**Introducción**

***Sobre este documento***

En esta sección se listan una serie de pruebas de regresión que se le aplican al sistema “Walkers of thecity” para verificar su correcto funcionamiento. Para realizar dichas pruebas se aplicoUnittesting a través de la librería JUnit de java. Aunque muchos de los test mostrados no están implementados, se listan para que se implementen cuando sea el momento; ya que se considera que son esenciales para mantener un mínimo de control sobre el funcionamiento del sistema.

**Modelo general de testing con JUnit**

***Template genérico***

Para hacer UnitTesting con JUnit desarrollamos el siguiente template que nos sirve como modelo general para desarrollar los casos de test, además de especificar ciertas convenciones de escritura y de programación.

**import**junit.framework.TestCase;

**publicclass** test\_[NOMBRE\_CLASE]**extends**TestCase {

[OBJETOS NECESARIOS PARA REALIZAR EL TEST]

**publicvoid**setUp(){

[INSTACIAR LOS OBJETOS NECESARIOS]

[LLEVAR LAS INSTACIAS AL ESTADO ADECUADO]

}

**publicvoid** test[NOMBRE\_METODO]() {

[PROCESO DE TESTING]

[USAR ALGUN TIPO DE *ASSERT*]

}

}

***Tipos de assert***

* assertEquals(expected, actual)
* assertEquals(message, expected, actual)
* assertEquals(expected, actual, delta) - used on doubles or floats, where delta is the difference in precision
* assertEquals(message, expected, actual, delta) - used on doubles or floats, where delta is the difference in precision
* assertFalse(condition)
* assertFalse(message, condition)
* assertNotNull(object)
* assertNotNull(message, object)
* assertNotSame(expected, actual)
* assertNotSame(message, expected, actual)
* assertNull(object)
* assertNull(message, object)
* assertSame(expected, actual)
* assertSame(message, expected, actual)
* assertTrue(condition)
* assertTrue(message, condition)

**UnitTesting**

***Caso N***

Objeto del test:[que se está testeando]

Estado:[implementado / no implementado]

Descripción del test:[como se está testeando (en lo referente a la implementación)]

Condiciones de ejecución:[condiciones previas a la ejecución del test]

Resultado:[éxito / falla]

**UnitTesting**

***Caso 1***

Objeto del test:Conexión tcp de parte del cliente al servidor Localhost y puerto 5000(estado de conexión de un objeto de la clase Conexión)

Estado:implementado

Descripción del test:se testea la dirección del puerto de origen del socket correspondiente, de ser 0 es porque no se produjo la conexión.

Condiciones de ejecución:Se debe tener un puerto abierto por parte del servidor

Resultado:Falla (no esta habilitado el puerto)

**import** junit.framework.TestCase;

**publicclass** test\_Conexion **extends** TestCase {

Conexion c;

**publicvoid** setUp()

{

c = **new** Conexion();

}

**publicvoid** testconectar() {

//fail("Not yet implemented");

c.conectar("localhost", 5000);

*assertEquals*("no hay conexion", 0, c.getSocket().getPort());

}

/\*De no producirselaconexion el puertoen el socket quedacomo 0

getPort() Returns:

the remote port number to which this socket is connected,

or 0 if the socket is not connected yet.\*/

}