**Design Patterns**

*Dependency Injection* (injeção de dependência).

Os padrões de projetos inicialmente lançado por Chistopher Alexander, com o objetivo de resolver problemas enfrentados pela á arquitetura, logo ganhou mercado, onde teve publicações para outras áreas de conhecimento, exemplo disto foi o lançamento em 1995 de 23 padrões para uso específico em desenvolvimento de softwares por a famosa GoF, onde deixou de ser apenas um guia de 23 padrões e começou a ser realizado publicações em outros livros e catálogos.

Em catálogos de padrões de software, foi lançado um conhecido como *Dependency Injection* ou Injeção de independência, que em uma visão geral é uma forma de implementar um outro padrão dentro do seu código (Inversão de Controle), onde se é utilizado quando precisamos de desenvolver um sistema com diferentes módulos que seu nível de acoplamento seja extremamente baixo.

Existem 4 maneiras de implementar esse padrão:

I – Constructor – Modo em que é implementado a injeção de dependência na definição de construtores das classes.

II – Getter and Setter – Modo em que é implementado na definição dos gets e sets das classes.

III – Interface Implementation – Modo em que é usado interfaces para realizar as injeções de dependências

IV – Service Locador – Modo em que é contruido classes que servem como localizadoras de objetos que iremos instanciar em nossas outras classes.

No spring MVC, Framework Java muito poderoso, possibilita a criação dessas injeções de dependências através de anotações como @Autowired, @Component, @Service:

@Component – indica que é uma classe componente, isto é, quando o Spring efetuar o scanning do código ele deve criar uma instância para a classe. Funciona exatamente como a tag na configuração do XML.

@Service – ele extende o @Component, funciona da mesma maneira só que indica que é uma classe específica de serviço.

@Autowired – busca a instância da classe criada pelo Spring e insere no atributo.

Na anotação @Autowired, indica-se onde o spring deve injetar dependência, segue o código abaixo:

interface AlgoritmoDeEncriptografia {

String encriptografar(String mensagem);

}

/\*\*

\* O encritografador depende do algoritmo de encriptografia.

\* Este será injetado nesta classe por outra classe.

\*/

class Encriptografador {

AlgoritmoDeEncriptografia algEnc;

//O algoritmo de encriptografia será injetado no construtor

public Encriptografador(AlgoritmoDeEncriptografia algEnc) {

this.algEnc = algEnc;

}

public String encriptografar(String mensagem) {

return algEnc.encriptografar(mensagem);

}

}

// indica classe componente

@Component

public class Operacao {

// indica injeção de dependência

    @Autowired

    private ControleFilme cf;

    private boolean isAvaliacaoFilme(){

      boolean result = false;

// indica que se a variavel receber algo diferente de 0 retorna a string dentro do parênteses

      if  (this.cf.buscaFilme("lord of the rings") != null)

          result = true;

      return result;

   {

}