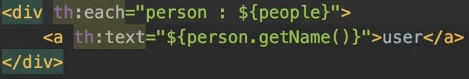
**Thymeleaf команды**

**1. th:text –** атрибут, который позволяет вставить текст.



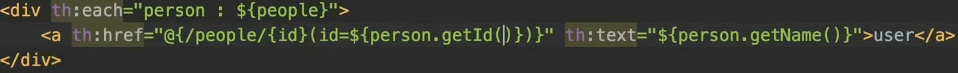
При обращении к ключу **message** с помощью такого синтаксиса **${}** Thymeleaf увидит этот код, обратиться к моделе, получит значение по ключу и вставить значение в это пространство тега <p>.

**2. th:each –** атрибут, который позволяет итерироваться по коллекциям.

  
  
  
  
people – название коллекции.  
person – переменная которая указывает на каждую из элементов коллекции.

В th:text вызываем геттер, который есть у класса person.

**3. th:href** – атрибут, который позволяет указать ссылку на которую будет осуществляться переход.

В этом коде:

1. мы получаем *id* каждого *person* с помощью геттера *getId()*

2. кладем во временную переменную полученный *id*

3. врменную переменную используем для формирования динамического адреса на который будет переходить пользователь при нажатии на эту ссылку.

**4. HTML формы**

У формы есть атрибту *method* и атрибут *action.*

***method -*** пишется тот HTTP метод который будет использоваться при отправке этой формы.

В атрибуте ***action***пишется тот адрес на который будут отправляться данные с этой формы.

***Input*** *–* это тег который создает поле для ввода.

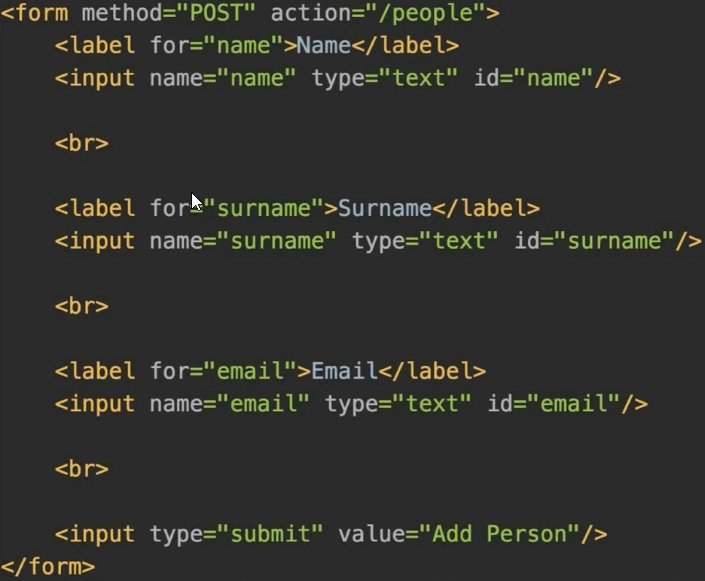
На скриншоте видно, что первое поле **текстовое** и оно имеет имя *name.* Когда мы отправляем эту форму на контроллер, в контроллере мы можем по этим названияем получить доступ к значениям каждого из полей.

У ***input*** так есть другой тип ввода — это ***submit.***

При нажатии на эту кнопку все данные с этой формы отправляются POST запросом на адрес «/people”. Вот то тело POST запроса которое придет: name=ИМЯ&surname=ФАМИЛИЯ&email=EMAI L

Соответственно эти параметры мы можем считать на контроллере.

***label* -** эта та надпись которая даёт понять пользователю, что вводить в это поле.



**HTML формы в Thymeleaf**

HTML формы в Thymeleaf очень похожи на просто формы в HTML коде, но есть некоторые функции:

1. HTML формы в Thymeleaf принимают на вход объект для которого эта форма создана. Тоесть если мы хотим создать форму для человека мы должны в контроллере который будет отдавать шаблон с этой формы нужно в модель поместить нового пустого человека. На скришноте видно, что мы создаем новый объект класса Person и помещаем его в модель под ключом “person”. Потом возвращаем этот шаблон в этом контроллере и в этом шаблоне мы должны получать доуступ к объекту класса person, тоесть к пустому человеку, к человеку у которого не назначены значения для его полей.

В шаблоне Thymeleaf так же есть *method, action* и нужно указать тот объект(**th:object**) для которого эта форма предназначается. Дальше в коде мы используем те поля которые есть в классе person с помощью **th:field**.



**5. Валидация форм.**

**th:if –** условный атрибут. В конструкции **${}** указываем значение которое вернёт **истину** или **ложь**. С помощью символа **‘#’** получаем доступ к полям объекта который указан в **th:object.**

С помощью метода *hasErrors()* - проверяем если есть какие нибудь ошибки для полей которые мы указали в урагментах этого метода.

**th:errors –** выводит ошибки если у поля которое находится в конструкции **\*{}** есть ошибки.

В этом примере мы проверяем три поля объекта **person** на наличие ошибок.

Если в **th:if** метод *hasErrors()* возрващает истину, то тег <div> будет отображаться на форме и в этот тег будут вставляться сообщения об ошибках которые мы указали в самом классе **Person** с помощью аннотаций **@NotEmpty**, **@Size** и пр. С помощью атрибута th:errors мы говорим, что именно для этого поля должна выводится ошибки.