# Linux Kernel Maintainers

## Why are they so grumpy?

Greg Kroah-Hartman
gregkh@linuxfoudation.org

> “Publicly making fun of people is half the fun of open source programming.”
>
> – Linus Torvalds

# 2,833 developers
# 373 companies

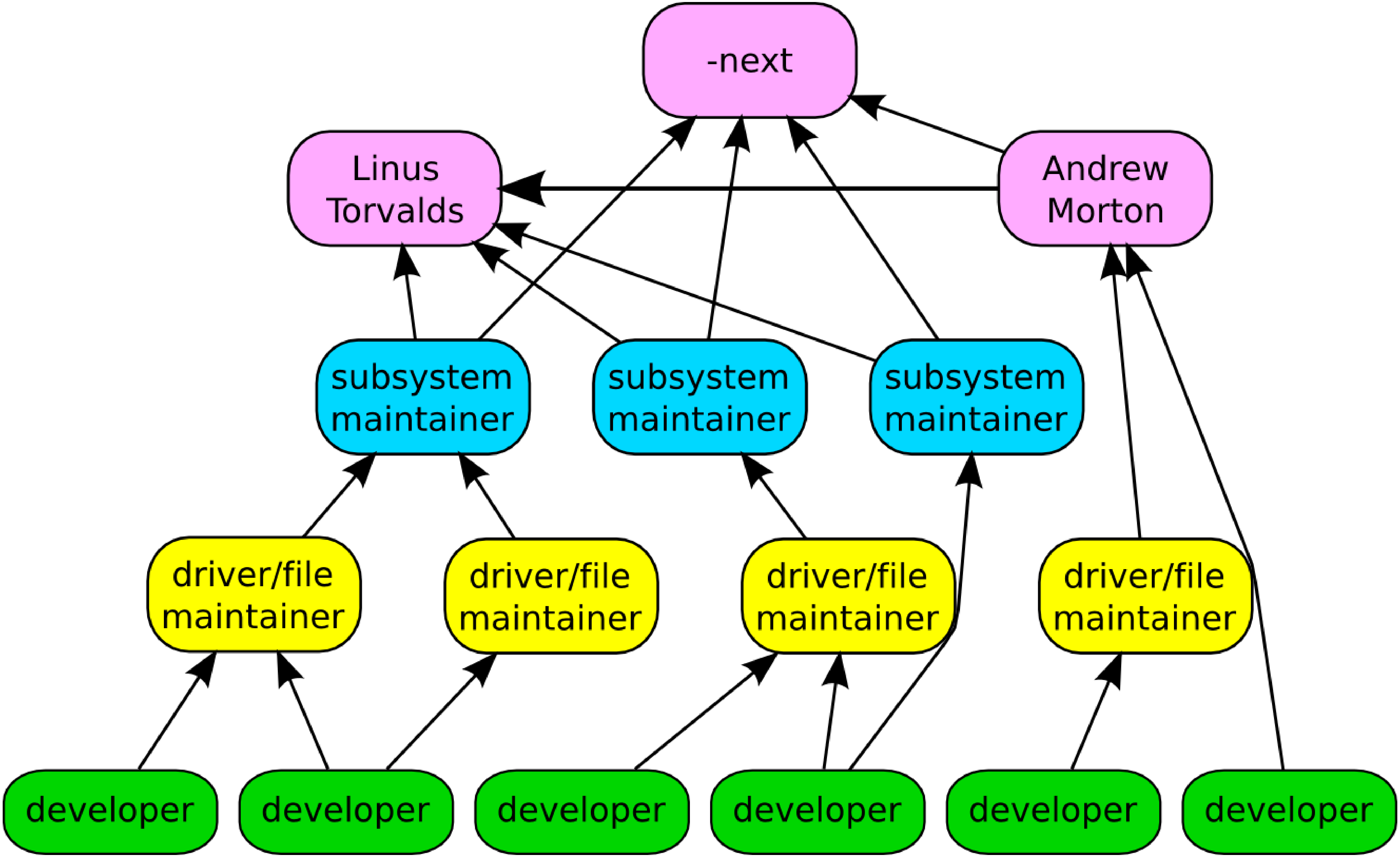
Kernel releases 3.0.0 – 3.4.0
May 2011 – May 2012

# 5.79 changes per hour

Kernel releases 3.0.0 – 3.4.0
May 2011 – May 2012

7.21 changes per hour

3.4.0 release


-next
Linus Torvalds
Andrew Morton
subsystem maintainer
subsystem maintainer
subsystem maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
driver/file maintainer
developer
developer
developer
developer
developer
developer

