```
from huilib import *
import lod_and_bake
import re

import time

import threading

def get_valid_name(name): # works for Houdini name and Houdini ui name
    return re.sub("[^0-9a-zA-Z\.]+", "_", name).lower() # the lower is something nice but extra

class MegascansFixerDialog(HDialog):
        def __init__(self, megascans_asset_object):
                super(MegascansFixerDialog, self).__init__("Namegoeshere", "Megascans Fixer")
                self.megascans_asset_object = megascans_asset_object # Note, this importantly calls the 'execute_fix' method when the go button is pressed
                # ^ continued: otherwise, use however you please (perhaps have it show the path of the megascans asset).


                self.setWindowLayout('vertical')
                self.setWindowAttributes(stretch = True, margin = 0.1, spacing = 0.1, min_width = 5)

                # using conventions that i'm coming up with to keep this organised
                # assigning a Gadget to self.variablename (a personalised variable name too) when I know i'll access it later

                # Prep

                self.displacement_type_menu_list = ["Displacement", "Vector Displacement"]
                self.displacement_resolution_menu_list = ["8K", "4K", "2K", "1K"]


                #-------------- General

                sep = HSeparator()
                sep.setAttributes(look = 'bevel')

                #-------------- Header
                label0 = HLabel("Megascans Asset Subnet: {}".format(self.megascans_asset_object.megascans_asset_subnet.path()))
                self.addGadget(label0)

                self.addGadget(sep)

                #-------------- Custom LOD options
                label1 = HLabel("Custom LOD options:")
                self.addGadget(label1)


                lod_options_row_layout = HRowLayout()
                self.addLayout(lod_options_row_layout)


                space_column_layout1 = HColumnLayout()
                space_column_layout1.setAttributes(width = 0.25)
                lod_options_row_layout.addLayout(space_column_layout1)

                text_column_layout1 = HColumnLayout()
                text_column_layout1.setAttributes(width = 1.5)
                lod_options_row_layout.addLayout(text_column_layout1)

                other_column_layout1 = HColumnLayout()
                lod_options_row_layout.addLayout(other_column_layout1)


                #space column layout1
                space_label1 = HLabel("")
                space_column_layout1.addGadget(space_label1)

                #text_column_layout1
                text_label1 = HLabel("Polyreduce Percentage:")
                text_column_layout1.addGadget(text_label1)

                #other_column_layout1
                self.polyreduce_percentage_slider = HFloatSlider("polyreduce_percentage_slider", "") # let's see if that works
                self.polyreduce_percentage_slider.setRange((0, 100))
```

```
74                 self.polyreduce_percentage_slider.setValue(50)
75                 self.polyreduce_percentage_slider.lockRange()
76                 other_column_layout1.addGadget(self.polyreduce_percentage_slider)
77 
78                 self.addGadget(sep)
79 
80                 #-------------- Displacement Baking options (everything layout/widget without a personalised baking name has '2' on it)
81                 label2 = HLabel("Displacement Baking options:")
82                 self.addGadget(label2)
83 
84 
85                 displacement_baking_options_row_layout = HRowLayout()
86                 self.addLayout(displacement_baking_options_row_layout)
87 
88 
89                 space_column_layout2 = HColumnLayout()
90                 space column layout2.setAttributes(width = 0.25)
91                 displacement_baking_options_row_layout.addLayout(space_column_layout2)
92 
93                 other column layout2 = HColumnLayout()
94                 displacement_baking_options_row_layout.addLayout(other_column_layout2)
95 
96 
97                 #space_column_layout2
98                 space_label2 = HLabel("")
99                 space_column_layout2.addGadget(space_label2)
100 
101                 #other_column_layout2
102                 self.displacement_type_menu = HStringMenu("displacement_type_menu", "Displacement Map Type", self.displacement_type_menu_list)
103                 other_column_layout2.addGadget(self.displacement_type_menu)
104 
105                 self.displacement_resolution_menu = HStringMenu("displacement_resolution_menu", "Displacement Map Resolution", self.displacement_resolution_menu_list)
106                 other_column_layout2.addGadget(self.displacement_resolution_menu)
107 
108                 self.use temp displacement checkbox = HCheckbox("use_temp_displacement_checkbox", "Temporarily use 1K Resoluton while above is baked (does nothing if 1K is chosen above)")
109                 self.use_temp_displacement_checkbox.setValue("0") # because it's not set to ? anything? by default (but should be set to 0 as that corresponds to unticked - so doing it here), which is FUCKED
110                 other_column_layout2.addGadget(self.use_temp_displacement_checkbox)
111 
112                 # sep
113                 self.addGadget(sep)
114 
115                 #-------------- Other baking options
116                 label2 = HLabel("Other Baking options:")
117                 self.addGadget(label2)
118 
119 
120                 other baking options row layout = HRowLayout()
121                 self.addLayout(other_baking_options_row_layout)
122 
123 
124                 space_column_layout3 = HColumnLayout()
125                 space_column_layout3.setAttributes(width = 0.25)
126                 other_baking_options_row_layout.addLayout(space_column_layout3)
127 
128 
129                 other_baking_options_collapser_layout = HCollapserLayout(label = "", layout = "vertical")
130                 other_baking_options_row_layout.addLayout(other_baking_options_collapser_layout)
131 
132 
133                 map_names_list = list(lod_and_bake.Bake.map_name_and_houdini_parameter_name_dict.keys())
134                 map_names_list.remove("Displacement") # delete because used above
135                 map_names_list.remove("Vector Displacement") # ^
136                 map_names_list.remove("Tangent-Space Vector Displacement") # (TODO) the rest of my code hasn't accounted for Tangent-Space Vector Displacement, removing for now
137 
138                 self.other_maps_to_bake_checkbox_dict = dict()
139                 for map_name in map_names_list:
140                         checkbox_name = get_valid_name(map_name) # checkbox label is map_name
141                         a_checkbox = HCheckbox(checkbox_name, map_name)
142                         a_checkbox.setValue("0") # this is the fix re: huilib having problems with the default value
143 
144                         other_baking_options_collapser_layout.addGadget(a_checkbox)
```

```
145
146                         self.other_maps_to_bake_checkbox_dict[map_name] = a_checkbox # save for a rainy day
147
148                 #hou.ui.displayMessage("hi")
149
150                 # sep
151                 self.addGadget(sep)
152
153
154                 # add columnslayouts to rowlayouts
155
156                 #-------------- Go Button
157                 bottom_row_layout = HRowLayout()
158                 self.addLayout(bottom_row_layout)
159
160
161                 self.go_button = HButton('go button', "Go!")
162                 bottom_row_layout.addGadget(self.go_button)
163
164                 #-------------- "connect call backs"
165
166                 self.go_button.connect(self.cb_go_button) # close is an inherited method
167
168                 #-------------- Initialize
169                 self.initUI()
170
171         def cb_go_button(self):
172                 self.close() # inherited call back method to close the UI
173
174                 a_thread_one = threading.Thread(target = self.close)
175                 a_thread_one.start()
176
177                 a_thread_two = threading.Thread(target = self.blah)
178                 a_thread_two.start()
179
180         def blah(self):
181                 # Get information from UI (I have types in the variable names for the sake of clarity)
182                 polyreduce_percentage_float = float(self.polyreduce_percentage_slider.getValue())
183
184                 displacement_type_str = self.displacement_type_menu_list[self.displacement_type_menu.getValue()]
185                 displacement_resolution_str = self.displacement_resolution_menu_list[self.displacement_resolution_menu.getValue()]
186
187                 use_temp_displacement_checkbox_value = self.use_temp_displacement_checkbox.isChecked() # checkbox_value is "0" or "1", but I want it False or True for clarity
188                 if use_temp_displacement_checkbox_value == 0: # 0 corresponds to unticked, and 1 corresponds to ticked
189                         use_temp_displacement_bool = False
190                 else:
191                         use_temp_displacement_bool = True
192
193                 # construct maps_to_bake_dict
194                 maps_to_bake_dict = lod_and_bake.Bake.maps_to_bake_dict_template
195
196                 maps_to_bake_dict[displacement_type_str] = True # sort out displacement
197
198                 for map_name in self.other_maps_to_bake_checkbox_dict.keys():
199                         checkbox_value = self.other_maps_to_bake_checkbox_dict[map_name].isChecked() # like the above
200                         hou.ui.displayMessage("{}: {}".format(map_name, checkbox_value))
201                         if checkbox_value == 0:
202                                 bake_bool = False
203                         else:
204                                 bake_bool = True
205                         maps_to_bake_dict[map_name] = bake_bool
206
207                 # for testing
208                 message_string = "polyreduce percentage: {}\ndisplacement type: {}\ndisplacement resolution: {}\nuse_temp_displacement: {}\nmaps_to_bake_dict: {}".format(polyreduce_percentage_float, displacement_type_str, displacement_resolution_str, self.use_temp_displacement_checkbox.isChecked(), maps_to_bake_dict)
209                 hou.ui.displayMessage(message_string)
210                 # using displacement resolution as resolution for all maps you ask it to bake! I need to a discussion with Muggy on how it should be dealt with
211                 self.megascans_asset_object.execute_fix(polyreduce_percentage_float, maps_to_bake_dict, displacement_resolution_str, use_temp_displacement_bool)
212
```