Daily Coding Problem #120

Problem

This problem was asked by Microsoft.

Implement the singleton pattern with a twist. First, instead of storing one instance, store two instances. And in every even call of `getInstance()`, return the first instance and in every odd call of `getInstance()`, return the second instance.

Solution

This question is more about programming and design patterns than computer science.

The singleton pattern allows you to limit the number of objects of a class to one instance. This is helpful in a large application either to conserve resources such as memory or to make correctness easier to reason about. For example, to represent configuration of a system, it would be helpful to have one centralized object.

In this particular question, we ask for a twist on the classic singleton by allowing two instances of a class. We do this by adding another static field as well as `calls` variable to keep track of the number of calls made to `getInstance`.

```
public class Service {
   private static Service instanceOne = null;
   private static Service instanceTwo = null;

   private static int calls = 0;

   private Service() {
      // Disallow creation through the constructor
   }

   public static Service getInstance() {
      if(instanceOne == null) {
         instanceOne = new Service();
         instanceTwo = new Service();
      }

      if (calls++ % 2 == 0) {
        return instanceOne;
      }
      return instanceTwo;
   }
}
```