# Patches I received in the past 2 weeks

# Patches I received in the past 2 weeks

480

# Subject: [PATCH 48/48] …

# 15 patch series, no order given

# Patches 1, 3-10

# “Signed-off-by:” in signature

# Signature saying email was confidential

# Tabs were converted to spaces

# Leading spaces removed

# diff in non-unified format

# Patch created in driver directory

# Patch created in /usr/src/linux-2.6.32

# Made against different tree

# Wrong coding style

# Wrong coding style, and acknowledged it

# Would not compile

# Broke the build on patch 3/6

Broke the build on patch 3/6 and fixed it on 6/6

# Broke the build on patch 5/8

# Broke the build on patch 5/8

Contained note that fix would be sent later

# Patches that had nothing to do with me

# 1 patch, 450kb big (4500 lines added)

# Obviously wrong kerneldoc

This was a calm two weeks

# It is in my self-interest to ignore your patch

# Give me no excuse to reject your patch

# Proper coding style

# scripts/checkpatch.pl clean

# Sent to proper people and lists

# Sent to proper people and lists

scripts/get_maintainer.pl

# Proper Subject:

# Proper changelog comment

# “obviously” correct

# Small incremental change

# Description of WHY it is needed

# Which tree it was made against

# If multiple patches, state the order

# Has to build properly

# Make sure it works, if possible

# Don't ignore review comments

# Don't resend without saying why

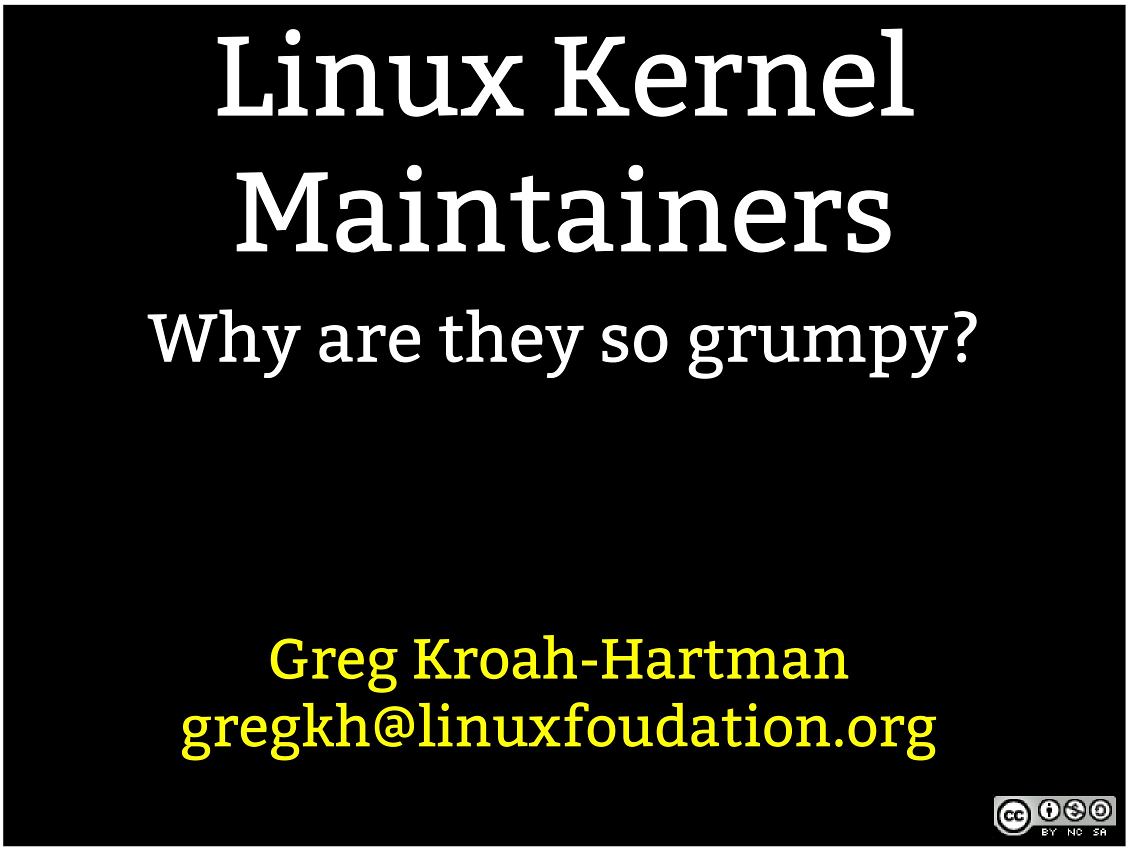
# What I will do for you:

