

Proposed resources schema change

[Browse code](#)


```
models_new.py: A rough draft of the new model schema.

resources_new.txt: Schema brainstorming I did in December 2012. Explains
some of the schema decisions I made in models_new.py, but is not
completely accurate, since I changed my mind on some details when
writing the models.

resources.txt: Notes on the old schema, how it works, and why it is bad.

inventory_requests.pdf: An e-mail thread with requests for the new
schema. I believe that the proposed schema can implement all requests.

And, of course, models.py has the old schema, for comparison.
```

 **jmdowd** authored 2 days ago

1 parent 5f70daf commit 4cdc12ab3c8508dc5f7b666114993274fb294463

 Showing **4 changed files** with **357 additions** and **0 deletions**.

Show Diff Stats

**BIN** ■■■■■ esp/esp/resources/inventory\_requests.pdf [View file @ 4cdc12a](#)

Binary file not shown

196




esp/esp/resources/models\_new.py

☒ show inline notes

View file @ 4cdc12a

```
... @@ -0,0 +1,196 @@
1 +@reversion
2 +class Resource(models.Model):
3 +    """
4 +    An instance of a physical resource, e.g. Chromebook #1, ADP2 Mac adapter, etc.
5 +    All resources exist globally for use by all programs.
6 +    """
7 +    is_active = models.BooleanField()
```

14

 **luac** repo collab a day ago  

I noticed there are a lot of fields that are the same. Suggest use [abstract inheritance](#) to eliminate duplication?




Also, what is `custom_attributes / admin_custom_attributes` for, and how are they different?

**jmoldow** repo collab 21 hours ago

I will definitely use abstract inheritance in the actual models.

(admin\_)?custom\_attributes is a JSON string of any data that you want to associate with a resource in a structured / in a way that can be parsed by Python and Javascript. This is in contrast to notes and description, which are unstructured comments on the resource.

custom\_attributes would be visible to teachers+admins, and admin\_custom\_attributes would be visible to admins only. For example, the storage location in Walker might be something that goes in admin\_custom\_attributes, while properties of the item (dimensions of a board, model/OS of a laptop, etc.) would go in custom\_attributes.

 **luac** repo collab 21 hours ago  

So what is `teacher_notes` / `teacher_custom_attributes` in `Location`?



**jmoldow** repo collab 21 hours ago  

In the case of Location and Area, the audience includes not only admins and teachers, but students as well. Whereas only admins and teachers deal with Resources.




So with Location and Area, notes and custom\_attributes would be viewable to anyone[\*]. teacher\_notes and teacher\_custom\_attributes would be viewable to teachers and admins, and admin\_notes and admin\_custom\_attributes would be viewable to admins only.

My convention was, no modifier means accessible to anyone would might be interacting with this object, and modifier means accessible to that user type or higher. So no modifier means students+ in one scenario, and teachers+ in another. This seems consistent and intuitive to me, but is obviously inconsistent from a different point of view. So I'd be amenable to changing all of the Resource notes / custom\_attributes to teacher\_notes / teacher\_custom\_attributes, if that is preferred.

[\*] Viewable in any place we choose to show it, that is. Since we don't release classrooms in advance, students wouldn't be able to see notes and custom\_attributes in most situations anyway. But we could include them in the printed schedules. We could design a room / building look-up that students and parents could access during a program.




 **luac** repo collab 20 hours ago  

Also, how would custom\_attributes be displayed? Is it basically just like Tags where /theoretically/ any keys and values are allowed, but there are a small set of them that mean something to some part of the website (but are not a core Resource functionality and so aren't fields on the model themselves)?

 **jmoldow** repo collab 19 hours ago  

(I replied to this already, but it seems it didn't keep in Github. If someone got an e-mail with my original comment, can you copy-paste it here? Otherwise, I'm just going to tl;dr my original comment.)

This is not a Tag. I'm hoping that custom\_attributes will just be used for metadata. So it would be displayed in some interfaces, but would have no effect on any parts of the site. This could possibly be used for program-specific overrides or site-specific features, but I hope not. It could also be used in custom shell scripts, for example, a script run by MIT to create a CSV of all inventory items and their Walker storage locations (the latter of which would be stored in admin\_custom\_attributes).




 **luac** repo collab 19 hours ago  

I see.

For display I was thinking that raw JSON is a little... raw, but I guess it could use a sort of "pretty description list" format where `{ foo: 1, bar_baz: "quux" }` becomes

```
Foo: 1
Bar baz: quux
```

or something.




 **luac** repo collab 19 hours ago  

or in html





```
<d1>
  <dt>Foo</dt> <dd>1</dd>
  <dt>Bar baz</dt> <dd>quux</dd>
</d1>
```

 **jmoldow** repo collab 19 hours ago  

We can definitely prettify it for display.

 **pricem** repo collab 13 hours ago  

I like the currently proposed methods of storing and displaying custom attributes. However, do we really need to have some attributes that only admins can see? I can imagine this being used for "non human readable" information used by the autoscheduler, but I'm not sure what's wrong with showing teachers information added by admins. The same goes for the notes and description. I would initially recommend having one text field (notes/description) and one JSON field (attributes).

		<div><div>jmoldow</div><div>repo collab</div><div>11 hours ago</div><div><div></div><div></div></div></div> <div>I thought that the <code>_admin</code> fields could be used for storing information that teachers don't need to see and would clutter the view (such as where we rented a particular item from / their contact information, or any autoscheduler configs), that would confuse the teachers (such as where we store the items, since the teachers might confuse that with where they should return the items), or that are too sensitive to be seen by teachers (perhaps a root password for a computer).</div> <div>I don't know if we <i>need</i> that, but that's how I thought of it.</div>
		<div><div>benjaminjkraft</div><div>repo collab</div><div>11 hours ago</div><div><div></div><div></div></div></div> <div>I doubt that we want to show teachers the json field, because I think anything that's json will probably confuse them, and they will screw up the formatting if they edit it. So maybe we can use that for admin-only things and have a notes/description that is public? (Room codes probably shouldn't go in the DB anyway if at all possible.)</div>
		<div><div>jmoldow</div><div>repo collab</div><div>10 hours ago</div><div><div></div><div></div></div></div> <div>I don't think that any of the fields need to be teacher editable. They just need to be displayable, and in the case of JSON, pretty-displayable.</div> <div>JSON might confuse admins too. Some admins might have admin-only comments that they prefer to write in an unstructured way. I'm not sure that forcing all admin-only comments to be JSON is the right approach either.</div>
		<div><div>benjaminjkraft</div><div>repo collab</div><div>10 hours ago</div><div><div></div><div></div></div></div> <div>If JSON is too hard, we could use YAML or the format that ConfigParser uses, although then that's another language for us to deal with.</div>
		<div>Add a line note</div>
8	+	<code>abstraction = models.ForeignKey(AbstractResource)</code>
9	+	<code>identifier = models.CharField()</code>
10	+	<code># unique (abstraction, identifier) when is_active</code>
11	+	<code>availability = models.ManyToManyField(Event)</code>
4		<div><div>benjaminjkraft</div><div>repo collab</div><div>20 hours ago</div><div><div></div><div></div></div></div> <div>Should these have through-models or does it not matter?</div>
		<div><div>jmoldow</div><div>repo collab</div><div>20 hours ago</div><div><div></div><div></div></div></div> <div>I originally said they should, but then I couldn't remember why, so I didn't do that. I think it was because I wanted to be able to reversion it, or because I might want to add additional properties. But reversioning can be taken care of by reversioning Resource, and I can't think of any additional properties I would want to add to the intermediate table. If we can think of something we want, or some other reason to do it, I'll make that change.</div>
		<div><div>benjaminjkraft</div><div>repo collab</div><div>20 hours ago</div><div><div></div><div></div></div></div> <div>It looks like resources.txt suggests the autoscheduler might want a lock field, but I'm not sure why; could that be it?</div>
		<div><div>jmoldow</div><div>repo collab</div><div>20 hours ago</div><div><div></div><div></div></div></div> <div>I think that was a typo. I think the autoscheduler lock is only on assignments.</div>
		<div>Add a line note</div>
12	+	<code>custom_attributes = models.JSONField()</code>
13	+	<code>admin_custom_attributes = models.JSONField()</code>
14	+	<code>description = models.TextField()</code>
2		<div><div>benjaminjkraft</div><div>repo collab</div><div>20 hours ago</div><div><div></div><div></div></div></div> <div>Is there a difference between description and (admin_)notes?</div>
		<div><div>jmoldow</div><div>repo collab</div><div>20 hours ago</div><div><div></div><div></div></div></div>

Maybe, maybe not. I envisioned that description would be a short, rarely (if ever) changing description of the resource. Whereas notes could be appended to as frequently as necessary, with any updates to a resource's status, condition, location, planned movement, caveats, etc. Basically, I thought that notes would be the type of stuff that we currently record in e-mail exchanges or spreadsheets.

It might be the case that description isn't necessary, and that notes is sufficient. Or that description should only exist on the Abstract object. Or maybe description is fine, and notes should actually be implemented using <https://docs.djangoproject.com/en/1.4/ref/contrib/comments/>.

Add a line note

15

+ notes = models.TextField()

16

+ admin\_notes = models.TextField()

17

+

18

+@reversion

19

+class AbstractResource(models.Model):

20

+ """

21

+ Represents an abstract version of a specific type of Resource, e.g. Thinkpad Linux computer, mini-display-port to

22

+ Resources of the same AbstractResource are identical.

23

+ All abstract resources exist globally for use by all programs.

24

+ """

25

+ is\_active = models.BooleanField()

26

+ resource\_type = models.ForeignKey(ResourceType)

27

+ name = models.CharField()

28

+ # unique (resource\_type, name) when is\_active

29

+ is\_reusable = models.BooleanField()

4

pricem

repo collab

13 hours ago

What does is\_reusable mean?

jmoldow

repo collab

11 hours ago

Can this resource be assigned more than once, or will its use in a class destroy it (such as a food item)? This defaults to True (can be reused), with the possibility of it being False (can't be reused, its use destroys it).

benjaminjkraft

repo collab

11 hours ago

I think non-reusable resources probably won't be stored in the website anyway, since we get them for a single class anyway, although it might be worth asking ESP Entropy what they think about that.

jmoldow

repo collab

5 hours ago

Entropy said they might use this feature.

Add a line note

30

+ is\_requestable = models.BooleanField()

31

+ custom\_attributes = models.JSONField()

32

+ admin\_custom\_attributes = models.JSONField()

33

+ description = models.TextField()

34

+ notes = models.TextField()

35

+ admin\_notes = models.TextField()

36

+

37

+@reversion

38

+class ResourceType(models.Model):

39

+ """

40

+ Represents a type of resource, e.g. Linux computer, Mac adapter, chalkboard, etc.

41

+ Distinct AbstractResources in the same ResourceType are similar and can potentially serve as substitutes, but are

42

+ All resource types exist globally for use by all programs.

43

+ """

44

+ is\_active = models.BooleanField()

45

+ family = models.ForeignKey(ResourceFamily, null=True)

46

+ name = models.CharField()

4 of 18

8/14/13 8:24 PM

47

+ # unique (resource\_type, name) when is\_active

48

+ is\_requestable = models.BooleanField()

49

+ # if not family, then is\_requestable

50

+ custom\_attributes = models.JSONField()

51

+ admin\_custom\_attributes = models.JSONField()

52

+ description = models.TextField()

53

+ notes = models.TextField()

54

+ admin\_notes = models.TextField()

55

+

56

+@reversion

57

+class ResourceFamily(models.Model):

8

luac

repo collab

19 hours ago

ResourceFamily is basically the same schema as ResourceType. They have the same fields and are interchangeable in every way except that ResourceType is a distinguished layer of the tree -- it isn't allowed to be its own parent, and can only be the parent of an AbstractResource (which are in my mind the real leaves of this tree).

