**Swing –г цааш нь судлах нь**

Бид өмнө swing –тэй холбоотой ерөнхий ойлголтуудыг судалсан бол энэ хичээлээр бид swing –г цааш нь дэлгэрэнгүй судлах болно.

**JLabel, ImageIcon классын тухай**

Бид өмнө JLabel –н талаар үзсэн болохоор энэ бидэнд танил болсон гэж бодож байна. JLabel –г цонхон дээр тогтмол текст болон зургийг дүрслэхэд хэрэглэдэг.

**Жишээ**

**package** swing;

**import** javax.swing.ImageIcon;

**import** javax.swing.JApplet;

**import** javax.swing.JLabel;

**import** javax.swing.SwingUtilities;

**public** **class** JLabelDemo **extends** JApplet{

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

// Create an icon.

ImageIcon image = **new** ImageIcon("hourglass.jpeg");

// Create a label.

JLabel jl = **new** JLabel("Hourglass", image, JLabel.***CENTER***);

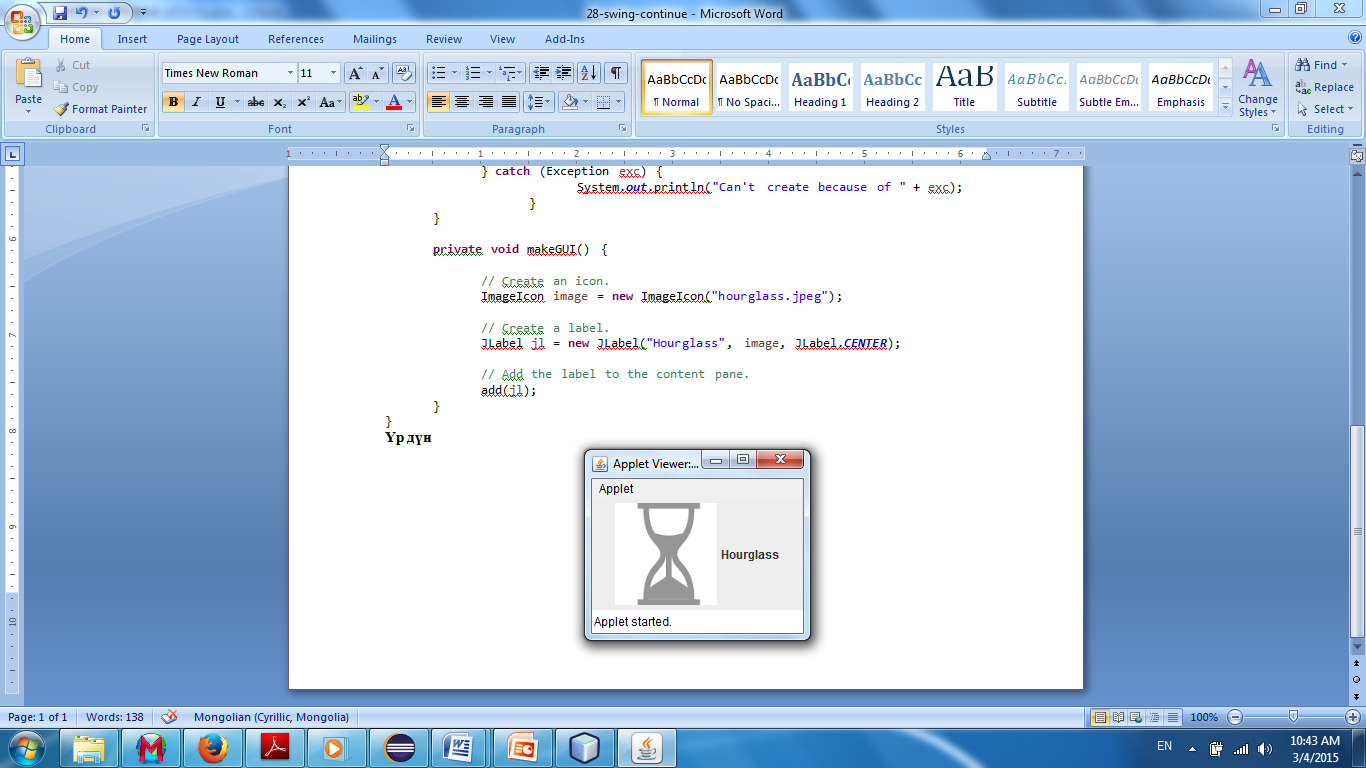
// Add the label to the content pane.

add(jl);

}

}

**Үр дүн**

****

**JTextField, JPasswordField классуудын тухай**

JTextField класс нь хэрэглэгч текст оруулж болох нэг мөр объект. JPasswordField класс нь хэрэглэгчийн оруулсан мэдээллийг нууцлах нэг мөр объект ба JTextField классаас удамшина.

**Жишээ**

**package** swing;

**import** java.awt.FlowLayout;

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

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

**public** **class** JTextFieldDemo **extends** JApplet{

JTextField jtf;

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

// Change to flow layout.

setLayout(**new** FlowLayout());

// Add text field to content pane.

jtf = **new** JTextField(15);

add(jtf);

jtf.addActionListener(**new** ActionListener() {

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

// Show text when user presses ENTER.

showStatus(jtf.getText());

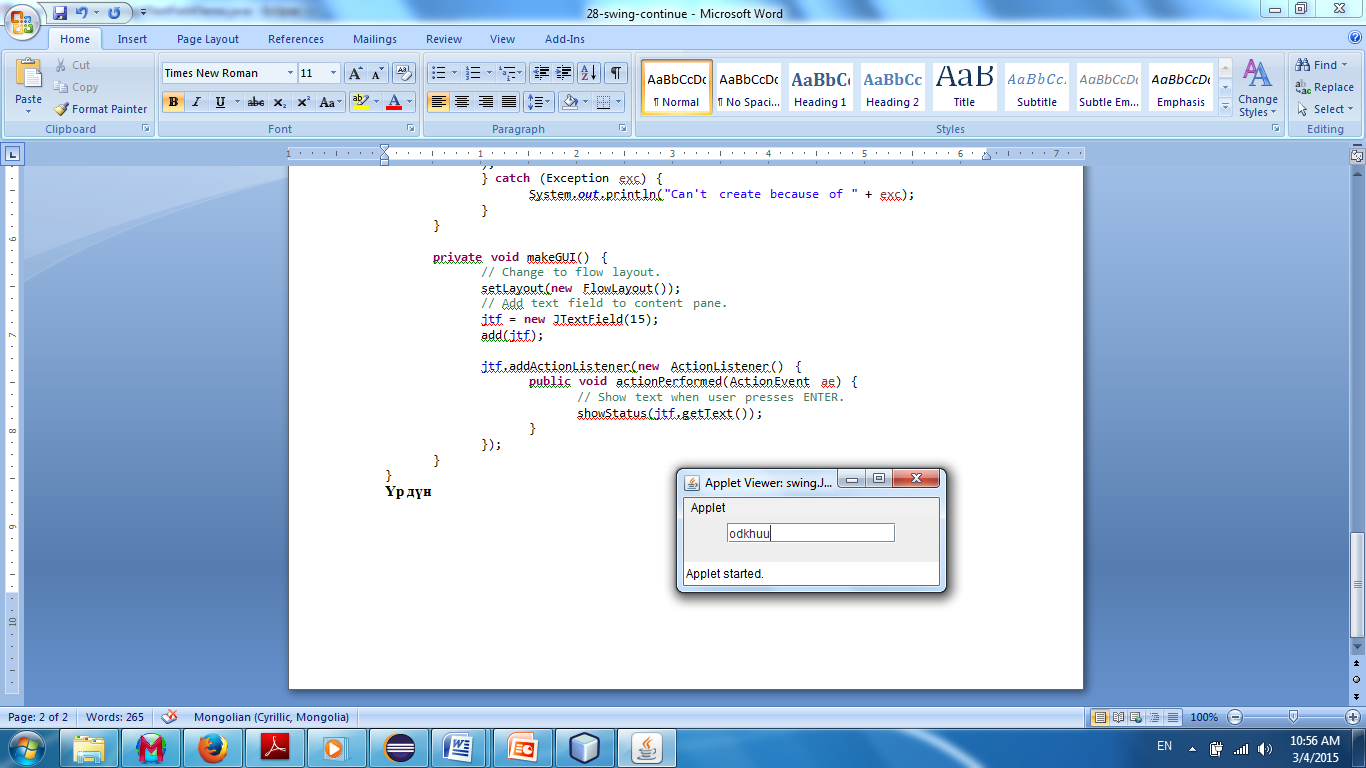
}

});

}

}

**Үр дүн**

****

**package** swing;

**import** java.awt.FlowLayout;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** javax.swing.JApplet;

**import** javax.swing.JPasswordField;

**import** javax.swing.SwingUtilities;

**public** **class** JPasswordFieldDemo **extends** JApplet{

JPasswordField jpf;

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

// Change to flow layout.

setLayout(**new** FlowLayout());

// Add text field to content pane.

jpf = **new** JPasswordField(15);

add(jpf);

jpf.addActionListener(**new** ActionListener() {

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

// Show text when user presses ENTER.

showStatus(jpf.~~getText~~());

