CheckR: A User-Friendly Collection Validation Script for Depositors and Users

Irene Yi, Claire Bowern

9/3/2022

This is an [R Markdown](http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

# Introduction

Welcome to CheckR: A User-Friendly Collection Validation Script for Depositors and Users. This tool is designed to function as a sanity check for depositors and users of digital language collections. We have included comments (noted by a “#” symbol in front of the associated comment line) to help you understand the function of each part of the code, as well as be able to edit the lines of code in very simple ways to choose what functions’ outputs you do or do not want to see in a given run of the code.

## Setup

First, set the fileroot - that’s the folder that you want to analyze. This chunk also sets some R options. Your “fileroot” will be your file path to your collection folder. In this example, it is /Users/ireneyi/Downloads/toyfiles because the example collection I am using, a folder called “toyfiles”, is located in my computer’s downloads folder. You will replace the part within the quotes ” ” with your own file path. Note that if you are using a Windows computer, you’ll need to use something like c:\\users\ireneyi\Downloads.

fileroot = c("/Users/xxx/path.to.files")  
  
local({r <- getOption("repos")  
 r["CRAN"] <- "http://cran.r-project.org"  
 options(repos=r)})  
knitr::opts\_knit$set(root.dir = fileroot)  
  
knitr::opts\_chunk$set(echo = FALSE)

**The first time you use this code, RUN this section of the code before doing anything else; this will ensure that you have all the packages you need to run the rest of the code.** To run it, change eval=FALSE to read eval=TRUE. You only need to install the packages once.

This chunk loads the packages you’ll need.

# Read in directory and list files

This block of code reads in your collection and gets a list of all of your files, as well as the directories/file paths (for purposes of reading files into code, or for locating where in subfolders files exist).

## [1] "" "Itelmen" "Itelmen/output"   
## [4] "Itelmen/Textgrids" "Itelmenout" "Ixil"   
## [7] "Ixil-Out" "IxilSeg" "jc1"   
## [10] "jc1aud" "jc1MFA" "jc1MFA2"   
## [13] "jc1MFA2out" "jc1test" "Langi"   
## [16] "LangiOutput" "Orig" "Yami"   
## [19] "Yami-orig" "Yami-origOUT" "Yami-output"   
## [22] "Yami-retry" "YamiOutput-old"

## Counting files

This next block of code counts the total number of files within your collection, lists every single type of file extension present in your collection, and counts the number of files that have each type of file extension.

## [1] "You have 1562 total files in your entire collection."

## [1] "Here is a table with every type of file extension that exists in your collection, along with how many files have each extensions (for example, if it says '226' under 'pfs', it means you have 226 files with the .pfs extension):"

##   
## -annoteTextGrid 001 csv dict eaf   
## 1 1 1 1 11   
## g2p lab mp3 par pdf   
## 1 384 13 1 8   
## pfsx phon praat R TextGrid   
## 9 1 1 1 659   
## txt wav xml yaml   
## 33 430 5 1

# Elan files

This next part of the script will count (and optionally list out) all the .eaf (Elan) files in your collection.

## [1] "You have 11 total .eaf files."

## [1] "/Users/xxx/path.to.files/Ixil/IXIL-CJL-CER-PMM-2016-11-02-0051.eaf"  
## [2] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0504.eaf"  
## [3] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0505.eaf"  
## [4] "/Users/xxx/path.to.files/jc1/JC1-KA01-Annotation.eaf"   
## [5] "/Users/xxx/path.to.files/jc1/JC1-KA02-Annotation.eaf"   
## [6] "/Users/xxx/path.to.files/jc1/JC1-KA03-Annotation.eaf"   
## [7] "/Users/xxx/path.to.files/jc1/JC1-KA04-Annotation.eaf"   
## [8] "/Users/xxx/path.to.files/jc1/JC1-KA05-Annotation.eaf"   
## [9] "/Users/xxx/path.to.files/jc1/JC1-KA06-Annotation.eaf"   
## [10] "/Users/xxx/path.to.files/jc1/JC1-KA07-Annotation.eaf"   
## [11] "/Users/xxx/path.to.files/jc1/JC1-KA08-Annotation.eaf"

## Missing audio files

The next block of code checks to see how many .eaf files are missing corresponding audio files (in the form of .wav or .mp4).