# Review your patch within 1-2 weeks

# Offer semi-constructive criticism

# Let you know the status of your patch

github.com/gregkh/presentation-maintainer



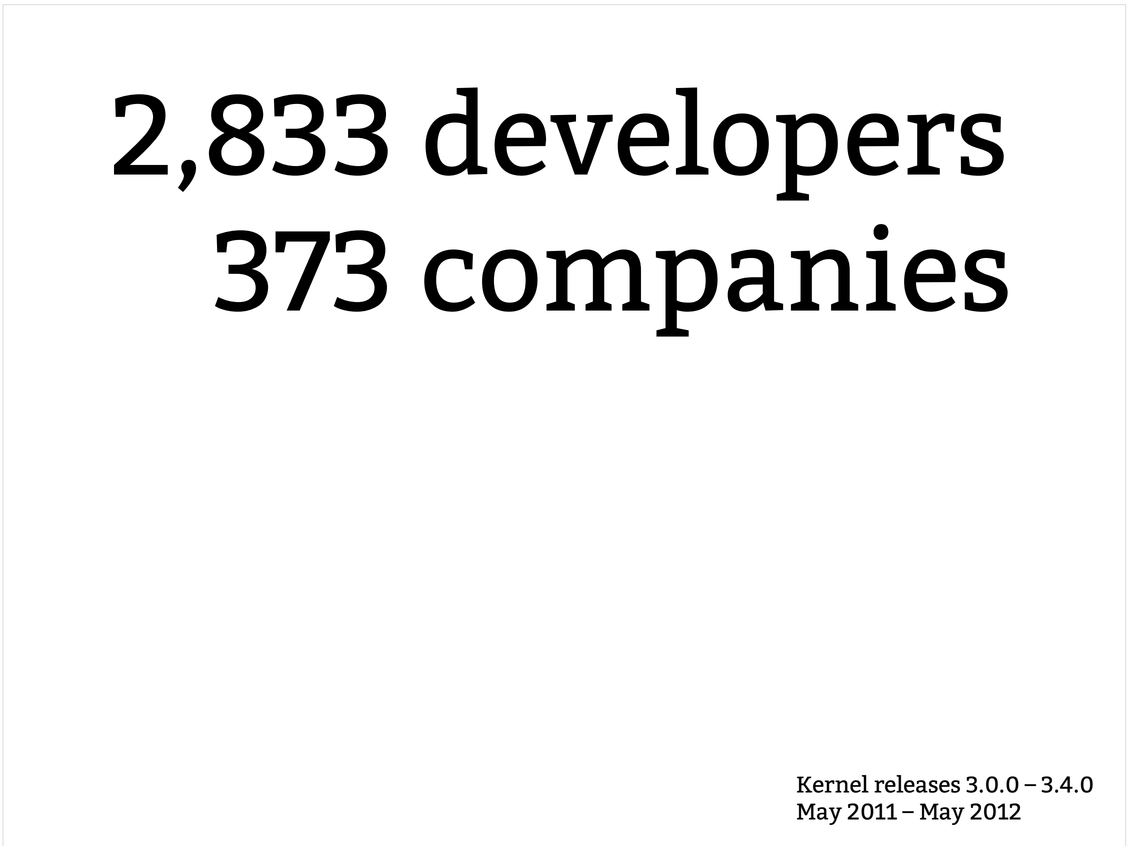
# Linux Kernel Maintainers

## Why are they so grumpy?

Greg Kroah-Hartman
gregkh@linuxfoudation.org

I'm going to discuss the how fast the kernel is moving, how we do it all, and how you can get involved.

> “Publicly making fun of people is half the fun of open source programming.”
>
> – Linus Torvalds

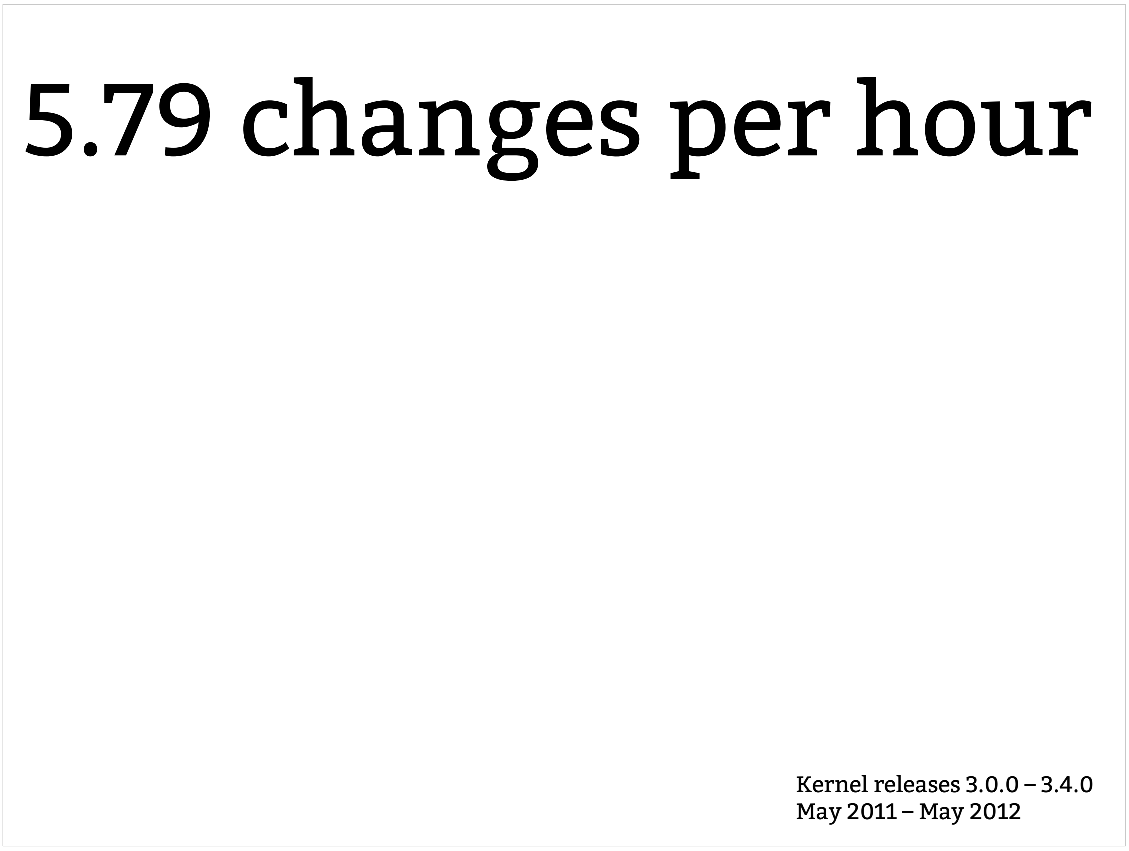


2,833 developers
373 companies

Kernel releases 3.0.0 – 3.4.0
May 2011 – May 2012

This makes the Linux kernel the largest contributed body of software out there that we know of.

This is just the number of companies that we know about, there are more that we do not, and as the responses to our inquiries come in, this number will go up.



