

Testes Automatizados com xUnit

Princípios e boas práticas

Cláudio Alves – claudio.d.silva@itau-unibanco.com.br

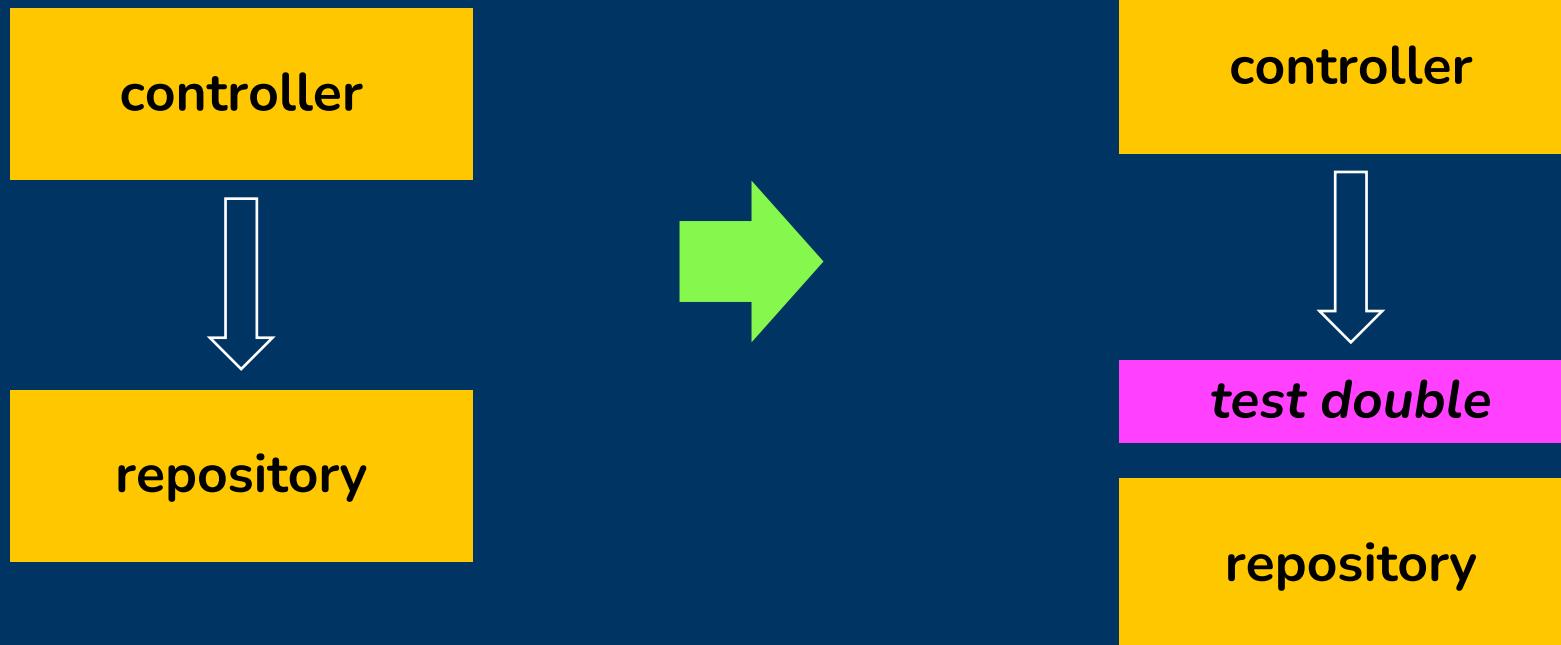
Testes de unidade

Testar as unidades de uma aplicação
de forma isolada sem usar qualquer
dependências externas



Técnica utilizada

isole as unidades

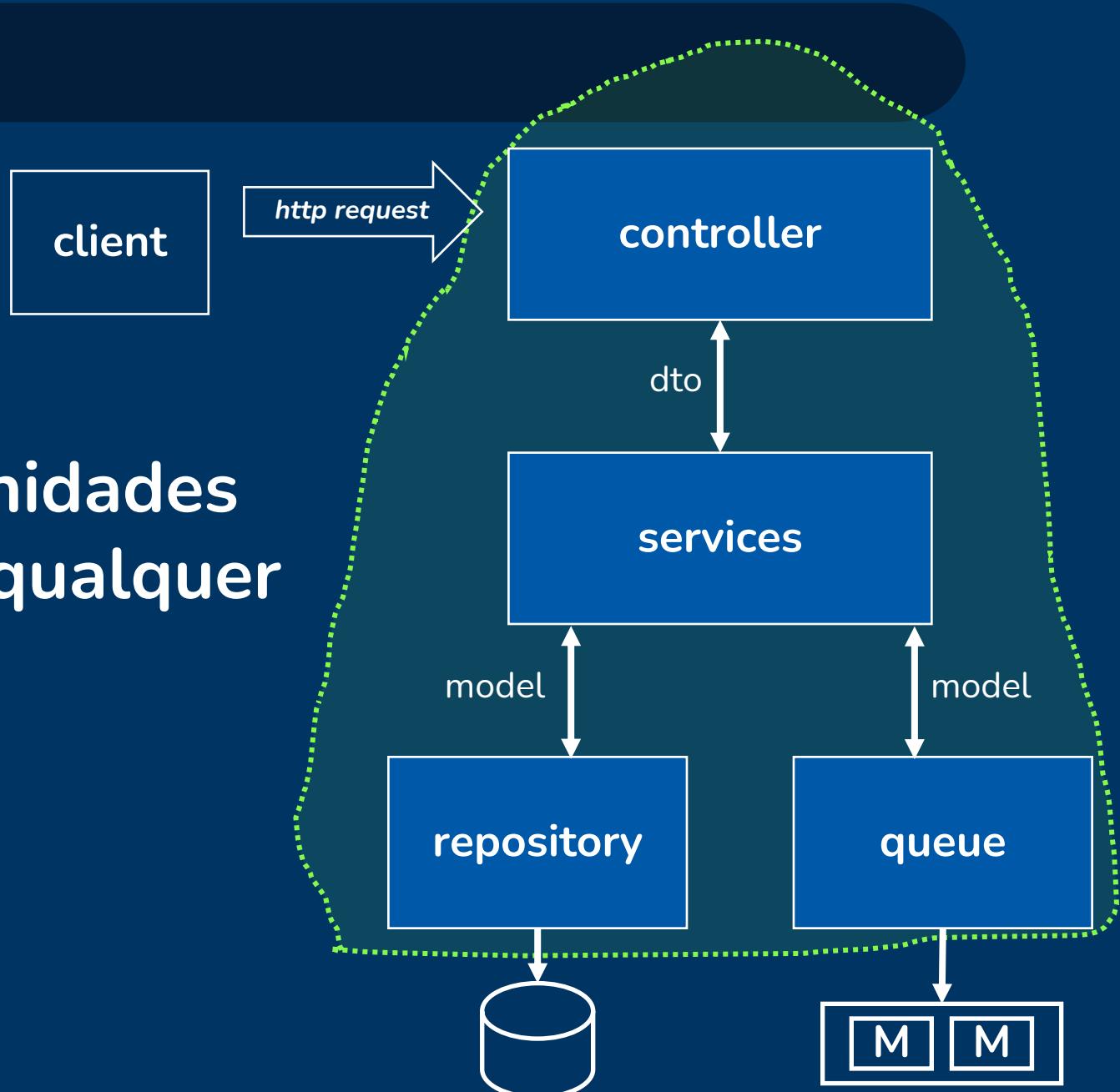


- Mock
- Stub
- Fake

a unidade tipicamente é uma classe

Testes de integração

Testar as integrações das unidades de uma aplicação sem usar qualquer dependências externas



Características de bons testes automatizados

rápido – em alguns milissegundos

isolado de todas as dependências externas

validação automática

sempre produzir o mesmo resultado

não deve conter lógica (if, else, foreach, ...)

um teste não pode depender do resultado de outro teste

Estabeleça um padrão de nomenclatura

Sugestão para projetos

<nome do projeto>.<tipo de teste>

```
✓  📂 src
    > 📂 Labs.Feedback.API

✓  📂 tests
    > 📂 Labs.Feedback.API.IntegrationTests
    > 📂 Labs.Feedback.API.UnitTests
```

Estabeleça um padrão de nomenclatura

Sugestão para classes

<nome da classe>Tests

public class FeedbackController

public class FeedbackControllerTests

Estabeleça um padrão de nomenclatura

Sugestão para métodos

<nome do método>_<cenário de teste>_<resultado esperado>

```
public bool CancelarReserva(Reserva reserva)
```



