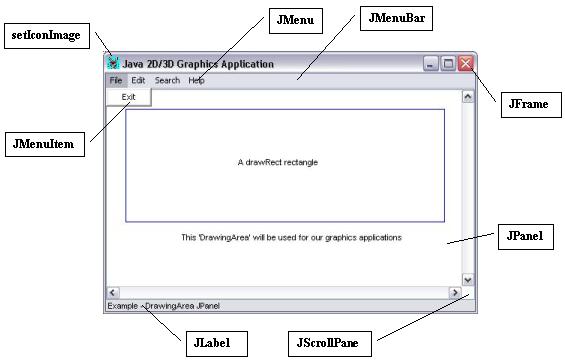
**Swing Menu –нүүд**

Жава дээр бид цэс үүсгэх болон цэстэй холбоотой ажилахдаа жава хэлний өөрийнх нь классууд болох JMenuBar, JMenu, JMenuItem, JCheckBoxMenuItem, JRadioButtonMenuItem, JSeparator, JPopupMenu –г ашигладаг.

Цонхонд цэс үүсгэхэд дараах алхамуудаар үүсгэнэ:

* JMenuBar –г үүсгэнэ.
* JMenu –нүүдийг үүсгэнэ.
* JMenuItem –үүдийг үүсгэнэ.
* JMenu –д харгалзах JMenuItem –үүдийг нэмнэ.
* JMenuBar –т үүсгэсэн JMenu –нүүдийг нэмнэ.
* JMenuBar –г цонхонд нэмнэ.



Цонхонд цэсний мөрийг нэмэхдээ setJMenuBar( ) функцыг ашигладаг. Уг функц нь цэсний мөрийг цонхны дээд хэсэгт автоматаар байрлуулдаг ба цэсийг JFrame-ийн RootPane давхаргад нэмдэг.

**Жишээ**

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**class** MenuDemo **implements** ActionListener{

JLabel jlab;

MenuDemo() {

JFrame jfrm = **new** JFrame("Menu Demo");

jfrm.setLayout(**new** FlowLayout());

jfrm.setSize(220, 200);

jfrm.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

jlab = **new** JLabel();

JMenuBar jmb = **new** JMenuBar();

// Create the File menu.

JMenu jmFile = **new** JMenu("File");

JMenuItem jmiOpen = **new** JMenuItem("Open");

JMenuItem jmiClose = **new** JMenuItem("Close");

JMenuItem jmiSave = **new** JMenuItem("Save");

JMenuItem jmiExit = **new** JMenuItem("Exit");

jmFile.add(jmiOpen);

jmFile.add(jmiClose);

jmFile.add(jmiSave);

jmFile.addSeparator();

jmFile.add(jmiExit);

jmb.add(jmFile);

// Create the Options menu.

JMenu jmOptions = **new** JMenu("Options");

// Create the Colors submenu.

JMenu jmColors = **new** JMenu("Colors");

JMenuItem jmiRed = **new** JMenuItem("Red");

JMenuItem jmiGreen = **new** JMenuItem("Green");

JMenuItem jmiBlue = **new** JMenuItem("Blue");

jmColors.add(jmiRed);

jmColors.add(jmiGreen);

jmColors.add(jmiBlue);

jmOptions.add(jmColors);

// Create the Priority submenu.

JMenu jmPriority = **new** JMenu("Priority");

JMenuItem jmiHigh = **new** JMenuItem("High");

JMenuItem jmiLow = **new** JMenuItem("Low");

jmPriority.add(jmiHigh);

jmPriority.add(jmiLow);

jmOptions.add(jmPriority);

// Create the Reset menu item.

JMenuItem jmiReset = **new** JMenuItem("Reset");

jmOptions.addSeparator();

jmOptions.add(jmiReset);

// Finally, add the entire options menu to the menu bar

jmb.add(jmOptions);

// Create the Help menu.

JMenu jmHelp = **new** JMenu("Help");

JMenuItem jmiAbout = **new** JMenuItem("About");

jmHelp.add(jmiAbout);

jmb.add(jmHelp);

// Add action listeners for the menu items.

jmiOpen.addActionListener(**this**);

jmiClose.addActionListener(**this**);

jmiSave.addActionListener(**this**);

jmiExit.addActionListener(**this**);

jmiRed.addActionListener(**this**);

jmiGreen.addActionListener(**this**);

jmiBlue.addActionListener(**this**);

jmiHigh.addActionListener(**this**);

jmiLow.addActionListener(**this**);

jmiReset.addActionListener(**this**);

jmiAbout.addActionListener(**this**);

// Add the label to the content pane.

jfrm.add(jlab);

// Add the menu bar to the frame.

jfrm.setJMenuBar(jmb);

// Display the frame.

jfrm.setVisible(**true**);

}

// Handle menu item action events.

**public** **void** actionPerformed(ActionEvent ae) {

// Get the action command from the menu selection.

String comStr = ae.getActionCommand();

// If user chooses Exit, then exit the program.

**if**(comStr.equals("Exit")) System.*exit*(0);

// Otherwise, display the selection.

jlab.setText(comStr + " Selected");

}

**public** **static** **void** main(String args[]) {

SwingUtilities.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** MenuDemo();

