**Fluent Validation:**

It’s used for strongly typed validation rules in .net library.

There are two types of validators

1. Build-in Validator
2. Custom Validator

Git source : https://github.com/jayamoorthi/MinimalAPIAutoDIRegister/commit/c287f134165fc92adbeb7454ac37240fc51fbdb6

Build-in Validator.

*It already has been exists some build in validation methods like, NotNull(), NotEmpty(), EmailAddresses(), NotEqual(), Equal(), Length(),MaxLength(),MinLength(),LessThan(),LessThanOrEqualTo(),*

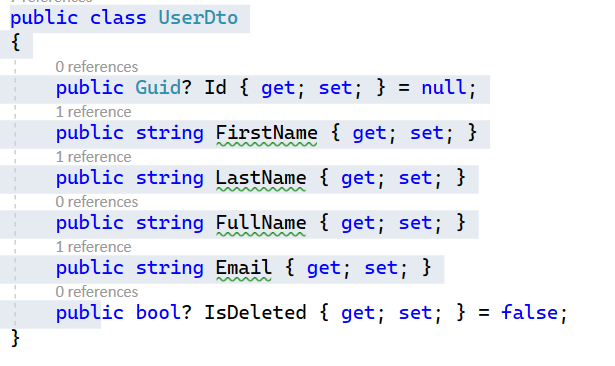
RuleFor(user => user.Email).EmailAddress().WithMessage("Enter valid email address");

RuleFor(user => user.LastName).MaximumLength(100)

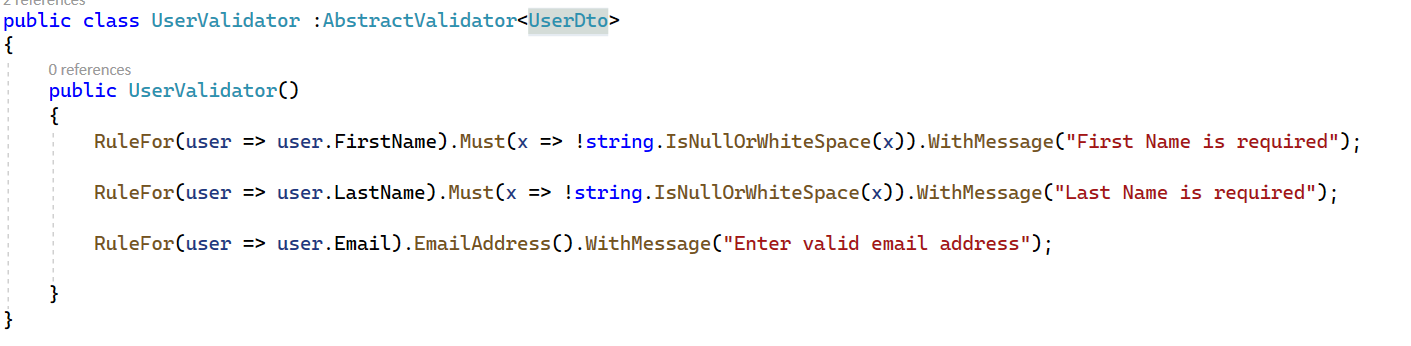
RuleFor(user => user.FirstName).MinimumLength(10)

Custom Validator:

Create our own custom validator class and inherit AbstractValidator<Model> class on it, then write our own custom logic which is need to validate in our model.

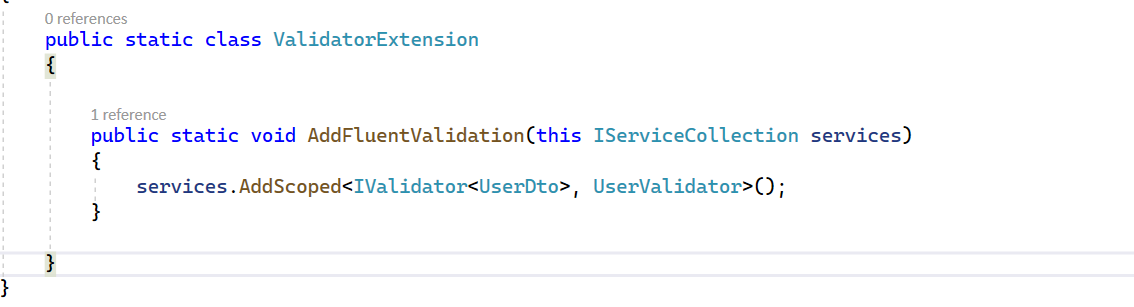


Rule for the model need to validates.



DI register all the custom validator in service using extension method

Method 1: registering in service collection



Method 2: loading all DI from assembly in using AddValidatorFromAssesmbly()



How do validate our custom validator?

We can inject our custom validator in our userservice class and call Validate(), if rule is success then Isvalid is true, else return errors collection of each property.

