

COS 214 Project

1

Generated by Doxygen 1.8.13

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	File Index	7
3.1	File List	7
4	Class Documentation	9
4.1	AggressiveDriver Class Reference	9
4.1.1	Detailed Description	10
4.1.2	Constructor & Destructor Documentation	10
4.1.2.1	AggressiveDriver()	10
4.2	AverageDriver Class Reference	10
4.2.1	Constructor & Destructor Documentation	11
4.2.1.1	AverageDriver()	11
4.3	BigBrother Class Reference	11
4.3.1	Detailed Description	12
4.3.2	Member Function Documentation	12
4.3.2.1	visit() [1/7]	12
4.3.2.2	visit() [2/7]	12
4.3.2.3	visit() [3/7]	13
4.3.2.4	visit() [4/7]	13
4.3.2.5	visit() [5/7]	13

4.3.2.6	visit() [6/7]	14
4.3.2.7	visit() [7/7]	14
4.4	Car Class Reference	14
4.4.1	Detailed Description	17
4.4.2	Constructor & Destructor Documentation	17
4.4.2.1	Car() [1/3]	17
4.4.2.2	Car() [2/3]	18
4.4.2.3	Car() [3/3]	18
4.4.2.4	~Car()	18
4.4.3	Member Function Documentation	18
4.4.3.1	add()	18
4.4.3.2	clone()	19
4.4.3.3	FullClone()	19
4.4.3.4	getAcceleration()	19
4.4.3.5	getCarDamage()	20
4.4.3.6	getCarFuel()	20
4.4.3.7	getCarID()	20
4.4.3.8	getCarTyre()	20
4.4.3.9	getCarTyres()	21
4.4.3.10	getDescription()	21
4.4.3.11	getDriver()	21
4.4.3.12	getHandling()	21
4.4.3.13	getLap()	22
4.4.3.14	getManager()	22
4.4.3.15	getModelNumber()	22
4.4.3.16	getModelType()	22
4.4.3.17	getNumTyres()	23
4.4.3.18	getSpeed()	23
4.4.3.19	getState()	23
4.4.3.20	getTeam()	23

4.4.3.21	getTrackPart()	24
4.4.3.22	getTrackTime()	24
4.4.3.23	notifyTeam()	24
4.4.3.24	racing()	24
4.4.3.25	ready()	24
4.4.3.26	RegistrationNotify()	24
4.4.3.27	setAcceleration()	25
4.4.3.28	setCarDamage()	25
4.4.3.29	setCarFuel()	25
4.4.3.30	setCarTyre()	26
4.4.3.31	setChanged()	26
4.4.3.32	setDescription()	26
4.4.3.33	setDriver()	26
4.4.3.34	setHandling()	27
4.4.3.35	setLap()	27
4.4.3.36	setManager()	27
4.4.3.37	setRefuel()	28
4.4.3.38	setRepair()	28
4.4.3.39	setSpeed()	28
4.4.3.40	setState()	28
4.4.3.41	setTeam()	29
4.4.3.42	setTrackPart()	29
4.4.3.43	setTrackTime()	29
4.4.3.44	showCarCondition()	30
4.4.3.45	showCarStats()	30
4.4.3.46	stopped()	30
4.4.3.47	toString()	30
4.4.4	Member Data Documentation	30
4.4.4.1	carDecorate	31
4.5	CarFactory Class Reference	31

4.5.1	Detailed Description	31
4.5.2	Member Function Documentation	32
4.5.2.1	produceElectric()	32
4.5.2.2	produceSports()	32
4.5.2.3	produceStandard()	32
4.6	ConcreteBigBrother Class Reference	33
4.6.1	Detailed Description	34
4.6.2	Member Function Documentation	34
4.6.2.1	visit() [1/7]	34
4.6.2.2	visit() [2/7]	34
4.6.2.3	visit() [3/7]	35
4.6.2.4	visit() [4/7]	35
4.6.2.5	visit() [5/7]	35
4.6.2.6	visit() [6/7]	35
4.6.2.7	visit() [7/7]	37
4.7	ConcreteMediator Class Reference	37
4.7.1	Detailed Description	38
4.7.2	Constructor & Destructor Documentation	38
4.7.2.1	~ConcreteMediator()	38
4.7.3	Member Function Documentation	38
4.7.3.1	addMember()	38
4.7.3.2	getManager()	39
4.7.3.3	notify()	39
4.7.3.4	notifyManager()	39
4.8	ConcreteRaceManager Class Reference	40
4.8.1	Detailed Description	41
4.8.2	Member Function Documentation	41
4.8.2.1	addCars()	41
4.8.2.2	addRacetrack()	41
4.8.2.3	getCarInfo()	42

4.8.2.4	getLap()	42
4.8.2.5	getLapMax()	42
4.8.2.6	pauseRace()	42
4.8.2.7	printLeaderBoard()	43
4.8.2.8	readyRace()	43
4.8.2.9	resumeRace()	43
4.8.2.10	setLapCount()	43
4.8.2.11	setLapMax()	44
4.8.2.12	startRace()	44
4.8.2.13	stopRace()	44
4.9	ConcreteRegistrationManager Class Reference	45
4.9.1	Detailed Description	46
4.9.2	Constructor & Destructor Documentation	46
4.9.2.1	ConcreteRegistrationManager()	46
4.9.2.2	~ConcreteRegistrationManager()	46
4.9.3	Member Function Documentation	46
4.9.3.1	addCar()	46
4.9.3.2	addTrack()	47
4.9.3.3	getCars()	47
4.9.3.4	getTrack()	47
4.10	Driver Class Reference	48
4.10.1	Detailed Description	49
4.10.2	Member Function Documentation	49
4.10.2.1	getDriverAbilty()	49
4.10.2.2	getFuelAbilty()	49
4.10.2.3	getTyreAbilty()	49
4.10.2.4	setAbility()	49
4.10.2.5	setFuelAbility()	50
4.10.2.6	setTyreAbility()	50
4.11	ElectricCar Class Reference	50

4.11.1 Detailed Description	51
4.11.2 Constructor & Destructor Documentation	52
4.11.2.1 ElectricCar() [1/2]	52
4.11.2.2 ElectricCar() [2/2]	52
4.11.2.3 ~ElectricCar()	52
4.11.3 Member Function Documentation	53
4.11.3.1 clone()	53
4.11.3.2 FullClone()	53
4.11.3.3 getDescription()	53
4.12 ElectricFormulaOne Class Reference	54
4.12.1 Detailed Description	55
4.12.2 Constructor & Destructor Documentation	55
4.12.2.1 ElectricFormulaOne() [1/2]	55
4.12.2.2 ElectricFormulaOne() [2/2]	55
4.12.3 Member Function Documentation	55
4.12.3.1 clone()	55
4.13 ElectricGoKart Class Reference	56
4.13.1 Detailed Description	57
4.13.2 Constructor & Destructor Documentation	57
4.13.2.1 ElectricGoKart() [1/2]	57
4.13.2.2 ElectricGoKart() [2/2]	58
4.13.3 Member Function Documentation	59
4.13.3.1 clone()	59
4.14 ElectricRoadster Class Reference	59
4.14.1 Detailed Description	60
4.14.2 Constructor & Destructor Documentation	61
4.14.2.1 ElectricRoadster() [1/2]	61
4.14.2.2 ElectricRoadster() [2/2]	61
4.14.3 Member Function Documentation	61
4.14.3.1 clone()	61

4.15 Facade Class Reference	62
4.15.1 Detailed Description	62
4.15.2 Constructor & Destructor Documentation	63
4.15.2.1 Facade()	63
4.15.2.2 ~Facade()	63
4.15.3 Member Function Documentation	63
4.15.3.1 copyCar()	63
4.15.3.2 createCustomCar()	63
4.15.3.3 createCustomRaceTrack()	64
4.15.3.4 createDriver()	64
4.15.3.5 createTeam()	64
4.15.3.6 prepRace()	64
4.15.3.7 registerCar() [1/2]	65
4.15.3.8 registerCar() [2/2]	65
4.15.3.9 registerTrack()	65
4.15.3.10 StartRace()	65
4.16 FinishDecorator Class Reference	66
4.16.1 Detailed Description	67
4.16.2 Constructor & Destructor Documentation	67
4.16.2.1 FinishDecorator()	67
4.16.2.2 ~FinishDecorator()	67
4.17 FlameVinyl Class Reference	68
4.17.1 Detailed Description	69
4.17.2 Constructor & Destructor Documentation	69
4.17.2.1 FlameVinyl() [1/2]	69
4.17.2.2 ~FlameVinyl()	69
4.17.2.3 FlameVinyl() [2/2]	69
4.17.3 Member Function Documentation	70
4.17.3.1 FullClone()	70
4.18 FormulaOneFactory Class Reference	70

4.18.1 Detailed Description	71
4.18.2 Member Function Documentation	71
4.18.2.1 produceElectric()	71
4.18.2.2 produceSports()	71
4.18.2.3 produceStandard()	72
4.19 GoKartFactory Class Reference	72
4.19.1 Detailed Description	73
4.19.2 Member Function Documentation	73
4.19.2.1 produceElectric()	73
4.19.2.2 produceSports()	73
4.19.2.3 produceStandard()	74
4.20 LeftEighth Class Reference	74
4.20.1 Detailed Description	75
4.20.2 Constructor & Destructor Documentation	75
4.20.2.1 LeftEighth()	75
4.20.2.2 ~LeftEighth()	75
4.20.3 Member Function Documentation	75
4.20.3.1 accept()	75
4.20.3.2 add()	76
4.20.3.3 addTime()	76
4.20.3.4 getAverageTime()	76
4.20.3.5 print()	77
4.21 LeftPeelOff Class Reference	77
4.21.1 Detailed Description	78
4.21.2 Constructor & Destructor Documentation	78
4.21.2.1 LeftPeelOff()	78
4.21.2.2 ~LeftPeelOff()	78
4.21.3 Member Function Documentation	78
4.21.3.1 accept()	78
4.21.3.2 add()	79

4.21.3.3	addTime()	79
4.21.3.4	getAverageTime()	79
4.21.3.5	print()	80
4.22	LeftPeelOn Class Reference	80
4.22.1	Detailed Description	81
4.22.2	Constructor & Destructor Documentation	81
4.22.2.1	LeftPeelOn()	81
4.22.2.2	~LeftPeelOn()	81
4.22.3	Member Function Documentation	81
4.22.3.1	accept()	81
4.22.3.2	add()	82
4.22.3.3	addTime()	82
4.22.3.4	getAverageTime()	82
4.22.3.5	print()	83
4.23	Manager Class Reference	83
4.23.1	Detailed Description	84
4.23.2	Constructor & Destructor Documentation	84
4.23.2.1	Manager()	84
4.23.3	Member Function Documentation	84
4.23.3.1	getDamage()	84
4.23.3.2	getFuelLevel()	85
4.23.3.3	getTyreCondition()	85
4.23.3.4	setDamage()	85
4.23.3.5	setFuelLevel()	86
4.23.3.6	setTyreCondition()	86
4.23.3.7	update()	86
4.24	Mechanic Class Reference	87
4.24.1	Detailed Description	88
4.24.2	Constructor & Destructor Documentation	88
4.24.2.1	Mechanic()	88

4.24.3	Member Function Documentation	88
4.24.3.1	getDamage()	89
4.24.3.2	getFuelLevel()	89
4.24.3.3	getTyreCondition()	89
4.24.3.4	repair()	89
4.24.3.5	setDamage()	89
4.24.3.6	setFuelLevel()	90
4.24.3.7	setTyreCondition()	90
4.24.3.8	update()	90
4.25	Mediator Class Reference	91
4.25.1	Detailed Description	91
4.25.2	Constructor & Destructor Documentation	92
4.25.2.1	~Mediator()	92
4.25.3	Member Function Documentation	92
4.25.3.1	addMember()	92
4.25.3.2	notify()	92
4.25.3.3	notifyManager()	92
4.26	Nitro Class Reference	93
4.26.1	Detailed Description	94
4.26.2	Constructor & Destructor Documentation	94
4.26.2.1	Nitro() [1/2]	94
4.26.2.2	~Nitro()	95
4.26.2.3	Nitro() [2/2]	95
4.26.3	Member Function Documentation	95
4.26.3.1	FullClone()	95
4.27	PassiveDriver Class Reference	96
4.27.1	Constructor & Destructor Documentation	96
4.27.1.1	PassiveDriver()	96
4.28	PimpMyRide Class Reference	97
4.28.1	Detailed Description	98

4.28.2	Constructor & Destructor Documentation	98
4.28.2.1	PimpMyRide()	98
4.28.2.2	~PimpMyRide()	98
4.28.3	Member Function Documentation	98
4.28.3.1	add()	98
4.28.3.2	clone()	99
4.28.3.3	FullClone()	99
4.28.3.4	showCarStats()	100
4.29	PimpMyTrack Class Reference	100
4.29.1	Detailed Description	101
4.29.2	Constructor & Destructor Documentation	101
4.29.2.1	PimpMyTrack()	101
4.29.2.2	~PimpMyTrack()	101
4.29.3	Member Function Documentation	101
4.29.3.1	accept()	101
4.29.3.2	add()	102
4.29.3.3	addTime()	102
4.29.3.4	print()	102
4.30	PitCrew Class Reference	103
4.30.1	Detailed Description	104
4.30.2	Constructor & Destructor Documentation	104
4.30.2.1	PitCrew()	104
4.30.3	Member Function Documentation	104
4.30.3.1	changed()	105
4.30.3.2	changedCar()	105
4.30.3.3	getDamage()	105
4.30.3.4	getDescription()	105
4.30.3.5	getFuelLevel()	105
4.30.3.6	getTyreCondition()	106
4.30.3.7	registerWork()	106

4.30.3.8	setDamage()	106
4.30.3.9	setDescription()	106
4.30.3.10	setFuelLevel()	107
4.30.3.11	setTyreCondition()	107
4.30.3.12	update()	107
4.31	PitStop Class Reference	108
4.31.1	Detailed Description	109
4.31.2	Constructor & Destructor Documentation	109
4.31.2.1	PitStop()	109
4.31.2.2	~PitStop()	109
4.31.3	Member Function Documentation	109
4.31.3.1	addCar()	109
4.31.3.2	attach()	110
4.31.3.3	attachManager()	110
4.31.3.4	detach()	110
4.31.3.5	getCar()	111
4.31.3.6	getCarStats()	111
4.31.3.7	getManager()	111
4.31.3.8	getMember()	111
4.31.3.9	getName()	111
4.31.3.10	getNumMembers()	112
4.31.3.11	notify()	112
4.31.3.12	setDamage()	112
4.31.3.13	setFuelLevel()	112
4.31.3.14	setTyreCondition()	113
4.31.3.15	showCar()	113
4.31.3.16	showCrew()	113
4.31.3.17	showManager()	113
4.31.3.18	toString()	114
4.32	PitStopDecorator Class Reference	114

4.32.1 Detailed Description	115
4.32.2 Constructor & Destructor Documentation	115
4.32.2.1 PitStopDecorator()	115
4.32.2.2 ~PitStopDecorator()	116
4.33 RaceManager Class Reference	116
4.33.1 Detailed Description	116
4.33.2 Member Function Documentation	117
4.33.2.1 addCars()	117
4.33.2.2 addRacetrack()	117
4.33.2.3 pauseRace()	117
4.33.2.4 printLeaderBoard()	118
4.33.2.5 readyRace()	118
4.33.2.6 resumeRace()	118
4.33.2.7 startRace()	118
4.33.2.8 stopRace()	118
4.34 RaceTrack Class Reference	119
4.34.1 Detailed Description	120
4.34.2 Constructor & Destructor Documentation	120
4.34.2.1 RaceTrack()	120
4.34.2.2 ~RaceTrack()	120
4.34.3 Member Function Documentation	120
4.34.3.1 accept()	120
4.34.3.2 add()	121
4.34.3.3 addAllCars()	121
4.34.3.4 addTime()	121
4.34.3.5 getAllCars()	121
4.34.3.6 getNumComponents()	122
4.34.3.7 getRaceTrackID()	122
4.34.3.8 makeAccept()	122
4.34.3.9 moveCar()	123

4.34.3.10 print()	123
4.34.3.11 removeAllCars()	123
4.34.3.12 show()	124
4.35 RaceTrackComponent Class Reference	124
4.35.1 Detailed Description	125
4.35.2 Constructor & Destructor Documentation	126
4.35.2.1 RaceTrackComponent()	126
4.35.2.2 ~RaceTrackComponent()	126
4.35.3 Member Function Documentation	126
4.35.3.1 accept()	126
4.35.3.2 add()	126
4.35.3.3 addAllCars()	127
4.35.3.4 addCar()	127
4.35.3.5 addTime()	127
4.35.3.6 decorateTrack()	127
4.35.3.7 getAllCars()	128
4.35.3.8 getCars()	128
4.35.3.9 getDecorator()	128
4.35.3.10 getDescription()	129
4.35.3.11 getNumComponents()	129
4.35.3.12 makeAccept()	129
4.35.3.13 moveCar()	129
4.35.3.14 print()	130
4.35.3.15 removeAllCars()	130
4.35.3.16 removeCar()	130
4.35.3.17 setDescription()	131
4.35.3.18 show()	131
4.35.4 Member Data Documentation	131
4.35.4.1 cars	131
4.35.4.2 decorate	131

4.36	Racing Class Reference	132
4.37	Ready Class Reference	133
4.38	Refueller Class Reference	133
4.38.1	Detailed Description	135
4.38.2	Constructor & Destructor Documentation	135
4.38.2.1	Refueller()	135
4.38.3	Member Function Documentation	135
4.38.3.1	getDamage()	135
4.38.3.2	getFuelLevel()	136
4.38.3.3	getTyreCondition()	136
4.38.3.4	refuel()	136
4.38.3.5	setDamage()	136
4.38.3.6	setFuelLevel()	137
4.38.3.7	setTyreCondition()	137
4.38.3.8	update()	137
4.39	RegistratcionManager Class Reference	138
4.39.1	Member Function Documentation	138
4.39.1.1	addCar()	138
4.39.1.2	addTrack()	138
4.39.1.3	getCars()	138
4.39.1.4	getTrack()	139
4.40	RightEighth Class Reference	139
4.40.1	Detailed Description	140
4.40.2	Constructor & Destructor Documentation	140
4.40.2.1	RightEighth()	140
4.40.2.2	~RightEighth()	141
4.40.3	Member Function Documentation	141
4.40.3.1	accept()	141
4.40.3.2	add()	141
4.40.3.3	addTime()	141

4.40.3.4	getAverageTime()	142
4.40.3.5	print()	142
4.41	RightPeelOff Class Reference	142
4.41.1	Detailed Description	143
4.41.2	Constructor & Destructor Documentation	143
4.41.2.1	RightPeelOff()	143
4.41.2.2	~RightPeelOff()	144
4.41.3	Member Function Documentation	144
4.41.3.1	accept()	144
4.41.3.2	add()	144
4.41.3.3	addTime()	144
4.41.3.4	getAverageTime()	145
4.41.3.5	print()	145
4.42	RightPeelOn Class Reference	145
4.42.1	Detailed Description	146
4.42.2	Constructor & Destructor Documentation	146
4.42.2.1	RightPeelOn()	146
4.42.2.2	~RightPeelOn()	147
4.42.3	Member Function Documentation	147
4.42.3.1	accept()	147
4.42.3.2	add()	147
4.42.3.3	addTime()	147
4.42.3.4	getAverageTime()	148
4.42.3.5	print()	148
4.43	RoadsterFactory Class Reference	148
4.43.1	Detailed Description	149
4.43.2	Member Function Documentation	149
4.43.2.1	produceElectric()	149
4.43.2.2	produceSports()	150
4.43.2.3	produceStandard()	150

4.44 SandPitsDecorator Class Reference	150
4.44.1 Detailed Description	151
4.44.2 Constructor & Destructor Documentation	151
4.44.2.1 SandPitsDecorator()	151
4.44.2.2 ~SandPitsDecorator()	152
4.45 SkullVinyl Class Reference	152
4.45.1 Detailed Description	153
4.45.2 Constructor & Destructor Documentation	153
4.45.2.1 SkullVinyl() [1/2]	153
4.45.2.2 ~SkullVinyl()	153
4.45.2.3 SkullVinyl() [2/2]	153
4.45.3 Member Function Documentation	154
4.45.3.1 FullClone()	154
4.46 Slick Class Reference	154
4.46.1 Detailed Description	155
4.46.2 Constructor & Destructor Documentation	155
4.46.2.1 Slick() [1/2]	155
4.46.2.2 ~Slick()	156
4.46.2.3 Slick() [2/2]	156
4.46.3 Member Function Documentation	156
4.46.3.1 FullClone()	156
4.47 Spoiler Class Reference	157
4.47.1 Detailed Description	158
4.47.2 Constructor & Destructor Documentation	158
4.47.2.1 Spoiler() [1/2]	158
4.47.2.2 ~Spoiler()	158
4.47.2.3 Spoiler() [2/2]	158
4.47.3 Member Function Documentation	159
4.47.3.1 FullClone()	159
4.48 SportsCar Class Reference	159

4.48.1	Detailed Description	160
4.48.2	Constructor & Destructor Documentation	160
4.48.2.1	SportsCar() [1/2]	160
4.48.2.2	SportsCar() [2/2]	161
4.48.2.3	~SportsCar()	161
4.48.3	Member Function Documentation	161
4.48.3.1	clone()	161
4.48.3.2	FullClone()	162
4.48.3.3	getDescription()	162
4.49	SportsFormulaOne Class Reference	162
4.49.1	Detailed Description	163
4.49.2	Constructor & Destructor Documentation	163
4.49.2.1	SportsFormulaOne() [1/2]	163
4.49.2.2	SportsFormulaOne() [2/2]	164
4.49.3	Member Function Documentation	165
4.49.3.1	clone()	165
4.50	SportsGoKart Class Reference	165
4.50.1	Detailed Description	166
4.50.2	Constructor & Destructor Documentation	167
4.50.2.1	SportsGoKart() [1/2]	167
4.50.2.2	SportsGoKart() [2/2]	167
4.50.3	Member Function Documentation	167
4.50.3.1	clone()	167
4.51	SportsRoadster Class Reference	168
4.51.1	Detailed Description	169
4.51.2	Constructor & Destructor Documentation	169
4.51.2.1	SportsRoadster() [1/2]	169
4.51.2.2	SportsRoadster() [2/2]	170
4.51.3	Member Function Documentation	171
4.51.3.1	clone()	171

4.52 StandardCar Class Reference	171
4.52.1 Detailed Description	172
4.52.2 Constructor & Destructor Documentation	173
4.52.2.1 StandardCar() [1/2]	173
4.52.2.2 StandardCar() [2/2]	173
4.52.2.3 ~StandardCar()	173
4.52.3 Member Function Documentation	174
4.52.3.1 clone()	174
4.52.3.2 FullClone()	174
4.52.3.3 getDescription()	174
4.53 StandardFormulaOne Class Reference	175
4.53.1 Detailed Description	176
4.53.2 Constructor & Destructor Documentation	176
4.53.2.1 StandardFormulaOne() [1/2]	176
4.53.2.2 StandardFormulaOne() [2/2]	176
4.53.3 Member Function Documentation	176
4.53.3.1 clone()	176
4.54 StandardGoKart Class Reference	177
4.54.1 Detailed Description	178
4.54.2 Constructor & Destructor Documentation	178
4.54.2.1 StandardGoKart() [1/2]	178
4.54.2.2 StandardGoKart() [2/2]	179
4.54.3 Member Function Documentation	180
4.54.3.1 clone()	180
4.55 StandardRoadster Class Reference	180
4.55.1 Detailed Description	181
4.55.2 Constructor & Destructor Documentation	182
4.55.2.1 StandardRoadster() [1/2]	182
4.55.2.2 StandardRoadster() [2/2]	182
4.55.3 Member Function Documentation	182

4.55.3.1	clone()	182
4.56	StartDecorator Class Reference	183
4.56.1	Detailed Description	184
4.56.2	Constructor & Destructor Documentation	184
4.56.2.1	StartDecorator()	184
4.56.2.2	~StartDecorator()	185
4.57	State Class Reference	185
4.57.1	Detailed Description	185
4.57.2	Member Function Documentation	185
4.57.2.1	racing()	185
4.57.2.2	ready()	186
4.57.2.3	stopped()	186
4.57.2.4	toString()	186
4.58	Stopped Class Reference	187
4.59	Straight Class Reference	187
4.59.1	Detailed Description	188
4.59.2	Constructor & Destructor Documentation	189
4.59.2.1	Straight()	189
4.59.2.2	~Straight()	189
4.59.3	Member Function Documentation	189
4.59.3.1	accept()	189
4.59.3.2	add()	189
4.59.3.3	addTime()	190
4.59.3.4	getAverageTime()	190
4.59.3.5	print()	190
4.60	Team Class Reference	191
4.60.1	Detailed Description	191
4.60.2	Constructor & Destructor Documentation	192
4.60.2.1	Team()	192
4.60.3	Member Function Documentation	192

4.60.3.1	getCarStats()	192
4.61	TyreChanger Class Reference	192
4.61.1	Detailed Description	193
4.61.2	Constructor & Destructor Documentation	193
4.61.2.1	TyreChanger()	194
4.61.3	Member Function Documentation	194
4.61.3.1	changeTyre()	194
4.61.3.2	getDamage()	194
4.61.3.3	getFuelLevel()	194
4.61.3.4	getTyreCondition()	195
4.61.3.5	setDamage()	195
4.61.3.6	setFuelLevel()	195
4.61.3.7	setTyreCondition()	195
4.61.3.8	update()	196
5	File Documentation	197
5.1	AggressiveDriver.h File Reference	197
5.2	BigBrother.h File Reference	198
5.3	Car.h File Reference	198
5.4	CarFactory.h File Reference	199
5.5	ConcreteBigBrother.h File Reference	200
5.6	ConcreteMediator.h File Reference	201
5.7	ConcreteRaceManager.h File Reference	202
5.8	ConcreteRegistrationManager.h File Reference	203
5.9	Driver.h File Reference	205
5.10	ElectricCar.h File Reference	205
5.11	ElectricFormulaOne.h File Reference	206
5.12	ElectricRoadster.h File Reference	208
5.13	Facade.h File Reference	209
5.14	FinishDecorator.h File Reference	210
5.15	FlameVinyl.h File Reference	211

5.16 GoKartFactory.h File Reference	212
5.17 LeftEighth.h File Reference	213
5.18 LeftPeelOff.h File Reference	213
5.19 LeftPeelOn.h File Reference	214
5.20 Manager.h File Reference	215
5.21 Mechanic.h File Reference	216
5.22 Mediator.h File Reference	217
5.23 Nitro.h File Reference	218
5.24 PimpMyRide.h File Reference	219
5.25 PimpMyTrack.h File Reference	220
5.26 PitCrew.h File Reference	221
5.27 PitStop.h File Reference	222
5.28 PitStopDecorator.h File Reference	224
5.29 RaceManager.h File Reference	224
5.30 RaceTrack.h File Reference	225
5.31 RaceTrackComponent.h File Reference	226
5.32 Refueller.h File Reference	227
5.33 RightEighth.h File Reference	228
5.34 RightPeelOff.h File Reference	229
5.35 RightPeelOn.h File Reference	230
5.36 RoadsterFactory.h File Reference	231
5.37 SandPitsDecorator.h File Reference	232
5.38 SkullVinyl.h File Reference	233
5.39 Slick.h File Reference	234
5.40 Spoiler.h File Reference	235
5.41 SportsCar.h File Reference	236
5.42 SportsFormulaOne.h File Reference	237
5.43 SportsGoKart.h File Reference	239
5.44 SportsRoadster.h File Reference	240
5.45 StandardCar.h File Reference	241
5.46 StandardFormulaOne.h File Reference	242
5.47 StandardGoKart.h File Reference	243
5.48 StandardRoadster.h File Reference	244
5.49 StartDecorator.h File Reference	245
5.50 State.h File Reference	246
5.51 Straight.h File Reference	246
5.52 Team.h File Reference	247
5.53 TyreChanger.h File Reference	248

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

BigBrother	11
ConcreteBigBrother	33
Car	14
ElectricCar	50
ElectricFormulaOne	54
ElectricGoKart	56
ElectricRoadster	59
PimpMyRide	97
FlameVinyl	68
Nitro	93
SkullVinyl	152
Slick	154
Spoiler	157
SportsCar	159
SportsFormulaOne	162
SportsGoKart	165
SportsRoadster	168
StandardCar	171
StandardFormulaOne	175
StandardGoKart	177
StandardRoadster	180
CarFactory	31
FormulaOneFactory	70
GoKartFactory	72
RoadsterFactory	148
Driver	48
AggressiveDriver	9
AverageDriver	10
PassiveDriver	96
Facade	62
Mediator	91
ConcreteMediator	37
PitCrew	103

Manager	83
Mechanic	87
Refueller	133
TyreChanger	192
PitStop	108
Team	191
RaceManager	116
ConcreteRaceManager	40
RaceTrackComponent	124
LeftEighth	74
LeftPeelOff	77
LeftPeelOn	80
PimpMyTrack	100
FinishDecorator	66
PitStopDecorator	114
SandPitsDecorator	150
StartDecorator	183
RaceTrack	119
RightEighth	139
RightPeelOff	142
RightPeelOn	145
Straight	187
RegistratcionManager	138
ConcreteRegistrationManager	45
State	185
Racing	132
Ready	133
Stopped	187

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

AggressiveDriver	
ConcreteStrategy for strategy design pattern	9
AverageDriver	10
BigBrother	
Visitor class in visitor pattern	11
Car	
Abstract Product for Abstract Factory Pattern and Component for Decorator Pattern	14
CarFactory	
Abstract Factory for Abstract Factory Pattern	31
ConcreteBigBrother	
Concrete visitor class in visitor pattern	33
ConcreteMediator	37
ConcreteRaceManager	
Concrete Observer in observer pattern	40
ConcreteRegistrationManager	
ConcreteMediator for mediator design pattern	45
Driver	
Strategy for strategy design pattern	48
ElectricCar	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	50
ElectricFormulaOne	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	54
ElectricGoKart	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	56
ElectricRoadster	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	59
Facade	
Facade pattern	62
FinishDecorator	
ConcreteDecorator for Decorator design pattern	66
FlameVinyl	
Concrete Decorator for Decorator Pattern	68
FormulaOneFactory	
Concrete Factory for Abstract Factory Pattern	70
GoKartFactory	
Concrete Factory for Abstract Factory Pattern	72

LeftEighth	
Leaf for Composite design pattern	74
LeftPeelOff	
Leaf for Composite design pattern	77
LeftPeelOn	
Leaf for Composite design pattern	80
Manager	83
Mechanic	87
Mediator	91
Nitro	
Concrete Decorator for Decorator Pattern	93
PassiveDriver	96
PimpMyRide	
Decorator for Decorator Pattern	97
PimpMyTrack	
Abstract Decorator for Decorator design pattern	100
PitCrew	103
PitStop	108
PitStopDecorator	
ConcreteDecorator for Decorator design pattern	114
RaceManager	
Observer class for Observer pattern	116
RaceTrack	
Composite class for composite pattern	119
RaceTrackComponent	
Abstract leaf class for composite pattern	124
Racing	132
Ready	133
Refueller	133
RegistratcionManager	138
RightEighth	
Leaf for Composite design pattern	139
RightPeelOff	
Leaf for Composite design pattern	142
RightPeelOn	
Leaf for Composite design pattern	145
RoadsterFactory	
Concrete Factory for Abstract Factory Pattern	148
SandPitsDecorator	
ConcreteDecorator for Decorator design pattern	150
SkullVinyl	
Concrete Decorator for Decorator Pattern	152
Slick	
Concrete Decorator for Decorator Pattern	154
Spoiler	
Concrete Decorator for Decorator Pattern	157
SportsCar	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	159
SportsFormulaOne	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	162
SportsGoKart	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	165
SportsRoadster	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	168
StandardCar	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	171
StandardFormulaOne	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	175

StandardGoKart	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	177
StandardRoadster	
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern	180
StartDecorator	
ConcreteDecorator for Decorator design pattern	183
State	185
Stopped	187
Straight	
Leaf for Composite design pattern	187
Team	191
TyreChanger	192

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

AggressiveDriver.h	197
AverageDriver.h	??
BigBrother.h	198
Car.h	198
CarFactory.h	199
ConcreteBigBrother.h	200
ConcreteMediator.h	201
ConcreteRaceManager.h	202
ConcreteRegistrationManager.h	203
Driver.h	205
ElectricCar.h	205
ElectricFormulaOne.h	206
ElectricGokart.h	??
ElectricRoadster.h	208
Facade.h	209
FinishDecorator.h	210
FlameVinyl.h	211
ForumlaOneFactory.h	??
GoKartFactory.h	212
LeftEighth.h	213
LeftPeelOff.h	213
LeftPeelOn.h	214
Manager.h	215
Mechanic.h	216
Mediator.h	217
Nitro.h	218
PassiveDriver.h	??
PimpMyRide.h	219
PimpMyTrack.h	220
PitCrew.h	221
PitStop.h	222
PitStopDecorator.h	224
RaceManager.h	224
RaceTrack.h	225
RaceTrackComponent.h	226

Refueller.h	227
RegistrationManager.h	??
RightEighth.h	228
RightPeelOff.h	229
RightPeelOn.h	230
RoadsterFactory.h	231
SandPitsDecorator.h	232
SkullVinyl.h	233
Slick.h	234
Spoiler.h	235
SportsCar.h	236
SportsFormulaOne.h	237
SportsGoKart.h	239
SportsRoadster.h	240
StandardCar.h	241
StandardFormulaOne.h	242
StandardGoKart.h	243
StandardRoadster.h	244
StartDecorator.h	245
State.h	246
Straight.h	246
Team.h	247
TyreChanger.h	248

Chapter 4

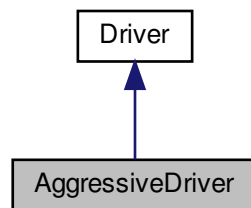
Class Documentation

4.1 AggressiveDriver Class Reference

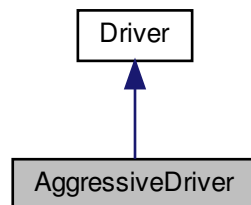
concreteStrategy for strategy design pattern

```
#include <AggressiveDriver.h>
```

Inheritance diagram for AggressiveDriver:



Collaboration diagram for AggressiveDriver:



Public Member Functions

- [AggressiveDriver](#) ()

4.1.1 Detailed Description

concreteStrategy for strategy design pattern

Authors

Duncan + Tjaart

Version

1.0.0

4.1.2 Constructor & Destructor Documentation

4.1.2.1 AggressiveDriver()

```
AggressiveDriver::AggressiveDriver ( ) [inline]
```

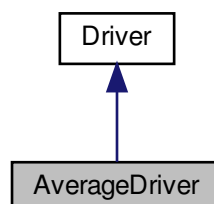
constructor to set fuel, tyre and driving ability

The documentation for this class was generated from the following file:

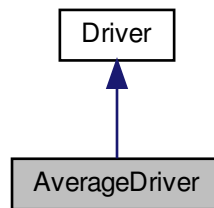
- [AggressiveDriver.h](#)

4.2 AverageDriver Class Reference

Inheritance diagram for AverageDriver:



Collaboration diagram for AverageDriver:



Public Member Functions

- [AverageDriver\(\)](#)

4.2.1 Constructor & Destructor Documentation

4.2.1.1 AverageDriver()

```
AverageDriver::AverageDriver ( ) [inline]
```

constructor to set fuel, tyre and driving ability

The documentation for this class was generated from the following file:

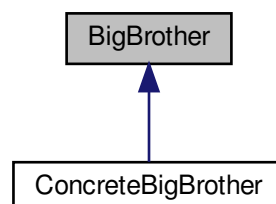
- AverageDriver.h

4.3 BigBrother Class Reference

visitor class in visitor pattern

```
#include <BigBrother.h>
```

Inheritance diagram for BigBrother:



Public Member Functions

- virtual void [visit](#) ([LeftEighth](#) *leftEighth)=0
- virtual void [visit](#) ([RightEighth](#) *rightEighth)=0
- virtual void [visit](#) ([LeftPeelOff](#) *leftPeelOff)=0
- virtual void [visit](#) ([RightPeelOff](#) *rightPeelOff)=0
- virtual void [visit](#) ([LeftPeelOn](#) *leftPeelOn)=0
- virtual void [visit](#) ([RightPeelOn](#) *rightPeelOn)=0
- virtual void [visit](#) ([Straight](#) *straight)=0

4.3.1 Detailed Description

visitor class in visitor pattern

Authors

Duncan + Tjaart

Version

1.0.0

4.3.2 Member Function Documentation

4.3.2.1 [visit\(\)](#) [1/7]

```
virtual void BigBrother::visit (
    LeftEighth * leftEighth ) [pure virtual]
```

virtual visit that visits Left Eighth object

Parameters

<i>leftEighth</i>	
-------------------	--

Implemented in [ConcreteBigBrother](#).

4.3.2.2 [visit\(\)](#) [2/7]

```
virtual void BigBrother::visit (
    RightEighth * rightEighth ) [pure virtual]
```

virtual visit that visits rightEighth object

Parameters

<i>rightEighth</i>	
--------------------	--

Implemented in [ConcreteBigBrother](#).

4.3.2.3 visit() [3/7]

```
virtual void BigBrother::visit (  
    LeftPeelOff * leftPeelOff ) [pure virtual]
```

virtual visit that visits leftPeelOff object

Parameters

<i>leftPeelOff</i>	
--------------------	--

Implemented in [ConcreteBigBrother](#).

4.3.2.4 visit() [4/7]

```
virtual void BigBrother::visit (  
    RightPeelOff * rightPeelOff ) [pure virtual]
```

virtual visit that visits rightPeelOff object

Parameters

<i>rightPeelOff</i>	
---------------------	--

Implemented in [ConcreteBigBrother](#).

4.3.2.5 visit() [5/7]

```
virtual void BigBrother::visit (  
    LeftPeelOn * leftPeelOn ) [pure virtual]
```

virtual visit that visits leftPeelOn object

Parameters

<i>leftPeelOn</i>	
-------------------	--

Implemented in [ConcreteBigBrother](#).

4.3.2.6 visit() [6/7]

```
virtual void BigBrother::visit (
    RightPeelOn * rightPeelOn ) [pure virtual]
```

virtual visit that visits rightPeelOn object

Parameters

<i>rightPeelOn</i>	
--------------------	--

Implemented in [ConcreteBigBrother](#).

4.3.2.7 visit() [7/7]

```
virtual void BigBrother::visit (
    Straight * straight ) [pure virtual]
```

virtual visit that visits straight object

Parameters

<i>straight</i>	
-----------------	--

Implemented in [ConcreteBigBrother](#).

