

Chunked, Compressed, & Cloud-native
N-dimensional arrays

From Tensors to Clouds—Guide to Zarr-Python 3

Sanket Verma

Community and OSS @ Zarr



<https://sanketverma.com/>

— — —

Slides

https://bit.ly/zp3_pycon_de



Slides & Notebook

https://bit.ly/zp3_more



What I'll be talking about?

- What is Zarr?
- What's new in the Zarr-Python 3?
- Demo
- How to get involved?

How Zarr works?

How Zarr works?

Arrays are container of items of the same data-type & size (in bits). The number of dimensions and items in container are described by the shape.

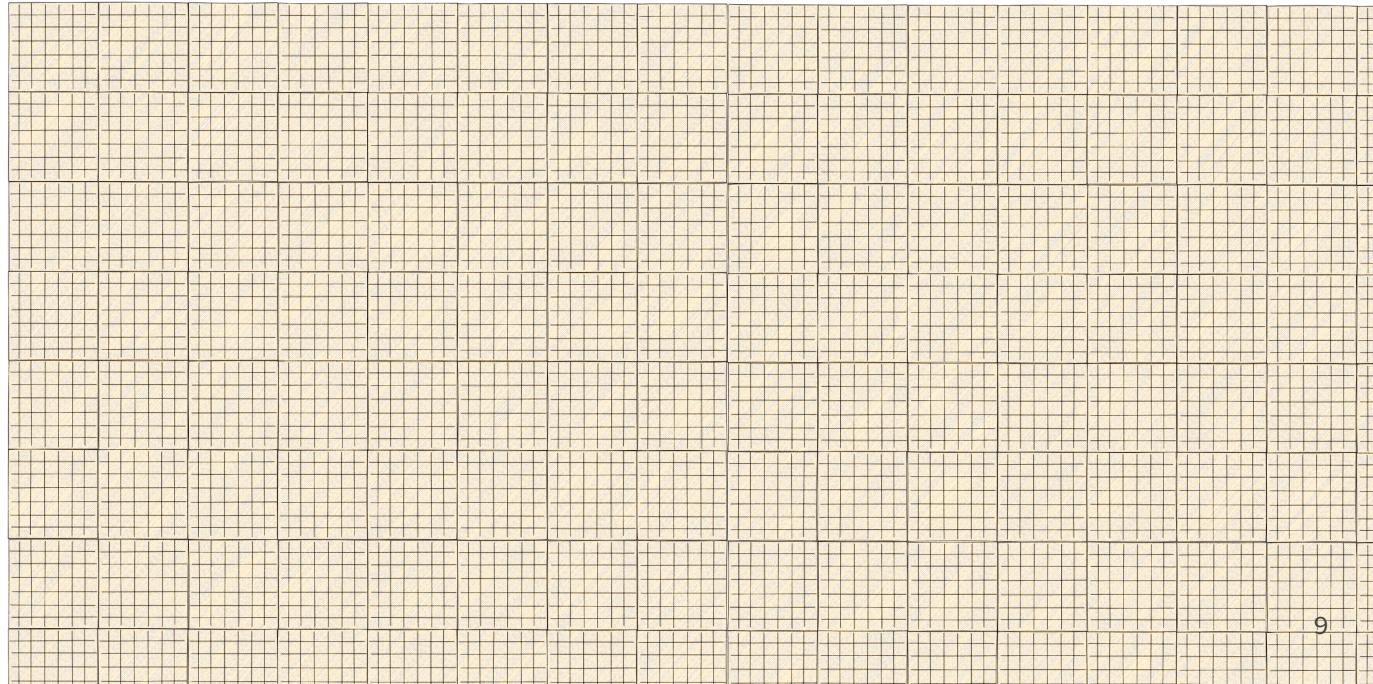
shape	(7,)	(7,7)
# dimensions	1D	2D
# items	7	$7 * 7$

How Zarr works?



