MapShaper tutorial

1. Download shapefiles. These can usually be found by Googling [your place] + shapefile  
     
   EXAMPLE: From here <https://data-wi-dnr.opendata.arcgis.com/datasets/county-boundaries-24k/explore> download the Shapefile (file previously generated on \_\_\_\_\_\_)
2. Navigate to mapshaper.org.
3. Upload shapefiles – either the entire Zip folder, or select and drag all the files.  
     
   A map should appear on screen.
4. Upload a data file that has the same FIPS to be joined. You can click and drag, or use Add a File from the central dropdown.  
     
   EXAMPLE: We upload the **2021\_04\_06-spring\_election\_agg** file from data/cleaned.
5. Open the console in the top right of mapshaper. For this next portion, you will need to navigate between the shapefile and your uploaded data. Switch between the two by selecting from the dropdown at the top.
6. Type **-info** in the console to view field names for both the shapefile and your data file. In order to join your data, you will need a common field called a key.

EXAMPLE: the County\_Boundaries\_24K example has the field **COUNTY\_FIP** field. It is a string with varying lengths. The 2021\_04\_06-spring\_election\_agg file has a **FIPS** field with a 5-digit code. So we need the County\_Boundaries field to end up as a 5-digit FIPS code, and can generate it via console commands.

1. Some light transformation can be done in console to create a join key.   
     
    **each county\_fip=String(COUNTY\_FIP).padStart(3,"0")**The **each** command creates a new field, **county\_fip**, from the original COUNTY\_FIP column. The **padStart** command makes each string length 3 by adding the character in quotes – in this case **0** – to the front if it is not length 3.  
     
   Next we run:  
     
    **each GEOID="55"+String(county\_fip)**  
     
   We create another new field from our previously generated county\_fip field. This time we add “55” to every entry. Now we have a 5-digit string to join on.

We’re not quite ready to join. The field in our shapefile is a string; in our data file, it’s a number. So we navigate over to the datafile, open the console, and run:  
  
 **each fips=String(FIPS)**

1. Time to join! Navigate back to the shapefile. In console, type:  
     
    **join [datafile] keys=[Key1],[Key2]**(without the brackets, substituting your datafile and keys)  
     
   EXAMPLE:  
     
    **join 2021\_04\_06-spring\_election\_agg keys=GEOID,fips**Now if you run **-info**, you should see the shape attributes as well as the data attributes!
2. Chloropleth maps can be generated using the **classify** command. **classify** starts by specifying a field to map. It can then intake several additional commands.  
    -**colors** can specify a color scheme (such as reds, blues, greens, etc)  
    -**classes** can specify how many different groupings you want to see  
    -**breaks** can be used to specify how to split the classes (eg, between 10-20 gets one color, between 20-30 the next, etc). Always input **1 fewer break** than classes (since classes get created before & above the first and last break)  
    **-key-style** can be used to determine what kind of key you want. (**simple** or **dataviz**  
     
   Overall it would look something like this:  
     
    **classify** [**your\_field] colors=[your\_colors] classes=[# classes] breaks = [break intervals]**

EXAMPLE:  
  
 **classify absentee\_%\_of\_voters colors=greens classes=5 key-style=simple**  
will allow the breaks to be auto-generated, or you might manually specify with:  
  
 **classify absentee\_%\_of\_voters colors=greens classes=5 breaks=0.15,0.20,0.25,0.30  
 key-style=simple**

1. The county lines have gone! These can be added back in. Simply re-upload the entire shapefiles again. The chloropleth map will vanish – but it isn’t gone. Go to the dropdown and click the eye icon next to the original dataset. It should now overlay!