Hosta.ai Interview-Task

1. Introduction

At Hosta we work a lot on object-object relationships within rooms. In our pipeline, each image we receive will be automatically processed and each object will be assigned a `unique_id` and a hierarchical relationships established. An `interior_wall` is considered a parent which can hold multiple child-nodes such as `doors`, `windows` and also furniture etc. All these relationships are captured on an image-by-image basis, and for each image a corresponding `json` file is generated.
Sometimes these relationships within the `json` however, are misaligned and need to updated using a `csv` file.

2. The Files

2.1 JSON Files

You will be working with the three Jsons contained in this folder:

```
3d3fde25-fc47-47ad-bda4-0b438196045b.json
763fdd40-9408-45bb-b532-3f90b5c7c5d1.json
b73070b3-7625-4975-872a-967b2297a458.json
```

As described above each of these jsons represent an image taken from the same room. Each of these Jsons has a field called `ops_3d` which contains all of the objects found in the picture. These objects have their own `unique_id` field, which allows for identification of the object.
Some of the objects also contain a field called `imageIds`, which contains the image-ids as they are used in the `EXP_ObjectID_HostID.csv` file.

2.1 CSV File

The `EXP_ObjectID_HostID.csv` file is used when relationships between objects in the `Jsons` need to be updated.
The *columns* named `Object_ID` and `Host_ID` in the `csv` contain ids which also describe these object-object hierarchies, however they are *not* directly matching the `unique_id`'s in the `json` files.

- The *column* `Object_ID` contains a unique id per object.
- The *column* `Host_ID` is the unique id of the parent object. For example, a *door* - object would have a wall as their `Host_ID`.
- The *columns* `ImageX_Object_ID` contain the `imageIds` linking them back to the `json`. One object can show up in different images therefore the object can have multiple ids. Cells which contain either a `0` or a `''` can be disregarded.
- *Rows* which do not have an `Object_ID` or `Host_ID` can be ignored.

3. Objective

The child-parent relationships in the *Json* files have been broken. The goal for you is to restore as many child-parent object relationships as possible by updating the *Jsons*, using the `csv` file. Using the `imageIds` field in the *Json* and the `ImageX_Object_ID` columns in the `csv` a relationship between objects in `csv` and `json` can be established.
Now it is upon you to find possible parents for some of the objects contained in the

Json files and update the child-object in the *Jsons* by adding a `parent_id` field with the corresponding parent id.

The fields you add to the Jsons will look like this. `"parent_id": "a1104a24-9052-11ec-9e6a-26d6033014c5"`

4. Deliverable

Find the right parent-object for some of the objects described in the *Jsons* using a `Python` Script. Add the parent's `unique_id` to the child by adding a field called `parent_id`. Save all three updated Jsons. All of the *python* code you used to achieve this.

5. Criteria

We value concisely written code, with reusability in mind. Part of this means that the code is well documented and could be picked up and understood by somebody else.