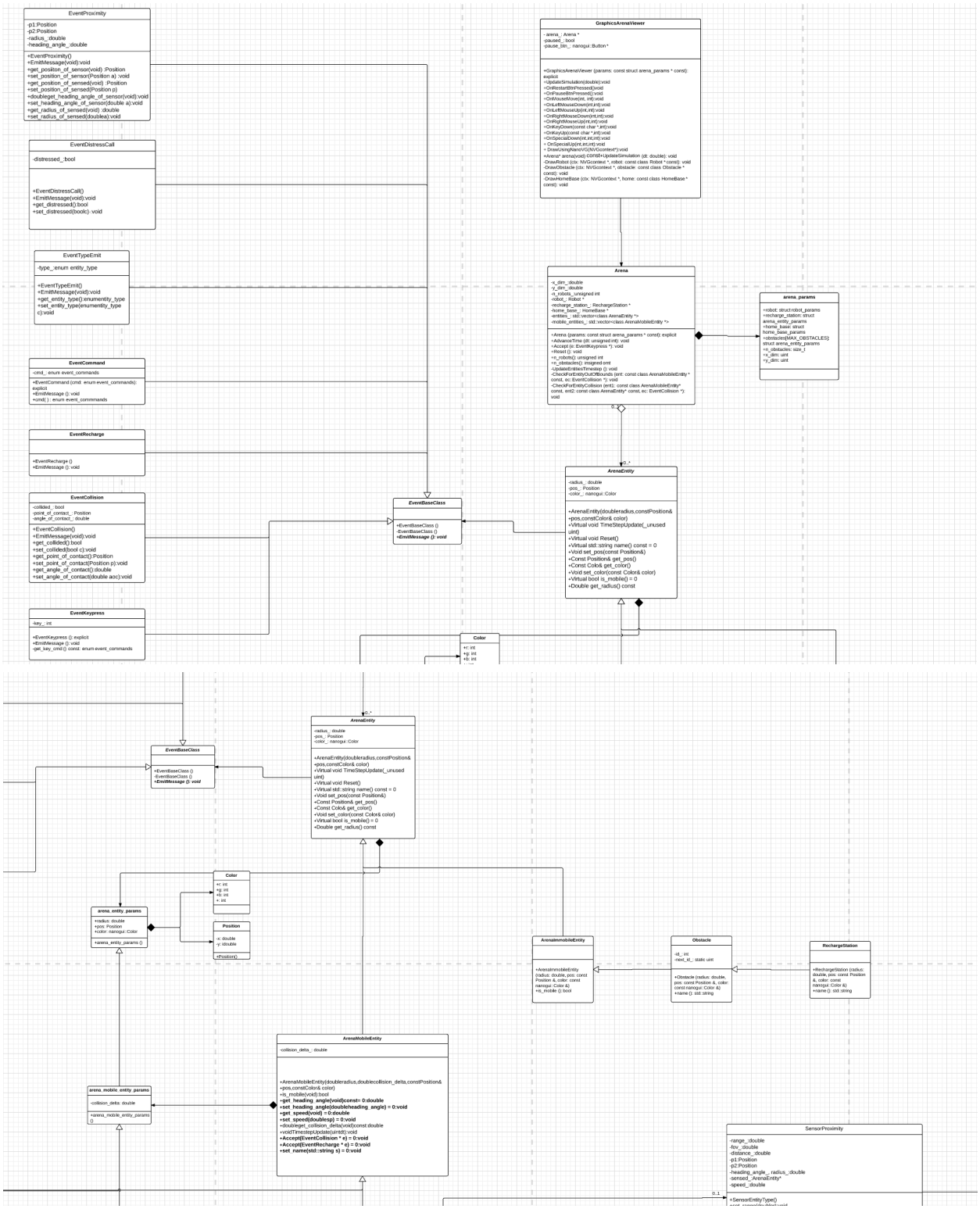
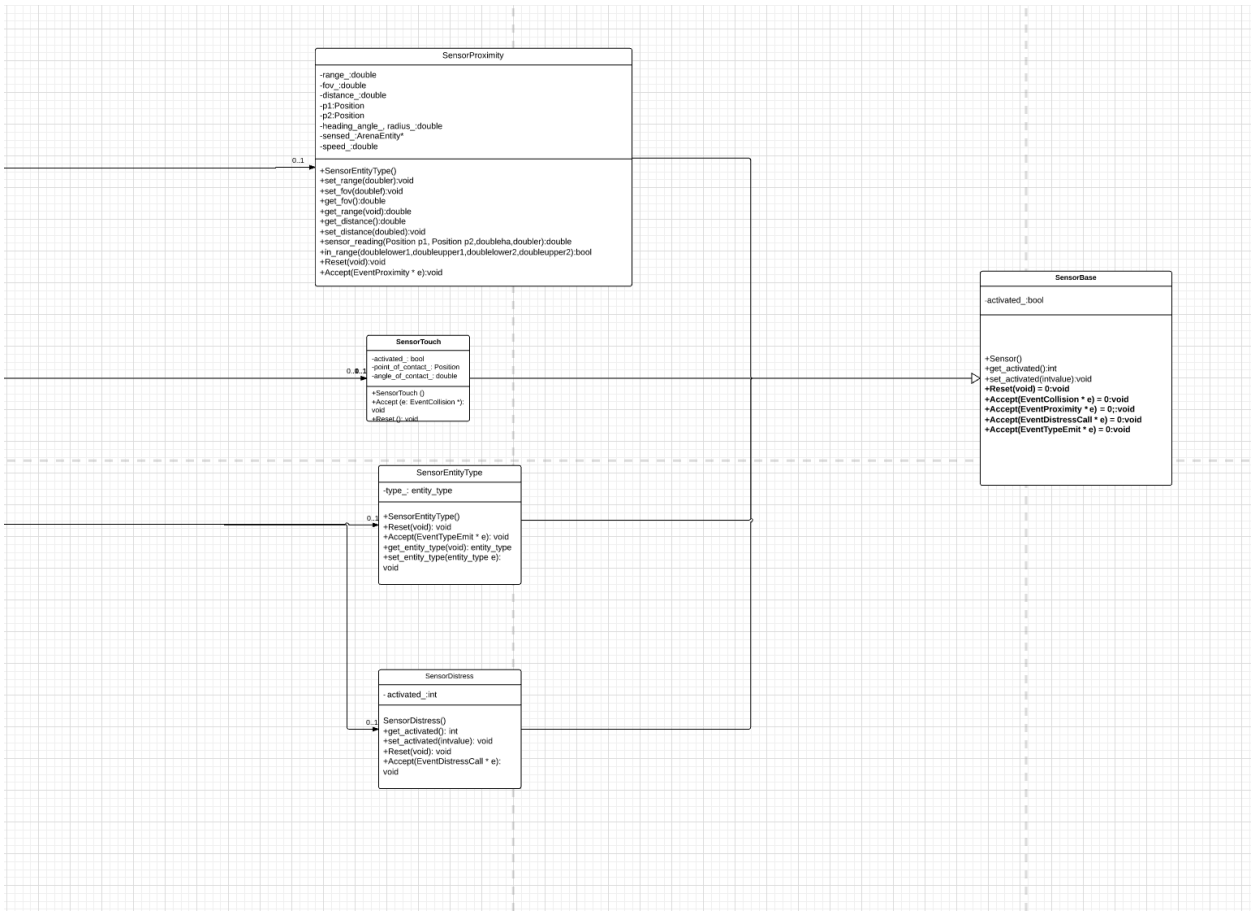
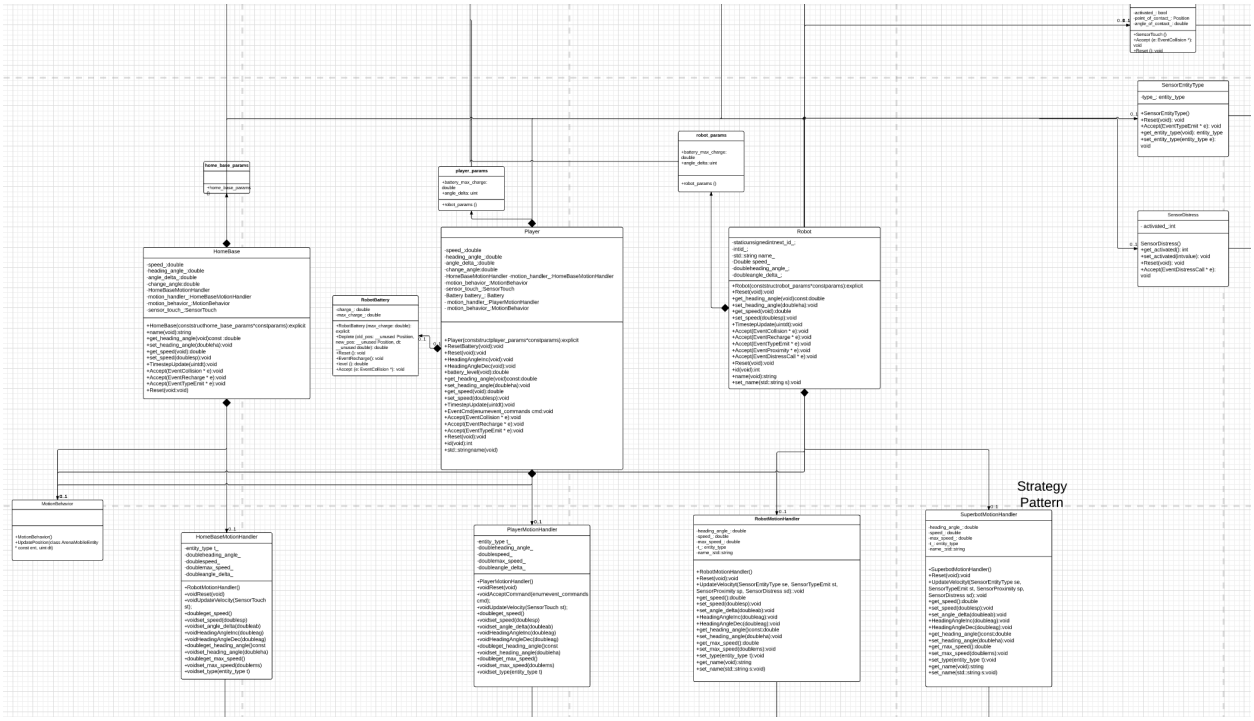
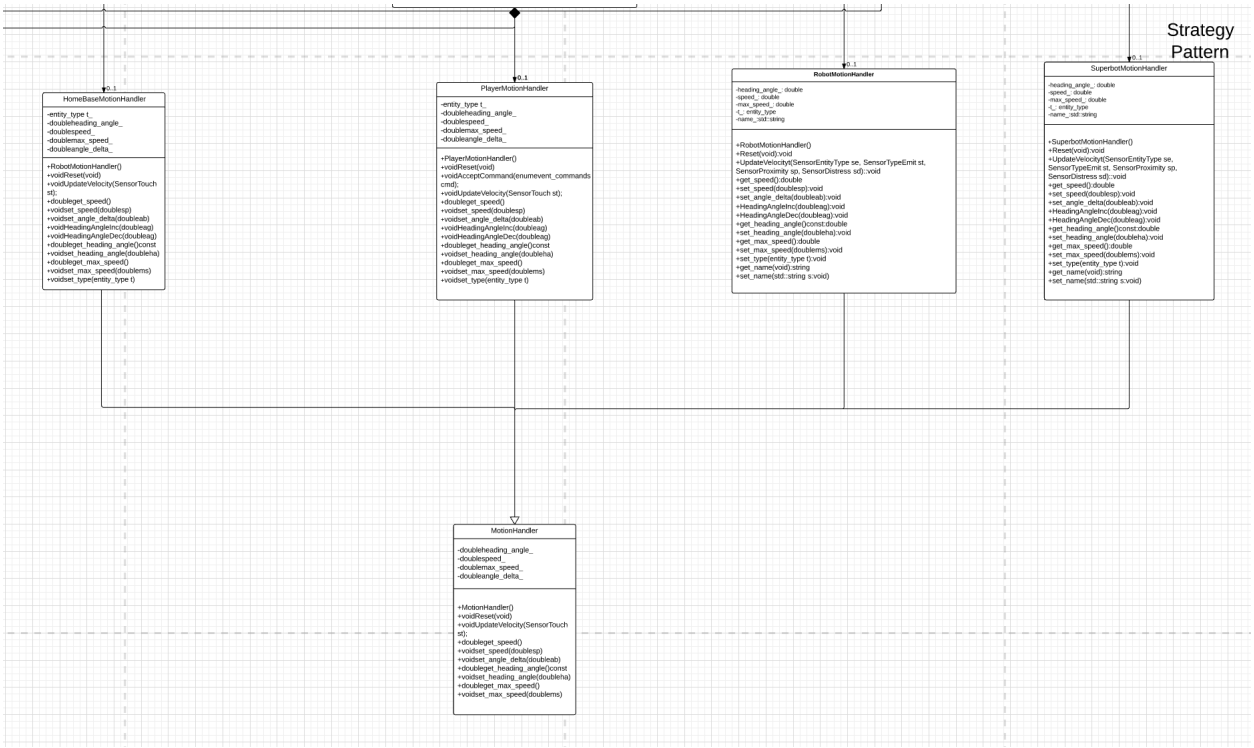
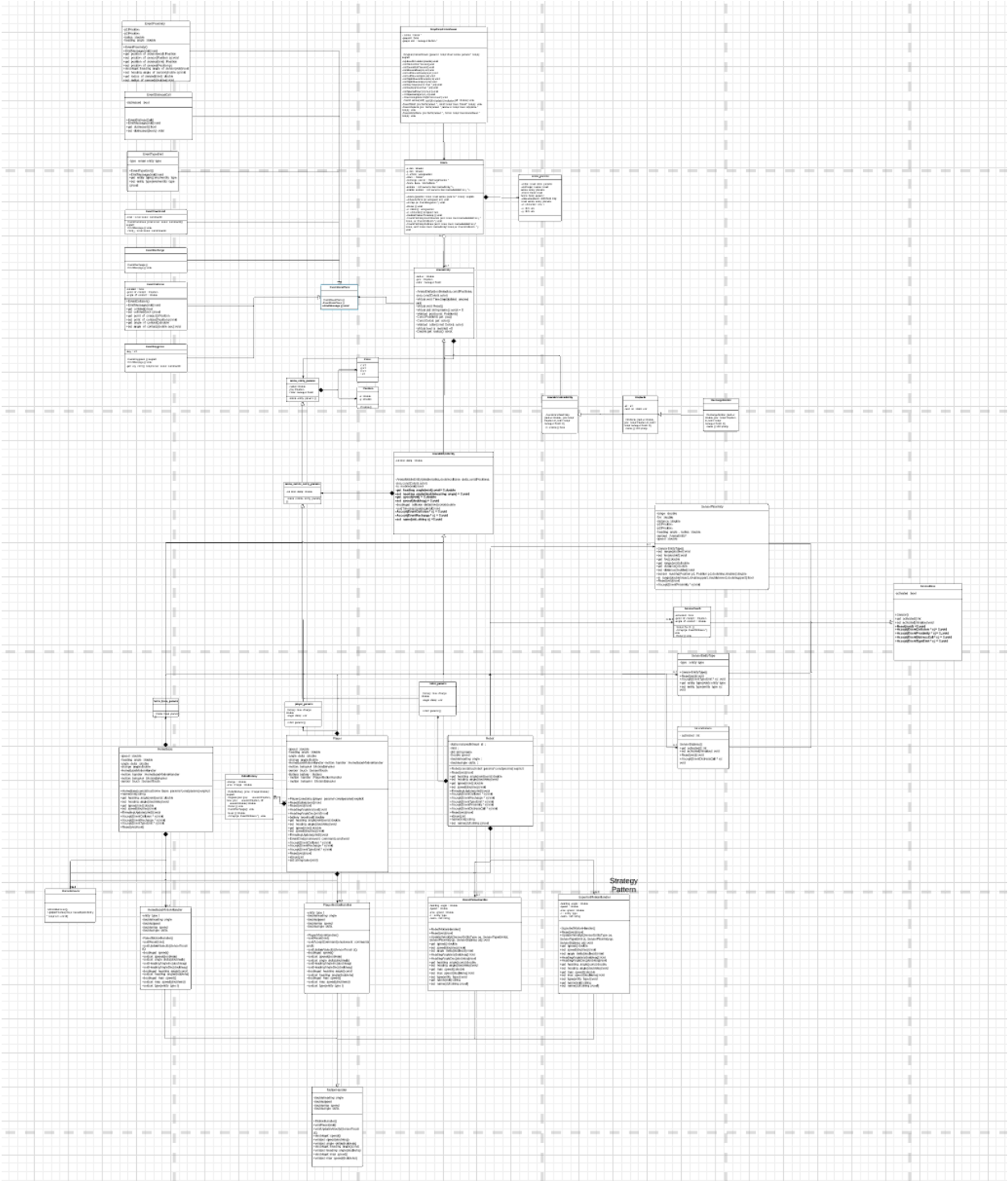

EventProximity
-p1:Position
-p2:Position
-radius_:double
-heading_angle_:double
+EventProximity()
+EmitMessage(void):void
+get_position_of_sensor(void) :Position
+set_position_of_sensor(Position a) :void
+get_position_of_sensed(void) :Position
+set_position_of_sensed(Position p)
+doubleget_heading_angle_of_sensor(void):void
+set_heading_angle_of_sensor(double a):void
+get_radius_of_sensed(void) :double
+set_radius_of_sensed(doublea):void
EventDistressCall
-distressed_:bool
+EventDistressCall()
+EmitMessage(void):void
+get_distressed():bool
+set_distressed(boolc) :void
EventTypeEmit
-type_:enum entity_type
+EventTypeEmit()
+EmitMessage(void):void
+get_entity_type():enumentity_type
+set_entity_type(enumentity_type c):void
EventCommand
-cmd_: enum event_commands
+EventCommand (cmd: enum event_commands): explicit
+EmitMessage (): void
+cmd() : enum event_commmands
EventRecharge
+EventRecharge ()
+EmitMessage (): void
EventCollision
-colided_:bool
-point_of_contact_: Position
-angle_of_contact_: double
