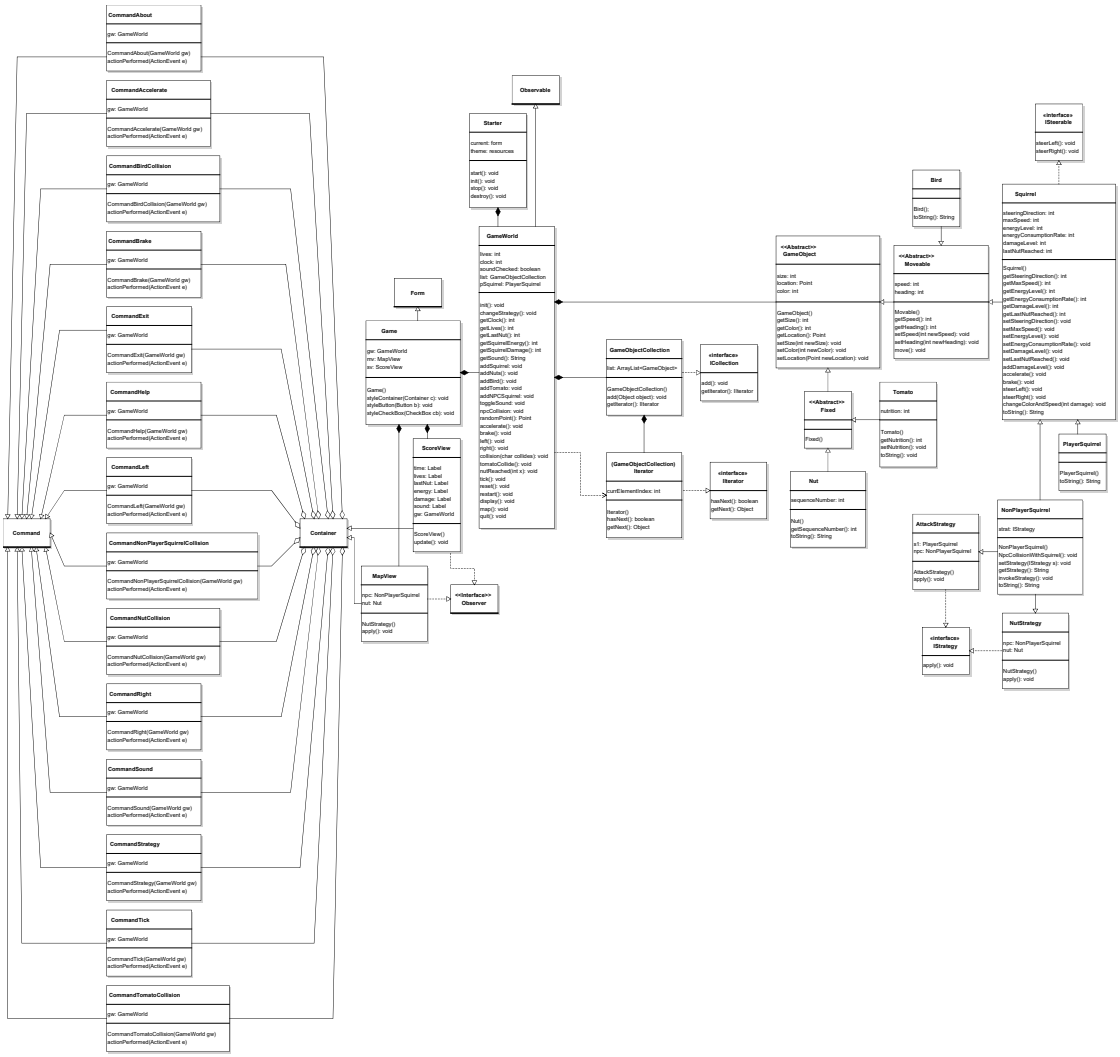
CommandAbout
gw: GameWorld
CommandAbout(GameWorld gw)
actionPerformed(ActionEvent e)
CommandAccelerate
gw: GameWorld
CommandAccelerate(GameWorld gw)
actionPerformed(ActionEvent e)
CommandBirdCollision
gw: GameWorld
CommandBirdCollision(GameWorld gw)
actionPerformed(ActionEvent e)
CommandBrake
gw: GameWorld
CommandBrake(GameWorld gw)
actionPerformed(ActionEvent e)
CommandExit
gw: GameWorld
CommandExit(GameWorld gw)
actionPerformed(ActionEvent e)
CommandHelp
gw: GameWorld
CommandHelp(GameWorld gw)
actionPerformed(ActionEvent e)
CommandLeft
gw: GameWorld
CommandLeft(GameWorld gw)
actionPerformed(ActionEvent e)
Command
CommandNonPlayerSquirrelCollision
gw: GameWorld
CommandNonPlayerSquirrelCollision(GameWorld gw)
actionPerformed(ActionEvent e)
Container
CommandNutCollision
gw: GameWorld
CommandNutCollision(GameWorld gw)
actionPerformed(ActionEvent e)
CommandRight
gw: GameWorld
CommandRight(GameWorld gw)
actionPerformed(ActionEvent e)
CommandSound
gw: GameWorld
CommandSound(GameWorld gw)
actionPerformed(ActionEvent e)
CommandStrategy
gw: GameWorld
CommandStrategy(GameWorld gw)
actionPerformed(ActionEvent e)
CommandTick
gw: GameWorld
CommandTick(GameWorld gw)
actionPerformed(ActionEvent e)
CommandTomatoCollision
gw: GameWorld
CommandTomatoCollision(GameWorld gw)
actionPerformed(ActionEvent e)
Observable
Starter
current: form
theme: resources
start(): void
init(): void
stop(): void
destroy(): void
GameWorld
lives: int
clock: int
soundChecked: boolean
list: GameObjectCollection
pSquirrel: PlayerSquirrel
init(): void
changeStrategy(): void
getClock(): int
getLives(): int
getLastNut(): int
getSquirrelEnergy(): int
getSquirrelDamage(): int
getSound(): String
addSquirrel: void
addNut(): void
addBird(): void
addTomato: void
addNPCSquirrel: void
toggleSound: void
npcCollision: void
randomPoint(): Point
accelerate(): void
brake(): void
left(): void
right(): void
collision(char collidee): void
tomatoCollide(): void
nutReached(int x): void
tick(): void
reset(): void
restart(): void
display(): void
map(): void
quit(): void
Form
Game
gw: GameWorld
mv: MapView
sv: ScoreView
Game()
styleContainer(Container c): void
styleButton(Button b): void
styleCheckBox(CheckBox cb): void
ScoreView
