

# Performance difference between annotating fields or getter methods in Hibernate / JPA

Asked 16 years ago   Modified 11 years, 9 months ago   Viewed 6k times



8



I was curious if anyone had any hard numbers around the performance difference between annotating Entities using private fields instead of public getter methods. I've heard people say that fields are slower because they're called "through reflection" but then again so are the getter methods, no? Hibernate needs to set the accessibility of the field to true before it tries to read it which I can see having some *slight* overhead. However wouldn't that be done at the Class-level in the scope of a Session or perhaps only once when the Configuration is read and SessionFactory is built?

Just curious if this is a myth or if there's really truth to it; I personally find annotating the fields to be a bit more readable.

java

hibernate

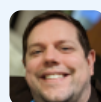
jpa

Share

Improve this question

Follow

asked Dec 1, 2008 at 23:25



[cliff.meyers](#)

17.7k ● 6 ● 53 ● 67

## 3 Answers

Sorted by:

Highest score (default)



9



Loaded 5000 records into a simple 3 column table. Mapped two classes to that table, one using annotated private fields and another using annotated public getters. Ran 30 runs of Spring's `HibernateTemplate.loadAll()` followed by a `HibernateTemplate.clear()` to purge the Session cache. Results in ms below...

methods total: 6510, average: 217

fields total: 6586, average: 219

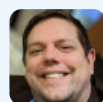
I should probably take another stab at it after adding more properties to each class but right now the difference doesn't appear to be statistically significant.

[Share](#) [Improve this answer](#)

[edited Mar 12, 2013 at 12:39](#)

[Follow](#)

answered Dec 2, 2008 at 1:38



[cliff.meyers](#)


17.7k ● 6 ● 53 ● 67

---

3 5000 records isn't statistically significant. Can you try it with a more complex entity structure (which would need joins), and with something like 100x the number of records? You will always have fast performance with small data sets.

– [Justin Standard](#) Apr 24, 2009 at 18:06

---

- 1 To anyone interested in the future. I've run very similar tests on [IMDb](#) dataset. I've fetch actors table (over 800k rows) altogether with their roles (1.3kk rows). Average results are: Field-based access: 17720 ms. Property-based access: 17914 ms. – [Aramka](#) Jul 9, 2020 at 18:58 
- 



1

ok, I can't give numbers haha, but I would guess that accessing the fields through reflection wouldn't be a 'one time' thing. Each object has its own private members.



I honestly don't know much about reflection but the getter/setters should be straight forward. In fact you can try setting one of the methods to private and I think it won't work because it can't find the method it needs.



There are other issues like proxies that will affect getter methods though depending on how you load your entities.

This is all I see in the documentation:

The access attribute lets you control how Hibernate will access the property at runtime. By default, Hibernate will call the property get/set pair. If you specify `access="field"`, Hibernate will bypass the get/set pair and access the field directly, using reflection. You may specify your own strategy for property access by naming a class that implements the interface `org.hibernate.property.PropertyAccessor`.

My guess is that reflection in general is going to be a higher cost though, but sorry.. no numbers :(

Share Improve this answer

answered Dec 1, 2008 at 23:39

Follow



Arthur Thomas

5,177 ● 1 ● 28 ● 31

- 
- 1 The get/set methods need to be invoked through reflection also. There is no way at compile time that Hibernate can know which get/set pairs it will need to invoke on a class. My point about setting the accessibility is that it's done at the Class level on a `java.lang.reflect.Field` instance...

– [cliff.meyers](#) Dec 1, 2008 at 23:46

---

ahh yes, I guess that makes sense since it has to look up the actual names. – [Arthur Thomas](#) Dec 1, 2008 at 23:57

- 
- 1 @cliff.meyers It's always compile time.... Hibernate uses class generation and AFAIK places a helper class into the entity package so it needs no reflection except for `private` members (package private is fine). – [maaartinus](#) Aug 7, 2015 at 21:48



0



Generally use annotation in above of getter methods, Because when class is loaded in JVM at that hibernate model or entity class is also loaded then if you use annotation above of field or peroperty then it will execute only once at time of each request. whereas if you placing above getter then in reflaction or any other layer when it will caled this class then getter method called then this become very useful

Share Improve this answer

answered Mar 12, 2013 at 7:57

Follow



Pradip Bhatt

658 ● 9 ● 15

- 
- 2 Could you please improve the wording of your answer? I don't think this provides anything new that hasn't already been covered in the existing answers, so would probably be better off in a comment to an existing marked answer. – [Dutts](#)  
Mar 12, 2013 at 8:18
-