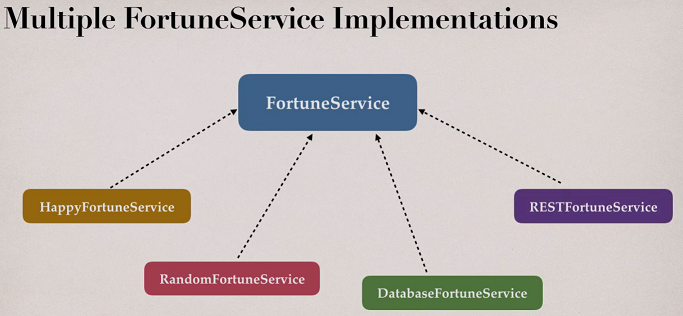
**Qualifiers for Dependency Injection**

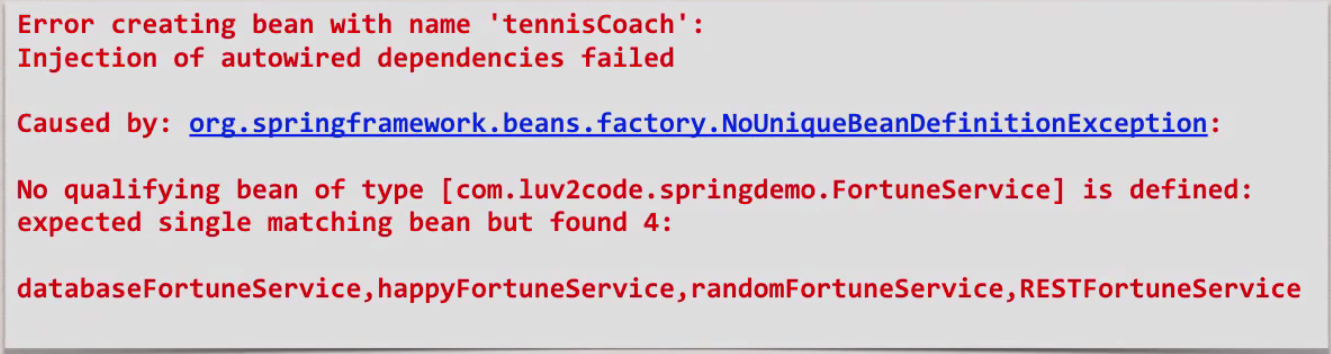
**Qualifier**:

There may be a situation when we create more than one bean of the same type and want to wire only one of them with a property. In such cases, we can use the @Qualifier annotation along with @Autowired to remove the confusion by specifying which exact bean will be wired.

Here we have a project "com.ruhul.odduu.qualifiers" and in this project we have an interface name "FortuneService" and multiple class like "HappyFortuneService", "RandomFortuneService", "RestFulFortuneService", and "DatabaseFortuneService" implements the "FortuneService" interface.



When we run the project, Spring don’t know which exact bean will be wired. We will get RuntimeException saying



We have to tell Spring which bean we have to use.