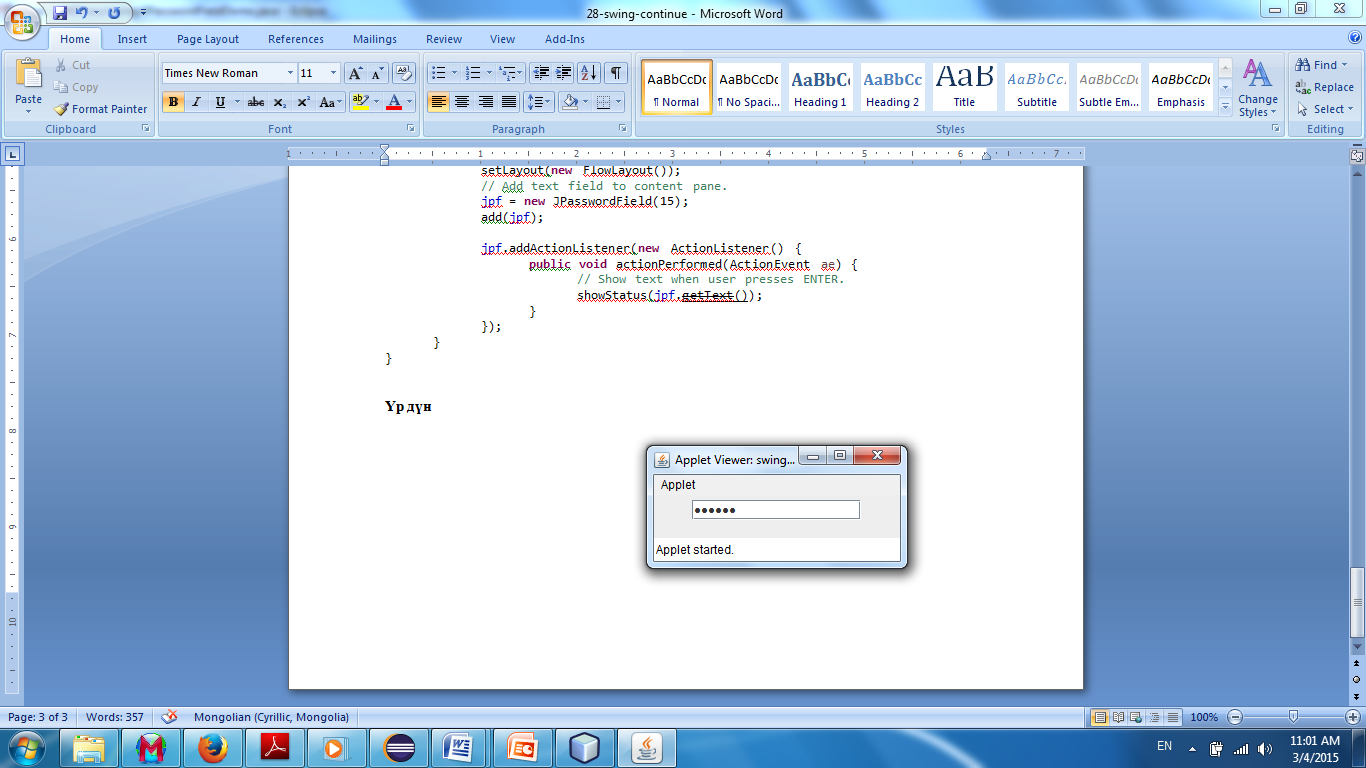
}

});

}

}

**Үр дүн**

****

**Swing –н товчнууд**

Swing нь дөрвөн төрлийн товчнуудыг тодорхойлдог: **JButton, JToggleButton, JCheckBox, JRadioButton. AbstractButton** классаас удамшсан классуудаас гардаг. Мөн дараах функцүүд байдаг: **void setDisabledIcon(Icon *di*), void setPressedIcon(Icon *pi*), void setSelectedIcon(Icon *si*) ,void setRolloverIcon(Icon *ri*).**

**JButton** класс:

Цонхон дээр товч гаргадаг класс. Дараах төрлийн байгуулагч функцүүдтэй.

**JButton(Icon *icon*)** - зурагтай товч

**JButton(String *str*)** - тэмдэгт мөртэй товч

**JButton(String *str*, Icon *icon*)** -тэмдэгт мөр болон зургатай товч

JButton класс дээр event –г барьж авахын тулд ActionListenr интерфейсийг имплемент хийн түүний actionPerformed функцийг тодорхойлж өгөх хэрэгтэй. Мөн энэ класс нь setActionCommand( ), getActionCommand( ) гэсэн функцүүдтэй.

**Жишээ**

**package** swing;

**import** java.awt.FlowLayout;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** javax.swing.ImageIcon;

**import** javax.swing.JApplet;

**import** javax.swing.JButton;

**import** javax.swing.JLabel;

**import** javax.swing.SwingUtilities;

**public** **class** JButtonDemo **extends** JApplet **implements** ActionListener{

JLabel jlab;

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

// Change to flow layout.

setLayout(**new** FlowLayout());

// Add buttons to content pane.

ImageIcon hourglass = **new** ImageIcon("hourglass.jpg");

JButton jb = **new** JButton(hourglass);

jb.setActionCommand("Hourglass");

jb.addActionListener(**this**);

add(jb);

ImageIcon analog = **new** ImageIcon("analog.jpg");

jb = **new** JButton(analog);

jb.setActionCommand("Analog Clock");

jb.addActionListener(**this**);

add(jb);

ImageIcon digital = **new** ImageIcon("digital.jpg");

jb = **new** JButton(digital);

jb.setActionCommand("Digital Clock");

jb.addActionListener(**this**);

add(jb);

ImageIcon stopwatch = **new** ImageIcon("stopwatch.jpg");

jb = **new** JButton(stopwatch);

jb.setActionCommand("Stopwatch");

jb.addActionListener(**this**);

add(jb);

// Create and add the label to content pane.

jlab = **new** JLabel("Choose a Timepiece");

add(jlab);

}

// Handle button events.

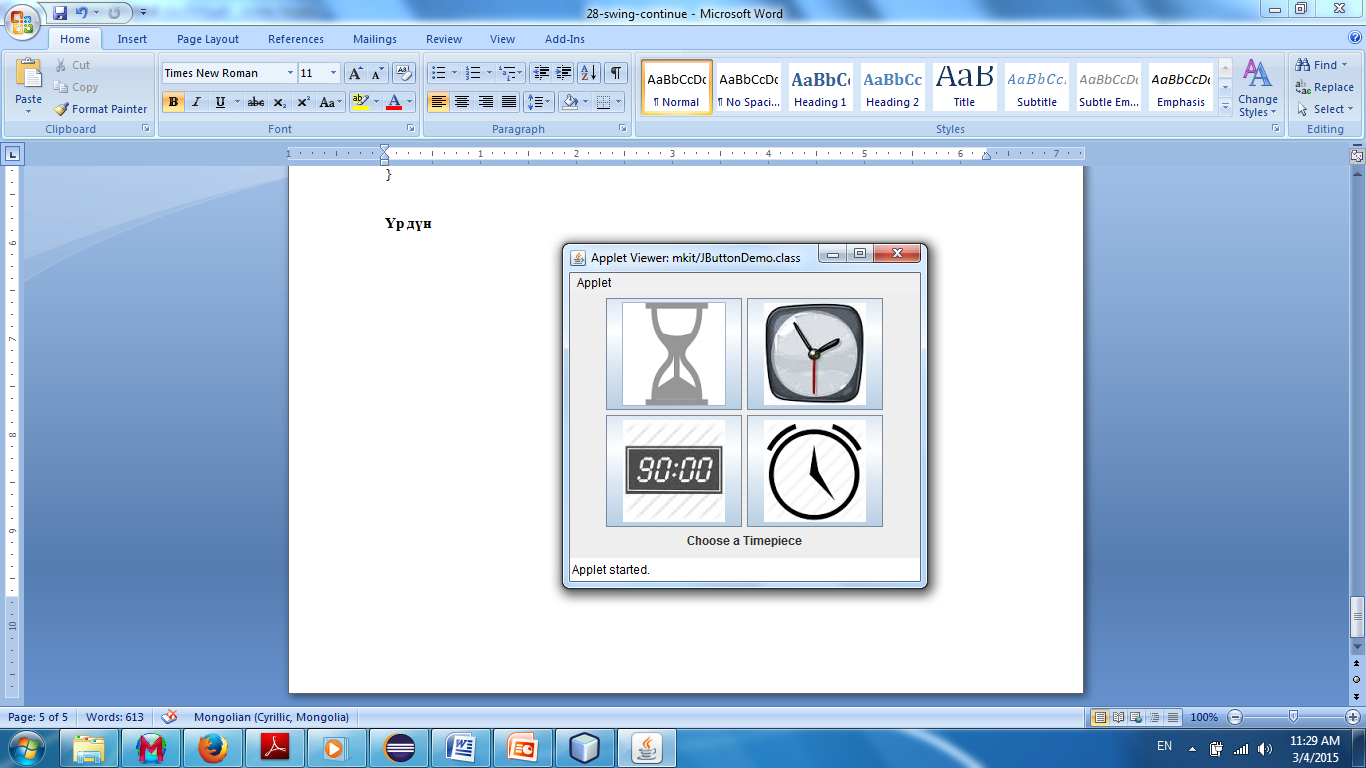
**public** **void** actionPerformed(ActionEvent ae) {

jlab.setText("You selected " + ae.getActionCommand());

}

}

**Үр дүн**

****

**JToggleButton** классын тухай

Энэ класс нь AbstractButton классаас удмашдаг ба өөрөө дотроо хоёр төрлийн JCheckBox болон JRadioButton гэсэн классуудыг агуулдаг. Тэгэхээр энэ классыг командын товч гэж хэлж болно. эдгээр дээрх event –г барьж авахын тулд ItemListener интерфейсийг имплемент хийж түүний itemStateChanged( ) функцийг тодорхойлж өгдөг. True / false гэсэн төлөвийн точв юм.

**Жишээ  
package** swing;  
**import** java.awt.\*;  
**import** java.awt.event.\*;  
**import** javax.swing.\*;  
**public** **class** JToggleButtonDemo **extends** JApplet{  
 JLabel jlab;  
 JToggleButton jtbn;  
 **public** **void** init() {  
 **try** {  
 SwingUtilities.*invokeAndWait*(  
 **new** Runnable() {  
 **public** **void** run() {  
 makeGUI();  
 }  
 });  
 } **catch** (Exception exc) {  
 System.***out***.println("Can't create because of " + exc);  
 }  
 }

**private** **void** makeGUI() {

setLayout(**new** FlowLayout());

jlab = **new** JLabel("Button is off.");

jtbn = **new** JToggleButton("On/Off");

jtbn.addItemListener(**new** ItemListener() {

**public** **void** itemStateChanged(ItemEvent ie) {

**if**(jtbn.isSelected())

jlab.setText("Button is on.");

**else**

jlab.setText("Button is off.");

}

});

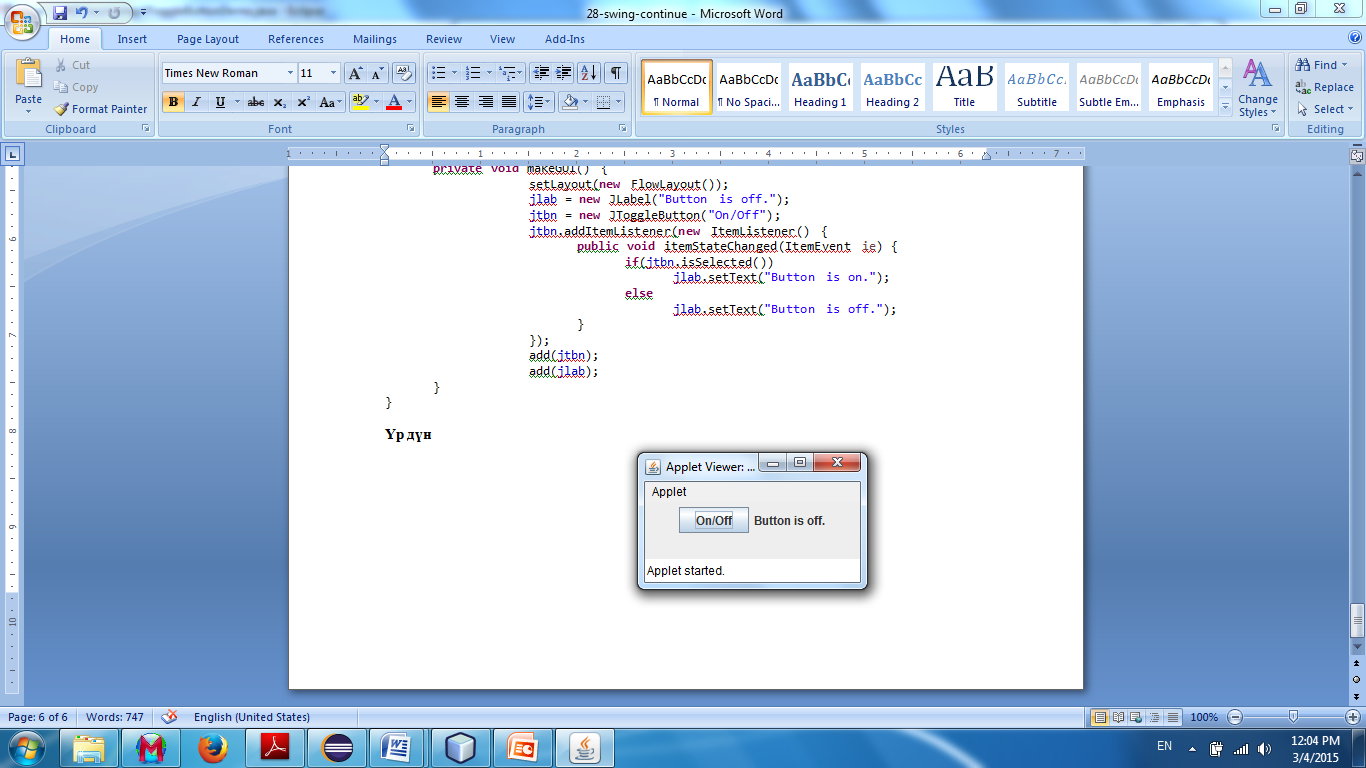
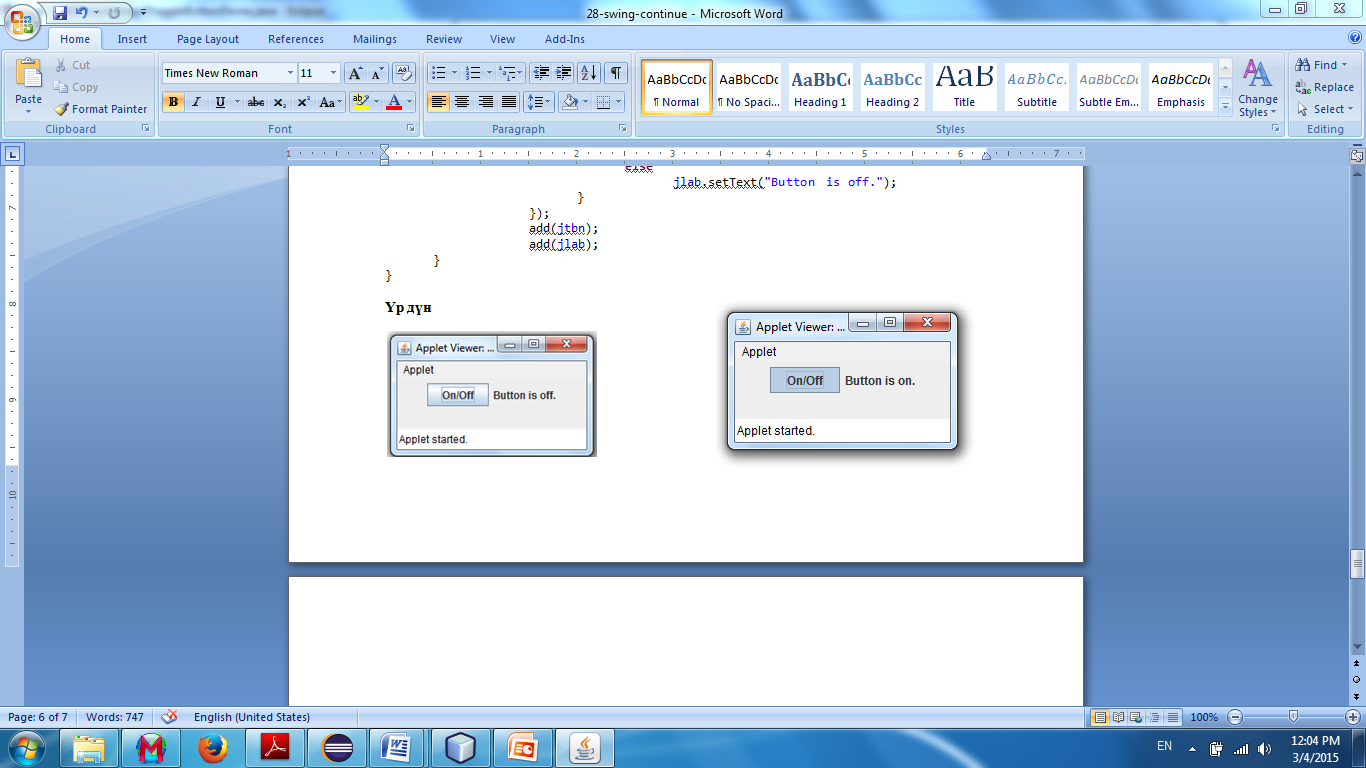
add(jtbn);

add(jlab);

}

}

**Үр дүн**

** **

**JCheckBox** классын тухай

Энэ класс нь JToggleButton классаас удамшдаг ба аливаа зүйлийг чеклэх үед хэрэглэгддэг. Энэ классын event –г барьж авхад JToggleButton класстай адилхан хийдэг.

**Жишээ**

**package** swing;

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

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

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

**public** **class** JCheckBoxDemo **extends** JApplet **implements** ItemListener{

JLabel jlab;

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

setLayout(**new** FlowLayout());

JCheckBox cb = **new** JCheckBox("C");

cb.addItemListener(**this**);

add(cb);

cb = **new** JCheckBox("C++");

cb.addItemListener(**this**);

add(cb);

cb = **new** JCheckBox("Java");

cb.addItemListener(**this**);

add(cb);

cb = **new** JCheckBox("Perl");

cb.addItemListener(**this**);

add(cb);

jlab = **new** JLabel("Select languages");

add(jlab);

}

**public** **void** itemStateChanged(ItemEvent ie) {

JCheckBox cb = (JCheckBox)ie.getItem();

**if**(cb.isSelected())

jlab.setText(cb.getText() + " is selected");