+EventCollision()
+EmitMessage(void):void
+get_collided():bool
+set_collided(bool c):void
+get_point_of_contact():Position
+set_point_of_contact(Position p):void
+get_angle_of_contact() double
+set_angle_of_contact(double aoc):void
EventKeypress
-key_: int
+EventKeypress (): explicit
+EmitMessage (): void
-get_key_cmd () :const. enum event_commands
GraphicsArenaViewer
- arena_ : Arena *
-paused_: bool
-pause_btn_: nanogui::Button *
+GraphicsArenaViewer (params: const struct arena_params * const): explicit
+UpdateSimulation(double):void
+OnRestartBtnPressed():void
+OnPauseBtnPressed():void
+OnMouseMove(int, int):void
+OnLeftMouseDown(int,int):void
+OnLeftMouseUp(int,int):void
+OnRightMouseDown(int,int):void
+OnRightMouseUp(int,int):void
+OnKeyDown(const char * ,int):void
+OnKeyUp(const char * ,int):void
+OnSpecialDown(int,int,int):void
+ OnSpecialUp(int,int,int):void
+ DrawUsingNanoVG(NVGcontext*):void
+Arena* arena(void) COnSt&+UpdateSimulation (dt: double): void
-DrawRobot (ctx: NVGcontext *, robot: const class Robot * const): void
-DrawObstacle (ctx: NVGcontext *, obstacle: const class Obstacle * const): void
-DrawHomeBase (ctx: NVGcontext *, home: const class HomeBase * const): void
Arena
-x_dim_:double
