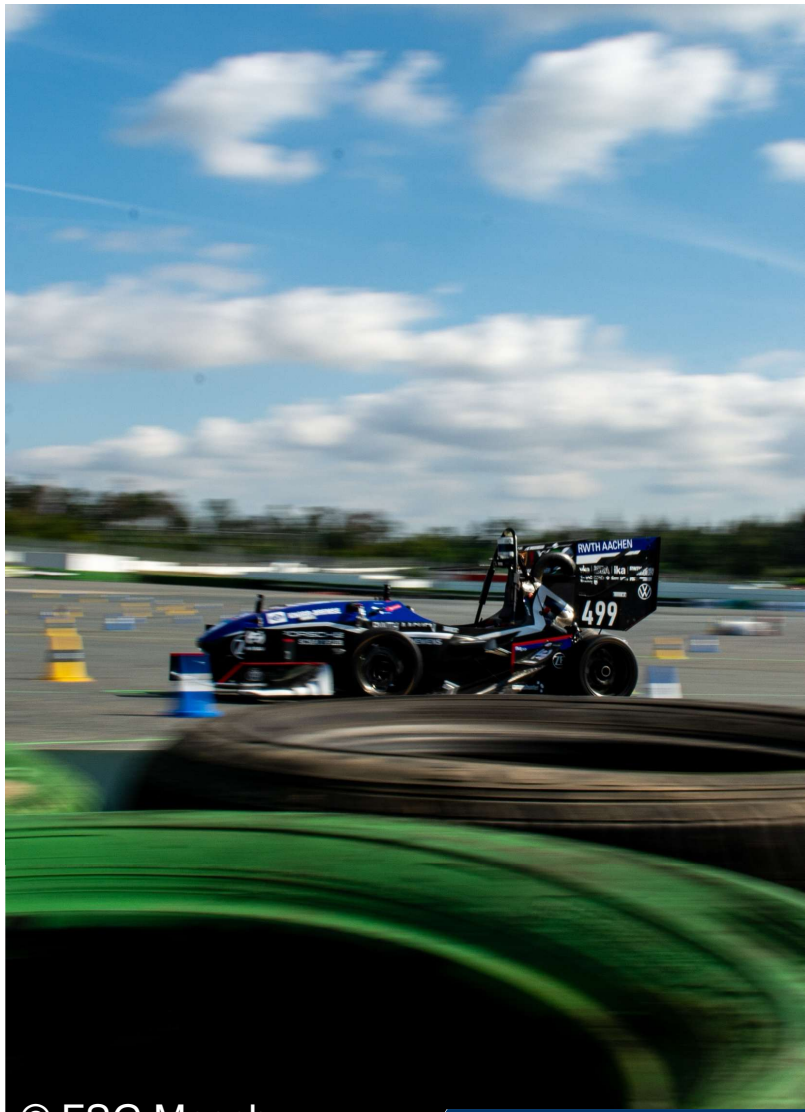

ecurieaix
RWTH AACHEN
699
RWTH AACHEN
SIEMENS
HUMBEL
KNORR-BREMSE
SCHAEFFLER
BOSCH
PORSCHE
aconity3D
bertrandt
SCHULTE
HSM
HARWIN
VECTOR
CNC T.H
dSPACE
WENU
Learning from Industry Standards in Collaborative Software Development
ecurieaix
RWTH AACHEN

OVERVIEW

Topics

- Using your Git correctly
- Documentation 101
- How to turn good code into great code
- To UnitTest or not to UnitTest, that is the question
- Simulating for Success
- Bringing it all together: Continuous Integration

How to Git

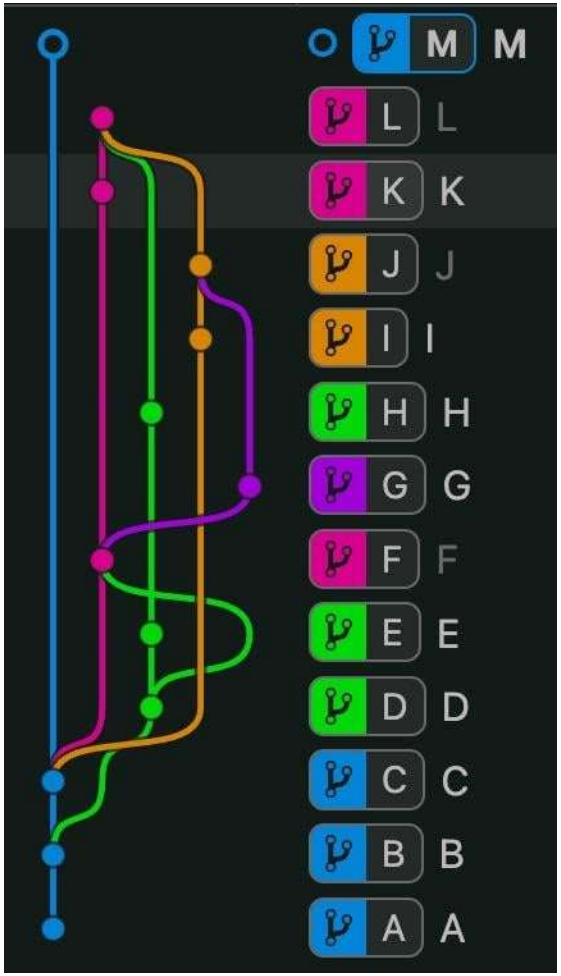


© FSG Mosch

Why a VCS (and why Git)?

