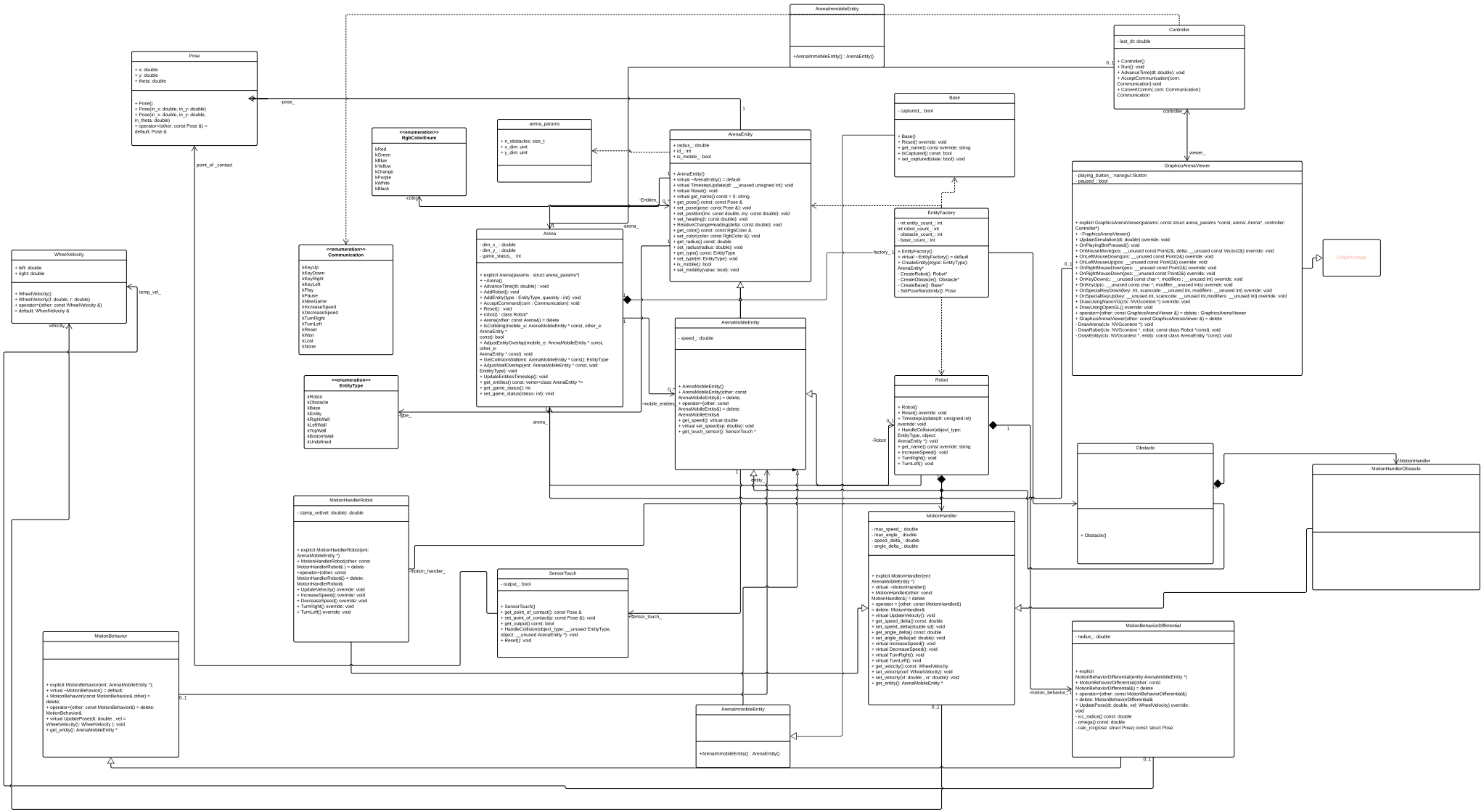
ArenaImmobileEntity
+ArenaImmobileEntity() : ArenaEntity()
Controller
- last_dt: double
+ Controller()
+ Run(): void
+ AdvanceTime(dt: double): void
+ AcceptCommunication(com: Communication):void
+ ConvertComm(com: Communication): Communication
Pose
+ x: double
+ y: double
+ theta: double
+ Pose()
+ Pose(in_x: double, in_y: double)
+ Pose(in_x: double, in_y: double, in_theta: double)
+ operator=(other: const Pose &) = default: Pose &
-pose_
point_of_ _contact
Base
- captured_: bool
+ Base()
+ Reset() override: void
+ get_name() const override: string
+ IsCaptured() const: bool
+ set_captured(state: bool): void
controller_
viewer_
arena_params
+ n_obstacles: size_t
+ x_dim: uint
+ y_dim: uint
<<enumeration>>
RgbColorEnum
kRed
kGreen
kBlue
kYellow
kOrange
kPurple
kWhite
kBlack
-colors_
ArenaEntity
+ radius_: double
+ id_: int
+ is_mobile_: bool
+ ArenaEntity()
+ virtual ~ArenaEntity() = default
+ virtual TimestepUpdate(dt: __unused unsigned int): void
+ virtual Reset(): void
+ virtual get_name() const = 0: string
+ get_pose() const: const Pose &
+ set_pose(pose: const Pose &): void
+ set_position(inx: const double, iny: const double): void
+ set_heading(t: const double): void
+ RelativeChangeHeading(delta: const double): void
+ get_color() const: const RgbColor &
+ set_color(color: const RgbColor &): void
+ get_radius() const: double
+ set_radius(status: double): void
+ get_type() const: EntityType
+ set_type(et: EntityType): void
+ is_mobile(): bool
+ set_mobility(value: bool): void
-Entities_
-arena_
GraphicsArenaViewer
- playing_button_: nanogui::Button
- paused_: bool
+ explicit GraphicsArenaViewer(params: const struct arena_params *const, arena: Arena*, controller: Controller*)
+ ~FraphicsArenaViewer()
+ UpdateSimulation(dt: double) override: void
+ OnPlayingBtnPressed(): void
+ OnMouseMove(pos: __unused const Point2&, delta: __unused const Vector2&) override: void
+ OnLeftMouseDown(pos: __unused const Point2&) override: void
+ OnLeftMouseUp(pos: __unused const Point2&) override: void
+ OnRightMouseDown(pos: __unused const Point2&) override: void
+ OnRightMouseDown(pos: __unused const Point2&) override: void
+ OnKeyDown(c: __unused const char *, modifiers: __unused int) override: void
+ OnKeyUp(c: __unused const char *, modifier: __unused ints) override: void
+ OnSpecialKeyDown(key: int, scancode: __unused int, modifiers: __unused int) override: void
+ OnSpecialKeyUp(key: __unused int, scancode: __unused int,modifiers: __unused int) override: void
