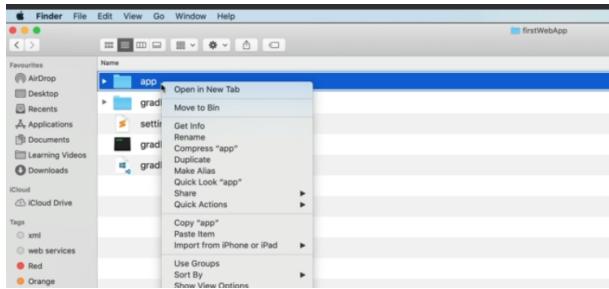
Split functionality across multiple subprojects?:
1: no - only one application project
2: yes - application and library projects
Enter selection (default: no - only one application project) [1..2] 1
```

```
Select build script DSL:
1: Groovy
2: Kotlin
Enter selection (default: Groovy) [1..2] 1

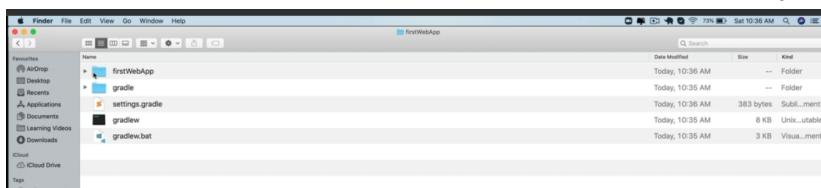
Generate build using new APIs and behavior (some features may change in the next minor re

Select test framework:
1: JUnit 4
2: TestNG
3: Spock
4: JUnit Jupiter
Enter selection (default: JUnit Jupiter) [1..4] 1

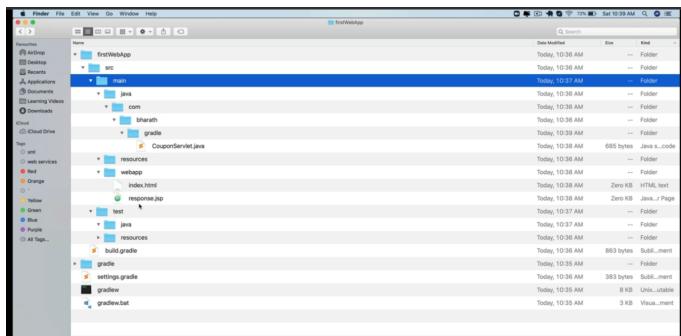
Project name (default: firstWebApp):
Source package (default: firstWebApp): com.bharath.gradle
```



Rename app folder to firstWebApp as shown below. Similarly in settings file.



Create webapp folder and create index.html and response.jsp skeleton files and in gradle file copy and paste CouponServlet.java



Copy the corresponding code and past it.

Change the plugin name from application to war. War is child plugin of war and cleanup build.gradle

```

1 /*
2  * This file was generated by the Gradle 'init' task.
3  *
4  * This generated file contains a sample Java application project.
5  * For more details take a look at the 'Building Java & JVM projects'
6  * User Manual available at https://docs.gradle.org/7.3.3/userguide/
7  */
8
9 plugins {
10     id 'war'
11 }
12
13 repositories {
14     // Use Maven Central for resolving dependencies.
15     mavenCentral()
16 }
17
18 dependencies {
19 }
20
21

```

We will see war task as shown below.

```
bharaththippireddy@bharaths-MacBook-Pro-2 firstWebApp % gradle tasks --all
> Task :tasks

Tasks runnable from root project 'firstWebApp'

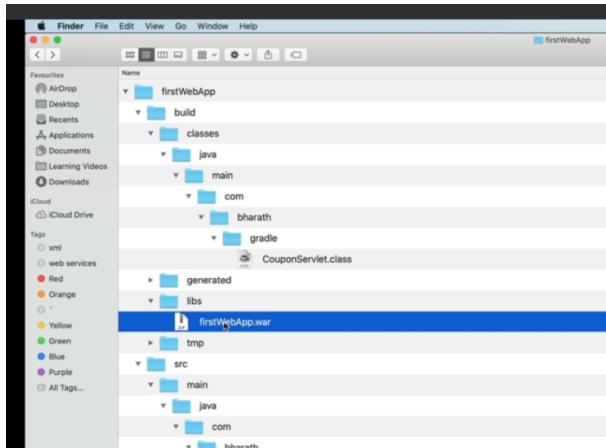
Build tasks
-----
firstWebApp:assemble - Assembles the outputs of this project.
firstWebApp:build - Assembles and tests this project.
firstWebApp:buildDependents - Assembles and tests this project and all projects
nd on it.
firstWebApp:buildNeeded - Assembles and tests this project and all projects it
depends on.
firstWebApp:classes - Assembles main classes.
firstWebApp:clean - Deletes the build directory.
firstWebApp:jar - Assembles a jar archive containing the main classes.
firstWebApp:testClasses - Assembles test classes.
firstWebApp:war - Generates a war archive with all the compiled classes, the web
content and the libraries.
```

Copy this dependency and paste it as shown below