We can apply @Qualifire annotation to

1. Constructor injection
2. Setter injection
3. Field injection

**Example**:

**File: Coach.java**:

**package** com.ruhul.odduu.qualifiers;

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

**public** String getDailyWorout();

**public** String getDailyFortune();

}

**File: FortuneService.java**:

**package** com.ruhul.odduu.qualifiers;

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

**public** String getFortune();

}

**File: RandomFortuneService.java**:

**package** com.ruhul.odduu.qualifiers;

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

@Component

**public** **class** RandomFortuneService **implements** FortuneService {

@Override

**public** String getFortune() {

**return** "Hello RandomFortuneService";

}

}

**File: DatabaseFortuneService.java**:

**package** com.ruhul.odduu.qualifiers;

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

@Component

**public** **class** DatabaseFortuneService **implements** FortuneService {

@Override

**public** String getFortune() {

**return** "Hello DatabaseFortuneService";

}

}

**File: RestFulFortuneService.java**:

**package** com.ruhul.odduu.qualifiers;

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

@Component

**public** **class** RestFulFortuneService **implements** FortuneService {

@Override

**public** String getFortune() {

**return** "Hello RestFulFortuneService";

}

}

**File: HappyFortuneService.java**:

**package** com.ruhul.odduu.qualifiers;

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

@Component

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

@Override

**public** String getFortune() {

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

}

}

**File: TennisCoach.java**:

**package** com.ruhul.odduu.qualifiers;

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

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

@Component

**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: AnnotationDemoApp.java**:

**package** com.ruhul.odduu.qualifiers;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** AnnotationDemoApp {

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

// read spring congig file

ClassPathXmlApplicationContext context = **new** ClassPathXmlApplicationContext("applicationContextQualifiers.xml");

// get the bean from container

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

// call a method on the bean

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

// call method to get the daily fortune

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

// close the context

context.close();

}

}

**File: applicationContextQualifiers.xml**:

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xmlns:context=*"http://www.springframework.org/schema/context"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context.xsd"*>

<!-- add entry to enable component scanning -->

<context:component-scan

base-package=*"com.ruhul.odduu.qualifiers"* />

</beans>

/\*

**Output**:

>> TennisCoach: inside default constructor

Mar 10, 2019 2:29:24 PM org.springframework.context.support.AbstractApplicationContext refresh

WARNING: Exception encountered during context initialization - cancelling refresh attempt: org.springframework.beans.factory.UnsatisfiedDependencyException: Error creating bean with name 'tennisCoach': Unsatisfied dependency expressed through field 'fortuneService'; nested exception is org.springframework.beans.factory.NoUniqueBeanDefinitionException: No qualifying bean of type 'com.ruhul.odduu.qualifiers.FortuneService' available: expected single matching bean but found 4: databaseFortuneService,happyFortuneService,randomFortuneService,restFulFortuneServic

\*/

We have to tell Spring which bean we have to use.

Now add qualifier in TennisCoach.java class. Add "@Qualifier("happyFortuneService")" in the TennisCoach.java class on the top of "private FortuneService fortuneService;". Here "happyFortuneService" is the bean name to inject. This is default bean is the name is same as class but the first letter is lower case.

**File: TennisCoach.java**:

**package** com.ruhul.odduu.qualifiers;

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

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

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

@Component

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

// field injection

@Autowired

@Qualifier("happyFortuneService")

**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();

}

}

/\*

**Output**:

>> TennisCoach: inside default constructor

Practice your backhand volley

Today is your lucky day!!!

\*/

Now we use the "RandomFortuneService" class as bean. Let’s implement the class "RandomFortuneService" for generate random value.

**File: RandomFortuneService.class**:

**package** com.ruhul.odduu.qualifiers;

**import** java.util.Random;

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

@Component

**public** **class** RandomFortuneService **implements** FortuneService {

// create an array of strings

**private** String[] data = {

"Odduu Health Care Service",

"Odduu VDI Project",

"Odduu WASHA Project",

"Odduu Tructor Project"

};

//create a random number generator

**private** Random myRandom = **new** Random();

@Override

**public** String getFortune() {

// pick a random string from the array

**int** index = myRandom.nextInt(data.length);

String theFortune = data[index];

**return** theFortune;

}

}

**File: TennisCoach.java**:

**package** com.ruhul.odduu.qualifiers;

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

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

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

@Component

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

// field injection

@Autowired

@Qualifier("randomFortuneService")

**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: AnnotationDemoApp.java**:

**package** com.ruhul.odduu.qualifiers;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** AnnotationDemoApp {

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

// read spring config file

ClassPathXmlApplicationContext context = **new** ClassPathXmlApplicationContext("applicationContextQualifiers.xml");

// get the bean from container

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

// call a method on the bean

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

// call method to get the daily fortune

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

// close the context

context.close();

}

}

/\*

**Output**:

>> TennisCoach: inside default constructor

Practice your backhand volley

Odduu WASHA Project

\*/

If we run the project more and more the value of “RandomFortuneService” change every time.

Qualifiers for Dependency Injection