

PROGRAMACIÓN ORIENTADA A OBJETOS

Herencia e interfaces

ADEMÁS Java desde consola

Septiembre 2017

Laboratorio 3/6

OBJETIVOS

Desarrollar competencias básicas para:

1. Aprovechar los mecanismos de la herencia y el uso de interfaces.
2. Organizar las fuentes en paquetes.
3. Usar la utilidad `jar` de java para entregar una aplicación.
4. Extender una aplicación cumpliendo especificaciones de diseño, estándares y verificando su corrección.
5. Vivenciar las prácticas XP : Use [collective ownership](#). Only one pair [integrates code at a time](#).
6. Utilizar los programas básicos de java (`javac`, `java`, `javadoc`, `jar`), desde la consola.

ENTREGA

- ➔ Incluyan en un archivo `.zip` los archivos correspondientes al laboratorio. El nombre debe ser los dos apellidos de los miembros del equipo ordenados alfabéticamente.
- ➔ En el foro de entrega deben indicar el estado de avance de su laboratorio y los problemas pendientes por resolver.
- ➔ Deben publicar el avance al final de la sesión y la versión definitiva en la fecha indicada en los espacios preparados para tal fin.

DESARROLLO

Contexto

En esta aplicación se presenta un salón de un gimnasio, en nuestro caso el BodyTic.

Conociendo [En lab03.doc y BodyTic.asta]

1. En el directorio descarguen los archivos contenidos en `BodyTic.zip`. Revisen el código de la aplicación a) ¿Cuántos paquetes tiene? b) ¿Cuántas clases tiene en total? c) ¿Cuál es la clase ejecutiva? ¿Por qué?
2. Ejecuten el programa. ¿Qué funcionalidades ofrece? ¿Qué hace actualmente? ¿Por qué?

Arquitectura general. [En lab03.doc y BodyTic.asta]

1. Consulte el significado de las palabras `package` e `import` de java. Explique su uso con un ejemplo del proyecto.
2. ¿Qué es un paquete? ¿Para qué sirve? Inicie el diseño con un diagrama de paquetes en el que se presente los componentes y las relaciones entre ellos.
3. Revise el contenido del directorio de trabajo y sus subdirectorios. Describa su contenido. ¿Qué coincidencia hay entre paquetes y directorios?

Arquitectura detallada. [En lab03.doc y BodyTic.asta]

1. Usando ingeniería reversa. Presente el diseño actual de la aplicación.
2. ¿Qué observaciones haría al respecto?

Ciclo 1. Iniciando con los deportistas normales [En lab04.doc y *.java]

(NO OLVIDE BDD - MDD)

1. Estudie la clase **Salon** ¿Qué tipo de colección se usa para albergar los elementos? ¿Puede recibir deportistas? ¿Por qué?
2. Estudie el código de la clase deportista ¿de qué color es? ¿qué palabras dice? ¿cómo inician? ¿cómo paran? ¿cómo deciden?
3. En este punto vamos a construir (diseño y código) el método que atiende el clic del botón **Entren** de la interfaz: el método llamado **entrada()** de la clase **Salon**. En ese método creen dos deportistas en diferentes posiciones llámelos **edward** y **bella**. Ejecute el programa y hagan clic en el botón **Entren**. Capture una pantalla significativa.
4. En este punto vamos a construir (diseño y código) el método que atiende el clic del botón **Inicien** de la interfaz: el método llamado **inicie()** de la clase **Salon** Ejecute el programa y haga tres clic en el botón **Inicien**. ¿Cómo actúan **edward** y **bella** ? Capture una pantalla significativa.
5. En este punto vamos a construir (diseño y código) el método que atiende el clic del botón **Paren** de la interfaz: el método llamado **pare()** de la clase **Salon** Construya el método, ejecute el programa y haga clic en el botón **Paren**. ¿Cómo quedan todos los deportistas? Capture una pantalla significativa.
6. En este punto vamos a construir (diseño y código) el método que atiende el clic del botón **Decidan** de la interfaz: el método llamado **decida()** de la clase **Salon** Construya el método, ejecute el programa y haga clic en el botón **Decidan**. ¿Como quedan los deportistas? Capture una pantalla significativa.
7. En este punto vamos a construir (diseño y código) el método que atiende el clic del botón **Salgan** de la interfaz: el método llamado **salga()** de la clase **Salon** Construya el método, ejecute el programa y haga clic en el botón **Salgan**. ¿Como queda el salón? Capture una pantalla significativa.

