

bankingapp-with-junit - bankingapp-with-junit/src/test/AccountServiceHashMapImplTest.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

```
AccountServiceHashMapImplTest.java x
1 package test;
2
3 import org.junit.Before;
4 import org.junit.Test;
5
6 import com.indium.bankingapp.model.Account;
7 import com.indium.bankingapp.service.AccountService;
8 import com.indium.bankingapp.service.AccountServiceHashMapImpl;
9
10 import static org.junit.Assert.assertEquals;
11 import static org.junit.Assert.assertFalse;
12 import static org.junit.Assert.assertNull;
13 import static org.junit.Assert.assertTrue;
14
15 public class AccountServiceHashMapImplTest {
16     private AccountService<Account> accountService;
17
18     @Before
19     public void setUp() {
20         accountService = new AccountServiceHashMapImpl();
21     }
22
23     @Test
24     public void testCreateAccount() {
25         Account account = new Account(1, "sunny", 1000.0, 0, null);
26
27         // Create a new account
28         assertTrue(accountService.createAccount(account));
29
30         // Try to create an account with the same ID (should fail)
31         assertFalse(accountService.createAccount(account));
32     }
33
34     @Test
35     public void testUpdateAccount() {
36         Account account = new Account(1, "sunny", 1000.0, 0, null);
37     }
```

bankingapp-with-junit - bankingapp-with-junit/src/test/AccountServiceHashMapImplTest.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

```
AccountServiceHashMapImplTest.java x
34= @Test
35 public void testUpdateAccount() {
36     Account account = new Account(1, "sunny", 1000.0, 0, null);
37
38     // Try to update an account that doesn't exist (should fail)
39     assertFalse(accountService.updateAccount(1, account));
40
41     // Create a new account
42     accountService.createAccount(account);
43
44     // Update the account
45     Account updatedAccount = new Account(1, "Updated sunny", 2000.0, 0, null);
46     assertTrue(accountService.updateAccount(1, updatedAccount));
47
48     // Check if the account has been updated
49     assertEquals(updatedAccount, accountService.getAccount(1));
50 }
51
52= @Test
53 public void testDeleteAccount() {
54     Account account = new Account(1, "sunny", 1000.0, 0, null);
55
56     // Try to delete an account that doesn't exist (should fail)
57     assertFalse(accountService.deleteAccount(1));
58
59     // Create a new account
60     accountService.createAccount(account);
61
62     // Delete the account
63     assertTrue(accountService.deleteAccount(1));
64
65     // Try to get the deleted account (should return null)
66     assertNull(accountService.getAccount(1));
67 }
68
69= @Test
70 public void testGetAccount() {
```

bankingapp-with-junit - bankingapp-with-junit/src/test/AccountServiceHashMapImplTest.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

AccountServiceHashMapImplTest.java x

```
60     accountService.createAccount(account);
61
62     // Delete the account
63     assertTrue(accountService.deleteAccount(1));
64
65     // Try to get the deleted account (should return null)
66     assertNull(accountService.getAccount(1));
67 }
68
69 @Test
70 public void testGetAccount() {
71     Account account = new Account(1, "sunny", 1000.0, 0, null);
72
73     // Try to get an account that doesn't exist (should return null)
74     assertNull(accountService.getAccount(1));
75
76     // Create a new account
77     accountService.createAccount(account);
78
79     // Get the created account
80     assertEquals(account, accountService.getAccount(1));
81 }
82
83 @Test
84 public void testGetAllAccounts() {
85     Account account1 = new Account(1, "sunny", 1000.0, 0, null);
86     Account account2 = new Account(2, "kumar", 2000.0, 0, null);
87
88     // Create two accounts
89     accountService.createAccount(account1);
90     accountService.createAccount(account2);
91
92     // Get all accounts and check if both are present
93     assertTrue(accountService.getAllAccounts().contains(account1));
94     assertTrue(accountService.getAllAccounts().contains(account2));
95 }
96 }
```