# Package: tidyff (via r-universe)

August 31, 2024

**Title** Create Native raster Files With ff

**Version** 0.0.2.9001

**Description** Couplings between raster and ff files.

**Depends** R (>= 3.3.0)

**Imports** activate, configr, ff, raster, tidync, tibble, yesno

**License** GPL-3

**LazyData** true

**RoxygenNote** 6.0.1

**Suggests** digest, covr, knitr, rmarkdown, testthat, viridis

**VignetteBuilder** knitr

**Remotes** r-gris/activate, r-gris/tidync

**Repository** https://hypertidy.r-universe.dev

**RemoteUrl** https://github.com/hypertidy/tidyff

**RemoteRef** HEAD

**RemoteSha** a7dc53317baae9e72af8a3425fe8e66c112780bf

## Contents

## activate *activate*

### Description

activate

### Usage

```
active(x, value)

## S3 method for class 'tidyff'
activate(.data, what)
```

### Arguments

| | |
|---|---|
| x | object |
| value | activation name |
| .data | object with sub-element |
| what | sub-element to activate |

## dim_order *dimension order*

### Description

The dimension order as used by ff and raster native grid, see ‘raster::writeRaster‘. The raster package uses the named conventions "BSQ", "BIL" and "BIP" which are domain-specialized ways to record the dimension order of (virtual) 3D arrays. They stand for "band sequential" "band interleaved" and ... correspond to c(1, 2, 3) etc. etc.

### Usage

```
dim_order(x, ...)

## S3 method for class 'BasicRaster'
dim_order(x, ...)

## S3 method for class 'character'
dim_order(x, ...)
```

### Arguments

| | |
|---|---|
| x | filename or raster object or ff object |
| ... | ignored |

## Details

This function always returns an integer vector indicating the order of the dims of a ‘length(dim_order(x))‘ array oriented relative to the R array convention.

## Value

integer vector of dimension positions (1-based)

## Examples

```
r <- raster::raster(volcano)
dim_order(r)
w <- raster::writeRaster(r, raster::rasterTmpFile(), bandorder = "BIP")
dim_order(w)
dim_order(raster::filename(w))
```

# ff_object *Raster and ’ff’ file-backed arrays*

## Description

Create an file-backed ’ff’ object from a raster, or a raster-enabled file.

## Usage

```
ff_object(x, readonly = TRUE, filename = NULL, ...)

## S3 method for class 'BasicRaster'
ff_object(x, readonly = TRUE, filename = NULL, ...)

## S3 method for class 'character'
ff_object(x, readonly = TRUE, filename = NULL, ...)
```

## Arguments

| | |
|---|---|
| `x` | raster or raster-able file |
| `readonly` | open in read-only mode (TRUE by default) |
| `filename` | path to file to create |
| `...` | arguments to methods |

## Details

When the object is created from a filename, ‘raster::brick‘ is used. The ’dim’ension for raster and for ff in this context always keeps degenerate singletons. Please get in touch if this causes you problems.

## Value

‘ff‘ object

## Examples

```
f <- system.file("extdata", "raster", "sst.grd", package  = "tidyff")
ff_object(raster::brick(f))
if (interactive()) {
  arr <- ff_object(f, filename = "afile.grd")
 }
```

---

# `ff_type` *ff type from raster type*

---

## Description

ff type from raster type

## Usage

```
ff_type(x, ...)

## S3 method for class 'character'
ff_type(x, ...)

## S3 method for class 'BasicRaster'
ff_type(x, ...)

## S3 method for class 'raster_ini'
ff_type(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | filename, raster object, or ini object |
| `...` | ignored |

## Value

type of ff, see ‘ff::vmode‘

## hyper_filter *hyper filter*

### Description

hyper filter

### Usage

```
## S3 method for class 'tidyff'
hyper_filter(x, ...)
```

### Arguments

| | |
|---|---|
| x | object |
| ... | arguments as per ‘tidync::hyper_filter‘ |

## ini_file *raster ini file*

### Description

Read the configuration file in raw list form, you can use either the filename (.grd) or a raster object.

### Usage

```
ini_file(x)

## S3 method for class 'character'
ini_file(x)

## S3 method for class 'BasicRaster'
ini_file(x)
```

### Arguments

| | |
|---|---|
| x | file name or raster |

### Value

ini object from raster native binary

### Examples

```
f <- system.file("extdata", "raster", "sst.grd", package  = "tidyff")
ini_file(f)
```

## `tidyff` *tidyff.*

### Description

tidyff.

Tidy ff

### Usage

```
tidyff(x, ...)
```

### Arguments

| | |
|---|---|
| `x` | object |
| `...` | ignored |

# Index