# 5.79 changes per hour

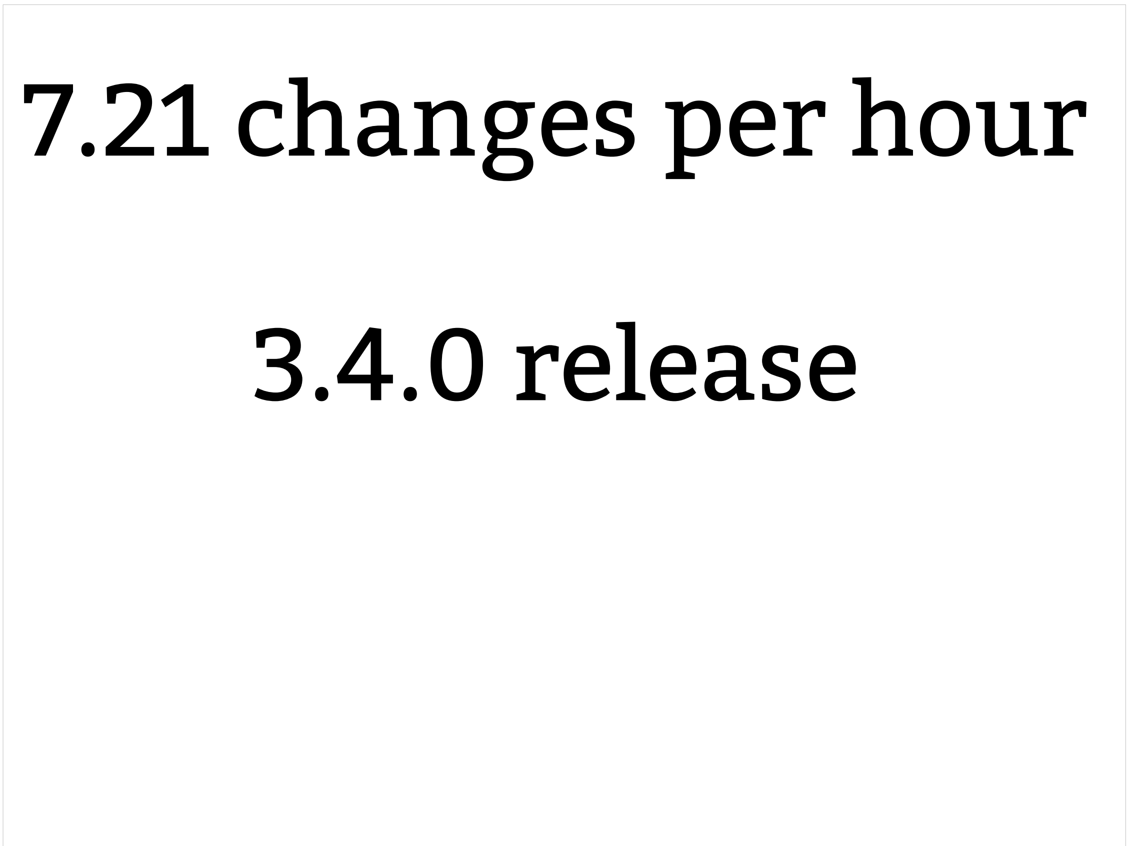
Kernel releases 3.0.0 – 3.4.0
May 2011 – May 2012

This is 24 hours a day, 7 days a week, for a full year.

We went this fast the year before this as well, this is an amazing rate of change.

Interesting note, all of these changes are all through the whole kernel.

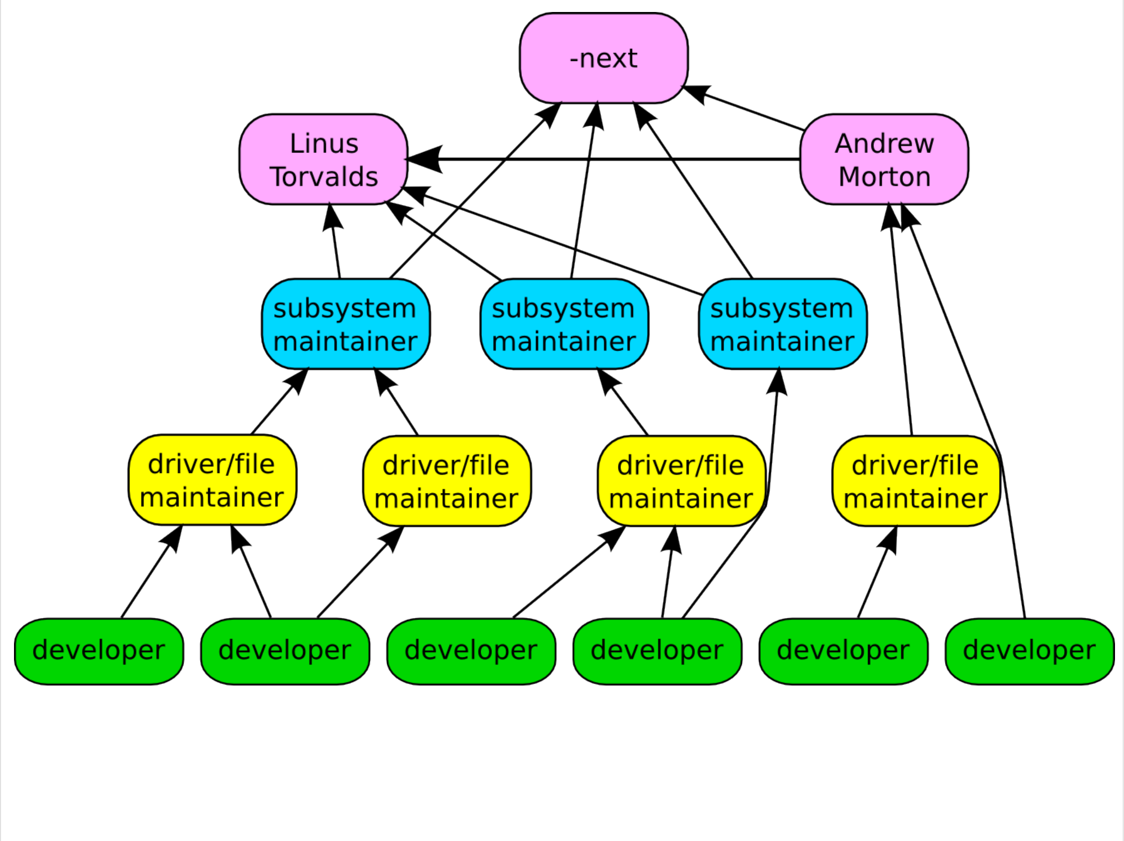
For example, the core kernel is only 5% of the code, and 5% of the change was to the core kernel. Drivers are 55%, and 55% was done to them, it's completely proportional all across the whole kernel.