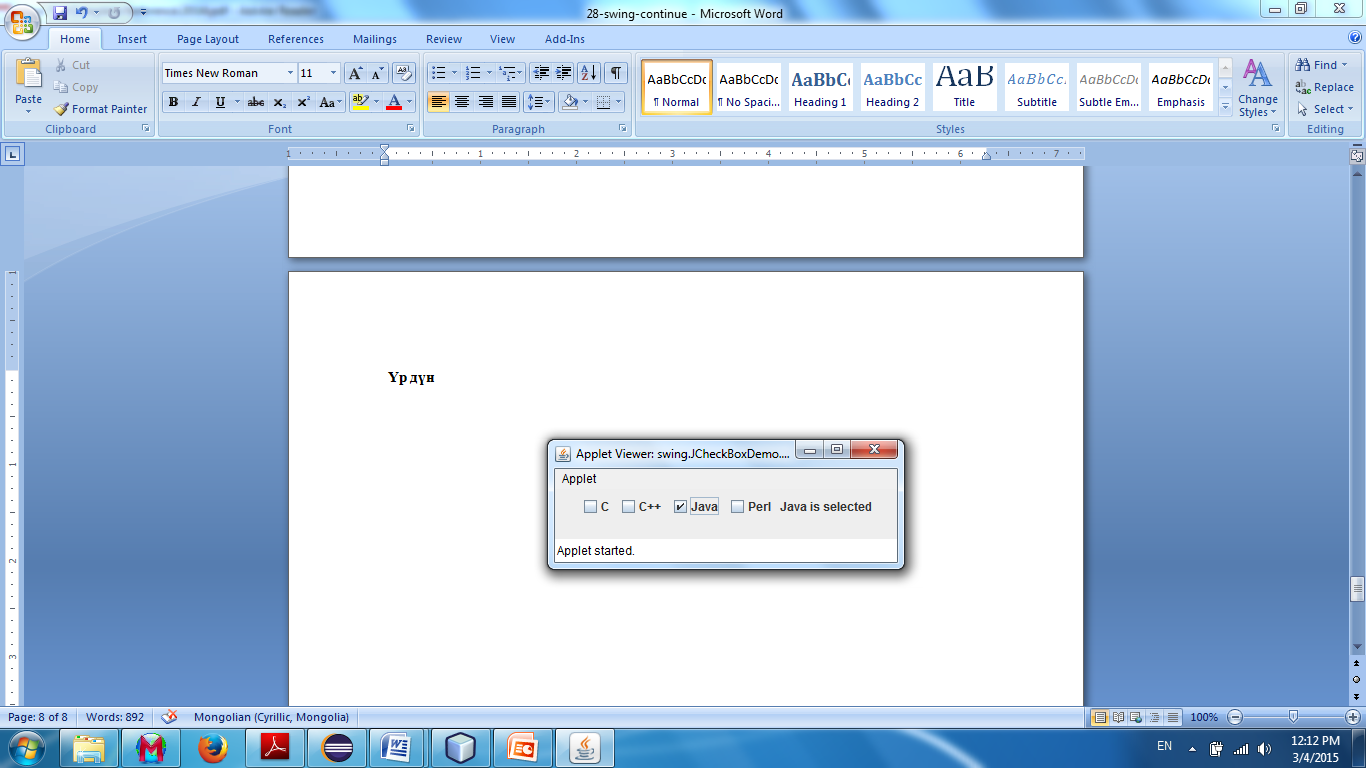
**else**

jlab.setText(cb.getText() + " is cleared");

}

}

**Үр дүн**



**JRadioButton** классын тухай

Энэ класс нь JToggleButton классаас удамшдаг ба аливаа бүлэг зүйлээс нэгийг чеклэж сонгох үед хэрэглэгддэг. Энэ классын event –г барьж авхад ActionListener интерфейсийг имплемент хийж түүний actionPerformed( ) функцийг тодорхойлох хэрэгтэй. Мөн өөрийнхөө товчнуудыг нэгтгэх зориулалттай ButtonGroup классыг ашиглан нэгтгэдэг.

**Жишээ**

**package** swing;

**import** java.awt.FlowLayout;

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

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

**public** **class** JRadioButtonDemo **extends** JApplet **implements** ActionListener{

JLabel jlab;

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

setLayout(**new** FlowLayout());

JRadioButton b1 = **new** JRadioButton("A");

b1.addActionListener(**this**);

add(b1);

JRadioButton b2 = **new** JRadioButton("B");

b2.addActionListener(**this**);

add(b2);

JRadioButton b3 = **new** JRadioButton("C");

b3.addActionListener(**this**);

add(b3);

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

bg.add(b1);

bg.add(b2);

bg.add(b3);

jlab = **new** JLabel("Select One");

add(jlab);

}

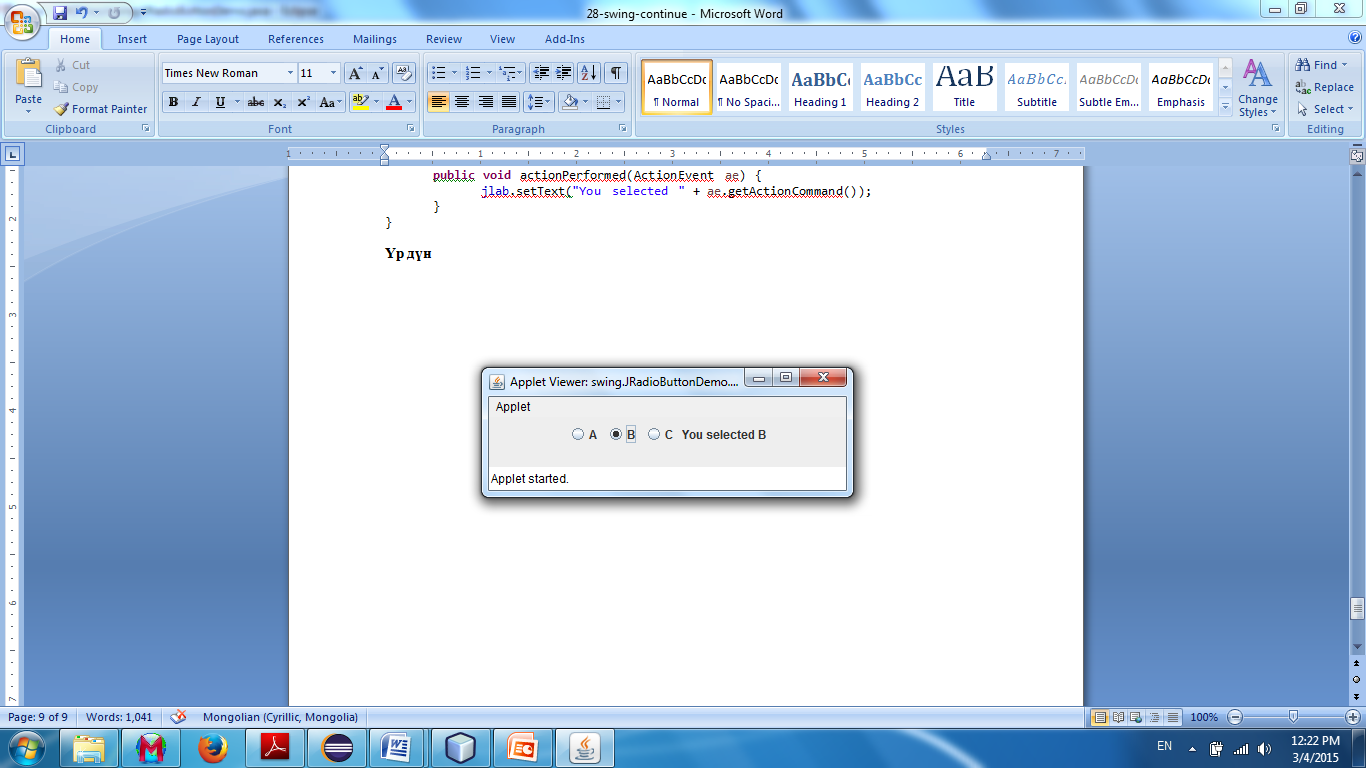
**public** **void** actionPerformed(ActionEvent ae) {

jlab.setText("You selected " + ae.getActionCommand());

}

}

**Үр дүн**



**JTabbedPane** классын тухай

Энэ класс нь component –үүдийг хэсэг хэсэгээр нь харуулна (үечилсэн бүтэц). Үе бүрийг JPanel хэлбэрээр оруулдаг. Tab ашиглан үе бүрт хандана.

**Жишээ**

**import** javax.swing.JApplet;

**import** javax.swing.JTabbedPane;

**import** javax.swing.SwingUtilities;

**public** **class** JTabbedPaneDemo **extends** JApplet{

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

JTabbedPane jtp = **new** JTabbedPane();

jtp.addTab("Cities", **new** CitiesPanel());

jtp.addTab("Colors", **new** ColorsPanel());

jtp.addTab("Flavors", **new** FlavorsPanel());

add(jtp);

}

}

**class** CitiesPanel **extends** JPanel{

**public** CitiesPanel() {

JButton b1 = **new** JButton("New York");

add(b1);

JButton b2 = **new** JButton("London");

add(b2);

JButton b3 = **new** JButton("Hong Kong");

add(b3);

JButton b4 = **new** JButton("Tokyo");

add(b4);

}

}

**class** ColorsPanel **extends** JPanel{

**public** ColorsPanel() {

JCheckBox cb1 = **new** JCheckBox("Red");

add(cb1);

JCheckBox cb2 = **new** JCheckBox("Green");

add(cb2);

JCheckBox cb3 = **new** JCheckBox("Blue");

add(cb3);

}

}

**class** FlavorsPanel **extends** JPanel{

**public** FlavorsPanel() {

JComboBox<String> jcb = **new** JComboBox<String>();

jcb.addItem("Vanilla");

jcb.addItem("Chocolate");

