LLVM developers have been working recently to land their new ThreadSanitizer run-time. The TSan as a reminder is the compiler instrumentation with associated run-time library for being able to detect data races.

ThreadSanitizer is successful at detecting data race conditions even within large and complex code-bases. But unfortunately it's quite burdensome to enable with performance slowing down in the range of 5~15x while the run-time memory overhead can be in the range of 5~10x.

xcrun: error: invalid active developer path - buildVirtual
The PHP Foundation is an effort by 10 key PHP vendors to assure adequate funding to keep the popular scripting language viable.

We have released Qt 5.12.12 today. This is the last release from Qt 5.12 LTS series and the standard support of Qt 5.12 LTS ends in December 2021.

Qt 5.12.12 contains ~ 30 bug fixes compared to the Qt 5.12.11. Please check details about the release from Qt 5.12.12 Release Note.

Note that Qt 5.12 LTS standard support ends in December 2021. It has been quite a long journey with it; big thanks to everyone involved!

The part to note in this definition is that LWJGL provides access to native APIs through Java. That it is a wrapper over the APIs doesn't mean that you should not be familiar with the semantics of the underlying API. Hence to get the most out of LWJGL a good understanding of the native APIs is essential too.

At this point it is important to disambiguate between a library and a framework. LWJGL is a library and as such is low level; it is not a gamedev framework like libgdx (which itself uses LWJGL under the covers!) or a gamedev engine like GoDot which provide higher level of abstractions. For this reason, it is not recommended for novice programmers to start out writing games with it.

And, of course, it is debatable whether Java is a good language for gamedev over the classic value of C++. Some advantages of using Java are its support of multiple operating systems and, of course, the easy learning curve in comparison to C++. Minuses could be garbage collection, performance and a smaller dev community. In any case, it depends on the use case; as they say, choose the best tool for the job at hand.

What happens when a single bit gets corrupted in an image file you cherish? The results can range from absolutely nothing to an imperceptible visual change to a complete loss of the image. The hero image below is somewhere in the middle of the scale; where the top half of
the image is perfect, and the lower half is reduced to meaningless digital noise.

Whether due to mechanical failure or transmission interference like cosmic radiation: bitrot happens. A rotted bit, or flipped bit, is when one bit of RAM or persistent storage unintentionally flips its state between zero and one. You can only do so much to protect your system from random failure. Multiple backups and data verification is the only proven strategy to protect against it.

Traditional JPEG images (referred to as JPEG throughout the article as opposed to JXL), especially with progressive encoding, handle bitrot remarkably well. You might see a single pixel shift its color almost imperceptibly, or one of the encoding layers may shift slightly. The effects are so well understood that you can even find free software that can automatically recover corrupted JPEG photos.

However, the next-generation image formats pack data much more densely than in the legacy image formats. There’s isn’t just less redundancy, but every single bit means more to the complete image. This means the effects of bitrot produce a much greater loss of visual fidelity and decodes to more abstract results. The newer encoding techniques include predictive models that can get thrown off completely by a single bit out of place. The digital hellfire in the lower half of the above hero image is a perfect example of this.

- **OpenFaaS: How to Add Python Requirements and Dependencies - Anto. / Online**

  This guide will show you how to add requirements and dependencies for a Python project using OpenFaaS.

  Python dependencies are software components that your project needs for it to work. You can manually use PyPI (the Python Package Index) to provide packages that you need, but OpenFaaS can automate this for you.

- **Perl Weekly Challenge 140: Add Binary**

- **My Favorite Warnings: experimental | Tom Wyant [blogs.perl.org]**

  Perl has had experimental features ever since I started using it at about version 5.6. These were things that were considered useful, but about which there was doubt -- about their final form, whether a satisfactory implementation existed, or whatever.

  Until Perl 5.18, experimental features were simply documented as experimental. At that point, an experimental warning category was added, with sub-categories experimental::lexical_subs, experimental::lexical_topic, experimental::regex_sets, and experimental::smartmatch.
Most of the features covered by the original Perl 5.18 warning categories were actually introduced in Perl 5.10 as back-ports from Raku (or Perl 6, as it was then called), and not documented as experimental. My impression was that the relevant experimental:: warnings were introduced because the corresponding features were recognized as being more experimental than originally believed. Programmers already familiar with a feature might not notice an extra sentence in the documentation, but they will surely notice if their code starts spitting out experimental warnings.

**Development**

Source URL: [http://www.tuxmachines.org/node/158434](http://www.tuxmachines.org/node/158434)

Links:
3. [https://buildvirtual.net/xcrun-error-invalid-active-developer-path/](https://buildvirtual.net/xcrun-error-invalid-active-developer-path/)
7. [https://www.ctrl.blog/entry/bitrot-avif-jxl-comparison.html](https://www.ctrl.blog/entry/bitrot-avif-jxl-comparison.html)
8. [https://anto.online/code/openfaas-how-to-add-requirements/](https://anto.online/code/openfaas-how-to-add-requirements/)