0301502 ADVANCED JAVA

UNIT	MODULES	WEIGHTAGE
1	File Handling	20 %
2	Java Collection Framework	20 %
3	Event Handling, Swing and GUI Components	20 %
4	Swing, GUI Components and Layout Manager	20 %
5	Database Connectivity (JDBC)	20 %

UNIT -4 Swing, GUI Components and Layout Manager

- JtoggleButton
- JRadionButton
- JCheckBox
- JList
- JScrollBar
- JTextField
- JPasswordField
- JTextArea
- JComboBox
- JMenuItem, JMenu, JmenuBar
- LayoutManagers

UNIT – 4 JToggleButton

- The **JtoggleButton** is a concrete subclass of Abstract Button.
- It is a **superclass for JcheckBox and JradioButton classes**.
- Toggle buttons are **two state graphical components.**
- They will be in **selected or deselected state.**
- The toggle buttons generates following Events:
  - *ActionEvent*
  - *ChangeEvent*
  - *ItemEvent*

UNIT - 4 JToggleButton Constuctors

- A **JToggleButton** Constructors:
  - *JtoggleButton()*
  - *JtoggleButton(Icon i)*
  - *JtoggleButton(Icon i, boolean state)*
  - *JtoggleButton(String label)*
  - *JtoggleButton(String label, boolean state)*
  - *JtoggleButton(String label, Icon i)*
  - *JtoggleButton(String label, Icon i, boolean state)*

UNIT – 4 JToggleButton

- Examples :
  - TEST_demo7.java

UNIT – 4 JRadioButton

- JRadio Buttons are like check boxes.
- In Jradio Button **out of several options, only one will be in selected** state and all the remaining are in deselected state.
- JradioButton class is a **subclass of JtoggleButton class.**
- The JradioButton **must be placed in a button group.**
- **The Button group is created using the ButtonGroup class**.
- The JradioButtons are to be **added to the ButtonGroup using add() method.**

UNIT – 4 JRadioButton

- JradioButton generates follwoing event:
  - *ActionEvent*
  - *ItemEvent*
  - *ChangeEvent*
- If the radio buttons **are not grouped using ButtonGroup, then each radio button will be behave exactly like JcheckBox.**

UNIT – 4 JRadioButton

- A **JRadioButton** Constructors:
  - *JRadioButton()*
  - *JRadioButton(Icon icon)*
  - *JRadioButton(Icon icon, boolean selected)*
  - *JRadioButton(String text)*
  - *JRadioButton(String text, boolean selected)*
  - *JRadioButton(String text, Icon icon)*
  - *JRadioButton(String text, Icon icon, boolean selected)*

UNIT – 4 JRadioButton -Methods

Method Name	Purpose of Method
void setText(String s)	It is used to set specified text on button.
String getText()	It is used to return the text of the button.
void setEnabled(boolean b)	It is used to enable or disable the button.
void setIcon(Icon b)	It is used to set the specified Icon on the button.
Icon getIcon()	It is used to get the Icon of the button.
void setMnemonic(int a)	It is used to set the mnemonic on the button.
boolean isSelected()	It return true if button is selected.

UNIT – 4 JRadioButton

- Examples :
  - TEST_demo13.java

UNIT – 4 JCheckBox

- JcheckBox is a **subclass of JtoggleButton.**
- A check **box is a two state graphical cmponent that will be in either a select(True) or a deselected(False) state.**
- In JCheckBox **user is given an option to select any number of options out of several options given.**
- **JradioButton generates follwoing event:**
  - *ActionEvent*
  - *ItemEvent*
  - *ChangeEvent*

UNIT – 4 JCheckBox

- A **JCheckBox** Constructors:
  - *JCheckBox()*
  - *JCheckBox(Icon icon)*
  - *JCheckBox(Icon icon, boolean selected)*
  - *JCheckBox(String text)*
  - *JCheckBox(String text, boolean selected)*
  - *JCheckBox(String text, Icon icon)*
  - *JCheckBox(String text, Icon icon, boolean selected)*

UNIT – 4 JCheckBox - Methods

Method Name	Purpose of Method
void setText(String s)	It is used to set specified text on button.
String getText()	It is used to return the text of the button.
void setIcon(Icon b)	It is used to set the specified Icon on the button.
Icon getIcon()	It is used to get the Icon of the button.
Void setSelected(boolean state)	Sets the check box to the specified state.
boolean isSelected()	It return true if button is selected.

UNIT – 4 JCheckBox

- Examples :
  - TEST_demo10.java

UNIT – 4 JList

- JList is a subclass of JComponent.
- It allows user to select one or more items from the list.
- A list is used when the number of items for selection is large.
- Either strings or images canbe element of the list.
- Selection of an item is done by clicking on the item itself.
- A Swing list does not have a scroll bar to display the list. Hence, the list is to be plaed inside a **JscrollPane() object.**

UNIT – 4 JList

- A **JList** Constructors:
  - JList()
  - JList(Vector vec)
  - JList(Object[] obj)
  - JList(ListModel lm)

UNIT – 4 JList - Methods

Method Name	Purpose of Method
void setVisibleRowCount(int c)	Sets the number of rows in the list to be displayed.
void addListSelectionListener(ListSelectionListener ls)	Adds a list selection listener to this list.
int getFirstVisibleIndex()	Returns the index of the topmost item that is visible.
int getLastVisibleIndex()	Returns the index of the bottom item that is visible.
int getSelectedIndex()	Returns the index of the first seleted item.
object getSelectedValue()	Returns the topmost selected item.
object[] getSelectedValues()	Returns an array of all selected items.
int getVisibleRowCount()	Returns the number of rows visible in the Jlist.

UNIT – 4 JList

- Examples :
  - JlistExample.java
  - JlistExample_1.java
  - JlistExample_2.java

UNIT – 4 JScrollBar

- JScrollBar is a subclass of Jcomponent.
- The scroll bar are available in two orientations
  - Horizontal ---- JScrollBar.HORIZONTAL
  - Vertical -------- JScrollBar.VERTICAL

UNIT – 4 JScrollBar

- A **JScrollBar** Constructors:
  - *JScrollBar()*
  - *JScrollBar(int Orientation)*
  - *JscrollBar(int Orientation, int scollpos,int visible,int minimum, int maximum)*

UNIT – 4 JScrollBar - Methods

Method Name	Purpose of Method
Void addAdjustmentListener(Adjustment Listener al)	Adds adjustment listener to this component.
Int getBlockDecrement()	Returns the amount of scroll bar units that give the distance through which the slider moves when block decrement is clicked.
Int getBlockIncrement()	Returns the amount of scroll bar units that give
Int getMaximum()	Returns the scroll bar’s maximum value.
Int getMinimum()	Returns the scroll bar’s minimum value.
Int getOrientation()	Returns the orientation of the scroll bar.
Int getUnitIncrement(int orientation)	Returns the amount of the slider which will be incremented when the scroll bar’s increment / decrement arrow is clicked.

UNIT – 4 JScrollBar - Methods

Method Name	Purpose of Method
Int getValue()	Returns the current value of the osition of the slider
Void setMaximum(int max)	Sets the scroll bar’s maximum value in scroll bar unit to the specified value.
Void setMinimum(int min)	Sets the scroll bar’s minimum value, in scroll bar unit, to the specified value.
Void setOrientation(int orientation)	Sets the orientation of the scroll bar.
Void setUnitIncrement(int inc)	Sets the amount slider should move in scroll bar units when the incrrement/ decrement arrow is clicked.

UNIT – 4 JScrollBar

- Examples :
  - *ScrollBarExample.java*

UNIT – 4 JTextField

- Swing’s **text component deals with two types of text :**
  - Simple text of one font and one color of text
  - Styled text with multiple fonts and multiple colors.
- Simple **type texts are deal by:**
  - *JTextFiled*
  - *JPasswordField*
  - *JTextArea*
- The styled texts are handled in :
  - *JEditorPane*
  - *JTextPane*

UNIT – 4 JTextField

- Simple type texts are deal by **JtextField, JpasswordField and JtextArea classes**
- **JtextField** is a subclass of **JTextComponent,** which is a subclass of **JComponent.**
- **JtextField can display one line of editable text** of one font and color at a time.
- The object of a JTextField class is a text component that allows the editing of a single line text.

UNIT – 4 JTextField

- Alignment can set using:
  - *JTextField.LEFT*
  - *JTextField.CENTER*
  - *JTextField.RIGHT*

UNIT – 4 JTextField

- A **JTextField** Constructors:
  - *JTextField()*
  - *JtextField(String s)*
  - *JtextField(int c)*
  - *JtextField(String s ,int c)*

UNIT – 4 JTextField - Methods

Method Name	Purpose of Method
Void addActionListener(ActionLister al)	Adds the action listeer to receive action events from this text field
Int getColumns()	Returns the number of columns set for this text field
Void removeActionListener(ActionListener al)	Remove the action listener from this text field
Void setColumns(int columns)	Sets the specified number of columns for this text field
Void setText(String text)	Sets the specified text as the text for this text filed

UNIT – 4 JTextField - Methods

Method Name	Purpose of Method
String getText()	Returns the text contained in this text field
String getSelectedText()	Returns the selectted text contained in this text field
Void setEditable(boolean edit)	Sets the text filed to editable or note editable

UNIT – 4 JTextFiled

- Examples :
  - *Demo1t.java*
  - *Demo2t.java*

UNIT – 4 JPassword Filed

- JpasswordField creates a display for text field similar to JtextFiled. The difference is that when text is displayed, the actual characters are replaced by * Characters.
- JPasswordField is a subclass of JTextComponent
- The object of a JPasswordField class is a text component specialized for password entry. It allows the editing of a single line of text. It inherits JTextField class.

UNIT – 4 JPassword Filed

- A **Jpassword Field** Constructors:
  - *public JPasswordField ()*
  - *public JPasswordField(String text)*
  - *public JPasswordField(int columns)*
  - *public JPasswordField(String text, int columns)*

UNIT – 4 JPassword Filed - Methods

Method Name	Purpose of Method
Boolean echoCharIsSet()	Returns true if an echo character has been set
Char getEchoChar()	Returns the echo character set for this field
Void setEchoChar(char c)	Sets the specified character as echo character for this field

UNIT – 4 JPassword Filed

- Examples :
  - *Demo1p.java*

UNIT – 4 JTextArea

- JTextArea is a **subclass of JTextComponent.**
- JTextArea **component displays multiple lines** of text in one color and with one font.
- There is **no scroll bar to view the text.**
- If the text is large, then a **JscrollPane has to be created using the text area component.**

UNIT – 4 JTextArea

- A **JTextArea** Constructors:
  - *Public JTextArea()*
  - *public JTextArea(int row, int column)*
  - *public JTextArea(String text, int row, int column)*

UNIT – 4 JtextArea

Field

- *static int SCROLLBARS_BOTH*
  - Create and display both vertical and horizontal scrollbars.
- *static int SCROLLBARS_HORIZONTAL_ONLY*
  - Create and display horizontal scrollbar only.
- *static int SCROLLBARS_NONE*
  - Do not create or display any scrollbars for the text area.
- *static int SCROLLBARS_VERTICAL_ONLY*
  - Create and display vertical scrollbar only.

UNIT – 4 JtextArea - Methods

Method Name	Purpose of Method
Void append(String text)	Adds the specified text at the end of the text area.
Void copy()	Copies the selected text into the system clipboard.
Void cut()	Cuts the selected textinto the system clipboard.
Int getCaretPosition()	Returns the current caret(cursor) position inside the text area.
Int getColumn()	Returns the number of columns set for the text area.
Int getLineCount()	Return the number of lines in the text area.

UNIT – 4 JtextArea - Methods

Method Name	Purpose of Method
Boolean getLineWrap()	Returns true if line wrap has been set for the text area.
Int getRows()	Returns the numbers of rows set for the text area
String getSelectedText()	Returns the selected text.
Int getSelectionEnd()	Returns the index next to the last charaacter of the seleted text.
Int getSelectionStart()	Returns the index of the first character of the selected text.
String getText()	Returns the entire text of the text area.

