# JUNIT TEST FOR BOOK Test .JAVA

**Version History**

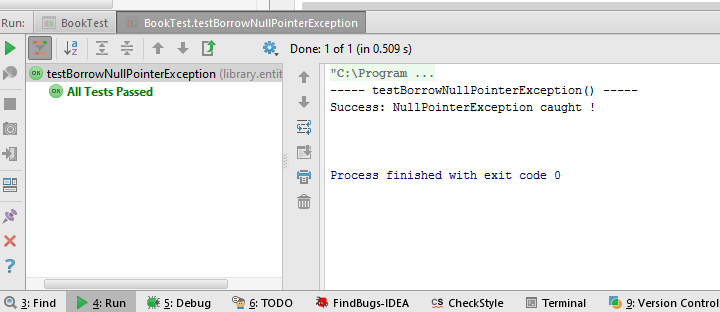
|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Description** | **Author** |
| 1.0 | 1/10/2016 | Initial version | KAMAL RAJ KANDEL |

# Methods:



@Test  
*/\*\*  
 \* Test if NullPointerException catches while Book.borrow()'s parameter(ILoan) is NULL.  
 \* if (loan == null) { throw new NullPointerException(); }  
 \*/***public void** testBorrowNullPointerException() **throws** Exception {  
 System.***out***.println(**"----- testBorrowNullPointerException() -----"**);  
 **loan** = **null**;  
  
 **try** {  
 **this**.**book**.borrow(**loan**);  
 } **catch**(NullPointerException e) {  
 *assertThat*(e, *instanceOf*(NullPointerException.**class**));  
 System.***out***.println(**"Success: NullPointerException caught !"**);  
 }  
 System.***out***.println(**"\n"**);  
}

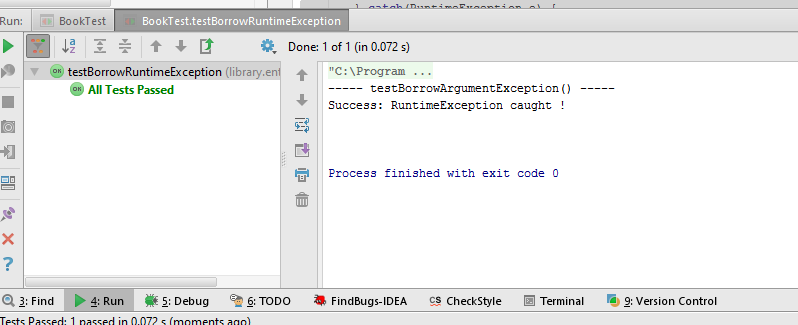
Result on the screen



2.

@Test  
*/\*\*  
 \* Test if it catches RunTimeException when state != EBookState.Available.  
 \* if(state != EBookState.Available) { throw new RuntimeException(); }  
 \*/***public void** testBorrowRuntimeException() **throws** Exception {  
 System.***out***.println(**"----- testBorrowArgumentException() -----"**);  
 **state** = EBookState.***ON\_LOAN***;  
  
 **try** {  
 **book**.borrow(**loan**);  
 } **catch**(RuntimeException e) {  
 *assertThat*(e, *instanceOf*(RuntimeException.**class**));  
 System.***out***.println(**"Success: RuntimeException caught !"**);  
 }  
 System.***out***.println(**"\n"**);  
}

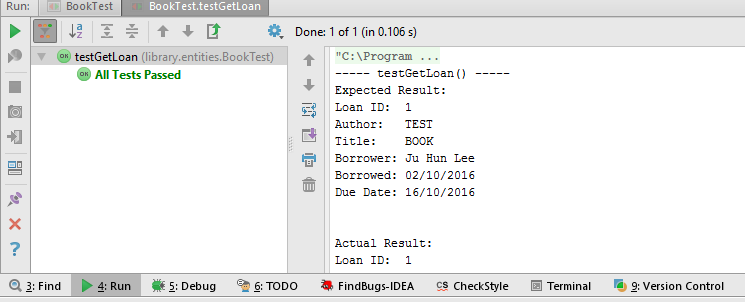
Result on the screen



3.