7.21 changes per hour

3.4.0 release

The 3.4 kernel release was the fastest one ever created



Every 3 months, when the merge window opens up, everything gets sent to Linus from the subsystem maintainers and Andrew Morton.

The merge window is 2 weeks long, and thousands of patches get merged in that short time.

All of the patches merged to Linus should have been in the linux-next release, but that isn't always the case for various reasons.

Linux-next can not just be sent to Linus as there are things in there that sometimes are not good enough to be merged just yet, it is up to the individual subsystem maintainer to decide what to merge.

# Patches I received in the past 2 weeks

Patches I received in the past 2 weeks

480

# Subject: [PATCH 48/48] ...

# 15 patch series, no order given

# Patches 1, 3-10

# “Signed-off-by:” in signature

Signature saying email was confidential

# Tabs were converted to spaces

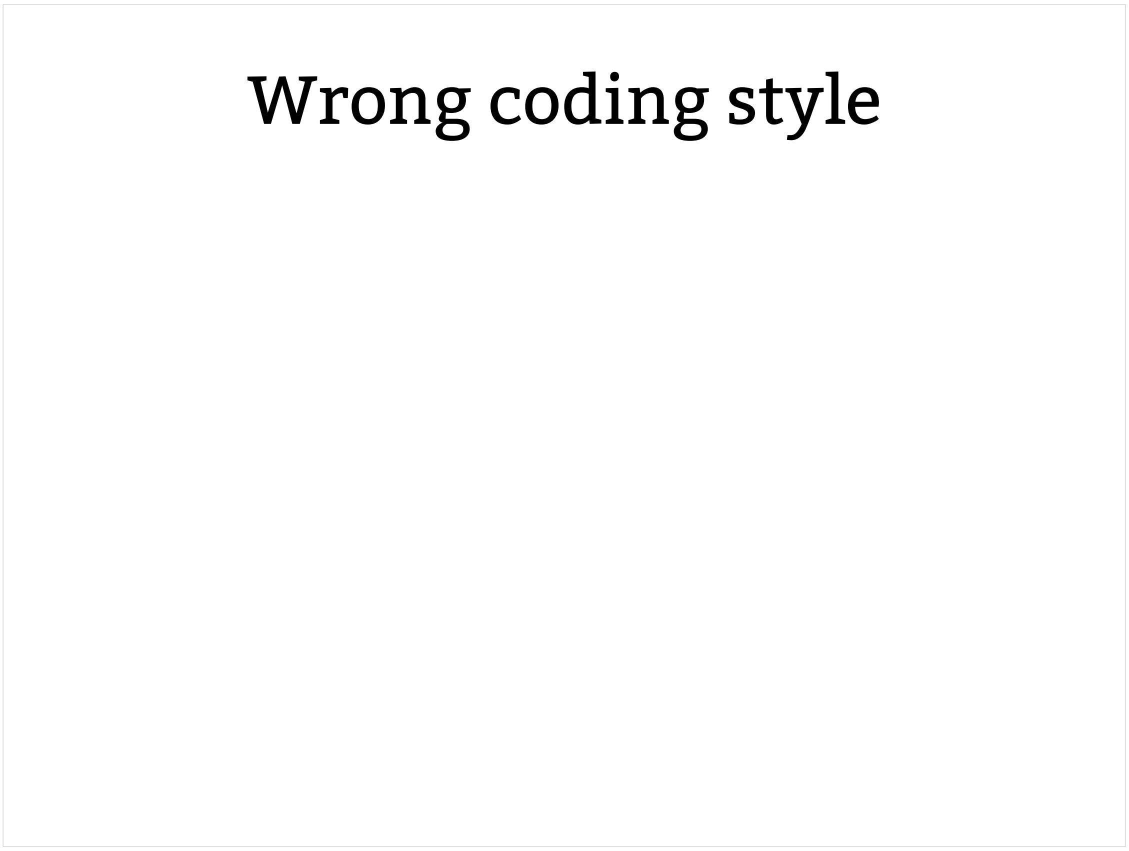
# Leading spaces removed

# diff in non-unified format

# Patch created in driver directory

# Patch created in /usr/src/linux-2.6.32

Made against different tree



# Wrong coding style

# Wrong coding style, and acknowledged it

Would not compile

# Broke the build on patch 3/6

# Broke the build on patch 3/6 and fixed it on 6/6

# Broke the build on patch 5/8

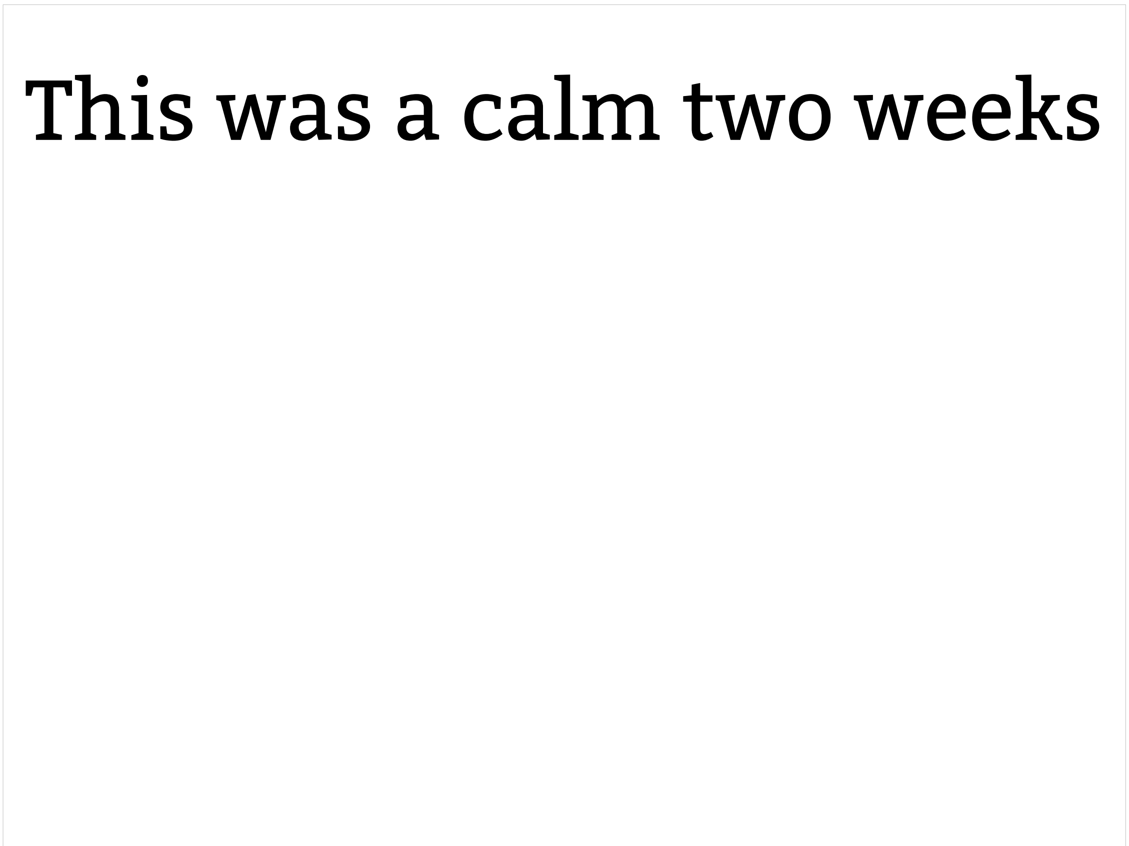
# Broke the build on patch 5/8

Contained note that fix would be sent later

# Patches that had nothing to do with me

# 1 patch, 450kb big (4500 lines added)

# Obviously wrong kerneldoc

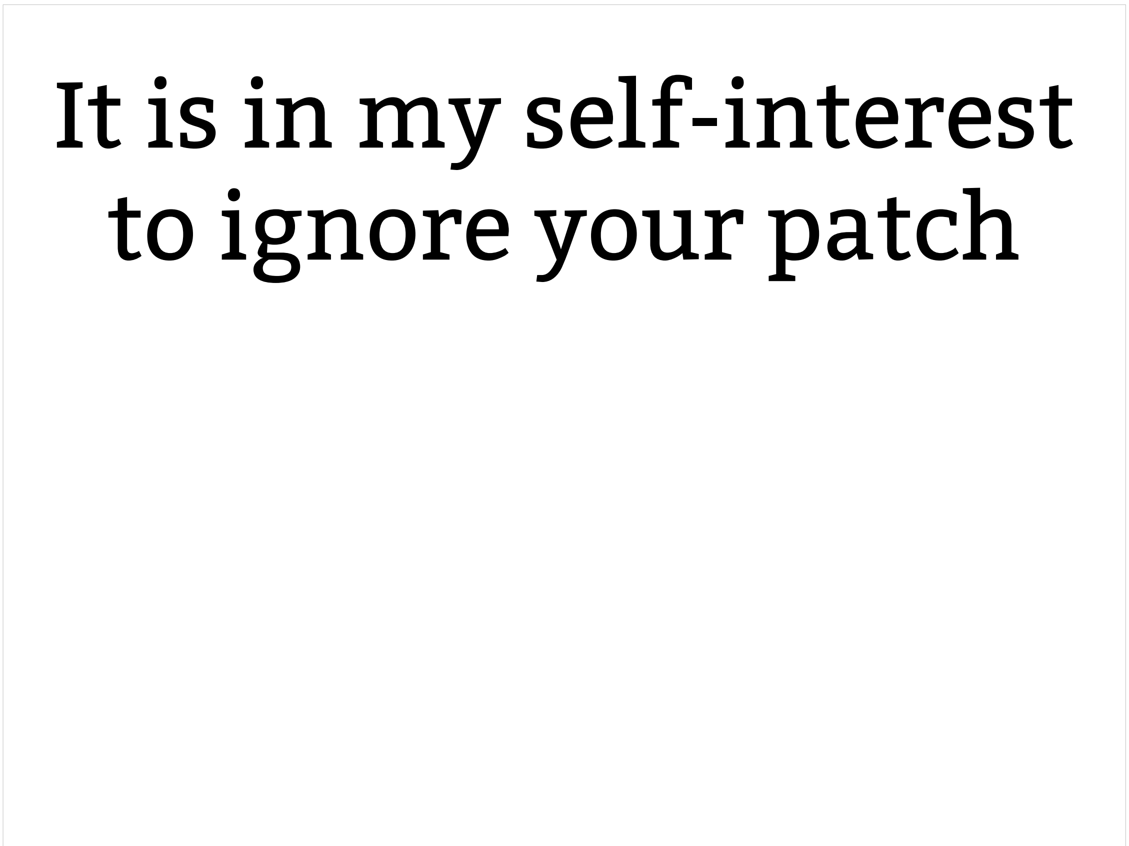


# This was a calm two weeks

Now, I'm not asking you to take pity on me, just realize that this is the level of incompetence that every single one of those 700 developers encounter on a constant  basis.

