

```

1  #!/usr/bin/env python
2  """
3  type will select all objects of that type
4  selector will select a specific group within an object type
5  unique will get at a specific object in a selector
6  """
7  import time
8  from pages import formfield
9
10 class user(object):
11     def __init__(self, _UID=None, _name=None, _email=None, _default_foods=[], _p
rovide=[]):
12         self.type = 'user'
13         self.UID = str(_UID) # unique
14         self.name = _name
15         self.selector = self.name
16         self.ts = None
17
18         # formfield constructor is var_name, human_name, help text (optional), typ
e (defaults to normal), role
19         self.form_fields = [
20
21             # creation = only appears when creating master object for the first time
22             # editmaster = only appears when editing a master
23             # edit = only appears when editing a unique object
24             # admin = only appears when admin viewing
25
26             formfield('UID', 'Unique ID', 'Don\'t touch this unless you\'re Lucas', 'no
rmal', 'admin'),
27             formfield('name', 'Username', 'Don\'t be a faggot and choose something stu
pid', 'normal', 'creation'),
28             formfield('email', 'Email Address', 'Privacy policy? how bout I spam the s
hit out of you', 'normal', 'creation, editmaster, edit'),
29             formfield('default_foods', 'Foods that by default <b>only you</b> consume
', 'e.g. for Lucas, he would put \'grape juice, roast beef\'', etc.', 'large', 'c
reation, editmaster, edit'),
30             formfield('food_owned', 'Food Owned', 'List of food you have directly purc
hased. (Comma seperated)', 'large', 'admin'),
31             formfield('food_consumed', 'Food Consumed', 'List of food you have consume
d. (Comma seperated)', 'large', 'admin'),
32             formfield('meals_owned', 'Meals Cooked', 'List of meals you have cooked. (
Comma seperated)', 'large', 'admin'),
33             formfield('meals_consumed', 'Meals Eaten', 'List of meals you have eaten.
(Comma seperated)', 'large', 'admin'),
34             formfield('receipts_payed', 'Receipts Owned', 'List of receipts you have p
ayed for. (Comma seperated)', 'large', 'admin')
35         ]
36
37         self.email = _email
38         # pw (maybe later?)
39
40         # what they eat (list of food selectors)
41         self.default_foods = _default_foods
42
43         self.food_owned = []
44         self.food_consumed = []
45
46         self.meals_owned = []

```

```

47     self.meals_consumed = []
48
49     self.receipts_payed = []
50
51     def sync(self):
52         self.selector = self.name
53
54     def key(self):
55         self.selector = self.name
56         return str(self.UID) + '.' + str(self.selector) + '.' + str(self.type)
57
58     def print_default_food(self):
59         return ', '.join(self.consume)
60
61     def set_default_vals(self):
62         self.form_fields[0].default_val = self.UID
63         self.form_fields[1].default_val = self.name
64         self.form_fields[2].default_val = self.email
65         self.form_fields[3].default_val = self.default_foods
66         self.form_fields[4].default_val = self.food_owned
67         self.form_fields[5].default_val = self.food_consumed
68         self.form_fields[6].default_val = self.meals_owned
69         self.form_fields[7].default_val = self.receipts_payed
70         self.form_fields[8].default_val = self.meals_consumed
71
72
73 class food(object):
74     def __init__(self, _UID=None, _name=None, _price=None):
75         self.type = 'food'
76         self.UID = str(_UID) # unique
77         self.name = _name # selector
78         self.selector = self.name
79         self.ts = None
80
81         # formfield constructor is var_name, human_name, help text (optional), typ
82         e (defaults to normal)
83         self.form_fields = [
84             formfield('UID', 'Unique ID', 'Don\'t touch this unless you\'re Lucas', 'no
85             rmal', 'admin'),
86             formfield('name', 'Name of food', 'e.g. \'apple\' or \'rocket grenade\'', '
87             normal', 'creation'),
88             formfield('category', 'Categories', 'e.g. For milk, this could be somethin
89             g like \'dairy, drink\'', 'large', 'creation, editmaster, edit'),
90             formfield('price', 'Price', 'Price for one of this type of food, e.g. \'$2
91             0.00\'', 'normal', 'creation, editmaster, edit'),
92             formfield('owner', 'Owner', 'Who paid for this?', 'normal', 'edit'),
93             formfield('eaten', 'Eaten / consumed', 'Has been eaten or consumed, Has <b
94             >not</b> been eaten or consumed, ', 'boolean', 'edit')
95         ]
96
97         self.price = None
98         self.owner = None # who paid, user
99         self.eaten = False # has this been used?
100        self.category = [] # list of strings, e.g. 'milk' would be under 'dairy'
101
102     def sync(self):
103         self.selector = self.name

```

```

99     def key(self):
100         self.selector = self.name
101         return str(self.UID) + '.' + str(self.selector) + '.' + str(self.type)
102
103     def set_default_vals(self):
104         self.form_fields[0].default_val = self.UID
105         self.form_fields[1].default_val = self.name
106         self.form_fields[2].default_val = self.category
107         self.form_fields[3].default_val = self.price
108         self.form_fields[4].default_val = self.owner
109         self.form_fields[5].default_val = self.eaten
110
111 class meal(object):
112     def __init__(self, _UID=None, _name=''):
113         self.type = 'meal'
114         self.UID = _UID # unique
115         self.name = _name # selector
116         self.selector = self.name
117         self.ts = None
118
119         # formfield constructor is var_name, human_name, help text (optional), type (defaults to normal)
120         self.form_fields = [
121             formfield('UID', 'Unique ID', 'Don\'t touch this unless you\'re Lucas', 'normal', 'admin'),
122             formfield('name', 'Name of meal', 'e.g. \'chicken parm\' or \'genital herpes\'', 'normal', 'creation'),
123             formfield('ingrediants', 'Ingrediants', 'Comma seperated list of foods in this dish. <b>IMPORTANT:</b> For now, if the dish uses more than one quantity of some food, enter it in that many times. e.g., if \'chicken parm\' has two boxes of spaghetti, do \'spaghetti, spaghetti\'', 'large', 'creation, editmaster, edit'),
124             formfield('participants', 'Who is eating at the meal', 'Comma seperated list of users who will eat the meal.', 'normal', 'creation, editmaster, edit'),
125             formfield('cooks', 'Chefs', 'Comma seperated list of people responsible for cooking this meal.', 'normal', 'editmaster, edit, creation'),
126             formfield('date', 'Date', 'What day will this meal be eaten <b>MUST BE IN THIS FORM:</b> 9/9/2011', 'normal', 'edit'),
127             formfield('eaten', 'Has this meal been eaten?', 'Yes, No', 'boolean', 'edit')
128         ]
129
130         self.participants = [] # list of user
131         self.price = None
132         self.eaten = None
133         self.cooks = [] # list user
134         self.date = None # date it is gonna be cooked / eaten
135         self.ingrediants = [] # list of food
136
137     def sync(self):
138         self.selector = self.name
139
140     def key(self):
141         self.selector = self.name
142         return str(self.UID) + '.' + str(self.selector) + '.' + str(self.type)
143
144     def set_default_vals(self):
145         self.form_fields[0].default_val = self.UID

```

```

146     self.form_fields[1].default_val = self.name
147     self.form_fields[2].default_val = self.ingredients
148     self.form_fields[3].default_val = self.participants
149     self.form_fields[4].default_val = self.cooks
150     self.form_fields[5].default_val = self.date
151     self.form_fields[6].default_val = self.eaten
152
153 class menu(object):
154     def __init__(self, _UID=None, start_date=''):
155         self.type = 'menu'
156         self.UID = _UID
157         self.start_date = start_date
158         self.selector = self.start_date
159         self.ts = None
160
161         self.form_fields = [
162             formfield('UID', 'Unique ID', 'Don\'t touch this unless you\'re Lucas', 'normal', 'admin'),
163             formfield('start_date', 'Start Date', 'What date will this menu be starting? <b>MUST BE IN THIS FORM:</b> 9/9/2011', 'normal', 'creation, edit'),
164             formfield('meals', 'Meals', 'What meals are on the menu?', 'large', 'creation, editmaster, edit'),
165             formfield('done', 'Done<br>Is the menu over with?', 'Yes, No', 'boolean', 'edit')
166         ]
167
168         #self.people
169         self.meals = [] # meals for the week, maybe change this to dict
170         self.done = None
171
172     def sync(self):
173         self.selector = self.UID
174
175     def key(self):
176         self.selector = self.name
177         return str(self.UID) + '.' + str(self.selector) + '.' + str(self.type)
178
179     def set_default_vals(self):
180         self.form_fields[0].default_val = self.UID
181         self.form_fields[1].default_val = self.start_date
182         self.form_fields[2].default_val = self.meals
183         self.form_fields[3].default_val = self.done
184
185
186 class receipt(object):
187     def __init__(self, _UID=None, purchase_date=''):
188         self.type = 'receipt'
189         self.UID = _UID
190         self.purchase_date = purchase_date
191         self.selector = self.purchase_date
192         self.ts = None
193
194         self.form_fields = [
195             formfield('UID', 'Unique ID', 'Don\'t touch this unless you\'re Lucas', 'normal', 'admin'),
196             formfield('purchase_date', 'Purchase Date', 'When did you buy this? <b>MUST BE IN THIS FORM:</b> 9/9/2011', 'normal', 'creation, editmaster, edit'),
197             formfield('owner', 'Who paid?', 'User who paid for the receipt', 'normal',

```

```
198     creation, editmaster, edit'),
199         formfield('contents','Contents','What foods are on the receipt','large',
200         'creation, editmaster, edit')
201     ]
202     self.owner = None # user who paid
203     self.contents = [] # list of food
204
205     def sync(self):
206         self.selector = self.start_date
207
208     def key(self):
209         self.selector = self.start_date
210         return str(self.UID) + '.' + str(self.selector) + '.' + str(self.type)
211
212     def set_default_vals(self):
213         self.form_fields[0].default_val = self.UID
214         self.form_fields[1].default_val = self.purchase_date
215         self.form_fields[2].default_val = self.owner
216         self.form_fields[3].default_val = self.contents
```