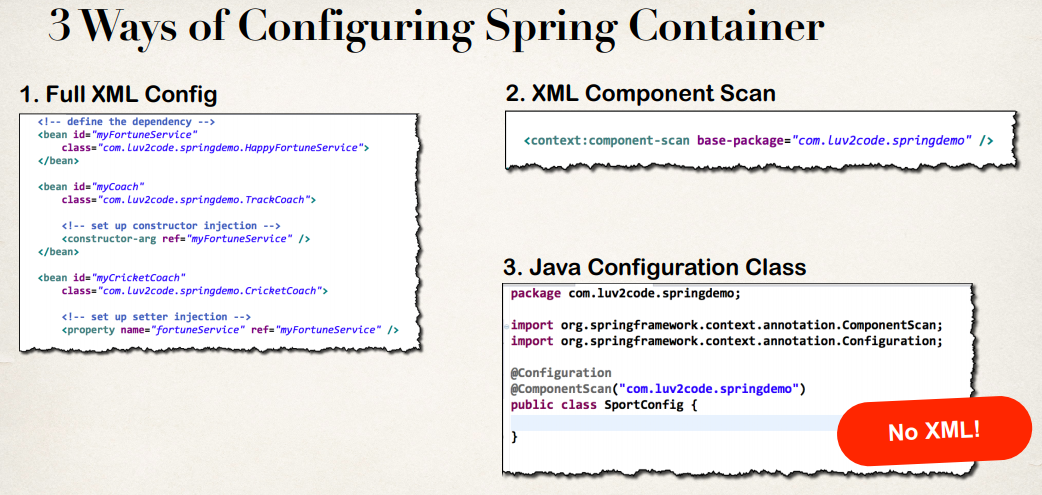
**10. Spring Configuration with Java Code**

**Java Configuration**:

Instead of configuring our container using XML, we can use Java source code to configure the container. We actually no requirement for XML. This is a pure Java configuration.

**Three ways of Configuring Spring Container**:

1. Full XML configuration
2. XML Component Scan
3. Java Configuration Scan



**Development process (Step-By-Step)**:

1. Create a Java class and annotate as @Configuration
2. Add component scanning support: @ComponentScan (optional)
3. Read Spring Java configuration class in our main application.
4. Retrieve bean from Spring container

**Step 1: Create a Java class and annotate as @Configuration**:

@Configuration

public class SportConfig {

}

**Step 2: Add component scanning support: @ComponentScan (Optional)**:

@Configuration

@ComponentScan("com.ruhul.odduu")

public class SportConfig {

}

**Step 3: Read Spring Java configuration class in our main application**:

AnnotationConfigApplicationContext context =

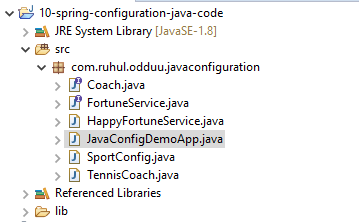
new AnnotationConfigApplicationContext(SportConfig.class);

**Step 4: Retrieve bean from Spring container**:

Coach theCoach = context.getBean("tennisCoach", Coach.class);

**Example**:

**Project Structure**:



**File: Coach.java**:

**package** com.ruhul.odduu.javaconfiguration;

**public** **interface** Coach {

**public** String getDailyWorout();

**public** String getDailyFortune();

}

**File: FortuneService.java**:

**package** com.ruhul.odduu.javaconfiguration;

**public** **interface** FortuneService {

**public** String getFortune();

}

**File: HappyFortuneService.java**:

**package** com.ruhul.odduu.javaconfiguration;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** HappyFortuneService **implements** FortuneService {

@Override

**public** String getFortune() {

**return** "Today is your lucky day!!!";

}

}

**File: SportConfig.java**:

**package** com.ruhul.odduu.javaconfiguration;

**import** org.springframework.context.annotation.ComponentScan;

**import** org.springframework.context.annotation.Configuration;

@Configuration

@ComponentScan("com.ruhul.odduu.javaconfiguration")

**public** **class** SportConfig {

}

**File: TennisCoach.java**:

**package** com.ruhul.odduu.javaconfiguration;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.context.annotation.Scope;

**import** org.springframework.stereotype.Component;

@Component

@Scope("prototype")

**public** **class** TennisCoach **implements** Coach {

// field injection

@Autowired

**private** FortuneService fortuneService;

// default constructor

**public** TennisCoach() {

System.***out***.println(">> TennisCoach: inside default constructor");

}

@Override

**public** String getDailyWorout() {

**return** "Practice your backhand volley";

}

@Override

**public** String getDailyFortune() {

**return** fortuneService.getFortune();

}

}

**File: JavaConfigDemoApp.java**:

**package** com.ruhul.odduu.javaconfiguration;

**import** org.springframework.context.annotation.AnnotationConfigApplicationContext;

**public** **class** JavaConfigDemoApp {

**public** **static** **void** main(String[] args) {

// read spring configuration java class

AnnotationConfigApplicationContext context = **new** AnnotationConfigApplicationContext(SportConfig.**class**);

// retrieve bean from spring container

Coach theCoach = context.getBean("tennisCoach", Coach.**class**);

// call a method on the bean

System.***out***.println(theCoach.getDailyWorout());

// call a method to the daily fortune

System.***out***.println(theCoach.getDailyFortune());

// close the context

context.close();

}

}

/\*

**Output**:

>> TennisCoach: inside default constructor

Practice your backhand volley

Today is your lucky day!!!

\*/

10. Spring Configuration with Java Code