JUnit 4.* Параметризовані тести.

```
@RunWith(Parameterized.class)
public class Util1Test {
Util1 ut1=new Util1();
private int par1;
private int par2;
private int rezultat;

public Util1Test ( int atr1 , int atr2 , int rez ) {
   this.par1=atr1;
   this.par2=atr2;
   this.rezultat=rez;
   }

@Parameters
public static Collection my_param(){
   ArrayList<Object[]> arls1=new ArrayList<Object[]>();
   Object[] ar11={10,10,20};
   Object[] ar12={15,15,30};
   arls1.add(ar11);
   arls1.add(ar12);
   return arls1;
}
   @BeforeClass
   public static void setUpClass() throws Exception {
      System.out.println("Это разовая фикстура перед "
            + "запуском тестового класса");
   }
   @AfterClass
   public static void tearDownClass() throws Exception {
   }
   @Before
   public void setUp() {
   }
   @After
   public void tearDown() {
   }

  @Test
   public void test_sum_metod() {
      int temp=ut1.sum(this.par1,this.par2);
     assertEquals(temp, this.rezultat);
   }
}
```

Ключовий момент № 1
Ось така Аннотація

Ключовий момент № 2
необхідність приватних атрибутів класу , кількість та тип яких відповідає параметрам тестуємого методу + атрибут для результату , який повертає метод

Ключовий момент № 3
Відповідний конструктор

Це наші тестові набори.

Ключовий момент № 4
Наявність статичного методу , перед яким застосована аннотація @Parameters та тип повертаємого значення є любий клас , який імплементує інтерфейс Collection

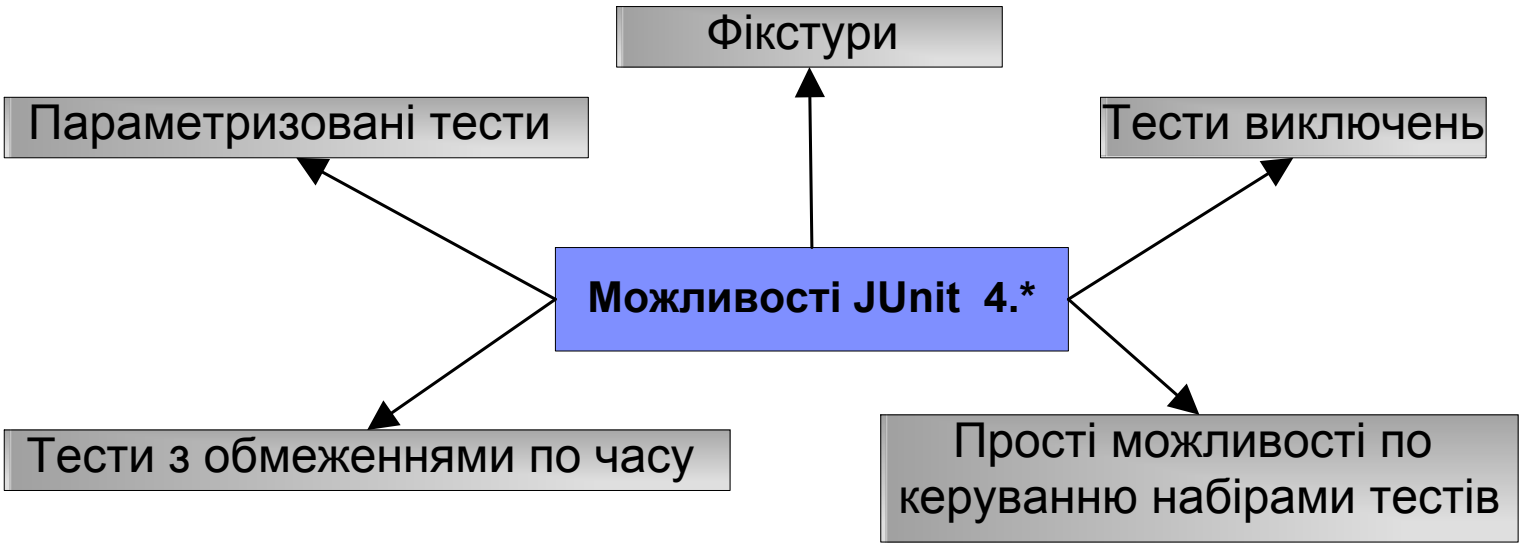
Ключовий момент № 5
У виклику методу, який тестуємо , у якості параметрів використовуємо атрибути класу.

