JAVA version: java.awt.color

**Packages**  **Description**

**Java.awt.color** Provides classes for color spaces

|  |
| --- |
| **Interfaces** |

|  |
| --- |
| **Classes**  ColorSpace -This abstract class is used to serve as a color space tag to identify the specific color space of a Color object or, via a ColorModel object, of an Image, a BufferedImage, or a GraphicsDevice.  ICC\_ColorSpace-The ICC\_ColorSpace class is an implementation of the abstract ColorSpace class.  ICC\_Profile-A representation of color profile data for device independent and device dependent color spaces based on the International Color Consortium Specification ICC.1:2001-12, File Format for Color Profiles, (see http://www.color.org).  ICC\_ProfileGray-A subclass of the ICC\_Profile class which represents profiles which meet the following criteria: the color space type of the profile is TYPE\_GRAY and the profile includes the grayTRCTag and mediaWhitePointTag tags.  ICC\_ProfileRGB-The ICC\_ProfileRGB class is a subclass of the ICC\_Profile class that represents profiles which meet the following criteria: The profile's color space type is RGB. The profile includes the redColorantTag, greenColorantTag, blueColorantTag, redTRCTag, greenTRCTag, blueTRCTag, and mediaWhitePointTag tags. The ICC\_Profile getInstance method will return an ICC\_ProfileRGB object when these conditions are met. |

|  |
| --- |
| **Exceptions**  CMMException-This exception is thrown if the native CMM returns an error.  ProfileDataException-This exception is thrown when an error occurs in accessing or processing an ICC\_Profile object. |

JAVA version:[**java.awt.datatransfer**](https://docs.oracle.com/javase/7/docs/api/java/awt/datatransfer/package-summary.html)

**Packages**  **Description**

[**java.awt.datatransfer**](https://docs.oracle.com/javase/7/docs/api/java/awt/datatransfer/package-summary.html) Provides interfaces and classes for transferring data between and within applications.

|  |
| --- |
| **Interfaces**  ClipboardOwner-Defines the interface for classes that will provide data to a clipboard.  FlavorListener-Defines an object which listens for FlavorEvents.  FlavorMap-A two-way Map between "natives" (Strings), which correspond to platform- specfic data formats, and "flavors" (DataFlavors), which corerspond to platform-independent MIME types.  FlavorTable-A FlavorMap which relaxes the traditional 1-to-1 restriction of a Map.  Transferable-Defines the interface for classes that can be used to provide data for a transfer operation. |

|  |
| --- |
| **Classes**  Clipboard-A class that implements a mechanism to transfer data using cut/copy/paste operations.  DataFlavor-A DataFlavor provides meta information about data.  FlavorEvent-FlavorEvent is used to notify interested parties that available DataFlavors have changed in the Clipboard (the event source).  StringSelection -A Transferable which implements the capability required to transfer a String.  SystemFlavorMap-The SystemFlavorMap is a configurable map between "natives" (Strings), which correspond to platform-specific data formats, and "flavors" (DataFlavors), which correspond to platform-independent MIME types. |

|  |
| --- |
| **Exceptions**  MimeTypeParseException-A class to encapsulate MimeType parsing related exceptions  UnsupportedFlavorException -Signals that the requested data is not supported in this flavor. |

## JAVA version: java.awt.dnd

**Packages**  **Description**

java.awt.dnd Drag and Drop is a direct manipulation gesture found in many Graphical User Interface systems that provides a mechanism to transfer information between two entities logically associated with presentation elements in the GUI.

|  |
| --- |
| **Interfaces**  Autoscroll-During DnD operations it is possible that a user may wish to drop the subject of the operation on a region of a scrollable GUI control that is not currently visible to the user.  DragGestureListener-The listener interface for receiving drag gesture events.  DragSourceListener-The DragSourceListener defines the event interface for originators of Drag and Drop operations to track the state of the user's gesture, and to provide appropriate "drag over" feedback to the user throughout the Drag and Drop operation.  DragSourceMotionListener-A listener interface for receiving mouse motion events during a drag operation.  DropTargetListener-The DropTargetListener interface is the callback interface used by the DropTarget class to provide notification of DnD operations that involve the subject DropTarget |

