* **DEMO PREP**
  + make sure OpenLayers plugin is uninstalled
  + remove toolbars and readd during demo
  + have back-up of OSM extract, SQLite database, and final GeoJSON
* **Install QGIS**
  + https://www.qgis.org/en/site/forusers/download.html
* **Tour of QGIS interface**
  + when you open QGIS, you'll see a set of standard toolbars and buttons for
    - connecting to data,
    - editing data,
    - and various navigation tools
    - we can also add other toolbars by right clicking on the gray space and selecting the different toolbars we want
* **Activate plugins**
  + the first thing we're going to do is add a plugin we’ll be using for this demo
  + from the *Plugins* menu, choose *Manage and Install Plugins*
    - we see a list of plugins already installed with QGIS
      * we're looking for the *OpenLayers* pluginwhich isn’t installed by default
        + to get it, we’ll click on *Get More*
        + and type the word *open* into the search box
        + when the OpenLayers Plugin appears, highlight it and click the option to *Install plugin*
        + and choose *Close* to close the plugins box
* **Add OSM basemap to map canvas**
  + the OpenLayers Plugin can now be accessed from the *Plugins* menu
  + choose the OpenLayers plugin
    - and you can see the various tiled basemap layers than can be used in your project
  + we'll go ahead and Add the *OpenStreetMap basemap layer*
    - a new layer loads in the *layers* panel
    - this is a pre-rendered tiled raster basemap of OpenStreetMap data
  + let’s zoom into our area of interest which is downtown Denver
    - use the zoom tool to draw a box and zoom in
* **Download OSM Data**
  + What we’re looking at here is a tiled version of OSM data rendered in QGIS as a basemap.
  + we want the actual data we see in the OSM basemap, so we'll use the OpenStreetMap tools from the Vector file menu
    - the projection of our map canvas is 3857 or Web Mercator because that is the projection of the tiled basemap
    - we want to download these data using lat/long so we’ll change the map canvas projection to WGS84
      * click on the globe icon in the bottom right hand corner of the map canvas and let’s change the projection to WGS84
    - next, we’ll download the raw OSM extract which is XML for this area
      * from the Vector menu select OpenStreetMap>Download Data option
    - we’ll select the following options in the dialogue
      * download the data using the extent of the Map Canvas
      * choose a name and place to download the .osm file locally we’ll name it denver.osm
* **Add extracted data to QGIS**
  + Now we want to bring our extract into QGIS so we can look at it and extract just what we need for our restaurant map
  + Click on the icon for “Add Vector Layer” and let’s navigate to our denver.osm file
    - from the “Select vector layers to add…” dialogue, we’ll select all of the available features
    - and then click “Ok”
    - to see things a little better, let’s remove the OpenStreetMap basemap
    - let’s also change the symbology for polygons to a gray and points to a yellow
    - next, we’ll switch the layer ordering a bit so we can see the points and lines on top of the polygons
* **Import raw OSM data into SQLite**
  + The next step is to read our raw .osm XML file into a SQLite database
    - SQLite is a lightweight database that is stored as one file on your computer
  + From the Vector menu select OpenStreetMap > Import Topology from XML
  + In the OpenStreetMap Import dialogue:
    - Navigate to our denver.osm extract
    - Give our output database a name
    - And keep the option checked to Create a connection after import
* **Create a point layer with the tags that we want**
  + By selecting the tags that we want from the data, we can create a custom layer for our map. We can use the layer as is, or we can export it to a different format.
  + Since we’re making a map of restaurant and pub locations in Denver, we’ll want to only select those features from our extract
    - Back in the Vector>OpenStreetMap tools let’s select “Export Topology to SpatialLite”
      * For the input, we’ll choose the database we created in the previous step denver.osm.db
      * We’ll choose Points (nodes) under export type
      * Give our output a layer name denver\_restaurants
      * and click “Load from DB”
      * a list of all the tags available in the points layer populate in the list
      * from the list, we’ll select
        + name, amenity, all of the address fields, and cuisine
        + this is information we’ll use in our popups
      * We’ll keep the option “Load into canvas when finished” checked so our layer will be added to our map canvas
* **Create final dataset for use in web map**
  + If we open the attribute table to our point layer denver\_restaurants, we can see that all of the tags that we selected are now attributes in our table
  + the attribute we’ll be using to select our data is “amenity”
    - if we sort on this field, you can see the different types of amenities that we have
  + let’s go ahead and symbolize this layer using the amenity attribute to see more clearly what we have
    - right click on the denver\_points layer
    - select Properties and click on the “Style” option
    - we’ll choose to symbolize the data based on categories
    - and choose amenity as our Column
    - click Classify and then OK
    - now we can see all of the types of amenities we have in our point data
  + in our final web map, we’re going to show restaurants, bars, cafes and pubs so we need to select those from the data
    - we’ll right click on our denver\_points layer and select Filter….
    - in the Query Builder window, we’ll build our query to select the data we want
      * `amenity" IN ('cafe','bar','restaurant','pub')
    - now our map canvas is updated with that data
  + the final step is to save the queried data into a format that I can easily share with the team
    - right click on the denver\_points layer and click Save As
    - we’ll choose GeoJSON for the format
    - Save as denver\_restaurants
    - and keep the default projection settings
  + we’ll add that GeoJSON to our map canvas to take one final look at what our dataset looks like
* Push data to GitHub repository