Hibernate Session get() vs load() difference with examples

APRIL 2, 2018 BY PANKAJ — 23 COMMENTS

Hibernate Session provide different methods to fetch data from database. Two of them are – **get()** and **load()**. There are also a lot of overloaded methods for these, that we can use in different circumstances.

At first look both `get()` and `load()` seems similar because both of them fetch the data from database, however there are few differences between them, let's look at them with a simple example.

```
package com.journaldev.hibernate.main;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

import com.journaldev.hibernate.model.Employee;
import com.journaldev.hibernate.util.HibernateUtil;
```

```
public class HibernateGetVsLoad {

	public static void main(String[] args) {

		//Prep Work
		SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
		Session session = sessionFactory.openSession();
		Transaction tx = session.beginTransaction();

		//Get Example
		Employee emp = (Employee) session.get(Employee.class, new Long(2));
		System.out.println("Employee get called");
		System.out.println("Employee ID= "+emp.getId());
```

When I execute above code, it produces following output.

```
Hibernate: select employee0_.emp_id as emp_id1_1_0_, employee0_.emp_name as
emp_name2_1_0_, employee0_.emp_salary as emp_sala3_1_0_, address1_.emp_id as
emp_id1_0_1_, address1_.address_line1 as address_2_0_1_, address1_.city as city3_0_1_,
address1_.zipcode as zipcode4_0_1_ from EMPLOYEE employee0_ left outer join ADDRESS
address1_ on employee0_.emp_id=address1_.emp_id where employee0_.emp_id=?
Employee get called
Employee ID= 2
Employee Get Details:: Id= 2, Name= David, Salary= 200.0, {Address= AddressLine1=
Arques Ave, City=Santa Clara, Zipcode=95051}

Employee load called
Employee ID= 1
Hibernate: select employee0_.emp_id as emp_id1_1_0_, employee0_.emp_name as
emp_name2_1_0_, employee0_.emp_salary as emp_sala3_1_0_, address1_.emp_id as
emp_id1_0_1_, address1_.address_line1 as address_2_0_1_, address1_.city as city3_0_1_,
address1_.zipcode as zipcode4_0_1_ from EMPLOYEE employee0_ left outer join ADDRESS
address1_ on employee0_.emp_id=address1_.emp_id where employee0_.emp_id=?
Employee load Details:: Id= 1, Name= Pankaj, Salary= 100.0, {Address= AddressLine1=
Albany Dr, City=San Jose, Zipcode=95129}
```

From the output it's clear that `get()` returns the object by fetching it from database or from hibernate cache whereas `load()` just returns the reference of an object that might not actually exists, it loads the data from database or cache only when you access other properties of the object.

Now let's try to fetch data that doesn't exists in the database.

```
//Get Example
try{
Employee emp = (Employee) session.get(Employee.class, new Long(200));
System.out.println("Employee get called");
if(emp != null){
System.out.println("Employee GET ID= "+emp.getId());
System.out.println("Employee Get Details:: "+emp+"\n");
}
}catch(Exception e){
        e.printStackTrace();
}

//load Example
try{
Employee emp1 = (Employee) session.load(Employee.class, new Long(100));
System.out.println("Employee load called");
System.out.println("Employee LOAD ID= "+emp1.getId());
System.out.println("Employee load Details:: "+emp1+"\n");
}catch(Exception e){
        e.printStackTrace();
}
```

