Spring AOP Use cases and Terminologies

**Use cases:**

* logging-related
* security checks
* transaction management
* tweaking of a legacy application
* Validation
* Exception Handling
* Performance measurement

**Aspect:** An Aspect is a class that implements concerns that cut across different classes such as logging. It is just a name.

**Advice:** Action taken by aspect at particular join point. For example: Before execution of getEmployeeName() method, put logging. So here, we are using before advice.

**Join point:** It is a point in execution of program such as execution of method. In Spring AOP, a join point always represents a method execution.

**Pointcut:** Pointcut is an expression that decides execution of advice at matched joint point. Spring uses the AspectJ pointcut expression language by default.

**Advices:**

**@Before:**  it executes before a join point.

**@After:** @After will be called regardless of whether the target threw an exception or not.

**@AfterReturning:**  it executes after a joint point completes normally.

**@AfterThrowing:**  it executes if method exits by throwing an exception.

**@Around:** It executes before and after a join point.