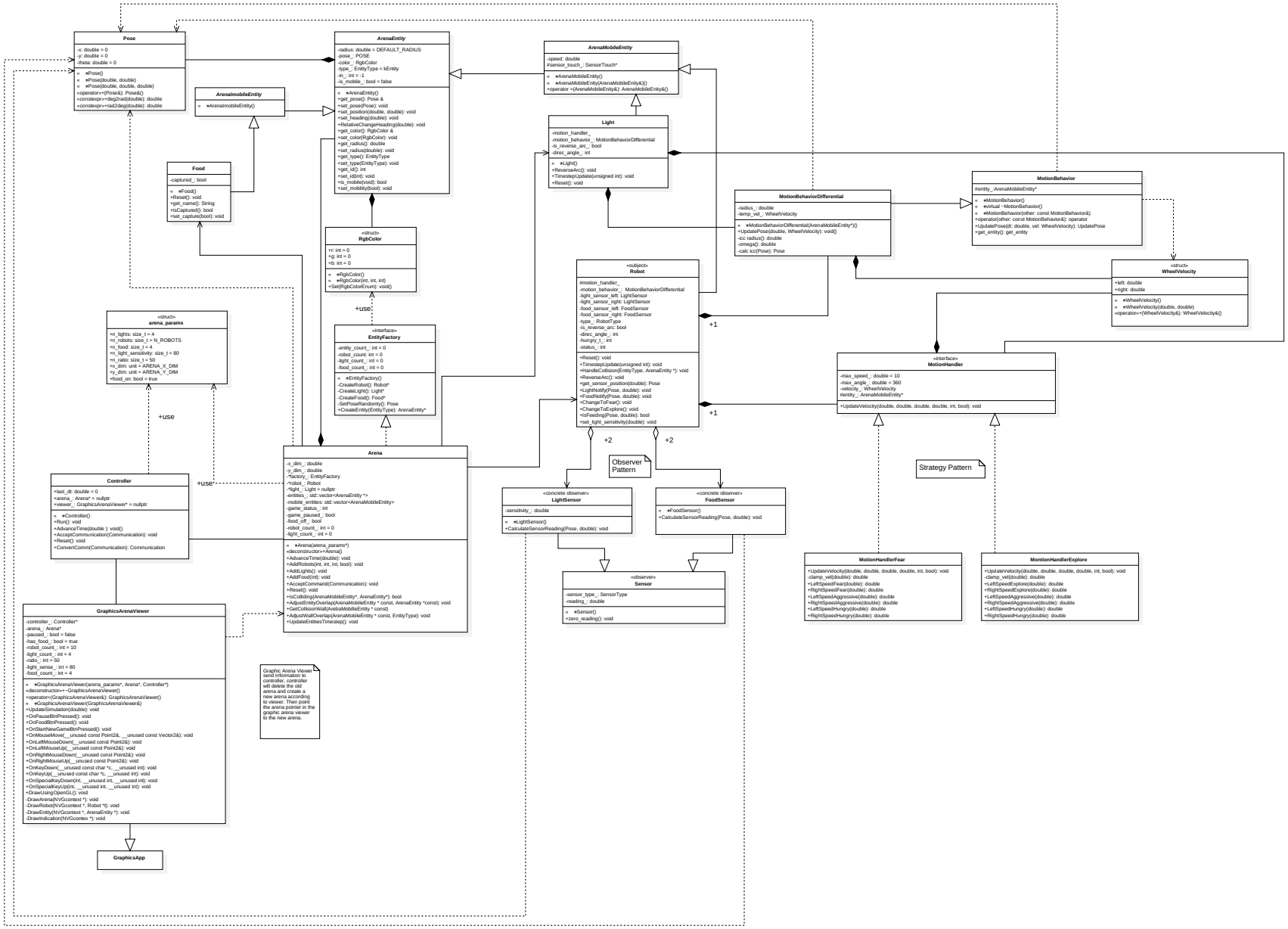
Pose
-x: double = 0
-y: double = 0
-theta: double = 0
= default()
= default(double, double)
= default(double, double, double)
+operator+=(Pose&): Pose&()
+constexpr+=deg2rad(double): double
+constexpr+=rad2deg(double): double
ArenaMobileEntity
= default()
ArenaEntity
-radius: double = DEFAULT_RADIUS
-pose_: POSE
-color_: RgbColor
-type_: EntityType = kEntity
-in_: int = -1
-is_mobile_: bool = false
= default()
+get_pose(): Pose &
+set_pose(Pose): void
+set_position(double, double): void
+set_heading(double): void
+RelativeChangeHeading(double): void
+get_color(): RgbColor &
+set_color(RgbColor): void
+get_radius(): double
+set_radius(double): void
+get_type(): EntityType
+set_type(EntityType): void
+get_id(): int
+set_id(int): void
+is_mobile(void): bool
+set_mobility(bool): void
ArenaMobileEntity
-speed: double
#sensor_touch_: SensorTouch*
= default()
= default(ArenaMobileEntity&)
+operator =(ArenaMobileEntity&): ArenaMobileEntity&()
Food
-captured_: bool
= default()
+Reset(): void
+get_name(): String
+isCaptured(): bool
+set_capture(bool): void
Light
-motion_handler_
-motion_behavior_: MotionBehaviorDifferential
-is_reverse_arc_: bool
-direc_angle_: int
= default()
+ReverseArc(): void
+Timestep()Update(unsigned int): void
+Reset(): void
«struct»
RgbColor
+r: int = 0
+g: int = 0
+b: int = 0
= default()
= default(int, int, int)
+Set(RgbColorEnum): void()
MotionBehaviorDifferential
-radius_: double
-temp_vel_: WheelVelocity
= default(MotionBehaviorDifferential(ArenaMobileEntity*)()
+UpdatePose(double, WheelVelocity): void()
-icc radius(): double
-omega(): double
-calc icc(Pose): Pose
MotionBehavior
#entity_:ArenaMobileEntity*
= default()
= default ~MotionBehavior()
= default(other: const MotionBehavior&)
+operator(other: const MotionBehavior&): operator
+UpdatePose(dt: double, vel: WheelVelocity): UpdatePose
+get_entity(): get_entity
«struct»
WheelVelocity
+left: double
+right: double
= default()
= default(double, double)
+operator+=(WheelVelocity&): WheelVelocity&()
«struct»
arena_params
+n_lights: size_t = 4
+n_robots: size_t = N_ROBOTS
+n_food: size_t = 4
+n_light_sensitivity: size_t = 80
+n_ratio: size_t = 50
+x_dim: unit = ARENA_X_DIM
+y_dim: unit = ARENA_Y_DIM
+food_on: bool = true