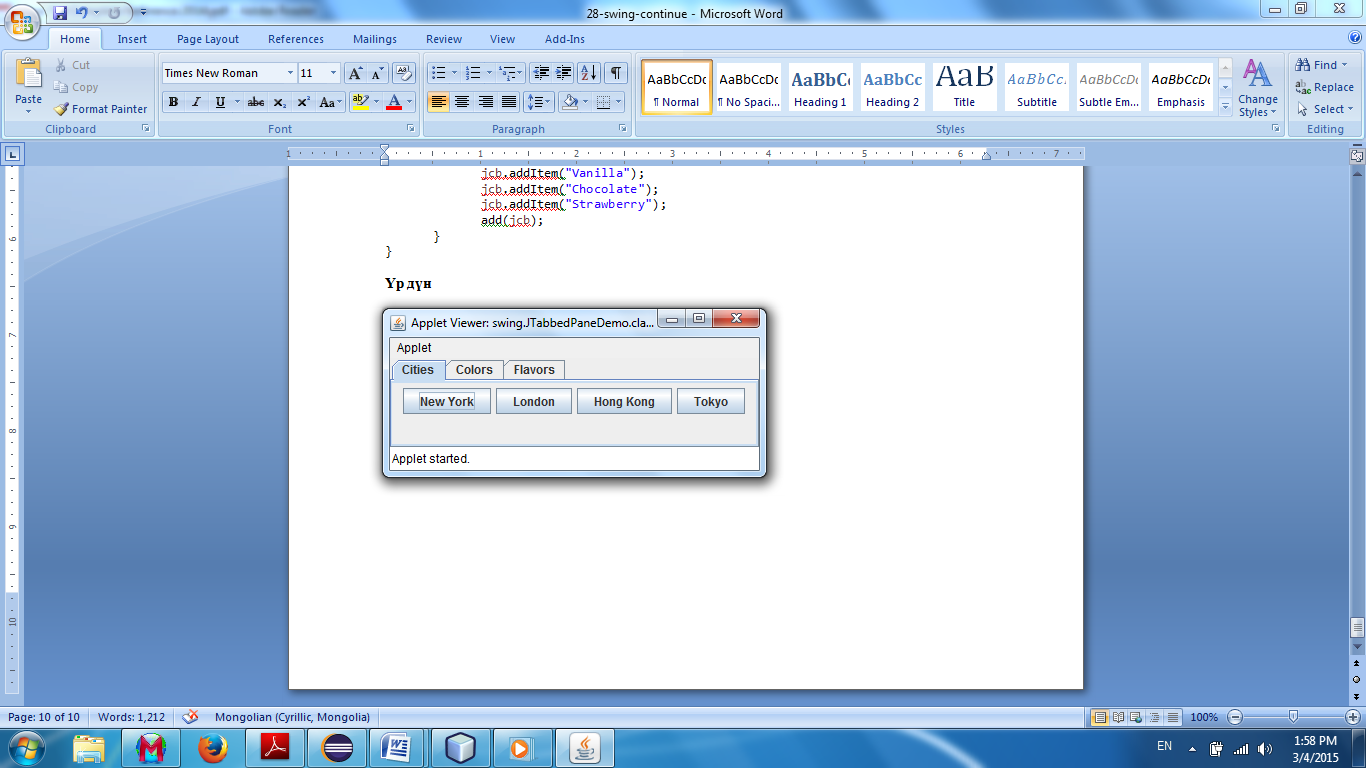
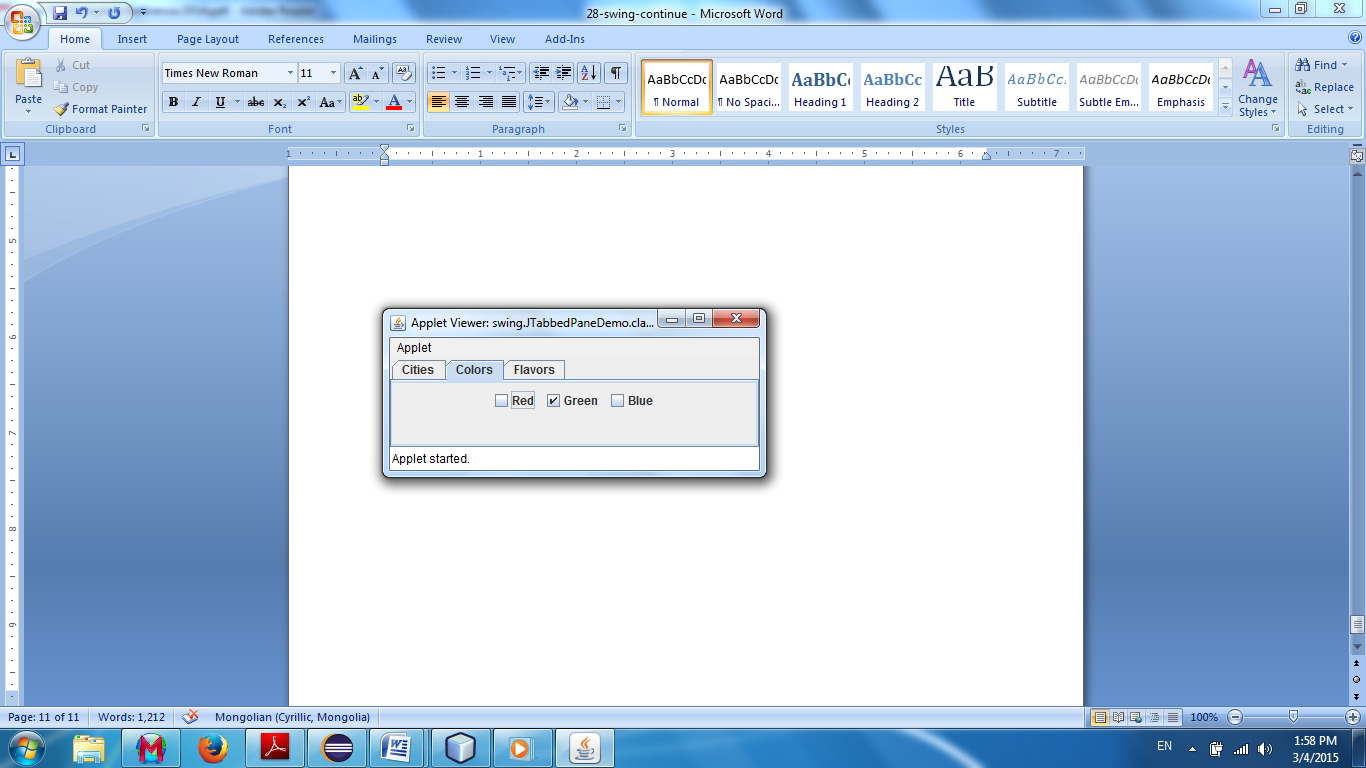
jcb.addItem("Strawberry");

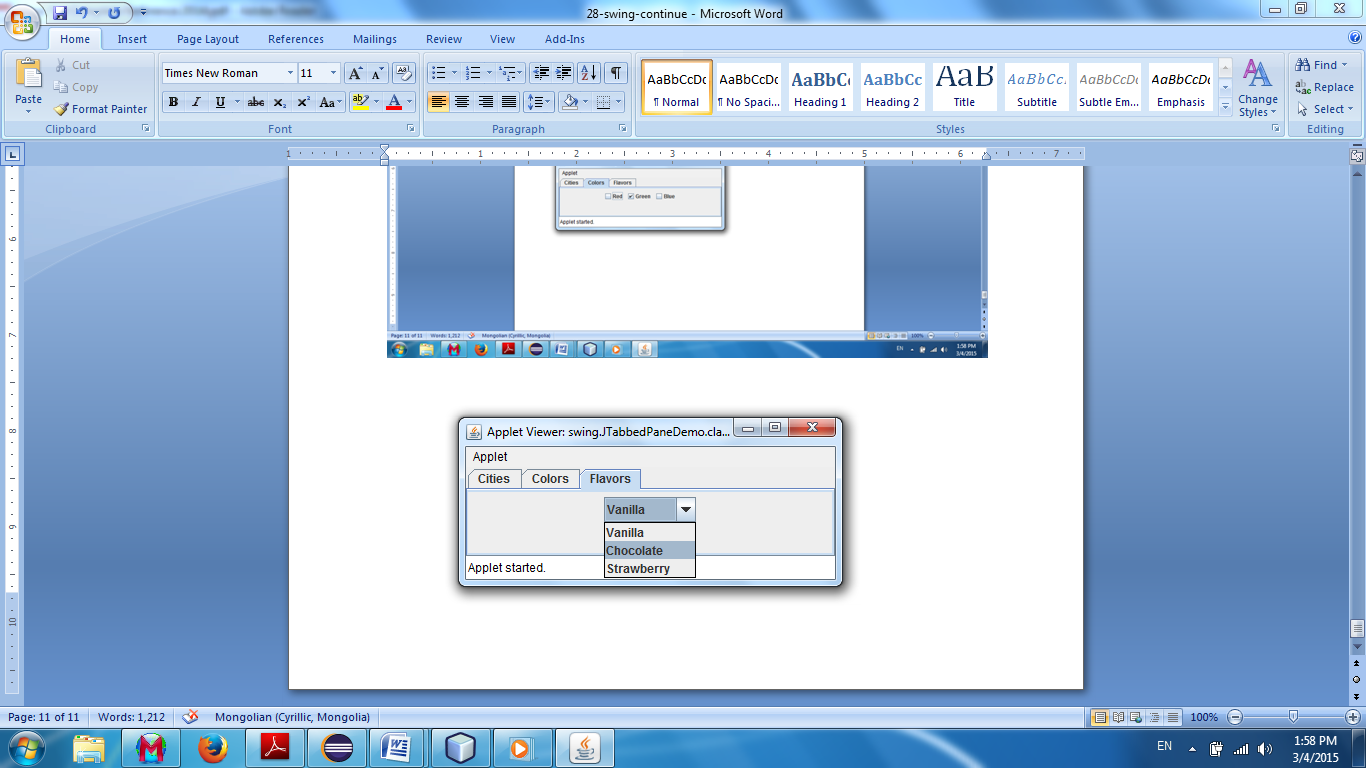
add(jcb);

}

}

**Үр дүн**



**JScrollPane** классын тухай

Энэ класс нь scroll бүхий panel юм. Swing –н ихэнх component өөрийн гэсэн scroll –гүй байдаг. Жишээ нь: JList –д scroll байдаггүй ба хэрэв листийн элементүүдийн хэмжээ нь их бол зарим элементүүд нь харагддаггүй. Иймээс эхлээд JScrollPane –д листээ хийгээд, JFrame, JPanel –руу scrollPane –ээ нэмдэг. JScrollPane –н хэмжээг өөрчилж болох боловч ихэнх тохиолдолд нь доторх агуулагдагч элементийн хэмжээгээр тодорхойлогддог. Мөн setViewportView( Component view) функцээр клиент component –г өгч болно.

