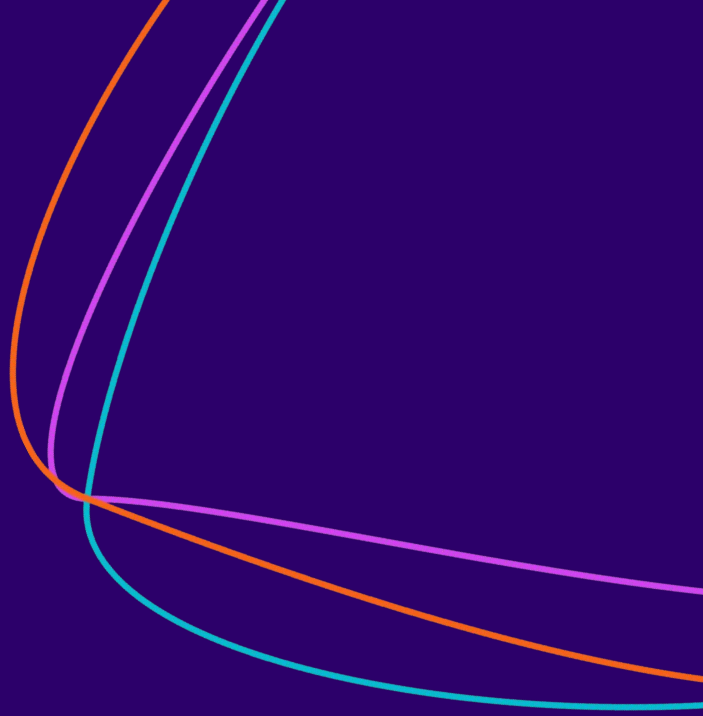
DIGIX

Não se limite aos testes de unidade e testes end-to-end!



Renan S Moreira

Tech leader @ Digix

Testes de integração

Limitações dos testes de unidade

```java
@Override
public ParcelaDePagamentoDoCenso buscarPeloId(
        ParcelaDePagamentoDoCensoId parcelaDePagamentoDoCensoId) {
    Query query = criarConsulta( hql: "SELECT parcela " +
            "FROM ParcelaDePagamentoDoCenso parcela " +
            "WHERE parcela.idd = :id");
    query.setParameter( s: "id",
            Integer.valueOf(parcelaDePagamentoDoCensoId.toString()));
    return (ParcelaDePagamentoDoCenso) query.uniqueResult();
}
```

Limitações dos testes de unidade

```java
@Test
void deveObterPeloId() {
    ParcelaDePagamentoDoCensoRepository repositorio =
            Mockito.mock(ParcelaDePagamentoDoCensoRepository.class);
    ParcelaDePagamentoDoCensoId identificadorDesejado =
            new ParcelaDePagamentoDoCensoId(UUID.randomUUID().toString());
    ParcelaDePagamentoDoCenso parcelaDePagamentoDesejada =
            new ParcelaDePagamentoDoCensoBuilder().comId(identificadorDesejado).criar();
    Mockito.when(repositorio.buscarPeloId(identificadorDesejado))
            .thenReturn(parcelaDePagamentoDesejada);

    ParcelaDePagamentoDoCenso parcelaDePagamentoObtida =
            repositorio.buscarPeloId(identificadorDesejado);

    Assertions
            .assertThat(parcelaDePagamentoObtida)
            .isEqualTo(parcelaDePagamentoDesejada);
}
```

Limitações dos testes de unidade

Limitações dos testes de unidade

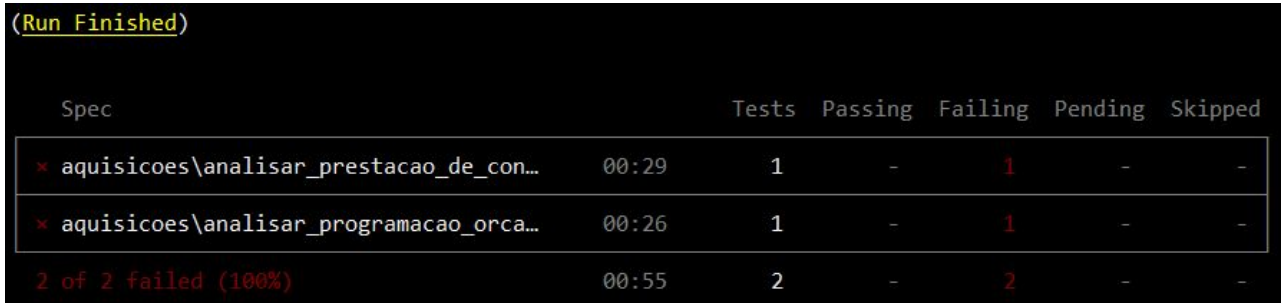
```java
@Entity
public class Contrato extends Entidade<ContratoId> {

    private ContratoId id;

    @OneToMany(cascade = CascadeType.ALL,
               orphanRemoval = true,
               fetch = FetchType.EAGER)
    @JoinColumn(name = "contrato_identificadorDoRepositorio")
    private List<ItemDoContrato> itens;
```

Limitações dos testes de unidade

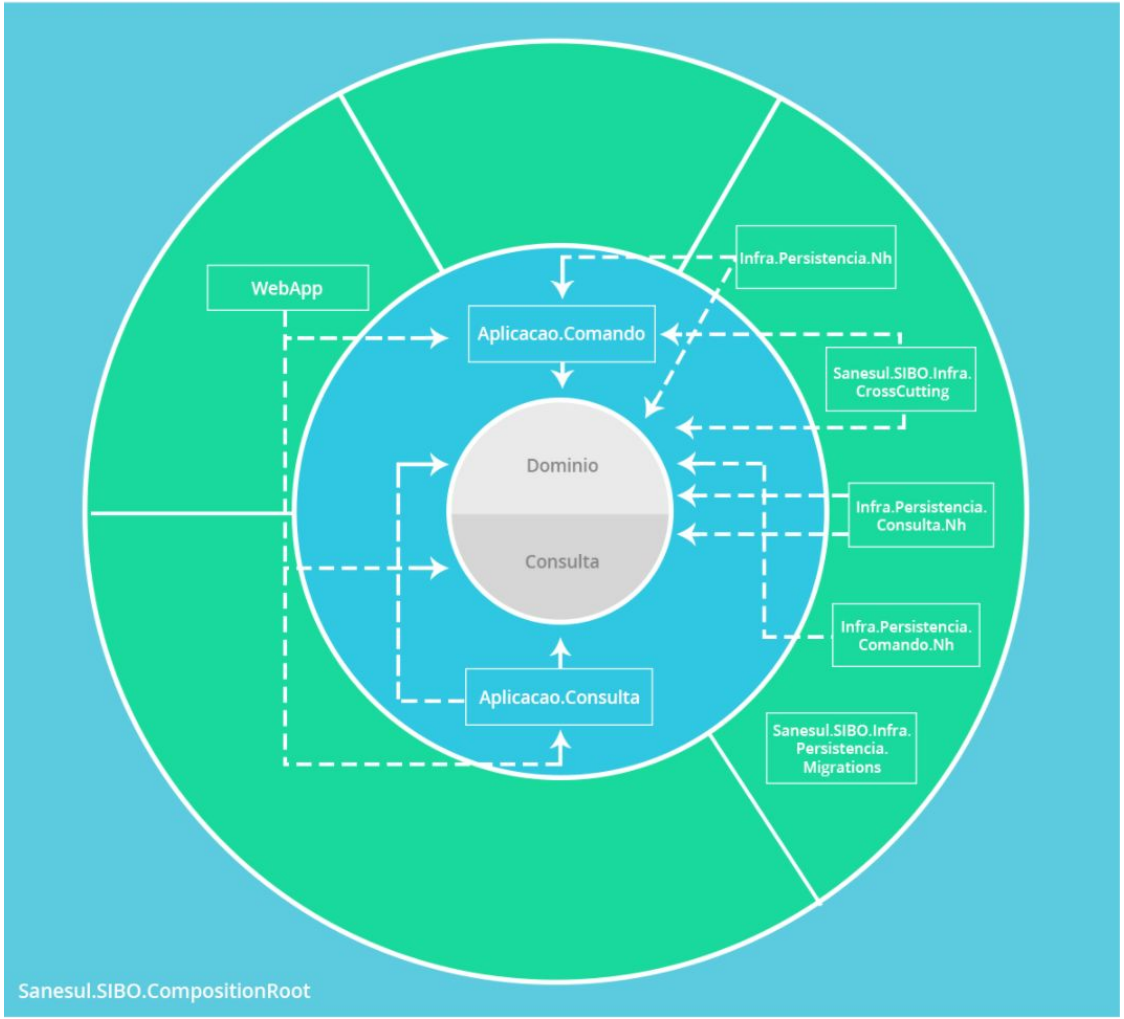
```
public class DecimalModelBinder : IModelBinder
{
    0 references
    public object BindModel(ControllerContext controllerContext,
        ModelBindingContext bindingContext)
    {
        var valueResult = bindingContext.
            ValueProvider
            .GetValue(bindingContext.ModelName);
        var modelState = new ModelState { Value = valueResult };

        object actualValue = null;
        try
        {
            actualValue = Convert.ToDecimal(
                valueResult.AttemptedValue,
                CultureInfo.CurrentCulture);
        }
        catch (FormatException e)
        {
            modelState.Errors.Add(e);
        }

        bindingContext.ModelState
            .Add(bindingContext.ModelName, modelState);

        return actualValue;
    }
}
```

Testes end-to-end

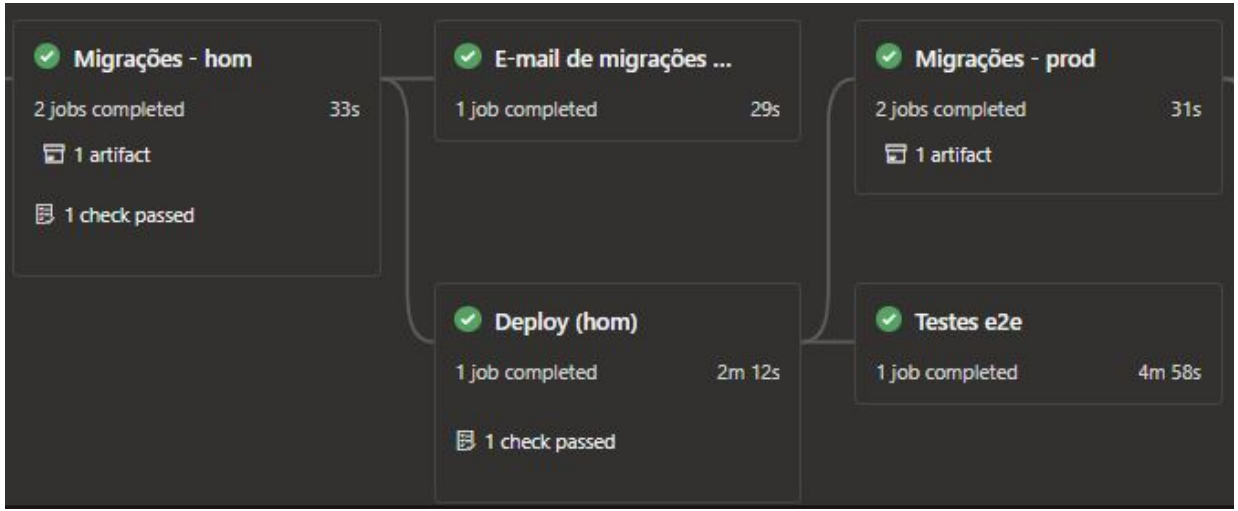
Problemas dos testes end-to-end



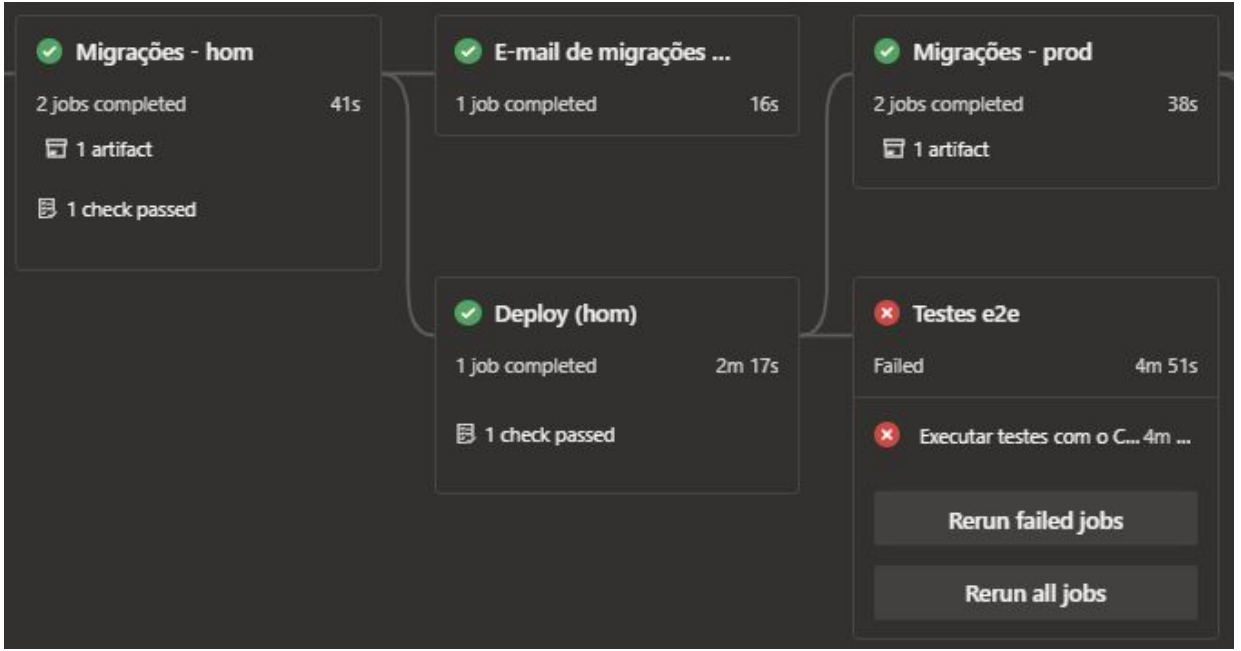
Problemas dos testes end-to-end



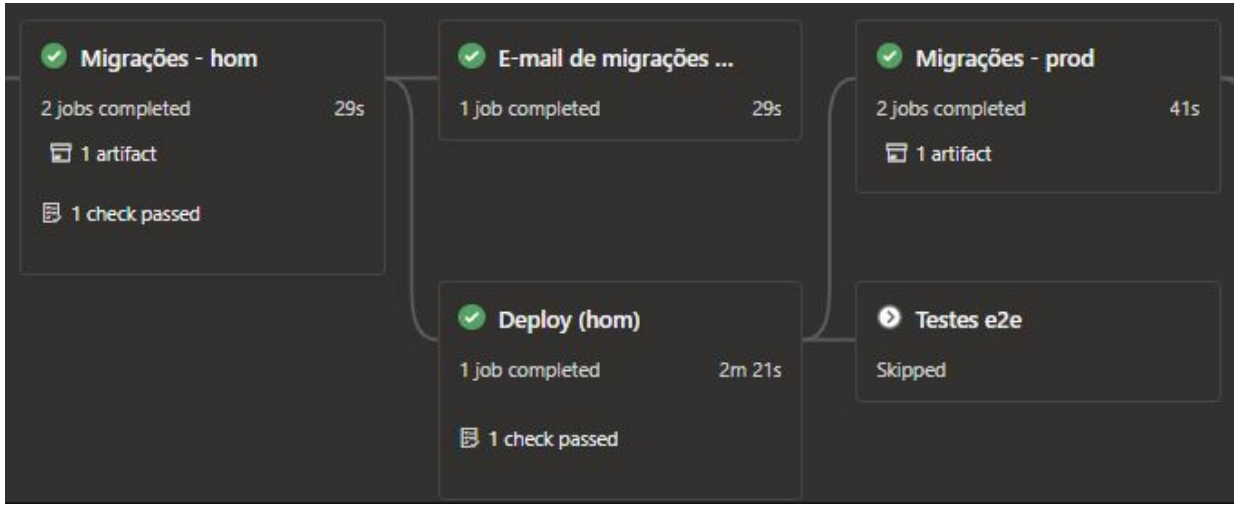
Problemas dos testes end-to-end



Problemas dos testes end-to-end



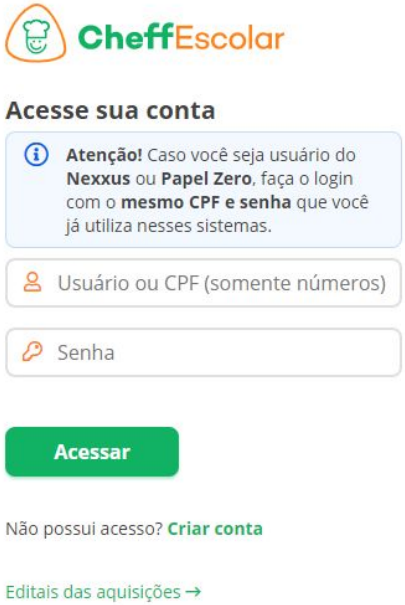
Problemas dos testes end-to-end



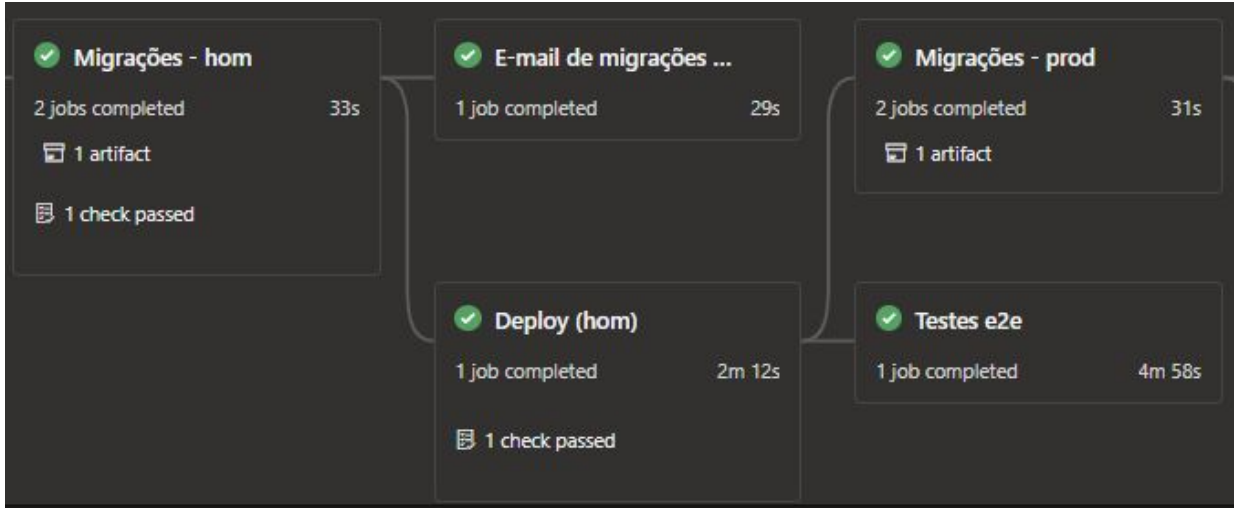
Como testar?

```java
@Override
public ParcelaDePagamentoDoCenso buscarPeloId(
        ParcelaDePagamentoDoCensoId parcelaDePagamentoDoCensoId) {
    Query query = criarConsulta( hql: "SELECT parcela " +
            "FROM ParcelaDePagamentoDoCenso parcela " +
            "WHERE parcela.idd = :id");
    query.setParameter( s: "id",
            Integer.valueOf(parcelaDePagamentoDoCensoId.toString()));
    return (ParcelaDePagamentoDoCenso) query.uniqueResult();
}
```