```
16 }
17
18 dependencies {
19     compileOnly group: 'jakarta.servlet', name: 'jakarta.servlet-api', version:
20 }
21
```

We are using this only for compile and we do not want this packaging that is why we had to keep it as compileOnly.

It creates war file.



Add gretty plugin as well as shown below. It make our web application run on jetty server.

```
9 plugins {
10     id 'war'
11     id 'org.gretty' version '4.0.3'
12 }
13

s (if any) to ${buildDir}/inplaceWebApp
firstWebApp:showClassPath - Shows classpath information
firstWebApp:tomcatRestart - Sends 'restart' command to a running server.
firstWebApp:tomcatRun - Starts web-app inplace, in interactive mode.
firstWebApp:tomcatRunDebug - Starts web-app inplace, in debug and interactive mode.
firstWebApp:tomcatRunWar - Starts web-app on WAR-file, in interactive mode.
firstWebApp:tomcatRunWarDebug - Starts web-app on WAR-file, in debug and interactive mode
```

Just run gradle appRun as shown below

```
bharaththippireddy@bharaths-MacBook-Pro-2 firstWebApp % gradle appRun
21:33:09.163 [main] DEBUG o.akhikhil.gretty.JettyServerManager - Jetty 11.0.3 starting.
21:33:09.420 [main] DEBUG o.a.gretty.JettyServerConfigurer - jetty context temp directory
: /Users/bharaththippireddy/Documents/gradle/firstWebApp/firstWebApp/build/serverBaseDir_
jetty11/webapps-exploded/firstWebApp
21:33:09.642 [main] INFO o.a.gretty.JettyServerStartInfo - Jetty 11.0.3 started and list
ening on port 8080
21:33:09.653 [main] INFO o.a.gretty.JettyServerStartInfo - firstWebApp runs at:
21:33:09.653 [main] INFO o.a.gretty.JettyServerStartInfo - http://localhost:8080/first
WebApp
21:33:09.658 [main] DEBUG org.akhikhil.gretty.ServiceProtocol - ServiceProtocol.send(52205
, {"status":"successfully started","host":"localhost","httpPort":8080,"contexts":[{"proto
col":"http","host":"localhost","port":8080,"contextPath":"/firstWebApp","baseURI":"http:/
localhost:8080/firstWebApp"}])
21:33:09.658 [main] DEBUG o.akhikhil.gretty.JettyServerManager - Jetty 11.0.3 started.

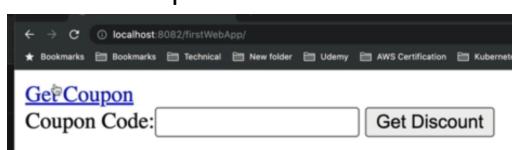
> Task :firstWebApp:appRun
Press any key to stop the server.
<=====--> 87% EXECUTING [9s]
> :firstWebApp:appRun
```

We can also run on tomcat as shown below.

```
Terminal Shell View Window Help
File Recent File Recent Minimise — Java Xcode6.x Xcode6.4m Gradle appRun --gradle-classpath /var/local/Cellar/gradle/7.3.3/libexec/lib/gradle-launcher-7.3.3.jar org.gradle.launcher.GradleMain tomcatRun
bharaththippireddy@bharaths-MacBook-Pro-2 firstWebApp % gradle tomcatRun
<=====--> 87% EXECUTING [618ms]
> :firstWebApp:tomcatRun
```