## [1] "You have 8 .eaf files in your collection that are missing corresponding audio files."

## [1] "/Users/xxx/path.to.files/jc1/JC1-KA01-Annotation.eaf"  
## [2] "/Users/xxx/path.to.files/jc1/JC1-KA02-Annotation.eaf"  
## [3] "/Users/xxx/path.to.files/jc1/JC1-KA03-Annotation.eaf"  
## [4] "/Users/xxx/path.to.files/jc1/JC1-KA04-Annotation.eaf"  
## [5] "/Users/xxx/path.to.files/jc1/JC1-KA05-Annotation.eaf"  
## [6] "/Users/xxx/path.to.files/jc1/JC1-KA06-Annotation.eaf"  
## [7] "/Users/xxx/path.to.files/jc1/JC1-KA07-Annotation.eaf"  
## [8] "/Users/xxx/path.to.files/jc1/JC1-KA08-Annotation.eaf"

## [1] "You have 3 .eaf files in your collection that have corresponding audio files."

## [1] "/Users/xxx/path.to.files/Ixil/IXIL-CJL-CER-PMM-2016-11-02-0051.eaf"  
## [2] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0504.eaf"  
## [3] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0505.eaf"

## Settings files

The next block of code checks to see how many .eaf files are missing corresponding settings files (either in .pfs or .pfsx file formats). Note that these files are mostly cosmetic, but they di include audio/transcript offset information. If that’s not archived, the link between transcripts and any offset is lost (meaning the transcripts no longer align with the audio).

## [1] "You have 2 .eaf files in your collection that are missing corresponding settings files."

## [1] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0505.eaf"  
## [2] "/Users/xxx/path.to.files/jc1/JC1-KA02-Annotation.eaf"

## [1] "You have 9 .eaf files in your collection that have corresponding settings files."

## [1] "/Users/xxx/path.to.files/Ixil/IXIL-CJL-CER-PMM-2016-11-02-0051.eaf"  
## [2] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0504.eaf"  
## [3] "/Users/xxx/path.to.files/jc1/JC1-KA01-Annotation.eaf"   
## [4] "/Users/xxx/path.to.files/jc1/JC1-KA03-Annotation.eaf"   
## [5] "/Users/xxx/path.to.files/jc1/JC1-KA04-Annotation.eaf"   
## [6] "/Users/xxx/path.to.files/jc1/JC1-KA05-Annotation.eaf"   
## [7] "/Users/xxx/path.to.files/jc1/JC1-KA06-Annotation.eaf"   
## [8] "/Users/xxx/path.to.files/jc1/JC1-KA07-Annotation.eaf"   
## [9] "/Users/xxx/path.to.files/jc1/JC1-KA08-Annotation.eaf"

## Missing transcripts

The next block of code checks to see how many .eaf files are missing corresponding transcript (.txt) files.

## [1] "You have 11 .eaf files in your collection that are missing corresponding transcript (.txt) files."

## [1] "/Users/xxx/path.to.files/Ixil/IXIL-CJL-CER-PMM-2016-11-02-0051.eaf"  
## [2] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0504.eaf"  
## [3] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0505.eaf"  
## [4] "/Users/xxx/path.to.files/jc1/JC1-KA01-Annotation.eaf"   
## [5] "/Users/xxx/path.to.files/jc1/JC1-KA02-Annotation.eaf"   
## [6] "/Users/xxx/path.to.files/jc1/JC1-KA03-Annotation.eaf"   
## [7] "/Users/xxx/path.to.files/jc1/JC1-KA04-Annotation.eaf"   
## [8] "/Users/xxx/path.to.files/jc1/JC1-KA05-Annotation.eaf"   
## [9] "/Users/xxx/path.to.files/jc1/JC1-KA06-Annotation.eaf"   
## [10] "/Users/xxx/path.to.files/jc1/JC1-KA07-Annotation.eaf"   
## [11] "/Users/xxx/path.to.files/jc1/JC1-KA08-Annotation.eaf"

## [1] "You have 0 .eaf files in your collection that have corresponding transcript (.txt) files."

## NULL

# Audio files

This next part of the script will count (and optionally list out) all the audio files in your collection.