-y_dim_:double
-n_robots_:unsigned int
-robot_: Robot *
-recharge_station_: RechargeStation *
-home_base_: HomeBase *
-entities_: std::vector<class ArenaEntity *>
-mobile_entities_: std::vector<class ArenaMobileEntity *>
+Arena (params: const struct arena_params * const): explicit
+AdvanceTime (dt: unsigned int): void
+Accept (e: EventKeypress *): void
+Reset () : void
+n_robots():unsigned int
+n_obstacles():unsigned int
-UpdateEntitiesTimestep () :void
-CheckForEntityOutOfBounds (ent: const class ArenaMobileEntity * const, ec: EventCollision *): void
-CheckForEntityCollision (ent1: const class ArenaMobileEntity* const, ent2: const class ArenaEntity* const, ec: EventCollision *): void
arena_params
+robot: struct robot_params
+recharge_station: struct arena_entity_params
+home_base: struct home_base_params
+obstacles[MAX_OBSTACLES]: struct arena_entity_params
+n_obstacles: size_t
+x_dim: uint
+y_dim: uint
0..1
0..*
0..*
EventBaseClass
+EventBaseClass ()
-EventBaseClass ()
+EmitMessage (): void
ArenaEntity
-radius_: double
-pos_: Position
-color_: nanogui::Color
+ArenaEntity(doubleradius,constPosition& +pos,constColor& color)
+Virtual void TimeStepUpdate(_unused uint)
+Virtual void Reset()
