//Control.h

#include <string>

#include <vector>

#include <algorithm>

#include <iostream>

using namespace std;

class Controller

{

private:

int row = 0;

int col = 0;

int \*\*matrix;

public:

Controller(int row, int col)

{

this->row = row;

this->col = col;

cout << row << string(L",") << col << endl;

createMatrix();

showMatrix();

}

virtual void showMatrix()

{

for (int i = 0; i < row; i++)

{

for (int j = 0; j < col; j++)

{

printf(L"%3d", matrix[i][j]);

}

cout << endl;

}

}

private:

void createMatrix()

{

matrix = RectangularArrays::ReturnRectangularIntArray(row, col);

Random \*rand = new Random();

int imgCount = 21;

int max = 4;

int arr[imgCount + 1];

ArrayList<Point> listPoint = new ArrayList<Point>();

for (int i = 0; i < row; i++)

{

for (int j = 0; j < col; j++)

{

listPoint.push\_back(new Point(i, j));

}

}

int i = 0;

do

{

int index = rand->nextInt(imgCount) + 1;

if (arr[index] < max)

{

arr[index] += 2;

for (int j = 0; j < 2; j++)

{

int size = listPoint.size();

int pointIndex = rand->nextInt(size);

matrix[listPoint[pointIndex]->x][listPoint[pointIndex]->y] = index;

listPoint.remove(pointIndex);

}

i++;

}

} while (i < row \* col / 2);

}

public:

virtual int getRow()

{

return row;

}

virtual void setRow(int row)

{

this->row = row;

}

virtual int getCol()

{

return col;

}

virtual void setCol(int col)

{

this->col = col;

}

virtual int \*\*getMatrix()

{

return matrix;

}

virtual void setMatrix(int \*matrix[])

{

this->matrix = matrix;

}

};

}

class RectangularArrays

{

public:

static int\*\* ReturnRectangularIntArray(int size1, int size2)

{

int\*\* newArray;

if (size1 > -1)

{

newArray = new int\*[size1];

if (size2 > -1)

{

for (int array1 = 0; array1 < size1; array1++)

{

newArray[array1] = new int[size2] ();

}

}

}

return newArray;

}

};

//ButtonEvent.h

#include <string>

class ButtonEvent : public JPanel, public ActionListener

{

private:

int row = 0;

int col = 0;

int bound = 2;

int size = 50;

JButton \*\*\*btn;

Controller \*controller;

Color \*backGroundColor = Color::lightGray;

MainFrame \*frame;

public:

ButtonEvent(MainFrame \*frame, int row, int col)

{

this->frame = frame;

this->row = row;

this->col = col;

setLayout(new GridLayout(row, col, bound, bound));

setBackground(backGroundColor);

setPreferredSize(new Dimension((size + bound) \* col, (size + bound) \* row));

setBorder(new EmptyBorder(10, 10, 10, 10));

setAlignmentY(JPanel::CENTER\_ALIGNMENT);

newGame();

}

virtual void newGame()

{

controller = new Controller(this->row, this->col);

addArrayButton();

}

private:

void addArrayButton()

{

btn = RectangularArrays::ReturnRectangularJavaxswingJButtonArray(row, col);

for (int i = 0; i < row; i++)

{

for (int j = 0; j < col; j++)

{

btn[i][j] = createButton(i + string(L",") + j);

Icon \*icon = getIcon(controller->getMatrix()[i][j]);

btn[i][j]->setIcon(icon);

add(btn[i][j]);

}

}

}

Icon \*getIcon(int index)

{

int width = 48, height = 48;

Image \*image = (new ImageIcon(getClass().getResource(string(L"/icon/") + index + string(L".png"))))->getImage();

Icon \*icon = new ImageIcon(image->getScaledInstance(width, height, image->SCALE\_SMOOTH));

return icon;

}

JButton \*createButton(const string &action)

{

JButton \*btn = new JButton();

btn->setActionCommand(action);

btn->setBorder(nullptr);

btn->addActionListener(this);

return btn;

}

public:

virtual void actionPerformed(ActionEvent \*e) override

{

}

};

}

class RectangularArrays

{

public:

static javax::swing::JButton\*\*\* ReturnRectangularJavaxswingJButtonArray(int size1, int size2)

{

javax::swing::JButton\*\*\* newArray;

if (size1 > -1)

{

newArray = new javax::swing::JButton\*\*[size1];

if (size2 > -1)

{

for (int array1 = 0; array1 < size1; array1++)

{

newArray[array1] = new javax::swing::JButton\*[size2] ();

}

}

}

return newArray;

}

};

//MainFrame.h

class MainFrame : public JFrame, public ActionListener, public Runnable

{

private:

int row = 8;

int col = 8;

int width = 700;

int height = 500;

ButtonEvent \*graphicsPanel;

JPanel \*mainPanel;

public:

MainFrame()

{

add(mainPanel = createMainPanel());

setTitle(L"Pokemon Game");

setResizable(false);

setDefaultCloseOperation(JFrame::EXIT\_ON\_CLOSE);

setSize(width, height);

setLocationRelativeTo(nullptr);

setVisible(true);

}

private:

JPanel \*createMainPanel()

{

JPanel \*panel = new JPanel(new BorderLayout());

panel->add(createGraphicsPanel(), BorderLayout::CENTER);

return panel;

}

JPanel \*createGraphicsPanel()

{

graphicsPanel = new ButtonEvent(this, row, col);

JPanel \*panel = new JPanel(new GridBagLayout());

panel->setBackground(Color::gray);

panel->add(graphicsPanel);

return panel;

}

public:

virtual void actionPerformed(ActionEvent \*e) override

{

}

virtual void run() override

{

}

};

}

//Main.cpp

Main::Main()

{

frame = new MainFrame();

}

void Main::main(string args[])

{

new Main();

}

}

Kết quả:

