.NET Framework 4

**Garbage Collection**

The .NET Framework's garbage collector manages the allocation and release of memory for your application. Each time you create a new object, the common language runtime allocates memory for the object from the managed heap. As long as address space is available in the managed heap, the runtime continues to allocate space for new objects. However, memory is not infinite. Eventually the garbage collector must perform a collection in order to free some memory. The garbage collector's optimizing engine determines the best time to perform a collection, based upon the allocations being made. When the garbage collector performs a collection, it checks for objects in the managed heap that are no longer being used by the application and performs the necessary operations to reclaim their memory.

.NET Framework Class Library

**GC Class**

Controls the system garbage collector, a service that automatically reclaims unused memory.

Description: http://i.msdn.microsoft.com/Global/Images/clear.gifInheritance Hierarchy

[System..::.Object](http://msdn.microsoft.com/en-us/library/system.object.aspx)  
**System..::.GC**

**Namespace:** [System](http://msdn.microsoft.com/en-us/library/system.aspx)  
**Assembly:** mscorlib (in mscorlib.dll)

Description: http://i.msdn.microsoft.com/Global/Images/clear.gifSyntax

Visual Basic

Public NotInheritable Class GC

C#

public static class GC

Visual C++

public ref class GC abstract sealed

F#

[<AbstractClass>]

[<Sealed>]

type GC = class end

The **GC** type exposes the following members.

Description: http://i.msdn.microsoft.com/Global/Images/clear.gifProperties

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Description: Public propertyDescription: Static memberDescription: Supported by the XNA Framework | [MaxGeneration](http://msdn.microsoft.com/en-us/library/system.gc.maxgeneration.aspx) | Gets the maximum number of generations that the system currently supports. |

[Top](http://msdn.microsoft.com/en-us/library/system.gc.aspx#mainBody)

