public void filterByKingdom(View view)

{

String kingdom = filterEditText.getText().toString();

kingdomFilterHelper(this, kingdom);

}

public void kingdomFilterHelper(Context context, String kingdom)

{

kingdomArrayList.clear();

Log.*i*("filter by kingdom", "kingdom");

final String query = "https://bison.usgs.gov/api/search.jsonp?params=kingdom%3A(%22" + kingdom + "%22)&start=0&count=10";

RequestQueue requestQueue = Volley.*newRequestQueue*(context);

// get search results, put in structure to intersect with current search results

JsonObjectRequest searchRequest = new JsonObjectRequest(Request.Method.*GET*, query, null, new Response.Listener<JSONObject>()

{

@Override

public void onResponse(JSONObject response)

{

String scientificName = null;

String commonName = null;

boolean match = false;

// closes the loading, please wait dialog

dialog.dismiss();

try

{

JSONArray occurrences = response.getJSONArray("data");

Log.*i*("occurrences:", Integer.*toString*(occurrences.length()));

for(int i = 0; i < occurrences.length(); i++)

{

scientificName = occurrences.getJSONObject(i).getString("name");

Log.*i*("sciName", scientificName);

if (!kingdomArrayList.contains(scientificName)) {

kingdomArrayList.add(scientificName);

}

}

Log.*i*("array", Integer.*toString*(kingdomArrayList.size()));

}

catch(Exception exception)

{

}

}

}, new Response.ErrorListener()

{

@Override

public void onErrorResponse(VolleyError error)

{

Log.*e*("Error: ", error.toString());

dialog.dismiss();

}

});

// change timeout length

searchRequest.setRetryPolicy(new RetryPolicy() {

@Override

public int getCurrentTimeout() {

return 100000;

}

@Override

public int getCurrentRetryCount() {

return 100000;

}

@Override

public void retry(VolleyError error) throws VolleyError {

}

});

requestQueue.add(searchRequest);

}