



Programación

Tema 8: Estilo y pruebas

The logo for Cartagena99 features the text "Cartagena99" in a stylized, green, serif font. The "99" is larger and more prominent. The text is set against a background of light blue and orange wavy lines.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

1

Contenidos

- Estilo
- Pruebas
- Depuración

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Calidad del Software

- Estilo, prueba, depuración: aspectos relacionados con el...
- Aseguramiento de la **Calidad del Software** (QA, **Quality Assurance**):
Conjunto de políticas y actividades para:
 - Definir objetivos de calidad
 - Ayudar a asegurar que los productos y procesos software cumplen esos objetivos
 - Saber en qué medida lo hacen

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Estilo de código

- Comprensible
- Cohesión
- Desacoplamiento
- Formato (sangrado...)
- Nombres consistentes
- Ocultamiento de información
- Comentarios significativos

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Estilo: objetivos

- Disminuir errores
- Asegurar **legibilidad**
 - El código se escribe para ser leído por otros (no solo por el compilador)
- Abaratar/**facilitar mantenimiento**
- Estándares de codificación en Java:
 - The Elements of Java Style
Vermeulen et.al. SIGS Books

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

- - -

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Estilo: reglas generales

- Simplicidad
 - **Clases y métodos simples**
- Claridad
 - Propósito claro de cada elemento: dónde, cuándo, porque y cómo usarlo
- Completitud
 - Documentar todos los aspectos necesarios
- Consistencia
 - Entidades similares deben tener aspecto y comportamiento similar
- Robustez
 - Comportamiento predecible y documentado en respuesta a errores y excenciones



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Estilo: nombrado

- Clases, Interfaces: Mayuscula-Mayuscula
 - Ej. Punto, PolinomioGrado2
- Métodos: minuscula-Mayuscula
 - Ej. getX(), distancia(), buscaMayor()
- Atributos: minuscula-Mayuscula
 - Ej. nombrePropio, x, y, apellidos
- Constantes: MAYUSCULA + “_”
 - Ej. ORIGEN_X

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

- - -

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Estilo: parámetros

- Orden de parámetros:
 - Entrada/modificación/salida
 - Consistente en todos los métodos
- Usar todos los parámetros
- Limitar número de parámetros (nunca >7)
- Documentar suposiciones sobre parámetros:
 - **Tipo de acceso, unidades, rangos, valores no válidos**

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Estilo: comentarios

- Comentarios de código

```
/* ...
```

```
*/
```

```
// comentario táctico: hasta fin de línea
```

- Comentarios de documentación

```
/**
```

```
**/
```

- Herramienta **javadoc** para generar páginas

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Programación DIT-UPM

- Estilo
- Pruebas
- Depuración

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

- Valen para detectar errores
- Una prueba es tanto mejor cuantos menos errores pasan desapercibidos
- Hay que programar, ejecutar y documentar las pruebas antes de dar por acabado una clase
- Permiten ver cómo se usa la clase
- Al hacer pruebas hay que **intentar que falle** el programa o la clase por todos los medios

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Tipos de pruebas funcionales

- **Pruebas unitarias:** prueban el menor elemento posible, una clase o un método
- **Pruebas de integración:** interacción entre clases o paquetes
- **Pruebas de sistema:** prueban el programa en contexto real
- **Pruebas de aceptación:** prueba del programa para ver que satisface los requisitos

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Enfoques de prueba

- De **caja negra**
 - cuando conocemos la parte pública de una clase
- De **caja blanca**
 - cuando conocemos la parte privada y forzamos la ejecución de todo el código

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

- - -

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Casos de prueba

- ¿Qué hay que probar?
 - ejecutar al menos una vez cada sentencia
 - cobertura de sentencias
 - ejecutar al menos una vez cada condición con resultado cierto y falso
 - cobertura de ramas
- Método
 - si va tachando el código probado
100% cuando todo esté tachado

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Casos de prueba

- if (...)
 - si T, si F
- switch (...)
 - cada 'case' + 'default'
- cobertura de bucles
 - for -> 3 pruebas: 0 veces, 1 vez, n>1 veces
 - repeat -> 2 pruebas: 1 vez, n>1 veces
 - while -> 3 pruebas: 0 veces, 1 vez, n>1 veces

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

- - -

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Datos de prueba

- ¿Con qué datos se prueba?
 - divida el espacio de datos en **clases de equivalencia**
 - clase de equivalencia:
datos que provocan el “mismo comportamiento”
 - no parece que deban provocar comportamientos diferentes
 - elija un **dato “normal”** de clase de equivalencia
 - pruebe con todos los **datos “frontera”**:
valores extremos de la clase de equivalencia
- Añada aquellos **casos en los que sospeche** que el programador puede haberse equivocado
- Pruebe todas las combinaciones

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Pruebas automáticas: JUnit

- **Framework** para pruebas unitarias de clases
 - Facilita la escritura de pruebas
 - Automatiza su ejecución
 - Uso independiente (desde línea de comandos) o **integrado en IDEs** (Eclipse, Netbeans)
 - Autores: Kent Beck and Erich Gamma.
 - Web: junit.org
 - Versión estable: 4.11 (nov.2012)

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

- - -

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Casos de prueba

- Conjunto de **métodos java (anotados)**:
 - secuencia de operaciones, entradas y valores esperados (escritos por quien prueba)
 - Sustituyen el uso de main() para probar
- **Añade aserciones** (clase) para comprobar:
 - Si dos objetos son iguales
 - si dos referencias a objetos son idénticas
 - Si una referencia a un objeto es nula o no

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

- - -

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Prueba con JUnit 4.x

- **Importar clases** JUnit

```
import org.junit.*;  
import static org.junit.Assert.*;
```

(import static: para referenciar métodos/atributos sin citar la clase)

- Declarar clase de prueba (normal)

```
public class MiClaseTest {
```

- Declarar variables para prueba:

```
MiClase c;...
```

- Uso de **anotaciones** (metadatos) @...

