System.Collections.Generic.Dictionary<TKey, TValue> Class

```
[ILAsm]
.class public serializable
System.Collections.Generic.Dictionary`2<TKey,TValue> extends System.Object
implements class System.Collections.Generic.IDictionary`2<!0,!1>,
System.Collections.Generic.ICollection`1<valuetype
System.Collections.Generic.KeyValuePair`2<!0,!1>>,
System.Collections.Generic.IEnumerable`1<valuetype
System.Collections.Generic.KeyValuePair`2<!0,!1>>,
System.Collections.IDictionary, System.Collections.ICollection,
System.Collections.IEnumerable

[C#]
public class Dictionary<TKey,TValue>: IDictionary<TKey,TValue>,
ICollection<KeyValuePair<TKey,TValue>>,
IEnumerable<KeyValuePair<TKey,TValue>>, IDictionary, ICollection,
IEnumerable
```

Assembly Info:

- *Name:* mscorlib
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- *Version:* 2.0.x.x
- *Attributes:*
    - CLSCompliantAttribute(true)

Implements:

- **System.Collections.ICollection**
- **System.Collections.IDictionary**
- **System.Collections.IEnumerable**
- **System.Collections.Generic.ICollection<KeyValuePair<TKey,TValue>>**
- **System.Collections.Generic.IDictionary<TKey,TValue>**
- **System.Collections.Generic.IEnumerable<KeyValuePair<TKey,TValue>>**

Summary

Represents a collection of key/value pairs that are organized based on the key.

Inherits From: System.Object

Library: BCL

Thread Safety: Static members of this type are thread safe. Any instance members are not guaranteed to be thread safe. A dictionary can support multiple readers concurrently, as long as the collection is not modified. Even so, enumerating through a collection is intrinsically not a thread-safe procedure. To guarantee thread safety during enumeration,

you can lock the collection during the entire enumeration. To allow the collection to be accessed by multiple threads for reading and writing, you must implement your own synchronization.

Description

Each element is a key/value pair that can be retrieved as a `System.Collections.Generic.KeyValuePair<TKey,TValue>` object.

`System.Collections.Generic.Dictionary<TKey,TValue>` requires an equality comparer implementation to perform comparisons. If no equality comparer is provided, the following default equality comparer approach is used: If type `TKey` implements `System.IEquatable<TKey>`, that implementation is used; otherwise, `TKey`'s implementations of `System.Object.Equals` and `System.Object.GetHashCode` are used. In any case, you can specify a `System.Collections.Generic.IEqualityComparer<TKey>` implementation in a constructor overload that accepts an equality comparer parameter.

After its insertion in a dictionary, changes to the value of a key that affect the equality comparer render the dictionary's behavior unspecified. Every key in a dictionary must be unique according to the equality comparer. A key cannot be `null`, but a value can be, if the value type `TValue` is a reference type.

The capacity of a dictionary is the number of elements that dictionary can hold. As elements are added to a dictionary, the capacity is automatically increased.

This type contains a member that is a nested type, called `Enumerator`. Although `Enumerator` is a member of this type, `Enumerator` is not described here; instead, it is described in its own entry, `Dictionary<TKey,TValue>.Enumerator`.

Dictionary<TKey,TValue>() Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor()

[C#]
public Dictionary()
```

Summary

Initializes a new dictionary that is empty, has the default initial capacity, and uses the default equality comparer.

Dictionary<TKey,TValue>(System.Collections.Generic.IEqualityComparer<TKey>) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(class
System.Collections.Generic.IEqualityComparer`1<!0> comparer)

[C#]
public Dictionary(IEqualityComparer<TKey> comparer)
```

Summary

Initializes a new dictionary that is empty, has the default initial capacity, and uses the specified equality comparer.

Parameters

Parameter	Description
comparer	The equality comparer implementation to use when comparing keys. -or- `null` to use the default equality comparer for the type of the key.