+Virtual std::string name() const = 0
+Void set_pos(const Position&)
+Const Position& get_pos()
+Const Colo& get_color()
+Void set_color(const Color& color)
+Virtual bool is_mobile() = 0
+Double get_radius() const
Color
+r: int
+g: int
+b: int
+ int
0..*
EventBaseClass
+EventBaseClass ()
-EventBaseClass ()
+EmitMessage (): void
ArenaEntity
-radius_: double
-pos_: Position
-color_: nanogui::Color
+ArenaEntity(doubleradius,constPosition& +pos,constColor& color)
+Virtual void TimeStepUpdate(_unused uint)
+Virtual void Reset()
+Virtual std::string name() const = 0
+Void set_pos(const Position&)
+Const Position& get_pos()
+Const Colo& get_color()
+Void set_color(const Color& color)
+Virtual bool is_mobile() = 0
+Double get_radius() const
Color
+r: int
+g: int
+b: int
+ int
arena_entity_params
+radius: double
+pos: Position
+color: nanogui::Color
+arena_entity_params ()
Position
-x: double
-y: double
+Position()
ArenaImmobileEntity
+ArenaImmobileEntity (radius: double, pos: const Position &, color: const nanogui::Color &)
+is_mobile () :bool
Obstacle
-id_: int
-next_id_: static uint
+Obstacle (radius: double, pos: const Position &, color: const nanogui::Color &)
+name () :std::string
RechargeStation
+RechargeStation (radius: double, pos: const Position &, color: const nanogui::Color &)
+name () :std::string
ArenaMobileEntity
-collision_delta_: double
+ArenaMobileEntity(doubleradius,doublecollision_delta,constPosition& +pos,constColor& color)
+is_mobile(void):bool
+get_heading_angle(void)const= 0:double
+set_heading_angle(doubleheading_angle) = 0:void
+get_speed(void) = 0:double
+set_speed(doublesp) = 0:void
+doubleget_collision_delta(void)const:double
+void TimestepUpdate(uint8):void
+Accept(EventCollision * e) = 0:void
+Accept(EventRecharge * e) = 0:void
+set_name(std::string s) = 0:void
arena_mobile_entity_params
-collision_delta: double
+arena_mobile_entity_params ()
SensorProximity
-range_:double
-fov_:double
-distance_:double
-p1:Position
-p2:Position
-heading_angle_ radius_:double
-sensed_:ArenaEntity*
-speed_:double
+SensorEntityType()
0..1


SensorProximity
-range_:double
-fov_:double
-distance_:double
-p1:Position
-p2:Position
-heading_angle_, radius_:double
-sensed_:ArenaEntity*
-speed_:double
+SensorEntityType()
+set_range(doubler):void
+set_fov(doublef):void
+get_fov():double
+get_range(void):double
+get_distance():double
+set_distance(doubled):void
+sensor_reading(Position p1, Position p2,doubleha,doubler):double