Abordagem
end-to-end



Abordagem end-to-end

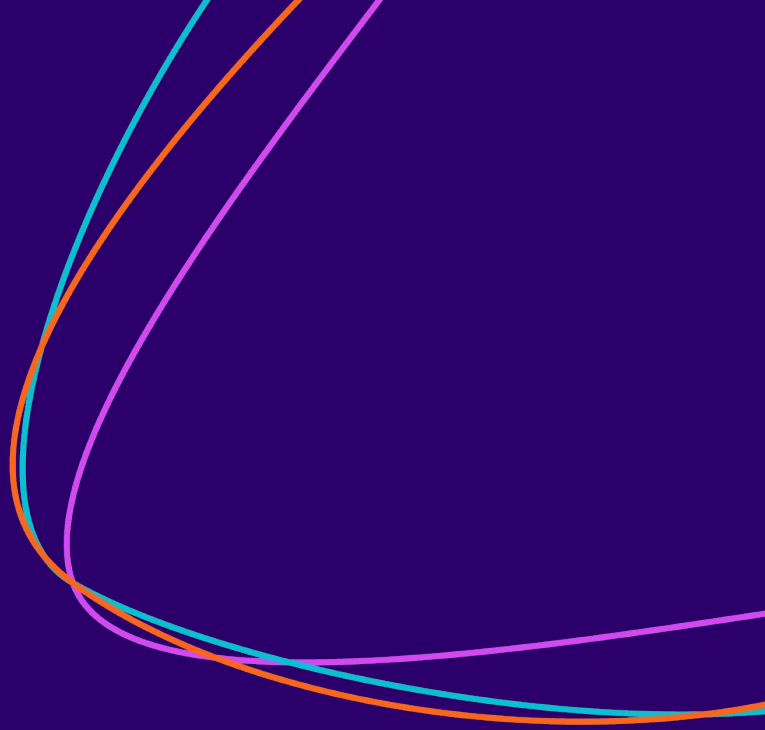


Como resolver?

Abordagem de teste direta

```
public class ParcelaDePagamentoDoCensoRepositoryHibernate
        extends HibernateRepository<ParcelaDePagamentoDoCenso>
        implements ParcelaDePagamentoDoCensoRepository {

    @Autowired
    public ParcelaDePagamentoDoCensoRepositoryHibernate(
            SessionFactory sessionFactory) {
        super(sessionFactory);
    }

    @Override
    public ParcelaDePagamentoDoCenso buscarPeloId(
            ParcelaDePagamentoDoCensoId parcelaDePagamentoDoCensoId) {
        Query query = criarConsulta( hql: "SELECT parcela " +
                "FROM ParcelaDePagamentoDoCenso parcela " +
                "WHERE parcela.idd = :id");
        query.setParameter( s: "id",
                Integer.valueOf(parcelaDePagamentoDoCensoId.toString()));
        return (ParcelaDePagamentoDoCenso) query.uniqueResult();
    }
```

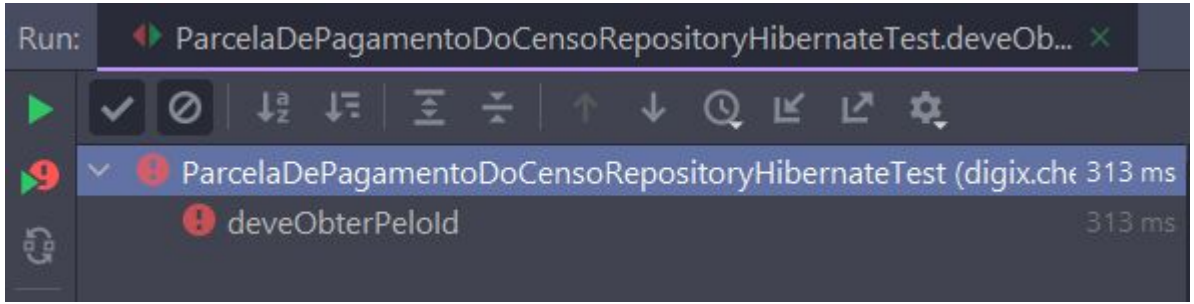
1.
Conheça sua stack



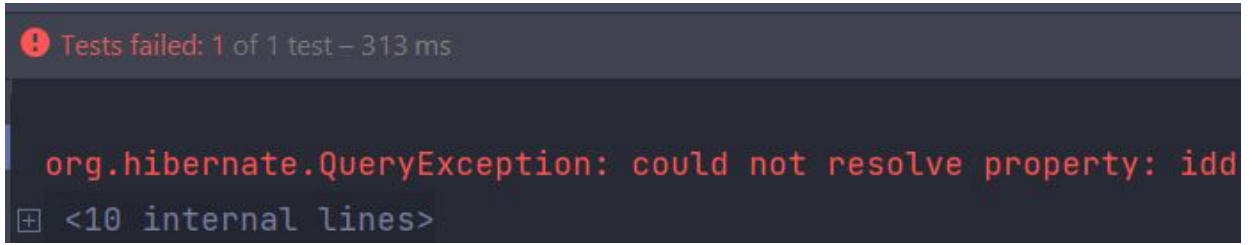
Abordagem de teste direta

```java
@RunWith(SpringRunner.class)
@SpringBootTest
@ContextConfiguration(classes = {ConfiguracaoDeTesteParaCamadaDeAplicacao.class})
public class ParcelaDePagamentoDoCensoRepositoryHibernateTest {

    @Autowired
    private ParcelaDePagamentoDoCensoRepository repositorio;

    @Test
    public void deveObterPeloId() {
        ParcelaDePagamentoDoCensoId identificadorDesejado =
                new ParcelaDePagamentoDoCensoId(UUID.randomUUID().toString());
        ParcelaDePagamentoDoCenso parcelaDePagamentoDesejada =
                new ParcelaDePagamentoDoCensoBuilder()
                        .comId(identificadorDesejado).criar();
        repositorio.incluir(parcelaDePagamentoDesejada);

        ParcelaDePagamentoDoCenso parcelaDePagamentoObtida =
                repositorio.buscarPeloId(identificadorDesejado);

        Assertions
                .assertThat(parcelaDePagamentoObtida)
                .isEqualTo(parcelaDePagamentoDesejada);
    }
```

Abordagem de teste direta



Abordagem de teste direta



Comparando resultados

Comparando resultados

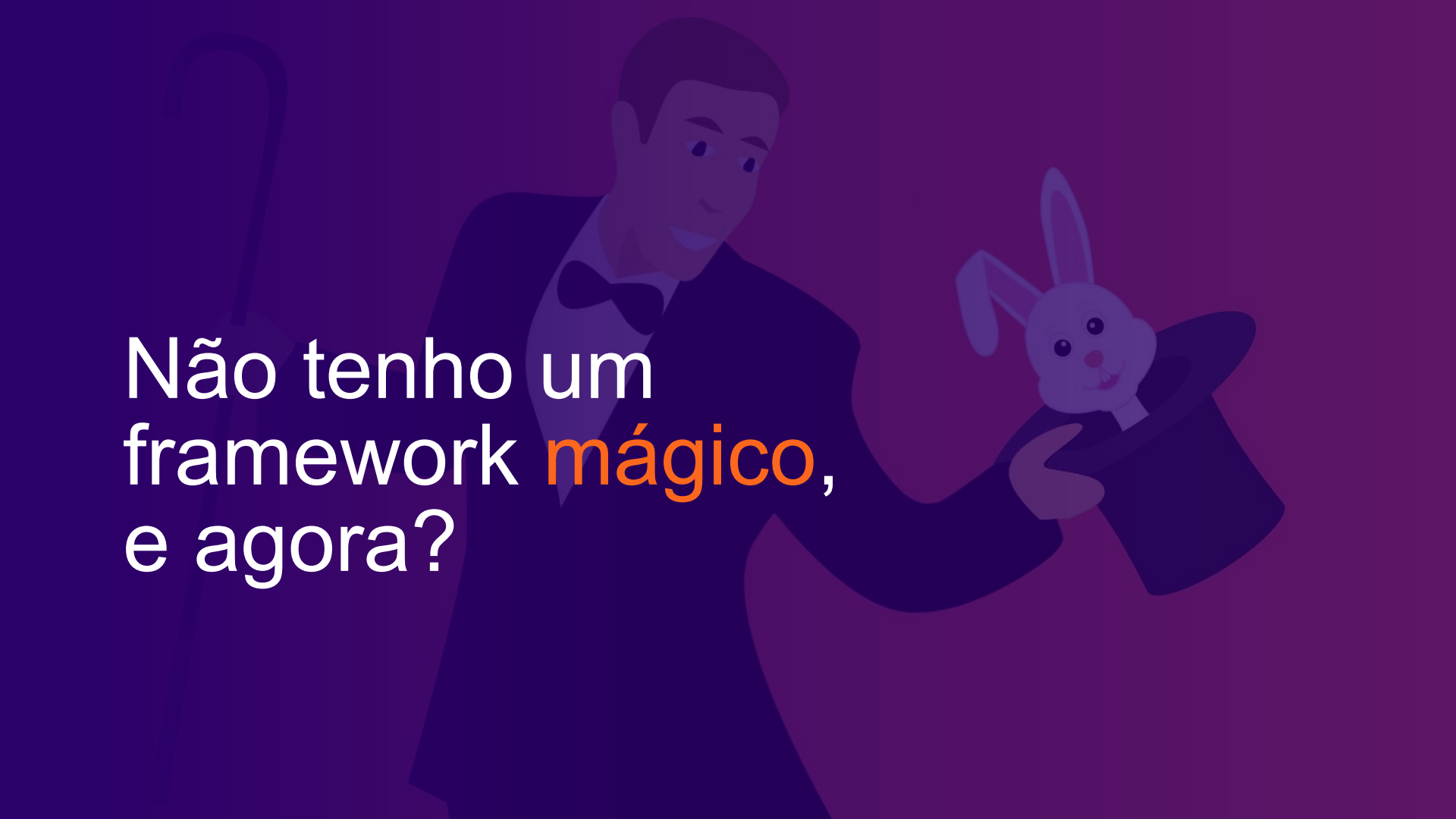
Aspecto	End-to-end	Integrado
Tempo de execução	~30 segundos	~1 segundo
Tempo de feedback ao time	~10 minutos	~2 minutos
Tempo de desenvolvimento	~40 minutos	~10 minutos
Resistência à refatorações	Baixa	Alta
Possibilidade de falsos positivos ou negativos	Muito alta	Baixa