UNIT – 4 JtextArea - Methods

Method Name	Purpose of Method
Void insert(String str, int pos)	Inserts the specified string at the specified position pos.
Boolean isEditable()	Returns the boolean indicating whetheer the text area is editable or not
Void setLineWrap(boolean b)	Sets the line wrap for the text area as specified by the boolean
Void paste()	Pastes the string from system clipboard at the current cursor position.
Void replaceSelection(String str)	Replaces the seected text with the specified string.
Void setEditable(boolean b)	Sets the text to the editable or non editable mode

UNIT – 4 JtextArea - Methods

Method Name	Purpose of Method
Void setSeletionEnd(int end)	Sets the index at which the selection should end
Void setSelectionStart(int start)	Sets the index at which the selection should start
Void setText(String text)	Setss the text for the text area.

UNIT – 4 JtextArea

- Examples :
  - *Demota.java*
  - *demo_txtarea.java*

UNIT – 4 JComboBox

- JComboBox is a **subclass of JComponent.**
- It is a **combination of JList & JtextField.**
- In JcomboBox, only **one item is visible at a time.**
- In Combox, **the items can be edited by setting the JComboBox editable.**
- JcomboBox **has a model view structure.**
- It has **DefaultComboBoxModel class**, which can be add more flexible methods to JComboBox.

UNIT – 4 JComboBox

- Constructors:
  - *JComboBox()*
  - *JComboBox(ComboBoxModel model)*
  - *JComboBox(Object[] arry)*
  - *JcomboBox(Vector v)*

UNIT – 4 JComboBox - Methods

Method Name	Purpose of Method
Void actionPerformed(ActionEvent e)	This method is to be implemented when addActionListener isused.
Void addActionListener(ActionListener al)	Adds an action listener to the combo box.
Vod addItem(Object obj)	Adds the specified object to the list.
Void addItemListener(ItemListener il)	Adds an item listener to the combo box.
String getActioncommand()	Returns the action command.
Obeject getItemAt(int index)	Returns item at the specified index in the list.

UNIT – 4 JComboBox - Methods

Method Name	Purpose of Method
Int getItemCount()	Returns the number of items in the list.
Int getSelectedIndex()	Returns the number of items in the list.
Object getSelectedItem()	Returns the currently selected item.
Boolean isEditable()	Returns a boolean specifying whether the combobox items are editable or not.
Void removeAllItems()	Removes all items from the list
Void removeItem(Object obj)	Removes the specified item from the list.

UNIT – 4 JComboBox - Methods

Method Name	Purpose of Method
Void removeItemAt(int index)	Removes the item at the specified index from the list.
Void setActionCommand(String cmd)	Sets the specified string as the action command for the combo box.
Void setEditable(boolean edit)	Sets the boolean value indicating whether the items in the list are editable or not.
Void setEditor(ComboBoxEditor editor)	Sets an editor for the combo box
Void setModel(ComboBoxModel model)	Sets a model for the combo box.

UNIT – 4 JComboBox

- Examples :
  - *DemoCombo.java*

UNIT – 4 JMenuItem, JMenu & JMenuBar

- Menu can be created using
  - *JMenuItem*
  - *JMenu*
  - *JMenuBar*
- JMenu is subclass of JComponent & JmenuItem.
- JMenuBar is subclass of JComponent.
- JMenu contains several JmenuItem.
- A JMenuItem is like Button.

UNIT – 4 JmenuItem, Jmenu & JMenuBar

- ***A JMenuItem is to be attached to a JMenu object.***
- ***A JMenu is to be attached to a JMenuBar***
- ***A JMenuBar is to be attached to a JFrame***

UNIT – 4 JmenuItem, Jmenu & JMenuBar

- To create a menu window, the following steps are to be followed:
  - **Create JMenuItem (Many)**
  - **Create JMenu (Many)**
  - **Create JMenuBar**
  - **Create JFrame**
  - **Add all JMenuItem to JMenu**
  - **Add all JMenu to JMenuBar**
  - **Add JMenuBar to JFrame**

UNIT – 4 JMenuItem

- A JmenuItem is a part of Jmenu. When a **Jmenu is clicked, a popup menu appears, displaying all menu items contained in it.**
- Each **menu item acts like a button**. Clicking a menu iteam, **action can be initiated.**
- Constructor:
  - *JMenuItem()*
  - *JMenuItem(Icon img)*
  - *JMenuItem(String str)*
  - *JMenuItem(String text, Icon img)*

UNIT – 4 JMenuItem - Methods

Method Name	Purpose of Method
Void addActionListener(ActionListener al)	Adds action listener to the menu item.
Void addMenuDragMouseListener(MenuDragMouseListener mdl)	Adds menu drag mouse listener to this menu item.
String getActionCommand()	Returns the action command set for this menu item.
Icon getIcon()	Returns the icon of this menu item.
String getText()	Returns text of this menu item.