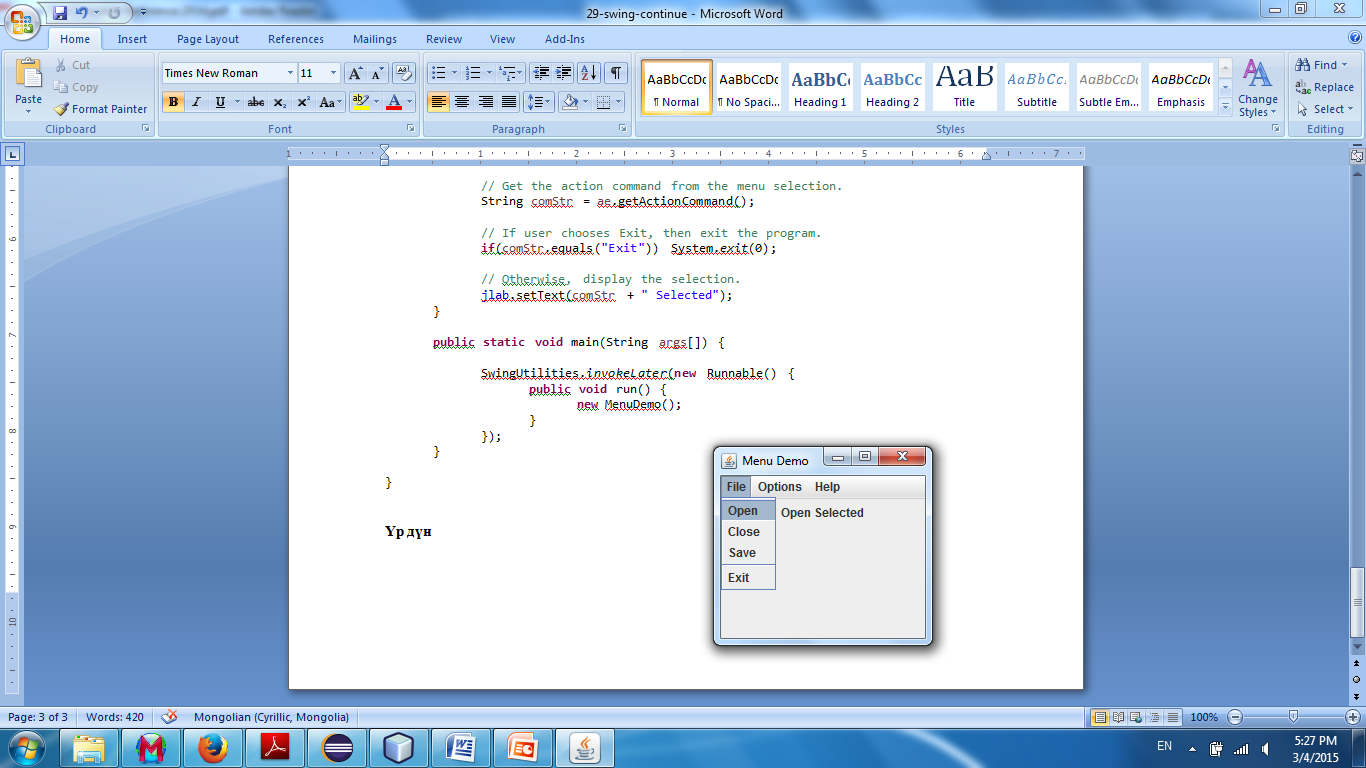
}

});

}

}

**Үр дүн**



Цэсийг гараас нээх боломжтой болгох /mnemonic/ Цэсийг гараас нээх боломжтой болгохдоо (Alt + Үсэг) JMenu классын setMnemonic(char mnemonic) функцыг ашиглана. Уг функцын аргументаар дамжуулж буй тэмдэгт нь цэсний нэрэнд агуулагдсан байх ёстой гэдгийг анхаараарай.

Дараах байдалтай бичнэ: үр дүнг хархад File цэсний доор доогуур зураас гарсан байна. Тиймээс гарнын Alt + key хослолыг хэрэглэхэд File цэс нээгдэнэ гэсэн үг.

// Create the File menu.

JMenu jmFile = **new** JMenu("File");

jmFile.setMnemonic('F');

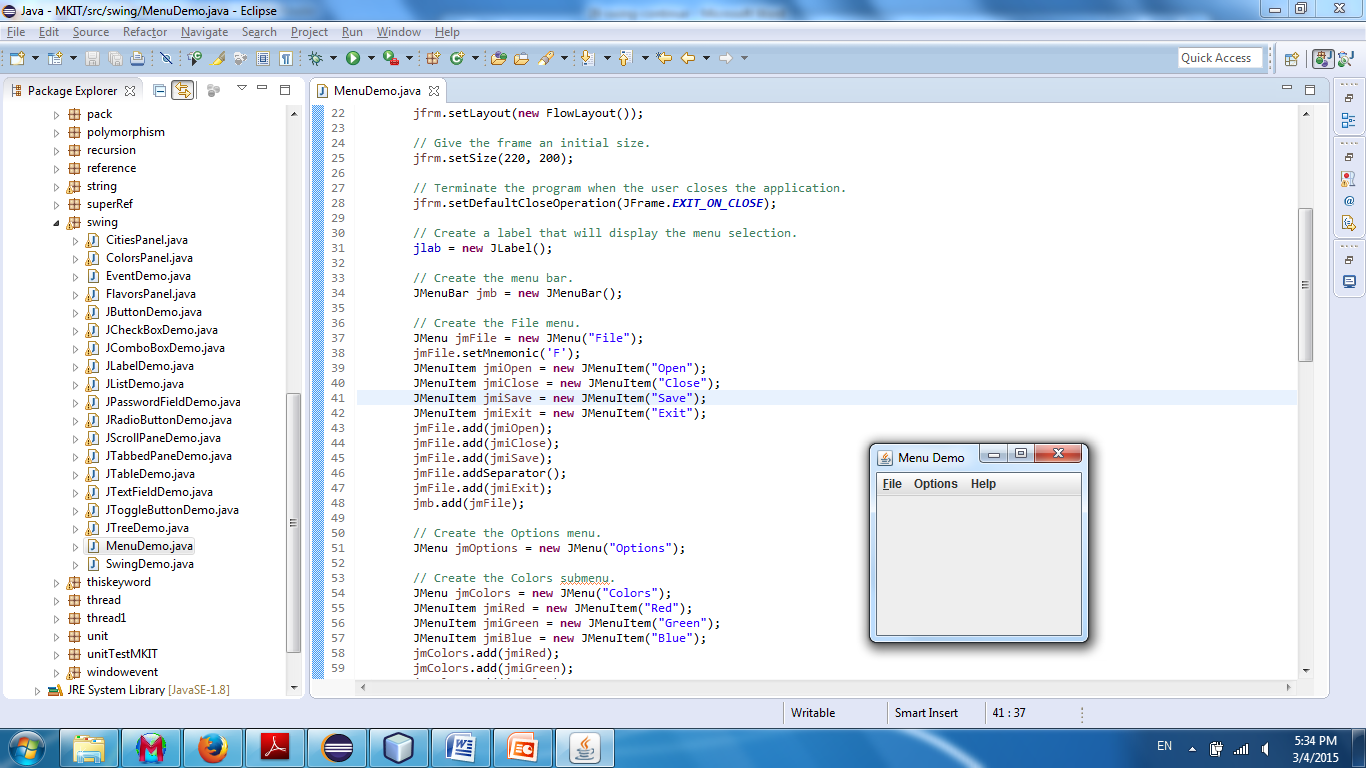
Мөн өөрөөр бас доорх маягаар бичиж болно.

// Create the File menu.

JMenu jmFile = **new** JMenu("File");

jmFile.setMnemonic(KeyEvent.***VK\_F***);

**Үр дүн**

****

Дэд цэсэнд гарын хослол нэмэх: **JMenuItem** классын **setAccelerator(KeyStroke arg)** функцыг ашиглана. Дараах байдлаар бичнэ.

JMenuItem jmiOpen = **new** JMenuItem("Open",KeyEvent.***VK\_O***);

jmiOpen.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_O***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiClose = **new** JMenuItem("Close",KeyEvent.***VK\_C***);

jmiClose.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_C***,InputEvent.***CTRL\_DOWN\_MASK***));