- A **VCS** enables you to **work on multiple versions**
  - Allows **snapshotting** and **working on multiple features**
  - **Mark versions as important** or as less important
  - **Find regression bugs quickly** by checking the history

- Git is, by far, the most popular VCS
  - **Distributed, Decentralized, Fast**
  - Incredibly versatile, running your own server takes less than 2 minutes to set up
  - Easy to learn, hard to master



Why Git, specifically?

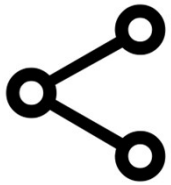
Incredibly Fast

Customizable

Lightweight

Features you (probably) didn‘t know

- **Hooks**
  - Git enables you to write custom hooks
  - Demo: Only allow commits when compile successful
    Demo: How to override the commit hook (using --no-verify)
- **Bisect**
  - Useful to catch where regressions come from
- **Merging unrelated histories**
  - You got multiple git projects on accident? Git can still bring them together if told so!
- **Filter Trees**
  - Download partial project
  - Useful if the project got too large or big files got pushed on accident

A simple Git Hook

- Inside the .git folder git stores its own data
  - Inside the hooks folder there's your hooks

- Hooks are just scripts that can be executed!
  - That means they are super easy to edit and to adapt to your needs.

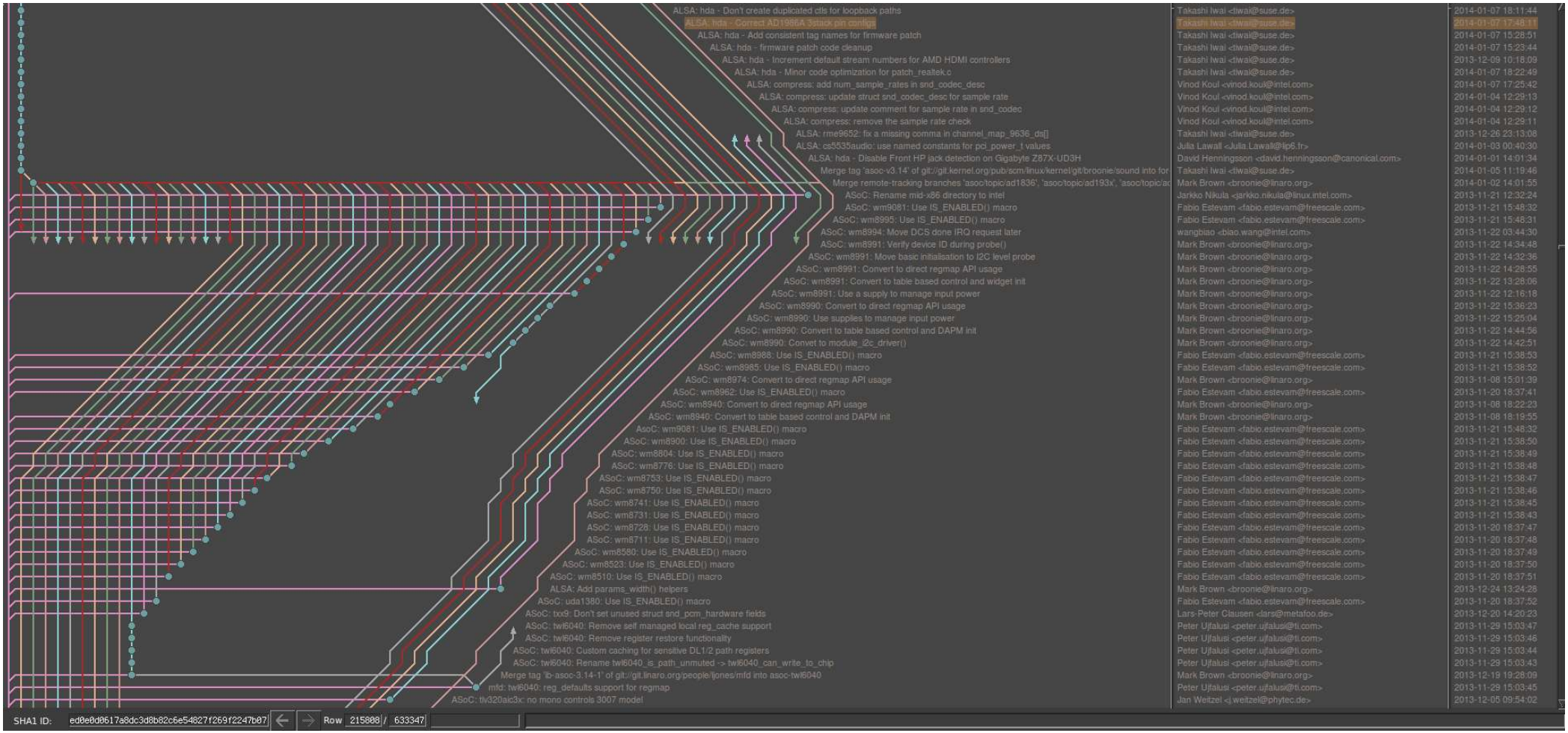
What your Merges shouldn‘t look like

21st of January 2014 @ Linux Kernel Mailing List

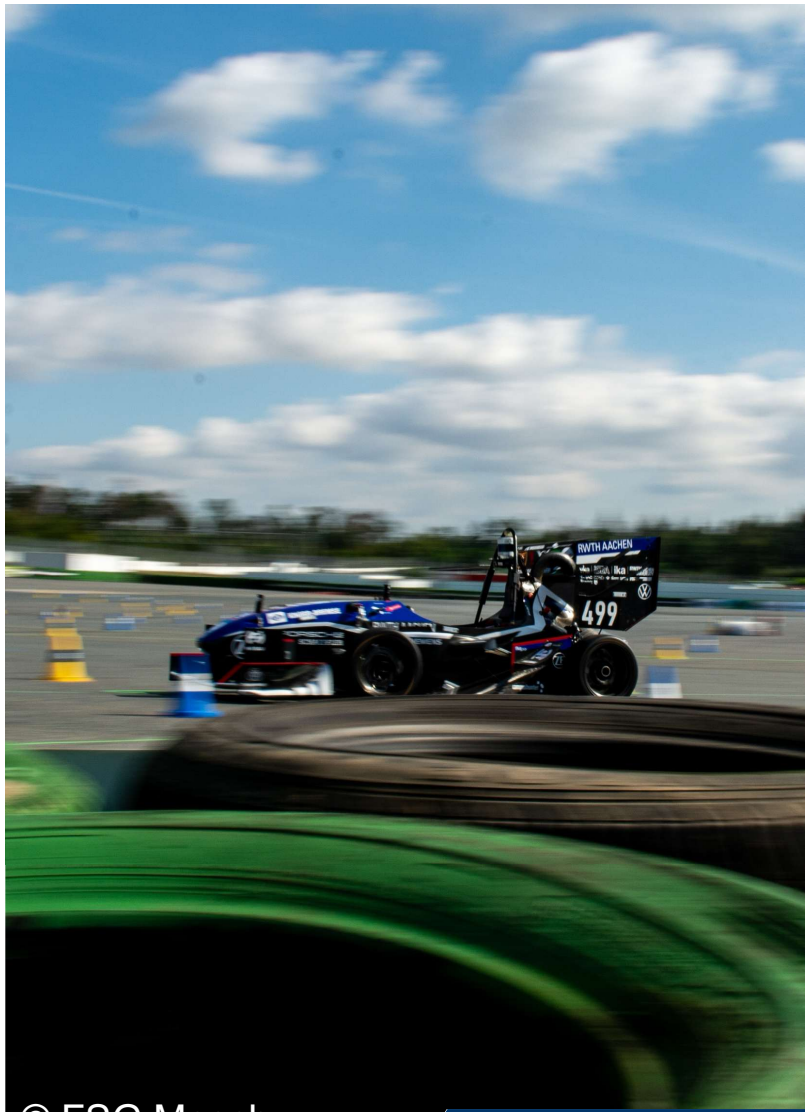
> If you're OK with octopus merges for things like this I'll definitely
> take another look at using them, the enormous stack of merge commits
> always looks noisy to me in the logs and pull requests and for things
> like driver updates there's unlikely to be much doubt about which branch
> it was if there's a problem.

Christ. When you start doing octopus merges,
you don't do it by half measures, do you? I just pulled the sound updates
from Takashi, and as a result got your merge commit 2cde51fbd0f3.
That one has 66 parents.
[...]
It's pulled, and it's fine, but **there's clearly a balance between**
"octopus merges are fine" and
"Christ, that's not an octopus, that's a Cthulhu merge".