|  |
| --- |
| **Classes**  DnDConstants-This class contains constant values representing the type of action(s) to be performed by a Drag and Drop operation.  DragGestureEvent-A DragGestureEvent is passed to DragGestureListener's dragGestureRecognized() method when a particular DragGestureRecognizer detects that a platform dependent drag initiating gesture has occurred on the Component that it is tracking. |

|  |
| --- |
| DragGestureRecognizer-The DragGestureRecognizer is an abstract base class for the specification of a platform-dependent listener that can be associated with a particular Component in order to identify platform-dependent drag initiating gestures.  DragSource-The DragSource is the entity responsible for the initiation of the Drag and Drop operation, and may be used in a number of scenarios: 1 default instance per JVM for the lifetime of that JVM.  DragSourceAdapter-An abstract adapter class for receiving drag source events.  DragSourceContext-The DragSourceContext class is responsible for managing the initiator side of the Drag and Drop protocol.  DragSourceDragEvent-The DragSourceDragEvent is delivered from the DragSourceContextPeer, via the DragSourceContext, to the DragSourceListener registered with that DragSourceContext and with its associated DragSource.  DragSourceDropEvent-The DragSourceDropEvent is delivered from the DragSourceContextPeer, via the DragSourceContext, to the dragDropEnd method of DragSourceListeners registered with that DragSourceContext and with its associated DragSource.  DragSourceEvent-This class is the base class for DragSourceDragEvent and DragSourceDropEvent.  DropTarget-The DropTarget is associated with a Component when that Component wishes to accept drops during Drag and Drop operations.  DropTarget.DropTargetAutoScroller-this protected nested class implements autoscrolling  DropTargetAdapter-An abstract adapter class for receiving drop target events.  DropTargetContext-A DropTargetContext is created whenever the logical cursor associated with a Drag and Drop operation coincides with the visible geometry of a Component associated with a DropTarget.  DropTargetDragEvent-The DropTargetDragEvent is delivered to a DropTargetListener via its dragEnter() and dragOver() methods.  DropTargetDropEvent-The DropTargetDropEvent is delivered via the DropTargetListener drop() method.  DropTargetEvent-The DropTargetEvent is the base class for both the DropTargetDragEvent and the DropTargetDropEvent.  MouseDragGestureRecognizer-This abstract subclass of DragGestureRecognizer defines a DragGestureRecognizer for mouse-based gestures. |

|  |
| --- |
| **Exceptions**  InvalidDnDOperationException-This exception is thrown by various methods in the java.awt.dnd package. |

JAVA version: **java.awt.font**

**Packages**  **Description**

**java.awt.font** Provides classes and interface relating to fonts.

|  |
| --- |
| **Interfaces**  MultipleMaster-The MultipleMaster interface represents Type 1 Multiple Master fonts.  OpenType-The OpenType interface represents OpenType and TrueType fonts. |

|  |
| --- |
| **Classes**  FontRenderContext-The FontRenderContext class is a container for the information needed to correctly measure text.  GlyphJustificationInfo-The GlyphJustificationInfo class represents information about the justification properties of a glyph.  GlyphMetrics-The GlyphMetrics class represents infomation for a single glyph.  GlyphVector-A GlyphVector object is a collection of glyphs containing geometric information for the placement of each glyph in a transformed coordinate space which corresponds to the device on which the GlyphVector is ultimately displayed.  GraphicAttribute-This class is used with the CHAR\_REPLACEMENT attribute.  ImageGraphicAttribute-The ImageGraphicAttribute class is an implementation of GraphicAttribute which draws images in a TextLayout.  LayoutPath-LayoutPath provides a mapping between locations relative to the baseline and points in user space.  LineBreakMeasurer-The LineBreakMeasurer class allows styled text to be broken into lines (or segments) that fit within a particular visual advance. |
| LineMetrics-The LineMetrics class allows access to the metrics needed to layout characters along a line and to layout of a set of lines.  NumericShaper-The NumericShaper class is used to convert Latin-1 (European) digits to other Unicode decimal digits.  ShapeGraphicAttribute-The ShapeGraphicAttribute class is an implementation of GraphicAttribute that draws shapes in a TextLayout.  TextAttribute-The TextAttribute class defines attribute keys and attribute values used for text rendering.  TextHitInfo-The TextHitInfo class represents a character position in a text model, and a bias, or "side," of the character.  TextLayout-TextLayout is an immutable graphical representation of styled character data.  TextLayout.CaretPolicy-Defines a policy for determining the strong caret location.  TextMeasurer-The TextMeasurer class provides the primitive operations needed for line break: measuring up to a given advance, determining the advance of a range of characters, and generating a TextLayout for a range of characters.  TransformAttribute-The TransformAttribute class provides an immutable wrapper for a transform so that it is safe to use as an  attribute. | | |

