MC636 – Verificação e Validação de Software

JUnit

Teste de Unidade

"Teste focado nas menores unidades do software, que podem ser funções, procedimentos, métodos ou classes, para avaliar se os aspectos de implementação de cada unidade estão corretos "

Geralmente, há frameworks que ajudam na construção desse tipo de teste. No caso de aplicações Java, existe o JUnit, um framework open-source que verifica se os métodos fazem o que se espera, ou seja, retornam o resultado esperado.

Para mais informações sobre o Junit: http://www.junit.org/

- **Como aplicar?**

Abaixo há um exemplo de como são criados os testes de unidade.

Considere um programinha estilo batalha naval, porém, o programa define a posição dos objetos e o usuário "chuta" a posição de cada um. Abaixo há apenas uma classe, o programa completo pode ser visto em http://www.students.ic.unicamp.br/~ra059664/mc636/dotCom.zip.

Classe *DotCom*, é a classe que configura uma instância *DotCom* para ser posicionada no tabuleiro:

```
//classe para configuracao de cada objeto DotCOm
public class DotCom{
	//array para colocar a posicao dos objetoos
	private ArrayList<String> locationCells;
	private String name;

	//metodo que atualiza o local de um objeto DotCom
	//local aleatorio fornecido pelo metodo placeDotCom da classe
GameHelper
	public void setLocationCells(ArrayList<String> loc){
		locationCells = loc;
	}

	public ArrayList<String> getLocationCells(){
		return locationCells;
	}

	public void setName(String n){
		name = n;
	}

	public String getName(){
		return name;
	}

	//verifica se o palpite do usuario foi correto, se ja eliminou
um objeto ou se foi errado
```

```
	public String checkYourself(String userInput){
		String result = "errado";
		int index = locationCells.indexOf(userInput);
		if(index >= 0){
			locationCells.remove(index);
			if(locationCells.isEmpty()){
				result = "eliminar";
				System.out.println("Voce afundou " + name);
			}
			else{
				result = "correto";
			}
		}
		return result;
	}
}
```

Para esta classe, foram testados os métodos *setName()* e *setLocationCells()*. Para isso, criou-se uma classe do tipo JunitTestCase de nome *DotComTeste* que ficou da seguinte forma:

```
public class DotComTeste extends TestCase {

	DotCom teste = new DotCom();
	private ArrayList<String> locationCells = new
ArrayList<String>();
	private ArrayList<String> coordenadas = new ArrayList<String>();

	protected void setUp() throws Exception {
		super.setUp();
		System.out.println("Iniciando...");
		setaLocalidades();
	}

	public void setaLocalidades(){
		locationCells.add("b1");
		locationCells.add("b2");
		locationCells.add("b3");

		coordenadas.add("b1");
		coordenadas.add("b2");
		coordenadas.add("b3");
	}

	public void testSetLocationCells() {
		//setaLocalidades();

		teste.setLocationCells(coordenadas);

		assertEquals(teste.getLocationCells(), locationCells);
	}

	public void testSetName() {
		teste.setName("TesteNome");

		assertEquals(teste.getName(), "TesteNome");
	}

	protected void tearDown() throws Exception {
		super.tearDown();
		 System.out.println("Finalizando...");
	}
```

```
}
```

Explicando o código:

Os métodos *setUp()* e *tearDown()* servem para inicializar e finalizar os casos de teste, respectivamente. Não são obrigados a existirem no código de teste. No caso do exemplo acima, esses métodos mostram que os testes estão iniciando, inicializam os atributos necessários e depois, mostra a finalização dos testes.

O método *setaLocalidades()* é apenas um auxílio da classe para setar os atributos.

Os métodos *testSetLocationCells()* e *testSetName()* testarão os métodos *setLocationCells()* e *setName()* da classe *DotCom*.

O JUnit possue várias asserções que podem ser utilizadas nos testes, como

1. assertEquals(objEsperado, objRecebido);
2. assertArrayEquals(objEsperado, objRecebido);
3. assertTrue(expBooleana);
4. assertNull(obj);
5. assertNotNull(obj);
6. assertSame(obj1, obj2);
7. fail(mensagem);

Para mais informações: http://kentbeck.github.com/junit/javadoc/latest/org/junit/Assert.html

- **Instalando o JUnit**

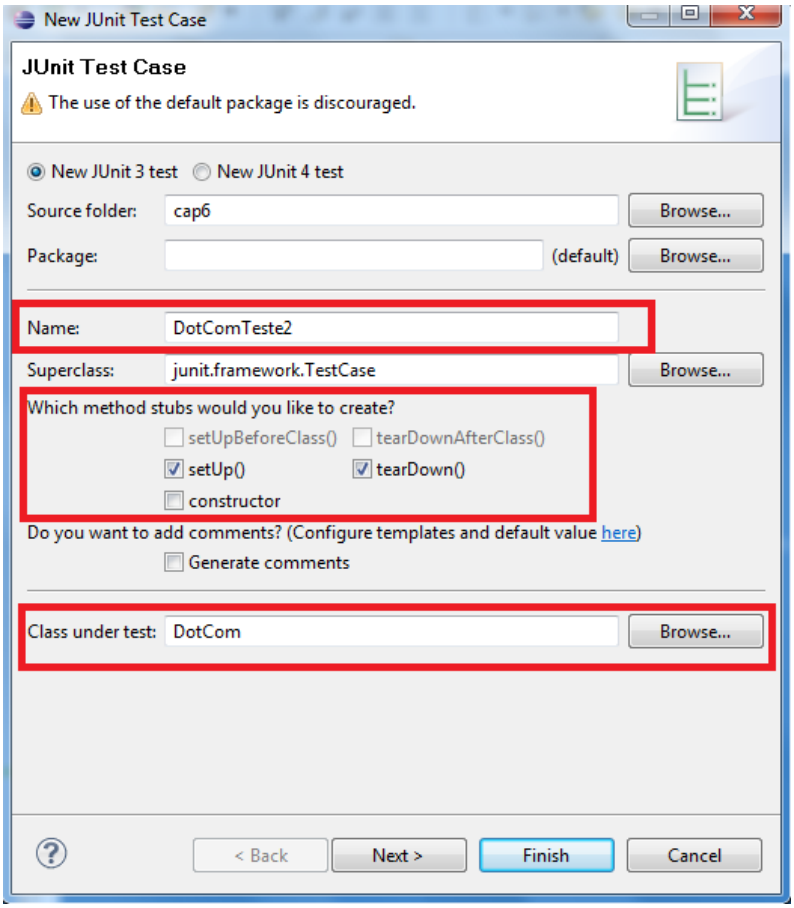
Para instalar o JUnit no Eclipse Helios:
Observação: os computadores do IC3, nas salas 302 e 303 já estão com o JUnit instalado.

1. Baixar o jar do JUnit em https://github.com/KentBeck/junit/downloads
2. Abrir o Eclipse. Clicar em **Project -> Properties**
3. Selecionar **Java Build Path**, **Libraries**
4. Clicar em **Add External JARs** e procurar onde a pasta do JUnit. Escolha o junit.jar e clique em **Open**.
5. Você verá que JUnit aparecerá na lista de bibliotecas. Clique em OK e o Eclipse reconstruirá os build paths.

- **Classe de teste**

Para criar uma classe de teste:

1. No seu projeto do Eclipse, clicar com o botão direito no pacote que está a classe a ser testada, New -> JUnit Test Case
2. Na janela que abrir, se houver a opção New JUnit 3 test, escolher essa. Caso não, só possuir a 4, olhar observação abaixo, pois o código muda um pouco. Colocar o nome da sua classe de teste, escolher se deseja que possua setUp e tearDown, e também pode escolher a classe que irá testar. Se escolher a classe de teste, o Eclipse mostrará as opções de métodos que deseja testar clicando em Next.



3. Selecionar Finish e a classe será gerada. Basta agora completar o código

Observação: Caso apenas a opção New JUnit 4 test estiver disponível, fazer os mesmos passos anteriores, e após clicar em Finish, aparecerá o código abaixo:

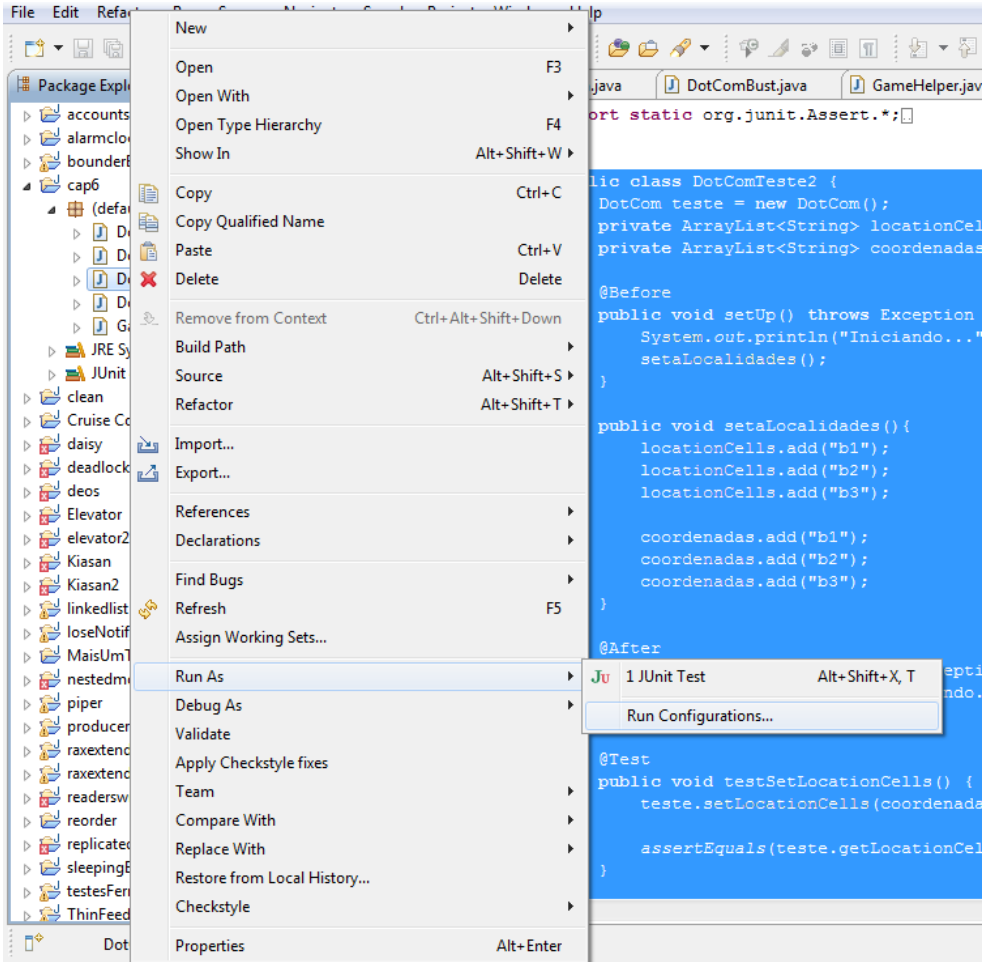
```
public class DotComTeste2 {
	DotCom teste = new DotCom();
	private ArrayList<String> locationCells = new
ArrayList<String>();
	private ArrayList<String> coordenadas = new ArrayList<String>();

	@Before
	public void setUp() throws Exception {
		System.out.println("Iniciando...");
		setaLocalidades();
	}

	public void setaLocalidades(){
		locationCells.add("b1");
		locationCells.add("b2");
		locationCells.add("b3");

		coordenadas.add("b1");
		coordenadas.add("b2");
		coordenadas.add("b3");
	}

	@After
	public void tearDown() throws Exception {
		System.out.println("Finalizando...");
```

```
}

@Test
public void testSetLocationCells() {
	teste.setLocationCells(coordenadas);

	assertEquals(teste.getLocationCells(), locationCells);
}

@Test
public void testSetName() {
	teste.setName("TesteNome");

	assertEquals(teste.getName(), "TesteNome");
}
```

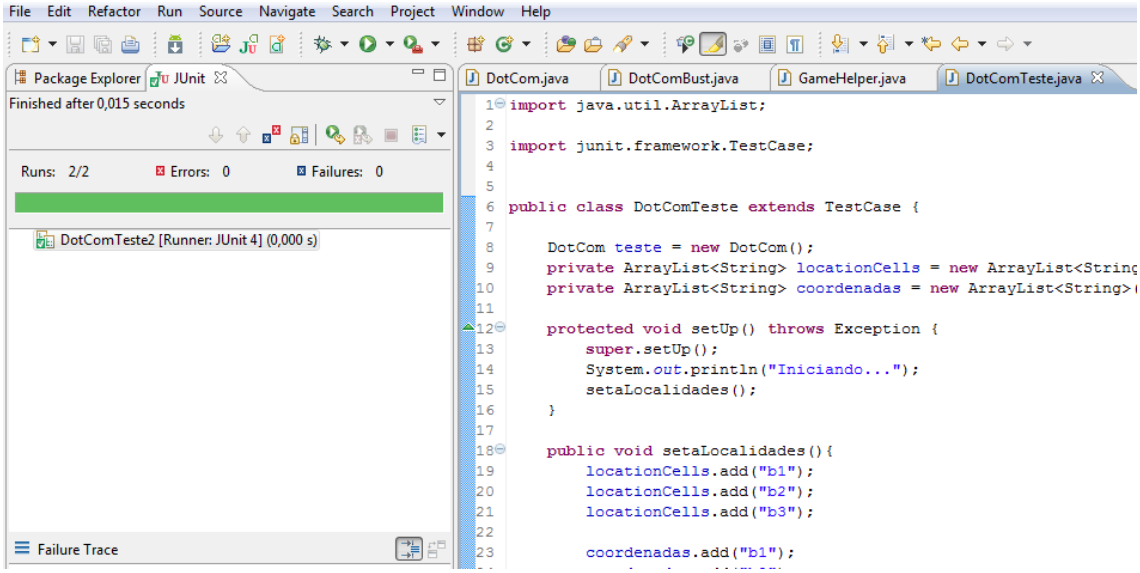
Os textos antes de cada método são parte de uma sintaxe formal, utilizada para saber a ordem das coisas, como no caso @after e @before e o que ocorrerá, como @Test. No caso, os métodos setUp() e tearDown() não precisam do uso do super.

- **<u>Rodar o teste</u>**

Para rodar um teste:
Após seus testes serem criados, clicar com o botão direito em cima do nome da classe de teste, ir em Run As -> JUnit Test Case



Uma janela com o status do teste será aberta. Se os casos de teste passar, aparecerá uma barra verde mostrando que os testes passaram.



Caso o algum teste falhe, a barra ficará vermelha e o Eclipse mostrará o que ocorreu de errado: