



Programación Orientada a Aspectos

- Diego Alejandro Gutierrez Rojas
- Juan Camilo Pineda Vargas
- Cristian David Tafur Devia
- Christian Camilo Rojas Tapias





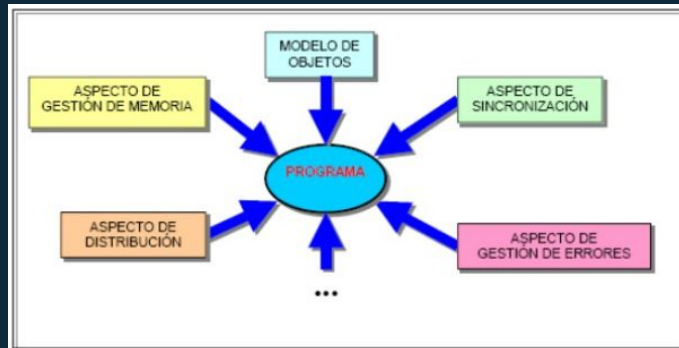
Contenido:

1. **Introducción**
2. **Historia**
3. **Filosofía**
4. **Conceptos clave**
5. **Ventajas y desventajas**
6. **Lenguajes de Programación**
7. **Aplicaciones**
8. **Ejemplos**
9. **Referencias**

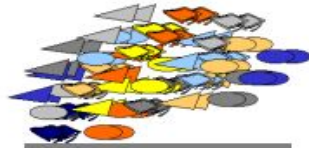
Introducción

1

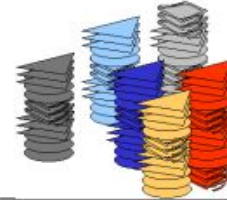
- El POA es un paradigma de programación que tiene como objetivo aumentar la modularidad al permitir la separación de preocupaciones transversales
- El paradigma de orientación a aspectos surgió de necesidades muy concretas en la programación.



Evolución de los sistemas de software:



1ª Generación:
Código espagueti



2ª y 3ª Generación:
Descomposición funcional

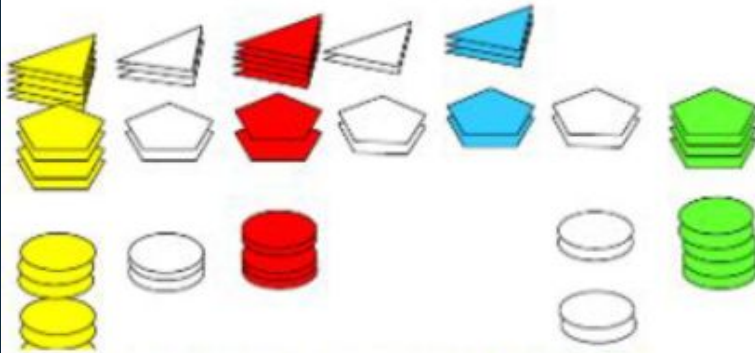


4ª Generación:
Descomposición en objetos

Software = Datos (formas)
+ Funciones (colores)



5° generación



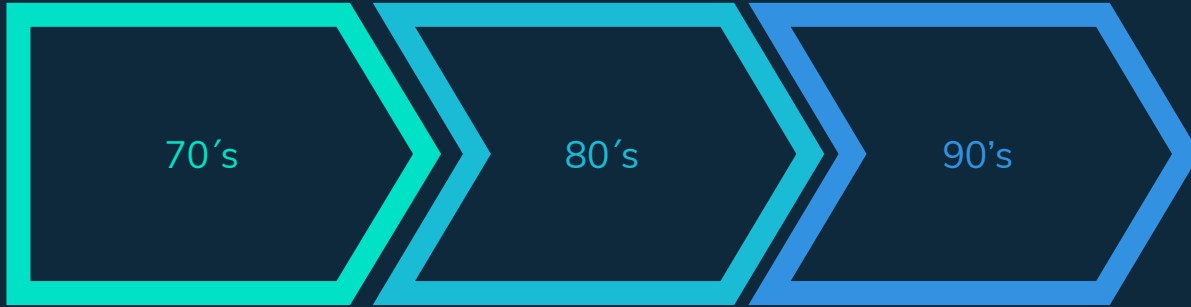
Descomposición en Aspectos

Software= Datos(formas) +
Funciones(colores)



