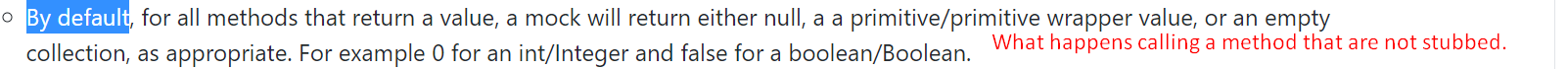
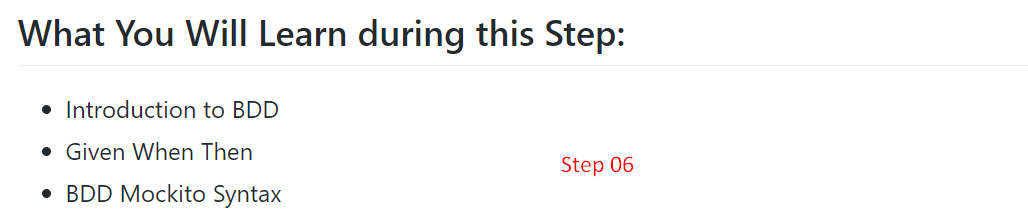
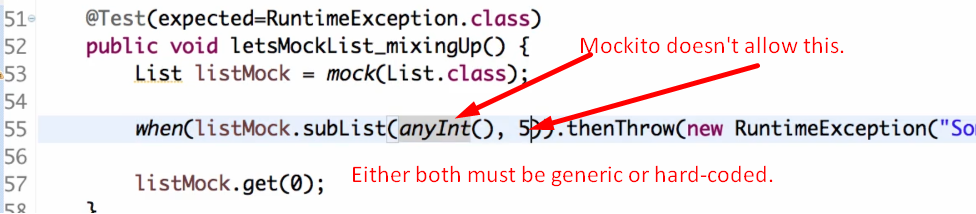
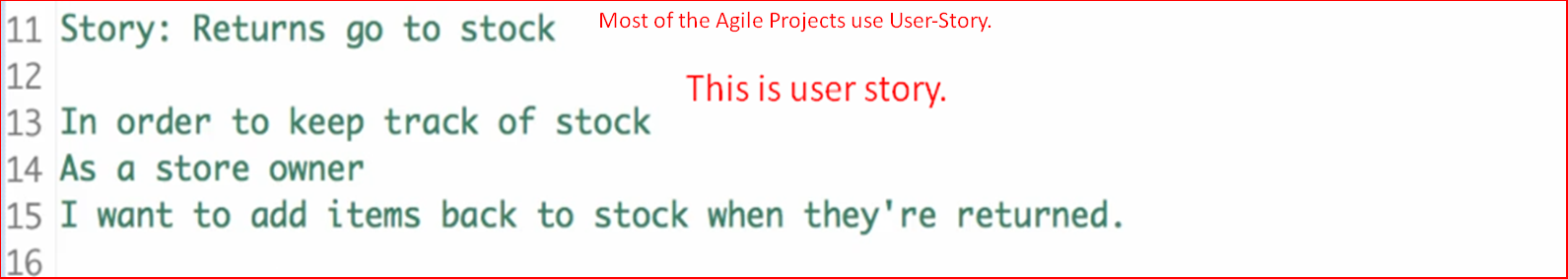
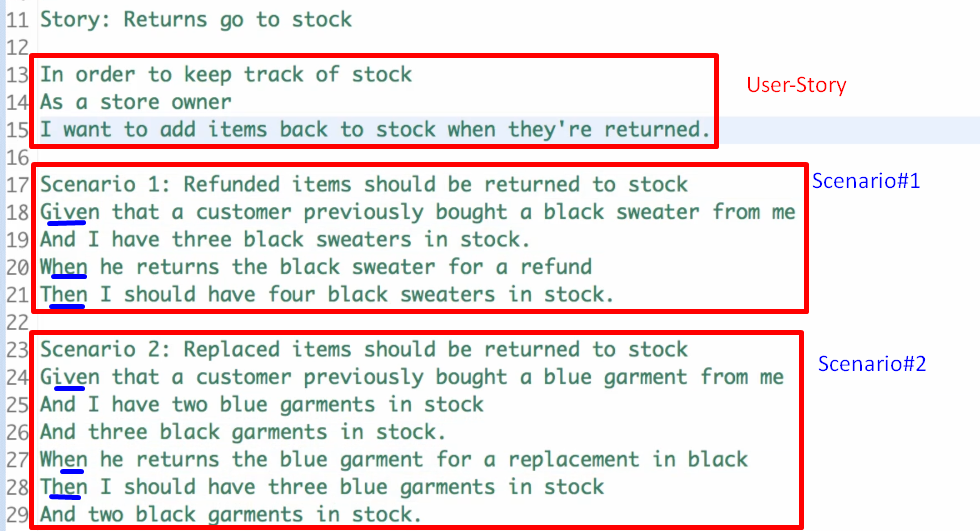
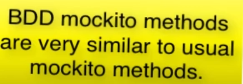
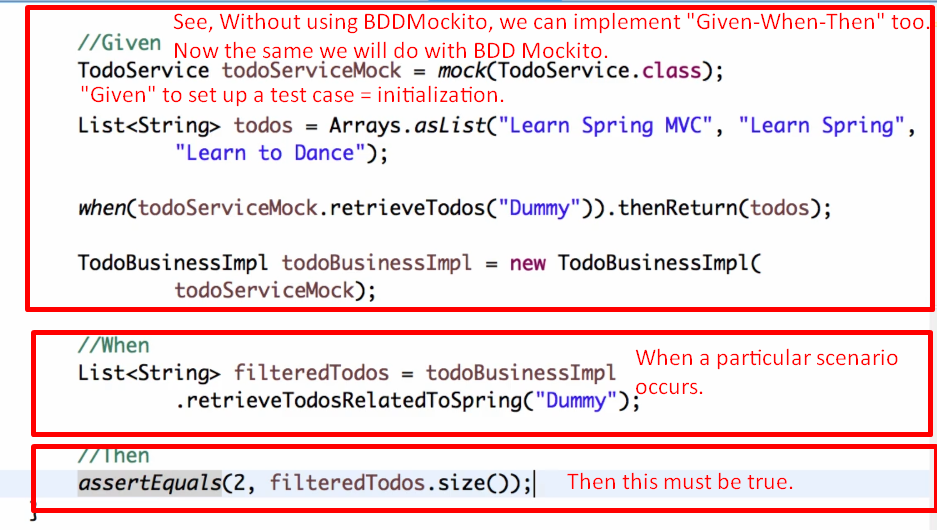
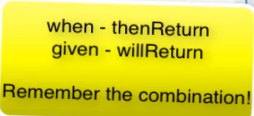
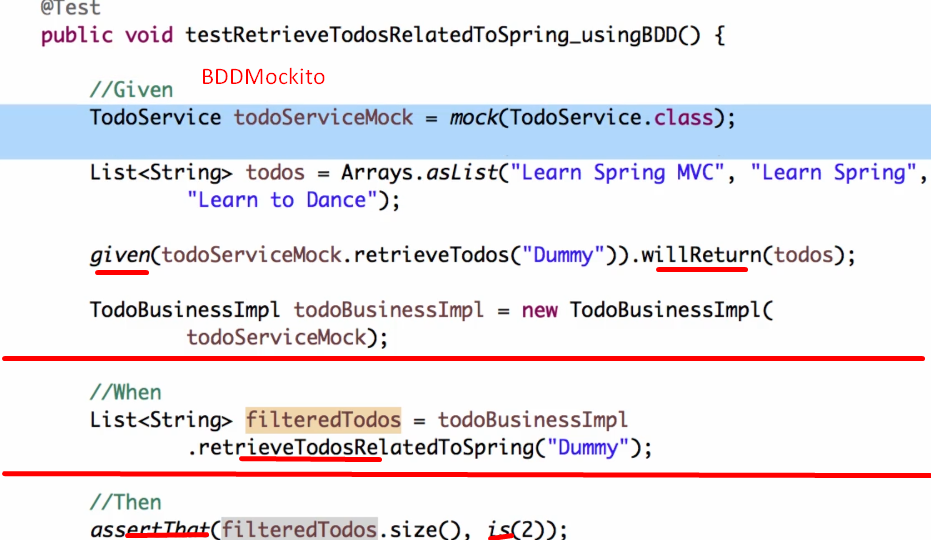
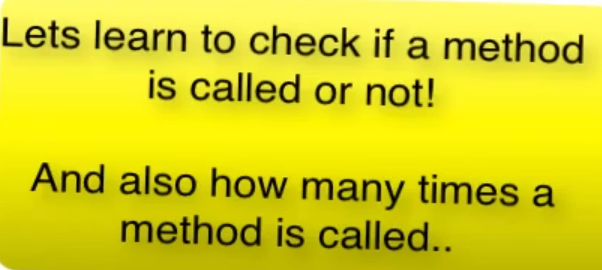
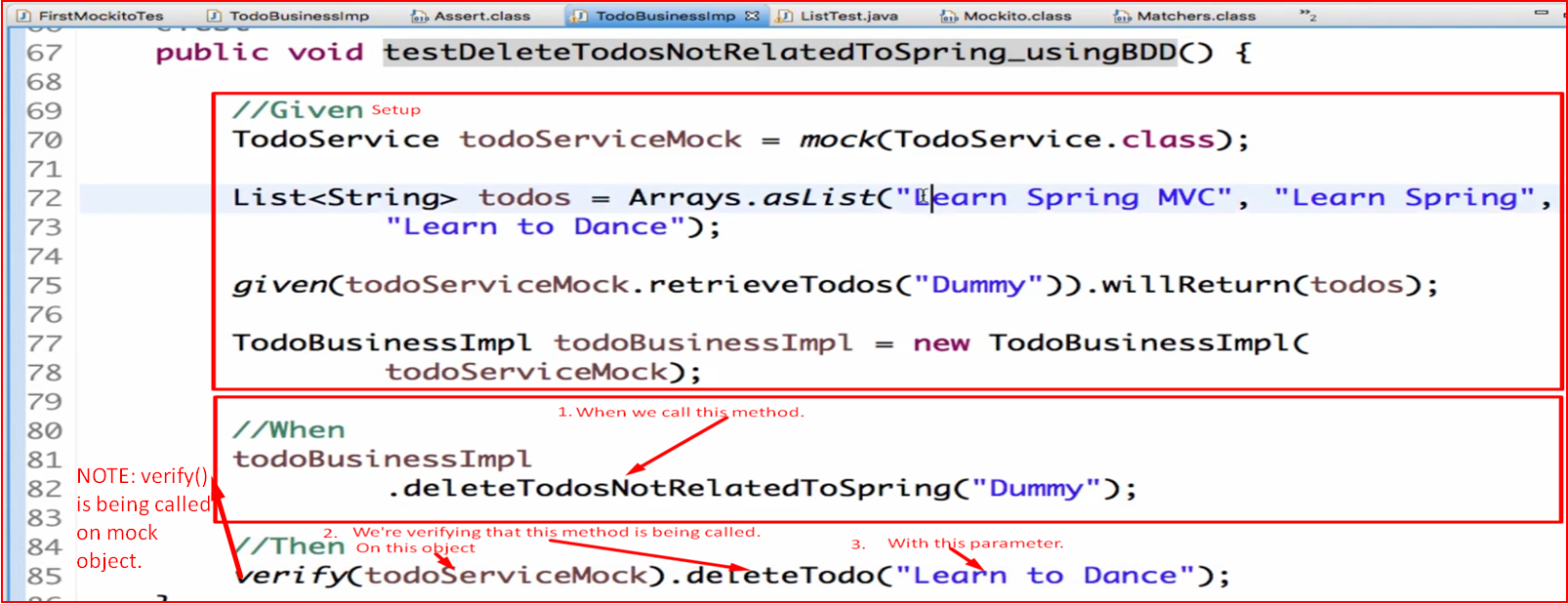
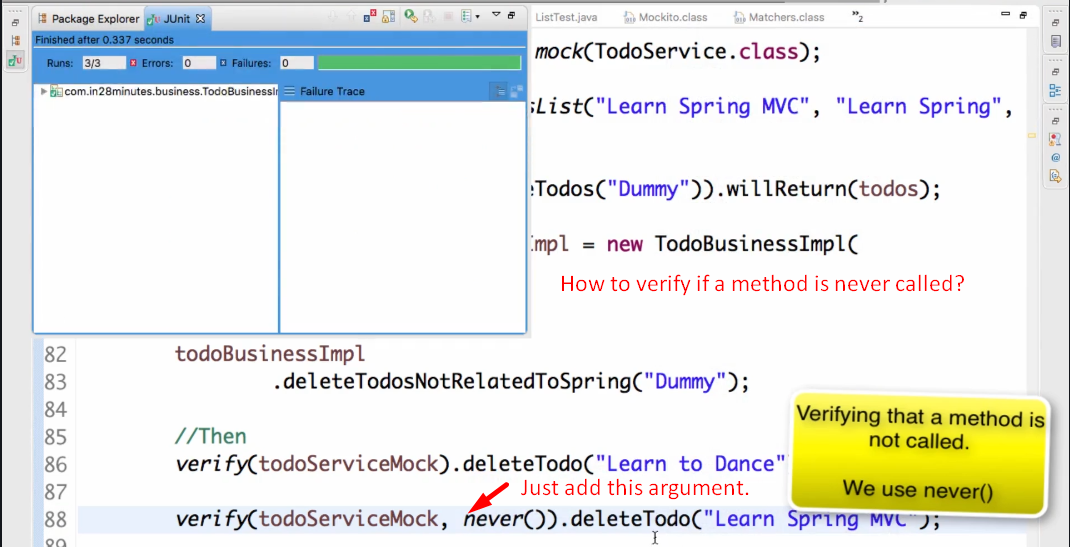
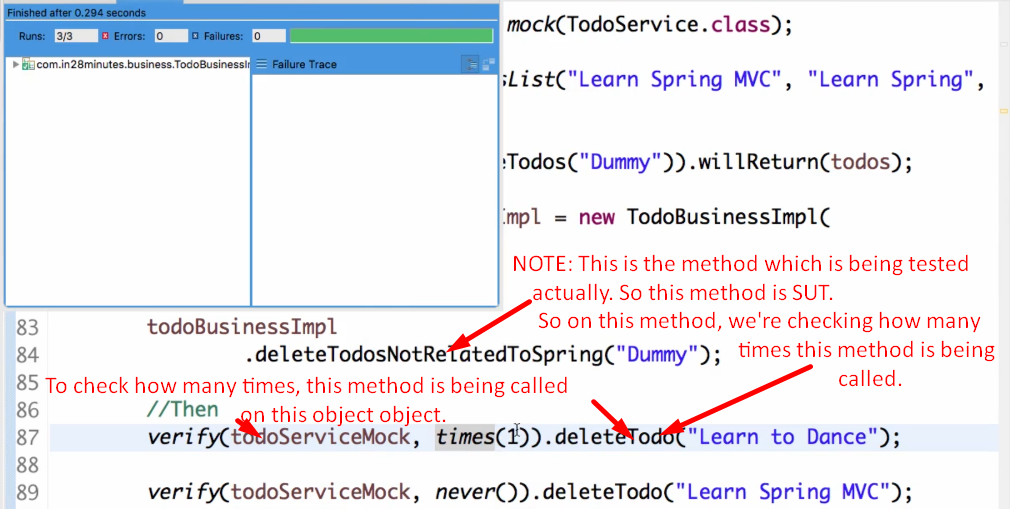
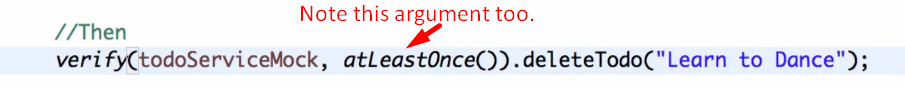
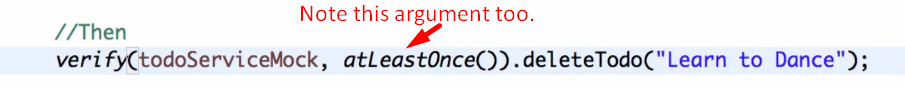
1. **Mockito**:
   1. **when**(mock(List.class).size()).thenReturn(2).thenReturn(3);
      1. Mocking List Interface.
      2. 1st call to size() would return 2 and   
         2nd call to size() would return 3
   2. **when**(mock(List.class).get(0)).thenReturn(“Jatin”);
   3. assertEquals(“Jatin”, mockList().get(1));
      1. We didn’t mock for index 1, so default   
         NOTE: Whereas EasyMock API doesn’t do this.
   4. **when**(mock(List.class).get(anyInt())).thenReturn(“Jatin”);
   5. 
   6. when(mockList.get(anyInt())).thenThrow(new RuntimeException("Something is Wrong"));
2. 

BDD

1. 
2. 
3. 
4. **NOTE**: We write a scenario in “**Given-When-Then**” format.  
   Actually, we split a user story into multiple scenarios (“**Given-When-Then**”)
5. 
6. “**Given-When-Then**”:
   1. **Given**:
      1. It is setup for the scenario.
      2. **Example**: I’ve 3 black sweaters in stock.
   2. **When**:
      1. It is the actual method call means the System Under Test (SUT). The method which we want to test.
      2. **Example**: When a customer returns a black sweater for a refund.
   3. **Then**:
      1. It is a set of asserts that something has happened.
      2. **Example**: Now the stock must have 4 black sweaters.
7. BDDMockito **🡺 Mockito Class specific to write Test Cases in BDD  
   **
8. Writing BDD by using simple plain Mockito without using BDDMockito  
   
9. BDDMockito Replacements: 
10. 
11. 
12. 
13. 
14. Checking how many times a method is being called on mock object.  
    
15. 
16. 
17. 