*/\*\*  
 \*  
 \** ***@throws*** *Exception  
 \*/*@Test  
**public void** testGetLoan() **throws** Exception {  
 **try** {  
 System.***out***.println(**"----- testGetLoan() -----"**);  
 String expectedResult = **book**.getLoan().toString();  
 String result = **loan**.toString();  
 *assertEquals*(expectedResult,result);  
 System.***out***.println(**"Expected Result: \n"** + expectedResult);  
 System.***out***.println(**"\n"**);  
 System.***out***.println(**"Actual Result: \n"** + result + **"\n"**);  
 System.***out***.println(**"Success: Identical !"**);  
 } **catch**(NullPointerException e) {  
 *fail*(**"Fail: Loan is null"**);  
 }  
 System.***out***.println(**"\n"**);  
}

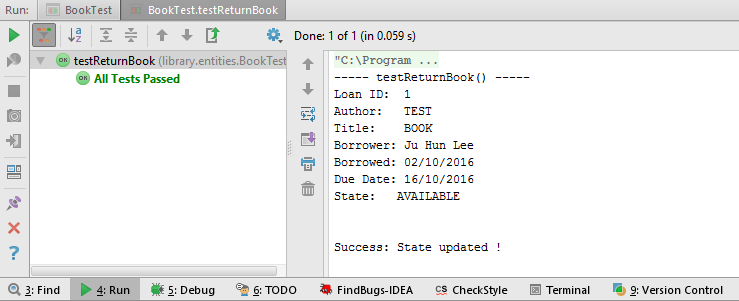
Result on the screen



4.

@Test  
*/\*\*  
 \* Test if Book's state updated when book is not damaged  
 \* this.state = damaged ? EBookState.DAMAGED : EBookState.AVAILABLE;  
 \*/***public void** testReturnBook() **throws** Exception {  
 **boolean** state = **false**;  
 EBookState expectedResult = EBookState.***AVAILABLE***;  
  
 **try** {  
 System.***out***.println(**"----- testReturnBook() -----"**);  
 **book**.returnBook(state);  
 System.***out***.println(**loan**);  
 System.***out***.println(**"State: \t "** + **book**.getState());  
 *assertEquals*(expectedResult,**book**.getState());  
 System.***out***.println(**"\n"**);  
 System.***out***.println(**"Success: State updated !"**);  
 } **catch**(NullPointerException e) {  
 *fail*(**"Fail: Null value submitted"**);  
 }  
 System.***out***.println(**"\n"**);  
}

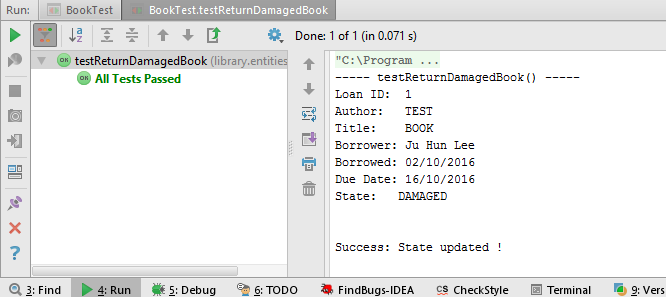
Result on the screen



5.

@Test  
*/\*\*  
 \* Test if Book's state updated when book is damaged  
 \* this.state = damaged ? EBookState.DAMAGED : EBookState.AVAILABLE;  
 \*/***public void** testReturnDamagedBook() **throws** Exception {  
 **boolean** state = **true**;  
 EBookState expectedResult = EBookState.***DAMAGED***;  
  
 **try** {  
 System.***out***.println(**"----- testReturnDamagedBook() -----"**);  
 **book**.returnBook(state);  
 System.***out***.println(**loan**);  
 System.***out***.println(**"State: \t "** + **book**.getState());  
 *assertEquals*(expectedResult,**book**.getState());  
 System.***out***.println(**"\n"**);  
 System.***out***.println(**"Success: State updated !"**);  
 } **catch**(NullPointerException e) {  
 *fail*(**"Fail: Null value submitted"**);  
 }  
 System.***out***.println(**"\n"**);  
}

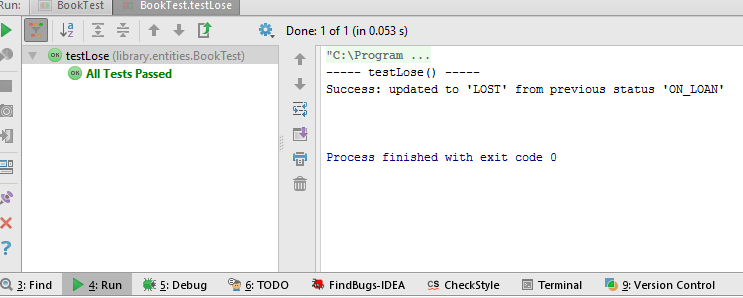
Result on the Screen



6.

