
openSSL
CORPORATION

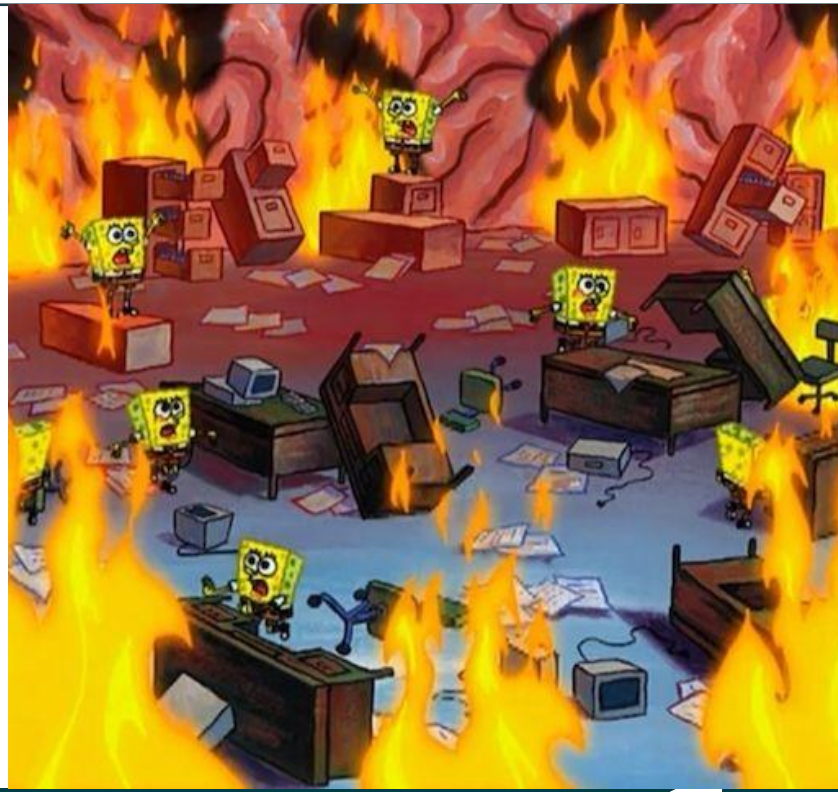
# I am going to break all the things.

Bob Beck

beck@openssl.org

# We are going to reformat the OpenSSL source code (again).

OMG..  Why why why why why….



# We want to improve OpenSSL development.

- We would like to continue to modernize our development efforts, and not sit still.
- Part of this includes efforts to have better tooling around reviews, and code quality
- We would like to have better and quicker code review
- We would like it to be easier for people to contribute without barriers to contribution that are easy to remove.

# Humans enforcing code formatting is for the naughties

- Humans are bad at complying to bespoke formatting styles when coding
- Humans are bad at enforcing bespoke formatting styles when reviewing

The result of this is both a lot of time can be spent on review over style nits and yet the actual standard enforced in the code base is varying.

Consistent formatting is supposed to make review easier, not consume review resources to maintain it.

Back in the day this was the only way to do this. That day has past, the world has moved on.

Modern development practices expect formatting can be handled by editor/IDE/submission processes. Developers and contributors do not want to learn a bespoke format by trial and error.

# clang-format has emerged as the defacto standard.

- There have been many tools over the years that do formatting… but…
- clang-format integrates into git based development processes.
- clang-format can be used in a CI pipeline, and can automatically check changes before commit.
- clang-format supports a bunch of ‘standard’ code styles - which are in turn likely to be known and supported in other tools, as well as recognizable to developers.

Like all tools it is somewhat imperfect, but good enough.

We want to use it on commit, and require formatting to a standard clang-format style.

Our existing bespoke style can not be completely expressed in clang-format and will require reformatting anyway.

We have chosen to reformat in the WebKit clang-format standard style. A .clang-format file to do that will be committed to the tree.  Our style guide and requirements will change to reflect this.



# This is a first step towards better review, and better code.

- Our git CI will enforce clang-format on PR submission.
- We will later extend clang-format to enforce further things
  - We hope to make includes self-contained and sorted, in the future.
  - We will endeavor that files include what they use.
  - These changes will be reflected in the style guide, and where possible enforced by clang-format.
- We will adopt other pre-submission checks
  - Spelling
  - Other analysis including agentic pre-review.

The goal is not to automate the review process, the goal is to make the review process better, and allow human reviewers more time to review the things humans should be reviewing.



# You killed my patches… You…

- Yes this is an event if you maintain patch stacks.
- It is not insurmountable.
- We will tag before and after the reformat on every branch.
  - Check out two copies of the tree at the before tag.
  - Apply your patches to one copy
    - Verify this still works as expected.
  - Run clang-format to reformat **your changed files** in **both** copies of the tree.
    - Again, verify it still works as expected.
  - Regenerate your patches from the unpatched to patched copy.
  - Now check out the tree at the after tag.
    - Verify your patch applies
    - Verify it still works as expected.
    - 


# Questions?