Dictionary<TKey,TValue>(System.Collections.Generic.IDictionary<TKey,TValue>) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(class
System.Collections.Generic.IDictionary`2<!0,!1> dictionary)

[C#]
public Dictionary(IDictionary<TKey,TValue> dictionary)
```

Summary

Initializes a new dictionary that contains elements copied from the specified dictionary, has sufficient capacity to accommodate the number of elements copied, and uses the default equality comparer.

Parameters

Parameter	Description
dictionary	The dictionary whose elements are to be copied to the new dictionary.

Description

Every key in a dictionary must be unique according to the default equality comparer; otherwise, a `System.ArgumentException` is thrown; likewise, every key in the source dictionary must also be unique according to the default equality comparer.

Exceptions

Exception	Condition
System.ArgumentException	*dictionary* contains one or more duplicate keys.
System.ArgumentNullException	*dictionary* is `null`.

Dictionary<TKey,TValue>(System.Int32) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(int32 capacity)

[C#]
public Dictionary(int capacity)
```

Summary

Initializes a new dictionary that is empty, has the specified initial capacity, and uses the default equality comparer.

Parameters

Parameter	Description
capacity	The initial number of elements that the dictionary can contain.

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	*capacity* is less than zero.

Dictionary<TKey,TValue>(System.Collections.Generic.IDictionary<TKey,TValue>, System.Collections.Generic.IEqualityComparer<TKey>) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(class
System.Collections.Generic.IDictionary`2<!0,!1> dictionary, class
System.Collections.Generic.IEqualityComparer`1<!0> comparer)

[C#]
public Dictionary(IDictionary<TKey,TValue> dictionary,
IEqualityComparer<TKey> comparer)
```

Summary

Initializes a new dictionary that contains elements copied from the specified dictionary, has sufficient capacity to accommodate the number of elements copied, and uses the specified equality comparer.

Parameters

Parameter	Description
dictionary	The dictionary whose elements are to be copied to the new dictionary.
comparer	The equality comparer implementation to use when comparing keys. -or- `null` to use the default equality comparer for the type of the key.

Description

Every key in a dictionary must be unique according to the specified; otherwise, a `System.ArgumentException` is thrown; likewise, every key in the source dictionary must also be unique according to the specified equality comparer.

Exceptions

Exception	Condition

System.ArgumentException	*dictionary* contains one or more duplicate keys.
System.ArgumentNullException	*dictionary* is `null`.

Dictionary<TKey,TValue>(System.Int32, System.Collections.Generic.IEqualityComparer<TKey>) Constructor

```
[ILAsm]
public rtspecialname specialname instance void .ctor(int32 capacity, class
System.Collections.Generic.IEqualityComparer`1<!0> comparer)

[C#]
public Dictionary(int capacity, IEqualityComparer<TKey> comparer)
```

Summary

Initializes a new dictionary that is empty, has the specified initial capacity, and uses the specified equality comparer.

Parameters

Parameter	Description
capacity	The initial number of elements that the dictionary can contain.
comparer	The equality comparer implementation to use when comparing keys. -or- `null` to use the default equality comparer for the type of the key.

Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	*capacity* is less than zero.

Dictionary<TKey,TValue>.Add(TKey, TValue) Method

```
[ILAsm]
.method public hidebysig void Add(!0 key, !1 value)

[C#]
public void Add(TKey key, TValue value)
```

Summary

Adds an element with the specified key and value to the dictionary.

Parameters

Parameter	Description
key	The key of the element to add to the dictionary.
value	The value of the element to add to the dictionary.

Description

You can also use the `System.Collections.Generic.Dictionary<TKey,TValue>.Item(TKey)` property to add new elements by setting the value of a key that does not exist in the dictionary. However, if the specified key already exists in the dictionary, setting the `System.Collections.Generic.Dictionary<TKey,TValue>.Item(TKey)` property overwrites the old value. In contrast, the `System.Collections.Generic.Dictionary<TKey,TValue>.Add(TKey,TValue)` method does not modify existing elements.

If `System.Collections.Generic.Dictionary<TKey,TValue>.Count` already equals the capacity, the capacity of the dictionary is increased.

A key cannot be `null`, but a value can be, if the value type TValue is a reference type.

Exceptions

Exception	Condition
System.ArgumentException	An element with the same key already exists in the dictionary.
System.ArgumentNullException	*key* is `null`.

1

2

Dictionary<TKey,TValue>.Clear() Method

```
[ILAsm]
.method public hidebysig void Clear()

[C#]
public void Clear()
```

Summary

Removes all elements from the dictionary.

Description

[*Note:* This method is implemented to support the `System.Collections.IDictionary` interface.

]

`System.Collections.Generic.ICollection<TKey>.Count` gets set to zero, and references to other objects from elements of the collection are also released. The capacity remains unchanged.

Dictionary<TKey,TValue>.ContainsKey(TKey) Method

