SE 3XA3: Module Interface Specification
Tetris Tussle

Nicholas Lobo	Matthew Paulin	David Carrie
lobon3	paulinm	carriedd
400179304	400187147	000661652

Contents

1 Module Hierarchy **4**

2 MIS of Input Module **4**
2.1 Interface Syntax . . . 4
2.1.1 Exported Access Programs . . . 4
2.2 Interface Semantics . . . 4
2.2.1 State Variables . . . 4
2.2.2 Environmental Variables . . . 4
2.2.3 Assumptions . . . 4
2.2.4 Access Program Semantics . . . 4

3 MIS of Main View Module **5**
3.1 Interface Syntax . . . 5
3.1.1 Exported Access Programs . . . 5
3.2 Interface Semantics . . . 5
3.2.1 State Variables . . . 5
3.2.2 Environmental Variables . . . 5
3.2.3 Assumptions . . . 5
3.2.4 Access Program Semantics . . . 5

4 MIS of Leaderboard Module **5**
4.1 Interface Syntax . . . 5
4.1.1 Exported Access Programs . . . 5
4.2 Interface Semantics . . . 5
4.2.1 State Variables . . . 5
4.2.2 Environmental Variables . . . 6
4.2.3 Assumptions . . . 6
4.2.4 Access Program Semantics . . . 6

5 MIS of Menu Module **6**
5.1 Interface Syntax . . . 6
5.1.1 Exported Access Programs . . . 6
5.2 Interface Semantics . . . 6
5.2.1 State Variables . . . 6
5.2.2 Environmental Variables . . . 6
5.2.3 Assumptions . . . 6
5.2.4 Access Program Semantics . . . 6

6 MIS of Singleplayer View Module **8**
6.1 Interface Syntax . . . 8
6.1.1 Exported Access Programs . . . 8
6.2 Interface Semantics . . . 8
6.2.1 State Variables . . . 8
6.2.2 Environmental Variables . . . 8
6.2.3 Assumptions . . . 8
6.2.4 Access Program Semantics . . . 8

7 MIS of Multiplayer View Module **10**
7.1 Interface Syntax . . . 10
7.1.1 Exported Access Programs . . . 10
7.2 Interface Semantics . . . 10
7.2.1 State Variables . . . 10
7.2.2 Environmental Variables . . . 10
7.2.3 Assumptions . . . 10
7.2.4 Access Program Semantics . . . 10

8 MIS of Tetromino Module **12**
8.1 Interface Syntax . . . 12
8.1.1 Exported Access Programs . . . 12
8.2 Interface Semantics . . . 12
8.2.1 State Variables . . . 12
8.2.2 Environmental Variables . . . 12
8.2.3 Assumptions . . . 12
8.2.4 Access Program Semantics . . . 12

9 MIS of Board Module **14**
9.1 Interface Syntax . . . 14
9.1.1 Exported Access Programs . . . 14
9.2 Interface Semantics . . . 14
9.2.1 State Variables . . . 14
9.2.2 Environmental Variables . . . 14
9.2.3 Assumptions . . . 14
9.2.4 Access Program Semantics . . . 14

10 MIS of Player Module **15**
10.1 Interface Syntax . . . 15
10.1.1 Exported Access Programs . . . 15
10.2 Interface Semantics . . . 15
10.2.1 State Variables . . . 15
10.2.2 Environmental Variables . . . 15
10.2.3 Assumptions . . . 15
10.2.4 Access Program Semantics . . . 15

11 MIS of Singleplayer Module **17**
11.1 Interface Syntax . . . 17
11.1.1 Exported Access Programs . . . 17
11.2 Interface Semantics . . . 17
11.2.1 State Variables . . . 17
11.2.2 Environmental Variables . . . 17
11.2.3 Assumptions . . . 17
11.2.4 Access Program Semantics . . . 17

12 Major Revision History **17**

1 Module Hierarchy

Level 1	Level 2
Hardware-Hiding Module	N/A
Behaviour-Hiding Module	User Inputs Module Main View Module Leaderboard Module Menu Module Singleplayer View Module Multiplayer View Module Singleplayer Module Multiplayer Module Player Module Tetromino Module Board Module
Software Decision Module	Server Module

Table 1: Module Hierarchy

2 MIS of Input Module

2.1 Interface Syntax

2.1.1 Exported Access Programs

Name	In	Out	Exceptions
onkeydown	KeyboardEvent Object	-	-

2.2 Interface Semantics

2.2.1 State Variables

keys: map (int $\rightarrow$ string) - A mapping of keyboard key codes to the corresponding game control input

2.2.2 Environmental Variables

N/A

2.2.3 Assumptions

N/A

2.2.4 Access Program Semantics

onkeydown(event):

Input: Object event, a JavaScript Keyboard Event Object representing a user's keyboard input

Transition: Translate the key code into a control input and pass the value to the Player module through a call of the keyPress function

Output: None

Exceptions: None

3 MIS of Main View Module

3.1 Interface Syntax

3.1.1 Exported Access Programs

Name	In	Out	Exceptions
mouseClicked	MouseEvent Object	GUI	-

3.2 Interface Semantics

3.2.1 State Variables

viewState: string - an indication of the current view to be displayed

3.2.2 Environmental Variables

N/A

3.2.3 Assumptions

N/A

3.2.4 Access Program Semantics

mouseClicked(event):

Input: Object event, a JavaScript Mouse Event Object representing a user's mouse input

Transition: Translate the mouse input into a control input and adjusts the viewState variable

Output: The visual elements to be displayed

Exceptions: None

4 MIS of Leaderboard Module

4.1 Interface Syntax

4.1.1 Exported Access Programs

Name	In	Out	Exceptions
updateLeaderboard	string, int	GUI	-

4.2 Interface Semantics

4.2.1 State Variables

highScores: List : int

4.2.2 Environmental Variables

N/A

4.2.3 Assumptions

N/A

4.2.4 Access Program Semantics

updateLeaderboard(name,score):
Input: name: string - the name of the player, score: int - the point total for a completed game
Transition: updates the menu view module and the highScores state variable
Output: A list of high scores shown on screen
Exception - None

5 MIS of Menu Module

5.1 Interface Syntax

5.1.1 Exported Access Programs

Name	In	Out	Exceptions
changeView	string	-	-

5.2 Interface Semantics

5.2.1 State Variables

Not Applicable

5.2.2 Environmental Variables

frameRate: integer representing frame rate.
captureArea: Rectangle object representing the area of the screen to be recorded
isRecording: boolean representing whether the system is recording or not recording

5.2.3 Assumptions

If getFrameRate() is called before setFrameRate(), then the default value will be returned.

If getDimensions() is called before setDimensions(), then the default value will be returned.

If getRecordingStatus() is called before setRecordingStatus(), then the default value of false will be returned.

createRectangle() will be called after setDimensions().

5.2.4 Access Program Semantics

setFrameRate(i):
Input: integer i
Transition: sets variable frameRate to i
Exception: Invalid Input if i is out of range by being zero or negative or over 240

setDimensions(p1,p2):

Input: two Point2D objects, p1 and p2

Transition: calls createRectangle and passes p1 and p2 to the access program. Sets variable captureArea to the returned Rectangle object

Exception: Out of Range if the x or y coordinates are negative or if it is the same point

setRecordingStatus(b):

Input: boolean indicating whether the the system currently has a request to record or to not record the screen

Transition: sets isRecording to the value of b

Exception: none

getFrameRate():

Input: none

Transition: accesses variable frameRate and retrieves value

Output: returns value of variable frameRate

Exception: none

getDimensions():

Input: none

Transition: accesses variable captureArea and retrieves value

Output: returns value of variable captureArea

Exception: none

getRecordingStatus():

Input: none

Transition: accesses variable isRecording and retrieves the value

Output: returns value of variable isRecording

Exception: none

createRectangle(p1,p2):

Input: two Point2D objects, p1 and p2

Transition: creates a Rectangle object from p1 and p2

Output: a Rectangle object with dimensions described by p1 and p2

Exception: IllegalArgumentException if the coordinates do not properly form a Rectangle (this exception comes from the Rectangle class)

6 MIS of Singleplayer View Module

6.1 Interface Syntax

6.1.1 Exported Access Programs

Name	In	Out	Exceptions
setup	-	GUI	-
draw	-	-	-
display	Player	GUI	-
windowResized	-	GUI	-
drawBlock	integer,integer,integer,integer,string,integer,	GUI	-
drawBorder	integer, List(integer)	GUI	-
drawGrid	integer, List(integer)	GUI	-
drawBoard	integer, List(integer)	GUI	-
drawPlaced	integer, List(integer), Board	GUI	-
drawCurrent	integer, List(integer), List(integer), integer, string, Tetromino, integer	GUI	-

6.2 Interface Semantics

6.2.1 State Variables

COLORS: List: String - list of different colour hex codes that represents the colours of the Tetromino

6.2.2 Environmental Variables

N/A

6.2.3 Assumptions

N/A

6.2.4 Access Program Semantics

setup():
    Input:
    Transition: displays p5.js canvas on screen
    Output: GUI
    Exception - none

draw():
    Input:
    Transition: needed for the processing library to function correctly
    Output:
    Exception - none

display(player):
    Input: player - player object
    Transition: uses other methods to display game state on screen
    Output:
    Exception - none

windowResized():

Input: player -
Transition: Updates size of canvas to match size of browser window
Output: GUI
Exception - none

drawBlock(x,y,outline,strokecolor,color,unit):
Input: x - x position of a block, y - y position of block, outline - stroke weight of the block outline, color - color of the block, unit- size of the block
Transition: fills in a grid square
Output: GUI
Exception - none

drawBorder(units, topleft):
Input: units - size of a grid square, topleft- x,y position of the board
Transition: displays border for the board
Output: GUI
Exception - none

drawGrid(units, topleft):
Input: units - size of a grid square, topleft- x,y position of the board
Transition: displays the grid lines for the board
Output: GUI
Exception - none

drawBoard(units, topleft):
Input: units - size of a grid square, topleft- x,y position of the board
Transition: calls the drawGrid and drawBoard function
Output: GUI
Exception - none

drawPlaced(units, topleft, board):
Input: units - size of a grid square, topleft- x,y position of the board, board - board object
Transition: draws the Tetrominos placed on the board
Output: GUI
Exception - none

drawCurrent(unit, position, topleft, shadow. color, shape):
Input: units - size of a grid square, position - position of the Tetromino, topleft- x,y position of the board, shadow - The vertical position of the shadow piece, color - the color of the Tetromino, shape - an integer giving the binary representation of the Tetromino
Transition: draws the Tetrominos placed on the board
Output: GUI
Exception - none

7 MIS of Multiplayer View Module

7.1 Interface Syntax

7.1.1 Exported Access Programs

Name	In	Out	Exceptions
setup	-	GUI	-
draw	-	-	-
display	Player, Player	GUI	-
windowResized	-	GUI	-
drawBlock	integer,integer,integer,integer,string,integer,	GUI	-
drawBorder	integer, List(integer)	GUI	-
drawGrid	integer, List(integer)	GUI	-
drawBoard	integer, List(integer)	GUI	-
drawPlaced	integer, List(integer), Board	GUI	-
drawCurrent	integer, List(integer), List(integer),integer, string, Tetromino, integer	GUI	-

7.2 Interface Semantics

7.2.1 State Variables

COLORS: List: String - list of different colour hex codes that represents the colours of the Tetromino

7.2.2 Environmental Variables

N/A

7.2.3 Assumptions

N/A

7.2.4 Access Program Semantics

setup():
Input:
Transition: displays p5.js canvas on screen
Output: GUI
Exception - none

draw():
Input:
Transition: needed for the processing library to function correctly
Output:
Exception - none

display(player,player2):
Input: player - player object
Transition: uses other methods to display game state on screen for two players
Output:
Exception - none

windowResized():

Input: player -
Transition: Updates size of canvas to match size of browser window
Output: GUI
Exception - none

drawBlock(x,y,outline,strokecolor,color,unit):
Input: x - x position of a block, y - y position of block, outline - stroke weight of the block outline, color - color of the block, unit- size of the block
Transition: fills in a grid square
Output: GUI
Exception - none

drawBorder(units, topleft):
Input: units - size of a grid square, topleft- x,y position of the board
Transition: displays border for the board
Output: GUI
Exception - none

drawGrid(units, topleft):
Input: units - size of a grid square, topleft- x,y position of the board
Transition: displays the grid lines for the board
Output: GUI
Exception - none

drawBoard(units, topleft):
Input: units - size of a grid square, topleft- x,y position of the board
Transition: calls the drawGrid and drawBoard function
Output: GUI
Exception - none

drawPlaced(units, topleft, board):
Input: units - size of a grid square, topleft- x,y position of the board, board - board object
Transition: draws the Tetrominos placed on the board
Output: GUI
Exception - none

drawCurrent(unit, position, topleft, shadow. color, shape):
Input: units - size of a grid square, position - position of the Tetromino, topleft- x,y position of the board, shadow - The vertical position of the shadow piece, color - the color of the Tetromino, shape - an integer giving the binary representation of the Tetromino
Transition: draws the Tetrominos placed on the board
Output: GUI
Exception - none

8 MIS of Tetromino Module

8.1 Interface Syntax

8.1.1 Exported Access Programs

Name	In	Out	Exceptions
Tetromino	-	Tetromino Object	-
getPosition	-	List(integer)	-
getShape	-	integer	-
getState	-	List(integer)	-
getRotation	-	integer	-
rotate	-	-	-
getShadow	-	integer	-
setShadow	integer	-	-
gameTick	-	-	-
move	string	-	-
drop	integer	-	-

8.2 Interface Semantics

8.2.1 State Variables

shape: integer - represents corresponding index in SHAPES
state: List(integer) - lists of binary encoded all rotations of each shape
rotation: integer - specifies the rotation of shape and corresponds to index in shape array
shadow: integer - represents the vertical position of the shadow
position: List(integer) - x and y position of the Tetromino

8.2.2 Environmental Variables

SHAPES: List(List(integer)) - stores the binary representations of each Tetromino for each rotation position

8.2.3 Assumptions

Variables should be set before trying to access them

8.2.4 Access Program Semantics

Tetromino():
- Input:
- Transition: creates a random Tetromino object in its default position
- Output: Tetromino Object
- Exception: none

getPosition():
- Input: none
- Transition: none
- Output: a list of integers containing the x and y position of the Tetromino
- Exception: none

getShape():
- Input: none
- Transition: none

Output: An integer representing the Tetromino type's corresponding location in the SHAPES list
Exception: none

getState():
Input: none
Transition: none
Output: A list of integers containing the binary encoded versions of the Tetromino in all its possible rotations
Exception: none

getRotation():
Input: none
Transition: none
Output: An integer representing the current rotation of the Tetromino
Exception: none

rotate():
Input: none
Transition: rotates the Tetromino 90 degrees
Output: none
Exception: none

getShadow():
Input: none
Transition: none
Output: An integer representing the vertical position of the Tetromino's shadow
Exception: none

setShadow(shadow):
Input: shadow - An integer representing the vertical position of the Tetromino's shadow
Transition: update the Tetromino's shadow
Output: none
Exception: none

gameTick():
Input: none
Transition: move the Tetromino downwards
Output: none
Exception: none

move(direction):
Input: direction - A string representing the direction of movement
Transition: update the Tetromino's position
Output: none
Exception: none

drop(y):
Input: y - an integer representing the vertical amount to move
Transition: update the Tetromino's position
Output: none
Exception: none

9 MIS of Board Module

9.1 Interface Syntax

9.1.1 Exported Access Programs

Name	In	Out	Exceptions
Board	-	Board Object	-
getElems	-	List(List(Cell))	-
addToBoard	Tetromino	-	-
checkCollisions	Tetromino, List(integer), integer	boolean	-
hardDrop	Tetromino	integer	-
clearLine	-	-	-

9.2 Interface Semantics

9.2.1 State Variables

elems: List(List(cells)) - represents one grid unit of the board, if its filled and the current color of the grid unit.

9.2.2 Environmental Variables

ROWS: integer - number of rows on the board COLS: integer - number of columns on the board

9.2.3 Assumptions

None

9.2.4 Access Program Semantics

Board():
Input:
Transition: creates an empty board object
Output: Board Object
Exception: none

getElems():
Input: none
Transition: none
Output: Elems
Exception: none

addToBoard(Tetromino):
Input: Tetromino object
Transition: fills in cells containing the tetromino on the board
Output: none
Exception: none

checkCollision(Tetromino, direction, rotation):
Input: Tetromino Object, direction - List(integer) representing horizontal and vertical direction of the Tetromino, rotation - integer representing the way the Tetromino will rotate
Transition: Checks collisions between Tetromino and the board
Output: Boolean representing if a collision occurred

Exception: none

hardDrop(Tetromino):
Input: Tetromino - a Tetromino object
Transition: Lowest position the Tetromino can move to without colliding with the board
Output: An integer representing the vertical position of the lowest point on the board
Exception: none

clearLine():
Input: none
Transition: Delete filled in rows on the board and shifts all filled pieces on the board down one unit.
Output: none
Exception: none

10 MIS of Player Module

10.1 Interface Syntax

10.1.1 Exported Access Programs

Name	In	Out	Exceptions
Player	integer, Board, Tetromino, boolean	Player Object	-
getId	-	integer	-
getBoard	-	Board	-
getTetromino	-	Tetromino	-
isPaused	-	boolean	-
setBoard	Board	-	-
setTetromino	Tetromino	-	-
togglePaused	-	-	-

10.2 Interface Semantics

10.2.1 State Variables

id: integer - represents a unique player identifier
board: Board - current board with respect to player id
tetromino: Tetromino - current Tetromino with respect to player id
paused: boolean - checks whether the game is paused

10.2.2 Environmental Variables

10.2.3 Assumptions

None

10.2.4 Access Program Semantics

Player(id, board, tetromino, paused): Input: id - integer that represents a unique player identifier, board - Board - current board with respect to player id, tetromino - current Tetromino with respect to player id, paused - boolean that checks whether the game is paused
Transition: creates a Player object
Output: Player Object

Exception: none

getId():
Input: none
Transition: gets the unique id of the player object
Output: id
Exception: none

getBoard():
Input: none
Transition: gets the unique Board object of the player object
Output: Board object
Exception: none

getTetromino():
Input: none
Transition: gets the unique Tetromino object of the player object
Output: Tetromino object
Exception: none

isPaused():
Input: none
Transition: gets the value of the isPaused state variable for the player object
Output: boolean
Exception: none

setTetromino():
Input: Tetromino
Transition: gets the unique Tetromino object of the player object
Output: Tetromino object
Exception: none

setBoard():
Input: Board
Transition: sets the attributes of the board for the player object
Output: none
Exception: none

togglePause():
Input: none
Transition: changes the value of the isPaused state variable
Output: none
Exception: none

11 MIS of Singleplayer Module

11.1 Interface Syntax

11.1.1 Exported Access Programs

Name	In	Out	Exceptions
tick	-	GUI	-
getState	-	Board	-
display	Player Object	GUI	-
newGame	-	GUI	-
keyPress	integer	GUI	-

11.2 Interface Semantics

11.2.1 State Variables

11.2.2 Environmental Variables

11.2.3 Assumptions

None

11.2.4 Access Program Semantics

tick(): Input: none
Transition: Advances the game at a set interval and updates the Player accordingly
Output: updates GUI
Exception: none

12 Major Revision History

November 3, 2015 - Rough draft of sections
November 5, 2015 - Revised sections
November 6, 2015 - Revision 0 complete