

Package: movepub (via r-universe)

October 15, 2026

Title Prepare Movebank Data for Publication

Version 0.4.0.9000

Date 2025-08-27

Description Prepare animal tracking data from 'Movebank' (<https://www.movebank.org/>) for publication in a research repository. With 'movepub' you can document data with metadata following the Data Package standard and transform these to Darwin Core and Ecological Metadata Language ('EML') for publication to the Global Biodiversity Information Facility ('GBIF') and the Ocean Biodiversity Information System ('OBIS').

License MIT + file LICENSE

URL <https://github.com/inbo/movepub>, <https://inbo.github.io/movepub/>

BugReports <https://github.com/inbo/movepub/issues>

Depends R (>= 4.1.0)

Imports cli, dplyr (>= 1.2.0), EML, frictionless (>= 1.3.0), jsonlite, lifecycle, move2 (>= 0.4.0), purrr, readr, rlang, stringr, tidyr, uuid, worrms, xml2

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.1.0

Config/pak/sysreqs

libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make libicu-dev libjq-dev libuv1-dev libxml2-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libnode-dev libx11-dev

Repository <https://inbo.r-universe.dev>

Date/Publication 2026-10-15 14:08:09 UTC

```
RemoteUrl https://github.com/inbo/movepub
RemoteRef HEAD
RemoteSha d9ca56f021ccc48e61d099f4382947e003b4187e
```

Contents

add_resource	2
datacite_to_eml	3
get_aphia_id	3
html_to_docbook	4
o_assen	5
write_dwc	6
write_eml	8

Index	11
-------	----

add_resource	<i>Add Movebank data to a Data Package</i>
--------------	--

Description

Adds Movebank data (reference-data, gps, acceleration, accessory-measurements) as a Data Resource to a Data Package. The function extends `frictionless::add_resource()` by adding the following to the Table Schema of the resource:

- The title, definition, format and URI for each field, from the latest version of the **Movebank Attribute Dictionary**.
- The primary key of the resource and foreign keys between resources.

Usage

```
add_resource(package, resource_name, files, keys = TRUE)
```

Arguments

package	Data Package object, as returned by <code>read_package()</code> or <code>create_package()</code> .
resource_name	Name of the Data Resource.
files	One or more paths to CSV file(s) that contain the data for this resource, as a character (vector).
keys	If TRUE, primaryKey and foreignKey properties are added to the Table Schema.

Details

See `vignette("movepub")` for examples.

Value

package with one additional resource.

datacite_to_eml	<i>Get DataCite metadata as EML</i>
-----------------	-------------------------------------

Description

Get metadata from **DataCite** and transform to EML.

Usage

```
datacite_to_eml(doi)
```

Arguments

doi DOI of a dataset.

Value

EML list that can be extended and/or written to file with `EML::write_eml()`.

See Also

Other support functions: `get_aphia_id()`, `html_to_docbook()`

Examples

```
datacite_to_eml("10.5281/zenodo.10053903")
```

get_aphia_id	<i>Get WoRMS AphiaID from a taxonomic name</i>
--------------	--

Description

This function wraps `worrms::wm_name2id_()` so that it returns a data frame rather than a list. It also silences "not found" warnings, returning NA instead.

Usage

```
get_aphia_id(x)
```

Arguments

x A (vector with) taxonomic name(s).

Value

Data frame with name, `aphia_id`, `aphia_lsid` and `aphia_url`.

See Also

Other support functions: [datacite_to_eml\(\)](#), [html_to_docbook\(\)](#)

Examples

```
get_aphia_id("Mola mola")
get_aphia_id(c("Mola mola", "not_a_name"))
```

html_to_docbook	<i>Convert text with HTML to DocBook</i>
-----------------	--

Description

Converts text with HTML syntax to **DocBook**, splitting paragraphs and headers into separate elements. Only a subset of HTML tags are supported (see transformation details), all other HTML syntax is removed.

Usage

```
html_to_docbook(string)
```

Arguments

string Character (vector) that may contain HTML syntax.

Value

A character vector with HTML converted to DocBook.

Transformation details

The function splits text into a character vector, with one element for each paragraph, header or line break (\n). The remaining HTML is converted to DocBook, but only tags those supported by EML for **paragraphs**. All other HTML/DocBook syntax is sanitized and empty elements are removed.

Input	Output
...	... (separate element)
...	... (separate element)
...	... (separate element)
...	... (separate element)
...	... (separate element)
...	... (separate element)
...	... (separate element)
...	... (separate element)
... \n	... (separate element)
...	<itemizedlist>...</itemizedlist>
...	<orderedlist>...</orderedlist>

...	<listitem><para>...</para></listitem>
...	<emphasis>...</emphasis>
<i>...</i>	<emphasis>...</emphasis>
...	<emphasis>...</emphasis>
...	<emphasis>...</emphasis>
_{...}	<subscript>...</subscript>
^{...}	<superscript>...</superscript>
<pre>...</pre>	<literalLayout>...</literalLayout>
...	<ulink url="https://example.com"><citetitle>...</citetitle></ulink>
<code>...</code>	... (HTML element sanitized)
<foo>...</foo>	... (HTML element sanitized)
...	... (HTML property sanitized)
<p class="small">...</p>	... (HTML property sanitized)
	empty string (HTML element sanitized)
<emphasis>...</emphasis>	... (DocBook element sanitized)

Use with EML

- 1. Capture EML with `eml <- movepub::write_eml()` or read with `EML::read_eml()`.
- 2. Assign output of `html_to_docbook()` to `eml$dataset$abstract$para`.
- 3. Write EML with `EML::write_eml()`.

See Also

Other support functions: [datacite_to_eml\(\)](#), [get_aphia_id\(\)](#)

Examples

```
html_to_docbook(  
  c(  
    "This is <b>bold</b>.\nParagraph 1\n\nParagraph 2<p></p>",  
    "What follows is a list: <ul><li>Item 1</li><li>Item 2</li></ul>"  
  )  
)
```

o_assen	Sample Movebank dataset with GPS tracking data
---------	--

Description

A sample Movebank dataset with GPS tracking data, formatted as a **Data Package** and read by [read_package\(\)](#).

Usage

```
o_assen
```

Format

An object of class datapackage (inherits from list) of length 6.

Details

This sample is derived from the Zenodo-deposited dataset [Dijkstra et al. \(2023\)](#), but excludes the acceleration data.

Source

[doi:10.5281/zenodo.10053903](https://doi.org/10.5281/zenodo.10053903)

Examples

```
## Not run:
# The data in o_assen was created with the code below
o_assen <-
  read_package("https://zenodo.org/records/10053903/files/datapackage.json") |>
  remove_resource("acceleration")
o_assen$title <- paste(
  "O_ASSEN - Eurasian oystercatchers (Haematopus ostralegus, Haematopodidae)",
  "breeding in Assen (the Netherlands)"
)
o_assen$licenses[[1]]$name <- "CC0-1.0"
o_assen$contributors[[1]]$title <- "Vogelwerkgroep Assen"
o_assen$contributors[[1]]$role <- "rightsHolder"
usethis::use_data(o_assen, overwrite = TRUE)

## End(Not run)
```

write_dwc

Transform a Data Package with Movebank data to a Darwin Core Archive

Description

Transforms a Data Package with Movebank data to a [Darwin Core Archive](#).

Usage

```
write_dwc(
  package,
  directory,
  dataset_id = package$id,
  dataset_name = package$title,
  license = NULL,
  rights_holder = NULL
)
```

Arguments

package	A Data Package with Movebank data, as returned by <code>read_package()</code> . It is expected to contain a reference-data and gps resource.
directory	Path to local directory to write files to.
dataset_id	Identifier for the dataset.
dataset_name	Title of the dataset.
license	License of the dataset.
rights_holder	Acronym of the organization owning or managing the rights over the data.

Details

The resulting files can be uploaded to an **IPT** for publication to GBIF and/or OBIS. A corresponding eml.xml metadata file can be created with `write_eml()`. See `vignette("movepub")` for an example.

Value

CSV and meta.xml files written to disk. And invisibly, a list of data frames with the transformed data.

Transformation details

This function **follows recommendations** suggested by Peter Desmet, Sarah Davidson, John Wiczorek and others and transforms data to:

- An **Occurrence core**.
- An **Extended Measurement Or Facts extension**
- A meta.xml file.

Key features of the Darwin Core transformation:

- Deployments (animal+tag associations) are parent events, with tag attachment (a human observation) and GPS positions (machine observations) as child events. No information about the parent event is provided other than its ID, meaning that data can be expressed in an Occurrence core with one row per observation and parentEventID shared by all occurrences in a deployment.
- The tag attachment event often contains metadata about the animal (sex, life stage, comments) and deployment as a whole. Sex and life stage are additionally provided in an Extended Measurement Or Facts extension, where values are mapped to a controlled vocabulary recommended by **OBIS**.
- No event/occurrence is created for the deployment end, since the end date is often undefined, unreliable and/or does not represent an animal occurrence.
- Only visible (non-outlier) GPS records that fall within a deployment are included.
- GPS positions are subsampled to the **first GPS position per hour**, to reduce the size of high-frequency data. It is possible for a deployment to contain no GPS positions, e.g. if the tag malfunctioned right after deployment.

- Parameters or metadata are used to set the following record-level terms:
 - dwc:datasetID: dataset_id, defaulting to package\$id.
 - dwc:datasetName: dataset_name, defaulting to package\$title.
 - dcterms:license: license, defaulting to the first license name (e.g. CC0-1.0) in package\$licenses.
 - dcterms:rightsHolder: rights_holder, defaulting to the first contributor in package\$contributors with role rightsHolder.

Required data

The source data should have the following resources and fields:

- **reference-data** with at least the fields animal-id, animal-taxon, and tag-id. Records must have a deploy-on-date to be retained.
- **gps** with at least the fields individual-local-identifier, tag-local-identifier, and timestamp. Records must have a location-lat, visible = TRUE and a link with the reference data to be retained.

See Also

Other transformation functions: [write_eml\(\)](#)

Examples

```
write_dwc(o_assen, directory = "my_directory")

# Clean up (don't do this if you want to keep your files)
unlink("my_directory", recursive = TRUE)
```

write_eml

Transform Movebank metadata to EML

Description

Transforms the metadata of a published Movebank dataset (with a DOI) to an **Ecological Metadata Language (EML)** file.

Usage

```
write_eml(
  doi,
  directory,
  contact = NULL,
  study_id = NULL,
  derived_paragraph = TRUE
)
```

Arguments

doi	DOI of the original dataset, used to get metadata.
directory	Path to local directory to write files to.
contact	Person to be set as resource contact and metadata provider. To be provided as a person() .
study_id	Identifier of the Movebank study from which the dataset was derived (e.g. 1605797471 for this study).
derived_paragraph	If TRUE, a paragraph will be added to the abstract, indicating that data have been transformed using write_dwc() .

Details

The resulting EML file can be uploaded to an [IPT](#) for publication to GBIF and/or OBIS. A corresponding Darwin Core Archive can be created with [write_dwc\(\)](#). See `vignette("movepub")` for an example.

Value

eml.xml file written to disk. And invisibly, an [EML::eml](#) object.

Transformation details

Metadata are derived from the original dataset by looking up its doi in DataCite ([example](#)) and transforming these to EML. The following properties are set:

- **title:** Original dataset title.
- **description:** Original dataset description. If `derived_paragraph = TRUE` a generated paragraph is added, e.g.:
Data have been standardized to Darwin Core using the [movepub](#) R package and are subsampled to the first GPS position per hour. The original data are available in Dijkstra et al. (2023, [doi:10.5281/zenodo.10053903](#)), a deposit of Movebank study [1605797471](#).
- **license:** License of the original dataset.
- **creators:** Creators of the original dataset.
- **contact:** contact or first creator of the original dataset.
- **metadata provider:** contact or first creator of the original dataset.
- **keywords:** Keywords of the original dataset.
- **alternative identifier:** DOI of the original dataset. As a result, no new DOI will be created when publishing to GBIF.
- **external link and alternative identifier:** URL created from `study_id` or the first derived from related identifier in the original dataset.

The following properties are not set:

- **type**
- **subtype**

- **update frequency**
- **publishing organization**
- **geographic coverage**
- **taxonomic coverage**
- **temporal coverage**
- **associated parties**
- **project data**
- **sampling methods**
- **citations**
- **collection data**: not applicable.

See Also

Other transformation functions: [write_dwc\(\)](#)

Examples

```
(write_eml(doi = "10.5281/zenodo.10053903", directory = "my_directory"))  
  
# Clean up (don't do this if you want to keep your files)  
unlink("my_directory", recursive = TRUE)
```

Index

- * **datasets**
 - o_assen, [5](#)
- * **frictionless functions**
 - add_resource, [2](#)
- * **sample data**
 - o_assen, [5](#)
- * **support functions**
 - datacite_to_eml, [3](#)
 - get_aphia_id, [3](#)
 - html_to_docbook, [4](#)
- * **transformation functions**
 - write_dwc, [6](#)
 - write_eml, [8](#)

add_resource, [2](#)

create_package(), [2](#)

datacite_to_eml, [3](#)

datacite_to_eml(), [4](#), [5](#)

EML::eml, [9](#)

EML::write_eml(), [3](#)

frictionless::add_resource(), [2](#)

get_aphia_id, [3](#)

get_aphia_id(), [3](#), [5](#)

html_to_docbook, [4](#)

html_to_docbook(), [3](#), [4](#)

o_assen, [5](#)

person(), [9](#)

read_package(), [2](#), [5](#), [7](#)

worrms::wm_name2id(), [3](#)

write_dwc, [6](#)

write_dwc(), [9](#), [10](#)

write_eml, [8](#)

write_eml(), [7](#), [8](#)