Description: http://i.msdn.microsoft.com/Global/Images/clear.gifMethods

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Description** |
| Description: Public methodDescription: Static member | [AddMemoryPressure](http://msdn.microsoft.com/en-us/library/system.gc.addmemorypressure.aspx) | Informs the runtime of a large allocation of unmanaged memory that should be taken into account when scheduling garbage collection. |
| Description: Public methodDescription: Static member | [CancelFullGCNotification](http://msdn.microsoft.com/en-us/library/system.gc.cancelfullgcnotification.aspx) | Cancels the registration of a garbage collection notification. |
| Description: Public methodDescription: Static memberDescription: Supported by the XNA Framework | [Collect()()()](http://msdn.microsoft.com/en-us/library/xe0c2357.aspx) | Forces an immediate garbage collection of all generations. |
| Description: Public methodDescription: Static member | [Collect(Int32)](http://msdn.microsoft.com/en-us/library/y46kxc5e.aspx) | Forces an immediate garbage collection from generation zero through a specified generation. |
| Description: Public methodDescription: Static member | [Collect(Int32, GCCollectionMode)](http://msdn.microsoft.com/en-us/library/bb356724.aspx) | Forces a garbage collection from generation zero through a specified generation, at a time specified by a [GCCollectionMode](http://msdn.microsoft.com/en-us/library/bb495757.aspx) value. |
| Description: Public methodDescription: Static member | [CollectionCount](http://msdn.microsoft.com/en-us/library/system.gc.collectioncount.aspx) | Returns the number of times garbage collection has occurred for the specified generation of objects. |
| Description: Public methodDescription: Static member | [GetGeneration(Object)](http://msdn.microsoft.com/en-us/library/ts8yd2d0.aspx) | Returns the current generation number of the specified object. |
| Description: Public methodDescription: Static member | [GetGeneration(WeakReference)](http://msdn.microsoft.com/en-us/library/h28z0ez9.aspx) | Returns the current generation number of the target of a specified weak reference. |
| Description: Public methodDescription: Static memberDescription: Supported by the XNA Framework | [GetTotalMemory](http://msdn.microsoft.com/en-us/library/system.gc.gettotalmemory.aspx) | Retrieves the number of bytes currently thought to be allocated. A parameter indicates whether this method can wait a short interval before returning, to allow the system to collect garbage and finalize objects. |
| Description: Public methodDescription: Static memberDescription: Supported by the XNA FrameworkDescription: Supported by Portable Class Library | [KeepAlive](http://msdn.microsoft.com/en-us/library/system.gc.keepalive.aspx) | References the specified object, which makes it ineligible for garbage collection from the start of the current routine to the point where this method is called. |
| Description: Public methodDescription: Static member | [RegisterForFullGCNotification](http://msdn.microsoft.com/en-us/library/system.gc.registerforfullgcnotification.aspx) | Specifies that a garbage collection notification should be raised when conditions favor full garbage collection and when the collection has been completed. |
| Description: Public methodDescription: Static member | [RemoveMemoryPressure](http://msdn.microsoft.com/en-us/library/system.gc.removememorypressure.aspx) | Informs the runtime that unmanaged memory has been released and no longer needs to be taken into account when scheduling garbage collection. |
| Description: Public methodDescription: Static memberDescription: Supported by the XNA FrameworkDescription: Supported by Portable Class Library | [ReRegisterForFinalize](http://msdn.microsoft.com/en-us/library/system.gc.reregisterforfinalize.aspx) | Requests that the system call the finalizer for the specified object for which [SuppressFinalize](http://msdn.microsoft.com/en-us/library/system.gc.suppressfinalize.aspx) has previously been called. |
| Description: Public methodDescription: Static memberDescription: Supported by the XNA FrameworkDescription: Supported by Portable Class Library | [SuppressFinalize](http://msdn.microsoft.com/en-us/library/system.gc.suppressfinalize.aspx) | Requests that the system not call the finalizer for the specified object. |
| Description: Public methodDescription: Static member | [WaitForFullGCApproach()()()](http://msdn.microsoft.com/en-us/library/cc647054.aspx) | Returns the status of a registered notification for determining whether a full garbage collection by the common langauge runtime is imminent. |
| Description: Public methodDescription: Static member | [WaitForFullGCApproach(Int32)](http://msdn.microsoft.com/en-us/library/cc647167.aspx) | Returns, in a specified time-out period, the status of a registered notification for determining whether a full garbage collection by the common language runtime is imminent. |
| Description: Public methodDescription: Static member | [WaitForFullGCComplete()()()](http://msdn.microsoft.com/en-us/library/cc647068.aspx) | Returns the status of a registered notification for determining whether a full garbage collection by the common language runtime has completed. |
| Description: Public methodDescription: Static member | [WaitForFullGCComplete(Int32)](http://msdn.microsoft.com/en-us/library/cc647221.aspx) | Returns, in a specified time-out period, the status of a registered notification for determining whether a full garbage collection by common language the runtime has completed. |
| Description: Public methodDescription: Static memberDescription: Supported by the XNA Framework | [WaitForPendingFinalizers](http://msdn.microsoft.com/en-us/library/system.gc.waitforpendingfinalizers.aspx) | Suspends the current thread until the thread that is processing the queue of finalizers has emptied that queue. |

[Top](http://msdn.microsoft.com/en-us/library/system.gc.aspx#mainBody)

Description: http://i.msdn.microsoft.com/Global/Images/clear.gifRemarks

The methods in this class influence when garbage collection is performed on an object and when resources allocated by an object are released. Properties in this class provide information about the total amount of memory available in the system and the age category, or generation, of memory allocated to an object.

The garbage collector tracks and reclaims objects allocated in managed memory. Periodically, the garbage collector performs garbage collection to reclaim memory allocated to objects for which there are no valid references. Garbage collection happens automatically when a request for memory cannot be satisfied using available free memory. Alternatively, an application can force garbage collection using the [Collect](http://msdn.microsoft.com/en-us/library/system.gc.collect.aspx) method.

Garbage collection consists of the following steps:

1. The garbage collector searches for managed objects that are referenced in managed code.
2. The garbage collector tries to finalize objects that are not referenced.
3. The garbage collector frees objects that are not referenced and reclaims their memory.

