TutorialKart

C# Dictionary

C# Dictionary

In C#, Dictionary is a collection of <Key, Value> pairs.

In a C# Dictionary, Key has to be unique. Value could be a duplicate.

Define a C# Dictionary

To use a Dictionary, you have to use System.Collections.Generic namespace. To define a Dictionary, use Dictionary class with the following syntax.

```
Dictionary<keyDatatype, valueDatatype> dictionaryName = new Dictionary<keyDatatype, valu
```

Example

In the following example, we defined two C# dictionaries named `dict1` and `dict2`.

Program.cs

```
using System.Collections.Generic;

namespace CSharpExamples {
    class Program {
        static void Main(string[] args) {
            Dictionary<int, string> dict1 = new Dictionary<int, string>();
            Dictionary<string, string> dict2 = new Dictionary<string, string>();
        }
    }
}
```

Dictionary `dict1` stores key-value pairs where key is of `int` datatype and value is of `string` datatype.

Dictionary `dict2` stores key-value pairs where key is of `string` datatype and value is of `string` datatype.

Initialize C# Dictionary with Entries

There are two methods to initialize a C# Dictionary.

The first method is to define a dictionary and add elements to it using Dictionary.Add() method.

Program.cs

```
using System.Collections.Generic;

namespace CSharpExamples {
    class Program {
        static void Main(string[] args) {
            Dictionary<int, string> dict1 = new Dictionary<int, string>();
            dict1.Add(1,"Tesla");
            dict1.Add(2,"Toyota");
        }
    }
}
```

The second method is to provide the key-value pairs in the definition as shown below.

Program.cs

```
using System.Collections.Generic;

namespace CSharpExamples {
    class Program {
        static void Main(string[] args) {
            Dictionary<int, string> dict1 = new Dictionary<int, string>(){
                {1, "Tesla"},
                {2, "Honda"},
                {3, "Toyota"}
            };
        }
    }
}
```

Print C# Dictionary

You can use foreach to access each of the item in Dictionary. In the following example, we access each item of the dictionary, and then get the key and values separately.

Program.cs

```
using System;
using System.Collections.Generic;

namespace CSharpExamples {
    class Program {
        static void Main(string[] args) {
```

```
            Dictionary<int, string> dict1 = new Dictionary<int, string>(){
                {1, "Tesla"},
                {2, "Honda"},
                {3, "Toyota"}
            };

            foreach(KeyValuePair<int, string> item in dict1) {
                Console.WriteLine(item.Key+" - "+item.Value);
            }
        }
    }
}
```

Output

```
PS D:\workspace\csharp\HelloWorld> dotnet run
1 - Tesla
2 - Honda
3 - Toyota
```

Count Number of Elements in Dictionary

To get the number of elements in a Dictionary, you can use Dictionary.Count property.

Program.cs

```
using System;
using System.Collections.Generic;

namespace CSharpExamples {
    class Program {
        static void Main(string[] args) {
            Dictionary<int, string> dict1 = new Dictionary<int, string>(){
                {1, "Tesla"},
                {2, "Honda"},
                {3, "Toyota"}
            };
            int count = dict1.Count;
            Console.WriteLine("Number of items in Dictionary : "+count);
        }
    }
}
```

Output

```
PS D:\workspace\csharp\HelloWorld> dotnet run
Number of items in Dictionary : 3
```

Update a Value for Key in Dictionary

To update a value for a key in dictionary, you can use the key as an index and dictionary as an array.

```
myDictionary[key] = value;
```

Program.cs

```
using System;
using System.Collections.Generic;

namespace CSharpExamples {
    class Program {
        static void Main(string[] args) {
            Dictionary<int, string> dict1 = new Dictionary<int, string>(){
                {1, "Tesla"},
                {2, "Honda"},
                {3, "Toyota"}
            };

            dict1[2] = "Volkswagen";

            foreach(KeyValuePair<int, string> item in dict1) {
                Console.WriteLine(item.Key+" - "+item.Value);
            }
        }
    }
}
```

Output

```
PS D:\workspace\csharp\HelloWorld> dotnet run
1 - Tesla
2 - Volkswagen
3 - Toyota
```

Delete an Item from C# Dictionary

To delete an Item from C# Dictionary, use `Dictionary.Remove(key)` method. The item with key passed as argument, shall be deleted.

Program.cs

```
using System;
using System.Collections.Generic;

namespace CSharpExamples {
    class Program {
        static void Main(string[] args) {
            Dictionary<int, string> dict1 = new Dictionary<int, string>(){
                {1, "Tesla"},
                {2, "Honda"},
                {3, "Toyota"}
            };

            dict1.Remove(2);
```

```
            foreach(KeyValuePair<int, string> item in dict1) {
                Console.WriteLine(item.Key+" - "+item.Value);
            }
        }
    }
}
```

Output

```
PS D:\workspace\csharp\HelloWorld> dotnet run
1 - Tesla
3 - Toyota
```

Conclusion

In this C# Tutorial, we learned what a Dictionary is in C#, how to define a dictionary and initialize it, access the items, modify or remove some of the items, etc.

C# Collections

- C# List
- C# SortedList
- C# HashSet
- C# SortedSet
- C# Stack
- C# Queue
- C# LinkedList
- **C# Dicionary**
- C# SortedDictionary

C# Errors [Solved]

- C# Error: Class does not contain a constructor that takes arguments