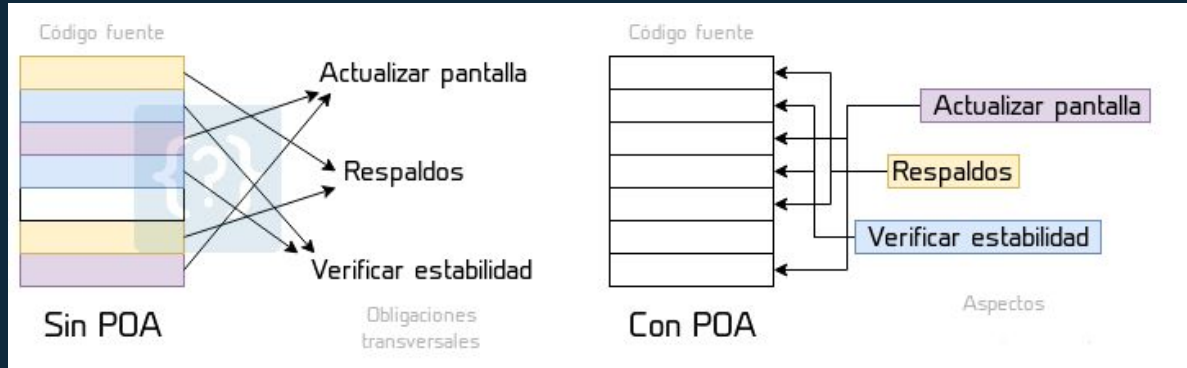
2

Historia



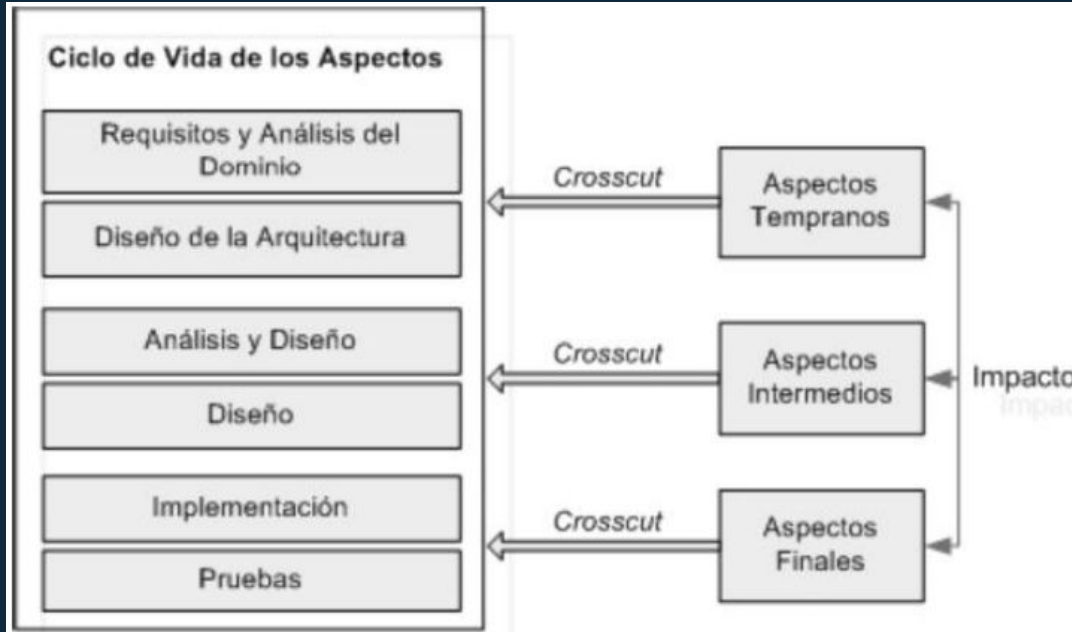
3

Filosofía



POO vs POA

Ciclo de vida





4

Conceptos Clave



Aspecto (Aspect)

- ◇ Funcionalidad transversal que será implementada.
- ◇ Representa la sección de código que se separará del programa.





Consejo (Advice)

◇ Before

```
@Aspect
public class EjemploBefore {

    @Before("execution(public * get*())")
    public void controlaPermisos() {
        // ...
    }
}
```





Consejo (Advice)

◇ AfterReturning

```
@Aspect
public class EjemploAfterReturning {

    @AfterReturning("execution(public * get*())")
    public void log() {
        // ...
    }
}
```





Consejo (Advice)

◇ AfterThrowing

```
@Aspect
public class EjemploAfterThrowing {

    @AfterThrowing(
        pointcut="execution(public * get*())",
        throwing="daoException")
    public void logException(DAOException daoException) {
        // ...
    }
}
```





Consejo (Advice)

◇ Around

```
@Aspect
public class EjemploAround {

    @Around("execution(public * get*())")
    public Object ejemploAround(ProceedingJoinPoint pjp) throws Throwable {
        System.out.println("ANTES");
        Object valorRetorno = pjp.proceed();
        System.out.println("DESPUES");
        return valorRetorno;
    }
}
```



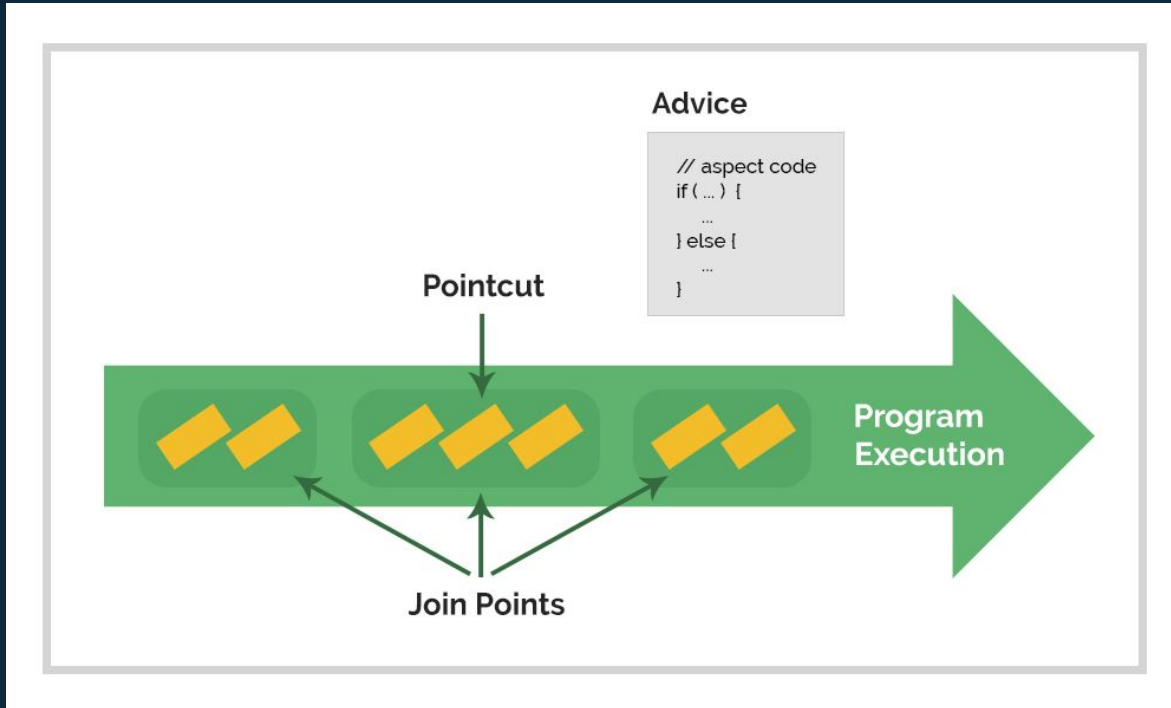


Punto de union (join-point)

- ◇ Puntos definidos en el flujo de un programa.
- ◇ Ejemplo:
 - Llamadas a un método.
 - Ejecución de constructores.
 - Manejo de excepciones.



Punto de corte (pointcut)





Tejedores (Weaver)



Entrelazado Estático

Se realiza en tiempo de compilación.

- ◇ Mezcla los componentes con los aspectos.
- ◇ Usa los join point para realizar la inserción de las sentencias.