Propose merging Resource{Family,Type} for simplicity?

pricem

repo collab

13 hours ago

I think this is too many models. Given that ResourceFamily is a tree, I think that's sufficient capability for structuring groups of resources. You could even have a field like 'is\_equivalent' to denote when resources within the family can be substituted for one another. To go beyond what Anthony said, I would ask for a justification of why we need more than one model for grouping resources (rather than two).

jmoldow

repo collab

11 hours ago

I think my original reasoning was that ResourceFamily wouldn't be a tree. So you would always go ResourceFamily -> ResourceType -> AbstractResource. Then, when I was actually writing the models, I made Family a tree, and didn't realize that this made ResourceType unnecessary. So yeah, we can merge them.

luac

repo collab

11 hours ago

Yeah, I actually initially didn't notice that ResourceFamily was a tree, and I was actually going to propose eliminating it and making ResourceType a tree.

benjaminjkraft

repo collab

10 hours ago

Ooh, I like having "is\_equivalent" so we can know when things are interchangeable rather than using a separate model for it.

jmoldow

repo collab

10 hours ago

"is\_equivalent" would be a BooleanField on a Resource{Type,Family}?

I think I like is\_substitutable or is\_interchangeable better, since is\_equivalent is sort of what Resources under an AbstractResource are.

benjaminjkraft

repo collab

9 hours ago

I think the idea was to remove AbstractResource, add a field is\_equivalent on ResourceFamily, and use ResourceFamily with is\_equivalent=False to be what ResourceFamily currently is, and ResourceFamily with is\_equivalent=True to be what AbstractResource is, so we only have a single type instead of two or three.

luac

repo collab

9 hours ago

The current proposed schema enforces that an AbstractResource can't have any "subcategories" so to speak.

If we have just one model unifying AbstractResource, ResourceType, and ResourceFamily, then it's possible that a resource type has descendants even though it has is\_equivalent (i.e. it's supposed to be a leaf). But this also doesn't absolutely need to be enforced through the schema.

Add a line note

58

+ """

59

+ Represents a family of resources, e.g. computers, A/V, boards, etc.


```
60 + Resources descended from the same ResourceFamily have something in common, but are not necessarily substitutes.
61 + A ResourceType may, but is not required to, be a member of a ResourceFamily.
62 + A ResourceFamily may, but is not required to, be a member of a parent ResourceFamily.
63 + In this way, ResourceFamilies + ResourceTypes form a forest, with ResourceTypes as the leaves and with an arbitra
64 + The trees will never be very tall, and probably not very wide, so this does not pose the problems that the DataTr
65 + All resource families exist globally for use by all programs.
66 + """
67 + is_active = models.BooleanField()
68 + parent = models.ForeignKey(ResourceFamily, null=True)
69 + name = models.CharField()
70 + # unique (resource_type, name) when is_active
71 + is_requestable = models.BooleanField()
72 + # if not family, then is_requestable
73 + custom_attributes = models.JSONField()
74 + admin_custom_attributes = models.JSONField()
75 + description = models.TextField()
76 + notes = models.TextField()
77 + admin_notes = models.TextField()
78 +
79 +@reversion
80 +class Location(models.Model):
81 + """
82 + A.k.a. a classroom.
83 + All locations exist globally for use by all programs.
84 + """
85 + is_active = models.BooleanField()
86 + area = models.ForeignKey(Area, null=True)
87 + name = models.CharField()
88 + # unique (area, name)
89 + display_name_override = models.CharField() # Some string template that includes %(location)s and, optionally, %(a
```


8

pricem

repo collab

13 hours ago







Is this the mechanism for handling things like meeting point instructions?

jmoldow

repo collab

11 hours ago






It certainly could be, though I admit that wasn't what I was thinking at the time.

I was thinking more that you could override the display of the name for special types of classrooms. So at MIT, the average building would have a display\_name of "%(area)s-%(location)s" or just "%(location)s", for example to get Building 2-147 or 2-147 (which we choose would depend on whether MIT sets the names of classrooms as "2-147" or "147"). The display\_name\_override could be used so that, instead of displaying "Building 24-619" (adhering to "%(area)s-%(location)s"), we could display something like "The kitchen of %(area)s-%(location)s".

My inspiration for this was that we name some rooms pretty funky things. So we can accomplish that with something like this, or by simply making the name of the location always be the display text, and not make the building+location be the display text.


I don't think this is the best way to handle meeting point instructions though, since we want a teacher to be able to see both the real location and the meeting point, but we want students to be able to see only the meeting point. So I'll add something else for that, and possibly remove this if it isn't useful.




benjaminjkraft

repo collab


11 hours ago





I would think meeting point instructions can go in (teacher\_)notes, unless we're getting rid of those anyway.


FWIW I don't think the room name should be assumed to be "%(area)s-%(location)s". This is not true in general at places that aren't MIT (where it's usually "%(area)s %(location)s") and it will be confusing to have the "Building " in every room name (or not to have it in Areas. Also, if we do that, the location names will be things like "147" instead of "2-147" which is entirely useless. I'd suggest that we have a location name intended to stand on its own (e.g. "2-147" or "ESG kitchen") and anything else can go wherever we put meeting point instructions.




benjaminjkraft















repo collab



10 hours ago





Oh, and I think we often want both student-facing meeting point instructions, and non-student-facing ones. For example, T-Club lounge at

