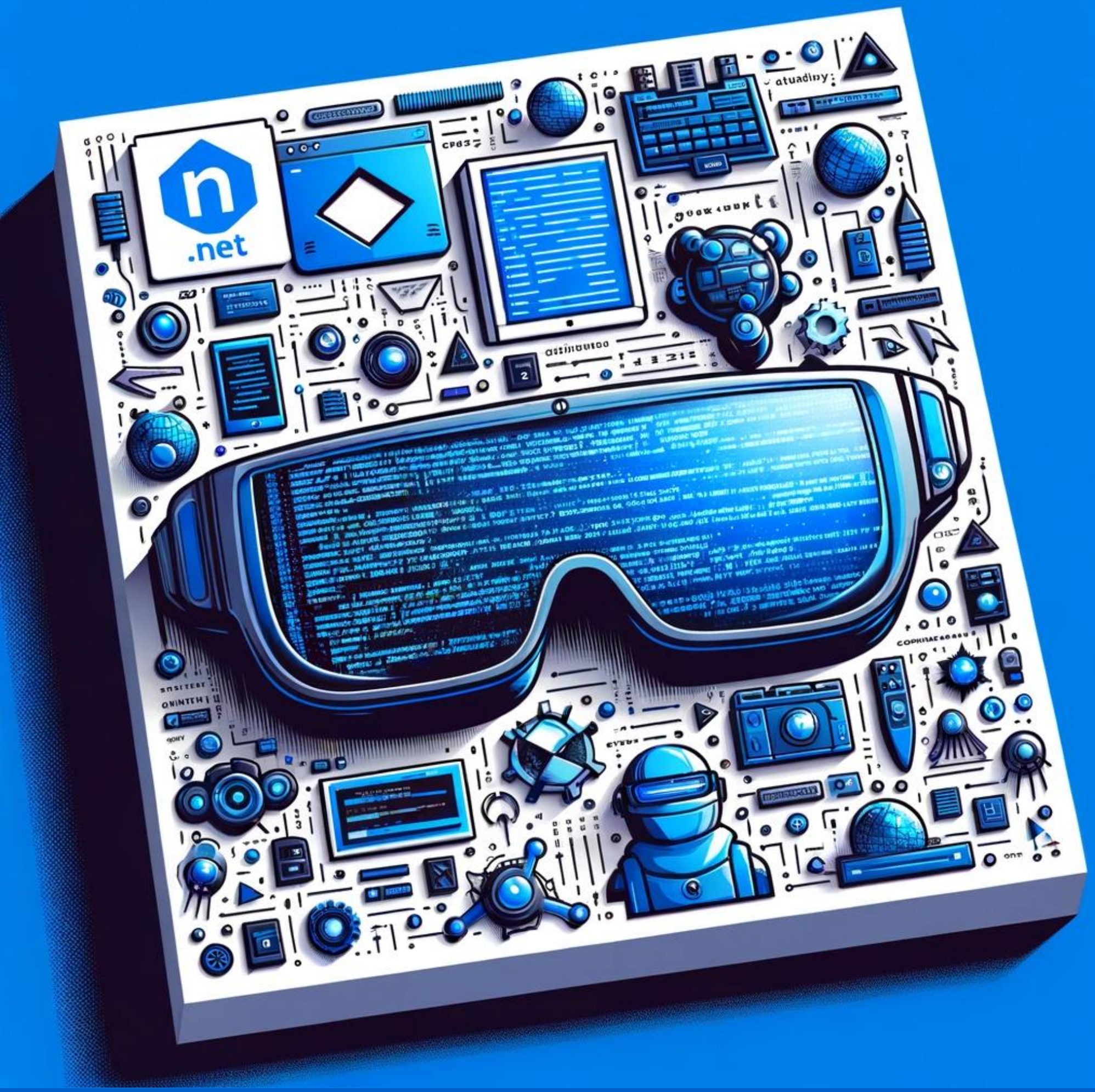
Guia do Código Secreto dos Heróis: Mestre em .NET



ELIANE BARROS

Guia Prático de .NET para Desenvolvimento de Aplicações em .NET

Introdução ao .NET

O que é .NET?

.NET é uma plataforma de desenvolvimento criada pela Microsoft que permite criar diversos tipos de aplicações, como web, desktop e mobile.

01

Estrutura do .NET

1. Estrutura do .NET

1.1. Framework vs Core vs 5+

- .NET Framework: Antigo, mais usado em aplicações Windows.
- .NET Core: Multi-plataforma, ideal para novos projetos.
- .NET 5+: Evolução do Core, unificando o desenvolvimento.

