# The anatomy of Hibernate dirty checking mechanism

AUGUST 21, 2014 / VLADMIHALCEA

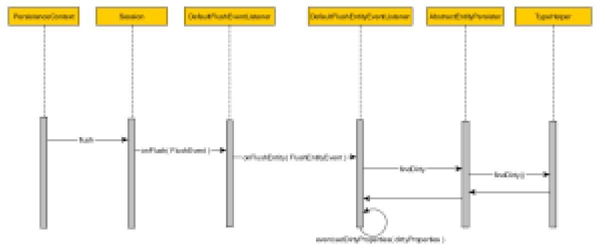
*(Last Updated On: January 29, 2018)*

## Introduction

The persistence context enqueues entity state transitions that get translated to database statements upon flushing. For managed entities, Hibernate can auto-detect incoming changes and schedule SQL UPDATES on our behalf. This mechanism is called **automatic dirty checking**.

## The default dirty checking strategy

By default Hibernate checks all managed entity properties. Every time an entity is loaded, Hibernate makes an additional copy of all entity property values. At flush time, every managed entity property is matched against the loading-time snapshot value:


PersistenceContext
Session
DefaultFlushEventListener
DefaultFlushEntityEventListener
AbstractEntityPersister
TypeHelper
flush
onFlush( FlushEvent )
onFlushEntity( FlushEntityEvent )
findDirty
findDirty()
dirtyProperties( dirtyProperties )

So the number of individual **dirty checks** is given by the following formula:

$$N = \sum_{k=1}^{n} p_k$$

where

n = The number of managed entities
p = The number of properties of a given entity

Even if only one property of a single entity has ever changed, Hibernate will still check all managed entities. For a large number of managed entities, the default dirty checking mechanism may have a significant CPU and memory footprint. Since the initial entity snapshot is held separately, the persistence context requires twice as much memory as all managed entities would normally occupy.

## Bytecode instrumentation

A more efficient approach would be to mark dirty properties upon value changing. Analogue to the original deep comparison strategy, it’s good practice to decouple the domain model structures from the change detection logic. The automatic entity change detection mechanism is a cross-cutting concern, that can be woven either at build-time or at runtime.

The entity class can be appended with bytecode level instructions implementing the automatic dirty checking mechanism.

## Weaving types

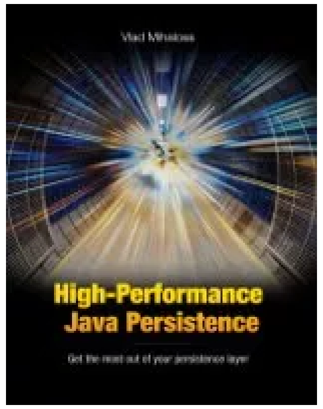
The bytecode enhancement can happen at:

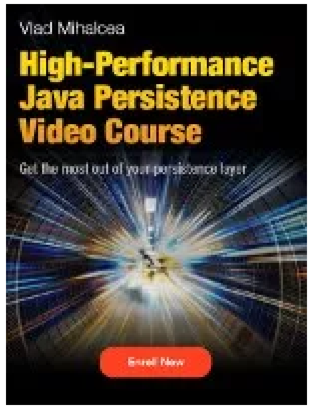
- Build-time

  After the hibernate entities are compiled, the build tool (e.g. ANT, Maven) will insert bytecode level instructions into each compiled entity class. Because the classes are enhanced at build-time, this process exhibits no extra runtime penalty. Testing can be done against enhanced class versions, so that the actual production code is validated before the project gets built.
- Runtime

  The runtime weaving can be done using:

  - A Java agent, doing bytecode enhancement upon entity class loading
  - A runtime container (e.g. Spring), using JDK Instrumentation support





## Hibernate 5 improvements

Hibernate 3 has been offering bytecode instrumentation through an ANT target but it never became mainstream and most Hibernate projects are still currently using the default deep comparison approach.
Hibernate 5 has redesigned the bytecode enhancement mechanism, is more reliable than it used to be.

- **3 chapters** from my book, **High-Performance Java Persistence**,
- a **10% discount** coupon for my book.

Get the most out of your persistence layer!

Subscribe

**Related**

How to enable bytecode enhancement dirty checking in Hibernate
February 11, 2016
In "Hibernate"

How to customize Hibernate dirty checking mechanism
August 29, 2014
In "Hibernate"

How to prevent lost updates in long conversations
September 22, 2014
In "Hibernate"

Categories: Hibernate, Java     Tags: automatic dirty checking, hibernate, Session flush, Training, Tutorial

← How does AUTO flush strategy work in JPA and Hibernate     How to customize Hibernate dirty checking mechanism →

## Leave a Reply

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

**Comment**

Name *

Email *

Website

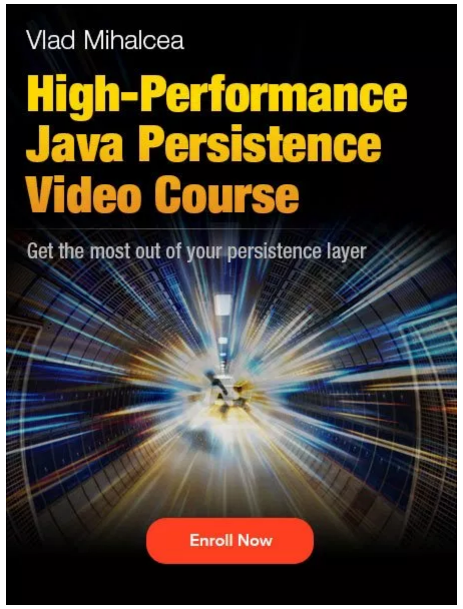
☐ Save my name, email, and website in this browser for the next time I comment.