	<div><div>MIT might have directions to wait in Johnson lobby for students, and directors for how to actually get there for teachers.</div><div><div>jmdow</div><div>repo collab</div><div>10 hours ago</div><div></div><div></div></div><div><div>@benjaminjkraft, with regards to your second-to-last comment: I currently have it so that "%(area)s-%(location)s", or the equivalent, would be set in Area, in the display_name field. Then that String template would apply to all Locations in that Area. I wasn't suggesting that all chapters would be forced to use "%(area)s-%(location)s". A good default probably could be just "%(location)s" or "%(area)s %(location)s".</div></div></div>
	<div><div><div>jmdow</div><div>repo collab</div><div>9 hours ago</div><div></div><div></div></div><div><div>With regards to meeting points, we need to be careful to distinguish between actual meeting points, versus instructions to get to a room.</div><div>Instructions to get to a room could go in notes, or in a special field for this purpose.</div><div>Meeting points (i.e., meet in Lobby 7, even though the class actually takes place in a lab) should go in the ResourceAssignment, since that is class-specific and not global. And in the case where, for some reason, we don't want the students to know the real location of the class, and only the meeting point, we can have a mechanism for overriding the Location name with the meeting point name.</div><div>Thanks for bringing this up, I forgot about this use case.</div></div></div>
	<div><div><div></div><div>benjaminjkraft</div><div>repo collab</div><div>9 hours ago</div><div></div><div></div></div><div><div>Also, if we do that, the location names will be things like "147" instead of "2-147" which is entirely useless. I'd suggest that we have a location name intended to stand on its own (e.g. "2-147" or "ESG kitchen") and anything else can go wherever we put meeting point instructions.</div><div>Ah, ok. I think this part still stands, though; if the default is just "%(location)s", which I think would make more sense, what's the point in having the override?</div></div></div>
	<div><div><div>jmdow</div><div>repo collab</div><div>9 hours ago</div><div></div><div></div></div><div><div>Because you could choose to set display_name="%(area)s-%(location)s" in the "Building 2" Area, and then have all Building 2 classrooms be named "147" instead of "2-147", which would display them as "Building 2-147".</div><div>I'm saying the default would be "%(location)s", but that it could be something else if you wanted it to be.</div><div>And if this is not useful and confusing, it can be removed.</div></div></div>
	<div><div>Add a line note</div></div>
90	+ capacity = models.IntegerField()
91	+ furnishings = property() # AbstractResources, ResourceTypes, and ResourceFamilies
2	<div><div><div></div><div>benjaminjkraft</div><div>repo collab</div><div>10 hours ago</div><div></div><div></div></div><div><div>Is this just the reverse ForeignKey?</div></div></div>
	<div><div><div>jmdow</div><div>repo collab</div><div>9 hours ago</div><div></div><div></div></div><div><div>No.</div><div>The reverse ForeignKey would be to Furnishing, not to AbstractResources, ResourceTypes, and ResourceFamilies.</div><div>And simply doing furnishing.values('resource') wouldn't work, because (if this is even supported) it will be a set of IDs of different models, which will be yucky to work with.</div><div>The furnishings property will be a method that collects all of the related Furnishings via the actual reverse ForeignKey, and puts all of the AbstractResources, ResourceTypes, and ResourceFamilies into a list that is returned.</div></div></div>
	<div><div>Add a line note</div></div>
92	+ availability = models.ManyToManyField(Event)
93	+ custom_attributes = models.JSONField()
94	+ teacher_custom_attributes = models.JSONField()
95	+ admin_custom_attributes = models.JSONField()

<div>96 + description = models.TextField() 97 + notes = models.TextField() 98 + teacher_notes = models.TextField() 99 + admin_notes = models.TextField() 100 + is_requestable = models.BooleanField() 101 + url = models.URLField() 102 + administrator_email = models.EmailField()</div>	<div><div>3</div><div><div><div>pricem repo collab13 hours ago</div><div>I like the idea of tracking who is "in control" of each classroom, like the departmental or university administrators; maybe this should be grouped into a simple Maintainer[?] model that Location is foreign keyed to?</div></div><div><div>j moldow repo collab11 hours ago</div><div>Sounds good to me.</div></div><div><div> benjaminjkraft repo collab11 hours ago</div><div>++; most rooms are in groups of at least a few with the same owner and reservation process.</div></div><div>Add a line note</div></div></div>
<div>103 + 104 +@reversion 105 +class Area(models.Model): 106 + """ 107 + A.k.a. building, or floor, or wing, etc. 108 + A set of Locations that are sufficiently close together for scheduling purposes, and form some distinct geographi 109 + All areas exist globally for use by all programs. 110 + """ 111 + is_active = models.BooleanField() 112 + name = models.CharField() 113 + display_name = models.CharField() # Some string template that includes %(location)s and, optionally, %(area)s. Ap 114 + adjacent_areas = models.ManyToManyField(symmetric=True) # Other sets of Locations that are also sufficiently close tog 115 + latitude = models.DecimalField() 116 + longitude = models.DecimalField() 117 + map_pixel_x = models.IntegerField() 118 + map_pixel_y = models.IntegerField()</div>	<div><div><div>2</div><div><div><div> benjaminjkraft repo collab10 hours ago</div><div>Can the map stuff be nullable? I don't know if we will always want to use it, and small chapters will probably not. Also, what's the difference between map_pixel_{x,y} and lat/long?</div></div><div><div>j moldow repo collab9 hours ago</div><div>lat/long could be used to automatically show the building on Google Maps (or equivalent service), whereas map_pixel could be linked to a high-res map image that is saved on the site. Yes, they can be nullable.  url below can also be nullable, and can be used to link to something like whereis.mit.edu, or to a building's website.  I chose lat/long because not all buildings will have distinct addresses that can be searched on Google Maps, but all buildings will have GPS coords that can be searched for.</div></div><div>Add a line note</div></div></div></div>
<div>119 + custom_attributes = models.JSONField() 120 + teacher_custom_attributes = models.JSONField() 121 + admin_custom_attributes = models.JSONField() 122 + description = models.TextField() 123 + notes = models.TextField() 124 + teacher_notes = models.TextField() 125 + admin_notes = models.TextField() 126 + is_requestable = models.BooleanField() 127 + url = models.URLField()</div>	



128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

+ administrator\_email = models.EmailField()

+

+@reversion

+class Furnishing(models.Model):

+ """

+ A permanent, fixed resource in a Location.

+ Global for all programs.

+ Can be an AbstractResource, a ResourceType, or a ResourceFamily, depending on the level of detail that is cared a

+ """

+ is\_active = models.BooleanField()

+ custom\_attributes = models.JSONField()

+ admin\_custom\_attributes = models.JSONField()

+ description = models.TextField()

+ notes = models.TextField()

+ admin\_notes = models.TextField()

+ resource = models.ForeignKey(ContentType, choices=(AbstractResource,ResourceType,ResourceFamily))

2

luac

repo collab

20 hours ago

Does this do the right thing? It looks like it only holds the class (AbstractResource, ResourceType, or ResourceFamily) and not the ID of the object it's referring to, which I assume you want.

jmoldow

repo collab

19 hours ago

I didn't look up the correct syntax, and was using shorthand. I intend for it to point to the object/ID, even if I didn't write it that way.