1.2. Hello World em .NET (csharp)

```csharp
using System;

class Program
{
    static void Main()
    {
        Console.WriteLine("Hello, World!");
    }
}
```

02

Aplicações Web com ASP.NET Core

2. Aplicações Web com ASP.NET Core

2.1. Criando um Projeto ASP.NET Core

1. Instale o .NET SDK.
2. Crie um projeto: dotnet new webapp -o MinhaWebApp.

2.2. Estrutura de um Projeto ASP.NET Core

- Program.cs: Ponto de entrada da aplicação.
- Startup.cs: Configurações de serviços e pipeline de middleware.

2.3. Exemplo de Rota Simples (csharp)

```csharp
app.MapGet("/", () => "Hello, World!");
```

2.4. Controladores e APIs (csharp)

```csharp
[ApiController]
[Route("[controller]")]
public class WeatherForecastController : ControllerBase
{
    [HttpGet]
    public IEnumerable<WeatherForecast> Get()
    {
        // Implementação do método
    }
}
```

03

Acesso a Dados com Entity Framework Core

3. Acesso a Dados com Entity Framework Core

3.1. Configurando o EF Core

1. Adicione o pacote: dotnet add package Microsoft.EntityFrameworkCore.
2. Configure o contexto no Startup.cs.

3. Acesso a Dados com Entity Framework Core

3.2. Criando um Modelo e Contexto (csharp)

```csharp
public class Produto
{
    public int Id { get; set; }
    public string Nome { get; set; }
    public decimal Preco { get; set; }
}

public class AppDbContext : DbContext
{
    public DbSet<Produto> Produtos { get; set; }
}
```

3.3. Migrations e Atualização do Banco de Dados (sh)

```sh
dotnet ef migrations add InitialCreate
dotnet ef database update
```

04

Aplicações Desktop com WPF

4. Aplicações Desktop com WPF

4.1. Criando um Projeto WPF

1. Crie um projeto: dotnet new wpf -o MinhaWpfApp.

4.2. Estrutura Básica de um Projeto WPF

- App.xaml: Definições globais da aplicação.
- MainWindow.xaml: Interface principal.

4.3. Exemplo de Interface Simples (xml)

```xml
<Window x:Class="MinhaWpfApp.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        Title="MainWindow" Height="350" Width="525">
    <Grid>
        <Button Content="Click Me" HorizontalAlignment="Left" VerticalAlignment="Top" Width="75" Click="Button_Click"/>
    </Grid>
</Window>
```

4.4. Manipulando Eventos (csharp)

```csharp
private void Button_Click(object sender, RoutedEventArgs e)
{
    MessageBox.Show("Button clicked!");
}
```

05

Desenvolvimento Mobile com Xamarin

5. Desenvolvimento Mobile com Xamarin

5.1. Criando um Projeto Xamarin.Forms

1. Crie um projeto: dotnet new maui -o MinhaXamarinApp.

5.2. Estrutura de um Projeto Xamarin.Forms

- MainPage.xaml: Página principal da aplicação.
- App.xaml.cs: Configurações iniciais da aplicação.

5.3. Exemplo de Interface Simples (xml)

```xml
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="MinhaXamarinApp.MainPage">

    <StackLayout>
        <Label Text="Welcome to Xamarin.Forms!"
               VerticalOptions="CenterAndExpand"
               HorizontalOptions="CenterAndExpand" />
    </StackLayout>
</ContentPage>
```

5.4. Navegação entre Páginas (csharp)

```csharp
await Navigation.PushAsync(new SecondPage());
```

06

Boas Práticas de Desenvolvimento

2.Escreva testes para métodos críticos.

6. Boas Práticas de Desenvolvimento

6.1. Clean Code

1. Clareza: Código deve ser fácil de ler e entender.
2. Simplicidade: Evite complexidade desnecessária.

6.2. Testes Unitários

- Crie um projeto de testes: dotnet new xunit -o MinhaApp.Tests.
- Escreva testes para métodos críticos.

6.3. Exemplo de Teste Unitário (csharp)

```csharp
public class CalculadoraTests
{
    [Fact]
    public void Soma_DeveRetornarResultadoCorreto()
    {
        // Arrange
        var calc = new Calculadora();

        // Act
        var resultado = calc.Soma(2, 3);

        // Assert
        Assert.Equal(5, resultado);
    }
}
```

07

Conclusão

2.Escreva testes para métodos críticos.

7. Conclusão

7.1. Pratique e Aprenda

1. Experimente criar pequenos projetos para fixar o conhecimento. Pratique diariamente e explore a vasta documentação disponível.

08

Agradecimentos

2.Escreva testes para métodos críticos.

OBRIGADO POR LER ATÉ AQUI

Esse E-book foi gerado por IA e diagramado por humano.

O conteúdo desse e-book foi gerado com fins didáticos de construção para aprendizado de utilização das ferramentas IA, não foi realizada uma validação humana podendo conter erros gerados por uma IA.