|  |
| --- |
| **Exceptions** |

JAVA version: **java.awt.geom**

**Packages**  **Description**

**java.awt.geom** Provides the Java 2D classes for defining and performing operations on objects related to two-dimensional geometry.

|  |
| --- |
| **Interfaces**  MultipleMaster-The MultipleMaster interface represents Type 1 Multiple Master fonts.  OpenType-The OpenType interface represents OpenType and TrueType fonts. |

|  |
| --- |
| **Classes**  **FontRenderContext-The FontRenderContext class is a container for the information needed to correctly measure text.**  **GlyphJustificationInfo-The GlyphJustificationInfo class represents information about the justification properties of a glyph.**  **GlyphMetrics-The GlyphMetrics class represents infomation for a single glyph.**  **GlyphVector-A GlyphVector object is a collection of glyphs containing geometric information for the placement of each glyph in a transformed coordinate space which corresponds to the device on which the GlyphVector is ultimately displayed.**  **GraphicAttribute-This class is used with the CHAR\_REPLACEMENT attribute.**  **ImageGraphicAttribute-The ImageGraphicAttribute class is an implementation of GraphicAttribute which draws images in a TextLayout-LayoutPath-LayoutPath provides a mapping between locations relative to the baseline and points in user space.**  **LineBreakMeasurer-The LineBreakMeasurer class allows styled text to be broken into lines (or segments) that fit within a particular visual advance.** **AffineTransform**  **The AffineTransform class represents a 2D affine transform that performs a linear mapping from 2D coordinates to other 2D coordinates that preserves the "straightness" and "parallelness" of lines.**  **Arc2D-Arc2D is the abstract superclass for all objects that store a 2D arc defined by a framing rectangle, start angle, angular extent (length of the arc), and a closure type (OPEN, CHORD, or PIE).**  **Arc2D.Double-This class defines an arc specified in double precision.**  **Arc2D.Float-This class defines an arc specified in float precision.**  **Area-An Area object stores and manipulates a resolution-independent description of an enclosed area of 2-dimensional space.**  **CubicCurve2D-The CubicCurve2D class defines a cubic parametric curve segment in (x,y) coordinate space.**  **CubicCurve2D.Double-A cubic parametric curve segment specified with double coordinates.**  **CubicCurve2D.Float-A cubic parametric curve segment specified with float coordinates.**  **Dimension2D-The Dimension2D class is to encapsulate a width and a height dimension.**  **Ellipse2D-The Ellipse2D class describes an ellipse that is defined by a framing rectangle.**  **Ellipse2D.Double-The Double class defines an ellipse specified in double precision.**  **Ellipse2D.Float-The Float class defines an ellipse specified in float precision.**  **FlatteningPathIterator-The FlatteningPathIterator class returns a flattened view of another PathIterator object.**  **GeneralPath-The GeneralPath class represents a geometric path constructed from straight lines, and quadratic and cubic (Bézier) curves.**  **Line2D-This Line2D represents a line segment in (x,y) coordinate space.**  **Line2D.Double-A line segment specified with double coordinates.**  **Line2D.Float-A line segment specified with float coordinates.**  **Path2D-The Path2D class provides a simple, yet flexible shape which represents an arbitrary geometric path.**  **Path2D.Double-The Double class defines a geometric path with coordinates stored in double precision floating point.**  **Path2D.Float-The Float class defines a geometric path with coordinates stored in single precision floating point.**  **Point2D -The Point2D class defines a point representing a location in (x,y) coordinate space.**  **Point2D.Double-The Double class defines a point specified in double precision.**  **Point2D.Float-The Float class defines a point specified in float precision.**  **QuadCurve2D-The QuadCurve2D class defines a quadratic parametric curve segment in (x,y) coordinate space.**  **QuadCurve2D.Double-A quadratic parametric curve segment specified with double coordinates.**  **QuadCurve2D.Float-A quadratic parametric curve segment specified with float coordinates.**  **Rectangle2D-The Rectangle2D class describes a rectangle defined by a location (x,y) and dimension (w x h).**  **Rectangle2D.Double-The Double class defines a rectangle specified in double coordinates.**  **Rectangle2D.Float-The Float class defines a rectangle specified in float coordinates.**  **RectangularShape-RectangularShape is the base class for a number of Shape objects whose geometry is defined by a rectangular frame.**  **RoundRectangle2D-The RoundRectangle2D class defines a rectangle with rounded corners defined by a location (x,y), a dimension (w x h), and the width and height of an arc with which to round the corners.**  **RoundRectangle2D.Double-The Double class defines a rectangle with rounded corners all specified in double coordinates.**  **RoundRectangle2D.Float-The Float class defines a rectangle with rounded corners all specified in float coordinates.** |

