Классы и объекты

№ урока: 2 **Курс:** C# Essential

Средства обучения: Компьютер с установленной Visual Studio

Обзор, цель и назначение урока

Рассмотрение частичных классов и частичных методов.
Рассмотрение основных парадигм ООП.
Введение в диаграммы классов UML.
Связи отношений между классами (самоассоциация, ассоциация)

Изучив материал данного занятия, учащийся сможет:

- Создавать и реализовывать частичные методы и классы.
- Понимать и различать основные парадигмы ООП
- Применять в работе диаграммы классов UML

Содержание урока

1. Рассмотрение частичных классов
2. Рассмотрение частичных методов
3. Поля доступные только для чтения (`readonly`).
4. Основные парадигмы ООП.
5. Диаграммы классов UML.
6. Связи отношения между классами (самоассоциация, ассоциация)

Резюме

- ООП – Объектно-ориентированное программирование – парадигма программирования, в которой основными концепциями являются понятия объектов и классов.
- В C# реализована возможность разделить создание класса или метода (структуры, интерфейса) между двумя или более исходными файлами или модулями. Каждый исходный файл содержит определение типа или метода, и все части объединяются при компиляции приложения.
- Существует несколько ситуаций, при которых желательно разделение создания класса:
  1) При работе над большими проектами распределение класса между различными файлами позволяет нескольким программистам работать с ним одновременно.
  2) При работе с использованием автоматически создаваемого источника, код можно добавлять в класс без повторного создания файла источника. Например: при создании форм Windows Forms.
- Чтобы разделить класс на несколько частей, используйте ключевое слово `partial`.
- Ключевое слово `partial` указывает на то, что другие части класса (структуры или интерфейса) могут быть созданы в этом пространстве имен. Все части должны использовать ключевое слово `partial`. Все части должны иметь одинаковые модификаторы доступа, например `public`, `private` и т.д.
- Частичный класс или структура могут содержать частичный метод.
- **Частичные методы** – это методы, где «прототип» или сигнатура метода определена при создании частичного класса, а реализация выполняется в любой другой (только одной) части этого класса.
- Преимущество использования таких методов состоит в том, что их реализацию можно не выполнять. В таком случае, при компилировании кода данный метод даже не компилируется и не попадает в IL-код. То есть, как будто этот метод вообще не существовал.
- Правила использования частичных методов:
  1) Частичные методы должны быть определенны только в частичных классах

CyberBionic systematics
CyberBionic Systematics ® 2015
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

2) Частичные методы должны быть помечены ключевым словом `partial`
3) Частичные методы являются скрытыми (`private`), но явное использование с ними модификатора доступа приведет к ошибке
4) Частичные методы должны возвращать `void`
5) Частичные методы могут быть нереализованными
6) Частичные методы могут не иметь аргументов

- Ключевое слово `readonly` – это модификатор, который можно использовать для полей.
- Если созданное поле содержит модификатор `readonly`, присвоить значение такому полю можно только непосредственно в месте создания или в конструкторе в того же класса.
- Ключевое слово `readonly` отличается от ключевого слова `const` тем, что поле с модификатором `const` может быть инициализировано только при создании поля. Поле с модификатором `readonly` может быть инициализировано при создании или в конструкторе. Следовательно, поля с модификатором `readonly` могут иметь различные значения в зависимости от использованного конструктора.
- **Парадигма программирования** – это система идей и понятий, определяющих стиль написания компьютерных программ, а также образ мышления программиста.
- К основным парадигмам ООП относятся:
  1) Инкапсуляция – это свойство системы, позволяющее объединить данные и методы, работающие с ними, в классе и скрыть детали реализации от пользователя.
  2) Наследование – это свойство системы, позволяющее описать новый класс на основе уже существующего.
  3) Полиморфизм – возможность объектов с одинаковой спецификацией иметь различную реализацию.
  4) Абстракция – это придание объекту характеристик, которые четко определяют его концептуальные границы, отличая от всех других объектов. Позволяет работать с объектами, не вдаваясь в особенности их реализации.
  5) Посылка сообщений – это способ передачи управления объекту. Если объект должен «отвечать» на это сообщение, то у него должен быть метод, соответствующий данному сообщению.
  6) Повторное использование – парадигма ООП в которой утверждается, что программы (компьютерная программа, программный модуль) частично либо полностью должны составляться из частей, написанных ранее компонентов и/или частей другой программы (системы). Это основная методология, которая применяется для сокращения трудозатрат при разработке сложных систем.
- UML (англ. Unified Modeling Language – унифицированный язык моделирования) – язык графического описания для объектного моделирования в области разработки программного обеспечения.
- UML является языком широкого профиля, это открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML-моделью. UML был создан для определения, визуализации, проектирования и документирования в основном программных систем. UML не является языком программирования
- Разработан в 1994 г. Гради Бучем, Джеймсом Рамбо и Иваром Якобсоном.
- Диаграмма классов (Class diagram) – статическая структурная диаграмма, описывающая структуру системы, она демонстрирует классы системы, их атрибуты, методы и зависимости между классами.
- Взаимосвязь – это особый тип логических отношений между сущностями, показанных на диаграммах классов и объектов.
- Ассоциация показывает, что объекты одной сущности (класса) связаны с объектами другой сущности.
- Агрегация – это разновидность ассоциации при отношении между целым и его частями.(Отношение типа: «Я знаю о… и без этого могу существовать»). Одно отношение агрегации не может включать более двух классов (контейнер и содержимое).
- Композиция – более строгий вариант агрегации. Композиция имеет жёсткую зависимость времени существования экземпляров класса-контейнера и экземпляров содержащихся классов.

