أكاديمية طويق
Tuwaiq Academy

Exercise (Spring Container)

What is the output?

Q1:

```
1 usage
@SpringBootApplication
public class SpringPollApplication {

    public static void main(String[] args) { SpringApplication.run(SpringPollApplication.class, args); }


    @Bean
    public String getMessage1(){
        System.out.println("hey from message1");
        return "1";
    }
}
```

A1:

{hey from message1}.

because there is only one method that doesn't depend on anything.

Q2:

```
1 usage
@SpringBootApplication
public class SpringPollApplication {

    public static void main(String[] args) { SpringApplication.run(SpringPollApplication.class, args); }


    @Bean
    @Qualifier("1")
    public String getMessage1(){
        System.out.println("hey from message1");
        return "1";
    }
    @Bean
    public String getMessage2(@Qualifier("1") String data ){
        System.out.println("hey from message2");
        return data ;
    }
```

A2:

{hey from message1, hey from message2}.

Because the @Qualifier has been used here getMessage2 depened on getMessage1, so getMessage2 can't run before getMessage1.

Q3:

```java
@Bean
@Qualifier("1")
public String getMessage1(){
    System.out.println("hey from message1");
    return "1";
}

@Bean
@Qualifier("2")
public String getMessage2(@Qualifier("3") String data ){
    System.out.println("hey from message2");
    return data;
}

@Bean
@Qualifier("3")
public String getMessage3(){
    System.out.println("hey from message3");
    return "3" ;
}
```

A3:
{hey from message3, hey from message2, hey from message1}.
{hey from message1, hey from message3, hey from message2}.

Because the symbol used in getMessage2's @Qualifier is 3 which will make getMessage2 depends on getMessage3 but for getMessage1 we can't be sure if it will run at the beginning or at the end.

Q4:

```java
    @Bean
    @Qualifier("1")
    public String getMessage1(){
        System.out.println("hey from message1");
        return "1";
    }

    @Bean
    @Qualifier("2")
    public String getMessage2(@Qualifier("3") String data ){
        System.out.println("hey from message2");
        return data;
    }

    @Bean
    @Qualifier("3")
    public String getMessage3(){
        System.out.println("hey from message3");
        return "3" ;
    }
```

```java
@Component
public class MainController {

    1 usage
    String data;

    public MainController(@Qualifier("1") String data){
        this.data=data;
        System.out.println("hey from Main controller");
    }

}
```

A4:

{hey from message1, hey from Main controller, hey from message3, hey from message2}.

Because of Spring's injection order the constructor should be invoked first however the constructor has a qualifier notation with a symbol "1" which is located in getMessage1 that's why getMessage1 will be invoked first then the constructor will follow and getMessage3 because it has the qualifier symbol for getMessage2 then getMessage2 will be invoked at the end.

Q5:

```java
    @Bean
    @Qualifier("1")
    public String getMessage1(MainController mainController){
        System.out.println("hey from message1");
        return "1";
    }

    @Bean
    @Qualifier("2")
    public String getMessage2(@Qualifier("3") String data ){
        System.out.println("hey from message2");
        return data;
    }

    @Bean
    @Qualifier("3")
    public String getMessage3(){
        System.out.println("hey from message3");
        return "3" ;
    }
```

```java
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Component;

1 usage
@Component
public class MainController {

    1 usage
    String data;

    public MainController(@Qualifier("2") String data){
        this.data=data;
        System.out.println("hey from Main controller");
    }

}
```

A5:
{hey from message3, hey from message2, hey from Main controller, hey from message1}.

Because of Spring's injection order the constructor should be invoked first however the constructor has a qualifier notation with a symbol "2" which is located in getMessage2 but even getMessage2 has a qualifier notation in getMessage3 and getMessage1 has the constructor as an argument that's why getMessage3 will be invoked first then getMessage2 will follow and then the constructor and finally getMessage1 will be invoked.