**Instructions for the EML assembly line**

**Colin Smith (EDI)**

**IEP test-case comments Brittany Davis**

Will take about a day to become familiar with EML assembly line.

<https://github.com/EDIorg/EMLassemblyline>.

Edits by Rosemary Hartman, CDFW 7 Dec 2018

**Overview**

The EML assembly line will help you create high quality metadata for your dataset. Below is a set of step-by-step instructions for making EML metadata for tabular data. The assembly line will soon be capable of handling other data types including: spatial vector, spatial raster, and images.

**Installation (periodic reinstallation is recommended)**

The assembly line is under constant revision and improvement. Please reinstall the assembly line periodically to ensure a successful experience. Installation from GitHub requires the devtools package.

# Install devtools

install.packages("devtools")

# Load devtools

library(devtools)

# Install and load EMLassemblyline

install\_github("EDIorg/EMLassemblyline")

library(EMLassemblyline)

**Step 1: Create a directory for your dataset**

Create a new directory for your dataset. This is where the metadata parts created in the assembly line process will be stored and available for editing should you need to change the content of your EML.

Name this directory after your dataset. Replace spaces with underscores (e.g. name of your directory should be name\_of\_your\_directory).

\*R does not communicate well with the OneDrive, consider working more easily on the Desktop.

**Step 2: Move your dataset to the directory**

Move copies of the final versions of your data tables into this directory. These should be the final versions of the data you are ready to publish.

Rename these files following these rules:

**Step 3: Select an intellectual rights license**

There are 2 options for intellectual rights licenses:

1. **CC0** the most accomodating of data reuse ... This data package is released to the “public domain” under Creative Commons CC0 1.0 “No Rights Reserved” (see: <https://creativecommons.org/publicdomain/zero/1.0/>). It is considered professional etiquette to provide attribution of the original work if this data package is shared in whole or by individual components. A generic citation is provided for this data package on the website [https://portal.edirepository.org](https://portal.edirepository.org/) (herein “website”) in the summary metadata page. Communication (and collaboration) with the creators of this data package is recommended to prevent duplicate research or publication. This data package (and its components) is made available “as is” and with no warranty of accuracy or fitness for use. The creators of this data package and the website shall not be liable for any damages resulting from misinterpretation or misuse of the data package or its components. Periodic updates of this data package may be available from the website. Thank you.
2. **CCBY** requires attribution ... This information is released under the Creative Commons license - Attribution - CC BY (<https://creativecommons.org/licenses/by/4.0/>). The consumer of these data ("Data User" herein) is required to cite it appropriately in any publication that results from its use. The Data User should realize that these data may be actively used by others for ongoing research and that coordination may be necessary to prevent duplicate publication. The Data User is urged to contact the authors of these data if any questions about methodology or results occur. Where appropriate, the Data User is encouraged to consider collaboration or co-authorship with the authors. The Data User should realize that misinterpretation of data may occur if used out of context of the original study. While substantial efforts are made to ensure the accuracy of data and associated documentation, complete accuracy of data sets cannot be guaranteed. All data are made available "as is." The Data User should be aware, however, that data are updated periodically and it is the responsibility of the Data User to check for new versions of the data. The data authors and the repository where these data were obtained shall not be liable for damages resulting from any use or misinterpretation of the data. Thank you.

IEP can create a customized CC statement and alter in the uploaded text box

**Step 4: Identify the types of data in your dataset**

Currently, the assembly line only works for tabular data and .zip directories.

**table**

A flat file composed of columns containing variables and rows containing observations. Column names must follow these rules:

* replace symbols with words
* replace parentheses with underscores
* replace periods with underscores
* replace blank spaces with underscores

e.g. land.cover.use (%) should be percent\_land\_cover\_use

**.zip directory**

A .zip directory containing anything you want to put into it. .zip directory name should follow the same naming rules as for a table.

**Step 5: Import the core metadata templates**

Run the function import\_templates in the RStudio Console to populate the directory with metadata templates for you to complete. You will need to supply a few arguments to this function:

1. **path** A path to your dataset working directory.
2. **license** The license for your dataset ("CC0" or "CCBY").
3. **data.files** A list of the data tables of your dataset. File extension is not required. Do not include .zip directories here, they will be added in the make\_eml step.