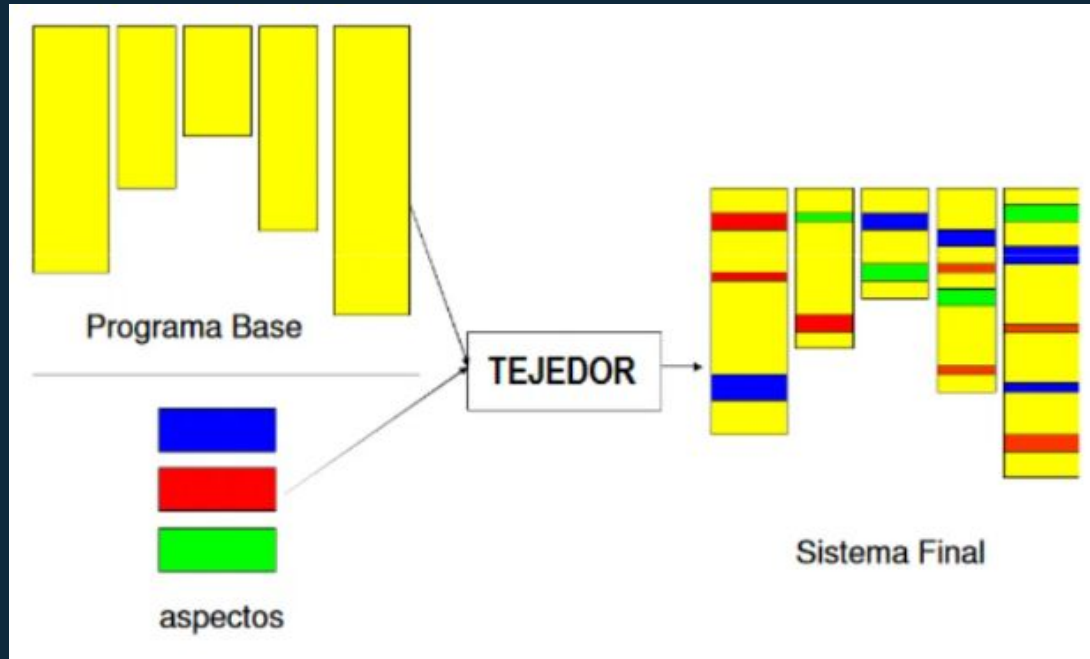
Entrelazado Dinámico

Se realiza en tiempo de ejecución.

- ◇ Yuxtaposición.
- ◇ Mezcla.
- ◇ Fusión.



Tejedores (Weaver)





5

Ventajas y Desventajas De la Programación Orientada a Aspectos

Ventajas

Se solucionan los problemas causados por Código mezclado y código diseminado

1

Mayor Evolucionabilidad

3

Resuelve el dilema del arquitecto

5

2

Implementación Modularizada

4

Capacidad de retrasar las decisiones de diseño

6

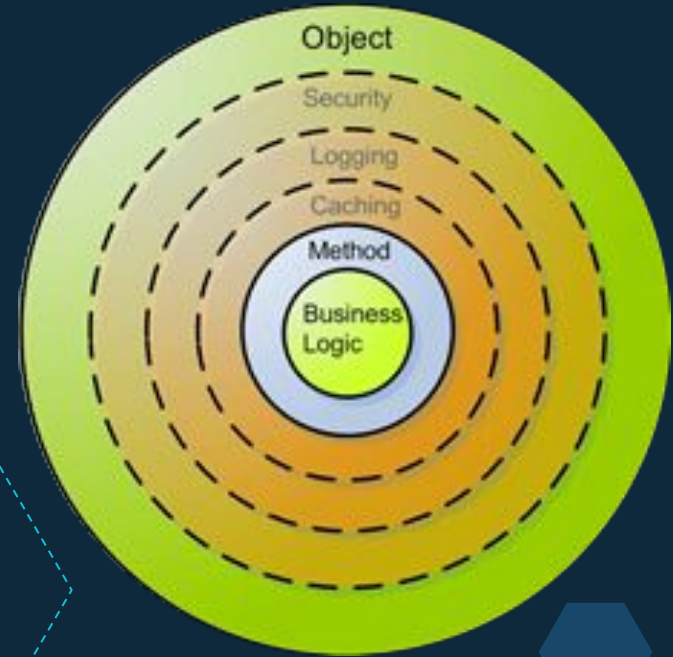
Mayor Reusabilidad

Ventajas

N-Dimensiones

Divide y
Vencerás

Mínimo
acoplamiento y
máxima
cohesión



Desventajas

1. Posibles choques entre el código funcional.
2. Posibles choques entre los aspectos.
3. Problemas propios del desarrollo.
4. Posibles choques código de aspectos y mecanismos del lenguaje.
5. Documentación,



6

Lenguajes de Programación



Lenguajes de Programación

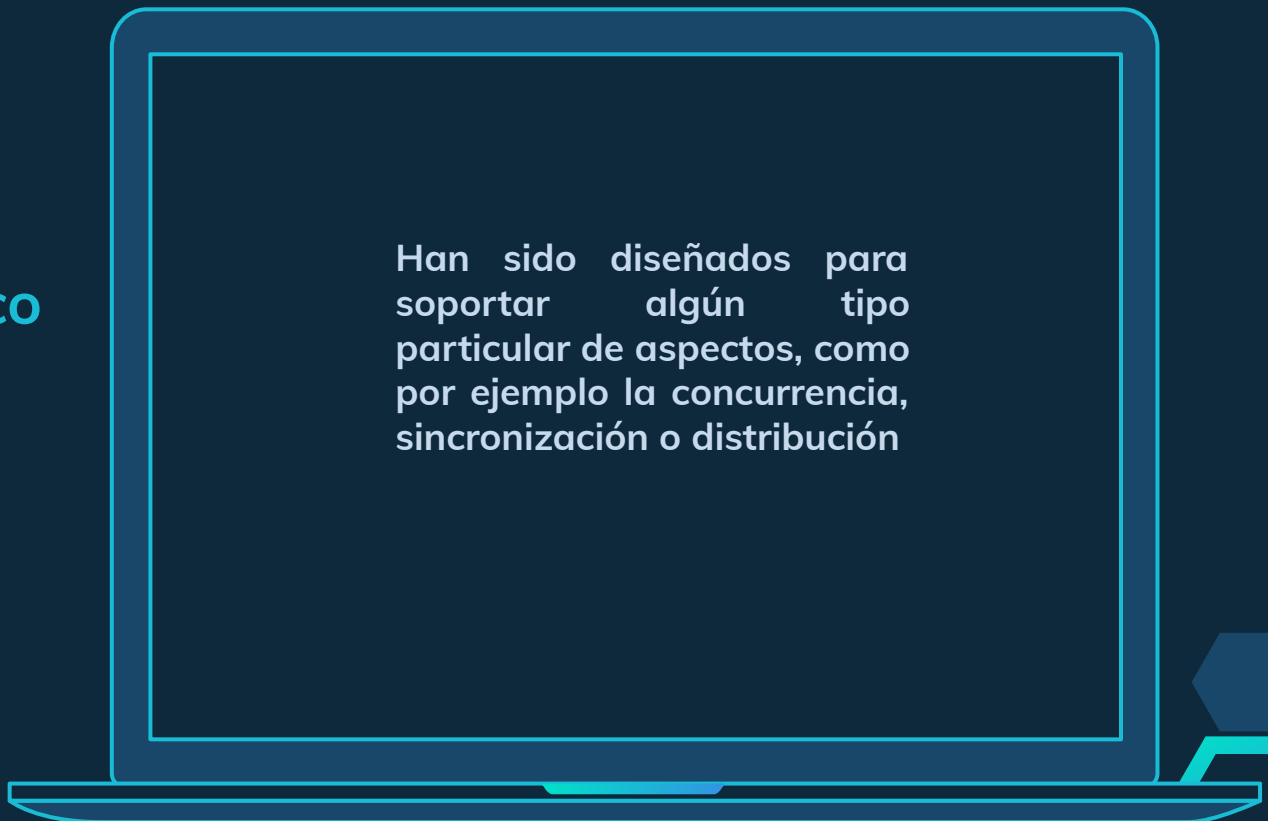
Son los que permiten separar la definición de la funcionalidad “principal” de la definición de los diferentes aspectos:

- Debe ser claramente identificable.
- Debe auto contenerse.
- Debe ser fácilmente modificable.
- No deben interferir entre ellos





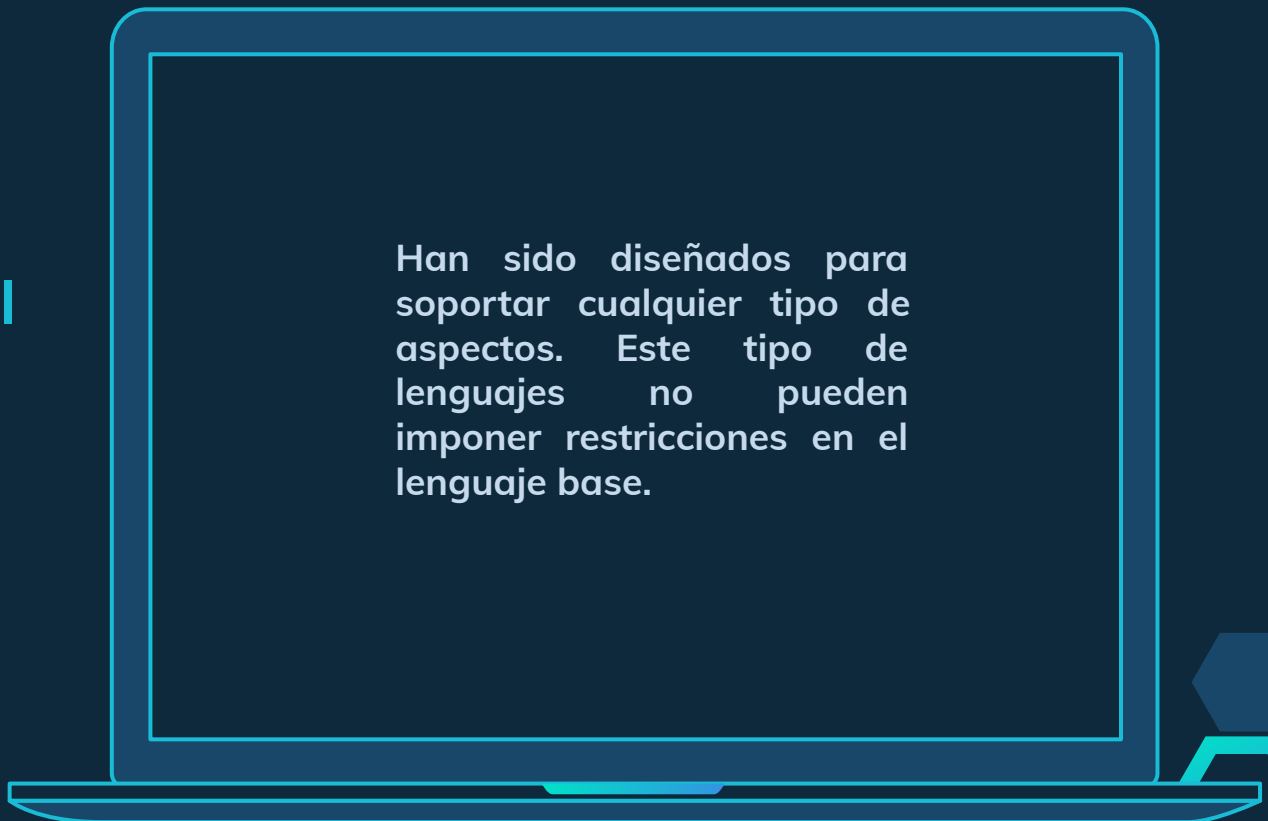
Lenguajes de Dominio Específico



Han sido diseñados para soportar algún tipo particular de aspectos, como por ejemplo la concurrencia, sincronización o distribución



Lenguajes de Propósito General



Han sido diseñados para soportar cualquier tipo de aspectos. Este tipo de lenguajes no pueden imponer restricciones en el lenguaje base.