Ciclo 2. Incluyendo a los deportistaa avanzados [En lab04.doc, BodyTic.asta y *.java]

El objetivo de este punto es permitir recibir en el salón deportistas avanzados.

(NO OLVIDE BDD - MDD)

1. Los deportistas avanzados, vestidos de naranja, se mueven más, sólo paran después de dos órdenes y, cuando les piden que decidan, siempre hacen ejercicio. Implemente este nuevo deportista. ¿cuáles métodos se sobrescriben (*overriding*)?
2. Adicione una pareja de deportistas avanzados, llámelos **neo** y **trinity**, ejecute el programa y pídale a todos que entren, inicien, decidan y paren. Capturen pantallas significativas. ¿Qué pasa?
3. Ahora, los avanzados quieren sorprender con su resistencia; es decir, sólo son paran muy pocas veces. ¿Qué modificaría para lograr este comportamiento? ¡Hágalo!
4. Nuevamente ejecute el programa y pídale a todos que entren, inicien, decidan y paren. Capturen una pantalla significativa. ¿Qué pasa?

Ciclo 3. Adicionando bolas [En lab03.doc, BodyTic.asta y *.java]

El objetivo de este punto es incluir en el **Salon** bolas de pilates de calor (sólo vamos a permitir un tipo de bolas). Las bolas se encienden cuando inician, apagan cuando paran y deciden de manera estándar.

(NO OLVIDE BDD - MDD)

5. Construya la clase **Balon** para poder adicionarla en el **Salon** ¿qué cambios incluyó
6. Para aceptar este elemento , ¿debe cambiar en el código del **Salon** en algo? ¿por qué?
7. Adicionen dos balones cerca al el centro del **Salon**, llámenlas **derecha** y **izquierda**, ejecuten el programa. Capturen una pantalla significativa. ¿Qué pasa? ¿es correcto?

Ciclo 4. Creando un nuevo deportista: el hablador [En lab03.doc, BodyTic.asta y *.java]

El objetivo de este punto es incluir el deportista hablador, considerando que:

(NO OLVIDE BDD - MDD)

- Está vestido con color rojo
 - Al ejercitarse alta (va subiendo los pies lentamente y luego los baja) y estando arriba se mueve a la derecha, y habla un poco.
 - Cuando descansa, habla mucho.
1. Implemente al deportista `Hablador`
 2. Ahora si, adicione ahora una pareja de deportistas habladores llámelos `han` y `leila`, Nuevamente ejecuten el programa y pídale a todos que entren, inicien, decidan y paren. Capturen pantallas significativas. Expliquen.

Ciclo 5. Nuevo deportista: Proponiendo y diseñando

El objetivo de este punto es permitir recibir en un nuevo tipo de deportista

(NO OLVIDE BDD - MDD)

1. Propongan, describan e implementen un nuevo tipo de deportista.
2. Incluyan una pareja de ellos con el nombre de ustedes. Ejecute el programa con dos casos significativos. Explique la intención de cada caso y capture las pantallas correspondientes.

Ciclo 6. Nuevo elemento: Proponiendo y diseñando

El objetivo de este punto es permitir recibir en un nuevo elemento (no deportista) en el salón.

(NO OLVIDE BDD - MDD)

1. Propongan, describan e implementen un nuevo tipo de elemento
2. Incluyan un par de ellos con el nombres semánticos. Ejecute el programa con dos casos significativos. Explique la intención de cada caso y capture las pantallas correspondientes.

Empaquetando la versión final para el usuario. [En lab03.doc, BodyTic.asta , *.java, BodyTic.jar]

1. Revise las opciones de `Bluej` para empaquetar su programa entregable en un archivo .jar. Genere el archivo correspondiente.
2. Consulte el comando `java` para ejecutar un archivo jar. Ejecutelo ¿qué pasa?
3. ¿Qué ventajas tiene esta forma de entregar los proyectos? Explique claramente.