Add a line note

144

145

146

147

148

+ location = models.ForeignKey(Location)

+ amount = models.IntegerField()

+

+@reversion

+class FloatingFurnishing(models.Model):

3

pricem

repo collab

13 hours ago

Can you help me understand how this model plays along with ResourceAssignment? Most "furnishing" resources are associated directly with a location via the Furnishing model, and cannot be assigned. The FloatingFurnishing model, as I see it, allows you to designate a resource as "floating" and schedule it into different locations at different times. Why couldn't you also use a ResourceAssignment to bind such floating resources to their location and class section?

jmoldow

repo collab

11 hours ago

I'm sorry I didn't comment this better. It is something I thought of at the last minute, and so isn't expressed in my notes.

In real life, there are two different ways you can assign a floating resource. You can assign it to a class, or you can assign it to a classroom. An example of the former is "assign a floating projector to a teacher, who has to pick it up and return it before/after his class". An example of the latter is "assign a floating projector to a classroom, and leave it there all day, so that all classes in that classroom can use it". MIT uses both of these systems. So a FloatingFurnishing augments the built-in Furnishings of a room, but can't be a regular Furnishing because it isn't permanent, and can't be a ResourceAssignment because it isn't tied to a particular class (if you have a class that is rescheduled out of a room, you want the furnishing to stay where it is, and not be moved or removed).

benjaminjkraft

repo collab

11 hours ago

I think even MIT literally only does this for a single-digit number of projectors at Splarks. Maybe we can just have a field on Furnishing, is\_temporary? Or have a meeting\_times key on them, where null is assumed to be "it's built-in, so all of them, for all times at programs" or both? (If we do that, we should allow a Furnishing to be a Resource as well.)

Add a line note

149

150

151

152

153


+ """

+ An assignment of a floating resource to a Location for some set of Events.


+ """

+ is\_active = models.BooleanField()

+ resource = models.ForeignKey(Resource)

154	+ location = models.ForeignKey(Location)	
2	<div><div>pricem repo collab13 hours ago</div><div>By the way, for "floating furnishings" it would be desirable to store the "home" location where it is stored between programs, if applicable.</div><div>jmoldow repo collab11 hours ago</div><div>You could store the "home" location in the notes or attributes of that Resource. FloatingFurnishing is just a temporary assignment of a Resource to a room, so no need to store the "home" location in that object.</div><div>Add a line note</div></div>	
155	+ meeting_times = models.ManyToManyField(Event)	
156	+ ignore_warnings = models.BooleanField()	
157	+ lock_level = models.IntegerField() # for autoscheduler	
158	+	
159	+@reversion	
160	+class ResourceAssignment(models.Model):	
161	+ """	
162	+ An assignment of a ClassSection to a Location and an Event, with Resources.	
163	+ """	
164	+ is_active = models.BooleanField()	
165	+ custom_attributes = models.JSONField()	
166	+ admin_custom_attributes = models.JSONField()	
167	+ notes = models.TextField()	
168	+ admin_notes = models.TextField()	
169	+ resource = models.ManyToManyField(Resource)	
3	<div><div>pricem repo collab13 hours ago</div><div>How does this compare to directly adding many-to-many fields on ClassSection (and thus having the location and assigned resources "belong" to the ClassSection)? I'm mainly asking this in case there is a way to avoid duplicating the times of a section in the schema.</div><div>jmoldow repo collab11 hours ago</div><div>I see this as replacing the meeting_times field on ClassSection/ClassSubject, and forgot to notate that.</div><div>jmoldow repo collab11 hours ago</div><div>Also, later when I have time I'll explain why I grouped all four of these things (Location, ClassSection, Event, and Resources) into one assignment model. I think I hinted at it in my notes, but might not have explained it well.</div><div>Add a line note</div></div>	
170	+ location = models.ForeignKey(Location)	
171	+ section = models.ForeignKey(ClassSection)	
172	+ meeting_time = models.ForeignKey(Event)	
173	+ # unique (resource, location, section) if is_active	
174	+ lock_level = models.IntegerField() # for autoscheduler	
175	+ ignore_warnings = models.BooleanField()	
2	<div><div> benjaminjkraft repo collab10 hours ago</div><div>What's this used for?</div><div>jmoldow repo collab9 hours ago</div><div>Use case for "we purposely are assigning a resource/classroom to two different classes that meet at the same time, or we are assigning two classrooms to a single class, don't issue warnings about this, we're doing it on purpose".</div><div>Add a line note</div></div>	
176	+	
177	+@reversion	
178	+class ResourceRequest(models.Model):	



2




benjaminjkraft

repo collab

20 hours ago





Can/should this have a way for a teacher to say how important their request is, even just as simple as a two-option "do you need this or would it just be useful?" I can see this happening for things like sound systems.



jmoldow

repo collab

20 hours ago



Good idea.

Add a line note

179

+ """

180

+ A request for a ClassSubject (all of its ClassSections) to be assigned a resource.

181

+ Can be either an AbstractResource, ResourceType, or ResourceFamily, depending on the allowed and needed level of

182

+ (the model object requested must have is\_requestable == True).

183

+ """

184

+ is\_active = models.BooleanField()

185

+ resource = models.ForeignKey(ContentType, choices=(AbstractResource,ResourceType,ResourceFamily where is\_requestable == True),

186

+ subject = models.ForeignKey(ClassSubject)


187

+ amount = models.IntegerField(null=True)

188

+ pcnt\_of\_capacity = models.IntegerField(null=True)



3




benjaminjkraft

repo collab

10 hours ago





I'm not sure this is actually a useful field. Nearly all the resources we have, you either have or you don't, and I think the ones that are arbitrarily divisible are all also things that are non-reusable and we just buy more of if we need; even if we're allowing partial usage or requests, I don't think percentage is the way to do it.




jmoldow

repo collab

9 hours ago





I was thinking of something like "I want one laptop/dissection kit for every two students, so the number I need depends on how big a room you assign me". If that's something that never happens, we can remove it.



benjaminjkraft

repo collab

8 hours ago



I see, so it's percentage of the room capacity needed, not percentage of the resource needed. I think in practice that's something we will need to handle manually, and it's only a small number of cases anyway; most of the scalable classes are things the teacher is buying or can buy more of, IME. But I can also see having this exist now that I understand what it does.

