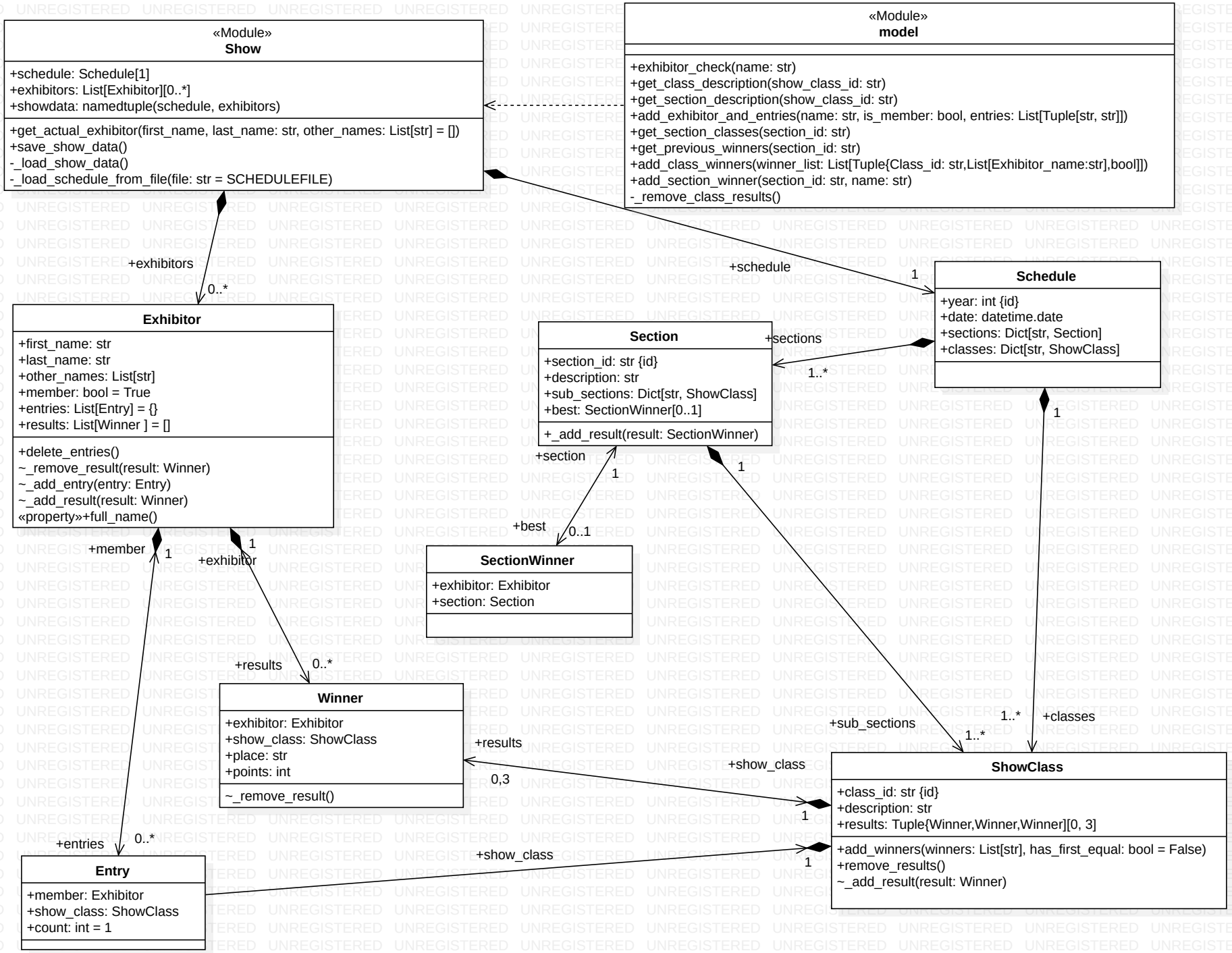
Model::Main


«Module»
Show
+schedule: Schedule[1]
+exhibitors: List[Exhibitor][0..*]
+showdata: namedtuple(schedule, exhibitors)
+get_actual_exhibitor(first_name, last_name: str, other_names: List[str] = [])
+save_show_data()
-_load_show_data()
-_load_schedule_from_file(file: str = SCHEDULEFILE)
«Module»
model
+exhibitor_check(name: str)
+get_class_description(show_class_id: str)
+get_section_description(show_class_id: str)
+add_exhibitor_and_entries(name: str, is_member: bool, entries: List[Tuple[str, str]])
+get_section_classes(section_id: str)
+get_previous_winners(section_id: str)
+add_class_winners(winner_list: List[Tuple{Class_id: str,List[Exhibitor_name:str],bool]])
+add_section_winner(section_id: str, name: str)
-_remove_class_results()
+exhibitors
0..*
+schedule
1
Schedule
+year: int {id}
+date: datetime.date
+sections: Dict[str, Section]
+classes: Dict[str, ShowClass]
Exhibitor
+first_name: str
+last_name: str
+other_names: List[str]
+member: bool = True
+entries: List[Entry] = {}
+results: List[Winner] = []
+delete_entries()
~_remove_result(result: Winner)
~_add_entry(entry: Entry)
~_add_result(result: Winner)
«property»+full_name()
Section
+section_id: str {id}
+description: str
+sub_sections: Dict[str, ShowClass]
+best: SectionWinner[0..1]
+_add_result(result: SectionWinner)
+sections
1..*
+section
1
+best
0..1
SectionWinner
+exhibitor: Exhibitor
+section: Section
+member
1
+exhibitor
1
+results
0..*
Winner
+exhibitor: Exhibitor
+show_class: ShowClass
+place: str
+points: int
~_remove_result()
+results
0,3
+sub_sections
1..*
1..*
+classes
+show_class
1
ShowClass
+class_id: str {id}
+description: str
+results: Tuple{Winner,Winner,Winner][0, 3]
+add_winners(winners: List[str], has_first_equal: bool = False)
+remove_results()
~_add_result(result: Winner)
+entries
0..*
+show_class
1
Entry
+member: Exhibitor
+show_class: ShowClass
+count: int = 1