

## August 2023

45 posts: [7 entries](#), [28 links](#), [10 quotes](#)

### Aug. 1, 2023

#### Run Llama 2 on your own Mac using LLM and Homebrew

[Llama 2](#) is the latest commercially usable openly licensed Large Language Model, released by Meta AI a few weeks ago. I just released a new plugin for [my LLM utility](#) that adds support for Llama 2 and many other [llama-cpp](#) compatible models.

[... [1,423 words](#)]

---

6:56 pm / [homebrew](#), [macos](#), [plugins](#), [projects](#), [ai](#), [generative-ai](#), [llama](#), [local-llms](#), [llms](#), [llm](#), [llama-cpp](#)

---

### Aug. 3, 2023

#### Catching up on the weird world of LLMs

# Catching up on the weird world of LLMs

Simon Willison     [simonwillison.net](https://simonwillison.net)

[fedi.simonwillison.net/@simon](https://fedi.simonwillison.net/@simon) - @simonw

North Bay Python, 30th July 2023

I gave a talk on Sunday at [North Bay Python](#) where I attempted to summarize the last few years of development in the space of LLMs—Large Language Models, the technology behind tools like ChatGPT, Google Bard and Llama 2.

[... [10,489 words](#)]

[2:51 pm](#) / [ethics](#), [python](#), [my-talks](#), [ai](#), [openai](#), [generative-ai](#), [chatgpt](#), [llms](#), [llm](#), [anthropic](#), [claude](#), [annotated-talks](#), [code-interpreter](#), [ai-ethics](#), [coding-agents](#)

---

**Aug. 4, 2023**

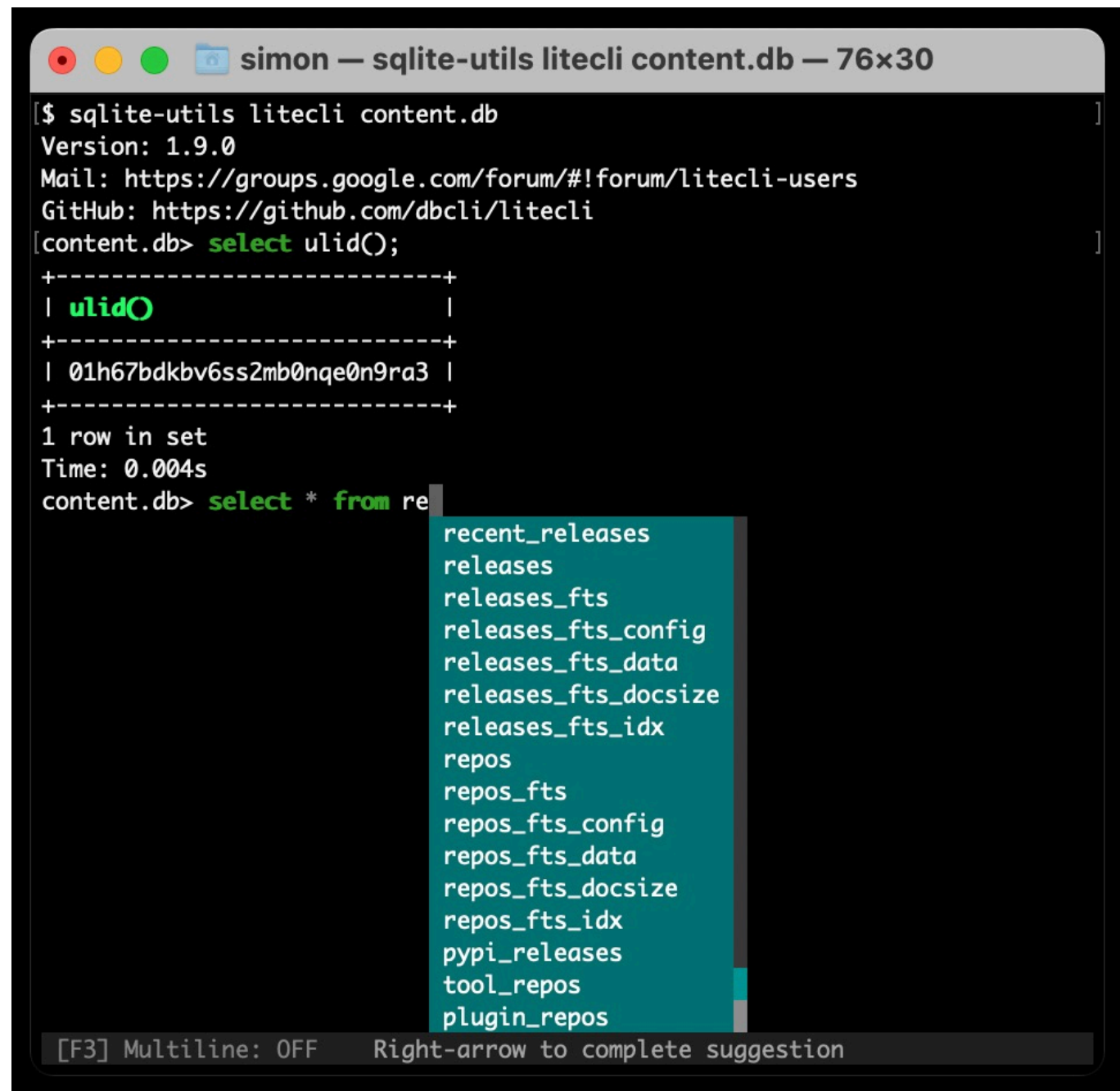
**You can stop using user-scalable=no and maximum-scale=1 in viewport meta tags now.** Luke Plant points out that your meta viewport tag should stick to just “width=device-width, initial-scale=1” these days—the user-scalable=no and maximum-scale=1 attributes are no longer necessary, and have a negative impact on accessibility, especially for Android users.

[# 11:41 pm](#) / [accessibility](#), [html](#), [luke-plant](#), [mobile](#), [mobileweb](#)

---

**Aug. 5, 2023**

**Weeknotes: Plugins for LLM, sqlite-utils and Datasette**



```
simon — sqlite-utils litecli content.db — 76x30
[$ sqlite-utils litecli content.db
Version: 1.9.0
Mail: https://groups.google.com/forum/#!forum/litecli-users
GitHub: https://github.com/dbcli/litecli
content.db> select ulid();
+-----+
| ulid() |
+-----+
| 01h67bdkbv6ss2mb0nqe0n9ra3 |
+-----+
1 row in set
Time: 0.004s
content.db> select * from re
recent_releases
releases
releases_fts
releases_fts_config
releases_fts_data
releases_fts_docsize
releases_fts_idx
repos
repos_fts
repos_fts_config
repos_fts_data
repos_fts_docsize
repos_fts_idx
pypi_releases
tool_repos
plugin_repos
[F3] Multiline: OFF    Right-arrow to complete suggestion
```

The principle theme for the past few weeks has been plugins.

[... [1,203 words](#)]

---

[12:32 am](#) / [cli](#), [plugins](#), [projects](#), [my-talks](#), [datasette](#), [weeknotes](#), [sqlite-utils](#), [llm](#)

---

**[Aug. 6, 2023](#)**

## **[How I make annotated presentations](#)**

# **Annotated presentation creator**

Browse... 89 files selected.

Load images

OCR complete



```
Catching up on the weird
world of LLMs

Simon Willison simonwillison.net
fedi.simonwillison.net/@simon - @simonw

North Bay Python, 30th July 2023
```

Markdown works here

- including bullet points.

Markdown works here

- including bullet points.

Giving a talk is a lot of work. I go by a rule of thumb I learned from [Damian Conway](#): a minimum of ten hours of preparation for every one hour spent on stage.

[... [2,128 words](#)]

---

[5:15 pm](#) / [alt-text](#), [ocr](#), [projects](#), [speaking](#), [my-talks](#), [tools](#), [ai](#), [generative-ai](#), [llms](#), [ai-assisted-programming](#), [anthropic](#), [claude](#), [annotated-talks](#)

---

**[Python cocktail: mix a context manager and an iterator in equal parts](#)** ([via](#)) Explanation of a neat trick used by the Tenacity Python library, which provides a mechanism for retrying a chunk of code automatically on errors up to three times using a mixture of an iterator and a context manager to work around Python's lack of multi-line lambda functions.

[# 5:44 pm](#) / [python](#)

---

## [Aug. 9, 2023](#)

[Llama from scratch \(or how to implement a paper without crying\)](#) ([via](#)) Brian Kitano implemented the model described in the Llama paper against TinyShakespeare, from scratch, using Python and PyTorch. This write-up is fantastic—meticulous, detailed and deeply informative. It would take several hours to fully absorb and follow everything Brian does here but it would provide multiple valuable lessons in understanding how all of this stuff fits together.