«interface»
EntityFactory
-entity_count_: int = 0
-robot_count: int = 0
-light_count_: int = 0
-food_count_: int = 0
= default()
-CreateRobot(): Robot*
-CreateLight(): Light*
-CreateFood(): Food*
-SetPoseRandomly(): Pose
+CreateEntity(EntityType): ArenaEntity*
«subject»
Robot
#motion_handler_
-motion_behavior_: MotionBehaviorDifferential
-light_sensor_left: LightSensor
-light_sensor_right: LightSensor
-food_sensor_left: FoodSensor
-food_sensor_right: FoodSensor
-type_: RobotType
-is_reverse_arc: bool
-direc_angle_: int
-hungry_t_: int
-status_: int
+Reset(): void
+Timestep()Update(unsigned int): void
+HandleCollision(EntityType, ArenaEntity *): void
+ReverseArc(): void
+get_sensor_position(double): Pose
+LightNotify(Pose, double): void
+FoodNotify(Pose, double): void
+ChangeToFear(): void
+ChangeToExplore(): void
+IsFeeding(Pose, double): bool
+set_light_sensitivity(double): void
«interface»
MotionHandler
-max_speed_: double = 10
-max_angle_: double = 360
-velocity_: WheelVelocity
#entity_: ArenaMobileEntity*
+UpdateVelocity(double, double, double, double, int, bool): void
+1
+1
+2
+2
Observer Pattern
Strategy Pattern
+use
+use
+use
Controller
+last_dt: double = 0
+arena_: Arena* = nullptr
+viewer_: GraphicsArenaViewer* = nullptr
= default()
+Run(): void
+AdvanceTime(double): void()
+AcceptCommunication(Communication): void
+Reset(): void
+ConvertComm(Communication): Communication
Arena
-x_dim_: double
-y_dim_: double
-*factory_: EntityFactory
-*robot_: Robot
-*light_: Light = nullptr
-entities_: std::vector<ArenaEntity *>
-mobile_entities: std::vector<ArenaMobileEntity>
-game_status_: int
-game_paused_: bool
-food_off_: bool
-robot_count_: int = 0
-light_count_: int = 0
= default(arena_params*)
«deconstructor»+Arena()
+AdvanceTime(double): void
+AddRobots(int, int, int, bool): void
+AddLights(): void
+AddFood(int): void
+AcceptCommand(Communication): void
+Reset(): void
+IsCollidingArenaMobileEntity*, ArenaEntity*): bool
+AdjustEntityOverlap(ArenaMobileEntity * const, ArenaEntity *const): void
+GetCollisionWallArenaMobileEntity * const)
+AdjustWallOverlap(ArenaMobileEntity * const, EntityType): void
+UpdateEntitiesTimestep(): void
«concrete observer»
LightSensor
-sensitivity_: double
= default()
+CalculateSensorReading(Pose, double): void