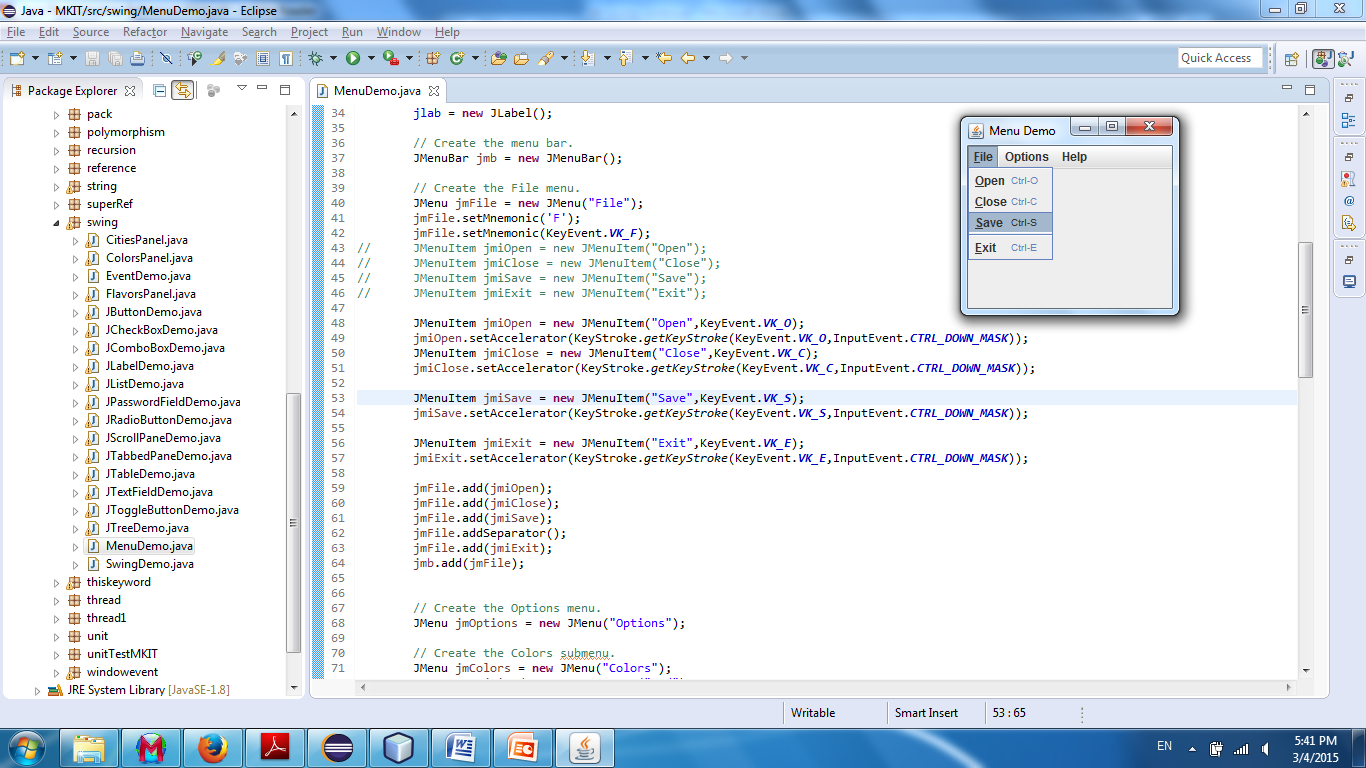
JMenuItem jmiSave = **new** JMenuItem("Save",KeyEvent.***VK\_S***);

jmiSave.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_S***,InputEvent.***CTRL\_DOWN\_MASK***));

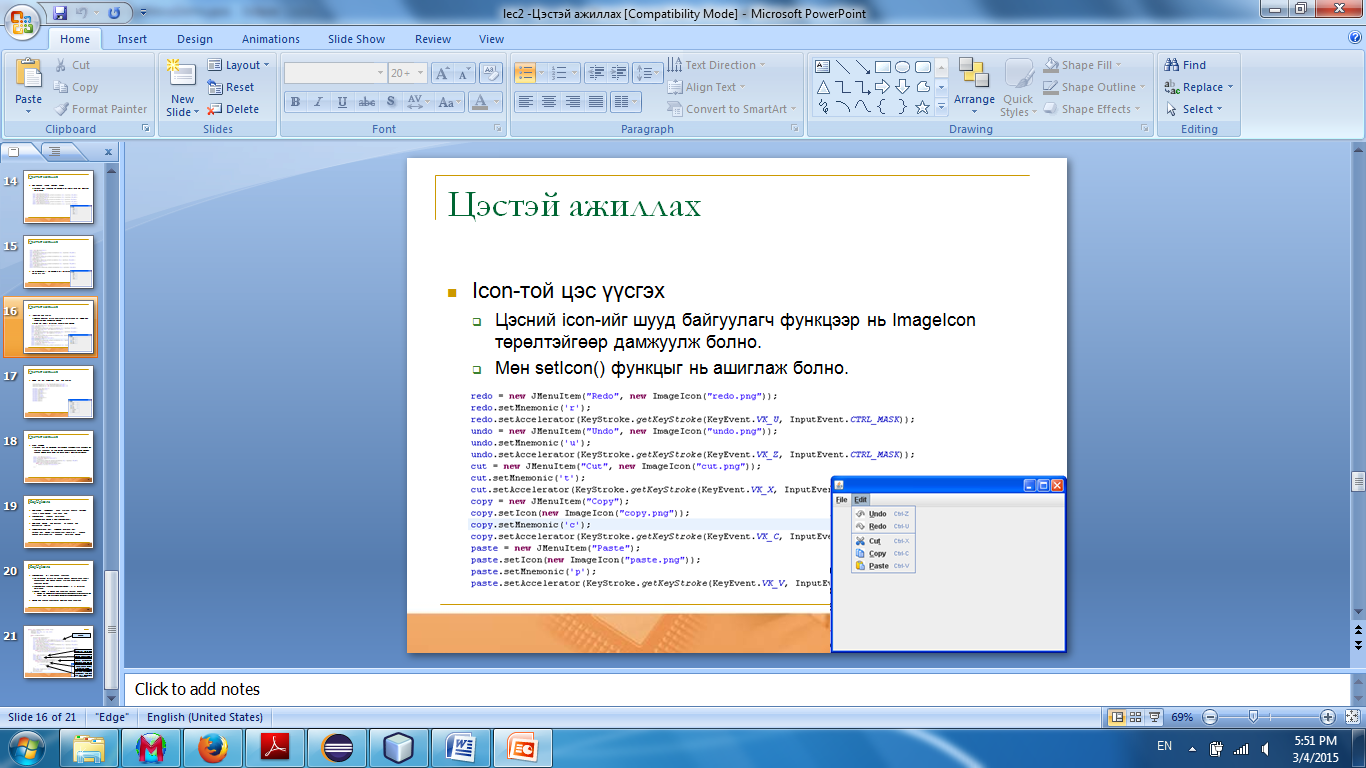
JMenuItem jmiExit = **new** JMenuItem("Exit",KeyEvent.***VK\_E***);

jmiExit.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_E***,InputEvent.***CTRL\_DOWN\_MASK***));

**Үр дүн**



Мөн цэсэнд icon нэмхэд хоёр янзаар хийж өгч болно. Цэсний icon –г шууд байгуулагч функцээр нь ImageIcon төрөлтэйгээр дамжуулж болно. Нөгөө нэг арга нь setIcon( ) функцийг ашиглаж болно.



**JRadioButtonMenuItem** болон **JCheckBoxMenuItem**  классуудын тухай

Эдгээр класс нь цонхны дэд цэсэнд radio button болон checkbox цэс нэмхэд хэрэглэгддэг классууд. Эдгээр классын event –г барьж авахын тулд ActionListener интерфейсийг имплемент хийн түүний actionPerformed( ) функцийг тодорхойлж өгнө.