+ DrawUsingNanoVG(ctx: NVGcontext *) override: void
+ DrawUsingOpenGL() override: void
+ operator=(other: const GraphicsArenaViewer &) = delete : GraphicsArenaViewer
+ GraphicsArenaViewer(other: const GraphicsArenaViewer &) = delete
- DrawArena(ctx: NVGcontext *): void
- DrawRobot(ctx: NVGcontext *, robot: const class Robot *const): void
- DrawEntity(ctx: NVGcontext *, entity: const class ArenaEntity *const): void
GraphicsApp
WheelVelocity
+ left: double
+ right: double
+ WheelVelocity()
+ WheelVelocity(l: double, r: double)
+ operator=(other: const WheelVelocity &) = default: WheelVelocity &
velocity_
temp_vel_
<<enumeration>>
Communication
kKeyUp
kKeyDown
kKeyRight
kKeyLeft
kPlay
kPause
kNewGame
kIncreaseSpeed
kDecreaseSpeed
kTurnRight
kTurnLeft
kReset
kWon
kLost
kNone
Arena
- dim_x_: double
- dim_y_: double
- game_status_: int
+ explicit Arena(params : struct arena_params*)
+ ~Arena()
+ AdvanceTime(dt: double) : void
+ AddRobot(): void
+ AddEntity(type : EntityType, quantity : int): void
+ AcceptCommand(com : Communication): void
+ Reset() : void
+ robot() : class Robot*
+ Arena(other: const Arena&) = delete
+ IsColliding(mobile_e: ArenaMobileEntity * const, other_e: ArenaEntity * const): bool
+ AdjustEntityOverlap(mobile_e: ArenaMobileEntity * const, other_e: ArenaEntity * const): void
+ GetCollisionWall(ent: ArenaMobileEntity * const): EntityType
+ AdjustWallOverlap(ent: ArenaMobileEntity * const, wall: EntityType): void
+ UpdateEntitiesTimestep(): void
+ get_entities() const: vector<class ArenaEntity *>
+ get_game_status(): int
+ set_game_status(status : int): void
EntityFactory
- int entity_count_: int
int robot_count_: int
- obstacle_count_: int
- base_count_: int
+ EntityFactory()
+ virtual ~EntityFactory() = default
+ CreateEntity(etype: EntityType): ArenaEntity*
- CreateRobot(): Robot*
- CreateObstacle(): Obstacle*
- CreateBase(): Base*
- SetPoseRandomly(): Pose
-factory_
<<enumeration>>
EntityType
kRobot
kObstacle
kBase
kEmtiy
kRightWall
kLeftWall
kTopWall
kBottomWall
kUndefined
-type_
arena_
ArenaMobileEntity
- speed_: double
+ ArenaMobileEntity()
+ ArenaMobileEntity(other: const ArenaMobileEntity&) = delete;
+ operator=(other: const ArenaMobileEntity&) = delete: ArenaMobileEntity&
+ get_speed(): virtual double
+ virtual set_speed(sp: double): void
+ get_touch_sensor(): SensorTouch *
-mobile_entities_
Robot
+ Robot()
+ Reset() override: void
+ TimestepUpdate(dt: unsigned int) override: void
+ HandleCollision(object_type: EntityType, object: ArenaEntity *): void
+ get_name() const override: string
