Package: romeo (via r-universe)

July 3, 2026

Title Minimal R 'OME-Zarr' Reader

Version 0.99.1

Description A minimal R package to reading, writing and validating multiscale OME-Zarr images.

License MIT + file LICENSE

URL https://huber-group-embl.github.io/romeo/, https://github.com/Huber-group-EMBL/romeo

BugReports https://github.com/Huber-group-EMBL/romeo/issues

Depends R (>= 4.6)

Imports cli, EBImage, jsonlite, jsonvalidate, methods, Rarr, ZarrArray

Suggests BiocStyle, knitr, spatialdataR, testthat, utils, withr

VignetteBuilder knitr

biocViews DataImport

Config/roxygen2/version 8.0.0

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/pak/sysreqs libblosc-dev cmake libfftw3-dev make libjpeg-dev libpng-dev libtiff-dev libuv1-dev libxml2-dev libzstd-dev libssl-dev libnode-dev zlib1g-dev

Repository https://bisaloo.r-universe.dev

Date/Publication 2026-06-24 09:46:54 UTC

RemoteUrl https://github.com/Huber-group-EMBL/romeo

RemoteRef HEAD

RemoteSha c6c4d0eb7adf6607dbd740ba986479b911a11c5c

Contents

`[.ome_zarr` *Subset an* `ome-zarr` *object*

Description

Subset operation is applied on all levels of the multiscale `ome-zarr` object. The result is an `ome-zarr` object with the same number of levels, but each level is subsetted according to the provided indices.

Usage

```
## S3 method for class 'ome_zarr'
x[...]
```

Arguments

`x`	An `ome-zarr` object.
`...`	Indices to subset the `ome-zarr` object.

Details

The first image is subsetted using the provided indices, and the resulting dimensions are used to subset the remaining levels, while conserving the same scaling factor across levels

Value

A subset of an object of `ome_zarr` (OME-Zarr) class representing an image or label pyramid.

Examples

```
omezarrzip <- system.file("extdata",
                          "test_ngff_image_v04.ome.zarr.zip",
                          package = "romeo")
dir.create(td <- tempfile())
unzip(omezarrzip, exdir = td)
x <- ome_read(td)
y <- x[1:5,1:5]
plot(y, level = 2)
```

extract_levels *Extract specific levels from a multiscale* ome-zarr *object*

Description

Extract specific levels from a multiscale ome-zarr object

Usage

```
extract_levels(x, levels)
```

Arguments

x	An ome-zarr object.
levels	Integer vector specifying the levels to extract.

Value

- If levels is of length 1, an array
- If levels is of length more than 1, an ome-zarr object

An object of ome_zarr (OME-Zarr) class representing an image or label pyramid.

Examples

```
omezarrzip <- system.file("extdata",
                          "test_ngff_image_v04.ome.zarr.zip",
                          package = "romeo")
dir.create(td <- tempfile())
unzip(omezarrzip, exdir = td)
x <- ome_read(td)
extract_levels(x, c(1, 3))
extract_levels(x, 2)
```

ome_read *Read a multiscale OME-Zarr file*

Description

Read a multiscale OME-Zarr file

Usage

```
ome_read(path, s3_client = NULL, lazy = TRUE, validate = TRUE)
```

Arguments

path	Path to the OME-Zarr file.
s3_client	Object created by paws.storage::s3(). Only required for a file on S3. Leave as NULL for a file on local storage.
lazy	Logical. If TRUE (the default), use **ZarrArray** to read data lazily. If FALSE, read data into memory using **Rarr**. If the data can fit into memory, setting lazy = FALSE may result in better performance.
validate	Logical.If TRUE (the default), validate the OME-Zarr file.

Value

An object of ome_zarr (OME-Zarr) class representing an image or label pyramid.

Examples

```
omezarrzip <- system.file("extdata",
                          "test_ngff_image_v04.ome.zarr.zip",
                          package = "romeo")
dir.create(td <- tempfile())
unzip(omezarrzip, exdir = td)
x <- ome_read(td)
```

---

ome_validate *Validate a multiscale OME-Zarr file*

Description

Validate a multiscale OME-Zarr file

Usage

```
ome_validate(path, s3_client = NULL)
```

Arguments

path	Path to the OME-Zarr file.
s3_client	Object created by paws.storage::s3(). Only required for a file on S3. Leave as NULL for a file on local storage.

Value

This function is used for its side-effect and will return the type of the OME-Zarr schema (image, label), otherwise will invoke an error when passed an invalid OME-Zarr file

Examples

```
omezarrzip <- system.file("extdata",
                          "test_ngff_image_v04.ome.zarr.zip",
                          package = "romeo")
dir.create(td <- tempfile())
unzip(omezarrzip, exdir = td)
ome_validate(td)
```

---

ome_write *Write a multiscale OME-Zarr file*

Description

Writes an ome image to the zarr path according to ome-zarr specification

Usage

```
ome_write(
  image,
  path = "/",
  axes = NULL,
  scalefactors = c(2, 2, 2, 2),
  version = c("0.5", "0.4"),
  storage_options = NULL,
  type = c("image", "label"),
  label_name = NULL,
  label_metadata = NULL
)

## S4 method for signature 'array'
ome_write(
  image,
  path = "/",
  axes = NULL,
  scalefactors = c(2, 2, 2, 2),
  version = c("0.5", "0.4"),
  storage_options = NULL,
  type = c("image", "label"),
  label_name = NULL,
  label_metadata = NULL
)

## S4 method for signature 'Image'
ome_write(
  image,
  path = "/",
  axes = NULL,
```

```
  scalefactors = c(2, 2, 2, 2),
  version = c("0.5", "0.4"),
  storage_options = NULL,
  type = c("image", "label"),
  label_name = NULL,
  label_metadata = NULL
)
```

Arguments

image	an n-dimensional (or a path to an) array representing the image data (1<n<6)
path	the path to writing ome.zarr
axes	a character vector specifying the axes of the image (e.g. c("t", "c", "z", "y", "x"))
scalefactors	Scale factors to apply to construct a multiscale image. Importantly, each scale factor is relative to the previous scale factor. For example, if the scale factors are c(2, 2, 2), the returned multiscale image will have 4 scales.
version	OME-Zarr version (0.4 or 0.5), lower versions are not supported for writing.
storage_options	a list of storage options for the zarr array (e.g. chunks)
type	The type of OME pyramid written: 'image' (default) or 'label'.
label_name	The name of the label data.
label_metadata	label metadata added to attributes.

Value

An object of ome_zarr (OME-Zarr) class representing an image or label pyramid.

Examples

```
library(EBImage)
nuc <- readImage(system.file("images", "nuclei.tif", package="EBImage"))
nuc <- getFrames(nuc)[[1]]
td <- tempfile(fileext = ".ome.zarr")

# write image pyramid
ome_nuc <- ome_write(nuc,
                     path = td,
                     version = "0.4",
                     storage_options = list(chunk_dim = c(64,64)))

# nuclei segmentation using otsu's method
nuc_th = nuc > otsu(nuc)

# write label pyramid
ome_nuc_th <- ome_write(nuc_th,
                        path = td,
                        version = "0.4",
                        scalefactors = c(2,2,3),
```

```
                        storage_options = list(chunk_dim = c(64,64)),
                        label_name = "nuclei_segmentation",
                        type = "label")
```

`plot.ome_zarr` *Plot an* `ome_zarr` *object.*

Description

Plot an `ome_zarr` object.

Usage

```
## S3 method for class 'ome_zarr'
plot(x, level = 1, ...)
```

Arguments

`x`	An `ome_zarr` object.
`level`	Integer. The scale level to plot. Defaults to `1` (the highest resolution).
`...`	Additional arguments passed to `plot()`.

Value

None

Examples

```
omezarrzip <- system.file("extdata",
                          "test_ngff_image_v04.ome.zarr.zip",
                          package = "romeo")
dir.create(td <- tempfile())
unzip(omezarrzip, exdir = td)
x <- ome_read(td)
plot(x)
plot(x, 2)
plot(x, all = TRUE)
```

Index