Lenguajes de Programación

JPAL

JPAL

RIDL

RIDL

Phyton



COOL

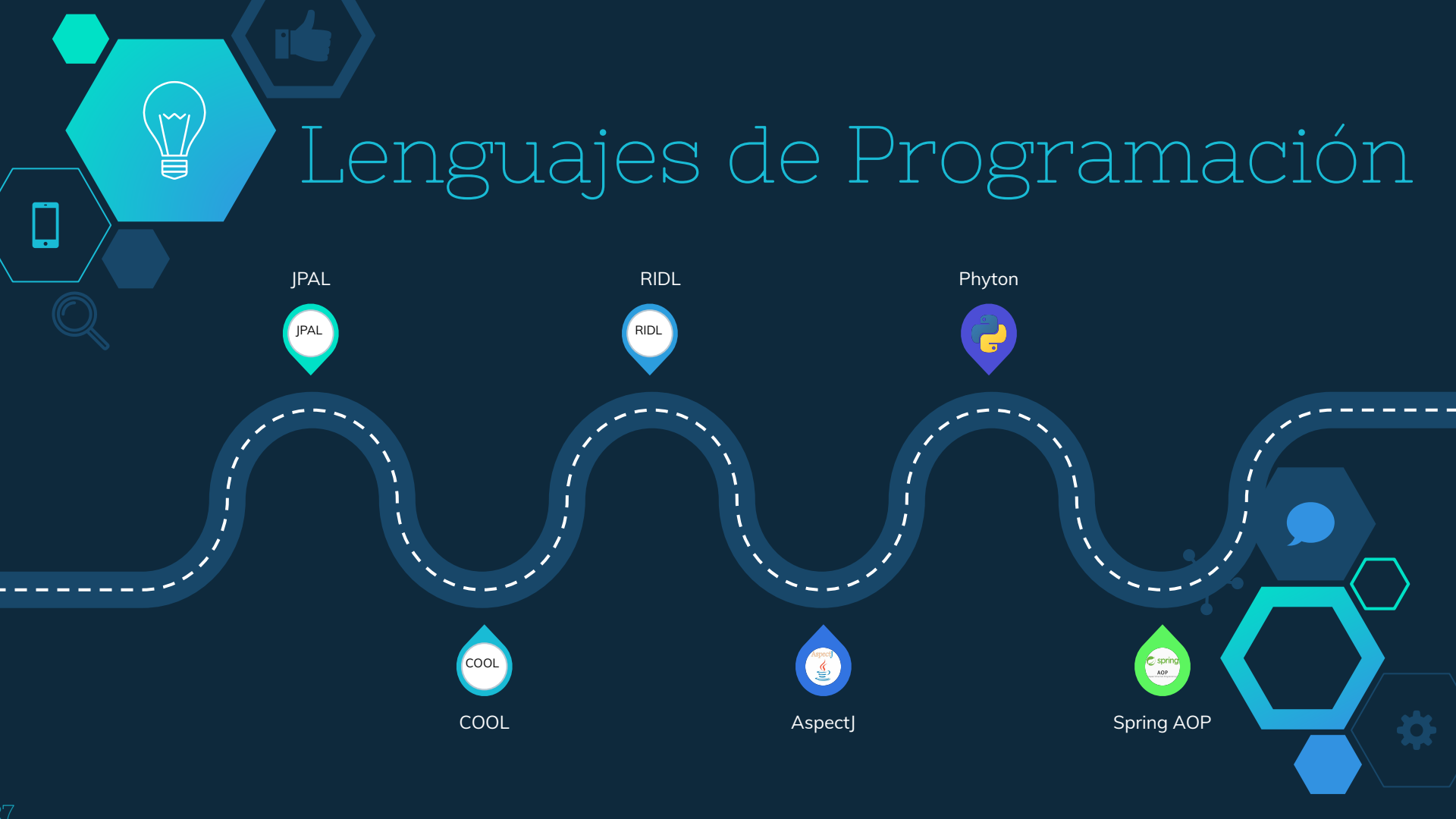
COOL

AspectJ

AspectJ

spring AOP

Spring AOP





Aplicaciones:

Manejo de memoria

- Optimización de recursos de memoria. No habrá código duplicado
- Procesos más eficientes

1

2

Tracing

- Mejor entendimiento del flujo del programa y progresión de datos

3

4

- Uso transversal de datos

Transacciones

- Optimiza el control de errores

Manejo de errores



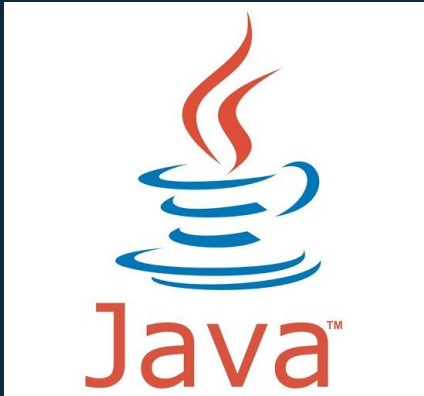
Aplicaciones:



Proceso de compilación

Javac

Java compiler

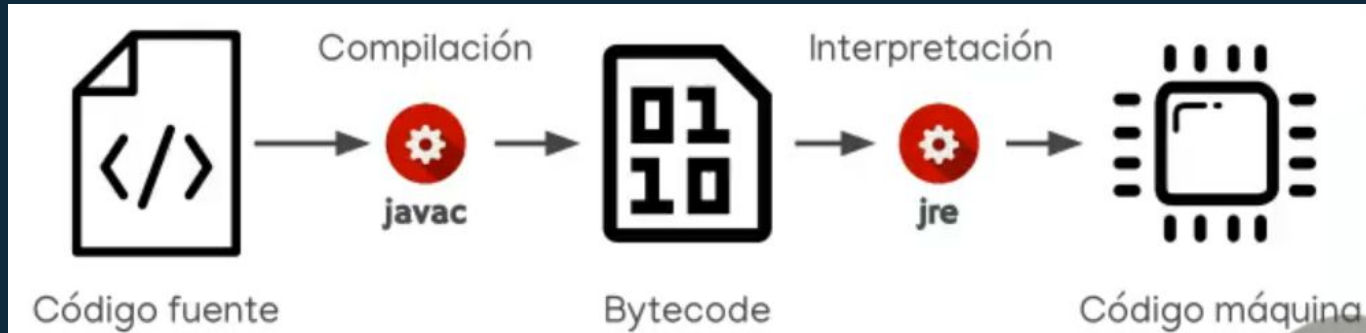


AJC

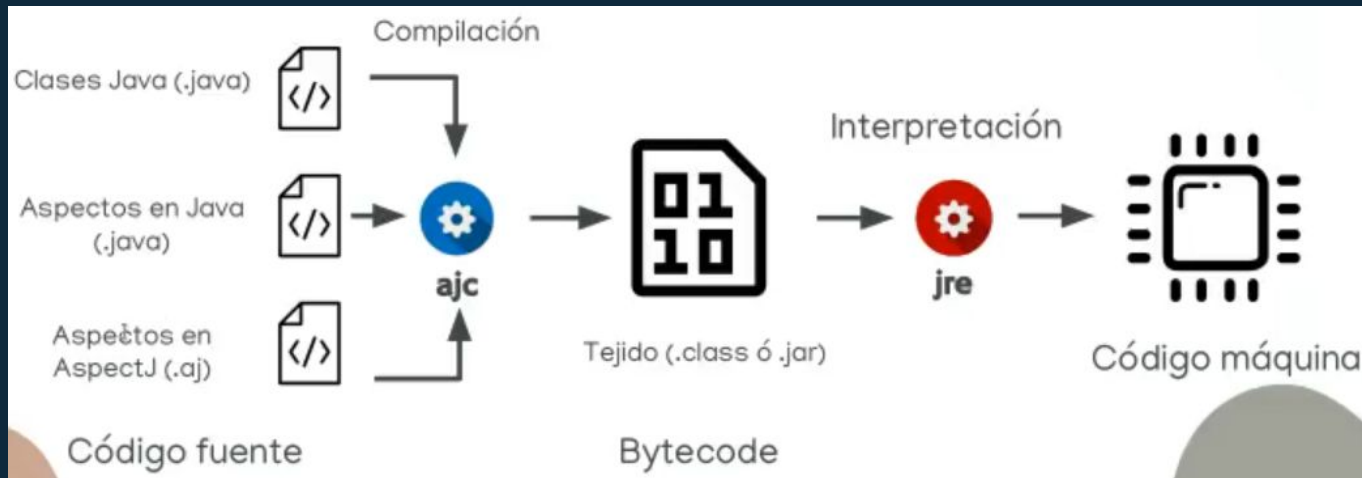
Aspectj compiler



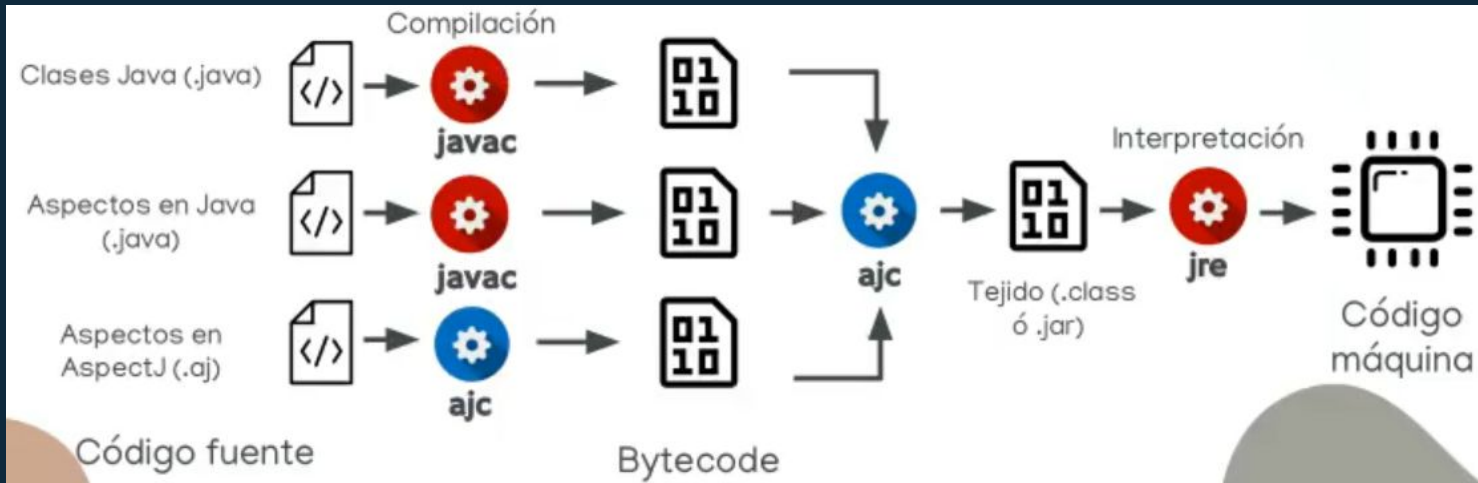
Proceso de compilación Java



Proceso de compilación Java + Aspectj



Proceso de compilación Java + Aspectj





EJEMPLO





Ejemplo POA

```
public void insertarCliente (Cliente elCliente, String IdUsuario){  
  
    Session miSesion=sessionFactory.getCurrentSession();  
  
    miSesion.save (elCliente);  
  
}
```





Ejemplo POA

```
public void insertarCliente (Cliente elCliente, String IdUsuario){  
  
    // código para login  
  
    Session miSesion=sessionFactory.getCurrentSession();  
  
    miSesion.save (elCliente);  
  
}
```



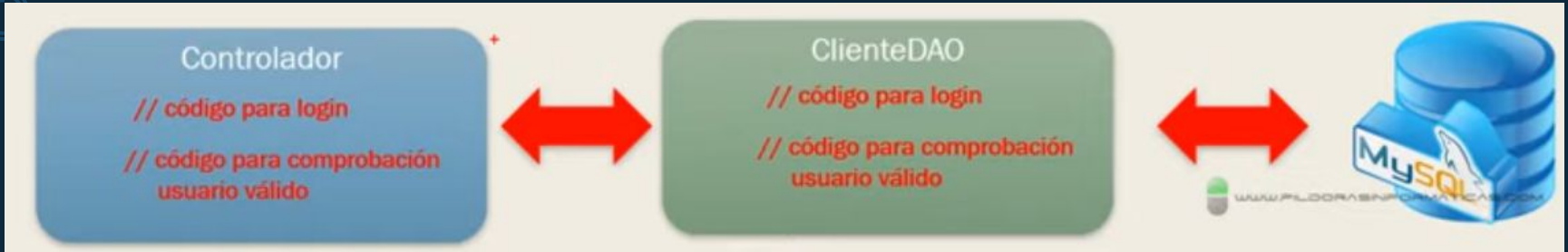


Ejemplo POA

```
public void insertarCliente (Cliente elCliente, String IdUsuario){  
  
    // código para login  
  
    // código para comprobación usuario válido  
  
    Session miSesion=sessionFactory.getCurrentSession();  
  
    miSesion.save (elCliente);  
  
}
```