@Test  
*/\*\*  
 \* Test if it changes state from book to 'LOST' from 'ON\_LOAN'  
 \* if (this.state != EBookState.ON\_LOAN) { throw new RuntimeException() }  
 \*/***public void** testLose() **throws** Exception {  
 System.***out***.println(**"----- testLose() -----"**);  
 **book**.setState(EBookState.***ON\_LOAN***);  
 EBookState prevState = **book**.getState();  
 EBookState expectedResult = EBookState.***LOST***;  
 **try** {  
 **book**.lose();  
 *assertEquals*(expectedResult, **book**.getState());  
 System.***out***.println(**"Success: updated to '"** + expectedResult + **"' from previous status '"** + prevState + **"'"**);  
 } **catch**(RuntimeException e) {  
 *fail*(**"Fail: state should be ON\_LOAN"**);  
 }  
 System.***out***.println(**"\n"**);  
}

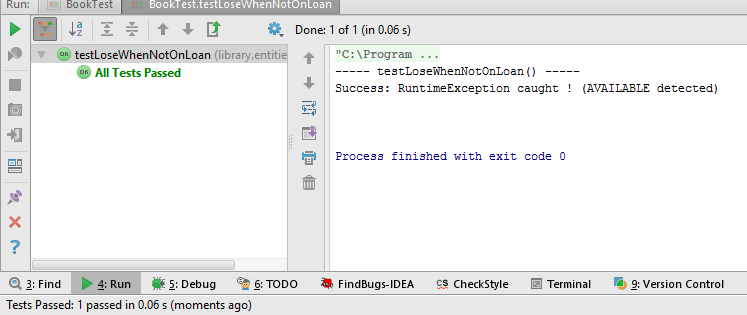
Result on the Screen



7.

@Test  
*/\*\*  
 \* Test if it catches RuntimeException when state is not 'ON\_LOAN'  
 \* if (this.state != EBookState.ON\_LOAN) { throw new RuntimeException() }  
 \*/***public void** testLoseWhenNotOnLoan() **throws** Exception {  
 System.***out***.println(**"----- testLoseWhenNotOnLoan() -----"**);  
 **book**.setState(EBookState.***AVAILABLE***);  
 **try** {  
 **book**.lose();  
 } **catch**(RuntimeException e) {  
 *assertThat*(e,*instanceOf*(RuntimeException.**class**));  
 System.***out***.println(**"Success: RuntimeException caught ! ("** + **book**.getState() + **" detected)"**);  
 }  
 System.***out***.println(**"\n"**);  
}

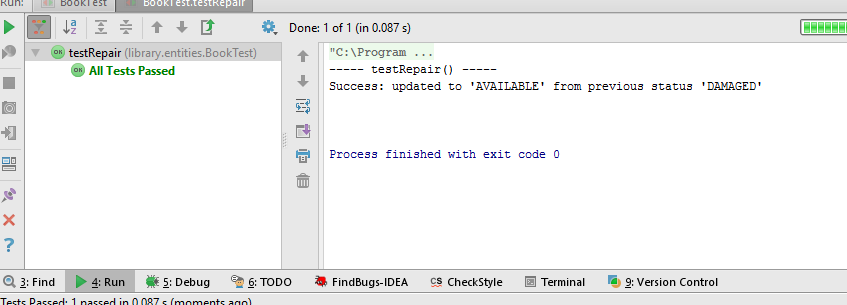
Result on the screen



8.