--- Linus Torvalds


ALSA: hda - Don't create duplicated ctls for loopback paths
ALSA: hda - Correct AD1986A 3stack pin configs
ALSA: hda - Add consistent tag names for firmware patch
ALSA: hda - firmware patch code cleanup
ALSA: hda - Increment default stream numbers for AMD HDMI controllers
ALSA: hda - Minor code optimization for patch_realtek.c
ALSA: compress: add num_sample_rates in snd_codec_desc
ALSA: compress: update struct snd_codec_desc for sample rate
ALSA: compress: update comment for sample rate in snd_codec
ALSA: compress: remove the sample rate check
ALSA: rme9652: fix a missing comma in channel_map_9636_ds[]
ALSA: cs5535audio: use named constants for pci_power_t values
ALSA: hda - Disable Front HP jack detection on Gigabyte Z87X-UD3H
Merge tag 'asoc-v3.14' of git://git.kernel.org/pub/scm/linux/kernel/git/broonie/sound into for
Merge remote-tracking branches 'asoc/topic/ad1836', 'asoc/topic/ad193x', 'asoc/topic/a
ASoC: Rename mid-x86 directory to intel
ASoC: wm9081: Use IS_ENABLED() macro
ASoC: wm8995: Use IS_ENABLED() macro
ASoC: wm8994: Move DCS done IRQ request later
ASoC: wm8991: Verify device ID during probe()
ASoC: wm8991: Move basic initialisation to I2C level probe
ASoC: wm8991: Convert to direct regmap API usage
ASoC: wm8991: Convert to table based control and widget init
ASoC: wm8991: Use a supply to manage input power
ASoC: wm8990: Convert to direct regmap API usage
ASoC: wm8990: Use supplies to manage input power
ASoC: wm8990: Convert to table based control and DAPM init
ASoC: wm8990: Convet to module_i2c_driver()
ASoC: wm8988: Use IS_ENABLED() macro
ASoC: wm8985: Use IS_ENABLED() macro
ASoC: wm8974: Convert to direct regmap API usage
ASoC: wm8962: Use IS_ENABLED() macro
ASoC: wm8940: Convert to direct regmap API usage
ASoC: wm8940: Convert to table based control and DAPM init
AsoC: wm9081: Use IS_ENABLED() macro
ASoC: wm8900: Use IS_ENABLED() macro
ASoC: wm8804: Use IS_ENABLED() macro
ASoC: wm8776: Use IS_ENABLED() macro
ASoC: wm8753: Use IS_ENABLED() macro
ASoC: wm8750: Use IS_ENABLED() macro
ASoC: wm8741: Use IS_ENABLED() macro
ASoC: wm8731: Use IS_ENABLED() macro
ASoC: wm8728: Use IS_ENABLED() macro
ASoC: wm8711: Use IS_ENABLED() macro
ASoC: wm8580: Use IS_ENABLED() macro
ASoC: wm8523: Use IS_ENABLED() macro
ASoC: wm8510: Use IS_ENABLED() macro
ALSA: Add params_width() helpers
ASoC: uda1380: Use IS_ENABLED() macro
ASoC: txx9: Don't set unused struct snd_pcm_hardware fields
ASoC: twl6040: Remove self managed local reg_cache support
ASoC: twl6040: Remove register restore functionality
ASoC: twl6040: Custom caching for sensitive DL1/2 path registers
ASoC: twl6040: Rename twl6040_is_path_unmuted -> twl6040_can_write_to_chip
Merge tag 'lb-asoc-3.14-1' of git://git.linaro.org/people/ljones/mfd into asoc-twl6040
mfd: twl6040: reg_defaults support for regmap
ASoC: tlv320aic3x: no mono controls 3007 model
Takashi Iwai <tiwai@suse.de> 2014-01-07 18:11:44
Takashi Iwai <tiwai@suse.de> 2014-01-07 17:48:11
Takashi Iwai <tiwai@suse.de> 2014-01-07 15:28:51
Takashi Iwai <tiwai@suse.de> 2014-01-07 15:23:44
Takashi Iwai <tiwai@suse.de> 2013-12-09 10:18:09
Takashi Iwai <tiwai@suse.de> 2014-01-07 18:22:49
Vinod Koul <vinod.koul@intel.com> 2014-01-07 17:25:42
Vinod Koul <vinod.koul@intel.com> 2014-01-04 12:29:13
Vinod Koul <vinod.koul@intel.com> 2014-01-04 12:29:12
Vinod Koul <vinod.koul@intel.com> 2014-01-04 12:29:11
Takashi Iwai <tiwai@suse.de> 2013-12-26 23:13:08
Julia Lawall <Julia.Lawall@lip6.fr> 2014-01-03 00:40:30
David Henningsson <david.henningsson@canonical.com> 2014-01-01 14:01:34
Takashi Iwai <tiwai@suse.de> 2014-01-05 11:19:46
Mark Brown <broonie@linaro.org> 2014-01-02 14:01:55
Jarkko Nikula <jarkko.nikula@linux.intel.com> 2013-11-21 12:32:24
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:48:32
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:48:31
wangbiao <biao.wang@intel.com> 2013-11-22 03:44:30
Mark Brown <broonie@linaro.org> 2013-11-22 14:34:48
Mark Brown <broonie@linaro.org> 2013-11-22 14:32:36
Mark Brown <broonie@linaro.org> 2013-11-22 14:28:55
Mark Brown <broonie@linaro.org> 2013-11-22 13:28:06
Mark Brown <broonie@linaro.org> 2013-11-22 12:16:18
Mark Brown <broonie@linaro.org> 2013-11-22 15:36:23
Mark Brown <broonie@linaro.org> 2013-11-22 15:25:04
Mark Brown <broonie@linaro.org> 2013-11-22 14:44:56
Mark Brown <broonie@linaro.org> 2013-11-22 14:42:51
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:38:53
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:38:52
Mark Brown <broonie@linaro.org> 2013-11-08 15:01:39
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-20 18:37:41
Mark Brown <broonie@linaro.org> 2013-11-08 18:22:23
Mark Brown <broonie@linaro.org> 2013-11-08 18:19:55
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:48:32
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:38:50
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:38:49
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:38:48
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:38:47
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:38:46
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:38:45
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-21 15:38:43
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-20 18:37:47
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-20 18:37:48
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-20 18:37:49
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-20 18:37:50
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-20 18:37:51
Mark Brown <broonie@linaro.org> 2013-12-24 13:24:28
Fabio Estevam <fabio.estevam@freescale.com> 2013-11-20 18:37:52
Lars-Peter Clausen <lars@metafoo.de> 2013-12-20 14:20:23
Peter Ujfalusi <peter.ujfalusi@ti.com> 2013-11-29 15:03:47
Peter Ujfalusi <peter.ujfalusi@ti.com> 2013-11-29 15:03:46
Peter Ujfalusi <peter.ujfalusi@ti.com> 2013-11-29 15:03:44
Peter Ujfalusi <peter.ujfalusi@ti.com> 2013-11-29 15:03:43
Mark Brown <broonie@linaro.org> 2013-12-19 19:28:09
Peter Ujfalusi <peter.ujfalusi@ti.com> 2013-11-29 15:03:45
Jan Weitzel <j.weitzel@phytec.de> 2013-12-05 09:54:02
SHA1 ID: ed0e0d0617a8dc3d8b82c6e54627f269f2247b07
Row 215808 / 633347