# First load the EMLassemblyline package

library(EMLassemblyline)

# View documentation for this function

?import\_templates

# Import templates for an example dataset licensed under CC0, with 2 tables.

import\_templates(path = "/Users/csmith/Desktop/gleon\_chloride",

license = "CC0",

data.files = c("lake\_chloride\_concentrations",

"lake\_characteristics"))

* replace symbols with words
* replace parentheses with underscores
* replace periods with underscores
* replace blank spaces with underscores

e.g. name.of.(your) d@t@.file should be name\_of\_your\_data\_file

\*\* If any column names have spaces or periods you will get an ERROR message to correct before text attributes created.

FOR EXAMPLE

> import\_templates(path = "C:/Users/bedavis/OneDrive - California Department of Water Resources/My Documents/IEP/EDI/metadata\_EMLassemblyLine",

+ license="CC0",

+ data.files=c("YBFMP\_fish and water quality data\_07252018\_QA",

+ "YBFMP\_Fish\_Taxonomy",

+ "YBFMP\_Trap\_Effort",

+ "YBFMP\_Site\_locations\_latitude\_and\_longitude"))

Checking input arguments.

Importing abstract.txt.

Importing additional\_info.txt.

Importing bounding\_boxes.txt.

Importing custom\_units.txt.

Importing instructions.html.

Importing intellectual\_rights.txt.

Importing keywords.txt.

Importing methods.txt.

Importing my\_workflow.R.

Importing personnel.txt.

Checking data.files for valid column names.

Error in import\_templates(path = "C:/Users/bedavis/OneDrive - California Department of Water Resources/My Documents/IEP/EDI/metadata\_EMLassemblyLine", :

Invalid column names detected in YBFMP\_fish and water quality data\_07252018\_QA.csv: Genetic.ID Replace characters located at periods "." in the above listed column names with underscores "\_"

-make the correction to the .csv file, re-save, and re-run. You will now see it has created attribute files that were not created before.

|  |
| --- |
| + "YBFMP\_Site\_locations\_latitude\_and\_longitude"))  Checking input arguments.  abstract.txt already exists!  additional\_info.txt already exists!  bounding\_boxes.txt already exists!  custom\_units.txt already exists!  instructions.html already exists!  intellectual\_rights.txt already exists!  keywords.txt already exists!  methods.txt already exists!  my\_workflow.R already exists!  personnel.txt already exists!  Checking data.files for valid column names.  Detecting attributes of YBFMP\_fish and water quality data\_07252018\_QA.csv.  Importing attributes\_YBFMP\_fish and water quality data\_07252018\_QA.txt.  Detecting attributes of YBFMP\_Fish\_Taxonomy.csv.  Importing attributes\_YBFMP\_Fish\_Taxonomy.txt.  Detecting attributes of YBFMP\_Trap\_Effort.csv.  Importing attributes\_YBFMP\_Trap\_Effort.txt.  Detecting attributes of YBFMP\_Site\_locations\_latitude\_and\_longitude.csv.  Importing attributes\_YBFMP\_Site\_locations\_latitude\_and\_longitude.txt.  Done. |
|  |
| |  | | --- | | > | |

-Keep in mind if you make corrections to files and re-run the import\_templates, if the file already exists it will not re-write it. You must delete the specific text file and then re-run the import\_templates.

**Step 6: Script your workflow**

Open my\_workflow.R in RStudio. This is a blank script for you to build an assembly line workflow, which can be revisited or modified for future assembly line runs.

\*\*You will have already started an R workflow, hence at this point save yours in the appropriate working directory.

**Step 7: Abstract**

Open the file abstract.txt and write an abstract for your dataset. The abstract should cover what, why, when, where, and how for your dataset. Write your abstract in plain text.

Do not use special characters, symbols, formatting, or hyperlinks (URLs are acceptable). The reason for this is that the EML schema only allows characters that are apart of the unicode character set.

NOTE: You can create your abstract in Microsoft Word and then copy over to abstract.txt but first you will need to remove any non-unicode characters. To do this go to [this web service](http://utils.paranoiaworks.org/diacriticsremover/) and paste your abstract into the window. Click the button "Remove Diacritics" to remove these non-compliant characters, then copy the resultant text into abstract.txt. You will want to give your abstract one last look over after performing this operation to ensure no information has been lost.

**Step 8: Methods**

Open the file methods.txt and describe the methods for your dataset. Be specific, include instrument descriptions, or point to a protocol online. If this dataset is a synthesis of other datasets please specify dataset origins, preferably their DOI or URL plus general citation information.

Do not use special characters, symbols, formatting, or hyperlinks (URLs are acceptable). The reason for this is that the EML schema only allows characters that are apart of the unicode character set.

NOTE: You can create your methods in Microsoft Word and then copy over to methods.txt but first you will need to remove any non-unicode characters. To do this go to [this web service](http://utils.paranoiaworks.org/diacriticsremover/) and paste your methods into the window. Click the button "Remove Diacritics" to remove these non-compliant characters, then copy the resultant text into methods.txt. You will want to give your methods one last look over after performing this operation to ensure no information has been lost.

Rosie’s note: I ended up moving some tables from the “methods” section to their own data files (sample types look-up and gear specifications)

Format methods following the IEP metadata standards.

|  |  |  |
| --- | --- | --- |
| 1. Data collection methods | Metadata must include enough information on methods to make the data usable. Minimum methods information should be similar to the "methods" section of a scientific paper. This may include diagrams and pictures of sampling equipment. | REQUIRED |
| 1. Link to blank datasheet | Available upon request (see data contact information) | REQUIRED – 3yrs |
| 1. Instrument and equipment specifications, including qaqc methods and frequency | May be references to external SOPs instead of included in metadata | REQUIRED |
| 1. Analysis methods & SOPs | Any analyses done to produce the data set (such as CPUE calculations). This is not analyses done to produce later publications.  Link to SOPS: Specific SOPs used to generate the data. It is understood that not all programs will have this information available right away, but should be prepared to provide them within three years. | REQUIRED |
| 1. Project history | List of any changes in methods and sampling locations, with dates changes were implemented | REQUIRED – If App. |
| 1. QA/QC –   Methods:  Data: | Methods: Protocols for quality assurance during data collection  Data: Protocols for quality assurance during data entry and analysis | REQUIRED – 3yrs |
| 1. Contractor information | Chain of custody procedures and contact information for any outside labs used to produce the data. | REQUIRED – If App. |
| 1. External review process | Any other review of data done by entities other than the PI to help with quality assurance. We just need a description of the process, not the reviews themselves | REQUIRED – If App. |
| 1. Methods references | Citations for publications from which methods were drawn. | REQUIRED – If App. |

**Step 9: Additional information**

additional\_info.txt is a good place for text based information about your dataset that doesn't fall under the scope of the abstract or methods (e.g. a list of research articles or theses derived from this dataset). If you have this information and would like to share it then open additional\_info.txt in a text editor and add it. You can delete this file if you won't be using it, or you can keep it around in case you change your mind.

Do not use special characters, symbols, formatting, or hyperlinks (URLs are acceptable). The reason for this is that the EML schema only allows characters that are apart of the unicode character set.

NOTE: You can create your additional information in Microsoft Word and then copy over to additional\_info.txt but first you will need to remove any non-unicode characters. To do this go to [this web service](http://utils.paranoiaworks.org/diacriticsremover/) and paste your additional information into the window. Click the button "Remove Diacritics" to remove these non-compliant characters, then copy the resultant text into additional\_info.txt. You will want to give your additional information one last look over after performing this operation to ensure no information has been lost.

\* Brittany added a list of peer-reviewed literature that has already used this dataset that users can reference for additional details.

**Step 10: Keywords**

Open the tab delimited file keywords.txt in a spreadsheet editor and list the keywords that best describe your dataset. DO NOT edit this file in a text editor. [Consult the LTER controlled vocabulary](http://vocab.lternet.edu/vocab/vocab/index.php) for keywords. In addition to keywords describing the data, you may want to include keywords that describe your lab, station, and project (e.g. OBFS, LTREB, etc.).

Definitions for columns of this file:

* **keyword** A keyword describing your dataset.
* **keywordThesaurus** A keywordThesaurus (i.e. a controlled vocabulary like the resource listed above) corresponding to the keyword listed in the keyword column. If the keyword is not from a thesaurus or controlled vocabulary, leave corresponding entry in the keywordThesaurus column blank.

Include IEP key term “Interagency Ecological Program for the San Francisco Bay Delta Estuary” and “IEP”

You can check to see if the keywords are in the LTER Thesaurus by leaving the keywordThesaurus column blank and running the “validate\_keywords” function:

validate\_keywords(path = getwd(), cv = 'lter')

The function will automatically fill-in the “keywordThesaurus” column for any words in the LTER vocabulary

**Step 11: Personnel**

Open the tab delimited file personnel.txt in a spreadsheet editor and enter information about the personnel associated with this dataset.

