Module 06:

"Validation"

Agenda

▸ Introduction

▸ **Validation using FluentValidation**

▸ MediatR Pipelines and Behaviors

▸ Summary

Types of Validation

▸ Model Validation
- In Presentation (or in entry to Application)

▸ Domain Validation
- Business rules in Domain

FluentValidation

▸ Freely available nuget package
  - https://docs.fluentvalidation.net/en/latest/

▸ **FluentValidation** is a .NET library for building strongly-typed validation rules using a Fluent API syntax

▸ Additionally;
  - FluentValidation.AspNetCore

FluentValidation Basics

▸ Define type-safe validator for specified object

```csharp
public class CustomerDtoValidator : AbstractValidator<CustomerDto>
{
    public CustomerDtoValidator()
    {
        RuleFor(customer => customer.Age).LessThan(100);
        RuleFor(customer => customer.FullName).NotNull();
    }
}
```

```csharp
record class CustomerDto

    int Age { ... }
    public string FullName { ... }
}
```

```csharp
CustomerDto customer = new CustomerDto() { ... };
CustomerDtoValidator validator = new CustomerDtoValidator();

ValidationResult result = validator.Validate(customer);
```
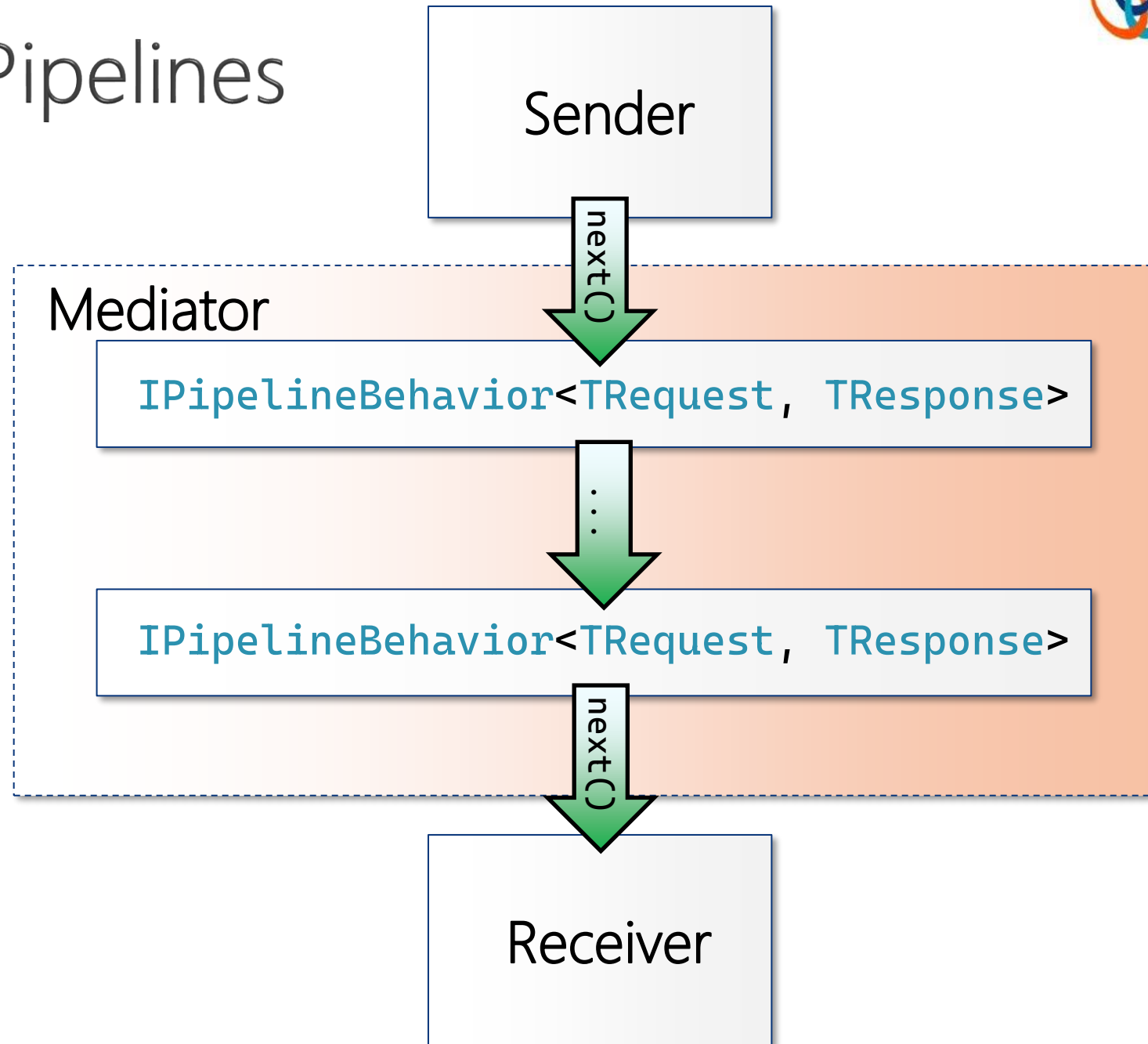
Agenda

▸ Introduction
▸ Validation using FluentValidation
▸ **MediatR Pipelines and Behaviors**
▸ Summary

MediatR Pipelines

Sender

Mediator

IPipelineBehavior<TRequest, TResponse>

...

IPipelineBehavior<TRequest, TResponse>

next()

next()

Receiver

WINCUBATE

Registering MediatR Pipeline Behaviors

```
services.AddMediatR(cfg =>
{
    cfg.AddBehavior<IPipelineBehavior<TRequest,TResponse>,ValidatorBehavior>();
});
```

```
services.AddMediatR(cfg =>
{
    cfg.AddOpenBehavior(typeof(GenericValidatorBehavior<,>);
});
```

Registering All Validators

▸ Use add-on nuget package **FluentValidation.AspNetCore**

```
services
    .AddValidatorsFromAssemblyContaining(typeof(DependencyInjection));
```

▸ <u>Note:</u> This registers all <u>public</u> validators in the specific assembly

Summary

- Introduction
- Validation using FluentValidation
- MediatR Pipelines and Behaviors