SE 3XA3: Module Interface Specification Tetrileet

Team 15, AAA Solutions
Student 1 Abdallah Taha, tahaa8
Student 2 Andrew Carvalino, carvalia
Student 3 Ali Tabar, sahraeia

March 18, 2021

This document is the Module Interface Specification (MIS) of AAA Solutions's Tetrileet.

Table 1: **Revision History**

Date	**Version**	**Notes**
March 17 2021	1.0	Added all modules from Game Controller to End Game
March 18 2021	1.1	Added all modules from BlockFall to Grid

Game Controller Module

Module

Game Controller Type

Uses

Grid

Syntax

Exported Constants

N/A

Exported Types

N/A

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
moveLeft	keyInput		Left_Edge_Grid
moveRight	keyInput		Right_Edge_Grid
moveDown	keyInput		Taken_Grid
rotate	keyInput		

Semantics

State Variables

None

Environment Variables

keyInput: {"key.W", "key.S", "key.A", "key.D"}

State Invariant

None

Assumptions

- Game Window is open and the start game button has been called.

Access Routine Semantics

moveLeft(key.A):

- transition: $grid.moveLeft()$
- output: None
- exception: Left_Edge_Grid

moveRight(key.D):

- transition: $grid.moveRight()$
- output: letter
- exception: Right_Edge_Grid

moveDown(key.S):

- transition: $grid.moveDown()$
- output: letter
- exception: Taken_Grid

rotate(key.W):

- transition: $grid.rotate()$
- output: letter
- exception: None

Local Constants

None

Game Window Module

Module

Game Window Type

Uses

None

Syntax

Exported Constants

N/A

Exported Types

Block

Exported Access Programs

Routine name	In	Out	Exceptions
draw			
eraseBlock			

Semantics

State Variables

Square

Environment Variables

None

State Invariant

$|Square| == 200$

Assumptions

- The HTML file is executed in a compatible browser.

Access Routine Semantics

draw():

- transition: $Square \rightarrow Block$
- output: None
- exception: None

eraseBlock():

- transition: $Block \rightarrow Square$
- output: None
- exception: None

Pause Game Module

Module

Pause Game Type

Uses

Game Window
BlockFall

Syntax

Exported Constants

N/A

Exported Types

$\mathbb{B}$

Exported Access Programs

Routine name	In	Out	Exceptions
Pause	keyInput		

Semantics

State Variables

Paused

Environment Variables

keyInput: {"key.P"}

State Invariant

$Paused = True \vee False$

Assumptions

N/A

Access Routine Semantics

Pause(key.P):

- transition: $Paused \equiv True \implies Paused := False \vee Paused \equiv False \implies Paused := True$
- output: None
- exception: None

Start Game Module

Module

Start Game Type

Uses

Game Window, BlockFall

Syntax

Exported Constants

N/A

Exported Types

$\mathbb{B}$

Exported Access Programs

Routine name	In	Out	Exceptions
Start	keyInput		

Semantics

State Variables

Started

Environment Variables

keyInput: {"mouse.leftClick"}

State Invariant

$Started = True \vee False$

Assumptions

N/A

Access Routine Semantics

Start(mouse.leftClick):

- transition: $Started \equiv False \implies Started := True$
- output: None
- exception: None

End Game Module

Module

End Game Type

Uses

Game Window, Grid

Syntax

Exported Constants

N/A

Exported Types

$\mathbb{B}$

Exported Access Programs

Routine name	In	Out	Exceptions
gameOver			

Semantics

State Variables

Ended

Environment Variables

keyInput: {”mouse.leftClick”}

State Invariant

$Ended = True \lor False$

Assumptions

Game has started and is in a running state.

Access Routine Semantics

gameOver():

- transition: $Ended \equiv False \rightarrow Ended := True$
- output: None
- exception: None

BlockFall Module

Module

BlockFall Type

Uses

Game Window, Grid, Shape

Syntax

Exported Constants

N/A

Exported Types

N/A

Exported Access Programs

Routine name	In	Out	Exceptions
falling			Taken_Grid

Semantics

State Variables

None

Environment Variables

None

State Invariant

N/A

Assumptions

Game has started and is in a running state, and is not paused.

Access Routine Semantics

falling():

- transition: Game will call $grid.moveDown()$ every 1,000 milliseconds
- output: none
- exception: Taken_Grid

BlockStack Module

Module

BlockStack Type

Uses

Shape, BlockFall

Syntax

Exported Constants

$\mathbb{B}$

Exported Types

N/A

Exported Access Programs

Routine name	In	Out	Exceptions
freezeBlock			

Semantics

State Variables

N/A

Environment Variables

N/A

State Invariant

N/A

Assumptions

Game has started and is in a running state, and is not paused.

Access Routine Semantics

freezeBlock():

- transition: $Shape.block$:= "Taken"
- output: None
- exception: None

Shape Module

Module

Shape Type

Uses

Grid

Exported Constants

N/A

Exported Types

N/A

Exported Access Programs

Routine name	In	Out	Exceptions
createShape			

Semantics

State Variables

None

Environment Variables

N/A

State Invariant

N/A

Assumptions

Game has started and is in a running state, and is not paused.

Access Routine Semantics

createShape():

- transition: Takes coordinates from a predetermined array, and selects the square with the tag 'block', and gives them a colour based on the corresponding shape
- output:
- exception: None

RowCheck Module

Uses

Grid

Syntax

Exported Constants

N/A

Exported Types

N/A

Exported Access Programs

Routine name	In	Out	Exceptions
rowCheck		$\mathbb{B}$	
rowClear			
addScore		$\mathbb{R}$	

Semantics

State Variables

clear, rowIndex, addToScore

Environment Variables

None

State Invariant

- $Clear \equiv True \vee False$
- $0 \leq rowIndex \leq 20$
- $addToScore \equiv 10$

Assumptions

N/A

Access Routine Semantics

rowCheck():

- transition: $\forall\, rowIndex \;\in\; row \mid row[rowIndex].block \;\equiv\; 'taken' \implies clear := true$
- output: clear
- exception: None

RowClear():

- transition: $clear := true \implies \forall\, rowIndex \in row \mid row[rowIndex].remove('block') \land row[rowIndex].remove('taken')$
- output: None
- exception: None

addScore():

- transition: $clear == true \implies addToScore+ = 10$
- output: addToScore
- exception: None

Grid

Uses

Game Window

Syntax

Exported Constants

Square

Exported Types

N/A

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
createGrid()			
displayShape()			
moveRight()			Right_Edge_Grid
moveLeft()			Left_Edge_Grid
moveDown()			Taken_Grid
rotate()			Right_Edge_Grid $\wedge$ Left_Edge_Grid $\wedge$ Taken_Grid

Semantics

State Variables

[Square], width, height

Environment Variables

N/A

State Invariant

$width = 10 \wedge height = 20$
$|[Square]| = 200$

Assumptions

N/A

Access Routine Semantics

createGrid:

- transition: Creates a grid with a height of 20 squares and a width of 10 squares. By taking the squares from the Game Window and assigning them a 'Block' tag.
- output: None
- exception: None

displayShape():

- transition: Takes the predetermined square coordinates from an array that have the tag 'block' and recolors aforementioned squares. It then gives them a 'taken' tag.
- output: None
- exception: None

moveRight():

- transition: Moves all squares with the tags 'block' and 'taken' from the current shape over by one column to the right. Then, recolours the newly taken squares and gives them a 'taken' tag. Proceeds to remove the colour and 'taken' tag from the squares that are no longer taken.
- output: None
- exception: None

moveLeft():

- transition: Moves all squares with the tags 'block' and 'taken' from the current shape over by one column to the left. Then, recolours the newly taken squares and gives them a 'taken' tag. Proceeds to remove the colour and 'taken' tag from the squares that are no longer taken.
- output: None
- exception: None

moveDown():

- transition: Moves all squares with the tags 'block' and 'taken' from the current shape over by one row down. Then, recolours the newly taken squares and gives them a 'taken' tag. Proceeds to remove the colour and 'taken' tag from the squares that are no longer taken.
- output: None
- exception: None

rotate():

- transition: Receives the change coordinates for the current shape from a predetermined array. Then, uses those coordinates to remove 'taken' tags and color from unused blocks after rotation. Finally, the new blocks are coloured and given a 'taken' tag.
- output: None
- exception: None