## [1] "You have 430 total audio files."

## [1] "/Users/xxx/path.to.files/Itelmen/1\_Uspenskaia.wav"   
## [2] "/Users/xxx/path.to.files/Itelmen/2\_Zaporotskaia.wav"   
## [3] "/Users/xxx/path.to.files/Itelmen/3\_Zaporotski.wav"   
## [4] "/Users/xxx/path.to.files/Itelmen/4\_Ivashova.wav"   
## [5] "/Users/xxx/path.to.files/Itelmen/4\_Pravdoshina\_NEU.wav"   
## [6] "/Users/xxx/path.to.files/Itelmen/5\_Khan.wav"   
## [7] "/Users/xxx/path.to.files/Itelmen/6\_Gutorova.wav"   
## [8] "/Users/xxx/path.to.files/Ixil/IXIL-CJL-CER-PMM-2016-11-02-0051.wav"   
## [9] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0504.wav"   
## [10] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0505.wav"   
…

"   
## [426] "/Users/xxx/path.to.files/test.wav"   
## [427] "/Users/xxx/path.to.files/Yami-orig/25-65\_FishingAtNight.wav"   
## [428] "/Users/xxx/path.to.files/Yami-retry/25-65\_FishingAtNight.wav"   
## [429] "/Users/xxx/path.to.files/Yami/25-65\_FishingAtNightA.wav"   
## [430] "/Users/xxx/path.to.files/Yami/25-65\_FishingAtNightB.wav"

## Missing audio files

The next block of code checks to see how many audio files are missing .eaf/ELAN files (in .eaf format).

## [1] "You have 427 audio files in your collection that are missing corresponding ELAN (.eaf) files."

## [1] "/Users/xxx/path.to.files/Itelmen/1\_Uspenskaia.wav"   
## [2] "/Users/xxx/path.to.files/Itelmen/2\_Zaporotskaia.wav"   
## [3] "/Users/xxx/path.to.files/Itelmen/3\_Zaporotski.wav"   
## [4] "/Users/xxx/path.to.files/Itelmen/4\_Ivashova.wav"   
## [5] "/Users/xxx/path.to.files/Itelmen/4\_Pravdoshina\_NEU.wav"   
## [6] "/Users/xxx/path.to.files/Itelmen/5\_Khan.wav"   
## [7] "/Users/xxx/path.to.files/Itelmen/6\_Gutorova.wav"   
## [8] "/Users/xxx/path.to.files/IxilSeg/IXIL-CJL-CER-PMM-2016-11-02-0051\_aetzej\_u\_tumen\_aetzej\_umeyo\_aetzej\_u\_tzichin\_puaq\_478.wav"   
…

"   
## [425] "/Users/xxx/path.to.files/Yami-retry/25-65\_FishingAtNight.wav"   
## [426] "/Users/xxx/path.to.files/Yami/25-65\_FishingAtNightA.wav"   
## [427] "/Users/xxx/path.to.files/Yami/25-65\_FishingAtNightB.wav"

## [1] "You have 3 audio files in your collection that have corresponding ELAN (.eaf) files."

## [1] "/Users/xxx/path.to.files/Ixil/IXIL-CJL-CER-PMM-2016-11-02-0051.wav"  
## [2] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0504.wav"  
## [3] "/Users/xxx/path.to.files/Ixil/IXIL-CTZ-DES-GLT-2016-06-21-0505.wav"

# Export results to textfiles

This section takes all the results and writes them to text files. The files.txt file gives a full list of files. The missing\_export.txt file summarizes the eaf files that have missing audio (and vice versa), while matched\_export.txt is a list of the files that are not missing corresponding files. The filetypes.txt file provides a list of the filetypes in the collection (and how many of each there are).

# Further Elan file statistics

##Load Elan functions

For this section, CheckR borrows functions written by Dale Barr (<https://github.com/dalejbarr>), which have been compiled into a public Github repository that can be found here: https://github.com/dalejbarr/elan. An accompanying piece of R code, titled “functions.R”, can be downloaded from the Github repo and should be run before running the following chunks of code. Barr’s code allows for further analysis of ELAN tiers and data, but CheckR is merely using surface functions of Barr’s code to extract basic information about ELAN tiers for specific EAF files, as CheckR is used as a sanity check for users and depositors of collections.

For further uses of Barr’s code functions, please follow the documentation from the linked Github repository.

## Parse structure of Elan files

# For bugs, feature requests, etc

Please submit issues (bug reports or requests for new features) at <https://github.com/chirila/archive-audit/issues/>