+ IncreaseSpeed(): void
+ TurnRight(): void
+ TurnLeft(): void
-Robot
Obstacle
+ Obstacle()
MotionHandler
MotionHandlerObstacle
-entity_
MotionHandlerRobot
- clamp_vel(vel: double): double
+ explicit MotionHandlerRobot(ent: ArenaMobileEntity *)
+ MotionHandlerRobot(other: const MotionHandlerRobot&) = delete
+operator=(other: const MotionHandlerRobot&) = delete: MotionHandlerRobot&
+ UpdateVelocity() override: void
+ IncreaseSpeed() override: void
+ DecreaseSpeed() override: void
+ TurnRight() override: void
+ TurnLeft() override: void
-motion_handler_
SensorTouch
- output_: bool
+ SensorTouch()
+ get_point_of_contact(): const Pose &
+ set_point_of_contact(p: const Pose &): void
+ get_output() const: bool
+ HandleCollision(object_type: __unused EntityType, object: __unused ArenaEntity *): void
+ Reset(): void
-sensor_touch_
MotionHandler
- max_speed_: double
- max_angle_: double
- speed_delta_: double
- angle_delta_: double
+ explicit MotionHandler(ent: ArenaMobileEntity *)
+ virtual ~MotionHandler()
+ MotionHandler(other: const MotionHandler&) = delete
+ operator = (other: const MotionHandler&) = delete: MotionHandler&
+ virtual UpdateVelocity(): void
+ get_speed_delta() const: double
+ set_speed_delta(double sd): void
+ get_angle_delta() const: double
+ set_angle_delta(ad: double): void
+ virtual IncreaseSpeed(): void
+ virtual DecreaseSpeed(): void
+ virtual TurnRight(): void
+ virtual TurnLeft(): void
+ get_velocity() const: WheelVelocity
+ set_velocity(vel: WheelVelocity): void
+ set_velocity(vl: double , vr: double): void
+ get_entity(): ArenaMobileEntity *
-motion_behavior_
MotionBehavior
+ explicit MotionBehavior(ent: ArenaMobileEntity *);
+ virtual ~MotionBehavior() = default;
+ MotionBehavior(const MotionBehavior& other) = delete;
+ operator=(other: const MotionBehavior&) = delete: MotionBehavior&
+ virtual UpdatePose(dt: double , vel = WheelVelocity(): WheelVelocity): void
+ get_entity(): ArenaMobileEntity *
MotionBehaviorDifferential
- radius_: double
+ explicit MotionBehaviorDifferential(entity:ArenaMobileEntity *)
+ MotionBehaviorDifferential(other: const MotionBehaviorDifferential&) = delete
+ operator=(other: const MotionBehaviorDifferential&) = delete: MotionBehaviorDifferential&
+ UpdatePose(dt: double, vel: WheelVelocity) override: void
- icc_radius() const: double
- omega() const: double
- calc_icc(pose: struct Pose) const: struct Pose
ArenaImmobileEntity
+ArenaImmobileEntity() : ArenaEntity()
1..1
0..1
1
0..*
1..*
1..1
0..1
0..1
0..1
0..1
0..1
0..1
0..1
1
1