Abordagem de teste direta

```
class SendCommunique():
    def execute(self, **kwargs):
        title = kwargs.get('title')
        message = kwargs.get('message')
        user = kwargs.get('user')

        staff_member = StaffMember.objects.get(user=user)
        new_communique = Communique.create(title, message, staff_member)
        new_communique.save()

        return { 'id': new_communique.pk }
```

Abordagem de teste direta

```python
class SendCommuniqueTests(TestCase):
    def setUp(self):
        user = User.objects.create_user(username='professor')
        user.save()

        StaffMember.objects.create(user=user).save()

        self.command = SendCommunique()

    def test_that_should_create_a_new_communique(self):
        command_arguments = {
            'title': 'Random title',
            'message': 'Hi everyone, this week...',
            'user': User.objects.get(username='professor')
        }

        return_data = self.command.execute(**self.command_arguments)

        new_communique = Communique.objects.get(pk=return_data['id'])
        self.assertIsNotNone(new_communique)
```


Não tenho um
framework mágico,
e agora?

2.
Conheça sua arquitetura

Abordagem de teste direta

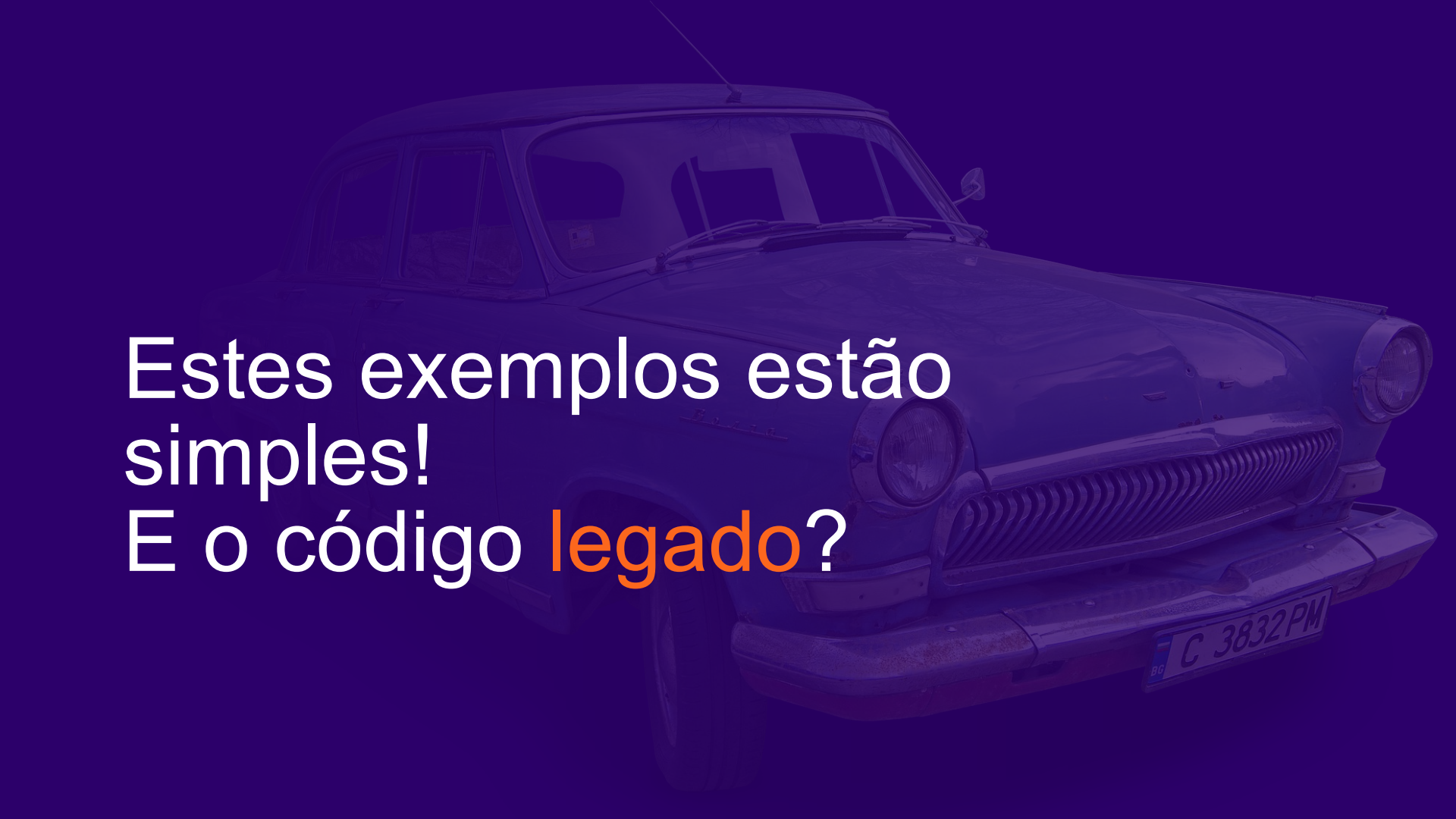
```
public class Orders : IOrders
{
    private readonly DatabaseConnection _databaseConnection;

    public Orders(DatabaseConnection databaseConnection)
    {
        _databaseConnection = databaseConnection;
    }

    public void Save(Order order)
    {
        using var connection = _databaseConnection.Open();
        connection.Execute("INSERT INTO 'Order' (Id, CustomerId, State) " +
            "VALUES (@Id, @CustomerId, @State)", new OrderPersistenceModel
            {
                Id = order.Id,
                CustomerId = order.Customer.Id,
                State = (int)order.State
            });
    }
```

Abordagem de teste direta

```
public class DatabaseConnection
{
    private readonly DatabaseConfig _databaseConfig;

    public DatabaseConnection(DatabaseConfig databaseConfig)
    {
        _databaseConfig = databaseConfig;
    }

    public DbConnection Open()
    {
        return new SQLiteConnection(_databaseConfig.ConnectionString);
    }
}
```

Abordagem de teste direta

```
[Fact]
public void Should_save_a_new_order()
{
    var databaseConnection = new DatabaseConnection(new DatabaseConfig
    {
        ConnectionString = "Data Source=database.sqlite"
    });
    var orders = new Orders(databaseConnection);
    var order = OrderBuilder.New().WithState(OrderState.Placed).Build();

    orders.Save(order);

    var recentlyAddedOrder = orders.GetById(order.Id);
    Assert.Equal(order.Id, recentlyAddedOrder.Id);
    Assert.Equal(OrderState.Placed, recentlyAddedOrder.State);
}
```



Estes exemplos estão simples!
E o código legado?

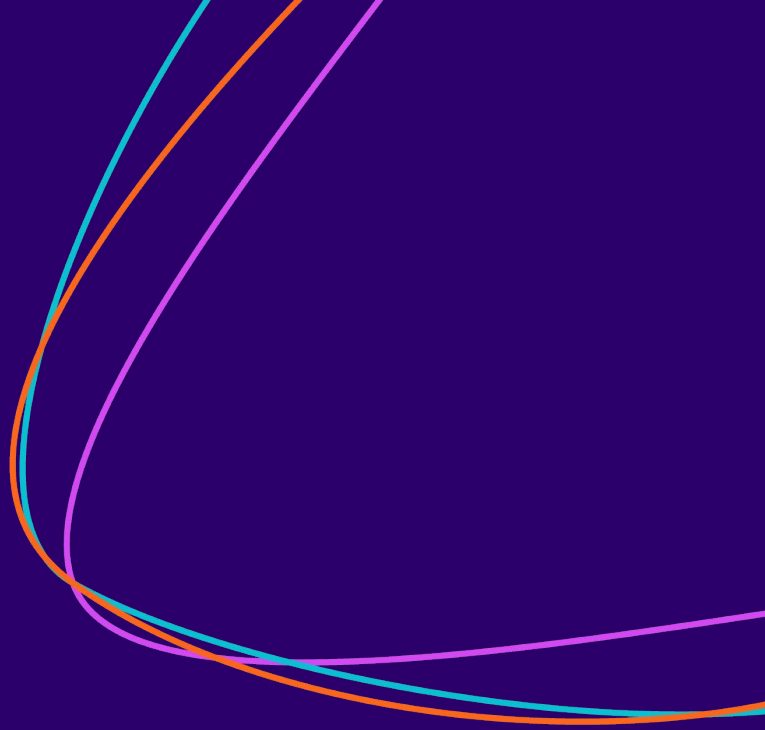
Testando código legado

```
public InscricaoBo(IProtocoloDao protocoloDao, IPessoaDao pessoaDao, IComposicaoRendaDao composicaoRendaDao,
    IPessoaProtocoloDao pessoaProtocoloDao, IContatoDao contatoDao, IUsuarioLogadoBo usuarioLogadoBo,
    IComposicaoRendaBo composicaoRendaBo, IImovelDao imovelDao, IJuncaoProtocoloDao juncaoProtocoloDao,
    ISinalizarMenu sinalizarMenu, IGeradorDeProtocolo geradorDeProtocolo, ITipoDeLogradouroDao tipoDeLogradouroDao,
    ICidadeDao cidadeDao, IBairroDao bairroDao, IEtniaDao etniaDao, INacionalidadeDao nacionalidadeDao, IReligiaoDao
    IMaterialDeMoradiaDao materialDeMoradiaDao, IGrauDeInstrucaoDao grauDeInstrucaoDao, IProfissaoDao profissaoDao,
    ICondicaoDeMoradiaDao condicaoDeMoradiaDao, IMeioTrasporteDao meioDeTrasporteDao, IProtocoloBo protocoloBo,
    IValidadorDeCpfDePretendente validadorDeCpfDePretendente, IRegimeDeCasamentoDao regimeDeCasamentoDao,
    IGrupoEtnicoDao grupoEtnicoDao, IUfDao ufDao, ITipoDeEmancipacaoDao tipoEmancipacaoDao, ICidDao cidDao,
    IValidadorDeCid validadorDeCid, IDistritoDao distritoDao,
    IAtualizacaoDeProtocolosIncompletos atualizacaoDeProtocolosIncompletos,
    IServicoDeAplicacao<AtualizacaoDeProtocoloComSelecaoCmd> atualizacaoDeProtocoloComSelecao,
    ISeparacaoProtocoloDao separacaoProtocoloDao, IValidacaoDeSeparacao validacaoDeSeparacao,
    IValidacaoDeJuncao validacaoDeJuncao, IPaisDao paisDao, ILogsDeEventos logsDeEventos,
    IProgramaSocialProtocoloDao programaSocialProtocoloDao,
    IValidacaoDePermissaoParaAlteracaoDeProtocolos validacaoDePermissaoParaAlteracaoDeProtocolos,
    IServicoDeCriacaoDeImovel servicoDeCriacaoDeImovel,
    IVerificadorDePermissaoParaAlteracaoDeProtocolos verificadorDePermissaoParaAlteracaoDeProtocolos,
    ILogDeComunicacaoDao logDeComunicacaoDao,
    IEmpreendimentoDao empreendimentoDao)
```


