Proposta de Exercícios Práticos sobre Hierarquia de Herança, Enumeradores e Verificação de Tipo

Contexto: Sistema de Gerenciamento de Tarefas

Vamos criar um sistema de gerenciamento de tarefas para uma equipe de desenvolvimento de software. O sistema deverá gerenciar diferentes tipos de tarefas (Desenvolvimento, Design, Teste) e permitir a identificação do tipo de tarefa em tempo de execução utilizando enumeradores e operadores como `is`, `as` e `typeof`.

Exercício 1: Melhorar a Classe Tarefa com Enums

1. **Crie um Enum TipoTarefa**
    - Valores: Desenvolvimento, Design, Teste
2. **Modifique a Classe Tarefa**
    - Substitua a propriedade `Tipo` de string para `TipoTarefa`
    - Atualize os construtores das classes derivadas para utilizar o enum
3. **Teste a Implementação**
    - Crie instâncias das classes `TarefaDesenvolvimento`, `TarefaDesign`, e `TarefaTeste`
    - Verifique a propriedade `Tipo` utilizando enums

Código Exemplo

```
public enum TipoTarefa
{
    Desenvolvimento,
    Design,
    Teste
}

public class Tarefa
{
    public TipoTarefa Tipo { get; set; }
    public string Titulo { get; set; }
    public string Descricao { get; set; }

    public Tarefa(TipoTarefa tipo, string titulo, string descricao)
    {
        Tipo = tipo;
        Titulo = titulo;
        Descricao = descricao;
    }

    public void MostrarDetalhes()
    {
        Console.WriteLine($"Título: {Titulo}, Descrição: {Descricao}, Tipo: {Tipo}");
    }
}

public class TarefaDesenvolvimento : Tarefa
```

```
{
    public TarefaDesenvolvimento(string titulo, string descricao)
        : base(TipoTarefa.Desenvolvimento, titulo, descricao) { }
}

public class TarefaDesign : Tarefa
{
    public TarefaDesign(string titulo, string descricao)
        : base(TipoTarefa.Design, titulo, descricao) { }
}

public class TarefaTeste : Tarefa
{
    public TarefaTeste(string titulo, string descricao)
        : base(TipoTarefa.Teste, titulo, descricao) { }
}
```

Exercício 2: Implementar Verificação de Tipo em Tempo de Execução

1. **Crie um Método IdentificarTarefa**
    - Utilize operadores `is` para identificar o tipo do objeto
    - Exiba uma mensagem indicando o tipo da tarefa
2. **Crie um Método IdentificarTarefaComCast**
    - Utilize operadores `as` e cast explícito para identificar o tipo do objeto
    - Exiba uma mensagem indicando o tipo da tarefa
3. **Crie um Método IdentificarTarefaComTypeOf**
    - Utilize o operador `typeof` para comparar o tipo do objeto
    - Exiba uma mensagem indicando o tipo da tarefa
4. **Teste a Implementação**
    - Crie instâncias das classes `TarefaDesenvolvimento`, `TarefaDesign`, e `TarefaTeste`
    - Passe essas instâncias para os métodos `IdentificarTarefa`, `IdentificarTarefaComCast` e `IdentificarTarefaComTypeOf`

Código Exemplo

```
public class Program
{
    public static void Main()
    {
        Tarefa tarefaDesenvolvimento = new TarefaDesenvolvimento("Implementar API",
"Desenvolver endpoints REST");
        Tarefa tarefaDesign = new TarefaDesign("Criar Wireframe", "Desenhar o layout
da página inicial");
        Tarefa tarefaTeste = new TarefaTeste("Testar API", "Testar os endpoints
desenvolvidos");

        IdentificarTarefa(tarefaDesenvolvimento);
```

```
        IdentificarTarefa(tarefaDesign);
        IdentificarTarefa(tarefaTeste);

        IdentificarTarefaComCast(tarefaDesenvolvimento);
        IdentificarTarefaComCast(tarefaDesign);
        IdentificarTarefaComCast(tarefaTeste);

        IdentificarTarefaComTypeOf(tarefaDesenvolvimento);
        IdentificarTarefaComTypeOf(tarefaDesign);
        IdentificarTarefaComTypeOf(tarefaTeste);
    }

    public static void IdentificarTarefa(Tarefa tarefa)
    {
        if (tarefa is TarefaDesenvolvimento)
        {
            Console.WriteLine($"{tarefa.Titulo} é uma tarefa de Desenvolvimento.");
        }
        else if (tarefa is TarefaDesign)
        {
            Console.WriteLine($"{tarefa.Titulo} é uma tarefa de Design.");
        }
        else if (tarefa is TarefaTeste)
        {
            Console.WriteLine($"{tarefa.Titulo} é uma tarefa de Teste.");
        }
        else
        {
            Console.WriteLine($"{tarefa.Titulo} é de um tipo desconhecido.");
        }
    }

    public static void IdentificarTarefaComCast(Tarefa tarefa)
    {
        TarefaDesenvolvimento tarefaDesenvolvimento = tarefa as TarefaDesenvolvimento;
        if (tarefaDesenvolvimento != null)
        {
            Console.WriteLine($"{tarefa.Titulo} é uma tarefa de Desenvolvimento.");
            return;
        }

        TarefaDesign tarefaDesign = tarefa as TarefaDesign;
        if (tarefaDesign != null)
        {
            Console.WriteLine($"{tarefa.Titulo} é uma tarefa de Design.");
            return;
        }

        TarefaTeste tarefaTeste = tarefa as TarefaTeste;
        if (tarefaTeste != null)
        {
            Console.WriteLine($"{tarefa.Titulo} é uma tarefa de Teste.");
```

```
            return;
        }

        Console.WriteLine($"{tarefa.Titulo} é de um tipo desconhecido.");
    }

    public static void IdentificarTarefaComTypeOf(Tarefa tarefa)
    {
        if (tarefa.GetType() == typeof(TarefaDesenvolvimento))
        {
            Console.WriteLine($"{tarefa.Titulo} é uma tarefa de Desenvolvimento.");
        }
        else if (tarefa.GetType() == typeof(TarefaDesign))
        {
            Console.WriteLine($"{tarefa.Titulo} é uma tarefa de Design.");
        }
        else if (tarefa.GetType() == typeof(TarefaTeste))
        {
            Console.WriteLine($"{tarefa.Titulo} é uma tarefa de Teste.");
        }
        else
        {
            Console.WriteLine($"{tarefa.Titulo} é de um tipo desconhecido.");
        }
    }
}
```

Resumo

1. **Enums**: Substituir strings por enums para tipificar objetos proporciona maior segurança e clareza ao código.
2. **Verificação de Tipo**: Utilizar operadores `is`, `as` e `typeof` para identificar o tipo de um objeto em tempo de execução.
3. **Exercícios Práticos**: Implementar e testar as técnicas em um sistema de gerenciamento de tarefas, aplicando o conhecimento de hierarquia de herança, enumeradores e verificação de tipo.

Esses exercícios e exemplos ajudarão os alunos a compreenderem melhor como utilizar herança, enumeradores e técnicas de verificação de tipo em um contexto de desenvolvimento de software real.