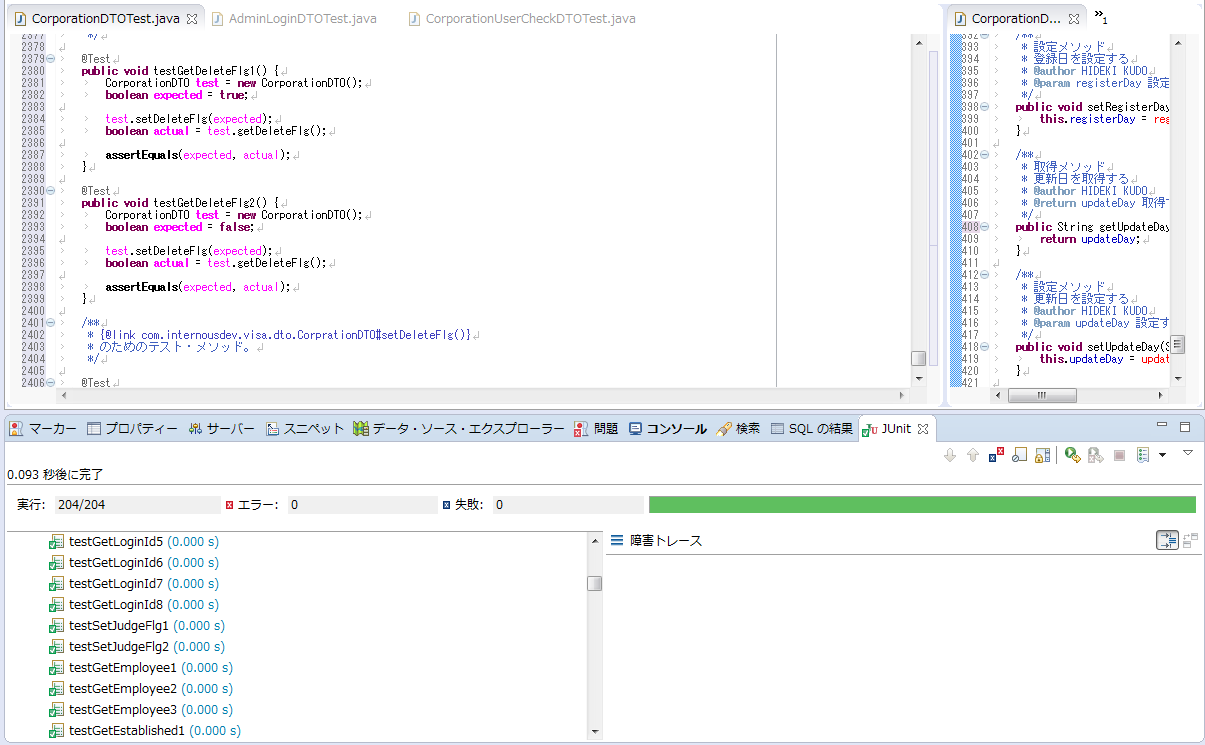
**単体試験参考資料**



・試験方法はJUnitを使用

・試験パターンは以下で行った。

Int … 　0、Integer.MAX\_VALUE、Integer.MIN\_VALUE、

　　　　　　　レンジ範囲外（Exception）

String … 　null、""（空白）、" "（半角スペース）、"　"（全角スペース）、

"abc123"、"あいう１２３"、

　　　　　　　"abc123あいう１２３"、"ａｂｃ１２３あいう漢字"

Boolean… 　true、false

---下記テストコード例---

【int】

/\*\*

\* {@link com.internousdev.visa.dto.UserDTO#getPostal()} のためのテスト・メソッド。

\*/

@Test

名前が被らないように末を１，２，３などと変える

**public** **void** testGetPostal1() {

UserDTO test = **new** UserDTO();

**int** expected = 0;

test.setPostal(expected);

各色別に対応箇所あり。

実装の際は、

使用するフォルダ名などに変更する。

*assertEquals*(expected, test.getPostal());

}

@Test

**public** **void** testGetPostal2() {

UserDTO test = **new** UserDTO();

**int** expected = -2147483648;

test.setPostal(expected);

*assertEquals*(expected, test.getPostal());

}

@Test

**public** **void** testGetPostal3() {

UserDTO test = **new** UserDTO();

**int** expected = 2147483647;

test.setPostal(expected);

ここからtry catch文を使って

レンジ範囲外をテスト

*assertEquals*(expected, test.getPostal());

}

@Test

**public** **void** testGetPostal4() **throws** Exception {

UserDTO test = **new** UserDTO();

**try** {

**int** postalMin = Integer.*parseInt*("-2147483649");

test.setPostal(postalMin);

} **catch** (RuntimeException e) {

assertThat1(e.getMessage(), "For input string: \"-2147483649\"");

}

}

@Test

**public** **void** testGetPostal5() **throws** Exception {

UserDTO test = **new** UserDTO();

**try** {

**int** postalMax = Integer.*parseInt*("2147483648");

test.setPostal(postalMax);

} **catch** (RuntimeException e) {

assertThat1(e.getMessage(), "For input string: \"2147483648\"");

}

}

**private** **void** assertThat1(String message, String string) {

}

/\*\*

\* {@link com.internousdev.visa.dto.UserDTO#setPostal()} のためのテスト・メソッド。

\*/

Get　set　両方作成する

**public** **void** testSetPostal3() {

UserDTO test = **new** UserDTO();

**int** expected = 0;

test.setPostal(expected);

**int** actual = test.getPostal();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetPostal1() {

UserDTO test = **new** UserDTO();

**int** expected = -2147483648;

test.setPostal(expected);

**int** actual = test.getPostal();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetPostal2() {

UserDTO test = **new** UserDTO();

