



# LES COLLECTIONS C#

TP

# LES COLLECTIONS

Les types de collections C# sont conçus pour stocker, gérer et manipuler des données similaires plus efficacement.

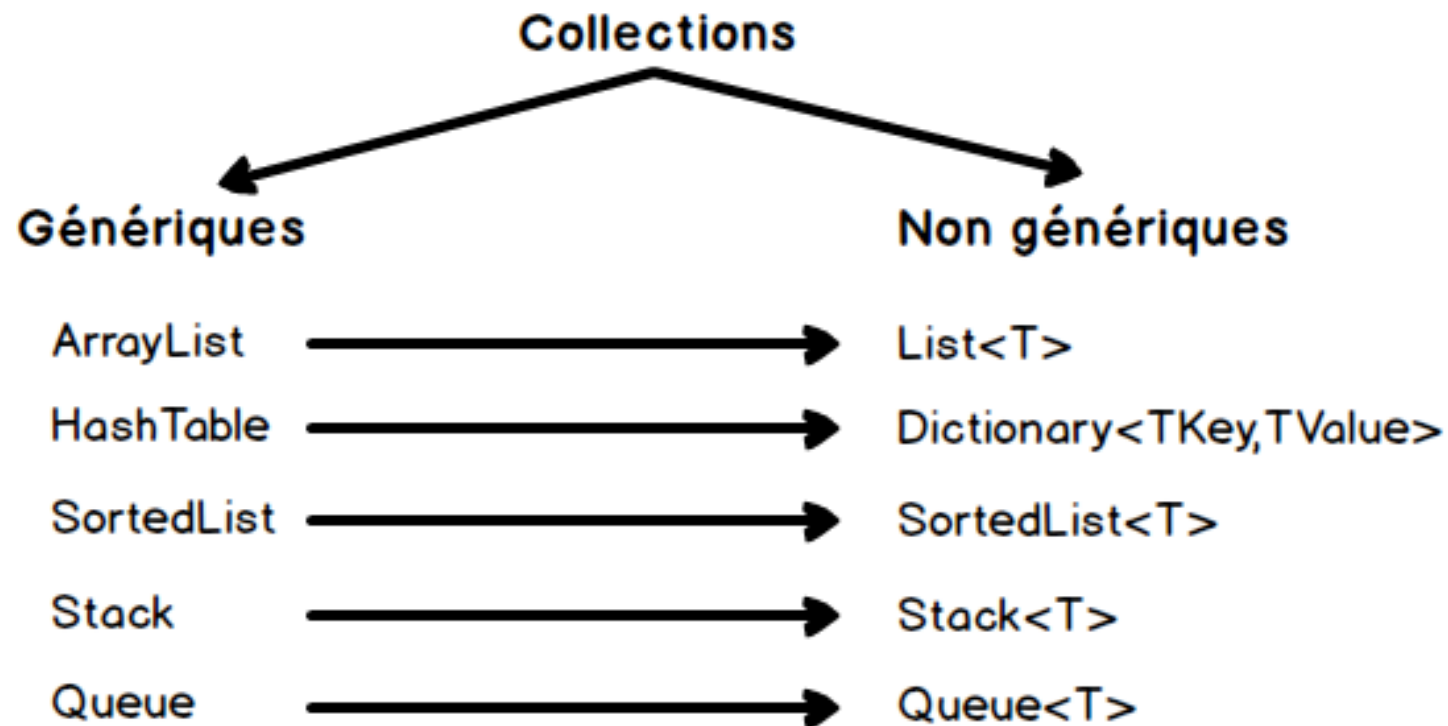
permettent de regrouper et gérer plusieurs objets de taille et/ou de type dynamiques (des tableaux multi-types extensibles). Les types de collection implémentent les fonctionnalités communes suivantes:

- Ajouter et insérer des éléments dans une collection
- Supprimer des éléments d'une collection
- Recherche, tri, recherche d'articles
- Remplacement d'éléments
- Copier et cloner des éléments
- Propriétés Capacité et Nombre pour trouver la capacité de la collection et le nombre d'éléments dans la collection

.NET prend en charge deux types de collections, les collections génériques et les collections non génériques

# LES COLLECTIONS

- Elles sont définies dans le NameSpace **System.Collections**;
- Elles implémentent l'interface **System.Collections.ICollection**



# COLLECTIONS NON GÉNÉRIQUES

Dans les collections non génériques, chaque élément peut représenter une valeur d'un type différent. La taille de la collection n'est pas fixe. Les éléments de la collection peuvent être ajoutés ou supprimés lors de l'exécution.