@Test  
*/\*\*  
 \* Test if state changes into 'AVAILABLE' after book.repair() executed from given state 'DAMAGED'  
 \* if (this.state != EBookState.DAMAGED) { throw new RuntimeException() }  
 \*/***public void** testRepair() **throws** Exception {  
 System.***out***.println(**"----- testRepair() -----"**);  
 EBookState prevState = EBookState.***DAMAGED***;  
 EBookState expectedResult = EBookState.***AVAILABLE***;  
 **book**.setState(prevState);  
 **try** {  
 **book**.repair();  
 *assertEquals*(expectedResult, **book**.getState());  
 System.***out***.println(**"Success: updated to '"** + expectedResult + **"' from previous status '"** + prevState + **"'"**);  
 } **catch**(RuntimeException e) {  
 *fail*(**"Fail: State should be 'DAMAGED', but passed state is '"** + **book**.getState() + **"'"**);  
 }  
 System.***out***.println(**"\n"**);  
}

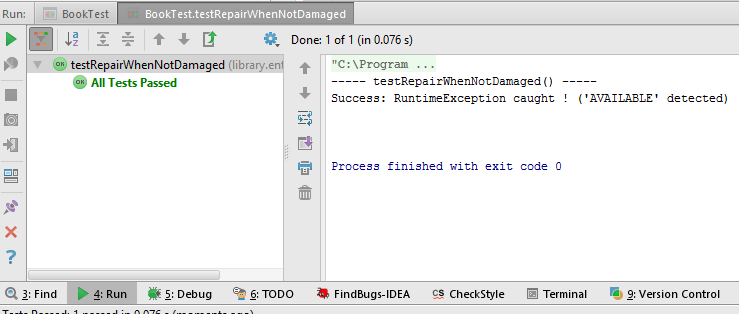
Result on the Screen



9.

@Test  
*/\*\*  
 \* Test if the method catch RuntimeException when current state is not 'DAMAGED'  
 \* if (this.state != EBookState.DAMAGED) { throw new RuntimeException() }  
 \*/***public void** testRepairWhenNotDamaged() **throws** Exception {  
 System.***out***.println(**"----- testRepairWhenNotDamaged() -----"**);  
 EBookState prevState = EBookState.***AVAILABLE***;  
 EBookState expectedResult = EBookState.***AVAILABLE***;  
 **book**.setState(prevState);  
 **try** {  
 **book**.repair();  
 } **catch**(RuntimeException e) {  
 *assertThat*(e, *instanceOf*(RuntimeException.**class**));  
 System.***out***.println(**"Success: RuntimeException caught ! ('"** + prevState + **"' detected)"**);  
 }  
 System.***out***.println(**"\n"**);  
}

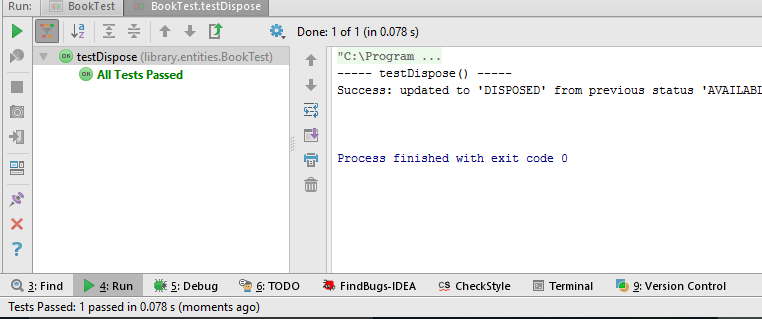
Result on the Screen



10.

@Test  
*/\*\*  
 \* Test if current state changes into 'DISPOSED' from 'AVAILABLE'  
 \* if (this.state != EBookState.AVAILABLE && this.state != EBookState.DAMAGED && this.state != EBookState.LOST) {throw new RuntimeException() }  
 \*/***public void** testDispose() **throws** Exception {  
 System.***out***.println(**"----- testDispose() -----"**);  
 EBookState prevState = EBookState.***AVAILABLE***;  
 EBookState expectedResult = EBookState.***DISPOSED***;  
 **book**.setState(prevState);  
 **try** {  
 **book**.dispose();  
 *assertEquals*(expectedResult, **book**.getState());  
 System.***out***.println(**"Success: updated to '"** + expectedResult + **"' from previous status '"** + prevState + **"'"**);  
 } **catch**(RuntimeException e) {  
 *fail*(**"Fail: Book's state should be 'AVAILABLE', 'DAMAGED', 'LOST'. ('"** + prevState + **"' detected)"**);  
 }  
 System.***out***.println(**"\n"**);  
}

