C# Essential

Абстрактные классы и интерфейсы

После урока обязательно

Повторите этот урок в видео формате на ITVDN.com

Проверьте как Вы усвоили данный материал на TestProvider.com

Тема

Абстрактные классы и интерфейсы

Понятие абстракции

В развитии науки осуществляется закон движения познания, который можно сформулировать следующим образом:

“Мышление, восходя от конкретного к абстрактному, не отходит – если оно правильное ... – от истины, а подходит к ней. Абстракция материи, закона природы, абстракция стоимости и т.д., одним словом все научные (правильные, серьезные, не вздорные) абстракции отражают природу глубже, вернее, полнее. От живого созерцания к абстрактному мышлению и от него к практике – таков диалектический путь познания истины, познания объективное реальности”.

Абстракция

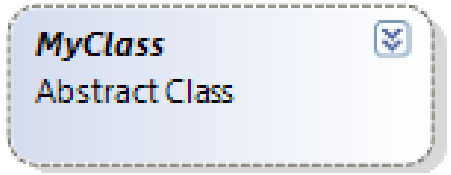
Понятие абстракции

Абстракция в ООП – это придание объекту характеристик, которые отличают его от всех других объектов, четко определяя его концептуальные границы.

Абстрагирование в ООП – это способ выделить набор значимых характеристик объекта, исключая из рассмотрения незначимые. Соответственно, абстракция — это набор всех таких характеристик.

Абстрактные классы

Абстрактный класс в объектно-ориентированном программировании – это базовый класс, который не предполагает создания экземпляров через вызов конструктора напрямую, но экземпляр абстрактного класса создается неявно при построении экземпляра производного конкретного класса.




```
abstract class MyClass
{

}
```

Абстракция

Ключевое слово abstract

Ключевое слово `abstract` может использоваться с ***классами, методами, свойствами, индексаторами и событиями.***

Абстрактные классы

Возможности и ограничения абстрактных классов:

- Экземпляр абстрактного класса создать нельзя через вызов конструктора напрямую, но экземпляр абстрактного класса создается неявно при построении экземпляра производного конкретного класса.
- Абстрактные классы могут содержать как абстрактные, так и не абстрактные члены.
- Не абстрактный (конкретный) класс, являющийся производным от абстрактного, должен содержать фактические реализации всех наследуемых абстрактных членов.

Абстрактные методы

Возможности абстрактных методов:

- Абстрактный метод является неявным виртуальным методом.
- Создание абстрактных методов допускается только в абстрактных классах.
- Тело абстрактного метода отсутствует; создание метода просто заканчивается двоеточием, а после сигнатуры ставить фигурные скобки ({ }) не нужно.
- Реализация предоставляется методом переопределения `override`, который является членом неабстрактного класса.

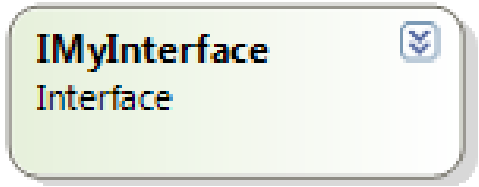
Интерфейсы

Интерфейс – семантическая и синтаксическая конструкция в коде программы, используемая для специфицирования услуг, предоставляемых классом или компонентом.

Интерфейс – стереотип, являющийся аналогом чистого абстрактного класса, в котором запрещена любая реализация.

Создание интерфейсов

Для имени интерфейса следует применять в качестве префикса букву "I".
Это подсказывает, что данный тип является интерфейсом.




```
interface IMyInterface
{

}
```

Интерфейсы

Правила использования интерфейсов:

- Невозможно создать экземпляр интерфейса.
- Интерфейсы и члены интерфейсов являются абстрактными. Интерфейсы не имеют реализации в C#.
- Интерфейс может содержать только абстрактные члены (методы, свойства, события или индексаторы).
- Члены интерфейсов автоматически являются открытыми, абстрактными, и они не могут иметь модификаторов доступа.
- Интерфейсы не могут содержать константы, поля, операторы, конструкторы экземпляров, деструкторы или вложенные типы (интерфейсы в том числе).
- Класс или структура, которые реализуют интерфейс, должны реализовать члены этого интерфейса, указанные при его создании.

Интерфейсы

- Интерфейс может наследоваться от одного или нескольких базовых интерфейсов.

- Базовый класс также может реализовать члены интерфейса с помощью виртуальных членов. В этом случае производный класс может изменить поведение интерфейса путем переопределения виртуальных членов.

- Если класс реализует два интерфейса, содержащих член с одинаковой сигнатурой, то при реализации этого члена в классе оба интерфейса будут использовать этот член для своей реализации.

- Если члены двух интерфейсов с одинаковой сигнатурой методов должны выполнять различные действия при их реализации, необходимо воспользоваться явной реализацией члена интерфейса — техникой явного указания в имени члена, имени интерфейса, которому принадлежит данный член. Это достигается путем включения в имя члена класса имени интерфейса с точкой. Данный член в производном классе будет помечен по умолчанию как скрытый.

Интерфейсы

Преимущество использования интерфейсов:

- Класс или структура может реализовать несколько интерфейсов (допустимо множественное наследование от интерфейсов).
- Если класс или структура реализует интерфейс, она получает только имена и сигнатуры метода.
- Интерфейсы определяют поведение экземпляров производных классов.
- Базовый класс может обладать ненужным функционалом, полученным от других его базовых классов, чего можно избежать, применяя интерфейсы.

Q&A

Информационный видеосервис для разработчиков программного обеспечения

ITVDN

IT VIDEO DEVELOPERS NETWORK

TestProvider PROMETRIC TEST CENTER CyberBionic systematics Microsoft Partner Silver Learning Windows Azure Cloud Partner EBA European Business Association