The C# Programming Language

Professional

Пользовательские коллекции

Microsoft .NET

CyberBionic systematics

Коллекция

Collection

Коллекция – это класс, предназначенный для группировки связанных объектов, управления ими и обработки их в циклах.

Коллекции являются важным инструментом программиста, но решение о их применении не всегда оказывается очевидным.

Применение коллекций

Use of Collections

Коллекции стоит применять, если:

- Отдельные элементы используются для одинаковых целей и одинаково важны.
- На момент компиляции число элементов не известно или не зафиксировано.
- Необходима поддержка операции перебора всех элементов.
- Необходима поддержка упорядочивания элементов.
- Необходимо использовать элементы из библиотеки, от которой потребитель ожидает наличия типа коллекции.

Интерфейс

IEnumerable

Методы интерфейса `IEnumerable`:

`IEnumerator GetEnumerator()` - возвращает перечислитель, который можно использовать для навигации по коллекции.

CyberBionic systematics

Интерфейс

IEnumerator

Свойства интерфейса `IEnumerator`:

`object Current { get; }` – возвращает текущий элемент коллекции.

Методы интерфейса `IEnumerator`:

`bool MoveNext()` – перемещает перечислитель на следующий элемент коллекции.

`void Reset()` – возвращает перечислитель на начало коллекции.

CyberBionic systematics

Интерфейс

IEnumerable<T>

`IEnumerable<T>` - унаследован от `IEnumerable`

Методы интерфейса `IEnumerable<T>`:

`IEnumerator<T> GetEnumerator()` - возвращает обобщенный перечислитель, который можно использовать для навигации по коллекции.

CyberBionic systematics

Интерфейс

IEnumerator<T>

`IEnumerator<T>` унаследован от `IDisposable` и `IEnumerator`

Свойства интерфейса `IEnumerator<T>`:

`T Current { get;}` – возвращает текущий элемент коллекции.

Интерфейс

ICollection

`ICollection` Унаследован от: `IEnumerable`

Свойства интерфейса `ICollection`:

- `int Count { get; }` – возвращает количество элементов, хранящихся в коллекции.
- `bool IsSynchronized { get; }` – признак синхронизации доступа к коллекции.
- `object SyncRoot { get; }` – объект синхронизации доступа к коллекции.

Методы интерфейса `ICollection`:

- `void CopyTo(Array array, int index)` – выполняет копирование элементов коллекции в массив, начиная с указанного индекса.

CyberBionic systematics

Интерфейс

ICollection<T>

`ICollection<T>` унаследован от `IEnumerable<T>` и `IEnumerable`

Свойства интерфейса `ICollection<T>`:

- `int Count { get; }` – возвращает количество элементов, хранящихся в коллекции.
- `bool IsReadOnly { get; }` – показывает, является ли коллекция доступной только для чтения.

Методы интерфейса `ICollection<T>`:

- `void Add(T item)` – позволяет помещать элементы в коллекцию.
- `void Clear()` – очищает содержимое коллекции.
- `bool Contains(T item)` – определяет, содержится ли в коллекции указанное значение.
- `void CopyTo(T[] array, int arrayIndex)` - выполняет копирование элементов коллекции в массив, начиная с указанного индекса.
- `bool Remove(T item)` – удаляет из коллекции первый элемент, значение которого соответствует значению аргумента.

CyberBionic systematics

Интерфейс

IList

`IList` Унаследован от `ICollection` и `IEnumerable`

Свойства интерфейса `IList`:

`bool IsFixedSize { get; }` – показывает, является ли размер списка фиксированным.
`bool IsReadOnly { get; }` - показывает, является ли список доступным только для чтения.
`object this[int index] { get; set; }` – индексатор, позволяющий помещать либо извлекать значения по указанному индексу.

Методы интерфейса `IList`:

`int Add(object value)` – позволяет помещать элементы в список.
`void Clear()` – очищает содержание списка.
`bool Contains(object value)` – определяет, содержится ли в списке указанное значение.
`int IndexOf(object value)` – возвращает индекс элемента с указанным значением.
`void Insert(int index, object value)` – помещает элемент в список по указанному индексу.
`void Remove(object value)` – удаляет из списка первый элемент, значение которого соответствует значению аргумента.
`void RemoveAt(int index)` – удаляет элемент по указанному индексу.

CyberBionic systematics

Ключевое слово

yield

- Блок, в котором содержится ключевое слово `yield`, расценивается компилятором, как блок итератора.
- Ключевое слово `return` используется для предоставления значения объекту перечислителя.
- Ключевое слово `break` используется для обозначения конца итерации.

