**Класс Dictionary<TKey, TValue>**

Класс Dictionary<TKey, TValue> позволяет хранить пары "ключ-значение"

в коллекции как в словаре. Значения доступны в словаре по соответствующим ключам.

В этом отношении данный класс аналогичен необобщенному классу Hashtable.

В классе Dictionary<TKey, TValue> реализуются интерфейсы IDictionary,

IDictionary<TKey, TValue>, ICollection, ICollection<KeyValuePair<TKey,

TValue>>, IEnumerable, IEnumerable<KeyValuePair<TKey, TValue>>,

ISerializable и IDeserializationCallback. В двух последних интерфейсах поддерживается

сериализация списка. Словари имеют динамический характер, расширяясь по

мере необходимости.

В классе Dictionary<TKey, TValue> предоставляется немало конструкторов.

Ниже перечислены наиболее часто используемые из них.

public Dictionary()

public Dictionary(IDictionary<TKey, TValue> dictionary)

public Dictionary(int capacity)

В первом конструкторе создается пустой словарь с выбираемой по умолчанию

первоначальной емкостью. Во втором конструкторе создается словарь с указанным количеством

элементов *dictionary. А* в третьем конструкторе с помощью параметра

*сараcity* указывается емкость коллекции, создаваемой в виде словаря. Если размер

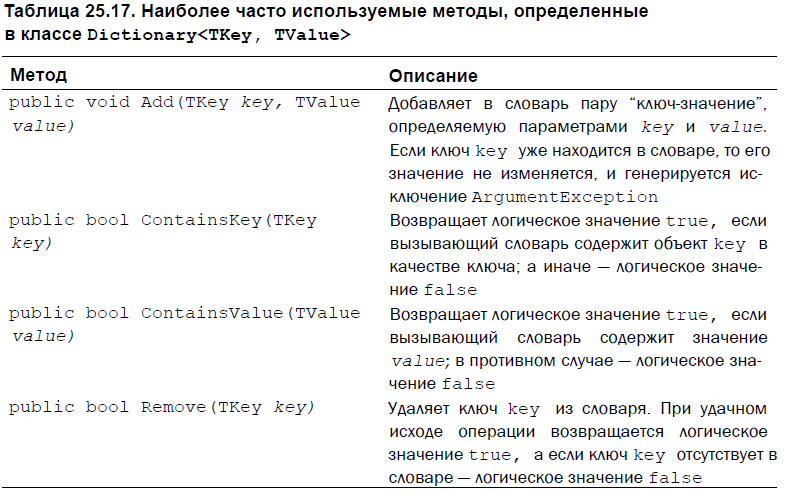
словаря заранее известен, то, указав емкость создаваемой коллекции, можно исключить

изменение размера словаря во время выполнения, что, как правило, требует дополнительных

затрат вычислительных ресурсов.

В классе Dictionary<TKey, TValue> определяется также ряд методов. Некоторые

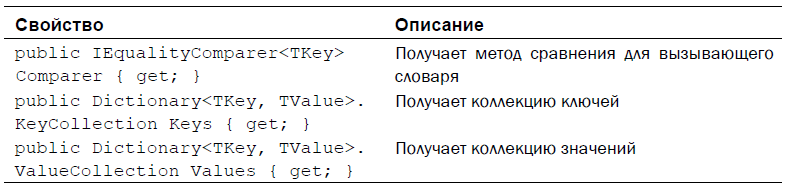
наиболее часто используемые методы этого класса сведены в табл. 25.17.



Кроме того, в классе Dictionary<TKey, TValue> определяются собственные

свойства, помимо тех, что уже объявлены в интерфейсах, которые в нем реализуются.

Эти свойства приведены ниже.



Следует иметь в виду, что ключи и значения, содержащиеся в коллекции, доступны

отдельными списками с помощью свойств Keys и Values. В коллекциях типа

Dictionary<TKey, TValue>.KeyCollection и Dictionary<TKey, TValue>.

ValueCollection реализуются как обобщенные, так и необобщенные формы интерфейсов

ICoirection и IEnumerable.

И наконец, в классе Dictionary<TKey, TValue> реализуется приведенный ниже

индексатор, определенный в интерфейсе IDictionary<TKey, TValue>.

public TValue this[TKey key] { get; set; }

Этот индексатор служит для получения и установки значения элемента коллекции,

а также для добавления в коллекцию нового элемента. Но в качестве индекса в данном

случае служит ключ элемента, а не сам индекс.

При перечислении коллекции типа Dictionary<TKey, TValue> из нее возвращаются

пары "ключ-значение" в форме структуры KeyValuePair<TKey, TValue>.

Напомним, что в этой структуре определяются два поля.

public TKey Key;

public TValue Value;

В этих полях содержится ключ или значение соответствующего элемента коллекции.

Как правило, структура KeyValuePair<TKey, TValue> не используется непосредственно,

поскольку средства класса Dictionary<TKey, TValue> позволяют

работать с ключами и значениями по отдельности. Но при перечислении коллекции

типа Dictionary<TKey, TValue>, например, в цикле foreach перечисляемыми

объектами являются пары типа KeyValuePair.

Все ключи в коллекции типа Dictionary<TKey, TValue> должны быть уникальными,

причем ключ не должен изменяться до тех пор, пока он служит в качестве

ключа. В то же время значения не обязательно должны быть уникальными. К тому же

объекты не хранятся в коллекции типа Dictionary<TKey, TValue> в отсортированном

порядке.

(***glava25\_11***)

class GenDictionaryDemo

{

static void Main()

{

//make dictionary for name lastname

//employes and sallary

Dictionary<string, double> dict =

new Dictionary<string, double>();

dict.Add("Batler, John", 73000);

dict.Add("Shwarz, Sarah", 59000);

dict.Add("Pike, Thomas", 45000);

dict.Add("Frank, Ed", 99000);

//get key collection lastname name

ICollection<string> c = dict.Keys;

//use keys to get sallary

foreach (var str in c)

Console.WriteLine("{0}, salarry: {1:C}", str, dict[str]);

}

}