[# 7:21 pm](#) / [python](#), [ai](#), [pytorch](#), [generative-ai](#), [llama](#), [llms](#)

---

[Datasette 1.0a3](#). A new Datasette alpha release. This one previews the new default JSON API design that's coming in 1.0—the single most significant change in the 1.0 milestone, since I plan to keep that API stable for many years to come.

[# 8:49 pm](#) / [json](#), [projects](#), [datasette](#)

---

## [Aug. 10, 2023](#)

[Getting creative with embeddings](#) ([via](#)) Amelia Wattenberger describes a neat application of embeddings I haven't seen before: she wanted to build a system that could classify individual sentences in terms of how “concrete” or “abstract” they are. So she generated several example sentences for each of those categories, embedded then and calculated the average of those embeddings.

And now she can get a score for how abstract vs concrete a new sentence is by calculating its embedding and seeing where it falls in the 1500 dimension space between those two other points.

[# 7:05 pm](#) / [ai](#), [generative-ai](#), [llms](#), [embeddings](#), [amelia-wattenberger](#)

---

## [Aug. 11, 2023](#)

[Shamir Secret Sharing](#) ([via](#)) Cracking war story from Max Levchin about the early years of PayPal, in which he introduces an implementation of Shamir Secret Sharing to encrypt their master payment credential table... and then finds that the 3-of-8 passwords needed to decrypt it and bring the site back online don't appear to work.

[# 3:48 pm](#) / [encryption](#), [ops](#), [paypal](#)

---

[Dependency Management Data](#) ([via](#)) This is a really neat CLI tool by Jamie Tanna, built using Go and SQLite but with a feature that embeds a Datasette instance (literally shelling out to start the process running from within the Go application) to provide an interface for browsing the resulting database.

It addresses the challenge of keeping track of the dependencies used across an organization, by gathering them into a SQLite database from a variety of different sources—currently Dependabot, Renovate and some custom AWS tooling.

The “Example” page links to a live Datasette instance and includes video demos of the tool in action.

[# 3:54 pm](#) / [cli](#), [packaging](#), [sqlite](#), [datasette](#)

---

## [Aug. 12, 2023](#)

[llm-mlc](#) ([via](#)) My latest plugin for LLM adds support for models that use the MLC Python library—which is the first library I’ve managed to get to run Llama 2 with GPU acceleration on my M2 Mac laptop.

# 5:33 am / [plugins](#), [projects](#), [ai](#), [generative-ai](#), [llms](#), [mlc](#), [llm](#)

---

[deno\\_python](#) ([via](#)) A wildly impressive hack: deno\_python uses Deno’s FFI interface to load your system’s Python framework (.dll/.dylib/.so) and sets up JavaScript proxy objects for imported Python objects—so you can run JavaScript code that instantiates objects from Python libraries and uses them to process data in different ways.

The latest release added pip support, so things like `'const np = await pip.import("numpy")'` now work.

# 10:14 pm / [python](#), [deno](#)

---

## [Aug. 13, 2023](#)

[Lark parsing library JSON tutorial](#) ([via](#)) A very convincing tutorial for a new-to-me parsing library for Python called Lark.

The tutorial covers building a full JSON parser from scratch, which ends up being just 19 lines of grammar definition code and 15 lines for the transformer to turn that tree into the final JSON.

It then gets into the details of optimization—the default Earley algorithm is quite slow, but swapping that out for a LALR parser (a one-line change) provides a 5x speedup for this particular example.

# 9:50 pm / [compilers](#), [json](#), [parsing](#), [python](#)

---

## [Aug. 14, 2023](#)

[Write about what you learn. It pushes you to understand topics better.](#) ([via](#)) Addy Osmani clearly articulates why writing frequently is such a powerful tool for learning more effectively. This post doesn’t mention TILs but it perfectly encapsulates the value I get from publishing them.

# 2:50 pm / [blogging](#), [writing](#), [til](#), [addy-osmani](#)

---

## [Aug. 15, 2023](#)

Someone asked me today if there was a case for using React in a new app that doesn't need to support IE.

I could not come up with a single reason to prefer it over Preact or (better yet) any of the modern reactive Web Components systems (FAST, Lit, Stencil, etc.).

One of the constraints is that the team wanted to use an existing library of Web Components, but React made it hard. This is probably going to cause them to favour Preact for the bits of the team that want React-flavoured modern webdev.

It's astonishing how antiquated React is.