Циклическая конструкция

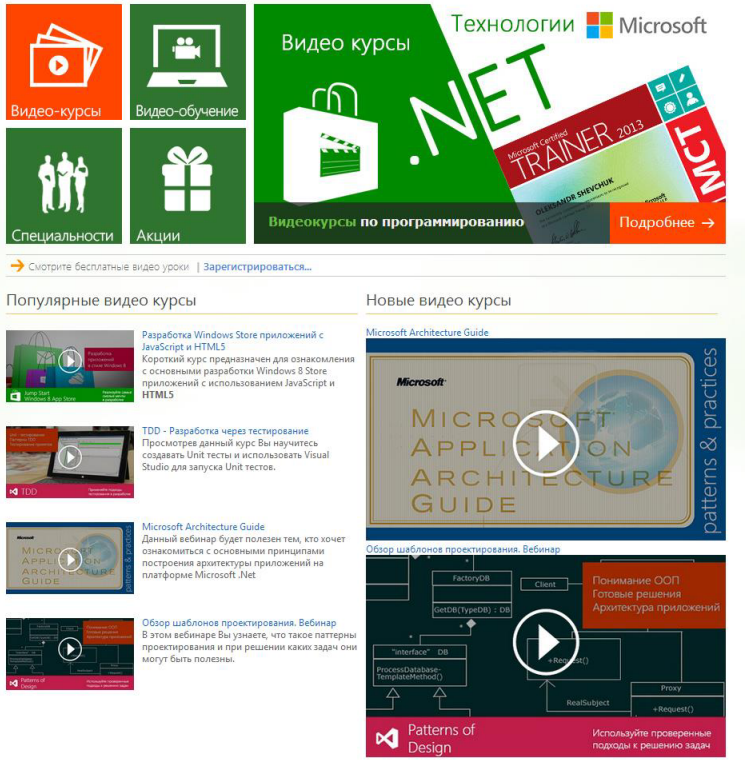
foreach

Циклическая конструкция `foreach` позволяет выполнять навигацию по коллекции, используя реализации интерфейсов `IEnumerable` и `IEnumerator`.

Q&A

ВИДЕО ОБУЧЕНИЕ

Microsoft Partner
Silver Learning



Перейти к видеопорталу
video.cbsystematics.com

Гарантия качества

Видео курсы Учебного центра CyberBionic Systematics - результат проверенной годами методики обучения программистов.
Они разработаны сертифицированными тренерами Microsoft для учебного центра CyberBionic Systematics

Преимущества видео обучения

- Вы можете просматривать учебный материал повторно необходимое количество раз
- Вы можете делать остановки в обучении для выполнения задания с учетом Вашей способности восприятия нового материала
- Вы обучаетесь у сертифицированных тренеров Microsoft

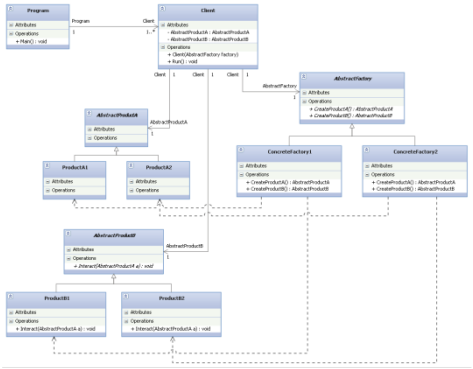
Видео курсы - это возможность обучаться самостоятельно, а также многократно просматривать и повторять материал учебного курса, если Вы обучаетесь очно или on-line. Мы рекомендуем видеообучение также специалистам, которым нужно систематизировать и углубить знания, полученные ранее в ВУЗе.

Windows Azure

CyberBionic Systematics® TestProvider

CyberBionic systematics

КУРС: Patterns of Design (GoF)