[Fact]

```
public void CancelarReserva_GerenteSolicitaCancelamentoDaReserva_True()...
```

[Fact]

```
public void CancelarReserva_FuncionarioSolicitaCancelamentoDeReservaComStatusAberta_True()...
```

[Fact]

```
public void CancelarReserva_FuncionarioSolicitaCancelamentoDeReservaComStatusConfirmada_False()...
```

Estrutura para o teste de unidade

Padrão dos 3 AAA

arrange

- Preparação dos dados
- Configuração dos dublês (mock, stub, fake, ...)

act

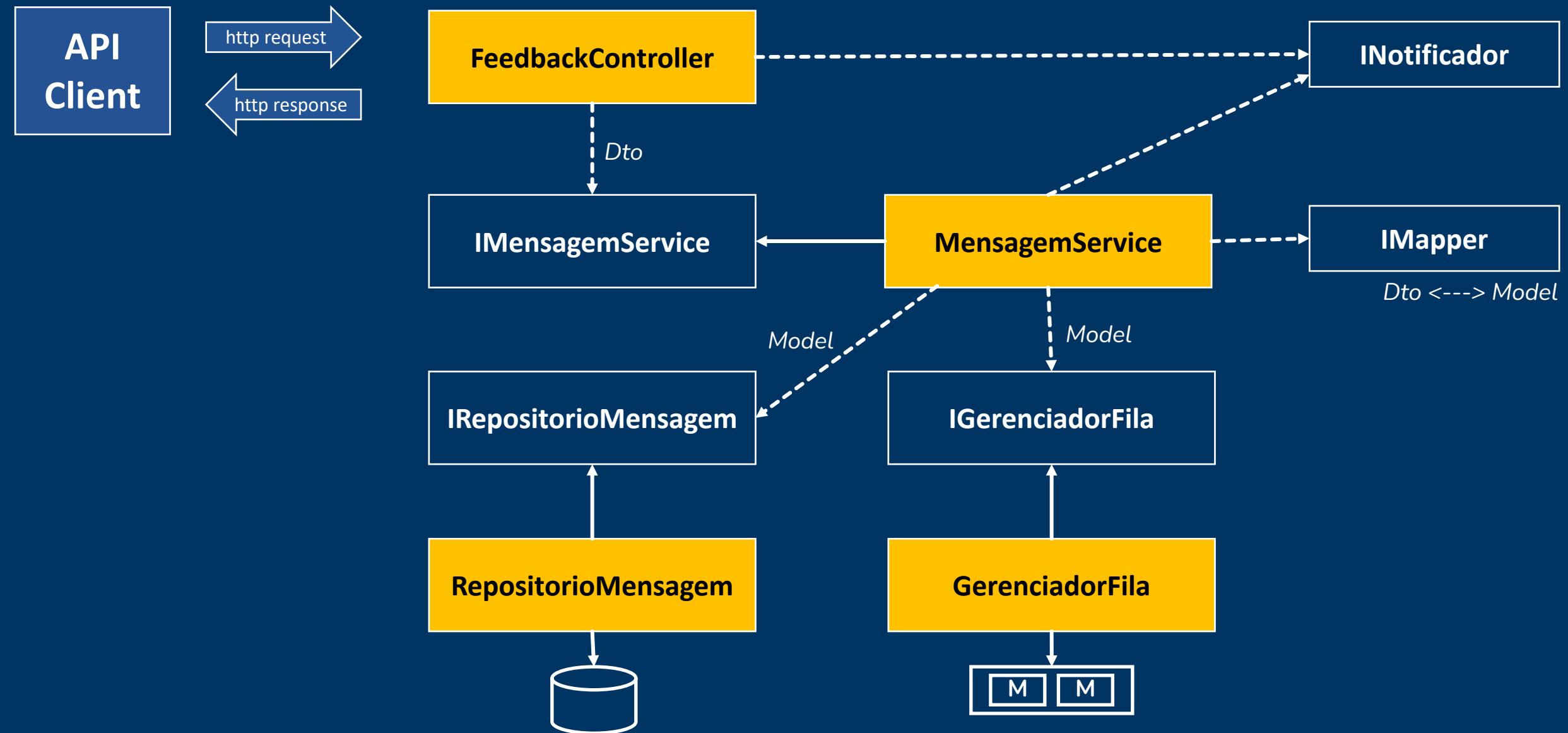
- Executa o método a ser testado
- Armazena o resultado

assert

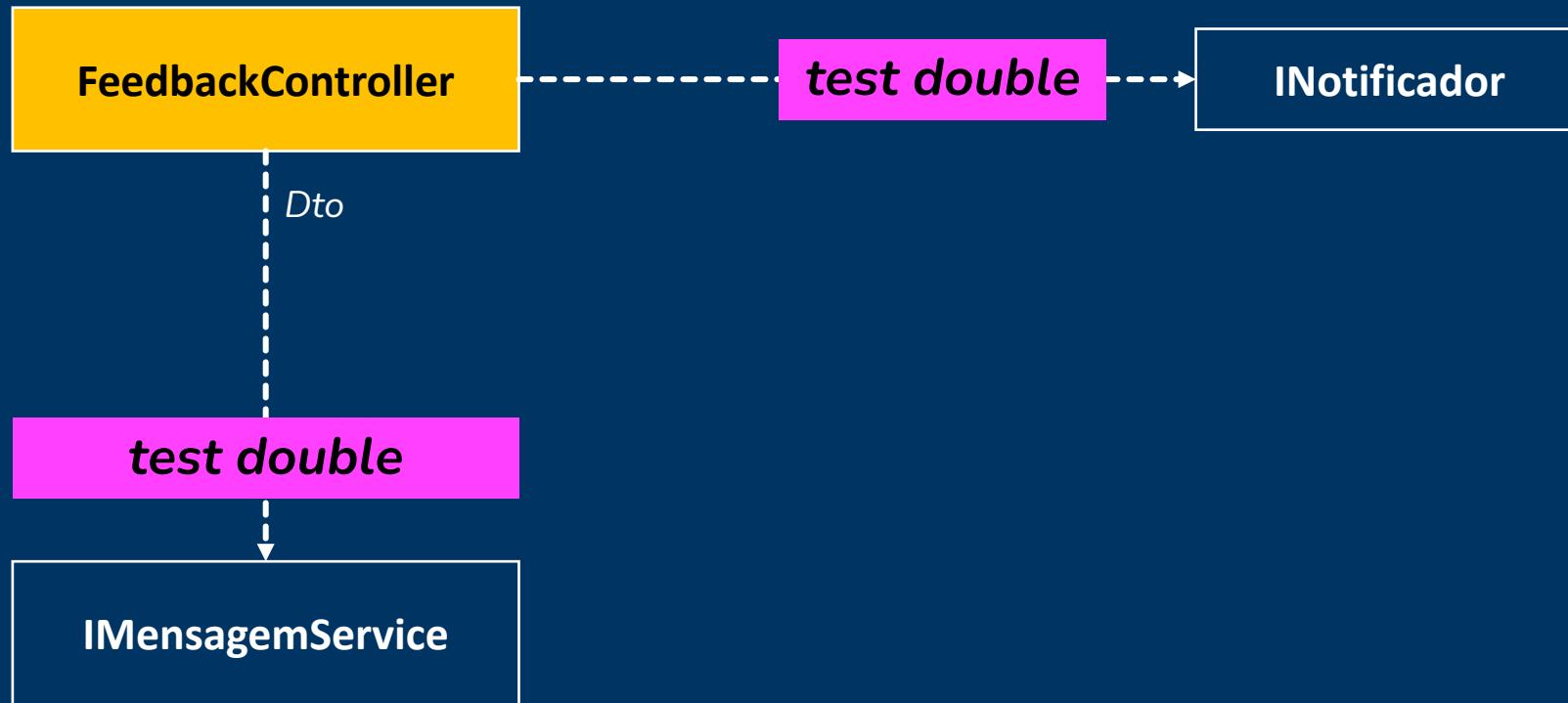
- Verifica o resultado com o valor esperado
- Verifica se o comportamento foi o esperado

Exemplo

<https://github.com/casje/labs-xunit>



Exemplo



técnica de *isolamento da unidade de teste*

Exemplo

Cenário básico – cadastrar feedback de erro

```
[HttpPost]  
3 references  
public IActionResult PostCadastrarMensagem(MensagemDto mensagemDto)  
{  
    var mensagem = this._mensagemService.CadastrarMensagem(mensagemDto);  
  
    if (_notificador.TemNotificacao())  
    {  
        return StatusCode(422, new {  
            data = _notificador.ObterNotificacoes()  
        });  
    }  
  
    if (mensagem == null)  
        return BadRequest();  
  
    return StatusCode(201, new {  
        data = mensagem  
    });  
}
```