**Жишээ**

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**class** MenuDemoRadioCheck **implements** ActionListener{

JLabel jlab;

ButtonGroup group;

MenuDemoRadioCheck() {

JFrame jfrm = **new** JFrame("Menu Demo");

jfrm.setLayout(**new** FlowLayout());

jfrm.setSize(220, 200);

jfrm.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

jlab = **new** JLabel();

JMenuBar jmb = **new** JMenuBar();

JMenu jmFile = **new** JMenu("File");

jmFile.setMnemonic('F');

jmFile.setMnemonic(KeyEvent.***VK\_F***);

JMenuItem jmiOpen = **new** JMenuItem("Open",KeyEvent.***VK\_O***);

jmiOpen.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_O***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiClose = **new** JMenuItem("Close",KeyEvent.***VK\_C***);

jmiClose.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_C***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiSave = **new** JMenuItem("Save",KeyEvent.***VK\_S***); jmiSave.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_S***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiExit = **new** JMenuItem("Exit",KeyEvent.***VK\_E***);

jmiExit.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_E***,InputEvent.***CTRL\_DOWN\_MASK***));

jmFile.add(jmiOpen);

jmFile.add(jmiClose);

jmFile.add(jmiSave);

jmFile.addSeparator();

jmFile.add(jmiExit);

jmb.add(jmFile);

JMenu jmOptions = **new** JMenu("Options");

JMenu jmColors = **new** JMenu("Colors");

JCheckBoxMenuItem jmiRed = **new** JCheckBoxMenuItem("Red");

JCheckBoxMenuItem jmiGreen = **new** JCheckBoxMenuItem("Green");

JCheckBoxMenuItem jmiBlue = **new** JCheckBoxMenuItem("Blue");

jmColors.add(jmiRed);

jmColors.add(jmiGreen);

jmColors.add(jmiBlue);

jmOptions.add(jmColors);

JMenu jmPriority = **new** JMenu("Priority");

JRadioButtonMenuItem jmiHigh = **new** JRadioButtonMenuItem("High", **true**);

JRadioButtonMenuItem jmiLow = **new** JRadioButtonMenuItem("Low");

group = **new** ButtonGroup();

group.add(jmiHigh);

group.add(jmiLow);

jmPriority.add(jmiHigh);

jmPriority.add(jmiLow);

jmOptions.add(jmPriority);

JMenuItem jmiReset = **new** JMenuItem("Reset");

jmOptions.addSeparator();

jmOptions.add(jmiReset);

jmb.add(jmOptions);

JMenu jmHelp = **new** JMenu("Help");

JMenuItem jmiAbout = **new** JMenuItem("About");

jmHelp.add(jmiAbout);

jmb.add(jmHelp);

jmiOpen.addActionListener(**this**);

jmiClose.addActionListener(**this**);

jmiSave.addActionListener(**this**);

jmiExit.addActionListener(**this**);

jmiRed.addActionListener(**this**);

jmiGreen.addActionListener(**this**);

jmiBlue.addActionListener(**this**);

jmiHigh.addActionListener(**this**);

jmiLow.addActionListener(**this**);

jmiReset.addActionListener(**this**);

jmiAbout.addActionListener(**this**);

jfrm.add(jlab);

jfrm.setJMenuBar(jmb);

jfrm.setVisible(**true**);

}

**public** **void** actionPerformed(ActionEvent ae) {

String comStr = ae.getActionCommand();

**if**(comStr.equals("Exit")) System.*exit*(0);

jlab.setText(comStr + " Selected");

}

**public** **static** **void** main(String args[]) {

SwingUtilities.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** MenuDemoRadioCheck();

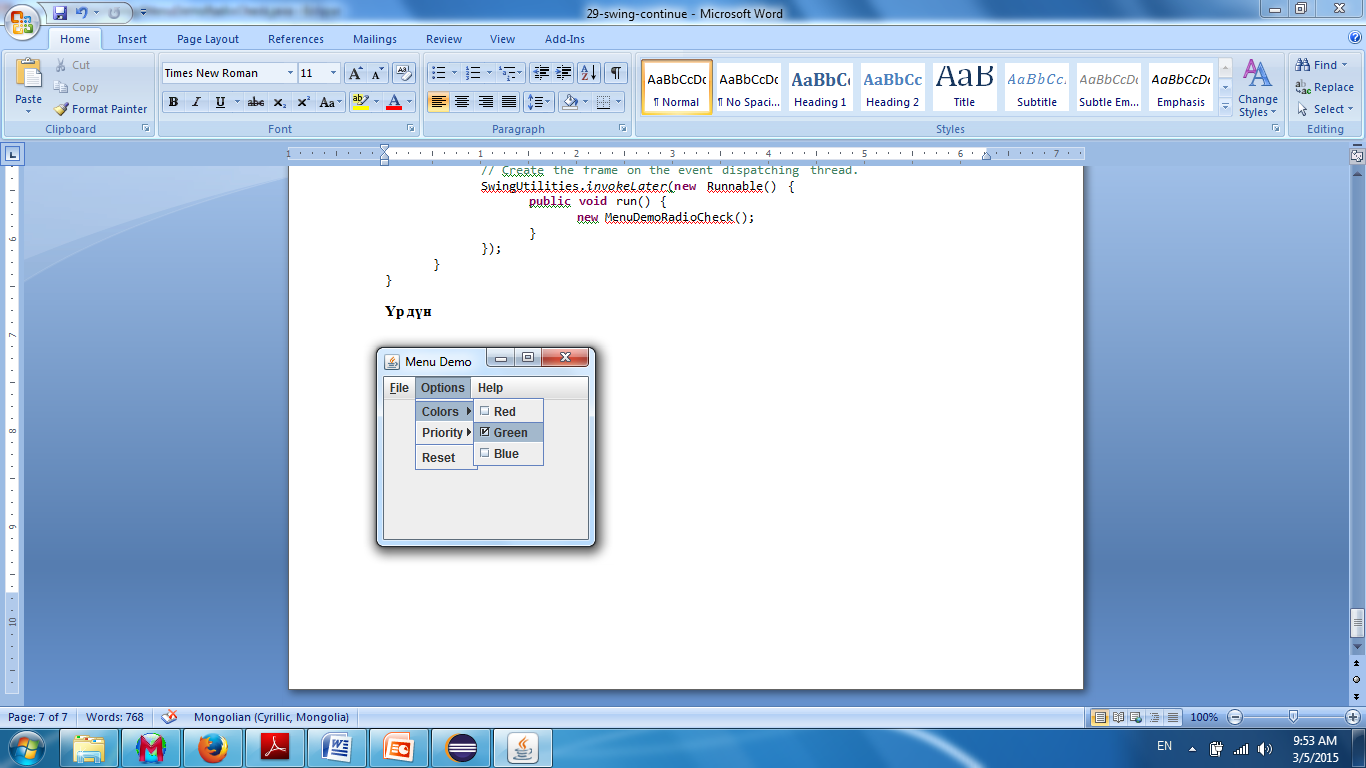
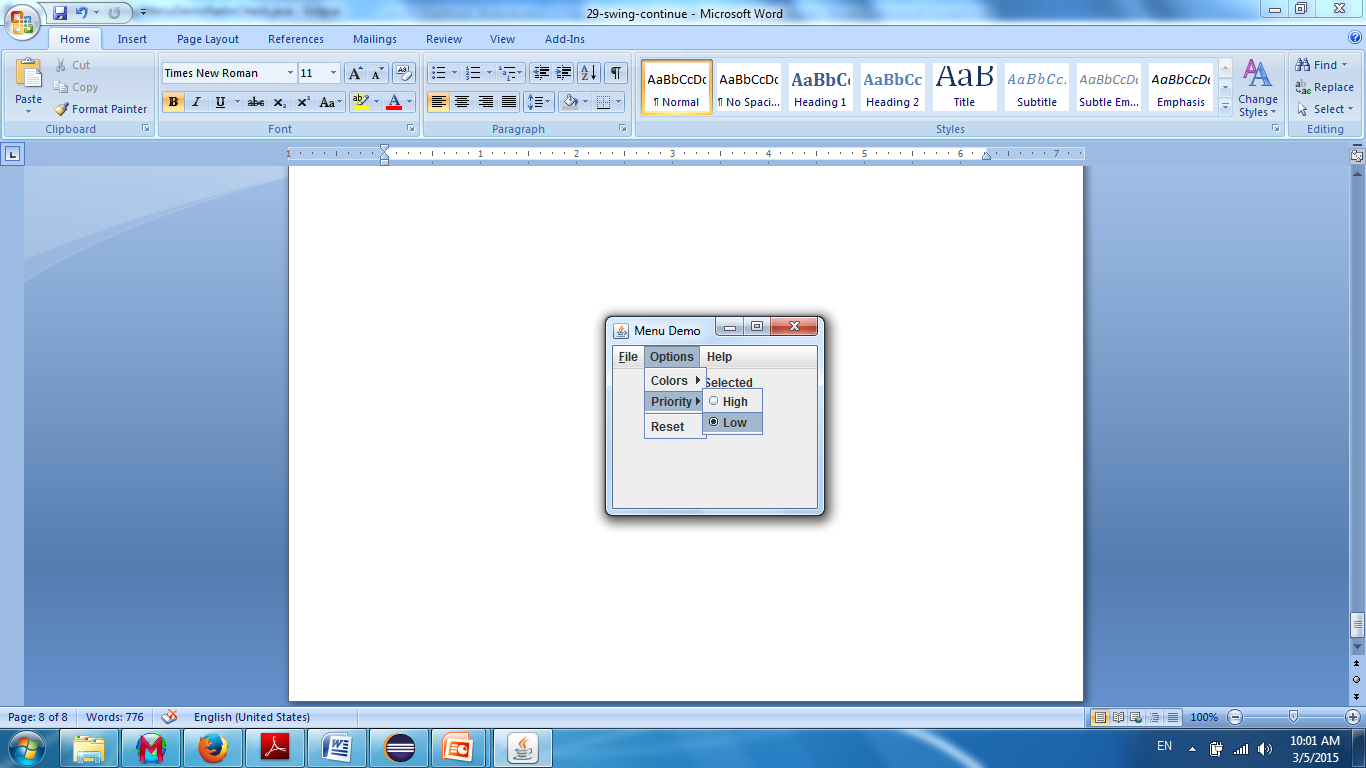
}