Definitions for columns of this file:

* **givenName** First name of person.
* **middleInitial** Middle initial of person.
* **surName** Last name of person.
* **organizationName** Name of organization the person is associated with.
* **electronicMailAddress** Email address of person.
* **userId** ORCID of person (not required). A valid entry for userId is the 16 digit ORCID number separated by dashes (i.e. XXXX-XXXX-XXXX-XXXX). An ORCID is like a social security number for scientists and links your dataset with your ORCID. [Create one here](https://orcid.org/).
* **role** Role of person with respect to this dataset. Valid entries for role are:
  + **creator** Dataset creator (required; at least 1 creator must be listed for your dataset).
  + **PI** Principal investigator associated with this dataset (not required).
  + **contact** Dataset contact (required; at least 1 contact must be listed for your dataset). The contact may be a person or a position at an organization. We recommend listing the contact as a person rather than a position. To list a position as a contact (e.g. Data Manager), Enter the position name in the givenName column and leave middleInitial and surName blank.
  + Any other entries into the 'role' column are acceptable and will be defined under the associated party element of this dataset with whatever value is entered under role.
  + If a person serves more than one role, duplicate this person’s information in another row but with the additional role.
  + Similarly if a role is shared among many people, list the individuals with the shared role on separate lines.
* **projectTitle** Title of the project this dataset was created under (optional). Project titles are only listed on lines where the personnel role is PI. If an auxiliary project was involved in creating this dataset then add a new row below the row containing the primary project and list the project title and associated PI. Do this for each auxiliary project.
* **fundingAgency** Name of the entity funding the creation of this dataset (optional). Only include an entry in this column for rows where role PI.
* **fundingNumber** Number of the grant or award that supported creation of this dataset (optional). Only include an entry in this column for rows where role PI.

\*Creators are the only ones that will actually be listed as the cited authors. See the metadata word document for more notes on how to manipulate the table to get other entries imported (e.g. having IEP listed in the SurName). The order of personnel in the table will be the order listed in the citation.

**Step 12: Attributes**

An attributes\_datatablename.txt file has been created for each of your data tables. Edit each of these tab delimited files in a spreadsheet editor. DO NOT edit this file in a text editor. You will see this file has been partially populated with information detected by the import\_templates function. You will have to double check values listed in all the columns except attributeName.

Instructions for completing the attribute table are as follows:

* **attributeName** Enter attribute names (i.e. column names) as they appear in the data table and in the same order as listed in the data table.
* **attributeDefinition** Enter definitions for each attribute. Be specific, it can be lengthy.
* **class** Enter the attribute class. This is the type of value stored under the attribute. Valid options for class are:
  + **numeric** For numeric variables.
  + **categorical** For categorical variables.
  + **character** For variables containing text or symbols that are not categorical.
  + **Date** For date time variables.
  + If an attribute has class of numeric or Date, then all values of this attribute must be either numeric or date time. If any character strings are present in an otherwise numeric attribute, this attribute must be classified as character. Similarly if any values of a "Date" attribute do not match the date time format string (details below), then this attribute must be classified as character.
* **unit** If an attributes class is numeric, then you must provide units. If the attribute is numeric but does not have units, enter dimensionless. If the attribute class is categorical, character, or Date then leave the unit field blank. If the attribute is numeric and has units search the standard unit dictionary for the unit of interest and enter the unit name as it appears in the dictionary (unit names are case sensitive). Open the dictionary by running these lines of code in the RStudio console window:

# View and search the standard units dictionary

view\_unit\_dictionary()