[Fact]

0 references | Run Test | Debug Test

```
public void PostCadastrarMensagem_CadastrarMensagemValida_MensagemComHttpStatusCode201()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().Build();

    var mockMensagemServices = new Mock<IMensagemService>();
    mockMensagemServices.Setup(m => m.CadastrarMensagem(It.IsAny<MensagemDto>())).Returns(mensagemDto);

    var mockNotificador = new Mock<INotificador>();
    mockNotificador.Setup(m => m.TemNotificacao()).Returns(false);

    var controller = new FeedbackController(mockMensagemServices.Object, mockNotificador.Object);

}
}
```

[Fact]

0 references | Run Test | Debug Test

```
public void PostCadastrarMensagem_CadastrarMensagemValida_MensagemComHttpStatusCode201()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().Build();

    var mockMensagemServices = new Mock<IMensagemService>();
    mockMensagemServices.Setup(m => m.CadastrarMensagem(It.IsAny<MensagemDto>())).Returns(mensagemDto);

    var mockNotificador = new Mock<INotificador>();
    mockNotificador.Setup(m => m.TemNotificacao()).Returns(false);

    var controller = new FeedbackController(mockMensagemServices.Object, mockNotificador.Object);

    // Act
    var response = controller.PostCadastrarMensagem(mensagemDto) as ObjectResult;
    var data = response.GetData<MensagemDto>();

}
```

Preparação de dados e comportamentos

Executa o método a ser testado e armazena o resultado

[Fact]

0 references | Run Test | Debug Test

```
public void PostCadastrarMensagem_CadastrarMensagemValida_MensagemComHttpStatusCode201()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().Build();

    var mockMensagemServices = new Mock<IMensagemService>();
    mockMensagemServices.Setup(m => m.CadastrarMensagem(It.IsAny<MensagemDto>())).Returns(mensagemDto);

    var mockNotificador = new Mock<INotificador>();
    mockNotificador.Setup(m => m.TemNotificacao()).Returns(false);

    var controller = new FeedbackController(mockMensagemServices.Object, mockNotificador.Object);

    // Act
    var response = controller.PostCadastrarMensagem(mensagemDto) as ObjectResult;
    var data = response.GetData<MensagemDto>();

    // Assert
    Assert.Equal(201, response.StatusCode);
    Assert.Equal(mensagemDto, data);
}
```

Preparação de dados e comportamentos

Executa o método a ser testado e armazena o resultado

Validação do resultado com o valor esperado

Exemplo

Cenário alternativo – mensagem inválida

```
[HttpPost]  
3 references  
public IActionResult PostCadastrarMensagem(MensagemDto mensagemDto)  
{  
    var mensagem = this._mensagemService.CadastrarMensagem(mensagemDto);  
  
    if (_notificador.TemNotificacao())  
    {  
        return StatusCode(422, new {  
            data = _notificador.ObterNotificacoes()  
        });  
    }  
  
    if (mensagem == null)  
        return BadRequest();  
  
    return StatusCode(201, new {  
        data = mensagem  
    });  
}
```

[Fact]

0 references | Run Test | Debug Test

```
public void PostCadastrarMensagem_CadastrarMensagemComDescricaoInvalida_NotificacaoDeDescricaoEStatusCode422()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().Build();

    var mockMensagemServices = new Mock<IMensagemService>();
    mockMensagemServices.Setup(m => m.CadastrarMensagem(It.IsAny<MensagemDto>())).Returns(mensagemDto);

    var mockNotificador = new Mock<INotificador>();
    mockNotificador.Setup(m => m.TemNotificacao()).Returns(true);
    mockNotificador.Setup(m => m.ObterNotificacoes()).Returns(new List<Notificacao> { new Notificacao("descricao invalida") });

    var controller = new FeedbackController(mockMensagemServices.Object, mockNotificador.Object);
}
```



Preparação de dados e comportamentos



Executa o método a ser testado e armazena o resultado



Validação do resultado com o valor esperado

[Fact]

0 references | Run Test | Debug Test

```
public void PostCadastrarMensagem_CadastrarMensagemComDescricaoInvalida_NoticacaoDeDescricaoEStatusCode422()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().Build();

    var mockMensagemServices = new Mock<IMensagemService>();
    mockMensagemServices.Setup(m => m.CadastrarMensagem(It.IsAny<MensagemDto>())).Returns(mensagemDto);

    var mockNotificador = new Mock<INotificador>();
    mockNotificador.Setup(m => m.TemNotificacao()).Returns(true);
    mockNotificador.Setup(m => m.ObterNotificacoes()).Returns(new List<Notificacao> { new Notificacao("descricao invalida") });

    var controller = new FeedbackController(mockMensagemServices.Object, mockNotificador.Object);

    // Act
    var response = controller.PostCadastrarMensagem(mensagemDto) as ObjectResult;
    var data = response.GetData<List<Notificacao>>();

}

}
```

Preparação de dados e comportamentos

Executa o método a ser testado e armazena o resultado

Validação do resultado com o valor esperado

[Fact]

0 references | Run Test | Debug Test

```
public void PostCadastrarMensagem_CadastrarMensagemComDescricaoInvalida_NotificacaoDeDescricaoEStatusCode422()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().Build();

    var mockMensagemServices = new Mock<IMensagemService>();
    mockMensagemServices.Setup(m => m.CadastrarMensagem(It.IsAny<MensagemDto>())).Returns(mensagemDto);

    var mockNotificador = new Mock<INotificador>();
    mockNotificador.Setup(m => m.TemNotificacao()).Returns(true);
    mockNotificador.Setup(m => m.ObterNotificacoes()).Returns(new List<Notificacao> { new Notificacao("descricao invalida") });

    var controller = new FeedbackController(mockMensagemServices.Object, mockNotificador.Object);

    // Act
    var response = controller.PostCadastrarMensagem(mensagemDto) as ObjectResult;
    var data = response.GetData<List<Notificacao>>();

    // Assert
    Assert.Equal(422, response.StatusCode);
    Assert.Single(data);
}
```

Preparação de dados e comportamentos

Executa o método a ser testado e armazena o resultado

Validação do resultado com o valor esperado

Exemplo

Cenário alternativo – mensagem nula

```
[HttpPost]
3 references
public IActionResult PostCadastrarMensagem(MensagemDto mensagemDto)
{
    var mensagem = this._mensagemService.CadastrarMensagem(mensagemDto);

    if (_notificador.TemNotificacao())
    {
        return StatusCode(422, new {
            data = _notificador.ObterNotificacoes()
        });
    }

    if (mensagem == null)
        return BadRequest();

    return StatusCode(201, new {
        data = mensagem
    });
}
```

[Fact]

0 references | Run Test | Debug Test

```
public void PostCadastrarMensagem_CadastrarMensagemComErroDuranteProcessamento_HttpStatusCode400()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().Build();
    var mockMensagemServices = new Mock<IMensagemService>();
    var mockNotificador = new Mock<INotificador>();
    mockNotificador.Setup(m => m.TemNotificacao()).Returns(false);

    var controller = new FeedbackController(mockMensagemServices.Object, mockNotificador.Object);
}
```

Preparação de dados e comportamentos



Executa o método a ser testado e armazena o resultado



Validação do resultado com o valor esperado

[Fact]

0 references | Run Test | Debug Test

```
public void PostCadastrarMensagem_CadastrarMensagemComErroDuranteProcessamento_HttpStatusCode400()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().Build();
    var mockMensagemServices = new Mock<IMensagemService>();
    var mockNotificador = new Mock<INotificador>();
    mockNotificador.Setup(m => m.TemNotificacao()).Returns(false);

    var controller = new FeedbackController(mockMensagemServices.Object, mockNotificador.Object);

    // Act
    var response = controller.PostCadastrarMensagem(mensagemDto);
}
```

Preparação de dados e comportamentos

Executa o método a ser testado e armazena o resultado

Validação do resultado com o valor esperado

[Fact]

0 references | Run Test | Debug Test

```
public void PostCadastrarMensagem_CadastrarMensagemComErroDuranteProcessamento_HttpStatusCode400()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().Build();
    var mockMensagemServices = new Mock<IMensagemService>();
    var mockNotificador = new Mock<INotificador>();
    mockNotificador.Setup(m => m.TemNotificacao()).Returns(false);

    var controller = new FeedbackController(mockMensagemServices.Object, mockNotificador.Object);

    // Act
    var response = controller.PostCadastrarMensagem(mensagemDto);

    // Assert
    Assert.IsType<BadRequestResult>(response);
}
```

