Package: arttools (via r-universe)

August 30, 2024

Type Package

Title Tools For Generative Art Workflows

Version 0.0.0.9000

Description A personal-use package for managing generative art workflows.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/djnavarro/arttools, http://djnavarro.net/arttools/

BugReports https://github.com/djnavarro/arttools/issues

Imports brio, cli, curl, fs, here, httr2, magick, readr, rlang, tibble, waldo

RoxygenNote 7.2.3

Config/Needs/website rmarkdown

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

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

RemoteUrl https://github.com/djnavarro/arttools

RemoteRef HEAD

RemoteSha 08d7225e69e7172d42c6431d4d4dc42e5ac35c14

Contents

`bucket_download` *Download series from remote bucket to local storage*

Description

Download series from remote bucket to local storage

Usage

```
bucket_download(
  series,
  local_path = bucket_local_path(),
  remote_url = bucket_remote_url()
)
```

Arguments

`series`	Name of the series
`local_path`	Local bucket folder to download to
`remote_url`	Remote bucket folder to download from

Value

This function is called for its side effect, downloading series files from a remote bucket storage to local storage. Invisibly returns a tibble containing the download status information, as reported by `curl::multi_download()`.

`bucket_remote_url` *Paths to object buckets and repositories*

Description

Paths to object buckets and repositories

Usage

```
bucket_remote_url(...)

bucket_local_path(...)

repo_remote_url(...)

repo_local_path(...)
```

Arguments

...	Names of subfolders

Value

Fully qualified path or url

---

create_resized_images *Create resized copies of all images in a directory*

Description

Create resized copies of all images in a directory

Usage

```
create_resized_images(
  series,
  images_from,
  images_to,
  pixels_wide,
  pixels_high = pixels_wide,
  origin = bucket_local_path()
)
```

Arguments

series	Name of the series
images_from	Folder within the series containing original images
images_to	Folder within the series to contain resized images
pixels_wide	Width of output images in pixels
pixels_high	Height of output images in pixels description
origin	Location in which to find the series

Value

Invisibly returns NULL. This may change

`manifest_read` *Read and write manifest files*

Description

The manifest for a series is a plain text file that lists the files contained within the series.

Usage

```
manifest_read(series, origin = bucket_remote_url())

manifest_write(series, date = Sys.Date(), origin = bucket_local_path())

manifest_build(series, date, origin = bucket_local_path())
```

Arguments

`series`	Name of the series (e.g., "series-rosemary")
`origin`	Location in which to find the series
`date`	Publication date for the series (defaults to current date)

Details

The manifest file is used to document the content of a published art series, and `manifest_*()` functions provide tools to work with manifests with a minimum of pain. The typical workflow is expected to be as follows. Once an art series has been finalised and all images have been written into the "local bucket folder", use `manifest_write()` to create a manifest file within the local series folder. Once that is done, you can upload the completed series to the "remote bucket folder".

Whenever you need to inspect the contents of the now-online series, read the manifest file from the remote bucket using `manifest_read()`. As an example, you can use this to programmatically construct an HTML document that displays all images in a series, by calling `manifest_read()` within a code chunk in a quarto or R markdown document.

This workflow is one in which manifests are constructed locally, published to a remote, and then read from the remote location. For that reason, the default behaviour is that `manifest_read()` sets the `origin` to `bucket_remote_url()`, whereas `manifest_write()` sets the `origin` and `destination` to `bucket_local_path()`.

In some cases it may be convenient to construct the manifest tibble from the series image files without writing it to a csv file. To that end there is also a `manifest_build()` function that does this.

Value

Tibble containing the manifest data, returned visibly to the user by `manifest_read()` and `manifest_build()`, and invisibly by `manifest_write()`. The tibble contains one row per image file in the series, and the following columns:

- `series_name` is a string with the series name (e.g., "series-rosemary")
- `series_date` contains the publication date in YYYY-MM-DD format
- `path` specifies the path to the image (e.g., "800/rosemary_001_1000_short-title.png")
- `folder` specifies the directory part of the path (e.g., "800")
- `file_name` specifies the file name part of the path (e.g., "rosemary_001_1000_short-title.png")
- `file_format` specifies the file format for the image (e.g., "png")
- `system_name` specifies the generative art system name (e.g., "rosemary")
- `system_version` specifies the generative art system version identifier (e.g., "001")
- `image_id` specifies the identifier string for the image (e.g., "1000")
- `image_short_title` specifies the short title for the image, or NA (e.g., "short-title")
- `image_long_title` specifies the long title for the image, or NA (e.g., "The Long Title")
- `manifest_version` specifies the version of the manifest format used (currently always 1)

For `manifest_read()` and `manifest_build()` returning this tibble is the only thing it does. For `manifest_write()` the tibble is written to a csv file whose location is specified using the `destination` argument.

`repo_create` *Create new art repository*

Description

Create new art repository

Usage

```
repo_create(
  series,
  license = NULL,
  local_path = repo_local_path(),
  remote_url = repo_remote_url()
)
```

Arguments

`series`	Name of the series (e.g., "series-rosemary")
`license`	License type for the series ("ccby", "cc0", or "mit")
`local_path`	Local folder in which to create the repository
`remote_url`	Remote URL to check for a pre-existing repository, or NULL to skip check

Value

Invisibly returns TRUE on success, FALSE on failure (this may change)

`status` *Status of a repository or bucket*

Description

Status of a repository or bucket

Usage

```
bucket_status(series, local_path = bucket_local_path())

repo_status(series, local_path = repo_local_path())
```

Arguments

`series`	Name of the series
`local_path`	Location in which to find the bucket or repository folder

Value

Invisibly returns TRUE if no problems are detected, FALSE otherwise

Index