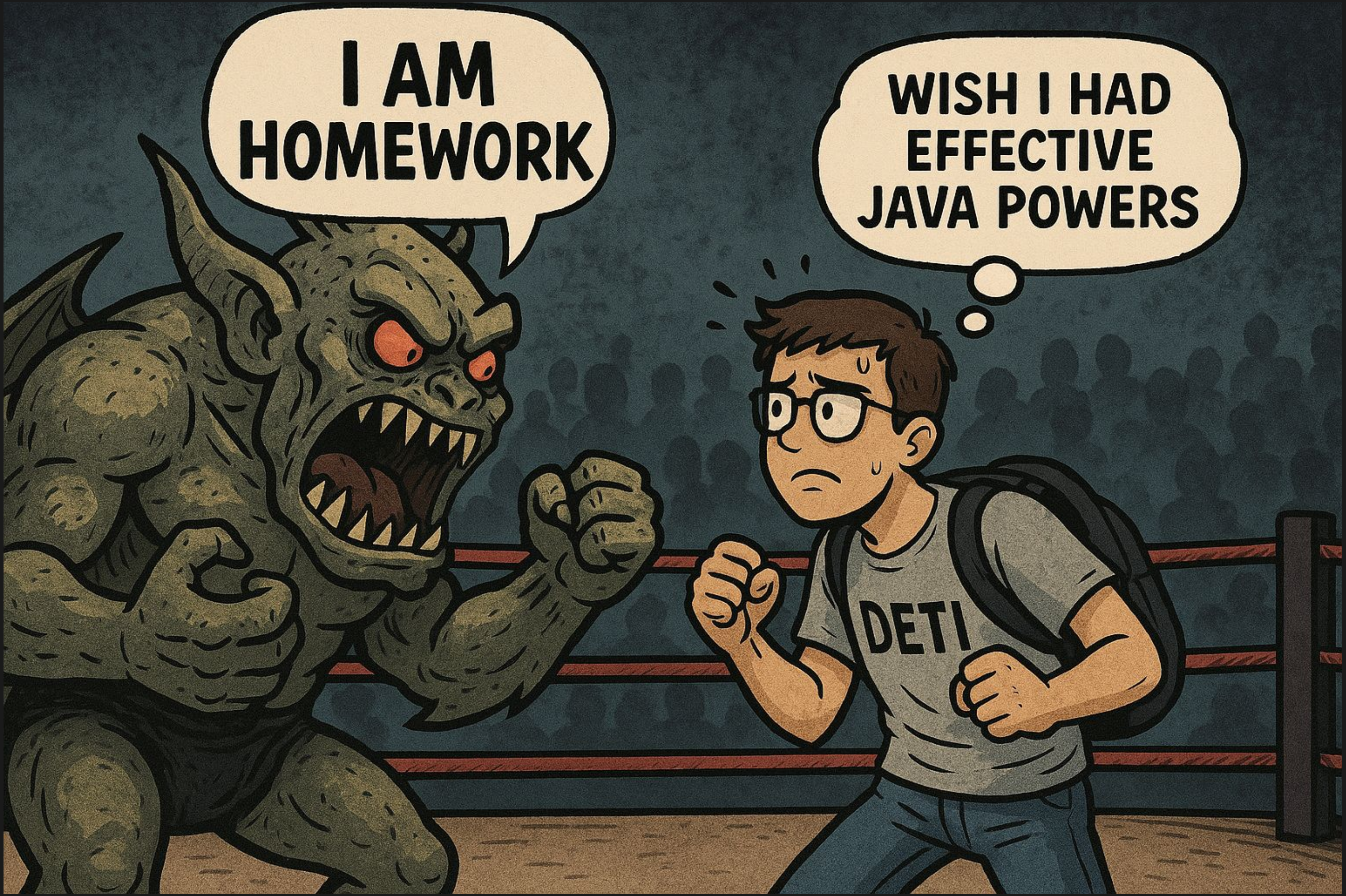
Effective java

Selected topics for Java developers
v2025-04-01

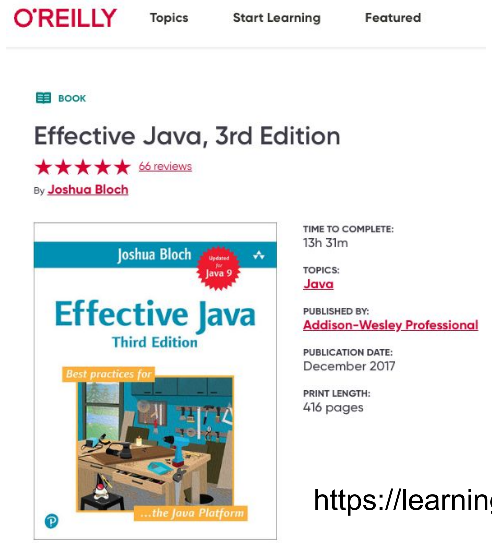


(AI generated image.)

