UNIVERSIDADE FEDERAL DO OESTE DO PARÁ
INSTITUTO DE ENGENHARIA E GEOCIÊNCIAS
PROGRAMA DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Alunos: Sandro Oliveira da Silva matrícula 201300824

Iolandino Xayukuma Wai Wai matrícula 201601234

Selena Crixi Miranda Leão matrícula 201300188

No texto com link a seguir, destaque 3 assuntos (tópico, sub temas, ...) que você considere como os mais importantes ou interessantes; e para cada assunto faça o seguinte:
1) Justifique a sua escolha, em no máximo 5 linhas.
2) Crie uma questão com resposta dissertativa e responda a mesma, em no máximo 5 linhas.
Link para o texto:
https://learn.microsoft.com/pt-br/dotnet/csharp/language-reference/attributes/nullable-analysis
Observo ainda que:
a) A atividade pode ser executada individualmente ou em equipe de no máximo 3 alunos.
b) O arquivo com a resposta a esta atividade deve ser devidamente compartilhado e seu link deverá ser enviado via formulário específico, o qual irei disponibilizar daqui a uma semanas.

Atributos para análise estática de estado nulo interpretados pelo compilador C#

Pré-condições: AllowNull e DisallowNull

Para adquirir conhecimento e habilidades atualizadas, é essencial compreender as condições de uso dos atributos AllowNull e DisallowNull em C#. Estes atributos são cruciais para manipular corretamente valores nulos em propriedades e métodos. Ao utilizá-los de forma apropriada, podemos especificar se um valor nulo é permitido ou não em um determinado contexto, o que garante a consistência e a segurança do código.

1 - Em quais situações o atributo AllowNull é mais adequado em comparação com o DisallowNull em C#? Justifique.

R: O uso do atributo AllowNull é mais adequado quando se deseja permitir valores nulos como parte da lógica do programa, enquanto o atributo DisallowNull é mais adequado quando se deseja garantir que valores nulos não sejam permitidos e representem uma situação inválida.

Pós-condições: MaybeNull e NotNull

Definem as condições de retorno de métodos, indicando a possibilidade de valores nulos através de [MaybeNull] e [NotNull]. [MaybeNull] permite um valor de retorno nulo, enquanto [NotNull] assegura que o argumento não seja nulo após a execução (por exemplo, ThrowWhenNull). Esses atributos são cruciais para documentar e garantir a manipulação correta de valores nulos, promovendo consistência e segurança no código.

2- Como a aplicação dos atributos MaybeNull e NotNull em métodos em C# pode ser utilizada para estabelecer pós-condições claras, assegurando que um valor de retorno não anulável possa ser nulo e que um valor de retorno anulável nunca será nulo, contribuindo assim para a consistência e a robustez do código? Justifique.

R: A utilização dos atributos MaybeNull e NotNull em métodos em C# é crucial para estabelecer clareza nas pós-condições, assegurar consistência e robustez do código, prevenir erros com valores nulos, melhorar a análise estática, facilitar a manutenção e promover um desenvolvimento eficiente e seguro, livre de problemas relacionados a valores nulos.

Método auxiliares: MemberNotNull e MemberNotNullWhen

Essenciais para refatorar construtores em métodos auxiliares, garantindo a inicialização correta de campos de referência não anuláveis e evitando o aviso CS8618 (inicialização não verificada). Promovem consistência e segurança na inicialização de campos, sendo cruciais para manter a integridade do código e prevenir falhas.

3 - Como garantir a inicialização correta de campos de referência não anuláveis em métodos auxiliares em C#? Justifique.

R: Permitir que o compilador análise e identifique a inicialização dos campos, evitando warnings de compilação como CS8618, os atributos MemberNotNull e MemberNotNullWhen são essenciais para garantir a correta inicialização de campos de referência não anuláveis em métodos auxiliares e promover consistência e segurança nesse processo. Eles são fundamentais para trabalhar com tipos de referência não anuláveis em C#.