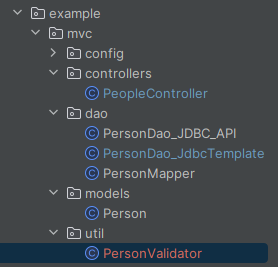
**Spring Validator**

Если случае простых валидаций мы используем валидацию полей **@Valid, @NotEmpty, @Size, @Min** и т.д, то в случае если мы например хотим проверять более сложные валидации, например обращаться к базе данных и проверять налчичие каких нибудь значение в ней, то нам нужно использовать другой механизм — **Spring Validator.**

**Работа с валидатором:**

**1.** Создаем пакет **util**. В этом паете обычно лежат вспомогательные классы, которые нам нужны в приложении. Классы валидаторы обычно лежат в этом пакете.



**2.** Создаем новый класс **PersonValidator**.

Для каждой сущности у нас обычно свой валидатор.

Чтобы этот класс являлся валидатором, нам необходимо реализовать интерфейс **Validator (Org.SprinFramework.validation)** и реализовать два метда – ***supports()*** и ***validate()***.

* В методе ***supports() -*** нам необходиом дать понять Spring`у к какому классу этот валидотор относиться тоесть на объекте какого класса этот валидатор можно использовать.
* Метод ***validate() -*** будет вызываться в контроллере.

В этом методе производим нужную нам проверку. В случае если проверка не будет пройдена в **errors** в метод ***rejectValue()*** качестве аргументов добавляем:

**1.** Название поля на котором у нас произошла ошибка, тоесть какое поле объекта у нас вызвало ошибку.

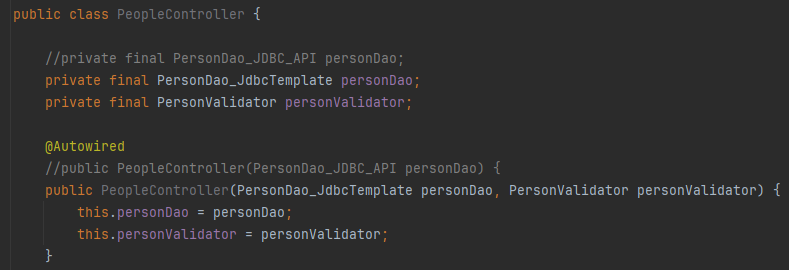
**2.** Код ошибки, указываем пустую строку.

**3.** Сообщение об ошибке.



**3.** Чтобы использвать наш **validator** его нужно:

1. Внедрить в наш контроллер с помощью **Spring**`а.



1. Вызвать метод **v*alidate()*** на нашем внедренном **validator`**e в том методе где мы хотим производить проверку полей. В качестве первого аргумента нужно передать объект который пришел с формы, а в качестве второго аргумента передаем **bindingResult,** в него будут складыватсья ошибки от аннотации **@Valid** (если она есть) а так же наши ошибки которые были пойманы с помощью **validator`a,** тоесть в **bindingResult** храним все ошибки, со всех валидаций.