```
[ILAsm]
.method public hidebysig virtual bool Contains(!0 key)

[C#]
public virtual bool ContainsKey(TKey key)
```

Summary

Determines whether the dictionary contains an element with a specific key.

Parameters

Parameter	Description
key	The key to locate in the dictionary.

Return Value

`true`, if an element whose key is *key* is found in the dictionary; otherwise, `false`.

Description

This implementation is close to O(1) in most cases.

Exceptions

Exception	Condition
System.ArgumentNullException	*key* is `null`.

Dictionary<TKey,TValue>.ContainsValue(TValue) Method

```
[ILAsm]
.method public hidebysig bool Contains(!1 value)

[C#]
public bool ContainsValue(TValue value)
```

Summary

Determines whether the dictionary contains an element with a specific value.

Parameters

Parameter	Description
value	The value to locate in the dictionary.

Return Value

`true`, if an element whose value is *value* is found in the dictionary; otherwise, `false`.

Description

This method determines equality using the default equality comparer for the value type TValue. If TValue implements `System.IEquatable<TValue>`, that type is used. Otherwise, `System.Object.Equals` is used.

This method performs a linear search; therefore, the average execution time is proportional to `System.Collections.Generic.Dictionary<TKey,TValue>.Count`. That is, this method is an O(n) operation, where n is `System.Collections.Generic.Dictionary<TKey,TValue>.Count`.

Dictionary<TKey,TValue>.GetEnumerator() Method

```
[ILAsm]
.method public hidebysig virtual class
System.Collections.Generic.Dictionary`2/Enumerator<!0,!1> GetEnumerator()

[C#]
public Dictionary<TKey,TValue>.Enumerator GetEnumerator()
```

Summary

Returns an enumerator that can be used to iterate over the dictionary.

Return Value

An enumerator for the dictionary.

Usage

For a detailed description regarding the use of an enumerator, see `System.Collections.Generic.IEnumerator<TKey>`.

Dictionary<TKey,TValue>.Remove(TKey) Method

```
[ILAsm]
.method public hidebysig bool Remove(!0 key)

[C#]
public bool Remove(TKey key)
```

Summary

Removes the element with the specified key from the dictionary.

Parameters

Parameter	Description
key	The key of the element to be removed from the dictionary.

Return Value

`true` if the element containing *key* is successfully removed; otherwise, `false`. This method also returns `false` if *key* was not found in the dictionary.

Description

If the dictionary does not contain an element with the specified key, the dictionary remains unchanged. No exception is thrown.

This method shall be implemented with efficiency that is at least an O(1) operation.

Exceptions

Exception	Condition
System.ArgumentNullException	*key* is `null`.

Dictionary<TKey,TValue>.System.Collections.Generic.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>.Add(System.Collections.Generic.KeyValuePair<TKey,TValue>) Method

```
[ILAsm]
.method private hidebysig virtual final void class
System.Collections.Generic.ICollection`1<System.Collections.Generic.KeyValuePair`2<!0,!1>>.Add(class
System.Collections.Generic.KeyValuePair`2<!0,!1> keyValuePair)

[C#]
void ICollection<KeyValuePair<TKey,TValue>>.Add(KeyValuePair<TKey,TValue>
keyValuePair)
```

Summary

This method is implemented to support the `System.Collections.Generics.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>` interface.

Dictionary<TKey,TValue>.System.Collections.Generic.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>.Contains(System.Collections.Generic.KeyValuePair<TKey,TValue>) Method

```
[ILAsm]
.method private hidebysig virtual final bool
System.Collections.Generic.ICollection`1<System.Collections.Generic.KeyValuePair`2<!0,!1>>.Contains(class
System.Collections.Generic.KeyValuePair`2<!0,!1> keyValuePair)

[C#]
bool
ICollection<KeyValuePair<TKey,TValue>>.Contains(KeyValuePair<TKey,TValue>
keyValuePair)
```

Summary

This method is implemented to support the `System.Collections.Generics.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>` interface.

Dictionary<TKey,TValue>.System.Collections.Generic.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>.CopyTo(System.Collections.Generic.KeyValuePair<TKey,TValue>[], System.Int32) Method

