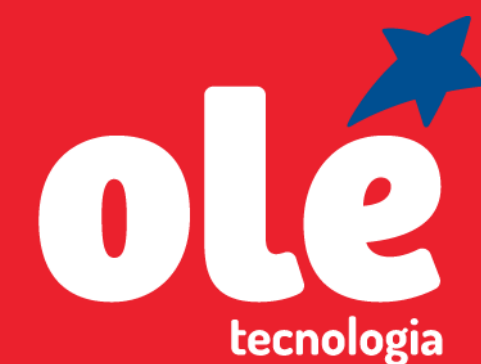


Arquitetura de Soluções

Uma empresa



#ArquiteturaDeSoluções

Exceções



- O que são exceções?
- Boas práticas para tratamento de exceções.
- É uma boa prática lançar exceções ou deve ser evitado?





O que são exceções ?

Sob ponto de vista técnico, é um mecanismo de desvio de fluxo de execução.

Ao lançar uma exceção (*throw*), o fluxo é desviado para o bloco de captura (*catch*) mais próximo (com assinatura compatível).

[illegible]

Evitar:

```
try
{
    ...
}
catch (Exception e)
{
    throw e;
}
```

[illegible]

Evitar:

```
try
{
    ...
}
catch (Exception e)
{
    throw new Exception(e.Message);
}
```

Evitar:

```
try
{
    ...
}
catch (Exception e)
{
    // Fluxo alternativo para erros em geral.
}
```




Boas práticas para tratamento de exceções.

Evitar:

```
try  
{  
    ...  
}  
catch {}
```



[illegible]

Boas práticas para tratamento de exceções.

```
try
{
    ...
}
catch (AssinaturaCoreException e)
{
    // Fluxo alternativo para problemas relacionados à
    // assinatura.
}
catch (PagamentoCoreException e)
{
    // Fluxo alternativo para problemas relacionados à
    // pagamento.
}
catch (Exception e)
{
    // Fluxo alternativo para erros em geral.
}
```


[illegible]

Boas práticas para tratamento de exceções.

```
try
{
    ...
}
catch (AssinaturaCoreException e)
{
    // Fluxo alternativo para problemas relacionados à
    // assinatura.
}
catch (PagamentoCoreException e)
{
    // Fluxo alternativo para problemas relacionados à
    // pagamento.
}
catch (Exception e)
{
    ...
}
```



Boas práticas para tratamento de exceções.

```
try
{
    var transactionId = Retirar(contaOrigem, valor);
    Depositar(contaDestino, valor);
}
catch (Exception e)
{
    RollbackTransaction(transactionId);

    throw;
}
```

Diferente de “**throw** e;” a pilha original é preservada (particularidade C#)

[illegible]

Boas práticas para tratamento de exceções.

```
try
{
    var transactionId = Retirar(contaOrigem, valor);
    Depositar(contaDestino, valor);
}
catch (Exception e)
{
    RollbackTransaction(transactionId);

    throw new TransacaoBancariaException(
        message: $"Erro ao realizar transferência {transactionId}"
        innerException: e);
}
```

“Exceções devem ser utilizadas somente em situações excepcionais.”

Uma empresa Santander

“Exceções devem ser utilizadas somente em situações excepcionais.”

O que é uma situação excepcional ?

O que é uma situação excepcional ?

- Precisamos de uma definição rigorosa, objetiva e mensurável para erros. Também precisamos de orientação sobre onde eles devem ser relatados e tratados.
- Precisamos de garantias de segurança claras e compreensíveis que possamos usar para descrever o tratamento de erros.
- Finalmente, precisamos ver quais erros devem ser relatados e tratados usando exceções.

Herb Sutter (2004) - <http://www.drdobbs.com/when-and-how-to-use-exceptions/184401836>



O que é uma situação excepcional ?

- Uma condição que impede que o método atenda a uma condição prévia (por exemplo, uma restrição de parâmetro) de outro método que deve ser chamado.
- Uma condição que impede que o método estabeleça uma de suas próprias pós-condições. Se o método tiver um valor de retorno, produzir um objeto de valor de retorno válido é uma pós-condição.
- Uma condição que impede que o método restabeleça uma invariável que é responsável por manter.

