C# 2.1

ООП Концепты

- Воспоминания
- Инкапсуляция
- Наследование
- Полиморфизм
- Абстрактный класс
- Финализаторы
- Операторы
- Виртуальная функция

Структуры[1]

Структуры – это пользовательский тип, использующийся, как правило, для инкапсуляции небольшого количества разнотипных данных.

Могут содержать поля, конструкторы, свойства, методы, индексаторы

Демонстрация

[1]*Структуры C# представляют частный случай класса*

Класс

Класс - это «чертеж» (описание) сущности предметной области. Позволяющий выделить некоторые общее характеристики состояние и поведение, зависящее от состояния.

Пример: класс «Человек», класс «Сотрудник», класс «Автомобиль», класс «Геометрическая Фигура».

** Предметная область — множество всех предметов (явлений) решаемой проблемы*

Объект

Объект - отдельный представитель класса, имеющий КОНКРЕТНОЕ состояние, и поведение, которое полностью определяется описанием класса.

Пример:

класс «Студент», экземпляр «Иван Крылов 41 ПМиФ»,

класс «Самолет», экземпляр «F-22 ID2014»,

класс «Геометрическая Фигура», экземпляр «Додекаэдр»,

класс «Сортировщики», экземпляр «Cassida C 200 N43-21»

Состояние и поведение

- Состояние - набор данных (полей, атрибутов, членов класса)
- Поведение - функции для работы с данными и выполнения полезной работы

Структура класса

Данные-члены
- Поля
- Константы
- События

Функции-члены
- Методы
- Свойства
- Конструкторы
- Финализаторы
- Операции
- Индексаторы

Данные-члены

Данные-члены — это данные класса.

- *Поля – переменные* ассоциированные с классом
- *Константы - аналогичны переменным*
- *События - уведомляют вызывающий код* о изменении состояния или факте некоторого взаимодействия

ООП

ООП - объектно-ориентированное программирование подход

Парадигма(*стиль, шаблон*) разработки ПО, основными понятиями которой являются *классы и* объекты.

Говорят, что разработка в стиле ООП ведется с использованием классов объектов, которые обладают состоянием и поведением, зависящим от этого состояния.

Инкапсуляция

Основные концепции: Инкапсуляция

Инкапсуляция – это свойство системы, позволяющее объединить данные и методы, работающие с ними, в классе, скрыв детали реализации и защитив от пользователя этого класса объектов.

Наследование

Основные концепции: Наследование

Наследование – это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью.

Класс, от которого производится наследование, называется базовым или родительским. Новый класс – потомком, наследником или производным классом

Полиморфизм

Основные концепции: Полиморфизм

Полиморфизм – это свойство системы, использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта.

Полиморфизм - способность использовать объект вне зависимости от его реализации, благодаря, полиморфной переменной – это переменная, которая может принимать значения разных типов.

Абстракция

Основные концепции: Абстракция

Абстракция – это свойство системы, позволяющее описать общие характеристики базового класса для всех его производных классов, а наполнение деталями предоставляется каждому из этих классов.

В абстрактном классе определяются лишь общие поля и характер методов, которые должны быть конкретно реализованы в производных классах, а не в самом базовом классе.

public abstract string Method_Name(); ** кто есть я*

Интерфейс

interface предназначен для описания исключительно общего поведения сущностей.

Частный случай: для гарантии, реализации некоторого поведения.

Аналогичен абстрактному классу со всеми абстрактными методами, но допускается множественное наследование.

*расширение поведение

** что я могу делать*

Интерфейс

Интерфейс - это частный случай класса. Он представляет собой полностью абстрактный класс, все методы, свойства, события и индексаторы которого абстрактны.

```
interface IExampleInterface
{
    void ExampleMethod(string messaege);
    int ExampleProperty { get; set; }
    event ExampleEventListener ChangeWidth;
    int this[int Index] { get; set; }
}
```

Финализатор

Деструктор -
метод вызываемый перед окончательным уничтожением объекта системой "сборки мусора".

```
~ИмяКласса ()
{
   код
}
```

Перегрузка операторов

Операции C#

+, -, !, ++, —, true, false – набор перегружаемых унарных операций

+, -, *, /, %, &, |, ^, <<, >>, ==, !=, <, >, <=, >= – набор перегружаемых бинарных операций*

C# требует совместной перегрузки «подобных» операций (т.е. < и >, <= и >=, == и !=)

Операция [] не может быть перегружена, альтернатива - индексаторы

+=, -=, *=, /=, %=, &=, |=, ^=, <<=, >>= Сокращенные операции автоматически, перегружаются при перегрузке соответствующей бинарной операции

static public ReturnType operator operation(par list)

Подробно

Спасибо за внимание