Above code produces following output.

```
Hibernate: select employee0_.emp_id as emp_id1_1_0_, employee0_.emp_name as
emp_name2_1_0_, employee0_.emp_salary as emp_sala3_1_0_, address1_.emp_id as
emp_id1_0_1_, address1_.address_line1 as address_2_0_1_, address1_.city as
city3_0_1_, address1_.zipcode as zipcode4_0_1_ from EMPLOYEE employee0_ left outer
join ADDRESS address1_ on employee0_.emp_id=address1_.emp_id where
employee0_.emp_id=?
Employee get called
Employee load called
Employee LOAD ID= 100
Hibernate: select employee0_.emp_id as emp_id1_1_0_, employee0_.emp_name as
emp_name2_1_0_, employee0_.emp_salary as emp_sala3_1_0_, address1_.emp_id as
emp_id1_0_1_, address1_.address_line1 as address_2_0_1_, address1_.city as
city3_0_1_, address1_.zipcode as zipcode4_0_1_ from EMPLOYEE employee0_ left outer
join ADDRESS address1_ on employee0_.emp_id=address1_.emp_id where
employee0_.emp_id=?
org.hibernate.ObjectNotFoundException: No row with the given identifier exists:
[com.journaldev.hibernate.model.Employee#100]
        at
```

```
org.hibernate.internal.SessionFactoryImpl$1$1.handleEntityNotFound(SessionFactoryImpl.j

        at
```

Look at the output closely, when we use `get()` to retrieve data that doesn't exists, it returns **null**. That makes sense because it try to load the data as soon as it's called.

With `load()`, we are able to print the id but as soon as we try to access other fields, it fires database query and throws `org.hibernate.ObjectNotFoundException` if there is no record found with the given identifier. It's hibernate specific Runtime Exception, so we don't need to catch it explicitly.

Let's look at some of the overloaded methods too. Above **get() and load()** methods could have been written as below too.

```
Employee emp = (Employee) session.get("com.journaldev.hibernate.model.Employee", new
Long(2));

Employee emp1 = (Employee) session.load("com.journaldev.hibernate.model.Employee", new
Long(1));

Employee emp2 = new Employee();
session.load(emp1, new Long(1));
```

There are other methods with `LockOptions` argument but I haven't used them. Notice that we need to pass full class name as argument.

Based on the above explanations we have following differences between **get() vs load()**:

1. `get()` loads the data as soon as it's called whereas `load()` returns a proxy object and loads data only when it's actually required, so `load()` is better because it support lazy loading.
2. Since `load()` throws exception when data is not found, we should use it only when we know data exists.
3. We should use `get()` when we want to make sure data exists in the database.

That's all for hibernate get and load methods, I hope it will clear some doubts and help you in deciding which one to use in different scenarios.

« PREVIOUS
Hibernate Named Query Example – @NamedQuery

NEXT »
[Solved] org.hibernate.AnnotationException: No identifier specified for entity Class

About Pankaj
If you have come this far, it means that you liked what you are reading. Why not reach little more and connect with me directly on **Google Plus**, **Facebook** or **Twitter**. I would love to hear your thoughts and opinions on my articles directly.

Recently I started creating video tutorials too, so do check out my videos on **Youtube**.

FILED UNDER: HIBERNATE

Comments

venky says
MAY 7, 2018 AT 11:14 PM

try{
Employee emp1 = (Employee) session.load(Employee.class, new Long(100));
System.out.println("Employee load called");
if(null!=emp1){
System.out.println("Employee LOAD ID= "+emp1.getId());
System.out.println("Employee load Details:: "+emp1+"\n");
}
}catch(Exception e){
e.printStackTrace();
}
above load method will throw error or not?
Reply

yashwanth potu says
APRIL 1, 2017 AT 5:13 PM
Thanks pankaj,It's precise and very explainatory
Reply

aman verma says
JANUARY 6, 2017 AT 11:36 AM
any other differences for load() and get() ?
Reply

Vinod Kumar says
OCTOBER 21, 2016 AT 8:31 AM
As Toral says that even we get the ID then it check for database. But In your explanation the query not hitting on getting ID. When I check in my system when I get ID is hitting database first after printing Id. Why is this plese let me know
Reply

Rakshit says
AUGUST 4, 2016 AT 1:32 AM
well explained
Reply

Sreekanth says

JULY 5, 2016 AT 10:46 PM

Good Explanation ..