```
19
20 dependencies {
21     // Use JUnit test frame
22     compileOnly(group:"jak
23 }
24
25 gretty.httpPort = 8082
```

We can also port like this





Question 1:
Which plugin can be used to run a web application

war
 tomcat
 jetty
 gretty



Question 2:
The war plugin also will bring in the java plugin tasks

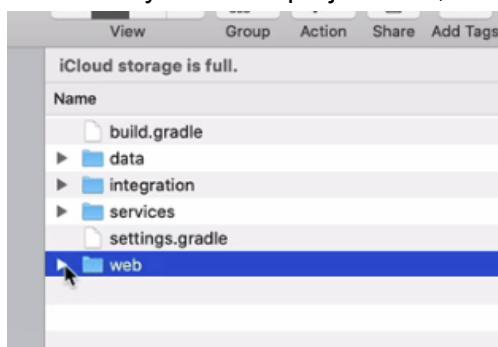
True
 False



Question 3:
Which task can be used to run the web application when using gretty plugin

runApp
 run
 app
 appRun

Different layers in the project web, service, integration and data layers



Include those modules in setting file

```
build.gradle — couponapp | settings.gradle | build.gradle — couponapp/services
1
2 * This file was generated by the Gradle 'init' task
3 *
4 * The settings file is used to specify which projects
5 * are included in the build.
6 * Detailed information about configuring a multi-project
7 * build is available at https://docs.gradle.org/6.0/userguide/multi_project_builds.html
8 */
9 rootProject.name = 'couponapp'
10 include('web', 'services', 'data', 'integration')
```

Or

```
9
10 rootProject.name = 'couponapp'
11 include('web')
12 include('services')
13 include('data')
14 include('integration')
15
```

```

10
17 project(':web'){
18     dependencies{
19         implementation project(':services')
20     }
21 }
22
23 project(':services'){
24     dependencies{
25         implementation project(':data')
26     }
27 }
28

```

```

Terminal Shell Edit View Window Help
○ Terminal → couponapp → java → bharathippireddybharaths-MacBook-Pro couponapp % gradle init
bharathippredy@bharaths-MacBook-Pro couponapp % gradle init

Select type of project to generate:
1: basic
2: application
3: library
4: Gradle plugin
Enter selection (default: basic) [1..4] 2

Select implementation language:
1: C++
2: Groovy
3: Java
4: Kotlin
5: Scala
6: Swift
Enter selection (default: Java) [1..6] 3

Split functionality across multiple subprojects?:
1: no - only one application project
2: yes - application and library projects
Enter selection (default: no - only one application project) [1..2] 1

```

```

2: yes - application and library projects
Enter selection (default: no - only one application project) [1..2] 1

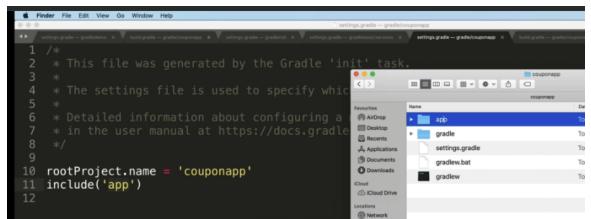
Select build script DSL:
1: Groovy
2: Kotlin
Enter selection (default: Groovy) [1..2] 1

Select test framework:
1: JUnit 4
2: TestNG
3: Spock
4: JUnit Jupiter
Enter selection (default: JUnit 4) [1..4]

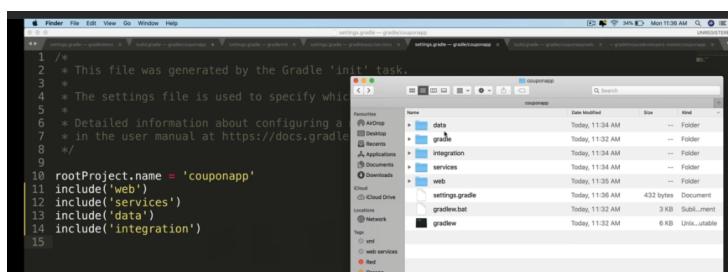
Project name (default: couponapp):
Source package (default: couponapp): com.bharath.gradle

```

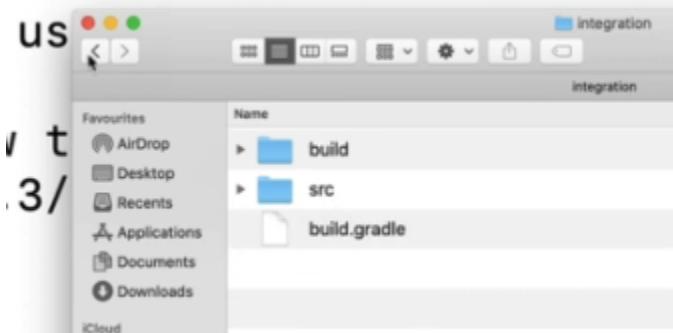
It creates app folder by default from from gradle 6.6 onwards



Create folders as shown below and include them in settings



After this we will see the build folder and build.gradle file in every layer.



Gradle clean will remove the build folder from all the folders.

If we run ‘gradle task’ from the root folder it will do all the tasks of every module. In order to see specific module tasks run the same command from the module folder.

Create a task in the main build.gradle file

The screenshot shows a Sublime Text window with the title bar "Sublime Text" and menu items "File", "Edit", "Selection", "Find", "View". The window has three tabs: "build.gradle" (the active tab), "gradle-wrapper.properties", and "gradle.properties". The "build.gradle" tab contains the following Groovy-like code:

```
task printProjectName{
    doLast(){
        println project.name
    }
}
```

Below the code editor is a terminal window with the following output:

```
bharaths-MacBook-Pro:couponapp bharathhippireddy$ gradle pPN  
> Task :printProjectName  
couponapp
```

When you try same command from web we get the error

```
BUILD SUCCESSFUL in 688ms
1 actionable task: 1 executed
bharaths-MacBook-Pro:couponapp bharaththippireddy$ cd web
bharaths-MacBook-Pro:web bharaththippireddy$ gradle pPN
```

FAILURE: Build failed with an exception.

* What went wrong:
Task 'pPN' not found in root project 'web'.

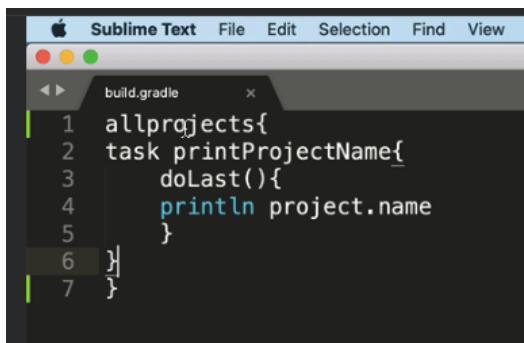
How to run the submodule tasks from the root module

```
Terminal Shell Edit View Window Help couponapp — bash — 84x23
bharaths-MacBook-Pro:couponapp bharaththippireddy$ pwd
/Users/bharaththippireddy/Documents/gradle/couponapp
bharaths-MacBook-Pro:couponapp bharaththippireddy$ gradle :web:clean

Deprecated Gradle features were used in this build, making it incompatible with Gradle 7.0.
Use '--warning-mode all' to show the individual deprecation warnings.
See https://docs.gradle.org/6.5.1/userguide/command_line_interface.html#sec:command_line_warnings

BUILD SUCCESSFUL in 655ms
1 actionable task: 1 executed
bharaths-MacBook-Pro:couponapp bharaththippireddy$
```

Similarly if we want to apply for all the projects use allproject method as shown below.



```
Sublime Text File Edit Selection Find View
build.gradle
1 allprojects{
2     task printProjectName{
3         doLast(){
4             println project.name
5         }
6     }
7 }
```

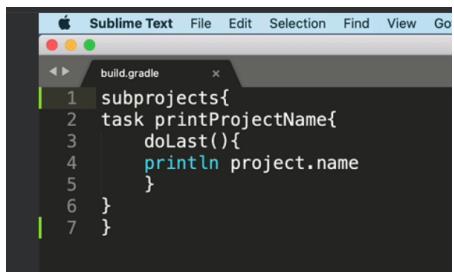
Now when i run the web module - that time we can see above task executing.