|  |
| --- |
| **Exceptions**  IllegalPathStateException-The IllegalPathStateException represents an exception that is thrown if an operation is performed on a path that is in an illegal state with respect to the particular operation being performed, such as appending a path segment to a GeneralPath without an initial moveto.  NoninvertibleTransformException-The NoninvertibleTransformException class represents an exception that is thrown if an operation is performed requiring the inverse of an AffineTransform object but the AffineTransform is in a non-invertible state. |

JAVA version:[**java.awt.im**](https://docs.oracle.com/javase/8/docs/api/java/awt/im/package-summary.html)

**Packages**  **Description**

**java.awt.im** Provides classes and interfaces for the input method framework.

|  |
| --- |
| **Interfaces**  InputMethodRequests -InputMethodRequests defines the requests that a text editing component has to handle in order to work with input methods. |

|  |
| --- |
| **Classes**  InputContext -Provides methods to control text input facilities such as input methods and keyboard layouts.  InputMethodHighlight -An InputMethodHighlight is used to describe the highlight attributes of text being composed.  InputSubset -Defines additional Unicode subsets for use by input methods. |

|  |
| --- |
| **Exceptions** |

JAVA version: **java.awt.im.spi**

**Packages**  **Description**

**java.awt.im.spi** Provides interfaces that enable the development of input methods that can be used with any Java runtime environment.

|  |
| --- |
| **Interfaces**  InputMethod-Defines the interface for an input method that supports complex text input.  InputMethodContext-Provides methods that input methods can use to communicate with their client components or to request other services.  InputMethodDescriptor-Defines methods that provide sufficient information about an input method to enable selection and loading of that input method. |

|  |
| --- |
| **Classes** |

|  |
| --- |
| **Exceptions** |

JAVA version: **java.awt.image.renderable**

**Packages**  **Description**

**java.awt.image.renderable** Provides classes and interfaces for producing rendering-independent images.

|  |
| --- |
| **Interfaces**  ContextualRenderedImageFactory-ContextualRenderedImageFactory provides an interface for the functionality that may differ between instances of RenderableImageOp.  RenderableImage-A RenderableImage is a common interface for rendering-independent images (a notion which subsumes resolution independence).  RenderedImageFactory-The RenderedImageFactory interface (often abbreviated RIF) is intended to be implemented by classes that wish to act as factories to produce different renderings, for example by executing a series of BufferedImageOps on a set of sources, depending on a specific set of parameters, properties, and rendering hints. |

|  |
| --- |
| **Classes**  ParameterBlock -A ParameterBlock encapsulates all the information about sources and parameters (Objects) required by a RenderableImageOp, or other classes that process images.  RenderableImageOp-This class handles the renderable aspects of an operation with help from its associated instance of a ContextualRenderedImageFactory.  RenderableImageProducer-An adapter class that implements ImageProducer to allow the asynchronous production of a RenderableImage.  RenderContext-A RenderContext encapsulates the information needed to produce a specific rendering from a RenderableImage. |

