Staroids, Module Interface Specification

Team 20, Staroids
Moziah San Vicente, 400091284, sanvicem
Eoin Lynagh, 400067675, lynaghe
Jason Nagy, 400055130, nagyj2

November 9, 2018

The following is a series of MISes for the modules that comprise the Staroids game

Table 1: **Revision History**

Date	**Version**	**Notes**
Nov 06/18	0.1	Added basic information to template
Nov 07/18	0.2	Added Head module specification
Nov 08/18	0.3	Added all module specifications
Nov 09/18	0.35	Tidied up
Nov 09/18	0.5	Finished Sound, Utilites, Head and Game State MIS

Utilities Module

Template Module

Utilities

Uses

CVS from Browser (Playing screen)
CTX from CVS (Screen coordinate system)
FONTSTYLE from Browser (Available fonts for printing)

Syntax

Exported Types

FPS=30
SHIP_SIZE=30
TURN_SPEED=180
SHIP_THRUST=0.2
SHIP_BREAK=0.98
MIN_SPEED=0.1
MAX_SPEED=20
MAX_ACC=2
CVS_WIDTH=780
CVS_HEIGHT=620
BULLET_EXTRA=5
KILLABLE={True,False}
MAX_ASTEROIDS=2
TEST={True,False}
ALIEN_SPAWN=700
KeyCode={UP,DOWN,RIGHT,LEFT,SPACE,M,P,R}
EPOCH=1
Key=?
Text=?
Game=?

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
Key		Key	
isDown	KeyCode	$\mathbb{N}$	
onKeydown	KeyCode	$\mathbb{N}$	
onKeyup	KeyCode		

Semantics

State Variables

d: sequence of $\mathbb{N}$

State Invariant

$\forall(c : \mathbb{N} | c \in d : c > 0)$

Assumptions

- Only known keys (as defined by KeyCode) will be put into the Key object as events to be processed.

Access Routine Semantics

Key():

- transition: $d :=$ seq of KeyCode
- output: $out := Key$
- exception: None

isDown(e):

- output: $e \in d \Rightarrow true \wedge e \notin d \Rightarrow false$
- exception: None

onKeydown(e):

- transition: $d[e] =$ EPOCH

- exception: None

onKeyup(e):

- output: $out := d[e]$
- exception: None

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
TEXT	CTX, FONTSTYLE	TEXT	
norm	$String, \mathbb{Z}, \mathbb{Z}$		
emph	$String, \mathbb{Z}, \mathbb{Z}$		

Semantics

State Variables

cvs: CTX fnt: FONTSTYLE

State Invariant

None

Assumptions

- Before the Text object is used, the initialization function must be run first.

Access Routine Semantics

norm(Str, x, y):

- transition: Displays Str to cvs at location (x, y) in standard font.
- exception: None

emph(Str, x, y):

- transition: Displays Str to cvs at location (x, y) in emphasized font.
- exception: None

Routine name	**In**	**Out**	**Exceptions**
Game			
reduceCounter	$String, \mathbb{Z}, \mathbb{Z}$		
resetMute			
resetPause			
drawLives			
addScore	$\mathbb{Z}$		
addSprites	OBJECT		
subLives	$\mathbb{Z}$		
subSprites	OBJECT		
getScore		$\mathbb{N}$	
getLives		$\mathbb{N}$	
getLevel		$\mathbb{N}$	
getAsteroids		$\mathbb{N}$	
getWidth		$\mathbb{N}$	
getHeight		$\mathbb{N}$	
getCvs		CVS	
getCtx		CTX	
getSprites		sequence of OBJECT	
getPlayer		PLAYER	
getAlien		ALIEN	
getText		TEXT	
getSound		SOUND	
getPaused		$\mathbb{B}$	
setScore	$\mathbb{N}$		
setLives	$\mathbb{N}$		
setLevel	$\mathbb{N}$		
setAsteroids	$\mathbb{N}$		
setWidth	$\mathbb{N}$		
setHeight	$\mathbb{N}$		
setCvs	CVS		
setCtx	CTX		
setSprites	sequence of OBJECT		
setPlayer	PLAYER		
setAlien	ALIEN		
setText	TEXT		
setSound	SOUND		
setPaused	$\mathbb{B}$		

