Spring Professional Exam Tutorial v5.0

Question 07

Question 07 - If shown pointcut expressions, would you understand them? For example, what would be the correct pointcut expression to match both getter and setter methods?

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator types supported by Spring AOP:

- `execution`
- `within`
- `args`
- `bean`
- `this`
- `target`
- `@annotation`
- `@args`
- `@within`
- `@target`

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `execution` - matches method execution

General Form:

```
execution([visibility modifiers] [return type] [package].[class].[method]([arguments]) [throws exceptions]
```

Description:

- `[visibility modifiers]` - public/protected, if omitted all are matched, can be used with negation, for example `!protected`
- `[return type]` - void, primitive or Object type, cannot be omitted, can be used with wildcard `*`, can be used with negation, for example `!int`
- `[package]` - package in which class is located, may be omitted if class is located within same package as aspect, wildcard `*` may be used to match all packages, wildcard `..` may be used to match all sub-packages
- `[class]` - Class name to match against, may be omitted, may be used with `*` wildcard, matches subclasses of the class as well
- `[method]` - Name of the method, whole or partial method name can be used with `*` wildcard
- `[arguments]` - May be empty to match methods without any arguments, may be used with wildcard `..` to match zero or more arguments, may be used with wildcard * to match all types of specific argument, may be used with ! Negation
- `[throws exceptions]` - Match method that throws exceptions from given list, can be used with negation !

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `within` - matches execution within specified class/classes, optionally you can specify class package

General Form:

```
within([package].[class])
```

Description:

- `[package]` - package where class is located, may be used with `..` wildcard (includes all sub-packages) or with `*` wildcard, may be omitted
- `[class]` - class against which match should happen, may be used with * wildcard

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `args` - matches execution of method with matching arguments

General Form:

```
args([parameter_type1, parameter_type2, ..., parameter_typeN])
```

Description:

- `[parameter_typeN]` - simple or object type, may be `*` to indicate one parameter of any type, may be `..` to indicate zero or more arguments, you can specify type with the package

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator – `bean` – matches execution of method with matching Spring Bean Name

General Form:

```
bean([beanName])
```

Description:

- `[beanName]` – name of the Spring Bean (automatically generated by framework, or set manually),

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `this` - matches execution against type of proxy that was generated by Spring AOP

General Form:

```
this([type])
```

Description:

- `[type]` - type of the proxy, matches if generated proxy is of specified type

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `target` - matches execution against type of the target object invoked by proxy