* If you cannot find a unit in the dictionary, create one and add it to custom\_units.txt. Open this tab delimited file in a spreadsheet editor. DO NOT edit this file in a text editor. If you have no custom units to report you may delete this file, but may also keep it around if you think it may be of future use. Valid custom units must be convertible to SI Units (i.e. International System of Units). If it cannot be converted to SI then list it in the attribute defintion and enter "dimensionless" in the unit field. To create a custom unit define the:
  + **id** This is equivalent to the unit name.
  + **unitType** The type of unit being defined. Reference the dictionary for examples.
  + **parentSI** The SI equivalent of the id you have entered.
  + **multiplierToSI** This is the multiplier to convert from your custom unit to the SI unit equivalent.
  + **description** A description of the custom unit. Reference the dictionary for examples.
* **dateTimeFormatString** Enter the date time format string for each attribute of "Date" class. Remember, a class of "Date" specifies the attribute as a date, time, or datetime. Enter the format string in this field. If the attribute class is not "Date", leave this field blank. Below are rules for constructing format strings.
  + **year** Use Y to denote a year string (e.g. 2017 is represented as YYYY).
  + **month** Use M to denote a month string (e.g. 2017-05 is represented as YYYY-MM).
  + **day** Use D to denote a day string (e.g. 2017-05-09 is represented as YYYY-MM-DD).
  + **hour** Use h to denote a hour string (e.g. 2017-05-09 13 is represented as YYYY-MM-DD hh).
  + **minute** use m to denote a minute string (e.g. 2017-05-09 13:15 is represented as YYYY-MM-DD hh:mm).
  + **second** use s to denote a second string (e.g. 2017-05-09 13:15:00 is represented as YYYY-MM-DD hh:mm:ss).
  + **Time zone format strings:** use + or - along with a time string to denote time zone offsets (e.g. 2017-05-09 13:15:00+05:00 is represented as YYYY-MM-DD hh:mm:ss+hh:mm).
* **missingValueCode** If a code for 'no data' is used, specify it here (e.g. NA, -99999, etc.). Only one missingValueCode is allowed for a single attribute.
* **missingValueCodeExplanation** Define the missing value code here.

\*You cannot have empty cells in the data tables, I added NA’s to fill cells in R (see my\_workflow\_YBFMP.R code). Then you can define NAs, do not add missing value descriptions if there are no actual NAs in that column, you’ll get an error code.

\*Depending on the data and time string format chose, you may get a warning in the final production report on EDI, but you will still be able to make the eml in R and publish.

**Step 13: Close files**

Make sure all files of your dataset directory are closed. Some functions will error out if these files are open.

\*If you edit any text file or recall in templates, they will not re-write the ones already in the working directory and you will get a message in the console that the files already exist.

**Step 14: Categorical variables**

If your data tables contain any attributes with the categorical class, you will need to supply definitions for the categorical codes. Use the function define\_catvars to do this. define\_catvars searches through each attribute file looking for attributes with a categorical class. If found, the function extracts unique categorical codes for each attribute and writes them to a file for you to define.

# View documentation for this function

?define\_catvars

# Run this function for your dataset

define\_catvars(path = "/Users/csmith/Desktop/gleon\_chloride")

A tab delimited **catvars\_datatablename.txt** will be created for each of your data tables containing categorical variables. Open these in a spreadsheet editor and add definitions for each code.

**Step 15: Geographic coverage**

If your dataset contains more than one sampling point or area, then you will want to add this information to your metadata. Often a data user will search for data within a geographic area.

**Geographic points**

Run the function extract\_geocoverage to get the unique latitude, longitude, and site name combinations from your data and write to file. extract\_geocoverage requires specific inputs that may require altering the latitude and longitude format of your data. See documentation for details.

Arguments required by this function are:

1. **path** A path to the dataset working directory containing the data table with geographic information.
2. **data.file** Name of the input data table containing geographic coverage data.
3. **lat.col** Name of latitude column. Values of this column must be in decimal degrees. Latitudes south of the equator must be prefixed with a minus sign (i.e. dash, "-").
4. **lon.col** Name of longitude column. Values of this column must be in decimal degrees. Longitudes west of the prime meridian must be prefixed with a minus sign (i.e. dash, "-").
5. **site.col** Name of site column. This column lists site specific names to be associated with the geographic coordinates.

# View documentation for this function

?extract\_geocoverage

# Run this function for your dataset

extract\_geocoverage(path = "/Users/csmith/Desktop/gleon\_chloride",

data.file = "lake\_characteristics.csv",

lat.col = "lake\_latitude",

lon.col = "lake\_longitude",

site.col = "lake\_name")

This function outputs a tab delimited file named geographic\_coverage.txt to your dataset directory. You may edit this in a spreadsheet editor if you'd like, but if the data table this information has been extracted from is accurate, then there is no need for editing.

**Geographic areas**

The import\_templates function created a tab delimited table (bounding\_boxes.txt) in your working directory. Instructions for completing this file:

* **geographicDescription** Enter a brief description for each geographic area.
* **westBoundingCoordinate** Enter the western most geographic coordinate (in decimal degrees) of the area. Longitudes west of the prime meridian are prefixed with a minus sign (i.e. dash -).
* **eastBoundingCoordinate** Enter the eastern most geographic coordinate (in decimal degrees) of the area. Longitudes west of the prime meridian are prefixed with a minus sign (i.e. dash -).
* **northBoundingCoordinate** Enter the northern most geographic coordinate (in decimal degrees) of the area. Latitudes south of the equator are prefixed with a minus sign (i.e. dash -).
* **southBoundingCoordinate** Enter the northern most geographic coordinate (in decimal degrees) of the area. Latitudes south of the equator are prefixed with a minus sign (i.e. dash -).