«concrete observer»
FoodSensor
= default()
+CalculateSensorReading(Pose, double): void
«observer»
Sensor
-sensor_type_: SensorType
-reading_: double
= default()
+zero_reading(): void
MotionHandlerFear
+UpdateVelocity(double, double, double, double, int, bool): void
-clamp_vel(double): double
+LeftSpeedFear(double): double
+RightSpeedFear(double): double
+LeftSpeedAggressive(double): double
+RightSpeedAggressive(double): double
+LeftSpeedHungry(double): double
+RightSpeedHungry(double): double
MontionHandlerExplore
+UpdateVelocity(double, double, double, double, int, bool): void
-clamp_vel(double): double
+LeftSpeedExplore(double): double
+RightSpeedExplore(double): double
+LeftSpeedAggressive(double): double
+RightSpeedAggressive(double): double
+LeftSpeedHungry(double): double
+RightSpeedHungry(double): double
GraphicsArenaViewer
-controller_: Controller*
-arena_: Arena*
-paused_: bool = false
-has_food_: bool = true
-robot_count_: int = 10
-light_count_: int = 4
-ratio_: int = 50
-light_sense_: int = 80
-food_count_: int = 4
= default(GraphicsArenaViewer(arena_params*, Arena*, Controller*)
«deconstructor»+~GraphicsArenaViewer()
+operator=(GraphicsArenaViewer&): GraphicsArenaViewer()
= default(GraphicsArenaViewer(GraphicsArenaViewer&)
+UpdateSimulation(double): void
+OnPauseBtnPressed(): void
+OnFoodBtnPressed(): void
+OnStartNewGameBtnPressed(): void
+OnMouseMove(__unused const Point2&, __unused const Vector2&): void
+OnLeftMouseDown(__unused const Point2&): void
+OnLeftMouseUp(__unused const Point2&): void
+OnRightMouseDown(__unused const Point2&): void
+OnRightMouseUp(__unused const Point2&): void
+OnKeyDown(__unused const char *c, __unused int): void
+OnKeyUp(__unused const char *c, __unused int): void
+OnSpecialKeyDown(int, __unused int, __unused int): void
+OnSpecialKeyUp(int, __unused int, __unused int): void
+DrawUsingOpenGL(): void
-DrawArena(NVGcontext *): void
-DrawRobot(NVGcontext *, Robot *): void
-DrawEntity(NVGcontext *, ArenaEntity *): void
-DrawIndication(NVGcontex *): void
GraphicsApp
Graphic Arena Viewer
send information to
controller, controller
will delete the old
arena and create a
new arena according
to viewer. Then point
the arena pointer in the
graphic arena viewer
to the new arena.