JUnit 4.* Тестові серії.

```
@RunWith(Suite.class)
@SuiteClasses({NewClass.class,Util1Test.class})
 public class NewClass1 {

}
```

Ключові моменти № 1
Повинні бути ці Аннотації з переліком класів , які містять тести



JUnit 4.* Тестування виключень.

```
@Test(expected=IndexOutOfBoundsException.class)
 public void verifyZipCodeGroupException() throws
Exception{
  Matcher mtcher = this.pattern.matcher("22101-
5051");
  boolean isValid = mtcher.matches();
  mtcher.group(2);
 }
```

Ключовий момент № 1
Поряд з Аннотією @Test ми повинні указати виключення , яке повинно бути отримано під час тесту

JUnit 4.* Тестування з обмеженнями по часу.

```
@Test(timeout=1000)
public void verifyFastZipCodeMatch() throws
Exception{
 Pattern pattern = Pattern.compile("^\\d{5}([\\
-]\\d{4})?$");
 Matcher mtcher = pattern.matcher("22011");
 boolean isValid = mtcher.matches();
 assertTrue("Pattern did not validate zip
code", isValid);
}
```

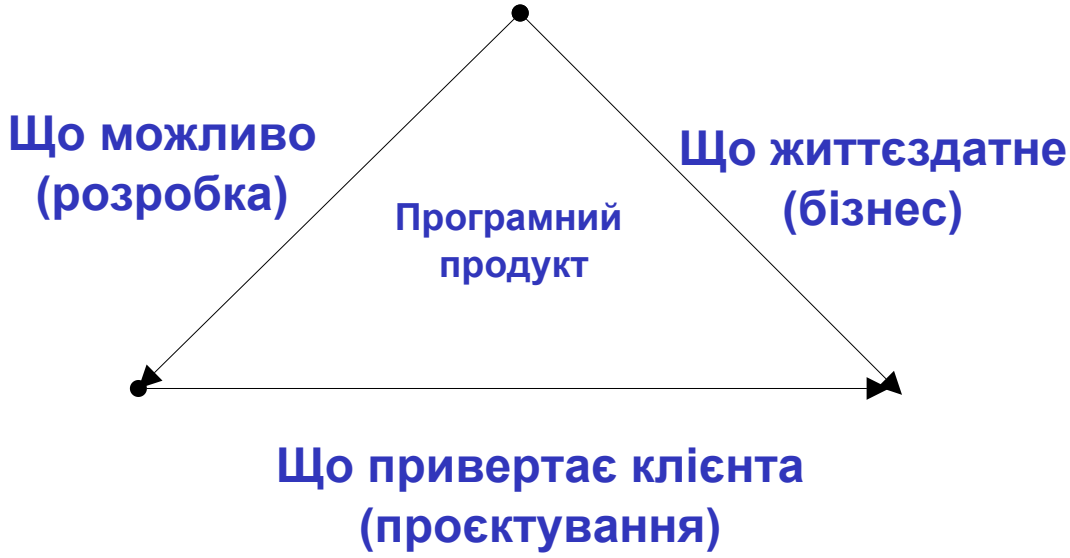
Ключові моменти № 1
Час вимірюється у міллісекундах.
Конкретно у цьому випадку , як що тест виконується більше ніж 1 сек. то він бракується.

JUnit 4.* Ігнорування тестів у разі необхідності.

```
@Ignore("this regular expression isn't working yet")
@Test
public void verifyZipCodeMatch() throws Exception{

 Pattern pattern = Pattern.compile("^\\d{5}([\\-
]\\d{4})");
 Matcher mtcher = pattern.matcher("22011");
 boolean isValid = mtcher.matches();
 assertTrue("Pattern did not validate zip code",
isValid);
}
```

Ключові моменти № 1
Ця Аннотація дозволяє тимчасово не запускати цей тест.


Що можливо
(розробка)
Програмний
продукт
Що життєздатне
(бізнес)
Що привертає клієнта
(проєктування)