CyberBionic systematics
CyberBionic Systematics ® 2015
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Если контейнер будет уничтожен, то всё его содержимое будет также уничтожено. (Отношение типа: «Я знаю о… и без этого не могу существовать»)

- Зависимость – это слабая форма отношения использования, при котором изменение в спецификации одного влечёт за собой изменение другого, причем обратное не обязательно.

Закрепление материала

- Что такое partial класс?
- Что такое partial метод?
- Когда нужно применять частичные классы и методы?
- Что такое ООП?
- Какие основные парадигмы ООП вы знаете?
- Что такое инкапсуляция?
- Что такое ассоциация?
- Какие связей отношений между классами вы знаете?
- Чем поля, помеченные ключевым словом readonly, отличаются от констант?

Дополнительное задание

Задание
Используя Visual Studio, создайте проект по шаблону Console Application.
Требуется:
Создать класс `User`, содержащий информацию о пользователе (логин, имя, фамилия, возраст, дата заполнения анкеты). Поле дата заполнения анкеты должно быть проинициализировано только один раз (при создании экземпляра данного класса) без возможности его дальнейшего изменения.
Реализуйте вывод на экран информации о пользователе.

Самостоятельная деятельность учащегося

Задание 1
Выучите основные конструкции и понятия, рассмотренные на уроке.

Задание 2
Используя Visual Studio, создайте проект по шаблону Console Application.
Требуется:
Создать класс `Converter`.
В теле класса создать пользовательский конструктор, который принимает три вещественных аргумента, и инициализирует поля соответствующие курсу 3-х основных валют, по отношению к гривне – `public Converter(double usd, double eur, double rub).`
Написать программу, которая будет выполнять конвертацию из гривны в одну из указанных валют, также программа должна производить конвертацию из указанных валют в гривну.

Задание 3
Используя Visual Studio, создайте проект по шаблону Console Application.
Требуется:
Создать класс `Employee`.
В теле класса создать пользовательский конструктор, который принимает два строковых аргумента, и инициализирует поля, соответствующие фамилии и имени сотрудника.
Создать метод рассчитывающий оклад сотрудника (в зависимости от должности и стажа) и налоговый сбор.
Написать программу, которая выводит на экран информацию о сотруднике (фамилия, имя, должность), оклад и налоговый сбор.

Задание 4
Используя Visual Studio, создайте проект по шаблону Console Application.

CyberBionic systematics
CyberBionic Systematics ® 2015
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Требуется:
Создать класс `Invoice`.
В теле класса создать три поля `int account`, `string customer`, `string provider`, которые должны быть проинициализированы один раз (при создании экземпляра данного класса) без возможности их дальнейшего изменения.
В теле класса создать два закрытых поля `string article`, `int quantity`
Создать метод расчета стоимости заказа с НДС и без НДС.
Написать программу, которая выводит на экран сумму оплаты заказанного товара с НДС или без НДС.

Задание 5
Зайдите на сайт MSDN.
Используя поисковые механизмы MSDN, найдите самостоятельно описание темы по каждому примеру, который был рассмотрен на уроке, так, как это представлено ниже, в разделе «Рекомендуемые ресурсы», описания данного урока. Сохраните ссылки и дайте им короткое описание.

Рекомендуемые ресурсы

MSDN: Частичные классы и методы (Руководство по программированию в C#)
http://msdn.microsoft.com/ru-ru/library/wa80x488.aspx

MSDN: readonly (Справочник по C#)
http://msdn.microsoft.com/ru-ru/library/acdd6hb7.aspx

CyberBionic systematics
CyberBionic Systematics ® 2015
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com