Ainda é possível
otimizar resultados

3.

Reduza complexidade



Reduzindo complexidade

```
public abstract class IntegrationTestBase : IDisposable
{
    private ModuleComposer _moduleComposer;

    public IntegrationTestBase()
    {
        _moduleComposer = ModuleComposer
            .Compose(new CompositionConfig
            {
                ForTesting = true
            })
            .AddCustomServices(new List<Type>
            {
                typeof(Orders),
                typeof(Clientele)
            })
            .FinishComposition();
    }

    protected T GetService<T>() where T : class
    {
        return _moduleComposer.GetService<T>();
    }
```

Reduzindo complexidade

```
public class OrdersTests : IntegrationTestBase
{
    [Fact]
    public void Should_get_an_order()
    {
        var orders = GetService<IOrders>();
        var order = OrderBuilder.New().WithState(OrderState.New).Build();
        orders.Save(order);

        var recentlyAddedOrder = orders.GetById(order.Id);

        Assert.Equal(order.Id, recentlyAddedOrder.Id);
        Assert.Equal(OrderState.New, recentlyAddedOrder.State);
    }
```

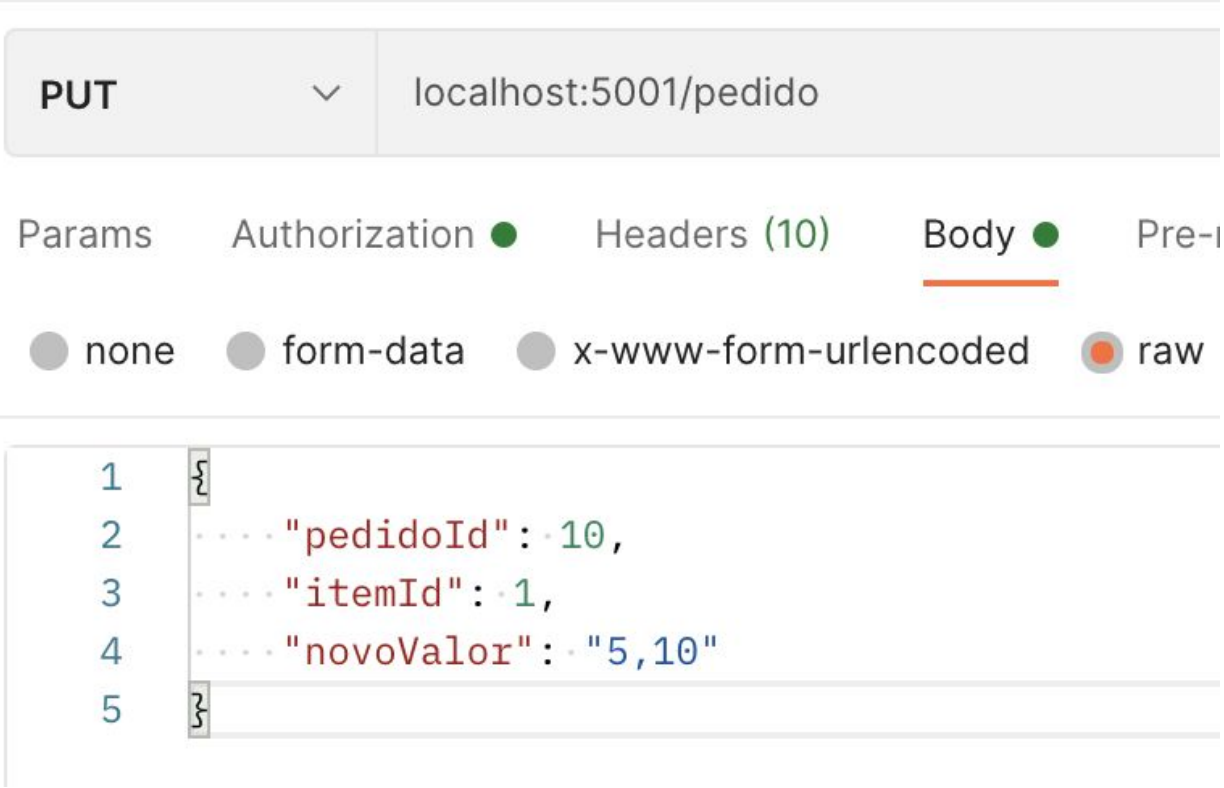
Reduzindo complexidade

```
public class ClienteleTests : IntegrationTestBase
{
    [Fact]
    public void Should_get_a_customer()
    {
        var clientele = GetService<IClientele>();
        var customer = CustomerBuilder.New().Build();
        clientele.Save(customer);

        var foundCustomer = clientele.GetById(customer.Id);

        Assert.Equal(customer.Id, foundCustomer.Id);
        Assert.Equal(customer.Name, foundCustomer.Name);
    }
}
```


Continue otimizando!

Outras formas de testes de integração

Testando requisições



Testando requisições

```
public class DecimalModelBinder : IModelBinder
{
    public object BindModel(ModelBindingContext bindingContext)
    {
        var valueResult = bindingContext
            .ValueProvider.GetValue(bindingContext.ModelName);
        var modelState = new ModelState { Value = valueResult };

        object actualValue = null;

        try
        {
            actualValue = Convert.ToDecimal(
                valueResult.AttemptedValue, CultureInfo.CurrentCulture);
        }
        catch (FormatException e)
        {
            modelState.Errors.Add(e);
        }

        bindingContext.ModelState.Add(
            bindingContext.ModelName, modelState);

        return actualValue;
    }
}
```

Testando requisições

```
public class Startup
{
    public Startup(IConfiguration configuration)
    {
        Configuration = configuration;
    }

    public IConfiguration Configuration { get; }

    public void ConfigureServices(IServiceCollection services)
    {
        services.AddControllers();
```

Testando requisições

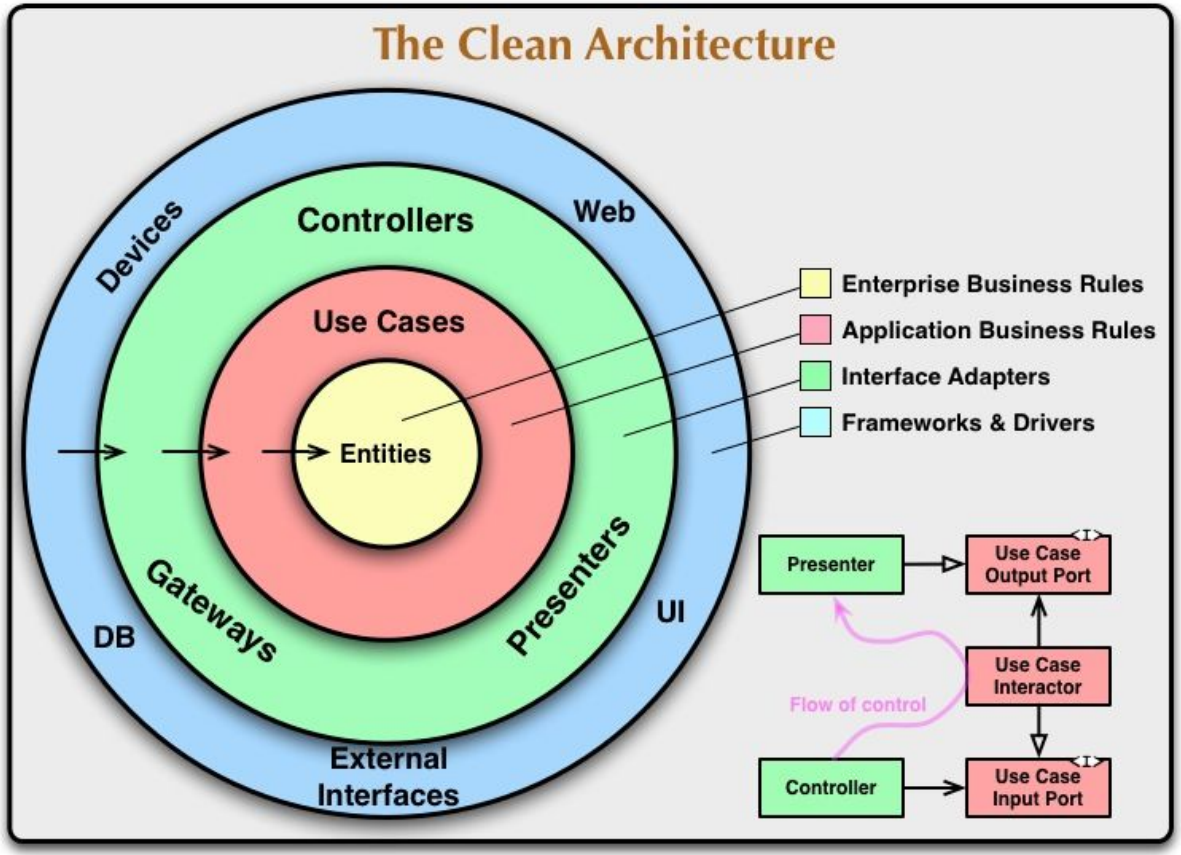
```
[Fact]
public async Task Should_put_a_valid_decimal_value()
{
    var testServer = new TestServer(new WebHostBuilder()
        .UseStartup<Startup>());
    var httpClient = testServer.CreateClient();
    var content = new HttpStringContent(
        $"{{ \"pedidoId\": 10,\"itemId\": 1,\"novoValor\":\"5,10\"}}",
        Encoding.UTF8, "application/json");

    var response = await httpClient
        .PutAsync(@"http://localhost:5001/pedido", content);

    var responseString = await response.Content.ReadAsStringAsync();
    Assert.Equal("{\"id\":1,\"valor\":5.10}", responseString);
}
```