Documentation 101



© FSG Mosch

What to document

- **Document ideas, not execution**
  - It is nice to have the execution explained, but this is rarely useful
    - If you have to explain your execution, you wrote bad code.
- **Document your intended behaviour**
  - Tell what it does, not how it does
- **Document (maybe) unexpected behaviour**
  - Example: Is the return value of -1 a bug or a feature? What does it mean?
- **Document known bugs**
  - You'll save others and yourself a lot of headaches.

What (not) to document

- **Don‘t explain your execution**
  - If you have to explain your execution, you wrote bad code
- **Documentation != Comments**
  - Documentation describes how your code is used, not what it does!

- **Comments don‘t get updated (usually)**
  - Outdated comments will persist into future versions
  - The compiler does not check comments for their correctness!
- **The more comments, the more confusion**
  - In most circumstances you don‘t even need comments

How to document

- **Choose one** approach to documentation and **stick to it**
  - More resources => More likely to get outdated
  - **Example:**
    - Documentation for high level overview
      - Recommendation: Create a „Concepts“´page and explain how your design works on a high level
      - Ideal onboarding resource for your new members
    - Comment known bugs and unexpected behaviour inline in your code
    - Use Git Issues for otherwise discovered bugs and project planning
- **Pick one flavor**
  - **Example:**
    - Use Doxygen syntax
    - Document how your code is used in a dedicated README.md
    - Comment what a class/function does above it

Example for good docs:
TigerBeetle

Improve your Code, One Line at a Time

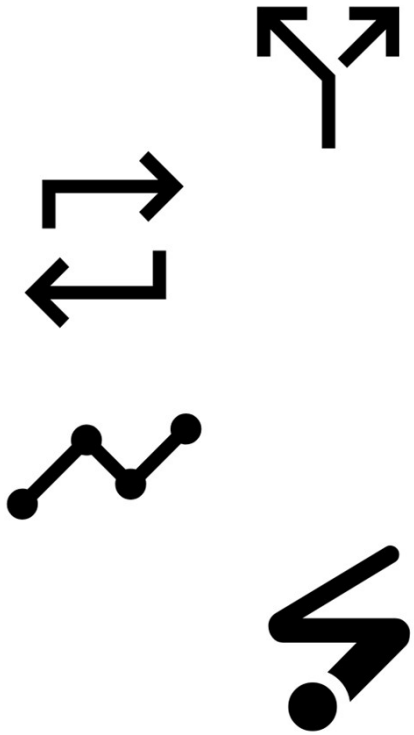
© FSG Mosch

What you can do

- **Pick conventions**
  - Stronger conventions = more consistency = easier to move across the code
    - BUT this isn't free: Stronger conventions = more time to get used to = slower initial output
    - Our experience: Slow to get used to, provide a lot of momentum once used to them
  - Migrate your old code to your new conventions!
    - Use tools like SonarQube (Open Source) to check these conventions
- **Understand your code**
  - If you see something weird, there's probably a reason for it
    - But is that reason still applicable? Challenge your assumptions!
- **Optimize what is needed, when it is needed**
  - "The real problem is that programmers have spent far too much time worrying about efficiency in the wrong places and at the wrong times; premature optimization is the root of all evil [...]"
    – Donald Knuth, "The Art of Computer Programming"