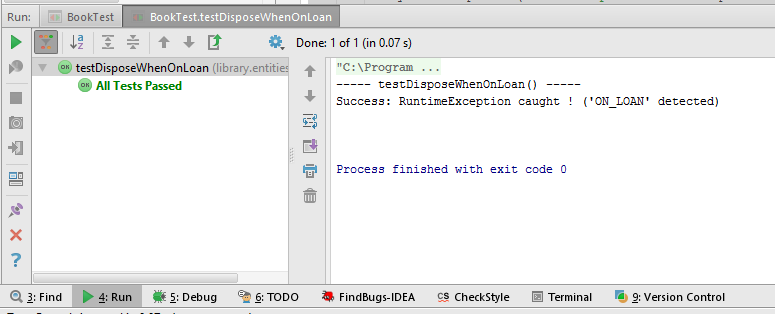
Result on the Screen



11.

@Test  
*/\*\*  
 \* Test if the method catches  
 \* if (this.state != EBookState.AVAILABLE && this.state != EBookState.DAMAGED && this.state != EBookState.LOST) {throw new RuntimeException() }  
 \*/***public void** testDisposeWhenOnLoan() **throws** Exception {  
 System.***out***.println(**"----- testDisposeWhenOnLoan() -----"**);  
 EBookState prevState = EBookState.***ON\_LOAN***;  
 EBookState expectedResult = EBookState.***DISPOSED***;  
 **book**.setState(prevState);  
 **try** {  
 **book**.dispose();  
 *assertEquals*(expectedResult, **book**.getState());  
 System.***out***.println(**"Success: updated to '"** + expectedResult + **"' from previous status '"** + prevState + **"'"**);  
 } **catch**(RuntimeException e) {  
 *assertThat*(e, *instanceOf*(RuntimeException.**class**));  
 System.***out***.println(**"Success: RuntimeException caught ! ('"** + prevState + **"' detected)"**);  
 }  
 System.***out***.println(**"\n"**);  
}

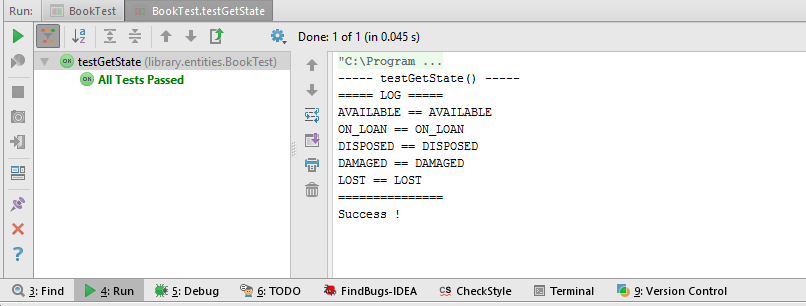
Result on the Screen



12.

@Test  
**public void** testGetState() **throws** Exception {  
 System.***out***.println(**"----- testGetState() -----"**);  
 EBookState expected1 = EBookState.***AVAILABLE***;  
 EBookState expected2 = EBookState.***ON\_LOAN***;  
 EBookState expected3 = EBookState.***DISPOSED***;  
 EBookState expected4 = EBookState.***DAMAGED***;  
 EBookState expected5 = EBookState.***LOST***;  
 **try** {  
 System.***out***.println(**"===== LOG ====="**);  
 **book**.setState(expected1);  
 *assertSame*(expected1, **book**.getState());  
 System.***out***.println(**"AVAILABLE == "** + **book**.getState());  
 **book**.setState(expected2);  
 *assertSame*(expected2,**book**.getState());  
 System.***out***.println(**"ON\_LOAN == "** + **book**.getState());  
 **book**.setState(expected3);  
 *assertSame*(expected3,**book**.getState());  
 System.***out***.println(**"DISPOSED == "** + **book**.getState());  
 **book**.setState(expected4);  
 *assertSame*(expected4,**book**.getState());  
 System.***out***.println(**"DAMAGED == "** + **book**.getState());  
 **book**.setState(expected5);  
 *assertSame*(expected5,**book**.getState());  
 System.***out***.println(**"LOST == "** + **book**.getState());  
 System.***out***.println(**"==============="**);  
 System.***out***.println(**"Success !"**);  
 } **catch**(Exception e) {  
 *fail*(**"Fail: Value does not match !"**);  
 }  
 System.***out***.println(**"\n"**);  
}

