Esatto

Memory Leaks in .NET

Common scenarios & analysis

Why should I care?

- Unnecessary memory usage, bigger, more CPU costly collections
- GC pressure
- OutOfMemory exception
- Cost

GC sweep is marked with an arrow

Finding the leaks

Proactively:

- Soak testing
- Load testing
- Memory targeted tests

When a problem occurs:

- Logs
- Memory dump

```
var memoryCheckPoint1 = dotMemory.Check();
foo.Bar();
var memoryCheckPoint2 = dotMemory.Check(memory =>
{   Assert.That(memory.GetTrafficFrom(memoryCheckPoint1)
            .Where(obj => obj.Interface.Is<IFoo>())
            .AllocatedMemory.SizeInBytes, Is.LessThan(1000));});
bar.Foo();
dotMemory.Check(memory =>
{   Assert.That(memory.GetTrafficFrom(memoryCheckPoint2)
        .Where( obj => obj.Type.Is<Bar>())
        .AllocatedMemory.ObjectsCount, Is.LessThan(10));});
```

```
var memoryCheckPoint = dotMemory.Check();

foo.Bar();

dotMemory.Check(memory =>
{
    Assert.That(memory.GetDifference(memoryCheckPoint)
                          .GetSurvivedObjects()
                          .GetObjects(where => where
                          .Namespace.Like("MyApp"))
                          .ObjectsCount, Is.EqualTo(0));
});
```

```
[DotMemoryUnit(SavingStrategy = SavingStrategy.OnAnyFail,
                    Directory = @"C:\tmp\Class",
                    WorkspaceNumberLimit = 200,
                    DiskSpaceLimit = 104857600)]
```

Esatto

Let’s get down to some practice

Esatto

Key tips

- Don't guess where the leak is
- Create more dumps
- Check for GC costs, but don't forget about allocation costs
- Monitor your app
- Use object pooling