2014-1 Programming Language

Design Project
- Object-oriented Programming for Tetris Game

June 1, 2014

Prof. Young-Tak Kim

Advanced Networking Technology Lab. (YU-ANTL)

Dept. of Information & Comm. Eng, College of Engineering,

Yeungnam University, KOREA

(Tel : +82-53-810-2497; Fax : +82-53-810-4742

http://antl.yu.ac.kr/; E-mail : ytkim@yu.ac.kr)

Outline

- Tetris, Tetromino
- Console output
  - special characters :
  - positioned output
- Class Hierarchy
- Class Shape
- Struct Tile
- Class Board
- Key Board Input
- main loop of Tetris grame

Tetris

- ***Tetris*** **(Russian: Тетрис)**
  - a tile-matching puzzle video game originally designed and programmed by Alexey Pajitnov in the Soviet Union
  - was released on June 6, 1984,[2] while he was working for the Dorodnicyn Computing Centre of the Academy of Science of the USSR in Moscow.
  - He derived its name from the Greek numerical prefix *tetra-* (all of the game's pieces contain four segments) and tennis, Pajitnov's favorite sport.[4][5]
  - the first entertainment software to be exported from the USSR to the U.S. and published by Spectrum Holobyte for Commodore 64 and IBM PC.
  - the *Tetris* game is a popular use of Tetrominoes, the four element special case of polynominoes. Polynominoes have been used in popular puzzles.

Tetromino

◆ Tetromino (1)

- game pieces shaped like Tetrominoes, geometric shapes composed of four square blocks each.
- A random sequence of Tetrominos fall down the playing field (a rectangular vertical shaft, called the "well" or "matrix").
- The objective of the game is to manipulate these Tetrominos, by moving each one sideways and rotating it by 90 degree units, with the aim of creating a horizontal line of ten blocks without gaps.
- When such a line is created, it disappears, and any block above the deleted line will fall.
- When a certain number of lines are cleared, the game enters a new level. As the game progresses, each level causes the Tetrominos to fall faster, and the game ends when the stack of Tetrominos reaches the top of the playing field and no new Tetrominos are able to enter. Some games also end after a finite number of levels or lines.

◆ Tetrominos (2)

- 7 one-sided Tetrominos: I, J, L, O, S, T, Z

piece	Shape	Color	Rotations
I		Cyan	
J		Blue	
L		Orange	
O		Yellow	
S		Green	
T		Purple	
Z		red	

◆ Tetrominos(3)

- All of the Tetrominos are capable of single and double clears. *I*, *J*, and *L* are able to clear triples.
- Only the *I* Tetromino has the capacity to clear four lines simultaneously, and this is referred to as a "tetris". (This may vary depending on the rotation and compensation rules of each specific *Tetris* implementation. For instance, in the *Super Rotation System* used in most recent implementations, certain situations allow *T*, *S*, and *Z* to 'snap' into tight spots and clear triples.)

Special Char on Console

◆ printing special char on console

- puts("■□●○"); // 한글 자음 'ㅁ' + '한자' key
- puts(" [ ] "); // 한글 자음 'ㄴ' + '한자' key
- puts("∞ ≠ ▽"); // 한글 자음 'ㄷ' + '한자' key
- puts("㉠ ㉡"); // 한글 자음 'ㅅ' + '한자' key
- puts("ⓐ ⓑ ⓒ"); // 한글 자음 'ㅇ' + '한자' key