```
BUILD FAILED in 640ms
bharaths-MacBook-Pro:couponapp bharaththippireddy$ gradle :web:pPN
> Task :web:printProjectName
web
```

We get the same output for other projects also.

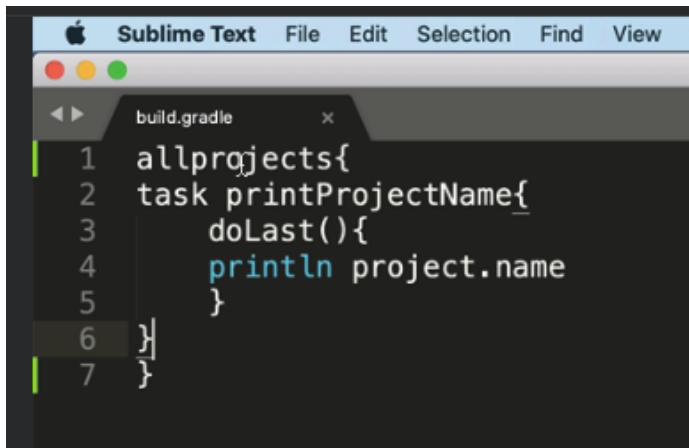
```
1 actionable task: 1 executed
bharaths-MacBook-Pro:couponapp bharaththippireddy$ gradle :services:pPN
> Task :services:printProjectName
services
```

If we want to apply only for sub projects follow the syntax. It will not execute root folder tasks

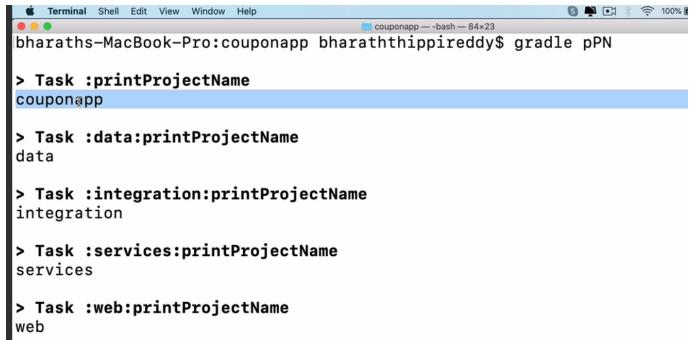


```
bharaths-MacBook-Pro:couponapp bharaththippireddy$ gradle pPN  
> Task :data:printProjectName  
data  
> Task :integration:printProjectName  
integration  
> Task :services:printProjectName  
services  
> Task :web:printProjectName  
web  
Deprecated Gradle features were used in this build, making it incompatible with Gradle 7.0.
```

If you run after changing to allprojects. It will run the root project also.



```
allprojects{  
task print projectName{  
    doLast(){  
        println project.name  
    }  
}  
}
```



```
bharaths-MacBook-Pro:couponapp bharaththippireddy$ gradle pPN  
> Task :print projectName  
couponapp  
> Task :data:print projectName  
data  
> Task :integration:print projectName  
integration  
> Task :services:print projectName  
services  
> Task :web:print projectName  
web
```

If we have common logic across all the projects - How to reuse.

```
Gradle for java developers
Sublime Text  File  Edit  Selection  Find  View  Goto  Tools  Project  Window  Help
build.gradle -- couponapp/services
1 plugins {
2     id 'application'
3 }
4 repositories {
5     mavenCentral()
6 }
7 dependencies {
8     testImplementation 'junit:junit:4.13'
9 }
10
11
12
```

Move it to main build.gradle file

```
Sublime Text  File  Edit  Selection  Find  View  Goto  Tools  Project  Window  Help
build.gradle -- couponapp
1 subprojects{
2     plugins {
3         id 'application'
4     }
5
6     repositories {
7         mavenCentral()
8     }
9
10    dependencies {
11        testImplementation 'junit:junit:4.13'
12    }
13 }
```

Syntax change for plugin

```
Sublime Text  File  Edit  Selection  Find  View  Goto  Tools  Project  Window  Help
build.gradle -- couponapp
1 subprojects{
2     apply plugin: 'application'
3
4     repositories {
5         mavenCentral()
6     }
7
8     dependencies {
9         testImplementation 'junit:junit:4.13'
10    }
11 }
```

After gradle clean we wont find any build folder

```
bharaththippireddy@bharaths-MacBook-Pro couponapp % gradle clean
Deprecated Gradle features were us
with Gradle 7.0.
Use '--warning-mode all' to show t
See https://docs.gradle.org/6.8.3/
c:command_line_warnings

BUILD SUCCESSFUL in 534ms
4 actionable tasks: 4 executed
bharaththippireddy@bharaths-MacBoo
```

Now build