UNIT – 4 JMenuItem - Methods

Method Name	Purpose of Method
Void setActionCommand(String cmd)	Sets the specified string as action command.
Void setIcon(Icon img)	Sets the icon for this menu item.
Void setRolloverIcon(Icon img)	Sets the roll over icon for this menu item.

UNIT – 4 JMenu

- JMenu is the **container of JMenuItem.** A Menu c**an be attached with several menu items.**
- When a **menu is clicked, a popup menu displays all the menu items.**
- **Several such menu can be attached to MenuBa**r
- JMenu also **contain JSeparator**, which displays a visual line separator between two menu items.
- Constructor :
  - *JMenu()*
  - *JMenu(String str)*
  - *JMenu(String text, boolean tearoff)*

UNIT – 4 JMenu - Methods

Method Name	Purpose of Method
Component add(Component com)	Append a component to the end of the menu.
Component add(Component com, int index)	Inserts the specified component at the specified location in the menu.
JmenuItem add(JMenuItem mitem)	Adds the specified menu item at the end of the menu.
JmenuItem add(String str)	Create a new menu item with the specified string and appends it to the end of the menu.
Void addSeparator()	Appends a separator to the menu.

UNIT – 4 JMenu - Methods

Method Name	Purpose of Method
Int getItemCount()	Returns the total number of items, including the separator, in the menu.
JmenuItem insert(JMenuItem mitem, int index)	Inserts the specified menu item at the specified index.
Void addActionListener(ActionListener al)	Adds action listener.

UNIT – 4 JMenuBar

- JmenuBar is a subclass of Jcomponent.
- A menu bar holds many menus in its bar.
- A menu bar to be attached to a Jframe window.
- Constructor:
  - *JMenuBar()*

UNIT – 4 JMenuBar - Methods

Method Name	Purpose of Method
JMenu add(JMenu jm)	Adds specified menu to the menu bar.
JMenu getHelpMenu()	Returns the help menu.
JMenu getMenu(int index)	Returns themenu at the specified location.
Int getMenuCount()	Returns the number of items in the menu bar.
Void setHelpMenu(JMenu jm)	Sets a help in the menu bar.

UNIT – 4 JMenuBar

- Examples :
  - DemoMenu.java

UNIT – 4 Layout Manager

- When you add more than one or two components to a Jframe, Japplet or any other container, you can spend a lot of time computing exactly where to place each component.
- An alternative is to use a layout manager.
- A Layout manager is an object **control the size and position** of components **inside a container object.**
- For **placing the component in a container, layout mangers are used. Each container has a sefault layout manager.**
- **Layout manager that you asssign to the window determines how the components are sized and positioned within the window.**

UNIT – 4 Layout Manager

- Java define **several layout managers.**
- Layout manager are interface classes that are part of the Java SDK, they align the component so they neither crowd each other nor overlap.
- Each layout manager arranges componets in equally spaced columns and rows.

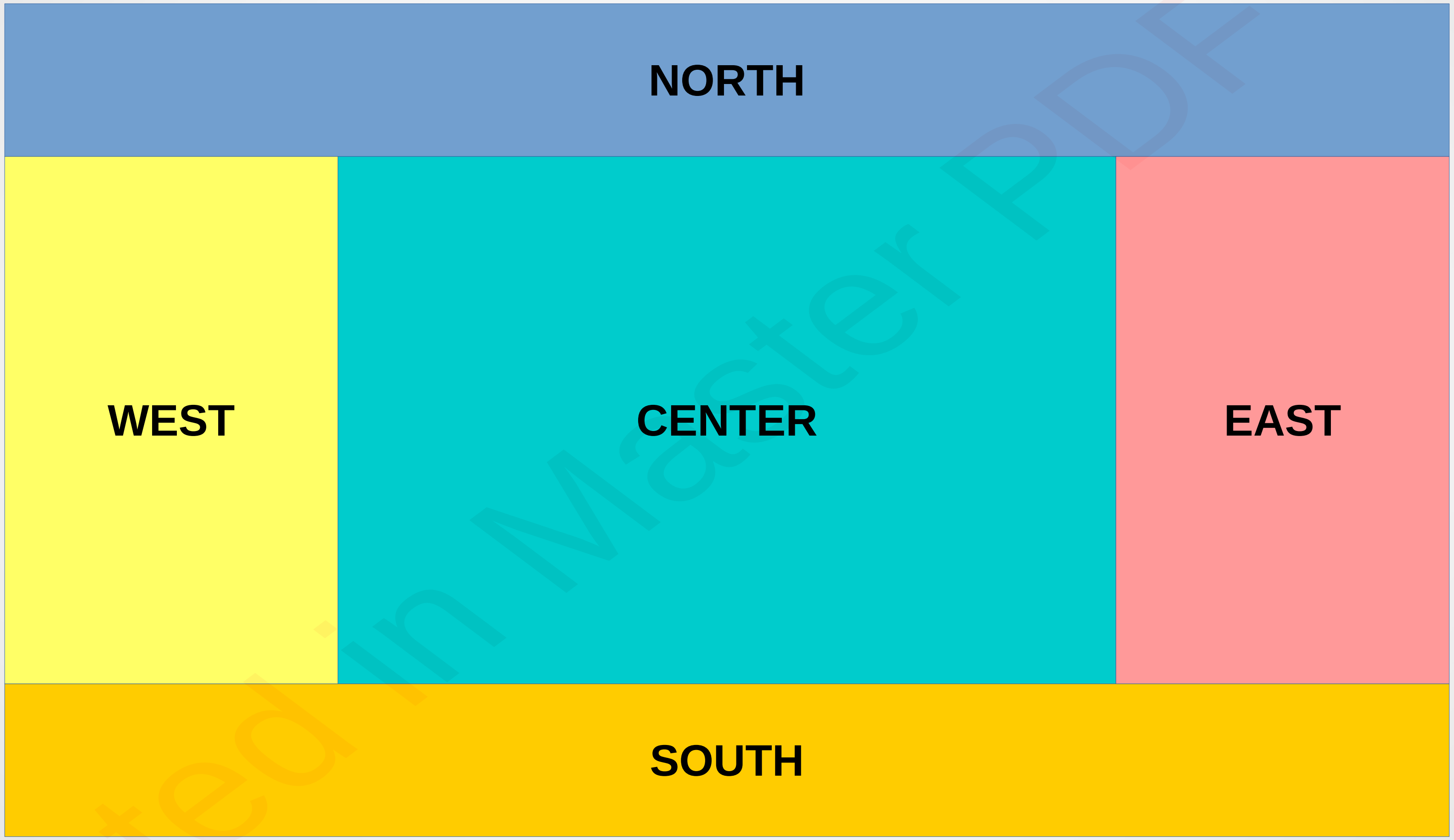
UNIT – 4 Layout Manager

Layout Manager	**When to Use**
Border Layout	Use when you add components to a maximum of five sections arranged in north, south, east, west and center positions.
Flow Layout	Use when you need to add components from left to right.
Grid Layout	Use when you need to add components into a grid fo row s and columns.
Card Layout	Use when you need to add components that are displayed one at a time.
Box Layout	Use when you need to add components into a single row or a single column
GridBag Layout	Use when you need to set size, placement and alignment constraints for every component that you add.