Preparação de dados e comportamentos

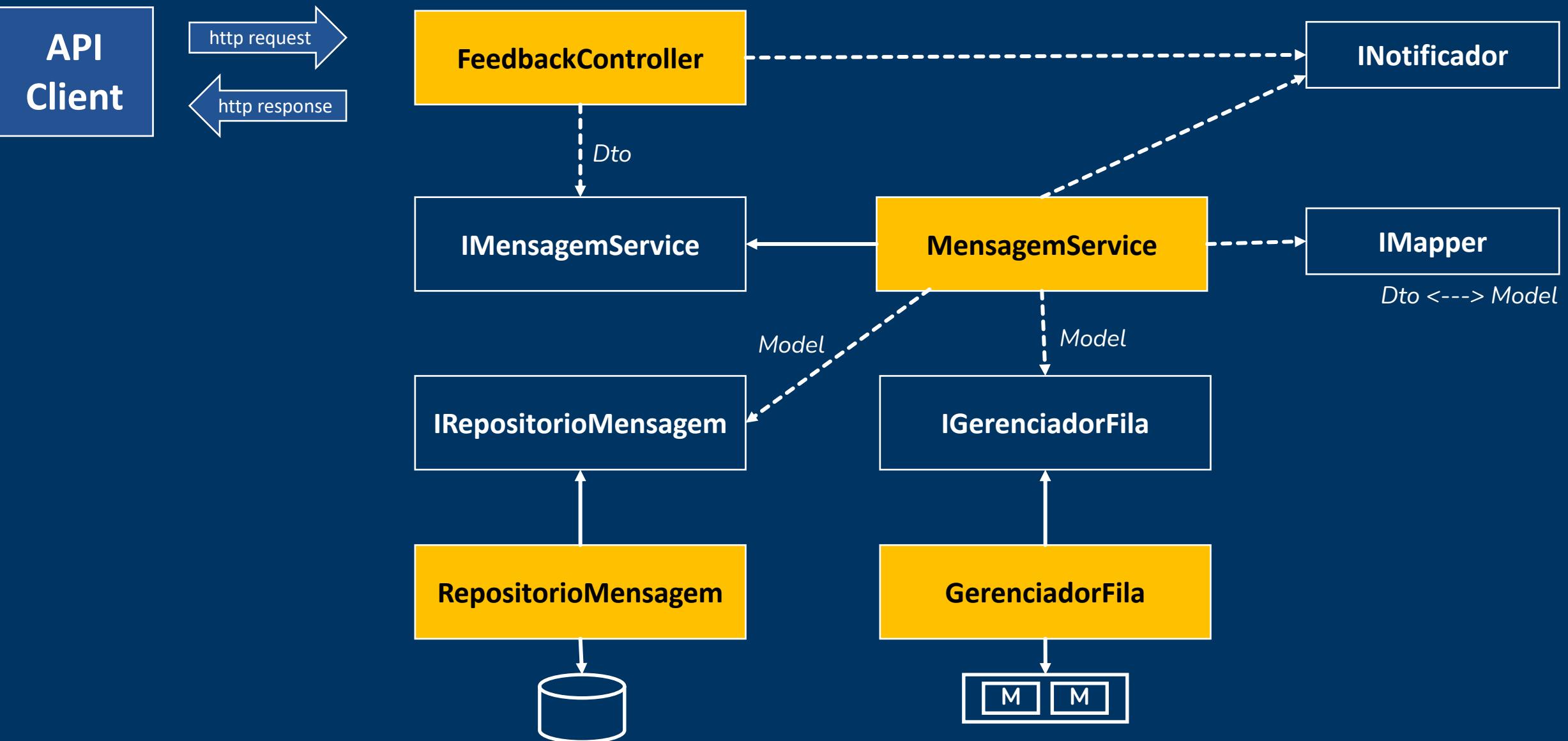


Executa o método a ser testado e armazena o resultado

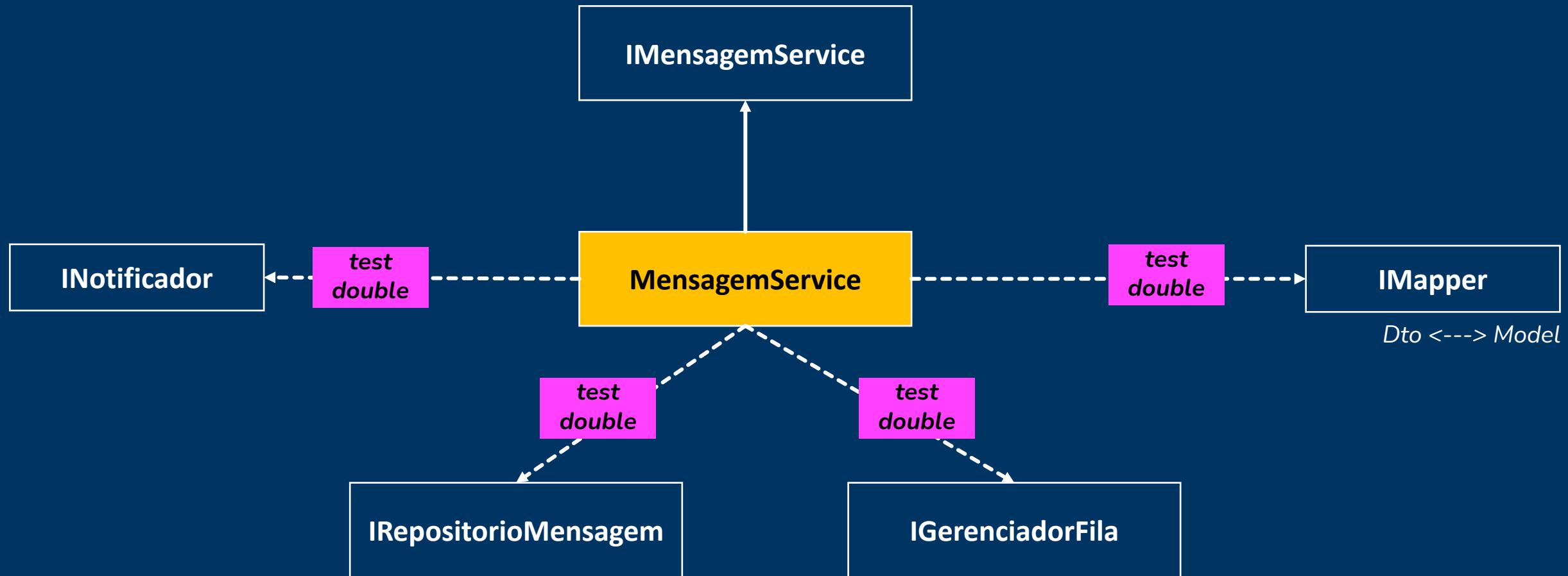


Validação do resultado com o valor esperado

Exemplo



Exemplo



Exemplo

Cenário básico – cadastrar feedback de erro

```
public MensagemDto CadastrarMensagem(MensagemDto mensagemDto)
{
    var mensagem = _mapper.Map<Mensagem>(mensagemDto);

    if (!ExecutarValidacao(new MensagemValidador(), mensagem)) return null;

    this._repositorioMensagem.AdicionarMensagem(mensagem);

    if (mensagem?.Categoria == Categoria.ERRO)
        this._gerenciadorFila.AdicionarItem(mensagem);

    return _mapper.Map<MensagemDto>(mensagem);
}
```

[Fact]

0 references | Run Test | Debug Test

```
public void CadastrarMensagem_ValidacaoDasInformacoesDaMensagemComDadosValidos_MensagemDtoComAsInformacoesDeCadastro()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().ComIdentDefault()
        .ComDescricao("Texto da mensagem").ComCategoria("ERRO").Build();
    var mensagem = MensagemBuilder.Criar().ComIdentDefault()
        .ComDescricao("Texto da mensagem").ComCategoria(Categoria.ERRO).Build();

    var mockNotification = new Mock<INotificador>();
    var mockMapper = new Mock<IMapper>();
    mockMapper.Setup(m => m.Map<Mensagem>(It.IsAny<MensagemDto>())).Returns(mensagem);
    mockMapper.Setup(m => m.Map<MensagemDto>(It.IsAny<Mensagem>())).Returns(mensagemDto);

    var mockRepositorio = new Mock<IRepositorioMensagem>();
    var mockGerenciadorFila = new Mock<IGerenciadorFila>();

    var mensagemService = new MensagemService(mockMapper.Object, mockNotification.Object
        , mockGerenciadorFila.Object, mockRepositorio.Object);
}
```