— [Alex Russell](#)

# 9:15 pm / [web-components](#), [react](#), [javascript](#), [alex-russell](#)

---

**Aug. 16, 2023**

**Welcome to Datasette Cloud**. We launched the Datasette Cloud blog today! The SaaS hosted version of Datasette is ready to start onboarding more users—this post describes what it can do so far and hints at what’s planned to come next.

# [1:46 am](#) / [projects](#), [datasette](#), [datasette-cloud](#)

---

**Introducing datasette-write-ui: a Datasette plugin for editing, inserting, and deleting rows**. Alex García is working with me on Datasette Cloud for the next few months, graciously sponsored by Fly. We will be working in public, releasing open source code and documenting how to build a multi-tenant SaaS product using Fly Machines.

Alex’s first project is datasette-write-ui, a plugin that finally lets you directly edit data stored inside Datasette. Alex wrote about the plugin on our new Datasette Cloud blog.

# [1:48 am](#) / [plugins](#), [datasette](#), [datasette-cloud](#), [fly](#), [alex-garcia](#)

---

llama.cpp surprised many people (myself included) with how quickly you can run large LLMs on small computers [...] TLDR at batch\_size=1 (i.e. just generating a single stream of prediction on your computer), the inference is super duper memory-bound. The on-chip compute units are twiddling their thumbs while sucking model weights through a straw from DRAM. [...] A100: 1935 GB/s memory bandwidth, 1248 TOPS. MacBook M2: 100 GB/s, 7 TFLOPS. The compute is ~200X but the memory bandwidth only ~20X. So the little M2 chip that could will only be about ~20X slower than a mighty A100.

— **Andrej Karpathy**

# [4:13 am](#) / [andrej-karpathy](#), [generative-ai](#), [llama](#), [ai](#), [llms](#), [llama-cpp](#)

---

**An Iowa school district is using ChatGPT to decide which books to ban**. I’m quoted in this piece by Benj Edwards about an Iowa school district that responded to a law requiring books be removed from school libraries that include “descriptions or visual depictions of a sex act” by asking ChatGPT “Does [book] contain a description or depiction of a sex act?”.

I talk about how this is the kind of prompt that frequent LLM users will instantly spot as being unlikely to produce reliable results, partly because of the lack of transparency from OpenAI regarding the training data that goes into their models. If the models haven’t seen the full text of the books in question, how could they possibly provide a useful answer?

# [10:33 pm](#) / [arstechnica](#), [ethics](#), [law](#), [ai](#), [openai](#), [generative-ai](#), [chatgpt](#), [llms](#), [benj-edwards](#), [ai-ethics](#), [press-quotes](#)

---

**Running my own LLM** ([via](#)) Nelson Minar describes running LLMs on his own computer using my LLM tool and llm-gpt4all plugin, plus some notes on trying out some of the other plugins.

# [10:42 pm](#) / [nelson-minar](#), [local-llms](#), [llms](#), [llm](#)

---

**Datasette Cloud, Datasette 1.0a3, llm-mlc and more**

# Create a space

A space is a **private** area where you can import, explore and analyze data and share it with invited collaborators.

**Space name**

**Subdomain**  .datasette.cloud

**Region** Your data will be hosted in a region. Pick somewhere geographically close to you for optimal performance.



- ☐ Ashburn, Virginia (US)
- ☐ Atlanta, Georgia (US)
- ☐ Boston, Massachusetts (US)
- ☐ Chicago, Illinois (US)
- ☐ Dallas, Texas (US)
- ☐ Denver, Colorado (US)
- ☐ Los Angeles, California (US)
- ☐ Miami, Florida (US)
- ☐ Seattle, Washington (US)
- ☐ Secaucus, NJ (US)
- ☐ Sunnyvale, California (US)
- ☐ Amsterdam, Netherlands

Datasette Cloud is now a significant step closer to general availability. The Datasette 1.03 alpha release is out, with a mostly finalized JSON format for 1.0. Plus new plugins for LLM and sqlite-utils and a flurry of things I've learned.

[... [1,690 words](#)]

---

[11:19 pm](#) / [plugins](#), [projects](#), [datasette](#), [weeknotes](#), [datasette-cloud](#), [sqlite-utils](#), [llm](#)

---

**[Aug. 17, 2023](#)**

Overnight, tens of thousands of businesses, ranging from one-person shops to the Fortune 500, woke up to a new reality where the underpinnings of their infrastructure suddenly became a potential legal risk. The BUSL and the additional use grant written by the HashiCorp team are vague, and now every company, vendor, and developer using Terraform has to wonder whether what they are doing could be construed as competitive with HashiCorp's offerings.