Our Recommendations

- **Split complex functions into reusable chunks**
  - Makes your code much more maintainable and readable
- **Don‘t Repeat Yourself**
  - You won‘t remember to fix the same bug everywhere
- **Avoid Recursion, Jump Labels, Try / Catch and Unbounded Loops**
  - Turn recursion and jumps into loops
  - Instead of Try/Catch, pass an error object with your returned value
  - Set an upper boundary to all loops
- **Pointers may not point to pointers**
  - Only allow direct dereferences

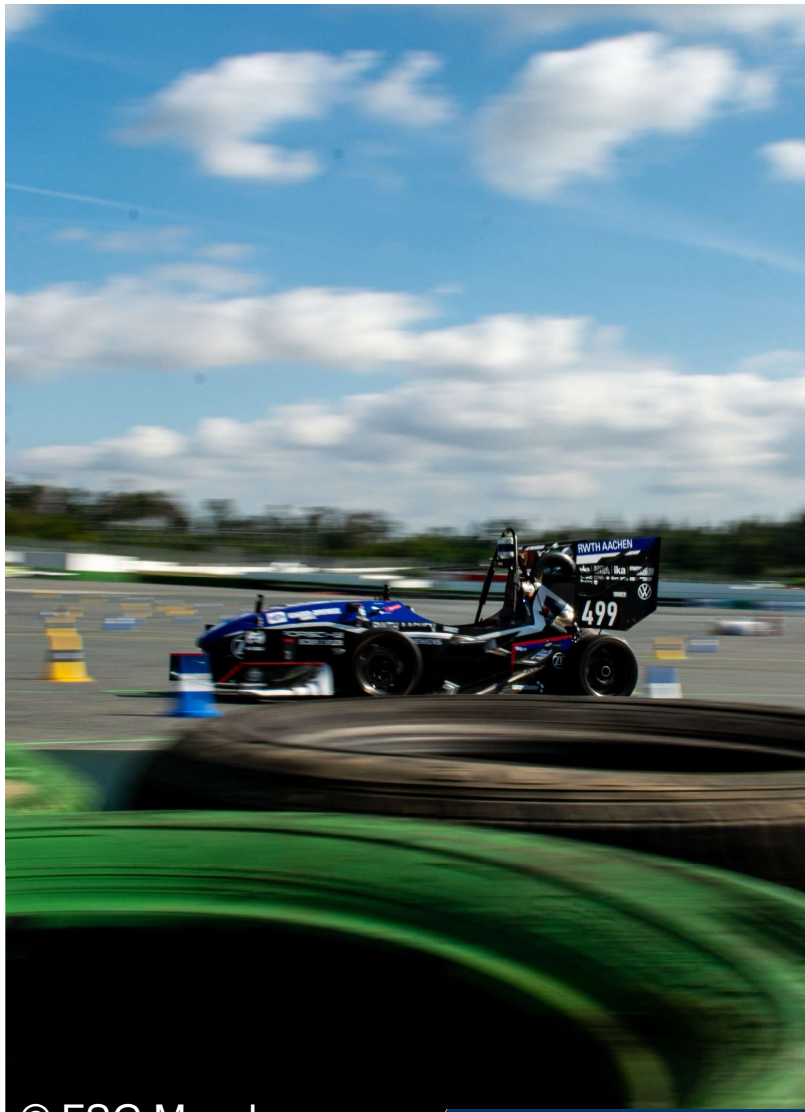


Futher Resources

NASA's „Power of Ten"

Tigerbeetle's „Tigerstyle"

Testing, testing, 1... 2... 3!



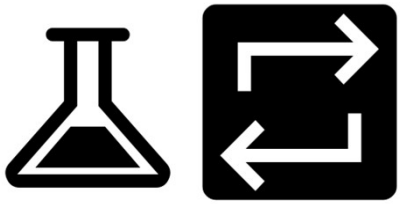
© FSG Mosch

Why Unit Test?

- **Check the DOs and DONTs**
  - Validate well-defined interfaces
  - Validate expected control flows
  - Check for accidental regression
  - Check for discovered bugs for future reference

- **Problem: Unit Tests require strict definition**
  - But we don't always know what we're looking for!
  - Exhaustive validation is expensive
    - Checking your entire model is almost impossible

Why FuzzTest?