[Fact]

0 references | Run Test | Debug Test

```
public void CadastrarMensagem_ValidacaoDasInformacoesDaMensagemComDadosValidos_MensagemDtoComAsInformacoesDeCadastro()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().ComIdentDefault()
        .ComDescricao("Texto da mensagem").ComCategoria("ERRO").Build();
    var mensagem = MensagemBuilder.Criar().ComIdentDefault()
        .ComDescricao("Texto da mensagem").ComCategoria(Categoria.ERRO).Build();

    var mockNotification = new Mock<INotificador>();
    var mockMapper = new Mock<IMapper>();
    mockMapper.Setup(m => m.Map<Mensagem>(It.IsAny<MensagemDto>())).Returns(mensagem);
    mockMapper.Setup(m => m.Map<MensagemDto>(It.IsAny<Mensagem>())).Returns(mensagemDto);

    var mockRepositorio = new Mock<IRepositorioMensagem>();
    var mockGerenciadorFila = new Mock<IGerenciadorFila>();

    var mensagemService = new MensagemService(mockMapper.Object, mockNotification.Object
        , mockGerenciadorFila.Object, mockRepositorio.Object);
    // Act
    var mensagemCadastrada = mensagemService.CadastrarMensagem(mensagemDto);
}
```

[Fact]

0 references | Run Test | Debug Test

```
public void CadastrarMensagem_ValidacaoDasInformacoesDaMensagemComDadosValidos_MensagemDtoComAsInformacoesDeCadastro()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().ComIdentDefault()
        .ComDescricao("Texto da mensagem").ComCategoria("ERRO").Build();
    var mensagem = MensagemBuilder.Criar().ComIdentDefault()
        .ComDescricao("Texto da mensagem").ComCategoria(Categoria.ERRO).Build();

    var mockNotification = new Mock<INotificador>();
    var mockMapper = new Mock<IMapper>();
    mockMapper.Setup(m => m.Map<Mensagem>(It.IsAny<MensagemDto>())).Returns(mensagem);
    mockMapper.Setup(m => m.Map<MensagemDto>(It.IsAny<Mensagem>())).Returns(mensagemDto);

    var mockRepositorio = new Mock<IRepositorioMensagem>();
    var mockGerenciadorFila = new Mock<IGerenciadorFila>();

    var mensagemService = new MensagemService(mockMapper.Object, mockNotification.Object
        , mockGerenciadorFila.Object, mockRepositorio.Object);
    // Act
    var mensagemCadastrada = mensagemService.CadastrarMensagem(mensagemDto);

    // Assert
    Assert.Equal(MensagemDtoBuilder.IDENT_DEFAULT, mensagemCadastrada.Ident, ignoreCase: true);
    Assert.Equal("Texto da mensagem", mensagemCadastrada.Descricao, ignoreCase: true);
    Assert.Equal("ERRO", mensagemCadastrada.Categoria, ignoreCase: true);
}
```