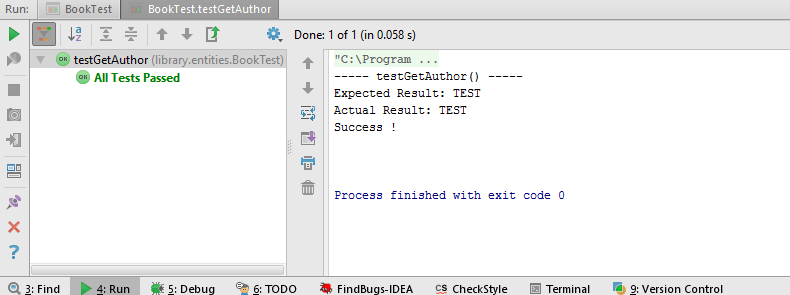
Result on the Screen



13.

@Test  
**public void** testGetAuthor() **throws** Exception {  
 System.***out***.println(**"----- testGetAuthor() -----"**);  
 String expectedResult = **"TEST"**;  
 **try** {  
 *assertEquals*(expectedResult, **book**.getAuthor());  
 System.***out***.println(**"Expected Result: "** + expectedResult);  
 System.***out***.println(**"Actual Result: "** + **book**.getAuthor());  
 System.***out***.println(**"Success !"**);  
 } **catch**(Exception e) {  
 *fail*(**"Fail: Value does not match !"**);  
 }  
 System.***out***.println(**"\n"**);  
}

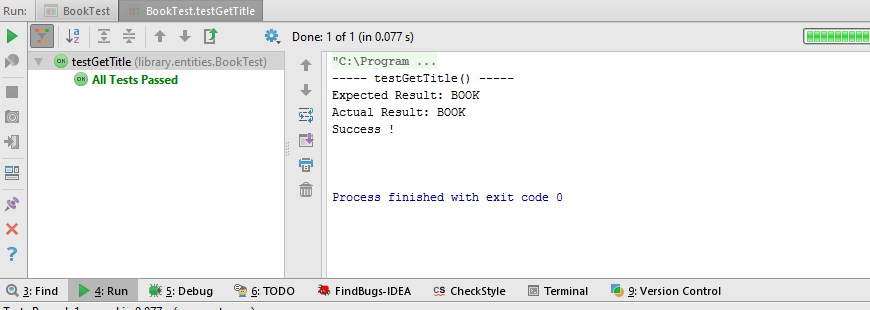
Result on the Screen



14.

@Test  
**public void** testGetTitle() **throws** Exception {  
 System.***out***.println(**"----- testGetTitle() -----"**);  
 String expectedResult = **"BOOK"**;  
 **try** {  
 *assertEquals*(expectedResult, **book**.getTitle());  
 System.***out***.println(**"Expected Result: "** + expectedResult);  
 System.***out***.println(**"Actual Result: "** + **book**.getTitle());  
 System.***out***.println(**"Success !"**);  
 } **catch**(Exception e) {  
 *fail*(**"Fail: Value does not match !"**);  
 }  
 System.***out***.println(**"\n"**);  
}

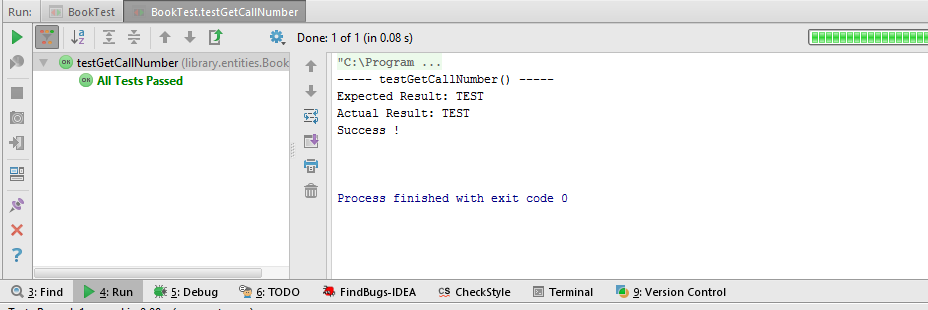
Result on the Screen



15.

