Robot Nitpicks

Never send a human to do a machine's job

Moshe Zadka – https://cobordism.com

PyBay 2018

The Easiest Most Useless Code Review

- Mechanical
- Annoying

Why?

- Speed bumps
- Feeling useful

Why not?

- Team solidarity
- Mentor and help

Nits

A taxonomy.

Coding Style

Cite chapter and verse.

Example – Line length

- PEP-8 says 80
- Black says 88
- Some say 100, 120...
- Literally just counting

Consistency

Compare two places, ask why.

Example – Literals or constructors

- `{}` vs `dict()`
- Paint the bikeshed fluourescent green

Best Practices

Plethora to pick from!

Local Practices

Especially annoying to new people.

Example – No print

...or fluourescent pink.

Humans are inconsistent

Easy to miss exceptions!

Humans get tired

"I've done most files in this PR."

Delay tactics

"I'm going to comment on every single line."

One upping

"I'm aware of the obscure rules in our local style guide, as well as historical versions of PEP-8"

Time pressure

"QA needs this yesterday, I don't have time to fix all of that at the last minute!"

Annoyance

"You're just doing this to me because..."

Configuring linters

Make the linter reflect current consensus.

Linter plugin

If the linter doesn't do something, implement it.

Ad-hoc programs

Just write a little script to check for copyright notices.

CI intergration

Automatic CI failure blocking merge on Lint errors.

Supporting legacy with diff

Only run lint on files in diff (or only fail on lint errors in diff)

Supporting legacy with white lists

Keep list of current failures, ignore "known" failures.

Simple rule

If CI doesn't care, the humans don't care!

Summary

- Humans gonna human
- Use that, don't fight that