
	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: Patrones en Java																					
OBJETIVO: Identificar los cambios importantes de Java Diseñar e Implementar las nuevas tecnicas de programación Entender los patrones de Java																							
INSTRUCCIONES (Detallar las instrucciones que se dará al estudiante):		1. Revisar los conceptos fundamentales de Java																					
		2. Establecer las características de Java basados en patrones de diseño																					
		3. Implementar y diseñar los nuevos patrones de Java																					
		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 Patrones de Diseño de Java																							
2. Diseñar e implementa cada estudiante un patron de diseño y verificar su funcionamiento. A continuación se detalla el patron a implementar:																							
<table border="1"> <thead> <tr> <th>Nombre</th> <th>Patron</th> </tr> </thead> <tbody> <tr> <td><u>NIXON ANDRES ALVARADO CALLE</u></td> <td>Factory Method</td> </tr> <tr> <td><u>ROMEL ANGEL AVILA FAICAN</u></td> <td>Builder</td> </tr> <tr> <td><u>JORGE SANTIAGO CABRERA ARIAS</u></td> <td>Abstract Factory</td> </tr> <tr> <td><u>EDITH ANAHI CABRERA BERMEO</u></td> <td>Prototype</td> </tr> <tr> <td><u>JUAN JOSE CORDOVA CALLE</u></td> <td>Chain of Responsibility</td> </tr> <tr> <td><u>DENYS ADRIAN DUTAN SANCHEZ</u></td> <td>Command</td> </tr> <tr> <td><u>JOHN XAVIER FAREZ VILLA</u></td> <td>Interpreter</td> </tr> <tr> <td><u>PAUL ALEXANDER GUAPUCAL CARDENAS</u></td> <td>Iterator</td> </tr> <tr> <td><u>PAUL SEBASTIAN IDROVO BERREZUETA</u></td> <td>Mediator</td> </tr> </tbody> </table>				Nombre	Patron	<u>NIXON ANDRES ALVARADO CALLE</u>	Factory Method	<u>ROMEL ANGEL AVILA FAICAN</u>	Builder	<u>JORGE SANTIAGO CABRERA ARIAS</u>	Abstract Factory	<u>EDITH ANAHI CABRERA BERMEO</u>	Prototype	<u>JUAN JOSE CORDOVA CALLE</u>	Chain of Responsibility	<u>DENYS ADRIAN DUTAN SANCHEZ</u>	Command	<u>JOHN XAVIER FAREZ VILLA</u>	Interpreter	<u>PAUL ALEXANDER GUAPUCAL CARDENAS</u>	Iterator	<u>PAUL SEBASTIAN IDROVO BERREZUETA</u>	Mediator
Nombre	Patron																						
<u>NIXON ANDRES ALVARADO CALLE</u>	Factory Method																						
<u>ROMEL ANGEL AVILA FAICAN</u>	Builder																						
<u>JORGE SANTIAGO CABRERA ARIAS</u>	Abstract Factory																						
<u>EDITH ANAHI CABRERA BERMEO</u>	Prototype																						
<u>JUAN JOSE CORDOVA CALLE</u>	Chain of Responsibility																						
<u>DENYS ADRIAN DUTAN SANCHEZ</u>	Command																						
<u>JOHN XAVIER FAREZ VILLA</u>	Interpreter																						
<u>PAUL ALEXANDER GUAPUCAL CARDENAS</u>	Iterator																						
<u>PAUL SEBASTIAN IDROVO BERREZUETA</u>	Mediator																						


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


	<u>ADOLFO SEBASTIAN JARA GAVILANES</u>	Observer
	<u>ADRIAN BERNARDO LOPEZ ARIZAGA</u>	State
	<u>ESTEBAN DANIEL LOPEZ GOMEZ</u>	Strategy
	<u>GEOVANNY NICOLAS ORELLANA JARAMILLO</u>	Visitor
	<u>NELSON PAUL ORTEGA SEGARRA</u>	Adapter
	<u>BRYAM EDUARDO PARRA ZAMBRANO</u>	Bridge
	<u>LISSETH CAROLINA REINOSO BAJAÑA</u>	Composite
	<u>MARTIN SEBASTIAN TOLEDO TORRES</u>	Decorator
	<u>SEBASTIAN ROBERTO UYAGUARI RAMON</u>	Flyweight
	<u>ARIEL RENATO VAZQUEZ CALLE</u>	Proxy
	CHRISTIAN ABEL JAPON CHAVEZ	Facade
3. Probar y modificar el patron de diseño a fin de generar cuales son las ventajas y desventajas.		
4. Realizar práctica codificando los codigos de los patrones y su estructura.		
RESULTADO(S) OBTENIDO(S): Realizar procesos de investigación sobre los patrones de diseño de Java Entender los patrones y su utilización dentro de aplicaciones Java. Entender las funcionalidades basadas en patrones.		
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. Revisar el siguiente link: https://refactoring.guru/es/design-patterns/java		

Docente / Técnico Docente: Ing: Diego Quisi

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

Firma: _____

		FORMATO DE INFORME DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA ESTUDIANTES	
CARRERA: Computación		ASIGNATURA: Programación Aplicada	
NRO. PRÁCTICA:		TÍTULO PRÁCTICA: Patrones de Diseño Java	
OBJETIVO ALCANZADO: Identificar los cambios importantes de Java Diseñar e Implementar las nuevas técnicas de programación Entender los patrones de Java			
ACTIVIDADES DESARROLLADAS			
<p>1. Revisar la teoría y conceptos de Patrones de Diseño de Java</p> <p>Patrón de Diseño Prototype:</p> <p>Es un patrón de diseño creacional que nos permite copiar objetos existentes sin que el código dependa de sus clases.</p> <p>Existen dos formas de clonar objetos:</p> <p>Copia Superficial: El objeto clonado tendrá los mismos valores que el original, guardando también referencias a otros objetos que contenga (por lo que si son modificados desde el objeto original o desde alguno de sus clones el cambio afectará a todos ellos).</p> <p>Copia Profunda: El objeto clonado tendrá los mismos valores que el original así como copias de los objetos que contenga el original (por lo que si son modificados por cualquiera de ellos, el resto no se verán afectados).</p> <p>El patrón declara una interfaz común para todos los objetos que soportan la clonación. Esta interfaz nos permite clonar un objeto sin acoplar el código a la clase de ese objeto. Dicha interfaz contiene un único método clonar.</p> <p>El método crea un objeto a partir de la clase actual y lleva todos los valores de campo del viejo objeto, al nuevo. Se puede incluso copiar campos privados, porque la mayoría de los lenguajes de programación permite a los objetos acceder a campos privados de otros objetos que pertenecen a la misma clase.</p>			
<p>2. Diseñar e implementa cada estudiante un patrón de diseño y verificar su funcionamiento</p> <p>Ejemplo en Java</p> <p>ec.edu.ups.modelo Clase Figura</p> <pre>package ec.edu.ups.modelo; import java.util.Objects; /** * * @author Anahi */ public abstract class Figura {</pre>			

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

```

public int x;
public int y;
public String color;

public Figura() {
}

public Figura(Figura f) {
    if (f != null) {
        this.x = f.x;
        this.y = f.y;
        this.color = f.color;
    }
}

public abstract Figura clonar();

@Override
public boolean equals(Object o) {
    if (!(o instanceof Figura)) return false;
    Figura figura2 = (Figura) o;
    return figura2.x == x && figura2.y == y && Objects.equals(figura2.color,
color);
}
}

```

Clase Círculo

```

package ec.edu.ups.modelo;
/**
 *
 * @author Anahi
 */
public class Circulo extends Figura{
    public int radio;


    public Circulo() {
    }

    public Circulo(Circulo c) {
        super(c);
        if (c != null) {
            this.radio = c.radio;
        }
    }

    @Override
    public Figura clonar() {
        return new Circulo(this);
    }

    @Override
    public boolean equals(Object o) {
        if (!(o instanceof Circulo) || !super.equals(o)) return false;
        Circulo figura2 = (Circulo) o;

```

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

```

        return figura2.radio == radio;
    }

}

Clase Rectángulo
package ec.edu.ups.modelo;

/**
 *
 * @author Anahi
 */
public class Rectangulo extends Figura {
    public int ancho;
    public int alto;

    public Rectangulo() {
    }

    public Rectangulo(Rectangulo r) {
        super(r);
        if (r != null) {
            this.ancho = r.ancho;
            this.alto = r.alto;
        }
    }

    @Override
    public Figura clonar() {
        return new Rectangulo(this);
    }

    @Override
    public boolean equals(Object o) {
        if (!(o instanceof Rectangulo) || !super.equals(o)) return false;
        Rectangulo figura2 = (Rectangulo) o;
        return figura2.ancho == ancho && figura2.alto == alto;
    }
}

ec.edu.ups.vista
Clase main Prueba
package ec.edu.ups.modelo.vista;
import ec.edu.ups.modelo.*;
import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author Anahi
 */
public class Prueba {
    public static void main(String[] args) {

```

```
List<Figura> figuras = new ArrayList<>();
List<Figura> figurasClonadas = new ArrayList<>();

Circulo c = new Circulo();
c.x = 10;
c.y = 20;
c.radio = 15;
c.color = "azul";
figuras.add(c);

Circulo c1 = (Circulo) c.clonar();
figuras.add(c1);
System.out.println(c1.x);
Rectangulo r = new Rectangulo();
r.ancho = 10;
r.alto = 20;
r.color = "amarillo";
figuras.add(r);

clonarYComparar(figuras, figurasClonadas);
}

private static void clonarYComparar(List<Figura> figuras, List<Figura>
figurasClonadas) {
    for (Figura figura : figuras) {
        figurasClonadas.add(figura.clonar());
    }
    for (int i = 0; i < figuras.size(); i++) {
        if (figuras.get(i) != figurasClonadas.get(i)) {
            System.out.println(i + ":Las figuras son diferentes");
            if (figuras.get(i).equals(figurasClonadas.get(i))) {
                System.out.println(i + ":y son idénticas (Bien Hecho!)");
            } else {
                System.out.println(i + ":Pero no son idénticas (Error)");
            }
        } else {
            System.out.println(i + ": Las figuras son las mismas (Error)");
        }
    }
}
}
```

3. Ventajas y Desventajas


Ventajas:

- ❖ Permite tener una copia de un objeto en ejecución.
- ❖ Permite crear copias tanto superficiales como a fondo.
- ❖ Los prototipos pueden servir como alternativa a las subclases

Desventajas:

- ❖ Es difícil implementar la clonación en una jerarquía de objetos.

Bibliografía:

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

- *Prototype*. (s. f.). Refactor GURU. Recuperado 18 de noviembre de 2020, de <https://refactoring.guru/es/design-patterns/prototype>
- *Patrón de diseño Prototype (creación)*. (s. f.). InformaticaPC.com. Recuperado 18 de noviembre de 2020, de <https://informaticapc.com/patrones-de-diseno/prototype.php>

RESULTADO(S) OBTENIDO(S):

Entender los patrones y su utilización dentro de aplicaciones Java.
Entender las funcionalidades basadas en patrones.

CONCLUSIONES:

El Prototype como patrón de diseño es fácil de usar y nos permite optimizar procesos en el código de programación.

RECOMENDACIONES:

Utilizar fuentes confiables para la investigación.
Utilizar buenas prácticas de Programación.

Nombre de estudiante: Edith Anahí Cabrera Bermeo