What is the meaning of the CascadeType.ALL for a @ManyToOne JPA association

Asked 7 years, 4 months ago Active 1 month ago Viewed 298k times



I think I misunderstood the meaning of cascading in the context of a @ManyToOne relationship.

198 The case:

public class User {

```
@OneToMany(fetch = FetchType.EAGER)
protected Set<Address> userAddresses;
```

4

74

```
public class Address {
    @ManyToOne(fetch = FetchType.LAZY, cascade = CascadeType.ALL)
    protected User addressOwner;
```

What is the meaning of the cascade = CascadeType.ALL ? For example, if I delete a certain address from the database, how does the fact that I added the cascade = CascadeType.ALL affect my data (the User, I guess)?

java jpa one-to-many cascade many-to-one

edited Feb 12 '19 at 15:15



4.7k 20 275

asked Oct 23 '12 at 9:21



,530 14 6

6 Answers

3 watchers 865 questions

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.



334

DETACH) to the relating entities.



It seems in your case to be a bad idea, as removing an Address would lead to removing the related User. As a user can have multiple addresses, the other addresses would become orphans. However the inverse case (annotating the User) would make sense - if an address belongs to a single user only, it is safe to propagate the removal of all addresses belonging to a user if this user is deleted.



BTW: you may want to add a mappedBy="address0wner" attribute to your User to signal to the persistence provider that the join column should be in the ADDRESS table.



edited May 1 '13 at 18:25

answered Oct 23 '12 at 9:33



kostja

3k 45 158 2

- +1 for the best and shortest explanation of mappedBy I've ever come across. Ridcully Sep 30 '14 at 6:19
- 4 It could be good to have CascadeType.ALL on @OneToMany side though. mvmn Mar 31 '17 at 11:14



See here for an example from the OpenJPA docs. cascadeType.ALL means it will do all actions.



Quote:



CascadeType.PERSIST: When persisting an entity, also persist the entities held in this field. We suggest liberal application of this cascade rule, because if the EntityManager finds a field that references a new entity during flush, and the field does not use CascadeType.PERSIST, it is an error.



CascadeType.REMOVE: When deleting an entity, also delete the entities held in this field.

CascadeType.REFRESH: When refreshing an entity, also refresh the entities held in this field.

CascadeType.MERGE: When merging entity state, also merge the entities held in this field.

Sebastian

edited Oct 10 '15 at 5:46

answered Oct 23 '12 at 9:27

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.



- 4 New in JPA, this information is usefull but what about Detach here? Sarz Dec 5 '14 at 4:55
- 1 In CascadeType.DETACH, when detaching an entity, em also detach the entities held by parent entity. Dorian Mejer Jun 4 '16 at 20:41



As I explained in this article and in my book, <u>High-Performance Java Persistence</u>, you should never use CascadeType.ALL on @ManyToOne since entity state transitions should propagate from Parent entities to Child ones.

25

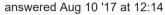
The @ManyToOne side is always the Child association since it should map the underlying FK.



Therefore, move the CascadeType.ALL from the @ManyToOne association to the @OneToMany which should use the mappedBy attribute since it's the most efficient one-to-many mapping.`



edited Apr 24 '19 at 6:10





Vlad Mihalcea

74.7k 20 275 593

From the EJB3.0 Specification:

18

Use of the cascade annotation element may be used to propagate the effect of an operation to associated entities. The cascade functionality is most typically used in parent-child relationships.



If X is a managed entity, the remove operation causes it to become removed. The remove operation is cascaded to entities referenced by X, if the relationships from X to these other entities is annotated with the cascade=REMOVE or cascade=ALL annotation element value.

()

So in a nutshell, entity relationships defined with CascadeType.All will ensure that all persistence events such as persist, refresh, merge and remove that occur on the parent, will be passed to the child. Defining other CascadeType options provides the developer with a more granular level of control over how the entity association handles persistence.

For example if I had an object Book that contained a List of pages and I add a page object within this list. If the <code>@OneToMany</code> annotation defining the association between Book and Page is marked as <code>CascadeType.All</code>, persisting the Book would result in the Page also being persisted to the database.

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.





2.982 5 25 41



15

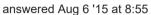


In JPA 2.0 if you want to delete an address if you removed it from a User entity you can add orphanRemoval=true (instead of CascadeType.REMOVE) to VOUR @OneToMany.

More explanation between orphanRemoval=true and CascadeType.REMOVE is here.



edited Oct 10 '15 at 5:09





Emilien Brigand



If you just want to delete the address assigned to the user and not to affect on User entity class you should try something like that:

```
@Entity
public class User {
  @OneToMany(mappedBy = "addressOwner", cascade = CascadeType.ALL)
  protected Set<Address> userAddresses = new HashSet<>();
@Entity
public class Addresses {
  @ManyToOne(cascade = CascadeType.REFRESH) @JoinColumn(name = "user id")
  protected User addressOwner;
```

This way you dont need to worry about using fetch in annotations. But remember when deleting the User you will also delete connected address to user object.

edited Jan 8 at 9:27



Negar Zamiri

answered Jan 19 '19 at 18:32

