Engineering Summer Workshop 2019

**Topic:** Make an On Screen Piano to Play Your Favorite Music

28 June 2019 (Friday), Rm 4210

Dr. Desmond Tsoi

Department of Computer Science & Engineering
The Hong Kong University of Science and Technology
Hong Kong SAR, China

Rm 3553, desmond@cse.ust.hk

---

**Instructor**

Dr. Desmond Yau-chat TSOI (Simply call me "Desmond" :) )

- Personal website: http://www.cse.ust.hk/~desmond
- E-mail: desmond@cse.ust.hk
- Office: Rm 3553 (Lift 27-28)

---

**Student Helpers**

- CHAN, Pak Ching
- CHENG, Man Hin
- CHUNG, Yuen Ting
- SUEN, Heung Ping
- XIA, Zihan
- YIP, Chun On
- ZHANG, Daofu
- ZHANG, Ziyang

---

**Software**

- In this workshop, you are going to use **Greenfoot** to construct an on-screen piano
  - Greenfoot is an interactive **Java** development environment for development of two-dimensional graphical applications, e.g., simulations and interactive games
  - Link to official site: https://www.greenfoot.org/
Goal: Make an On-Screen Piano to Play Music

How? Two Files: Piano.java and Key.java

A piano has a collection of keys (white and black keys)

Things to Do
1. Visit the Workshop Website
2. Download the Skeleton Code (middle icon)
3. Start Greenfoot (Please follow the verbal instructions)
4. Load up the code page:
   http://www.cse.ust.hk/~desmond/piano-workshop/code/

Note
Please keep your browser open as you need to refer to the code from time to time.
Five Parts

I. Making a white key
II. Making two white keys
III. Making all white keys (12 in total)
IV. Making all black keys (8 in total)
V. Making a music player

What is given?

- Piano Class (Right-click Piano icon and select “Open editor”)

```java
// (World, Actor, GreenfootImage, and Greenfoot)
import greenfoot.*;

class Piano extends World {
    public Piano() {
        super(800, 340, 1);
    }
}
```
What is given?

- Key Class (Right-click Key icon and select “Open editor”)

```java
// (World, Actor, GreenfootImage, and Greenfoot)
import greenfoot.*;

public class Key extends Actor {
  /*
   * Create a new key.
   */
  public Key() {
  }

  /*
   * Do the action for this key.
   */
  public void act() {
  }
}
```

Run it

- Press “Run”
- Right-click the “Key” icon and select “new Key()”

Place the key on the piano

- Image to Show the Key Down

- Update the act() method of Key class with the following

```java
// This method is called whenever // the “Act” or “Run” button gets // pressed in the environment

public void act() {
  // Check if key "g" is pressed
  if (Greenfoot.isKeyDown("g")) {
    // change to gray image
    setImage("white-key-down.png");
  } else {
    // change to original image, i.e. white
    setImage("white-key.png");
  }
}
```

Problem

No response when we press keys! :(  

```java
Rm 3553, desmond@cse.ust.hk
Engineering Workshop 2019
13 / 51

Rm 3553, desmond@cse.ust.hk
Engineering Workshop 2019
14 / 51

Rm 3553, desmond@cse.ust.hk
Engineering Workshop 2019
15 / 51

Rm 3553, desmond@cse.ust.hk
Engineering Workshop 2019
16 / 51
```
Oops...

Problem
Key always Down for First Press! :(
(Some versions of Greenfoot may not have this problem)

Change Once Only: boolean isDown

- Update the act() method of Key class again with the following code:

```java
public void act() {
    // if not isDown and "g" is down
    if (!isDown && Greenfoot.isKeyDown("g")) {
        setImage("white-key-down.png");
        isDown = true;
    }

    // if isDown and "g" is not down
    if (isDown && !Greenfoot.isKeyDown("g")) {
        setImage("white-key.png");
        isDown = false;
    }
}
```

Run it again and press "g". It should work! :)

Problem
No sound! :(

Produce the Sound
- The sounds folder has a collection of sound files, each of which contains the sounds for a single piano key

Play the Note
- Add play() method to the Key class as follows:

```java
import greenfoot.*;

public class Key extends Actor {
    // ...

    * Play the note of this key.
    */
    // Add the following code to the "Key" class
    public void play() {
        Greenfoot.playSound("3a.wav");
    }
}
```
Play the Note if "g" is down

- Put `play();` after the line `setImage("white-key-down.png");`

```java
public void act() {
    // if( not isDown and "g" is down )
    if ( !isDown && Greenfoot.isKeyDown("g") ) {
        setImage("white-key-down.png");
        // ----------------- // Add play() here // -----------------
        play();
        isDown = true;
    }
    // if( isDown and "g" is not down )
    if ( isDown && !Greenfoot.isKeyDown("g") ) {
        setImage("white-key.png");
        isDown = false;
    }
}
```

Run it again and press "g". It works! Perfect! :)

Add More Keys

- Now, add two keys and see what happen

```java
public class Key extends Actor {
    private boolean isDown;
    // Add two more variables
    private String key;
    private String sound;
    // Update the Key() method
    public Key(String keyName, String soundFile) {
        key = keyName;
        sound = soundFile;
    }
    public void act() {
        if (isDown && Greenfoot.isKeyDown(key)) {
            setImage("white-key-down.png");
            play();
            isDown = true;
        }
        if (isDown && !Greenfoot.isKeyDown(key)) {
            setImage("white-key.png");
            isDown = false;
        }
    }
    public void play() {
        Greenfoot.playSound(sound);
    }
}
```

Problem

All keys react the same way >.<

Part II

Making Two White Keys

- Add two more variables and update Key(...) method
Try: Add First Key
- Right-click "Key" and select "new Key"
  - Enter “g” and “3a.wav”
  3a.wav is “Do” sound

Try: Add Second Key
- Right-click “Key” and select “new Key”
  - Enter “h” and “3b.wav”
  3b.wav is “Rei” sound

Run it again. Press "g" and "h".

Add a Key at Specified Position When the Program is Run
- Use addObject method provided by Greenfoot
- The following statement add a Key at (300, 180) and link it with key “g” and sound file “3a.wav”

```
addObject(new Key("g", "3a.wav", 300, 180))
```

Run It and See What Happen
- Update Piano() of Piano class with the following

```
public class Piano extends World {
    public Piano() {
        super(800, 340, 1);
        // Add the following line
        addObject( new Key("g", "3a.wav", 300, 180 ) );
    }
}
```

Problem
Not in a nice position
Need Some Arithmetic!

The Key is 63 x 280
Therefore the Center of the Key
Would be 31 ½ x 140

Add Another Key

- Update Piano() of Piano class again
  ```java
  public class Piano extends World {
    public Piano() {
      super(800, 340, 1);
      // Add the following line
      addObject(new Key("g", "3a.wav"), 32, 140);
    }
  }
  ```
- Run it and see

Part III

Making All White Keys
Add All 12 White Keys

- Update Piano() of Piano class as follows
  
  ```java
  public class Piano extends World {
      public Piano() {
          super(800, 340, 1);
          for(int i=0; i<12; i++) // Repeat 12 times
              addObject(new Key("g", "3a.wav"), 32, 140);
      }
  }
  ```

Run It and See

- Oops... all overlapped

Add All 12 White Keys

- Update Piano() of Piano class
  
  ```java
  public class Piano extends World {
      public Piano() {
          super(800, 340, 1);
          for(int i=0; i<12; i++) // Repeat 12 times
              addObject(new Key("g", "3a.wav"), 32 + i*63, 140);
      }
  }
  ```

Run It and See

- Hmm... better, but not perfect!
Add All 12 White Keys

• Update Piano() of Piano class

```java
public class Piano extends World {
    public Piano() {
        super(800, 340, 1);
        // Width of piano: 800
        // Width of 12 keys: 12 * 63 = 756
        // Empty space = 800 - 756 = 44
        // Half the space on each side = 44 / 2 = 22
        for (int i = 0; i < 12; i++)
            addObject(new Key("g", "3a.wav"),
                       22 + 32 + i*63, 140);
    }
}
```

Run It and See

![Image of a piano keyboard]

**Problem**
Perfect! But ... all keys binded with "g" and with the same sound file

Make Each Key Different

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>s</td>
<td>d</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>;</td>
<td>'</td>
<td>\</td>
</tr>
</tbody>
</table>

```java
String[] whiteKeys = { "a", "s", "d", "f", "g", "h", "j", "k", "l", ";", ",", ",\"\"\};
```

How about Notes?

• We can do something similar

```java
String[] whiteNotes = { "3c", "3d", "3e", "3f", "3g", "3a", "3b", "4c", "4d", "4e", "4f", "4g"};
```

• Update Piano class as follows:

```java
public class Piano extends World {
    private String[] whiteKeys = { "a", "s", "d", "f", "g", "h", "j", "k", "l", ";", ",", ",\"\"\};
    private String[] whiteNotes = { "3c", "3d", "3e", "3f", "3g", "3a", "3b", "4c", "4d", "4e", "4f", "4g"};
    public Piano() {
        super(800, 340, 1);
        for (int i = 0; i < 12; i++)
            addObject(new Key(whiteKeys[i], whiteNotes[i] + ".wav"),
                       22 + 32 + i*63, 140);
    }
}
```
Run It and See

Your First Workable Piano. Play!! :D

Part IV
Making All Black Keys

Can Include Black Keys - Different Key Images
- First, add two variables and update `Key(...)` and `act()` method as follows:

```
public class Key extends Actor {
    private boolean isDown = false;
    private String key;
    private String sound;
    // Add two more variables below
    private String upImage;
    private String downImage;

    public Key(String keyName, String soundFile, String img1, String img2) {
        key = keyName;
        sound = soundFile;
        upImage = img1;
        downImage = img2;
        setImage(upImage);
    }

    public void act() {
        if(!isDown && Greenfoot.isKeyDown(key)) {
            setImage(downImage); // Change this play();
            isDown = true;
        } if(isDown && !Greenfoot.isKeyDown(key)) {
            setImage(upImage); // Change this
            isDown = false;
        }
    }
}
```

- Next, update `Piano` class as follows:

```
public class Piano extends World {

    private String[] whiteKeys = {
        "a", "a", "d", "f", "g", "b", "j", "k", "l", ";", ",", ");
    private String[] whiteNotes = {
        "3c", "3d", "3f", "3g", "3a", "3b", "4c", "4d", "4e", "4f", "4g";
    private String[] blackKeys = {
        "u", "e", "t", "y", "u", ",", ",", ",", ",", ");
    private String[] blackNotes = {
        "3c#", "3d#", ",", "3f#", "3g#", "4c#", "4d#", ",", "4f#";

    public Piano() {
        super(800, 340, 1);
        for(int i=0; i<12; i++) {
            Key key = new Key(whiteKeys[i], whiteNotes[i]+".wav",
                                "white-key.png", "white-key-down.png");
            addObject(key, 22 + 32 + i*63, 140);
        }
        for(int i=0; i<12-1; i++) {
            // Add another loop to create black keys
            if(!blackKeys[i].equals("")) {
                Key key = new Key(blackKeys[i], blackNotes[i]+".wav",
                                    "black-key.png", "black-key-down.png");
                addObject(key, 22 + (63/2) + 32 + i*63, 86);
            }
        }
    }
}
```
Add Methods to Control Key Up and Down

- Add `whiteKeyDownUp` and `blackKeyDownUp` methods to `Key` class

```java
public class Key extends Actor {
    // ...
    public void whiteKeyDownUp() {
        setImage("white-key-down.png");
        Greenfoot.playSound(sound);
        Greenfoot.delay(15);
        setImage("white-key.png");
    }

    public void blackKeyDownUp() {
        setImage("black-key-down.png");
        Greenfoot.playSound(sound);
        Greenfoot.delay(15);
        setImage("black-key.png");
    }
}
```

