// COMMENT (CLASS AND METHOD, PURPOSE)

public abstract class Retriever {

public abstract List<DatabaseObject> getAll();

}

public abstract class SingleRetriever extends Retriever {

public ArrayList<DatabaseObject> getAll() {

JsonArray json;

Iterator<JsonElement> iterator;

ArrayList<DatabaseObject> objects = new ArrayList<DatabaseObject>();

JsonObject temp;

Class objClass = getDatabaseObjectClass();

Constructor constructor = null;

try {

constructor = objClass.getConstructor(JsonObject.class);

} catch (NoSuchMethodException ex) {

// ERROR getting constructor from the provided class

ex.printStackTrace();

}

json = RestUtil.*getAll*(getJSONDatabaseString());

if (json != null) {

iterator = json.iterator();

while (iterator.hasNext()) {

temp = iterator.next().getAsJsonObject();

try {

objects.add((DatabaseObject)constructor.newInstance(temp));

} catch (InstantiationException ex) {

// ERROR instantiating database object

ex.printStackTrace();

} catch (IllegalAccessException ex) {

// ERROR instantiating database object

ex.printStackTrace();

} catch (InvocationTargetException ex) {

// ERROR instantiating database object

ex.printStackTrace();

}

}

}

return objects;

}

protected abstract Class getDatabaseObjectClass();

protected abstract String getJSONDatabaseString();

}

public abstract class DatabaseObject {

private String id;

private String name;

private String image;

private String description;

private HashMap<String, JsonElement> fields;

public DatabaseObject(JsonObject obj) {

fields = new HashMap<String, JsonElement>();

load(obj);

}

*/\*\**

*\* Gets a field from this object as a String. If the field can't be represented as a String*

*\* or if it does not exist, returns null.*

*\*/*

public String getString(String fieldName) {

JsonElement toReturn = fields.get(fieldName);

if (toReturn != null && toReturn.isJsonPrimitive()) {

return toReturn.getAsString();

}

else {

return null;

}

}

public JsonElement get(String fieldName) {

return fields.get(fieldName);

}

public String getId() {

return id;

}

public String getName() {

return name;

}

public String getImage() {

return image;

}

public String getDescription() {

return description;

}

private void load(JsonObject obj) {

for (Map.Entry<String, JsonElement> entry : obj.entrySet()) {

fields.put(entry.getKey(), entry.getValue());

}

if (fields.containsKey("\_id")) {

id = fields.get("\_id").getAsString();

}

if (fields.containsKey("name")) {

name = fields.get("name").getAsString();

}

if (fields.containsKey("image")) {

JsonObject imageObject = fields.get("image").getAsJsonObject();

image = imageObject.get("url").getAsString();

}

if (fields.containsKey("description")) {

description = fields.get("description").getAsString();

}

}

}

*/\*\**

*\* Parent class for fragments that will display lists of DatabaseObjects*

*\* as cards, pulled from a Retriever object.*

*\* Created by Gavin on 1/12/2016.*

*\*/*

public class CruCardScreen extends CruFragment {

private Retriever retriever;

public CruCardScreen() {

super();

}

public void setRetriever(final Retriever retriever) {

this.retriever = retriever;

}

public Retriever getRetriever() {

return retriever;

}

}

public class MinistryRetriever extends SingleRetriever {

@Override

protected Class getDatabaseObjectClass() {

return Ministry.class;

}

@Override

protected String getJSONDatabaseString() {

return Util.*getString*(R.string.*rest\_ministry\_all*);

}

}

public class Campus extends DatabaseObject {

private String name;

private String postcode;

private String state;

private String suburb;

private String street1;

private String country;

private String websiteUrl;

public Campus(JsonObject obj) {

super(obj);

name = this.getString("name");

websiteUrl = this.getString("url");

}

public String getName() {

return name;

}

public String getWebsiteUrl() {

return websiteUrl;

}

}