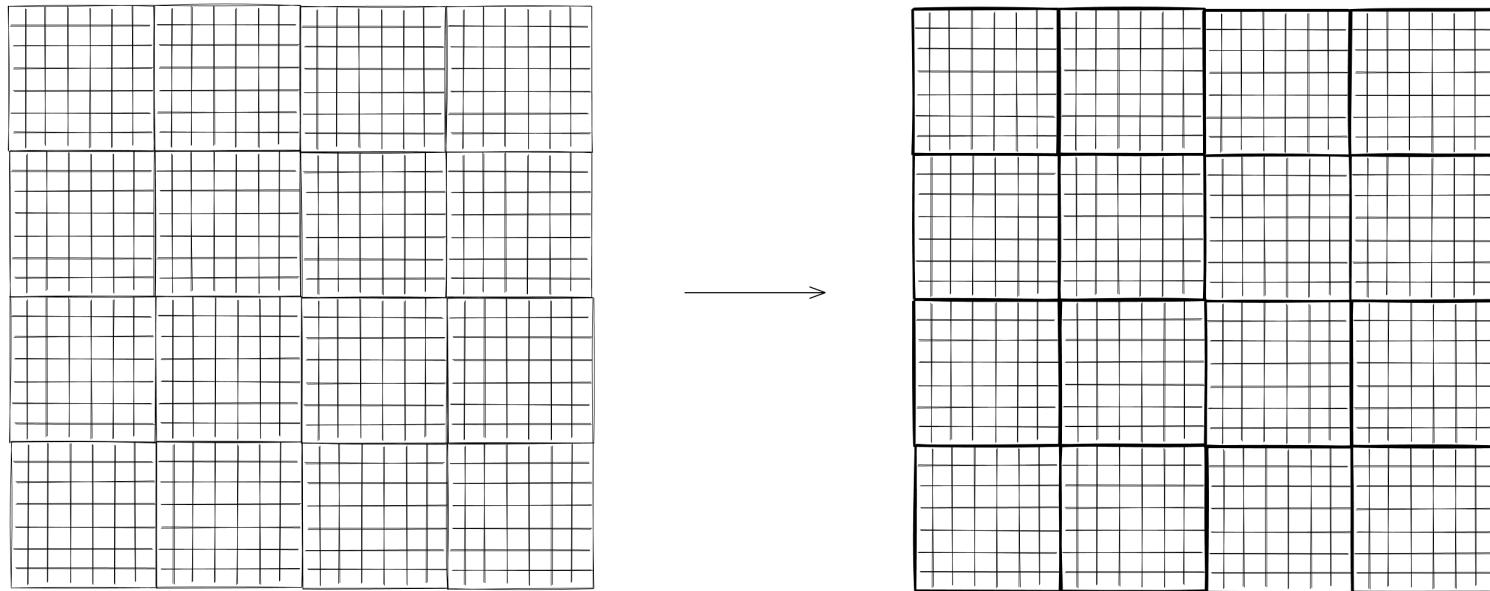
How Zarr works?

What if the data is too big to fit in memory?



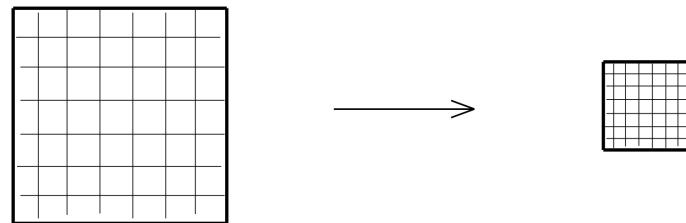
How Zarr works?

Divide array into chunks (Chunking)



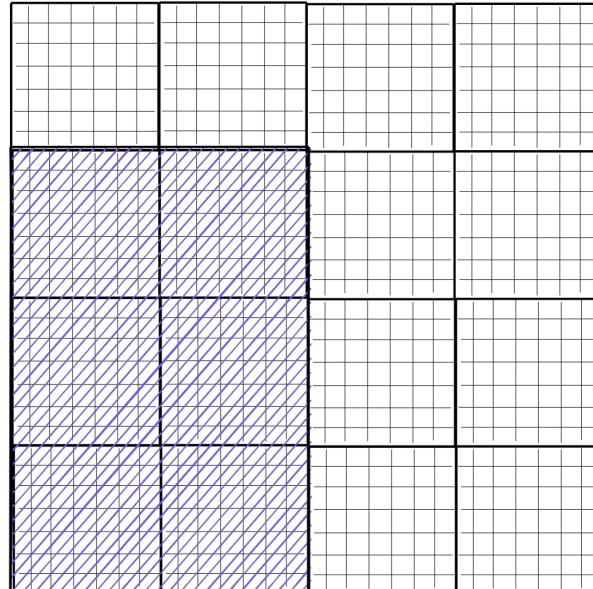
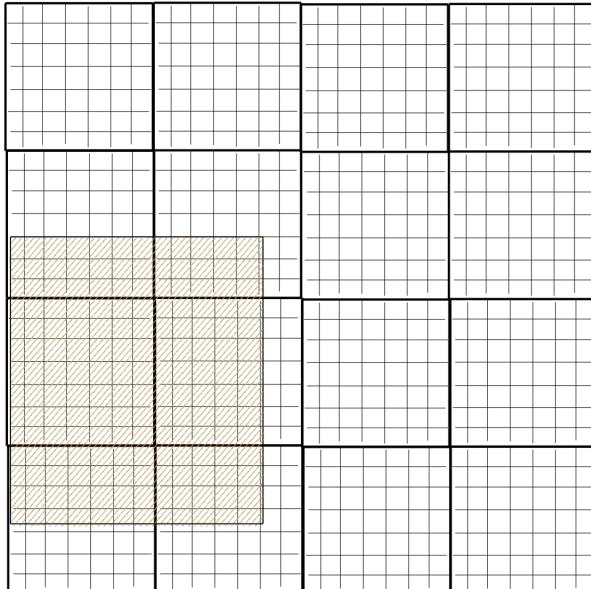
How Zarr works?