});

}

}

**Үр дүн**

**JPopupMenu** классын тухай

Програмын component дээр хулганы баруун товчоор (зүүн ч байж болно) гарч ирэх цэсийг хийхдээ дээрх классыг ашиглана. Програмд PopUp цэс оруулах нь үндсэн цэс оруулахтай төстэй. PopupMenu –г дэлгэцэнд харуулахдаа JPopupMenu классын show(component, x, y) функцийг ашигладаг. Popup-trigger гэдэг нь popup цэс харагдах товчийг дарах гэсэн үг. Өөрөөр хэлбэл Windows үйлдлийн систем дээр popup цэсийг хулганы баруун товч дарахад гарч ирдэг бол Motif үйлдлийн систем дээр хулганы голын товч дархад popup цэс гарч ирдэг. Тиймээс бид тухайн popup цэс дарагдсан event –г барьж авхад бидний урьд нь үзэж байсан ойлголт болох MouseListener интерфейсийг имплемент хийн түүний таван функцийг тодорхойлж өгнө.

**void mouseClicked(MouseEvent *me*)**

**void mouseEntered(MouseEvent *me*)**

**void mouseExited(MouseEvent *me*)**

**void mousePressed(MouseEvent *me*)**

**void mouseReleased(MouseEvent *me*)**

Popup цэс харуулах товч дарагдсан эсэхийг шалгахдаа **boolean isPopupTrigger( )** функцийг ашигладаг. **Component getComponent( )** функцээр тухайн popup цэсэнд байрлах component –үүдийг авхад ашиглагддаг.

**Жишээ**

**import** java.awt.FlowLayout;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**class** PopupMenuDemo **implements** ActionListener{

JLabel jlab;

JPopupMenu jpu;

**public** PopupMenuDemo() {

JFrame jfrm = **new** JFrame("Menu Demo");

jfrm.setLayout(**new** FlowLayout());

jfrm.setSize(220, 200);

jfrm.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

jlab = **new** JLabel();

JMenuBar jmb = **new** JMenuBar();

JMenu jmFile = **new** JMenu("File");

jmFile.setMnemonic('F');

jmFile.setMnemonic(KeyEvent.***VK\_F***);

JMenuItem jmiOpen = **new** JMenuItem("Open",KeyEvent.***VK\_O***);

jmiOpen.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_O***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiClose = **new** JMenuItem("Close",KeyEvent.***VK\_C***);

jmiClose.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_C***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiSave = **new** JMenuItem("Save",KeyEvent.***VK\_S***);

jmiSave.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_S***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiExit = **new** JMenuItem("Exit",KeyEvent.***VK\_E***);

jmiExit.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_E***,InputEvent.***CTRL\_DOWN\_MASK***));

jmFile.add(jmiOpen);

jmFile.add(jmiClose);

jmFile.add(jmiSave);

jmFile.addSeparator();

jmFile.add(jmiExit);

jmb.add(jmFile);

JMenu jmOptions = **new** JMenu("Options");

JMenu jmColors = **new** JMenu("Colors");

JMenuItem jmiRed = **new** JMenuItem("Red");

JMenuItem jmiGreen = **new** JMenuItem("Green");

JMenuItem jmiBlue = **new** JMenuItem("Blue");

jmColors.add(jmiRed);

jmColors.add(jmiGreen);

jmColors.add(jmiBlue);

jmOptions.add(jmColors);

JMenu jmPriority = **new** JMenu("Priority");

JMenuItem jmiHigh = **new** JMenuItem("High");

JMenuItem jmiLow = **new** JMenuItem("Low");

jmPriority.add(jmiHigh);

jmPriority.add(jmiLow);

jmOptions.add(jmPriority);

JMenuItem jmiReset = **new** JMenuItem("Reset");

jmOptions.addSeparator();

jmOptions.add(jmiReset);

jmb.add(jmOptions);

JMenu jmHelp = **new** JMenu("Help");

JMenuItem jmiAbout = **new** JMenuItem("About");

jmHelp.add(jmiAbout);

jmb.add(jmHelp);

jmiOpen.addActionListener(**this**);

jmiClose.addActionListener(**this**);

jmiSave.addActionListener(**this**);

jmiExit.addActionListener(**this**);

jmiRed.addActionListener(**this**);

jmiGreen.addActionListener(**this**);

jmiBlue.addActionListener(**this**);

jmiHigh.addActionListener(**this**);

jmiLow.addActionListener(**this**);

jmiReset.addActionListener(**this**);

jmiAbout.addActionListener(**this**);

// Create an Edit popup menu.

jpu = **new** JPopupMenu();

// Create the popup menu items.