During a collection, the garbage collector will not free an object if it finds one or more references to the object in managed code. However, the garbage collector does not recognize references to an object from unmanaged code, and might free objects that are being used exclusively in unmanaged code unless explicitly prevented from doing so. The [KeepAlive](http://msdn.microsoft.com/en-us/library/system.gc.keepalive.aspx) method provides a mechanism that prevents the garbage collector from collecting objects that are still in use in unmanaged code.

Aside from managed memory allocations, implementations of the garbage collector do not maintain information about resources held by an object, such as file handles or database connections. When a type uses unmanaged resources that must be released before instances of the type are reclaimed, the type can implement a finalizer.

In most cases, finalizers are implemented by overriding the [Object..::.Finalize](http://msdn.microsoft.com/en-us/library/system.object.finalize.aspx) method; however, types written in C# or C++ implement destructors, which compilers turn into an override of [Object..::.Finalize](http://msdn.microsoft.com/en-us/library/system.object.finalize.aspx). In most cases, if an object has a finalizer, the garbage collector calls it prior to freeing the object. However, the garbage collector is not required to call finalizers in all situations; for example, the [SuppressFinalize](http://msdn.microsoft.com/en-us/library/system.gc.suppressfinalize.aspx) method explicitly prevents a finalizer from being called. Also, the garbage collector is not required to use a specific thread to finalize objects, or guarantee the order in which finalizers are called for objects that reference each other but are otherwise available for garbage collection.

In scenarios where resources must be released at a specific time, classes can implement the [IDisposable](http://msdn.microsoft.com/en-us/library/system.idisposable.aspx) interface, which contains the [IDisposable..::.Dispose](http://msdn.microsoft.com/en-us/library/system.idisposable.dispose.aspx) method that performs resource management and cleanup tasks. Classes that implement [Dispose](http://msdn.microsoft.com/en-us/library/system.idisposable.dispose.aspx) must specify, as part of their class contract, if and when class consumers call the method to clean up the object. The garbage collector does not, by default, call the [Dispose](http://msdn.microsoft.com/en-us/library/system.idisposable.dispose.aspx) method; however, implementations of the [Dispose](http://msdn.microsoft.com/en-us/library/system.idisposable.dispose.aspx) method can call methods in the **GC** class to customize the finalization behavior of the garbage collector.

It is recommended, but not required, that garbage collectors support object aging using generations. A generation is a unit of measure of the relative age of objects in memory. The generation number, or age, of an object indicates the generation to which an object belongs. Objects created more recently are part of newer generations, and have lower generation numbers than objects created earlier in the application life cycle. Objects in the most recent generation are in generation zero.

**Notes to Implementers**

This implementation of the garbage collector supports three generations of objects.

[MaxGeneration](http://msdn.microsoft.com/en-us/library/system.gc.maxgeneration.aspx) is used to determine the maximum generation number supported by the system. Object aging allows applications to target garbage collection at a specific set of generations rather than requiring the garbage collector to evaluate all generations.

Description: http://i.msdn.microsoft.com/Global/Images/clear.gifExamples

The following example uses several GC methods to get generation and memory information about a block of unused objects and print it to the console. The unused objects are then collected, and the resulting memory totals are displayed.

Visual Basic

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl120_ctl00_ctl00_code');" \o "Copy Code)

Imports System

Namespace GCCollectInt\_Example

Class MyGCCollectClass

Private maxGarbage As Long = 10000

Public Shared Sub Main()

Dim myGCCol As New MyGCCollectClass

'Determine the maximum number of generations the system

'garbage collector currently supports.

Console.WriteLine("The highest generation is {0}", GC.MaxGeneration)

myGCCol.MakeSomeGarbage()

'Determine which generation myGCCol object is stored in.

Console.WriteLine("Generation: {0}", GC.GetGeneration(myGCCol))

'Determine the best available approximation of the number

'of bytes currently allocated in managed memory.

Console.WriteLine("Total Memory: {0}", GC.GetTotalMemory(False))

'Perform a collection of generation 0 only.