|  |
| --- |
| **Exceptions** |

JAVA version: **java.awt.print**

**Packages**  **Description**

**java.awt.print**  Provides classes and interfaces for a general printing API.

|  |
| --- |
| **Interfaces**  Pageable -The Pageable implementation represents a set of pages to be printed.  Printable -The Printable interface is implemented by the print methods of the current page painter, which is called by the printing system to render a page.  PrinterGraphics-The PrinterGraphics interface is implemented by Graphics objects that are passed to Printable objects to render a page. |

|  |
| --- |
| **Classes**  Book-The Book class provides a representation of a document in which pages may have different page formats and page painters.  PageFormat -The PageFormat class describes the size and orientation of a page to be printed.  Paper  The Paper class describes the physical characteristics of a piece of paper.  PrinterJob-The PrinterJob class is the principal class that controls printing. |

|  |
| --- |
| **Exceptions**  PrinterAbortException-The PrinterAbortException class is a subclass of PrinterException and is used to indicate that a user or application has terminated the print job while it was in the process of printing.  PrinterException-The PrinterException class and its subclasses are used to indicate that an exceptional condition has occurred in the print system.  PrinterIOException-The PrinterIOException class is a subclass of PrinterException and is used to indicate that an IO error of some sort has occurred while printing. |

JAVA version: **java.lang.annotation**

**Packages**  **Description**

**java.lang.annotation**  Provides library support for the Java programming language annotation facility.

|  |
| --- |
| **Interfaces**  Annotation-The common interface extended by all annotation types. |

|  |
| --- |
| **Classes** |

|  |
| --- |
| **Exceptions**  AnnotationTypeMismatchException-Thrown to indicate that a program has attempted to access an element of an annotation whose type has changed after the annotation was compiled (or serialized).  IncompleteAnnotationException-Thrown to indicate that a program has attempted to access an element of an annotation type that was added to the annotation type definition after the annotation was compiled (or serialized). |

JAVA version: **java.lang.instrument**

Packages Description

**java.lang.instrument** Provides services that allow Java programming language agents to instrument programs running on the JVM.

|  |
| --- |
| **Interfaces**  ClassFileTransformer-An agent provides an implementation of this interface in order to transform class files.  Instrumentation-This class provides services needed to instrument Java programming language code. |

|  |
| --- |
| **Classes**  ClassDefinition-This class serves as a parameter block to the Instrumentation.redefineClasses method. |

|  |
| --- |
| **Exceptions**  IllegalClassFormatException-Thrown by an implementation of ClassFileTransformer.transform when its input parameters are invalid.  UnmodifiableClassException-Thrown by an implementation of Instrumentation.redefineClasses when one of the specified classes cannot be modified. |

JAVA version: **java.lang.invoke**

Packages Description

**java.lang.invoke** The java.lang.invoke package contains dynamic language support provided directly by the Java core class libraries and virtual machine.

|  |
| --- |
| **Interfaces**  MethodHandleInfo-A symbolic reference obtained by cracking a direct method handle into its consitutent symbolic parts. |

|  |
| --- |
| **Classes**  CallSite-A CallSite is a holder for a variable MethodHandle, which is called its target.  ConstantCallSite-A ConstantCallSite is a CallSite whose target is permanent, and can never be changed.  LambdaMetafactory-Methods to facilitate the creation of simple "function objects" that implement one or more interfaces by delegation to a provided MethodHandle, possibly after type adaptation and partial evaluation of arguments.  MethodHandle-A method handle is a typed, directly executable reference to an underlying method, constructor, field, or similar low-level operation, with optional transformations of arguments or return values.  MethodHandleProxies-This class consists exclusively of static methods that help adapt method handles to other JVM types, such as interfaces.  MethodHandles-This class consists exclusively of static methods that operate on or return method handles.  MethodHandles.Lookup-A lookup object is a factory for creating method handles, when the creation requires access checking.  MethodType-A method type represents the arguments and return type accepted and returned by a method handle, or the arguments and return type passed and expected by a method handle caller.  MutableCallSite-A MutableCallSite is a CallSite whose target variable behaves like an ordinary field.  SerializedLambda-Serialized form of a lambda expression.  SwitchPoint-A SwitchPoint is an object which can publish state transitions to other threads.  VolatileCallSite-A VolatileCallSite is a CallSite whose target acts like a volatile variable. |

|  |
| --- |
| **Exceptions**  LambdaConversionException-LambdaConversionException  WrongMethodTypeException-Thrown to indicate that code has attempted to call a method handle via the wrong method type. |