JMenuItem jmiCut = **new** JMenuItem("Cut");

JMenuItem jmiCopy = **new** JMenuItem("Copy");

JMenuItem jmiPaste = **new** JMenuItem("Paste");

// Add the menu items to the popup menu.

jpu.add(jmiCut);

jpu.add(jmiCopy);

jpu.add(jmiPaste);

// Add a listener for the popup trigger.

jfrm.addMouseListener(**new** MouseAdapter() {

**public** **void** mousePressed(MouseEvent me) {

**if**(me.isPopupTrigger())

jpu.show(me.getComponent(), me.getX(), me.getY());

}

**public** **void** mouseReleased(MouseEvent me) {

**if**(me.isPopupTrigger())

jpu.show(me.getComponent(), me.getX(), me.getY());

}

});

jfrm.add(jlab);

jfrm.setJMenuBar(jmb);

jfrm.setVisible(**true**);

}

**public** **void** actionPerformed(ActionEvent ae) {

String comStr = ae.getActionCommand();

**if**(comStr.equals("Exit")) System.*exit*(0);

jlab.setText(comStr + " Selected");

}

**public** **static** **void** main(String args[]) {

SwingUtilities.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** PopupMenuDemo();

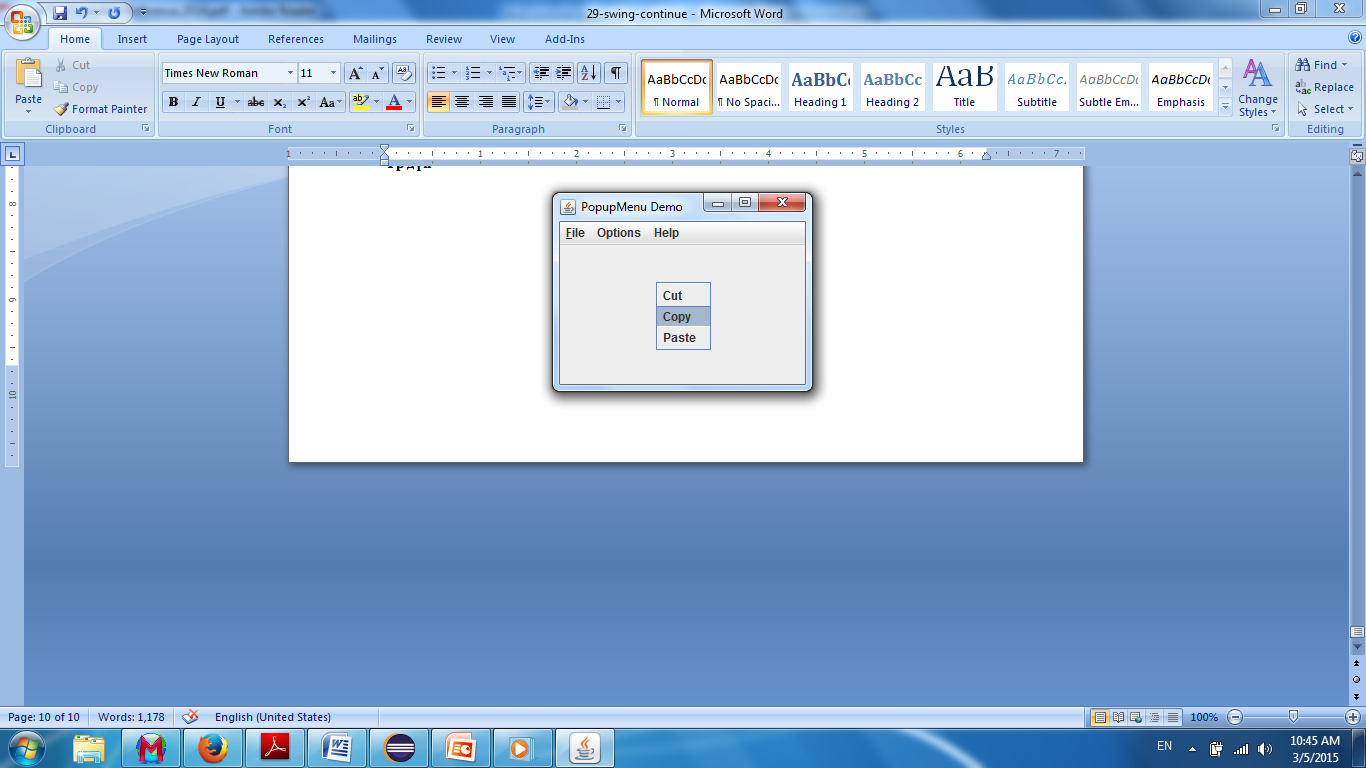
}

});

}

}

**Үр дүн**



**JToolBar** классын тухай

Энэ класс нь хөвдөг (хулганаар чирч өөр өөр газар байрлуулж болно.), агуулагч юм. setFloatable(boolean) функцд нь false утга дамжуулсанаар toolbar –г хөвдөггүй болгож болно. JToolBar –дээр ихэвчлэн icon –тэй товчнууд тавьдаг. JToolBar –дээр байрлах товчнуудын хэмжээ нь аль болох жижиг байх ёстой. JButton товчин дээрх текст, зураг дээд, доод талаасаа 2,2 px баруун, зүүн талаасаа 14, 14 px хэмжээтэй зай авдаг. Хэмжээг нь жижиг болгохын тулд энэ зайг 0,0,0,0 болгох хэрэгтэй. Тйимээс setMargins(insets) функцийг дуудна. Жишээ нь: **button.setMargins(new Insets(0,0,0,0));** гэж бичиж өгнө. Мөн товчны текст нь default-аараа зурагны баруун талд нь байрладаг.toolBar-ийн товчны хувьд текст нь зургийн доор байрлах нь тохиромжтой байдаг. **setVerticalTextPosition(BOTTOM) setHorizontalTextPosition(CENTER)** функцуудыг дуудаж текстийг доод хэсэгт голлуулна. Иймээс toolbar-т зориулсан **JButton** классаас удамшсан өөрийн классыг бичих нь ашигтай.

**Жишээ**

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**class** JToolBarDemo **implements** ActionListener {

JLabel jlab;

JPopupMenu jpu;

JToolBar jtb = **new** JToolBar("Debug");

**public** JToolBarDemo(){

JFrame jfrm = **new** JFrame("JToolBar Demo");

jfrm.setSize(220, 200);

jfrm.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

jlab = **new** JLabel();

JMenuBar jmb = **new** JMenuBar();

JMenu jmFile = **new** JMenu("File");

jmFile.setMnemonic('F');

jmFile.setMnemonic(KeyEvent.***VK\_F***);

JMenuItem jmiOpen = **new** JMenuItem("Open",KeyEvent.***VK\_O***);

jmiOpen.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_O***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiClose = **new** JMenuItem("Close",KeyEvent.***VK\_C***);

jmiClose.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_C***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiSave = **new** JMenuItem("Save",KeyEvent.***VK\_S***);

jmiSave.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_S***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiExit = **new** JMenuItem("Exit",KeyEvent.***VK\_E***);

jmiExit.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_E***,InputEvent.***CTRL\_DOWN\_MASK***));

jmFile.add(jmiOpen);

jmFile.add(jmiClose);

jmFile.add(jmiSave);

jmFile.addSeparator();

jmFile.add(jmiExit);

jmb.add(jmFile);

JMenu jmOptions = **new** JMenu("Options");

JMenu jmColors = **new** JMenu("Colors");

JMenuItem jmiRed = **new** JMenuItem("Red");

JMenuItem jmiGreen = **new** JMenuItem("Green");