Herb Sutter (2004) - <http://www.drdobbs.com/when-and-how-to-use-exceptions/184401836>



O que é uma situação excepcional ?

Precondition

- Uma condição que impede que o método atenda a uma condição prévia (por exemplo, uma restrição de parâmetro) de outro método que deve ser chamado.

Postcondition


- Uma condição que impede que o método estabeleça uma de suas próprias pós-condições. Se o método tiver um valor de retorno, produzir um objeto de valor de retorno válido é uma pós-condição.

Invariant

- Uma condição que impede que o método restabeleça uma invariável que é responsável por manter.

Herb Sutter (2004) - <http://www.drdobbs.com/when-and-how-to-use-exceptions/184401836>

Postcondition

- 
- O consumidor não consegue solucionar.

- Registrar o erro ocorrido (logs) e informar ao consumidor que houve um erro interno.
- Um identificador do registro do erro pode ser fornecido ao consumidor para que seja possível a equipe técnica rastrear o erro para fornecer um feedback ao consumidor, caso seja solicitado.



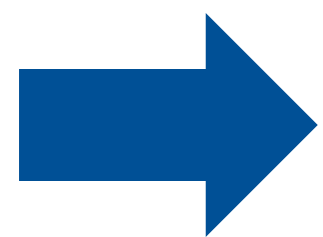
O que é uma situação excepcional ?

Precondition

- Falhas ocasionadas devido aos dados fornecidos; entrada não satisfazem as condições necessárias.

Invariant

- Produzido um estado inválido. Após a execução de uma operação, um modelo de dados terminou preenchido com dados não satisfazem as suas regras de validação.



Erro ~~pode ter sido~~ causado pelo consumidor.

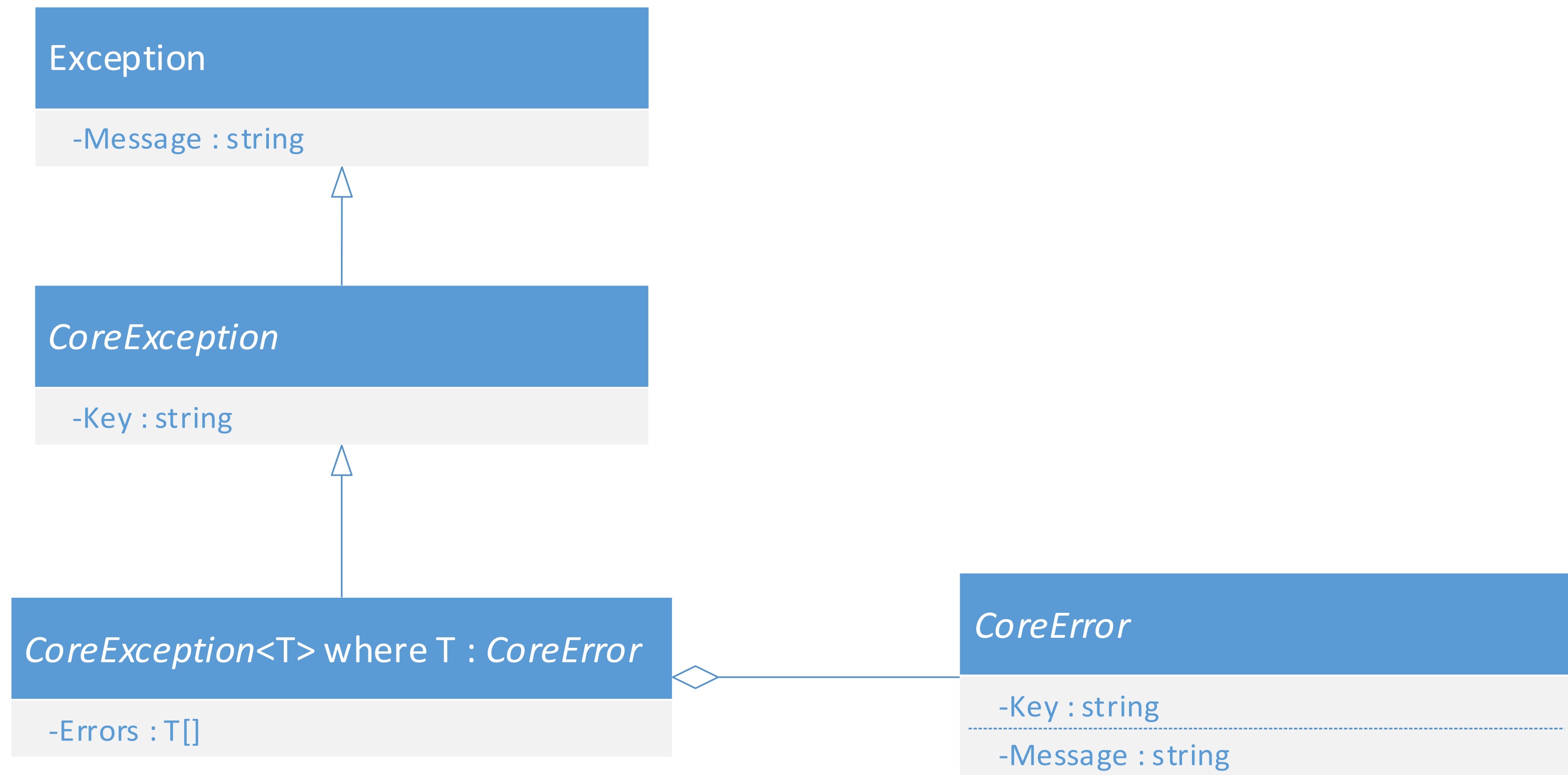
O que fazer?