Compress each chunk

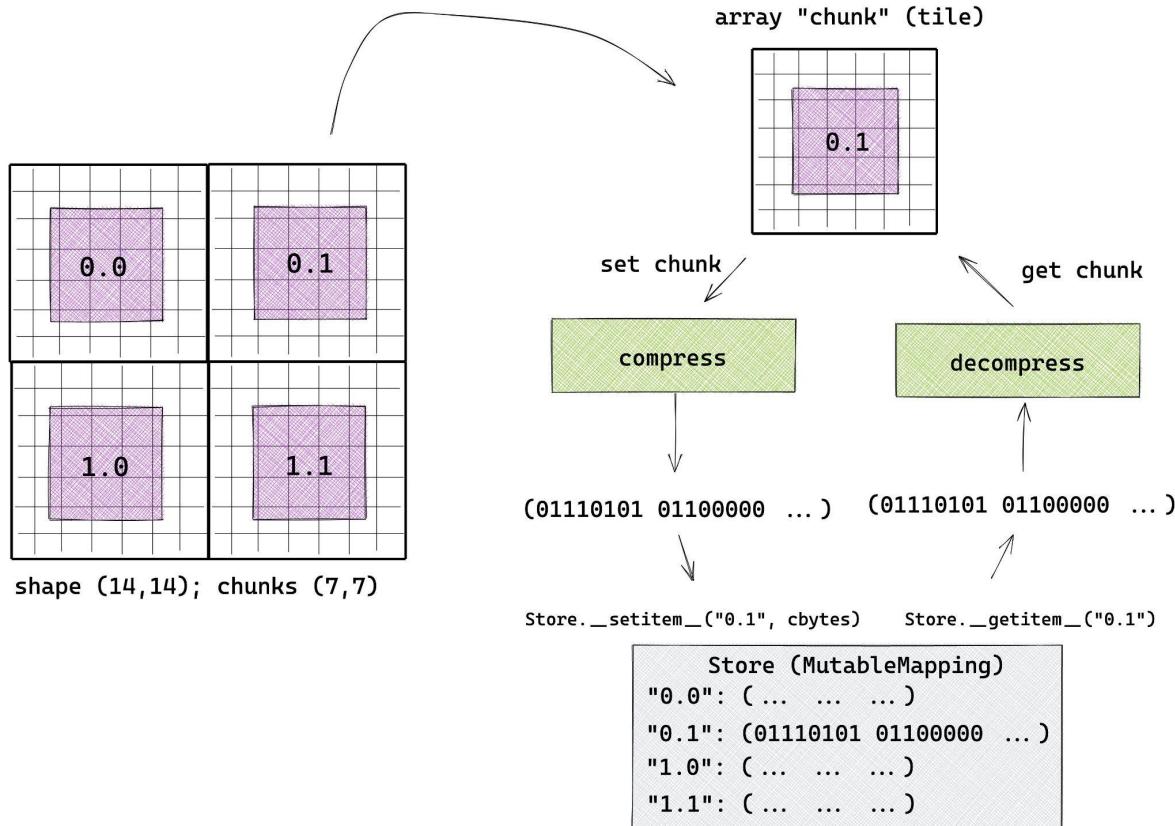


How Zarr works?