Homework

vs

Effective Java



O'REILLY Topics Start Learning Featured

BOOK

Effective Java, 3rd Edition

★★★★★ 66 reviews

By Joshua Bloch

TIME TO COMPLETE:
13h 31m

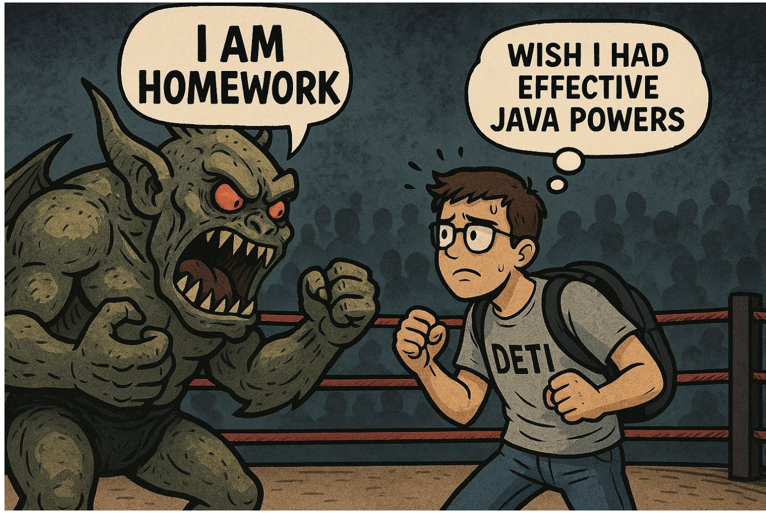
TOPICS:
Java

PUBLISHED BY:
Addison-Wesley Professional

PUBLICATION DATE:
December 2017

PRINT LENGTH:
416 pages

(AI generated image.)



https://learning.oreilly.com/library/view/effective-java-3rd/9780134686097/

Homework - comments

A long test method has several "sections" of code "clusters".

To clarify the intention of each block, the student inserted comments as if they were "section headers"

Any bad smells here?...

Martin's "Clean Code" → Explain yourself in code

Symptom: too many parameters in constructor

SonarQube reports a bad smell for my constructor which takes a large number of parameters… but I need them to initialize my entity and can't be any shorter.

Bloch's "Effective Java" → Item 2: Consider a **builder** when faced with many constructor parameters

+ code is self-documenting

+ specify only the parameters you need

+ constructs a final object that is typically immutable

+ easy to add new parameters

Symptom: hard to test my TTL

I set my cache entries to expire after 10sec (TTL); this is a short duration for the application logic, but is already too long for unit tests…

I have a security flag [in Sonar Analysis] because I've included the developer API key inline in the code.

"Effective Java" → Item 5: Prefer dependency injection to hardwiring resources

+ can easily change dependencies implementations without modifying the class.
+ enhances testability: easier to inject mock dependencies.
+ promotes loose coupling
+ externalizes configuration, making it easier to control behavior through constructors, factories, or frameworks

Symptom: I have a purpose-specific *cache*

My ExchangeRatesCache implements the cache behavior; my cache entries hold a “put” timestamp, the currency tag, and the exchange rate for a given currency (to Euro).

→ Item 29: Favor generic types

Symptom: returning *nulls*

The method searches for a Reservation by the reservation Id. If not found, or if there is a problem with the database connection, the caller will get a null return.

If the currency conversion rate is not yet available in the cache, then the lookup method returns -1.

“Clean Code” → Chap.7: Don’t return null; Use Exception rather than return codes

Symptom: string concatenation in logs

SonarQube is complaining about my log messages… how can it possible be a bad smell?

→ Item 63: Beware the performance of string concatenation

```
log.debug("User name is “ + userName +
“ and email is " + email);
```

```
logger.debug("User name is {} and email is {}",
userName, email);
```

Symptom: the catch-them-all option

SonarQube complained about catching the generic Exception, but, in this way, I avoid to write cases for 3 or 4 different "catches". Isn't this more efficient/clear?

→ Item 73: Throw exceptions appropriate to the abstraction

→ Item 77: Don't ignore exceptions

Symptom: do I need to create an Exception type?

I always use checked exceptions to force the program to handle exceptional behavior explicit, so it becomes more robust/secure.

→ Item 70: Use checked exceptions for recoverable conditions and runtime exceptions for programming errors

→ Item 72: Favor the use of standard exceptions

→ Item 49: Check parameters for validity *[fail quickly]*