JAVA version: **java.lang.management**

Packages Description

**java.lang.management** Provides the management interfaces for monitoring and management of the Java virtual machine and other components in the Java runtime.

|  |
| --- |
| **Interfaces**  BufferPoolMXBean-The management interface for a buffer pool, for example a pool of direct or mapped buffers.  ClassLoadingMXBean-The management interface for the class loading system of the Java virtual machine.  CompilationMXBean-The management interface for the compilation system of the Java virtual machine.  GarbageCollectorMXBean-The management interface for the garbage collection of the Java virtual machine.  MemoryManagerMXBean-The management interface for a memory manager.  MemoryMXBean -The management interface for the memory system of the Java virtual machine.  MemoryPoolMXBean-The management interface for a memory pool.  OperatingSystemMXBean-The management interface for the operating system on which the Java virtual machine is running.  PlatformLoggingMXBean-The management interface for the logging facility.  PlatformManagedObject-A platform managed object is a JMX MXBean for monitoring and managing a component in the Java platform.  RuntimeMXBean-The management interface for the runtime system of the Java virtual machine.  ThreadMXBean-The management interface for the thread system of the Java virtual machine. |

|  |
| --- |
| **Classes**  LockInfo-Information about a lock.  ManagementFactory-The ManagementFactory class is a factory class for getting managed beans for the Java platform.  ManagementPermission-The permission which the SecurityManager will check when code that is running with a SecurityManager calls methods defined in the management interface for the Java platform.  MemoryNotificationInfo-The information about a memory notification.  MemoryUsage-A MemoryUsage object represents a snapshot of memory usage.  MonitorInfo-Information about an object monitor lock.  ThreadInfo-Thread information. |

|  |
| --- |
| **Exceptions** |

JAVA version: **java.lang.ref**

Packages Description

**java.lang.ref** Provides reference-object classes, which support a limited degree of interaction with the garbage collector.

|  |
| --- |
| **Interfaces** |

|  |
| --- |
| **Classes**  PhantomReference<T>-Phantom reference objects, which are enqueued after the collector determines that their referents may otherwise be reclaimed.  Reference<T>-Abstract base class for reference objects.  ReferenceQueue<T>-Reference queues, to which registered reference objects are appended by the garbage collector after the appropriate reachability changes are detected.  SoftReference<T>-Soft reference objects, which are cleared at the discretion of the garbage collector in response to memory demand.  WeakReference<T> -Weak reference objects, which do not prevent their referents from being made finalizable, finalized, and then reclaimed. |

|  |
| --- |
| **Exceptions** |

JAVA version: **java.math**

Packages Description

**java.math**  Provides classes for performing arbitrary-precision integer arithmetic (BigInteger) and arbitrary-precision decimal arithmetic (BigDecimal).

|  |
| --- |
| **Interfaces** |

|  |
| --- |
| **Classes**  BigDecimal-Immutable, arbitrary-precision signed decimal numbers.  BigInteger-Immutable arbitrary-precision integers.  MathContext-Immutable objects which encapsulate the context settings which describe certain rules for numerical operators, such as those implemented by the BigDecimal class. |

|  |
| --- |
| **Exceptions** |

JAVA version: **java.net**

Packages Description

**java.net**  Provides the classes for implementing networking applications.

|  |
| --- |
| **Interfaces**  ContentHandlerFactory-This interface defines a factory for content handlers.  CookiePolicy-CookiePolicy implementations decide which cookies should be accepted and which should be rejected.  CookieStore-A CookieStore object represents a storage for cookie.  DatagramSocketImplFactory-This interface defines a factory for datagram socket implementations.  FileNameMap-A simple interface which provides a mechanism to map between a file name and a MIME type string.  ProtocolFamily-Represents a family of communication protocols.  SocketImplFactory-This interface defines a factory for socket implementations.  SocketOption<T>-A socket option associated with a socket.  SocketOptions-Interface of methods to get/set socket options.  URLStreamHandlerFactory-This interface defines a factory for URL stream protocol handlers. |