Retrieve chunks only when needed

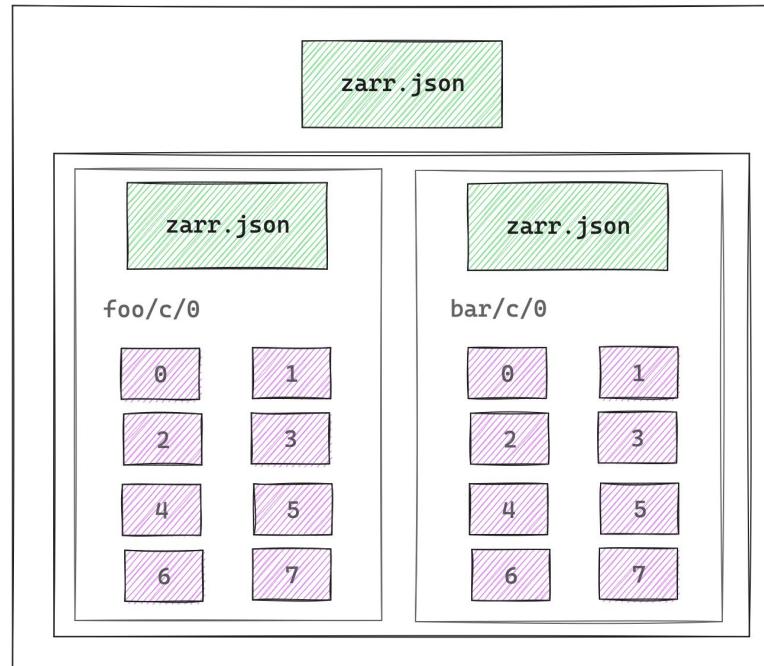


How Zarr works?



How Zarr works?

Multiple arrays can be organised in hierarchies of groups



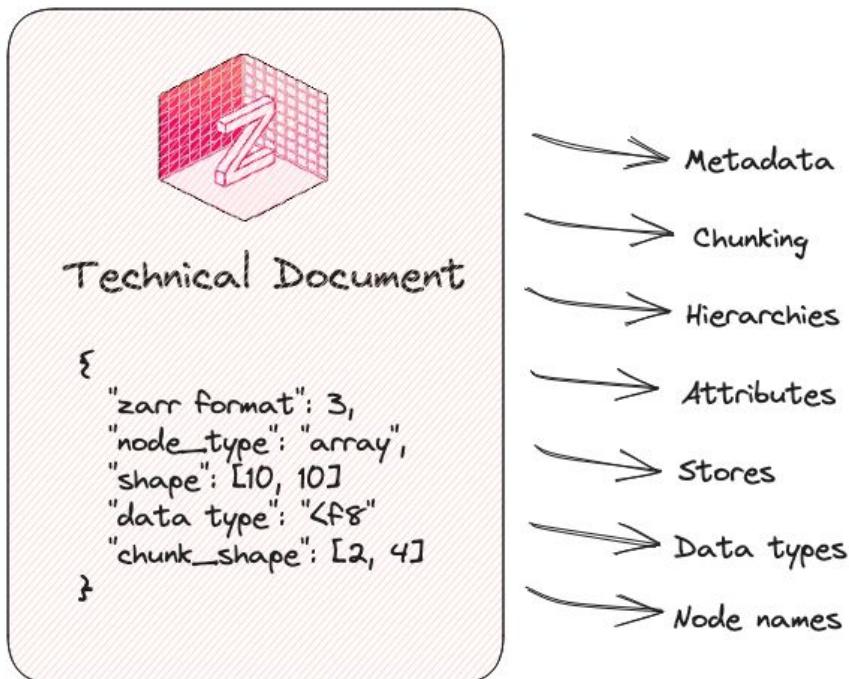
Zarr Specification

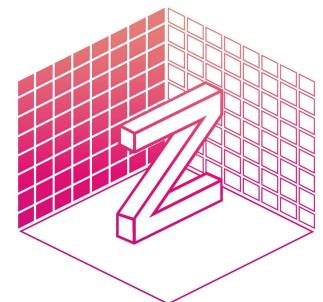
V1 → V2 → V3



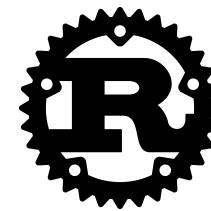
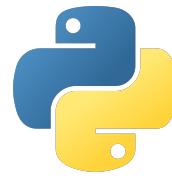
<https://zarr-specs.readthedocs.io/>

Zarr Specification





Zarr

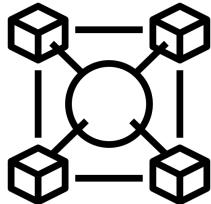


julia JS



What's new in Zarr-Python 3?