\*\*The bounding box text file is for multiple regions. For Yolo Bypass we have one general region (YB) with numerous sampling sites. So we’ll use geographic coverage as a single entry and delete the bounding box file in the working directory. If you leave the bounding boxes if will give you a replication ERROR and not run the xml.

**Step 16: Make EML**

Now you are ready to synthesize your completed metadata templates into EML. This step is relatively simple, but requires several arguments:

1. **path** A path to the dataset working directory.
2. **dataset.title** A character string specifying the title for your dataset. Be descriptive (more than 5 words). We recommend the following format: Project name: Broad description: Time span (e.g. "GLEON: Long term lake chloride concentrations from North America and Europe: 1940-2016").
3. **data.files** A list of character strings specifying the names of the data files of your dataset.
4. **data.files.description** A list of character strings briefly describing the data files listed in the data.files argument and in the same order as listed in the data.files argument.
5. **data.files.quote.character** A list of character strings defining the quote characters used in your data files and in the same order as listed in the data.files argument. If the quote character is a quotation, then enter "\"". If the quote character is an apostrophe, then enter "\'". If there is no quote character then don't use this argument when running make\_eml.
6. **data.files.url** A character string specifying the URL of where your data tables are stored on a publicly accessible server (i.e. does not require user ID or password). The EDI data repository software, PASTA+, will use this to upload your data into the repository. If you will be manually uploading your data tables, then don't use this argument when running make\_eml.
7. **zip.dir** A list of character strings specifying the names of the zip directories of your dataset.
8. **zip.dir.description** A list of character strings briefly describing the zip directories listed in the zip.dir argument and in the same order as listed in the zip.dir argument.
9. **temporal.coverage** A list of character strings specifying the beginning and ending dates of your dataset. Use the format YYYY-MM-DD.
10. **geographic.coordinates** A list of character strings specifying the spatial bounding coordinates of your dataset in decimal degrees. The list must follow this order: "North", "East", "South", "West". Longitudes west of the prime meridian and latitudes south of the equator are prefixed with a minus sign (i.e. dash -). If you don't have an area, but rather a point. Repeat the latitude value for North and South, and repeat the longitude value for East and West.
11. **geographic.description** A character string describing the geographic coverage of your dataset.
12. **maintenance.description** A character string specifying whether data collection for this dataset is "ongoing" or "completed".
13. **user.id** A character string specifying your EDI data repository user ID. If you don't have one, contact EDI ([info@environmentaldatainitiative.org](mailto:info@environmentaldatainitiative.org)) to get one, or don't use this argument when running make\_eml.
14. **package.id** A character string specifying the package ID for your data package. If you don't have a package ID, then don't use this argument when running make\_eml. A non-input package ID defaults to "edi.101.1".

# View documentation for this function

?make\_eml

# Run this function

make\_eml(path = "/Users/csmith/Desktop/gleon\_chloride",

dataset.title = "GLEON: Long term lake chloride concentrations from North America and Europe: 1940-2016",

data.files = c("lake\_chloride\_concentrations",

"lake\_characteristics"),

data.files.description = c("Chloride concentration data.",

"Climate, road density, and impervious surface data."),

data.files.quote.character = c("\"", "\""),

temporal.coverage = c("1940-01-31", "2016-01-01"),

geographic.description = "North America and Europe",

geographic.coordinates = c("69.0", "28.53", "28.38", "-119.95"),

maintenance.description = "completed",

user.id = "csmith",

package.id = "edi.201.1")

Your EML file will be written to your data directory with the name packageID.xml. If your EML is valid you will receive the message: EML passed validation!. If validation fails, open the EML file in an XML editor and look for the invalid section. Often a minor tweak to the EML can be made manually to bring it into compliance with the EML schema.

**Step 17: Upload your data package to the EDI repository**

Your data and metadata form a package that may be uploaded to the [EDI data repository](https://portal.edirepository.org/nis/home.jsp). Contact EDI for login credentials ([info@environmentaldatainitiative.org](mailto:info@environmentaldatainitiative.org)).

**\*\* See the next “Staging & Production” Document for IEP datasets**