|  |
| --- |
| **Classes**  Authenticator-The class Authenticator represents an object that knows how to obtain authentication for a network connection.  CacheRequest-Represents channels for storing resources in the ResponseCache.  CacheResponse-Represent channels for retrieving resources from the ResponseCache.  ContentHandler -The abstract class ContentHandler is the superclass of all classes that read an Object from a URLConnection.  CookieHandler-A CookieHandler object provides a callback mechanism to hook up a HTTP state management policy implementation into the HTTP protocol handler.  CookieManager-CookieManager provides a concrete implementation of CookieHandler, which separates the storage of cookies from the policy surrounding accepting and rejecting cookies.  DatagramPacket -This class represents a datagram packet.  DatagramSocket -This class represents a socket for sending and receiving datagram packets.  DatagramSocketImpl-Abstract datagram and multicast socket implementation base class.  HttpCookie-An HttpCookie object represents an HTTP cookie, which carries state information between server and user agent.  HttpURLConnection-A URLConnection with support for HTTP-specific features.  IDN -Provides methods to convert internationalized domain names (IDNs) between a normal Unicode representation and an ASCII Compatible Encoding (ACE) representation.  Inet4Address-This class represents an Internet Protocol version 4 (IPv4) address.  Inet6Address-This class represents an Internet Protocol version 6 (IPv6) address.  InetAddress -This class represents an Internet Protocol (IP) address.  InetSocketAddress-This class implements an IP Socket Address (IP address + port number) It can also be a pair (hostname + port number), in which case an attempt will be made to resolve the hostname.  InterfaceAddress -This class represents a Network Interface address.  JarURLConnection-A URL Connection to a Java ARchive (JAR) file or an entry in a JAR file.  MulticastSocket-The multicast datagram socket class is useful for sending and receiving IP multicast packets.  NetPermission-This class is for various network permissions.  NetworkInterface -This class represents a Network Interface made up of a name, and a list of IP addresses assigned to this interface.  PasswordAuthentication-The class PasswordAuthentication is a data holder that is used by Authenticator.  Proxy -This class represents a proxy setting, typically a type (http, socks) and a socket address.  ProxySelector-Selects the proxy server to use, if any, when connecting to the network resource referenced by a URL.  ResponseCache-Represents implementations of URLConnection caches.  SecureCacheResponse-Represents a cache response originally retrieved through secure means, such as TLS.  ServerSocket -This class implements server sockets.  Socket -This class implements client sockets (also called just "sockets").  SocketAddress -This class represents a Socket Address with no protocol attachment.  SocketImpl -The abstract class SocketImpl is a common superclass of all classes that actually implement sockets.  SocketPermission -This class represents access to a network via sockets.  StandardSocketOptions -Defines the standard socket options.  URI -Represents a Uniform Resource Identifier (URI) reference.  URL-Class URL represents a Uniform Resource Locator, a pointer to a "resource" on the World Wide Web.  URLClassLoader -This class loader is used to load classes and resources from a search path of URLs referring to both JAR files and directories.  URLConnection-The abstract class URLConnection is the superclass of all classes that represent a communications link between the application and a URL.  URLDecoder -Utility class for HTML form decoding.  URLEncoder -Utility class for HTML form encoding.  URLPermission-Represents permission to access a resource or set of resources defined by a given url, and for a given set of user-settable request methods and request headers.  URLStreamHandler -The abstract class URLStreamHandler is the common superclass for all stream protocol handlers. |

|  |
| --- |
| **Exceptions**  BindException-Signals that an error occurred while attempting to bind a socket to a local address and port.  ConnectException-Signals that an error occurred while attempting to connect a socket to a remote address and port.  HttpRetryException-Thrown to indicate that a HTTP request needs to be retried but cannot be retried automatically, due to streaming mode being enabled.  MalformedURLException-Thrown to indicate that a malformed URL has occurred.  NoRouteToHostException-Signals that an error occurred while attempting to connect a socket to a remote address and port.  PortUnreachableException-Signals that an ICMP Port Unreachable message has been received on a connected datagram.  ProtocolException-Thrown to indicate that there is an error in the underlying protocol, such as a TCP error.  SocketException-Thrown to indicate that there is an error creating or accessing a Socket.  SocketTimeoutException-Signals that a timeout has occurred on a socket read or accept.  UnknownHostException-Thrown to indicate that the IP address of a host could not be determined.  UnknownServiceException-Thrown to indicate that an unknown service exception has occurred.  URISyntaxException-Checked exception thrown to indicate that a string could not be parsed as a URI reference. |