The documentation for this class was generated from the following file:

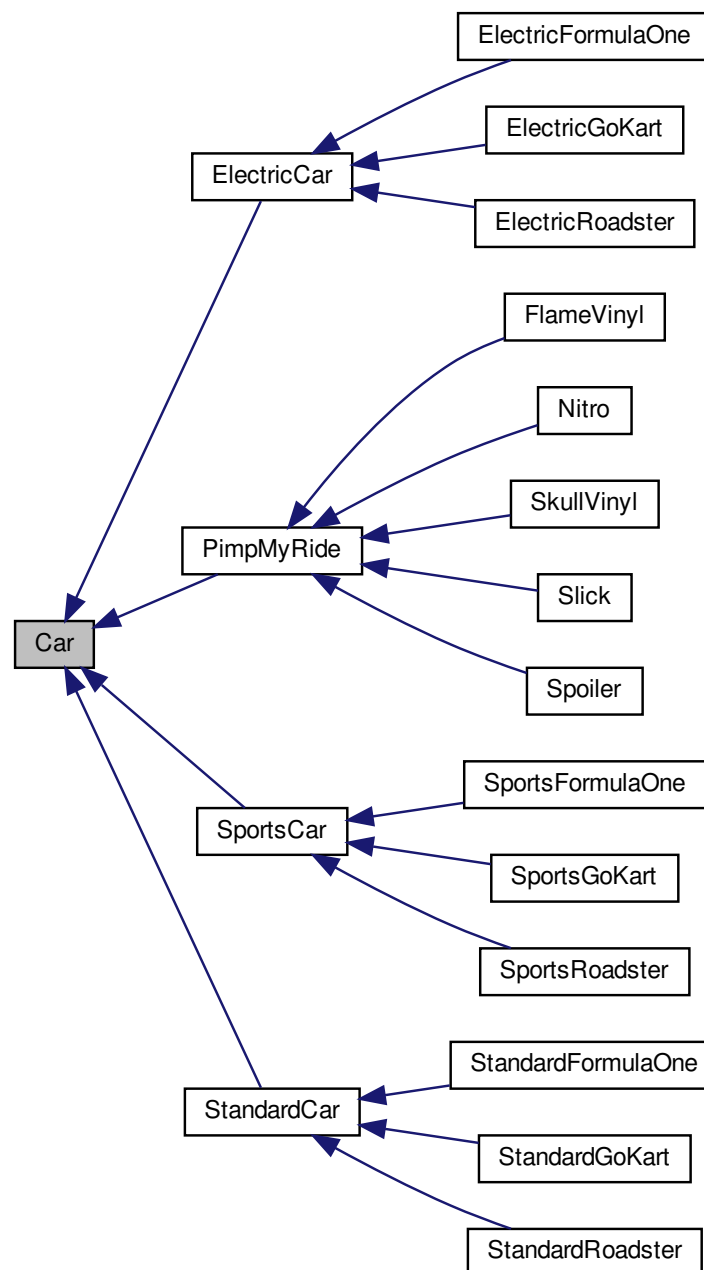
- [BigBrother.h](#)

4.4 Car Class Reference

Abstract Product for Abstract Factory Pattern and Component for Decorator Pattern.

```
#include <Car.h>
```

Inheritance diagram for Car:



Collaboration diagram for Car:



Public Member Functions

- [Car](#) (int tyres=4)
- [Car](#) (string modelType_, int tyres=4)
- [Car](#) (const [Car](#) &car_, bool flag_)
- virtual [~Car](#) ()
- virtual [Car](#) * [clone](#) (bool flag=false)=0
- virtual [Car](#) * [FullClone](#) ()=0
- virtual void [add](#) ([Car](#) *c)
- void [setDescription](#) (string des)
- string [getDescription](#) () const
- int [getModelNumber](#) () const
- virtual string [showCarStats](#) ()
- string [showCarCondition](#) ()
- void [setSpeed](#) (int speed)
- int [getSpeed](#) ()
- void [setHandling](#) (int H)
- int [getHandling](#) ()
- void [setAcceleration](#) (int A)
- int [getAcceleration](#) ()
- const string [getModelType](#) () const
- string [toString](#) ()
- int [getTrackTime](#) ()
- void [setTrackTime](#) (int i)
- int [getNumTyres](#) ()
- [PitCrew](#) * [getManager](#) ()
- void [setManager](#) ([PitCrew](#) *m)
- [PitStop](#) * [getTeam](#) ()
- void [setTeam](#) ([PitStop](#) *t)
- void [notifyTeam](#) ()
- int * [getCarTyres](#) ()
- int [getCarTyre](#) (int index)
- void [setCarTyre](#) (int index, int tyre)
- void [setChanged](#) (int index, int tyre)
- int [getCarFuel](#) ()
- void [setCarFuel](#) (int fuel)
- void [setRefuel](#) (int fuel)
- int [getCarDamage](#) ()
- void [setCarDamage](#) (int damage_)
- void [setRepair](#) (int damage_)
- int [getCarID](#) ()
- void [RegistrationNotify](#) (string msg)

- int [getLap](#) ()
- void [setLap](#) (int l)
- int [getTrackPart](#) ()
- void [setTrackPart](#) (int t)
- void [setState](#) ([State](#) *state)
- string [getState](#) ()
- void [ready](#) ()
- void [racing](#) ()
- void [stopped](#) ()
- void [setDriver](#) ([Driver](#) *driver1)
- [Driver](#) * [getDriver](#) ()

Public Attributes

- [Car](#) * [carDecorate](#)

4.4.1 Detailed Description

Abstract Product for Abstract Factory Pattern and Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.4.2 Constructor & Destructor Documentation

4.4.2.1 [Car](#)() [1/3]

```
Car::Car (
    int tyres = 4 )
```

Default constructor for [Car](#)

Parameters

<i>tyres</i>	- the amount of tyres the car has
--------------	-----------------------------------

4.4.2.2 Car() [2/3]

```
Car::Car (
    string modelType_,
    int tyres = 4 )
```

The base constructor for [Car](#)

Parameters

<i>modelType_</i>	- states whether the car is Electirc/Sports/Standard
<i>tyres</i>	- the amount of tyres the car has

4.4.2.3 Car() [3/3]

```
Car::Car (
    const Car & car_,
    bool flag_ )
```

The copy constructor for [Car](#)

Parameters

<i>car_</i>	- a Car object that will be copied
<i>flag_</i>	- bool var to decide if the base car or the whole car will be cloned

4.4.2.4 ~Car()

```
virtual Car::~~Car ( ) [inline], [virtual]
```

The virtual destructor for [Car](#)

4.4.3 Member Function Documentation

4.4.3.1 add()

```
void Car::add (
    Car * c ) [virtual]
```

abstract add function for decorator

Parameters

<i>c</i>	- A Car object
----------	--------------------------------

Reimplemented in [PimpMyRide](#).

4.4.3.2 clone()

```
virtual Car* Car::clone (
    bool flag = false ) [pure virtual]
```

Abstract clone function for the prototype design pattern

Parameters

<i>flag</i>	- bool var to decide if the base car or the whole car will be cloned
-------------	--

Returns

a [Car](#) object

Implemented in [SportsRoadster](#), [PimpMyRide](#), [SportsFormulaOne](#), [SportsGoKart](#), [StandardFormulaOne](#), [StandardGoKart](#), [StandardRoadster](#), [ElectricFormulaOne](#), [ElectricGoKart](#), [ElectricRoadster](#), [StandardCar](#), [ElectricCar](#), and [SportsCar](#).

4.4.3.3 FullClone()

```
virtual Car* Car::FullClone ( ) [pure virtual]
```

Abstract full clone implemented by decorator class to copy over the decorators

Returns

[Car](#) object

Implemented in [PimpMyRide](#), [Nitro](#), [Slick](#), [Spoiler](#), [StandardCar](#), [ElectricCar](#), [SportsCar](#), [FlameVinyl](#), and [SkullVinyl](#).

4.4.3.4 getAcceleration()

```
int Car::getAcceleration ( ) [inline]
```

Function to get Acceleration of a car

Returns

Acceleration

4.4.3.5 getCarDamage()

```
int Car::getCarDamage ( )
```

Get the damage the car has taken

Returns

the amount of damage since last repair

4.4.3.6 getCarFuel()

```
int Car::getCarFuel ( )
```

Get the fuel level of the car

Returns

int - the fuel level

4.4.3.7 getCarID()

```
int Car::getCarID ( ) [inline]
```

Get the ID of the car

Returns

returns the car ID

4.4.3.8 getCarTyre()

```
int Car::getCarTyre (
    int index )
```

Get the tyre with the specific index's condition

Parameters

<i>index</i>	- index for the tyre array
--------------	----------------------------

Returns

int - the condition of the tyre

4.4.3.9 getCarTyres()

```
int * Car::getCarTyres ( )
```

Get the condition of each tyre

Returns

int array showing the condition of each tyre

4.4.3.10 getDescription()

```
string Car::getDescription ( ) const
```

Get the description of the car

Returns

string

4.4.3.11 getDriver()

```
Driver* Car::getDriver ( ) [inline]
```

returns the driver of the car

Returns**4.4.3.12 getHandling()**

```
int Car::getHandling ( ) [inline]
```

Function to get Handling of a car

Returns

handling

4.4.3.13 getLap()

```
int Car::getLap ( ) [inline]
```

returns lapnumber

Returns

lapno

4.4.3.14 getManager()

```
PitCrew* Car::getManager ( ) [inline]
```

Gets the manager of the car

Returns

PitCrew pointer to the manager

4.4.3.15 getModelNumber()

```
int Car::getModelNumber ( ) const
```

Get the model number of the car

Returns

the car modelNumber

4.4.3.16 getModelType()

```
const string Car::getModelType ( ) const [inline]
```

function to get model type of a car

Returns

string (the model type)

4.4.3.17 getNumTyres()

```
int Car::getNumTyres ( )
```

function to get the amount of tyres the car have

Returns

the number of tyres

4.4.3.18 getSpeed()

```
int Car::getSpeed ( ) [inline]
```

Function to get speed of a car

Returns

speed - The speed the car has

4.4.3.19 getState()

```
string Car::getState ( )
```

Get the current state of the car

Returns

state

4.4.3.20 getTeam()

```
PitStop* Car::getTeam ( ) [inline]
```

Gets the team of the car

Returns

[PitStop](#) pointer to the team

4.4.3.21 getTrackPart()

```
int Car::getTrackPart ( ) [inline]
```

returns track part

Returns

trackPart

4.4.3.22 getTrackTime()

```
int Car::getTrackTime ( )
```

function to get the total time the car took in the race

Returns

int representing the total time

4.4.3.23 notifyTeam()

```
void Car::notifyTeam ( )
```

[Car](#) notifies the team that its variables has changed. [Car](#) will do this during a race

4.4.3.24 racing()

```
void Car::racing ( )
```

Change car into the racing state

4.4.3.25 ready()

```
void Car::ready ( )
```

Change car into the ready state

4.4.3.26 RegistrationNotify()

```
void Car::RegistrationNotify (
    string msg )
```

notifies the car which track it is registered for

Parameters

<i>msg</i>	the string to output
------------	----------------------

4.4.3.27 setAcceleration()

```
void Car::setAcceleration (
    int A ) [inline]
```

Function to set Acceleration of a car

Parameters

<i>A</i>	
----------	--

4.4.3.28 setCarDamage()

```
void Car::setCarDamage (
    int damage_ )
```

Set the damage of the car [Car](#) will then notify the team

Parameters

<i>damage</i>	- the new damage of the car
---------------	-----------------------------

4.4.3.29 setCarFuel()

```
void Car::setCarFuel (
    int fuel )
```

Set the fuel level of a car [Car](#) will notify the team

Parameters

<i>fuel</i>	- the new fuel level of the car
-------------	---------------------------------

4.4.3.30 setCarTyre()

```
void Car::setCarTyre (
    int index,
    int tyre )
```

Set the condition of the tyres after it has been changed [Car](#) will then notify the team

Parameters

<i>index</i>	- index for the tyre array
<i>tyre</i>	- the new condition of the tyre

4.4.3.31 setChanged()

```
void Car::setChanged (
    int index,
    int tyre )
```

Set the condition of the tyres after it has been changed [Car](#) will not notify the team

Parameters

<i>index</i>	- index for the tyre array
<i>tyre</i>	- the new condition of the tyre

4.4.3.32 setDescription()

```
void Car::setDescription (
    string des )
```

Set the description of the car

Parameters

<i>des</i>	string passed in
------------	------------------

4.4.3.33 setDriver()

```
void Car::setDriver (
    Driver * driver1 ) [inline]
```

sets the driver of the car

Parameters

<i>driver1</i>	
----------------	--

4.4.3.34 setHandling()

```
void Car::setHandling (
    int H ) [inline]
```

Function to set Handling of a car

Parameters

<i>H</i>	
----------	--

4.4.3.35 setLap()

```
void Car::setLap (
    int l ) [inline]
```

sets the lap number

Parameters

<i>l</i>	
----------	--

4.4.3.36 setManager()

```
void Car::setManager (
    PitCrew * m ) [inline]
```

Sets the manager of the car

Parameters

<i>m</i>	- PitCrew object which is the manager
----------	---

4.4.3.37 setRefuel()

```
void Car::setRefuel (
    int fuel )
```

Set the fuel level of a car [Car](#) will not notify the team

Parameters

<i>fuel</i>	- the new fuel level of the car
-------------	---------------------------------

4.4.3.38 setRepair()

```
void Car::setRepair (
    int damage_ )
```

Set the damage of the car [Car](#) will not notify the team

Parameters

<i>damage</i>	- the new damage of the car
---------------	-----------------------------

4.4.3.39 setSpeed()

```
void Car::setSpeed (
    int speed ) [inline]
```

Function to set speed of a car

Parameters

<i>speed</i>	- The speed the car has
--------------	-------------------------

4.4.3.40 setState()

```
void Car::setState (
    State * state )
```

Set the current state of the car

Parameters

<i>state</i>	- State object which it needs to be
--------------	---

4.4.3.41 `setTeam()`

```
void Car::setTeam (
    PitStop * t ) [inline]
```

Sets the team of the car

Parameters

<i>t</i>	- PitStop object which is the team
----------	--

4.4.3.42 `setTrackPart()`

```
void Car::setTrackPart (
    int t ) [inline]
```

sets track part of car

Parameters

<i>t</i>	
----------	--

4.4.3.43 `setTrackTime()`

```
void Car::setTrackTime (
    int i )
```

function to add time to the track time when the car is making a pit stop

Parameters

<i>i</i>	the amount of time to be added to the existing time
----------	---

4.4.3.44 showCarCondition()

```
string Car::showCarCondition ( )
```

Print the condition of a car during the race

Returns

string describing the condition

4.4.3.45 showCarStats()

```
string Car::showCarStats ( ) [virtual]
```

Abstract showCarStats function to show the stats of a car

Returns

string stating the stats

Reimplemented in [PimpMyRide](#).

4.4.3.46 stopped()

```
void Car::stopped ( )
```

Change car into the stopped state

4.4.3.47 toString()

```
string Car::toString ( )
```

function to return a full detail about the car

Returns

string of car details

4.4.4 Member Data Documentation

4.4.4.1 carDecorate

```
Car* Car::carDecorate
```

pointer to car object for decorator

The documentation for this class was generated from the following files:

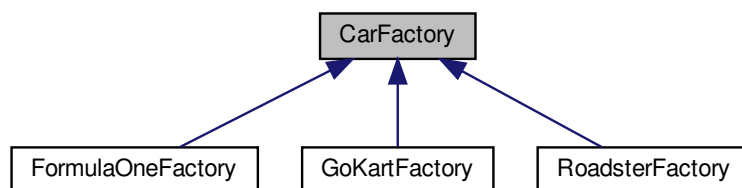
- [Car.h](#)
- [Car.cpp](#)

4.5 CarFactory Class Reference

Abstract Factory for Abstract Factory Pattern.

```
#include <CarFactory.h>
```

Inheritance diagram for CarFactory:



Public Member Functions

- virtual [ElectricCar](#) * [produceElectric](#) ()=0
- virtual [SportsCar](#) * [produceSports](#) ()=0
- virtual [StandardCar](#) * [produceStandard](#) ()=0

4.5.1 Detailed Description

Abstract Factory for Abstract Factory Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.5.2 Member Function Documentation

4.5.2.1 produceElectric()

```
virtual ElectricCar* CarFactory::produceElectric ( ) [pure virtual]
```

A Abstract Function to produce an [ElectricCar](#)

Returns

[ElectricCar](#)*

Implemented in [FormulaOneFactory](#), [GoKartFactory](#), and [RoadsterFactory](#).

4.5.2.2 produceSports()

```
virtual SportsCar* CarFactory::produceSports ( ) [pure virtual]
```

A Abstract Function to produce an [SportsCar](#)

Returns

[SportsCar](#)*

Implemented in [GoKartFactory](#), [FormulaOneFactory](#), and [RoadsterFactory](#).

4.5.2.3 produceStandard()

```
virtual StandardCar* CarFactory::produceStandard ( ) [pure virtual]
```

A Abstract Function to produce an [StandardCar](#)

Returns

[StandardCar](#)*

Implemented in [GoKartFactory](#), [FormulaOneFactory](#), and [RoadsterFactory](#).

The documentation for this class was generated from the following file:

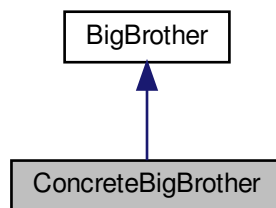
- [CarFactory.h](#)

4.6 ConcreteBigBrother Class Reference

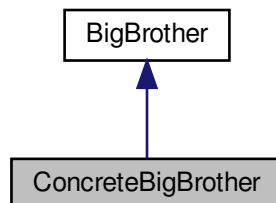
Concrete visitor class in visitor pattern.

```
#include <ConcreteBigBrother.h>
```

Inheritance diagram for ConcreteBigBrother:



Collaboration diagram for ConcreteBigBrother:



Public Member Functions

- virtual void [visit](#) ([LeftEighth](#) *leftEighth)
- virtual void [visit](#) ([RightEighth](#) *rightEighth)
- virtual void [visit](#) ([LeftPeelOff](#) *leftPeelOff)
- virtual void [visit](#) ([RightPeelOff](#) *rightPeelOff)
- virtual void [visit](#) ([LeftPeelOn](#) *leftPeelOn)
- virtual void [visit](#) ([RightPeelOn](#) *rightPeelOn)
- virtual void [visit](#) ([Straight](#) *straight)

4.6.1 Detailed Description

Concrete visitor class in visitor pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.6.2 Member Function Documentation

4.6.2.1 visit() [1/7]

```
virtual void ConcreteBigBrother::visit (  
    LeftEighth * leftEighth ) [inline], [virtual]
```

virtual visit that visits Left Eighth object

Parameters

<i>leftEighth</i>	
-------------------	--

Implements [BigBrother](#).

4.6.2.2 visit() [2/7]

```
virtual void ConcreteBigBrother::visit (  
    RightEighth * rightEighth ) [inline], [virtual]
```

virtual visit that visits rightEighth object

Parameters

<i>rightEighth</i>	
--------------------	--

Implements [BigBrother](#).

4.6.2.3 visit() [3/7]

```
virtual void ConcreteBigBrother::visit (
    LeftPeelOff * leftPeelOff ) [inline], [virtual]
```

virtual visit that visits leftPeelOff object

Parameters

<i>leftPeelOff</i>	
--------------------	--

Implements [BigBrother](#).

4.6.2.4 visit() [4/7]

```
virtual void ConcreteBigBrother::visit (
    RightPeelOff * rightPeelOff ) [inline], [virtual]
```

virtual visit that visits rightPeelOff object

Parameters

<i>rightPeelOff</i>	
---------------------	--

Implements [BigBrother](#).

4.6.2.5 visit() [5/7]

```
virtual void ConcreteBigBrother::visit (
    LeftPeelOn * leftPeelOn ) [inline], [virtual]
```

virtual visit that visits leftPeelOn object

Parameters

<i>leftPeelOn</i>	
-------------------	--

Implements [BigBrother](#).

4.6.2.6 visit() [6/7]

```
virtual void ConcreteBigBrother::visit (
    RightPeelOn * rightPeelOn ) [inline], [virtual]
```

virtual visit that visits rightPeelOn object

Parameters

<i>rightPeelOn</i>	
--------------------	--

Implements [BigBrother](#).

4.6.2.7 visit() [7/7]

```
virtual void ConcreteBigBrother::visit (  
    Straight * straight ) [inline], [virtual]
```

virtual visit that visits straight object

Parameters

<i>straight</i>	
-----------------	--

Implements [BigBrother](#).

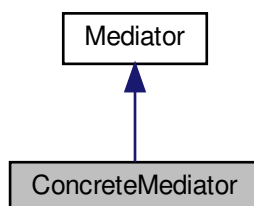
The documentation for this class was generated from the following file:

- [ConcreteBigBrother.h](#)

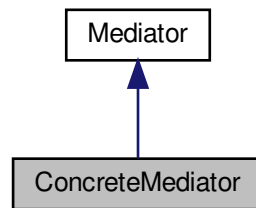
4.7 ConcreteMediator Class Reference

```
#include <ConcreteMediator.h>
```

Inheritance diagram for ConcreteMediator:



Collaboration diagram for ConcreteMediator:



Public Member Functions

- [~ConcreteMediator](#) ()
- virtual void [addMember](#) ([PitCrew](#) *member)
- virtual void [notify](#) ([PitCrew](#) *member)
- virtual void [notifyManager](#) ([PitCrew](#) *member)
- [PitCrew](#) * [getManager](#) ()

4.7.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.7.2 Constructor & Destructor Documentation

4.7.2.1 ~ConcreteMediator()

```
ConcreteMediator::~~ConcreteMediator ( )
```

Destructor for the [ConcreteMediator](#)

4.7.3 Member Function Documentation

4.7.3.1 addMember()

```
void ConcreteMediator::addMember (
    PitCrew * member ) [virtual]
```

Add a member of the pitcrew to the mediator to talk to other members

Parameters

<i>member</i>	
---------------	--

Implements [Mediator](#).

4.7.3.2 getManager()

```
PitCrew* ConcreteMediator::getManager ( ) [inline]
```

Get the manager of the pitcrew

Returns

4.7.3.3 notify()

```
void ConcreteMediator::notify (
    PitCrew * member ) [virtual]
```

Notify the team that the car has changed

Parameters

<i>member</i>	
---------------	--

Implements [Mediator](#).

4.7.3.4 notifyManager()

```
void ConcreteMediator::notifyManager (
    PitCrew * member ) [virtual]
```

Notify the manager that the car has changed

Parameters

<i>member</i>	
---------------	--

Implements [Mediator](#).

The documentation for this class was generated from the following files:

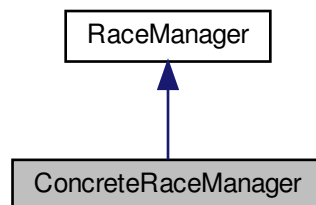
- [ConcreteMediator.h](#)
- ConcreteMediator.cpp

4.8 ConcreteRaceManager Class Reference

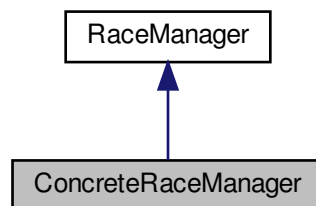
concrete Observer in observer pattern

```
#include <ConcreteRaceManager.h>
```

Inheritance diagram for ConcreteRaceManager:



Collaboration diagram for ConcreteRaceManager:



Public Member Functions

- virtual void [readyRace](#) ()
- virtual void [startRace](#) ()
- virtual void [stopRace](#) ()
- virtual void [pauseRace](#) (int numComponent)
- virtual void [resumeRace](#) (int numComponent)

- virtual void [printLeaderBoard](#) ()
- string [getCarInfo](#) ([Car](#) *_car)
- virtual void [addCars](#) (vector< [Car](#) *> _cars)
- virtual void [addRacetrack](#) ([RaceTrackComponent](#) *raceTrackComponent)
- void [setLapMax](#) (int i)
- int [getLapMax](#) ()
- void [setLapCount](#) (int i)
- int [getLap](#) ()

4.8.1 Detailed Description

concrete Observer in observer pattern

Authors

Duncan + Tjaart

Version

1.0.0

4.8.2 Member Function Documentation

4.8.2.1 addCars()

```
virtual void ConcreteRaceManager::addCars (
    vector< Car *> _cars ) [inline], [virtual]
```

add cars to the race manager

Parameters

_cars	
-----------------------	--

Implements [RaceManager](#).

4.8.2.2 addRacetrack()

```
virtual void ConcreteRaceManager::addRacetrack (
    RaceTrackComponent * raceTrackComponent ) [inline], [virtual]
```

adds the race

Parameters

<i>raceTrackComponent</i>	
---------------------------	--

Implements [RaceManager](#).

4.8.2.3 getCarInfo()

```
string ConcreteRaceManager::getCarInfo (
    Car * _car ) [inline]
```

gets the car info and returns a string

Parameters

<i>_car</i>	
-------------	--

Returns**4.8.2.4 getLap()**

```
int ConcreteRaceManager::getLap ( ) [inline]
```

returns the lap

Returns**4.8.2.5 getLapMax()**

```
int ConcreteRaceManager::getLapMax ( ) [inline]
```

returns the max laps

Returns**4.8.2.6 pauseRace()**

```
virtual void ConcreteRaceManager::pauseRace (
    int numComponent ) [inline], [virtual]
```

pauses the race

Parameters

<i>numComponent</i>	
---------------------	--

Implements [RaceManager](#).

4.8.2.7 printLeaderBoard()

```
virtual void ConcreteRaceManager::printLeaderBoard ( ) [inline], [virtual]
```

prints the cars in order according to track times

Implements [RaceManager](#).

4.8.2.8 readyRace()

```
virtual void ConcreteRaceManager::readyRace ( ) [inline], [virtual]
```

Moves all cars to starting point of track and sets the times to 0

Implements [RaceManager](#).

4.8.2.9 resumeRace()

```
virtual void ConcreteRaceManager::resumeRace (
    int numComponent ) [inline], [virtual]
```

resumes the race according to where it left

Parameters

<i>numComponent</i>	
---------------------	--

Implements [RaceManager](#).

4.8.2.10 setLapCount()

```
void ConcreteRaceManager::setLapCount (
    int i ) [inline]
```

sets the lap currently on

Parameters

<i>i</i>	
----------	--

4.8.2.11 setLapMax()

```
void ConcreteRaceManager::setLapMax (
    int i ) [inline]
```

sets the max amount of laps

Parameters

<i>i</i>	
----------	--

4.8.2.12 startRace()

```
virtual void ConcreteRaceManager::startRace ( ) [inline], [virtual]
```

starts to move the cars along the racetrack

Implements [RaceManager](#).

4.8.2.13 stopRace()

```
virtual void ConcreteRaceManager::stopRace ( ) [inline], [virtual]
```

announces when the race is finished and prints the final leaderboard

Implements [RaceManager](#).

The documentation for this class was generated from the following file:

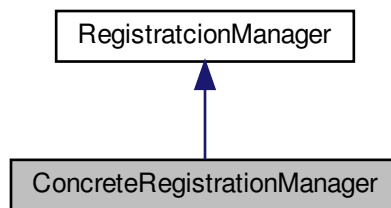
- [ConcreteRaceManager.h](#)

4.9 ConcreteRegistrationManager Class Reference

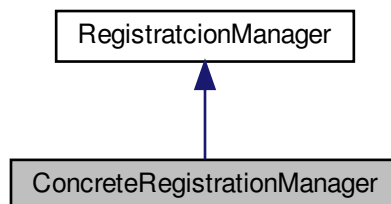
[ConcreteMediator](#) for mediator design pattern.

```
#include <ConcreteRegistrationManager.h>
```

Inheritance diagram for ConcreteRegistrationManager:



Collaboration diagram for ConcreteRegistrationManager:



Public Member Functions

- [ConcreteRegistrationManager](#) ()
- [~ConcreteRegistrationManager](#) ()
- virtual void [addCar](#) ([Car](#) * _car, int track)
- virtual void [addTrack](#) ([RaceTrackComponent](#) * _racetrack)
- virtual vector< [Car](#) * > [getCars](#) (int racetrack)
- virtual [RaceTrackComponent](#) * [getTrack](#) (int trackNo)

4.9.1 Detailed Description

[ConcreteMediator](#) for mediator design pattern.

[AbstractMediator](#) for mediator design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.9.2 Constructor & Destructor Documentation

4.9.2.1 ConcreteRegistrationManager()

```
ConcreteRegistrationManager::ConcreteRegistrationManager ( )
```

constructor for [ConcreteRegistrationManager](#)

Parameters

<code>_numTracks</code>	determines how many racetracks there are
-------------------------	--

4.9.2.2 ~ConcreteRegistrationManager()

```
ConcreteRegistrationManager::~~ConcreteRegistrationManager ( )
```

destructor for manager

4.9.3 Member Function Documentation

4.9.3.1 addCar()

```
void ConcreteRegistrationManager::addCar (
    Car * _car,
    int track ) [virtual]
```

implementation of absrtact function to add car into the cars array

Parameters

<i>_car</i>	car object to be placed in the array
<i>track</i>	specifies which track the car will be racing

Implements [RegistratcionManager](#).

4.9.3.2 addTrack()

```
void ConcreteRegistrationManager::addTrack (
    RaceTrackComponent * _racetrack ) [virtual]
```

implementation of absrtact function to add car into the cars array

Parameters

<i>_racetrack</i>	racetrack object to be placed in the array
-------------------	--

Implements [RegistratcionManager](#).

4.9.3.3 getCars()

```
virtual vector<Car*> ConcreteRegistrationManager::getCars (
    int racetrack ) [inline], [virtual]
```

returns the cars for a given racetrack

Parameters

<i>racetrack</i>	
------------------	--

Returns

vector of cars

Implements [RegistratcionManager](#).

4.9.3.4 getTrack()

```
virtual RaceTrackComponent* ConcreteRegistrationManager::getTrack (
    int trackNo ) [inline], [virtual]
```

returns a racetrack given a racetrack number

Parameters

<i>trackNo</i>	
----------------	--

Returns

a racetrack number

Implements [RegistratcionManager](#).

The documentation for this class was generated from the following files:

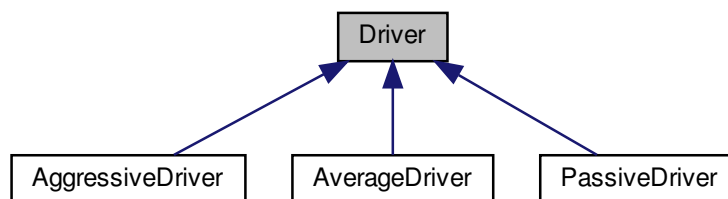
- [ConcreteRegistrationManager.h](#)
- [ConcreteRegistrationManager.cpp](#)

4.10 Driver Class Reference

Strategy for strategy design pattern.

```
#include <Driver.h>
```

Inheritance diagram for Driver:

**Public Member Functions**

- void [setAbility](#) (double ability)
- double [getDriverAbilty](#) ()
- void [setFuelAbility](#) (double ability)
- double [getFuelAbilty](#) ()
- void [setTyreAbility](#) (double ability)
- double [getTyreAbilty](#) ()

4.10.1 Detailed Description

Stratey for strategy design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.10.2 Member Function Documentation

4.10.2.1 `getDriverAbilty()`

```
double Driver::getDriverAbilty ( ) [inline]
```

returns the driving ability of the driver

Returns

4.10.2.2 `getFuelAbilty()`

```
double Driver::getFuelAbilty ( ) [inline]
```

returns the fuel ability of the driver

Returns

4.10.2.3 `getTyreAbilty()`

```
double Driver::getTyreAbilty ( ) [inline]
```

returns the tyre ability of the driver

Returns

4.10.2.4 `setAbility()`

```
void Driver::setAbility (
    double ability ) [inline]
```

sets the driving ability of the driver

Parameters

<i>ability</i>	
----------------	--

4.10.2.5 setFuelAbility()

```
void Driver::setFuelAbility (
    double ability ) [inline]
```

sets the fuel ability of the driver

Parameters

<i>ability</i>	
----------------	--

4.10.2.6 setTyreAbility()

```
void Driver::setTyreAbility (
    double ability ) [inline]
```

returns the tyre ability of the driver

Parameters

<i>ability</i>	
----------------	--

The documentation for this class was generated from the following file:

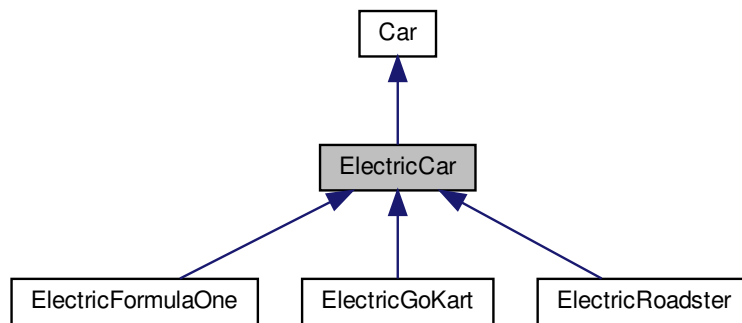
- [Driver.h](#)

4.11 ElectricCar Class Reference

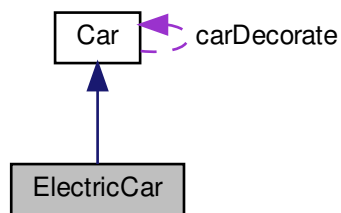
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <ElectricCar.h>
```

Inheritance diagram for ElectricCar:



Collaboration diagram for ElectricCar:



Public Member Functions

- [ElectricCar](#) (string modelType_)
- [ElectricCar](#) (const [Car](#) &car_, bool flag_)
- virtual [~ElectricCar](#) ()
- virtual string [getDescription](#) ()
- virtual [Car](#) * [clone](#) (bool flag_)
- virtual [Car](#) * [FullClone](#) ()

Additional Inherited Members

4.11.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.11.2 Constructor & Destructor Documentation

4.11.2.1 ElectricCar() [1/2]

```
ElectricCar::ElectricCar (
    string modelType_ )
```

Constructor for [ElectricCar](#)

Parameters

<i>modelType_</i>	states whether car is Electric/Sports/Standard
-------------------	--

4.11.2.2 ElectricCar() [2/2]

```
ElectricCar::ElectricCar (
    const Car & car_,
    bool flag_ )
```

The copy constructor for [ElectricCar](#)

Parameters

<i>car_</i>	is a Car object that will be copied
-------------	---

4.11.2.3 ~ElectricCar()

```
virtual ElectricCar::~~ElectricCar ( ) [inline], [virtual]
```

The virtual destructor for [ElectricCar](#)

4.11.3 Member Function Documentation

4.11.3.1 clone()

```
virtual Car* ElectricCar::clone (
    bool flag_ ) [inline], [virtual]
```

clone function for the prototype design pattern

Returns

a pointer to car object

Implements [Car](#).

Reimplemented in [ElectricFormulaOne](#), [ElectricGoKart](#), and [ElectricRoadster](#).

4.11.3.2 FullClone()

```
virtual Car* ElectricCar::FullClone ( ) [inline], [virtual]
```

implementation of Fullclone in [Car](#)

Returns

[Car](#) object with all decorated

Implements [Car](#).

4.11.3.3 getDescription()

```
string ElectricCar::getDescription ( ) [virtual]
```

a getDescription Function

Returns

a string that states the info about the car

The documentation for this class was generated from the following files:

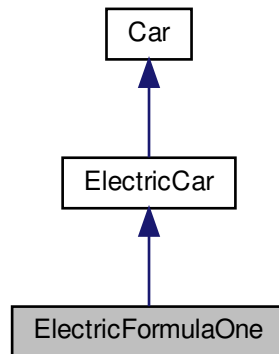
- [ElectricCar.h](#)
- [ElectricCar.cpp](#)

4.12 ElectricFormulaOne Class Reference

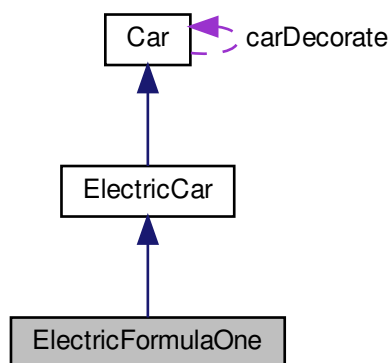
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <ElectricFormulaOne.h>
```

Inheritance diagram for ElectricFormulaOne:



Collaboration diagram for ElectricFormulaOne:



Public Member Functions

- [ElectricFormulaOne](#) ()
- [ElectricFormulaOne](#) (const [Car](#) &car_, bool flag_)
- virtual [Car](#) * [clone](#) (bool flag_=false)

Additional Inherited Members

4.12.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.12.2 Constructor & Destructor Documentation

4.12.2.1 ElectricFormulaOne() [1/2]

```
ElectricFormulaOne::ElectricFormulaOne ( ) [inline]
```

Constructor for [ElectricFormulaOne](#), calls Constructor of [ElectricCar](#)

4.12.2.2 ElectricFormulaOne() [2/2]

```
ElectricFormulaOne::ElectricFormulaOne (
    const Car & car_,
    bool flag_ ) [inline]
```

Copy constructor used for cloning

Parameters

<i>car</i> ↔ —	car object for copying
<i>flag</i> ↔ —	to determine if must be full clone or basic clone

4.12.3 Member Function Documentation

4.12.3.1 clone()

```
virtual Car* ElectricFormulaOne::clone (
    bool flag_ = false ) [inline], [virtual]
```

implementation of clone function

Parameters

<i>flag</i> ↔	determines if must be full clone or basic clone
—	

Returns

a copied car object

Reimplemented from [ElectricCar](#).

The documentation for this class was generated from the following file:

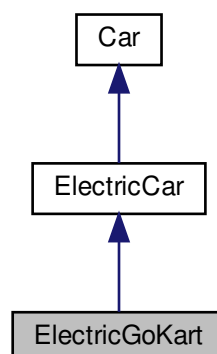
- [ElectricFormulaOne.h](#)

4.13 ElectricGoKart Class Reference

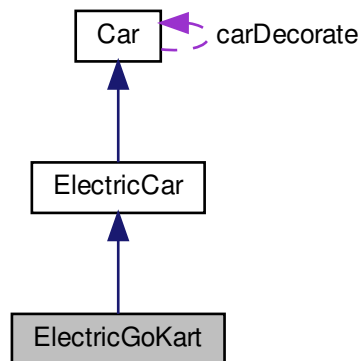
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <ElectricGokart.h>
```

Inheritance diagram for ElectricGoKart:



Collaboration diagram for ElectricGoKart:



Public Member Functions

- [ElectricGoKart](#) ()
- [ElectricGoKart](#) (const [Car](#) &car_, bool flag_)
- virtual [Car](#) * [clone](#) (bool flag_=false)

Additional Inherited Members

4.13.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.13.2 Constructor & Destructor Documentation

4.13.2.1 [ElectricGoKart](#)() [1/2]

```
ElectricGoKart::ElectricGoKart ( ) [inline]
```

Constructor for [ElectricGoKart](#), calls Constructor of [ElectricCar](#)

4.13.2.2 ElectricGoKart() [2/2]

```
ElectricGoKart::ElectricGoKart (
    const Car & car_,
    bool flag_ ) [inline]
```

Copy constructor used for cloning

Parameters

<i>car</i> ↔ —	car object for copying
<i>flag</i> ↔ —	to determine if must be full clone or basic clone

4.13.3 Member Function Documentation

4.13.3.1 clone()

```
virtual Car* ElectricGoKart::clone (
    bool flag_ = false ) [inline], [virtual]
```

implementation of clone function

Parameters

<i>flag</i> ↔ —	determines if must be full clone or basic clone
--------------------	---

Returns

a copied car object

Reimplemented from [ElectricCar](#).

The documentation for this class was generated from the following file:

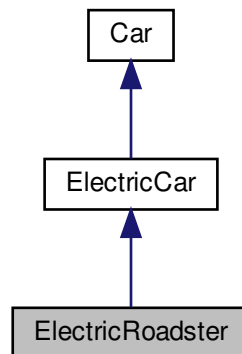
- [ElectricGokart.h](#)

4.14 ElectricRoadster Class Reference

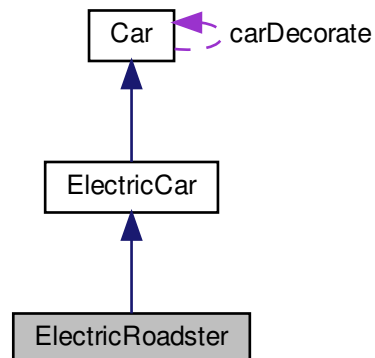
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <ElectricRoadster.h>
```

Inheritance diagram for ElectricRoadster:



Collaboration diagram for ElectricRoadster:



Public Member Functions

- [ElectricRoadster](#) ()
- [ElectricRoadster](#) (const [Car](#) &car_, bool flag_)
- virtual [Car](#) * [clone](#) (bool flag_=false)

Additional Inherited Members

4.14.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.14.2 Constructor & Destructor Documentation

4.14.2.1 ElectricRoadster() [1/2]

```
ElectricRoadster::ElectricRoadster ( ) [inline]
```

Constructor for [ElectricFormulaOne](#), calls Constructor of [ElectricCar](#)

4.14.2.2 ElectricRoadster() [2/2]

```
ElectricRoadster::ElectricRoadster (
    const Car & car_,
    bool flag_ ) [inline]
```

Copy constructor used for cloning

Parameters

<i>car</i> ↔ —	car object for copying
<i>flag</i> ↔ —	to determine if must be full clone or basic clone

4.14.3 Member Function Documentation

4.14.3.1 clone()

```
virtual Car* ElectricRoadster::clone (
    bool flag_ = false ) [inline], [virtual]
```

implementation of clone function

Parameters

<i>flag</i> ↔	determines if must be full clone or basic clone
—	

Returns

a copied car object

Reimplemented from [ElectricCar](#).