Add a line note

189

+ description = models.TextField()

190

+ custom\_attributes = models.JSONField()

191

+ admin\_custom\_attributes = models.JSONField()

192

+ notes = models.TextField()

193

+ admin\_notes = models.TextField()

194

+ wont\_satisfy = models.BooleanField(default=False) # set True if the request has been denied

195


+ is\_satisfied\_override = models.TextField(blank=True, default="") # If the request has been satisfied, but the request is not

196

+

43

esp/esp/resources/resources.txt



View file @ 4cdc12a

... @@ -0,0 +1,43 @@

1

+Resource types

2

+-- classrooms

3

+-- equipment (projector, box of chalk)

4

+-- furnishing

5

+-- consumable, non-consumable (not usable yet)

6

+-- distance function - completely unused

7

+-- only one - if True and ResourceRequestFormSet.static\_resource\_requests, makeaclass form only lets you choose one

8

+-- choices - uses survey's ListField

9

+-- program - can be null, for global

10

+-- autocreated - probably useless

11

+

12

+Resource groups

13  
14

13+- classroom + chalkboards

14+- example: group\_id 32618, classroom + LCD projector + Classroom space (Resource type with multiple values that are

4

benjaminjkraft

repo collab

21 hours ago

Just curious, what's the reasoning to having resource groups being a common id rather than a M2M or ForeignKey?

benjaminjkraft

repo collab

21 hours ago

Nevermind, this is the old schema, I get it.

jmoldow

repo collab

21 hours ago

I wish I could tell you. But I honestly have no idea why it was designed that way.  
  
(I think you think this is resources\_new.txt, but it is actually resources.txt. These are my observations of the current schema.)

pricem

repo collab

13 hours ago

It was designed incorrectly under time pressure.

Add a line note

15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43

15+- not any m2m or foreign key relationship

16+- matching integers on resource objects

17+- This is used only for classroom furnishings

18+

19+Resource requests (usually just a filter on resource type)

20+- foreign key to both section and subject

21+- foreign key to filtered resource type

22+- desired value - from ResourceType.choices

23+

24+Resource assignments (resource + event) (but actually, resource + class section)

25+- foreign key to both section and subject

26+- lock level (for autoscheduler)

27+

28+program resources module

29+

30+static resource requests - what is this? - it is string, should probably be Boolean (not using getBooleanTag)

31+

32+Resource

33+- num students - only used for classrooms

34+- group id - see above

35+- user - usually None, only used for teacher availability (now deprecated, not used in resources app)

36+- is unique - only used for resources pointing to the original, global resource types, and for things with group\_id

37+- event - gives availability for resource - one resource per timeslot - no direct foreign key to program - get from

38+- identical\_resources() - based on name alone, no relationship between them

39+- classrooms are special case, some fields used only for classrooms, many functions expect to be called on classroom

40+- when checking that a class's resource requests are satisfied, only considers furnishings of the assigned room. ad

41+- can assign multiple rooms, but only the first is used in many instances

42+

43+Uses old custom cache, possibly?

118■■■■■

esp/esp/resources/resources\_new.txt

show inline notes

View file @ 4cdc12a

...  
...  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10

@@ -0,0 +1,118 @@

+More thought needed about interaction between ClassSection, class length, Events (timeblocks for class), ResourceAs

+

+Potential Resolution:

+

+@reversion

+class ResourceAssignment:

+ section = ForeignKey(ClassSection)

+ meeting\_time = ForeignKey(Event)

+ classroom = ForeignKey(Classroom)

+ resources = ManyToManyField(Resource)

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

+ ctime = DateTime(auto\_on\_create)

+ mtime = DateTime(auto\_on\_modify)

+ last\_modified\_by = ForeignKey(ESPUser)

+ is\_active = BooleanField

+

+

+

+More thought needed about special resource requests

+

+Make use of fields that Taylor added in his Schema Simplification project.

+

+All m2m assignments should be done via intermediate tables, with created/modified times, and expirations rather tha

+

+Program-specific overwrites for global data

+

+Classroom Type

+ Not sure if this is necessary / optimal.

+ It would match up well with what the autoscheduler expects (I think).

+ Each classroom would have one or more classroom types.

+ In most cases, if not all, the classroom type actually just describes the existence of one or more permanent fixt

+ But some classroom types might not match up to physical resources, so maybe this would be useful.

+

+Classroom

+ Should the name be generalized (not all classes are located in rooms)?

+ "class location", "class space", "space", "location", etc. ?

+ Global (not per program like currently).

+ Per-program Event availabilities.

2

benjaminjkraft

repo collab

20 hours ago

These would be through ClassroomAvailability, right?

jmdow

repo collab

20 hours ago

Yes. I realized that I didn't need to implement this as written, because Events are already tied to programs. So what I implemented was "Event availabilities", with an implicit "per-program" rather than an explicit one.

Add a line note

38

+ m2m of ResourceTypes that the room is always furnishes with.

5

benjaminjkraft

repo collab

20 hours ago

Should this be ResourceTypes, AbstractResources, or Resources? It seems to me that it should be one of the latter.

jmdow

repo collab

20 hours ago

I ended up changing my mind on this. What I ended up implementing was AbstractResource OR ResourceType OR ResourceFamily, but NOT Resource.

If two Resources are of the same AbstractResource, they should be identical, except in identifier ("ADP1" vs "ADP2") or in properties we don't care about. So it seemed unnecessary to have to create individual Resources for each classroom, especially since those Resources are forever fixed and don't need to be scheduled or managed by us.