```

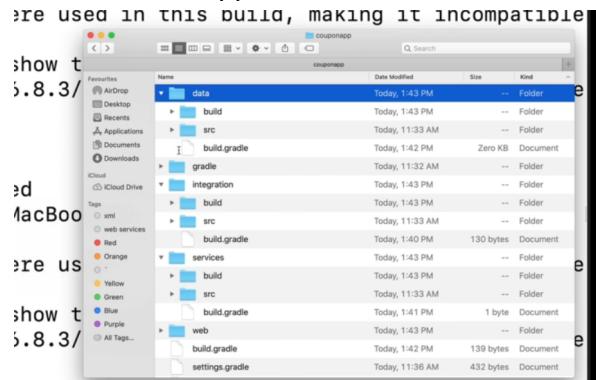
4 actionable tasks: 4 executed
bharaththippreddy@bharaths-MacBook-Pro couponapp % gradle build

Deprecated Gradle features were used in this build, making it incompatible
with Gradle 7.0.
Use '--warning-mode all' to show the individual deprecation warnings.
See https://docs.gradle.org/6.8.3/userguide/command_line_interface.html#se
c:command_line_warnings

BUILD SUCCESSFUL in 4s
7 actionable tasks: 7 executed

```

All build folder appears



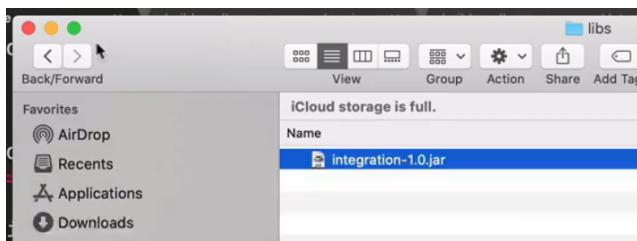
We can also apply properties for sub folders as follow. Line number 5 and 6

```

subprojects{
    apply plugin: 'java'
    version='1.0'
    group='com.bharath.gradle'
    repositories {
        mavenCentral()
    }
    dependencies {
        testImplementation 'junit:junit:4.13'
    }
}

```

See how the jar was created for the integration folder.



We have separate logic for web layer

```

1 /*
2  * This file was generated by the Gradle 'init' task.
3  *
4  * This generated file contains a sample Java project to get you started.
5  * For more details take a look at the Java Quickstart chapter in the Gradle
6  * User Manual available at https://docs.gradle.org/6.5.1/userguide/tutorial_java_projects
7 */
8
9 plugins {
10    // Apply the java plugin to add support for Java
11    id 'war'
12    id 'org.gretty' version '3.0.3'
13 }
14
15 repositories {
16    // Use jcenter for resolving dependencies.
17    // You can declare any Maven/Ivy/file repository here.
18    jcenter()
19 }
20
21 dependencies {
22    providedCompile 'javax.servlet:javax.servlet-api:3.1.0'
23    compile group: 'com.fasterxml.jackson.core', name: 'jackson-core', version: '2.11.2'
24 }
25
26 gretty.httpPort=8090

```

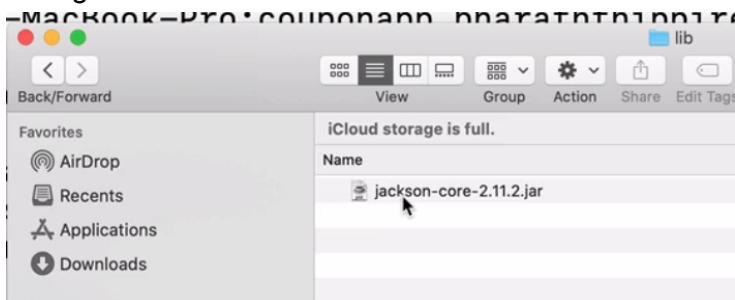
Run build command for build folder