The documentation for this class was generated from the following file:

- [ElectricRoadster.h](#)

4.15 Facade Class Reference

[Facade](#) pattern.

```
#include <Facade.h>
```

Public Member Functions

- [Facade](#) ()
- [~Facade](#) ()
- [PitStop](#) * [createTeam](#) ()
- [Car](#) * [createCustomCar](#) ()
- [RaceTrackComponent](#) * [createCustomRaceTrack](#) ()
- void [registerCar](#) ([Car](#) *c)
- void [registerCar](#) ()
- void [registerTrack](#) ([RaceTrackComponent](#) *rt)
- bool [prepRace](#) ()
- void [StartRace](#) ()
- [Driver](#) * [createDriver](#) ()
- [Car](#) * [copyCar](#) ()

4.15.1 Detailed Description

[Facade](#) pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.15.2 Constructor & Destructor Documentation

4.15.2.1 Facade()

```
Facade::Facade ( )
```

constructor that creates the necessary registration objects

4.15.2.2 ~Facade()

```
Facade::~~Facade ( )
```

destructor to delete all necessary things

4.15.3 Member Function Documentation

4.15.3.1 copyCar()

```
Car * Facade::copyCar ( )
```

function to clone a car

Returns

the cloned car

4.15.3.2 createCustomCar()

```
Car * Facade::createCustomCar ( )
```

create a custom car, will ask all options

Returns

the created car

4.15.3.3 createCustomRaceTrack()

```
RaceTrackComponent * Facade::createCustomRaceTrack ( )
```

create a custom track

Returns

the custom track

4.15.3.4 createDriver()

```
Driver * Facade::createDriver ( )
```

create a driver, asking which driver the user wants

Returns

the driver

4.15.3.5 createTeam()

```
PitStop * Facade::createTeam ( )
```

function to create a new team

Returns

a pitstop object

4.15.3.6 prepRace()

```
bool Facade::prepRace ( )
```

prepare the race by getting all the necessary info from the registration manager

Parameters

<i>rt</i>	
-----------	--

Returns

returns if you chose to go back

4.15.3.7 registerCar() [1/2]

```
void Facade::registerCar (
    Car * c )
```

register car to the track with the registration manager

Parameters

<i>c</i>	the car to add
<i>i</i>	the track number

4.15.3.8 registerCar() [2/2]

```
void Facade::registerCar ( )
```

overloaded parameter to state which car you want to register

4.15.3.9 registerTrack()

```
void Facade::registerTrack (
    RaceTrackComponent * rt )
```

register the track with registration manager

Parameters

<i>rt</i>	
-----------	--

4.15.3.10 StartRace()

```
void Facade::StartRace ( )
```

will start the race

The documentation for this class was generated from the following files:

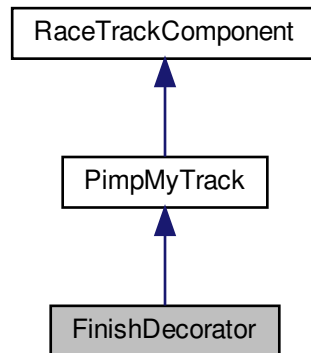
- [Facade.h](#)
- [Facade.cpp](#)

4.16 FinishDecorator Class Reference

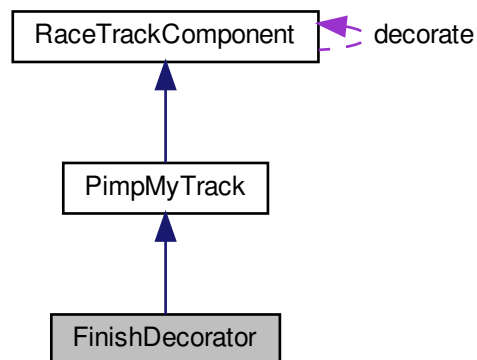
ConcreteDecorator for Decorator design pattern.

```
#include <FinishDecorator.h>
```

Inheritance diagram for FinishDecorator:



Collaboration diagram for FinishDecorator:



Public Member Functions

- [FinishDecorator](#) ()
- [~FinishDecorator](#) ()

Additional Inherited Members

4.16.1 Detailed Description

ConcreteDecorator for Decorator design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.16.2 Constructor & Destructor Documentation

4.16.2.1 FinishDecorator()

```
FinishDecorator::FinishDecorator ( ) [inline]
```

Constructor that calls constructor of pimpMyTrack and has a description

4.16.2.2 ~FinishDecorator()

```
FinishDecorator::~~FinishDecorator ( ) [inline]
```

destructor

The documentation for this class was generated from the following file:

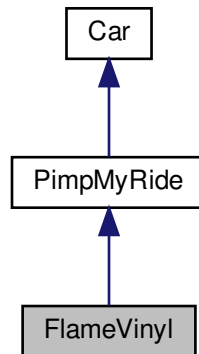
- [FinishDecorator.h](#)

4.17 FlameVinyl Class Reference

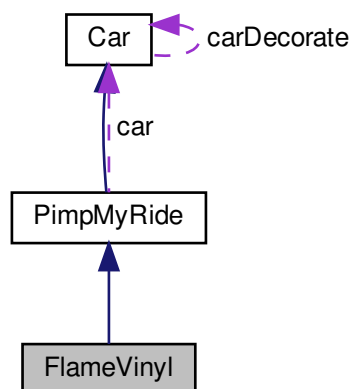
Concrete Decorator for Decorator Pattern.

```
#include <FlameVinyl.h>
```

Inheritance diagram for FlameVinyl:



Collaboration diagram for FlameVinyl:



Public Member Functions

- [FlameVinyl](#) ()
- [~FlameVinyl](#) ()
- [FlameVinyl](#) ([FlameVinyl](#) _Car, bool dummy)
- virtual [Car](#) * [FullClone](#) ()

Additional Inherited Members

4.17.1 Detailed Description

Concrete Decorator for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.17.2 Constructor & Destructor Documentation

4.17.2.1 FlameVinyl() [1/2]

```
FlameVinyl::FlameVinyl ( ) [inline]
```

constructor to set description

4.17.2.2 ~FlameVinyl()

```
FlameVinyl::~~FlameVinyl ( ) [inline]
```

destructor to delete the vinyl

4.17.2.3 FlameVinyl() [2/2]

```
FlameVinyl::FlameVinyl (
    FlameVinyl _Car,
    bool dummy ) [inline]
```

copy constructor used for cloning the decorators

Parameters

<i>_Car</i>	the car it copies
<i>dummy</i>	just there to use instead of default constructor

4.17.3 Member Function Documentation

4.17.3.1 FullClone()

```
virtual Car* FlameVinyl::FullClone ( ) [inline], [virtual]
```

implementation of FullClone to deep copy the decorator

Returns

[Car](#) object which is the decorator

Reimplemented from [PimpMyRide](#).

The documentation for this class was generated from the following file:

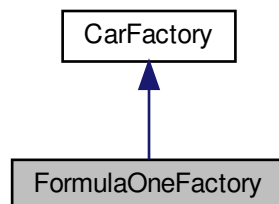
- [FlameVinyl.h](#)

4.18 FormulaOneFactory Class Reference

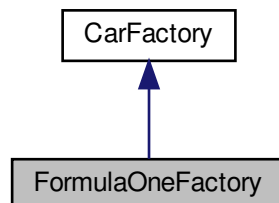
Concrete Factory for Abstract Factory Pattern.

```
#include <FormulaOneFactory.h>
```

Inheritance diagram for FormulaOneFactory:



Collaboration diagram for FormulaOneFactory:



Public Member Functions

- virtual [ElectricCar](#) * [produceElectric](#) ()
- virtual [SportsCar](#) * [produceSports](#) ()
- virtual [StandardCar](#) * [produceStandard](#) ()

4.18.1 Detailed Description

Concrete Factory for Abstract Factory Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.18.2 Member Function Documentation

4.18.2.1 [produceElectric\(\)](#)

```
virtual ElectricCar* FormulaOneFactory::produceElectric ( ) [inline], [virtual]
```

Implemented Function to produce an [ElectricCar](#)

Returns

[ElectricCar](#)*

Implements [CarFactory](#).

4.18.2.2 [produceSports\(\)](#)

```
virtual SportsCar* FormulaOneFactory::produceSports ( ) [inline], [virtual]
```

Implemented Function to produce an [SportsCar](#)

Returns

[SportsCar](#)*

Implements [CarFactory](#).

4.18.2.3 produceStandard()

```
virtual StandardCar* FormulaOneFactory::produceStandard ( ) [inline], [virtual]
```

Implemented Function to produce an [StandardCar](#)

Returns

[StandardCar](#)*

Implements [CarFactory](#).

The documentation for this class was generated from the following file:

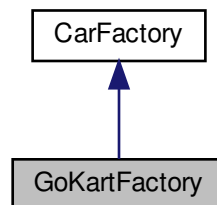
- ForumlaOneFactory.h

4.19 GoKartFactory Class Reference

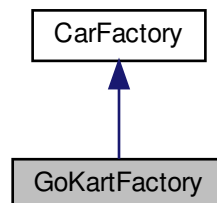
Concrete Factory for Abstract Factory Pattern.

```
#include <GoKartFactory.h>
```

Inheritance diagram for GoKartFactory:



Collaboration diagram for GoKartFactory:



Public Member Functions

- virtual [ElectricCar](#) * [produceElectric](#) ()
- virtual [SportsCar](#) * [produceSports](#) ()
- virtual [StandardCar](#) * [produceStandard](#) ()

4.19.1 Detailed Description

Concrete Factory for Abstract Factory Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.19.2 Member Function Documentation

4.19.2.1 [produceElectric\(\)](#)

```
virtual ElectricCar* GoKartFactory::produceElectric ( ) [inline], [virtual]
```

Implemented Function to produce an [ElectricCar](#)

Returns

[ElectricCar](#)*

Implements [CarFactory](#).

4.19.2.2 [produceSports\(\)](#)

```
virtual SportsCar* GoKartFactory::produceSports ( ) [inline], [virtual]
```

Implemented Function to produce an [SportsCar](#)

Returns

[SportsCar](#)*

Implements [CarFactory](#).

4.19.2.3 produceStandard()

```
virtual StandardCar* GoKartFactory::produceStandard ( ) [inline], [virtual]
```

Implemented Function to produce an [StandardCar](#)

Returns

[StandardCar](#)*

Implements [CarFactory](#).

The documentation for this class was generated from the following file:

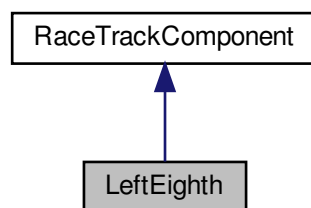
- [GoKartFactory.h](#)

4.20 LeftEighth Class Reference

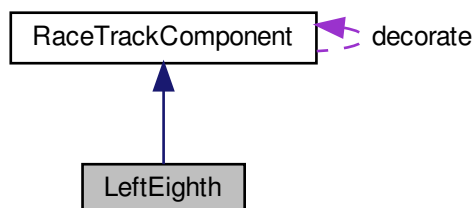
Leaf for Composite design pattern.

```
#include <LeftEighth.h>
```

Inheritance diagram for LeftEighth:



Collaboration diagram for LeftEighth:



Public Member Functions

- [LeftEighth](#) ()
- virtual [~LeftEighth](#) ()
- virtual void [add](#) ([RaceTrackComponent](#) *R)
- virtual void [print](#) ()
- int [getAverageTime](#) ()
- virtual void [accept](#) ([BigBrother](#) *v)
- virtual void [addTime](#) ()

Additional Inherited Members

4.20.1 Detailed Description

Leaf for Composite design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.20.2 Constructor & Destructor Documentation

4.20.2.1 [LeftEighth](#)()

```
LeftEighth::LeftEighth ( ) [inline]
```

constructor calls [RaceTrackComponent](#) and sets description

4.20.2.2 [~LeftEighth](#)()

```
virtual LeftEighth::~~LeftEighth ( ) [inline], [virtual]
```

destructor

4.20.3 Member Function Documentation

4.20.3.1 [accept](#)()

```
virtual void LeftEighth::accept (
    BigBrother * v ) [inline], [virtual]
```

accepts the visitor to go to the correct part of the visitor

Parameters

<i>V</i>	
----------	--

Implements [RaceTrackComponent](#).

4.20.3.2 add()

```
virtual void LeftEighth::add (  
    RaceTrackComponent * R ) [inline], [virtual]
```

implentation of add function, used by decorator

Parameters

<i>R</i>	
----------	--

Implements [RaceTrackComponent](#).

4.20.3.3 addTime()

```
virtual void LeftEighth::addTime ( ) [inline], [virtual]
```

adds the time and fuel and tyre conditions to the car

Implements [RaceTrackComponent](#).

4.20.3.4 getAverageTime()

```
int LeftEighth::getAverageTime ( ) [inline]
```

returns the average time for the track

Returns

4.20.3.5 print()

```
virtual void LeftEighth::print ( ) [inline], [virtual]
```

prints the description of the race track component /with decorators if it has

Implements [RaceTrackComponent](#).

The documentation for this class was generated from the following file:

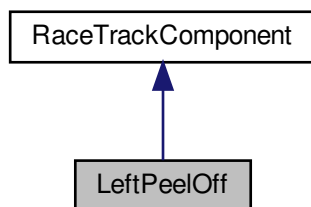
- [LeftEighth.h](#)

4.21 LeftPeelOff Class Reference

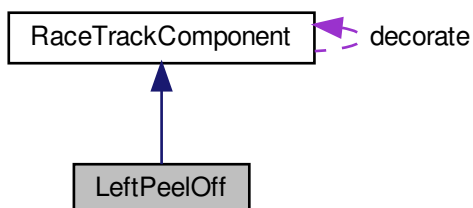
Leaf for Composite design pattern.

```
#include <LeftPeelOff.h>
```

Inheritance diagram for LeftPeelOff:



Collaboration diagram for LeftPeelOff:



Public Member Functions

- [LeftPeelOff](#) ()
- virtual [~LeftPeelOff](#) ()
- virtual void [add](#) ([RaceTrackComponent](#) *R)
- virtual void [print](#) ()
- int [getAverageTime](#) ()
- virtual void [accept](#) ([BigBrother](#) *v)
- virtual void [addTime](#) ()

Additional Inherited Members

4.21.1 Detailed Description

Leaf for Composite design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.21.2 Constructor & Destructor Documentation

4.21.2.1 [LeftPeelOff](#)()

```
LeftPeelOff::LeftPeelOff ( ) [inline]
```

constructor calls [RaceTrackComponent](#) and sets description

4.21.2.2 [~LeftPeelOff](#)()

```
virtual LeftPeelOff::~~LeftPeelOff ( ) [inline], [virtual]
```

destructor

4.21.3 Member Function Documentation

4.21.3.1 [accept](#)()

```
virtual void LeftPeelOff::accept (  
    BigBrother * v ) [inline], [virtual]
```

accepts the visitor to go to the correct part of the visitor

Parameters

V	
-----	--

Implements [RaceTrackComponent](#).

4.21.3.2 add()

```
virtual void LeftPeelOff::add (  
    RaceTrackComponent * R ) [inline], [virtual]
```

implentation of add function, used by decorator

Parameters

R	
-----	--

Implements [RaceTrackComponent](#).

4.21.3.3 addTime()

```
virtual void LeftPeelOff::addTime ( ) [inline], [virtual]
```

adds the time and fuel and tyre conditions to the car

Implements [RaceTrackComponent](#).

4.21.3.4 getAverageTime()

```
int LeftPeelOff::getAverageTime ( ) [inline]
```

returns the average time for the track

Returns

4.21.3.5 print()

```
virtual void LeftPeelOff::print ( ) [inline], [virtual]
```

prints the description of the race track component /with decorators if it has

Implements [RaceTrackComponent](#).

The documentation for this class was generated from the following file:

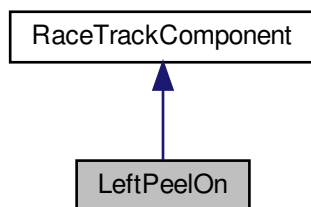
- [LeftPeelOff.h](#)

4.22 LeftPeelOn Class Reference

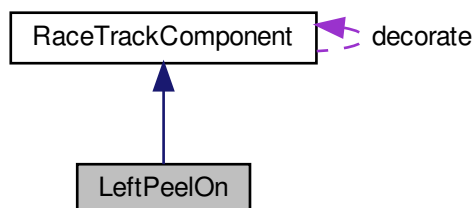
Leaf for Composite design pattern.

```
#include <LeftPeelOn.h>
```

Inheritance diagram for LeftPeelOn:



Collaboration diagram for LeftPeelOn:



Public Member Functions

- [LeftPeelOn](#) ()
- virtual [~LeftPeelOn](#) ()
- virtual void [add](#) ([RaceTrackComponent](#) *R)
- virtual void [print](#) ()
- int [getAverageTime](#) ()
- virtual void [accept](#) ([BigBrother](#) *v)
- virtual void [addTime](#) ()

Additional Inherited Members

4.22.1 Detailed Description

Leaf for Composite design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.22.2 Constructor & Destructor Documentation

4.22.2.1 [LeftPeelOn](#)()

```
LeftPeelOn::LeftPeelOn ( ) [inline]
```

constructor calls [RaceTrackComponent](#) and sets description

4.22.2.2 [~LeftPeelOn](#)()

```
virtual LeftPeelOn::~~LeftPeelOn ( ) [inline], [virtual]
```

destructor

4.22.3 Member Function Documentation

4.22.3.1 [accept](#)()

```
virtual void LeftPeelOn::accept (
    BigBrother * v ) [inline], [virtual]
```

accepts the visitor to go to the correct part of the visitor

Parameters

<i>V</i>	
----------	--

Implements [RaceTrackComponent](#).

4.22.3.2 add()

```
virtual void LeftPeelOn::add (  
    RaceTrackComponent * R ) [inline], [virtual]
```

implentation of add function, used by decorator

Parameters

<i>R</i>	
----------	--

Implements [RaceTrackComponent](#).

4.22.3.3 addTime()

```
virtual void LeftPeelOn::addTime ( ) [inline], [virtual]
```

adds the time and fuel and tyre conditions to the car

Implements [RaceTrackComponent](#).

4.22.3.4 getAverageTime()

```
int LeftPeelOn::getAverageTime ( ) [inline]
```

returns the average time for the track

Returns

4.22.3.5 print()

```
virtual void LeftPeelOn::print ( ) [inline], [virtual]
```

prints the description of the race track component /with decorators if it has

Implements [RaceTrackComponent](#).

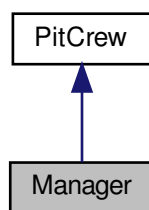
The documentation for this class was generated from the following file:

- [LeftPeelOn.h](#)

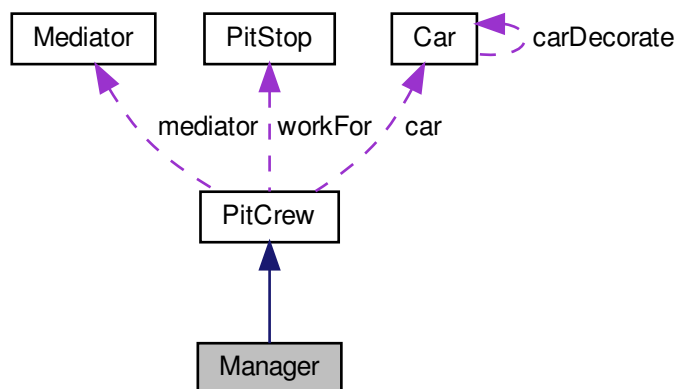
4.23 Manager Class Reference

```
#include <Manager.h>
```

Inheritance diagram for Manager:



Collaboration diagram for Manager:



Public Member Functions

- [Manager](#) ([Mediator](#) *med, [Car](#) *car)
- virtual bool * [getTyreCondition](#) ()
- virtual void [setTyreCondition](#) (bool *status)
- virtual bool [getFuelLevel](#) ()
- virtual void [setFuelLevel](#) (bool status)
- virtual bool [getDamage](#) ()
- virtual void [setDamage](#) (bool status)
- virtual void [update](#) (bool *tyreCondition, bool fuelLevel, bool damage)

Additional Inherited Members

4.23.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.23.2 Constructor & Destructor Documentation

4.23.2.1 Manager()

```
Manager::Manager (
    Mediator * med,
    Car * car ) [inline]
```

Constructor for the [Manager](#)

Parameters

<i>med</i>	- Mediator for the team
<i>car</i>	- Car for the team

4.23.3 Member Function Documentation

4.23.3.1 getDamage()

```
virtual bool Manager::getDamage ( ) [inline], [virtual]
```

Get the damage for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.23.3.2 getFuelLevel()

```
virtual bool Manager::getFuelLevel ( ) [inline], [virtual]
```

Get the fuelLevel for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.23.3.3 getTyreCondition()

```
virtual bool* Manager::getTyreCondition ( ) [inline], [virtual]
```

Get the tyreCondition for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.23.3.4 setDamage()

```
virtual void Manager::setDamage (
    bool status ) [inline], [virtual]
```

Set the damage for the car

Parameters

bool	saying if there is a problem or not
------	-------------------------------------

Reimplemented from [PitCrew](#).

4.23.3.5 setFuelLevel()

```
virtual void Manager::setFuelLevel (
    bool status ) [inline], [virtual]
```

Set the fuelLevel for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.23.3.6 setTyreCondition()

```
virtual void Manager::setTyreCondition (
    bool * status ) [inline], [virtual]
```

Set the tyreCondition for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.23.3.7 update()

```
virtual void Manager::update (
    bool * tyreCondition,
    bool fuelLevel,
    bool damage ) [inline], [virtual]
```

Check if there is a problem with the car and change the state accordingly and notify other members

Parameters

<i>tyreCondition</i>	
<i>fuelLevel</i>	
<i>damage</i>	

Implements [PitCrew](#).

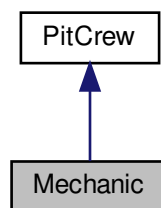
The documentation for this class was generated from the following file:

- [Manager.h](#)

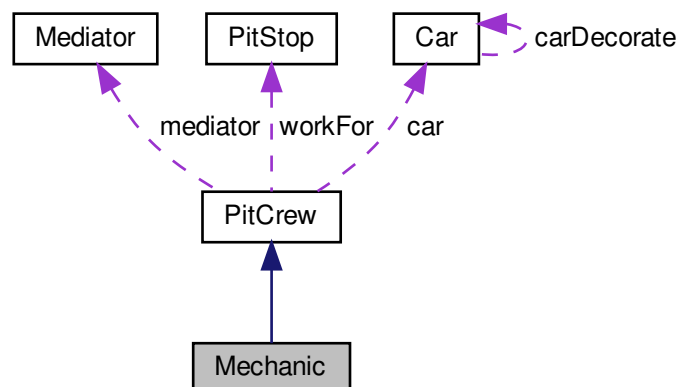
4.24 Mechanic Class Reference

```
#include <Mechanic.h>
```

Inheritance diagram for Mechanic:



Collaboration diagram for Mechanic:



Public Member Functions

- [Mechanic](#) ([Mediator](#) **med*, [Car](#) **car*)
- virtual bool * [getTyreCondition](#) ()
- virtual void [setTyreCondition](#) (bool **status*)
- virtual bool [getFuelLevel](#) ()
- virtual void [setFuelLevel](#) (bool *status*)
- virtual bool [getDamage](#) ()
- virtual void [setDamage](#) (bool *status*)
- virtual void [update](#) (bool **tyreCondition*, bool *fuelLevel*, bool *damage*)
- void [repair](#) ()

Additional Inherited Members

4.24.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.24.2 Constructor & Destructor Documentation

4.24.2.1 [Mechanic](#)()

```
Mechanic::Mechanic (  
    Mediator * med,  
    Car * car ) [inline]
```

Constructor for the [Mechanic](#)

Parameters

<i>med</i>	- Mediator for the team
<i>car</i>	- Car for the team

4.24.3 Member Function Documentation

4.24.3.1 `getDamage()`

```
virtual bool Mechanic::getDamage ( ) [inline], [virtual]
```

Get the damage for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.24.3.2 `getFuelLevel()`

```
virtual bool Mechanic::getFuelLevel ( ) [inline], [virtual]
```

Get the fuelLevel for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.24.3.3 `getTyreCondition()`

```
virtual bool* Mechanic::getTyreCondition ( ) [inline], [virtual]
```

Get the tyreCondition for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.24.3.4 `repair()`

```
void Mechanic::repair ( ) [inline]
```

Repair the car and notify the manager

4.24.3.5 `setDamage()`

```
virtual void Mechanic::setDamage (
    bool status ) [inline], [virtual]
```

Set the damage for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.24.3.6 setFuelLevel()

```
virtual void Mechanic::setFuelLevel (  
    bool status ) [inline], [virtual]
```

Set the fuelLevel for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.24.3.7 setTyreCondition()

```
virtual void Mechanic::setTyreCondition (  
    bool * status ) [inline], [virtual]
```

Set the tyreCondition for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.24.3.8 update()

```
virtual void Mechanic::update (  
    bool * tyreCondition,  
    bool fuelLevel,  
    bool damage ) [inline], [virtual]
```

Check if there is a problem with the car and change the state accordingly and notify other members

Parameters

<i>tyreCondition</i>	
<i>fuelLevel</i>	
<i>damage</i>	

Implements [PitCrew](#).

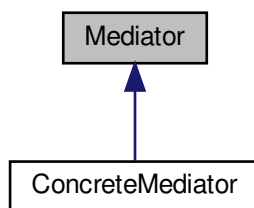
The documentation for this class was generated from the following file:

- [Mechanic.h](#)

4.25 Mediator Class Reference

```
#include <Mediator.h>
```

Inheritance diagram for Mediator:



Public Member Functions

- virtual void [notify](#) ([PitCrew](#) *member)=0
- virtual void [addMember](#) ([PitCrew](#) *member)=0
- virtual void [notifyManager](#) ([PitCrew](#) *member)=0
- virtual [~Mediator](#) ()

4.25.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.25.2 Constructor & Destructor Documentation

4.25.2.1 ~Mediator()

```
virtual Mediator::~~Mediator ( ) [inline], [virtual]
```

Virtual destructor for the mediator

4.25.3 Member Function Documentation

4.25.3.1 addMember()

```
virtual void Mediator::addMember (
    PitCrew * member ) [pure virtual]
```

Abstract method for adding a member to the mediator

Parameters

<i>member</i>	
---------------	--

Implemented in [ConcreteMediator](#).

4.25.3.2 notify()

```
virtual void Mediator::notify (
    PitCrew * member ) [pure virtual]
```

Abstract method for notifying other members of the team

Parameters

<i>member</i>	
---------------	--

Implemented in [ConcreteMediator](#).

4.25.3.3 notifyManager()

```
virtual void Mediator::notifyManager (
    PitCrew * member ) [pure virtual]
```

Abstract method for notifying the manager of the team

Parameters

<i>member</i>	
---------------	--

Implemented in [ConcreteMediator](#).

The documentation for this class was generated from the following file:

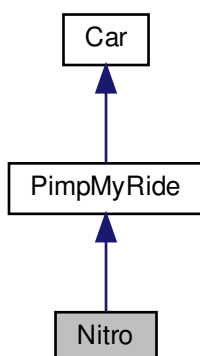
- [Mediator.h](#)

4.26 Nitro Class Reference

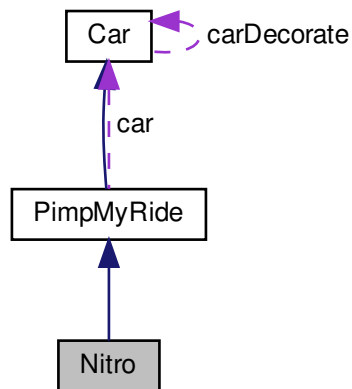
Concrete Decorator for Decorator Pattern.

```
#include <Nitro.h>
```

Inheritance diagram for Nitro:



Collaboration diagram for Nitro:



Public Member Functions

- [Nitro](#) ([Car](#) *Decorate)
- [~Nitro](#) ()
- [Nitro](#) ([Nitro](#) _Car, bool dummy)
- virtual [Car](#) * [FullClone](#) ()

Additional Inherited Members

4.26.1 Detailed Description

Concrete Decorator for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.26.2 Constructor & Destructor Documentation

4.26.2.1 Nitro() [1/2]

```

Nitro::Nitro (
    Car * Decorate )    [inline]
  
```

constructor which assigns description and alters behaviour of car

Parameters

<i>Decorate</i>	the car in which the behaviours are added
-----------------	---

4.26.2.2 ~Nitro()

```
Nitro::~~Nitro ( ) [inline]
```

destructor for [Nitro](#)

4.26.2.3 Nitro() [2/2]

```
Nitro::Nitro (
    Nitro _Car,
    bool dummy ) [inline]
```

copy constructor used for cloning the decorators

Parameters

<i>_Car</i>	the car it copies
<i>dummy</i>	just there to use instead of default constructor

4.26.3 Member Function Documentation

4.26.3.1 FullClone()

```
virtual Car* Nitro::FullClone ( ) [inline], [virtual]
```

implementation of FullClone to deep copy the decorator

Returns

[Car](#) object which is the decorator

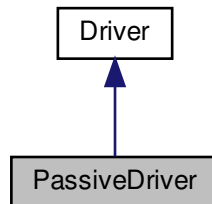
Reimplemented from [PimpMyRide](#).

The documentation for this class was generated from the following file:

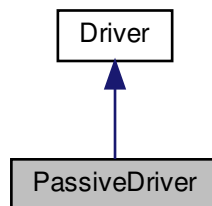
- [Nitro.h](#)

4.27 PassiveDriver Class Reference

Inheritance diagram for PassiveDriver:



Collaboration diagram for PassiveDriver:



Public Member Functions

- [PassiveDriver](#) ()

4.27.1 Constructor & Destructor Documentation

4.27.1.1 PassiveDriver()

```
PassiveDriver::PassiveDriver ( ) [inline]
```

constructor to set fuel, tyre and driving ability

The documentation for this class was generated from the following file:

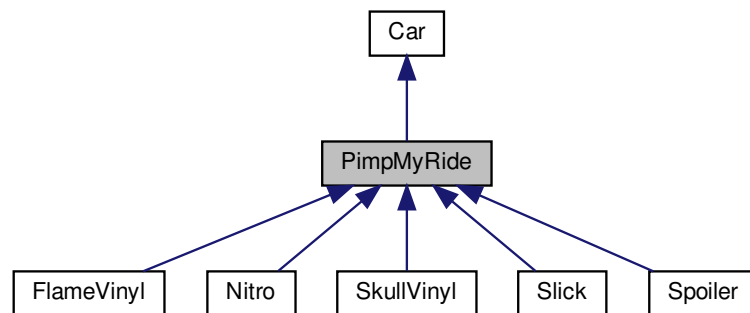
- `PassiveDriver.h`

4.28 PimpMyRide Class Reference

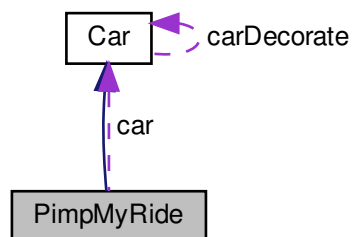
Decorcorator for Decorator Pattern.

```
#include <PimpMyRide.h>
```

Inheritance diagram for PimpMyRide:



Collaboration diagram for PimpMyRide:



Public Member Functions

- [PimpMyRide](#) ()
- [~PimpMyRide](#) ()
- virtual void [add](#) ([Car](#) * _car)
- virtual [Car](#) * [clone](#) (bool flag_)
- virtual string [showCarStats](#) ()
- virtual [Car](#) * [FullClone](#) ()

Public Attributes

- [Car](#) * [car](#)
a car object that will be decorated

4.28.1 Detailed Description

Decorcator for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.28.2 Constructor & Destructor Documentation

4.28.2.1 PimpMyRide()

```
PimpMyRide::PimpMyRide ( ) [inline]
```

Default constructor used for [PimpMyRide](#)

4.28.2.2 ~PimpMyRide()

```
PimpMyRide::~~PimpMyRide ( ) [inline]
```

The destructor for [Car](#)

4.28.3 Member Function Documentation

4.28.3.1 add()

```
virtual void PimpMyRide::add (  
    Car * _car ) [inline], [virtual]
```

add function for decorator

Parameters

<i>c</i>	is car object
----------	---------------

Reimplemented from [Car](#).

4.28.3.2 clone()

```
virtual Car* PimpMyRide::clone (
    bool flag_ ) [inline], [virtual]
```

for clone for prototype pattern

Parameters

<i>flag_</i>	to determine if must be full clone or basic
--------------	---

Returns

[Car](#) object

Implements [Car](#).

4.28.3.3 FullClone()

```
virtual Car* PimpMyRide::FullClone ( ) [inline], [virtual]
```

for Fullclone for prototype pattern of decorator

Returns

[Car](#) object

Implements [Car](#).

Reimplemented in [Nitro](#), [Slick](#), [Spoiler](#), [FlameVinyl](#), and [SkullVinyl](#).

4.28.3.4 showCarStats()

```
virtual string PimpMyRide::showCarStats ( ) [inline], [virtual]
```

showCarStats function to show the stats of a car

Returns

string stating the stats

Reimplemented from [Car](#).

The documentation for this class was generated from the following file:

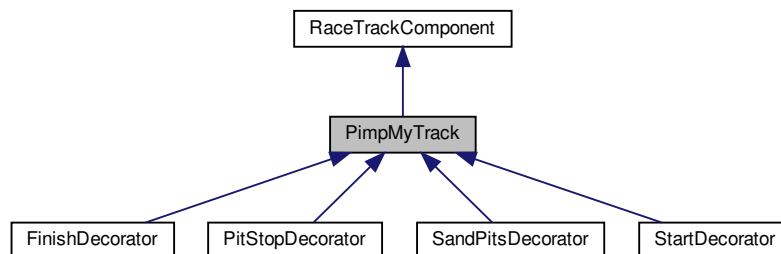
- [PimpMyRide.h](#)

4.29 PimpMyTrack Class Reference

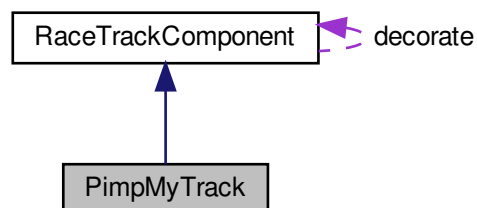
Abstract Decorator for Decorator design pattern.

```
#include <PimpMyTrack.h>
```

Inheritance diagram for PimpMyTrack:



Collaboration diagram for PimpMyTrack:



Public Member Functions

- [PimpMyTrack](#) ()
- [~PimpMyTrack](#) ()
- virtual void [print](#) ()
- virtual void [add](#) ([RaceTrackComponent](#) *)
- virtual void [addTime](#) ()
- virtual void [accept](#) ([BigBrother](#) *v)

Additional Inherited Members

4.29.1 Detailed Description

Abstract Decorator for Decorator design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.29.2 Constructor & Destructor Documentation

4.29.2.1 PimpMyTrack()

```
PimpMyTrack::PimpMyTrack ( ) [inline]
```

constructor calls parent constructor

4.29.2.2 ~PimpMyTrack()

```
PimpMyTrack::~~PimpMyTrack ( ) [inline]
```

destuctor

4.29.3 Member Function Documentation

4.29.3.1 accept()

```
virtual void PimpMyTrack::accept (
    BigBrother * v ) [inline], [virtual]
```

empty implementation of accept

Parameters

v	
---	--

Implements [RaceTrackComponent](#).

4.29.3.2 add()

```
virtual void PimpMyTrack::add (  
    RaceTrackComponent * ) [inline], [virtual]
```

empty implementation of add

Implements [RaceTrackComponent](#).

4.29.3.3 addTime()

```
virtual void PimpMyTrack::addTime ( ) [inline], [virtual]
```

empty implementation of add time

Implements [RaceTrackComponent](#).

4.29.3.4 print()

```
virtual void PimpMyTrack::print ( ) [inline], [virtual]
```

empty implementation of print

Implements [RaceTrackComponent](#).

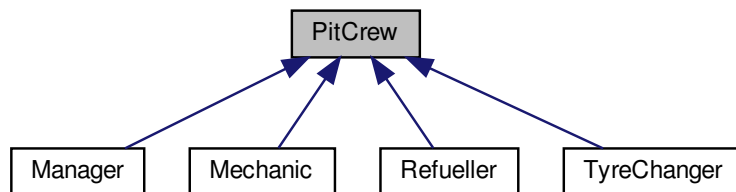
The documentation for this class was generated from the following file:

- [PimpMyTrack.h](#)

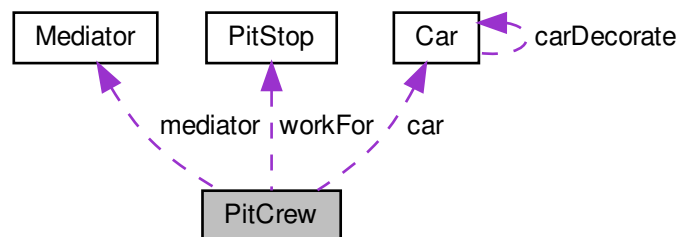
4.30 PitCrew Class Reference

```
#include <PitCrew.h>
```

Inheritance diagram for PitCrew:



Collaboration diagram for PitCrew:



Public Member Functions

- [PitCrew](#) ([Mediator](#) *med, [Car](#) *car)
- void [registerWork](#) ([PitStop](#) *pitStop)
- void [changed](#) ()
- void [changedCar](#) ()
- void [setDescription](#) (string des)
- string [getDescription](#) ()
- virtual void [update](#) (bool *tyreCondiiton, bool [fuelLevel](#), bool [damage](#))=0
- virtual bool * [getTyreCondition](#) ()
- virtual void [setTyreCondition](#) (bool *status)
- virtual bool [getFuelLevel](#) ()
- virtual void [setFuelLevel](#) (bool status)
- virtual bool [getDamage](#) ()
- virtual void [setDamage](#) (bool status)

Protected Attributes

- string [description](#)
description describing the crew member
- [PitStop](#) * [workFor](#)
the team/pitstop the member is working for
- [Car](#) * [car](#)
the car the member is working for
- bool * [tyreCondition](#)
tyrecondition for the car
- bool [fuelLevel](#)
fuel level for the car
- bool [damage](#)
damage for the car
- [Mediator](#) * [mediator](#)
mediator the car belongs to

4.30.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.30.2 Constructor & Destructor Documentation

4.30.2.1 PitCrew()

```
PitCrew::PitCrew (
    Mediator * med,
    Car * car )
```

default constructor for a new [PitCrew](#) member

Parameters

<i>med</i>	The mediator for which the member will communicate to
<i>car</i>	The car the member will work with

4.30.3 Member Function Documentation

4.30.3.1 changed()

```
void PitCrew::changed ( )
```

function to tell other colleagues that belongs to the same mediator that the car is changed

4.30.3.2 changedCar()

```
void PitCrew::changedCar ( )
```

function when the car has been changed but when the team should not be notified

4.30.3.3 getDamage()

```
bool PitCrew::getDamage ( ) [virtual]
```

Get the damage for the car

Returns

bool saying if there is a problem or not

Reimplemented in [Manager](#), [TyreChanger](#), [Mechanic](#), and [Refueller](#).

4.30.3.4 getDescription()

```
string PitCrew::getDescription ( )
```

function to get the description of the member

Returns

4.30.3.5 getFuelLevel()

```
bool PitCrew::getFuelLevel ( ) [virtual]
```

Get the fuelLevel for the car

Returns

bool saying if there is a problem or not

Reimplemented in [Manager](#), [TyreChanger](#), [Mechanic](#), and [Refueller](#).

4.30.3.6 getTyreCondition()

```
bool * PitCrew::getTyreCondition ( ) [virtual]
```

Get the tyreCondition for the car

Returns

bool saying if there is a problem or not

Reimplemented in [Manager](#), [TyreChanger](#), [Mechanic](#), and [Refueller](#).