GC.Collect(0)

'Determine which generation myGCCol object is stored in.

Console.WriteLine("Generation: {0}", GC.GetGeneration(myGCCol))

Console.WriteLine("Total Memory: {0}", GC.GetTotalMemory(False))

'Perform a collection of all generations up to and including 2.

GC.Collect(2)

'Determine which generation myGCCol object is stored in.

Console.WriteLine("Generation: {0}", GC.GetGeneration(myGCCol))

Console.WriteLine("Total Memory: {0}", GC.GetTotalMemory(False))

Console.Read()

End Sub

Sub MakeSomeGarbage()

Dim vt As Version

Dim i As Integer

For i = 0 To maxGarbage - 1

'Create objects and release them to fill up memory

'with unused objects.

vt = New Version

Next i

End Sub

End Class

End Namespace

C#

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl120_ctl00_ctl01_code');" \o "Copy Code)

using System;

namespace GCCollectIntExample

{

class MyGCCollectClass

{

private const long maxGarbage = 1000;

static void Main()

{

MyGCCollectClass myGCCol = new MyGCCollectClass();

// Determine the maximum number of generations the system

// garbage collector currently supports.

Console.WriteLine("The highest generation is {0}", GC.MaxGeneration);

myGCCol.MakeSomeGarbage();

// Determine which generation myGCCol object is stored in.

Console.WriteLine("Generation: {0}", GC.GetGeneration(myGCCol));

// Determine the best available approximation of the number

// of bytes currently allocated in managed memory.

Console.WriteLine("Total Memory: {0}", GC.GetTotalMemory(false));

// Perform a collection of generation 0 only.

GC.Collect(0);

// Determine which generation myGCCol object is stored in.

Console.WriteLine("Generation: {0}", GC.GetGeneration(myGCCol));

Console.WriteLine("Total Memory: {0}", GC.GetTotalMemory(false));

// Perform a collection of all generations up to and including 2.

GC.Collect(2);

// Determine which generation myGCCol object is stored in.

Console.WriteLine("Generation: {0}", GC.GetGeneration(myGCCol));

Console.WriteLine("Total Memory: {0}", GC.GetTotalMemory(false));

Console.Read();

}

void MakeSomeGarbage()

{

Version vt;

for(int i = 0; i < maxGarbage; i++)

{

// Create objects and release them to fill up memory

// with unused objects.

vt = new Version();

}

}

}

}

Visual C++

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl120_ctl00_ctl02_code');" \o "Copy Code)

using namespace System;

const long maxGarbage = 1000;

ref class MyGCCollectClass

{

public:

void MakeSomeGarbage()

{

Version^ vt;

for ( int i = 0; i < maxGarbage; i++ )

{

// Create objects and release them to fill up memory

// with unused objects.

vt = gcnew Version;

}

}

};

int main()

{

MyGCCollectClass^ myGCCol = gcnew MyGCCollectClass;

// Determine the maximum number of generations the system

// garbage collector currently supports.

Console::WriteLine( "The highest generation is {0}", GC::MaxGeneration );

myGCCol->MakeSomeGarbage();

// Determine which generation myGCCol object is stored in.

Console::WriteLine( "Generation: {0}", GC::GetGeneration( myGCCol ) );

// Determine the best available approximation of the number

// of bytes currently allocated in managed memory.

Console::WriteLine( "Total Memory: {0}", GC::GetTotalMemory( false ) );

// Perform a collection of generation 0 only.

GC::Collect( 0 );

// Determine which generation myGCCol object is stored in.

Console::WriteLine( "Generation: {0}", GC::GetGeneration( myGCCol ) );

Console::WriteLine( "Total Memory: {0}", GC::GetTotalMemory( false ) );

// Perform a collection of all generations up to and including 2.

GC::Collect( 2 );

// Determine which generation myGCCol object is stored in.

Console::WriteLine( "Generation: {0}", GC::GetGeneration( myGCCol ) );

Console::WriteLine( "Total Memory: {0}", GC::GetTotalMemory( false ) );

}