Interfaces X Classes Abstratas

Com o início do paradigma de desenvolvimento OO, alguns termos se tornam corriqueiros no mundo de programadores e analistas de sistema. Classes, herança, polimorfismo são alguns destes termos. Neste artigo mostro as diferenças e similaridades entre Interfaces e Classes Abstratas.

Com o início do paradigma de desenvolvimento OO, alguns termos se tornam corriqueiros no mundo de programadores e analistas de sistema. Classes, herança, polimorfismo são alguns destes termos.

Mas se tratando de Classes, senti que ainda persistem algumas dúvidas quanto às diferenças entre classes do tipo Abstratas e Interfaces.

São comuns os questionamentos: Quando devo usar uma classe abstrata? Quando devo usar uma Interface? Devo usar as duas?

Na verdade, uma Classe Abstrata sem qualquer implementação, tem o aspect parecido com uma Interface. Mas ambas possuem várias diferenças e similaridades entre si. Pensando neste tipo de dúvida, este pequeno artigo tenta elucidar algumas questões.

Interfaces:

· Uma interface não é considerada uma Classe e sim uma Entidade.

· Não possui implementação, apenas assinatura, ou seja, apenas a definição dos seus métodos sem o corpo.

· Todos os métodos são abstratos.

· Seus métodos são implicitamente Públicos e Abstratos.

· Não há como fazer uma instância de uma Interface e nem como criar um Construtor.

· Funcionam como um tipo de "contrato", onde são especificados os atributos, métodos e funções que as classes que implementem essa interface são obrigadas a implementar.

· Já que C# não suporta Heranças Múltiplas, as Interfaces são usadas para implementá-las.

Classes Abstratas:

· As classes abstratas devem conter pelo menos um método abstrato, que não tem corpo.

· É um tipo especial de classe que não há como criar instâncias dela.

· É usada apenas para ser herdada, funciona como uma super classe.

· Uma grande vantagem é que força a hierarquia para todas as sub-classes.

· É um tipo de contrato que faz com que as sub-classes contemplem as mesmas hierarquias e/ou padrões.

Overview:

Quando nos criamos uma Interface, nós estamos basicamente criando um set de métodos sem qualquer implementação que deve ser herdado por outras classes já implementadas. A vantagem é que desta forma consegue-se prover um caminho para uma classe ser parte de duas classes: uma herdada hierarquicamente e outra da

Interface.

Quando nos criamos uma Classe Abstrata, nós estamos criando uma classe base que pode ter um ou mais métodos completos, mas pelo menos um ou mais destes métodos tem que criados incompletos (sem corpo), isto caracteriza uma Classe Abstrata.

Vale lembrar que, se todos os método da Classe abstrata forem sem corpo, ela se torna uma Interface.

O propósito de uma Classe Abstrata é prover uma base de definições de como um set de Classes Derivadas irão trabalhar e então permitir os programadores de preencher as implementações nas Classes derivadas.

Abaixo um quadro comparativo para tornar mais fácil a compreensão entre diferenças e similaridades entre Classes Abstratas e Interfaces.

Característica	Interface	Classe Abstrata
Herança múltipla	Uma classe pode implementar diversas interfaces	Uma classe pode herdar somente uma classe
Implementação Padrão	Uma interface não pode conter qualquer tipo de código, muito menos código padrão.	Uma classe abstrata pode fornecer código completo, código padrão ou ter apenas a declaração de seu esqueleto para ser posteriormente sobrescrita.
Constantes	Suporte somente constantes do tipo estática.	Pode conter constantes estáticas e de instância.
Componentes de terceiros	Uma implementação de uma interface pode ser incluída a qualquer classe de terceiros.	Uma classe de terceiros precisa ser reescrita para estender somente a partir da classe abstrata.

Homogeneidade	Se todas as diversas implementações compartilham a assinatura do método então a interface funciona melhor.	Se as várias implementações são todas do tipo e compartilham um comportamento e status comum , então a classe abstrata funciona melhor.
Manutenção	Se o código do seu cliente conversa somente em termos de uma interface, você pode facilmente alterar a implementação concreta usando um método factory.	Idêntico
Velocidade	Lento, requer trabalho extra para encontrar o método correspondente na classe atual.	Rápido
Clareza	Todas as declarações de constantes em uma interface são presumidamente publicas ou estáticas.	Você pode por código compartilhado em uma classe abstrata. Você pode usar código para computar o valor inicial de suas constantes e variáveis de instância ou estáticas.
Funcionalidades Adicionais	Se você incluir um novo método em uma interface você precisa ajustar todas as implementações da interface.	Se você incluir um novo método em uma classe abstrata você tem a opção de fornecer uma implementação padrão para ele.

Publicado no Canal .net

por Luiz Agnelo
Guru .net e tecnologias MS

Gostou?

Sim (4) (0)

Ficou com alguma dúvida?

Todos os comentarios (7)

Postar dúvida / Comentário

Meus comentarios

Lucas Do Amaral
Na verdade, uma classe abstrata não PRECISA conter nenhum método abstrato.

A classe abstrata pode conter todos os métodos concretos, mas por ser abstrata, não poderá ser instanciada, fazendo com que para que seus métodos possam ser invocados, ela precise ser extendida por outra classe concreta.

[há +1 ano] - Responder

Henrique Machado Gasparotto
Olá Lucas, realmente, você está correto. Muito obrigado pela correção.

[há +1 ano] - Responder

Rodolfo Leão Duarte Silva
Obrigado pelo post, esclareceu algumas dúvidas. Gostaria de alertar que achei um conteúdo muito similar aqui: http://www.macoratti.net/net_ica1.htm

Mas nenhum dos autores colocaram referências.

[há +1 ano] - Responder

Joel Rodrigues
Prezado Rodolfo, agradeço pelo alerta, este é um artigo muito antigo em nosso site, ele foi publicado em 30/6/2009.

Infelizmente não temos como saber quando o artigo mencionado no site do Macoratti foi publicado, o que nos inviabiliza em fazer uma análise mais apurada. De qualquer forma, caso você acredite que existe um outro artigo no qual ambos (este e o do Macoratti) se basearam, agradecemos imensamente se você puder nos fornecer o link, então certamente faremos uma análise e estando comprovada qualquer cópia ou uso como referência, faremos as devidas correções.

No mais, agradeço mais uma vez a participação em nosso site.

Abs.

[há +1 ano] - Responder

Daniel

Tem como saber sim, basta limitar a busca no google com uma data inferior à 30/06/2009 (que foi a data da postagem desse artigo) e ver que a do Macoratti foi indexada em 2 de ago de 2005. Mas também a única coisa que vcs copiaram 100% foi a tabela, não vejo muito problema nisso...

[há +1 mês] - Responder

Herval Lemos Junior

No trecho que lista as definições das classes abstratas, num dos items temos: "É usada apenas para ser herdada, funciona como uma sub-classe."
O correto não seria: É usada apenas para ser herdada, funciona como uma SUPER CLASSE.
???

[há +1 ano] - Responder

Joel Rodrigues

Olá, Herval. Obrigado pelo comentário. Realmente você está certo, às vezes esses detalhes passam em branco. Já foi feita a correção.

[há +1 ano] - Responder