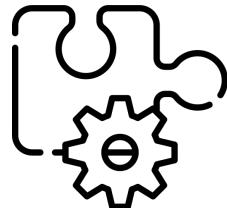
The Motivation



Interoperability



High-latency storage



Extensibility

Installation

```
$ pip install --upgrade zarr
```

or

```
$ conda install --channel conda-forge zarr
```

Installation

```
$ pip install "zarr[<extra>]"
```

where <extra> is:

gpu—support for GPUs

remote—reading/writing to remote data stores

Major design updates

New ABC Store (Abstract Base Class)

ABC Store

- Replaces MutableMapping base class
- Allows extensibility to add new custom stores
- Implements LocalStore, ZipStore and FSStore
- Removed various exotic stores like SQLite, MongoDB, Redis, DBMS, LMDBS Stores

Custom **Codecs** via Entrypoint Mechanism

Custom Codecs

- Enable flexible data compression and encoding strategies
- Zarr includes Blosc, Zstandard and Gzip compressors only
- Numcodecs has been adapted to use Zarr's Codec entrypoint system

GPU support for Zarr arrays

Using GPUs with Zarr

- `zarr.config.enable_gpu()` configures Zarr to use GPU memory
- Zarr currently enable reading ndarray into GPU memory.
Encoding and decoding still uses CPU memory

Changes and deprecations

Changes and deprecations

- `zarr.create_array()` instead of `zarr.array()`
- `zarr.create_group()` instead of `zarr.group()`
- `from zarr.storage import LocalStore` instead of
`from zarr import DirectoryStore`

Changes and deprecations

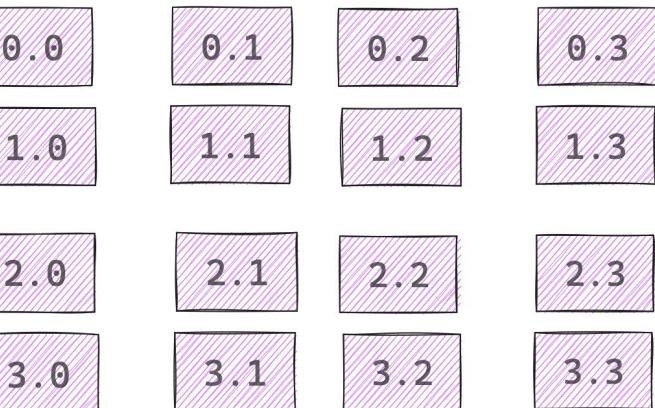
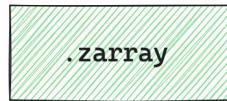
- `zarr.attrs` has gone, with no replacement
- `zarr.codecs` has gone, use `numcodecs` instead
- `zarr.context` has gone, with no replacement
- `zarr.core` remains but should be considered private API
- `zarr.hierarchy` has gone, with no replacement (use `zarr.Group` inplace of `zarr.hierarchy.Group`)
- `zarr.indexing` has gone, with no replacement
- `zarr.meta` has gone, with no replacement
- `zarr.meta_v1` has gone, with no replacement
- `zarr.sync` has gone, with no replacement
- `zarr.types` has gone, with no replacement
- `zarr.util` has gone, with no replacement
- `zarr.n5` has gone, see below for an alternative N5 options

Migration Guide

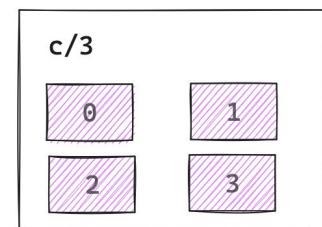
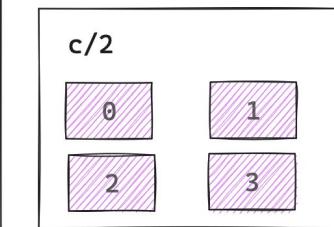
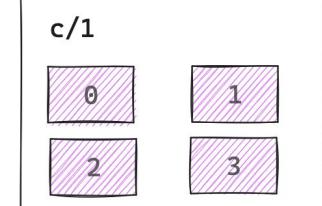
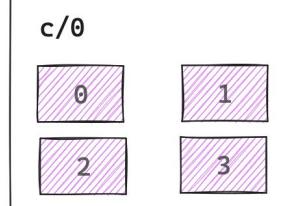
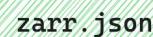
https://zarr.readthedocs.io/en/stable/user-guide/v3_migration.html