**int** expected = 2147483647;

test.setPostal(expected);

ここからtry catch文を使って

レンジ範囲外をテスト

**int** actual = test.getPostal();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetPostal4() **throws** Exception {

UserDTO test = **new** UserDTO();

**try** {

**int** postalMin = Integer.*parseInt*("-2147483649");

test.setPostal(postalMin);

} **catch** (RuntimeException e) {

assertThat2(e.getMessage(), "For input string: \"-2147483649\"");

}

}

@Test

**public** **void** testSetPostal5() **throws** Exception {

UserDTO test = **new** UserDTO();

**try** {

**int** postalMax = Integer.*parseInt*("2147483648");

test.setPostal(postalMax);

} **catch** (RuntimeException e) {

assertThat2(e.getMessage(), "For input string: \"2147483648\"");

}

}

**private** **void** assertThat2(String message, String string) {

}

基本的なやり方はIntと同じ。  
例外処理はない。

【String 】

/\*\*

\* {@link com.internousdev.visa.dto.UserDTO#getLoginId()}のためのテスト・メソッド。

\*/

@Test

**public** **void** testGetLoginId1() {

UserDTO test = **new** UserDTO();

String expected = **null**;

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testGetLoginId2() {

UserDTO test = **new** UserDTO();

String expected = "";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testGetLoginId3() {

UserDTO test = **new** UserDTO();

String expected = " ";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testGetLoginId4() {

UserDTO test = **new** UserDTO();

String expected = "　";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testGetLoginId5() {

UserDTO test = **new** UserDTO();

String expected = "abc123";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testGetLoginId6() {

UserDTO test = **new** UserDTO();

String expected = "あいう１２３";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testGetLoginId7() {

UserDTO test = **new** UserDTO();

String expected = "abc123あいう１２３";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testGetLoginId8() {

UserDTO test = **new** UserDTO();

String expected = "ａｂｃ１２３あいう漢字";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

/\*\*

\* {@link com.internousdev.visa.dto.UserDTO#setLoginId()} のためのテスト・メソッド。

\*/

@Test

**public** **void** testSetLoginId1() {

UserDTO test = **new** UserDTO();

String expected = **null**;

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetLoginId2() {

UserDTO test = **new** UserDTO();

String expected = "";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetLoginId3() {

UserDTO test = **new** UserDTO();

String expected = " ";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetLoginId4() {

UserDTO test = **new** UserDTO();

String expected = "　";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetLoginId5() {

UserDTO test = **new** UserDTO();

String expected = "abc123";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetLoginId6() {

UserDTO test = **new** UserDTO();

String expected = "あいう１２３";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetLoginId7() {

UserDTO test = **new** UserDTO();

String expected = "abc123あいう１２３";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testSetLoginId8() {

UserDTO test = **new** UserDTO();

String expected = "ａｂｃ１２３あいう漢字";

test.setLoginId(expected);

String actual = test.getLoginId();

*assertEquals*(expected, actual);

}

【**boolean**】

/\*\*

\* {@link com.internousdev.visa.dto.UserDTO#isDeleteFlg()} のためのテスト・メソッド。

\*/

@Test

**public** **void** testisDeleteFlg1() {

UserDTO test = **new** UserDTO();

**boolean** expected = **true**;

test.setDeleteFlg(expected);

**boolean** actual = test.isDeleteFlg();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testisDeleteFlg2() {

UserDTO test = **new** UserDTO();

**boolean** expected = **false**;

test.setDeleteFlg(expected);

**boolean** actual = test.isDeleteFlg();

*assertEquals*(expected, actual);

}

/\*\*

\* {@link com.internousdev.visa.dto.UserDTO#isDeleteFlg()} のためのテスト・メソッド。

\*/

@Test

**public** **void** testsetDeleteFlg1() {

UserDTO test = **new** UserDTO();

**boolean** expected = **true**;

test.setDeleteFlg(expected);

**boolean** actual = test.isDeleteFlg();

*assertEquals*(expected, actual);

}

@Test

**public** **void** testsetDeleteFlg2() {

UserDTO test = **new** UserDTO();

**boolean** expected = **false**;

test.setDeleteFlg(expected);

**boolean** actual = test.isDeleteFlg();

*assertEquals*(expected, actual);

}

**報告書例**

単体試験報告書

　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　2016年2月22日

　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　--名前--

・単体試験実施日程 2016年2月22日

・試験方法JUnitを使用

・試験パターンは以下で行った。

Int … 0、Integer.MAX\_VALUE、Integer.MIN\_VALUE

Float … Float.NaN、Float.MAX\_VALUE、Float.POSITIVE\_INFINITY、Float.NEGATIVE\_INFINITY、0.0F

String … null、""（空白）、" "（スペース）、"abc123"、"ａｂｃ１２３あいう漢字"

Boolean … true、false

・試験対象DTO ・合計試験件数

AdminCapacitySearchDTO 　　　　　　 AdminCapacitySearchDTOTest … 20件

AdminItemDTO 　　　　　　　　　　　　　　　 AdminItemDTOTest … 128件

AdminPurchaseDTO 　　　　　　　　　 AdminPurchaseDTOTest … 45件

ContactDTO 　　　　　　　　　　　　　　　 ContactDTOTest … 62件

GoAdminPurchaseDetailDTO 　　　　　　　 GoAdminPurchaseDetailDTOTest … 76件

GoAdminUserDTO GoAdminUserDTOTest … 48件

-----------などなど-----------

以上18件 合計 1249件

上記の作業が完了したことに伴い、結合試験に進んで良いのかの判断を伺いたい。

承認者：

日付： / /

承認者：

日付： / /

承認者：

日付： / /