+in_range(doublelower1,doubleupper1,doublelower2,doubleupper2):bool
+Reset(void):void
+Accept(EventProximity * e):void
0..1
SensorTouch
-activated_: bool
-point_of_contact_: Position
-angle_of_contact_: double
+SensorTouch ()
+Accept (e: EventCollision *): void
+Reset ():void
0..1
SensorBase
-activated_:bool
+Sensor()
+get_activated():int
+set_activated(intvalue):void
+Reset(void) = 0:void
+Accept(EventCollision * e) = 0:void
+Accept(EventProximity * e) = 0;:void
+Accept(EventDistressCall * e) = 0:void
+Accept(EventTypeEmit * e) = 0:void
SensorEntityType
-type_: entity_type
+SensorEntityType()
+Reset(void): void
+Accept(EventTypeEmit * e): void
+get_entity_type(void): entity_type
+set_entity_type(entity_type e): void
0..1
SensorDistress
- activated_:int
SensorDistress()
+get_activated(): int
+set_activated(intvalue): void
+Reset(void): void
+Accept(EventDistressCall * e): void
0..1


Strategy Pattern
HomeBase
Player
Robot
RobotBattery
SensorEntityType
SensorDistress
MotionBehavior
HomeBaseMotionHandler
PlayerMotionHandler
RobotMotionHandler
SuperbotMotionHandler


Strategy Pattern
HomeBaseMotionHandler
PlayerMotionHandler
RobotMotionHandler
SuperbotMotionHandler
MotionHandler


Strategy Pattern