using System;

using System.Collections;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

namespace ConsoleApp5

{

#region GC

//class Program

//{

// static void Main(string[] args)

// {

// Timer timer = new Timer(TimerCall, null, 0, 2000);

// Console.ReadLine();

// }

// private static void TimerCall(object state)

// {

// Console.WriteLine("Timer call : " + DateTime.Now);

// GC.Collect();

// }

//}

#endregion

//class File:IDisposable

//{

// //~File() // Finalyzer

// //{

// //}

// public void Dispose()

// {

// }

//}

public static class Program

{

public static void Main()

{

//ValueTypePerfTest();

ReferenceTypePerfTest();

}

private static void ValueTypePerfTest()

{

const int count = 100000000;

using (new OperationTimer("List<int>"))

{

List<int> l = new List<int>();

for (int n = 0; n < count; n++)

{

l.Add(n); // No boxing

int x = l[n]; // No unboxing

}

l = null; // Make sure this gets garbage collected

}

using (new OperationTimer("ArrayList of int"))

{

ArrayList a = new ArrayList();

for (int n = 0; n < count; n++)

{

a.Add(n); // Boxing

int x = (int)a[n]; // Unboxing

}

a = null; // Make sure this gets garbage collected

}

}

private static void ReferenceTypePerfTest()

{

const Int32 count = 100000000;

using (new OperationTimer("List<String>"))

{

List<String> l = new List<String>();

for (Int32 n = 0; n < count; n++)

{

l.Add("X"); // Reference copy

String x = l[n]; // Reference copy

}

l = null; // Make sure this gets garbage collected

}

using (new OperationTimer("ArrayList of String"))

{

ArrayList a = new ArrayList();

for (Int32 n = 0; n < count; n++)

{

a.Add("X"); // Reference copy

String x = (String)a[n]; // Cast check & reference copy

}

a = null; // Make sure this gets garbage collected

}

}

}

// This class is useful for doing operation performance timing

internal sealed class OperationTimer : IDisposable

{

private Stopwatch m\_stopwatch;

private String m\_text;

private Int32 m\_collectionCount;

public OperationTimer(String text)

{

PrepareForOperation();

m\_text = text;

m\_collectionCount = GC.CollectionCount(0);

// This should be the last statement in this

// method to keep timing as accurate as possible

m\_stopwatch = Stopwatch.StartNew();

}

public void Dispose()

{

Console.WriteLine("{0} (GCs={1,3}) {2}", (m\_stopwatch.Elapsed),

GC.CollectionCount(0) - m\_collectionCount, m\_text);

}

private static void PrepareForOperation()

{

GC.Collect();

GC.WaitForPendingFinalizers();

GC.Collect();

}

}

}