App.Models.constructor.Constructor

score: int
nationality: str
name: str
id: str

set_score(drivers): None

App.Models.client.Client

total_spent: int
tickets: Ticket[]
name: str
id: str
age: int

add_ticket(ticket): None

App.Models.race.Race

date: str
circuit: Circuit
round: str
name: str
restaurants: Restaurant
vip_seats: Seat[[],]
map: list
attendance: int
sold_tickets: int
general_seats: Seat[[],]

define_general_seats_matrix(): None
define_vip_seats_matrix(): None
print_general_seats(): None
print_vip_seats(): None
print_general_seats_tabulate(): None
print_vip_seats_tabulate(): None
pretty_print_attendance(): None
pretty_print_sold_tickets(): None

App.Models.circuit.Circuit

name: str
location: Location
circuitId: str

App.Models.location.Location

country: str
locality: str
lat: str
long: str

App.Models.restaurant.Restaurant

name: str
items: Refreshment[]

0..*

App.Models.driver.Driver

firstName: str
lastName: str
score: int
code: str
permanentNumber: str
nationality: str
dateOfBirth: str
id: str
team: str

App.Models.ticket.Ticket

race_round: str
code: str
total_price: int
discount: bool
type: TicketType

calculate_price(): None

1..*

App.Models.ticket.TicketType(Enum)

VIP: str
GENERAL: str

1

App.Models.seat.Seat

assisted: bool
code: str
taken: bool
position: str

show_all(): None

0..*

App.Models.refreshment.Refreshment

price: int
name: str
total_sold: int
stock: int

0..*

parent

App.Models.refreshment.Food

type: str

child

App.Models.refreshment.Drink

type: str

child