Colored Text Output on Console

◆ Color code

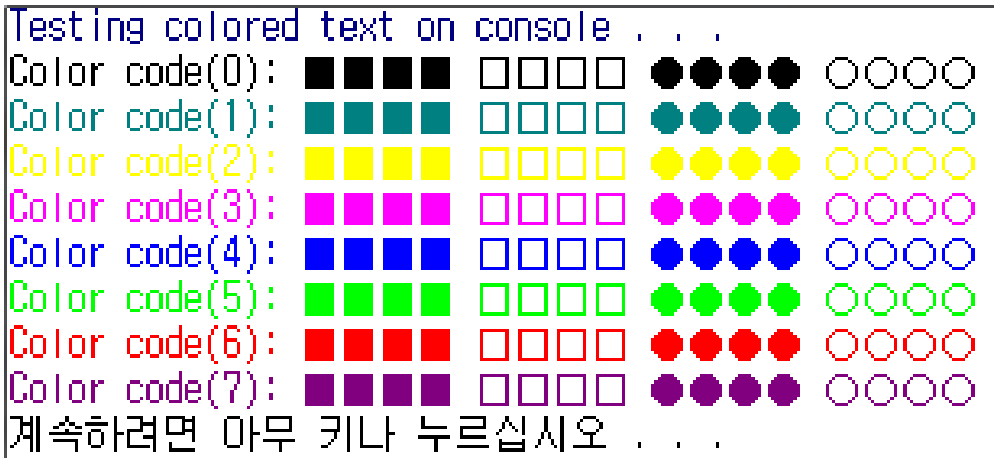
```
0 ................black
1 ................blue
2 ................green
3 ................cyan
4 ................red
5 ................purple
6 ................brown/yellow
7 ................white
```

◆ Background color and Foreground text Color

- 0x0F : Black background and White char
- 0xF0 : White background and Black char

```
#include <windows.h>
#include <iostream>
using namespace std;

#define WHITE_BACKGROUND 0xF0
int main(int argc, char* argv[])
{
    unsigned char colorTable[] = {0x00, 0x03, 0x0E, 0x0D, 0x09, 0x0A, 0x0C, 0x05};

    cout << "Testing colored text on console . . ." << endl;
    for (int i=0; i< 8; i++)
    {
        SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),
            colorTable[i] | WHITE_BACKGROUND);
        cout << "Color code(" << i << "): ";
        puts("■■■■□□□□●●●●○○○○"); // 한글자음'ㅁ' + '한자' 키

    }
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), WHITE_BACKGROUND);
      // white background, black foreground

    return 0;
}
```