JMenuItem jmiBlue = **new** JMenuItem("Blue");

jmColors.add(jmiRed);

jmColors.add(jmiGreen);

jmColors.add(jmiBlue);

jmOptions.add(jmColors);

JMenu jmPriority = **new** JMenu("Priority");

JMenuItem jmiHigh = **new** JMenuItem("High");

JMenuItem jmiLow = **new** JMenuItem("Low");

jmPriority.add(jmiHigh);

jmPriority.add(jmiLow);

jmOptions.add(jmPriority);

JMenuItem jmiReset = **new** JMenuItem("Reset");

jmOptions.addSeparator();

jmOptions.add(jmiReset);

jmb.add(jmOptions);

JMenu jmHelp = **new** JMenu("Help");

JMenuItem jmiAbout = **new** JMenuItem("About");

jmHelp.add(jmiAbout);

jmb.add(jmHelp);

jmiOpen.addActionListener(**this**);

jmiClose.addActionListener(**this**);

jmiSave.addActionListener(**this**);

jmiExit.addActionListener(**this**);

jmiRed.addActionListener(**this**);

jmiGreen.addActionListener(**this**);

jmiBlue.addActionListener(**this**);

jmiHigh.addActionListener(**this**);

jmiLow.addActionListener(**this**);

jmiReset.addActionListener(**this**);

jmiAbout.addActionListener(**this**);

jpu = **new** JPopupMenu();

JMenuItem jmiCut = **new** JMenuItem("Cut");

JMenuItem jmiCopy = **new** JMenuItem("Copy");

JMenuItem jmiPaste = **new** JMenuItem("Paste");

jpu.add(jmiCut);

jpu.add(jmiCopy);

jpu.add(jmiPaste);

jfrm.addMouseListener(**new** MouseAdapter() {

**public** **void** mousePressed(MouseEvent me) {

**if**(me.isPopupTrigger())

jpu.show(me.getComponent(), me.getX(), me.getY());

}

**public** **void** mouseReleased(MouseEvent me) {

**if**(me.isPopupTrigger())

jpu.show(me.getComponent(), me.getX(), me.getY());

}

});

jfrm.add(jlab, BorderLayout.***CENTER***);

jfrm.setJMenuBar(jmb);

jfrm.setVisible(**true**);

ImageIcon set = **new** ImageIcon("redo.png");

ImageIcon clear = **new** ImageIcon("undo.png");

ImageIcon resume = **new** ImageIcon("refresh.png");

JButton jbtnSet = **new** JButton(set);

jbtnSet.setActionCommand("Redo Breakpoint");

jbtnSet.setToolTipText("Redo Breakpoint");

JButton jbtnClear = **new** JButton(clear);

jbtnClear.setActionCommand("Undo Breakpoint");

jbtnClear.setToolTipText("Undo Breakpoint");

JButton jbtnResume = **new** JButton(resume);

jbtnResume.setActionCommand("Refresh");

jbtnResume.setToolTipText("Refresh");

jtb.add(jbtnSet);

jtb.add(jbtnClear);

jtb.add(jbtnResume);

jfrm.add(jtb, BorderLayout.***NORTH***);

}

**public** **void** actionPerformed(ActionEvent ae) {

String comStr = ae.getActionCommand();

**if**(comStr.equals("Exit")) System.*exit*(0);

jlab.setText(comStr + " Selected");

}

**public** **static** **void** main(String args[]) {

SwingUtilities.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** JToolBarDemo();

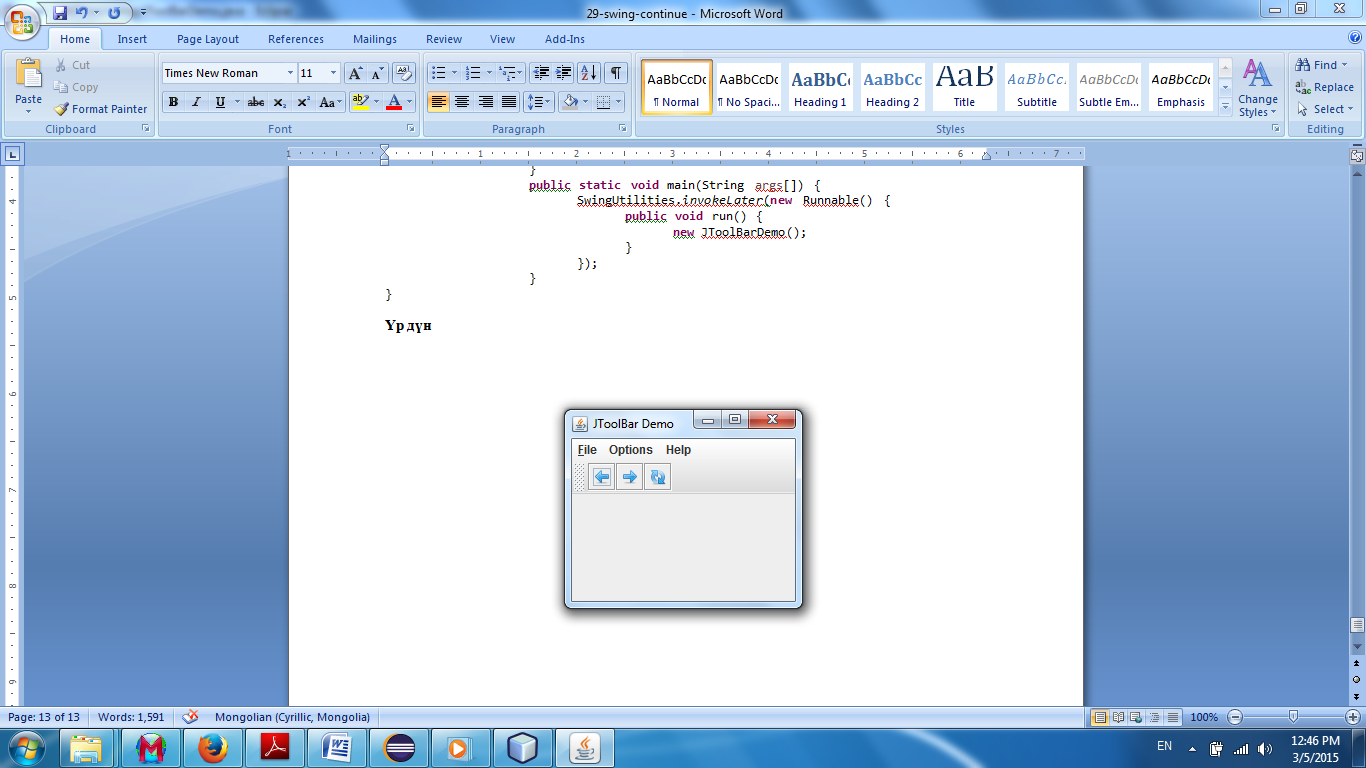
}

});

}

}

**Үр дүн**



Бүх жишээг нэгтгэв

**package** swing;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**class** MenuDemoAll **implements** ActionListener {

JLabel jlab;

JMenuBar jmb;

JToolBar jtb;

JPopupMenu jpu;

DebugAction setAct;

DebugAction clearAct;

DebugAction resumeAct;

