Assert library comparisons

Category	Junit4	Junit5	Truth	Assertj
Basic	Assert.assertEquals(“Frodo”, Frodo.getName()	Assert.assertEquals(“Frodo”, Frodo.getName()	*assertThat*(frodo.getName()).isEqualTo("Frodo")	*assertThat*(frodo.getName()).isEqualTo("Frodo")
	Assert.assertTrue(condition)	Assert.assertTrue(condition)		
	Assert.assertFalse(condition)	Assert.assertFalse(condition)		
	Assert.assertNull(object)	Assert.assertNull(object)		
	Assert.assertNotNull(object)	Assert.assertNotNull(object)	*assertThat*(frodo).isNotEqualTo(sauron)	*assertThat*(frodo).isNotEqualTo(sauron)
Extended data methods	N/A	N/A	assertThat(frodo.getName()).startsWith("Fro")	assertThat(frodo.getName()).startsWith("Fro")
			assertThat(frodo.getName()).endsWith("do")	assertThat(frodo.getName()).endsWith("do")
			assertThat(frodo.getName()).isEqualToIgnoringCas e("frodo")	assertThat(frodo.getName()).isEqualToIgnoringCase("fro do")
Array	Assert.assertArrayEquals(expectedArray, actualArray)	Assertions.assertArrayEquals(expectedArray, actualArray)		
Custom message	Assert.assertEquals("Custom message", expected, actual)	Assertions.assertEquals(expected, actual, "Custom message") Assertions.assertEquals(expected, actual, () -> "Custom message")		

Grouped assertions	N/A	Assertions.assertAll("Grouped assertion",
		  () -> Assertions.assertEquals(expected1,
		actual1),
		  () -> Assertions.assertEquals(expected2,
		actual2)
		)
			Assertions.assertAll("Grouped assertion",
			  () -> Assertions.assertEquals(expected1,
			actual1),
			  () -> Assertions.assertEquals(expected2,
			actual2)
			)
				Assertions.assertAll("Grouped assertion",
				  () -> Assertions.assertEquals(expected1, actual1),
				  () -> Assertions.assertEquals(expected2, actual2)
				)

Explicit failing test	Assert.fail("This test should fail")	Assert.fail("This test should fail")	Assert.fail("This test should fail")	Assert.fail("This test should fail")

Exception Testing	@Test(expected =
	IllegalArgumentException.class)
	public void testExceptionWithExpected()
	{
	throw new IllegalArgumentException("This
	is an error message"); }
		Assertions.assertThrows(
		ExpectedException.class, () -> {
		   // code that should throw the exception }
		)
			No direct support. Workaround:
			
			try {
			  throw new Exception("This is an error
			message") }
			catch (Exception e) {
			  // Use Truth to assert properties of the
			exception
			  assertThat(e).hasMessageContaining("This is an
			error message")
			  return // Test passes if exception is caught }
			
			// If no exception is thrown, the test should
			fail throw new AssertionError("Expected
			IllegalArgumentException to be thrown")
				assertThatThrownBy(() -> {
				   // Code that is expected to throw the exception
				   throw new IllegalArgumentException("This is an
				error message") })
				.isInstanceOf(IllegalArgumentException.class)
				.hasMessageContaining("This is an error message")

Timeout assertions	@Test(timeout = 2000)
	public void testTimeout() throws
	InterruptedException {
	  Thread.sleep(1500); // a delay
	}
		assertTimeout(Duration.ofSeconds(2), () -> {
		   // Code that should complete within 2
		seconds
		  Thread.sleep(1000);
		});
			Assertions.assertTimeout(Duration.ofMillis(100),
			() -> {
			   // code that should complete within 100
			milliseconds }
			)
				Assertions.assertTimeout(Duration.ofMillis(100), () ->
				{
				   // code that should complete within 100
				milliseconds }
				)

Fluent assertions	N/A	N/A	assertThat(frodo.getName()).startsWith("Fro")
			        .endsWith("do")
			        .isEqualToIgnoringCase("frodo")
				assertThat(frodo.getName()).startsWith("Fro")
				        .endsWith("do")
				        .isEqualToIgnoringCase("frodo")

Messaging	assertEquals("check " + frodo.getName()
	+ "'s age", frodo.getAge(), 33);
		assertEquals(frodo.getAge(), 33, "check " +
		frodo.getName() + "'s age");
			assertThat(frodo.getAge()).as("check %s's age",
			frodo.getName()).isEqualTo(33)
				assertThat(frodo.getAge()).as("check %s's age",
				frodo.getName()).isEqualTo(33)