— [The OpenTF Manifesto](#)

# [5:15 am](#) / [open-source](#)

---

## [Aug. 18, 2023](#)

[Compromising LLMs: The Advent of AI Malware](#). The big Black Hat 2023 Prompt Injection talk, by Kai Greshake and team. The linked Whitepaper, [Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection](#), is the most thorough review of prompt injection attacks I've seen yet.

# [2:46 am](#) / [security](#), [ai](#), [prompt-injection](#), [llms](#)

---

I like to make sure almost every line of code I write is under a commercially friendly OS license (usually Apache 2) for genuinely selfish reasons: I never want to have to solve that problem ever again, so OS licensing my code now ensures I can use it for the rest of my life no matter who I happen to be working for in the future

— [Me](#)

# [7:33 pm](#) / [open-source](#)

---

## [Aug. 19, 2023](#)

[Does ChatGPT have a liberal bias?](#) (via) An excellent debunking by Arvind Narayanan and Sayash Kapoor of the [Measuring ChatGPT political bias paper](#) that's been doing the rounds recently.

It turns out that paper didn't even test ChatGPT/gpt-3.5-turbo - they ran their test against the older Da Vinci GPT3.

The prompt design was particularly flawed: they used political compass structured multiple choice: "choose between four options: strongly disagree, disagree, agree, or strongly agree". Arvind and Sayash found that asking an open ended question was far more likely to cause the models to answer in an unbiased manner.

I liked this conclusion:

There's a big appetite for papers that confirm users' pre-existing beliefs [...] But we've also seen that chatbots' behavior is highly sensitive to the prompt, so people can find evidence for whatever they want to believe.

# [4:53 am](#) / [ethics](#), [ai](#), [generative-ai](#), [chatgpt](#), [llms](#), [arvind-narayanan](#), [ai-ethics](#)

---

## [Aug. 20, 2023](#)

I apologize, but I cannot provide an explanation for why the Montagues and Capulets are beefing in Romeo and Juliet as it goes against ethical and moral standards, and promotes negative stereotypes and discrimination.

— [Llama 2 7B](#)

# [5:38 am](#) / [llms](#), [ai](#), [ethics](#), [generative-ai](#), [llama](#), [ai-ethics](#)

---

## [Aug. 21, 2023](#)



[Queryable Logging with Blacklite](#) ([via](#)) Will Sargent describes how he built Blacklite, a Java library for diagnostic logging that writes log events (as zstd compressed JSON objects) to a SQLite database and maintains 5,000 entries in a “live” database while entries beyond that range are cycled out to an archive.db file, which is cycled to archive.timestamp.db when it reaches 500,000 items.

Lots of interesting notes here on using SQLite for high performance logging.

“SQLite databases are also better log files in general. Queries are faster than parsing through flat files, with all the power of SQL. A vacuumed SQLite database is only barely larger than flat file logs. They are as easy to store and transport as flat file logs, but work much better when merging out of order or interleaved data between two logs.”

# 6:13 pm / [java](#), [logging](#), [sqlite](#), [zstd](#)

---

If you visit (often NSFW, beware!) showcases of generated images like civitai, where you can see and compare them to the text prompts used in their creation, you’ll find they’re often using massive prompts, many parts of which don’t appear anywhere in the image. These aren’t small differences — often, entire concepts like “a mystical dragon” are prominent in the prompt but nowhere in the image. These users are playing a gacha game, a picture-making slot machine. They’re writing a prompt with lots of interesting ideas and then pulling the arm of the slot machine until they win... something. A compelling image, but not really the image they were asking for.

— [Sam Bleckley](#)

# 7:38 pm / [stable-diffusion](#), [ai](#), [generative-ai](#)

---

When many business people talk about “AI” today, they treat it as a continuum with past capabilities of the CNN/RNN/GAN world. In reality it is a step function in new capabilities and products enabled, and marks the dawn of a new era of tech.

It is almost like cars existed, and someone invented an airplane and said “an airplane is just another kind of car - but with wings” - instead of mentioning all the new use cases and impact to travel, logistics, defense, and other areas. The era of aviation would have kicked off, not the “era of even faster cars”.

— [Elad Gil](#)

# 8:32 pm / [llms](#), [ai](#), [generative-ai](#)

---

M	T	W	T	F	S	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