Anotación de métodos

- **@Test**: método que representa un caso de prueba
 - Comprueba si el resultado esperado es igual al real
 - Métodos Assert: assertTrue, assertFalse, assertEquals, assertNull, assertNotNull, assertEquals, assertEquals
- **@Before/@After**: métodos que deben ejecutarse antes/después de cualquier caso de prueba
- **@BeforeClass/@AfterClass**: se ejecutan **una vez, antes/después** de todos los test de la clase
 - Puede haber varios métodos etiquetados con

Cartagena99

CLASES PARTICULARES, TUTORIAS TECNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Ejemplo 1

```
import org.junit.*;
import static org.junit.Assert.*;

@Before
public void setUp() {
    ej = new ClaseEjemplo();
}

@Test
public void testSuma( ) {
    int a=3 , b=6;
    int resultEsperado = (a+b);
    int resultReal = ej.suma(a, b);
    assertEquals(reusultEsperado, resultReal);
}
```

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Métodos de clase de prueba

- Sugerencia: nombrar métodos que comiencen por **test**
 - Llama al método que se desea probar y obtiene el resultado
 - Comprueba que el resultado es el esperado
 - Se repiten estos pasos tantas veces como sea necesario



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

- - -

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

- **assertEquals (esperado, obtenido)**

- Comprueba si los objetos (o variables de tipo primitivo) son iguales, empleando el método equals() si está definido. Si no, emplea ==

- **assertArrayEquals (esperado, obtenido)**

- arrays de la misma longitud
- para cada valor del índice, se comprueba:
 assertEquals(expected[i],actual[i])
 o

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Aserciones (cont.)

- **assertTrue** (boolean comprobacion) - **assertFalse**
 - Éxito si la comprobación es cierta (falsa)
- **assertSame** (Objeto esperado, Objeto obtenido) - **assertNotSame**
 - Éxito si referencias iguales empleando == (o distintas)
- **assertNull** (Objeto objeto)
 - Éxito si el objeto es null (o no lo es)
- **fail** (String mensaje)
 - Causa fallo en la prueba. Para comprobar si una excepción se lanza y se captura.

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

- - -

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Ejemplo 2

```
public class Contador {  
    private int contador = 0;  
  
    public int incrementa() {  
        return ++contador;  
    }  
  
    public int decrementa() {  
        return --contador;  
    }  
  
    public int getContador() {  
        return contador;  
    }  
}
```

Los métodos triviales como getContador() no necesitan prueba asociada.

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Programación DIT-UPM

Ejemplo 2 (cont)

```
import static org.junit.Assert.*;
public class TestContador {
    private Contador miContador;

    @Before
    public void setUp() throws Exception {
        miContador= new Contador();
    }

    @After
    public void tearDown() throws Exception {}

    @Test
    public void testIncremental1() {
        assertEquals(1, miContador.incrementa());
    }

    @Test
    public void testIncrementa2() {
        miContador.incrementa();
        assertEquals(2, miContador.incrementa());
    }
}
```

Cada prueba comienza con un contador *nuevo*, con lo que no hay que preocuparse del orden de las pruebas

Cada prueba permite comprobar un elemento específico=> un *assert*



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Programación DIT-UPM

Aserciones: mensajes

- Cualquier aserción tiene una signatura adicional con un **parámetro String inicial**.
 - Es un mensaje adicional que se incluirá en el mensaje de fallo (si lo hubiera)
- Ejemplos:

`assertTrue (mensaje, comprobacion)`

`assertEquals (mensaje, esperado, real)`

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Aserciones sobre reales

- En la comparación de resultados reales se requiere un **parámetro adicional (delta)**

- Para evitar problemas de redondeo, se evalúa:

`Math.abs( resEsperado - resReal) <= delta`

- Ejemplo:

`assertEquals (resEsperado, resReal, 0.0001)`



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

Prueba de excepciones

- Pasando parámetro a la **anotación @Test con el tipo de excepción esperada:**

```
@Test(expected=ImporteNoDisponibleException.class)
public void reintegroExcesivo () throws
    ImporteNoDisponibleException {
    Cuenta c = new Cuenta("José Pérez", 1000);
    c.reintegro(1001);
}
```

- El test falla si no se lanza excepción (o si se lanza otra distinta de la esperada)



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Tiempo de ejecución

- Para evitar bucles infinitos, se pueden establecer **límites de tiempo** (en milisegundos)
- La prueba falla si el método se demora en exceso:

```
@Test (timeout=10)
public void testTimeout() {
    Cuenta c = new Cuenta ("José Pérez", 1000);
    c.bucleInfinito();
}
```



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

- Estilo
- Pruebas
- Depuración

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

- Consiste en encontrar y arreglar los errores detectados durante las pruebas
- Actividad difícil, manual
- Es necesario imaginar el flujo de ejecución y seguirlo para encontrar los errores
- Varias técnicas:
 - Revisión manual: revisar el listado del programa, llevando la cuenta de cómo están los objetos
 - Depuración en voz alta: leer el programa a otros

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70