UNIT – 4 Border Layout Manager

- Border Layout class has a layout manager that divides its container into five regions and fits the component into it.
- The five regions are :
  - North
  - South
  - East
  - West
  - Center

UNIT - 2 Adding Multipale Components



UNIT – 4 Border Layout Manager

- To add the componet – add() method uses two argument – the component and the region to which the component is added.
- The region are
  - *BorderLayout.NORTH*
  - *BorderLayout.SOUTH*
  - *BorderLayout.WEST*
  - *BorderLayout.EAST*
  - *BorderLayout.CENTER*

UNIT – 4 Border Layout Manager

- Constructor:
  - *BorderLayout()*
  - *BordeerLayout(int hgap, int vgap)*

UNIT – 4 Border Layout Manager - Methods

Method Name	Purpose of Method
Int getHgap()	Returns the horizontal gap between components.
Void setHgap(int hgap)	Sets the horizontal gap between components.
Public int getVgap()	Returns the vertical gap between components.
Void setVgap(int vgap)	Sets the vertical gap between components.
Void addLayoutComponent(Component comp, Object constraints)	Adds the specified component to the layout using the specified constraint object. The constraint must be one of the constraints – NORTH, SOUTH, EAST, WEST or CENTER

UNIT – 4 Border Layout Manager

- Examples :
  - *JdemoBorderLayout.java*

UNIT – 4 Flow Layout Manager

- The Flow layout manager class a**rrange components in rows across the width of container. Components are arranged in a left-to-right manner.**
- With Flow layout, each component that you **add is placed to the right of previously added ccomponents in a row, when no more component fit in a line, it is taken to the next line.**
- The Flow layout class contains three constants you can use to align components with a Container:
  - ***FlowLayout.LEFT***
  - ***FlowLayout.CENTER***
  - ***FlowLayout.RINGHT***
- If you don not specify alignment, compoents are center - align

UNIT – 4 Flow Layout Manager

- Constructor:
  - *FlowLayout()*
  - *FlowLayout(int align)*
  - *FlowLayout(int align, int hgap, int vgap)*

UNIT – 4 Flow Layout Manager - Methods

Method Name	Purpose of Method
Int getAlignment()	Returns the alignment value for this layout.
Void setAlignment(int align)	Sets the alignments value for this layout.
Int getHgap()	Returns the horizontal gap between components.
Void setHgap(int hgap)	Sets the horizontal gap between comonents.
Int getVgap()	Returns the vertical gap between components.
Void setVgap(int vgap)	Sets the vertical gap between components.

UNIT – 4 Flow Layout Manager

- Examples :
  - *JDemoFlowLayout.java*

UNIT – 4 Grid Layout Manager

- The Grid layout class manager **arrange the components into a rectangular, two dimensional grid of rows and columns.**
- If we want to **arrange components into equal rows and columns, we can use the GridLayout manager class.**
- For creating Grid Layout object, we indicate the number of rows and columns.

UNIT – 4 Grid Layout Manager

- Constructor:
  - *GridLayout()*
  - *GridLayout(int rows, int columns)*
  - *GridLayout(int rows, int columns, int hgap, int vgap)*

UNIT – 4 Grid Layout Manager - Methods

Method Name	Purpose of Method
Int getRows()	Returns the number of rows in this layout.
Void setRows(int rows)	Sets the numbers of rows in this layout.
Int getColumns()	Returns the numbers of columns in ths layout.
Void setColumns(int Columns)	Sets the number of columns in this layout.
Int getHgap()	Returns the horizontal gap between the components.
Void setHgap(int hgap)	Sets the horizontal gap between the components.
Int getVgap()	Returns the vertical gap between components.
Void setVgap(int vgap)	Sets the vertical gap between components.

UNIT – 4 Grid Layout Manager

- Examples :
  - *JdemoGridLayout.java*
  - *Puzzle.java*

UNIT – 4 Card Layout Manager

- The Card Layout manager generate a stack of container or components, one on top of another.
- Each component in the group is referred to as a card and each card be any component type.
- Constructor:
  - *CardLayout()*
  - *CardLayout(int hgap, int vgap)*

UNIT – 4 Card Layout Manager - Methods

Method Name	Purpose of Method
Void addLayoutComponenet(Componenet comp, Object Constraints)	Adds the specified component to this card layout.
Void first(Container parent)	Shows the first card of the container.
Void next(Container parent)	Shows the next card of the container.
Void previous(Container parent)	Shows the previous card of the container.
Void last(container parent)	Shows the last card of the container.
Void add(Component comp,Object Constraints)	Add the specified component to the end of this container.
Void add(Component comp,Object Constraints,int index)	Adds the specified component at the specified location.

UNIT – 4 Card Layout Manager

- Examples :
  - JdemoCardLayout.java

UNIT 4 COMPLETED