Spring Container

Exercise

WALEED MOHAMMED ALTHUBYANI

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

1)

Hey from message1

getMessage1(). Because it doesn't have a parameter to wait for. And there isn't any other method that could run beside it.

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

1)

hey from message1
hey from message2

getMessage1() then getMessage2(@Qualifier("1") String data). Because getMessage1() doesn't have any requirements but getMessage2(@Qualifier("1") String data) needs a String and it most come from 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

1)

hey from message1
hey from message3
hey from message2

getMessage1() then getMessage3() then getMessage2(@Qualifier("3") String data). Because there is no priority between getMessage1() and getMessage3() but getMessage2(@Qualifier("3") String data) must come after getMessage3().

2)

hey from message3
hey from message1
hey from message2

getMessage3() then getMessage1() then getMessage2(@Qualifier("3") String data). Because there is no priority between getMessage1() and getMessage3() but getMessage2(@Qualifier("3") String data) must come after getMessage3().

3)

hey from message3
hey from message2
hey from message1

getMessage3() then getMessage2(@Qualifier("3") String data) then getMessage1(). Because after getMessage3() get executed there will be no priority between getMessage2(@Qualifier("3") String data) and getMessage1().

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

1)

hey from message1
hey from Main controller
hey from message3
hey from message2

getMessage1() then MainController(@Qualifier("1") String data) then getMessage3() then getMessage2(@Qualifier("3") String data). Since spring boot create an object of each class in the beginning it will try to create an object of class MainController but because the constructor needs a String data that must come from getMessage1() it will be executed first followed by the constructor after that Message3() will be executed because it doesn't have any requirements then getMessage2(@Qualifier("3") String data) since it needs the outcome from getMessage3() to get executed.

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

1)

hey from message3
hey from message2
hey from Main controller
hey from message1

getMessage3() then getMessage2(@Qualifier("3") String data) then MainController(@Qualifier("2") String data) then getMessage1(). Since spring boot will try to create an instance of class MainController first it will need to get the outcome from getMessage2(@Qualifier("3") String data). But because it needs the outcome from getMessage3(), getMessage3() will be executed first. Then its outcome will go to getMessage2(@Qualifier("3") String data) and it will get executed. Its result will go to the constructor MainController(@Qualifier("2") String data). After executing it and creating an instance of the class MainController the remaining method getMessage1() will get executed.