MenuDemoAll() {

JFrame jfrm = **new** JFrame("Complete Menu Demo");

jfrm.setSize(360, 200);

jfrm.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

jlab = **new** JLabel();

jmb = **new** JMenuBar();

makeFileMenu();

makeActions();

makeToolBar();

makeOptionsMenu();

makeHelpMenu();

makeEditPUMenu();

jfrm.addMouseListener(**new** MouseAdapter() {

**public** **void** mousePressed(MouseEvent me) {

**if**(me.isPopupTrigger())

jpu.show(me.getComponent(), me.getX(), me.getY());

}

**public** **void** mouseReleased(MouseEvent me) {

**if**(me.isPopupTrigger())

jpu.show(me.getComponent(), me.getX(), me.getY());

}

});

jfrm.add(jlab, SwingConstants.***CENTER***);

jfrm.add(jtb, BorderLayout.***NORTH***);

jfrm.setJMenuBar(jmb);

jfrm.setVisible(**true**);

}

**public** **void** actionPerformed(ActionEvent ae) {

String comStr = ae.getActionCommand();

**if**(comStr.equals("Exit")) System.*exit*(0);

jlab.setText(comStr + " Selected");

}

**class** DebugAction **extends** AbstractAction {

**public** DebugAction(String name, Icon image, **int** mnem,**int** accel, String tTip) {

**super**(name, image);

putValue(***ACCELERATOR\_KEY***,

KeyStroke.*getKeyStroke*(accel,

InputEvent.***CTRL\_DOWN\_MASK***));

putValue(***MNEMONIC\_KEY***, **new** Integer(mnem));

putValue(***SHORT\_DESCRIPTION***, tTip);

}

**public** **void** actionPerformed(ActionEvent ae) {

String comStr = ae.getActionCommand();

jlab.setText(comStr + " Selected");

// Toggle the enabled status of the

// Set and Clear Breakpoint options.

**if**(comStr.equals("Set Breakpoint")) {

clearAct.setEnabled(**true**);

setAct.setEnabled(**false**);

} **else** **if**(comStr.equals("Clear Breakpoint")) {

clearAct.setEnabled(**false**);

setAct.setEnabled(**true**);

}

}

}

**void** makeFileMenu() {

JMenu jmFile = **new** JMenu("File");

jmFile.setMnemonic(KeyEvent.***VK\_F***);

JMenuItem jmiOpen = **new** JMenuItem("Open",KeyEvent.***VK\_O***);

jmiOpen.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_O***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiClose = **new** JMenuItem("Close", KeyEvent.***VK\_C***);

jmiClose.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_C***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiSave = **new** JMenuItem("Save",KeyEvent.***VK\_S***);

jmiSave.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_S***,InputEvent.***CTRL\_DOWN\_MASK***));

JMenuItem jmiExit = **new** JMenuItem("Exit",KeyEvent.***VK\_E***);

jmiExit.setAccelerator(KeyStroke.*getKeyStroke*(KeyEvent.***VK\_E***,InputEvent.***CTRL\_DOWN\_MASK***));

jmFile.add(jmiOpen);

jmFile.add(jmiClose);

jmFile.add(jmiSave);

jmFile.addSeparator();

jmFile.add(jmiExit);

jmb.add(jmFile);

jmiOpen.addActionListener(**this**);

jmiClose.addActionListener(**this**);

jmiSave.addActionListener(**this**);

jmiExit.addActionListener(**this**);

}

**void** makeOptionsMenu() {

JMenu jmOptions = **new** JMenu("Options");

JMenu jmColors = **new** JMenu("Colors");

JCheckBoxMenuItem jmiRed = **new** JCheckBoxMenuItem("Red");

JCheckBoxMenuItem jmiGreen = **new** JCheckBoxMenuItem("Green");

JCheckBoxMenuItem jmiBlue = **new** JCheckBoxMenuItem("Blue");

jmColors.add(jmiRed);

jmColors.add(jmiGreen);

jmColors.add(jmiBlue);

jmOptions.add(jmColors);

JMenu jmPriority = **new** JMenu("Priority");

JRadioButtonMenuItem jmiHigh = **new** JRadioButtonMenuItem("High", **true**);

JRadioButtonMenuItem jmiLow = **new** JRadioButtonMenuItem("Low");

jmPriority.add(jmiHigh);

jmPriority.add(jmiLow);

jmOptions.add(jmPriority);

ButtonGroup bg = **new** ButtonGroup();

bg.add(jmiHigh);

bg.add(jmiLow);

JMenu jmDebug = **new** JMenu("Debug");

JMenuItem jmiSetBP = **new** JMenuItem(setAct);

JMenuItem jmiClearBP = **new** JMenuItem(clearAct);

JMenuItem jmiResume = **new** JMenuItem(resumeAct);

jmDebug.add(jmiSetBP);

jmDebug.add(jmiClearBP);

jmDebug.add(jmiResume);

jmOptions.add(jmDebug);

JMenuItem jmiReset = **new** JMenuItem("Reset");

jmOptions.addSeparator();

jmOptions.add(jmiReset);

jmb.add(jmOptions);

jmiRed.addActionListener(**this**);

jmiGreen.addActionListener(**this**);

jmiBlue.addActionListener(**this**);

jmiHigh.addActionListener(**this**);

jmiLow.addActionListener(**this**);

jmiReset.addActionListener(**this**);

}

**void** makeHelpMenu() {

JMenu jmHelp = **new** JMenu("Help");

ImageIcon icon = **new** ImageIcon("AboutIcon.gif");

JMenuItem jmiAbout = **new** JMenuItem("About", icon);

jmiAbout.setToolTipText("Info about the MenuDemo program.");

jmHelp.add(jmiAbout);

jmb.add(jmHelp);

jmiAbout.addActionListener(**this**);

}

**void** makeActions() {

ImageIcon setIcon = **new** ImageIcon("redo.png");

ImageIcon clearIcon = **new** ImageIcon("undoe.png");

ImageIcon resumeIcon = **new** ImageIcon("refresh.png");

setAct = **new** DebugAction("Redo Breakpoint",setIcon,KeyEvent.***VK\_S***, KeyEvent.***VK\_B***, "Redo a break point.");

clearAct = **new** DebugAction("Undo Breakpoint", clearIcon, KeyEvent.***VK\_C***, KeyEvent.***VK\_L***,"Undo a break point.");

resumeAct = **new** DebugAction("Refresh",resumeIcon, KeyEvent.***VK\_R***, KeyEvent.***VK\_R***, "Refresh execution after breakpoint.");

clearAct.setEnabled(**false**);

}

**void** makeToolBar() {

JButton jbtnSet = **new** JButton(setAct);

JButton jbtnClear = **new** JButton(clearAct);

JButton jbtnResume = **new** JButton(resumeAct);

jtb = **new** JToolBar("Breakpoints");

jtb.add(jbtnSet);

jtb.add(jbtnClear);

jtb.add(jbtnResume);

}

**void** makeEditPUMenu() {

jpu = **new** JPopupMenu();

JMenuItem jmiCut = **new** JMenuItem("Cut");

JMenuItem jmiCopy = **new** JMenuItem("Copy");

JMenuItem jmiPaste = **new** JMenuItem("Paste");

jpu.add(jmiCut);

jpu.add(jmiCopy);

jpu.add(jmiPaste);

jmiCut.addActionListener(**this**);

jmiCopy.addActionListener(**this**);

jmiPaste.addActionListener(**this**);

}

**public** **static** **void** main(String args[]) {

SwingUtilities.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** MenuDemoAll();

}

});

}

}

**Үр дүн**