It seems clear why one would want a furnishing to be an AbstractResource. But why also a ResourceType or ResourceFamily? My thinking was that we might not always know, or care, the level of detail of a room furnishing like we would a floating resource that ESP owns and manages. We might not know whether a room's projector supports HDMI, only that it has "an LCD projector". So our object might look like

Family.....Type.....Abstract

A/V -----Projector-----HDMI

.....I.....I

.....I.....I--- non-HDMI

.....I

.....---Others

(dots used instead of whitespace because github markdown isn't verbatim)

We would use the HDMI / non-HDMI distinction for our floating projectors and any rooms that we happen to know the difference for, but would just point to Projector for all rooms that we don't have that level of detailed knowledge for.

In this simple example, the tree is shallow enough such that you wouldn't want to point to the ResourceFamily A/V, since that is too vague. But some Families might be specific enough to warrant them being the target of a Furnishing. For example,

```
Family.....Family.....Type.....Abstract
Display ----- Board -----Whiteboard----- (further distinction by size)
.....|
.....|-----Blackboard----- (further distinction by size)
```

In this case, it might be reasonable for a Furnishing to be a "Board", if you didn't know or didn't care whether it was a whiteboard or blackboard.

That's how I approach the problem, anyway. It's possible I'm overdesigning this, and it would be better / sufficient to allow only pointing to AbstractResources.



**benjaminjkraft** repo collab

20 hours ago

Okay, I see the argument for allowing it to point to a ResouceType, and if we're bothering with a ContentType key, we might as well allow a ResourceFamily too, although I think in most cases we do know exactly which rooms have what and should be as specific as possible. (Requests, on the other hand, should be encouraged to be fairly general if possible.)

**jmoldow** repo collab

19 hours ago

For Requests, I have is\_requestable fields on the Resource models. If we don't want people to be able to request blackboards or whiteboards specifically, we would set is\_requestable=False on them, and is\_requestable=True on Board. Then, in the Requests tree, they would be able to add Boards, but not blackboards or whiteboards.

Or, if we wanted them to be able to request blackboards and whiteboards, but not specific types/sizes, we could have is\_requestable=True on those ResourceTypes, but is\_requestable=False on all the AbstractResources.

But if we wanted them to be able to specify which type of Mac Adapter they wanted, we would set is\_requestable=True on all of the Mac Adapter AbstractResources.



**benjaminjkraft** repo collab

19 hours ago

Yeah, I saw that and I think it's a great idea. We can think about interfaces later, but essentially I think we should allow specificity but encourage generality in requests.

Add a line note

```
39 +- Instructions field (optional, global, both admin and teacher/student).
40 +- Comments (optional, global, admin).
41 +
42 +Building
43 +- Should the name be generalized (not all classes are located in buildings)?
44 +- "classroom group", "classroom collection", "classroom set", etc.
45 +- Global (not per program).
46 +- Instructions field (optional, global, both admin and teacher/student).
47 +- Comments (optional, global).
48 +- Should each classroom be required to be in a group (e.g., should there be a group for outdoor classes, or meeting
49 +- Should groups be allowed to nest (e.g. Building 4, Floor 2 inside Building 4, or ESG inside Building 24)? If so,
50 +- No functionality for this currently, so isn't strictly necessary; however
51 +- Some nice functionalities can be built for this:
52 +-- The autoscheduler tries to guess buildings based on individual room names, and only schedules back-to-back class
53 +--- The autoscheduler can continue doing what it's doing: only putting back-to-back classes in the same building.
54 +--- Each building can have a symmetric m2m of all the adjacent or nearby buildings where it is legal to schedule a
55 +--- Each pair of buildings can have a distance (probably a unit of distance, though possibly only a relative scale)
56 +-- Each building can be linked to a map location:
57 +--- Store an address, which can be linked to Google Maps.
58 +--- Store coordinates, which can be linked to Google Maps. Optionally, use GeoDjango <https://docs.djangoproject.co
59 +--- Globally store a campus map, and store pixel coordinates in this map for each building.
```

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89


90

```

+-- Store global and per-program notes for each building:
+--- Who to contact to reserve certain rooms.
+--- What doors we need to be sure don't become locked.
+--- What particularly confusing locations should be signed.
+--- Locations of signs for a program.
+--- Etc.
+
+Classroom Availability
+- Foreign key to Classroom
+- Foreign key to Program
+- m2m to Event
+- Lock level (for autoscheduler)
+
+ClassroomAssignment
+(alternatively, foreign key from ClassSection to Classroom, and m2m from ClassSection to Event; though this will ge
+- Foreign key to ClassSection
+- Foreign key to Classroom
+- m2m to Event (alternatively, foreign key to Event, and multiple ClassroomAssignments per ClassSection)
+
+ResourceCategory (other potential names: ResourceClass, ResourceFamily, ResourceSet, ResourceCollection, ResourceGr
+- A set of ResourceTypes.
+- A broad grouping of Resources/ResourceTypes that will appear under a common heading in the ResourceRequest form,
+- Global (not per program like currently).
+
+ResourceType
+- A type of resource, e.g. computer, Mac adapter, board, etc.
+- AbstractResources in the same ResourceType are similar, but not identical.
+- Global (not per program like currently).
+- Teachers can request to get Resources for a ResourceType, and can select which AbstractResources would work for t
+
+AbstractResource

```

7



**benjaminjkraft**

repo collab

21 hours ago




Do you have ideas for how to use all three of these grouping levels while keeping the interface fairly simple? Also, I think the name "AbstractResource" is confusing; maybe change ResourceType to ResourceGroup and make this ResourceType? I don't really have an idea I like, AbstractResource just seems very opaque to me.



**jmoldow**

repo collab

20 hours ago



I imagined an expandable/collapsible tree interface for teacherreg resource requests. Admittedly I didn't much think about the admin create/edit view, but that can probably either be the same tree interface with an inline create/edit form, or manual creation via the admin panel.

I had a hard time deciding on names. Kept changing my mind. I could be convinced to change the names again. My reasoning was that an AbstractResource is as specific as it gets; its descendent won't have more specific physical properties than it has. The only thing an AbstractResource lacks is a physical existence, which is what its descendent Resources have. I thought "abstract" captured that idea better than "type", but maybe "type" is less confusing and close enough to what I was thinking.