time: Label
lives: Label
lastNut: Label
energy: Label
damage: Label
sound: Label
gw: GameWorld
ScoreView()
update(): void
MapView
npc: NonPlayerSquirrel
nut: Nut
NutStrategy()
apply(): void
<<Interface>>
Observer
GameObjectCollection
list: ArrayList<GameObject>
GameObjectCollection()
add(Object object): void
getIterator(): Iterator
<interface>
ICollection
add(): void
getIterator(): Iterator
(GameObjectCollection)
Iterator
curElementIndex: int
Iterator()
hasNext(): boolean
getNext(): Object
<interface>
IIterator
hasNext(): boolean
getNext(): Object
<<Abstract>>
GameObject
size: int
location: Point
color: int
GameObject()
getSize(): int
getColor(): int
getLocation(): Point
setSize(int newSize): void
setColor(int newColor): void
setLocation(Point newLocation): void
<<Abstract>>
Fixed
Fixed()
Nut
sequenceNumber: int
Nut()
getSequenceNumber(): int
toString(): String
Bird
Bird();
toString(): String
<<Abstract>>
Moveable
speed: int
heading: int
Moveable()
getSpeed(): int
getHeading(): int
setSpeed(int newSpeed): void
setHeading(int newHeading): void
move(): void
Tomato
nutrition: int
Tomato()
getNutrition(): int
setNutrition(): void
toString(): void
<interface>
ISteerable
steerLeft(): void
steerRight(): void
Squirrel
steeringDirection: int
maxSpeed: int
energyLevel: int
energyConsumptionRate: int
damageLevel: int
lastNutReached: int
Squirrel()
getSteeringDirection(): int
getMaxSpeed(): int
getEnergyLevel(): int
getEnergyConsumptionRate(): int
getDamageLevel(): int
getLastNutReached(): int
setSteeringDirection(): void
setMaxSpeed(): void
setEnergyLevel(): void
setEnergyConsumptionRate(): void
setDamageLevel(): void
setLastNutReached(): void
addDamageLevel(): void
accelerate(): void
brake(): void
steerLeft(): void
steerRight(): void
changeColorAndSpeed(int damage): void
toString(): String
PlayerSquirrel
PlayerSquirrel()
toString(): String
NonPlayerSquirrel
strat: IStrategy
NonPlayerSquirrel()
NpcCollisionWithSquirrel(): void
setStrategy(IStrategy s): void
getStrategy(): String
invokeStrategy(): void
toString(): String
AttackStrategy
s1: PlayerSquirrel
npc: NonPlayerSquirrel
AttackStrategy()
apply(): void
<interface>
IStrategy
apply(): void
NutStrategy
npc: NonPlayerSquirrel
nut: Nut
NutStrategy()
apply(): void