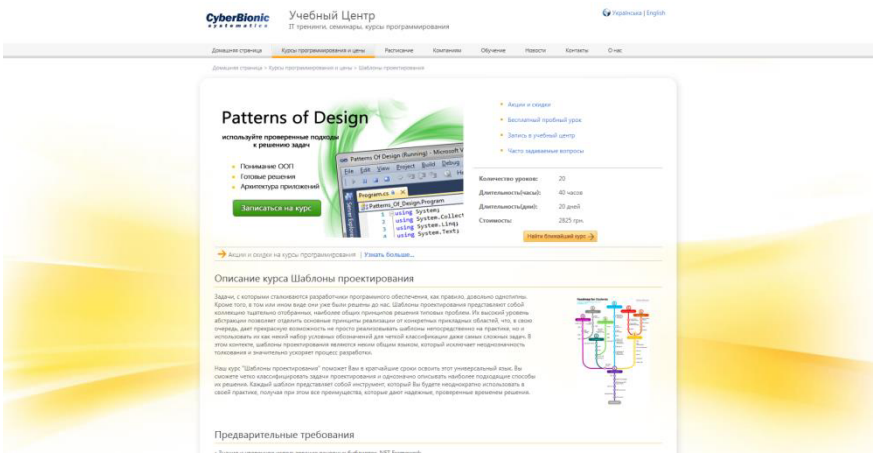
Задачи, с которыми сталкиваются разработчики программного обеспечения, как правило, довольно однотипны. Кроме того, в том или ином виде они уже были решены до нас. Шаблоны проектирования представляют собой коллекцию тщательно отобранных, наиболее общих принципов решения типовых проблем. Их высокий уровень абстракции позволяет отделить основные принципы реализации от конкретных прикладных областей, что, в свою очередь, дает прекрасную возможность не просто реализовывать шаблоны непосредственно на практике, но и использовать их как некий набор условных обозначений для четкой классификации даже самых сложных задач. В этом контексте, шаблоны проектирования являются неким общим языком, который исключает неоднозначность толкования и значительно ускоряет процесс разработки.

Описание курса:

Курс "Шаблоны проектирования" поможет Вам в кратчайшие сроки освоить приемы проектирования. Вы сможете четко классифицировать задачи проектирования и однозначно описывать наиболее подходящие способы их решения. Каждый шаблон представляет собой инструмент, который Вы будете неоднократно использовать в своей практике, получая при этом все преимущества, которые дают надежные, проверенные временем решения.

Длительность:

20 часов/10 дней.

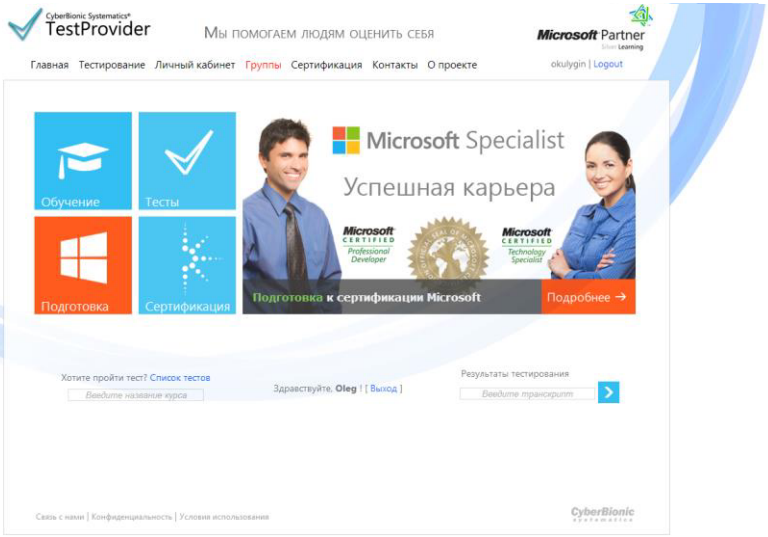


Узнать более подробно о курсе на сайте: edu.cbsystematics.com

CyberBionic systematics

ПРОВЕРКА ЗНАНИЙ

CyberBionic Systematics© TestProvider



Тестирование IT-специалистов

TestProvider обеспечивает надежную и объективную оценку технических знаний и опыта работы IT-специалиста с программными продуктами Microsoft.

Сертификация IT-специалистов

Подтвердите ваш практический опыт работы с технологиями ***Microsoft***, получив сертификацию, соответствующую той работе, которую вы выполняете сейчас или желаете получить в будущем.

Перейти к тестированию
www.TestProvider.com

Компании ***Microsoft***, ***CyberBionic Systematics*** и ***Intel*** на базе портала TestProvider компании CyberBionic Systematics с использованием платформы Microsoft Azure совместно с Министерством науки и образования Украины проводят Всеукраинское дистанционное мониторинговое исследование уровня сформированности у выпускников учебных заведений навыков использования информационно-коммуникативных технологий в практической деятельности.

Microsoft® intel® CyberBionic Systematics© TestProvider CyberBionic systematics

CyberBionic
systematics

Coevolution of humans and machines.