Exemplo

Cenário alternativo – feedback de dúvida

```
public MensagemDto CadastrarMensagem(MensagemDto mensagemDto)
{
    var mensagem = _mapper.Map<Mensagem>(mensagemDto);

    if (!ExecutarValidacao(new MensagemValidador(), mensagem)) return null;

    this._repositorioMensagem.AdicionarMensagem(mensagem);

    if (mensagem?.Categoria == Categoria.ERRO)
        this._gerenciadorFila.AdicionarItem(mensagem);

    return _mapper.Map<MensagemDto>(mensagem);
}
```

```
public void CadastrarMensagem_ValidacaoDoComportamentoDoCadastroComMensagemValidaECategoriaDuvida_ComportamentoDeCadastroParaCategoriaDuvida()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().ComIdentDefault().ComCategoria("Dúvida").Build();
    var mensagem = MensagemBuilder.Criar().ComIdentDefault().ComCategoria(Categoria.DUVIDA).Build();

    var mockNotification = new Mock<INotificador>();

    var mockMapper = new Mock<IMapper>();
    mockMapper.Setup(m => m.Map<Mensagem>(It.IsAny<MensagemDto>())).Returns(mensagem);
    mockMapper.Setup(m => m.Map<MensagemDto>(It.IsAny<Mensagem>())).Returns(mensagemDto);

    var mockRepositorio = new Mock<IRepositorioMensagem>();
    mockRepositorio.Setup(m => m.AdicionarMensagem(mensagem)).Returns(true);

    var mockGerenciadorFila = new Mock<IGerenciadorFila>();
    mockGerenciadorFila.Setup(m => m.AdicionarItem(mensagem)).Returns(true);

    var mensagemService = new MensagemService(mockMapper.Object, mockNotification.Object
        , mockGerenciadorFila.Object, mockRepositorio.Object);
}
```

```
public void CadastrarMensagem_ValidacaoDoComportamentoDoCadastroComMensagemValidaECategoriaDodia_ComportamentoDeCadastroParaCategoriaDodia()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().ComIdentDefault().ComCategoria("Dodia").Build();
    var mensagem = MensagemBuilder.Criar().ComIdentDefault().ComCategoria(Categoria.DODIA).Build();

    var mockNotification = new Mock<INotificador>();

    var mockMapper = new Mock<IMapper>();
    mockMapper.Setup(m => m.Map<Mensagem>(It.IsAny<MensagemDto>())).Returns(mensagem);
    mockMapper.Setup(m => m.Map<MensagemDto>(It.IsAny<Mensagem>())).Returns(mensagemDto);

    var mockRepositorio = new Mock<IRepositorioMensagem>();
    mockRepositorio.Setup(m => m.AdicionarMensagem(mensagem)).Returns(true);

    var mockGerenciadorFila = new Mock<IGerenciadorFila>();
    mockGerenciadorFila.Setup(m => m.AdicionarItem(mensagem)).Returns(true);

    var mensagemService = new MensagemService(mockMapper.Object, mockNotification.Object
        , mockGerenciadorFila.Object, mockRepositorio.Object);

    // Act
    var mensagemCadastrada = mensagemService.CadastrarMensagem(mensagemDto);
}
```

```
public void CadastrarMensagem_ValidacaoDoComportamentoDoCadastroComMensagemValidaECategoriaDuvida_ComportamentoDeCadastroParaCategoriaDuvida()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().ComIdentDefault().ComCategoria("Duvida").Build();
    var mensagem = MensagemBuilder.Criar().ComIdentDefault().ComCategoria(Categoria.DUVIDA).Build();

    var mockNotification = new Mock<INotificador>();

    var mockMapper = new Mock<IMapper>();
    mockMapper.Setup(m => m.Map<Mensagem>(It.IsAny<MensagemDto>())).Returns(mensagem);
    mockMapper.Setup(m => m.Map<MensagemDto>(It.IsAny<Mensagem>())).Returns(mensagemDto);

    var mockRepositorio = new Mock<IRepositorioMensagem>();
    mockRepositorio.Setup(m => m.AdicionarMensagem(mensagem)).Returns(true);

    var mockGerenciadorFila = new Mock<IGerenciadorFila>();
    mockGerenciadorFila.Setup(m => m.AdicionarItem(mensagem)).Returns(true);

    var mensagemService = new MensagemService(mockMapper.Object, mockNotification.Object
        , mockGerenciadorFila.Object, mockRepositorio.Object);

    // Act
    var mensagemCadastrada = mensagemService.CadastrarMensagem(mensagemDto);

    // Assert
    mockMapper.Verify(m => m.Map<Mensagem>(It.IsAny<MensagemDto>()), Times.Once());
    mockRepositorio.Verify(m => m.AdicionarMensagem(It.IsAny<Mensagem>()), Times.Once());
    mockGerenciadorFila.Verify(m => m.AdicionarItem(It.IsAny<Mensagem>()), Times.Never());
    mockMapper.Verify(m => m.Map<MensagemDto>(It.IsAny<Mensagem>()), Times.Once());
    Assert.NotNull(mensagemCadastrada);
}
```

```
public void CadastrarMensagem_ValidacaoDoComportamentoDoCadastroComMensagemValidaECategoriaDodia_ComportamentoDeCadastroParaCategoriaDodia()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().ComIdentDefault().ComCategoria("Dodia").Build();
    var mensagem = MensagemBuilder.Criar().ComIdentDefault().ComCategoria(Categoria.DODIA).Build();

    var mockNotification = new Mock<INotificador>();

    var mockMapper = new Mock<IMapper>();
    mockMapper.Setup(m => m.Map<Mensagem>(It.IsAny<MensagemDto>())).Returns(mensagem);
    mockMapper.Setup(m => m.Map<MensagemDto>(It.IsAny<Mensagem>())).Returns(mensagemDto);

    var mockRepositorio = new Mock<IRepositorioMensagem>();
    mockRepositorio.Setup(m => m.AdicionarMensagem(mensagem)).Returns(true);

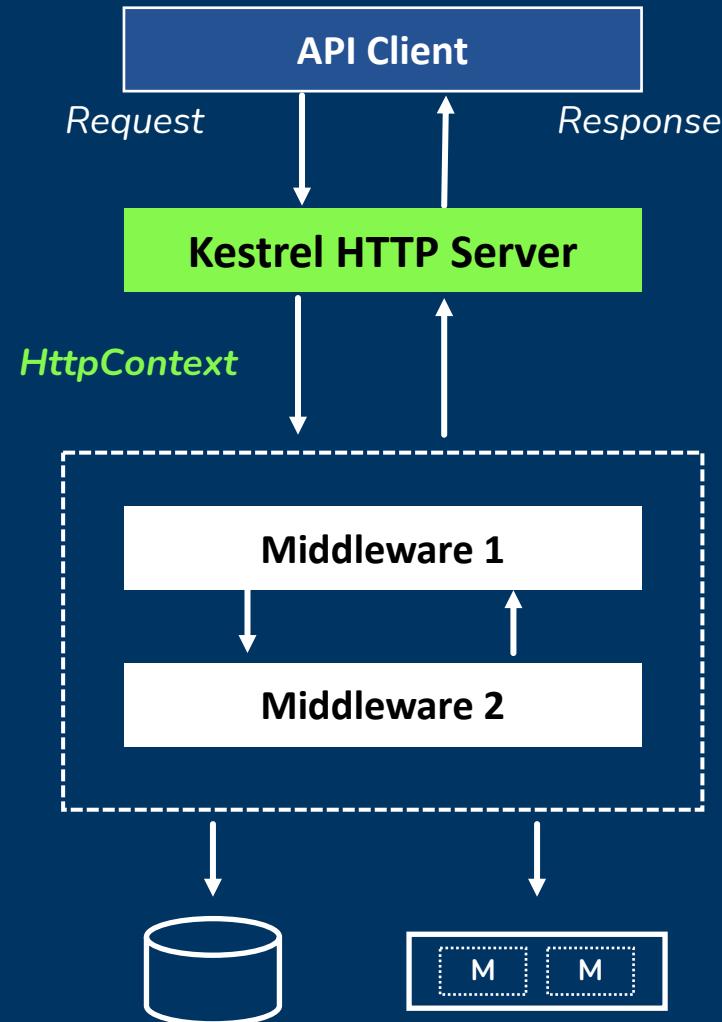
    var mockGerenciadorFila = new Mock<IGerenciadorFila>();
    mockGerenciadorFila.Setup(m => m.AdicionarItem(mensagem)).Returns(true);

    var mensagemService = new MensagemService(mockMapper.Object, mockNotification.Object
        , mockGerenciadorFila.Object, mockRepositorio.Object);

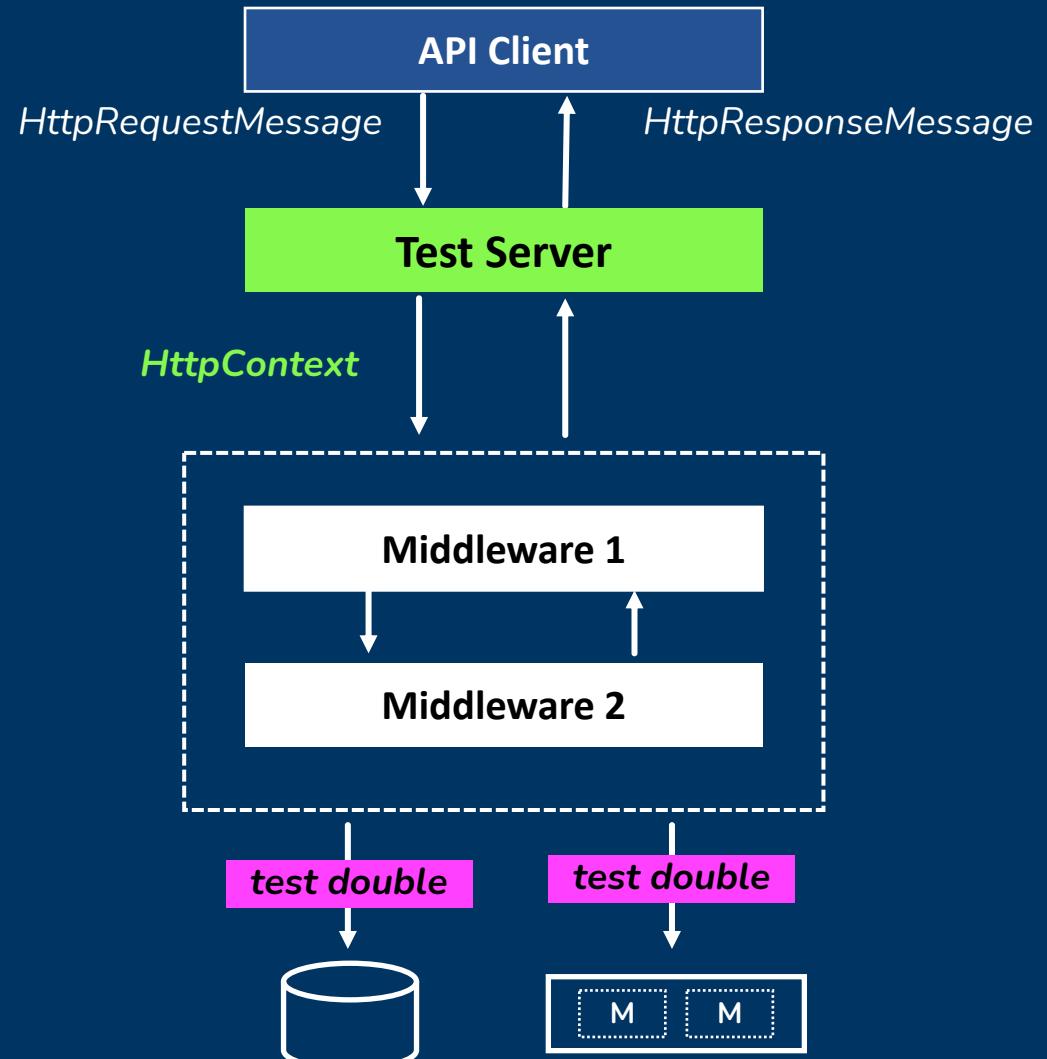
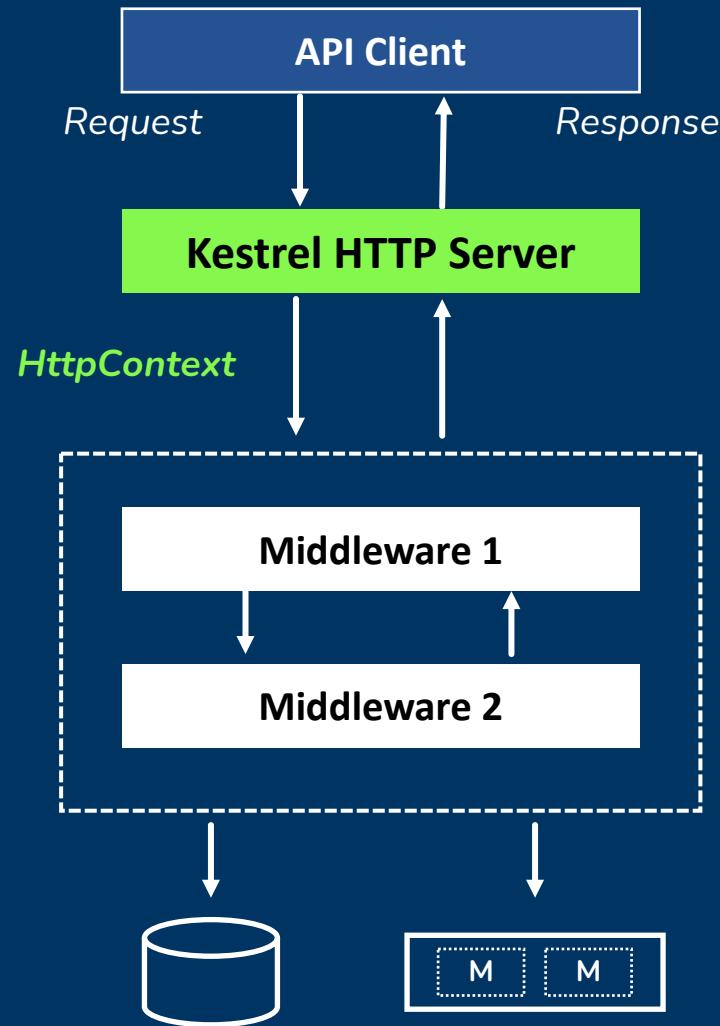
    // Act
    var mensagemCadastrada = mensagemService.CadastrarMensagem(mensagemDto);

    // Assert
    mockMapper.Verify(m => m.Map<Mensagem>(It.IsAny<MensagemDto>()), Times.Once());
    mockRepositorio.Verify(m => m.AdicionarMensagem(It.IsAny<Mensagem>()), Times.Once());
    mockGerenciadorFila.Verify(m => m.AdicionarItem(It.IsAny<Mensagem>()), Times.Never());
    mockMapper.Verify(m => m.Map<MensagemDto>(It.IsAny<Mensagem>()), Times.Once());
    Assert.NotNull(mensagemCadastrada);
}
```