4.30.3.7 registerWork()

```
void PitCrew::registerWork (
    PitStop * pitStop )
```

function for attaching the crewmembers to a pitstop allowing us to get a description of the crew members of a team

Parameters

<i>pitStop</i>	the pitstop the member is working for
----------------	---------------------------------------

4.30.3.8 setDamage()

```
void PitCrew::setDamage (
    bool status ) [virtual]
```

Set the damage for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented in [Manager](#), [TyreChanger](#), [Mechanic](#), and [Refueller](#).

4.30.3.9 setDescription()

```
void PitCrew::setDescription (
    string des )
```

function to set the description of a member allowing us to get a detailed view of the team

Parameters

<i>des</i>	
------------	--

4.30.3.10 setFuelLevel()

```
void PitCrew::setFuelLevel (
    bool status ) [virtual]
```

Set the fuelLevel for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented in [Manager](#), [TyreChanger](#), [Mechanic](#), and [Refueller](#).

4.30.3.11 setTyreCondition()

```
void PitCrew::setTyreCondition (
    bool * status ) [virtual]
```

Set the tyreCondition for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented in [Manager](#), [TyreChanger](#), [Mechanic](#), and [Refueller](#).

4.30.3.12 update()

```
virtual void PitCrew::update (
    bool * tyreCondiiton,
    bool fuelLevel,
    bool damage ) [pure virtual]
```

Abstract function to update the team members

Parameters

<i>tyreCondiiton</i>	
<i>fuelLevel</i>	
<i>damage</i>	

Implemented in [Manager](#), [TyreChanger](#), [Mechanic](#), and [Refueller](#).

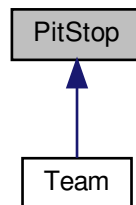
The documentation for this class was generated from the following files:

- [PitCrew.h](#)
- [PitCrew.cpp](#)

4.31 PitStop Class Reference

```
#include <PitStop.h>
```

Inheritance diagram for PitStop:



Public Member Functions

- [PitStop](#) (string name)
- [~PitStop](#) ()
- void [attach](#) ([PitCrew](#) *member)
- void [attachManager](#) ([PitCrew](#) *manager)
- void [detach](#) ([PitCrew](#) *member)
- void [notify](#) ()
- void [setTyreCondition](#) (bool *status)
- void [setFuelLevel](#) (bool status)
- void [setDamage](#) (bool status)
- virtual void [addCar](#) ([Car](#) *car)
- virtual [Car](#) * [getCar](#) ()
- [PitCrew](#) * [getManager](#) ()
- [PitCrew](#) * [getMember](#) (int index)
- int [getNumMembers](#) ()
- string [showCar](#) ()
- string [showManager](#) ()
- string [showCrew](#) ()
- virtual void [getCarStats](#) ()=0
- string [getName](#) ()
- string [toString](#) ()

4.31.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.31.2 Constructor & Destructor Documentation

4.31.2.1 PitStop()

```
PitStop::PitStop (
    string name )
```

default constructor for a pitstop

Parameters

<i>name</i>	of the team/pitstop
-------------	---------------------

4.31.2.2 ~PitStop()

```
PitStop::~~PitStop ( )
```

destructor for the pitstop class, also detaches all members from the pitcrew vector

4.31.3 Member Function Documentation

4.31.3.1 addCar()

```
void PitStop::addCar (
    Car * car ) [virtual]
```

function that adds a new car to the cars vector for a specific team

Parameters

<i>car</i>	to be added
------------	-------------

4.31.3.2 attach()

```
void PitStop::attach (
    PitCrew * member )
```

function for adding [PitCrew](#) members to the team/pitstop

Parameters

<i>member</i>	added to the pitcrew vector
---------------	-----------------------------

4.31.3.3 attachManager()

```
void PitStop::attachManager (
    PitCrew * manager )
```

function for adding [PitCrew](#) manager to the team/pitstop

Parameters

<i>manager</i>	added to the pitcrew vector
----------------	-----------------------------

4.31.3.4 detach()

```
void PitStop::detach (
    PitCrew * member )
```

function for adding [PitCrew](#) members to the team/pitstop

Parameters

<i>member</i>	added to the pitcrew vector
---------------	-----------------------------

4.31.3.5 getCar()

```
Car * PitStop::getCar ( ) [virtual]
```

function that gets a car from the cars vector for a specific team

4.31.3.6 getCarStats()

```
virtual void PitStop::getCarStats ( ) [pure virtual]
```

Abstract method to get the car stats

Implemented in [Team](#).

4.31.3.7 getManager()

```
PitCrew * PitStop::getManager ( )
```

function that gets the manager from the cars vector for a specific team

4.31.3.8 getMember()

```
PitCrew* PitStop::getMember (
    int index ) [inline]
```

function that gets the member with the index sent through

Parameters

<i>index</i>	
--------------	--

Returns

4.31.3.9 getName()

```
string PitStop::getName ( )
```

getter function to return the name of the team/pitstop

Returns

name of the team/pitstop

4.31.3.10 getNumMembers()

```
int PitStop::getNumMembers ( ) [inline]
```

function to get the number of members in the team

Returns

4.31.3.11 notify()

```
void PitStop::notify ( )
```

function that will notify the manager if the condition of the car changed

Parameters

<i>index</i>	the index of the car
--------------	----------------------

4.31.3.12 setDamage()

```
void PitStop::setDamage (
    bool status )
```

function that sets the condition of the cars damage depending if needs to be repaired or not

Parameters

<i>status</i>	a bool describing the status of the damage of the car
---------------	---

4.31.3.13 setFuelLevel()

```
void PitStop::setFuelLevel (
    bool status )
```

function that sets the condition of the fuel level of a car depending if it needs to be filled or not

Parameters

<i>status</i>	a bool describing the status of the fuel level
---------------	--

4.31.3.14 setTyreCondition()

```
void PitStop::setTyreCondition (
    bool * status )
```

function that sets the condition for all the tyres to true or false depending if they need change or not

Parameters

<i>status</i>	a bool array describing the status of each car
---------------	--

4.31.3.15 showCar()

```
string PitStop::showCar ( )
```

function that lists all the cars in the cars vector for the team

Returns

a string with all the cars

4.31.3.16 showCrew()

```
string PitStop::showCrew ( )
```

function that gets the crew of the team

Returns

a description of the crew

4.31.3.17 showManager()

```
string PitStop::showManager ( )
```

function that gets the managers of the team

Returns

a description of the manager

4.31.3.18 toString()

```
string PitStop::toString ( )
```

function to print the team in detail to see who is in the team

Returns

string description of the team

The documentation for this class was generated from the following files:

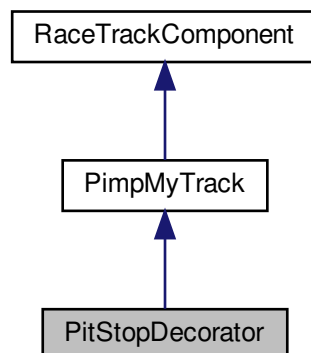
- [PitStop.h](#)
- PitStop.cpp

4.32 PitStopDecorator Class Reference

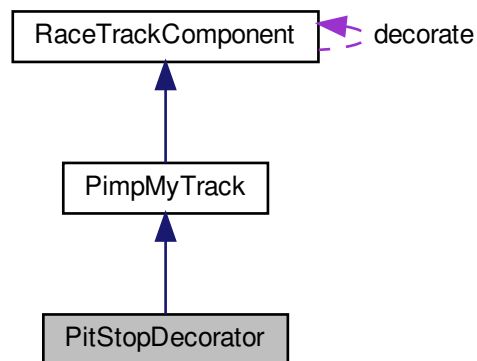
ConcreteDecorator for Decorator design pattern.

```
#include <PitStopDecorator.h>
```

Inheritance diagram for PitStopDecorator:



Collaboration diagram for PitStopDecorator:



Public Member Functions

- [PitStopDecorator\(\)](#)
- [~PitStopDecorator\(\)](#)

Additional Inherited Members

4.32.1 Detailed Description

ConcreteDecorator for Decorator design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.32.2 Constructor & Destructor Documentation

4.32.2.1 PitStopDecorator()

```
PitStopDecorator::PitStopDecorator ( ) [inline]
```

Constructor that calls constructor of pimpMyTrack and has a description

4.32.2.2 ~PitStopDecorator()

```
PitStopDecorator::~~PitStopDecorator ( ) [inline]
```

destructor

The documentation for this class was generated from the following file:

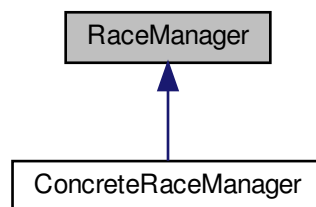
- [PitStopDecorator.h](#)

4.33 RaceManager Class Reference

Observer class for Observer pattern.

```
#include <RaceManager.h>
```

Inheritance diagram for RaceManager:



Public Member Functions

- virtual void [readyRace](#) ()=0
- virtual void [startRace](#) ()=0
- virtual void [stopRace](#) ()=0
- virtual void [pauseRace](#) (int numComponent)=0
- virtual void [resumeRace](#) (int numComponent)=0
- virtual void [printLeaderBoard](#) ()=0
- virtual void [addCars](#) (vector< [Car](#) *> _cars)=0
- virtual void [addRacetrack](#) ([RaceTrackComponent](#) *raceTrackComponent)=0

4.33.1 Detailed Description

Observer class for Observer pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.33.2 Member Function Documentation

4.33.2.1 addCars()

```
virtual void RaceManager::addCars (
    vector< Car *> _cars ) [pure virtual]
```

add cars to the race manager

Parameters

<i>_cars</i>	
--------------	--

Implemented in [ConcreteRaceManager](#).

4.33.2.2 addRacetrack()

```
virtual void RaceManager::addRacetrack (
    RaceTrackComponent * raceTrackComponent ) [pure virtual]
```

adds the race

Parameters

<i>raceTrackComponent</i>	
---------------------------	--

Implemented in [ConcreteRaceManager](#).

4.33.2.3 pauseRace()

```
virtual void RaceManager::pauseRace (
    int numComponent ) [pure virtual]
```

pauses the race

Parameters

<i>numComponent</i>	
---------------------	--

Implemented in [ConcreteRaceManager](#).

4.33.2.4 printLeaderBoard()

```
virtual void RaceManager::printLeaderBoard ( ) [pure virtual]
```

prints the cars in order according to track times

Implemented in [ConcreteRaceManager](#).

4.33.2.5 readyRace()

```
virtual void RaceManager::readyRace ( ) [pure virtual]
```

Moves all cars to starting point of track and sets the times to 0

Implemented in [ConcreteRaceManager](#).

4.33.2.6 resumeRace()

```
virtual void RaceManager::resumeRace (
    int numComponent ) [pure virtual]
```

resumes the race according to where it left

Parameters

<i>numComponent</i>	
---------------------	--

Implemented in [ConcreteRaceManager](#).

4.33.2.7 startRace()

```
virtual void RaceManager::startRace ( ) [pure virtual]
```

starts to move the cars along the racetrack

Implemented in [ConcreteRaceManager](#).

4.33.2.8 stopRace()

```
virtual void RaceManager::stopRace ( ) [pure virtual]
```

announces when the race is finished and prints the final leaderboard

Implemented in [ConcreteRaceManager](#).

The documentation for this class was generated from the following file:

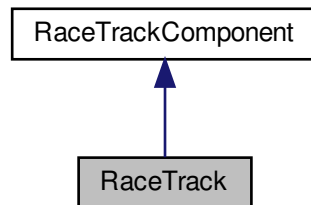
- [RaceManager.h](#)

4.34 RaceTrack Class Reference

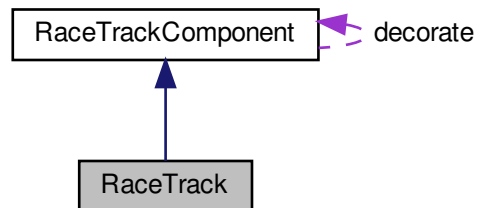
composite class for composite pattern

```
#include <RaceTrack.h>
```

Inheritance diagram for RaceTrack:



Collaboration diagram for RaceTrack:



Public Member Functions

- `RaceTrack` ()
- virtual void `print` ()
- virtual void `add` (`RaceTrackComponent` *Ra)
- virtual `~RaceTrack` ()
- void `show` ()
- void `moveCar` (`Car` *_car, int rt)
- int `getRaceTrackID` ()
- int `getNumComponents` ()
- void `addAllCars` (vector< `Car` *> _car, int rt)
- void `removeAllCars` (vector< `Car` *> _car, int rt)
- vector< `Car` * > `getAllCars` (int rt)
- virtual void `accept` (`BigBrother` *v)
- void `makeAccept` (`BigBrother` *v, int rt)
- virtual void `addTime` ()

Additional Inherited Members

4.34.1 Detailed Description

composite class for composite pattern

Authors

Duncan + Tjaart

Version

1.0.0

4.34.2 Constructor & Destructor Documentation

4.34.2.1 RaceTrack()

```
RaceTrack::RaceTrack ( ) [inline]
```

constructor calls parent constructor

4.34.2.2 ~RaceTrack()

```
virtual RaceTrack::~RaceTrack ( ) [inline], [virtual]
```

destructor that deletes the racetrack parts

4.34.3 Member Function Documentation

4.34.3.1 accept()

```
virtual void RaceTrack::accept (
    BigBrother * v ) [inline], [virtual]
```

empty implementation of abstract class

Parameters

v	
---	--

Implements [RaceTrackComponent](#).

4.34.3.2 add()

```
virtual void RaceTrack::add (
    RaceTrackComponent * Ra ) [inline], [virtual]
```

implementation of adding racetrack component to the track

Parameters

<i>Ra</i>	
-----------	--

Implements [RaceTrackComponent](#).

4.34.3.3 addAllCars()

```
void RaceTrack::addAllCars (
    vector< Car *> _car,
    int rt ) [inline], [virtual]
```

adds all the cars to the track part

Parameters

<i>_car</i>	
<i>rt</i>	

Reimplemented from [RaceTrackComponent](#).

4.34.3.4 addTime()

```
virtual void RaceTrack::addTime ( ) [inline], [virtual]
```

adds the time onto the car

Implements [RaceTrackComponent](#).

4.34.3.5 getAllCars()

```
vector<Car*> RaceTrack::getAllCars (
    int rt ) [inline], [virtual]
```

returns all the cars as a wector

Parameters

<i>rt</i>	
-----------	--

Returns

car vector

Reimplemented from [RaceTrackComponent](#).

4.34.3.6 getNumComponents()

```
int RaceTrack::getNumComponents ( ) [inline], [virtual]
```

returns the number of track components in the track

Returns

Reimplemented from [RaceTrackComponent](#).

4.34.3.7 getRaceTrackID()

```
int RaceTrack::getRaceTrackID ( ) [inline]
```

Returns

the RacetrackID

4.34.3.8 makeAccept()

```
void RaceTrack::makeAccept (
    BigBrother * v,
    int rt ) [inline], [virtual]
```

element accepts the visitor

Parameters

<i>v</i>	
<i>rt</i>	

Reimplemented from [RaceTrackComponent](#).

4.34.3.9 moveCar()

```
void RaceTrack::moveCar (
    Car * _car,
    int rt ) [inline], [virtual]
```

move car to certain track part

Parameters

<i>_car</i>	
<i>rt</i>	

Reimplemented from [RaceTrackComponent](#).

4.34.3.10 print()

```
virtual void RaceTrack::print ( ) [inline], [virtual]
```

starts iteration of printing the race track

Implements [RaceTrackComponent](#).

4.34.3.11 removeAllCars()

```
void RaceTrack::removeAllCars (
    vector< Car *> _car,
    int rt ) [inline], [virtual]
```

removes all the cars from the race track component

Parameters

<i>_car</i>	
<i>rt</i>	

Reimplemented from [RaceTrackComponent](#).

4.34.3.12 show()

```
void RaceTrack::show ( ) [inline], [virtual]
```

prints the race track parts nicely

Reimplemented from [RaceTrackComponent](#).

The documentation for this class was generated from the following files:

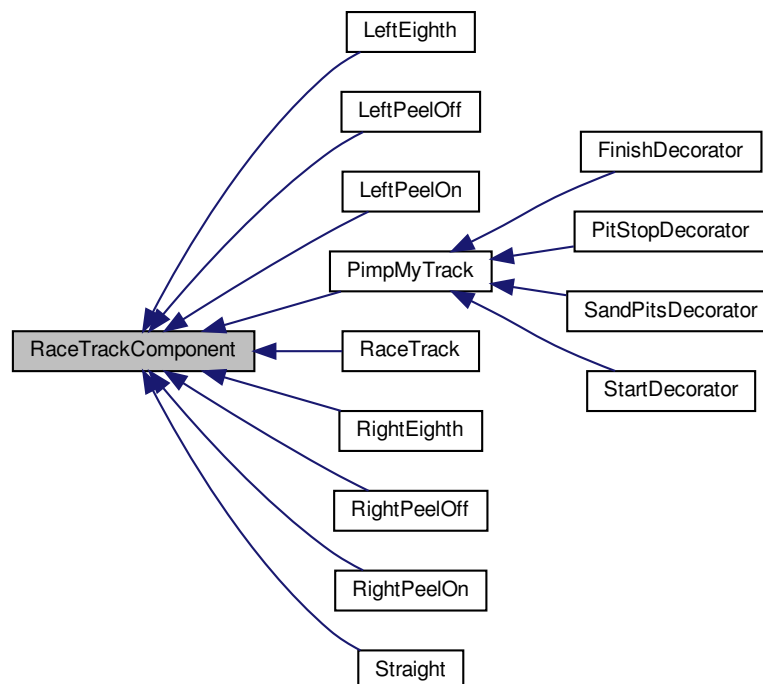
- [RaceTrack.h](#)
- RaceTrack.cpp

4.35 RaceTrackComponent Class Reference

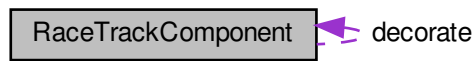
abstract leaf class for composite pattern

```
#include <RaceTrackComponent.h>
```

Inheritance diagram for RaceTrackComponent:



Collaboration diagram for RaceTrackComponent:



Public Member Functions

- [RaceTrackComponent](#) ()
- virtual void [decorateTrack](#) ([RaceTrackComponent](#) *R)
- virtual string [getDecorator](#) ()
- void [addCar](#) ([Car](#) *_car)
- void [removeCar](#) ([Car](#) *_car)
- virtual int [getNumComponents](#) ()
- virtual void [addAllCars](#) (vector< [Car](#) *> _car, int rt)
- virtual void [removeAllCars](#) (vector< [Car](#) *> _car, int rt)
- virtual vector< [Car](#) * > [getAllCars](#) (int rt)
- virtual void [makeAccept](#) ([BigBrother](#) *v, int rt)
- virtual void [moveCar](#) ([Car](#) *_car, int rt)
- virtual void [show](#) ()
- virtual void [print](#) ()=0
- virtual void [add](#) ([RaceTrackComponent](#) *)=0
- virtual [~RaceTrackComponent](#) ()
- void [setDescription](#) (string d)
- string [getDescription](#) ()
- virtual void [addTime](#) ()=0
- virtual void [accept](#) ([BigBrother](#) *v)=0
- vector< [Car](#) * > [getCars](#) ()

Public Attributes

- [RaceTrackComponent](#) * [decorate](#)
- vector< [Car](#) * > [cars](#)

4.35.1 Detailed Description

abstract leaf class for composite pattern

Authors

Duncan + Tjaart

Version

1.0.0

4.35.2 Constructor & Destructor Documentation

4.35.2.1 RaceTrackComponent()

```
RaceTrackComponent::RaceTrackComponent ( ) [inline]
```

Parent Constructor for the racetrack parys

4.35.2.2 ~RaceTrackComponent()

```
virtual RaceTrackComponent::~~RaceTrackComponent ( ) [inline], [virtual]
```

destructor

4.35.3 Member Function Documentation

4.35.3.1 accept()

```
virtual void RaceTrackComponent::accept (
    BigBrother * v ) [pure virtual]
```

accept visitor

Parameters

<i>v</i>	
----------	--

Implemented in [RaceTrack](#), [LeftEighth](#), [RightEighth](#), [RightPeelOff](#), [RightPeelOn](#), [LeftPeelOn](#), [Straight](#), [LeftPeelOff](#), and [PimpMyTrack](#).

4.35.3.2 add()

```
virtual void RaceTrackComponent::add (
    RaceTrackComponent * ) [pure virtual]
```

pure virtual implementation of add

Implemented in [RaceTrack](#), [PimpMyTrack](#), [LeftEighth](#), [RightPeelOff](#), [RightPeelOn](#), [LeftPeelOn](#), [RightEighth](#), [Straight](#), and [LeftPeelOff](#).

4.35.3.3 addAllCars()

```
virtual void RaceTrackComponent::addAllCars (
    vector< Car *> _car,
    int rt ) [inline], [virtual]
```

add the cars to the racetrack

Parameters

<code>_car</code>	
<code>rt</code>	

Reimplemented in [RaceTrack](#).

4.35.3.4 addCar()

```
void RaceTrackComponent::addCar (
    Car * _car ) [inline]
```

add car to the race track component

Parameters

<code>_car</code>	the car to add
-------------------	----------------

4.35.3.5 addTime()

```
virtual void RaceTrackComponent::addTime ( ) [pure virtual]
```

add time to car

Implemented in [RaceTrack](#), [LeftEighth](#), [RightEighth](#), [RightPeelOff](#), [RightPeelOn](#), [LeftPeelOn](#), [Straight](#), [LeftPeelOff](#), and [PimpMyTrack](#).

4.35.3.6 decorateTrack()

```
virtual void RaceTrackComponent::decorateTrack (
    RaceTrackComponent * R ) [inline], [virtual]
```

decorates the track with passed in paramater

Parameters

<i>R</i>	
----------	--

4.35.3.7 getAllCars()

```
virtual vector<Car*> RaceTrackComponent::getAllCars (
    int rt ) [inline], [virtual]
```

returns all the cars on a racetrack

Parameters

<i>rt</i>	
-----------	--

Returns

Reimplemented in [RaceTrack](#).

4.35.3.8 getCars()

```
vector<Car*> RaceTrackComponent::getCars ( ) [inline]
```

return the cars

Returns**4.35.3.9 getDecorator()**

```
virtual string RaceTrackComponent::getDecorator ( ) [inline], [virtual]
```

Returns

the decorators string

4.35.3.10 getDescription()

```
string RaceTrackComponent::getDescription ( ) [inline]
```

Returns

the description

4.35.3.11 getNumComponents()

```
virtual int RaceTrackComponent::getNumComponents ( ) [inline], [virtual]
```

returns the number of components

Returns

Reimplemented in [RaceTrack](#).

4.35.3.12 makeAccept()

```
virtual void RaceTrackComponent::makeAccept (
    BigBrother * v,
    int rt ) [inline], [virtual]
```

accept of visitor class

Parameters

<i>v</i>	
<i>rt</i>	

Reimplemented in [RaceTrack](#).

4.35.3.13 moveCar()

```
virtual void RaceTrackComponent::moveCar (
    Car * _car,
    int rt ) [inline], [virtual]
```

move car to certain track part

Parameters

<i>_car</i>	
<i>rt</i>	

Reimplemented in [RaceTrack](#).

4.35.3.14 print()

```
virtual void RaceTrackComponent::print ( ) [pure virtual]
```

pure virtual implementation of print

Implemented in [LeftEighth](#), [RightPeelOff](#), [RightPeelOn](#), [LeftPeelOn](#), [RightEighth](#), [Straight](#), [LeftPeelOff](#), [PimpMyTrack](#), and [RaceTrack](#).

4.35.3.15 removeAllCars()

```
virtual void RaceTrackComponent::removeAllCars (
    vector< Car *> _car,
    int rt ) [inline], [virtual]
```

removes the cars from the racetrack

Parameters

<i>_car</i>	
<i>rt</i>	

Reimplemented in [RaceTrack](#).

4.35.3.16 removeCar()

```
void RaceTrackComponent::removeCar (
    Car * _car ) [inline]
```

remove a car from the track component

Parameters

<i>_car</i>	the car to remove
-------------	-------------------

4.35.3.17 setDescription()

```
void RaceTrackComponent::setDescription (
    string d ) [inline]
```

sets the description

Parameters

<i>d</i>	the string to set the description
----------	-----------------------------------

4.35.3.18 show()

```
virtual void RaceTrackComponent::show ( ) [inline], [virtual]
```

empty implementation of show

Reimplemented in [RaceTrack](#).

4.35.4 Member Data Documentation

4.35.4.1 cars

```
vector<Car> RaceTrackComponent::cars
```

vector of cars on the track part

4.35.4.2 decorate

```
RaceTrackComponent* RaceTrackComponent::decorate
```

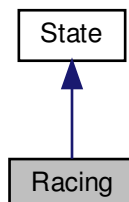
the decorate for the track component

The documentation for this class was generated from the following file:

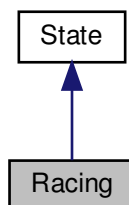
- [RaceTrackComponent.h](#)

4.36 Racing Class Reference

Inheritance diagram for Racing:



Collaboration diagram for Racing:



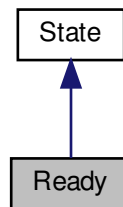
Additional Inherited Members

The documentation for this class was generated from the following files:

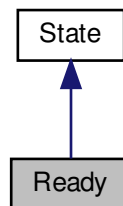
- [State.h](#)
- [State.cpp](#)

4.37 Ready Class Reference

Inheritance diagram for Ready:



Collaboration diagram for Ready:



Additional Inherited Members

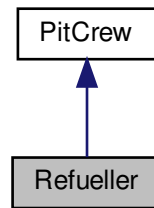
The documentation for this class was generated from the following files:

- [State.h](#)
- State.cpp

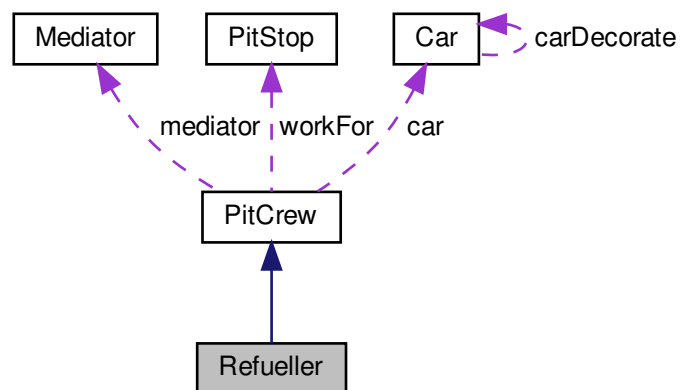
4.38 Refueller Class Reference

```
#include <Refueller.h>
```

Inheritance diagram for Refueller:



Collaboration diagram for Refueller:



Public Member Functions

- [Refueller](#) ([Mediator](#) *med, [Car](#) *car)
- virtual bool * [getTyreCondition](#) ()
- virtual void [setTyreCondition](#) (bool *status)
- virtual bool [getFuelLevel](#) ()
- virtual void [setFuelLevel](#) (bool status)
- virtual bool [getDamage](#) ()
- virtual void [setDamage](#) (bool status)
- virtual void [update](#) (bool *tyreCondition, bool fuelLevel, bool damage)
- void [refuel](#) ()

Additional Inherited Members

4.38.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.38.2 Constructor & Destructor Documentation

4.38.2.1 Refueller()

```
Refueller::Refueller (
    Mediator * med,
    Car * car ) [inline]
```

Constructor for the [Refueller](#)

Parameters

<i>med</i>	- Mediator for the team
<i>car</i>	- Car for the team

4.38.3 Member Function Documentation

4.38.3.1 getDamage()

```
virtual bool Refueller::getDamage ( ) [inline], [virtual]
```

Get the damage for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.38.3.2 getFuelLevel()

```
virtual bool Refueller::getFuelLevel ( ) [inline], [virtual]
```

Get the fuelLevel for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.38.3.3 getTyreCondition()

```
virtual bool* Refueller::getTyreCondition ( ) [inline], [virtual]
```

Get the tyreCondition for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.38.3.4 refuel()

```
void Refueller::refuel ( ) [inline]
```

Refuel the car and notify the manager

4.38.3.5 setDamage()

```
virtual void Refueller::setDamage (
    bool status ) [inline], [virtual]
```

Set the damage for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.38.3.6 setFuelLevel()

```
virtual void Refueller::setFuelLevel (
    bool status ) [inline], [virtual]
```

Set the fuelLevel for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.38.3.7 setTyreCondition()

```
virtual void Refueller::setTyreCondition (
    bool * status ) [inline], [virtual]
```

Set the tyreCondition for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.38.3.8 update()

```
virtual void Refueller::update (
    bool * tyreCondition,
    bool fuelLevel,
    bool damage ) [inline], [virtual]
```

Check if there is a problem with the car and change the state accordingly and notify other members

Parameters

<i>tyreCondition</i>	
<i>fuelLevel</i>	
<i>damage</i>	

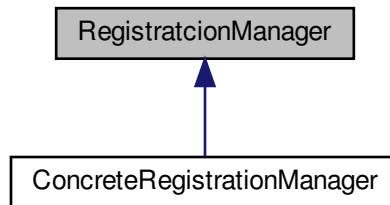
Implements [PitCrew](#).

The documentation for this class was generated from the following file:

- [Refueller.h](#)

4.39 RegistratcionManager Class Reference

Inheritance diagram for RegistratcionManager:



Public Member Functions

- virtual void [addCar](#) ([Car](#) *_car, int track)=0
- virtual void [addTrack](#) ([RaceTrackComponent](#) *_racetrack)=0
- virtual vector< [Car](#) * > [getCars](#) (int racetrack)=0
- virtual [RaceTrackComponent](#) * [getTrack](#) (int trackNo)=0

4.39.1 Member Function Documentation

4.39.1.1 addCar()

```
virtual void RegistratcionManager::addCar (  
    Car * _car,  
    int track ) [pure virtual]
```

abstract function to add cars into mediator

Implemented in [ConcreteRegistrationManager](#).

4.39.1.2 addTrack()

```
virtual void RegistratcionManager::addTrack (  
    RaceTrackComponent * _racetrack ) [pure virtual]
```

abstract function to add trackComponents into mediator

Implemented in [ConcreteRegistrationManager](#).

4.39.1.3 getCars()

```
virtual vector<Car*> RegistratcionManager::getCars (  
    int racetrack ) [pure virtual]
```

returns the cars on a certain racetrack

Parameters

<i>racetrack</i>	
------------------	--

Returns

Implemented in [ConcreteRegistrationManager](#).

4.39.1.4 getTrack()

```
virtual RaceTrackComponent* RegistratcionManager::getTrack (
    int trackNo ) [pure virtual]
```

returns the race track for a track number

Parameters

<i>trackNo</i>	
----------------	--

Returns

Implemented in [ConcreteRegistrationManager](#).

The documentation for this class was generated from the following file:

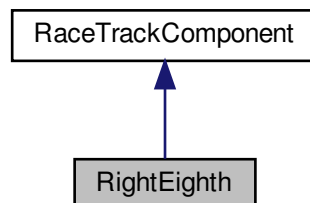
- RegistrationManager.h

4.40 RightEighth Class Reference

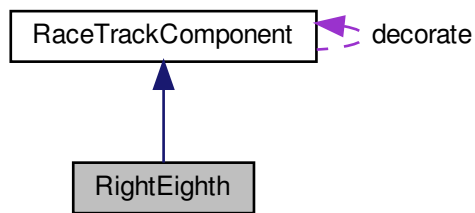
Leaf for Composite design pattern.

```
#include <RightEighth.h>
```

Inheritance diagram for RightEighth:



Collaboration diagram for RightEighth:



Public Member Functions

- [RightEighth](#) ()
- virtual [~RightEighth](#) ()
- virtual void [add](#) ([RaceTrackComponent](#) *R)
- virtual void [print](#) ()
- int [getAverageTime](#) ()
- virtual void [accept](#) ([BigBrother](#) *v)
- virtual void [addTime](#) ()

Additional Inherited Members

4.40.1 Detailed Description

Leaf for Composite design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.40.2 Constructor & Destructor Documentation

4.40.2.1 RightEighth()

```
RightEighth::RightEighth ( ) [inline]
```

constructor calls [RaceTrackComponent](#) and sets description

4.40.2.2 ~RightEighth()

```
virtual RightEighth::~~RightEighth ( ) [inline], [virtual]
```

destructor

4.40.3 Member Function Documentation

4.40.3.1 accept()

```
virtual void RightEighth::accept (
    BigBrother * v ) [inline], [virtual]
```

accepts the visitor to go to the correct part of the visitor

Parameters

<i>v</i>	
----------	--

Implements [RaceTrackComponent](#).

4.40.3.2 add()

```
virtual void RightEighth::add (
    RaceTrackComponent * R ) [inline], [virtual]
```

implementation of add function, used by decorator

Parameters

<i>R</i>	
----------	--

Implements [RaceTrackComponent](#).

4.40.3.3 addTime()

```
virtual void RightEighth::addTime ( ) [inline], [virtual]
```

adds the time and fuel and tyre conditions to the car

Implements [RaceTrackComponent](#).

4.40.3.4 `getAverageTime()`

```
int RightEighth::getAverageTime ( ) [inline]
```

returns the average time for the track

Returns

4.40.3.5 `print()`

```
virtual void RightEighth::print ( ) [inline], [virtual]
```

prints the description of the race track component /with decorators if it has

Implements [RaceTrackComponent](#).

The documentation for this class was generated from the following file:

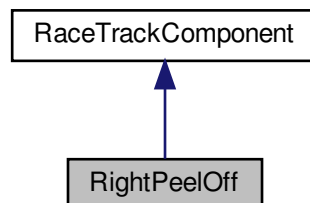
- [RightEighth.h](#)

4.41 RightPeelOff Class Reference

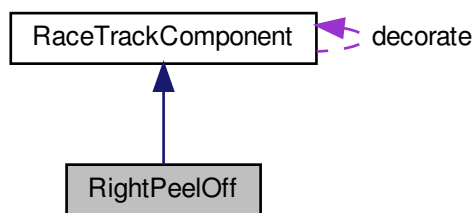
Leaf for Composite design pattern.

```
#include <RightPeelOff.h>
```

Inheritance diagram for RightPeelOff:



Collaboration diagram for RightPeelOff:



Public Member Functions

- [RightPeelOff](#) ()
- virtual [~RightPeelOff](#) ()
- virtual void [add](#) ([RaceTrackComponent](#) *R)
- virtual void [print](#) ()
- int [getAverageTime](#) ()
- virtual void [accept](#) ([BigBrother](#) *v)
- virtual void [addTime](#) ()

Additional Inherited Members

4.41.1 Detailed Description

Leaf for Composite design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.41.2 Constructor & Destructor Documentation

4.41.2.1 RightPeelOff()

```
RightPeelOff::RightPeelOff ( ) [inline]
```

constructor calls [RaceTrackComponent](#) and sets description

4.41.2.2 ~RightPeelOff()

```
virtual RightPeelOff::~~RightPeelOff ( ) [inline], [virtual]
```

destructor

4.41.3 Member Function Documentation

4.41.3.1 accept()

```
virtual void RightPeelOff::accept (
    BigBrother * v ) [inline], [virtual]
```

accepts the visitor to go to the correct part of the visitor

Parameters

<i>v</i>	
----------	--

Implements [RaceTrackComponent](#).

4.41.3.2 add()

```
virtual void RightPeelOff::add (
    RaceTrackComponent * R ) [inline], [virtual]
```

implementation of add function, used by decorator

Parameters

<i>R</i>	
----------	--

Implements [RaceTrackComponent](#).

4.41.3.3 addTime()

```
virtual void RightPeelOff::addTime ( ) [inline], [virtual]
```

adds the time and fuel and tyre conditions to the car

Implements [RaceTrackComponent](#).

4.41.3.4 getAverageTime()

```
int RightPeelOff::getAverageTime ( ) [inline]
```

returns the average time for the track

Returns

4.41.3.5 print()

```
virtual void RightPeelOff::print ( ) [inline], [virtual]
```

prints the description of the race track component /with decorators if it has

Implements [RaceTrackComponent](#).

The documentation for this class was generated from the following file:

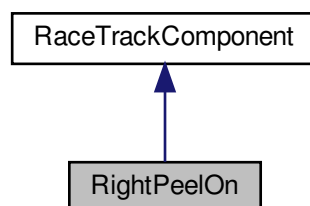
- [RightPeelOff.h](#)

4.42 RightPeelOn Class Reference

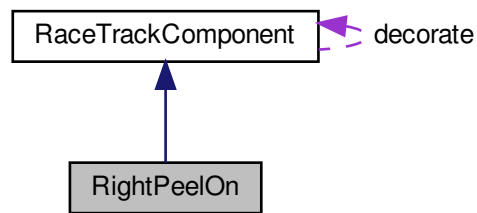
Leaf for Composite design pattern.

```
#include <RightPeelOn.h>
```

Inheritance diagram for RightPeelOn:



Collaboration diagram for RightPeelOn:



Public Member Functions

- [RightPeelOn](#) ()
- virtual [~RightPeelOn](#) ()
- virtual void [add](#) ([RaceTrackComponent](#) *R)
- virtual void [print](#) ()
- int [getAverageTime](#) ()
- virtual void [accept](#) ([BigBrother](#) *v)
- virtual void [addTime](#) ()

Additional Inherited Members

4.42.1 Detailed Description

Leaf for Composite design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.42.2 Constructor & Destructor Documentation

4.42.2.1 RightPeelOn()

```
RightPeelOn::RightPeelOn ( ) [inline]
```

constructor calls [RaceTrackComponent](#) and sets description

4.42.2.2 ~RightPeelOn()

```
virtual RightPeelOn::~~RightPeelOn ( ) [inline], [virtual]
```

destructor

4.42.3 Member Function Documentation

4.42.3.1 accept()

```
virtual void RightPeelOn::accept (
    BigBrother * v ) [inline], [virtual]
```

accepts the visitor to go to the correct part of the visitor

Parameters

<i>v</i>	
----------	--

Implements [RaceTrackComponent](#).

4.42.3.2 add()

```
virtual void RightPeelOn::add (
    RaceTrackComponent * R ) [inline], [virtual]
```

implementation of add function, used by decorator

Parameters

<i>R</i>	
----------	--

Implements [RaceTrackComponent](#).

4.42.3.3 addTime()

```
virtual void RightPeelOn::addTime ( ) [inline], [virtual]
```

adds the time and fuel and tyre conditions to the car

Implements [RaceTrackComponent](#).

4.42.3.4 `getAverageTime()`

```
int RightPeelOn::getAverageTime ( ) [inline]
```

returns the average time for the track

Returns

4.42.3.5 `print()`

```
virtual void RightPeelOn::print ( ) [inline], [virtual]
```

prints the description of the race track component /with decorators if it has

Implements [RaceTrackComponent](#).

The documentation for this class was generated from the following file:

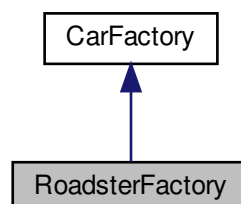
- [RightPeelOn.h](#)

4.43 RoadsterFactory Class Reference

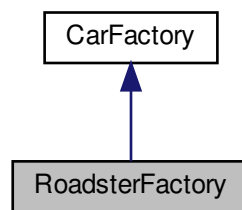
Concrete Factory for Abstract Factory Pattern.

```
#include <RoadsterFactory.h>
```

Inheritance diagram for RoadsterFactory:



Collaboration diagram for RoadsterFactory:



Public Member Functions

- virtual [ElectricCar](#) * [produceElectric](#) ()
- virtual [SportsCar](#) * [produceSports](#) ()
- virtual [StandardCar](#) * [produceStandard](#) ()

4.43.1 Detailed Description

Concrete Factory for Abstract Factory Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.43.2 Member Function Documentation

4.43.2.1 [produceElectric\(\)](#)

```
virtual ElectricCar* RoadsterFactory::produceElectric ( ) [inline], [virtual]
```

Implemented Function to produce an [ElectricCar](#)

Returns

[ElectricCar](#)*

Implements [CarFactory](#).

4.43.2.2 produceSports()

```
virtual SportsCar* RoadsterFactory::produceSports ( ) [inline], [virtual]
```

Implemented Function to produce an [SportsCar](#)

Returns

[SportsCar](#)*

Implements [CarFactory](#).

