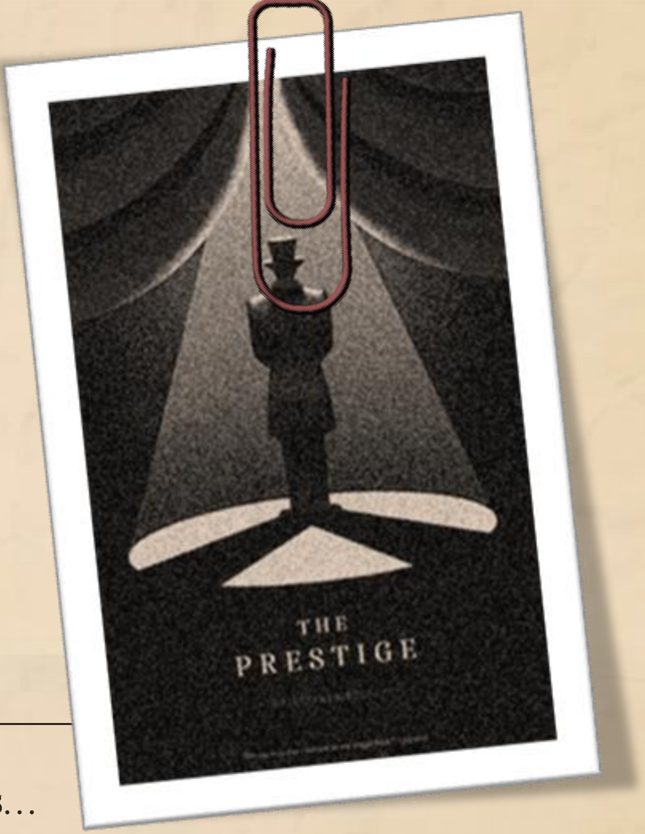
The Great Springhoff's masterpiece



No one cares about the man in the box, the man who disappears...

Case description

It’s been crowded the last few days in the city. We had a summer festival opening and it was attracting everyone’s attention. Couples in love, families with children, tourists who wandered in by chance - it seemed that everyone could find something interesting to do

I came specially to see The Great Springhoff. Since childhood I love everything connected with riddles and secrets, and the famous magician’s performance was something I just couldn’t miss

Posters were promising something intriguing: the magician was going to get into a box, vanish from it in front of all watchers and then appear back again

I was wondering if I’ll be able to understand, how he is doing his trick

Performance description on the poster

```java
public static void main(String[] args) {
    System.out.println("Ladies and Gents, the main event of this evening is about to happen");

    // magician enters the box
    System.out.println("1st: Let’s verify that The Great Springhoff is in the box");
    ConfigurableApplicationContext context = SpringApplication.run(Main.class, args);

    // dramatic pause
    System.out.println("2nd: Now, after just few seconds, we’ll see The Great  Springhoff is no more here");
    Watchable stage = context.getBean(Watchable.class);
    stage.watchVeryCarefully();

    // dramatic pause
    System.out.println("3rd: And now, just after few seconds again, let's open the box for the last time");
    context.close();

    // ovations
    System.out.println("That's all folks!");
}
```

So, the main trick was going to happen this way: the magician will enter the box, then he was going to disappear, and then appear again

We’ll be able to see what’s inside the box 3 times

Stage description

When I entered the hall where the performance was going to happen, I immediately looked on the stage to see how everything was organized

There was a box and only the festival workers had access to work with it

In other hand, the stage itself was completely watchable from all sides and all visitors could look on what's happening

```java
@Component
@RequiredArgsConstructor
public class Stage implements Watchable {

   private final Box box;

   @PostConstruct
   @PreDestroy
   @Attention
   public final void watchVeryCarefully() {
      System.out.println("Looking in the box: " + box);
   }

}
```

```java
public interface Watchable {
   void watchVeryCarefully();
}
```

```java
@Component
@ToString
public class Box {
   private final String magician = "Springhoff";
}
```

Attention on the performance

So, I decided to keep a high attention to all openings of the box

Whatever will happen, it should be somehow related to these openings (some hidden mechanism, reflective mirrors or something similar)

Right after I see anything suspicious, I'll understand how the trick is done

```java
@Component
@RequiredArgsConstructor
public class Stage implements Watchable {

    private final Box box;

    @PostConstruct
    @PreDestroy
    @Attention
    public final void watchVeryCarefully() {
        System.out.println("Looking in the box: " + box);
    }

}
```

```java
public @interface Attention {
}
```

```java
@Aspect
@Component
public class AttentionAspect {
    @After("@annotation(detectivecases.case3.Attention)")
    public void sawSomethingSuspicious() {
        System.out.println(“Now I understand how it works!");
    }
}
```

My expectations

```java
public static void main(String[] args) {
    System.out.println("Ladies and Gents, the main event of this evening is about to happen");
    // magician enters the box
    System.out.println("1st: Let's verify that The Great Springhoff is in the box");
    ConfigurableApplicationContext context = SpringApplication.run(Main.class, args);
    // dramatic pause
    System.out.println("2nd: Now, after just few seconds, we'll see The Great Springhoff is no more here");
    Watchable stage = context.getBean(Watchable.class);
    stage.watchVeryCarefully();
    // dramatic pause
    System.out.println("3rd: And now, just after few seconds again, let's open the box for the last time");
    context.close();
    // ovations
    System.out.println("That's all folks!");
}
```

```java
@Component
@RequiredArgsConstructor
public class Stage implements Watchable {

    private final Box box;

    @PostConstruct
    @PreDestroy
    @Attention
    public final void watchVeryCarefully() {
        System.out.println("Looking in the box: " + box);
    }

}
```

So, there will be 3 times to look inside the box:

1. Right after Springhoff gets into it (**@PostConstruct**)
2. In the middle of the trick (by calling **watchVeryCarefully** method)
3. In the very end to see the magician appearing back (**@PreDestroy**)
4. And each one opening will have my full attention

The show starts...

I was pretty sure The Great Springhoff couldn't do something that I wouldn't see. When the trick started, I starred my eyes on the stage:

```
Ladies and Gents, the main event of this evening is about to happen


1st: Let's see that The Great Springhoff is already in the box
Looking in the box: Box(magician=Springhoff)

2nd: Now, after just few seconds, let's open the box again
Looking in the box: null

3rd: And now, just after few seconds again, let's open the box for the last time
Looking in the box: Box(magician=Springhoff)


That's all folks!
```

was expecting to see "Now I understand how it works!"

I was really surprised!

1st – the magician vanished indeed, and I didn't get a bit of an idea how he did it

2nd and more important – the box was opened 3 times, but I didn't see anything at all

I was tricked, but how did it happen?

How did it happen?

You can find the project on:
https://github.com/WieRuindl/detective-cases/tree/master/case3

It contains all the code described on the previous slides

You're very welcome to take the project and try to figure out the case yourself ☺

There will be a **lunch talk on 16 May at 14:00 BG time** where I will explain step by step what and how exactly happened