Testes de integração



Testes de integração



```
private IHostBuilder CriarHostBuilderComGerenciador(IGerenciadorFila gerenciadorFila)
{
    var hostBuilder = new HostBuilder()
        .ConfigureWebHost(webHost => {
            // Add TestServer
            webHost.UseTestServer(); ← Usar o Test Server
            webHost.ConfigureServices(services =>
            {
                services.AddControllers();
                services.AddAutoMapper();
                services.AddScoped<INotificador, Notificador>();
                services.AddScoped<IMensagemService, MensagemService>();
                services.AddScoped<IRepositoryMensagem, RepositoryMensagem>();

                D U B L Ê // Gerenciador de Fila "fake"
                services.AddScoped<IGerenciadorFila>(sp => gerenciadorFila);
                // Banco de dados em memória
                services.AddDbContext<AppDbContext>(options => {
                    options.UseInMemoryDatabase(String.Concat("dbtest-", new Random().Next(10000000, 99999999)));
                });
            });
            webHost.Configure(app => {
                app.UseRouting();
                app.UseEndpoints(endpoints => { endpoints.MapControllers(); });
            });
        });
    return hostBuilder;
}
```

```
private IHostBuilder CriarHostBuilderComGerenciador(IGerenciadorFila gerenciadorFila)
{
    var hostBuilder = new HostBuilder()
        .ConfigureWebHost(webHost => {
            // Add TestServer
            webHost.UseTestServer();
            webHost.ConfigureServices(services =>
            {
                services.AddControllers();
                services.AddAutoMapper();
                services.AddScoped<INotificador, Notificador>();
                services.AddScoped<IMensagemService, MensagemService>();
                services.AddScoped<IRepositorioMensagem, RepositorioMensagem>();

                // Gerenciador de Fila "fake"
                services.AddScoped<IGerenciadorFila>(sp => gerenciadorFila);
                // Banco de dados em memória
                services.AddDbContext<AppDbContext>(options => {
                    options.UseInMemoryDatabase(String.Concat("dbtest-", new Random().Next(10000000, 99999999)));
                });
            });
            webHost.Configure(app => {
                app.UseRouting();
                app.UseEndpoints(endpoints => { endpoints.MapControllers(); });
            });
        });
    return hostBuilder;
}
```

Usar o Gerenciador de Fila Fake

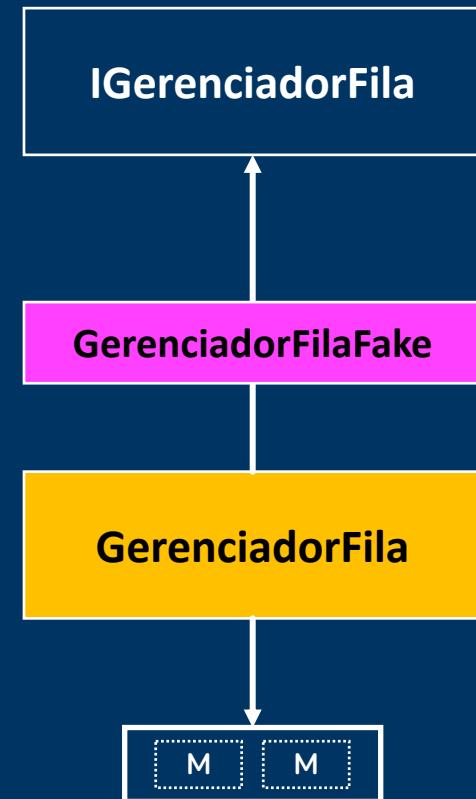
```
private IHostBuilder CriarHostBuilderComGerenciadorFake()
{
    var gerenciadorFilaFake = new GerenciadorFilaFake();
    return CriarHostBuilderComGerenciador(gerenciadorFilaFake);
}
```

D
U
B
L
Ê

Testes de integração

Teste dublê do tipo - Fake

```
1  using System;
2  using Labs.Feedback.API.Model;
3  using Labs.Feedback.API.Filas;
4
5  namespace Labs.Feedback.API.UtilTest
6  {
7      1 reference
8      public class GerenciadorFilaFake : IGerenciadorFila
9      {
10         6 references
11         public bool AdicionarItem(Mensagem mensagem)
12         {
13             return true;
14         }
15     }
16 }
```



Exemplo

Cenário básico – cadastrar feedback de erro

```
[HttpPost]  
3 references  
public IActionResult PostCadastrarMensagem(MensagemDto mensagemDto)  
{  
    var mensagem = this._mensagemService.CadastrarMensagem(mensagemDto);  
  
    if (_notificador.TemNotificacao())  
    {  
        return StatusCode(422, new {  
            data = _notificador.ObterNotificacoes()  
        });  
    }  
  
    if (mensagem == null)  
        return BadRequest();  
  
    return StatusCode(201, new {  
        data = mensagem  
    });  
}
```

Cenário básico – cadastrar feedback de erro

[Fact]

0 references | Run Test | Debug Test

```
public async Task CadastrarMensagem_RegistrarNovaMensagemDeFeedbackComCategoriaDeErro_RetornarHttpStatusCode201()
{
    // Arrange
    var hostBuilder = CriarHostBuilderComGerenciadorFake();
    var host = await hostBuilder.StartAsync();

    var client = host.GetTestClient();

    var mensagemDto = MensagemDtoBuilder.Criar().ComCategoria("Erro").Build();
    var body = new StringContent(MensagemDtoBuilder.ToJson(mensagemDto), Encoding.UTF8, "application/json");
}
```



Preparação de dados e comportamentos



Executa o método a ser testado e armazena o resultado



Validação do resultado com o valor esperado

Cenário básico – cadastrar feedback de erro

[Fact]

0 references | Run Test | Debug Test

```
public async Task CadastrarMensagem_RegistrarNovaMensagemDeFeedbackComCategoriaDeErro_RetornarHttpStatusCode201()
{
    // Arrange
    var hostBuilder = CriarHostBuilderComGerenciadorFake();
    var host = await hostBuilder.StartAsync();

    var client = host.GetTestClient();

    var mensagemDto = MensagemDtoBuilder.Criar().ComCategoria("Erro").Build();
    var body = new StringContent(MensagemDtoBuilder.ToJson(mensagemDto), Encoding.UTF8, "application/json");

    // Act
    var response = await client.PostAsync("http://exemplo.com/api/feedback", body);
    var mensagemRetorno = await response.Content.ReadAsStringAsync();
}
```



Preparação de dados e comportamentos



Executa o método a ser testado e armazena o resultado



Validação do resultado com o valor esperado

Cenário básico – cadastrar feedback de erro

[Fact]

0 references | Run Test | Debug Test

```
public async Task CadastrarMensagem_RegistrarNovaMensagemDeFeedbackComCategoriaDeErro_RetornarHttpStatusCode201()
{
    // Arrange
    var hostBuilder = CriarHostBuilderComGerenciadorFake();
    var host = await hostBuilder.StartAsync();

    var client = host.GetTestClient();

    var mensagemDto = MensagemDtoBuilder.Criar().ComCategoria("Erro").Build();
    var body = new StringContent(MensagemDtoBuilder.ToJson(mensagemDto), Encoding.UTF8, "application/json");

    // Act
    var response = await client.PostAsync("http://exemplo.com/api/feedback", body);
    var mensagemRetorno = await response.Content.ReadAsStringAsync();

    // Assert
    Assert.Equal(HttpStatusCode.Created, response.StatusCode);
    Assert.Contains(mensagemDto.Descricao, mensagemRetorno);
}
```