DE BLUEJ A CONSOLA

En esta sección del laboratorio vamos a aprender a usar java desde consola. Para esto se va a trabajar con el proyecto del punto anterior.

Comandos básicos del sistema operativo [En lab03.doc]

Antes de iniciar debemos repasar los comandos básicos del manejo de la consola.

1. Investiguen los comandos para moverse en la estructura de directorios: crear, borrar, listar su contenido y copiar o eliminar un archivo.
2. Organicen un nuevo directorio con la estructura propuesta para probar desde allí su habilidad con los comandos de consola. Consulten y capturen el contenido de su directorio

```
BodyTic
  src
    aplicacion
    presentacion
    pruebas
```

3. En el directorio copien únicamente los archivos *.java del paquete de aplicación . Consulte y capture el contenido de src/aplicacion

Estructura de proyectos java [En lab03.doc]

En java los proyectos se estructuran considerando tres directorios básicos.

```
automata
  src
  bin
  docs
```

1. Investiguen los archivos que deben quedar en cada una de esas carpetas y la organización interna de cada una de ellas.
2. ¿Qué archivos debería copiar del proyecto original al directorio bin? ¿Por qué? Cópielos y consulte y capture el contenido del directorio que modificó.

Comandos de java [En lab03.doc]

1. Consulte para qué sirven cada uno de los siguientes comandos:

```
javac
java
javadoc
jar
```

2. Cree una sesión de consola y consulte en línea las opciones de los comandos java y javac. Capture las pantallas.
3. Busque la opción que sirve para conocer la versión a que corresponden estos dos comandos. Documente el resultado.

Compilando [En lab03.doc]

1. Utilizando el comando javac, **desde el directorio raiz (desde bodyTic con una sola instrucción)**, compile el proyecto. ¿Qué instrucción completa tuvo que dar a la consola para compilar TODO el proyecto? Tenga presente que se pide un único comando y que los archivos compilados deben quedar en los directorios respectivos.
2. Revise de nuevo el contenido del directorio de trabajo y sus subdirectorios. ¿Cuáles nuevos archivos aparecen ahora y dónde se ubican?

Documentando [En lab03.doc]

1. Utilizando el comando javadoc, desde el directorio raiz, genere la documentación (API) en formato html, en este directorio. ¿cuál es el comando completo para generar esta documentación?
2. ¿Cuál archivo hay que abrir para empezar a navegar por la documentación? Ábralo y capture la pantalla.

Ejecutando [En lab03.doc]

1. Empleando el comando `java`, desde el directorio raíz, ejecute el programa. ¿Cómo utilizó este comando?

Probando [En lab03.doc]

1. Adicione ahora los archivos del directorio pruebas y trate de compilar nuevamente el programa. Tenga en cuenta que estas clases requieren la librería junit 4.8. ¿Cómo se incluye un paquete para compilar? ¿Qué instrucción completa tuvo que dar a la consola para compilar?
2. Ejecute desde consola las pruebas . ¿Cómo utilizó este comando?. Puede ver ejemplos de cómo ejecutar el "test runner" en: <http://junit.sourceforge.net/doc/cookbook/cookbook.htm>
3. Pegue en su documento el resultado de las pruebas

Empaquetando [En lab03.doc]

1. Consulte como utilizar desde consola el comando `jar` para empaquetar su programa entregable en un archivo .jar, que contenga los archivos bytecode necesarios (no las fuentes ni las clases de prueba), y que se pueda ejecutar al instalarlo en cualquier directorio, con solo tener la máquina virtual de java y su entorno de ejecución (JRE). ¿Cómo empaquetó `jar` ?
2. ¿Cómo se ejecuta el proyecto empaquetado?

RETROSPECTIVA

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/Hombre)
2. ¿Cuál es el estado actual del laboratorio? ¿Por qué?
3. Considerando las prácticas XP del laboratorio. ¿cuál fue la más útil? ¿por qué?
4. ¿Cuál consideran fue el mayor logro? ¿Por qué?
5. ¿Cuál consideran que fue el mayor problema técnico? ¿Qué hicieron para resolverlo?
6. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados?