AOP – аспекта ориентированное программирование, которое позволяет, добавлять сквозной функционал, но при этом не загромождать весь код.

К сквозному функционалу может относиться: Обработка exception, работа с транзакциями, кеширование, добавление логов и т.д.

AOP – состоит из 4 основных понятий:

1. Aspect – Что-то типа базового класса, в котором описаны Join point, pointcut, advice
2. Join point – эта некая точка в нашем методе, именно в эту точку мы хотим добавить сквозной функционал.
3. Advice – сам сквозной функционал. Advice находится между AOP Proxy и Нашим настоящим объектом (UserServise, UserController).
4. Pointcut – это предикат, который возвращает true or false. Он как бы говорит, стоит ли добавлять сквозной функционал в joint point (точка соединения)

**Всего существует два подхода реализации AOP в java, это:**

AspectJ (Библиотека) – добавление сквозного функционала происходит во время компиляции (compile time).

Spring AOP - добавление сквозного функционала происходит во время выполнения (runtime). Ограничение Spring AOP – мы можем добавить сквозной функционал, только для методов, используя **dynamic proxy or cglib proxy.**

**Использование сквозного функционала на уровне метода, покрывает почти все случаи.**

**Spring AOP тянется транзитивной зависимостью вместе с Spring boot.**