General Form:

```
target([type])
```

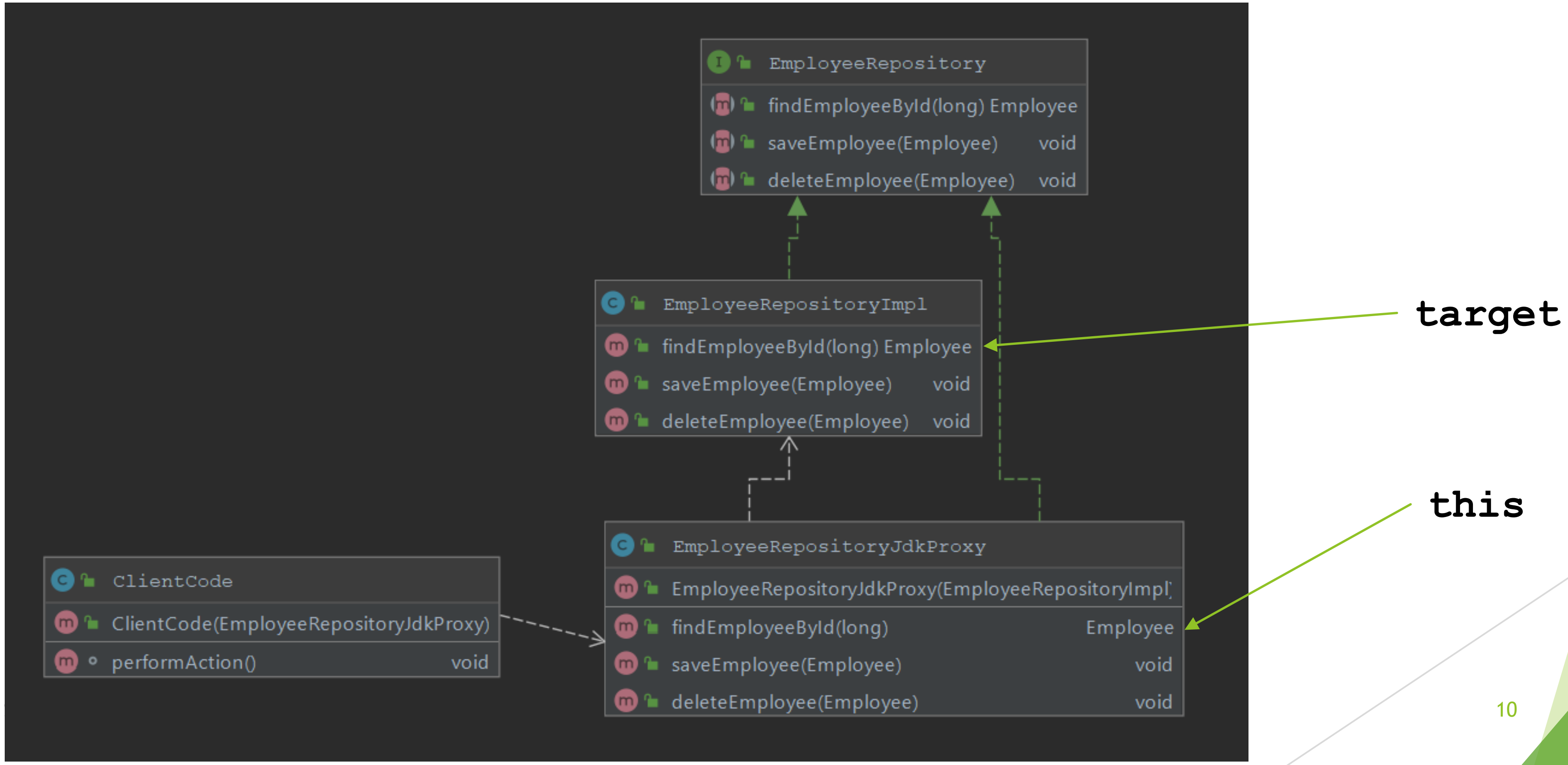
Description:

- `[type]` - type of the target object invoked by proxy, matches if target object is of specified type

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `this, target`

JDK Proxy

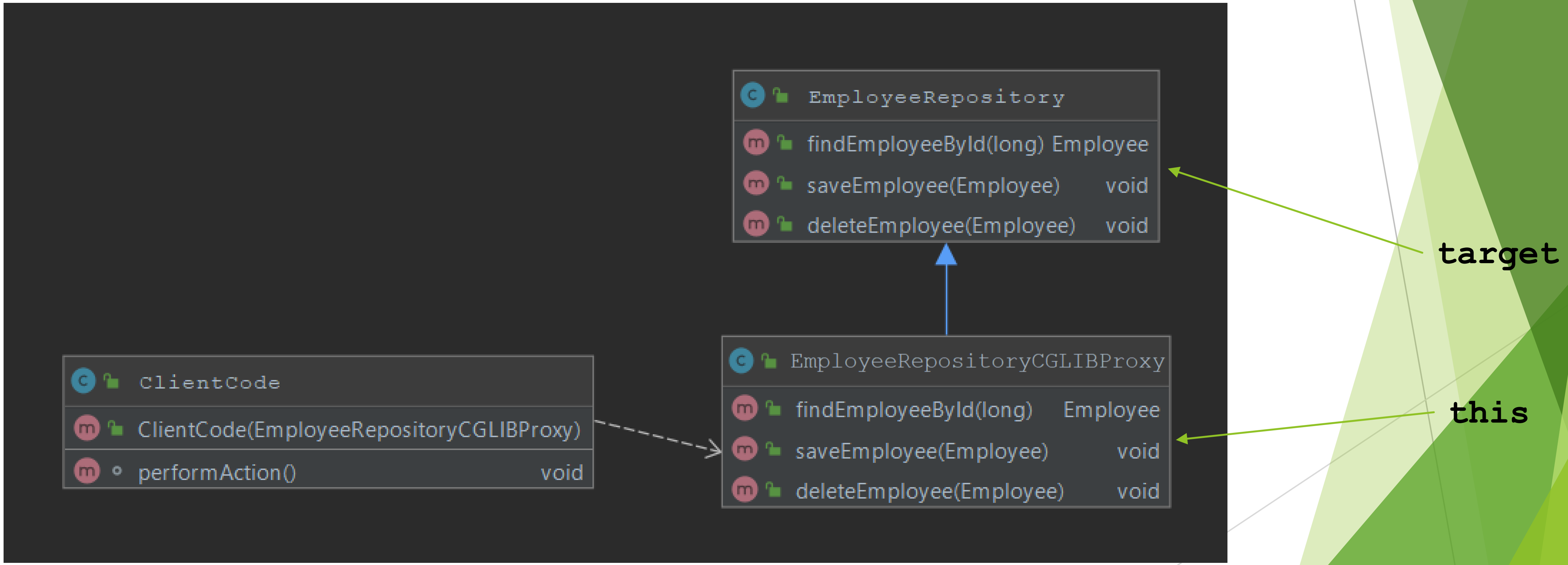


EmployeeRepository
findEmployeeById(long) Employee
saveEmployee(Employee) void
deleteEmployee(Employee) void
EmployeeRepositoryImpl
findEmployeeById(long) Employee
saveEmployee(Employee) void
deleteEmployee(Employee) void
EmployeeRepositoryJdkProxy
EmployeeRepositoryJdkProxy(EmployeeRepositoryImpl
findEmployeeById(long) Employee
saveEmployee(Employee) void
deleteEmployee(Employee) void
ClientCode
ClientCode(EmployeeRepositoryJdkProxy)
performAction() void
target
this

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `this, target`

CGLIB Proxy


EmployeeRepository
findEmployeeById(long) Employee
saveEmployee(Employee) void
deleteEmployee(Employee) void
target
ClientCode
ClientCode(EmployeeRepositoryCGLIBProxy)
performAction() void
EmployeeRepositoryCGLIBProxy
findEmployeeById(long) Employee
saveEmployee(Employee) void
deleteEmployee(Employee) void
this

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `@annotation` - matches method execution annotated with specified annotation

General Form:

```
@annotation([annotation_type])
```

Description:

- `[annotation_type]` - type of annotation used to annotated method which should match pointcut expression

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `@args` - matches method execution with argument, which types (classes) are annotated with specified annotation type, note that class should be annotated, not the argument of method itself

General Form:

```
@args([annotation_type])
```

Description:

- `[annotation_type]` - type of annotation used on top of class, which represents type of argument

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `@within` - matches method executions inside classes annotated with specified annotation

General Form:

```
@within([annotation_type])
```

Description:

- `[annotation_type]` - type of annotation used on top of class, inside which method execution should be matched

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut designator - `@target` - matches method executions inside proxied target class that is annotated with specific annotation

General Form:

```
@target([annotation_type])
```

Description:

- `[annotation_type]` - type of annotation used on top of proxied class, inside which method execution should be matched

Question 07 - If shown pointcut expressions, would you understand them? […]

Pointcut expressions can be combined together with usages of logical operators:

- ! – negation
- || - logical or
- && - logical and

Question 07 – [...] For example, what would be the correct pointcut expression to match both getter and setter methods?

To match getters and setter, `execution` Pointcut Expression can be used together with ability to combine them with logical operators.

Expression that will match getters and setter can look like this:

```
execution(* com.beans.EmployeeBean.get*()) ||
   execution(* com.beans.EmployeeBean.set*(*))
```