- Expor para o consumidor todos os detalhes sobre o erro ocorrido para que ele tenha a chance de corrigir os dados e tentar novamente.

[illegible]



Exceções de negócio (*CoreException*)



Exceções de negócio (*CoreException*)

```
public class AssinaturaCoreError : CoreError
{
    public static AssinaturaCoreError NaoExistePagamento =>
        new AssinaturaCoreError("NaoExistePagamento", "Não existe pagamento para essa assinatura");

    public static AssinaturaCoreError AssinaturaNaoEncontrada =>
        new AssinaturaCoreError("AssinaturaNaoEncontrada", "Assinatura não encontrada.");

    public static AssinaturaCoreError ClienteComAssinaturaAtiva =>
        new AssinaturaCoreError("ClienteComAssinaturaAtiva", "Cliente já tem uma assinatura ativa.");

    public static AssinaturaCoreError ClienteSemAssinatura =>
        new AssinaturaCoreError("ClienteSemAssinatura", "Não existem Assinaturas para o Cliente.");

    protected AssinaturaCoreError(string key, string message) : base(key, message)
    {
    }
}
```

Exceções de negócio (*CoreException*)

```
public class AssinaturaCoreException : CoreException<AssinaturaCoreError>
{
    public AssinaturaCoreException() : base()
    {
    }

    public AssinaturaCoreException(params AssinaturaCoreError[] errors)
    {
        AddError(errors);
    }
}
```

```
/// <exception cref="AssinaturaCoreException">Caso a assinatura nao exista.</exception>
