Dt : 27/10/2023

Summary of Objects generated from CoreJava:

=>Objects generated from CoreJava are categorized into three types:

1.Container Objects

2.Cursor Objects

3.Utility Objects

1.Container Objects:

=>The objects which are holding the data are known as Container Objects.

=>These Container Objects are categorized into seven types:

(a)User defined class Objects

(b)String-Objects

(c)WrapperClass-Objects

(d)Array-Objects

(e)Collection<E> Objects

(f)Map<K,V> Objects

(g)Enum<E> Objects

=>The following is the Complete list of Container Objects:

(a)User defined Class Objects

(i)Muatble Objects

(ii)Immutable Objects

(b)String-Objects

(i)String class - Immutable Objects

(ii)StringBuffer class - Mutable Objects - Synchronized class

(iii)StringBuilder class - Mutable Object - NonSynchronized Class

(c)WrapperClass Objects

(i)Byte Object

(ii)Short Object

(iii)Integer Object

(iv)Long Object

(v)Float Object

(vi)Double Object

(vii)Character Object

(viii)Boolean Object

(d)Array Objects

(i)Array holding User defined Class Objects

(ii)Array holding WrapperClass Objects

(iii)Array holding String-Objects

(iv)Array holding DisSimiler Objects(Object-Array)

(v)Array holding Array-Objects(Jagged Array)

(e)Collection<E> Objects

1.Set<E>

(i)HashSet<E>

(ii)LinkedHashSet<E>

(iii)TreeSet<E>

2.List<E>

(i)ArrayList<E>

(ii)LinkedList<E>

(iii)Vector<E>

=>Stack<E>

3.Queue<E>

(i)PriorityQueue<E>

=>Deque<E>

(ii)ArrayDeque<E>

(iii)LinkedList<E>

(f)Map<K,V> Objects

(i)HashMap<K,V>

(ii)LinkedHashMap<K,V>

(iii)TreeMap<K,V>

(iv)Hashtable<K,V>

(g)Enum<E> Objects

Total Container Objects : 31

2.Cursor Objects:

=>The Objects which are used to retrieve elements from collection objects are known as Cursor Objects.

=>The following are some important Cursor Statements used on collection Objects:

(a)Iterator<E>

(b)ListIterator<E>

(c)Enumeration<E>

(d)Spliterator<T>

3.Utility Objects:

=>The Objects which perform operations on Other Objects are known as Utility Objects.

=>The following are some important Utility Classes:

(a)Scanner

(b)StringTokenizer

(c)StringJoiner

(d)Arrays

(e)Collections

==========

=

**imp*

Multi-Threading in Java:

define program?

=>Program is a set-of-Instructions.

define Application?

=>Set-of-programs collected together to perform defined action is known as Application.

define Process?

=>According to OperatingSystem,the program under execution is a process.

=>According to Java,the Application under execution is a process.

define Task?

=>The part of process is known as Task

define Multi-Tasking?

=>Executing Multiple Tasks Simultaneously is known as Multi-Tasking.

(Simultaneously mean at-a-time but not parallel)

=>Multi-Tasking categorized into two types:

(i)Process based Multi-Tasking

(iii)Thread based Multi-Tasking

(i)Process based Multi-Tasking:

=>Executing multiple tasks from multiple processes is known as Process based Multi-tasking.

=>Process based Multi-Tasking is used in OperatingSystem design.

(iii)Thread based Multi-Tasking:

=>Executing Multiple tasks from the same process(Application) is known as Thread based Multi-Tasking

=>Thread based Multi-tasking is used in Application development

=>According to Application each program is one Task.

define Thread?

=>Part-of-Task is known as Thread.

Note:

(i)Thread is a Light-Weight and BackGround process

LightWeight - means consumes Less execution time.

BackGround - means separate identification is not available

(ii)Thread is also known as Child-Process

define Multi-Threading?

=>Executing multiple threads Simultaneously is known as Multi-Threading

Diagram: