# 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 |

# Utilities Module

## Template Module

Utilities

## Uses

N/A

## 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 | Screen, Font | Text | |
| norm | $String, \mathbb{Z}, \mathbb{Z}$ | | |
| emph | $String, \mathbb{Z}, \mathbb{Z}$ | | |

## Semantics

### State Variables

$cvs$: Screen $fnt$: Font

### 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: $cvs[x][y] = screenShow(Str, \text{NORMAL})$
- exception: None

emph($Str, x, y$):

- transition: $cvs[x][y] = screenShow(Str, \text{EMPHASIS})$
- 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 | | |

## Semantics

### State Variables

*score*: $\mathbb{N}$ *lives*: $\mathbb{N}$ *sprites*: sequence of OBJECT *muteSound*: $\mathbb{N}$ *pauseGame*: $\mathbb{N}$

### State Invariant

None

### Assumptions

None

### Access Routine Semantics

Game():

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

reduceCounter():

- transition: $muteSound := muteSound - 1 \land pauseGame := pauseGame - 1$
- 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: $in := x \in$ Sound
- transition: $in.muted := true$
- exception: None

unmute():

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

toggle():

- input: $in := x \in$ 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

N/A

## Syntax

### Exported Types

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

### Exported Constants

None

### Exported Access Programs

| Routine name | In | Out | Exceptions |
|---|---|---|---|
| GameObject | | GameObject | |
| update | | | |
| activate | | | |
| deactivate | | | |
| die | | | |
| interact | | | |
| move | | | |
| action | | | |
| draw | | | |
| reset | | | |
| pass | | | |
| collide | | | |
| update | | | |
| collide | | | |
| collide | | | |

GameObject():

- transition: $name, x, y, rot, a, visible, vel, acc, r$ = name, 0, 0, 0, 0, false, (0,0), (0,0), 0

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

| Routine name | In | Out | Exceptions |
|---|---|---|---|
| Player | | Player | |
| fire | | Bullet | |
| thrust | | | |
| turn | | | |
| brake | | | |

Player():

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

| Routine name | In | Out | Exceptions |
|---|---|---|---|
| Bullet | Player | Bullet | |

Bullet(p):

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

| Routine name | In | Out | Exceptions |
|---|---|---|---|
| Alien | | Alien | |

Alien():

- transition: $timeSpawn, timeOut, xOrY, lOrR, acc, r =$ ALIEN_SPAWN$, 50, true, true, (0, 0), 12.5$
- output: $out := Alien$
- exception: None

| Routine name | In | Out | Exceptions |
|---|---|---|---|
| AlienBullet | | AlienBullet | |

AlienBullet(a):

- transition: $timeOut, vel, x, y, r, velx, vely = 200, , getX(a)+4/3*getR(a)*cos(getHeading(a)), getY($
  $4/3*getR(a)*sin(getHeading(a)), 1, getVelX(a)+$BULLET_EXTRA$*cos(getHeading(a)), getVelY($
  BULLET_EXTRA $* -sin(getHeading(a))$
- output: $out := AlienBullet$
- exception: None

| **Routine name** | **In** | **Out** | **Exceptions** |
|---|---|---|---|
| Asteroid | | Asteroid | |

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(obj,other,r)

- input: $obj \in$ GameObject, $other \in$ GameObject, $obj \in \mathbb{Z}$

togglePause():

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

## Local Functions

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