
	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación

Universidad Politécnica Salesiana

Vicerrectorado Docente

Código del Formato:	GUIA-PRL-001
Versión:	VF1.0
Elaborado por:	Directores de Área del Conocimiento Integrantes Consejo Académico
Fecha de elaboración:	2016/04/01
Revisado por:	Consejo Académico
Fecha de revisión:	2016/04/06
Aprobado por:	Lauro Fernando Pesántez Avilés Vicerrector Docente
Fecha de aprobación:	2016/14/06
Nivel de confidencialidad:	Interno

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Descripción General

Propósito


El propósito del presente documento es definir un estándar para elaborar documentación de guías de práctica de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana, con la finalidad de lograr una homogenización en la presentación de la información por parte del personal académico y técnico docente.

Alcance

El presente estándar será aplicado a toda la documentación referente a informes de prácticas de laboratorio, talleres o centros de simulación de las Carreras de la Universidad Politécnica Salesiana.

Formatos


- Formato de Guía de Práctica de Laboratorio / Talleres / Centros de Simulación – para Docentes
- Formato de Informe de Práctica de Laboratorio / Talleres / Centros de Simulación – para Estudiantes


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		


		FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES	
CARRERA: COMPUTACIÓN		ASIGNATURA: Programación Aplicada	
NRO. PRÁCTICA:	1	TÍTULO PRÁCTICA: Investigación de las versiones de Java	
OBJETIVO: Identificar los cambios importantes de Java Diseñar e Implementar las nuevas técnicas de programación Entender la cada uno de las características nuevas en Java			
INSTRUCCIONES (Detallar las instrucciones que se dará al estudiante):		1. Revisar los conceptos fundamentales de las nuevas versiones de Java	
		2. Establecer las características de Java	
		3. Implementar y diseñar los nuevos componentes de programación	
		4. Realizar el informe respectivo según los datos solicitados.	
ACTIVIDADES POR DESARROLLAR (Anotar las actividades que deberá seguir el estudiante para el cumplimiento de la práctica)			
1. Revisar la teoría y conceptos de Java 9 ,10, 11, 12			
2. Diseñar e implementar las características de Java,			
3. Probar su funcionamiento y rendimiento dentro de los equipos de computo			
4. Realizar práctica codificando los códigos de las nuevas características de Java.			
RESULTADO(S) OBTENIDO(S): Realizar procesos de investigación sobre los cambios importantes de Java Entender las aplicaciones de codificación de las nuevas características Entender las funcionalidades adicionales de Java.			
CONCLUSIONES: Aprenden a trabajar en grupo dentro de plazos de tiempo establecidos, manejando el lenguaje de programación de Java.			
RECOMENDACIONES: Realizar el trabajo dentro del tiempo establecido.			

Docente / Técnico Docente: _____

Firma: _____

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

		FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES	
CARRERA: Ingeniería en computación		ASIGNATURA: Programación Aplicada	
NRO. PRÁCTICA:		TÍTULO PRÁCTICA: Investigación de las versiones de Java	
OBJETIVO ALCANZADO: <ul style="list-style-type: none"> - Investigar sobre las características y funcionalidades de java. - Diseñar e Implementar las nuevas técnicas de programación - Entender la cada uno de las características nuevas en Java 			
ACTIVIDADES DESARROLLADAS			
1. Revisar la teoría y conceptos de Java 11. <p>Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos, que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Java es, a partir de 2012, uno de los lenguajes de programación más populares en uso, particularmente para aplicaciones de cliente-servidor de web, con unos diez millones de usuarios reportados.</p> <p>Su sintaxis deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. Las aplicaciones de Java son compiladas a bytecode (clase Java), que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.</p>			
2. Diseñar e implementar las características de Java. <ul style="list-style-type: none"> - Soporte en java <p>Java 11 es la primera versión de soporte extendido publicada o LTS bajo el nuevo ciclo de publicaciones que adoptó Java en la versión 9. Añade varias novedades importantes en cuanto a seguridad y elimina otras que en versiones anteriores ya fueron marcadas como desaconsejadas.</p> - Eliminación de javaFx <p>Uno de los cambios más importantes que se ha implementado en java 11 es que javaFx ha sido eliminado para ser convertido en un módulo independiente.</p> <p>Esto viene con muchas novedades:</p> <ul style="list-style-type: none"> ○ Se han agregado nuevas características como: 			

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

➤ Ejecutar archivos Java directamente sin javac

- Este cambio lo que nos permite es ejecutar un file java directamente sin necesidad de compilarlo previamente con javac.
- Ahora se puede correr directamente tu programa, pero esto implica una compilación previa.

➤ Variables Locale para los parámetros Lambda

- En Java 10 se introdujo el uso de **var** con inferencia del tipo. Ahora podemos usar esto en la declaración de lambdas.

➤ Novedades que permiten el desempeño y la seguridad


En java se han agregado algunas novedades que abarcan posibilidades que permiten mejorar el desempeño y la seguridad como, por ejemplo:

- Los controles de acceso basados en Nest (nestmates) se encargan de revisitar la implementación de las clases internas y elimina la necesidad de insertar métodos puente por parte de los compiladores.
- Las constantes dinámicas de archivo de clase reducen el coste y la disrupción de la creación de nuevas formas de constantes de ficheros de clase materializables y abren la puerta a nuevos enfoques centrados en la plataforma y el rendimiento.
- Soporte experimental de ZGC, que es el nuevo recolector de basura diseñado para tiempos de pausa inferiores a 10 milisegundos y tiene como objetivo que la penalización no supere el 15% del rendimiento.
- Flight Recorder es un framework de recolección de datos de bajos recursos para resolver problemas detectados en las aplicaciones de Java y HotSpot JVM.
- La nueva biblioteca estándar HTTP se encarga de estandarizar la API incubada y habilita el soporte para permitir flujos basados en HTTP/1.1 y HTTP/2.
- Implementación de TLS 1.3.
- Sintaxis de variable local para los Parámetros Lambda, actualizando así la sintaxis Lambda para usar la inferencia de tipo de variable introducida en Java 10.
- En GNU/Linux, a partir de ahora se cargarán por defecto las bibliotecas de GTK3 en lugar de las de GTK2. Un movimiento hecho para adaptarse a los entornos de escritorios basados en GTK más modernos en GNU/Linux, posiblemente con GNOME en mente.
- Se han eliminado características como Web Start, las Applets (que ya fueron marcadas como obsoletas y cuyas funciones están ya ampliamente cubiertas por HTML, CSS y JavaScript) y el mencionado módulo JavaFX.

➤ Nuevo método de java String

Java ha agregado nuevos métodos para los String los cuales nos permiten hacer una variedad de cosas. Estos métodos son muy fáciles de usar y algunos de estos sustituyen a algunos métodos antiguos ya que les superan en funcionalidad.

Ejemplo:

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

- `isBlank()` Este nuevo método retorna un valor booleano según si el string contiene solo espacios en blanco y si está vacío.
- `lines()` Este metodo devolvera un array de strings 'spliteando' el texto por lineas.
- `strip()`, `stripLeading()`, `stripTrailing()` Remueve los caracteres en blanco. A diferencia del viejo `trim()`, este nuevo método `strip` si tiene en cuenta los caracteres Unicode.
- `repeat(int)` Repite el string las veces que se le indique.

➤ **Lectura y escritura de Strings en java**

En java 11 se agregaron métodos que simplifican la lectura y escritura de archivos esto hace que sea mucho más prácticos y sencillo de usar ya que antes para nosotros poder escribir y leer archivos se nos hacía mucho mas complicado por la cantidad de métodos que se debían utilizar.

- `readString(Path)`
- `readString(Path, Charset)`
- `writeString(Path, CharSequence, OpenOption...)`
- `writeString(Path, CharSequence, Charset, OpenOption...)`

➤ **Eliminación de paquetes Java EE y Corba**

Se han eliminado algunos paquetes incluidos en el JDK como:

- `Java.xml.bind`
- `Java.corba`
- `Java.activation`

En caso de que se necesiten estos paquetes solo hay que incluirlos como dependencia externa.

➤ **Eliminación de código no deseable**

A partir de Java 11 Oracle se ha vuelto más decidido en eliminar características que no eran recomendadas. Habitualmente las despreciaba pero ahora se ha decidido directamente en eliminarlas. Por ejemplo ha eliminado `Thread.destroy()` y `Thread.stop(Throwable)` . Así que ahora hay que estar más atentos a la compatibilidad con versiones previas al actualizar Java y a los ajustes que debemos implementar en nuestro código.

- 3. Probar su funcionamiento y rendimiento dentro de los equipos de cómputo.**
- 4. Realizar práctica codificando los códigos de las nuevas características de Java.**

➤ **Métodos de java String.**

```
1 public class HelloWorld {  
2     public static void main(String[] args) {  
3         System.out.println("Hello, World");  
4     }  
5 }
```

```
1 $ java HelloWorld.java  
2 Hello, World
```

➤ **Ejecutar archivos Java directamente sin javac**

```
System.out.println("isBlanck() example.");
```

Resultado por consola:

```
public void estaVacio(String texto) {  
  
    System.out.println(texto.isBlank());  
  
}
```

```
isBlanck() example.  
true  
false  
true
```

```
System.out.println("\n Strip example");
```

Resultados por consola:

```
public void removerEspacios(String texto) {  
  
    System.out.println(texto.strip());  
    System.out.println(texto.stripLeading());  
    System.out.println(texto.stripTrailing());  
    System.out.println(texto.trim());  
  
}
```

```
run:  
Hola como estas  
Hola como estas  
    Hola como estas  
Hola como estas
```

```
public void convertirListas(String texto) {
```

Resultados por consola

```
    System.out.println(texto.lines().collect(Collectors.toList()));
```

```
run:  
[1, 2, 3, 4, 5, 6]
```

```
}
```


```
public void repetir(String texto, int numeroVeces) {
```

Resultados por consola

```
    System.out.println(texto.repeat(numeroVeces));
```

```
run:  
Hola Hola Hola
```

```
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

➤ Lectura y escritura de Strings en archivos

```

public class ArchivosLecturaEscritura {

    public void leer() {

        try {

            String contenido = Files.readString(Path.of("prueba.txt"));
            System.out.println(contenido);

        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    public void escribir(String parametro) {

        try {

Path path = Files.writeString(Path.of("prueba.txt"), parametro, StandardOpenOption.APPEND);
            System.out.println(path);
            String s = Files.readString(path);
            System.out.println(s);

        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
}

class principal {

    public static void main(String[] args) {

        ArchivosLecturaEscritura archivo = new ArchivosLecturaEscritura();

        archivo.escribir(" Hola Mundo ");

    }
}

```

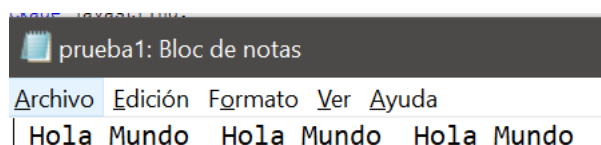
Resultados por consola:


```

prueba1.txt
Hola Mundo  Hola Mundo  Hola Mundo

```

Resultado en Bloc de notas



	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

➤ **Programa con el uso de var en lambda**

```
public class Lambda {

    public void probar(String seleccion) {

        switch (seleccion) {

            case "PAR":

                System.out.println("PAR");

                IntStream.of(1, 2, 3, 4, 5, 6, 7).filter((var i) -> i % 2 == 0).forEach(System.out::println);
                break;

            case "IMPAR":

                System.out.println("IMPAR");

                IntStream.of(1, 2, 3, 4, 5, 6, 7).filter((var i) -> i % 2 != 0).forEach(System.out::println);
                break;

            case "SUMAR":

                System.out.println("SUMA");

                IOperacion operacion = (var x, var y) -> (x + y);
                var resultado = operacion.calcular(2, 2);
                System.out.println(resultado);

                break;


        }
    }
}
```

- **Clase principal**

```
class prueba {

    public static void main(String[] args) {

        Lambda lambda = new Lambda();
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

```
lambda.probar("SUMAR");
lambda.probar("RESTAR");
lambda.probar("PAR");
lambda.probar("IMPAR");
```

```
}
```

```
}
```

- **Clase interface**

```
interface IOperacion {
```

```
    double calcular(double num1, double num2);
```


```
}
```

Resultados obtenidos por consola:

```
run:
SUMA
4.0
PAR
2
4
6
IMPAR
1
3
5
7
```

RESULTADO(S) OBTENIDO(S):

Realizar procesos de investigación sobre los cambios importantes de Java
Entender las aplicaciones de codificación de las nuevas características
Entender las funcionalidades adicionales de Java.

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

CONCLUSIONES:

Como conclusión decimos que aprender los nuevos funcionamientos y características de java 11 nos ayudan a tener una base mejor formada de cómo implementar estos métodos ya en la práctica. Creando así programas con los cuales nos empezamos a familiarizar con dichos métodos.

RECOMENDACIONES

Recomiendo usar las nuevas características de java 11 ya que nos facilita el uso de var en lambdas, además de una variedad de métodos para los Strings como `isBlank()`, `strip()`, `lines()`, y `repeat()`. Además que se agregaron métodos para lectura y escritura de archivos como `readString()`, y `writeString()`. Los cuales nos facilitan mucho a la hora de programar.

Referencias

Gustavo. (22 de Abril de 2019). *Experto*. Obtenido de <https://experto.dev/lo-nuevo-en-java-11/>
Medina, E. (26 de Septiembre de 2018). *muyLinux*. Obtenido de <https://www.muylinux.com/2018/09/26/java-11/>
picodotdev. (28 de 09 de 2018). *Blog Bitix*. Obtenido de <https://picodotdev.github.io/blog-bitix/2018/09/novedades-y-nuevas-caracteristicas-de-java-11/>

Nombre de estudiante: Jorge Santiago Cabrera Arias.

Firma de estudiante:



CS Escaneado con CamScanner