JSONMapDraw | A_Data | B_POI | C_Way | D_Polygon | E_InfoColors | GeoHelper

```
/*
This script allows you to take in
and draw basic GIS data from a JSON of GIS information

Nina Lutz
CUSW IAP 2019
*/

//First make a blank map
MercatorMap map;
PImage background;

void setup(){
  size(1000, 650);

  //Intiailize your data structures early in setup
  map = new MercatorMap(width, height, 42.3636, 42.3557, -71.1034, -71.0869, 0);
  polygons = new ArrayList<Polygon>();
  ways = new ArrayList<Way>();
  pois = new ArrayList<POI>();

  //Load in and parse your data in setup -- don't want to do this every frame!
  loadData();
  parseData();
}

void draw(){
  //background image from OSM
  image(background, 0, 0);
  fill(0, 120);
  rect(0, 0, width, height);

```

JSONMapDraw | A_Data | B_POI | C_Way | D_Polygon | E_InfoColors | GeoHelper

```
  ways = new ArrayList<Way>();
  pois = new ArrayList<POI>();

  //Load in and parse your data in setup -- don't want to do this every frame!
  loadData();
  parseData();
}

void draw(){
  //background image from OSM
  image(background, 0, 0);
  fill(0, 120);
  rect(0, 0, width, height);

  //Draw all the ways (roads, sidewalks, etc)
  for(int i = 0; i<ways.size(); i++){
    ways.get(i).draw();
  }

  //Draw all polygons
  for(int i = 0; i<polygons.size(); i++){
    polygons.get(i).draw();
  }

  //Draw all POIs
  for(int i = 0; i<pois.size(); i++){
    pois.get(i).draw();
  }

  drawInfo();

}
```

JSONMapDraw | A_Data | B_POI | C_Way | D_Polygon | E_InfoColors | GeoHelper

```
JSONObject example;
JSONArray features;
JSONObject wholeArea;
//Look at https://processing.org/reference/JSONObject.html for more info

void loadData(){
  //Load and resize background image
  background = loadImage("data/background.png");
  background.resize(width, height);

  //Small example area
  //example = loadJSONObject("data/example.json");
  //features = example.getJSONArray("features");

  //Whole Area
  wholeArea = loadJSONObject("data/wholeArea.json");
  features = wholeArea.getJSONArray("features");

  println("There are : ", features.size(), " features.");
}
```

JSONMapDraw | A_Data | B_POI | C_Way | D_Polygon | E_InfoColors | GeoHelper

```
void parseData(){
  //First do the general object
  JSONObject feature = features.getJSONObject(0);

  //Sort 3 types into our respective classes to draw
  for(int i = 0; i< features.size(); i++){
    //Idenitfy 3 main things; the properties, geometry, and type
    String type = features.getJSONObject(i).getJSONObject("geometry").getString("type");
    JSONObject geometry = features.getJSONObject(i).getJSONObject("geometry");
    JSONObject properties =  features.getJSONObject(i).getJSONObject("properties");
    String amenity = properties.getJSONObject("tags").getString("amenity");
    String dataAmenity = properties.getJSONObject("tags").getString("amenity");
    if(dataAmenity != null) amenity = dataAmenity;
    else amenity = "";
    //Make POIs if it's a point
    if(type.equals("Point")){
      //create new POI
      float lat = geometry.getJSONArray("coordinates").getFloat(1);
      float lon = geometry.getJSONArray("coordinates").getFloat(0);
      POI poi = new POI(lat, lon);
      poi.type = amenity;
      if(amenity.equals("atm")) poi.ATM = true;
      pois.add(poi);
    }

    //Polygons if polygon
    if(type.equals("Polygon")){
      ArrayList<PVector> coords = new ArrayList<PVector>();
      //get the coordinates and iterate through them
      JSONArray coordinates = geometry.getJSONArray("coordinates").getJSONArray(0);
      for(int j = 0; j<coordinates.size(); j++){
        float lat = coordinates.getJSONArray(j).getFloat(1);
        float lon = coordinates.getJSONArray(j).getFloat(0);
        //Make a PVector and add it
        PVector coordinate = new PVector(lat, lon);
        coords.add(coordinate);
      }
      //Create the Polygon with the coordinate PVectors
      Polygon poly = new Polygon(coords);
      polygons.add(poly);
    }

```

JSONMapDraw | A_Data | B_POI | C_Way | D_Polygon | E_InfoColors | GeoHelper

```
      JSONArray coordinates = geometry.getJSONArray("coordinates").getJSONArray(0);
      for(int j = 0; j<coordinates.size(); j++){
        float lat = coordinates.getJSONArray(j).getFloat(1);
        float lon = coordinates.getJSONArray(j).getFloat(0);
        //Make a PVector and add it
        PVector coordinate = new PVector(lat, lon);
        coords.add(coordinate);
      }
      //Create the Polygon with the coordinate PVectors
      Polygon poly = new Polygon(coords);
      polygons.add(poly);
    }

    //Way if a LineString
    if(type.equals("LineString")){
      ArrayList<PVector> coords = new ArrayList<PVector>();
      //get the coordinates and iterate through them
      JSONArray coordinates = geometry.getJSONArray("coordinates");
      for(int j = 0; j<coordinates.size(); j++){
        float lat = coordinates.getJSONArray(j).getFloat(1);
        float lon = coordinates.getJSONArray(j).getFloat(0);
        //Make a PVector and add it
        PVector coordinate = new PVector(lat, lon);
        coords.add(coordinate);
      }
      //Create the Way with the coordinate PVectors
      Way way = new Way(coords);
      ways.add(way);
    }

  }
}
```

JSONMapDraw | A_Data | B_POI | C_Way | D_Polygon | E_InfoColors | GeoHelper

```
ArrayList<POI> pois;

class POI {
  //What is the coordinate of the POI in lat, lon
  PVector coord;

  //Lat, lon values
  float lat;
  float lon;

  //Is ATM?
  boolean ATM;

  //String to hold the type -- defaults to empty if there is none
  String type;

  POI(float _lat, float _lon) {
    lat = _lat;
    lon = _lon;
    coord = new PVector(lat, lon);
  }

  void draw() {
    PVector screenLocation = map.getScreenLocation(coord);
    fill(poi_fill);
    noStroke();
    if (ATM) fill(atm);
    ellipse(screenLocation.x, screenLocation.y, 6, 6);
  }
}
```

JSONMapDraw | A_Data | B_POI | C_Way | D_Polygon | E_InfoColors | GeoHelper

```
ArrayList<Way> ways;

class Way{
  //Coordinates and color variables
  ArrayList<PVector>coordinates;

  //Empty constructor
  Way(){}

  //Constructor of coordinates
  Way(ArrayList<PVector> coords){
    coordinates =  coords;
  }

  //Draw the road
  void draw(){
    strokeWeight(4);
    stroke(road_color);
    for(int i = 0; i<coordinates.size()-1; i++){
        //iterate through the coordinates and draw lines
        PVector screenStart = map.getScreenLocation(coordinates.get(i));
        PVector screenEnd = map.getScreenLocation(coordinates.get(i+1));
        line(screenStart.x, screenStart.y, screenEnd.x, screenEnd.y);
  }
  }
}
```

JSONMapDraw | A_Data | B_POI | C_Way | D_Polygon | E_InfoColors | GeoHelper

```
class Polygon{
  //Shape, coordinates, and color variables
  PShape p;
  ArrayList<PVector>coordinates;

  //Empty constructor
  Polygon(){
    coordinates = new ArrayList<PVector>();
  }

  //Constructor with coordinates
  Polygon(ArrayList<PVector> coords){
    coordinates = coords;
    makeShape();
  }

  //Making the shape to draw
  void makeShape(){
    p = createShape();
    p.beginShape();
    p.fill(polygon_fill);
    p.strokeWeight(.5);
    p.stroke(255);
    for(int i = 0; i<coordinates.size(); i++){
        PVector screenLocation = map.getScreenLocation(coordinates.get(i));
        p.vertex(screenLocation.x, screenLocation.y);
    }
    p.endShape();
  }

  //Drawing shape
  void draw(){
    shape(p, 0, 0);
  }
}
```

JSONMapDraw | A_Data | B_POI | C_Way | D_Polygon | E_InfoColors | GeoHelper ▾

```
//fill color
color poi_fill = color(255,99,71);
color atm = color(255, 255, 0);
color polygon_fill = color(32, 178, 170);
color road_color = color(100,149,237);

void drawInfo(){
  fill(0);
  rect(20, 20, 125, 90);
  textSize(16);
  fill(poi_fill);
  text("POIs", 25, 40);
  fill(atm);
  text("ATM", 25, 60);
  fill(road_color);
  text("Roads", 25, 80);
  fill(polygon_fill);
  text("Buildings", 25, 100);
}

```