Semantics

State Variables

$score$: $\mathbb{N}$
$lives$: $\mathbb{N}$
$level$: $\mathbb{N}$
$asteroids$: $\mathbb{N}$
$width$: $\mathbb{N}$
$height$: $\mathbb{N}$
cvs: CVS
ctx: CTX
$sprites$: sequence of OBJECT
$player$: PLAYER
$alien$: ALIEN
$text$: TEXT
$sound$: SOUND
$paused$: $\mathbb{B}$
$muteSound$: $\mathbb{N}$
$pauseGame$: $\mathbb{N}$

State Invariant

None

Assumptions

None

Access Routine Semantics

Game():

- transition: $score = 0 \wedge lives = 3 \wedge sprites = seq.of\text{OBJECT} \wedge muteSound = FPS \wedge pauseGame = FPS$
- exception: None

getScore():

- output: $out := score$

- exception: None

getLives():

- output: $out := lives$
- exception: None

getLevel():

- output: $out := level$
- exception: None

getAsteroids():

- output: $out := asteroids$
- exception: None

getWidth():

- output: $out := width$
- exception: None

getHeight():

- output: $out := height$
- exception: None

getCvs():

- output: $out := cvs$
- exception: None

getCtx():

- output: $out := ctx$
- exception: None

getSprites():

- output: $out := sprites$

- exception: None

getPlayer():

- output: $out := player$
- exception: None

getAlien():

- output: $out := alien$
- exception: None

getText():

- output: $out := text$
- exception: None

getSound():

- output: $out := sound$
- exception: None

getPaused():

- output: $out := paused$
- exception: None

setScore(s):

- transition: $score = s$
- exception: None

setLives(l):

- transition: $lives = l$
- exception: None

setLevel(l):

- transition: $level = l$

- exception: None

setAsteroids(a):

- transition: $asteroids = a$
- exception: None

setWidth(w):

- transition: $width = w$
- exception: None

getHeight(h):

- transition: $height = h$
- exception: None

setCvs(c):

- transition: $cvs = c$
- exception: None

setCtx(c):

- transition: $cyx = c$
- exception: None

setSprites(s):

- transition: $sprites = s$
- exception: None

setPlayer(p):

- transition: $player = p$
- exception: None

setAlien():

- transition: $alien = a$

- exception: None

setText(t):

- transition: $text = t$
- exception: None

setSound(s):

- transition: $sound = s$
- exception: None

setPaused(b):

- transition: $paused = b$
- exception: None

reduceCounter():

- transition: $muteSound := muteSound - 1 \wedge pauseGame := pauseGame - 1$
- exception: None

resetMute():

- transition: $muteSound = FPS$
- exception: None

resetPause():

- transition: $pauseGame = FPS$
- exception: None

drawLives():

- transition: $\forall(i : \mathbb{N} | i < lives : drawTriangle(i * 15))$
- exception: None

addScore(amount):

- transition: $score + amount$

- exception: None

addSprite(obj):

- transition: $sprites = sprites || obj$
- exception: None

subLives(obj):

- transition: $lives - 1$
- exception: None

subSprite(obj):

- transition: $sprites = sprites \setminus obj$
- exception: None

Sound Module

Uses

AUDIO for Sound

Syntax

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
Sound		Sound	
play	Sound		
isPlay	Sound	Boolean	
pause	Sound		
unpause	Sound		
stop	Sound		
mute			
unmute			
toggle			

Semantics

State Variables

Sound: Audio object from file

State Invariant

None

Assumptions

- The constructor is called before other accesses
- The sound files are in the correct directory for the projectiles
- the sound files have the same name as expected.

