

# Semantic Reviews

Goodbye ambiguous comments!



# Ambiguous Code Review Comments

Did you think about [random unrelated library]?



Your function doesn't handle [use case].

What happens if someone passes null here?



You could have named this [...] instead.



# Solution: Semantic Comments

How? **Prefix your comments with a label to express intent.**

- Clarifies intent.
- Adds clear expectations
- Removes ambiguity.

Suggestion: [some library] might make this easier.



Important: Your function doesn't handle [use case].



# Based on a Blog Post by Kevin Schaal



<https://www.m31coding.com/blog/semantic-reviews.html>



# Remark

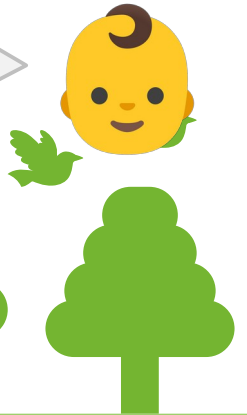
“I have something that I’d like to say.”



Remark: I like your approach here. Very clever.

- No change expected
- Good for positive feedback
- Basically an FYI

Remark: This is fine now but might start breaking when we scale up.



# Question

Question: How much will this cost per request?



“I have a genuine question and I’d like an answer.”



Question: Did you test this when [some edge case]?

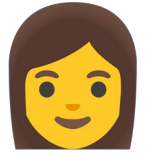
- No change expected
- Answer is expected
- Leads to conversation



# Hint

“I’d like to share something you might find interesting in the future.”

Hint: Check out [library], I think that it handles your user case.



- No change expected
- Good for teaching and sharing knowledge



Hint: You can avoid the nil check by initializing the map.



# Suggestion



Suggestion: Rename this function to [new name] to match what it really does.

“I have a suggestion and I expected you to think about it.”

Suggestion: Moving this try-catch to the caller will save a bunch of copy-paste.



- Change is not expected
- It should be considered
- Up to the PR author





# Important

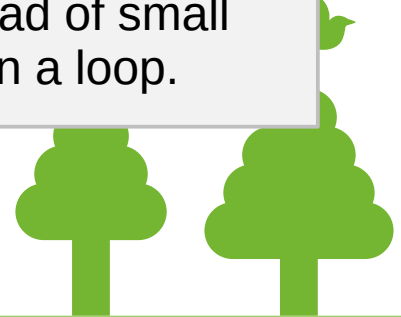
“This is important to me and I expect a change. I’m open to discussion.”

- Change is expected
- Open to conversation

Important: You’re better off crashing here. That way we’ll restart and recover.



Important: Do one big SQL query instead of small queries in a loop.



# Crucial



Crucial: This is vulnerable to SQL injection; use a prepared statement.

“This must not be merged. It has a severe issue and must be changed.”

Crucial: Your lambda has an infinite recursion here.



- Change expected
- Non-negotiable



Remark: That's all folks!



Suggestion: Applaud!

Hint: Find me after for questions.