Reply

preeti chauhan says

MAY 25, 2016 AT 5:49 PM

ya, i m very much staisfy with the above explanation.

Reply

NikhilJini says

FEBRUARY 14, 2016 AT 8:53 PM

thanks

Reply

Vijay Jetti says

APRIL 29, 2016 AT 12:01 AM

Thanks dude

Reply

Aakash Deep Sharma says

NOVEMBER 22, 2015 AT 7:45 AM

best explanation for this topic

Reply

kriti sinha says

SEPTEMBER 12, 2015 AT 12:28 AM

Best explanation i ever got for this concept.

Reply

Sudeshna says

AUGUST 13, 2015 AT 8:34 AM

Very clear and concise. Thank you.

Reply

Annu Upadhuaya says

JULY 17, 2015 AT 4:50 AM

Hi Pankaj,

Thank you so much for explaining it so perfectly.It helped me a lot.

Reply

Ravi says

JUNE 10, 2015 AT 5:25 AM

it's very good explanation

Reply

Dorel says

MAY 28, 2015 AT 8:21 PM

Well, quite bad explained, read more about get and load.

I will ask you: why not use ONLY .get() since it will retrieve the object or return null and forget about .load()?

Reply

RW says

FEBRUARY 17, 2015 AT 6:32 AM

Very well explained. Have a look at http://www.aurorasolutions.io/get-and-load-methods-in-hibernate-2/ . It's also one of the good ones.

Reply

Sumit Chauhan says

JANUARY 15, 2015 AT 3:51 AM

Good Explanation. Such an good example cleared the doubts.

Reply

Suresh Kumar Pandey says

DECEMBER 10, 2014 AT 1:12 AM

I used the same example and i am seeing there is no difference between the load and get apart from that
load() returns ObjectNotFoundException() and get() returns null pointer exception.

Reply

Suresh Kumar Pandey says

DECEMBER 10, 2014 AT 1:20 AM

Sorry for the comment above. I had missed the if else part. Thanks for the nice explanation.

Reply

Harman says

DECEMBER 15, 2014 AT 8:48 AM

Thanks for explanation

Reply

Gurdeep Singh says

NOVEMBER 10, 2014 AT 7:42 PM

Good Example…thanks for sharing

Reply

Toral says

SEPTEMBER 24, 2014 AT 11:38 AM

That's a great example. Clears the concept well. However, even in case of load, we cannot retrieve the Id.
I understand that to fetch any of the attributes including Id, a check is made in the database and the data retrieved. So, if the required record is not available, it will throw Object not found exception.
Please correct me if the understanding is wrong.

Reply

Vimal Kumar says

SEPTEMBER 22, 2014 AT 4:46 AM

Great!!! Seriously cleared my concept. Great Explanation with practical proofs

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Save my name, email, and website in this browser for the next time I comment.

POST COMMENT

Search for tutorials...

DOWNLOAD ANDROID APP

GET IT ON Google Play

HIBERNATE FRAMEWORK

- Access to DialectResolutionInfo
- get is not valid
- No CurrentSessionContext configured
- Hibernate Interview Questions

RECOMMENDED TUTORIALS

Java Tutorials

- Java IO
- Java Regular Expressions
- Multithreading in Java
- Java Logging
- Java Annotations
- Java XML
- Collections in Java
- Java Generics
- Exception Handling in Java
- Java Reflection
- Java Design Patterns
- JDBC Tutorial

Java EE Tutorials

- Servlet JSP Tutorial
- Struts2 Tutorial
- Spring Tutorial
- Hibernate Tutorial
- Primefaces Tutorial
- Apache Axis 2
- JAX-RS
- Memcached Tutorial

Employee Vacation Tracking

Onboarding New Employees

Employee Communication Apps

How to Retain Employees

Hiring Good Employees

Employee Engagement Activities

Employee Engagement Surveys

Employee Benefits Information