Access Routine Semantics

Sound():

- transition: $muted := true$
- exception: None

play():

- input: $in := x \in$ Sound
- transition: $!in.muted : in.play()$
- exception: None

isPlay():

- input: $in := x \in$ Sound
- output: $out :=!in.paused()$
- exception: None

pause():

- input: $in := x \in$ Sound
- transition: $in.paused := true$
- exception: None

unpause():

- input: $in := x \in$ Sound
- transition: $in.paused :=!true$
- exception: None

stop():

- input: $in := x \in$ Sound
- transition: $in.paused := true \wedge this.currentTime := 0$
- exception: None

mute():

- input: $in := x \in \text{Sound}$
- transition: $in.muted := true$
- exception: None

unmute():

- input: $in := x \in \text{Sound}$
- transition: $in.muted :=!true$
- exception: None

toggle():

- input: $in := x \in \text{Sound}$
- transition: $in.muted :=!in.muted$
- exception: None

Head Module

Uses

utilities.js, sound.js, gameobject.js, gamestate.js

Exported Constants

None

Exported Access Programs

Routine name	In	Out	Exceptions
dynamicallyLoadScript	any		

Semantics

State Variables

None

State Invariant

None

Assumptions

- The files are named the same way that the module expects

Access Routine Semantics

dynamicallyLoadScript():

- input: $in := x \in \{"utilities.js", "sound.js", "gameobject.js", "gamestate.js"\}$
- transition: $c := \{\}$
- output: $out := Head$
- exception: None

GameObject Module

Template Module

GameObject

Uses

for all draw function in GameObject, the object is just being drawn to the html canvas reffered to as CVS, using a '2d' context reffered to as CTX.

Syntax

Exported Types

GameObject=? Player=? Bullet=? Alien=? AlienBullet=? Asteroid=?

Exported Constants

None

Exported Access Programs

Routine name	In	Out	Exceptions
GameObject		GameObject	
getX		$\mathbb{Z}$	
getY		$\mathbb{Z}$	
getHeading		$\mathbb{R}$	
getActivity		$\mathbb{B}$	
getRadius		$\mathbb{Z}$	
getVel		$\mathbb{R}$	
getCtx		CTX	
getName		String	
setX	$\mathbb{Z}$		
setY	$\mathbb{Z}$		
setActivity	$\mathbb{B}$		

Semantics

State Variables

name: String
x: $\mathbb{R}$
y: $\mathbb{R}$
rot: $\mathbb{R}$
a: $\mathbb{R}$
r: $\mathbb{N}$
visible: $\mathbb{B}$
vel: sequence of $\mathbb{R}$
acc: sequence of $\mathbb{R}$
ctx: CTX

State Invariant

None

Assumptions

GameObject(name):

- transition: $name, x, y, rot, a, visible, vel, acc, r, ctx = \text{name}, 0, 0, 0, 0, false, (0,0), (0,0), 0, \text{CTX}$
- output: $out := GameObject$
- exception: None

getX():

- output: $out := x$
- exception: None

getY():

- output: $out := y$
- exception: None

getHeading():

- output: $out := a$

- exception: None

getActivity():

- output: $out := visible$
- exception: None

getRadius():

- output: $out := r$
- exception: None

getVel():

- output: $out := vel$
- exception: None

getAcc():

- output: $out := acc$
- exception: None

getCtx():

- output: $out := ctx$
- exception: None

getName():

- output: $out := name$
- exception: None

setX(x):

- input: $in := x \in \mathbb{Z}$
- exception: None

setY(y):

- input: $in := x \in \mathbb{Z}$

- exception: None

setActivity(activity):

- input: $in := x \in \mathbb{B}$
- exception: None

Routine name	**In**	**Out**	**Exceptions**
Player		Player	
fire		Bullet	
thrust			
turn			
brake			
interact	KeyCode		
brake			
brake			
brake			

Semantics

State Variables

None

State Invariant

None

Assumptions

Player():

- transition: $fire, thrust, turn, airbrake, bulletCountDownvel, acc, r = false, false, false, \text{FPS}/2, (0,$
- output: $out := Player$
- exception: None

fire():

- input: $in := spascebar \in \text{Sound}$
- transition: $in.paused := true \wedge this.currentTime := 0$

- exception: None

thrust():

- input: $in := x \in$ Sound
- transition: $in.paused := true \wedge this.currentTime := 0$
- exception: None

turn():

- input: $in := x \in$ Sound
- transition: $in.paused := true \wedge this.currentTime := 0$
- exception: None

brake():

- input: $in := x \in$ Sound
- transition: $up, spa := true \wedge this.currentTime := 0$
- exception: None

interact():

- input: $in := up \vee space \vee left \vee right \vee down \in$ KeyCode
- transition: $up, space, left, right, down := thrust = true, fire = true, turn = left, turn = right, airbrake = true$
- exception: None

