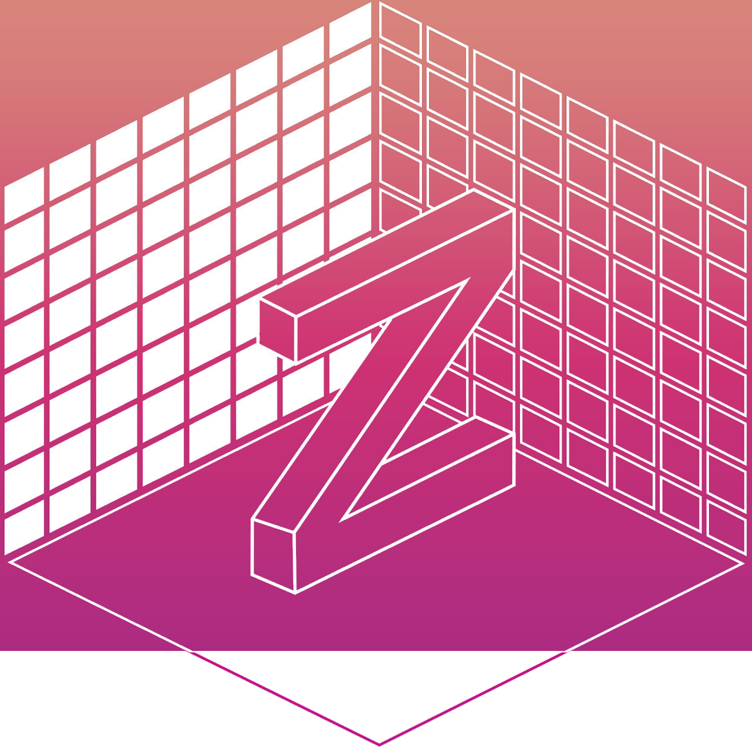


slides:

bit.ly/zarr-evolution-scipy-2023

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

Maintenance and Evolution of Zarr

slides:
bit.ly/zarr-evolution-scipy-2023

Josh Moore
Zarr Steering Council
@notjustmoore

Sanket Verma
Zarr Community Manager
@MSanKeys963

@zarr_dev

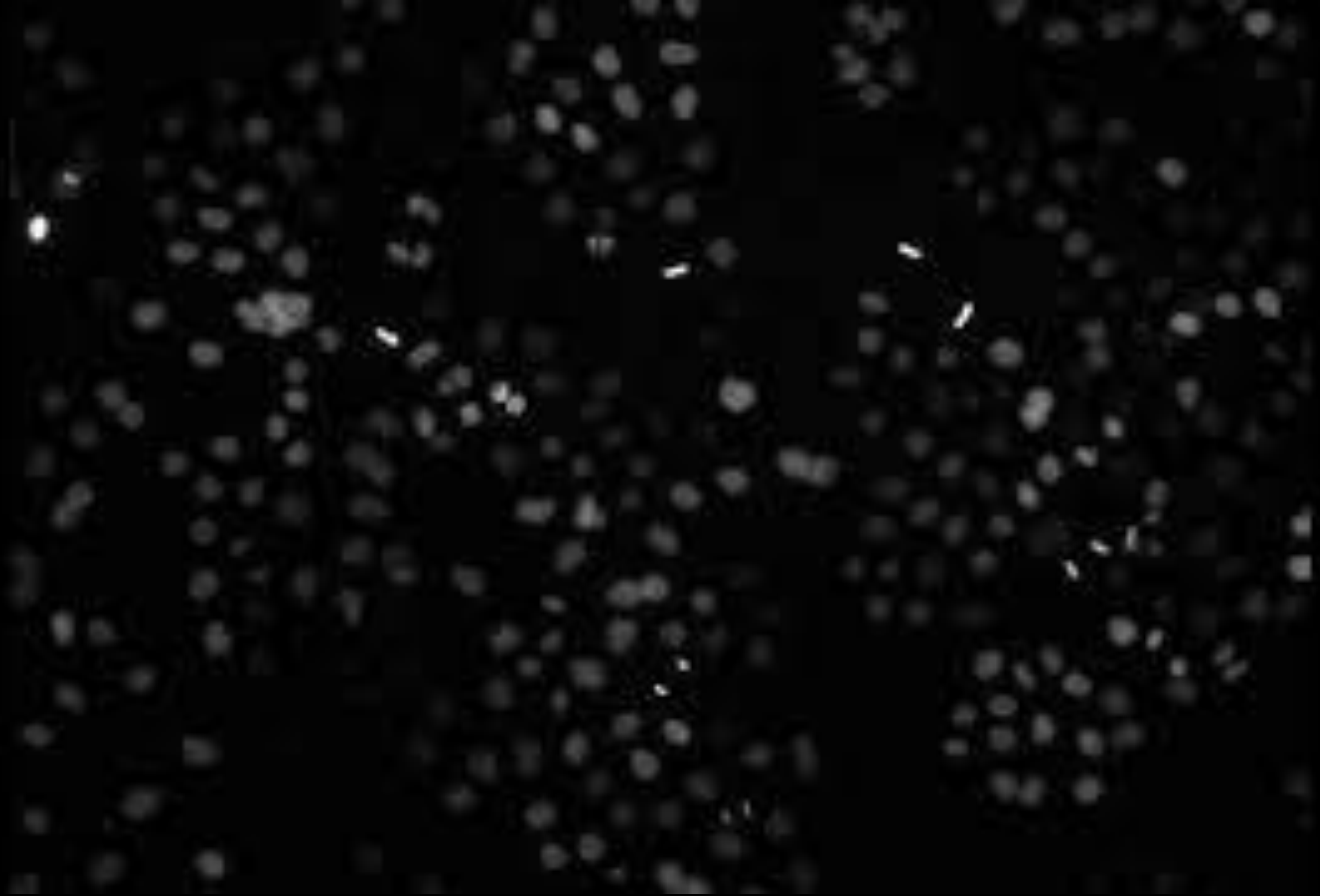
slides:
bit.ly/zarr-evolution-scipy-2023

Josh Moore
Zarr Steering Council
@joshmoore@fediscience.org

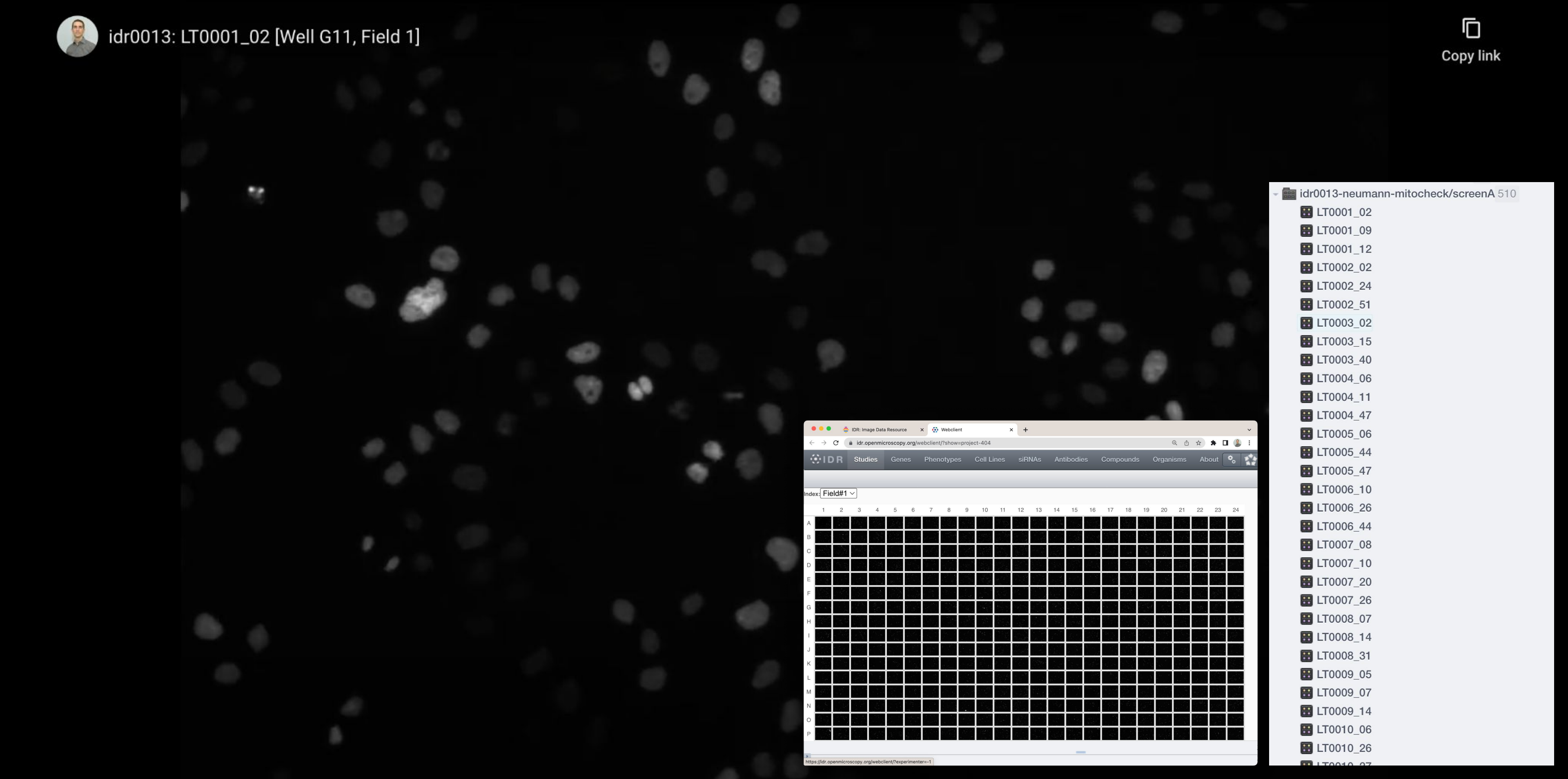
Sanket Verma
Zarr Community Manager

@zarr@fosstodon.org

So what’s my problem?



Mitocheck screen (~2003) **25TB, 200K videos** Credit: Neumann *et al.*, IDR (idr0013)



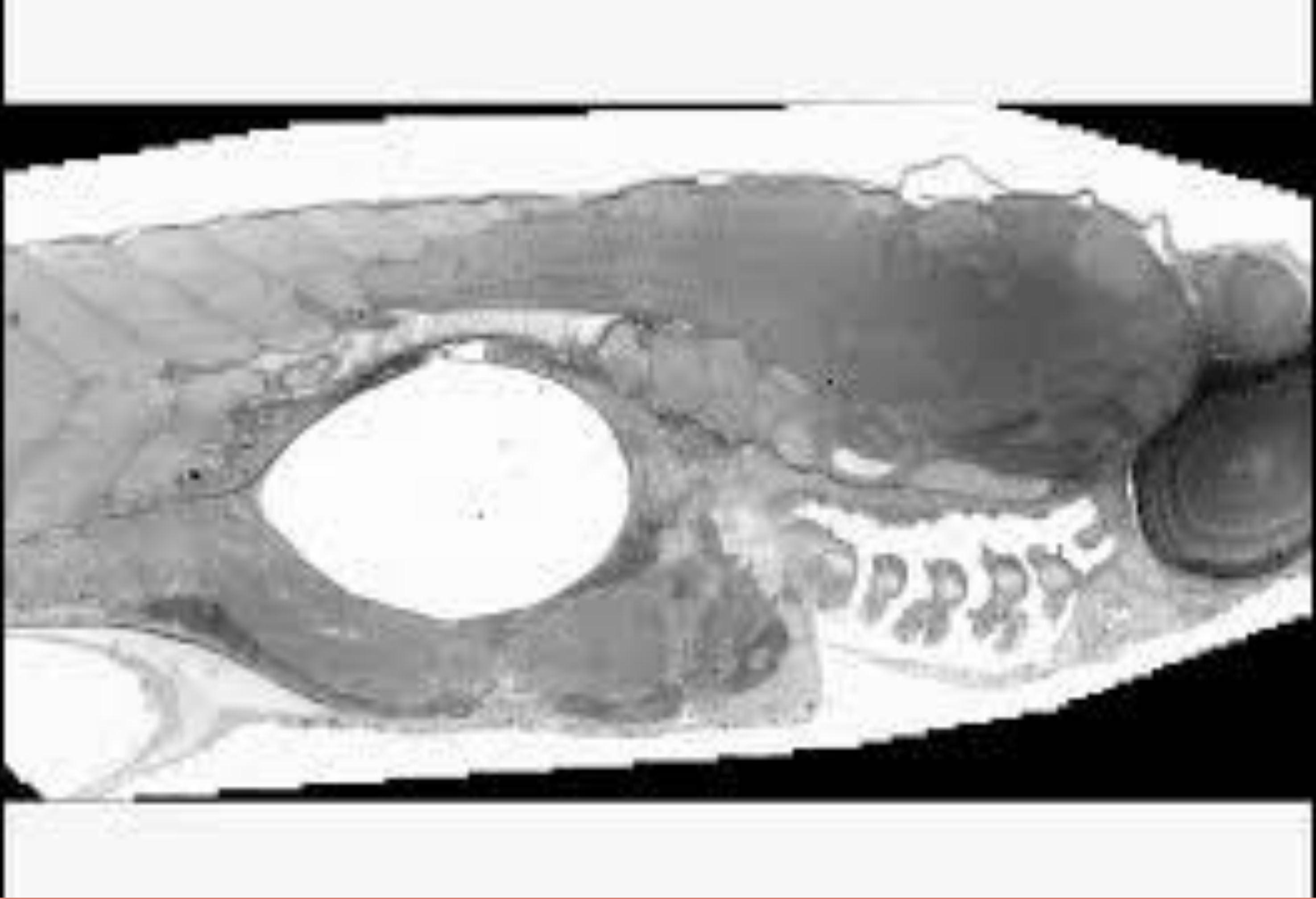
idr.openmicroscopy.org/webclient/?show=project-404

https://idr.openmicroscopy.org/webclient/?experimenter=-1

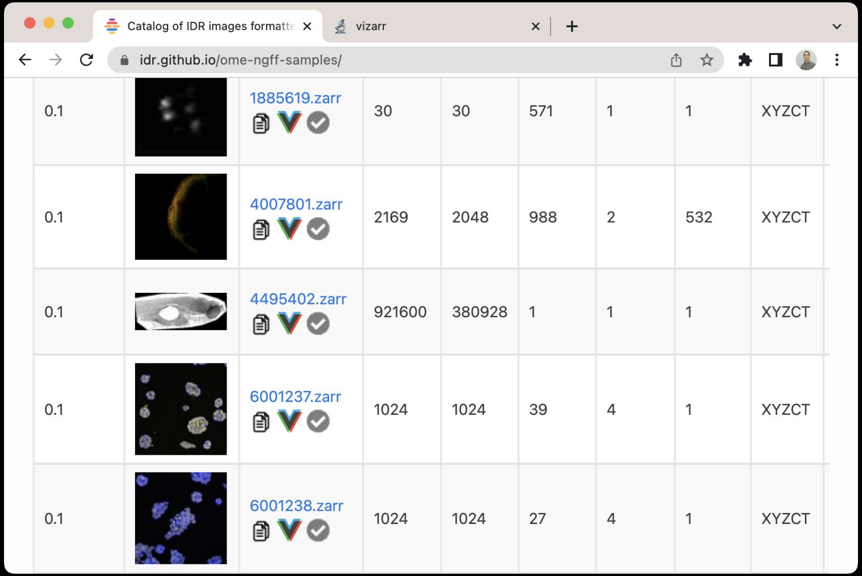
idr0013-neumann-mitocheck/screenA 510

- LT0001_02
- LT0001_09
- LT0001_12
- LT0002_02
- LT0002_24
- LT0002_51
- LT0003_02
- LT0003_15
- LT0003_40
- LT0004_06
- LT0004_11
- LT0004_47
- LT0005_06
- LT0005_44
- LT0005_47
- LT0006_10
- LT0006_26
- LT0006_44
- LT0007_08
- LT0007_10
- LT0007_20
- LT0007_26
- LT0008_07
- LT0008_14
- LT0008_31
- LT0009_05
- LT0009_07
- LT0009_14
- LT0010_06
- LT0010_26

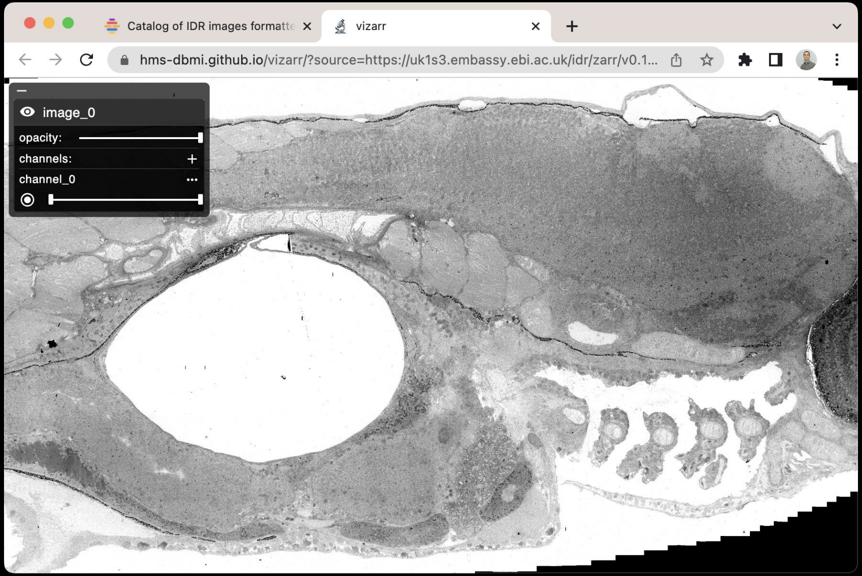
Mitocheck screen (~2003) **25TB, 200K videos** Credit: Neumann *et al.*, IDR (idr0013)



Zebrafish EM (~2012) **500 GB, 1 frame** Credit: Faas *et al.*, IDR (idr0053)


idr.github.io/ome-ngff-samples/

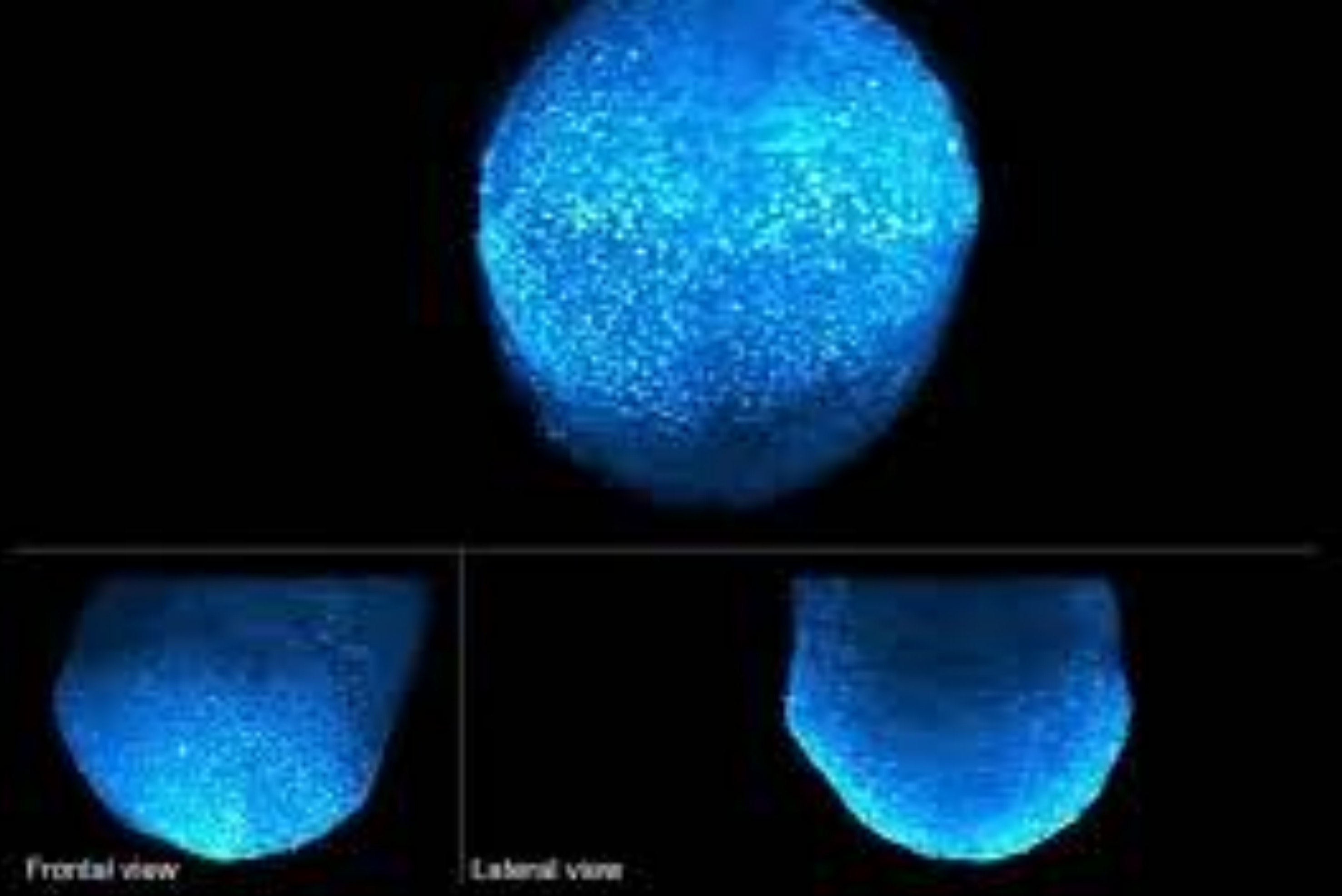
0.1		1885619.zarr	30	30	571	1	1	XYZCT
0.1		4007801.zarr	2169	2048	988	2	532	XYZCT
0.1		4495402.zarr	921600	380928	1	1	1	XYZCT
0.1		6001237.zarr	1024	1024	39	4	1	XYZCT
0.1		6001238.zarr	1024	1024	27	4	1	XYZCT


hms-dbmi.github.io/vizarr/?source=https://uk1s3.embassy.ebi.ac.uk/idr/zarr/v0.1...