◆ Result of colored text output



Positioned output on Windows Console

◆ COORD

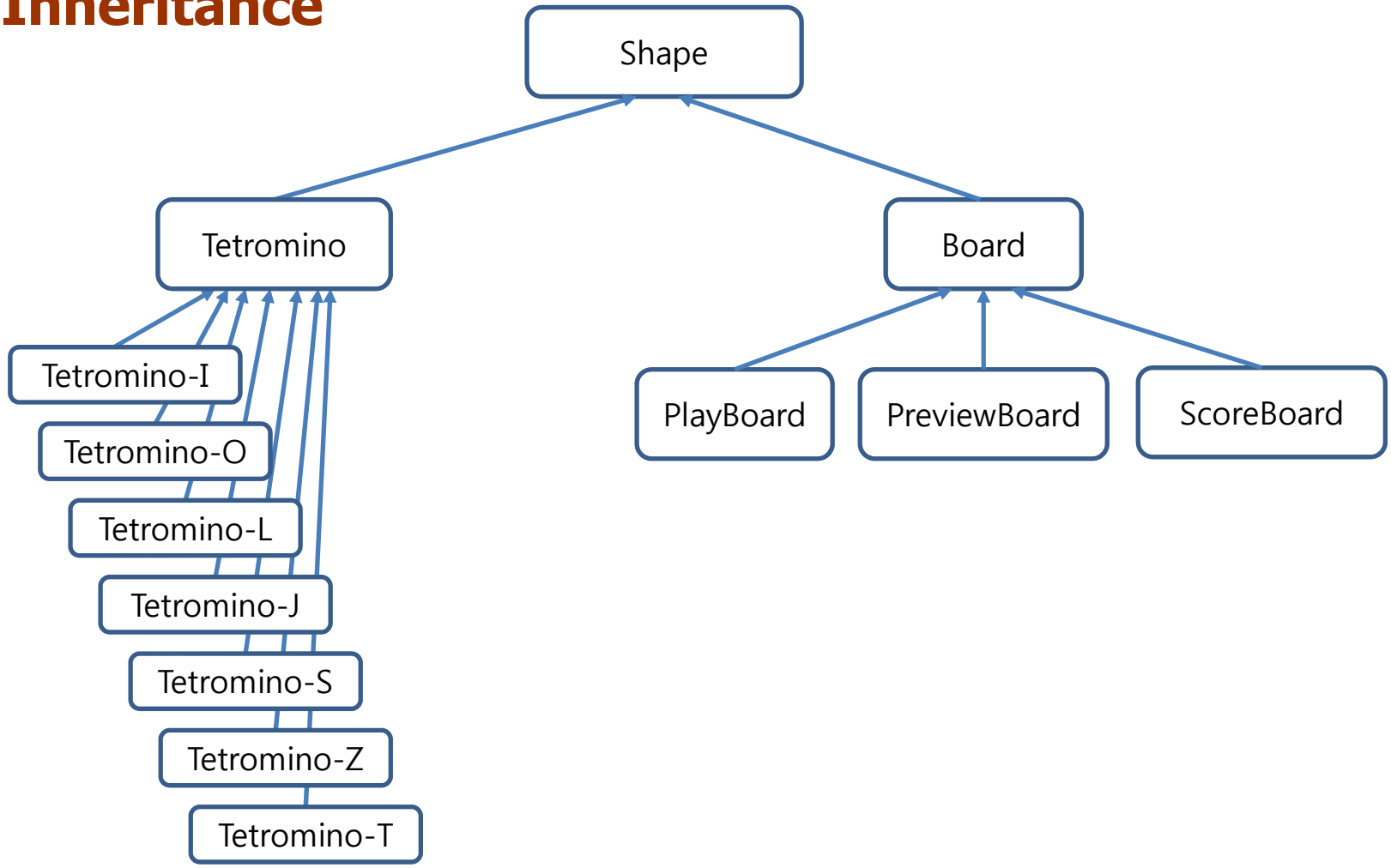
```
typedef struct _COORD
{
    SHORT X;
    SHORT Y;
} COORD;
```

◆ gotoXY()

```
. . . .
#include <stdlib.h>
#include <conio.h>
#include <windows.h>

void gotoXY(int x, int y)
{
    COORD xy = {x, y};
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), xy);
}
```

Class Hierarchy for Tetris game

◆ Inheritance



Class Shape

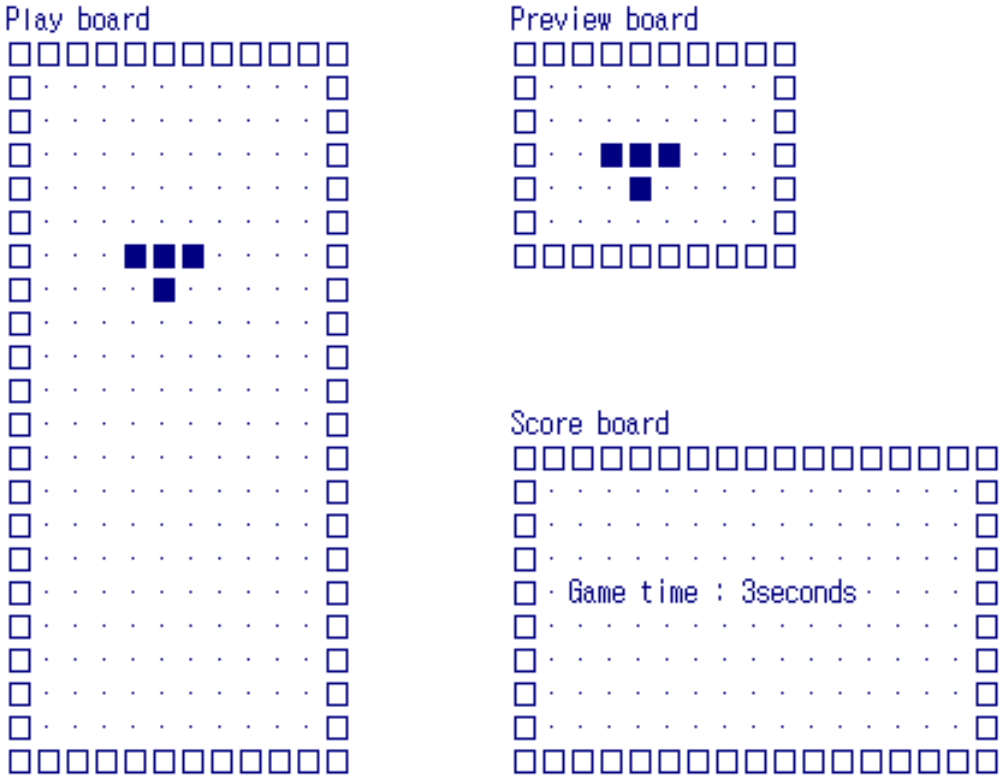
- **Class Shape contains following protected data members:**
  - int pos_x; // position coordinate
  - int pos_y; // position coordinate
- **Constructors**
  - The default constructor of shape, Shape::Shape(), initializes the data members to pos_x = 0, pos_y = 0.
  - Constructor, Shape::Shape(int x, int y), receives two arguments and initializes the attributes.
- **Member functions of Class Shape**
  - void gotoXY(int x, int y);
  - int getPos_x();
  - int getPos_y();
  - void setPos_x();
  - void setPos_y();.