☐ Sign me up for the newsletter!

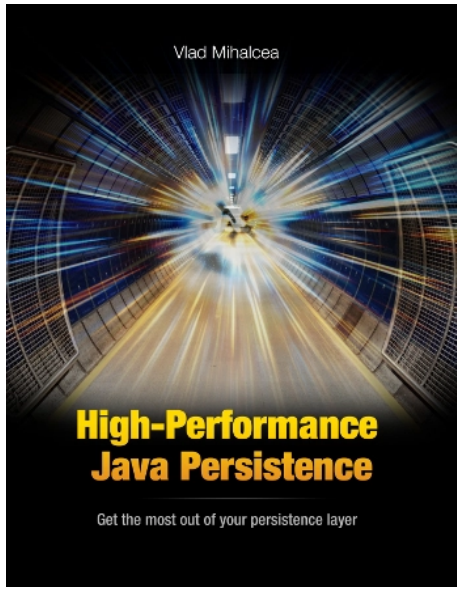
Post Comment

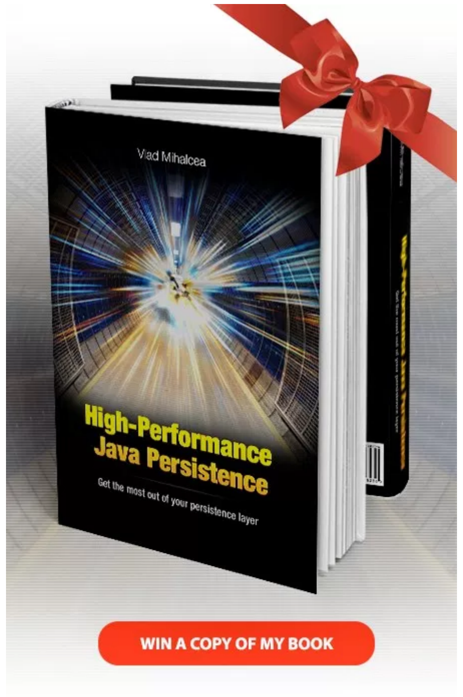
☐ Notify me of follow-up comments by email.

## VIDEO COURSE


Vlad Mihalcea
High-Performance Java Persistence Video Course
Get the most out of your persistence layer
Enroll Now

## HIGH-PERFORMANCE JAVA PERSISTENCE


Vlad Mihalcea
High-Performance Java Persistence
Get the most out of your persistence layer


Vlad Mihalcea
High-Performance Java Persistence
Get the most out of your persistence layer
WIN A COPY OF MY BOOK

Java Champions

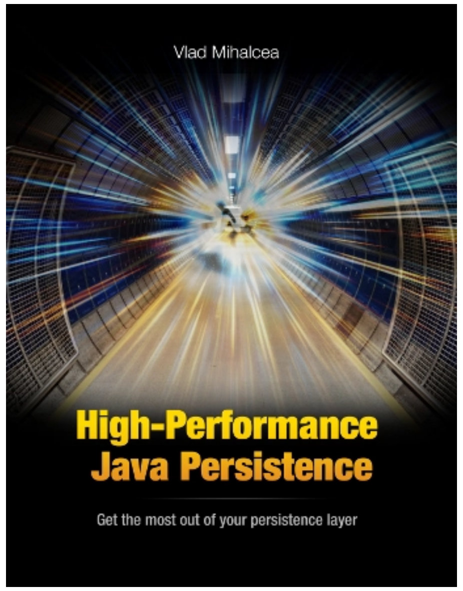
## HIBERNATE PERFORMANCE TUNING TIPS

## HYPERSISTENCE

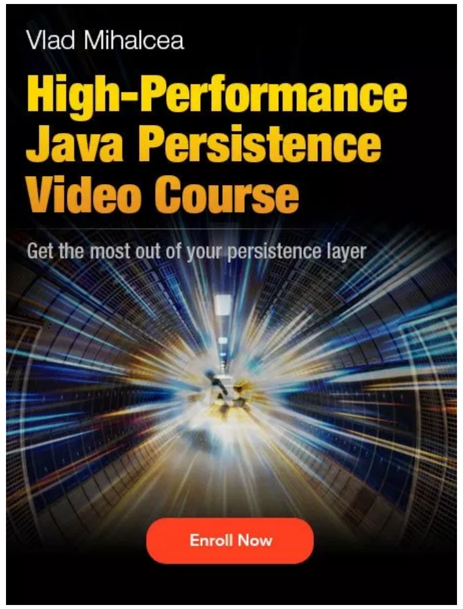
HYPERSISTENCE

## WIN A COPY OF MY BOOK!

## HIGH-PERFORMANCE JAVA PERSISTENCE


Vlad Mihalcea
High-Performance Java Persistence
Get the most out of your persistence layer

## VIDEO COURSE


Vlad Mihalcea
High-Performance Java Persistence Video Course
Get the most out of your persistence layer
Enroll Now

Search ...

RSS - Posts

RSS - Comments