Zarr Arrays → V2 vs. V3

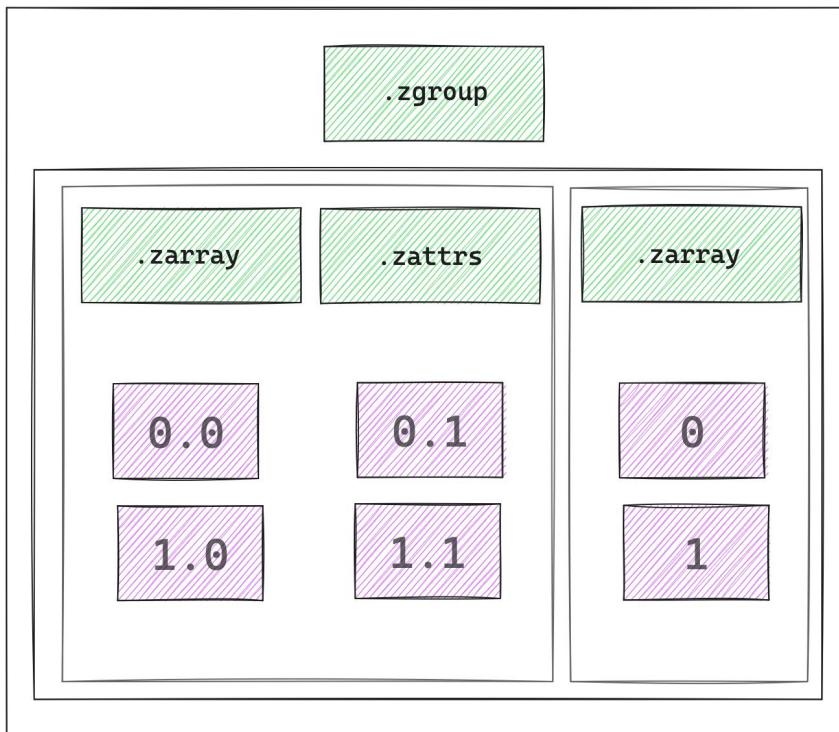


V2

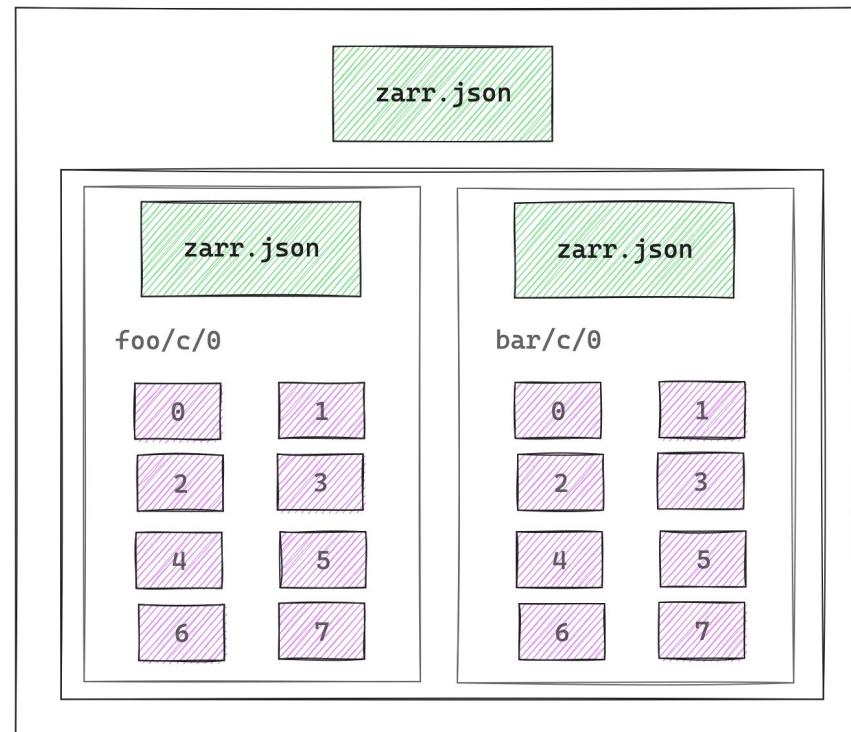


V3

Zarr Groups → V2 vs. V3



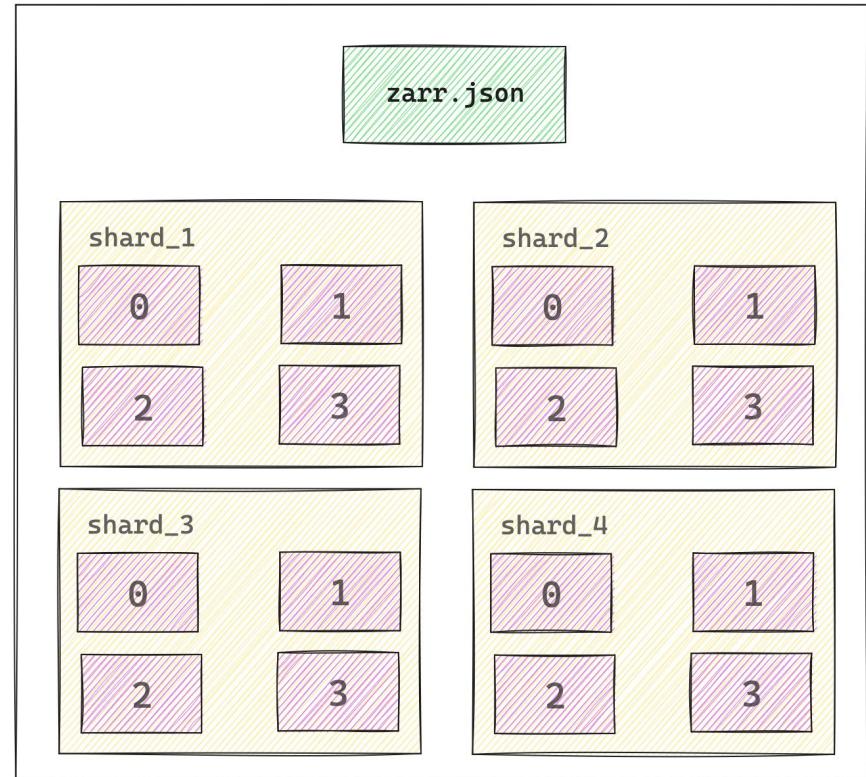
V2



V3

Sharding Codec

Via Extension Mechanism



chunk → compressible unit

shard → storage unit

Extensions



[zarrs](#)

A *Rust* library for the Zarr storage format

[GitHub](#) [Rust API](#) [Python API](#) [The zarrs Book](#) [Benchmarks](#) [CLI Tools](#)

zarrs-python

[pypi v0.1.3](#) [downloads/month 2k](#) [downloads 22k](#) [stars 34](#) [ci passing](#) [cd passing](#)

This project serves as a bridge between [zarrs](#) (Rust) and [zarr](#) (zarr-python) via [PyO3](#). The main goal of the project is to speed up i/o (see [zarr benchmarks](#)).

To use the project, simply install our package (which depends on `zarr-python>=3.0.0`), and run:

```
import zarr
import zarrs
zarr.config.set({"codec_pipeline.path": "zarrs.ZarrsCodecPipeline"})
```

You can then use your `zarr` as normal (with some caveats)!

Register your extensions