◆ Class Shape

```
/**  Shape.h   */
#ifndef SHAPE_H
#define SHAPE_H

using namespace std;

class Shape
{
public:
    Shape(int x, int y); // constructor
    ~Shape(); // destructor
    void gotoPos();
    void gotoXY(int x, int y);
    int getPos_x();
    int getPos_y();
    void setPos_x(int x);
    void setPos_y(int y);
private:
    int pos_x; // current reference position
    int pos_y;
};

#endif
```

Boards

◆ Boards: Play, Preview, Score



Class Board

- **Class Board inherits from class Shape, contains following private data members:**
  - int width;
  - int height;
  - Tile **ppTile // points a dynamically created two dimensional
    // array of width x height tiles.
- **Class Board contains member functions of**
  - Board(); // constructor
  - ~Board(); // destructor
  - void initBoard();
  - void drawBoard();
  - Tetromino *addTetr(int tetr_type);
  - void removeTetr(Tetromino *pTetr);
  - Tile **getppTile() {return ppTile};

Tile

◆ Struct Tile

- enum Tile {EMPTY, BRICK, WALL};

◆ Tile Color

- the color of the brick, wall, or Tetromino
- unsigned char colorTable[] = {0x00, 0x03, 0x0E, 0x0D, 0x09, 0x0A, 0x0C, 0x05};

```
/**  Board.h   (1)  */

#ifndef BOARD_H
#define BOARD_H

#include "Shape.h"
#include "Obj_Tetris.h"
#include "Tetromino.h"

using namespace std;


class Board :public Shape
{
public:
    Board(int pos_x, int pos_y, int w, int h); // constructor
    void initBoard();
    void drawBoard();
    bool moveDown(Tetromino *pTetr);
    bool moveRight(Tetromino *pTetr);
    bool moveLeft(Tetromino *pTetr);
    bool rotate(Tetromino *pTetr);
    Tetromino *addTetr(int tet_type);
    void removeTetr(Tetromino *pTetr);
    Tile **getppTile()    {return ppTile;}
```

```
/**  Board.h   (1)  */

    void reset();
    void setWidth(int w);
    void setHeight(int h);
    void gotoRefPos(int x, int y);  // goto reference position
    void clearBricksFromBoard(Tetromino *pTetr);
    void setBricksInBoard(Tetromino *pTetr);
    bool checkSpacesInBoard(Tetromino *pTetr);
    int removeFilledLines(); // remove filled lines,
                             // returns the number of processed lines

private:
    int width;
    int height;
    Tile **ppTile;
};

#endif
```