**benjaminjkraft**

repo collab

20 hours ago



That sounds reasonable for the teacher view. For the admin view, I think manual creation would be bad; one of the reasons that we seem to always screw up room reservations at Splarks, in my opinion, is that the interface is really really bad, so entering and checking rooms is a pain and error-prone. Part of this would be solved by having global resources so that the "import classrooms to this program" feature would be less clunky, but I think an interface would be important too.

For resources I like the idea of a tree view with an inline edit/create form. For locations, it would be awesome to have a grid that you paint with the availabilities, or alternately something where you enter pairs of start and end times, and the interface interpolates which Events we should have it for, since that's the form in which we usually get the reservations.


I guess AbstractResource makes me think it will be an abstract django model or something; we could ask a non-webmin what they think about the names, especially if we intend to use them in the frontend.



**luac**

repo collab

20 hours ago



benjaminjkraft

repo collab

20 hours ago

++ResourceInstance, I like that a lot better. It might still be worth running the set of names by a non-webmin.

jmoldow

repo collab

19 hours ago

AbstractResource is a model in its own right. It is not an abstract django model. In the naming, I was trying to capture a concept from a philosophy ([http://en.wikipedia.org/wiki/Metaphysics#Empirical\\_and\\_conceptual\\_objects](http://en.wikipedia.org/wiki/Metaphysics#Empirical_and_conceptual_objects)), and "abstract" was the best word I could think of that didn't sound silly and had some relevance to computer science.

But the naming Anthony proposed is also completely reasonable. I had considered something similar, and moved away from it because I wanted the basic, physical units of assignment to be called Resources. But since my nomenclature is confusing, I'm perfectly happy to make this change.

Your Location interface sounds pretty good, though I think we can also use the same interface for ResourceInstance availabilities. And have a clickable feature to initially paint them all available for all program, since that is true of most ResourceInstances and most Locations.

With global objects, setting up the 2nd+ programs will be as easy as adding brand-new Locations and Resources, "deleting" (i.e., setting is\_active=False) no-longer-existing Resources, setting availabilities, and making any other needed tweaks. The amount of adding and deleting, especially of Locations, will probably be minimal compared to present. For the adding and editing that is needed, I agree that a good interface will be better. But if we don't do a fantastic job in the beginning, it won't be the worst thing in the world, and the admin panel is still a good fallack.

Of course, the 1st program will be pretty annoying to set up, and that will require care. So a good interface would definitely make this better, I agree.

benjaminjkraft

repo collab

19 hours ago

Of course, the 1st program will be pretty annoying to set up, and that will require care. So a good interface would definitely make this better, I agree.

Or a script that scrapes roominv...

But in any case, yeah, that seems reasonable.

Add a line note

91

++ Represents an abstract version of a specific type of Resource, e.g. Linux computer, mini-display-port to VGA Mac

92

++ Resources of the same AbstractResource are identical (or similar enough such that teachers and admins don't care

93

++ Global (not per program like currently).

94

++ Is active setting (both global and per-program settings).

95

+

96

+Resource

97

++ An instance of a physical resource, e.g. Chromebook, ADP2 Mac adapter, etc.

98

++ Global (not per program like currently).

99

++ Global is active setting.

100

++ Per-program Event availabilities.

101

+

102

+Resource Availability

103

++ Foreign key to Resource

104

++ Foreign key to Program

105

++ m2m to Event

106

++ Lock level (for autoscheduler)

107

+

108

+ResourceRequest

109

++ m2m from ClassSubject to AbstractResource, through ResourceRequest intermediate model

110

++ notes from teacher

111

+

112

+ResourceAssignment

113

++ Foreign key to ClassSection OR foreign key to Classroom (can assign Resource to a class or to a classroom).

16 of 18

8/14/13 8:24 PM



114  
115  
116  
117  
118

+ - m2m to Event (timeblocks to assign resource for; can get rid of if we decide to always assign resources to class

+ - Foreign key to Resource.

+ - Lock level (for autoscheduler).

+

+All of the above global models should also have optional program-specific overrides.

4

benjaminjkraft

repo collab

21 hours ago

What's your vision for how program-specific overrides will work?

jmoldow

repo collab

20 hours ago

When writing the scheme, I omitted this because:

a) I couldn't think of any use cases where the override would need to be program-specific, as opposed to just updating the model.

b) I didn't want to have to define a program override model for each global model.

c) It seemed like it could potentially be computationally expensive to have to check for overrides on every resource operation.

If it where to become necessary to implement overrides, I would do it in one of the following ways, unless a better suggestion was made:

a)

class ResourceOverride(models.Model):

resource = models.ForeignKey(ContentType, choices=(one of the global Resource models))

program = models.ForeignKey(Program)

field = models.CharField()

value = models.TextField()

b) Enode the override in admin\_custom\_attributes.

In either case a) or b), all references to a resource object, after fetching from the database, would need to be passed, along with the current program, through some method that checks for and applies overrides and returns the new object for use.

Even these solutions are kind of nasty, so I'd prefer to avoid if possible.

benjaminjkraft

repo collab

20 hours ago

I think it would be fine to just not have overrides; certainly most of the use cases don't require them, and usually we should just be making a persistent change. The one that I can think of is if a program wants to override classroom capacities for some or all classrooms, and not have those saved permanently to confuse other programs. One way to handle this would be to keep the official capacity in the admin comment, and set the capacity that the website uses to whatever we feel is appropriate.

pricem

repo collab

13 hours ago

From our experience with the old resources schema, having different resources or resource types between different programs was a pain. I would suggest avoiding overrides and making different resources/families available as necessary for different programs.

Add a line note

1 note on commit 4cdc12a (86 line notes)

☐ Show line notes below

pricem

commented on 4cdc12a

13 hours ago

I just looked at this and I had many of the same reactions as Anthony. In particular, I think you've overthought this and proposed a schema that in the future will look too complicated. Hopefully we can come up with a good compromise between simplicity and functionality.

Write

Preview

Comments are parsed with GitHub Flavored Markdown

17 of 18

8/14/13 8:24 PM

Leave a comment

Attach images by dragging & dropping or [selecting them](#).

38

58

39


59

40

60

**Tip:** You can also add notes to lines in a file.  
Hover to the left of a line to make a note

Comment on this commit



Mute thread

You are receiving notifications because you authored the thread.