So when you think we are acting grumpy, remember, how would you act if you had to deal with this all of the time?

Let's get back to what the goal is here.  You want to create a patch that is accepted as it does something that you want to do in Linux.  The maintainer wants to reject it.

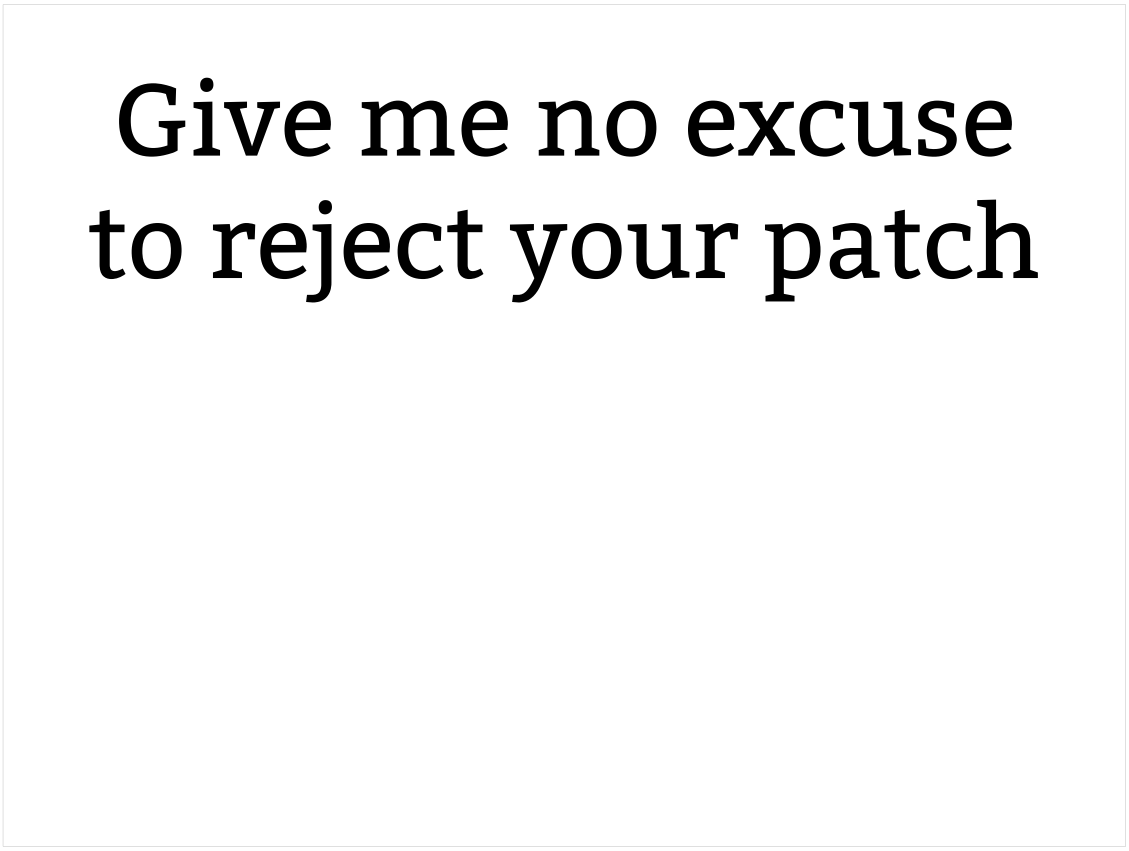


# It is in my self-interest to ignore your patch

Seriously.  It's easier for the maintainer to not accept your code at all. To accept it, it takes time to review it, apply it, send it on up the development chain, handle any problems that might happen with the patch, accept responsibility for the patch, possibly fix any problems that happen later on when you disappear, and maintain it for the next 20 years.

That's a lot of work that you are asking someone else to do on your behalf. You are asking someone who doesn't usually work for your company, who probably lives in a different country, who you have never met in person, to assume responsibility for your work, and to do extra work on top of the normal work they do in the kernel every day.

So you can see how it's in my interest to ignore your patch.  And it's in your interest to keep me from ignoring it, because you want it accepted.



