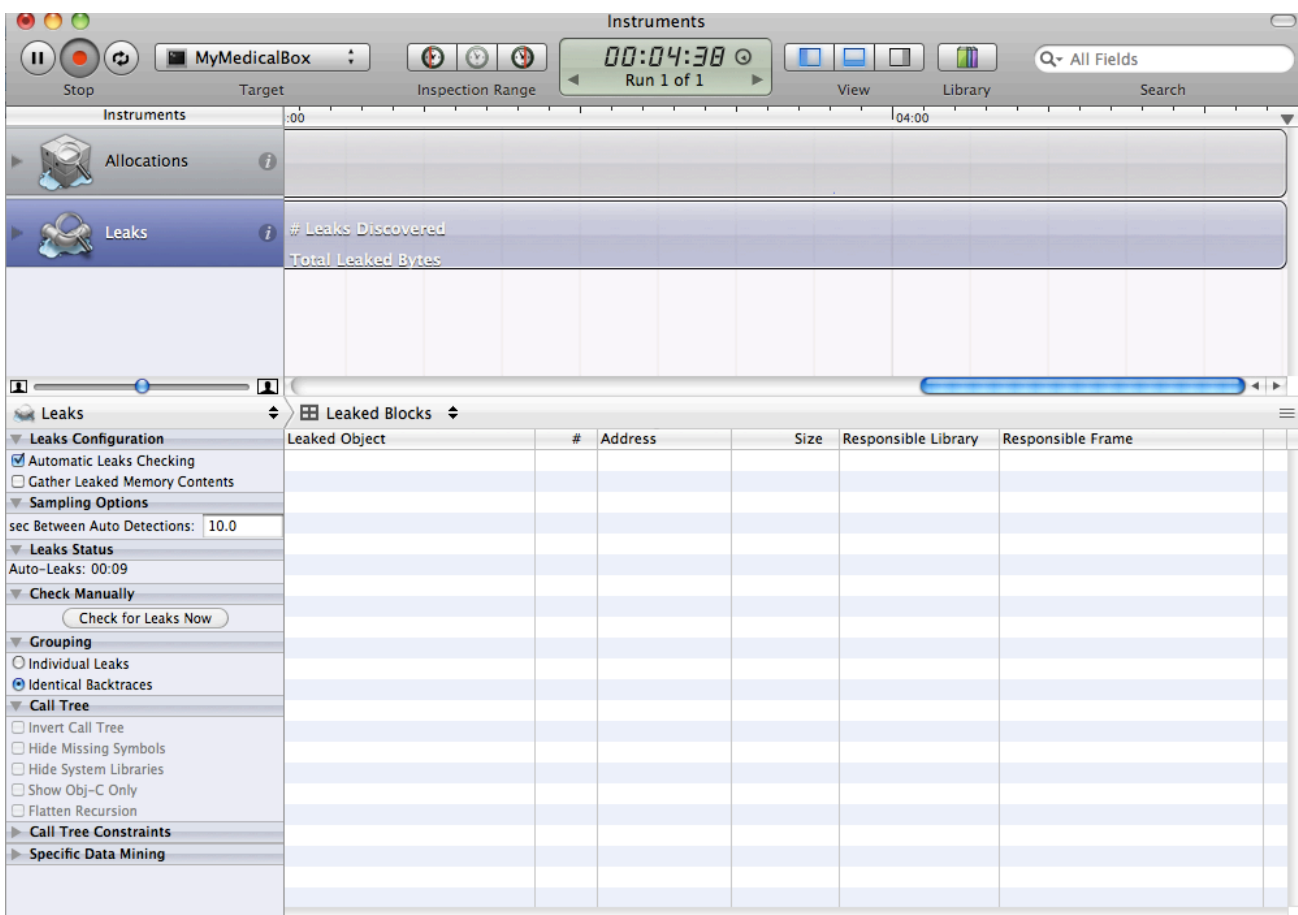




Checking for Memory Leaks

Suppose we develop an application without memory allocation considerations, for example allocate memory for new objects repetitively and without releasing it when they are no longer in use. Although xcode does come with a garbage collector, the problem is we never know when they will release unused memory and make that memory space available. So the habit is always release objects when they are no longer in use.

1. Using MyMedicalBox2-4.zip as the reference. Lets check for memory leaks. Open the project in xcode. Select Run -> Start With Performance Tool -> Leaks. Click the Leaks tab.