```
[ILAsm]
.method private hidebysig virtual final void
System.Collections.Generic.ICollection`1<System.Collections.Generic.KeyValuePair`2<!0,!1>>.CopyTo(class
System.Collections.Generic.KeyValuePair`2<!0,!1>[] array, int32 index)

[C#]
void
ICollection<KeyValuePair<TKey,TValue>>.CopyTo(KeyValuePair<TKey,TValue>[]
array, int index)
```

Summary

This method is implemented to support the `System.Collections.Generics.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>` interface.

Dictionary<TKey,TValue>.System.Collections.Generic.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>.Remove(System.Collections.Generic.KeyValuePair<TKey,TValue>) Method

```
[ILAsm]
.method private hidebysig virtual final bool
System.Collections.Generic.ICollection`1<System.Collections.Generic.KeyValuePair`2<!0,!1>>.Remove(class
System.Collections.Generic.KeyValuePair`2<!0,!1> keyValuePair)

[C#]
bool
ICollection<KeyValuePair<TKey,TValue>>.Remove(KeyValuePair<TKey,TValue>
keyValuePair)
```

Summary

This method is implemented to support the `System.Collections.Generics.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>` interface.

Dictionary<TKey,TValue>.System.Collections.Generic.IEnumerable<System.Collections.Generic.KeyValuePair<TKey,TValue>>.GetEnumerator() Method

```
[ILAsm]
.method private hidebysig virtual abstract class
System.Collections.Generic.IEnumerator<T>
System.Collections.Generic.IEnumerable<System.Collections.Generic.KeyValue
Pair`2<!0,!1>>.GetEnumerator()

[C#]
IEnumerator<T> IEnumerable<KeyValuePair<TKey,TValue>>.GetEnumerator()
```

Summary

This method is implemented to support the `System.Collections.Generics.IEnumerable<System.Collections.Generic.KeyValuePair<TKey,TValue>>` interface.

Dictionary<TKey,TValue>.System.Collections.ICollection.CopyTo(System.Array, System.Int32) Method

```
[ILAsm]
.method private hidebysig virtual final void
System.Collections.ICollection.CopyTo(class System.Array array, int32
index)

[C#]
void ICollection.CopyTo(Array array, int index)
```

Summary

This method is implemented to support the `System.Collections.ICollection` interface.

Dictionary<TKey,TValue>.System.Collections.IDictionary.Add(System.Object, System.Object) Method

```
[ILAsm]
.method private hidebysig virtual final void
System.Collections.IDictionary.Add(object key, object value)

[C#]
void IDictionary.Add(object key, object value)
```

Summary

This method is implemented to support the `System.Collections.IDictionary` interface.

Dictionary<TKey,TValue>.System.Collections.IDictionary.Contains(System.Object) Method

```
[ILAsm]
.method private hidebysig virtual final bool
System.Collections.IDictionary.Contains(object key)

[C#]
bool IDictionary.Contains(object key)
```

Summary

This method is implemented to support the `System.Collections.IDictionary` interface.

Dictionary<TKey,TValue>.System.Collections.IDictionary.GetEnumerator() Method

```
[ILAsm]
.method private hidebysig virtual final class
System.Collections.IDictionaryEnumerator
System.Collections.IDictionary.GetEnumerator()

[C#]
IDictionaryEnumerator IDictionary.GetEnumerator()
```

Summary

This method is implemented to support the `System.Collections.IDictionary` interface.

Dictionary<TKey,TValue>.System.Collections.IDictionary.Remove(System.Object) Method

```
[ILAsm]
.method private hidebysig virtual final void
System.Collections.IDictionary.Remove(object key)

[C#]
void IDictionary.Remove(object key)
```

Summary

This method is implemented to support the `System.Collections.IDictionary` interface.

Dictionary<TKey,TValue>.System.Collections.IEnumerable.GetEnumerator() Method

```
[ILAsm]
.method private hidebysig virtual final class
System.Collections.IEnumerator
System.Collections.IEnumerable.GetEnumerator()

[C#]
IEnumerator IEnumerable.GetEnumerator()
```

Summary

This method is implemented to support the `System.Collections.IEnumerable` interface.

Dictionary<TKey,TValue>.TryGetValue(TKey, TValue) Method

```
[ILAsm]
.method public hidebysig bool TryGetValue(!0 key, [out] !1 value)