Routine name	**In**	**Out**	**Exceptions**
Bullet	Player	Bullet	

Semantics

State Variables

None

State Invariant

None

Assumptions

Bullet(p):

- transition: $timeOut, vel, x, y, r, velx, vely = 200, , getX(p){+}4/3{*}getR(p){*}cos(getHeading(p)), getY($ $4/3{*}getR(p){*}sin(getHeading(p)), 1, getVelX(p){+}\text{BULLET\_EXTRA}{*}cos(getHeading(p)), getVelY($ $\text{BULLET\_EXTRA} * -sin(getHeading(p))$
- output: $out := Bullet$
- exception: None

Routine name	**In**	**Out**	**Exceptions**
Alien		Alien	

Semantics

State Variables

None

State Invariant

None

Assumptions

Alien():

- transition: $timeSpawn, timeOut, xOrY, lOrR, acc, r = \text{ALIEN\_SPAWN}, 50, true, true, (0, 0), 12.5$
- output: $out := Alien$
- exception: None

Routine name	**In**	**Out**	**Exceptions**
AlienBullet		AlienBullet	

Semantics

State Variables

None

State Invariant

None

Assumptions

AlienBullet(a):

- transition: $timeOut, vel, x, y, r, velx, vely = 200, , getX(a), getY(a), 2, getVelX(a)+$ BULLET_EXTRA$*cos(getHeading(a)), getVelY(a)+$BULLET_EXTRA$*-sin(getHeading(a))$
- output: $out := AlienBullet$
- exception: None

Routine name	**In**	**Out**	**Exceptions**
Asteroid		Asteroid	

Semantics

State Variables

None

State Invariant

None

Assumptions

Asteroid():

- transition: $x, y, scale, r, children, vel, velx, vely = 0, 0, scale, 5*scale, [], , \pm random(0,1)* 3, \pm random(0,1)*3$
- output: $out := Asteroid$
- exception: None

Game State Module

Uses

utilities.js, gameobject.js, head.js

Exported Constants

STATE={START,PREGAME,LOAD,PLAYING,POSTGAME,PAUSE,RELOAD}
StateMachine=?

Exported Access Programs

Routine name	In	Out	Exceptions
StateMachine			
isSafe	OBJECT, seq. of OBJECT		
generateAsteroids	$x \in \mathbb{Z}$		
checkCollision	GameObject, GameObject, GameObject		
togglePause			

Semantics

State Variables

$state$: String $stateSave$: String $paused$: $\mathbb{B}$

State Invariant

$state \neq stateSave$

Assumptions

None

Access Routine Semantics

StateMachine():

- transition: $state$ = start

- output: $out := StateMachine$
- exception: None

isSafe(obj,sprites)

- input: $in := object, in := sprites$
- return: $d := \forall s \in sprites : (getName(s) = "asteroid" \wedge getActivity(s) = false \wedge \exists c \in getChildren(s) : \neg isSafe(c) : false) \vee (getName(s) = "asteroid" \wedge getActivity(s) = true : checkCollision(obj, s, 50) : false) \vee (getName(s) \in \{"alien, "alienBullet"\} \wedge getActivity(s) = true \wedge checkCollision(obj, s, 50) : false)$

checkCollision(a,b,c)

- input: $a \in \text{GameObject}, b \in \text{GameObject}, c \in \mathbb{Z}$
- output: $out := (pyth(|a.getX() - b.getX()|, |a.getX() - b.getX()|) < c)$

togglePause():

- transition: $pause \Rightarrow (stateSave = state \wedge state = \text{PAUSE}) \vee \neg pause \Rightarrow (state = stateSave)$
- exception: None

Local Functions

screenShow: $String \times \{\text{NORMAL}, \text{EMPHASIS} \Rightarrow ?\}$ output: $out :=$
drawShape: $String \times \mathbb{R} \times \mathbb{R} \Rightarrow$ drawTriangle: $\mathbb{N} \Rightarrow$ output: $out :=$
getName: OBJECT $\Rightarrow$ output: $out :=$
getActivity: OBJECT $\Rightarrow$ output: $out :=$
getChildren: OBJECT $\Rightarrow$ output: $out :=$