zarr-developers / zarr-extensions

Type ⌘ to search

Code Issues 1 Pull requests 4 Actions Projects Security Insights

zarr-extensions Public

Watch 8 Fork 5 Star 7

main 3 Branches 0 Tags Go to file Add file Code

normanrz adds extensions that zarr-python defines (#1)	a0b7932 · 4 days ago	7 Commits
chunk-grids	Initial commit	2 months ago
chunk-key-encodings	Initial commit	2 months ago
codecs	adds extensions that zarr-python defines (#1)	4 days ago
data-types	adds extensions that zarr-python defines (#1)	4 days ago
storage-transformers	Initial commit	2 months ago
README.md	Update README.md	2 weeks ago

About

This repository contains the specifications for Zarr extensions for Zarr version 3.

Readme Code of conduct Activity Custom properties 7 stars 8 watching 5 forks Report repository

Demo

How does V3 data looks like?



WEBKNOSSOS

Citation:

Motta et al., Science 2019

Maintained by:



/normanrz

(Norman Rzepka)



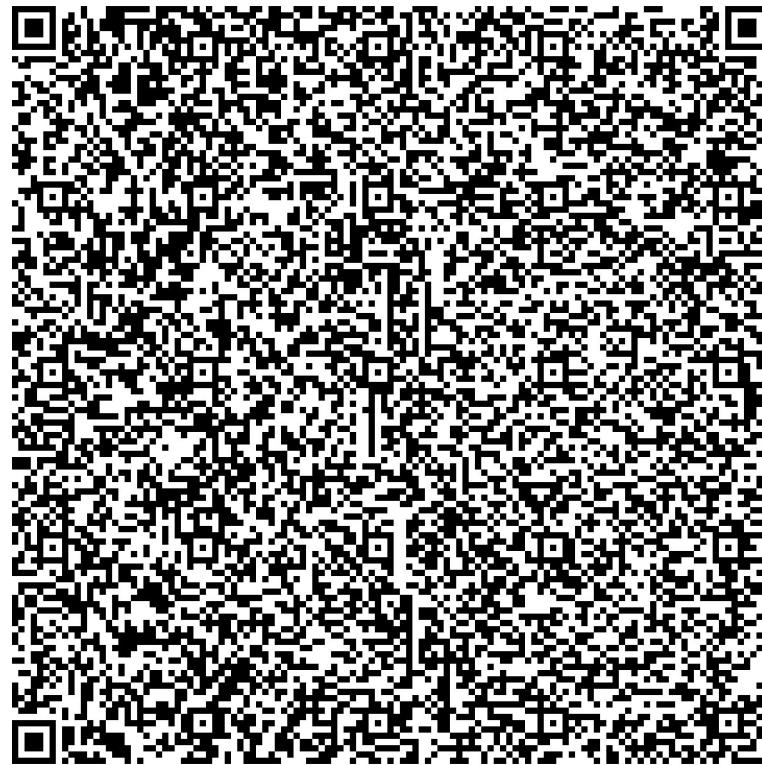
<https://webknossos.org/links/NxMfc4B65CmTVGhr>



