# [**JPA anotação @Autowired**](https://pt.stackoverflow.com/questions/85779/jpa-anota%c3%a7%c3%a3o-autowired)

O @Autowired é o mesmo que extends?

Vejamos um pequeno exemplo:

package com.dendetech.services;

import com.dendetech.entity.Participante; import com.dendetech.repositories.ParticipanteRepository; import org.springframework.beans.factory.annotation.Autowired; import org.springframework.stereotype.Service;

@Service public class ParticipanteService {

@Autowired

private ParticipanteRepository participanteRepository;

/\*

\* Crud Operations

\*/

<(Restante omitido). >

1 Resposta

[Ativos](https://pt.stackoverflow.com/questions/85779/jpa-anota%c3%a7%c3%a3o-autowired?answertab=active#tab-top)[Oldest](https://pt.stackoverflow.com/questions/85779/jpa-anota%c3%a7%c3%a3o-autowired?answertab=oldest#tab-top)[Votos](https://pt.stackoverflow.com/questions/85779/jpa-anota%c3%a7%c3%a3o-autowired?answertab=votes#tab-top)

1

Não, cada coisa é uma coisa.

Esta anotação está injetando automaticamente uma instancia de ParticipanteRepository.

Para que você não precise usar new ParticipanteRepository() ou outro método de criação sempre.

Extends é Herança, neste caso o Service herdaria do Repositor

# **Qual é a diferença entre @Inject e @Autowired no Spring Framework? Qual deles usar sob qual condição?**

Eu estou passando por alguns blogs na SpringSource e em um dos blogs, o autor está usando @Inject e suponho que ele também pode usar @Autowired.

Aqui está o pedaço de código:

@Inject private CustomerOrderService customerOrderService;

Não tenho certeza sobre a diferença entre @Inject e @Autowired e gostaria que alguém explicasse a diferença e qual deles usar em que situação?

[**java**](https://www.it-swarm.dev/pt/java/)[**spring**](https://www.it-swarm.dev/pt/spring/)[**dependency-injection**](https://www.it-swarm.dev/pt/dependency-injection/)[**autowired**](https://www.it-swarm.dev/pt/autowired/)[**inject**](https://www.it-swarm.dev/pt/inject/)

Supondo aqui que você está se referindo às anotações [javax.inject.Inject](http://download.oracle.com/javaee/6/api/javax/inject/Inject.html" \t "_blank) . @Inject faz parte do padrão Java CDI ( [Contextos e Injeção de Dependência](http://docs.oracle.com/javaee/6/tutorial/doc/giwhb.html" \t "_blank) ) introduzido no Java EE 6 (JSR-299), [leia mais](http://www.oracle.com/technetwork/articles/javaee/javaee6overview-141808.html" \t "_blank) . O Spring escolheu suportar o uso de @Inject como sinônimo de sua própria anotação @Autowired.

Então, para responder a sua pergunta, @Autowired é a anotação (legada) do próprio Spring. @Inject faz parte de uma nova tecnologia Java chamada CDI que define um padrão para injeção de dependência semelhante ao Spring. Em um aplicativo Spring, as duas anotações funcionam da mesma forma que o Spring decidiu suportar algumas anotações JSR-299 além das próprias.

A anotação @Autowired é definida no framework Spring.

@Inject annotation é uma anotação padrão, que é definida no padrão ["Injeção de Dependência para Java" (JSR-330)](https://jcp.org/en/jsr/detail?id=330) . Spring (desde a versão 3.0) suporta o modelo generalizado de injeção de dependência que é definido no padrão JSR-330. ( [frameworks Google Guice](https://en.wikipedia.org/wiki/Google_Guice) e [framework Picocontainer](http://picocontainer.com/web) também suportam este modelo).

Com @Inject pode ser injetada a referência à implementação da interface Provider, que permite injetar as referências adiadas.

Anotações @Inject e @Autowired- são analogias quase completas. Além da anotação @Autowired, a anotação @Inject pode ser usada para propriedades de ligação automática, métodos e construtores.

Em contraste com a anotação @Autowired, a anotação @Inject não possui o atributo required. Portanto, se as dependências não forem encontradas, será lançada uma exceção.

Existem também diferenças nos esclarecimentos das propriedades de ligação. Se houver ambiguidade na escolha dos componentes para a injeção, o qualificador @Named deve ser adicionado. Em uma situação similar para @Autowired, anotação será adicionada ao qualificador @Qualifier (a JSR-330 define sua própria anotação @Qualifier e por meio dessa anotação de qualificador @Named é definida).

<https://www.it-swarm.dev/pt/java/qual-e-diferenca-entre-inject-e-autowired-no-spring-framework-qual-deles-usar-sob-qual-condicao/940273742/>