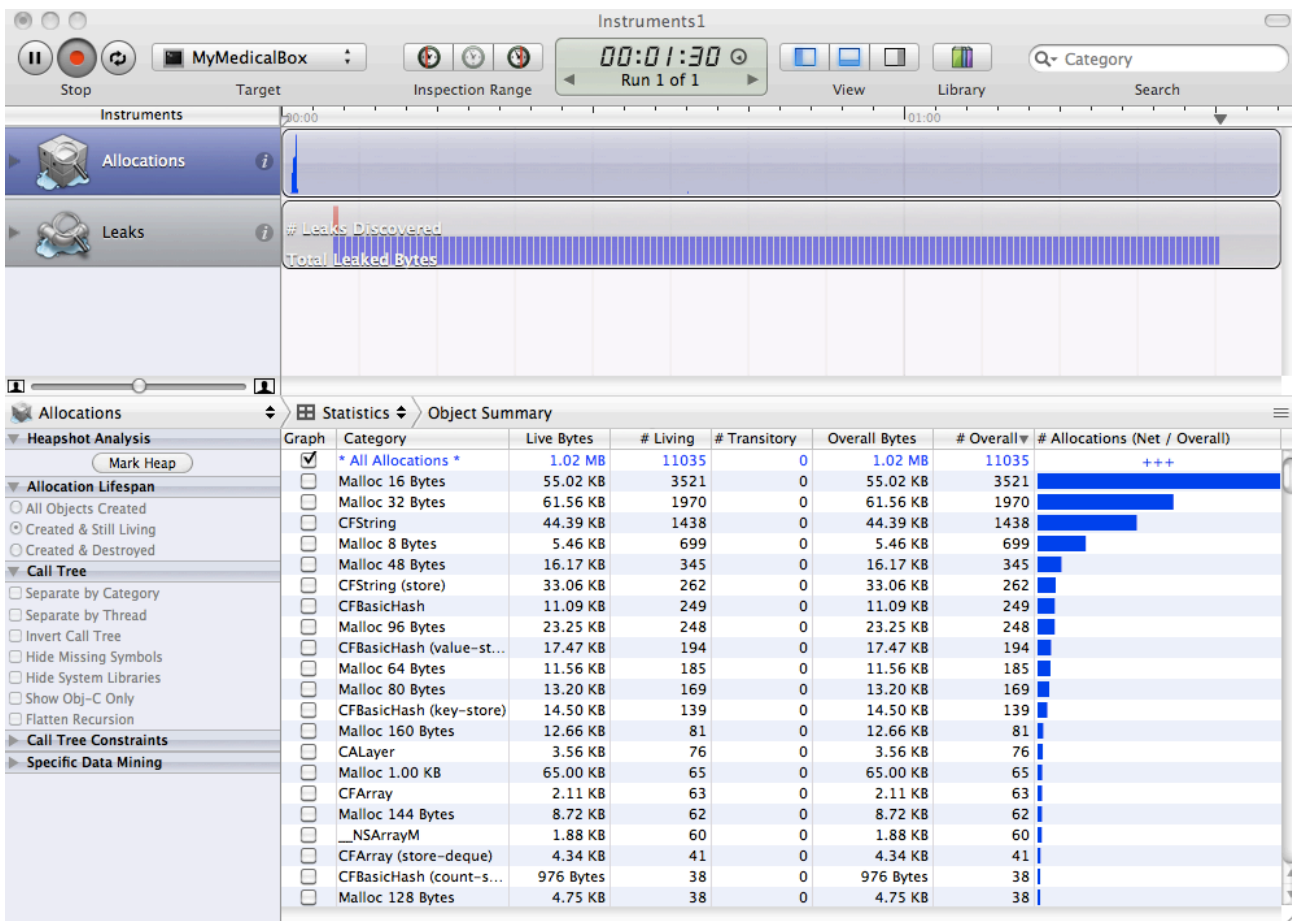


2. As we can see there are no leaked objects. Lets add a piece of code to make it a leaky app. Close the instrument. In the appDelegate.m add this code at the top in the didFinishLaunching method.

```
NSMutableString *test = [[NSMutableString alloc] init];  
[test appendString:@"Testing 1"];  
[test appendString:@"\nTesting 2"];
```



3. Build the project. And Run Start with performance tool again. This time we should see a leak.



4. Select the NSCFString leak object and Click View -> Extended Detail.



Leaked Object	#	Address	Size	Responsible Library
NSString		0x4b36bb0	32 Bytes	Foundation
Malloc 32 Bytes		0x4b36890	32 Bytes	Foundation

5. Double click the MyMedicalBoxAppDelegate.m to go to the line that contains the leak object.

```

@synthesize productsNavController;
@synthesize storedproductsNavController;
@synthesize databaseName;
@synthesize databasePath;
@synthesize products;

#pragma mark -
#pragma mark Application lifecycle

- (BOOL)application:(UIApplication *)application
    didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    NSMutableString *test = [[NSMutableString alloc] _init];
    [test appendString:@"Testing 1"];
    [test appendString:@"\nTesting 2"];

    // Setup some globals
    self.databaseName = @"myMedicalDatabase.sqlite";

    // Get the path to the documents directory and append the
    databaseName
    NSArray *documentPaths = NSSearchPathForDirectoriesInDomains
        (NSDocumentDirectory, NSUserDomainMask, YES);
    NSString *documentsDir = [documentPaths objectAtIndex:0];
    self.databasePath = [documentsDir stringByAppendingPathComponent:

```

