Generic Lists are one of the most commonly used collections in C#

Array vs List:

* Both Strongly typed, List uses generic type syntax.
* Array is fixed length and cannot be changed. A list can grow and shrink as elements are added.
* Array elements can’t be removed, only replaced. Lists elements can.
* Arrays can be multidimensional. Lists cannot.

Bottom line, lists are more flexible. Use arrays if you need to squeeze out a bit more performance or are using a large set of data with a set size.

Generic lists leverage generics so are of type T. List<T>

var Colours = new List<string>();

System.Collections.Generic;

Do:

Use generic lists to manage collections

Use Add over Insert

Use plural for name

Avoid:

Removing elements. It’s slow and inefficient. If you need to remove many elements, consider a LINQ collection.

Collection initialisers work for lists as well as arrays

Object initialisers – initialises an objects variable after it’s declaration using curly braces with member variables inside.

Object equality tests can be changed by overriding the Equals method of a class.

As – keyword to cast an object to a class type.

Foreach – although you can’t change the object at an element because the foreach var is read-only, you can change its member values.

System.Array

System.Collections.ArrayList(.NET1)

System.Collections.Generic(.NET2+)

* List<T>
* LinkedList<T>
* Queue<T>
* Stack<T>

ArrayList – stored type Object. This means type verification is not performed and it can hold any old type. Casting when retrieving is required. Obsolete and should not be used unless you’re targeting .NET1.

LinkedList – each element is linked to the element before and after it. Use it if you require a sequential list or are frequently inserting/removing from the middle of the list.

Queue – good for temporary storage and if you plan to discard value after retrieval.

Stack – same as queue but with a different retrieval order.

System.Collections.ObjectModel – contains more specialised collection types.

* ReadOnlyCollection. Can be read but not editable.
* ObservableCollection. Used for binding.
* Specialized. Speciality class collections.
* Concurrent. Thread safe list classes.

FAQ

1. When is it appropriate to use generic lists?

Use generic lists for type-safe collections of any type. For any list of things.

1. What are the key differences between an array and a generic list?

An array is of fixed size and elements cannot be deleted. A generic list can delete and insert elements and is growable and shrinkable. An array can be multidimensional. A list is one dimensional.

1. What is the difference between foreach and for loop when iterating through collections?

Foreach iterates through every element. Foreach variables are read-only. For loops can iterate through every element or a sub-set of elements. For loops can CRUD elements of a collection.

1. When using foreach on a list of objects are the foreach variables editable?

No, the object instance isn’t but it’s properties are.