4.43.2.3 produceStandard()

```
virtual StandardCar* RoadsterFactory::produceStandard ( ) [inline], [virtual]
```

Implemented Function to produce an [StandardCar](#)

Returns

[StandardCar](#)*

Implements [CarFactory](#).

The documentation for this class was generated from the following file:

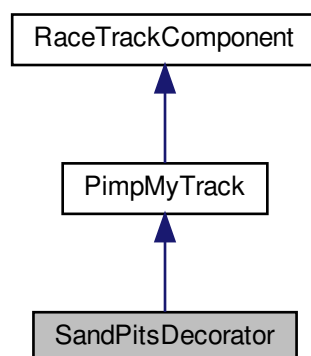
- [RoadsterFactory.h](#)

4.44 SandPitsDecorator Class Reference

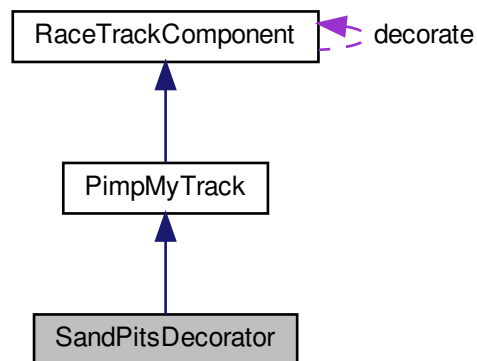
ConcreteDecorator for Decorator design pattern.

```
#include <SandPitsDecorator.h>
```

Inheritance diagram for SandPitsDecorator:



Collaboration diagram for SandPitsDecorator:



Public Member Functions

- [SandPitsDecorator\(\)](#)
- [~SandPitsDecorator\(\)](#)

Additional Inherited Members

4.44.1 Detailed Description

ConcreteDecorator for Decorator design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.44.2 Constructor & Destructor Documentation

4.44.2.1 SandPitsDecorator()

```
SandPitsDecorator::SandPitsDecorator ( ) [inline]
```

Constructor that calls constructor of pimpMyTrack and has a description

4.44.2.2 ~SandPitsDecorator()

```
SandPitsDecorator::~SandPitsDecorator ( ) [inline]
```

destructor

The documentation for this class was generated from the following file:

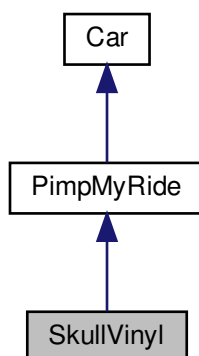
- [SandPitsDecorator.h](#)

4.45 SkullVinyl Class Reference

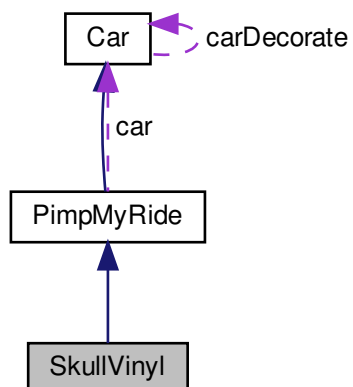
Concrete Decorator for Decorator Pattern.

```
#include <SkullVinyl.h>
```

Inheritance diagram for SkullVinyl:



Collaboration diagram for SkullVinyl:



Public Member Functions

- [SkullVinyl](#) ()
- [~SkullVinyl](#) ()
- [SkullVinyl](#) ([SkullVinyl](#) _Car, bool dummy)
- virtual [Car](#) * [FullClone](#) ()

Additional Inherited Members

4.45.1 Detailed Description

Concrete Decorator for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.45.2 Constructor & Destructor Documentation

4.45.2.1 [SkullVinyl\(\)](#) [1/2]

```
SkullVinyl::SkullVinyl ( ) [inline]
```

constructor to set description

4.45.2.2 [~SkullVinyl\(\)](#)

```
SkullVinyl::~~SkullVinyl ( ) [inline]
```

destructor to delete the vinyl

4.45.2.3 [SkullVinyl\(\)](#) [2/2]

```
SkullVinyl::SkullVinyl (
    SkullVinyl _Car,
    bool dummy ) [inline]
```

copy constructor used for cloning the decorators

Parameters

<code>_Car</code>	the car it copies
<code>dummy</code>	just there to use instead of default constructor

4.45.3 Member Function Documentation

4.45.3.1 FullClone()

```
virtual Car* SkullVinyl::FullClone ( ) [inline], [virtual]
```

implementation of FullClone to deep copy the decorater

Returns

[Car](#) object which is the decorator

Reimplemented from [PimpMyRide](#).

The documentation for this class was generated from the following file:

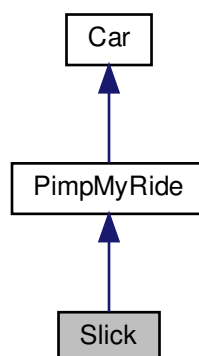
- [SkullVinyl.h](#)

4.46 Slick Class Reference

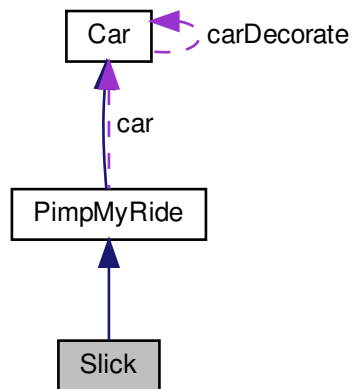
Concrete Decorator for Decorator Pattern.

```
#include <Slick.h>
```

Inheritance diagram for Slick:



Collaboration diagram for Slick:



Public Member Functions

- [Slick](#) ([Car](#) *DecorateCar)
- [~Slick](#) ()
- [Slick](#) ([Slick](#) _Car, bool dummy)
- virtual [Car](#) * [FullClone](#) ()

Additional Inherited Members

4.46.1 Detailed Description

Concrete Decorator for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.46.2 Constructor & Destructor Documentation

4.46.2.1 [Slick\(\)](#) [1/2]

```

Slick::Slick (
    Car * DecorateCar ) [inline]
  
```

constructor which assigns description and alters behaviour of car

Parameters

<i>Decorate</i>	the car in which the behaviours are added
-----------------	---

4.46.2.2 ~Slick()

```
Slick::~~Slick ( ) [inline]
```

destructor for [Slick](#)

4.46.2.3 Slick() [2/2]

```
Slick::Slick (
    Slick _Car,
    bool dummy ) [inline]
```

copy constructor used for cloning the decorators

Parameters

<i>_Car</i>	the car it copies
<i>dummy</i>	just there to use instead of default constructor

4.46.3 Member Function Documentation**4.46.3.1 FullClone()**

```
virtual Car* Slick::FullClone ( ) [inline], [virtual]
```

implementation of FullClone to deep copy the decorator

Returns

[Car](#) object which is the decorator

Reimplemented from [PimpMyRide](#).

The documentation for this class was generated from the following file:

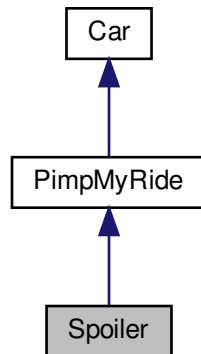
- [Slick.h](#)

4.47 Spoiler Class Reference

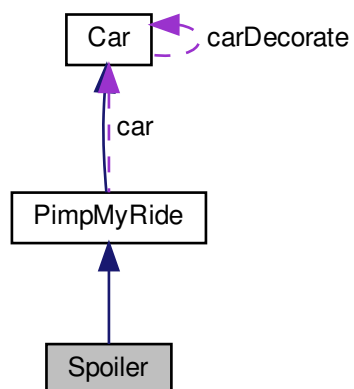
Concrete Decorator for Decorator Pattern.

```
#include <Spoiler.h>
```

Inheritance diagram for Spoiler:



Collaboration diagram for Spoiler:



Public Member Functions

- [Spoiler](#) ([Car](#) *Decorate)
- [~Spoiler](#) ()
- [Spoiler](#) ([Spoiler](#) _Car, bool dummy)
- virtual [Car](#) * [FullClone](#) ()

Additional Inherited Members

4.47.1 Detailed Description

Concrete Decorator for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.47.2 Constructor & Destructor Documentation

4.47.2.1 Spoiler() [1/2]

```
Spoiler::Spoiler (
    Car * Decorate ) [inline]
```

constructor which assigns description and alters behaviour of car

Parameters

<i>Decorate</i>	the car in which the behaviours are added
-----------------	---

4.47.2.2 ~Spoiler()

```
Spoiler::~~Spoiler ( ) [inline]
```

destructor for [Spoiler](#)

4.47.2.3 Spoiler() [2/2]

```
Spoiler::Spoiler (
    Spoiler _Car,
    bool dummy ) [inline]
```

copy constructor used for cloning the decorators

Parameters

<code>_Car</code>	the car it copies
<code>dummy</code>	just there to use instead of default constructor

4.47.3 Member Function Documentation

4.47.3.1 FullClone()

```
virtual Car* Spoiler::FullClone ( ) [inline], [virtual]
```

implementation of FullClone to deep copy the decorator

Returns

[Car](#) object which is the decorator

Reimplemented from [PimpMyRide](#).

The documentation for this class was generated from the following file:

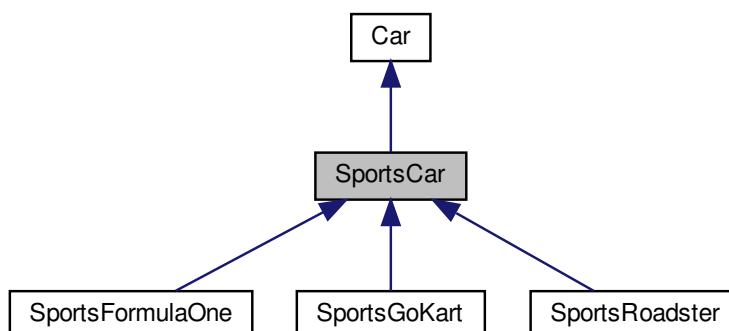
- [Spoiler.h](#)

4.48 SportsCar Class Reference

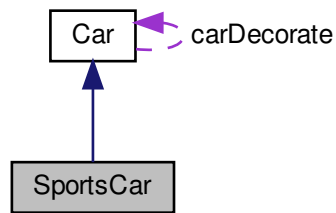
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <SportsCar.h>
```

Inheritance diagram for SportsCar:



Collaboration diagram for SportsCar:



Public Member Functions

- [SportsCar](#) (string modelType_)
- [SportsCar](#) (const [Car](#) &car_, bool flag_)
- virtual [~SportsCar](#) ()
- virtual string [getDescription](#) ()
- virtual [Car](#) * [clone](#) (bool flag_)
- virtual [Car](#) * [FullClone](#) ()

Additional Inherited Members

4.48.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.48.2 Constructor & Destructor Documentation

4.48.2.1 [SportsCar](#)() [1/2]

```
SportsCar::SportsCar (
    string modelType_ )
```

Constructor for [SportsCar](#)

Parameters

<i>model</i> ↵ <i>Type_</i>	states whether car is Electric/Sports/Standard
--------------------------------	--

4.48.2.2 SportsCar() [2/2]

```
SportsCar::SportsCar (
    const Car & car_,
    bool flag_ )
```

The copy constructor for [SportsCar](#)

Parameters

<i>car</i> ↵ —	is a Car object that will be copied
-------------------	---

4.48.2.3 ~SportsCar()

```
virtual SportsCar::~~SportsCar ( ) [inline], [virtual]
```

The virtual destructor for [SportsCar](#)

4.48.3 Member Function Documentation

4.48.3.1 clone()

```
virtual Car* SportsCar::clone (
    bool flag_ ) [inline], [virtual]
```

a abstract clone function for the prototype design pattern

Returns

a pointer to car object

Implements [Car](#).

Reimplemented in [SportsRoadster](#), [SportsFormulaOne](#), and [SportsGoKart](#).

4.48.3.2 FullClone()

```
virtual Car* SportsCar::FullClone ( ) [inline], [virtual]
```

implementation of Fullclone in [Car](#)

Returns

[Car](#) object with all decorated

Implements [Car](#).

4.48.3.3 getDescription()

```
string SportsCar::getDescription ( ) [virtual]
```

a getDescription Function

Returns

a string that states the info about the car

The documentation for this class was generated from the following files:

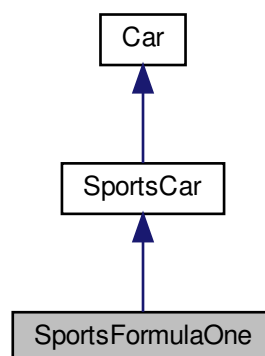
- [SportsCar.h](#)
- SportsCar.cpp

4.49 SportsFormulaOne Class Reference

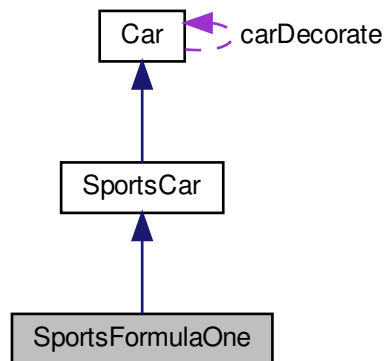
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <SportsFormulaOne.h>
```

Inheritance diagram for SportsFormulaOne:



Collaboration diagram for SportsFormulaOne:



Public Member Functions

- [SportsFormulaOne](#) ()
- [SportsFormulaOne](#) (const [Car](#) &car_, bool flag_)
- virtual [Car](#) * [clone](#) (bool flag_=false)

Additional Inherited Members

4.49.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.49.2 Constructor & Destructor Documentation

4.49.2.1 SportsFormulaOne() [1/2]

```
SportsFormulaOne::SportsFormulaOne ( ) [inline]
```

Constructor for [SportsFormulaOne](#), calls Constructor of [SportsCar](#)

4.49.2.2 SportsFormulaOne() [2/2]

```
SportsFormulaOne::SportsFormulaOne (
    const Car & car_,
    bool flag_ ) [inline]
```

Copy constructor used for cloning

Parameters

<i>car</i> ↔ —	car object for copying
<i>flag</i> ↔ —	to determine if must be full clone or basic clone

4.49.3 Member Function Documentation

4.49.3.1 clone()

```
virtual Car* SportsFormulaOne::clone (
    bool flag_ = false ) [inline], [virtual]
```

implementation of clone function

Parameters

<i>flag</i> ↔ —	determines if must be full clone or basic clone
--------------------	---

Returns

a copied car object

Reimplemented from [SportsCar](#).

The documentation for this class was generated from the following file:

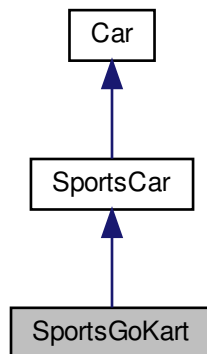
- [SportsFormulaOne.h](#)

4.50 SportsGoKart Class Reference

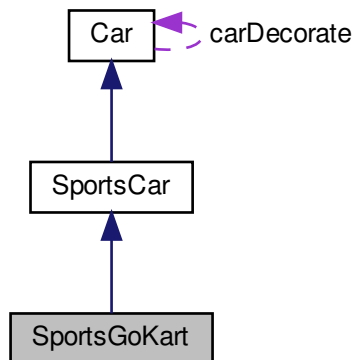
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <SportsGoKart.h>
```

Inheritance diagram for SportsGoKart:



Collaboration diagram for SportsGoKart:



Public Member Functions

- [SportsGoKart](#) ()
- [SportsGoKart](#) (const [Car](#) &car_, bool flag_)
- virtual [Car](#) * [clone](#) (bool flag_=false)

Additional Inherited Members

4.50.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.50.2 Constructor & Destructor Documentation

4.50.2.1 SportsGoKart() [1/2]

```
SportsGoKart::SportsGoKart ( ) [inline]
```

Constructor for [SportsGoKart](#), calls Constructor of [SportsCar](#)

4.50.2.2 SportsGoKart() [2/2]

```
SportsGoKart::SportsGoKart (
    const Car & car_,
    bool flag_ ) [inline]
```

Copy constructor used for cloning

Parameters

<i>car</i> ↔ —	car object for copying
<i>flag</i> ↔ —	to determine if must be full clone or basic clone

4.50.3 Member Function Documentation

4.50.3.1 clone()

```
virtual Car* SportsGoKart::clone (
    bool flag_ = false ) [inline], [virtual]
```

implementation of clone function

Parameters

<i>flag</i> ↔	determines if must be full clone or basic clone
—	

Returns

a copied car object

Reimplemented from [SportsCar](#).

The documentation for this class was generated from the following file:

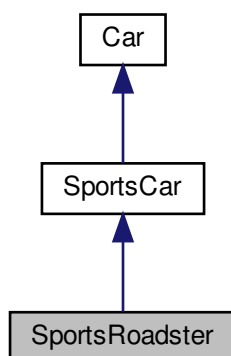
- [SportsGoKart.h](#)

4.51 SportsRoadster Class Reference

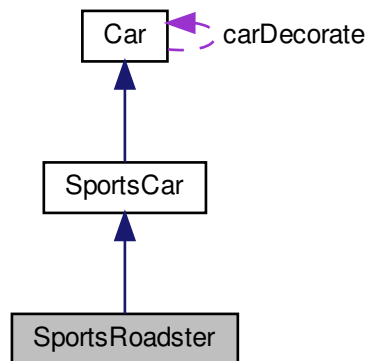
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <SportsRoadster.h>
```

Inheritance diagram for SportsRoadster:



Collaboration diagram for SportsRoadster:



Public Member Functions

- [SportsRoadster](#) ()
- [SportsRoadster](#) (const [Car](#) &car_, bool flag_)
- virtual [Car](#) * [clone](#) (bool flag_=false)

Additional Inherited Members

4.51.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.51.2 Constructor & Destructor Documentation

4.51.2.1 SportsRoadster() [1/2]

```
SportsRoadster::SportsRoadster ( ) [inline]
```

Constructor for [SportsRoadster](#), calls Constructor of [SportsCar](#)

4.51.2.2 SportsRoadster() [2/2]

```
SportsRoadster::SportsRoadster (
    const Car & car_,
    bool flag_ ) [inline]
```

Copy constructor used for cloning

Parameters

<i>car</i> ↔ —	car object for copying
<i>flag</i> ↔ —	to determine if must be full clone or basic clone

4.51.3 Member Function Documentation

4.51.3.1 clone()

```
virtual Car* SportsRoadster::clone (
    bool flag_ = false ) [inline], [virtual]
```

implementation of clone function

Parameters

<i>flag</i> ↔ —	determines if must be full clone or basic clone
--------------------	---

Returns

a copied car object

Reimplemented from [SportsCar](#).

The documentation for this class was generated from the following file:

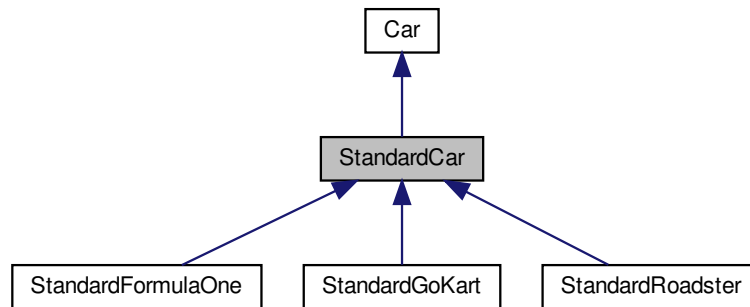
- [SportsRoadster.h](#)

4.52 StandardCar Class Reference

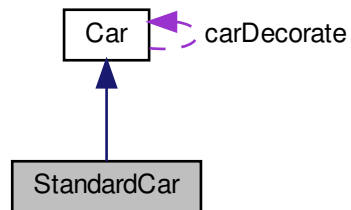
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <StandardCar.h>
```

Inheritance diagram for StandardCar:



Collaboration diagram for StandardCar:



Public Member Functions

- [StandardCar](#) (string modelType_)
- [StandardCar](#) (const [Car](#) &car_, bool flag_)
- virtual [~StandardCar](#) ()
- virtual string [getDescription](#) ()
- virtual [Car](#) * [clone](#) (bool flag_)
- virtual [Car](#) * [FullClone](#) ()

Additional Inherited Members

4.52.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.52.2 Constructor & Destructor Documentation

4.52.2.1 StandardCar() [1/2]

```
StandardCar::StandardCar (
    string modelType_ )
```

The base Constructor for [StandardCar](#)

Parameters

<i>modelType_</i>	states whether the car is FormulaOne/Roadster/GoKart
-------------------	--

4.52.2.2 StandardCar() [2/2]

```
StandardCar::StandardCar (
    const Car & car_,
    bool flag_ )
```

The copy constructor for [StandardCar](#)

Parameters

<i>car_</i>	is a Car object that will be copied
-------------	---

4.52.2.3 ~StandardCar()

```
virtual StandardCar::~~StandardCar ( ) [inline], [virtual]
```

The virtual destructor for [StandardCar](#)

4.52.3 Member Function Documentation

4.52.3.1 clone()

```
virtual Car* StandardCar::clone (
    bool flag_ ) [inline], [virtual]
```

a abstract clone function for the prototype design pattern

Returns

a [Car](#) object

Implements [Car](#).

Reimplemented in [StandardFormulaOne](#), [StandardGoKart](#), and [StandardRoadster](#).

4.52.3.2 FullClone()

```
virtual Car* StandardCar::FullClone ( ) [inline], [virtual]
```

implementation of Fullclone in [Car](#)

Returns

[Car](#) object with all decorated

Implements [Car](#).

4.52.3.3 getDescription()

```
string StandardCar::getDescription ( ) [virtual]
```

a getDescription Function

Returns

a string that states the info about the car

The documentation for this class was generated from the following files:

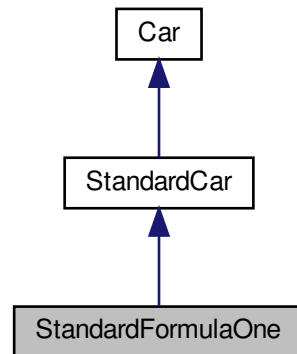
- [StandardCar.h](#)
- [StandardCar.cpp](#)

4.53 StandardFormulaOne Class Reference

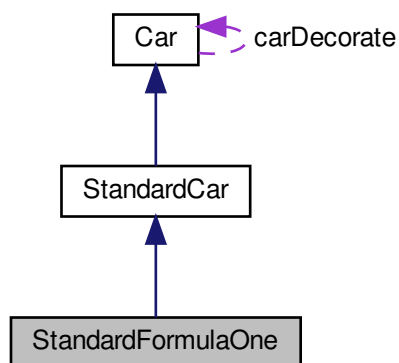
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <StandardFormulaOne.h>
```

Inheritance diagram for StandardFormulaOne:



Collaboration diagram for StandardFormulaOne:



Public Member Functions

- [StandardFormulaOne](#) ()
- [StandardFormulaOne](#) (const [Car](#) &car_, bool flag_)
- virtual [Car](#) * [clone](#) (bool flag_=false)

Additional Inherited Members

4.53.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.53.2 Constructor & Destructor Documentation

4.53.2.1 StandardFormulaOne() [1/2]

```
StandardFormulaOne::StandardFormulaOne ( ) [inline]
```

Constructor for [StandardFormulaOne](#), calls Constructor of [StandardCar](#)

4.53.2.2 StandardFormulaOne() [2/2]

```
StandardFormulaOne::StandardFormulaOne (
    const Car & car_,
    bool flag_ ) [inline]
```

Copy constructor used for cloning

Parameters

<i>car</i> ↔ —	car object for copying
<i>flag</i> ↔ —	to determine if must be full clone or basic clone

4.53.3 Member Function Documentation

4.53.3.1 clone()

```
virtual Car* StandardFormulaOne::clone (
    bool flag_ = false ) [inline], [virtual]
```


implementation of clone function

Parameters

<i>flag</i> ↔	determines if must be full clone or basic clone
—	

Returns

a copied car object

Reimplemented from [StandardCar](#).

The documentation for this class was generated from the following file:

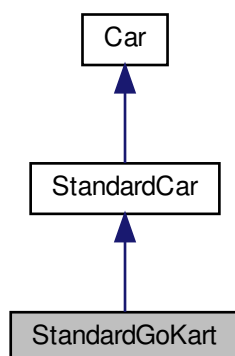
- [StandardFormulaOne.h](#)

4.54 StandardGoKart Class Reference

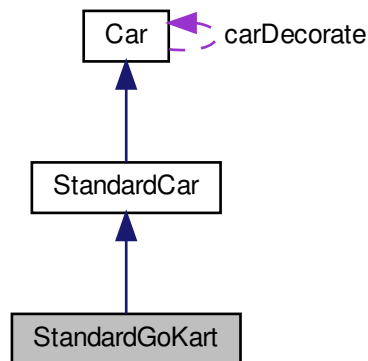
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <StandardGoKart.h>
```

Inheritance diagram for StandardGoKart:



Collaboration diagram for StandardGoKart:



Public Member Functions

- [StandardGoKart](#) ()
- [StandardGoKart](#) (const [Car](#) &car_, bool flag_)
- virtual [Car](#) * [clone](#) (bool flag_=false)

Additional Inherited Members

4.54.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.54.2 Constructor & Destructor Documentation

4.54.2.1 [StandardGoKart](#)() [1/2]

```
StandardGoKart::StandardGoKart ( ) [inline]
```

Constructor for [StandardGoKart](#), calls Constructor of [StandardCar](#)

4.54.2.2 StandardGoKart() [2/2]

```
StandardGoKart::StandardGoKart (
    const Car & car_,
    bool flag_ ) [inline]
```

Copy constructor used for cloning

Parameters

<i>car</i> ↔ —	car object for copying
<i>flag</i> ↔ —	to determine if must be full clone or basic clone

4.54.3 Member Function Documentation**4.54.3.1 clone()**

```
virtual Car* StandardGoKart::clone (
    bool flag_ = false ) [inline], [virtual]
```

implementation of clone function

Parameters

<i>flag</i> ↔ —	determines if must be full clone or basic clone
--------------------	---

Returns

a copied car object

Reimplemented from [StandardCar](#).

The documentation for this class was generated from the following file:

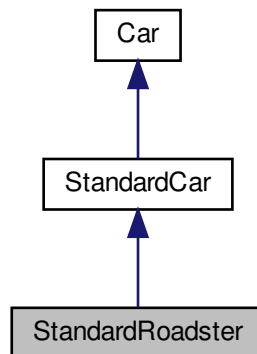
- [StandardGoKart.h](#)

4.55 StandardRoadster Class Reference

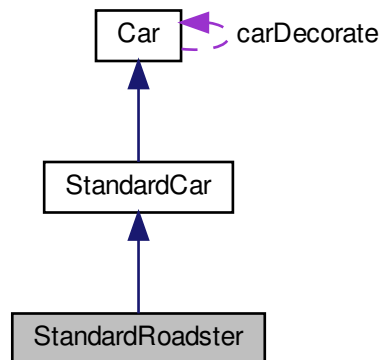
Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

```
#include <StandardRoadster.h>
```

Inheritance diagram for StandardRoadster:



Collaboration diagram for StandardRoadster:



Public Member Functions

- [StandardRoadster](#) ()
- [StandardRoadster](#) (const [Car](#) &car_, bool flag_)
- virtual [Car](#) * [clone](#) (bool flag_=false)

Additional Inherited Members

4.55.1 Detailed Description

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.55.2 Constructor & Destructor Documentation

4.55.2.1 StandardRoadster() [1/2]

```
StandardRoadster::StandardRoadster ( ) [inline]
```

Constructor for [StandardRoadster](#), calls Constructor of [StandardCar](#)

4.55.2.2 StandardRoadster() [2/2]

```
StandardRoadster::StandardRoadster (
    const Car & car_,
    bool flag_ ) [inline]
```

Copy constructor used for cloning

Parameters

<i>car</i> ↔ —	car object for copying
<i>flag</i> ↔ —	to determine if must be full clone or basic clone

4.55.3 Member Function Documentation

4.55.3.1 clone()

```
virtual Car* StandardRoadster::clone (
    bool flag_ = false ) [inline], [virtual]
```

implementation of clone function

Parameters

<i>flag</i> ↔	determines if must be full clone or basic clone
—	

Returns

a copied car object

Reimplemented from [StandardCar](#).

The documentation for this class was generated from the following file:

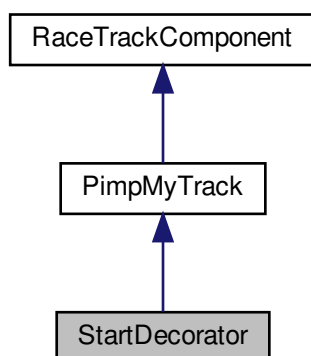
- [StandardRoadster.h](#)

4.56 StartDecorator Class Reference

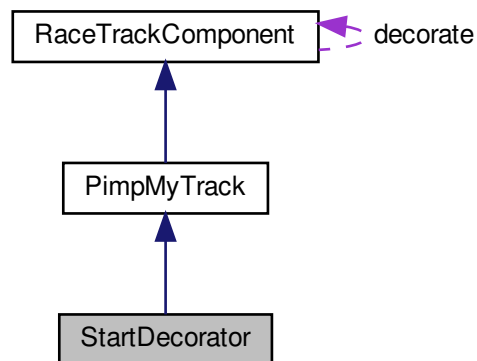
ConcreteDecorator for Decorator design pattern.

```
#include <StartDecorator.h>
```

Inheritance diagram for StartDecorator:



Collaboration diagram for StartDecorator:



Public Member Functions

- [StartDecorator](#) ()
- [~StartDecorator](#) ()

Additional Inherited Members

4.56.1 Detailed Description

ConcreteDecorator for Decorator design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.56.2 Constructor & Destructor Documentation

4.56.2.1 StartDecorator()

```
StartDecorator::StartDecorator ( ) [inline]
```

Constructor that calls constructor of `pimpMyTrack` and has a description

4.56.2.2 ~StartDecorator()

```
StartDecorator::~~StartDecorator ( ) [inline]
```

destructor

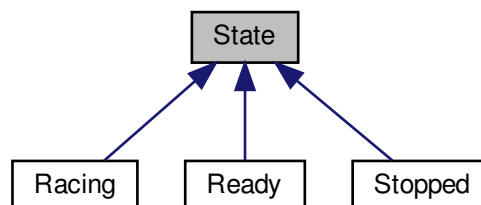
The documentation for this class was generated from the following file:

- [StartDecorator.h](#)

4.57 State Class Reference

```
#include <State.h>
```

Inheritance diagram for State:



Public Member Functions

- virtual void [ready](#) ([Car](#) *car)
- virtual void [racing](#) ([Car](#) *car)
- virtual void [stopped](#) ([Car](#) *car)
- virtual string [toString](#) ()=0

4.57.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.57.2 Member Function Documentation

4.57.2.1 racing()

```
virtual void State::racing (  
    Car * car ) [inline], [virtual]
```

Change car to racing state

Parameters

<i>car</i>	
------------	--

4.57.2.2 ready()

```
virtual void State::ready (
    Car * car ) [inline], [virtual]
```

Change car to ready state

Parameters

<i>car</i>	
------------	--

4.57.2.3 stopped()

```
virtual void State::stopped (
    Car * car ) [inline], [virtual]
```

Change car to stopped state

Parameters

<i>car</i>	
------------	--

4.57.2.4 toString()

```
virtual string State::toString ( ) [pure virtual]
```

Print the current state to the screen

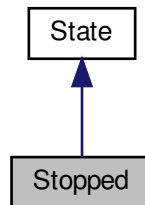
Returns

The documentation for this class was generated from the following file:

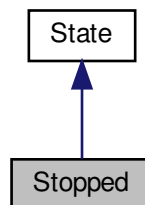
- [State.h](#)

4.58 Stopped Class Reference

Inheritance diagram for Stopped:



Collaboration diagram for Stopped:



Additional Inherited Members

The documentation for this class was generated from the following files:

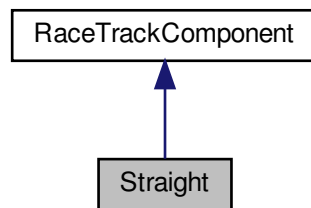
- [State.h](#)
- [State.cpp](#)

4.59 Straight Class Reference

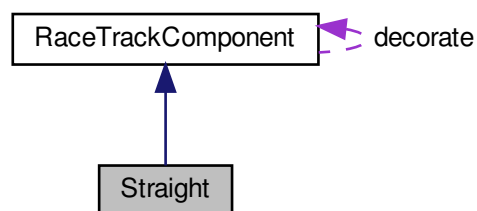
Leaf for Composite design pattern.

```
#include <Straight.h>
```

Inheritance diagram for Straight:



Collaboration diagram for Straight:



Public Member Functions

- [Straight](#) ()
- virtual [~Straight](#) ()
- virtual void [add](#) ([RaceTrackComponent](#) *R)
- virtual void [print](#) ()
- int [getAverageTime](#) ()
- virtual void [accept](#) ([BigBrother](#) *v)
- virtual void [addTime](#) ()

Additional Inherited Members

4.59.1 Detailed Description

Leaf for Composite design pattern.

Authors

Duncan + Tjaart

Version

1.0.0

4.59.2 Constructor & Destructor Documentation

4.59.2.1 Straight()

```
Straight::Straight ( ) [inline]
```

constructor calls [RaceTrackComponent](#) and sets description

4.59.2.2 ~Straight()

```
virtual Straight::~~Straight ( ) [inline], [virtual]
```

destructor

4.59.3 Member Function Documentation

4.59.3.1 accept()

```
virtual void Straight::accept (
    BigBrother * v ) [inline], [virtual]
```

accepts the visitor to go to the correct part of the visitor

Parameters

<i>v</i>	
----------	--

Implements [RaceTrackComponent](#).

4.59.3.2 add()

```
virtual void Straight::add (
    RaceTrackComponent * R ) [inline], [virtual]
```

implementation of add function, used by decorator

Parameters

<i>R</i>	
----------	--

Implements [RaceTrackComponent](#).

4.59.3.3 addTime()

```
virtual void Straight::addTime ( ) [inline], [virtual]
```

adds the time and fuel and tyre conditions to the car

Implements [RaceTrackComponent](#).

4.59.3.4 getAverageTime()

```
int Straight::getAverageTime ( ) [inline]
```

returns the average time for the track

Returns

4.59.3.5 print()

```
virtual void Straight::print ( ) [inline], [virtual]
```

prints the description of the race track component /with decorators if it has

Implements [RaceTrackComponent](#).

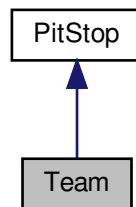
The documentation for this class was generated from the following file:

- [Straight.h](#)

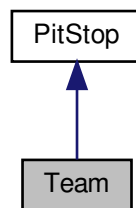
4.60 Team Class Reference

```
#include <Team.h>
```

Inheritance diagram for Team:



Collaboration diagram for Team:



Public Member Functions

- [Team](#) (string name)
- virtual void [getCarStats](#) ()

4.60.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.60.2 Constructor & Destructor Documentation

4.60.2.1 Team()

```
Team::Team (
    string name )
```

default constructor for when a team is created, calls pitstop constructor

Parameters

<i>name</i> ↔	the name of the team
—	

4.60.3 Member Function Documentation

4.60.3.1 getCarStats()

```
void Team::getCarStats ( ) [virtual]
```

Function to get the car stats

Implements [PitStop](#).

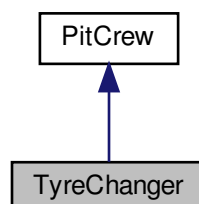
The documentation for this class was generated from the following files:

- [Team.h](#)
- Team.cpp

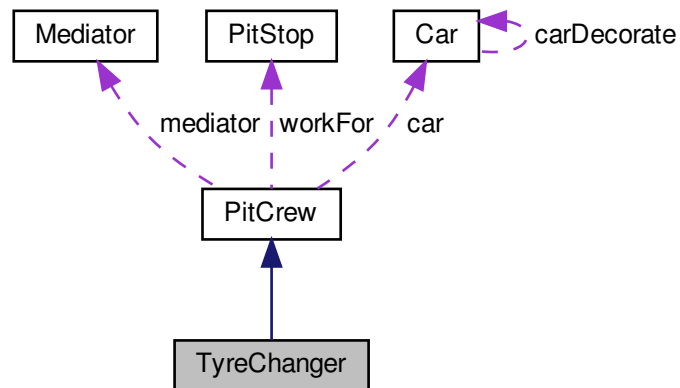
4.61 TyreChanger Class Reference

```
#include <TyreChanger.h>
```

Inheritance diagram for TyreChanger:



Collaboration diagram for TyreChanger:



Public Member Functions

- [TyreChanger](#) ([Mediator](#) *med, int id_, [Car](#) *car)
- virtual bool * [getTyreCondition](#) ()
- virtual void [setTyreCondition](#) (bool *status)
- virtual bool [getFuelLevel](#) ()
- virtual void [setFuelLevel](#) (bool status)
- virtual bool [getDamage](#) ()
- virtual void [setDamage](#) (bool status)
- virtual void [update](#) (bool *tyreCondition, bool fuelLevel, bool damage)
- void [changeTyre](#) ()

Additional Inherited Members

4.61.1 Detailed Description

Authors

Duncan + Tjaart

Version

1.0.0

4.61.2 Constructor & Destructor Documentation

4.61.2.1 TyreChanger()

```
TyreChanger::TyreChanger (
    Mediator * med,
    int id_,
    Car * car ) [inline]
```

Constructor for the [Refueller](#)

Parameters

<i>med</i>	- Mediator for the team
<i>car</i>	- Car for the team

4.61.3 Member Function Documentation

4.61.3.1 changeTyre()

```
void TyreChanger::changeTyre ( ) [inline]
```

Change the tyre of the car and notify the manager

4.61.3.2 getDamage()

```
virtual bool TyreChanger::getDamage ( ) [inline], [virtual]
```

Get the damage for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.61.3.3 getFuelLevel()

```
virtual bool TyreChanger::getFuelLevel ( ) [inline], [virtual]
```

Get the fuelLevel for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.61.3.4 getTyreCondition()

```
virtual bool* TyreChanger::getTyreCondition ( ) [inline], [virtual]
```

Get the tyreCondition for the car

Returns

bool saying if there is a problem or not

Reimplemented from [PitCrew](#).

4.61.3.5 setDamage()

```
virtual void TyreChanger::setDamage (
    bool status ) [inline], [virtual]
```

Set the damage for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.61.3.6 setFuelLevel()

```
virtual void TyreChanger::setFuelLevel (
    bool status ) [inline], [virtual]
```

Set the fuelLevel for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.61.3.7 setTyreCondition()

```
virtual void TyreChanger::setTyreCondition (
    bool * status ) [inline], [virtual]
```

Set the tyreCondition for the car

Parameters

<i>bool</i>	saying if there is a problem or not
-------------	-------------------------------------

Reimplemented from [PitCrew](#).

4.61.3.8 update()

```
virtual void TyreChanger::update (  
    bool * tyreCondition,  
    bool fuelLevel,  
    bool damage ) [inline], [virtual]
```

Check if there is a problem with the car and change the state accordingly and notify other members

Parameters

<i>tyreCondition</i>	
<i>fuelLevel</i>	
<i>damage</i>	

Implements [PitCrew](#).

The documentation for this class was generated from the following file:

- [TyreChanger.h](#)

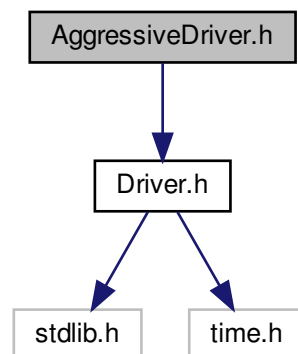
Chapter 5

File Documentation

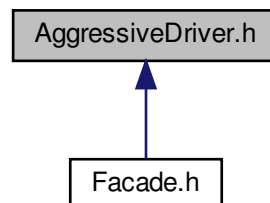
5.1 AggressiveDriver.h File Reference