[C#]
public bool TryGetValue(TKey key, out TValue value)
```

Summary

Gets the value associated with the specified key.

Parameters

Parameter	Description
key	The key of the element to locate in the dictionary.
value	When this method returns, the value associated with the specified key, if the key is found; otherwise, the default value for the type of this parameter.

Return Value

`true` if the dictionary contains an element with the specified key; otherwise, `false`.

Description

This method combines the functionality of the `System.Collections.Generic.Dictionary<TKey,TValue>.ContainsKey` method and the `System.Collections.Generic.Dictionary<TKey,TValue>.Item` property.

The default value for value types is zero while that for reference types is `null`.

This method is an O(1) operation.

Exceptions

Exception	Condition
System.ArgumentNullException	*key* is `null`.

Dictionary<TKey,TValue>.Count Property

```
[ILAsm]
.property int32 Count { public hidebysig specialname instance int32
get_Count () }

[C#]
public int Count { get; }
```

Summary

Gets the number of key/value pairs contained in the dictionary.

Property Value

The number of key/value pairs contained in the dictionary.

Description

This property is read-only.

Retrieving the value of this property is an O(1) operation.

Dictionary<TKey,TValue>.Item Property

```
[ILAsm]
.property !1 Item[!0 key] { public hidebysig specialname !1 get_Item(!0
key) public hidebysig specialname void set_Item(!0 key, !1 value) }

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

Summary

Gets or sets the value associated with the specified key.

Parameters

Parameter	Description
key	The key whose value is to be gotten or set.

Property Value

The value associated with the specified key. On a get attempt, if the specified key is not found, a `System.Collections.Generic.KeyNotFoundException` is thrown. On a set attempt, if the specified key is not found, a new element using the specified key is created.

Description

The default value for value types is zero while that for reference types is `null`.

You can also use the `System.Collections.Generic.Dictionary<TKey,TValue>.Item(TKey)` property to add new elements by setting the value of a key that does not exist in the dictionary. However, if the specified key already exists in the dictionary, setting the `System.Collections.Generic.Dictionary<TKey,TValue>.Item(TKey)` property overwrites the old value. In contrast, the `System.Collections.Generic.Dictionary<TKey,TValue>.Add(TKey,TValue)` method does not modify existing elements.

A key cannot be `null`, but a value can be, if the value type TValue is a reference type.

Retrieving the value of this property is an O(1) operation; setting the property is also an O(1) operation.

Exceptions

Exception	Condition

System.ArgumentNullException	*key* is `null`.
System.Collections.Generic.KeyNotFoundException	During a get attempt, *key* is not found in the dictionary.

1

2

Dictionary<TKey,TValue>.Keys Property

```
[ILAsm]
.property class
System.Collections.Generic.Dictionary`2/KeyCollection<!0,!1> Keys { public
hidebysig specialname instance class
System.Collections.Generic.Dictionary`2/KeyCollection<!0,!1> get_Keys() }

[C#]
public KeyCollection<TKey,TValue> Keys { get; }
```

Summary

Gets a collection that contains the keys in the dictionary.

Property Value

A collection of the keys in the dictionary.

Description

This property is read-only.

The order of the keys in the key collection is unspecified, but it is the same order as the associated values in the value collection returned by the `System.Collections.Generic.Dictionary<TKey,TValue>.Values` property.

If the dictionary is modified, or the value of any key in the dictionary is modified, the behavior of the key collection is unspecified.

Dictionary<TKey,TValue>.System.Collections.Generic.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>.IsReadOnly Property

```
[ILAsm]
.property bool
System.Collections.Generic.ICollection<System.Collections.Generic.KeyValue
Pair`2<!0,!1>>.IsReadOnly { private hidebysig virtual final specialname
bool get_IsReadOnly() }

[C#]
bool ICollection<KeyValuePair<TKey,TValue>>.IsReadOnly { get; }
```

Summary

This read-only property is implemented to support the `System.Collections.Generics.ICollection<System.Collections.Generic.KeyValuePair<TKey,TValue>>` interface.

Dictionary<TKey,TValue>.System.Collections.Generic.IDictionary<TKey,TValue>.Keys Property

```
[ILAsm]
.property class System.Collections.Generic.ICollection`1<!0>
System.Collections.Generic.IDictionary`2<!0,!1>.Keys { private hidebysig
virtual final specialname class
System.Collections.Generic.ICollection`1<!0> get_Keys() }

[C#]
ICollection<TKey> IDictionary<TKey,TValue>.Keys { get; }
```

Summary

This read-only property is implemented to support the `System.Collections.Generic.IDictionary<TKey,TValue>` interface.

