

# CS101 Default Final Project

**Objective:** Tetris is the default final project for CS101 Spring 2009. You do not have to choose the default project if you have something else more interesting to do. If you do not have other better choices, the default project is a good option. The final project is open-ended. You are free to change anything and make any extensions. So, try hard to push to the limit!

## 1 Tetris

Tetris is a popular video game that works on almost every game hardware platform. Now it is time to make the game yourself. You will find that making a game is as fun as playing a game.

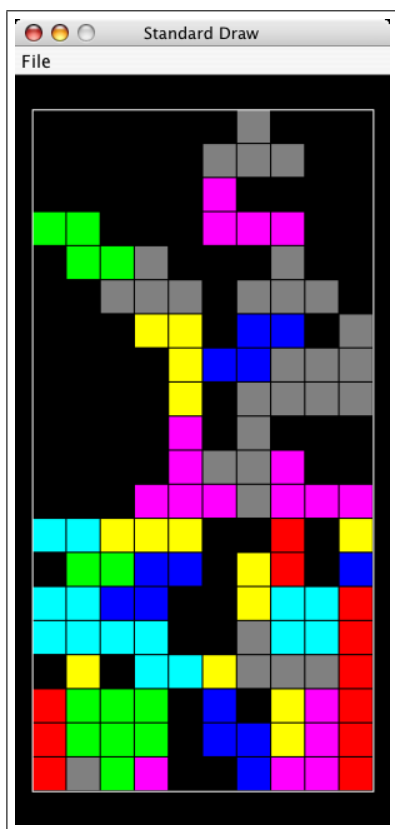


Figure 1: Tetris in Java.

different states about the game, creating and talking to the tetrads. The tetrads move around on the game panel and change its status. It is clear now that we need to make two classes: one for the game panel and the other one for tetrads.

Let's look at the game panel class `Tetris` first. The basic data member for the game panel is an array. Each element of the array corresponds to a square on the game panel. Tetrads can

The basic rule of Tetris is to move a sequence of falling tiles that are called tetrominoes or tetrads to make horizontal connected tiles. The movement of tetrads includes translation and rotation by 90 degrees. When a tetrad falls to the bottom, a new tetrad appears from the top. When tetrads pile up to the top, the game is over.

You will find that it is much simpler than you thought to make an real game that works as well as a commercial product. Moreover, you can have all the control about how the game should work. Fig. 1 shows a finished Tetris game using Java.

Creating a game such as Tetris is more than just good graphics. You need to think about how to process the data in the game. For instance, how to represent the game panel, how to represent the tetrads, how to keep their states, how to make them work with each other and interact with the player are some basic problems that you have to consider when you start to tackle the problem.

A good strategy of solving a problem like this is to look at it from a top-down view: instead of working on the details at the beginning, we build a framework first. It looks like there are two main objects in the game: the game panel and the tetrad. The game panel interacts with the player directly. It is responsible for starting the game, determining whether the game is over, keeping

traverse a sequence of squares on the game panel when they fall. The array can thus be used to keep the locations of the falling tetrads and those piling up at the bottom. We choose the following scheme to label the status of each square on the game panel: Empty squares on the game panel correspond to 0 value elements in the array. If a game panel square is on a tetrad and the tetrad is not at the bottom, the corresponding cell in the array is marked with the value  $7 + \text{tetradId}$ . Here `tetradId` is an integer from 1 to 7. When a tetrad hits the bottom, the corresponding cells in the array are labeled with `tetradId`. The reason that we use such a labeling scheme is that it is easy to distinguish the permanent labels at the bottom of the game panel and the temporary ones showing how tetrads fall. The label of each square in the game panel enables us to draw the squares with colors determined by the tetrad ids which are either the labels or the labels minus 7.

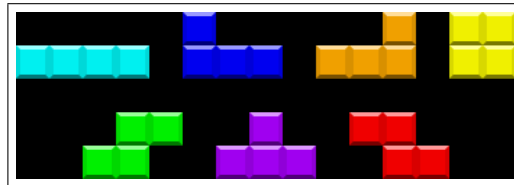


Figure 2: Tetrads.

The Tetris class also has a `draw` object with class `Draw`. The source code of `Draw.java` and the following `DrawListener.java` can be downloaded from the book web site [www.cs.princeton.edu/introcs](http://www.cs.princeton.edu/introcs) at the chapter of object oriented programming. The `draw` object is responsible for drawing the game panel. After defining the data structure for the game panel, drawing becomes the simple task of painting each square of the panel with the appropriate color corresponding to each element in the array.

To enable game panel to interact with users using the keyboard, we let class `Tetris` be a derived class from `DrawListener`. By overriding the functions of `KeyTyped`, we can let our program respond to the user key strokes in real time. We use key “i” to rotate the tetrad, “j” to move the tetrad to the left, “l” to the right and “k” down.

The game panel should also include methods to clear horizontal connected cells, determine scores and the level of the game and whether the game is over. These functions are not included in the skeleton code below. You should write them yourselves.

Now, it is time to think about the class of tetrad: `Tetromino`. In this class, we need to represent objects that have different shapes. One natural choice of the data structure is a mask array. The mask array labels squares on a tetrad as 1 and 0 otherwise. There are 7 different tetrads as shown in Fig. 2. Each tetrad is composed of 4 squares. We also would like to keep the location of a tetrad on the game panel, its size and id. Moving a tetrad therefore becomes translating or rotating a mask to fit into empty cells of the game panel array. You should make a tetrad to be a fully autonomous object. It moves around based on the messages from the game panel: some messages come from user keyboards input, others are from a regular timer in the game panel that triggers the time-to-fall message. A tetrad marks the game panel array to leave a trace about where it is. It also informs the game panel when it hits the bottom.

The game panel class and the tetrad class are closely coupled. The game panel class has an object of `Tetromino`. The tetrad object also shares a reference to the game panel array so that it can mark on it.

The skeleton code for the program can be downloaded from the course website. It contains a complete `Tetromino` class and a partially complete `Tetris` class. When running the skeleton code, you can move a randomly generated tetrad in the game panel until it hits the bottom. Based on

the skeleton code, you should be able to make a fully functioned Tetris game.

## **2 Possible Extensions**

Since this is the final course project, you are free to add any thing that is interesting. For example, you can make the graphics better, include the scoring schemes, make multiple levels, make the game to support two players and make a 3D Tetris game, etc.

## **3 What to Submit**

You should submit your Java programs. Pay attention to good Java programming style. Upload your Java files to webCT before the submission deadline. We will select the top projects to be presented in our last class.