@Test  
**public void** testGetCallNumber() **throws** Exception {  
 System.***out***.println(**"----- testGetCallNumber() -----"**);  
 String expectedResult = **"TEST"**;  
 **try** {  
 *assertEquals*(expectedResult, **book**.getCallNumber());  
 System.***out***.println(**"Expected Result: "** + expectedResult);  
 System.***out***.println(**"Actual Result: "** + **book**.getCallNumber());  
 System.***out***.println(**"Success !"**);  
 } **catch**(Exception e) {  
 *fail*(**"Fail: Value does not match !"**);  
 }  
 System.***out***.println(**"\n"**);  
}

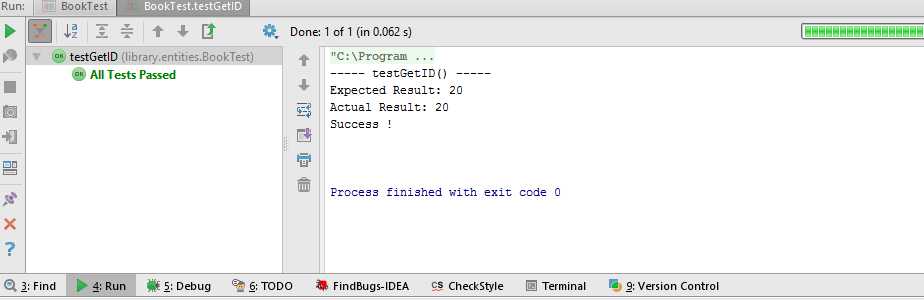
Result on the Screen



16.

@Test  
**public void** testGetID() **throws** Exception {  
 System.***out***.println(**"----- testGetID() -----"**);  
 **int** expectedResult = 20;  
 **try** {  
 *assertEquals*(expectedResult, **book**.getID());  
 System.***out***.println(**"Expected Result: "** + expectedResult);  
 System.***out***.println(**"Actual Result: "** + **book**.getID());  
 System.***out***.println(**"Success !"**);  
 } **catch**(Exception e) {  
 *fail*(**"Fail: Value does not match !"**);  
 }  
 System.***out***.println(**"\n"**);  
}

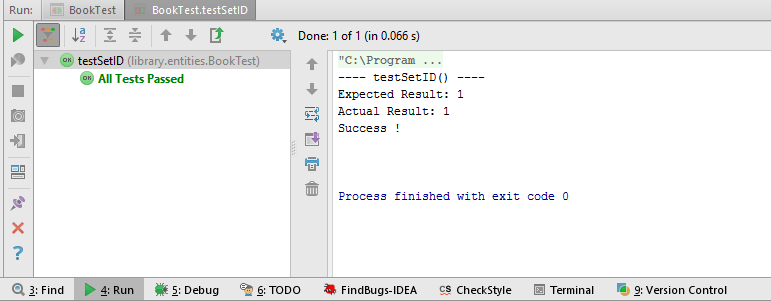
Result on the Screen



17.

@Test  
**public void** testSetID() **throws** Exception {  
 System.***out***.println(**"---- testSetID() ----"**);  
 **int** expectedID = 1;  
 **book**.setID(1);  
 **try** {  
 *assertEquals*(expectedID, **book**.getID());  
 System.***out***.println(**"Expected Result: "** + expectedID);  
 System.***out***.println(**"Actual Result: "** + **book**.getID());  
 System.***out***.println(**"Success !"**);  
 } **catch**(RuntimeException e) {  
 *fail*(**"Fail: Cannot update."**);  
 }  
 System.***out***.println(**"\n"**);  
}

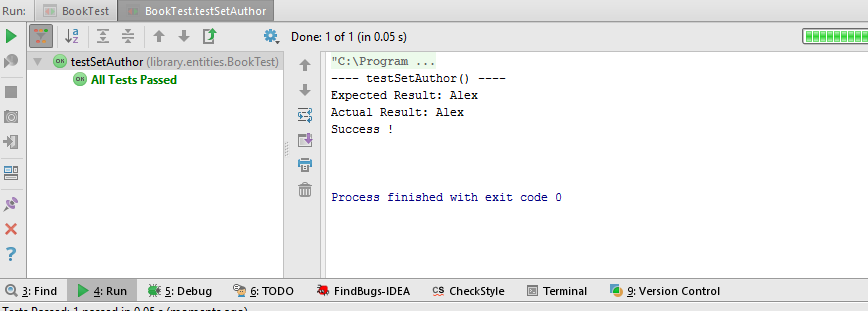
Result on the Screen



18.

