Unit Testing and TDD FAQs

Capgemini

Unit Testing and TDD

Q1.	**What is Unit Testing?**
Ans:	A unit test is a piece of code written by a developer that exercises a very small, specific area of functionality of the code being tested. Usually, a unit test exercises some method in a particular context.
Q2.	**Why Unit Testing?**
Ans:	It will make your designs better and drastically reduce the amount of time you spend debugging To avoid having to compile and run the entire application just to check a single function
Q3.	**What are advantages of Unit Testing?**
Ans:	Unit testing helps eliminate uncertainty in the pieces themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier. Unit testing provides a sort of "living document". Clients and other developers looking to learn how to use the class can look at the unit tests to determine how to use the class to fit their needs and gain a basic understanding of the API.
Q4.	**Who can perform unit testing?**
Ans:	Unit Testing is generally done at the development phase so that it can be performed by developers.
Q5.	**What is use of Test and TestFixture attribute?**
Ans:	The TestFixture attribute designates that a class is a test fixture. Classes thus designated contain setup, teardown, and unit tests. The Test attribute indicates that a method in the test fixture is a unit test.
Q6.	**Can we have more than one SetUp method and TearDown method in testFixture?**
Ans:	No. A test fixture can only have one SetUp method and TearDown method.
Q7.	**What is Code Coverage?**
Ans:	Code coverage is to determine to what portion of your project code is being tested by Unit testing; you can use the code coverage feature of Visual Studio.
Q8.	**What are 3'A of unit testing?**

Ans:	Arrange - Do necessary setup of the test
	Act - Execute the test
	Assert- Check and verify the returned result with expected results.
	For Example:
	`//Arrange test`
	`testClass objtest = new testClass();`
	`Boolean result;`
	`//Act test`
	`result = objtest.testFunction();`
	`//Assert test`

	Assert.AreEqual(true, result);
Q9.	**Which Tool is used for .net Unit testing?**
Ans:	NUnit Tool

Unit Testing and TDD- Lesson 2

Q10.	**What is Test Driven Development (TDD)?**
Ans:	TDD is a technique whereby you write your test cases before you write any implementation code. Tests drive or dictate the code that is developed. An indication of "intent" Tests provide a specification of "what" a piece of code actually does.
Q11.	**What are advantages of TDD?**
Ans:	Better program design and higher code quality. When writing tests for requirements, programmers immediately create a strict and detailed specification. TDD reduces the time required for project development. Code flexibility and easier maintenance. Refactoring becomes easier. Save project costs in the long run.
Q12.	**What do you know about the term Refactoring?**
Ans:	If you want to revamp any existing code, then this technique is used. It is generally done in small steps where only the code is changed, not the functionality or the logic. It helps in bug fixing too.
Q13.	**What is Mocking?**
Ans:	Mocking is primarily used in unit testing. An object under test may have dependencies on other (complex) objects. To isolate the behavior of the object you want to replace the other objects by mocks that simulate the behavior of the real objects. This is useful if the real objects are impractical to incorporate into the unit test.
Q14.	**What is stub?**
Ans:	A stub is simply an alternate implementation Stub object is used for State verification. Stub object vary their response based on input parameters
Q15.	**What are the types of Mock Objects supported by Rhino.Mocks framework?**
Ans:	Strict Mock Dynamic Mock Partial Mock