#Задача «Conditional приложение»

**Описание**

Как часто бывает в промышленной разработка, мы хотим в локальном или dev-окружении использовать упрощённые варианты наших сервисов.

Поэтому сегодня вы напишете приложение на Spring Boot, в котором в зависимости от параметров будут вызываться разные сервисы.

Для работы нужно подготовить несколько классов.

**Шаг 0.** Создайте Spring Boot приложение, как было сделано на лекции.

**Шаг 1.** Вам нужен интерфейс, реализации которого вы будете вызывать в зависимости от параметров вашего приложения:

public interface SystemProfile {

String getProfile();

}

Также нужны две реализации. Первая:

public class DevProfile implements SystemProfile {

@Override

public String getProfile() {

return "Current profile is dev";

}

}

И вторая:

public class ProductionProfile implements SystemProfile {

@Override

public String getProfile() {

return "Current profile is production";

}

}

**Шаг 2.** Вам нужно написать JavaConfig, в котором вы объявите бины классов DevProfile и ProductionProfile примерно так:

@Bean

public SystemProfile devProfile() {

return new DevProfile();

}

@Bean

public SystemProfile prodProfile() {

return new ProductionProfile();

}

**Шаг 3.** Дальше вам нужно создать application.properties в папке resources, если его нет, и добавить туда свой пользовательский параметр netology.profile.dev = true. Обратите внимание, что сейчас ваш параметр имеет значение true. В зависимости от значения параметра, вы и будете использовать ту или иную реализацию интерфейса SystemProfile.

**Шаг 4.** Теперь вам нужно использовать одну из аннотаций @Conditional и поместить её на бины в вашем JavaConfig. Советую использовать @ConditionalOnProperty, и, в зависимости от значения netology.profile.dev, будет создаваться бин интерфейса SystemProfile той или иной реализации.

**Шаг 5.** Чтобы проверить работоспособность логики, создайте контроллер, который будет возвращать вам значения из вашего сервиса SystemProfile:

@RestController

@RequestMapping("/")

public class ProfileController {

private SystemProfile profile;

public ProfileController(SystemProfile profile) {

this.profile = profile;

}

@GetMapping("profile")

public String getProfile() {

return profile.getProfile();

}

}