#### 

#### **CIBERTEC**

#### **Dirección de Extensión Profesional**

#### **DIVISIÓN DE ALTA TECNOLOGÍA (DAT)**

**Programa :** Java AdvancedApplicationDeveloper

**Curso :** Aspectos

**Duración :** 40 minutos

**Laboratorio Nº 3.1**

**Registra métricas en GCH (tiempos de ejecución)**

Inicie con el uso de AOP en la aplicación GCH en base al framework Spring AOP.

* Agregue el framework Spring AOP en la aplicación.
* Implemente el aspecto de logging en GCH.
* Configure la ejecución del aspecto en distintos point-cuts.

#### Para utilizar Spring AOP

applicationContext.xml

Configurar los espacios de nombres

Agregar el espacio de nombre:

xmlns:aop="http://www.springframework.org/schema/aop"

Agregar la referencia del esquema para poder auto completar

http://www.springframework.org/schema/aop/spring-aop-3.0.xsd

Configurar el AspectJ auto proxy

<aop:aspectj-autoproxy />

Crear los aspectos

Todos los aspectos son Java Classes

Hay dos tipos de aspectos:

1) Aspecto calificador

Define @Pointcut en todos sus métodos

Da un nivel de orden a la aplicacion y permite reutilizar los pointcuts como "accesos directos" a un conjunto de métodos

@Aspect

public class LogAspect {

@Pointcut("execution(\* \*(..))")

public void trace() {

}

}

2) Aspecto ejecutor

Define Advices o Bindings (@Before, @After, @Around) par aindicar en que momento se ejecuta el código

@Aspect

public class LogAspectImpl {

@Before("pe.edu.cibertec.javaadv.mod4.asistencias.aspect.LogAspect.trace()")

public void log() {

System.out.println("Hola desde los aspectos :3");

}

}

En Spring todo aspecto es un Spring Bean y tiene que estar declarado en applicationContext.xml

<bean

id="logAspect"

class="pe.edu.cibertec.javaadv.mod4.asistencias.aspect.LogAspect">

</bean>

<bean

id="logAspectImpl"

class="pe.edu.cibertec.javaadv.mod4.asistencias.aspect.LogAspectImpl">

</bean>