Ejemplo POA

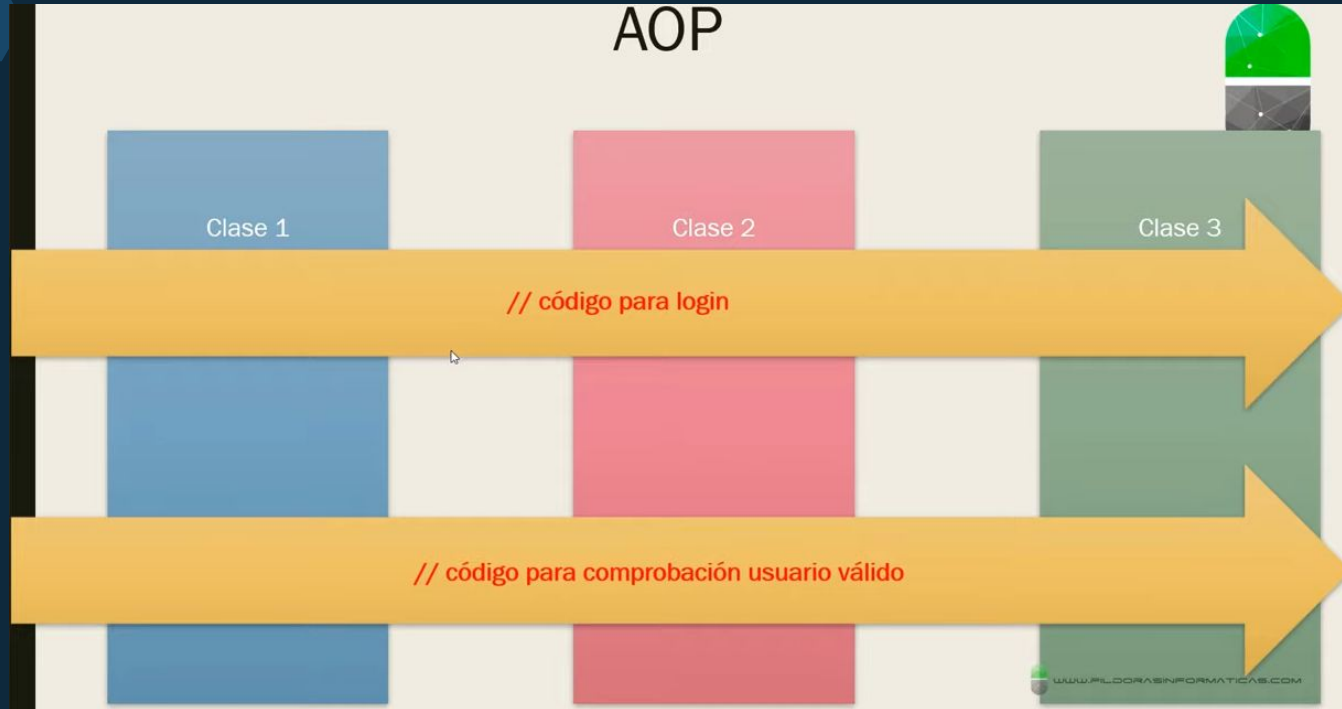



No solo clientes:

- Pedidos
- Productos
- etc

Dejaría varias clases con código duplicado

Ejemplo POA





Configuración proyecto Java con Aspectj en IntelliJ



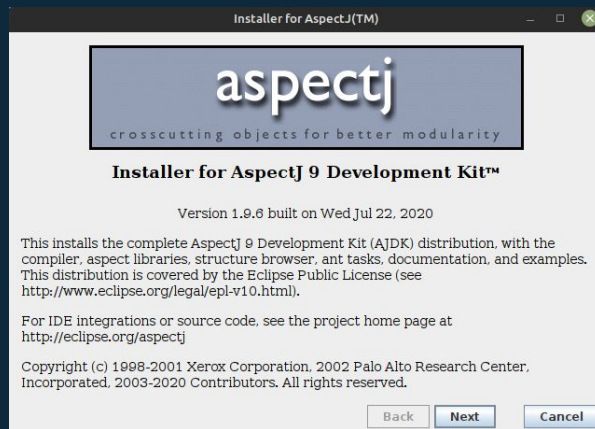


Instalación Aspectj

1. Descargar jar file de aspectj en <https://www.eclipse.org/downloads/download.php?file=/tools/aspectj/aspectj-1.9.6.jar>
2. Instalar aspectj con el comando
java -jar aspectj-1.9.6.jar

```
cdtafurd@cdtafurd:~/Downloads$ java -jar aspectj-1.9.6.jar
```

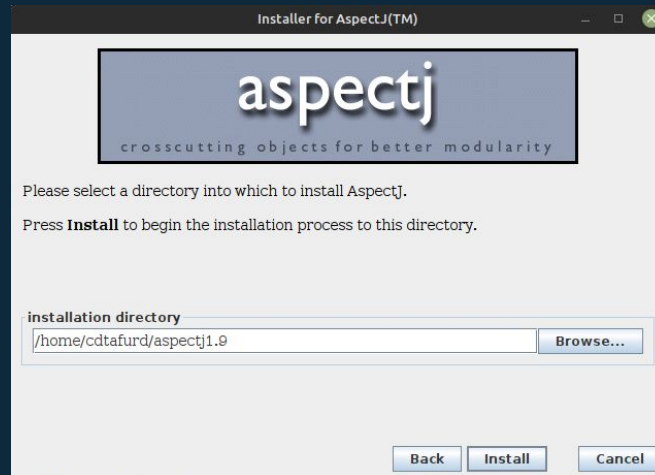
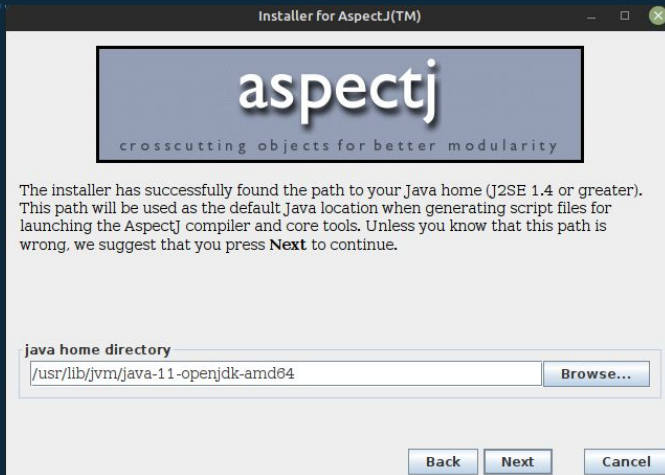
3. Se abre un instalador, dar Next





Instalación Aspectj

4. Se identifica el directorio de Java home, click next y escoja la ubicación donde quedará instalado y luego install





Instalación Aspectj

5. Inicie IntelliJ
6. Crear nuevo proyecto
7. Escoger proyecto Java
8. Escriba el nombre del proyecto y de click en Finish
9. Abrir configuración Ctrl+Alt+s
10. Build, Execution, Deployment > Compiler > Java compiler
11. Usar compilador AJC y escoger la ruta del archivo aspectjtools.jar previamente instalado





Settings

Build, Execution, Deployment > Compiler > Java Compiler

Use compiler: Ajc

Use '-release' option for cross-compilation (Java 9 and later)

Project bytecode version: Same as language level

Per-module bytecode version:

Module	Target bytecode version
All modules will be compiled with project bytecode version	

Ajc Options

Path to aspectjtools.jar: /home/cdtafurd/aspectj1.9/lib/aspectjtools.jar

Command line parameters:

Generate debug info

Delegate to Javac

Enable annotation processing options

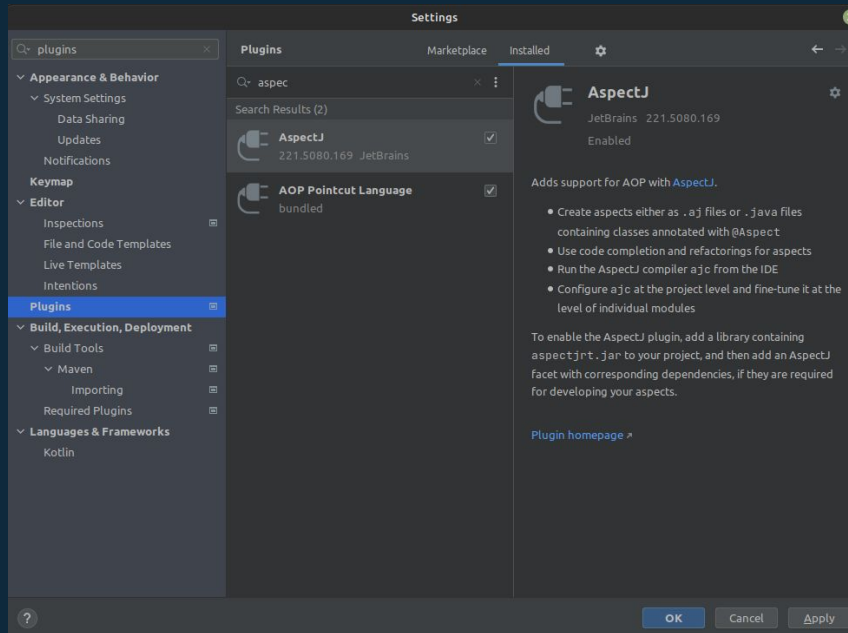
OK Cancel Apply



Instalación Aspectj

12. Teste y click en Ok

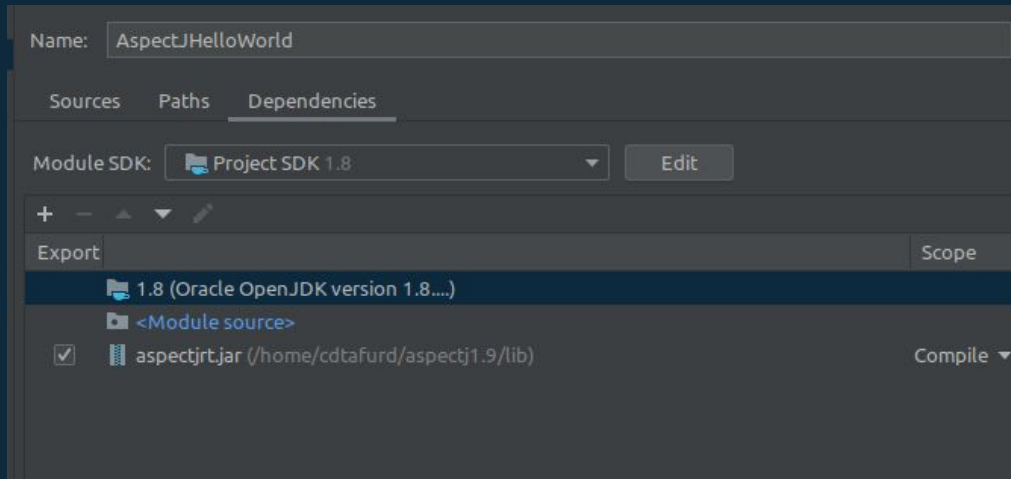
13. Instale el plugin de aspectj en intellij






Instalación Aspectj

14. Abrir las preferencias del proyecto (Ctrl + Alt + Shift +), click en Modules
15. Agregar aspectjrt.jar como dependencia del proyecto
16. Presionar Apply y luego ok





Referencias:

- ◇ El Desarrollo de Software Orientado a Aspectos: Un Caso Práctico para un Sistema de Ayuda en Línea: Marta S. Tabares B. Ph.d., Germán H. Alferez Salinas MSc., Edward M. Alferez Salinas MSc. <https://www.redalyc.org/pdf/1331/133115027009.pdf>
 - ◇ PROGRAMACIÓN ORIENTADA A ASPECTOS Análisis del paradigma: <https://www.angelfire.com/ri2/aspectos/Tesis/tesis.pdf>
 - ◇ Visión General de la Programación Orientada a Aspectos: <http://www.lsi.us.es/docs/informes/aopv3.pdf>
 - ◇ https://ferestrepoca.github.io/paradigmas-de-programacion/poa/poa_teoría/
 - ◇ http://www.jtech.ua.es/j2ee/publico/spring-2012-13/apendice_AOP-apuntes.html
 - ◇ <https://users.exa.unicen.edu.ar/catedras/deaspect/aspectj.pdf>
 - ◇ [https://codingornot.com/que-es-la-programacion-orientada-a-aspectos-aop#:~:text=La%20programaci%C3%B3n%20orientada%20a%20aspectos%20\(POA\)%20es%20un%20paradigma%20de,una%20correcta%20separaci%C3%B3n%20de%20responsabilidades](https://codingornot.com/que-es-la-programacion-orientada-a-aspectos-aop#:~:text=La%20programaci%C3%B3n%20orientada%20a%20aspectos%20(POA)%20es%20un%20paradigma%20de,una%20correcta%20separaci%C3%B3n%20de%20responsabilidades)
 - ◇ <http://tzachsolomon.blogspot.com/2015/08/how-to-create-hello-world-with-intellij.html>
- 



Muchas
gracias!

¿Preguntas?