public async Task RenovarAssinaturaAsync(Guid assinaturaId)
{
    if (!ExisteAssinatura(assinaturaId))
        throw new AssinaturaCoreException(AssinaturaCoreError.AssinaturaNaoEncontrada);

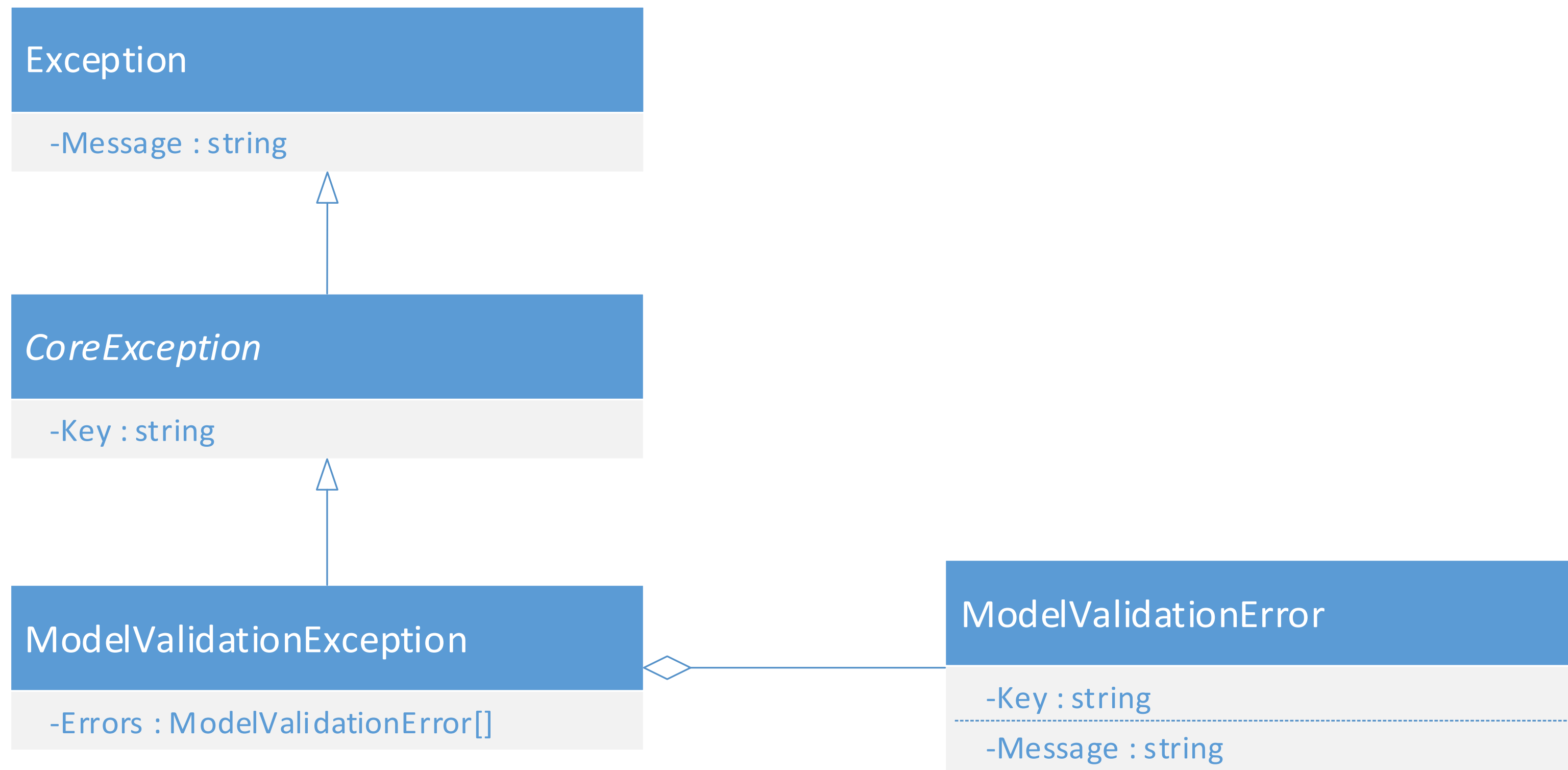
    ...
}
```

```
public class Endereco
{
    [Required(ErrorMessage = "RuaVazio")]
    public string Rua { get; set; }

    [Required(ErrorMessage = "NumeroVazio")]
    public string Numero { get; set; }
}
```


[illegible]

Validação de modelos de dados



Validação de modelos de dados

HTTP 400

```
{
  "key": "ModelValidationException",
  "message": "Modelo de dados inválido.",
  "errors": [
    {
      "key": "RuaVazio",
      "message": "O campo Rua é obrigatório."
    },
    {
      "key": "NumeroVazio",
      "message": "O campo Numero é obrigatório."
    }
  ]
}
```



agile development software

team processes code project waterfall study early progress work business change methods projects rate significant design lifecycle must techniques hand adapting product detailed activity productivity manifesto stakeholders specifications evidence defect valuable knowing teams improving adapting lines may project waterfall study early progress work business change methods projects rate significant design lifecycle must techniques hand adapting product detailed activity productivity manifesto stakeholders specifications evidence defect valuable knowing teams improving adapting lines may