```
#include "Driver.h"
```

Include dependency graph for AggressiveDriver.h:



This graph shows which files directly or indirectly include this file:

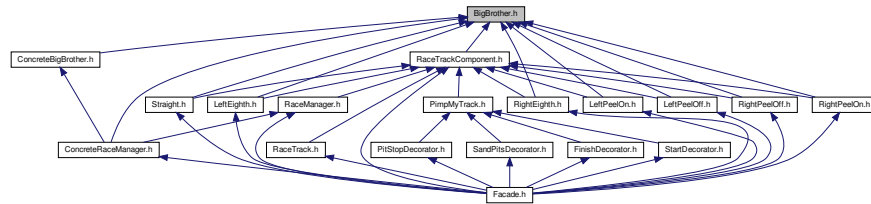


Classes

- class [AggressiveDriver](#)
concreteStrategy for strategy design pattern

5.2 BigBrother.h File Reference

This graph shows which files directly or indirectly include this file:



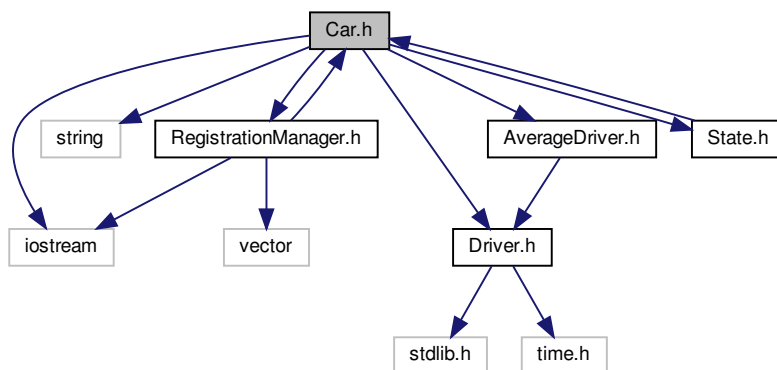
Classes

- class [BigBrother](#)
visitor class in visitor pattern

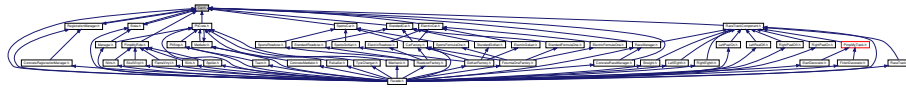
5.3 Car.h File Reference

```
#include <iostream>
#include <string>
#include "RegistrationManager.h"
#include "Driver.h"
#include "AverageDriver.h"
#include "State.h"
```

Include dependency graph for `Car.h`:



This graph shows which files directly or indirectly include this file:



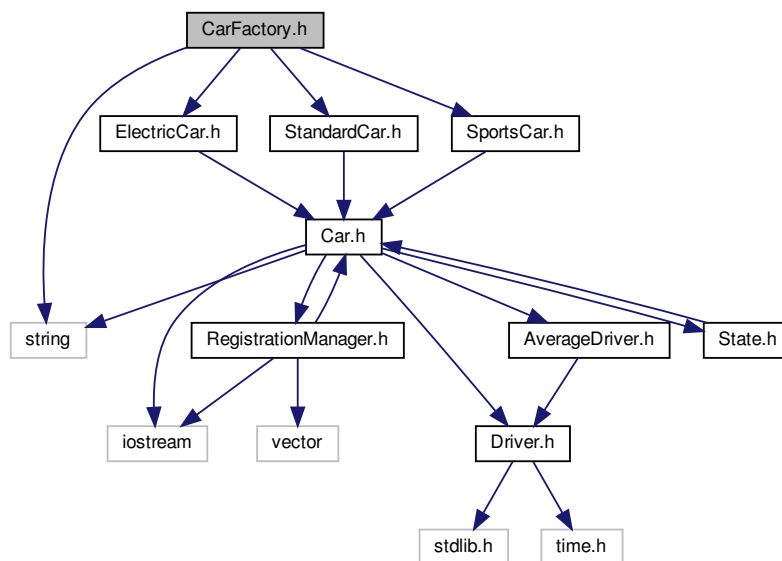
Classes

- class [Car](#)

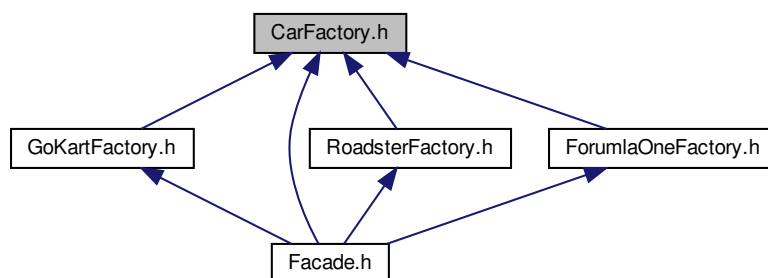
Abstract Product for Abstract Factory Pattern and Component for Decorator Pattern.

5.4 CarFactory.h File Reference

```
#include <string>
#include "ElectricCar.h"
#include "StandardCar.h"
#include "SportsCar.h"
Include dependency graph for CarFactory.h:
```



This graph shows which files directly or indirectly include this file:



Classes

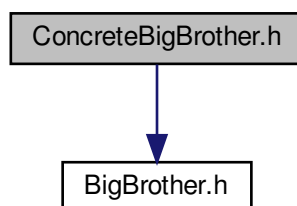
- class [CarFactory](#)

Abstract Factory for Abstract Factory Pattern.

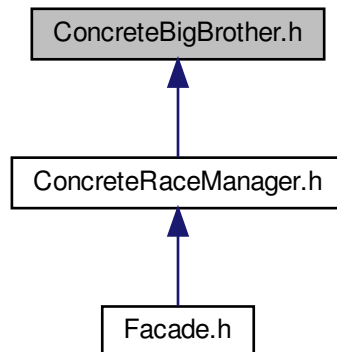
5.5 ConcreteBigBrother.h File Reference

```
#include "BigBrother.h"
```

Include dependency graph for ConcreteBigBrother.h:



This graph shows which files directly or indirectly include this file:



Classes

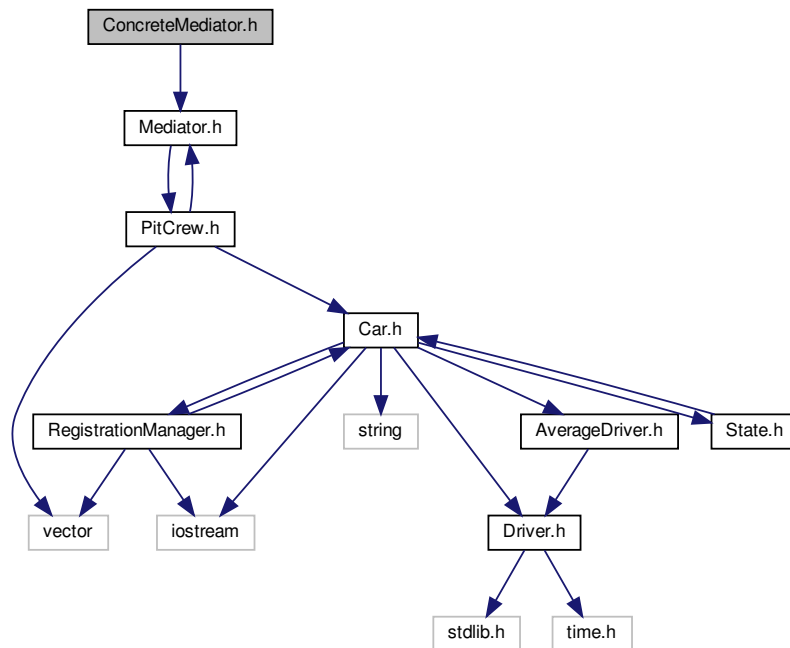
- class [ConcreteBigBrother](#)

Concrete visitor class in visitor pattern.

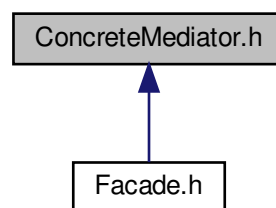
5.6 ConcreteMediator.h File Reference

```
#include "Mediator.h"
```

Include dependency graph for ConcreteMediator.h:



This graph shows which files directly or indirectly include this file:



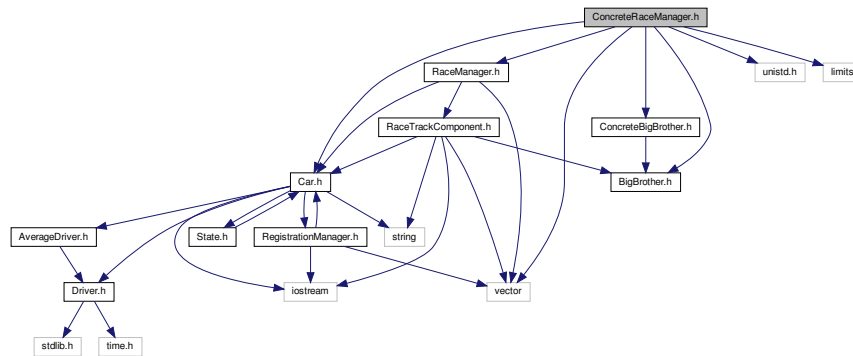
Classes

- class [ConcreteMediator](#)

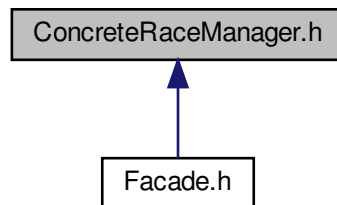
5.7 ConcreteRaceManager.h File Reference

```
#include "RaceManager.h"
#include "Car.h"
```

```
#include "BigBrother.h"
#include "ConcreteBigBrother.h"
#include <vector>
#include <unistd.h>
#include <limits>
Include dependency graph for ConcreteRaceManager.h:
```



This graph shows which files directly or indirectly include this file:



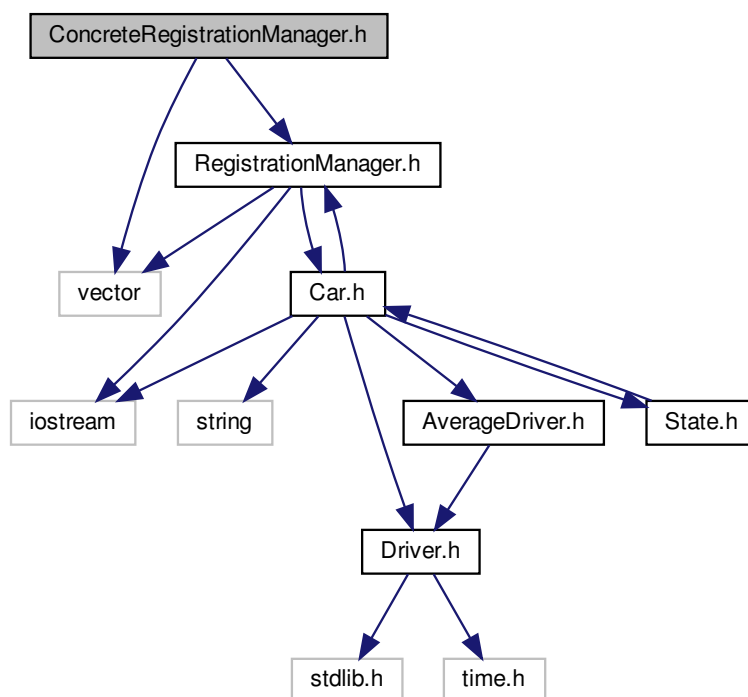
Classes

- class [ConcreteRaceManager](#)
concrete Observer in observer pattern

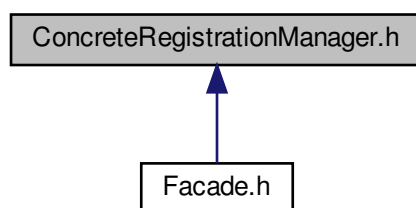
5.8 ConcreteRegistrationManager.h File Reference

```
#include <vector>
#include "RegistrationManager.h"
```

Include dependency graph for ConcreteRegistrationManager.h:



This graph shows which files directly or indirectly include this file:



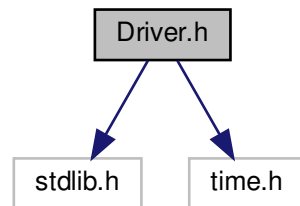
Classes

- class [ConcreteRegistrationManager](#)
ConcreteMediator for mediator design pattern.

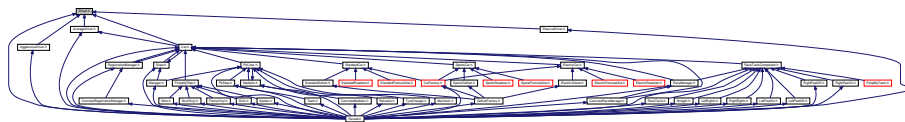
5.9 Driver.h File Reference

```
#include <stdlib.h>
#include <time.h>
```

Include dependency graph for Driver.h:



This graph shows which files directly or indirectly include this file:



Classes

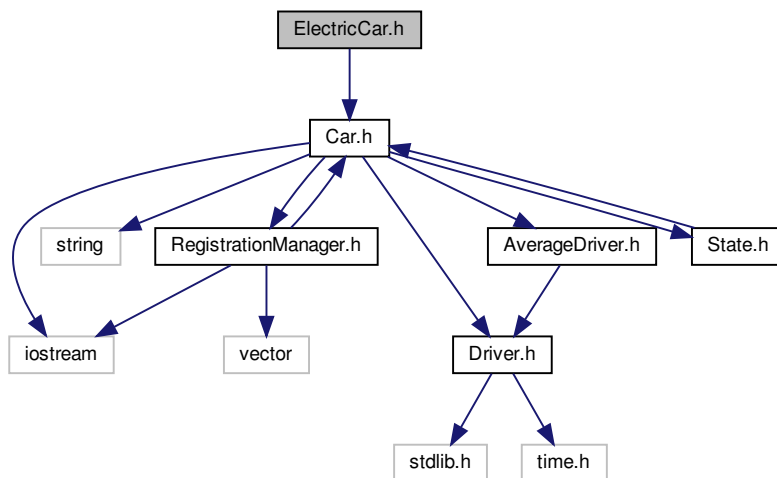
- class [Driver](#)

Stratey for strategy design pattern.

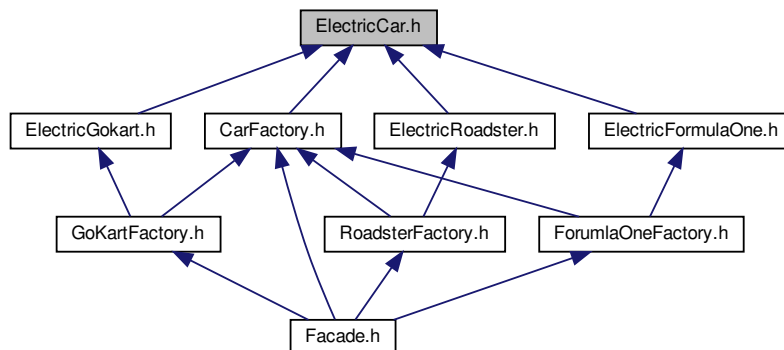
5.10 ElectricCar.h File Reference

```
#include "Car.h"
```

Include dependency graph for ElectricCar.h:



This graph shows which files directly or indirectly include this file:



Classes

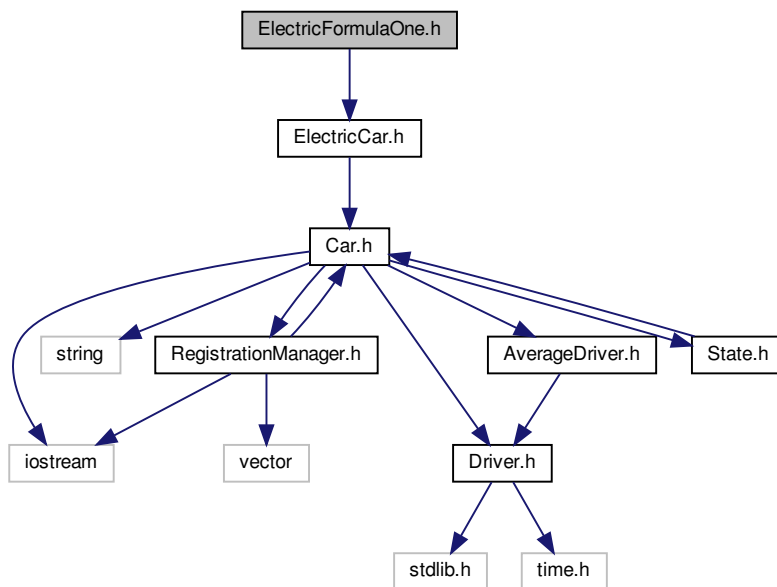
- class [ElectricCar](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

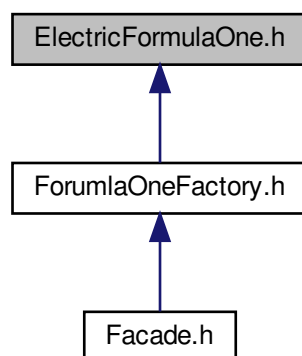
5.11 ElectricFormulaOne.h File Reference

```
#include "ElectricCar.h"
```

Include dependency graph for ElectricFormulaOne.h:



This graph shows which files directly or indirectly include this file:



Classes

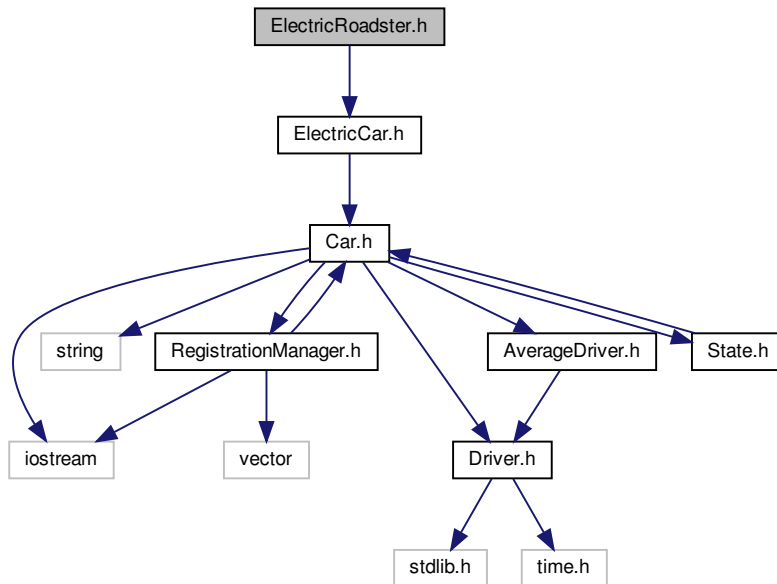
- class [ElectricFormulaOne](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

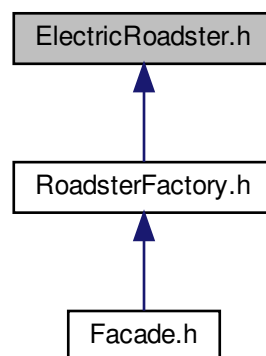
5.12 ElectricRoadster.h File Reference

```
#include "ElectricCar.h"
```

Include dependency graph for ElectricRoadster.h:



This graph shows which files directly or indirectly include this file:



Classes

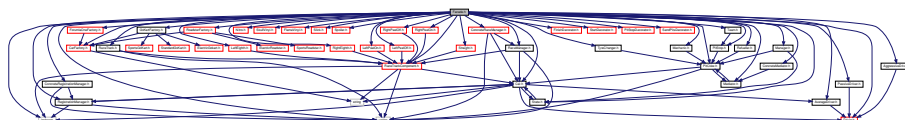
- class [ElectricRoadster](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

5.13 Facade.h File Reference

```
#include <iostream>
#include <vector>
#include "Car.h"
#include "GoKartFactory.h"
#include "RoadsterFactory.h"
#include "FormulaOneFactory.h"
#include "CarFactory.h"
#include "Nitro.h"
#include "SkullVinyl.h"
#include "FlameVinyl.h"
#include "Slick.h"
#include "Spoiler.h"
#include "RegistrationManager.h"
#include "ConcreteRegistrationManager.h"
#include "RaceTrackComponent.h"
#include "RaceTrack.h"
#include "Straight.h"
#include "LeftEighth.h"
#include "RightEighth.h"
#include "LeftPeelOn.h"
#include "LeftPeelOff.h"
#include "RightPeelOff.h"
#include "RightPeelOn.h"
#include "FinishDecorator.h"
#include "StartDecorator.h"
#include "PitStopDecorator.h"
#include "SandPitsDecorator.h"
#include "Team.h"
#include "PitStop.h"
#include "PitCrew.h"
#include "Refueller.h"
#include "TyreChanger.h"
#include "Mechanic.h"
#include "Manager.h"
#include "Mediator.h"
#include "ConcreteMediator.h"
#include "RaceManager.h"
#include "ConcreteRaceManager.h"
#include "State.h"
#include "Driver.h"
#include "AggressiveDriver.h"
#include "AverageDriver.h"
#include "PassiveDriver.h"
```

Include dependency graph for Facade.h:



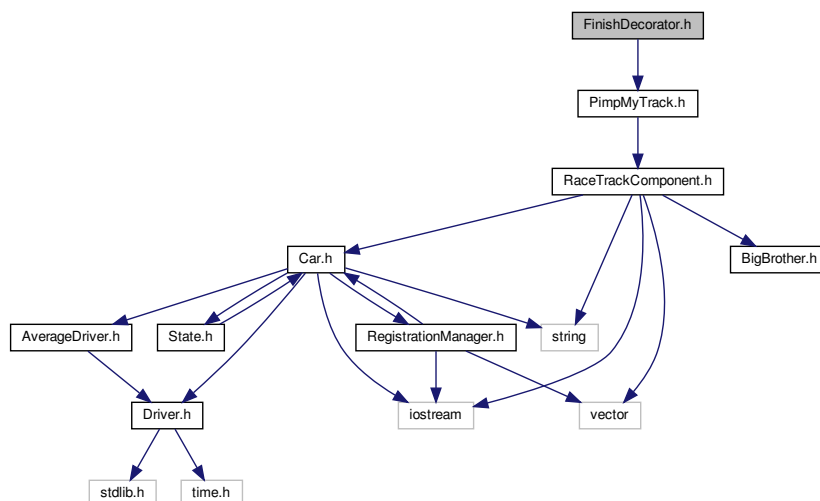
Classes

- class [Facade](#)
Facade pattern.

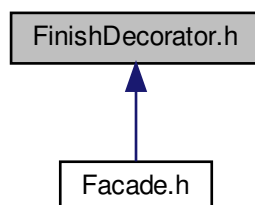
5.14 FinishDecorator.h File Reference

```
#include "PimpMyTrack.h"
```

Include dependency graph for FinishDecorator.h:



This graph shows which files directly or indirectly include this file:



Classes

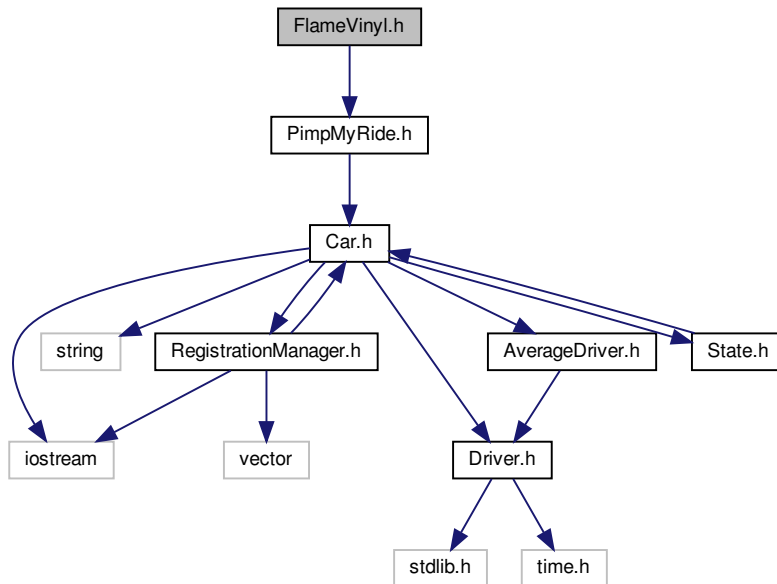
- class [FinishDecorator](#)

ConcreteDecorator for Decorator design pattern.

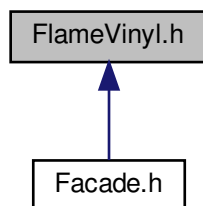
5.15 FlameVinyl.h File Reference

```
#include "PimpMyRide.h"
```

Include dependency graph for FlameVinyl.h:



This graph shows which files directly or indirectly include this file:



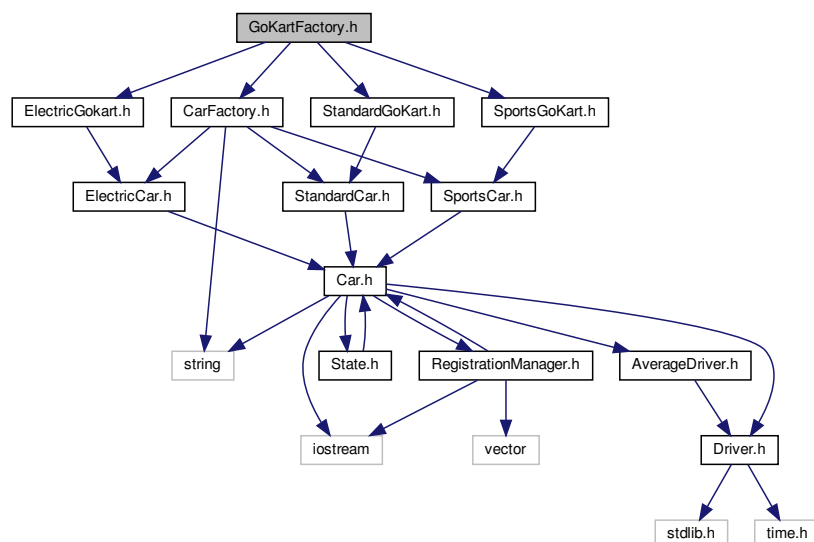
Classes

- class [FlameVinyl](#)

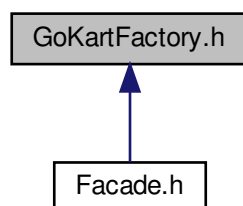
Concrete Decorator for Decorator Pattern.

5.16 GoKartFactory.h File Reference

```
#include "CarFactory.h"
#include "SportsGoKart.h"
#include "StandardGoKart.h"
#include "ElectricGokart.h"
Include dependency graph for GoKartFactory.h:
```



This graph shows which files directly or indirectly include this file:

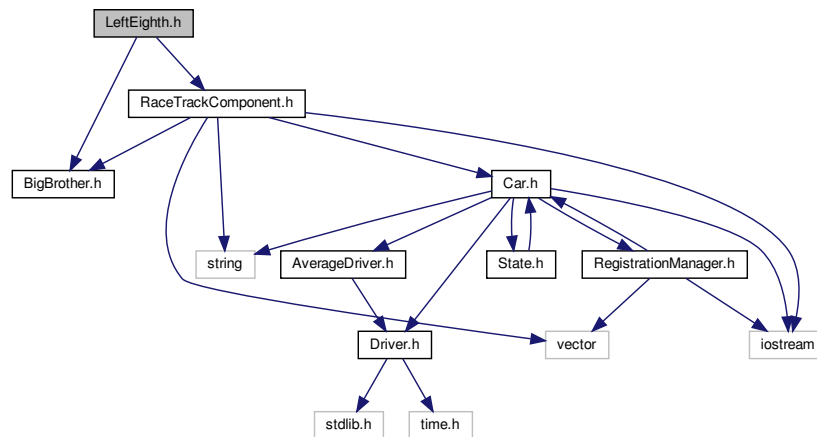


Classes

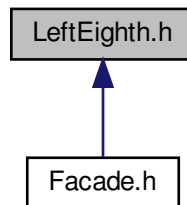
- class [GoKartFactory](#)
Concrete Factory for Abstract Factory Pattern.

5.17 LeftEighth.h File Reference

```
#include "BigBrother.h"
#include "RaceTrackComponent.h"
Include dependency graph for LeftEighth.h:
```



This graph shows which files directly or indirectly include this file:



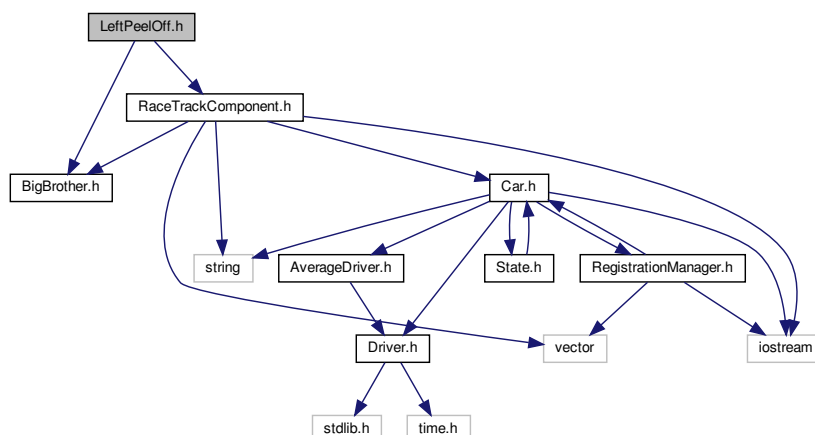
Classes

- class [LeftEighth](#)
Leaf for Composite design pattern.

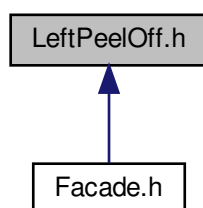
5.18 LeftPeelOff.h File Reference

```
#include "BigBrother.h"
#include "RaceTrackComponent.h"
```

Include dependency graph for LeftPeelOff.h:



This graph shows which files directly or indirectly include this file:



Classes

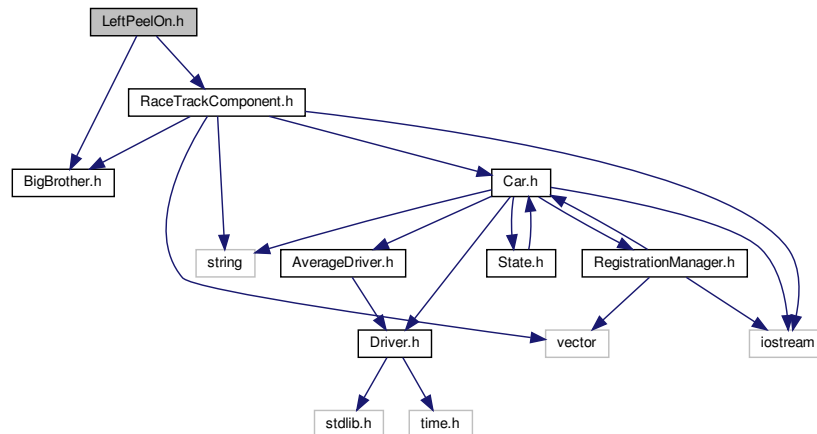
- class [LeftPeelOff](#)

Leaf for Composite design pattern.

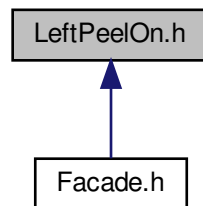
5.19 LeftPeelOn.h File Reference

```
#include "BigBrother.h"
#include "RaceTrackComponent.h"
```

Include dependency graph for LeftPeelOn.h:



This graph shows which files directly or indirectly include this file:



Classes

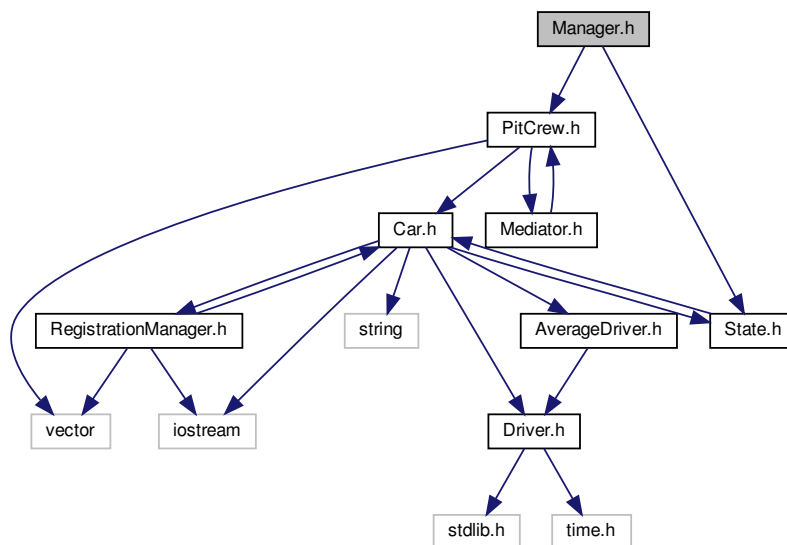
- class [LeftPeelOn](#)

Leaf for Composite design pattern.

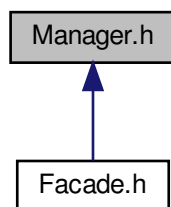
5.20 Manager.h File Reference

```
#include "PitCrew.h"
#include "State.h"
```

Include dependency graph for Manager.h:



This graph shows which files directly or indirectly include this file:



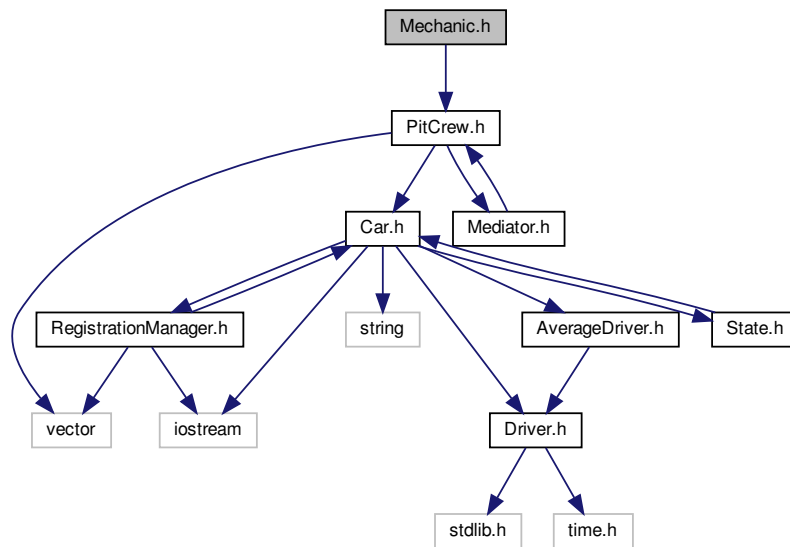
Classes

- class [Manager](#)

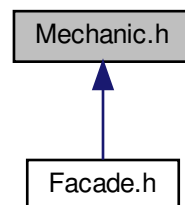
5.21 Mechanic.h File Reference

```
#include "PitCrew.h"
```


Include dependency graph for Mechanic.h:



This graph shows which files directly or indirectly include this file:



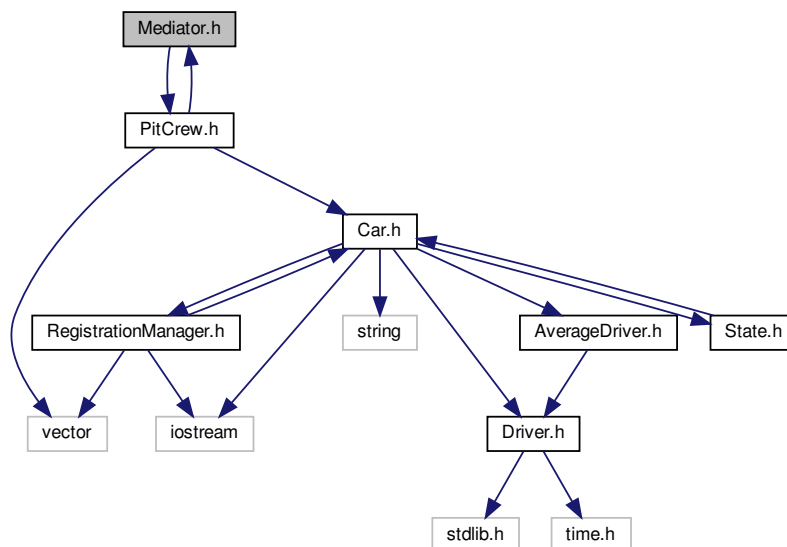
Classes

- class [Mechanic](#)

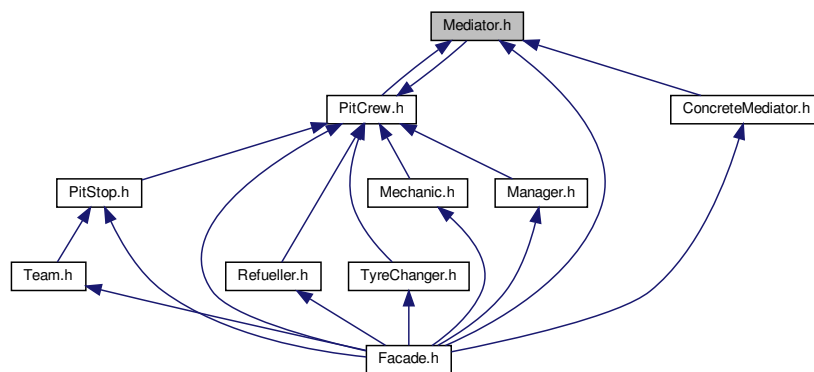
5.22 Mediator.h File Reference

```
#include "PitCrew.h"
```

Include dependency graph for Mediator.h:



This graph shows which files directly or indirectly include this file:



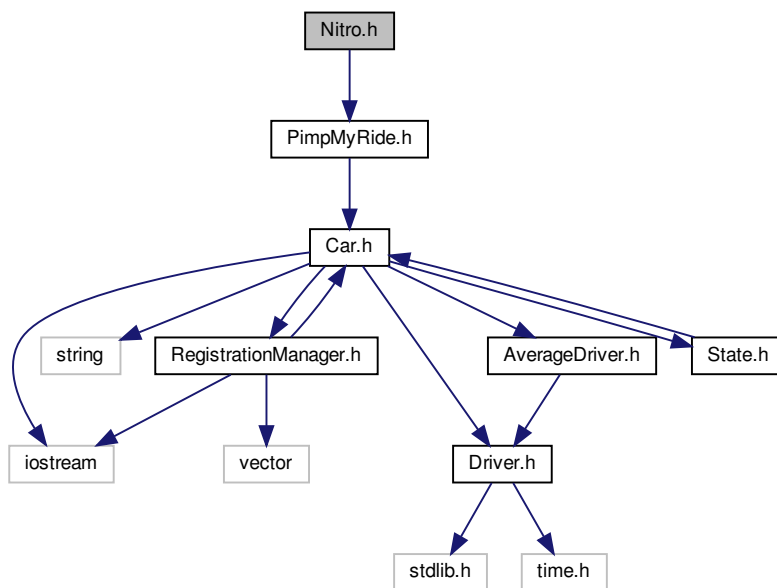
Classes

- class [Mediator](#)

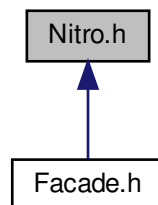
5.23 Nitro.h File Reference

```
#include "PimpMyRide.h"
```

Include dependency graph for Nitro.h:



This graph shows which files directly or indirectly include this file:



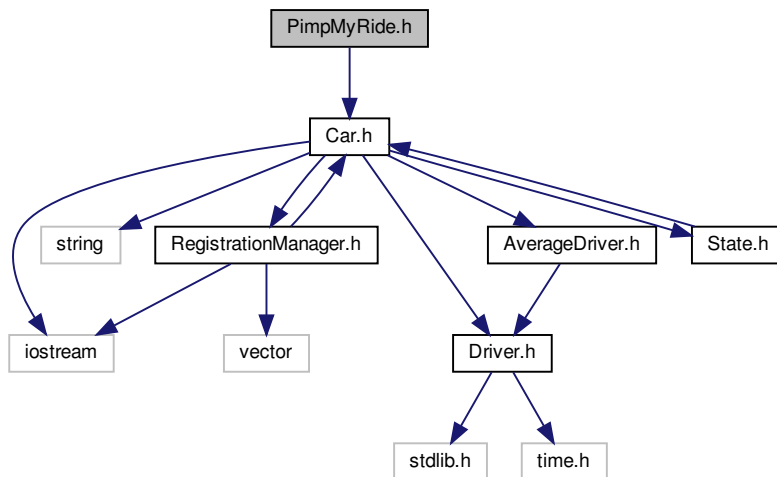
Classes

- class [Nitro](#)
Concrete Decorator for Decorator Pattern.

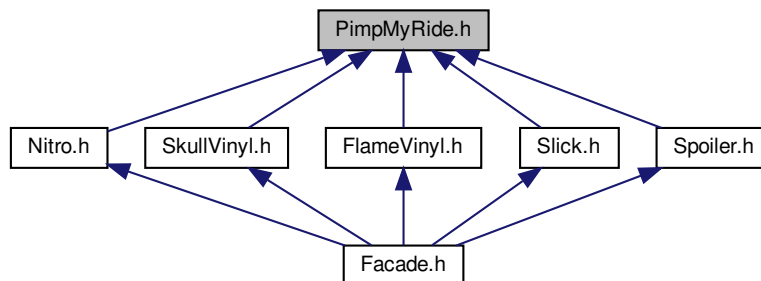
5.24 PimpMyRide.h File Reference