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

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

**public** **class** JScrollPaneDemo **extends** JApplet{

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

// Add 400 buttons to a panel.

JPanel jp = **new** JPanel();

jp.setLayout(**new** GridLayout(20, 20));

**int** b = 0;

**for**(**int** i = 0; i < 20; i++) {

**for**(**int** j = 0; j < 20; j++) {

jp.add(**new** JButton("Button " + b));

++b;

}

}

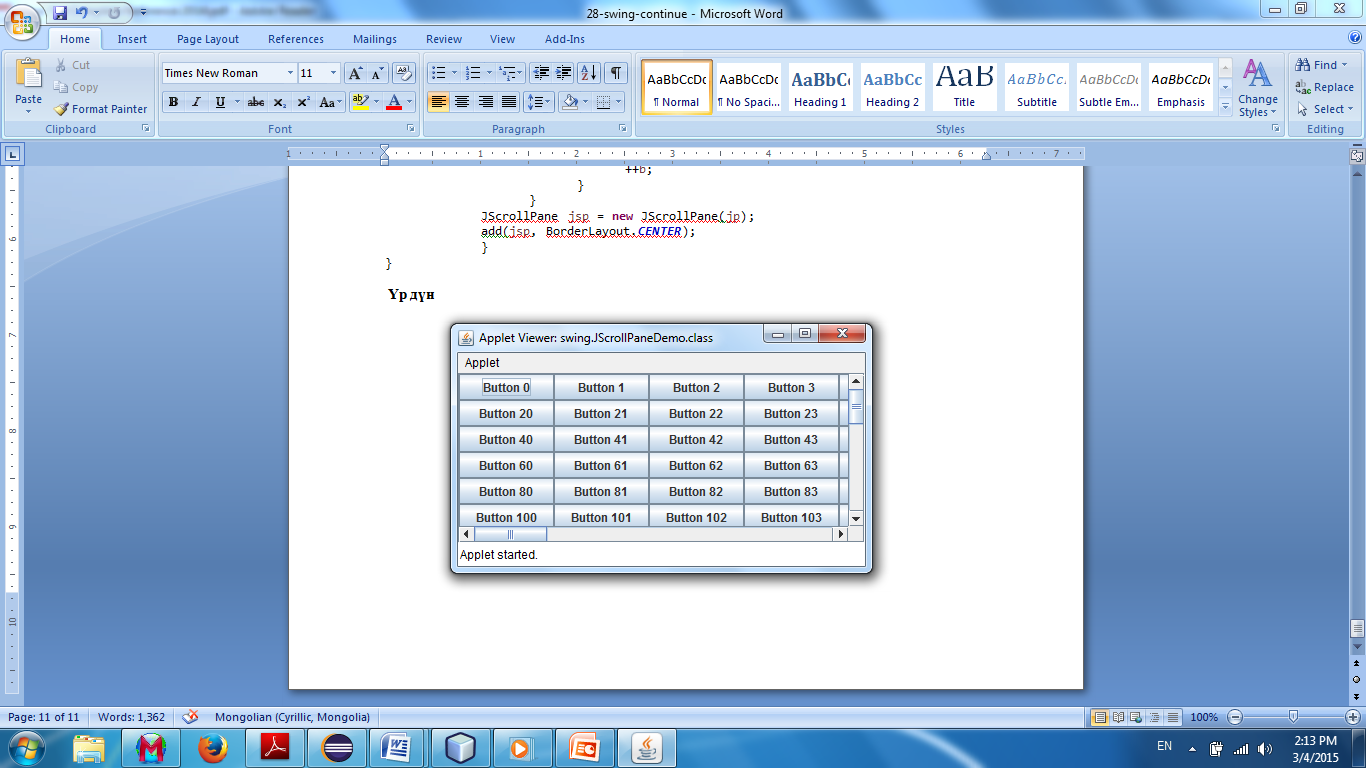
JScrollPane jsp = **new** JScrollPane(jp);

add(jsp, BorderLayout.***CENTER***);

}

}

**Үр дүн**



**JList** классын тухай

Энэ класс нь бүлэг элементүүдийг харуулах зорилоготой класс ба хэрэглэгч эдгээр элементүүдээс нь нэгийг эсвэл олныг сонгож болно. Гэхдээ сонгох төлөвөө бичихийн тулд setSelectionMode( ) функцд ListSelectionModel сонголтын төлөвүүдээс нэгийг нь сонгож аван тодорхойлно:

SINGLE\_SELECTION,

SINGLE\_INTERVAL\_SELECTION

MULTIPLE\_INTERVAL\_SELECTION

Бичигдэх хэлбэр нь: **setSelectionMode(ListSelectionModel.SINGLE\_SELECTION)**

**JDK7** хувилбараас эхлээд **JList** класс нь **generic** –г дэмждэг болсон. Харин **JList** классын **event** –г барьж авахын тулд **ListSelectionListener** интерфейсийг имплемент хийн түүний **valueChanged( )** функцийг тодорхойлох ёстой. Мөн энэ класс нь getSelectedIndex( ), getSelectedValue( ) гэсэн функцүүдтэй.

**Жишээ**

**package** swing;

**import** java.awt.Dimension;

**import** java.awt.FlowLayout;

**import** javax.swing.JApplet;

**import** javax.swing.JLabel;

**import** javax.swing.JList;

**import** javax.swing.JScrollPane;

**import** javax.swing.ListSelectionModel;

**import** javax.swing.SwingUtilities;

**import** javax.swing.event.ListSelectionEvent;

**import** javax.swing.event.ListSelectionListener;

**public** **class** JListDemo **extends** JApplet{

JList<String> jlst;

JLabel jlab;

JScrollPane jscrlp;

// Create an array of cities.

String Cities[] = { "New York", "Chicago", "Houston","Denver", "Los Angeles", "Seattle","London", "Paris", "New Delhi", "Hong Kong", "Tokyo", "Sydney" };

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

// Change to flow layout.

setLayout(**new** FlowLayout());

// Create a JList.

jlst = **new** JList<String>(Cities);

// Set the list selection mode to single selection.

jlst.setSelectionMode(ListSelectionModel.***SINGLE\_SELECTION***);

// Add the list to a scroll pane.

jscrlp = **new** JScrollPane(jlst);

// Set the preferred size of the scroll pane.

jscrlp.setPreferredSize(**new** Dimension(120, 90));

// Make a label that displays the selection.

jlab = **new** JLabel("Choose a City");

// Add selection listener for the list.

jlst.addListSelectionListener(**new** ListSelectionListener() {

**public** **void** valueChanged(ListSelectionEvent le) {

// Get the index of the changed item.

**int** idx = jlst.getSelectedIndex();

// Display selection, if item was selected.

**if**(idx != -1)

jlab.setText("Current selection: " + Cities[idx]);

**else** // Otherwise, reprompt.

jlab.setText("Choose a City");

}

});

// Add the list and label to the content pane.

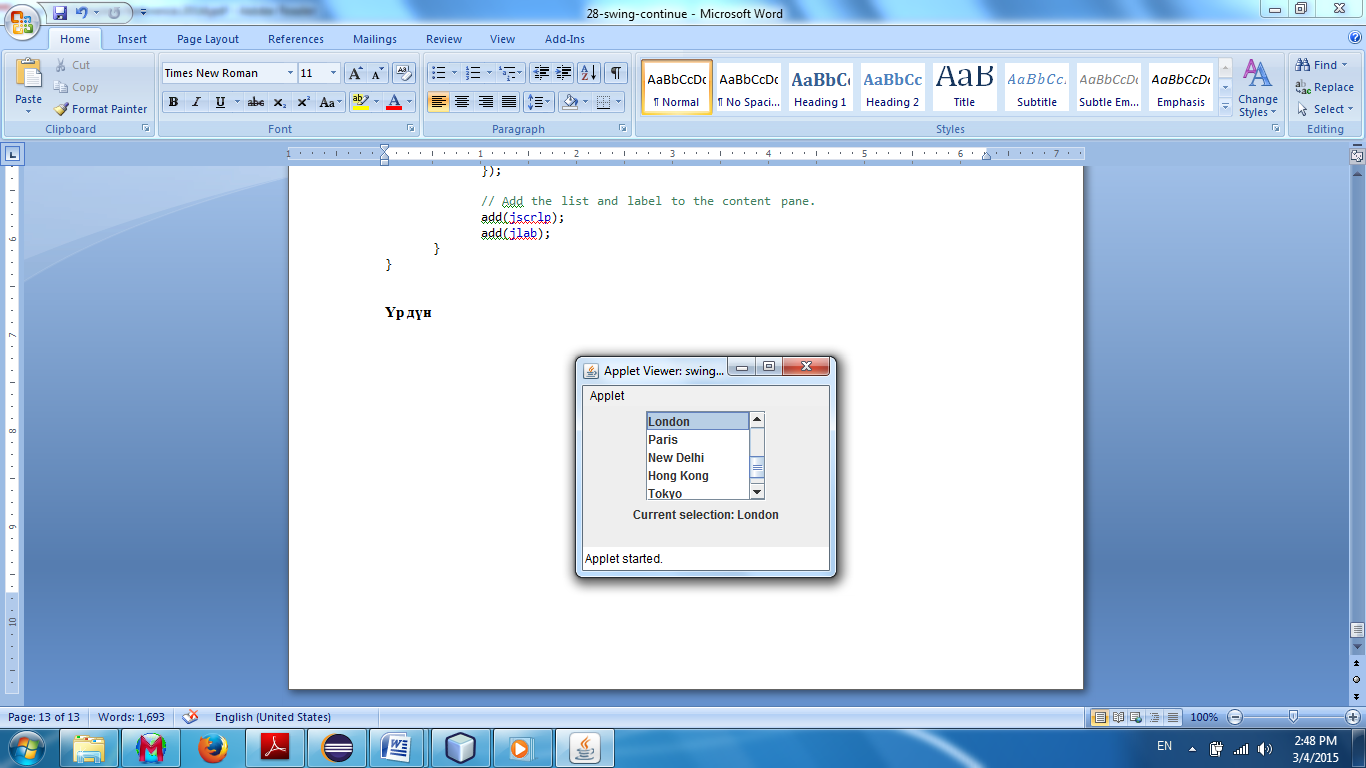
add(jscrlp);