```

Terminal Shell Edit View Window Help
baharath-MacBook-Pro:couponapp bharaththippireddy$ gradle :web:build
Deprecation Features were used in this build, making it incompatible with Gradle 7.0.
Use '--warning-mode all' to show the individual deprecation warnings.
See https://docs.gradle.org/6.5.1/userguide/command_line_interface.html#sec:command_line_warnings
BUILD SUCCESSFUL in 795ms
2 actionable tasks: 2 executed
baharath-MacBook-Pro:couponapp bharaththippireddy$ 

```

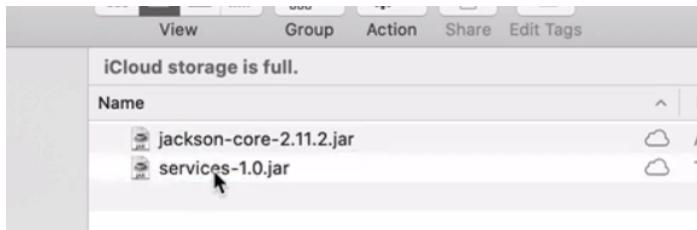
Extract the jar file and go to build folder you will find only dependencies that are part of web build.gradle.



But our web also depends on the service layer and other layers. So specify them in the main build.gradle as shown below. (dependencies between module)

```
1 subprojects{  
2     apply plugin: 'java'  
3     version = '1.0'  
4     group='com.bharath.gradle'  
5     repositories {  
6         mavenCentral()  
7     }  
8     dependencies {  
9         testImplementation 'junit:junit:4.13'  
10    }  
11 }  
12 project(':web'){  
13     dependencies{  
14         implementation project(':services')  
15     }  
16 }  
17 }
```

Now build web project again and check lib folder after extracting jar file



Now introduce some more project dependencies as shown below

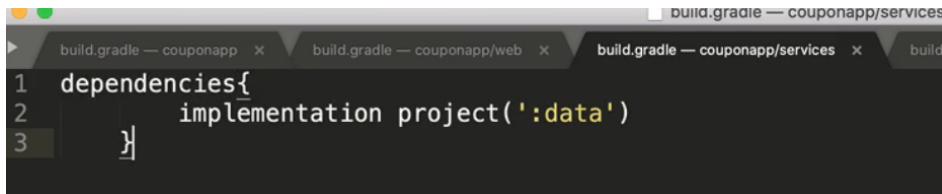
```
11 dependencies {  
12     testImplementation 'junit:junit:4.13'  
13 }  
14  
15  
16 project(':web'){  
17     dependencies{  
18         implementation project(':services')  
19     }  
20 }  
21  
22 project(':services'){  
23     dependencies{  
24         implementation project(':data')  
25     }  
26 }  
27  
28  
29  
30  
31 project(':data'){  
32     dependencies{  
33         implementation project(':integration')  
34     }  
35 }
```

Now build and check lib folder



The above syntax is for root level dependencies.

Whereas in project level when one module depends on the other module we can also specify like this. Service module depends on the data module.

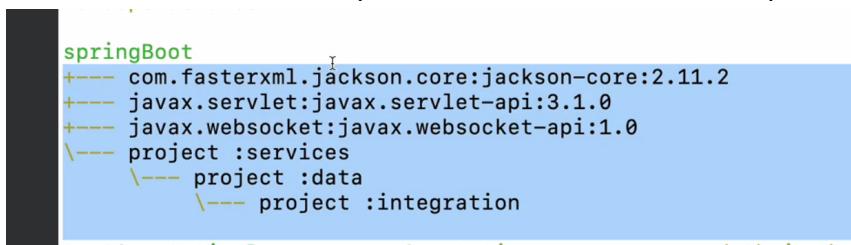


```
build.gradle — couponapp  X  build.gradle — couponapp/web  X  build.gradle — couponapp/services  X  build.gradle — couponapp/integration  X
1 dependencies{
2     implementation project(':data')
3 }
```

Show dependencies

```
BUILD SUCCESSFUL in 800ms
1 actionable task: 1 executed
bharaths-MacBook-Pro:couponapp bharaththippireddy$ gradle web:dependencies
```

It shows what are all the dependencies and its transitive dependencies as well as shown below.



```
springBoot
+--- com.fasterxml.jackson.core:jackson-core:2.11.2
+--- javax.servlet:javax.servlet-api:3.1.0
+--- javax.websocket:javax.websocket-api:1.0
\--- project :services
    \--- project :data
        \--- project :integration
```

Good job!

Question 1:
Which of the following is the correct syntax to declare child projects in a parent's settings.gradle

- include 'web';'services';'data';'integration'
- rootProject.name = 'couponapp'
- include 'web';'services';'data';'integration'

Good job!

Question 2:
Which of the following methods will apply build logic across projects including the root project

- subprojects
- projects
- someprojects
- allprojects

Good job!

Question 3:

We can control the dependencies of a child project or module from the parent project's build.gradle

- True
- False