```
#include "Car.h"
```

Include dependency graph for PimpMyRide.h:



This graph shows which files directly or indirectly include this file:



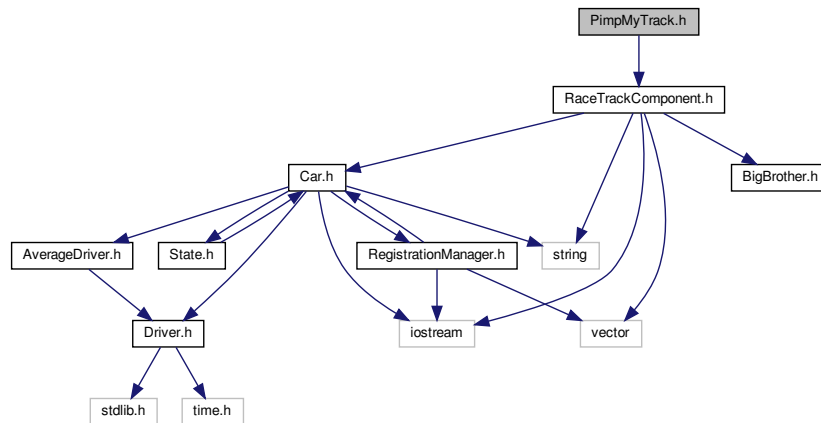
Classes

- class [PimpMyRide](#)
Decoractor for Decorator Pattern.

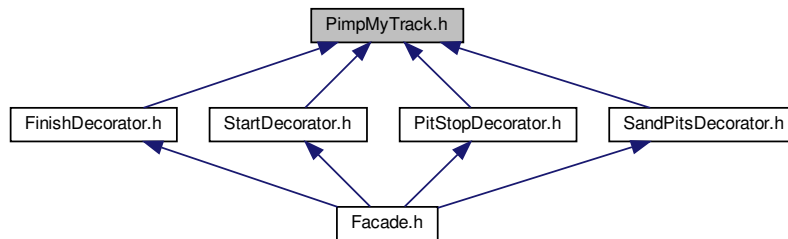
5.25 PimpMyTrack.h File Reference

```
#include "RaceTrackComponent.h"
```

Include dependency graph for PimpMyTrack.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [PimpMyTrack](#)

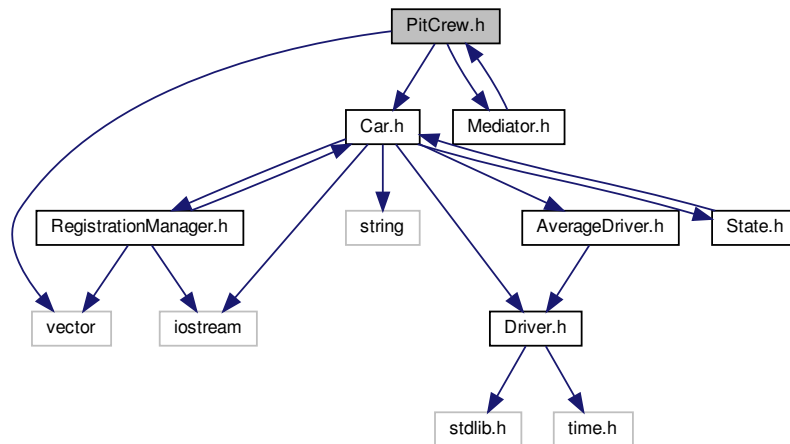
Abstract Decorator for Decorator design pattern.

5.26 PitCrew.h File Reference

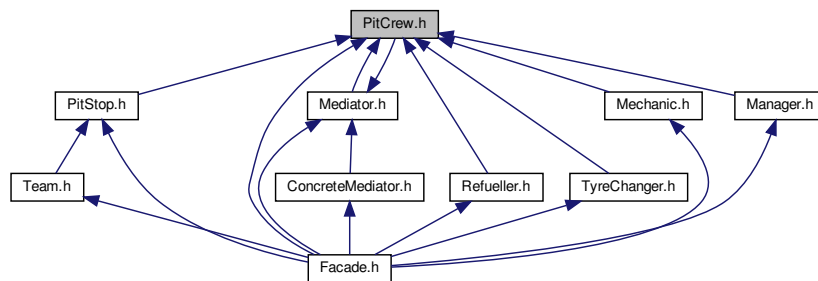
```
#include <vector>
#include "Car.h"
```

```
#include "Mediator.h"
```

Include dependency graph for PitCrew.h:



This graph shows which files directly or indirectly include this file:



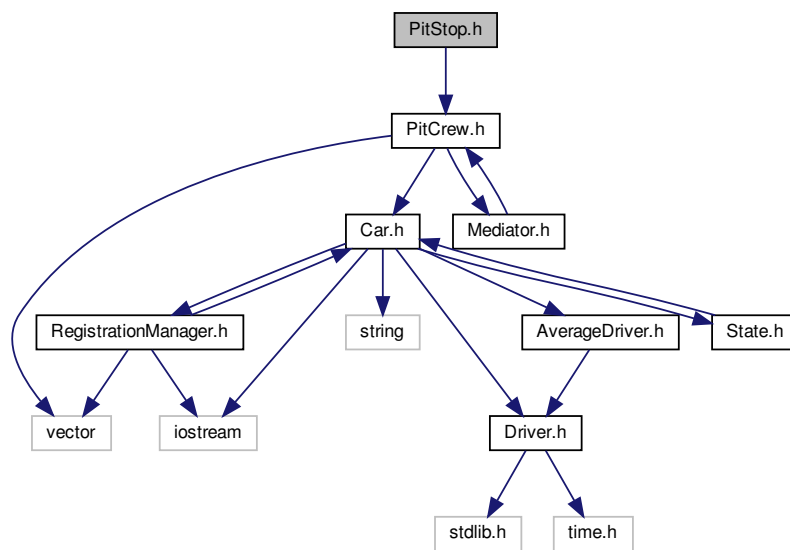
Classes

- class [PitCrew](#)

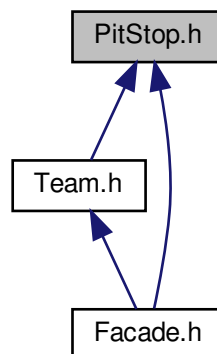
5.27 PitStop.h File Reference

```
#include "PitCrew.h"
```

Include dependency graph for PitStop.h:



This graph shows which files directly or indirectly include this file:



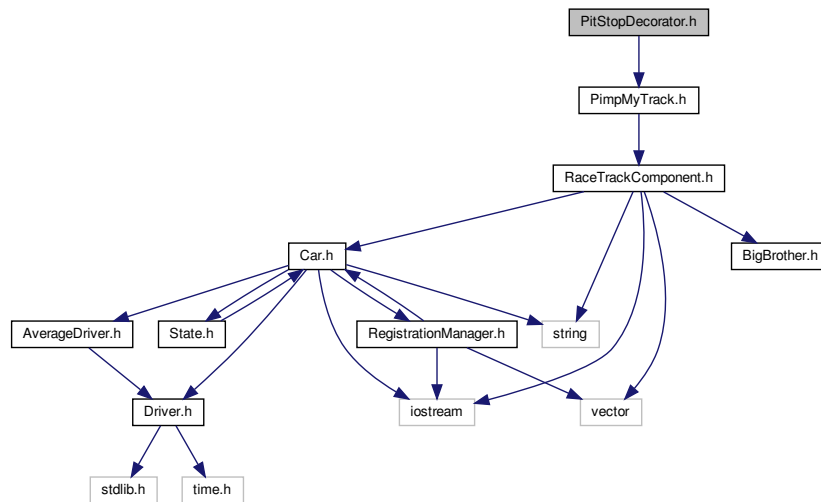
Classes

- class [PitStop](#)

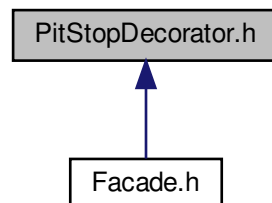
5.28 PitStopDecorator.h File Reference

```
#include "PimpMyTrack.h"
```

Include dependency graph for PitStopDecorator.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [PitStopDecorator](#)
ConcreteDecorator for Decorator design pattern.

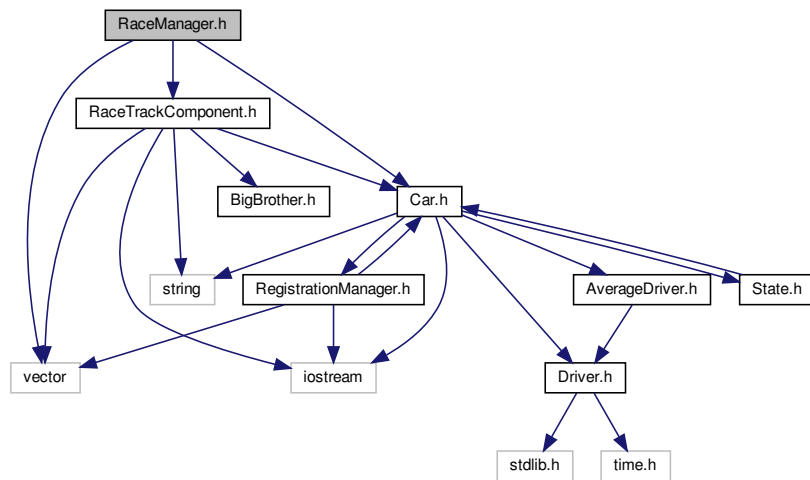
5.29 RaceManager.h File Reference

```
#include <vector>
#include "Car.h"
```

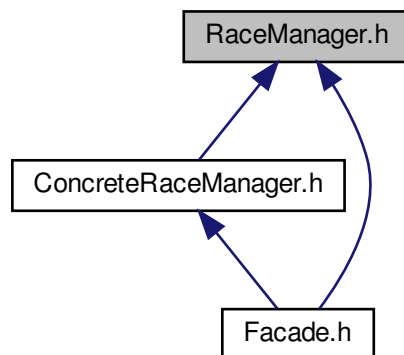


```
#include "RaceTrackComponent.h"
```

Include dependency graph for RaceManager.h:



This graph shows which files directly or indirectly include this file:



Classes

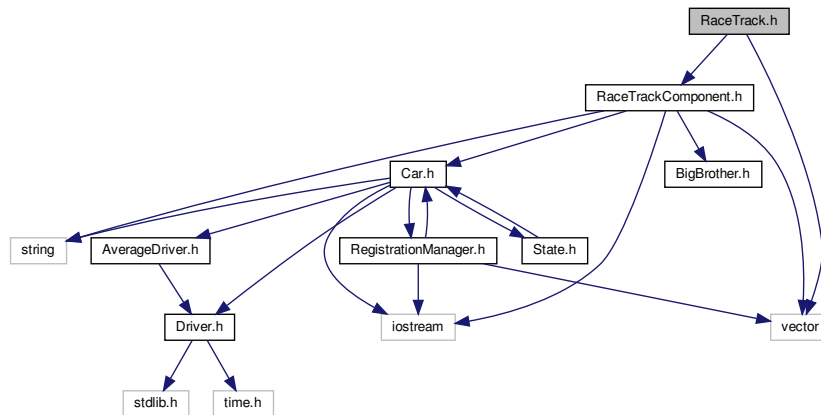
- class [RaceManager](#)
Observer class for Observer pattern.

5.30 RaceTrack.h File Reference

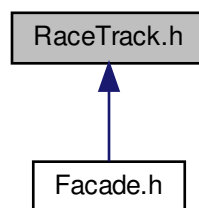
```
#include <vector>
```

```
#include "RaceTrackComponent.h"
```

Include dependency graph for RaceTrack.h:



This graph shows which files directly or indirectly include this file:



Classes

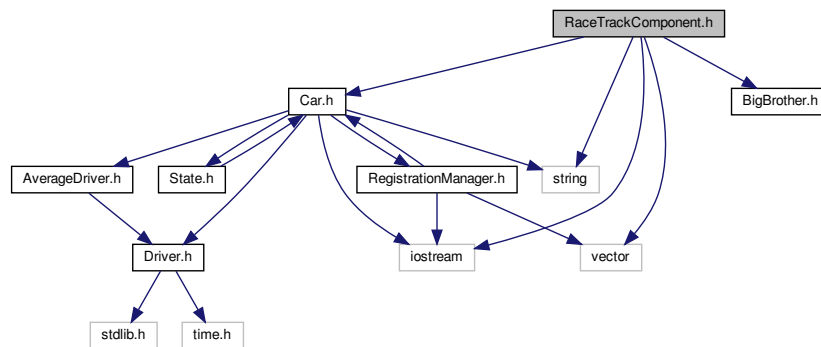
- class [RaceTrack](#)
composite class for composite pattern

5.31 RaceTrackComponent.h File Reference

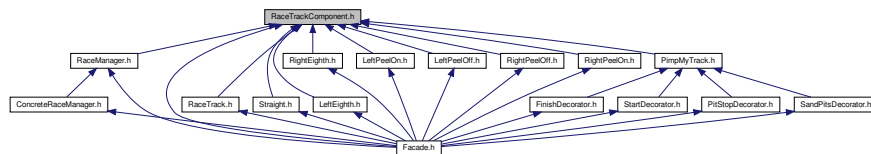
```
#include "Car.h"
#include "BigBrother.h"
#include <string>
#include <vector>
```

```
#include <iostream>
```

Include dependency graph for RaceTrackComponent.h:



This graph shows which files directly or indirectly include this file:



Classes

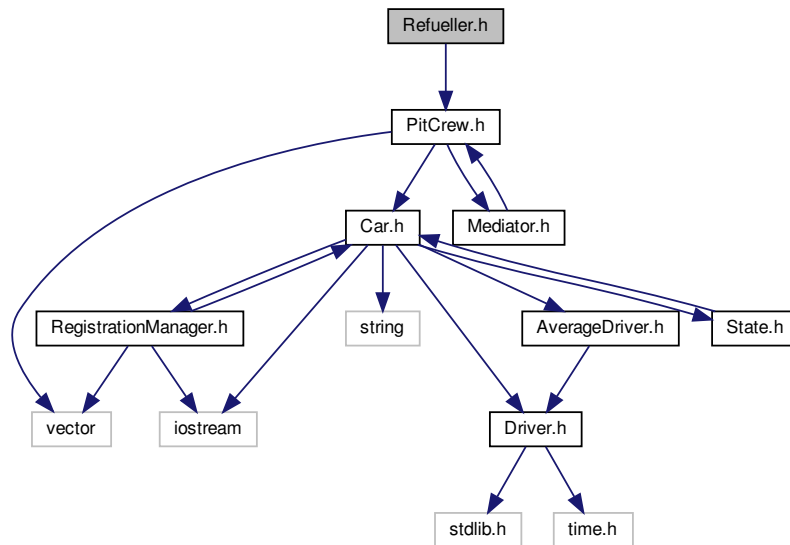
- class [RaceTrackComponent](#)

abstract leaf class for composite pattern

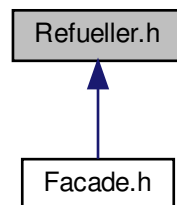
5.32 Refueller.h File Reference

```
#include "PitCrew.h"
```

Include dependency graph for Refueller.h:



This graph shows which files directly or indirectly include this file:



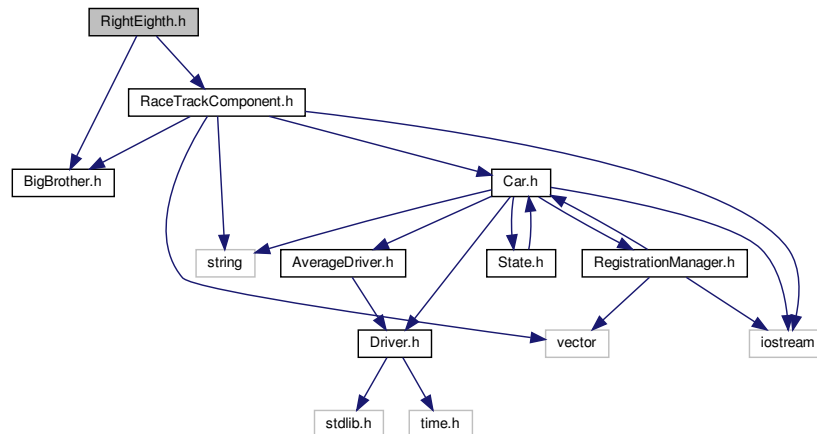
Classes

- class [Refueller](#)

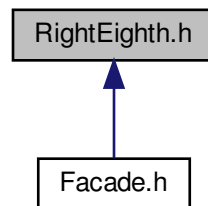
5.33 RightEighth.h File Reference

```
#include "BigBrother.h"
#include "RaceTrackComponent.h"
```

Include dependency graph for RightEighth.h:



This graph shows which files directly or indirectly include this file:



Classes

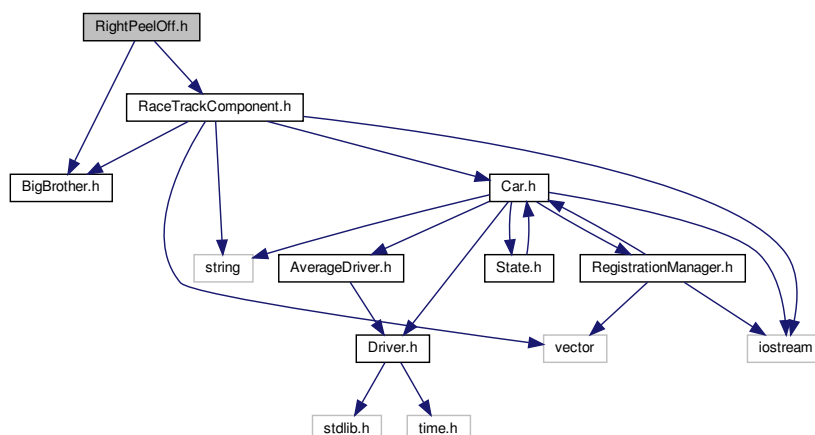
- class [RightEighth](#)

Leaf for Composite design pattern.

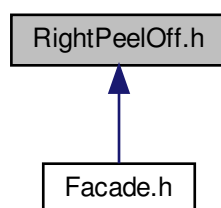
5.34 RightPeelOff.h File Reference

```
#include "BigBrother.h"
#include "RaceTrackComponent.h"
```

Include dependency graph for RightPeelOff.h:



This graph shows which files directly or indirectly include this file:



Classes

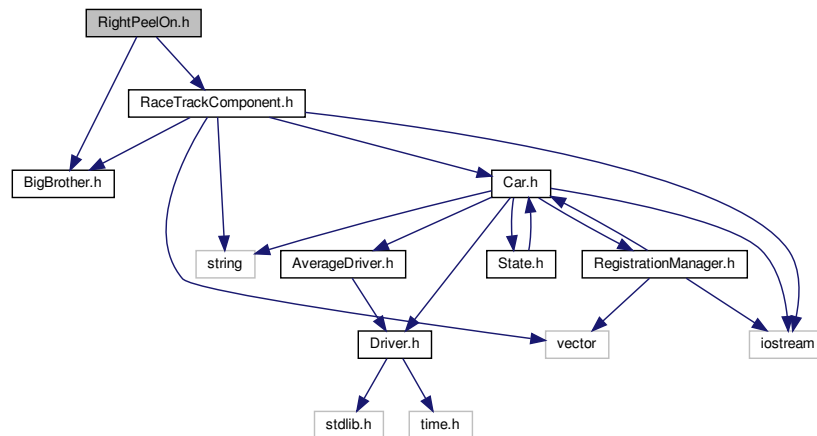
- class [RightPeelOff](#)

Leaf for Composite design pattern.

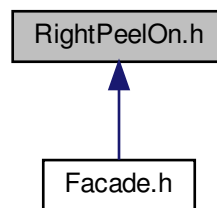
5.35 RightPeelOn.h File Reference

```
#include "BigBrother.h"
#include "RaceTrackComponent.h"
```

Include dependency graph for RightPeelOn.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [RightPeelOn](#)

Leaf for Composite design pattern.

5.36 RoadsterFactory.h File Reference

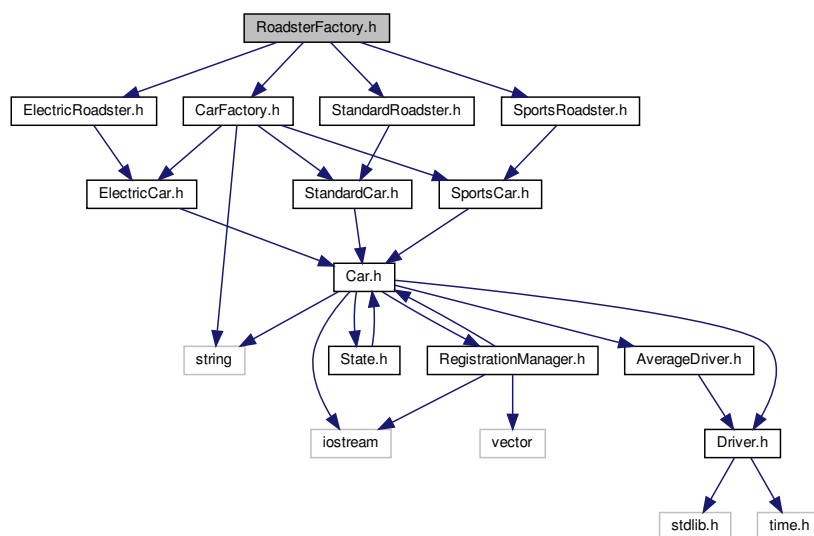
```

#include "CarFactory.h"
#include "ElectricRoadster.h"
#include "SportsRoadster.h"

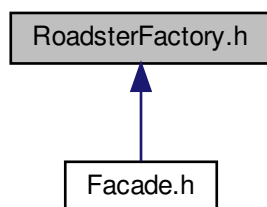
```

```
#include "StandardRoadster.h"
```

Include dependency graph for RoadsterFactory.h:



This graph shows which files directly or indirectly include this file:



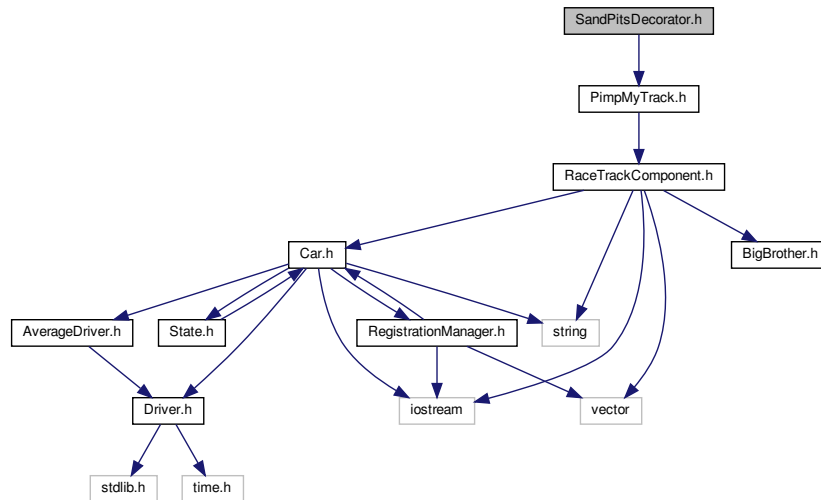
Classes

- class [RoadsterFactory](#)
Concrete Factory for Abstract Factory Pattern.

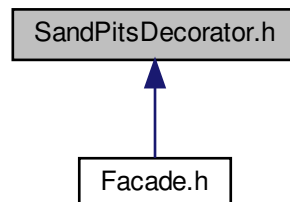
5.37 SandPitsDecorator.h File Reference

```
#include "PimpMyTrack.h"
```


Include dependency graph for SandPitsDecorator.h:



This graph shows which files directly or indirectly include this file:



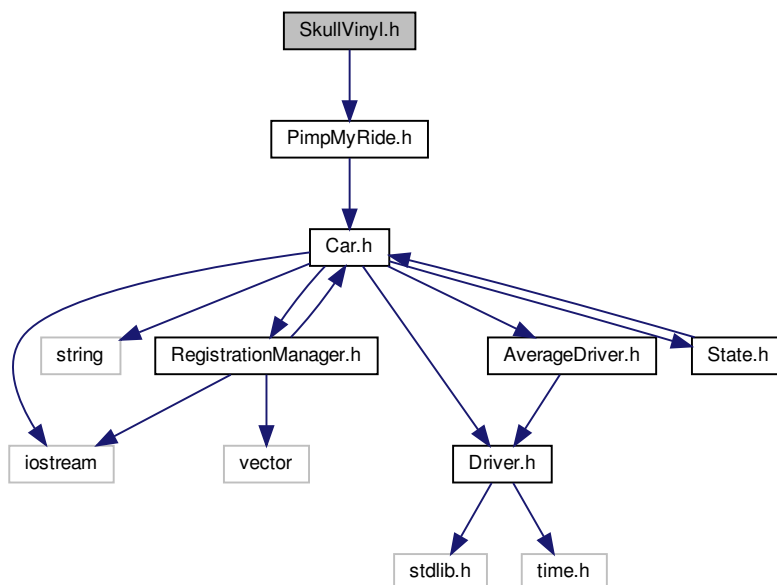
Classes

- class [SandPitsDecorator](#)
ConcreteDecorator for Decorator design pattern.

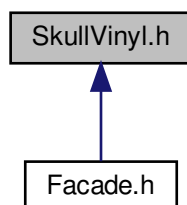
5.38 SkullVinyl.h File Reference

```
#include "PimpMyRide.h"
```

Include dependency graph for SkullVinyl.h:



This graph shows which files directly or indirectly include this file:



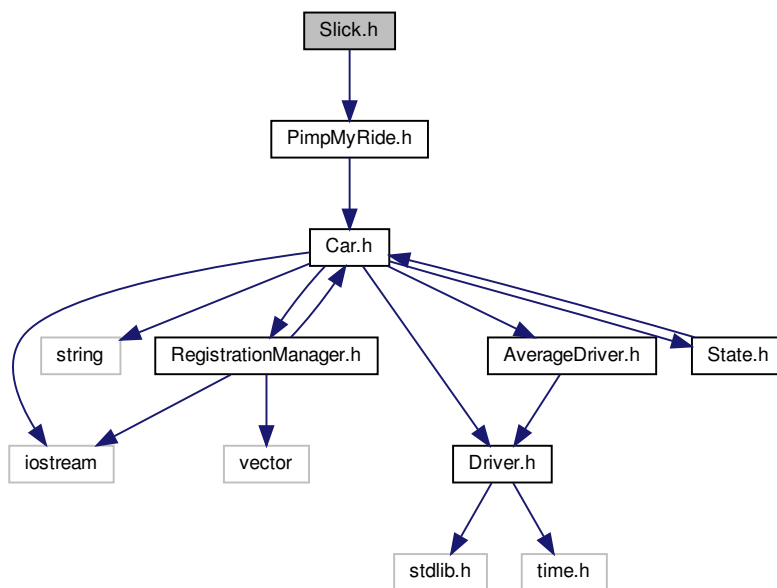
Classes

- class [SkullVinyl](#)
Concrete Decorator for Decorator Pattern.

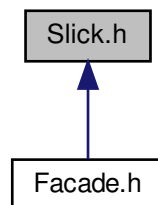
5.39 Slick.h File Reference

```
#include "PimpMyRide.h"
```

Include dependency graph for Slick.h:



This graph shows which files directly or indirectly include this file:



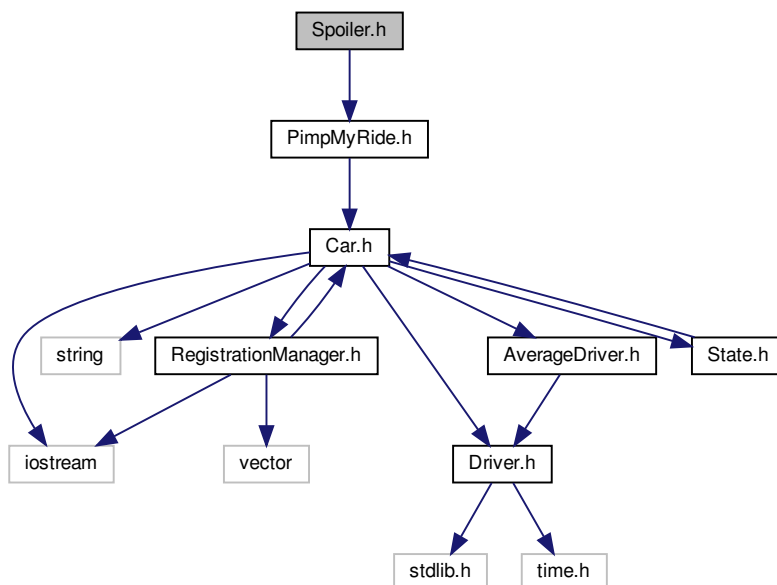
Classes

- class [Slick](#)
Concrete Decorator for Decorator Pattern.

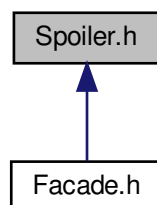
5.40 Spoiler.h File Reference

```
#include "PimpMyRide.h"
```

Include dependency graph for Spoiler.h:



This graph shows which files directly or indirectly include this file:



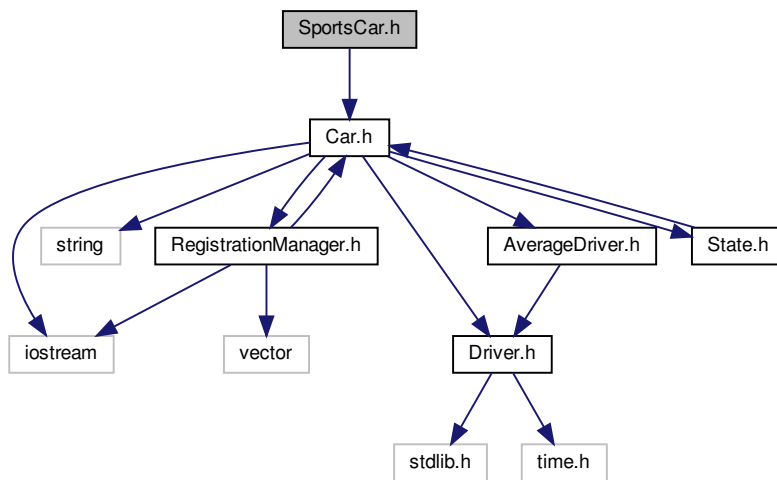
Classes

- class [Spoiler](#)
Concrete Decorator for Decorator Pattern.

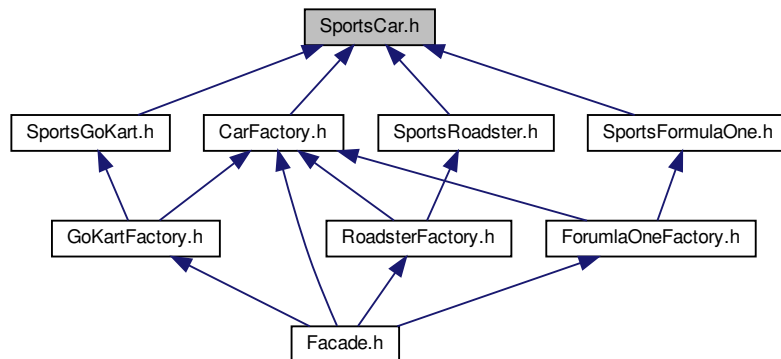
5.41 SportsCar.h File Reference

```
#include "Car.h"
```

Include dependency graph for SportsCar.h:



This graph shows which files directly or indirectly include this file:



Classes

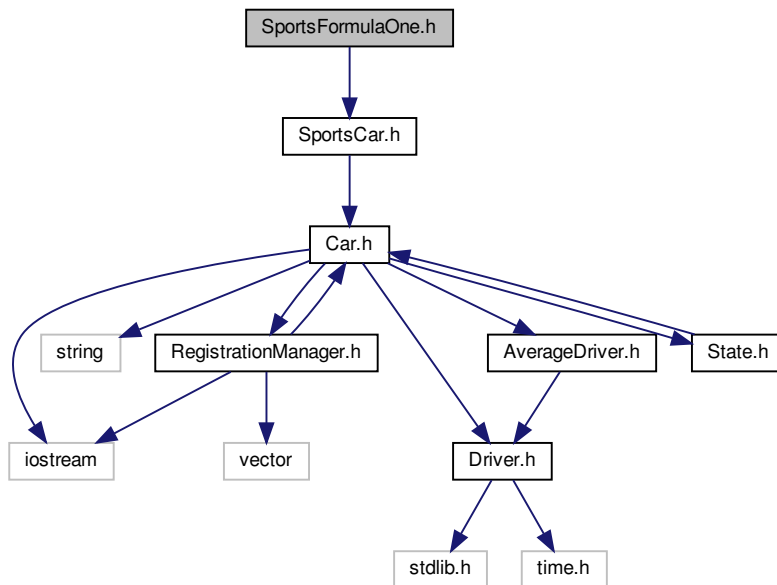
- class [SportsCar](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

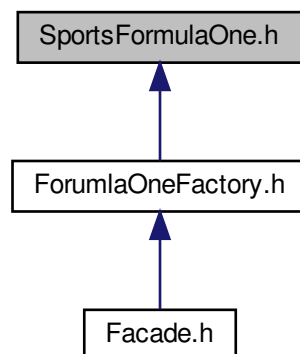
5.42 SportsFormulaOne.h File Reference

```
#include "SportsCar.h"
```

Include dependency graph for SportsFormulaOne.h:



This graph shows which files directly or indirectly include this file:



Classes

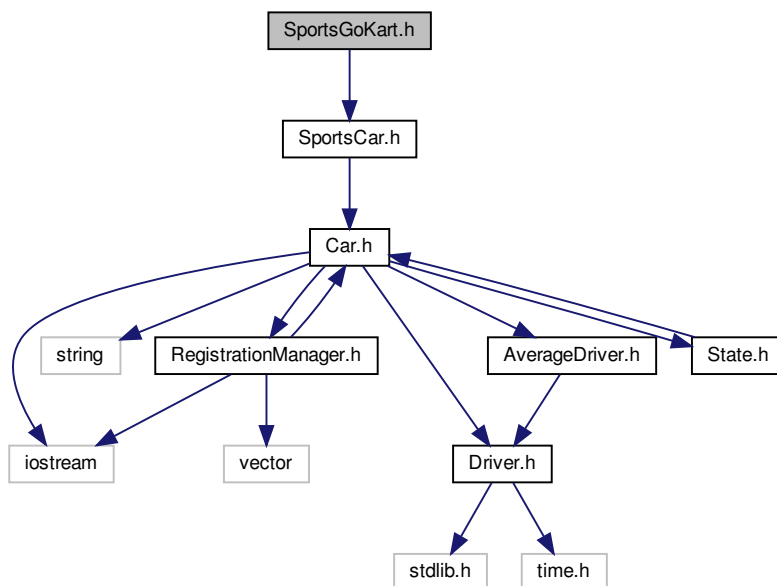
- class [SportsFormulaOne](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

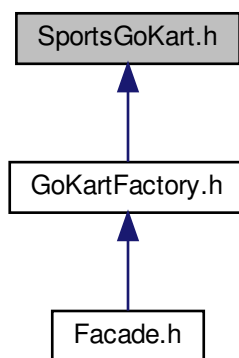
5.43 SportsGoKart.h File Reference

```
#include "SportsCar.h"
```

Include dependency graph for SportsGoKart.h:



This graph shows which files directly or indirectly include this file:



Classes

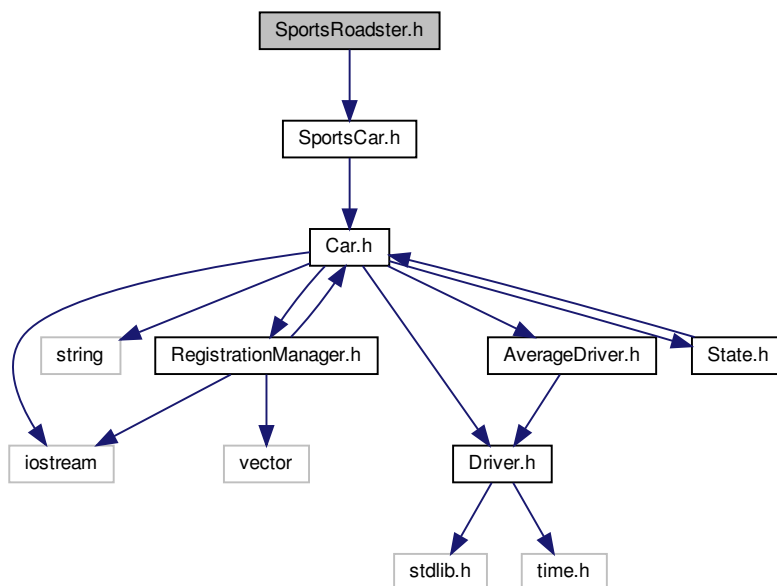
- class [SportsGoKart](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

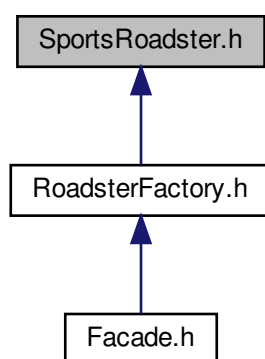
5.44 SportsRoadster.h File Reference

```
#include "SportsCar.h"
```

Include dependency graph for SportsRoadster.h:



This graph shows which files directly or indirectly include this file:



Classes

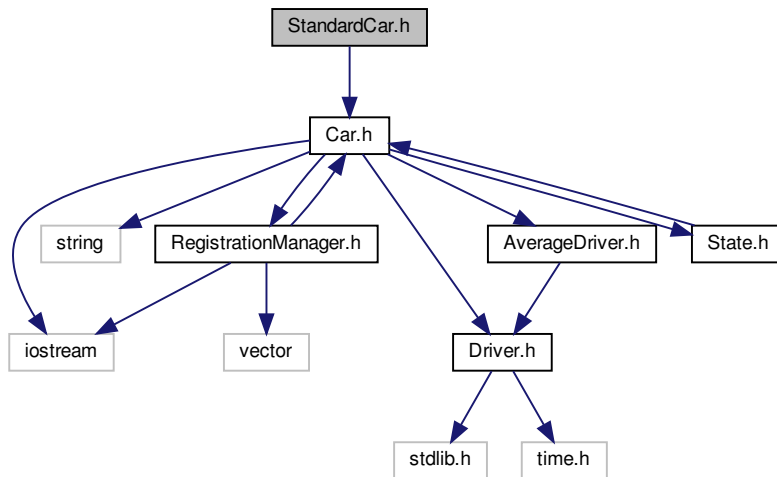
- class [SportsRoadster](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

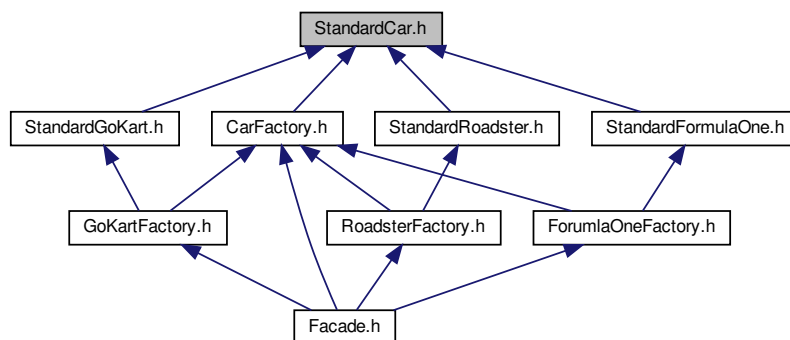
5.45 StandardCar.h File Reference