<https://github.com/google/neuroglancer>

Maintained by:  /jbms

(Jeremy Maitin-Shephard)



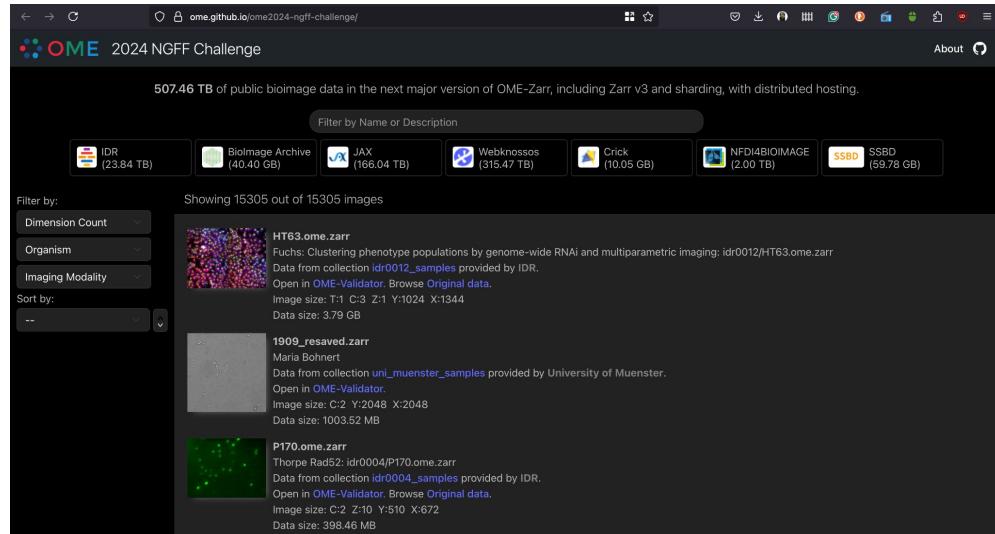
[Neuroglancer visualiser](#)



<https://ome.github.io/ome2024-ngff-challenge/>

Maintained by:  /joshmoore

(Josh Moore)



The screenshot shows a web browser displaying the "OME 2024 NGFF Challenge" page. The page header indicates "507.46 TB of public bioimage data". Below the header, there is a navigation bar with links to various datasets: IDR (23.84 TB), BioImage Archive (40.40 GB), JAX (166.04 TB), Webknossos (315.47 TB), Crick (10.05 GB), NFDI4BIOIMAGE (2.00 TB), and SSBD (59.78 GB). A search bar is present above a list of 15305 images. The list includes entries such as HT63.ome.zarr, 1909_resaved.zarr, and P170.ome.zarr, each with a thumbnail image, a brief description, and a link to the original data.

Join us at community, ZEP, core-dev meetings or office hours!

The calendar displays the following events:

- April 2:** 17:00 GeoZarr I, 20:00 Zarr Core
- April 4:** 16:00 Zarr-Pyth, 20:00 Virtualiz
- April 9:** 20:00 Zarr Offic
- April 10:** 18:00 Zarr Stee
- April 11:** 16:00 Zarr-Pyth
- April 16:** 20:00 Zarr Core
- April 18:** 16:00 Zarr-Pyth, 20:00 Virtualiz
- April 22:** 20:00 Zarr Offic (circled in blue)
- April 23:** 18:00 Zarr Stee
- April 24:** 16:00 Zarr-Pyth
- April 30:** 20:00 Zarr Core
- May 1:** 16:00 Zarr-Pyth, 20:00 Virtualiz

Zarr Public Calendar
Terminanzeige in der Zeitzone: (GMT+02:00) Mitteleuropäische Zeit - Berlin
[Zu Google Kalender hinzufügen](#)

<https://zarr.dev/community-calls/>

Google Kalender

Thank you!



<https://zarr.dev/>