**jUnitin sundugu standart assetion'lar.**

**assertEquals,**

**assertNotEquals,**

**assertTrue,**

**assertFalse,**

**assertNull,**

**assertNotNull,**

**assertArrayEquals,**

**assertSame**

**assertEquals**(expected, actual):

* ilk parametre beklenen değer, ikinci parametre kod çalıstıgında(test ettiğimiz senaryo tamamlandıktan sonra dönen değer) dönen değer olmak üzere iki parametre alır.
* Bu iki değeri javadaki equals methodu ile karşılaştırır.
* Actual ile expected değer aynı ise hata vermez, farklı ise hata verir ve test fail olur.
* 3 . parametre(String) olarak hata durumunda gösterilmesinin istediğimiz mesajı da verebiliriz.

Example:



Test Cıktısı:



**assertNotEquals():**

assertEquals' un tersidir. Verilen parametrerlerin eşit olmaması durumunda testi geçer. Eşit olması durumunda fail olur.

**assertTrue():**

boolean bir condition veya boolean supplier alıp expected(beklenen) değer ile karşılaştırır.

**assertFalse():**

assertTrue'nun tam tersi

**assertArrayEquals():**

iki tane array' i karşılaştırır

Array' lerin elemanlarının sırası ve değerleri bir biriyle uyumlu ise, test true alır ve herhangi bir hata vermez.

Opsiyonel olarak 3. parametrede hata mesajı verilebilir.

**assertSame():**

verilen iki objenin referans değeri olarak birebir aynı obje olup olmadıgının kontrolünü yapar.

**assertNotSame():**

assertSame' in tam tersidir.

**assertNull(object):**Verilen objenin null olup olmadığına bakar, null ise test passed(başarılı test) olacaktır.

**assertNotNull(object):**Verilen objenin null olup olmadığına bakar, null değilse test passed(başarılı test) olacaktır.

**assertAll:** Birden fazla asserti aynı anda çalıştırmak için kullanırız. Neden ayrı ayrı çalıştırınca bundan farkı ne diyeceksin farkı şu; normal assertleri alt alta yazsaydık eğer diyelim ilk test passed olamadı başarısız oldu , diğer testlere geçmiyor fakat assertAll ile test passed olsun yada olmasın tüm testleri çalıştıracak.

**assertThrows(Exception,excepted Exception):** Bu assert ile dönmesini beklediğimiz hatayı test edebiliriz. Örneğin: Burada bir createUser metodu var ve parametresi olan user boş gelirse İllegalArgumentException fırlatacak.

public User createUser*(*User user*){* if*(*Objects.*isNull(*user*)){* throw new IllegalArgumentException*()*;  
 *}* return user;  
*}*

Burada ise onun test metodu var. Bu assert, test ettiği metod hata verirse passed olacak\*\*\*Yani user boş gelirse ve createUser metodu illegalArgumentExc fırlatırsa test başarılı olacak

*assertThrows(*IllegalArgumentException.class, *()* -> user1.createUser*(*user1*))*;