Testando requisições

```java
@Test
public void deveAtualizarUmDocumento() {
    ByteBuffer novosDadosDoArquivo = ByteBuffer
            .wrap(new byte[]{89, 14, 5, 79, 12, 25});
    Documento documentoArmazenado = new DocumentoBuilder().criar();
    documentoRepositorioDaInfra.adicionar(documentoArmazenado);
    HttpEntity<MultiValueMap<String, Object>> request =
            montarRequisicaoParaAtualizarArquivo(novosDadosDoArquivo);

    this.restTemplate.put( url: RECURSO + "/{id}",
            request, documentoArmazenado.toString());

    ByteBuffer dadosDoDocumentoAlterado =
            documentoRepositorioDaInfra.obterArquivo(documentoArmazenado.id());
    Assertions.assertThat(dadosDoDocumentoAlterado)
            .isEqualTo(novosDadosDoArquivo);
}
```

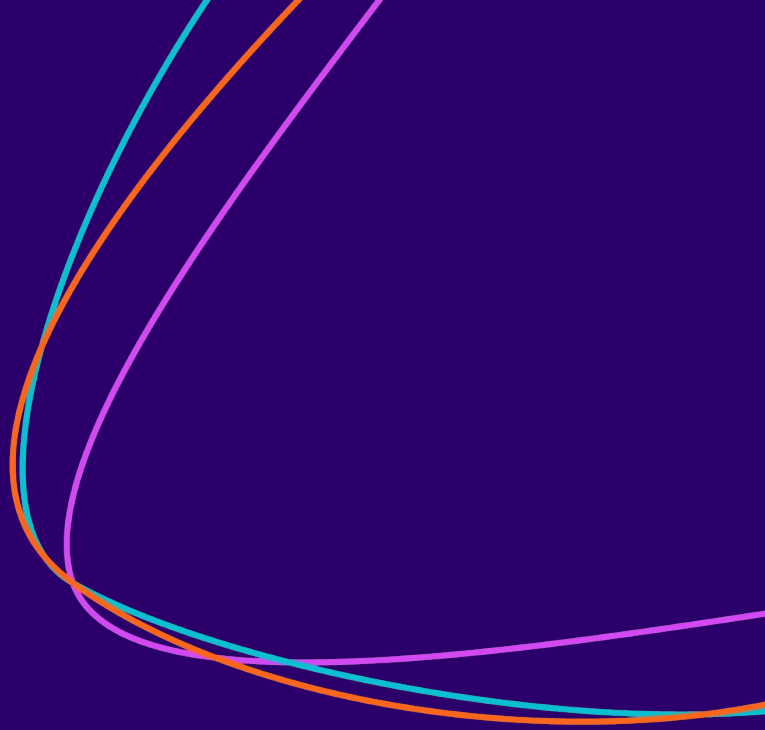
Um teste de requisição seria um teste end-to-end?

Testes
end-to-end

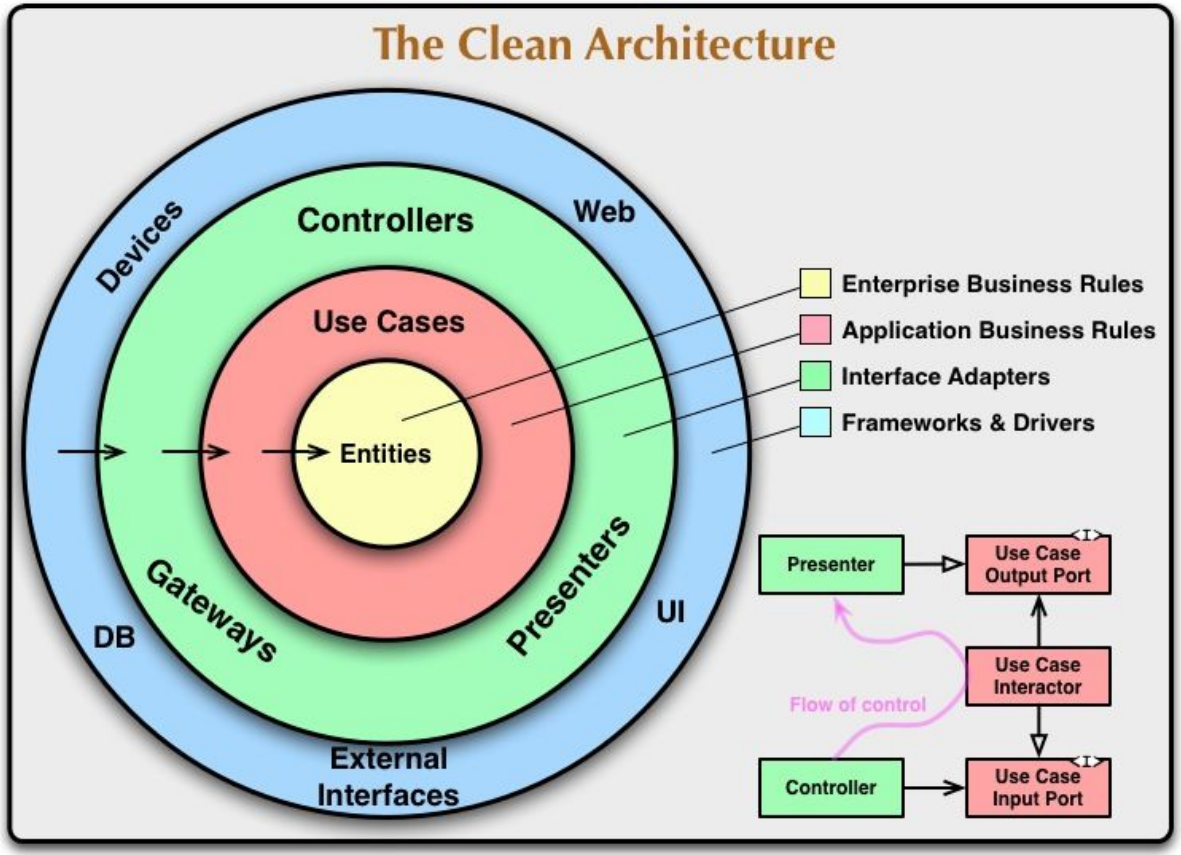


4.

Entenda o resultado desejado



Focando no que importa



Focando no que importa

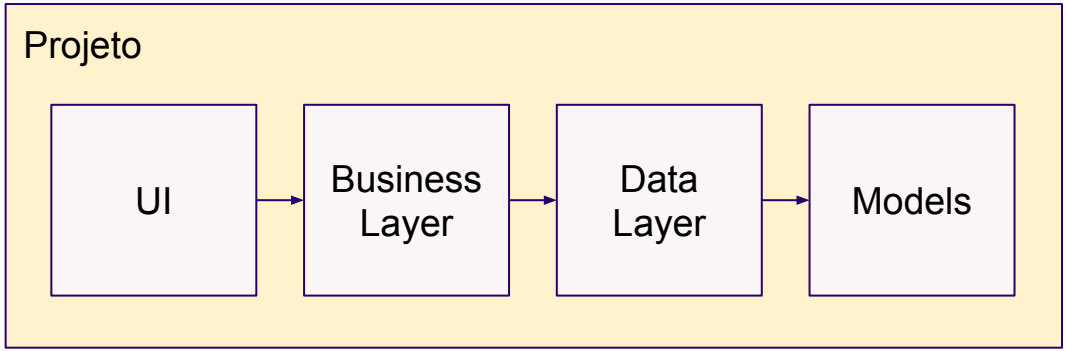
```java
public class ArmazenaDocumentosTest extends TesteDeIntegracao {
    @Autowired
    private ArmazenaDocumentos armazenaDocumentos;

    @Autowired
    private DocumentoRepositorioDaInfra documentoRepositorioDaInfra;

    @Test
    public void deveArmazenarUmDocumento() {
        Documento documentoParaArmazenar =
                new DocumentoBuilder().criar();
        ArmazenarDocumentos comando = ArmazenarDocumentos
                .criar(documentoParaArmazenar);

        armazenaDocumentos.executar(comando);

        ByteBuffer arquivoConsultado = documentoRepositorioDaInfra
                .obterArquivo(documentoParaArmazenar.id());
        assertEquals(arquivoConsultado, documentoParaArmazenar.arquivo());
    }
}
```

Lidando com o legado

Testando código legado

```
public InscricaoBo(IProtocoloDao protocoloDao, IPessoaDao pessoaDao, IComposicaoRendaDao composicaoRendaDao,
    IPessoaProtocoloDao pessoaProtocoloDao, IContatoDao contatoDao, IUsuarioLogadoBo usuarioLogadoBo,
    IComposicaoRendaBo composicaoRendaBo, IImovelDao imovelDao, IJuncaoProtocoloDao juncaoProtocoloDao,
    ISinalizarMenu sinalizarMenu, IGeradorDeProtocolo geradorDeProtocolo, ITipoDeLogradouroDao tipoDeLogradouroDao,
    ICidadeDao cidadeDao, IBairroDao bairroDao, IEtniaDao etniaDao, INacionalidadeDao nacionalidadeDao, IReligiaoDao
    IMaterialDeMoradiaDao materialDeMoradiaDao, IGrauDeInstrucaoDao grauDeInstrucaoDao, IProfissaoDao profissaoDao,
    ICondicaoDeMoradiaDao condicaoDeMoradiaDao, IMeioTrasporteDao meioDeTrasporteDao, IProtocoloBo protocoloBo,
    IValidadorDeCpfDePretendente validadorDeCpfDePretendente, IRegimeDeCasamentoDao regimeDeCasamentoDao,
    IGrupoEtnicoDao grupoEtnicoDao, IUfDao ufDao, ITipoDeEmancipacaoDao tipoEmancipacaoDao, ICidDao cidDao,
    IValidadorDeCid validadorDeCid, IDistritoDao distritoDao,
    IAtualizacaoDeProtocolosIncompletos atualizacaoDeProtocolosIncompletos,
    IServicoDeAplicacao<AtualizacaoDeProtocoloComSelecaoCmd> atualizacaoDeProtocoloComSelecao,
    ISeparacaoProtocoloDao separacaoProtocoloDao, IValidacaoDeSeparacao validacaoDeSeparacao,
    IValidacaoDeJuncao validacaoDeJuncao, IPaisDao paisDao, ILogsDeEventos logsDeEventos,
    IProgramaSocialProtocoloDao programaSocialProtocoloDao,
    IValidacaoDePermissaoParaAlteracaoDeProtocolos validacaoDePermissaoParaAlteracaoDeProtocolos,
    IServicoDeCriacaoDeImovel servicoDeCriacaoDeImovel,
    IVerificadorDePermissaoParaAlteracaoDeProtocolos verificadorDePermissaoParaAlteracaoDeProtocolos,
    ILogDeComunicacaoDao logDeComunicacaoDao,
    IEmpreendimentoDao empreendimentoDao)
```

Testando código legado



Testando código legado

```
  42      public class InscricaoBo : IInscricaoBo
  43  >   {…
1123      }
```

Testando código legado

```
public Protocolo Selecionar(int codProtocolo)
{
    const string sql =
        @"SELECT * FROM Protocolo WHERE codProtocolo = @codProtocolo ";
    Protocolo protocolo;

    using (var cmd = (SqlCommand)Conexao.CreateCommand())
    {
        cmd.CommandText = sql;
        cmd.Parameters.Add("@codProtocolo",
            SqlDbType.Int).Value = codProtocolo;
        protocolo = Load(cmd);
    }

    return protocolo;
}
```

Testando código legado

```
public class Protocolo
{
    public int CodProtocolo { get; set; }
    public int CodLocalAtendimentoInscricao { get; set; }
    public int CodLocalAtendimentoUltimaAtualizacao { get; set; }
    public int CodCidadePretensao { get; set; }
    public int CodUsuarioInclusao { get; set; }
    public int CodUsuarioAlteracao { get; set; }
    public string NumeroDoProtocolo { get; set; }
    public DateTime? DataInscricao { get; set; }
    public DateTime? DataUltimaAtualizacao { get; set; }
    public DateTime? DataFim { get; set; }
    public bool? DesejaParticiparDoLoteamentoIpe { get; set; }
    public bool? DesejaParticiparDoLoteamentoElioFernando { get; set; }
```

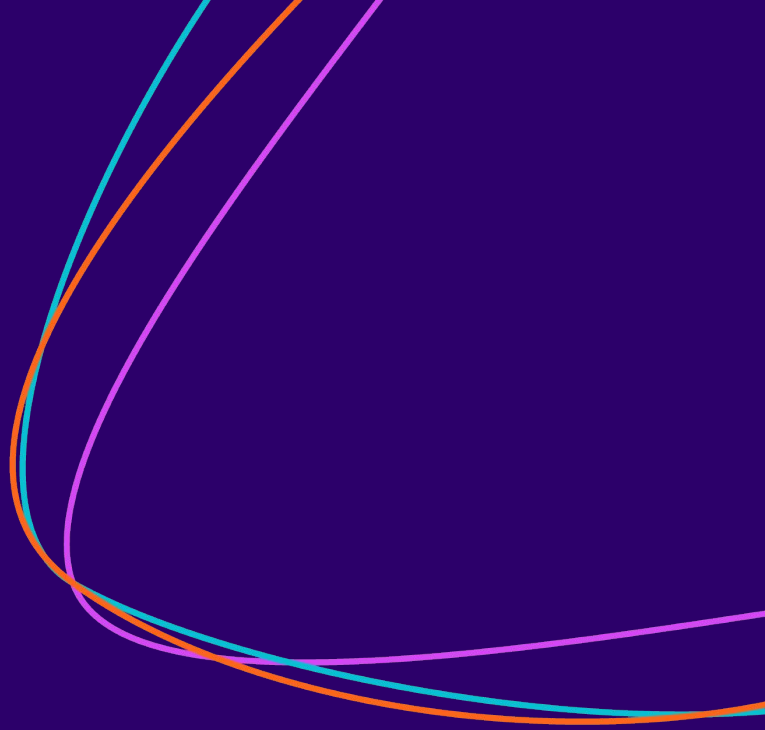
Testando código legado

Testando código legado

```
var conexao = new AdoConnection(new SqlConnection
{
    ConnectionString = ConfigurationManager
        .ConnectionStrings["NomeDaConexao"].ConnectionString
});

// Outros processamentos aqui
if (listaImoveis.Any())
{
    // listaImoveis = List<Imovel>
    foreach (var imovelAux in listaImoveis)
    {
        using (var cmd = (SqlCommand)conexao.CreateCommand())
        {
            // Erro na sintaxe de update
            cmd.CommandText = @"UPDATE Imovel
                SET cod1Protocolo = @codProtocolo
                WHERE codImovel = @codImovel ";
            cmd.Parameters.Add("@codImovel", SqlDbType.Int).Value = imovelAux.CodImovel;
            cmd.Parameters.Add("@codProtocolo", SqlDbType.Int).Value = codProtocoloNovo;
            cmd.ExecuteNonQuery();
        }
    }
}
// Outros processamentos aqui
```

5.

Isole o que deve ser testado



Testando código legado

```
var conexao = new AdoConnection(new SqlConnection
{
    ConnectionString = ConfigurationManager
        .ConnectionStrings["NomeDaConexao"].ConnectionString
});

// Outros processamentos aqui
if (listaImoveis.Any())
{
    // listaImoveis = List<Imovel>
    foreach (var imovelAux in listaImoveis)
    {
        using (var cmd = (SqlCommand)conexao.CreateCommand())
        {
            // Erro na sintaxe de update
            cmd.CommandText = @"UPDATE Imovel
                SET cod1Protocolo = @codProtocolo
                WHERE codImovel = @codImovel ";
            cmd.Parameters.Add("@codImovel", SqlDbType.Int).Value = imovelAux.CodImovel;
            cmd.Parameters.Add("@codProtocolo", SqlDbType.Int).Value = codProtocoloNovo;
            cmd.ExecuteNonQuery();
        }
    }
}
// Outros processamentos aqui
```

Testando código legado

```
public class ImovelDao
{
    private readonly IDbConnection _conexao;

    public ImovelDao(IDbConnection conexao)
    {
        _conexao = conexao;
    }

    public void Atualizar(int codImovel, int codProtocolo)
    {
        using (var cmd = (SqlCommand)_conexao.CreateCommand())
        {
            // Erro na sintaxe de update
            cmd.CommandText = @"UPDATE Imovel
                SET cod1Protocolo = @codProtocolo
                WHERE codImovel = @codImovel ";
            cmd.Parameters.Add("@codImovel", SqlDbType.Int).Value = codImovel;
            cmd.Parameters.Add("@codProtocolo", SqlDbType.Int).Value = codProtocolo;
            cmd.ExecuteNonQuery();
        }
    }
```

Testando código legado

```
[Fact]
public void Deve_atualizar_um_imovel()
{
    var novoCodProtocolo = "100";
    var conexao = new AdoConnection(new SqlConnection
    {
        ConnectionString = "Data Source=database.sqlite"
    });
    var imovelDao = new ImovelDao(conexao);
    var imovelCriado = new ImovelBuilder().criar();
    imovelDao.Salvar(imovelCriado);

    imovelDao.Atualizar(imovelCriado.CodImovel, novoCodProtocolo);

    Imovel imovelAlterado = imovelDao.Obter(imovelCriado.CodImovel);
    Assert.Equal(novoCodProtocolo, imovelAlterado.CodProtocolo);
}
```

Testando código legado

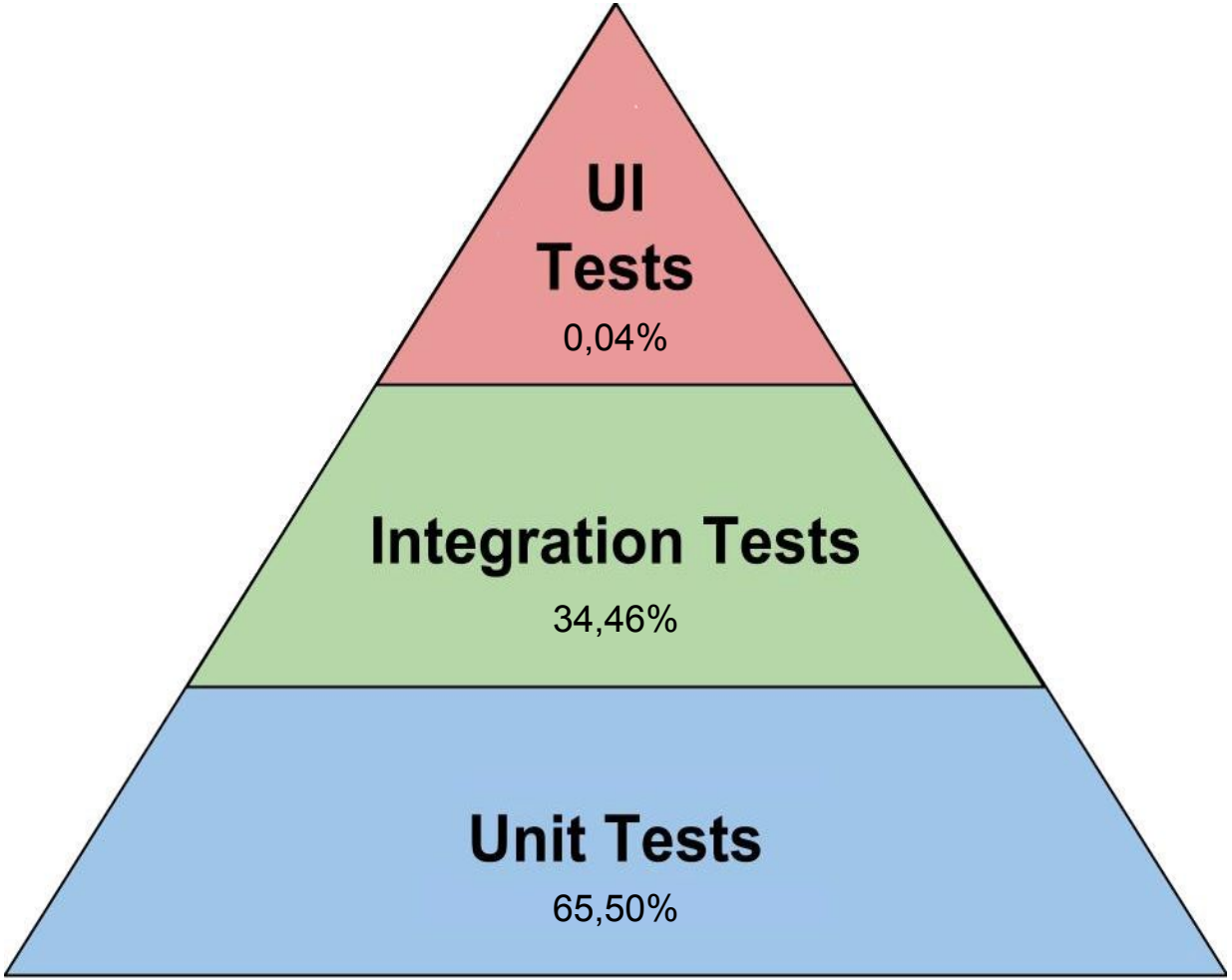
```
var conexao = new AdoConnection(new SqlConnection
{
    ConnectionString = ConfigurationManager
        .ConnectionStrings["NomeDaConexao"].ConnectionString
});

// Outros processamentos aqui
if (listaImoveis.Any())
{
    var imovelDao = new ImovelDao(conexao);

    foreach (var imovelAux in listaImoveis)
        imovelDao.Atualizar(imovelAux.CodImovel, codProtocoloNovo);
}
// Outros processamentos aqui
```

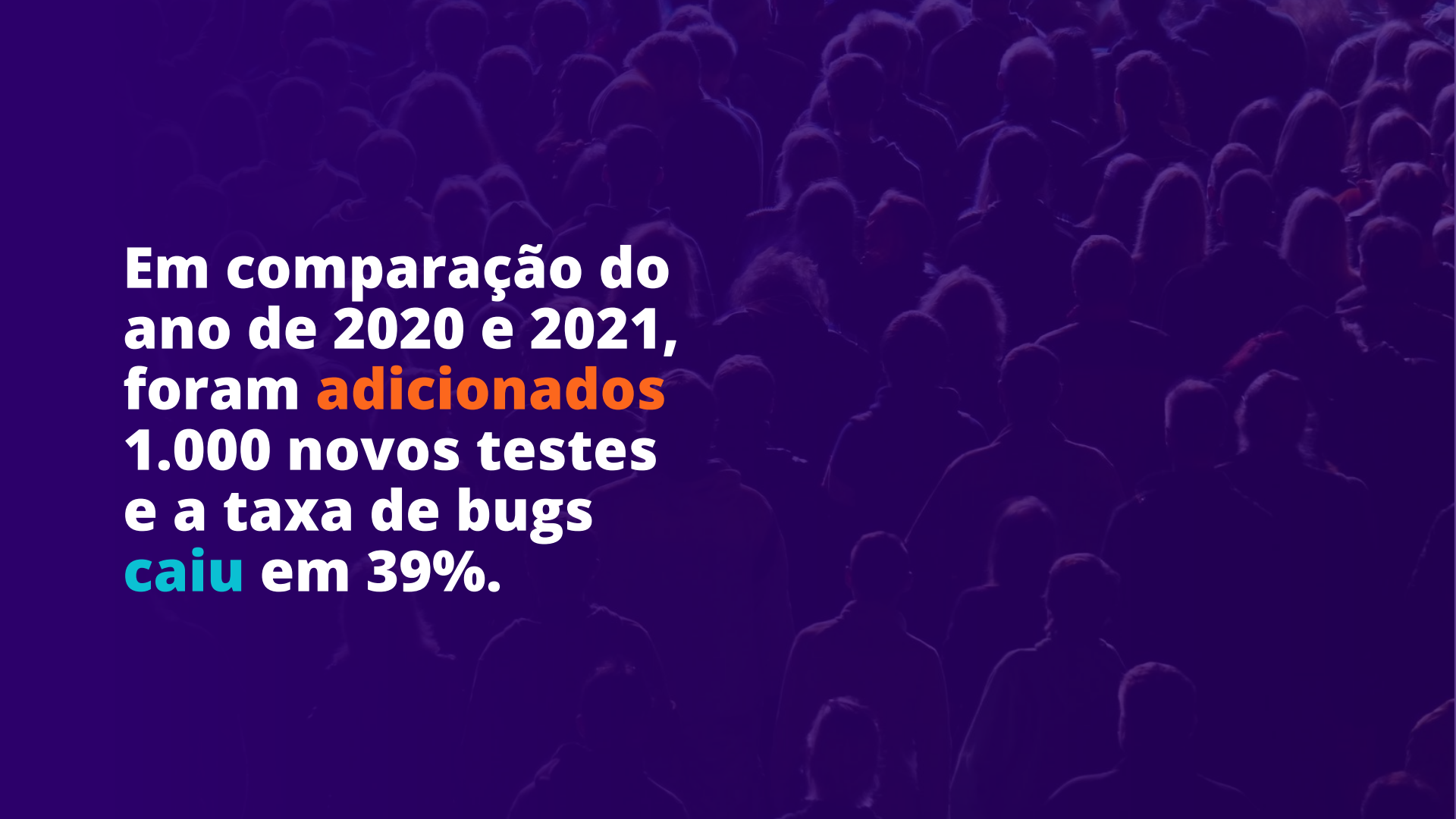
Bancos em memória ou bancos reais?

Quando usar cada um deles

Cenário	Em memória	Real
Queries complexas (UNION, PIVOT e outros)	❌	✅
Acesso à dados com ORM	✅	✅
Uso de sintaxes exclusivas do banco real	❌	✅
Transação importa no cenário de testes	❌	✅
Testes que envolvem performance	❌	✅
API compatível entre o banco real e o em memória	✅	✅
Testes que não têm como foco o acesso aos dados	✅	✅

Proporções




Em comparação do ano de 2020 e 2021, foram adicionados 1.000 novos testes e a taxa de bugs caiu em 39%.

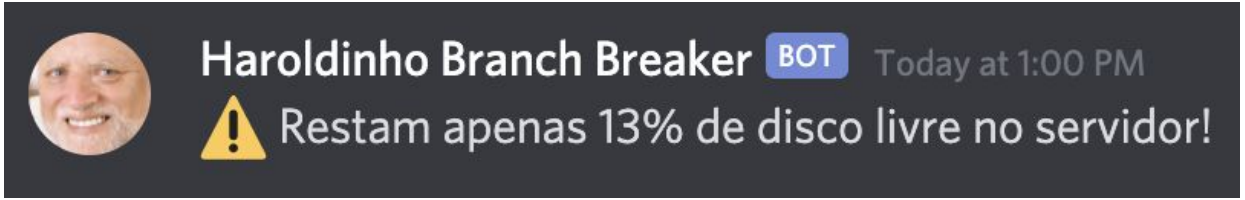
Indo além


```
"diskSpace": {
    "status": "UP",
    "details": {
        "total": 18238930944,
        "free": 14711382016,
        "threshold": 10485760
    }
},
"ping": {
    "status": "UP"
}
}
```

Indo além

```
async function analisarUsoDeMemoria(contexto, informacoesParaColetaDeDados) {
    try {
        const informacoesDoProduto = await coletorDeMetricas
            .coletar(contexto, informacoesParaColetaDeDados.urls);
        informacoesParaColetaDeDados.adaptadorDosDadosDoProduto
            .adicionarMetricas(informacoesDoProduto);

        await enviarMensagemSeNecessario(contexto,
            informacoesParaColetaDeDados.urlDoWebhook,
            informacoesParaColetaDeDados.adaptadorDosDadosDoProduto);
    } catch(erro) {
        contexto.log(erro);
        await enviadorDeMensagens.enviarFalha(contexto, informacoesParaColetaD
        throw erro;
    }
```

Indo além


Haroldinho Branch Breaker
BOT
Today at 1:00 PM
Restam apenas 13% de disco livre no servidor!

Diversos tipos de testes

Ainda dá pra garantir ainda mais aspectos
do software com outras técnicas de testes, tais como:

1. Testes mutantes;
2. Testes de contrato;
3. Testes de unidade no front-end;
4. Entre outros testes.

www.digix.com.br