https://idr.github.io/ome-ngff-samples/

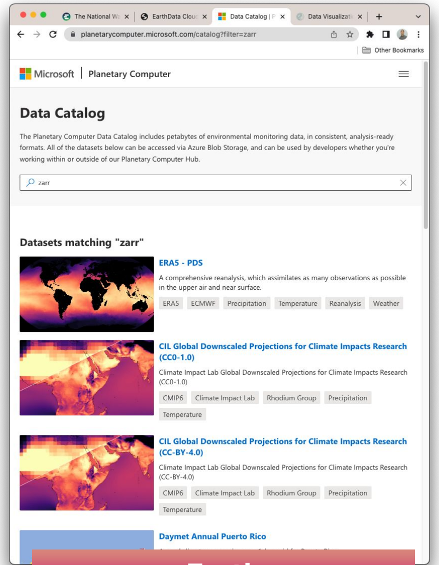
https://github.com/hms-dbmi/vizarr
https://www.buymeacoffee.com/manzt

Zebrafish EM (~2012) **500 GB, 1 frame** Credit: Faas *et al.*, IDR (idr0053) 8



Mouse embryo (~2018) **5 TB, 1 volume** Credit: McDole *et al.* Cell (10.1016/j.cell.2018.09.031)

TBD **10 PB, “1 brain”** Credit: https://knowyourmeme.com/


planetarycomputer.microsoft.com/catalog?filter=zarr

Microsoft | Planetary Computer

Data Catalog

The Planetary Computer Data Catalog includes petabytes of environmental monitoring data, in consistent, analysis-ready formats. All of the datasets below can be accessed via Azure Blob Storage, and can be used by developers whether you're working within or outside of our Planetary Computer Hub.

zarr

Datasets matching "zarr"

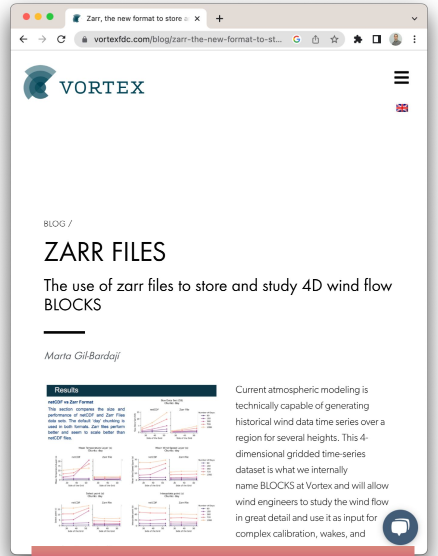
ERA5 - PDS
A comprehensive reanalysis, which assimilates as many observations as possible in the upper air and near surface.
ERA5 ECMWF Precipitation Temperature Reanalysis Weather

CIL Global Downscaled Projections for Climate Impacts Research (CC0-1.0)
Climate Impact Lab Global Downscaled Projections for Climate Impacts Research (CC0-1.0)
CMIP6 Climate Impact Lab Rhodium Group Precipitation Temperature

CIL Global Downscaled Projections for Climate Impacts Research (CC-BY-4.0)
Climate Impact Lab Global Downscaled Projections for Climate Impacts Research (CC-BY-4.0)
CMIP6 Climate Impact Lab Rhodium Group Precipitation Temperature

Daymet Annual Puerto Rico

Earth


vortexfdc.com/blog/zarr-the-new-format-to-st...

VORTEX

BLOG /

ZARR FILES

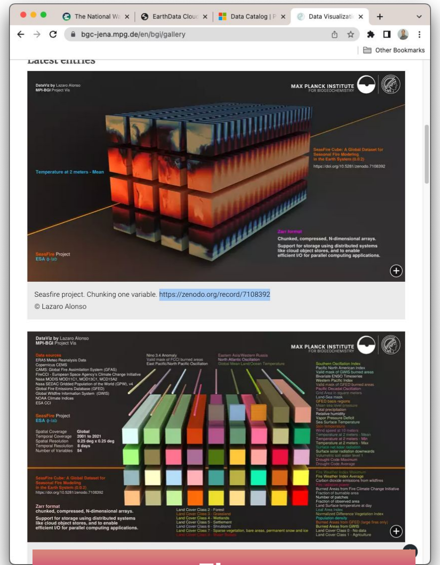
The use of zarr files to store and study 4D wind flow BLOCKS

Marta Gil-Bardají

Results
netCDF vs Zarr Format
This section compares the size and performance of netCDF and Zarr Files data sets. The default 'day' chunking is used in both formats. Zarr files perform better and seem to scale better than netCDF files.

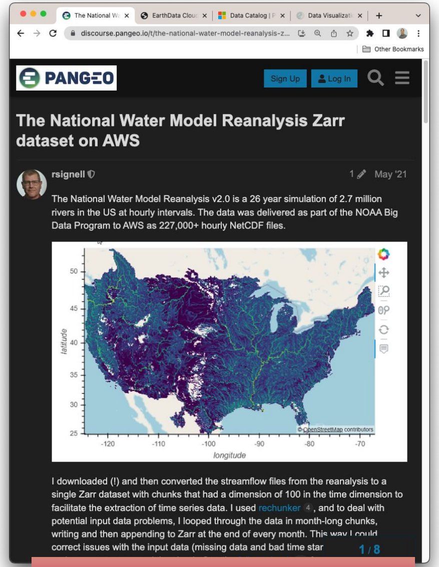
Current atmospheric modeling is technically capable of generating historical wind data time series over a region for several heights. This 4-dimensional gridded time-series dataset is what we internally name BLOCKS at Vortex and will allow wind engineers to study the wind flow in great detail and use it as input for complex calibration, wakes, and

Wind


bgc-jena.mpg.de/en/bgi/gallery

Seasfire project. Chunking one variable. https://zenodo.org/record/7108392
© Lazaro Alonso

Fire


discourse.pangeo.io/t/the-national-water-model-reanalysis-z...

PANGEO

The National Water Model Reanalysis Zarr dataset on AWS

rsignell 1 May '21

The National Water Model Reanalysis v2.0 is a 26 year simulation of 2.7 million rivers in the US at hourly intervals. The data was delivered as part of the NOAA Big Data Program to AWS as 227,000+ hourly NetCDF files.

I downloaded (!) and then converted the streamflow files from the reanalysis to a single Zarr dataset with chunks that had a dimension of 100 in the time dimension to facilitate the extraction of time series data. I used rechunker 4, and to deal with potential input data problems, I looped through the data in month-long chunks, writing and then appending to Zarr at the end of every month. This way I could correct issues with the input data (missing data and bad time star

1 / 8

(Water, etc.)

Benefits of Zarr

Distributed & Cloud Storage

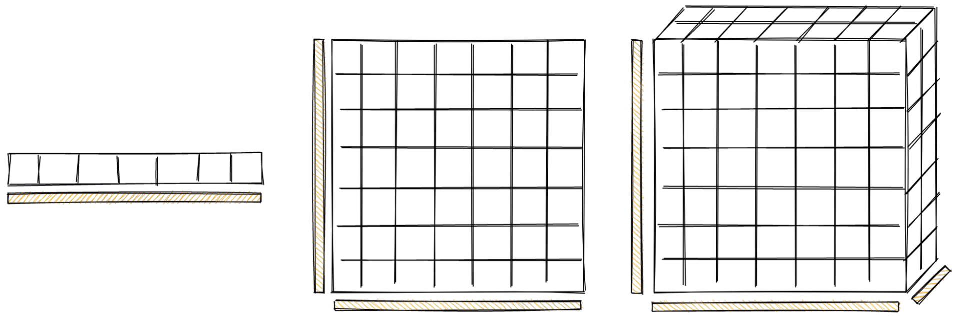
Chunked Storage

Built-In Compression

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.

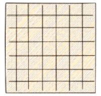


shape	(7,)	(7,7)	(2,7,7)
# dimensions	1D	2D	3D
# items	7	7 * 7	7 * 7 * 2

Credit: @trevmanz

How Zarr works?

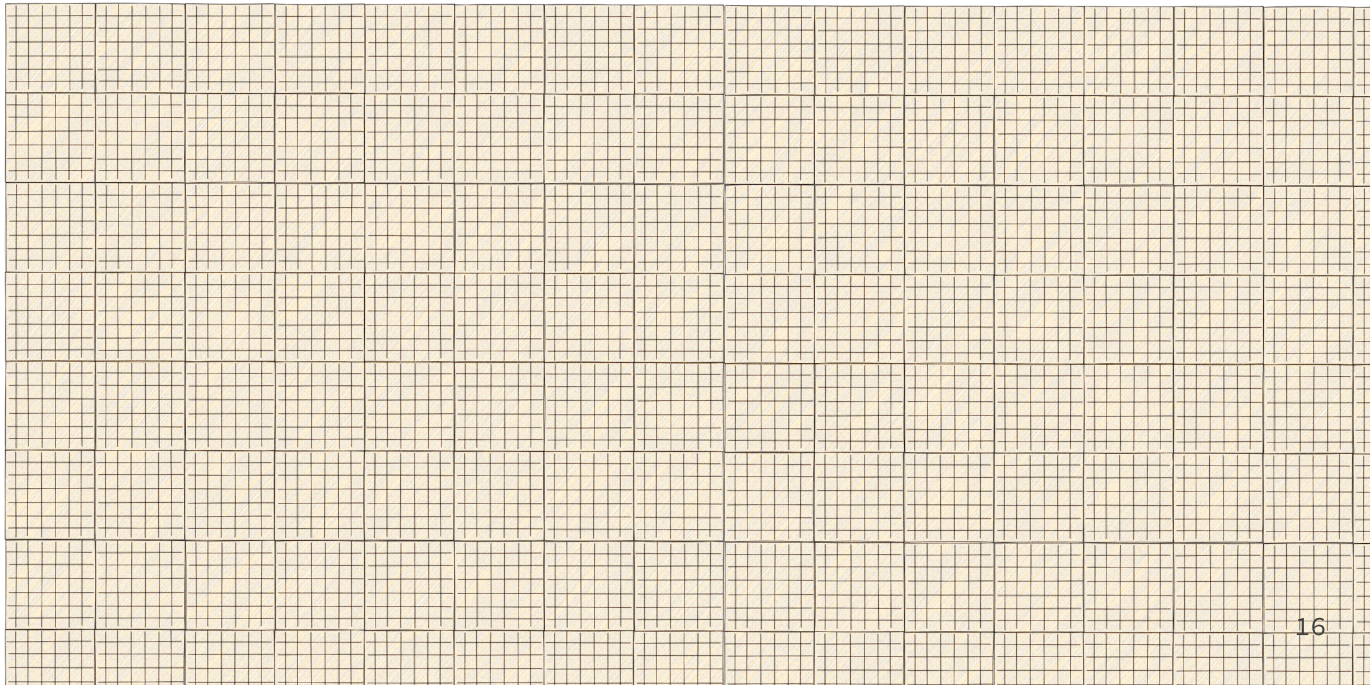
So take the arrays you know and love?



Credit: @trevmanz

How Zarr works?

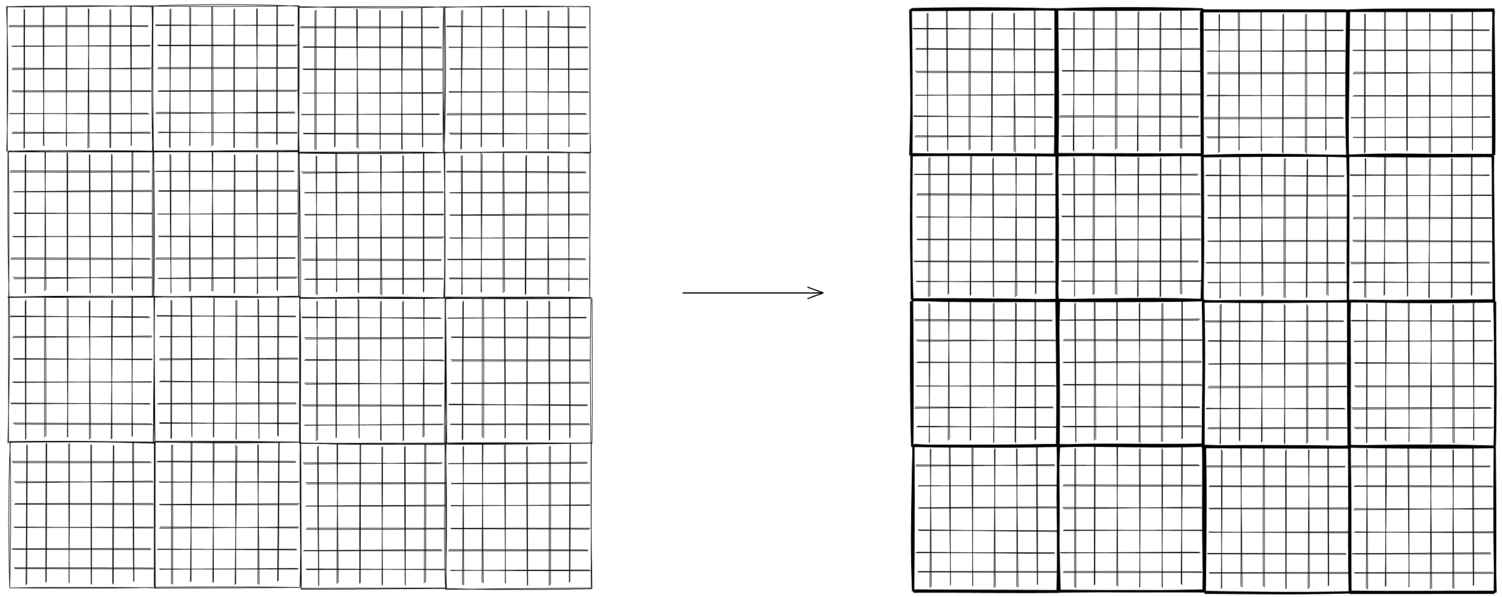
What if the data is too big to fit in memory?



Credit: @trevmanz

How Zarr works?

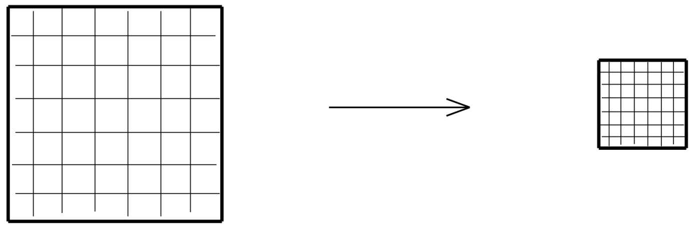
Divide array into chunks (Chunking)



Credit: @trevmanz

How Zarr works?

Compress each chunk

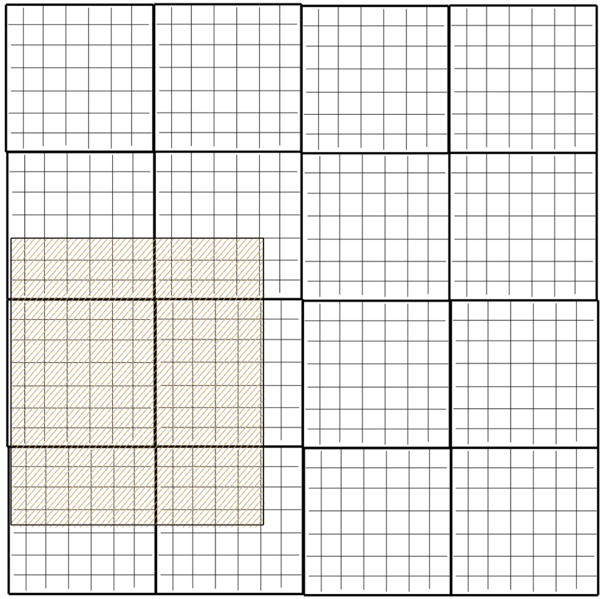


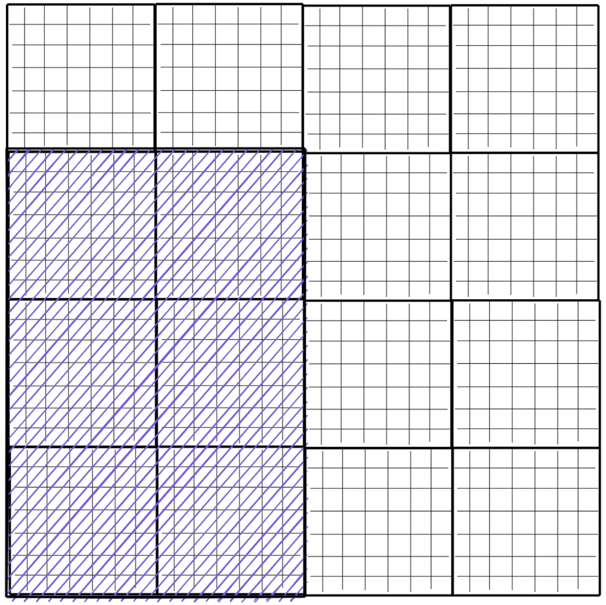
Over 20 supported compressors (BLOSC, Zstd, Zlib etc)

Credit: @trevmanz

How Zarr works?

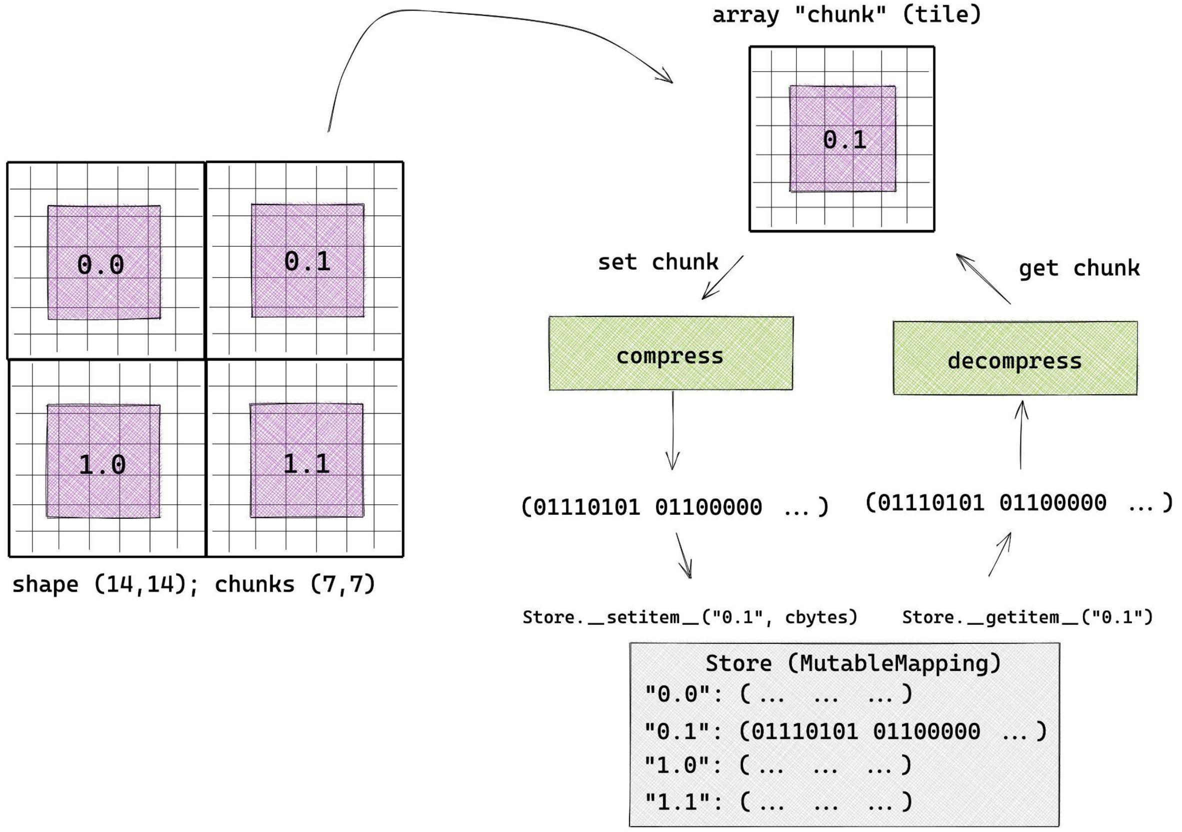
Retrieve chunks only when needed





Credit: @trevmanz

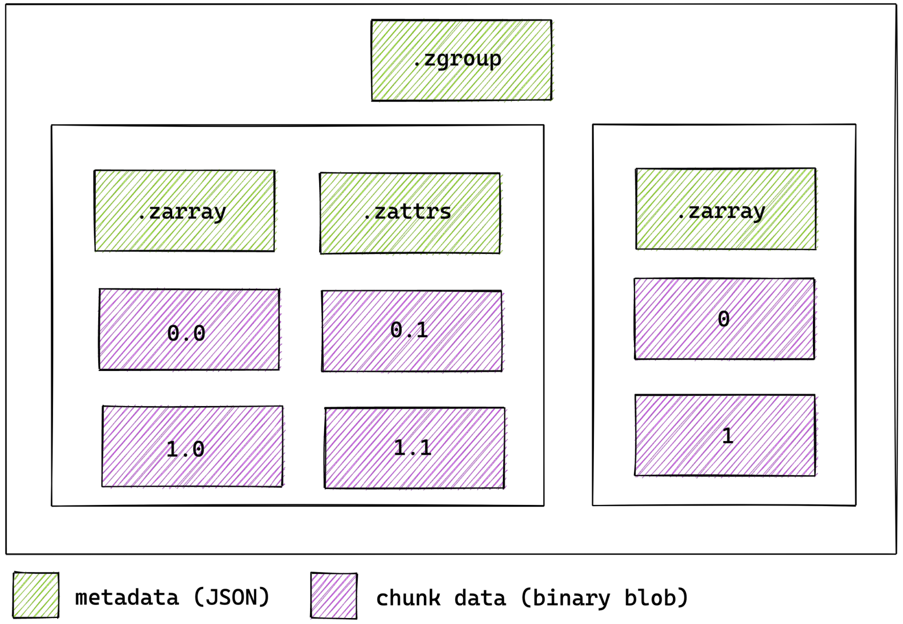
How Zarr works?



Credit: @trevmanz

How Zarr works?

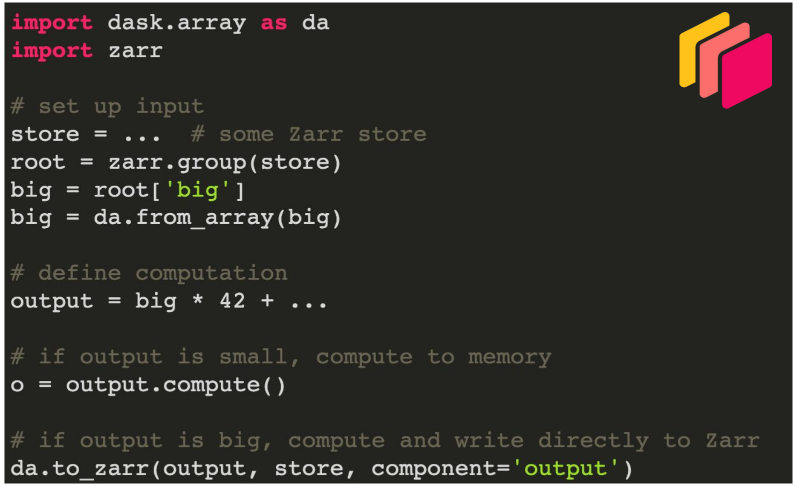
Multiple arrays can be organised in hierarchies of groups



Credit: @trevmanz

da.from_array & xr.open_zarr

Integrated methods through the PyData ecosystem



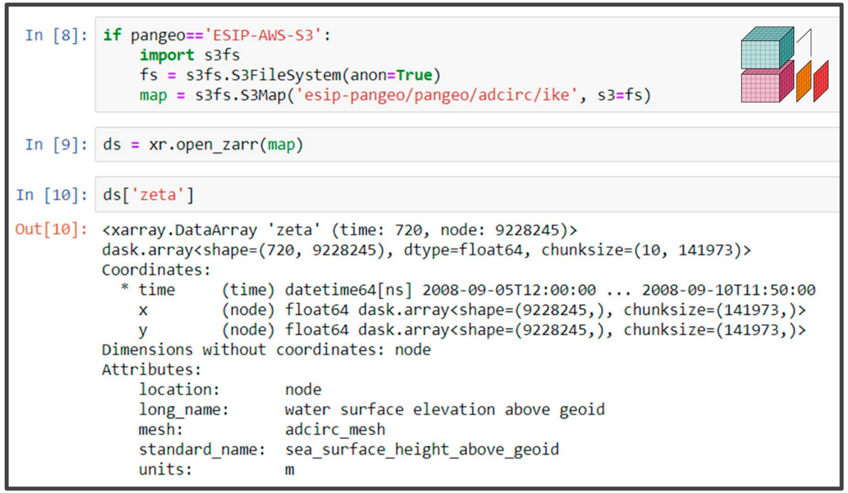
```python
import dask.array as da
import zarr

# set up input
store = ...  # some Zarr store
root = zarr.group(store)
big = root['big']
big = da.from_array(big)

# define computation
output = big * 42 + ...

# if output is small, compute to memory
o = output.compute()

# if output is big, compute and write directly to Zarr
da.to_zarr(output, store, component='output')
```



```
In [8]:  if pangeo=='ESIP-AWS-S3':
             import s3fs
             fs = s3fs.S3FileSystem(anon=True)
             map = s3fs.S3Map('esip-pangeo/pangeo/adcirc/ike', s3=fs)

In [9]:  ds = xr.open_zarr(map)

In [10]: ds['zeta']

Out[10]: <xarray.DataArray 'zeta' (time: 720, node: 9228245)>
         dask.array<shape=(720, 9228245), dtype=float64, chunksize=(10, 141973)>
         Coordinates:
           * time     (time) datetime64[ns] 2008-09-05T12:00:00 ... 2008-09-10T11:50:00
             x        (node) float64 dask.array<shape=(9228245,), chunksize=(141973,)>
             y        (node) float64 dask.array<shape=(9228245,), chunksize=(141973,)>
         Dimensions without coordinates: node
         Attributes:
             location:       node
             long_name:      water surface elevation above geoid
             mesh:           adcirc_mesh
             standard_name:  sea_surface_height_above_geoid
             units:          m
```

Credit: Alistair Miles
https://zarr.dev/slides/scipy-2019.html#/8/1

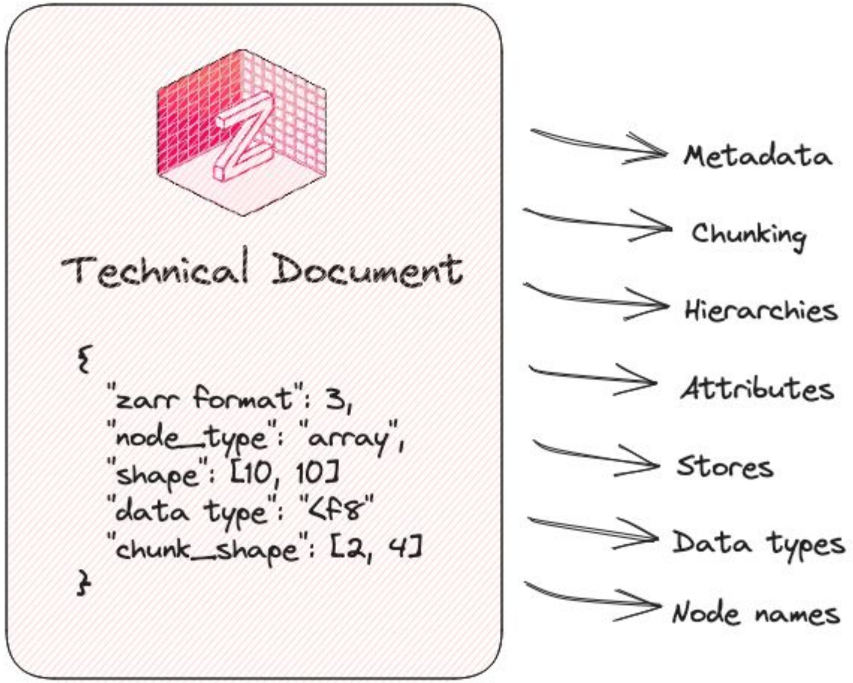
Credit: Richard Signell
https://www.mdpi.com/2077-1312/7/4/110

Zarr Specification

V1 → V2 → V3

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

Zarr Specification



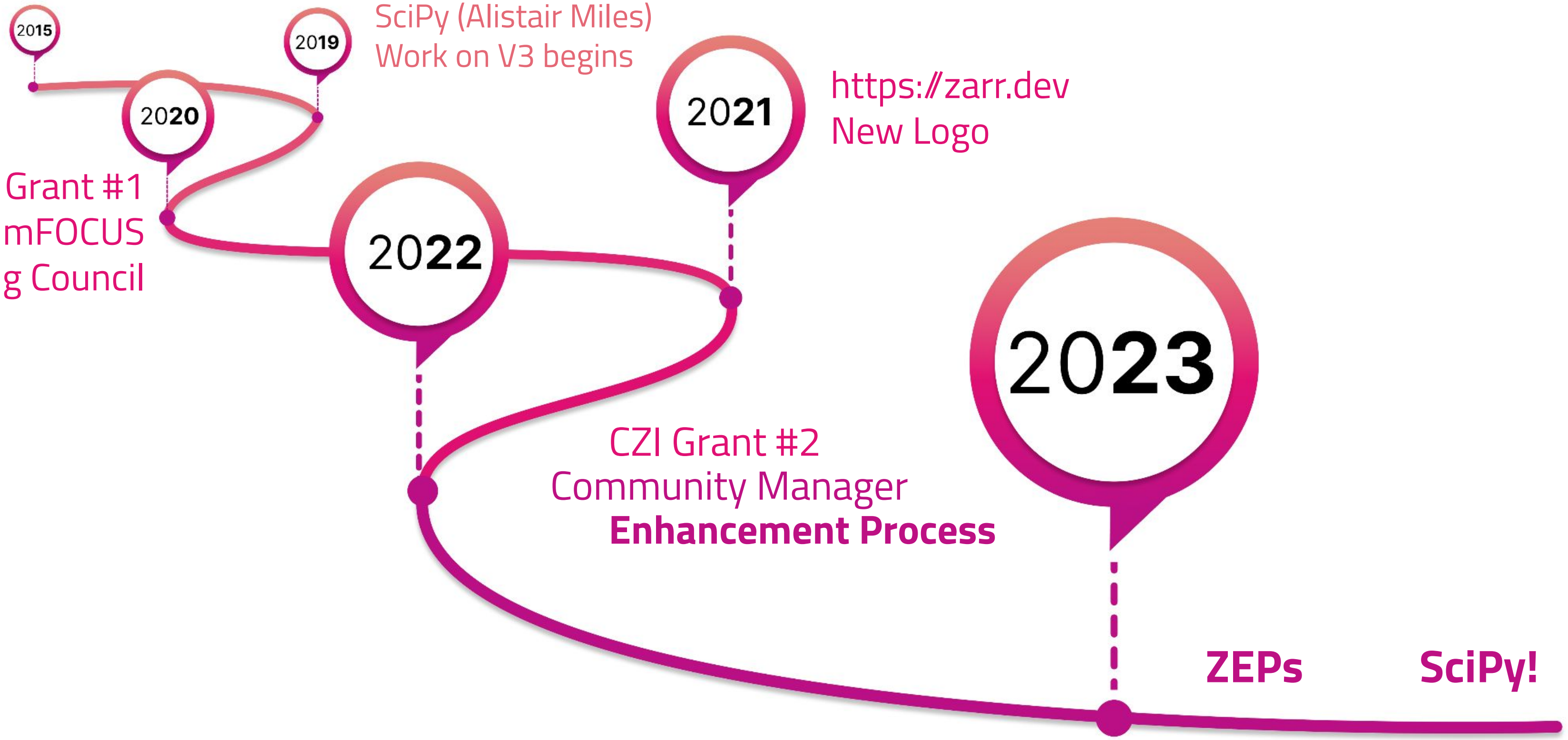
```
{
  "zarr format": 3,
  "node_type": "array",
  "shape": [10, 10]
  "data type": "<f8"
  "chunk_shape": [2, 4]
}
```

Zarr

↔

julia JS Java

Timeline

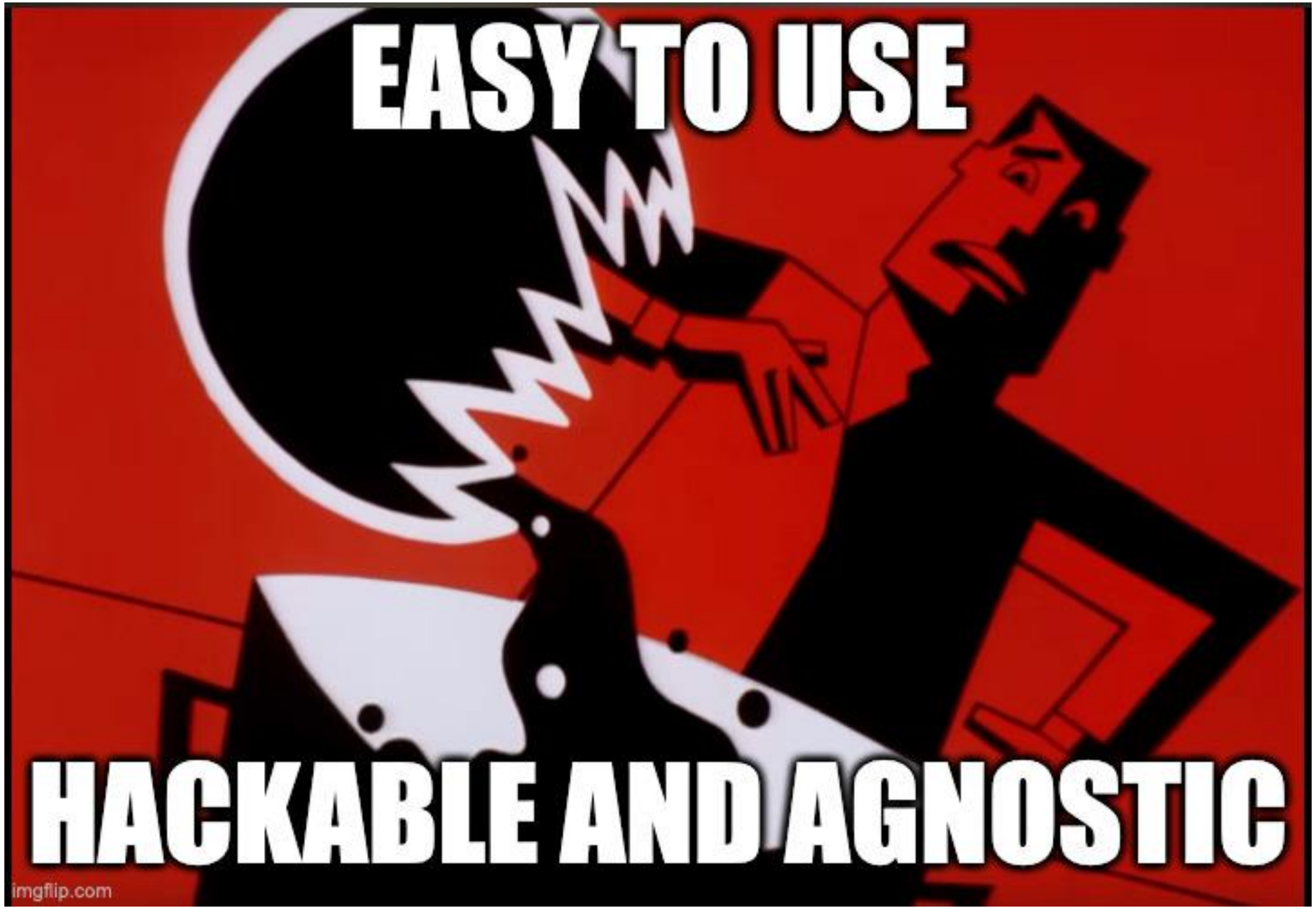

First commit
2015
2019
SciPy (Alistair Miles)
Work on V3 begins
2020
CZI Grant #1
NumFOCUS
Steering Council
2021
https://zarr.dev
New Logo
2022
CZI Grant #2
Community Manager
Enhancement Process
2023
ZEPs
SciPy!

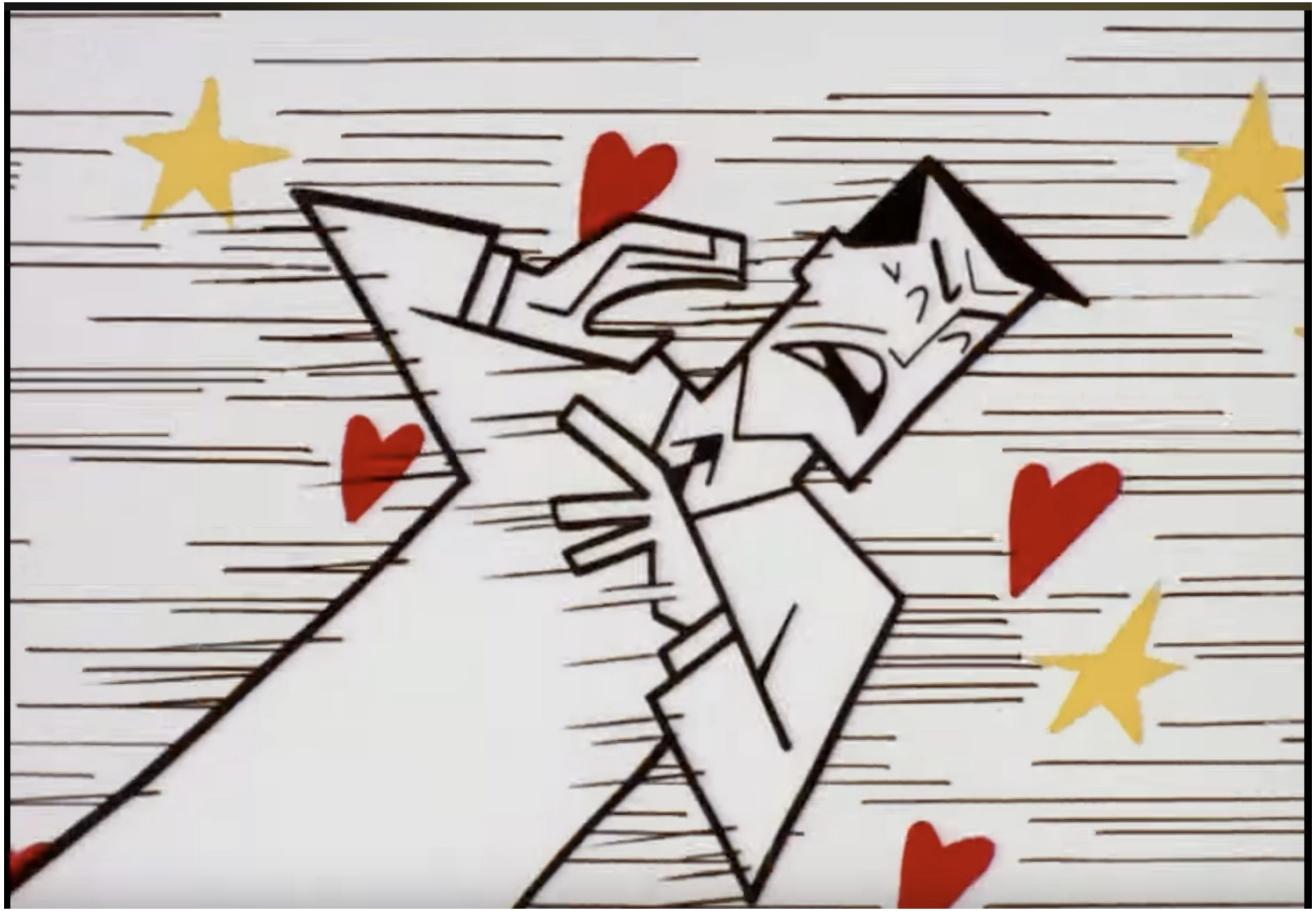
Zarr Community

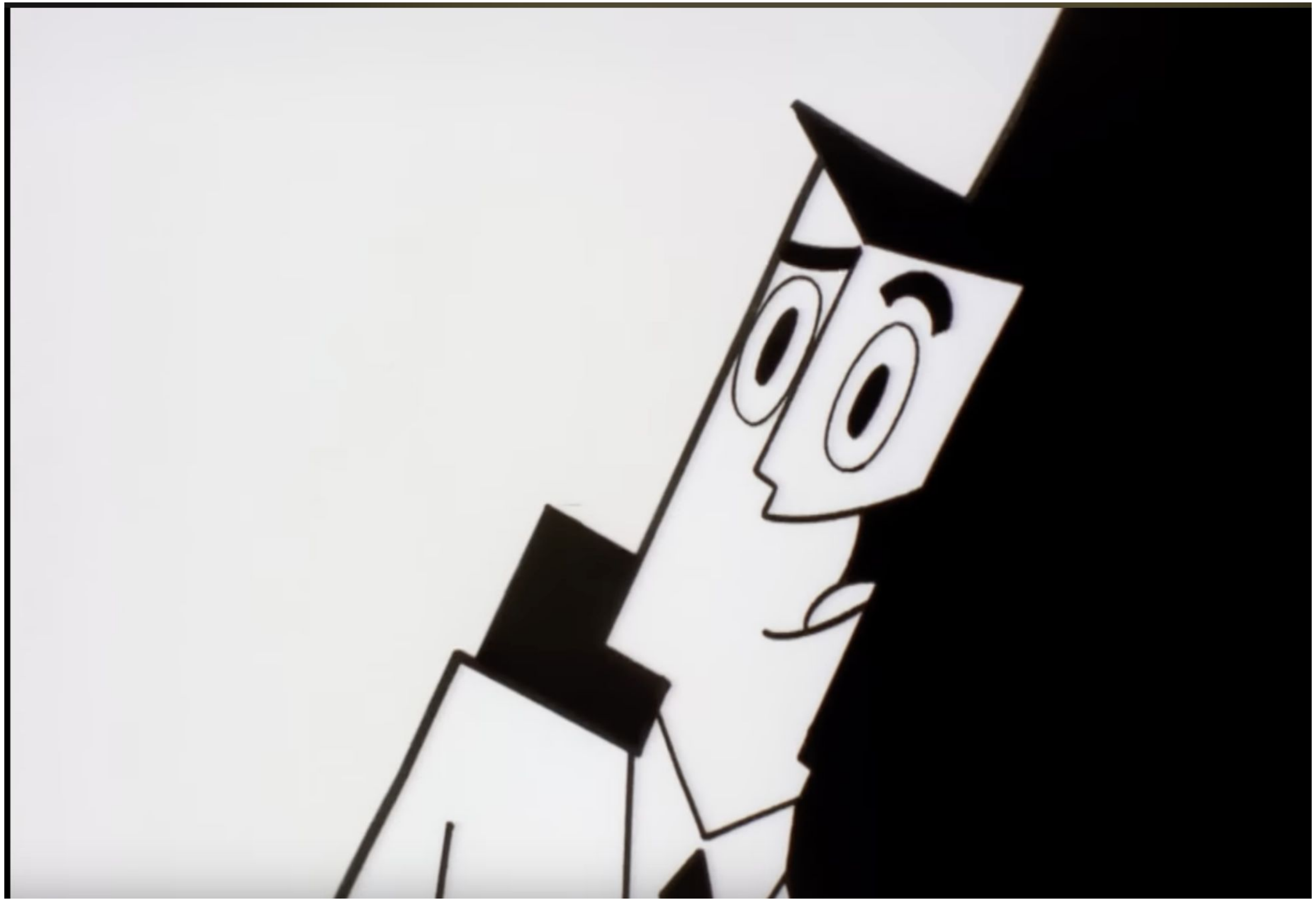
ZARR OSS
SUGAR
AND SPECIFICATION
imgflip.com

USERS & DEVS
CONTRIBUTORS & MAINTAINERS
imgflip.com

GOVERNANCE AND COUNCIL
imgflip.com


EASY TO USE
HACKABLE AND AGNOSTIC
imgflip.com





Zarr

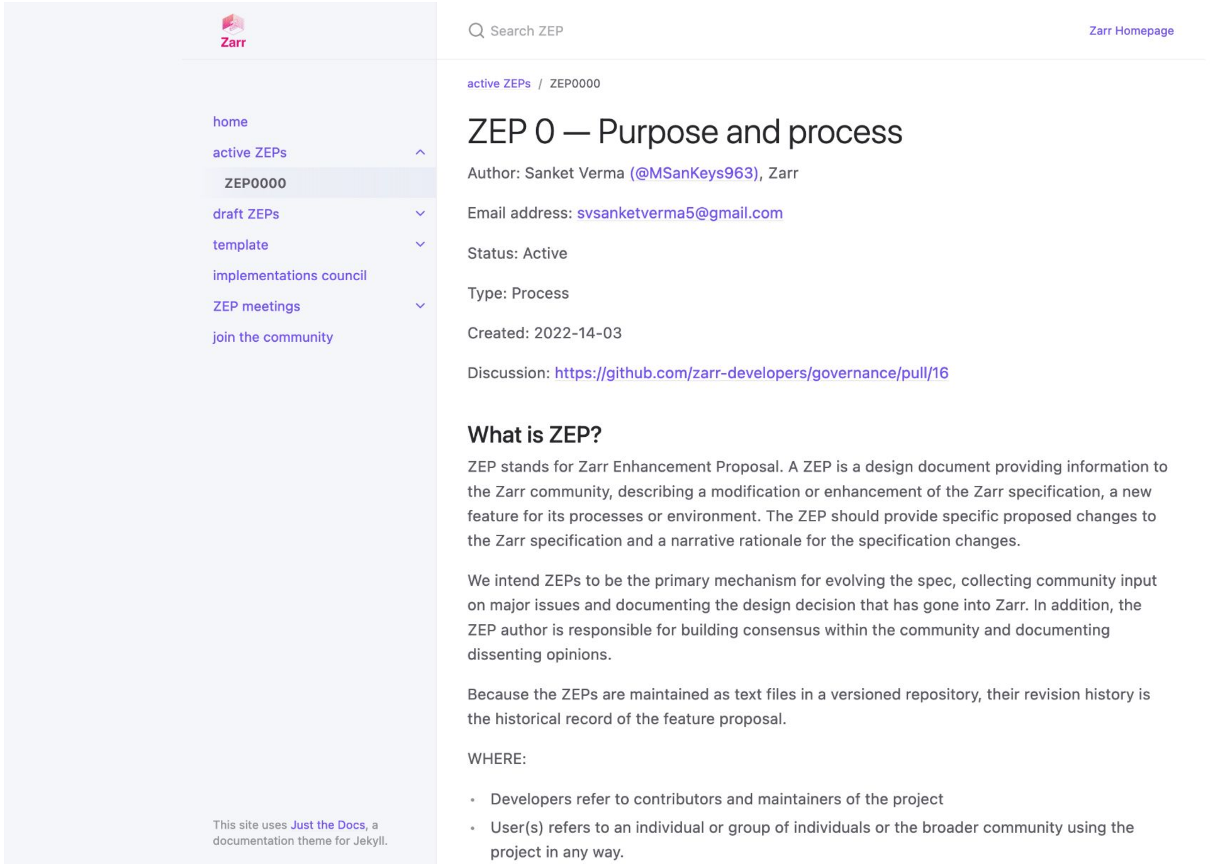
We have a large and diverse active community!

But...👀

STANDARDIZING DATA FORMAT(S)
Zarr
ACHIEVING CONSENSUS
Zarr
STANDARDIZING DATA FORMAT(S)
imgflip.com

We needed a structured way to solicit and *process* the feedback!

zarr.dev/zeps



active ZEPs / ZEP0000

ZEP 0 — Purpose and process

Author: Sanket Verma (@MSanKeys963), Zarr

Email address: svsanketverma5@gmail.com

Status: Active

Type: Process

Created: 2022-14-03

Discussion: https://github.com/zarr-developers/governance/pull/16

What is ZEP?

ZEP stands for Zarr Enhancement Proposal. A ZEP is a design document providing information to the Zarr community, describing a modification or enhancement of the Zarr specification, a new feature for its processes or environment. The ZEP should provide specific proposed changes to the Zarr specification and a narrative rationale for the specification changes.

We intend ZEPs to be the primary mechanism for evolving the spec, collecting community input on major issues and documenting the design decision that has gone into Zarr. In addition, the ZEP author is responsible for building consensus within the community and documenting dissenting opinions.

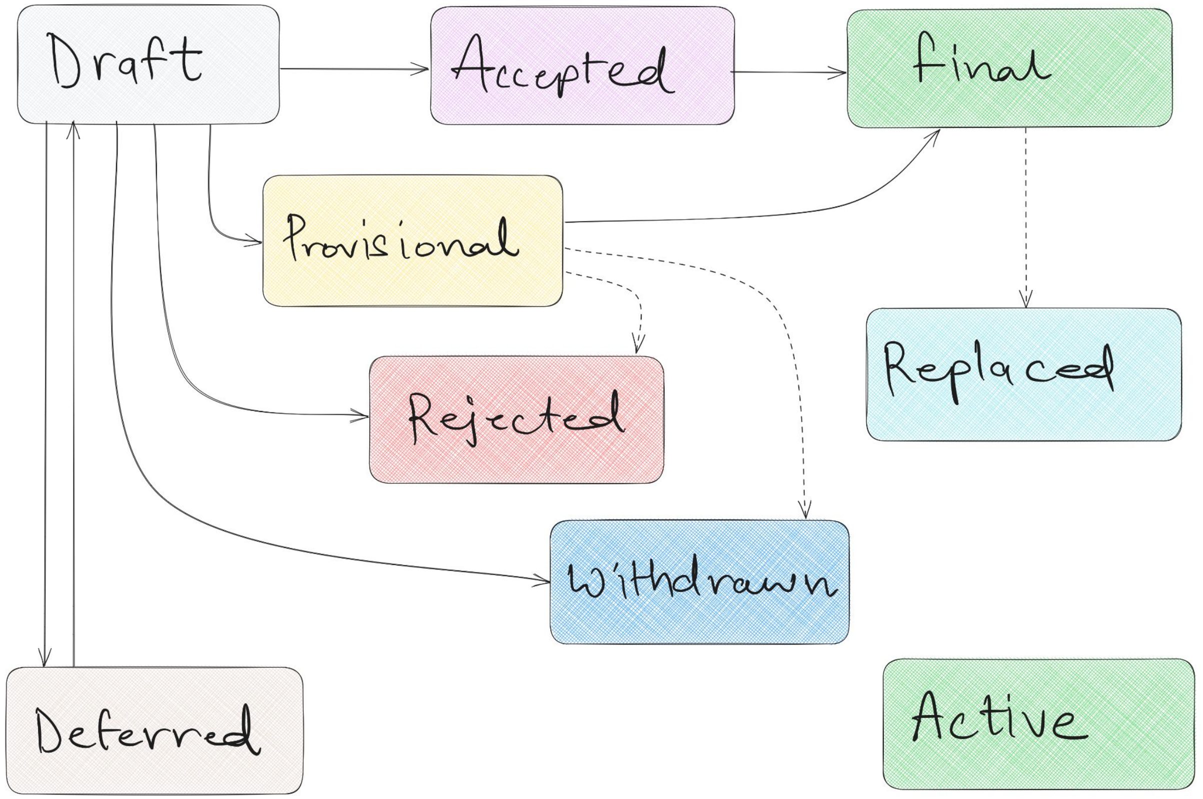
Because the ZEPs are maintained as text files in a versioned repository, their revision history is the historical record of the feature proposal.

WHERE:

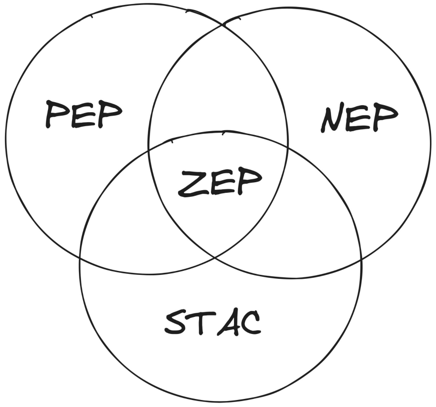
- Developers refer to contributors and maintainers of the project
- User(s) refers to an individual or group of individuals or the broader community using the project in any way.

This site uses Just the Docs, a documentation theme for Jekyll.

ZEP Flowchart ↱↖↔

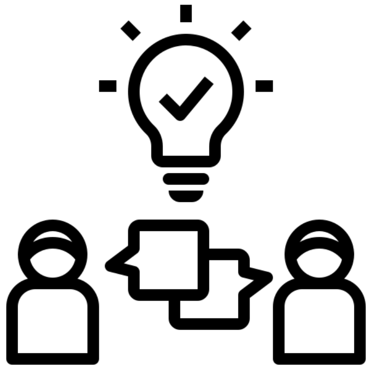


How we did it? 🚶



Lots of Reading

Previous Experience

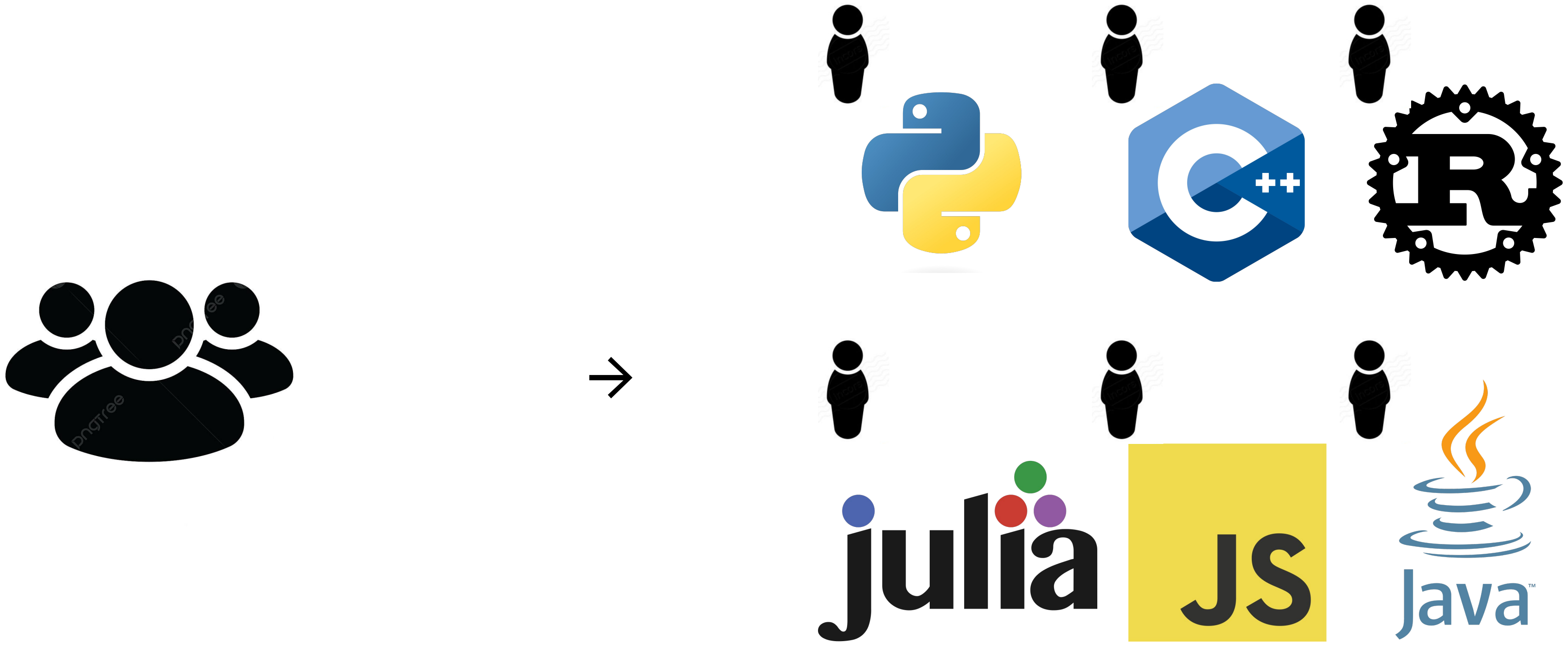


Understanding the needs of the community

Braindump 🧠

ZEP Inception Blog post

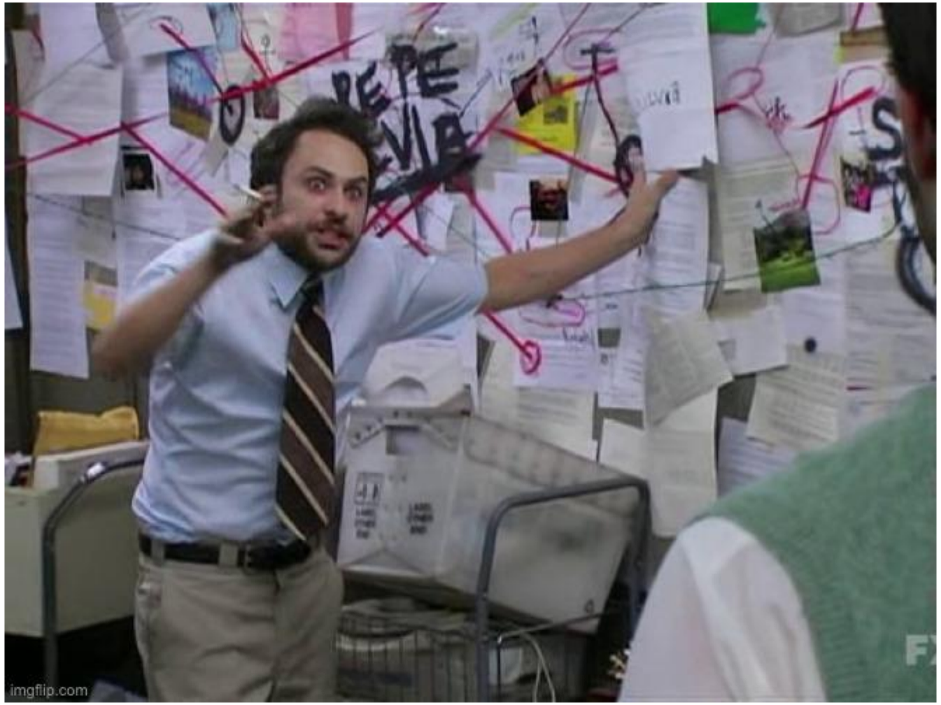
How do we adopt a ZEP?


julia
JS
Java

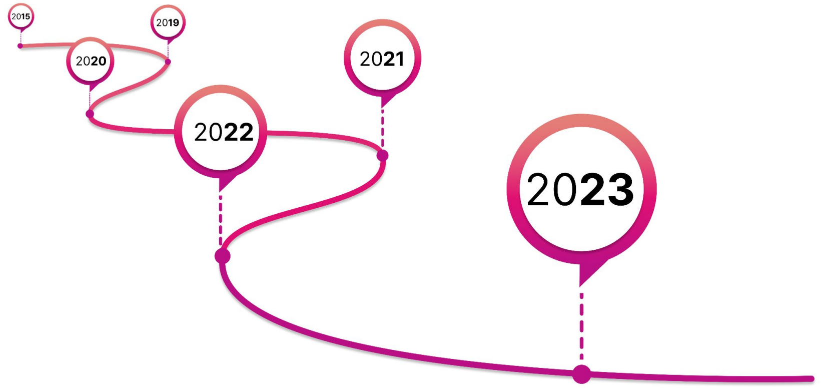
ZARR COMMUNITY
ZIC
ZSC
ADOPTING A ZEP
imgflip.com

🚧

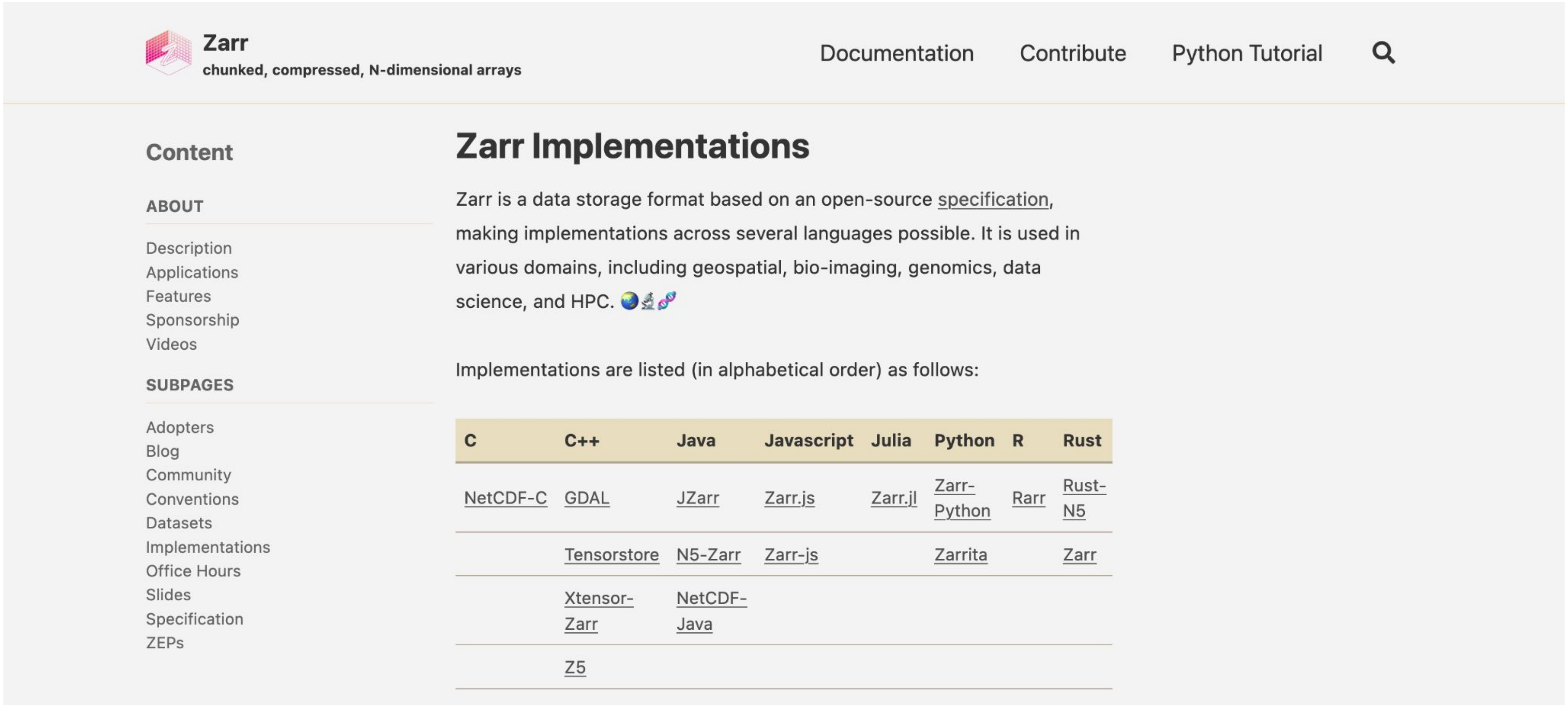
Work in progress! 👷



Evolution



zarr.dev/implementations



Zarr
chunked, compressed, N-dimensional arrays

Documentation Contribute Python Tutorial

Content

ABOUT

Description
Applications
Features
Sponsorship
Videos

SUBPAGES

Adopters
Blog
Community
Conventions
Datasets
Implementations
Office Hours
Slides
Specification
ZEPs

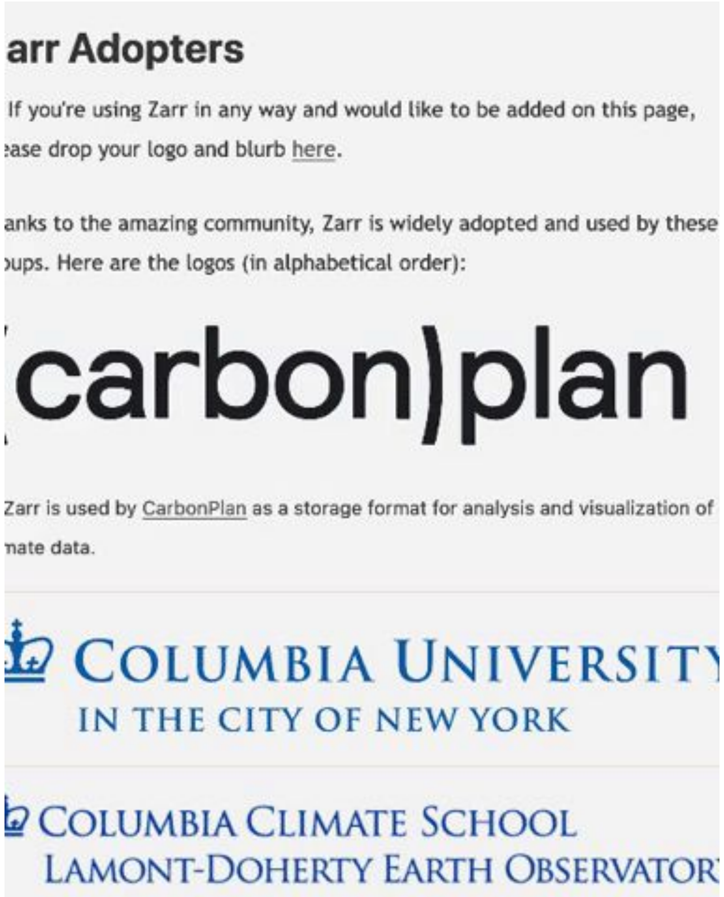
Zarr Implementations

Zarr is a data storage format based on an open-source specification, making implementations across several languages possible. It is used in various domains, including geospatial, bio-imaging, genomics, data science, and HPC. 🌏🔬🧬

Implementations are listed (in alphabetical order) as follows:

C	C++	Java	Javascript	Julia	Python	R	Rust
NetCDF-C	GDAL	JZarr	Zarr.js	Zarr.jl	Zarr-Python	Rarr	Rust-N5
	Tensorstore	N5-Zarr	Zarr-js		Zarrita		Zarr
	Xtensor-Zarr	NetCDF-Java					
	Z5						

zarr.dev/adopters



arr Adopters

If you're using Zarr in any way and would like to be added on this page, ease drop your logo and blurb here.

anks to the amazing community, Zarr is widely adopted and used by these oups. Here are the logos (in alphabetical order):

carbon)plan

Zarr is used by CarbonPlan as a storage format for analysis and visualization of nate data.

COLUMBIA UNIVERSITY IN THE CITY OF NEW YORK

COLUMBIA CLIMATE SCHOOL LAMONT-DOHERTY EARTH OBSERVATOR

/zarr-developers/{gsoc,outreachy}

Google Summer of Code

OUTREACHY

WHAT DID WE LEARN?
imgflip.com

Lessons 📖

→ Creative ways to foster the community & OSS 🎨

→ Establishing trust is critical for standardization 🤝

→ Everyone on the same page - difficult but achievable 🔗

→ Be considerate to everyone! 🫀

Thank you! 🙏

@zarr_dev