Dictionary<TKey,TValue>.System.Collections.Generic.IDictionary<TKey,TValue>.Values Property

```
[ILAsm]
.property class System.Collections.Generic.ICollection`1<!1>
System.Collections.Generic.IDictionary`2<!0,!1>.Values { private hidebysig
virtual final specialname class
System.Collections.Generic.ICollection`1<!1> get_Values() }

[C#]
ICollection<TValue> IDictionary<TKey,TValue>.Values { get; }
```

Summary

This read-only property is implemented to support the `System.Collections.Generic.IDictionary<TKey,TValue>` interface.

Dictionary<TKey,TValue>.System.Collections.ICollection.IsSynchronized Property

```
[ILAsm]
.property bool System.Collections.ICollection.IsSynchronized { private
hidebysig virtual final bool get_IsSynchronized() }

[C#]
bool ICollection.IsSynchronized { get; }
```

Summary

This read-only property is implemented to support the `System.Collections.ICollection` interface.

Dictionary<TKey,TValue>.System.Collections.ICollection.SyncRoot Property

```
[ILAsm]
.property object System.Collections.ICollection.SyncRoot { private
hidebysig virtual final object get_SyncRoot() }

[C#]
object ICollection.SyncRoot { get; }
```

Summary

This read-only property is implemented to support the `System.Collections.ICollection` interface.

Dictionary<TKey,TValue>.System.Collections.IDictionary.IsFixedSize Property

```
[ILAsm]
.property bool System.Collections.IDictionary.IsFixedSize { private
hidebysig virtual final specialname bool get_IsFixedSize() }

[C#]
bool IDictionary.IsFixedSize { get; }
```

Summary

This read-only property is implemented to support the `System.Collections.IDictionary` interface.

Dictionary<TKey,TValue>.System.Collections.IDictionary.IsReadOnly Property

```
[ILAsm]
.property bool System.Collections.IDictionary.IsReadOnly { private
hidebysig virtual final specialname bool get_IsReadOnly() }

[C#]
bool IDictionary.IsReadOnly { get; }
```

Summary

This read-only property is implemented to support the `System.Collections.IDictionary` interface.

Dictionary<TKey,TValue>.System.Collections.IDictionary.Item Property

```
[ILAsm]
.property object System.Collections.IDictionary.Item(object key) { private
hidebysig virtual final specialname object get_Item(object key) private
hidebysig virtual final specialname void set_Item(object key, object
value) }

[C#]
object IDictionary.this[object key] { get; set; }
```

Summary

This read-only property is implemented to support the `System.Collections.IDictionary` interface.

Dictionary<TKey,TValue>.System.Collections.IDictionary.Keys Property

```
[ILAsm]
.property class System.Collections.ICollection
System.Collections.IDictionary.Keys { private hidebysig virtual final
specialname class System.Collections.ICollection get_Keys() }

[C#]
ICollection IDictionary.Keys { get; }
```

Summary

This read-only property is implemented to support the `System.Collections.IDictionary` interface.

Dictionary<TKey,TValue>.System.Collections.IDictionary.Values Property

```
[ILAsm]
.property class System.Collections.ICollection
System.Collections.IDictionary.Values { private hidebysig virtual final
specialname class System.Collections.ICollection get_Values() }

[C#]
ICollection IDictionary.Values { get; }
```

Summary

This read-only property is implemented to support the `System.Collections.IDictionary` interface.

Dictionary<TKey,TValue>.Values Property

```
[ILAsm]
.property class
System.Collections.Generic.Dictionary`2/ValueCollection<!0,!1> Values {
public hidebysig specialname instance class
System.Collections.Generic.Dictionary`2/ValueCollection<!0,!1>
get_Values() }

[C#]
public ValueCollection<TKey,TValue> Values { get; }
```

Summary

Gets a collection that contains the values in the dictionary.

Property Value

A collection of the values in the dictionary.

Description

This property is read-only.

The order of the values in the value collection is unspecified, but it is the same order as the associated values in the key collection returned by the `System.Collections.Generic.Dictionary<TKey,TValue>.Keys` property.

The returned value collection is not a static copy; instead, it refers back to the values in the original dictionary. Therefore, changes to the dictionary continue to be reflected in the value collection.