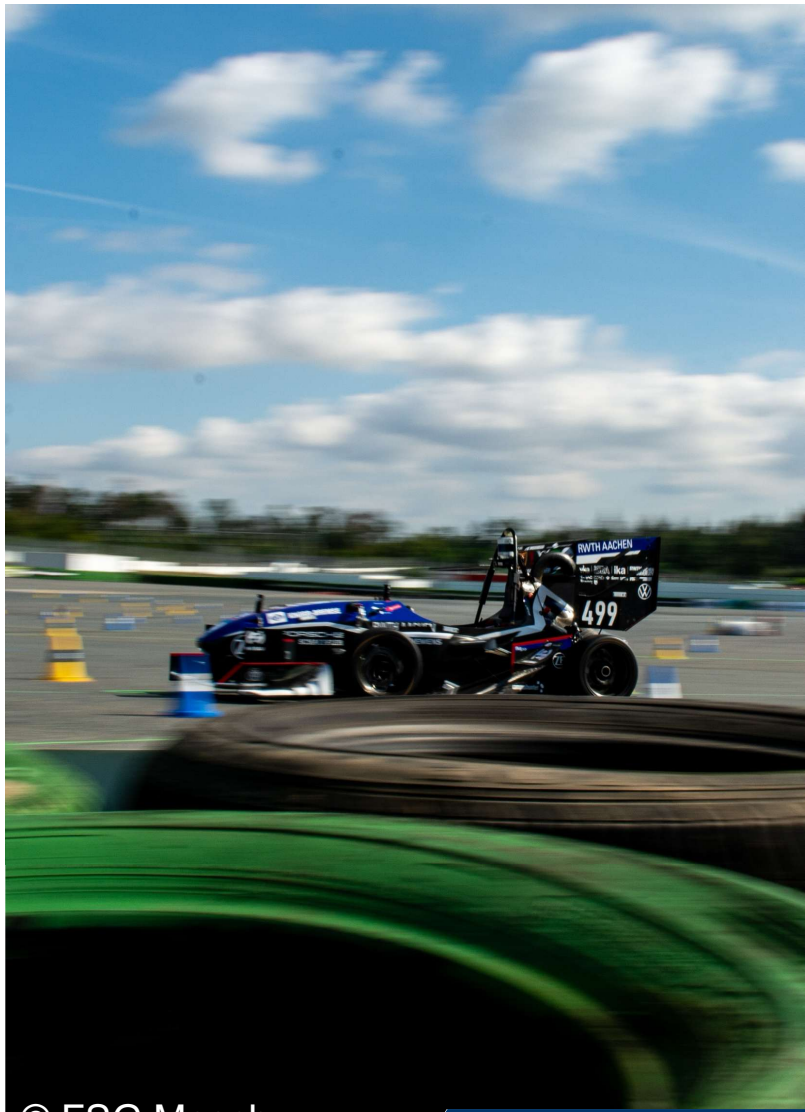


- **FuzzTests expect the unexpected**
  - Run your function against random inputs and see how it behaves
  - Goal: Identify scenarios that shouldn't occur

- **Fuzz Testing is fast and easy to setup**
  - Run your function against a random input and compare it with expected output
  - Usually a function can be expressed in multiple ways
    - Example: An optimized version of a function vs its old definition

- **Fuzz Testing can be used to benchmark**
  - Catch performance regressions easily before something happens

Simulating for Success



© FSG Mosch

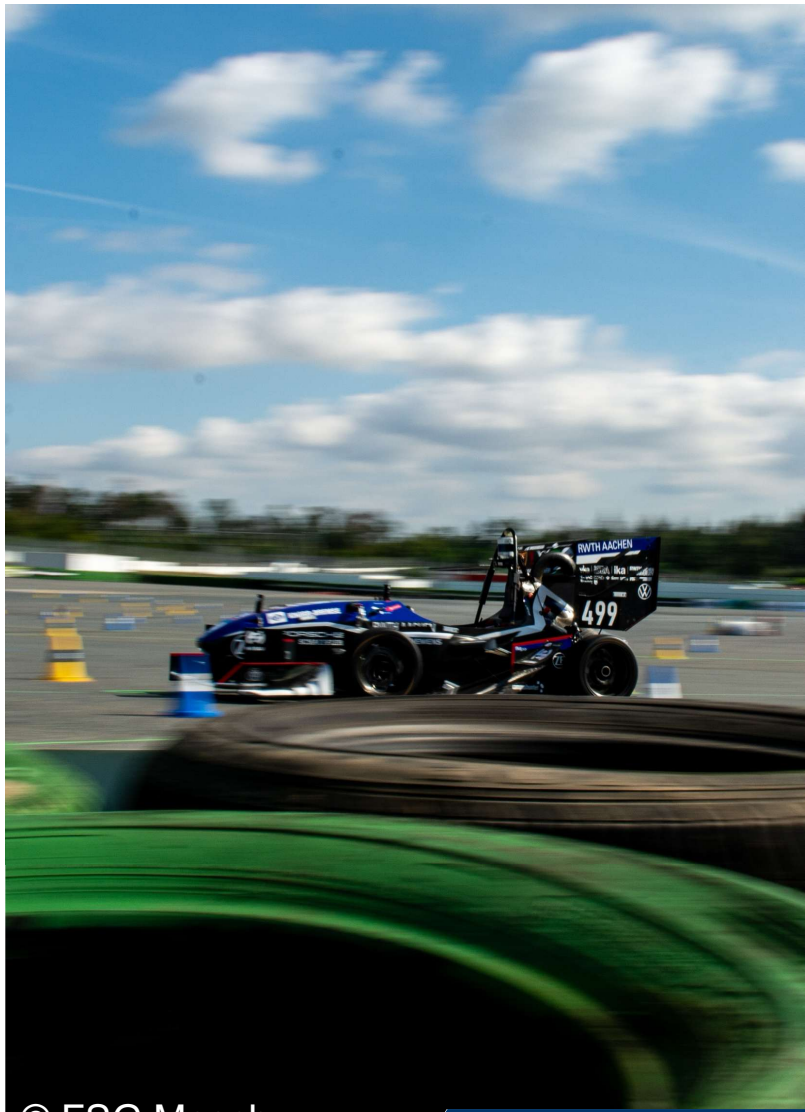
Why simulate?

- **Best representation of the real deal**
  - Fuzzing and Unit Testing only gets you so far
  - Allows you to see the entire stack in action

- **Get feedback on next-to-real data**
  - Turn your merges into more than just snapshots in time

- **Feed the data back to the development process**
  - Automatic evaluation enables identifying potentials early
  - Speed up the development cycle
  - Catch issues as early as possible

Integrate, Deploy, Monitor



© FSG Mosch

Bringing it all together

- **Use your VCS correctly**
  - Perform automatic linting, check if code compiles before push
- **Write good documentation and code**
  - Make it easy to follow your ideas
  - Easy to maintain = Easy to debug
- **Test for the expected AND unexpected**
  - Helps keep the codebase healthy
- **Simulate & Integrate**
  - Get some extra time for free

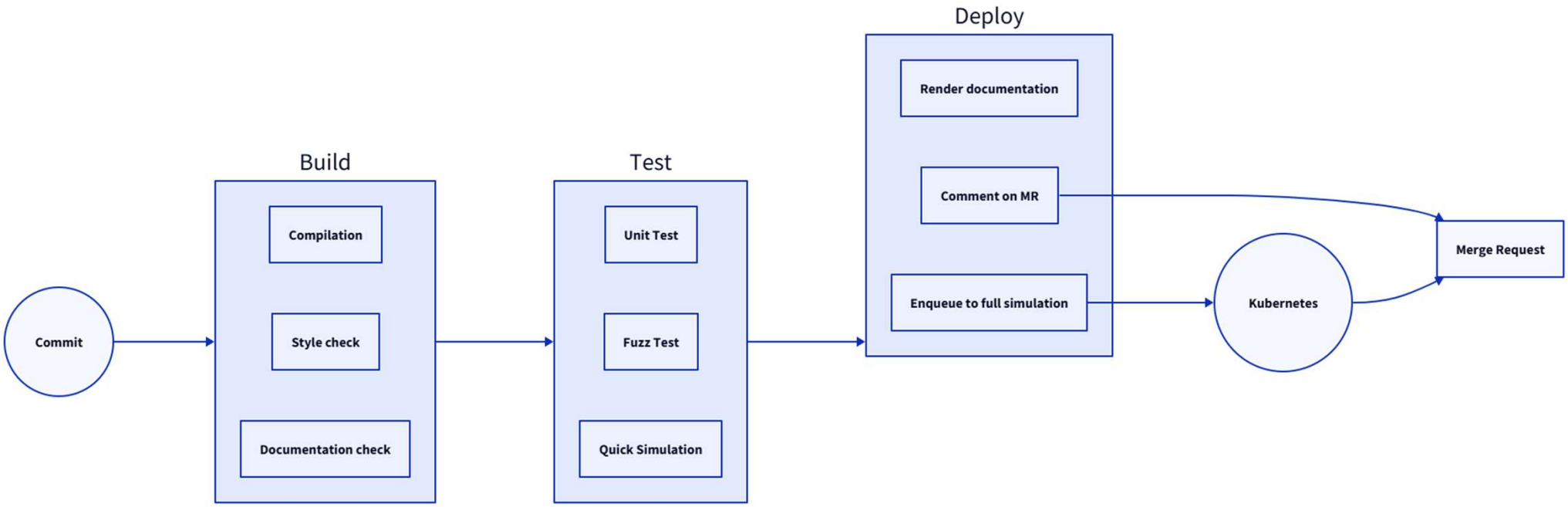
- **Be the most performat you can be.**

How can I do all that easily?

How can I do all that easily?

Continuous Integration

Example Integration



More about CI/CD

We hosted a presentation
dedicated only to CI/CD last year!

If you‘re interested you can find it here: