|  |  |
| --- | --- |
| Objective | Teach basics of RED GREEN BLUE, Test Driven and Behavior Driven Development for an entry level audience |
| Key Points | * TDD is emergent * Red is a form of testing your tests * Refactoring is inherent to the process * Common IDE features and Test Coverage mitigate most overhead that TDD is perceived to introduce. |
| Time | 45 minutes (unverified) |
| Materials | Projector or Large Format Display, laptop, an instance of Visual Studio Community with NUnit Test Runner Adapter installed. |
| Procedure | Optional: Ask for a volunteer from the class to run the keyboard   1. Introduce the problem of Fizz Buzz. The program should print out numbers 1 – 100. Multiples of 3 should display as “Fizz,” multiples of 5 should display as “Buzz”, and multiples of 3 and 5 should display as “FizzBuzz” 2. Demonstrate the code in the console application, and describe briefly the architecture of the program (FizzBuzz class is our business logic, utilized by the console UI). 3. Do the return 1 and 2 test case. Refactor to return int as a string. 4. Do the 3 test case. 5. Refactor tests to use Nunit’s TestCase to demonstrate importance of test maintenance. 6. Do the 5 test case. 7. Refactor code to simplest form, emphasizing small descriptive methods. Demonstrate how code coverage will catch us if we make a mistake. 8. When complete, run the console program to demonstrate success. 9. Discuss edge cases (e.g., negative numbers, 0) and gold plating. 10. Optional – Multiples of 10 should display “TDD Rocks” 11. Optional – Add exceptions for edge cases that may need it. It works fine as is, we only need to discuss edge cases when they arise. 12. Optional – Can we ship this (yes, there aren’t any bugs. Make new tests for things that could be bugs). |