SES'S L.S.RAHEJA COLLEGE OF ARTS AND COMMERCE

Course: Core Java Unit: IV Prepared by: Ms. Prajakta Joshi/ Ms. Srushty Padate

Java Arrays

Normally, an array is a collection of similar type of elements which have a contiguous memory location.

Java array is an object which contains elements of a similar data type. Additionally, The elements of an array are stored in a contiguous memory location. It is a data structure where we store similar elements. We can store only a fixed set of elements in a Java array

Types of Array in java

There are two types of array.

- Single Dimensional Array
- Multidimensional Array

Single Dimensional Array in Java

Syntax to Declare an Array in Java

```
dataType[] arr; (or)
dataType []arr; (or)
dataType arr[];
```

Multidimensional Array in Java

In such case, data is stored in row and column based index (also known as matrix form).

Syntax to Declare Multidimensional Array in Java

```
dataType[][] arrayRefVar; (or)

dataType [][]arrayRefVar; (or)

dataType arrayRefVar[][]; (or)

dataType []arrayRefVar[];
```

Java Vector Class

Java Vector class comes under the java.util package. The vector class implements a growable array of objects. Like an array, it contains the component that can be accessed using an integer index.

Vector is very useful if we don't know the size of an array in advance or we need one that can change the size over the lifetime of a program.

Vector implements a dynamic array that means it can grow or shrink as required. It is similar to the ArrayList, but with two differences-

- Vector is synchronized.
- The vector contains many legacy methods that are not the part of a collections framework

1)	add()	It is used to append the specified element in the given vector.
2)	addAll()	It is used to append all of the elements in the specified collection to the end of this Vector.
3)	addElement()	It is used to append the specified component to the end of this vector. It increases the vector size by one.
4)	capacity()	It is used to get the current capacity of this vector.
5)	clear()	It is used to delete all of the elements from this vector.
6)	clone()	It returns a clone of this vector.
7)	contains()	It returns true if the vector contains the specified element.
8)	containsAll()	It returns true if the vector contains all of the elements in the specified collection.
9)	copyInto()	It is used to copy the components of the vector into the specified array.
10)	elementAt()	It is used to get the component at the specified index.
11)	elements()	It returns an enumeration of the components of a vector.

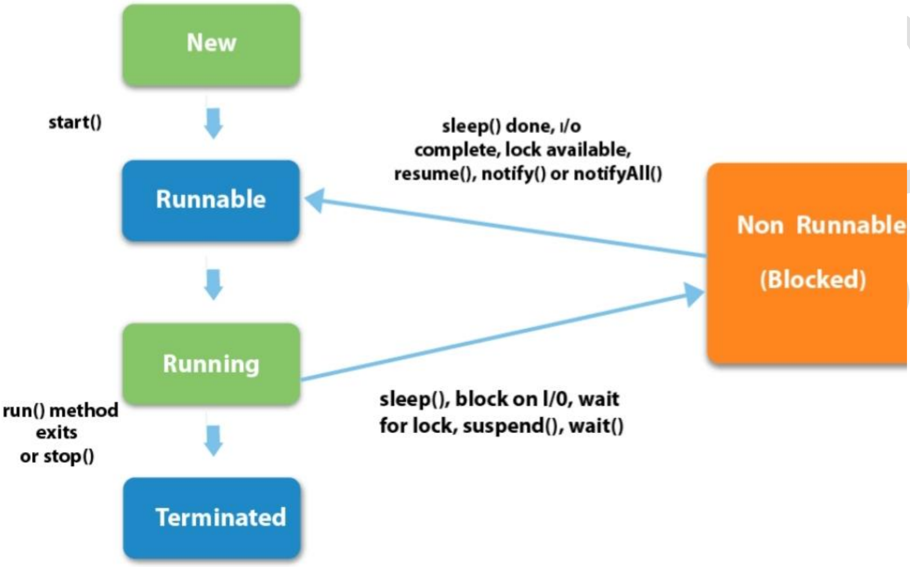
Multithreading in Java

Multithreading in Java is a process of executing multiple threads simultaneously.

A thread is a lightweight sub-process, the smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking

Life cycle of a Thread (Thread States)

A thread can be in one of the five states. According to sun, there is only 4 states in thread life cycle in java new, runnable, non-runnable and terminated. There is no running state.


New
start()
Runnable
sleep() done, I/o complete, lock available, resume(), notify() or notifyAll()
Non Runnable (Blocked)
Running
sleep(), block on I/0, wait for lock, suspend(), wait()
run() method exits or stop()
Terminated

What is Exception in Java

Dictionary Meaning: Exception is an abnormal condition.

In Java, an exception is an event that disrupts the normal flow of the program. It is an object which is thrown at runtime.

Java try block

Java **try** block is used to enclose the code that might throw an exception. It must be used within the method.

Java catch block

Java catch block is used to handle the Exception by declaring the type of exception within the parameter

Java finally block

Java finally block is a block that is used to execute important code such as closing connection, stream etc.