add(jlab);

}

}

**Үр дүн**



**JComboBox** классын тухай

Энэ класс нь доторх элементүүдээс нь хэрэглэгч сонгох боломжтой ба өөрөөр бас drop-down list гэж нэрлэгддэг. JDK7 хувилбараас эхлэн generic –г дэмждэг болсон. Харин энэ классын event –г барьж авахын тулд ActionListener интерфейсийг имплемент хийн түүний actionPerformed( ) функцийг тодорхойлох ёстой. Мөн өөрийн элементүүдийг сонгож авах getSelectedItem( ) функцтэй.

**Жишээ**

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

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

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

**public** **class** JComboBoxDemo **extends** JApplet{

JLabel jlab;

ImageIcon hourglass, analog, digital, stopwatch;

JComboBox<String> jcb;

String timepieces[] = { "Hourglass", "Analog", "Digital", "Stopwatch" };

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

});

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}}

**private** **void** makeGUI() {

setLayout(**new** FlowLayout());

jcb = **new** JComboBox<String>(timepieces);

add(jcb);

jcb.addActionListener(**new** ActionListener() {

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

String s = (String) jcb.getSelectedItem();

jlab.setIcon(**new** ImageIcon(s + ".jpg"));

}

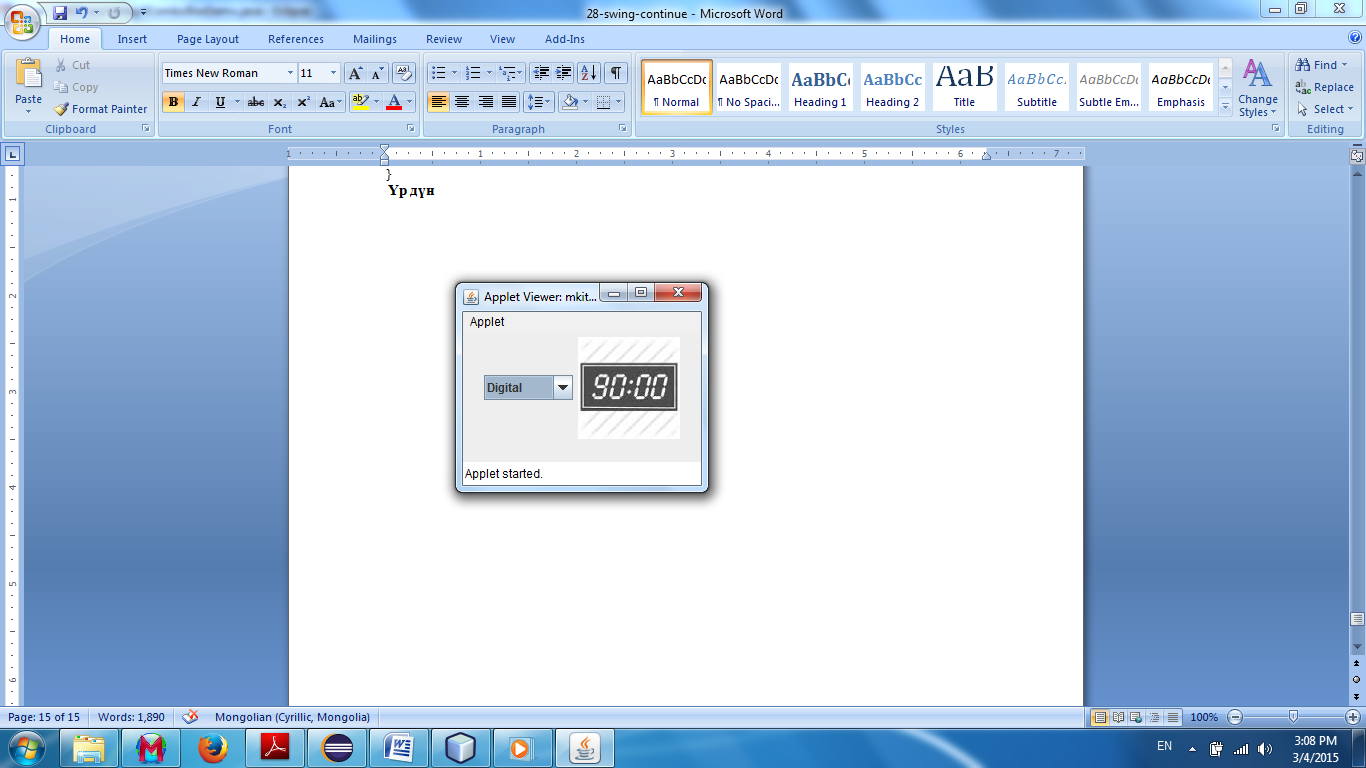
});

jlab = **new** JLabel(**new** ImageIcon("hourglass.jpg"));

add(jlab);

}}

**Үр дүн**

****

**JTree** классын тухай

Энэ класс нь өгөгдлийг мод хэлбэрээр дүрслэн үзүүлдэг. Энэ классын объект нь өгөгдлийг хадгалдаггүй харин үзүүлэгч component юм. Харин өгөгдөл нь түүний модел дотор байдаг. Тухайн классын элементүүд нь: Эх зангилаа (root), энгийн зангилаа (node), навч (leaf) дахни цааш задрахгүй зангилаа. Зангилаа үүсгэхийн тулд DefaultMutableTreeNode классыг модны зангилааг үүсгэдэг. Үүсгэсэн зангилааг JTree классын байгуулагчид дамжуулан модыг үүсгэнэ. Модыг scroll –тойгоор үүсгэх болбол уг модыг JScrollPane дотор хийж өгөх хэрэгтэй. Модон дээрх Event –г барьж авахын тулд TreeSelectionListener интерфейсийг имплемент түүний valueChanged( ) функцийг тодорхойлж өгөх хэрэгтэй.

**Жишээ**

**package** swing;

**import** java.awt.BorderLayout;

**import** javax.swing.JApplet;

**import** javax.swing.JLabel;

**import** javax.swing.JScrollPane;

**import** javax.swing.JTree;

**import** javax.swing.SwingUtilities;

**import** javax.swing.event.TreeSelectionEvent;

**import** javax.swing.event.TreeSelectionListener;

**import** javax.swing.tree.DefaultMutableTreeNode;

**public** **class** JTreeDemo **extends** JApplet{

JTree tree;

JLabel jlab;

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

// Create top node of tree.

DefaultMutableTreeNode top = **new** DefaultMutableTreeNode("Options");

// Create subtree of "A".

DefaultMutableTreeNode a = **new** DefaultMutableTreeNode("A");

top.add(a);

DefaultMutableTreeNode a1 = **new** DefaultMutableTreeNode("A1");

a.add(a1);

DefaultMutableTreeNode a2 = **new** DefaultMutableTreeNode("A2");

a.add(a2);

// Create subtree of "B"

DefaultMutableTreeNode b = **new** DefaultMutableTreeNode("B");

top.add(b);

DefaultMutableTreeNode b1 = **new** DefaultMutableTreeNode("B1");

b.add(b1);

DefaultMutableTreeNode b2 = **new** DefaultMutableTreeNode("B2");

b.add(b2);

DefaultMutableTreeNode b3 = **new** DefaultMutableTreeNode("B3");

b.add(b3);

// Create the tree.

tree = **new** JTree(top);

// Add the tree to a scroll pane.