```
#include "Car.h"
```

Include dependency graph for StandardCar.h:



This graph shows which files directly or indirectly include this file:



Classes

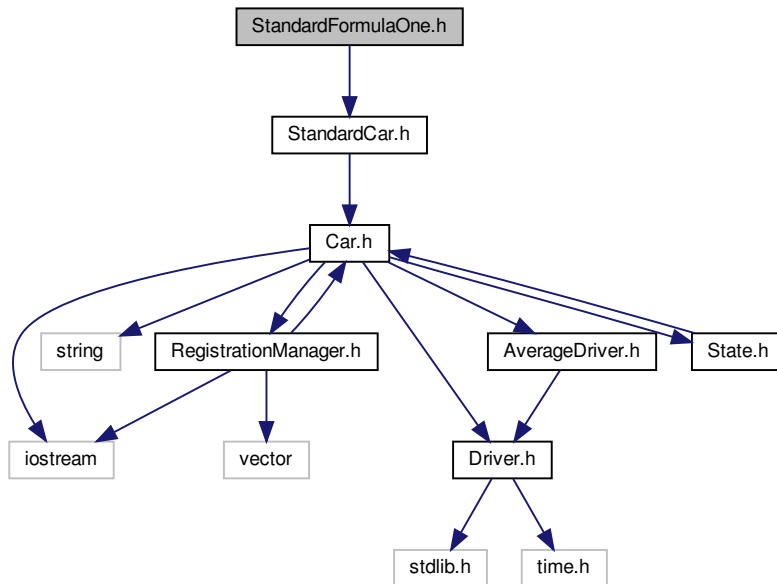
- class [StandardCar](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

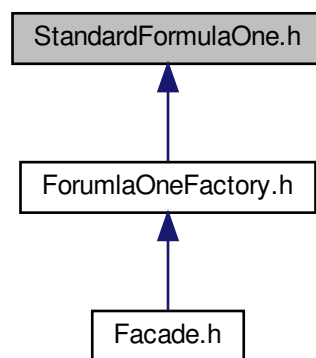
5.46 StandardFormulaOne.h File Reference

```
#include "StandardCar.h"
```

Include dependency graph for StandardFormulaOne.h:



This graph shows which files directly or indirectly include this file:



Classes

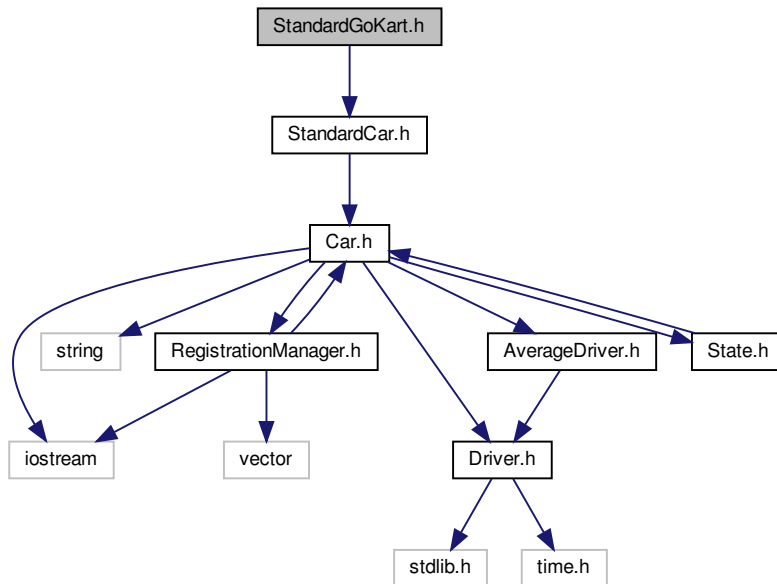
- class [StandardFormulaOne](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

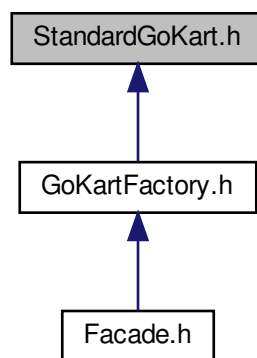
5.47 StandardGoKart.h File Reference

```
#include "StandardCar.h"
```

Include dependency graph for StandardGoKart.h:



This graph shows which files directly or indirectly include this file:



Classes

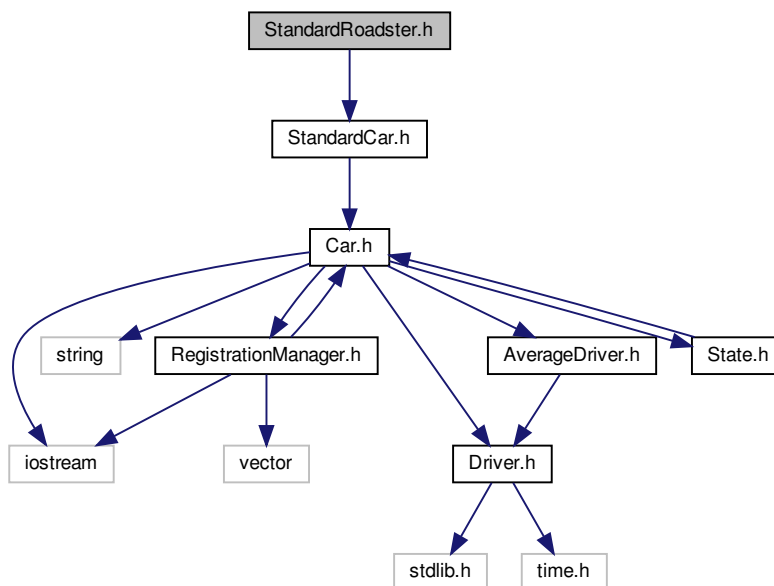
- class [StandardGoKart](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

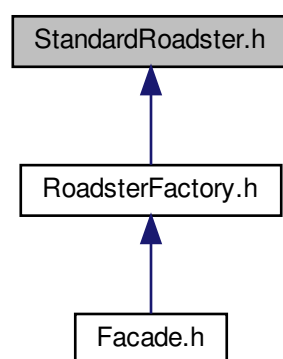
5.48 StandardRoadster.h File Reference

```
#include "StandardCar.h"
```

Include dependency graph for StandardRoadster.h:



This graph shows which files directly or indirectly include this file:



Classes

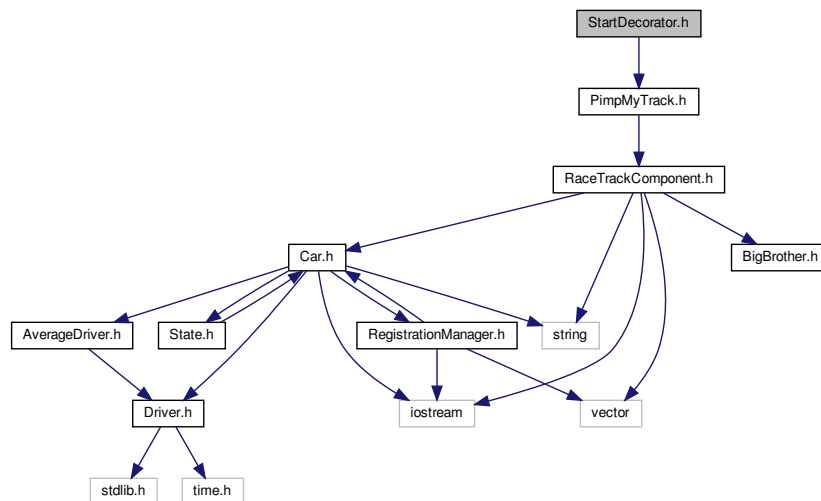
- class [StandardRoadster](#)

Concrete Product for Abstract Factory Pattern and Concrete Component for Decorator Pattern.

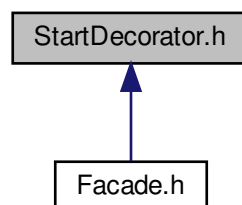
5.49 StartDecorator.h File Reference

```
#include "PimpMyTrack.h"
```

Include dependency graph for StartDecorator.h:



This graph shows which files directly or indirectly include this file:



Classes

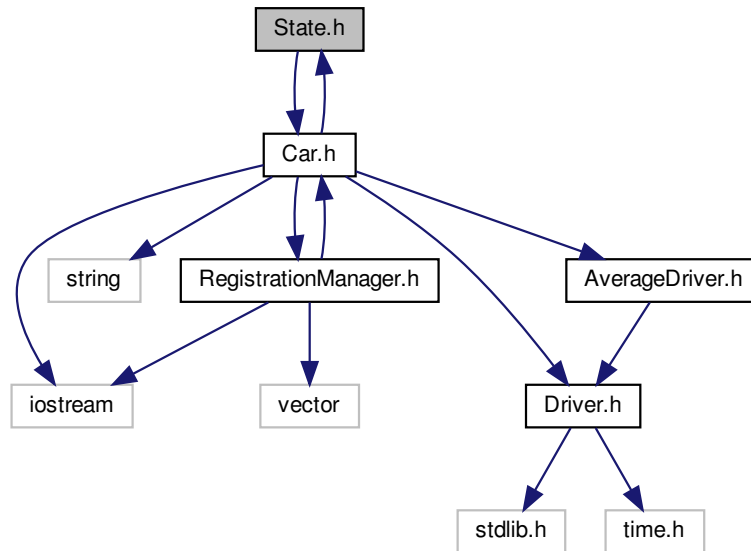
- class [StartDecorator](#)

ConcreteDecorator for Decorator design pattern.

5.50 State.h File Reference

```
#include "Car.h"
```

Include dependency graph for State.h:



This graph shows which files directly or indirectly include this file:



Classes

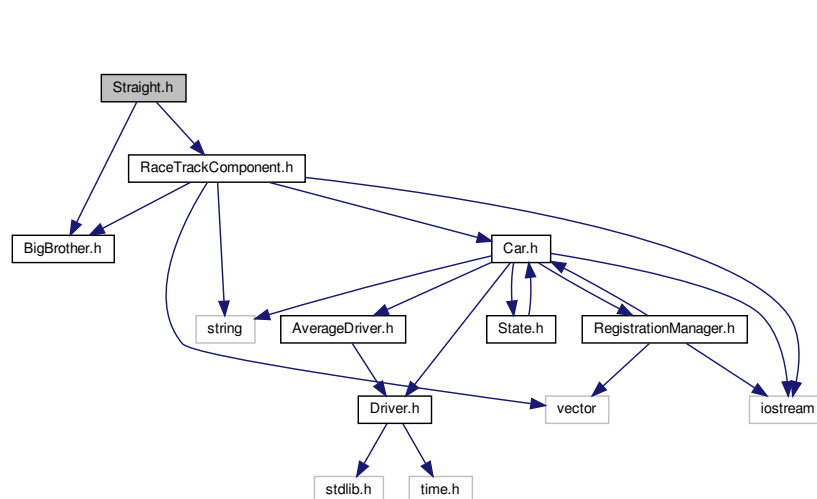
- class [State](#)
- class [Ready](#)
- class [Racing](#)
- class [Stopped](#)

5.51 Straight.h File Reference

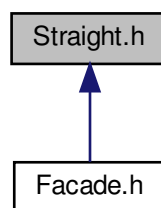
```
#include "BigBrother.h"
```

```
#include "RaceTrackComponent.h"
```

Include dependency graph for Straight.h:



This graph shows which files directly or indirectly include this file:



Classes

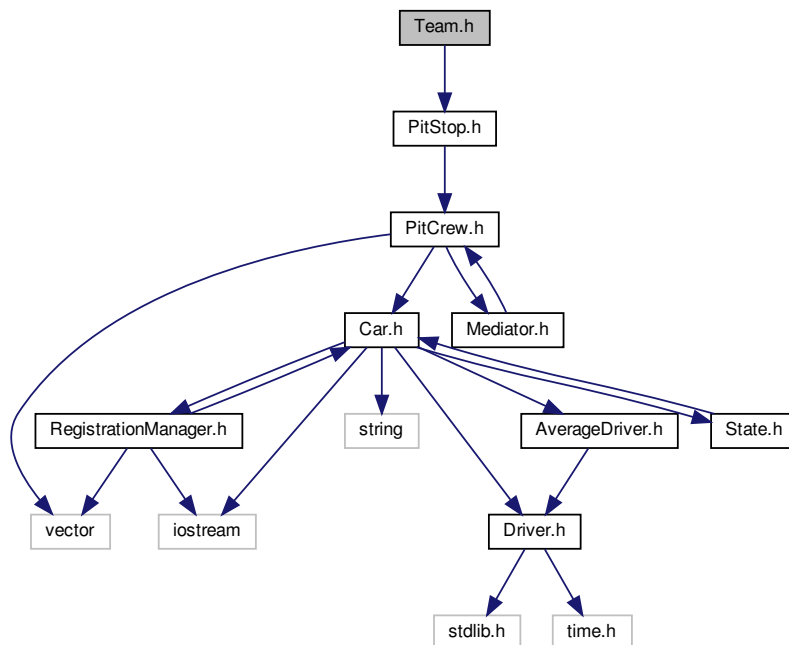
- class [Straight](#)

Leaf for Composite design pattern.

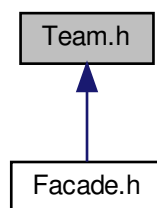
5.52 Team.h File Reference

```
#include "PitStop.h"
```

Include dependency graph for Team.h:



This graph shows which files directly or indirectly include this file:



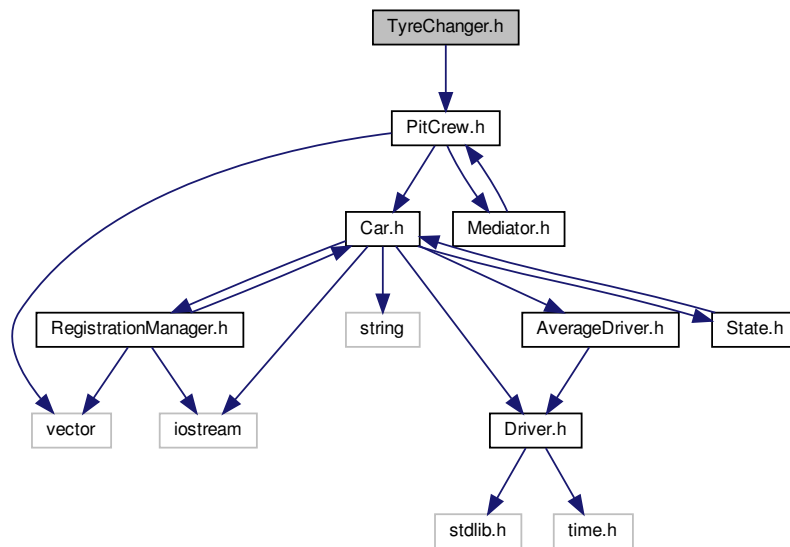
Classes

- class [Team](#)

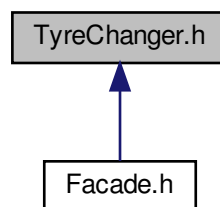
5.53 TyreChanger.h File Reference

```
#include "PitCrew.h"
```


Include dependency graph for TyreChanger.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [TyreChanger](#)

Index

- ~Car
 - Car, [18](#)
- ~ConcreteMediator
 - ConcreteMediator, [38](#)
- ~ConcreteRegistrationManager
 - ConcreteRegistrationManager, [46](#)
- ~ElectricCar
 - ElectricCar, [52](#)
- ~Facade
 - Facade, [63](#)
- ~FinishDecorator
 - FinishDecorator, [67](#)
- ~FlameVinyl
 - FlameVinyl, [69](#)
- ~LeftEighth
 - LeftEighth, [75](#)
- ~LeftPeelOff
 - LeftPeelOff, [78](#)
- ~LeftPeelOn
 - LeftPeelOn, [81](#)
- ~Mediator
 - Mediator, [92](#)
- ~Nitro
 - Nitro, [95](#)
- ~PimpMyRide
 - PimpMyRide, [98](#)
- ~PimpMyTrack
 - PimpMyTrack, [101](#)
- ~PitStop
 - PitStop, [109](#)
- ~PitStopDecorator
 - PitStopDecorator, [115](#)
- ~RaceTrack
 - RaceTrack, [120](#)
- ~RaceTrackComponent
 - RaceTrackComponent, [126](#)
- ~RightEighth
 - RightEighth, [140](#)
- ~RightPeelOff
 - RightPeelOff, [143](#)
- ~RightPeelOn
 - RightPeelOn, [146](#)
- ~SandPitsDecorator
 - SandPitsDecorator, [151](#)
- ~SkullVinyl
 - SkullVinyl, [153](#)
- ~Slick
 - Slick, [156](#)
- ~Spoiler
 - Spoiler, [158](#)
- ~SportsCar
 - SportsCar, [161](#)
- ~StandardCar
 - StandardCar, [173](#)
- ~StartDecorator
 - StartDecorator, [184](#)
- ~Straight
 - Straight, [189](#)
- accept
 - LeftEighth, [75](#)
 - LeftPeelOff, [78](#)
 - LeftPeelOn, [81](#)
 - PimpMyTrack, [101](#)
 - RaceTrack, [120](#)
 - RaceTrackComponent, [126](#)
 - RightEighth, [141](#)
 - RightPeelOff, [144](#)
 - RightPeelOn, [147](#)
 - Straight, [189](#)
- add
 - Car, [18](#)
 - LeftEighth, [76](#)
 - LeftPeelOff, [79](#)
 - LeftPeelOn, [82](#)
 - PimpMyRide, [98](#)
 - PimpMyTrack, [102](#)
 - RaceTrack, [121](#)
 - RaceTrackComponent, [126](#)
 - RightEighth, [141](#)
 - RightPeelOff, [144](#)
 - RightPeelOn, [147](#)
 - Straight, [189](#)
- addAllCars
 - RaceTrack, [121](#)
 - RaceTrackComponent, [126](#)
- addCar
 - ConcreteRegistrationManager, [46](#)
 - PitStop, [109](#)
 - RaceTrackComponent, [127](#)
 - RegistratcionManager, [138](#)
- addCars
 - ConcreteRaceManager, [41](#)
 - RaceManager, [117](#)
- addMember
 - ConcreteMediator, [38](#)
 - Mediator, [92](#)
- addRacetrack
 - ConcreteRaceManager, [41](#)

- RaceManager, 117
- addTime
 - LeftEighth, 76
 - LeftPeelOff, 79
 - LeftPeelOn, 82
 - PimpMyTrack, 102
 - RaceTrack, 121
 - RaceTrackComponent, 127
 - RightEighth, 141
 - RightPeelOff, 144
 - RightPeelOn, 147
 - Straight, 190
- addTrack
 - ConcreteRegistrationManager, 47
 - RegistratcionManager, 138
- AggressiveDriver, 9
 - AggressiveDriver, 10
- AggressiveDriver.h, 197
- attach
 - PitStop, 110
- attachManager
 - PitStop, 110
- AverageDriver, 10
 - AverageDriver, 11
- BigBrother, 11
 - visit, 12–14
- BigBrother.h, 198
- Car, 14
 - ~Car, 18
 - add, 18
 - Car, 17, 18
 - carDecorate, 30
 - clone, 19
 - FullClone, 19
 - getAcceleration, 19
 - getCarDamage, 19
 - getCarFuel, 20
 - getCarID, 20
 - getCarTyre, 20
 - getCarTyres, 21
 - getDescription, 21
 - getDriver, 21
 - getHandling, 21
 - getLap, 21
 - getManager, 22
 - getModelNumber, 22
 - getModelType, 22
 - getNumTyres, 22
 - getSpeed, 23
 - getState, 23
 - getTeam, 23
 - getTrackPart, 23
 - getTrackTime, 24
 - notifyTeam, 24
 - racing, 24
 - ready, 24
 - RegistrationNotify, 24
 - setAcceleration, 25
 - setCarDamage, 25
 - setCarFuel, 25
 - setCarTyre, 25
 - setChanged, 26
 - setDescription, 26
 - setDriver, 26
 - setHandling, 27
 - setLap, 27
 - setManager, 27
 - setRefuel, 27
 - setRepair, 28
 - setSpeed, 28
 - setState, 28
 - setTeam, 29
 - setTrackPart, 29
 - setTrackTime, 29
 - showCarCondition, 29
 - showCarStats, 30
 - stopped, 30
 - toString, 30
- Car.h, 198
- carDecorate
 - Car, 30
- CarFactory, 31
 - produceElectric, 32
 - produceSports, 32
 - produceStandard, 32
- CarFactory.h, 199
- cars
 - RaceTrackComponent, 131
- changeTyre
 - TyreChanger, 194
- changed
 - PitCrew, 104
- changedCar
 - PitCrew, 105
- clone
 - Car, 19
 - ElectricCar, 53
 - ElectricFormulaOne, 55
 - ElectricGoKart, 59
 - ElectricRoadster, 61
 - PimpMyRide, 99
 - SportsCar, 161
 - SportsFormulaOne, 165
 - SportsGoKart, 167
 - SportsRoadster, 171
 - StandardCar, 174
 - StandardFormulaOne, 176
 - StandardGoKart, 180
 - StandardRoadster, 182
- ConcreteBigBrother, 33
 - visit, 34, 35, 37
- ConcreteBigBrother.h, 200
- ConcreteMediator, 37
 - ~ConcreteMediator, 38
 - addMember, 38

- getManager, 39
 - notify, 39
 - notifyManager, 39
- ConcreteMediator.h, 201
- ConcreteRaceManager, 40
 - addCars, 41
 - addRacetrack, 41
 - getCarInfo, 42
 - getLap, 42
 - getLapMax, 42
 - pauseRace, 42
 - printLeaderBoard, 43
 - readyRace, 43
 - resumeRace, 43
 - setLapCount, 43
 - setLapMax, 44
 - startRace, 44
 - stopRace, 44
- ConcreteRaceManager.h, 202
- ConcreteRegistrationManager, 45
 - ~ConcreteRegistrationManager, 46
 - addCar, 46
 - addTrack, 47
 - ConcreteRegistrationManager, 46
 - getCars, 47
 - getTrack, 47
- ConcreteRegistrationManager.h, 203
- copyCar
 - Facade, 63
- createCustomCar
 - Facade, 63
- createCustomeRacetrack
 - Facade, 63
- createDriver
 - Facade, 64
- createTeam
 - Facade, 64
- decorate
 - RaceTrackComponent, 131
- decorateTrack
 - RaceTrackComponent, 127
- detach
 - PitStop, 110
- Driver, 48
 - getDriverAbilty, 49
 - getFuelAbilty, 49
 - getTyreAbilty, 49
 - setAbility, 49
 - setFuelAbility, 50
 - setTyreAbility, 50
- Driver.h, 205
- ElectricCar, 50
 - ~ElectricCar, 52
 - clone, 53
 - ElectricCar, 52
 - FullClone, 53
 - getDescription, 53
- ElectricCar.h, 205
- ElectricFormulaOne, 54
 - clone, 55
 - ElectricFormulaOne, 55
- ElectricFormulaOne.h, 206
- ElectricGoKart, 56
 - clone, 59
 - ElectricGoKart, 57
- ElectricRoadster, 59
 - clone, 61
 - ElectricRoadster, 61
- ElectricRoadster.h, 208
- Facade, 62
 - ~Facade, 63
 - copyCar, 63
 - createCustomCar, 63
 - createCustomeRacetrack, 63
 - createDriver, 64
 - createTeam, 64
 - Facade, 63
 - prepRace, 64
 - registerCar, 65
 - registerTrack, 65
 - StartRace, 65
- Facade.h, 209
- FinishDecorator, 66
 - ~FinishDecorator, 67
 - FinishDecorator, 67
- FinishDecorator.h, 210
- FlameVinyl, 68
 - ~FlameVinyl, 69
 - FlameVinyl, 69
 - FullClone, 70
- FlameVinyl.h, 211
- FormulaOneFactory, 70
 - produceElectric, 71
 - produceSports, 71
 - produceStandard, 71
- FullClone
 - Car, 19
 - ElectricCar, 53
 - FlameVinyl, 70
 - Nitro, 95
 - PimpMyRide, 99
 - SkullVinyl, 154
 - Slick, 156
 - Spoiler, 159
 - SportsCar, 161
 - StandardCar, 174
- getAcceleration
 - Car, 19
- getAllCars
 - RaceTrack, 121
 - RaceTrackComponent, 128
- getAverageTime
 - LeftEighth, 76
 - LeftPeelOff, 79

- LeftPeelOn, [82](#)
- RightEighth, [141](#)
- RightPeelOff, [144](#)
- RightPeelOn, [147](#)
- Straight, [190](#)
- getCar
 - PitStop, [110](#)
- getCarDamage
 - Car, [19](#)
- getCarFuel
 - Car, [20](#)
- getCarID
 - Car, [20](#)
- getCarInfo
 - ConcreteRaceManager, [42](#)
- getCarStats
 - PitStop, [111](#)
 - Team, [192](#)
- getCarTyre
 - Car, [20](#)
- getCarTyres
 - Car, [21](#)
- getCars
 - ConcreteRegistrationManager, [47](#)
 - RaceTrackComponent, [128](#)
 - RegistratcionManager, [138](#)
- getDamage
 - Manager, [84](#)
 - Mechanic, [88](#)
 - PitCrew, [105](#)
 - Refueller, [135](#)
 - TyreChanger, [194](#)
- getDecorator
 - RaceTrackComponent, [128](#)
- getDescription
 - Car, [21](#)
 - ElectricCar, [53](#)
 - PitCrew, [105](#)
 - RaceTrackComponent, [128](#)
 - SportsCar, [162](#)
 - StandardCar, [174](#)
- getDriver
 - Car, [21](#)
- getDriverAbility
 - Driver, [49](#)
- getFuelAbility
 - Driver, [49](#)
- getFuelLevel
 - Manager, [85](#)
 - Mechanic, [89](#)
 - PitCrew, [105](#)
 - Refueller, [135](#)
 - TyreChanger, [194](#)
- getHandling
 - Car, [21](#)
- getLap
 - Car, [21](#)
 - ConcreteRaceManager, [42](#)
- getLapMax
 - ConcreteRaceManager, [42](#)
- getManager
 - Car, [22](#)
 - ConcreteMediator, [39](#)
 - PitStop, [111](#)
- getMember
 - PitStop, [111](#)
- getModelNumber
 - Car, [22](#)
- getModelType
 - Car, [22](#)
- getName
 - PitStop, [111](#)
- getNumComponents
 - RaceTrack, [122](#)
 - RaceTrackComponent, [129](#)
- getNumMembers
 - PitStop, [111](#)
- getNumTyres
 - Car, [22](#)
- getRaceTrackID
 - RaceTrack, [122](#)
- getSpeed
 - Car, [23](#)
- getState
 - Car, [23](#)
- getTeam
 - Car, [23](#)
- getTrack
 - ConcreteRegistrationManager, [47](#)
 - RegistratcionManager, [139](#)
- getTrackPart
 - Car, [23](#)
- getTrackTime
 - Car, [24](#)
- getTyreAbility
 - Driver, [49](#)
- getTyreCondition
 - Manager, [85](#)
 - Mechanic, [89](#)
 - PitCrew, [105](#)
 - Refueller, [136](#)
 - TyreChanger, [194](#)
- GoKartFactory, [72](#)
 - produceElectric, [73](#)
 - produceSports, [73](#)
 - produceStandard, [73](#)
- GoKartFactory.h, [212](#)
- LeftEighth, [74](#)
 - ~LeftEighth, [75](#)
 - accept, [75](#)
 - add, [76](#)
 - addTime, [76](#)
 - getAverageTime, [76](#)
 - LeftEighth, [75](#)
 - print, [76](#)
- LeftEighth.h, [213](#)

- LeftPeelOff, 77
 - ~LeftPeelOff, 78
 - accept, 78
 - add, 79
 - addTime, 79
 - getAverageTime, 79
 - LeftPeelOff, 78
 - print, 79
- LeftPeelOff.h, 213
- LeftPeelOn, 80
 - ~LeftPeelOn, 81
 - accept, 81
 - add, 82
 - addTime, 82
 - getAverageTime, 82
 - LeftPeelOn, 81
 - print, 82
- LeftPeelOn.h, 214
- makeAccept
 - RaceTrack, 122
 - RaceTrackComponent, 129
- Manager, 83
 - getDamage, 84
 - getFuelLevel, 85
 - getTyreCondition, 85
 - Manager, 84
 - setDamage, 85
 - setFuelLevel, 86
 - setTyreCondition, 86
 - update, 86
- Manager.h, 215
- Mechanic, 87
 - getDamage, 88
 - getFuelLevel, 89
 - getTyreCondition, 89
 - Mechanic, 88
 - repair, 89
 - setDamage, 89
 - setFuelLevel, 90
 - setTyreCondition, 90
 - update, 90
- Mechanic.h, 216
- Mediator, 91
 - ~Mediator, 92
 - addMember, 92
 - notify, 92
 - notifyManager, 92
- Mediator.h, 217
- moveCar
 - RaceTrack, 123
 - RaceTrackComponent, 129
- Nitro, 93
 - ~Nitro, 95
 - FullClone, 95
 - Nitro, 94, 95
- Nitro.h, 218
- notify
 - ConcreteMediator, 39
 - Mediator, 92
 - PitStop, 112
- notifyManager
 - ConcreteMediator, 39
 - Mediator, 92
- notifyTeam
 - Car, 24
- PassiveDriver, 96
 - PassiveDriver, 96
- pauseRace
 - ConcreteRaceManager, 42
 - RaceManager, 117
- PimpMyRide, 97
 - ~PimpMyRide, 98
 - add, 98
 - clone, 99
 - FullClone, 99
 - PimpMyRide, 98
 - showCarStats, 99
- PimpMyRide.h, 219
- PimpMyTrack, 100
 - ~PimpMyTrack, 101
 - accept, 101
 - add, 102
 - addTime, 102
 - PimpMyTrack, 101
 - print, 102
- PimpMyTrack.h, 220
- PitCrew, 103
 - changed, 104
 - changedCar, 105
 - getDamage, 105
 - getDescription, 105
 - getFuelLevel, 105
 - getTyreCondition, 105
 - PitCrew, 104
 - registerWork, 106
 - setDamage, 106
 - setDescription, 106
 - setFuelLevel, 107
 - setTyreCondition, 107
 - update, 107
- PitCrew.h, 221
- PitStop, 108
 - ~PitStop, 109
 - addCar, 109
 - attach, 110
 - attachManager, 110
 - detach, 110
 - getCar, 110
 - getCarStats, 111
 - getManager, 111
 - getMember, 111
 - getName, 111
 - getNumMembers, 111
 - notify, 112
 - PitStop, 109

- setDamage, 112
 - setFuelLevel, 112
 - setTyreCondition, 113
 - showCar, 113
 - showCrew, 113
 - showManager, 113
 - toString, 113
- PitStop.h, 222
- PitStopDecorator, 114
 - ~PitStopDecorator, 115
 - PitStopDecorator, 115
- PitStopDecorator.h, 224
- prepRace
 - Facade, 64
- print
 - LeftEighth, 76
 - LeftPeelOff, 79
 - LeftPeelOn, 82
 - PimpMyTrack, 102
 - RaceTrack, 123
 - RaceTrackComponent, 130
 - RightEighth, 142
 - RightPeelOff, 145
 - RightPeelOn, 148
 - Straight, 190
- printLeaderBoard
 - ConcreteRaceManager, 43
 - RaceManager, 117
- produceElectric
 - CarFactory, 32
 - FormulaOneFactory, 71
 - GoKartFactory, 73
 - RoadsterFactory, 149
- produceSports
 - CarFactory, 32
 - FormulaOneFactory, 71
 - GoKartFactory, 73
 - RoadsterFactory, 149
- produceStandard
 - CarFactory, 32
 - FormulaOneFactory, 71
 - GoKartFactory, 73
 - RoadsterFactory, 150
- RaceManager, 116
 - addCars, 117
 - addRacetrack, 117
 - pauseRace, 117
 - printLeaderBoard, 117
 - readyRace, 118
 - resumeRace, 118
 - startRace, 118
 - stopRace, 118
- RaceManager.h, 224
- RaceTrack, 119
 - ~RaceTrack, 120
 - accept, 120
 - add, 121
 - addAllCars, 121
 - addTime, 121
 - getAllCars, 121
 - getNumComponents, 122
 - getRaceTrackID, 122
 - makeAccept, 122
 - moveCar, 123
 - print, 123
 - RaceTrack, 120
 - removeAllCars, 123
 - show, 123
- RaceTrack.h, 225
- RaceTrackComponent, 124
 - ~RaceTrackComponent, 126
 - accept, 126
 - add, 126
 - addAllCars, 126
 - addCar, 127
 - addTime, 127
 - cars, 131
 - decorate, 131
 - decorateTrack, 127
 - getAllCars, 128
 - getCars, 128
 - getDecorator, 128
 - getDescription, 128
 - getNumComponents, 129
 - makeAccept, 129
 - moveCar, 129
 - print, 130
 - RaceTrackComponent, 126
 - removeAllCars, 130
 - removeCar, 130
 - setDescription, 130
 - show, 131
- RaceTrackComponent.h, 226
- Racing, 132
- racing
 - Car, 24
 - State, 185
- Ready, 133
- ready
 - Car, 24
 - State, 186
- readyRace
 - ConcreteRaceManager, 43
 - RaceManager, 118
- refuel
 - Refueller, 136
- Refueller, 133
 - getDamage, 135
 - getFuelLevel, 135
 - getTyreCondition, 136
 - refuel, 136
 - Refueller, 135
 - setDamage, 136
 - setFuelLevel, 136
 - setTyreCondition, 137
 - update, 137

Refueller.h, 227
registerCar
 Facade, 65
registerTrack
 Facade, 65
registerWork
 PitCrew, 106
RegistratcionManager, 138
 addCar, 138
 addTrack, 138
 getCars, 138
 getTrack, 139
RegistrationNotify
 Car, 24
removeAllCars
 RaceTrack, 123
 RaceTrackComponent, 130
removeCar
 RaceTrackComponent, 130
repair
 Mechanic, 89
resumeRace
 ConcreteRaceManager, 43
 RaceManager, 118
RightEighth, 139
 ~RightEighth, 140
 accept, 141
 add, 141
 addTime, 141
 getAverageTime, 141
 print, 142
 RightEighth, 140
RightEighth.h, 228
RightPeelOff, 142
 ~RightPeelOff, 143
 accept, 144
 add, 144
 addTime, 144
 getAverageTime, 144
 print, 145
 RightPeelOff, 143
RightPeelOff.h, 229
RightPeelOn, 145
 ~RightPeelOn, 146
 accept, 147
 add, 147
 addTime, 147
 getAverageTime, 147
 print, 148
 RightPeelOn, 146
RightPeelOn.h, 230
RoadsterFactory, 148
 produceElectric, 149
 produceSports, 149
 produceStandard, 150
RoadsterFactory.h, 231
SandPitsDecorator, 150
 ~SandPitsDecorator, 151
 SandPitsDecorator, 151
SandPitsDecorator.h, 232
setAbility
 Driver, 49
setAcceleration
 Car, 25
setCarDamage
 Car, 25
setCarFuel
 Car, 25
setCarTyre
 Car, 25
setChanged
 Car, 26
setDamage
 Manager, 85
 Mechanic, 89
 PitCrew, 106
 PitStop, 112
 Refueller, 136
 TyreChanger, 195
setDescription
 Car, 26
 PitCrew, 106
 RaceTrackComponent, 130
setDriver
 Car, 26
setFuelAbility
 Driver, 50
setFuelLevel
 Manager, 86
 Mechanic, 90
 PitCrew, 107
 PitStop, 112
 Refueller, 136
 TyreChanger, 195
setHandling
 Car, 27
setLap
 Car, 27
setLapCount
 ConcreteRaceManager, 43
setLapMax
 ConcreteRaceManager, 44
setManager
 Car, 27
setRefuel
 Car, 27
setRepair
 Car, 28
setSpeed
 Car, 28
setState
 Car, 28
setTeam
 Car, 29
setTrackPart
 Car, 29

- setTrackTime
 - Car, [29](#)
- setTyreAbility
 - Driver, [50](#)
- setTyreCondition
 - Manager, [86](#)
 - Mechanic, [90](#)
 - PitCrew, [107](#)
 - PitStop, [113](#)
 - Refueller, [137](#)
 - TyreChanger, [195](#)
- show
 - RaceTrack, [123](#)
 - RaceTrackComponent, [131](#)
- showCar
 - PitStop, [113](#)
- showCarCondition
 - Car, [29](#)
- showCarStats
 - Car, [30](#)
 - PimpMyRide, [99](#)
- showCrew
 - PitStop, [113](#)
- showManager
 - PitStop, [113](#)
- SkullVinyl, [152](#)
 - ~SkullVinyl, [153](#)
 - FullClone, [154](#)
 - SkullVinyl, [153](#)
- SkullVinyl.h, [233](#)
- Slick, [154](#)
 - ~Slick, [156](#)
 - FullClone, [156](#)
 - Slick, [155](#), [156](#)
- Slick.h, [234](#)
- Spoiler, [157](#)
 - ~Spoiler, [158](#)
 - FullClone, [159](#)
 - Spoiler, [158](#)
- Spoiler.h, [235](#)
- SportsCar, [159](#)
 - ~SportsCar, [161](#)
 - clone, [161](#)
 - FullClone, [161](#)
 - getDescription, [162](#)
 - SportsCar, [160](#), [161](#)
- SportsCar.h, [236](#)
- SportsFormulaOne, [162](#)
 - clone, [165](#)
 - SportsFormulaOne, [163](#)
- SportsFormulaOne.h, [237](#)
- SportsGoKart, [165](#)
 - clone, [167](#)
 - SportsGoKart, [167](#)
- SportsGoKart.h, [239](#)
- SportsRoadster, [168](#)
 - clone, [171](#)
 - SportsRoadster, [169](#)
- SportsRoadster.h, [240](#)
- StandardCar, [171](#)
 - ~StandardCar, [173](#)
 - clone, [174](#)
 - FullClone, [174](#)
 - getDescription, [174](#)
 - StandardCar, [173](#)
- StandardCar.h, [241](#)
- StandardFormulaOne, [175](#)
 - clone, [176](#)
 - StandardFormulaOne, [176](#)
- StandardFormulaOne.h, [242](#)
- StandardGoKart, [177](#)
 - clone, [180](#)
 - StandardGoKart, [178](#)
- StandardGoKart.h, [243](#)
- StandardRoadster, [180](#)
 - clone, [182](#)
 - StandardRoadster, [182](#)
- StandardRoadster.h, [244](#)
- StartDecorator, [183](#)
 - ~StartDecorator, [184](#)
 - StartDecorator, [184](#)
- StartDecorator.h, [245](#)
- StartRace
 - Facade, [65](#)
- startRace
 - ConcreteRaceManager, [44](#)
 - RaceManager, [118](#)
- State, [185](#)
 - racing, [185](#)
 - ready, [186](#)
 - stopped, [186](#)
 - toString, [186](#)
- State.h, [246](#)
- stopRace
 - ConcreteRaceManager, [44](#)
 - RaceManager, [118](#)
- Stopped, [187](#)
- stopped
 - Car, [30](#)
 - State, [186](#)
- Straight, [187](#)
 - ~Straight, [189](#)
 - accept, [189](#)
 - add, [189](#)
 - addTime, [190](#)
 - getAverageTime, [190](#)
 - print, [190](#)
 - Straight, [189](#)
- Straight.h, [246](#)
- Team, [191](#)
 - getCarStats, [192](#)
 - Team, [192](#)
- Team.h, [247](#)
- toString
 - Car, [30](#)
 - PitStop, [113](#)

- State, [186](#)
- TyreChanger, [192](#)
 - changeTyre, [194](#)
 - getDamage, [194](#)
 - getFuelLevel, [194](#)
 - getTyreCondition, [194](#)
 - setDamage, [195](#)
 - setFuelLevel, [195](#)
 - setTyreCondition, [195](#)
 - TyreChanger, [193](#)
 - update, [196](#)
- TyreChanger.h, [248](#)
- update
 - Manager, [86](#)
 - Mechanic, [90](#)
 - PitCrew, [107](#)
 - Refueller, [137](#)
 - TyreChanger, [196](#)
- visit
 - BigBrother, [12–14](#)
 - ConcreteBigBrother, [34](#), [35](#), [37](#)