Update Piano Class

```java
public class Piano extends World {
    private String[] whiteKeys = {"a", "e", "f", "g", "h", "j", "k", "l", ",", ",", ","};
    private String[] whiteNotes = {"3c", "3d", "3e", "3f", "3g", "3a", "3b", "4c", "4d", "4e", "4f", "4g"};
    private Key[] pianoWhiteKey = new Key[12];
    private String[] blackKeys = {"w", "e", "r", "t", "y", "u", ",", "i", "o", "p", ",", ","};
    private String[] blackNotes = {"3c#", "3d#", "3e#", "3f#", "3g#", "3a#", ",", "4c#", "4d#", ",", ","};
    private Key[] pianoBlackKey = new Key[11];
    public Piano() {
        super(800, 340, 1);
        for(int i=0; i<12; i++) {
            pianoWhiteKey[i] = new Key(whiteKeys[i], whiteNotes[i]+".wav", "white-key.png", "white-key-down.png");
            addObject(pianoWhiteKey[i], 22 + 32 * i, 140);
        }
    }
    for(int i=0; i<12-1; i++) {
        // Add another loop to create black keys
        if(!blackKeys[i].equals(")\n            pianoBlackKey[i] = new Key(blackKeys[i], blackNotes[i]+".wav", "black-key.png", "black-key-down.png");
            addObject(pianoBlackKey[i], 22 + (63/2) + 32 * i, 86);
        // Shifted by half-width of white key
    }
```

Part V
Making a Music Player
Add pressKey and playSong Method to Piano Class

- Add pressKey() and playSong() method to Piano class as follows:

```java
public class Piano extends World {
    // ...
    private void pressKey(int i) {
        if (i >= 0 && i < 90) {
            if (i <= 20)
                pianoWhiteKey[i].whiteKeyDownUp(); // i <= 20 are for white keys
            if (i >= 50 && i != 52 && i != 56 && i != 59)
                pianoBlackKey[i-50].blackKeyDownUp(); // i >= 50: some are for black keys
        }
    }

    public void playSong() {
        // Sound of Music
        int[] notes = 
            {1,1,2,3,99,1,1,3,99,2,3,4,4,3,2,4,99,3,4,5,99,3,5,3,99,4,5,6,6,5,4,99,5,99,1,2,3,4,5,6,99,6,99,2,3,5,4,5,6,7,99,7,99,3,54,56,7,8,9,99,7,56,6,4,7,5,8,7,3,2,0};

        int i = 0;
        while(notes[i] != 0) {
            if((notes[i] >= 1 && notes[i] <= 12) || (notes[i] >= 51 && notes[i] <= 61))
                pressKey(notes[i]-1);
            else
                Greenfoot.delay(15);
            i++;
        }
    }
}
```

Other Songs

```java
int[] wedding = 
    {99,6,99,6,7,7,8,8,7,7,6,6,3,3,1,1,5,5,4,4,3,4,5,4,99,99,99,4,4,5,5,6,6, 7,7,5,5,2,2,4,4,3,3,2,3,4,3,99,99,99,99,10,99,6,8,10,9,10,99,6,8,10,9,10, 99,6,8,11,10,11,99,6,8,11,10,11,99,4,3,4,5,99,5,6,6,3,99,99,99,99,10, 99,6,8,10,9,10,99,6,8,10,9,10,99,6,8,11,10,11,99,6,8,11,10,11,99,4,3,4,5,4, 5,99,5,6,5,6,3,99,99,99,99,0};
```

```java
int[] jasmin = 
```

```java
int[] happyBirthday = 
    {99,5,5,6,5,8,7,9,99,5,6,9,8,99,5,5,12,10,8,7,6,13,99,11,11,10,8,9,8,99,0};
```

```java
int[] ohSusanna = 
    {99,1,2,3,5,5,99,6,5,3,1,99,2,3,3,2,1,2,99,1,2,3,5,5,99,6,5,3,1,99,2,3,3,2,2, 1,99,9,4,99,4,99,5,6,6,99,5,3,2,1,99,1,2,3,5,5,6,3,1,99,2,3,3,2,2,1, 99,0};
```

```java
int[] ShanghaiBeach = 
    {3,5,6,99,3,5,2,99,3,5,6,8,6,5,1,3,2,99,2,3,5,99,99,2,3,6,6,1,2,3,2,7,6,5,1,99, 8,8,6,99,6,6,6,5,3,6,5,1,2,1,3,99,3,3,2,3,99,8,7,6,99,3,3,2,3,8,7,6,3,5,99,3,5,6,99,3,5,2,99,3,5,6,8,6,5,1,3,2,99,3,5,2,3,6,99,99,6,1,2,3,2,7,6,5, 1,0};
```

That's all!
Any questions?

Thank you!