@Test  
**public void** testSetAuthor() **throws** Exception {  
 System.***out***.println(**"---- testSetAuthor() ----"**);  
 String expectedResult = **"Alex"**;  
 **book**.setAuthor(**"Alex"**);  
 **try** {  
 *assertEquals*(expectedResult, **book**.getAuthor());  
 System.***out***.println(**"Expected Result: "** + expectedResult);  
 System.***out***.println(**"Actual Result: "** + **book**.getAuthor());  
 System.***out***.println(**"Success !"**);  
 } **catch**(RuntimeException e) {  
 *fail*(**"Fail: Cannot update."**);  
 }  
 System.***out***.println(**"\n"**);  
}

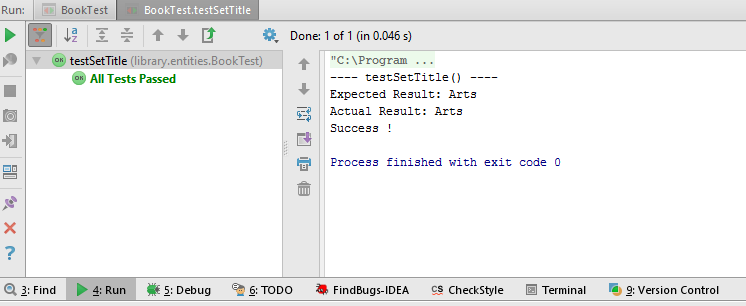
Result on the Screen



19.

@Test  
**public void** testSetTitle() **throws** Exception {  
 System.***out***.println(**"---- testSetTitle() ----"**);  
 String expectedResult = **"Arts"**;  
 **book**.setTitle(**"Arts"**);  
 **try** {  
 *assertEquals*(expectedResult, **book**.getTitle());  
 System.***out***.println(**"Expected Result: "** + expectedResult);  
 System.***out***.println(**"Actual Result: "** + **book**.getTitle());  
 System.***out***.println(**"Success !"**);  
 } **catch**(RuntimeException e) {  
 *fail*(**"Fail: Cannot update."**);  
 }  
}

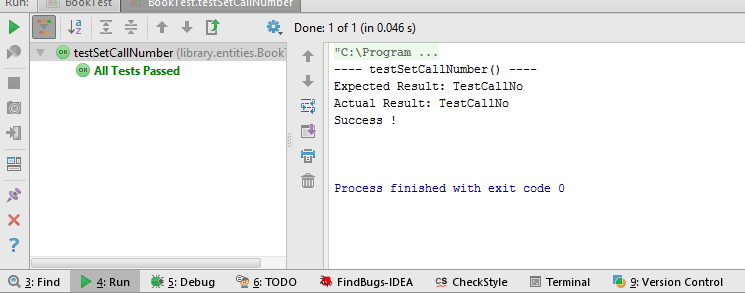
Result on the Screen



20.

@Test  
**public void** testSetCallNumber() **throws** Exception {  
 System.***out***.println(**"---- testSetCallNumber() ----"**);  
 String expectedResult = **"TestCallNo"**;  
 **book**.setCallNumber(**"TestCallNo"**);  
 **try** {  
 *assertEquals*(expectedResult, **book**.getCallNumber());  
 System.***out***.println(**"Expected Result: "** + expectedResult);  
 System.***out***.println(**"Actual Result: "** + **book**.getCallNumber());  
 System.***out***.println(**"Success !"**);  
 } **catch**(RuntimeException e) {  
 *fail*(**"Fail: Cannot update."**);  
 }  
 System.***out***.println(**"\n"**);  
}

Result on the Screen



21.

@Test  
**public void** testToString() **throws** Exception {  
 System.***out***.println(**"----- testToString() -----"**);  
 System.***out***.println(**book**.toString());  
 System.***out***.println(**"\n"**);  
}

