(*Final HTML map with popups*)

Hi Everyone, and thank you for checking out The Map Guy’s channel. My name is Lee and I’m going to show you today how you can quickly create a map of your cell phone photos *similar to this* in three easy and completely free steps. So, have you ever gone on a trip (vacation, work trip) where you took several photos with your cell phone at different locations and later you got the brilliant idea of seeing these photos mapped (for personal reasons or to share with others)? Well, as you probably should know, today’s cell phones have a built-in GPS and when the phone’s location feature is enabled, geospatial metadata are stored with each digital photo that you take with your phone. This geospatial spatial information (lat/long) is stored as part of the photo’s EXIF metadata. I’m going to show you how you can quickly and easily extract that geospatial EXIF data to view your photos in a map using R (an open source computing software). If you’re not familiar with R, don’t worry! I’m going to make this tutorial very simple so that even the newest R users can create a map. There are three very basic steps:

1. Setup (download R and save your photos to your computer)

2. Extract the EXIF data from the photos

3. Map the photos using the EXIF information

There is a little bit of programming involved here, but it’s very simple and I’ll provide links to the code in the Description below where you can essentially copy & paste to produce your results. (see, I told you this was simple). Assuming you already have R and R Studio installed on your computer, you can probably follow these instructions to produce a map in under 20 minutes. I’m going to try to keep this video under 8 minutes though, so feel free to pause along the way.

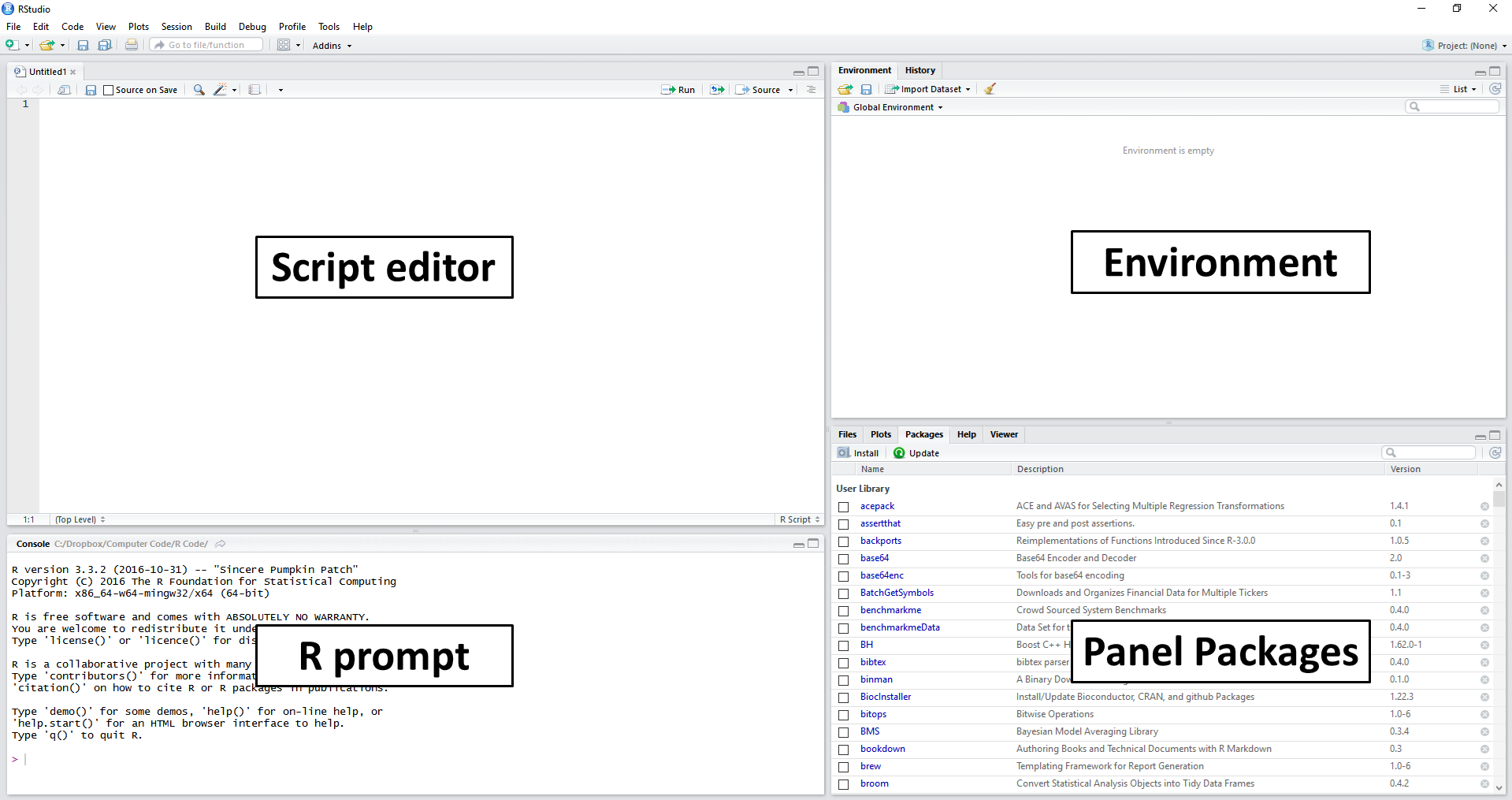
**Step 1: Setup (this is the longest step if you are starting from scratch….no R)**