JScrollPane jsp = **new** JScrollPane(tree);

// Add the scroll pane to the content pane.

add(jsp);

// Add the label to the content pane.

jlab = **new** JLabel();

add(jlab, BorderLayout.***SOUTH***);

// Handle tree selection events.

tree.addTreeSelectionListener(**new** TreeSelectionListener() {

**public** **void** valueChanged(TreeSelectionEvent tse) {

jlab.setText("Selection is " + tse.getPath());

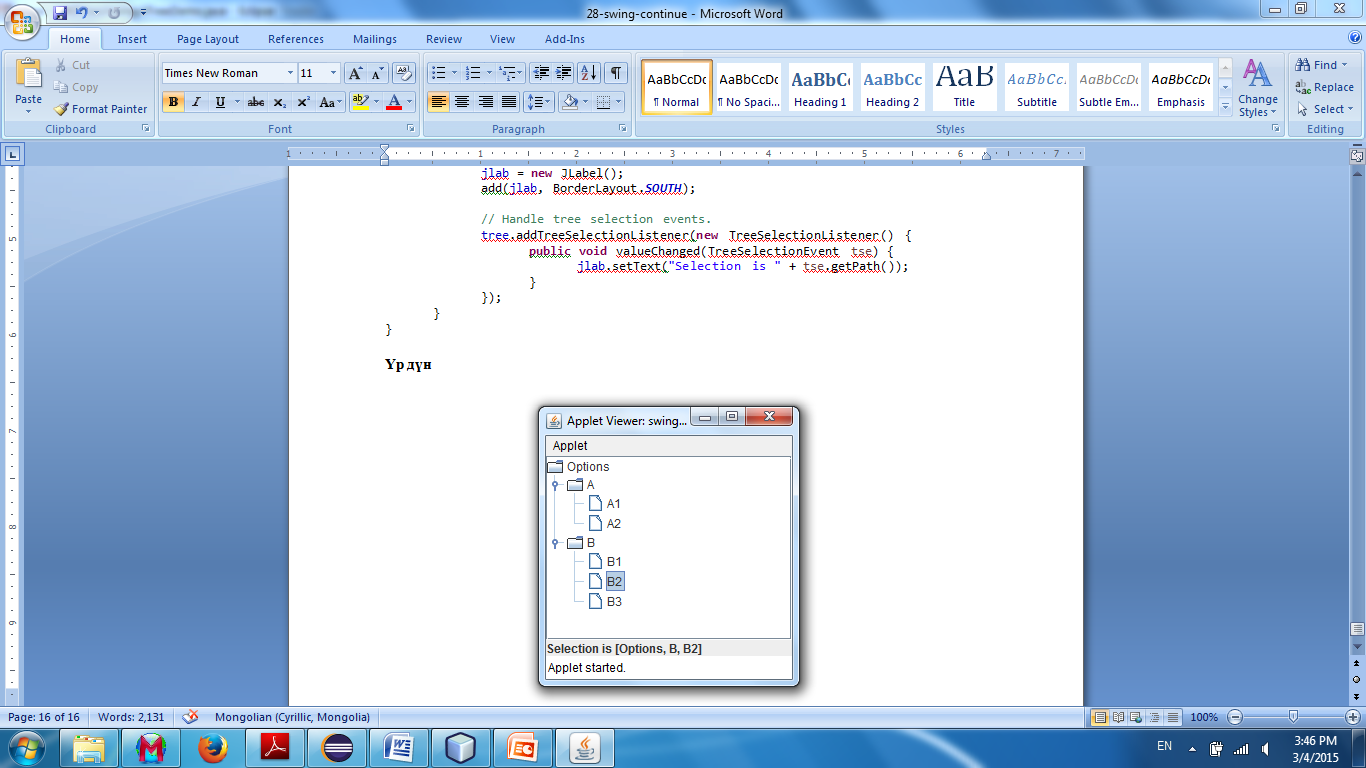
}

});

}

}

**Үр дүн**



Модны зангилаануудыг мөн хулганаар болон гараар идэвхжүүлэн сонгож болно. Сонголтын хэд хэдэн горим байдаг нь :

* TreeSelectionModel.SINGLE\_TREE\_SELECTION : Модны ганц элементийг л тэмдэглэх горим
* TreeSelectionModel.CONTIGUOUS\_TREE\_SELECTION : Дараалалсан олон элементийг сонгох боломжтой горим.
* TreeSelectionModel.DISCONTIGUOUS\_TREE\_SELECTION : Дараалалсан бус олон элемент сонгох боломж**.**

Сонголтын горимыг зааж өгөхдөө дээрх код дээр ингэж нэмж болно :

tree.getSelectionModel().setSelectionMode(TreeSelectionModel.***SINGLE\_TREE\_SELECTION***);

Модонд сонголтын листенерийг холбохдоо модын addTreeSelectionListener( ) функцийг ашиглана. Жишээ нь дээрх кодонд дараах байдлаар бичиж өгсөн байна.

tree.addTreeSelectionListener(**new** TreeSelectionListener() {

**public** **void** valueChanged(TreeSelectionEvent tse) {

jlab.setText("Selection is " + tse.getPath());

}

});

Модонд сонгогдсон байгаа хамгийн сүүлчийн элементийг авахдаа getLastSelectedPathComponent( ) функцийг ашигладаг. Модны getSelectionPath( ) функц нь модны эх зангилаанаас модонд одоо тэмдэглэгдсэн байгаа зангилаа хүртэлх замыг буцаадаг. Энэ зам нь TreePath классын обьект байдаг.

TreePath обьектийн дараах функцүүдийг ашиглаж болно :

* getLastPathComponent( ) : Уг замын хамгийн эцэст байгаа зангилааг өгнө.
* getParentPath( ) : Уг замын сүүлийн элементийг хассан замыг өгнө.
* getPath() : Энэ функц нь уг зам дахь бүх зангилаануудыг Object[ ] массивт хийн өгдөг. Энэ массивын эхний элемент нь мэдээж root зангилаа байна.
* getPathComponent(int index) : Замын заагдсан индекс дээр оршиж буй зангилааг өгнө.

Энэ замын ачаар бид сонгогдсон элемент хүртэлх дамжсан зангилаануудыг барьж авах боломжтой болж байгаа юм.

Модонд өгөгдөл оруулах, хасах, засварлахдаа модны моделийг ашиглана.

* DefaultTreeModel классыг ашиглана.
* Хэрэгтэй бол энэ классаас удамшсан шинэ моделийг хийж өгнө.

Модонд зангилаа нэмэхдээ моделийн insertNodeTo(node, parent, index) функцийг ашиглана. Үүнд :

* Node : Оруулах зангилаа (DefaultMutableTreeNode)
* Parent : Оруулах зангилааг агуулах зангилаа (DefaultMutableTreeNode)
* Index : Эх зангилааны хэддүгээр хүү зангилаа болж орохыг заана.

**JTable** классын тухай

Энэ класс нь өгөгдлийг хүснэгт хэлбэрээр гаргахад зориулсан класс юм. Энэ класс нь долоон байгуулагч функцтэй байдаг.

* JTable() : Хоосон хүснэгт үүсгэх.
* JTable(int, int) : Мөр болон баганын тоог зааж өгч үүсгэх.
* JTable(Object[ ][ ], Object[ ]) : Хүснэгтийн өгөгдлийг хоёр хэмжээст массиваар, толгойн мэдээллүүдийг нэг хэмжээст массиваар дамжуулах.
* JTable(Vector, Vector) : Мөр болон толгойн мэдээллийг Vector-р дамжуулах.
* JTable(TableModel) : Хүснэгтийн өгөгдлийг моделоор дамжуулах.
* JTable(TableModel, ColumnModel) : Хүснэгт болон баганыг моделоор дамжуулах.
* JTable(TableModel, ColumnModel, ListSelectionModel) : Хүснэгт болон баганыг моделоор мөн листенэртэйгээр дамжуулах.

**JTable –г ScrollPane –д хийж өгөхдөө:**

JScrollPane jsp = **new** JScrollPane(table);

Энэ тохиолдолд ScrollPane –г scroll хийхэд хүснэгтийн толгой нь хөдлөхгүй. ScrollPane дотор байгаа хүснэгтийг агуулагчид хийхдээ ScrollPane-г нь хийхэд л болно. Хэрэв ScrollPane гүй хүснэгт хийсэн бол хүснэгтийн толгой болон өгөгдлийг нь тус тусд нь хийж өгч байж бүтэн хүснэгт харагддаг: Тийм учираас доорх байдалтай бичих ёстой. Хүснэгт дээрх event –г барьж авахдаа **ListSelectionListener** интерфейсийг имплемент хийн түүний **valueChanged( )** функцийг тодорхойлох ёстой.

container.setLayout(**new** BorderLayout());

container.add(table.getTableHeader(), BorderLayout.***NORTH***);

container.add(table, BorderLayout.***CENTER***);

**Жишээ**

**package** swing;

**import** javax.swing.JApplet;

**import** javax.swing.JScrollPane;

**import** javax.swing.JTable;

**import** javax.swing.SwingUtilities;

**public** **class** JTableDemo **extends** JApplet{

**public** **void** init() {

**try** {

SwingUtilities.*invokeAndWait*(

**new** Runnable() {

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

makeGUI();

}

}

);

} **catch** (Exception exc) {

System.***out***.println("Can't create because of " + exc);

}

}

**private** **void** makeGUI() {

// Initialize column headings.

String[] colHeads = { "Name", "Extension", "ID#" };

// Initialize data.

Object[][] data = {

{ "Gail", "4567", "865" },

{ "Ken", "7566", "555" },

{ "Viviane", "5634", "587" },

{ "Melanie", "7345", "922" },

{ "Anne", "1237", "333" },

{ "John", "5656", "314" },

{ "Matt", "5672", "217" },

{ "Claire", "6741", "444" },

{ "Erwin", "9023", "519" },

{ "Ellen", "1134", "532" },

{ "Jennifer", "5689", "112" },

{ "Ed", "9030", "133" },

{ "Helen", "6751", "145" }

};

// Create the table.

JTable table = **new** JTable(data, colHeads);

// Add the table to a scroll pane.

JScrollPane jsp = **new** JScrollPane(table);

// Add the scroll pane to the content pane.

add(jsp);

}

}

**Үр дүн**