JAVA version: **java.nio**

Packages Description

**java.nio Defines buffers, which are containers for data, and provides an overview of the other NIO packages.**

|  |
| --- |
| **Interfaces** |

|  |
| --- |
| **Classes**  Buffer -A container for data of a specific primitive type.  ByteBuffer -A byte buffer.  ByteOrder -A typesafe enumeration for byte orders.  CharBuffer-A char buffer.  DoubleBuffer-A double buffer.  FloatBuffer-A float buffer.  IntBuffer-An int buffer.  LongBuffer-A long buffer.  MappedByteBuffer-A direct byte buffer whose content is a memory-mapped region of a file.  ShortBuffer-A short buffer. |

|  |
| --- |
| **Exceptions**  BufferOverflowException-Unchecked exception thrown when a relative put operation reaches the target buffer's limit.  BufferUnderflowException-Unchecked exception thrown when a relative get operation reaches the source buffer's limit.  InvalidMarkException-Unchecked exception thrown when an attempt is made to reset a buffer when its mark is not defined.  ReadOnlyBufferException-Unchecked exception thrown when a content-mutation method such as put or compact is invoked upon a read-only buffer. |

JAVA version: **java.nio.channels.spi**

Packages Description

**java.nio.channels.spi**  Service-provider classes for the java.nio.channels package.

|  |
| --- |
| **Interfaces** |

|  |
| --- |
| **Classes**  AbstractInterruptibleChannel-Base implementation class for interruptible channels.  AbstractSelectableChannel-Base implementation class for selectable channels.  AbstractSelectionKey-Base implementation class for selection keys.  AbstractSelector-Base implementation class for selectors.  AsynchronousChannelProvider-Service-provider class for asynchronous channels.  SelectorProvider-Service-provider class for selectors and selectable channels. |

|  |
| --- |
| **Exceptions** |

JAVA version: **java.nio.charset**

Packages Description

**java.nio.charset**  Defines charsets, decoders, and encoders, for translating between bytes and Unicode characters.

|  |
| --- |
| **Interfaces** |

|  |
| --- |
| **Classes**  Charset-A named mapping between sequences of sixteen-bit Unicode code units and sequences of bytes.  CharsetDecoder-An engine that can transform a sequence of bytes in a specific charset into a sequence of sixteen-bit Unicode characters.  CharsetEncoder-An engine that can transform a sequence of sixteen-bit Unicode characters into a sequence of bytes in a specific charset.  CoderResult-A description of the result state of a coder.  CodingErrorAction-A typesafe enumeration for coding-error actions.  StandardCharsets-Constant definitions for the standard Charsets. |

|  |
| --- |
| **Exceptions**  CharacterCodingException-Checked exception thrown when a character encoding or decoding error occurs.  IllegalCharsetNameException-Unchecked exception thrown when a string that is not a legal charset name is used as such.  MalformedInputException-Checked exception thrown when an input byte sequence is not legal for given charset, or an input character sequence is not a legal sixteen-bit Unicode sequence.  UnmappableCharacterException-Checked exception thrown when an input character (or byte) sequence is valid but cannot be mapped to an output byte (or character) sequence.  UnsupportedCharsetException-Unchecked exception thrown when no support is available for a requested charset. |

JAVA version: **java.nio.charset.spi**

Packages Description

**java.nio.charset.spi**  Service-provider classes for the java.nio.charset package.

|  |
| --- |
| **Interfaces** |

|  |
| --- |
| **Classes**  CharsetProvider-Charset service-provider class. |

|  |
| --- |
| **Exceptions** |