## ArrayList

La classe ArrayList est une collection qui peut être utilisée pour n'importe quel type ou objet.

- ArrayList est une classe similaire à un tableau, mais elle peut être utilisée pour stocker des valeurs de différents types.
- Un ArrayList n'a pas de taille spécifique.
- N'importe quel nombre d'éléments peut être stocké.

```
using System.Collections;

protected void Button1_Click(object sender, EventArgs e)
{
    ArrayList al = new ArrayList();
    string str = "kiran teja jallepalli";
    int x = 7;
    DateTime d = DateTime.Parse("8-oct-1985");
    al.Add(str);
    al.Add(x);
    al.Add(d);

    foreach (object o in al)
    {
        Response.Write(o);
        Response.Write("<br>");
    }
}
```

# COLLECTIONS NON GÉNÉRIQUES

## HashTable

HashTable est similaire à arraylist mais **représente les éléments comme une combinaison d'une clé et d'une valeur.**

```
using System.Collections;
```

```
protected void Button2_Click(object sender, EventArgs e)
{
    Hashtable ht = new Hashtable();
    ht.Add("ora", "oracle");
    ht.Add("vb", "vb.net");
    ht.Add("cs", "cs.net");
    ht.Add("asp", "asp.net");

    foreach (DictionaryEntry d in ht)
    {
        Response.Write (d.Key + " " + d.Value);

        Response.Write("<br>");
    }
}
```

### Output

```
vb vb.net
asp asp.net
cs cs.net
ora oracle
```

# COLLECTIONS NON GÉNÉRIQUES

## SortedList

- Représente les données sous forme de paire clé / valeur.
- Organise tous les éléments dans un ordre trié..

```
using System.Collections;
```

```
protected void Button3_Click(object sender, EventArgs e)
{
    SortedList sl = new SortedList();
    sl.Add("ora", "oracle");
    sl.Add("vb", "vb.net");
    sl.Add("cs", "cs.net");
    sl.Add("asp", "asp.net");

    foreach (DictionaryEntry d in sl)
    {
        Response.Write(d.Key + " " + d.Value);
        Response.Write("<br>");
    }
}
```

## Output

```
asp asp.net
cs cs.net
ora oracle
vb vb.net
```

# COLLECTIONS NON GÉNÉRIQUES

## Stack

Représente une simple collection non générique d'objets de type dernier entré, premier sorti (LIFO)

```
protected void Button4_Click(object sender, EventArgs e)
{
    Stack stk = new Stack();
    stk.Push("cs.net");
    stk.Push("vb.net");
    stk.Push("asp.net");
    stk.Push("sqlserver");

    foreach (object o in stk)
    {
        Response.Write(o + "<br>");
    }
}
```

## Output

sqlserver  
asp.net  
vb.net  
cs.net

# COLLECTIONS NON GÉNÉRIQUES

## Queue

Représente une collection d'objets premier entré, premier sorti (FIFO)

```
using System.Collections;
```

```
protected void Button5_Click(object sender, EventArgs e)
{
    Queue q = new Queue();
    q.Enqueue("cs.net");
    q.Enqueue("vb.net");
    q.Enqueue("asp.net");
    q.Enqueue("sqlserver");

    foreach (object o in q)
    {
        Response.Write(o + "<br>");
    }
}
```

## Output

```
cs.net
vb.net
asp.net
sqlserver
```



# COLLECTIONS GÉNÉRIQUES

Les collections génériques fonctionnent sur un type spécifique tandis que les collections non génériques fonctionnent sur le type d'objet.

- Type spécifique
- La taille du tableau n'est pas fixe
- Les éléments peuvent être ajoutés / supprimés lors de l'exécution.

## List

La classe `List` est une collection avec accès par indice très polyvalente, et performante pour la lecture et l'ajout.

```
using System.Collections.Generic;

void protégé Button1_Click ( expéditeur d' objet , EventArgs e)
{
    List < int > lst = new List < int > ();
    lst.Add (100);
    lst.Add (200);
    lst.Add (300);
    lst.Add (400);
    foreach ( int i in lst)
    {
        Response.Write (i + "<br>");
    }
}
```

# COLLECTIONS GÉNÉRIQUES

## Dictionary

C'est une collection générique qui fonctionne sur des paires de clés et de valeurs. La paire clé / valeur peut avoir différents types de données ou les mêmes types de données ou tout type personnalisé (c'est-à-dire des objets de classe)

```
using System.Collections.Generic;

protected void Button1_Click(object sender, EventArgs e)
{
    Dictionary<int, string> dct = new Dictionary<int, string>();
    dct.Add(1, "cs.net");
    dct.Add(2, "vb.net");
    dct.Add(3, "vb.net");
    dct.Add(4, "vb.net");
    foreach (KeyValuePair<int, string> kvp in dct)
    {
        Response.Write(kvp.Key + " " + kvp.Value);
        Response.Write("<br>");
    }
}
```

# COLLECTIONS GÉNÉRIQUES

## SortedList

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

```
protected void Button3_Click(object sender, EventArgs e)
{
    SortedList<string, string> sl = new SortedList<string, string>();
    sl.Add("ora", "oracle");
    sl.Add("vb", "vb.net");
    sl.Add("cs", "cs.net");
    sl.Add("asp", "asp.net");

    foreach (KeyValuePair<string, string> kvp in sl)
    {
        Response.Write(kvp.Key + " " + kvp.Value);
        Response.Write("<br>");
    }
}
```

# COLLECTIONS GÉNÉRIQUES

## Stack

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

```
protected void Button4_Click(object sender, EventArgs e)
{
    Stack<string> stk = new Stack<string>();
    stk.Push("cs.net");
    stk.Push("vb.net");
    stk.Push("asp.net");
    stk.Push("sqlserver");
}
```

```
foreach (string s in stk)
{
    Response.Write(s + "<br>");
}
}
```

# COLLECTIONS GÉNÉRIQUES

## Queue

```
using System.Collections.Generic;
protected void Button1_Click(object sender, EventArgs e)
{
    Queue<string> q = new Queue<string>();

    q.Enqueue("cs.net");
    q.Enqueue("vb.net");
    q.Enqueue("asp.net");
    q.Enqueue("sqlserver");

    foreach (string s in q)
    {
        Response.Write(s + "<br>");
    }
}
```

# EXERCICE

Reference: <https://www.dotnetperls.com/list>

Manipuler les collections de type List: Méthodes (classique de gestion de liste)

- **public int Add( object elt );** → Ajoute l'élément elt à List.
- **public void Clear( );** → Supprime tous les éléments de List.
- **public bool Contains( object elt );** → Indique si List contient l'élément elt.
- **public int IndexOf( object elt );** → Indique le rang de l'élément elt dans List.
- **public void Insert( int rang , object elt );** → Insère l'élément elt dans List à la position spécifiée par rang.
- **public void Remove( object elt );** → Supprime la première occurrence de l'objet elt de List.
- **public void RemoveAt (int rang);** → Supprime l'élément de List dont le rang est spécifié.

# EXERCICE

Soit à développer une application pour la gestion d'un stock.

Un article est caractérisé par son **numéro** de référence, son **nom**, son **prix** de vente et une **quantité** en stock.

Le stock est représenté par une collection d'articles.

## Travail à faire:

Créer la classe article contenant les éléments suivants :

Les attributs/propriétés.

Un constructeur d'initialisation.

# EXERCICE

Dans la classe Program créer :

Le stock sous forme d'une collection d'articles de votre choix.

Un menu présentant les fonctionnalités suivantes :

1. Rechercher un article par référence.
2. Ajouter un article au stock en vérifiant l'unicité de la référence.
3. Supprimer un article par référence.
4. Modifier un article par référence.
5. Rechercher un article par nom.
6. Afficher tous les articles.
7. Quitter