(*Show R download page*)

Now I’m going to assume that you’ve probably already heard about R. It’s free. It’s open source. It’s powerful. I like it. We need it for this example. We’re also using R Studio. It is a great editor for R code and makes R much easier to use. If you don’t already have R and R Studio, download links are below and here is an example download screen. Download the latest version and you should be set. And, yes, both Mac and PC users can do this!

(*Show R Studio console*)

Once installed, your R Studio Screen should look like this. You have the script editor where we’ll be working mostly, there’s the console/terminal that will show you whether/how the code is executed. Then you have the Environments and the window for files, plots, packages, and the viewer. This is where we’ll see map results. This is not an R tutorial, so I advise you to check out some introductory videos if you’d like to learn more about R. I’ll put some intro tutorials in the description below. Nevertheless, I assure you that even the newest R user can follow along and use these instructions to make a map of their cell phone videos. Now that you have R Studio installed and up and running. Let’s look at your cell phone photos.



(*Show folder with cell phone photos*)

The next part of the setup is to transfer the photos you want to map to your computer. There are many ways to do this. I just used AirDrop to transfer select photos from my iPhone to my MacBook. Whatever way you do it, just make sure all photos are in the same folder that you call later find in R. In this example, I have several photos that I took with my cell phone on a recent vacation to Florida. We’ve called this folder “Vacation” and you can see there are 9 photos present in the folder (with original file names). Once you’ve transferred all photos to the folder, you’re now ready to go back to R and move on to the next steps. It get’s easier from here!

**Step 2: Use R to Extract the EXIF data from the photos**

(*Show R Studio console*)

So, now we’re back in the R Studio console. Let me quickly tell you about how we’re going to do this. Our second step in this process is to extract the exif data from all of our photos; then we need to map them. This is all done in R. R uses open source libraries to get things done. There are many many libraries out there, but they don’t come pre-installed with R. You have to install them and then call the libraries. The ones we are going to use today are:

* exifr – read and extract the exif data from the photos
* leaflet – this is the mapping library
* mapview – adds some visualization functionality to leaflet (popups)
* htmlwidgets – allows us to save the resulting html map to our computer

Without going into a lot of detail, we first need to install these four packages. We do that by typing up here in the script editor:

**install.packages(‘exifr’)**

**install.packages(‘leaflet’)**

**install.packages(‘mapview’)**

**install.packages(‘htmlwidgets’)**

run them by highlighting them, and then click run up here…

Alternatively, you can install these packages by going over here to the panel packages frame and clicking on

Packages – Install – and then typing in the names of these packages in the window. Then click Install.

It takes about a minute or so to install all the packages. I already have the packages installed on my computer so we’ll speed things up to the next step.

Once you have the packages installed, next we need to call those libraries so we can used them. We do this by using the library statement:

**library(exifr)**

**library(leaflet)**

**library(mapview)**

**library(htmlwidgets)**

Just to confirm, you shouldn’t see any error messages down here in the console or terminal as you’re loading the libraries.

Ok, now that we have our libraries loaded, let’s move on to working with our images.

To do this, we need to set the working directory to the folder containing our images. We do this with the setwd statement:

**setwd('/Users/OikoEco/Vacation')** – remember to use quotes, single or double, use forward slashes, and click run

we can confirm the working directory using **getwd()**

This let’s us see that we are in fact in the correct directory

Now we get the names of all the JPG photos in that directory using the list.files command

It creates a list