Preparação de dados e comportamentos



Executa o método a ser testado e armazena o resultado



Validação do resultado com o valor esperado

Exemplo

Cenário alternativo – feedback com categoria inválida

```
[HttpPost]  
3 references  
public IActionResult PostCadastrarMensagem(MensagemDto mensagemDto)  
{  
    var mensagem = this._mensagemService.CadastrarMensagem(mensagemDto);  
  
    if (_notificador.TemNotificacao())  
    {  
        return StatusCode(422, new {  
            data = _notificador.ObterNotificacoes()  
        });  
    }  
  
    if (mensagem == null)  
        return BadRequest();  
  
    return StatusCode(201, new {  
        data = mensagem  
    });  
}
```

Cenário básico – cadastrar feedback com categoria inválida

[Fact]

0 references | Run Test | Debug Test

```
public async Task CadastrarMensagem_RegistrarNovaMensagemComCategoriaInvalida_RetornarHttpStatusCode422()
{
    // Arrange
    var hostBuilder = CriarHostBuilderComGerenciadorFake();
    var host = await hostBuilder.StartAsync();

    var client = host.GetTestClient();

    var mensagemDto = MensagemDtoBuilder.Criar().ComCategoria("qualquer-coisa").Build();
    var body = new StringContent(MensagemDtoBuilder.ToJson(mensagemDto), Encoding.UTF8, "application/json");

}
```



Preparação de dados e comportamentos



Executa o método a ser testado e armazena o resultado



Validação do resultado com o valor esperado

Cenário básico – cadastrar feedback com categoria inválida

[Fact]

0 references | Run Test | Debug Test

```
public async Task CadastrarMensagem_RegistrarNovaMensagemComCategoriaInvalida_RetornarHttpStatusCode422()
{
    // Arrange
    var hostBuilder = CriarHostBuilderComGerenciadorFake();
    var host = await hostBuilder.StartAsync();

    var client = host.GetTestClient();

    var mensagemDto = MensagemDtoBuilder.Criar().ComCategoria("qualquer-coisa").Build();
    var body = new StringContent(MensagemDtoBuilder.ToJson(mensagemDto), Encoding.UTF8, "application/json");

    // Act
    var response = await client.PostAsync("http://exemplo.com/api/feedback", body);
    var mensagemRetorno = await response.Content.ReadAsStringAsync();
}
```

Cenário básico – cadastrar feedback com categoria inválida

[Fact]

0 references | Run Test | Debug Test

```
public async Task CadastrarMensagem_RegistrarNovaMensagemComCategoriaInvalida_RetornarHttpStatusCode422()
{
    // Arrange
    var hostBuilder = CriarHostBuilderComGerenciadorFake();
    var host = await hostBuilder.StartAsync();

    var client = host.GetTestClient();

    var mensagemDto = MensagemDtoBuilder.Criar().ComCategoria("qualquer-coisa").Build();
    var body = new StringContent(MensagemDtoBuilder.ToJson(mensagemDto), Encoding.UTF8, "application/json");

    // Act
    var response = await client.PostAsync("http://exemplo.com/api/feedback", body);
    var mensagemRetorno = await response.Content.ReadAsStringAsync();

    // Assert
    Assert.Equal(HttpStatusCode.UnprocessableEntity, response.StatusCode); //422
    Assert.Contains(Mensagens.CATEGORIA_INVALIDA, mensagemRetorno);
}
```



Preparação de dados e comportamentos



Executa o método a ser testado e armazena o resultado



Validação do resultado com o valor esperado

Boas práticas



Boas práticas

Evite a duplicidade do código de teste

```
public void PraticaRuim_CadastrarMensagem_ValidacaoDoComportamentoDoCadastroComMensagemValidaECategoriaDuvida_ComportamentoDeCadastroPa
{
    // Arrange
    MensagemDto mensagemDto = new MensagemDto
    {
        Ident = "dc5f6c82-2514-47f9-95ff-c3ce36ffdca1",
        Descricao = "Mensagem de feedback para o teste de unidade",
        Categoria = "Duvida"
    };

    Mensagem mensagem = new Mensagem
    {
        Ident = Guid.Parse("dc5f6c82-2514-47f9-95ff-c3ce36ffdca1"),
        Descricao = "Mensagem de feedback para o teste de unidade",
        Categoria = Categoria.DUVIDA
    };

    #region teste...
    // Assert
    mockMapper.Verify(m => m.Map<Mensagem>(It.IsAny<Mensaje
}

```

The diagram illustrates code duplication in a test setup. A brace on the right side groups the two identical 'Arrange' blocks. Another brace at the bottom groups the entire 'Arrange' section with the 'Assert' section below it.

ruim

Boas práticas

Evite a duplicidade do código de teste

```
public void CadastrarMensagem_ValidacaoDoComportamentoDoCadastroComMensagemValidaECategoriaDuvida_ComportamentoDeCadastroParaCategoriaDuvida()
{
    // Arrange
    var mensagemDto = MensagemDtoBuilder.Criar().ComIdentDefault().ComCategoria("Dúvida").Build();
    var mensagem = MensagemBuilder.Criar().ComIdentDefault().ComCategoria(Categoria.DUVIDA).Build(); } melhor

#region teste...
// Assert
mockMapper.Verify(m => m.Map<Mensagem>(It.IsAny<MensagemDto>()), Times.Once());
mockRepositorio.Verify(m => m.AdicionarMensagem(It.IsAny<Mensagem>()), Times.Once());
mockGerenciadorFila.Verify(m => m.AdicionarItem(It.IsAny<Mensagem>()), Times.Never());
mockMapper.Verify(m => m.Map<MensagemDto>(It.IsAny<Mensagem>()), Times.Once());
Assert.NotNull(mensagemCadastrada);
}
```

Boas práticas

Tente incluir apenas um único cenário para a validação do teste

ruim

```
[Fact]
public void Add_EdgeCases_ThrowsArgumentExceptions()
{
    Assert.Throws<ArgumentException>(() => stringCalculator.Add(null));
    Assert.Throws<ArgumentException>(() => stringCalculator.Add("a"));
}
```

melhor

```
[Theory]
[InlineData(null)]
[InlineData("a")]
public void Add_InputNullOrAlphabetic_ThrowsArgumentException(string input)
{
    var stringCalculator = new StringCalculator();

    Action actual = () => stringCalculator.Add(input);

    Assert.Throws<ArgumentException>(actual);
}
```

Boas práticas

Evite incluir lógica no código de teste

ruim

```
[Fact]
public void Add_MultipleNumbers_ReturnsCorrectResults()
{
    var stringCalculator = new StringCalculator();
    var expected = 0;
    var testCases = new[]
    {
        "0,0,0",
        "0,1,2",
        "1,2,3"
    };

    foreach (var test in testCases)
    {
        Assert.Equal(expected, stringCalculator.Add(test));
        expected += 3;
    }
}
```

melhor

```
[Theory]
[InlineData("0,0,0", 0)]
[InlineData("0,1,2", 3)]
[InlineData("1,2,3", 6)]
public void Add_MultipleNumbers_ReturnsSumOfNumbers(string input, int expected)
{
    var stringCalculator = new StringCalculator();

    var actual = stringCalculator.Add(input);

    Assert.Equal(expected, actual);
}
```

Boas práticas

Evite a criação de novos testes para a variação de valores

melhor: use testes parametrizados

```
[Theory]
[InlineData(1, 2, 3)]
[InlineData(-4, -6, -10)]
[InlineData(-2, 2, 0)]
[InlineData(int.MinValue, -1, int.MaxValue)]
public void CanAddTheory(int value1, int value2, int expected)
{
    var calculator = new Calculator();

    var result = calculator.Add(value1, value2);

    Assert.Equal(expected, result);
}
```

Boas práticas

Não deixe o teste para a ultima coisa a ser feita

- Os testes colaboram para a melhoria do design da aplicação
- Incentivam a refatoração

Boas práticas

Cuidado com os testes viciados

<https://www.nuget.org/packages/Bogus/>

```
var faker = new Faker("pt_BR");

Console.WriteLine(faker.Address.City());
Console.WriteLine(faker.Person.Cpf());
Console.WriteLine(faker.Company.Cnpj());
Console.WriteLine(faker.Random.Word());
Console.WriteLine(faker.Random.Decimal());
Console.WriteLine(faker.Random.Int());
Console.WriteLine(faker.Finance.Amount(1, 100));
```

Como validar se o teste está bom?

Deve possuir as características de bons testes

Altere o valor esperado na validação para quebrar o teste

Altere o código da aplicação para quebrar o teste

Obrigado!