Stream

A stream is a sequence of data. In Java, a stream is composed of bytes. It's called a stream because it is like a stream of water that continues to flow.

1) System.out: standard output stream

2) System.in: standard input stream

3) System.err: standard error stream

NOTE: THESE POINTS ARE INDICATIVE AND NOT EXHAUSTIVE. PLEASE ELLABORATE THE ANSWERS WITH PROPER EXAMPLES WHEREVER APPLICABLE.

Write answers for following

1 What is exception handling in java.

2 Write a short note on life cycle of a thread.

3 What is a stream? Explain input and output streams.

4 How to write binary data to a file?

5 What is a vector? How to create a vector in java?

6 How is a vector different from array? Explain.

7 How to create a new thread in java?

8 Draw a labelled diagram and explain life cycle of thread.

9 Can we handle multiple exceptions using single catch block? Justify your answer.

10 Write a code to demonstrate use of file input stream. Take input file name at command line.

11 Write code to create two threads substantiated from same class. Each iterated for 10 loops and displays message ‘Hello' with a delay of 200miliseconds.

12 List any five vector methods and their functionalities.

13 Explain finally clause with example

14 Explain throws clause with example.

15 How to create a thread and extend the thread class.

16 Write a code to create a vector and add elements to it.
17 Explain with example on how to catch runtime exceptions.

18 What are different thread control methods?

19 Write a short note on main thread.

20 Write short note on reading and writing console input.

Event and Listener (Java Event Handling)

Changing the state of an object is known as an event. For example, click on button, dragging mouse etc. The java.awt.event package provides many event classes and Listener interfaces for event handling.

Java Event classes and Listener interfaces

Event Classes	Listener Interfaces
ActionEvent	ActionListener
MouseEvent	MouseListener and MouseMotionListener
MouseWheelEvent	MouseWheelListener
KeyEvent	KeyListener
ItemEvent	ItemListener
TextEvent	TextListener
AdjustmentEvent	AdjustmentListener
WindowEvent	WindowListener
ComponentEvent	ComponentListener

ContainerEvent	ContainerListener
FocusEvent	FocusListener

NOTE: ALL THE ABOVE EVENTS AND LISTENERS SHOULD BE EXPLAINED.

Java AWT Button

The button class is used to create a labeled button that has platform independent implementation. The application result in some action when the button is pushed.

Java AWT Label

The object of Label class is a component for placing text in a container. It is used to display a single line of read only text. The text can be changed by an application but a user cannot edit it directly.

Java AWT TextField

The object of a TextField class is a text component that allows the editing of a single line text. It inherits TextComponent class.

Java AWT TextArea

The object of a TextArea class is a multi line region that displays text. It allows the editing of multiple line text. It inherits TextComponent class.

Java AWT Checkbox

The Checkbox class is used to create a checkbox. It is used to turn an option on (true) or off (false). Clicking on a Checkbox changes its state from "on" to "off" or from "off" to "on".

Java AWT CheckboxGroup

The object of CheckboxGroup class is used to group together a set of Checkbox. At a time only one check box button is allowed to be in "on" state and remaining check box button in "off" state. It inherits the object class.

Java AWT Toolkit

Toolkit class is the abstract superclass of every implementation in the Abstract Window Toolkit. Subclasses of Toolkit are used to bind various components. It inherits Object class.

Java LayoutManagers

The LayoutManagers are used to arrange components in a particular manner. LayoutManager is an interface that is implemented by all the classes of layout managers. There are following classes that represents the layout managers:

1. java.awt.BorderLayout
2. java.awt.FlowLayout
3. java.awt.GridLayout
4. java.awt.CardLayout

Java BorderLayout

The BorderLayout is used to arrange the components in five regions: north, south, east, west and center. Each region (area) may contain one component only

Java GridLayout

The GridLayout is used to arrange the components in rectangular grid. One component is displayed in each rectangle.

Java FlowLayout

The FlowLayout is used to arrange the components in a line, one after another (in a flow). It is the default layout of applet or panel.

Java BoxLayout

The BoxLayout is used to arrange the components either vertically or horizontally.

NOTE: THESE POINTS ARE INDICATIVE AND NOT EXHAUSTIVE. PLEASE ELLABORATE THE ANSWERS WITH PROPER EXAMPLES WHEREVER APPLICABLE.

Write answers for following

1 Explain keylistener interface.

2 Explain mouselistener interface.

3 What is layout manager? Expalin any two layouts.

4 Explain text field, button and label controls.

5 Explain grid layout.

6 What are event and event sources?

7 List any five action listener with interface.

8 Write a code to implement the concept of event and action listener.

9 Can we handle multiple exceptions using single catch block? Justify your answer.

10 Write a code to demonstrate the use of canvas.

11 Write a short note on flow layout.

12 List and explain different components of abstract window toolkit.
13 Write a short note on border layout.

14 Explain delegation event model.

15 Write a short note on card layout.

16 Explain frames and panels.

17 What is the role of layout manager and which is the default layout?

18 What are adapter classes?

19 What are inner classes?

20 Explain action event and adjustment event class.