Spring Framework hasText(..) method

In this post we will see whether the String or implementation of CharSequence Interface has text or not.

In previous post we have seen methods isEmpty() and hasLength(..)

This method is overloaded. One method has signature of String and another has signature of CharSequence.

Both the methods are null-safe, means tit won’t throw NullPointerException.

Implementation of method that accepts “String” as parameter.

**public** **static** **boolean** hasLength(String str) {

**return** *hasLength*((CharSequence) str);

}

Implementation of method that accepts “CharSequence” as parameter.

**public** **static** **boolean** hasText(CharSequence str) {

/\*\*

\* hasLength(str)

\* returns true if str is neither null not length is 0

\* returns false if str has text.

\* \*/

**if** (!hasLength(str)) {

**return** **false**;

}

**int** strLen = str.length();

/\*\*

\* If all the characters are whitespace the return false

\* else if there is atleast one white space then return true.

\* \*/

**for** (**int** i = 0; i < strLen; i++) {

**if** (!Character.*isWhitespace*(str.charAt(i))) {

**return** **true**;

}

}

**return** **false**;

}

hasText(CharSequence str) - If the string is not null, length is greater than 0 and it has atleast one non whitespace then it will return true else not.

**package** stringutils;

**import** org.springframework.util.StringUtils;

**public** **class** SpringHasTextDemo {

**public** **static** **boolean** hasTextDemo(String str) {

**boolean** hasText = StringUtils.*hasText*(str);

**return** hasText;

}

**public** **static** **boolean** hasTextDemo(CharSequence chrSeq) {

**boolean** hasText = StringUtils.*hasText*(chrSeq);

**return** hasText;

}

}

Below are the test cases for testing both the methods

**package** stringutils;

**import** junit.framework.Assert;

**import** org.junit.Test;

**public** **class** SpringHasTextDemoTestCase {

@Test

**public** **void** testForEmpty() {

Assert.*assertFalse*(SpringHasTextDemo.*hasTextDemo*(""));

}

@Test

**public** **void** testForNull() {

Assert.*assertFalse*(SpringHasTextDemo.*hasTextDemo*(**null**));

}

@Test

**public** **void** testForString1() {

Assert.*assertTrue*(SpringHasTextDemo.*hasTextDemo*("a"));

}

@Test

**public** **void** testForString2() {

Assert.*assertTrue*(SpringHasTextDemo.*hasTextDemo*("123"));

}

@Test

**public** **void** testForString3() {

Assert.*assertTrue*(SpringHasTextDemo.*hasTextDemo*("Spring framework"));

}

//----------------------------------------------------------------------------

@Test

**public** **void** testForEmptyCharSequence() {

StringBuilder NULL=**new** StringBuilder();

NULL=**null**;

Assert.*assertFalse*(SpringHasTextDemo.*hasTextDemo*(NULL));

}

@Test

**public** **void** testForNullCharSequence() {

StringBuilder EMPTY=**new** StringBuilder("");

Assert.*assertFalse*(SpringHasTextDemo.*hasTextDemo*(EMPTY));

}

@Test

**public** **void** testForCharSequence1() {

StringBuilder STRING=**new** StringBuilder("a");

Assert.*assertTrue*(SpringHasTextDemo.*hasTextDemo*(STRING));

}

@Test

**public** **void** testForCharSequence2() {

StringBuilder WORD1=**new** StringBuilder("word");

Assert.*assertTrue*(SpringHasTextDemo.*hasTextDemo*(WORD1));

}

@Test

**public** **void** testForCharSequence3() {

StringBuilder WORD2=**new** StringBuilder("Spring Framework");

Assert.*assertTrue*(SpringHasTextDemo.*hasTextDemo*(WORD2));

}

}

Test Cases Execution