# Give me no excuse to reject your patch

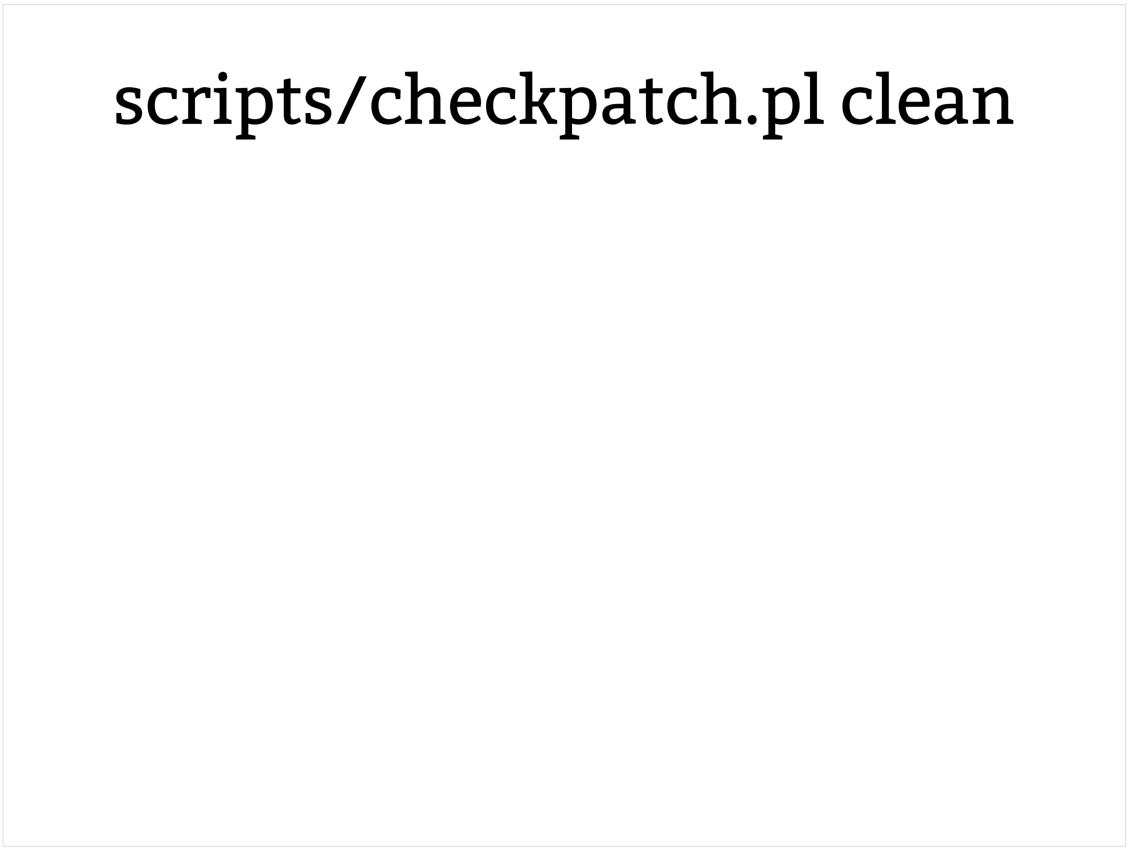
So your goal is, when sending a patch, is to give me NO excuse to not accept it.  To make it such that if I ignore it, or reject it, I am the one that is the problem here, not you.

What can you do to keep me from rejecting your patch outright

.

First off, don't do any of the things I listed above, that's obvious, right?  But that's a "do not do" list, how about a list of what to do:

# Proper coding style



# scripts/checkpatch.pl clean

# Sent to proper people and lists

# Sent to proper people and lists

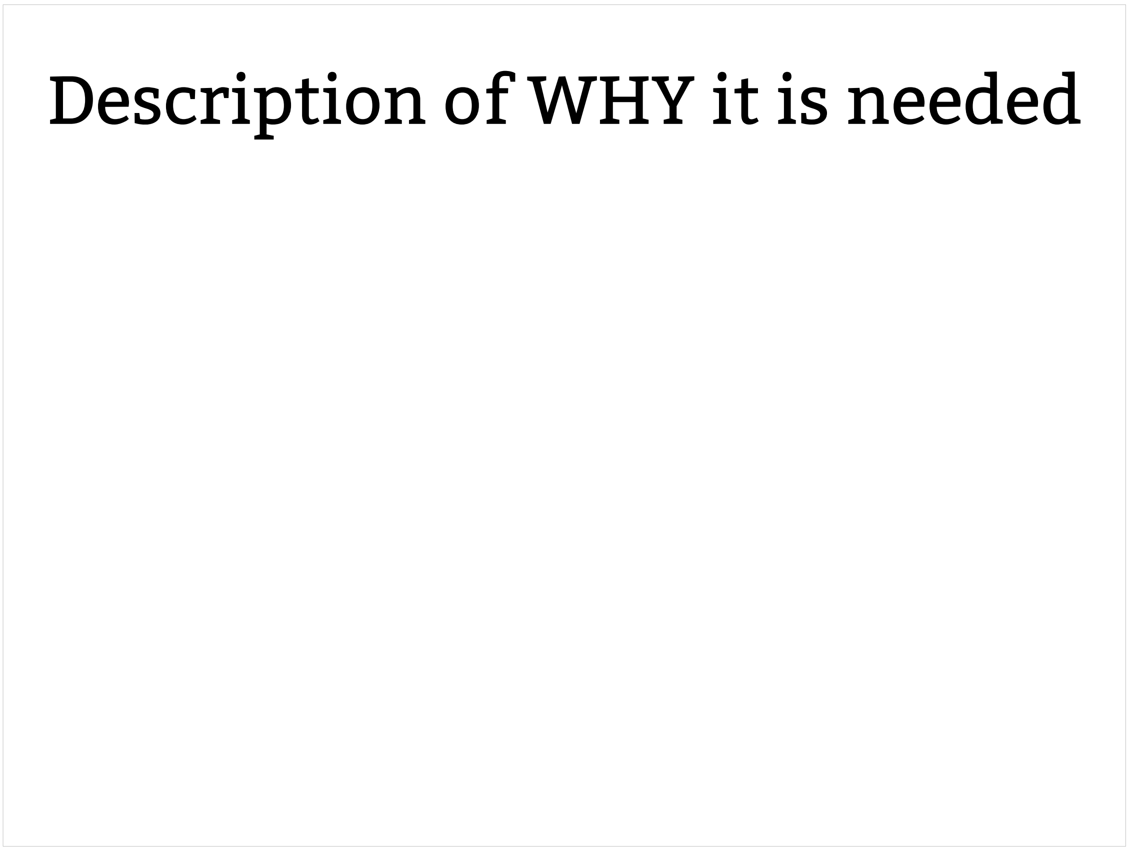
scripts/get_maintainer.pl

# Proper Subject:

# Proper changelog comment

# “obviously” correct

# Small incremental change



# Description of WHY it is needed

# Which tree it was made against

# If multiple patches, state the order

# Has to build properly

# Make sure it works, if possible

# Don't ignore review comments

# Don't resend without saying why

# What I will do for you:

# Review your patch within 1-2 weeks

# Offer semi-constructive criticism

# Let you know the status of your patch

Obligatory Penguin Picture

[redacted]