Module java.base

Package java.lang

`package java.lang`

Provides classes that are fundamental to the design of the Java programming language. The most important classes are `Object`, which is the root of the class hierarchy, and `Class`, instances of which represent classes at run time.

Frequently it is necessary to represent a value of primitive type as if it were an object. The wrapper classes `Boolean`, `Character`, `Integer`, `Long`, `Float`, and `Double` serve this purpose. An object of type `Double`, for example, contains a field whose type is double, representing that value in such a way that a reference to it can be stored in a variable of reference type. These classes also provide a number of methods for converting among primitive values, as well as supporting such standard methods as equals and hashCode. The `Void` class is a non-instantiable class that holds a reference to a `Class` object representing the type void.

The class `Math` provides commonly used mathematical functions such as sine, cosine, and square root. The classes `String`, `StringBuffer`, and `StringBuilder` similarly provide commonly used operations on character strings.

Classes `ClassLoader`, `Process`, `ProcessBuilder`, `Runtime`, `SecurityManager`, and `System` provide "system operations" that manage the dynamic loading of classes, creation of external processes, host environment inquiries such as the time of day, and enforcement of security policies.

Class `Throwable` encompasses objects that may be thrown by the `throw` statement. Subclasses of `Throwable` represent errors and exceptions.

Character Encodings

The specification of the `java.nio.charset.Charset` class describes the naming conventions for character encodings as well as the set of standard encodings that must be supported by every implementation of the Java platform.

Since:
1.0

Related Packages

Module	Package	Description
java.base	**java.lang.annotation**	Provides library support for the Java programming language annotation facility.
java.base	**java.lang.constant**	Classes and interfaces to represent *nominal descriptors* for run-time entities such as classes or method handles, and classfile entities such as constant pool entries or `invokedynamic` call sites.
java.instrument	**java.lang.instrument**	Provides services that allow Java programming language agents to instrument programs running on the JVM.
java.base	**java.lang.invoke**	The `java.lang.invoke` package provides low-level primitives for interacting with the Java Virtual Machine.
java.management	**java.lang.management**	Provides the management interfaces for monitoring and management of the Java virtual machine and other components in the Java runtime.
java.base	**java.lang.module**	Classes to support module descriptors and creating configurations of modules by means of resolution and service binding.
java.base	**java.lang.ref**	Provides reference-object classes, which support a limited degree of interaction with the garbage collector.
java.base	**java.lang.reflect**	Provides classes and interfaces for obtaining reflective information about classes and objects.
java.base	**java.lang.runtime**	The `java.lang.runtime` package provides low-level runtime support for the Java language.

All Classes and Interfaces | Interfaces | Classes | Enum Classes | Exception Classes | Annotation Interfaces

Class	Description
AbstractMethodError	Thrown when an application tries to call an abstract method.
Appendable	An object to which `char` sequences and values can be appended.
ArithmeticException	Thrown when an exceptional arithmetic condition has occurred.
ArrayIndexOutOfBoundsException	Thrown to indicate that an array has been accessed with an illegal index.
ArrayStoreException	Thrown to indicate that an attempt has been made to store the wrong type of object into an array of objects.
AssertionError	Thrown to indicate that an assertion has failed.
AutoCloseable	An object that may hold resources (such as file or socket handles) until it is closed.

Boolean	The Boolean class wraps a value of the primitive type `boolean` in an object.
BootstrapMethodError	Thrown to indicate that an `invokedynamic` instruction or a dynamic constant failed to resolve its bootstrap method and arguments, or for `invokedynamic` instruction the bootstrap method has failed to provide a call site with a target of the correct method type, or for a dynamic constant the bootstrap method has failed to provide a constant value of the required type.
Byte	The `Byte` class wraps a value of primitive type `byte` in an object.
Character	The `Character` class wraps a value of the primitive type `char` in an object.
Character.Subset	Instances of this class represent particular subsets of the Unicode character set.
Character.UnicodeBlock	A family of character subsets representing the character blocks in the Unicode specification.
Character.UnicodeScript	A family of character subsets representing the character scripts defined in the *Unicode Standard Annex #24: Script Names*.
CharSequence	A `CharSequence` is a readable sequence of `char` values.
Class<T>	Instances of the class `Class` represent classes and interfaces in a running Java application.
ClassCastException	Thrown to indicate that the code has attempted to cast an object to a subclass of which it is not an instance.
ClassCircularityError	Thrown when the Java Virtual Machine detects a circularity in the superclass hierarchy of a class being loaded.
ClassFormatError	Thrown when the Java Virtual Machine attempts to read a class file and determines that the file is malformed or otherwise cannot be interpreted as a class file.
ClassLoader	A class loader is an object that is responsible for loading classes.
ClassNotFoundException	Thrown when an application tries to load in a class through its string name using: The `forName` method in class `Class`.
ClassValue<T>	Lazily associate a computed value with (potentially) every type.
Cloneable	A class implements the `Cloneable` interface to indicate to the `Object.clone()` method that it is legal for that method to make a field-for-field copy of instances of that class.
CloneNotSupportedException	Thrown to indicate that the `clone` method in class `Object` has been called to clone an object, but that the object's class does not implement the `Cloneable` interface.
Comparable<T>	This interface imposes a total ordering on the objects of each class that implements it.
Compiler	Deprecated, for removal: This API element is subject to removal in a future version. *JIT compilers and their technologies vary too widely to be controlled effectively by a standardized interface.*
Deprecated	A program element annotated `@Deprecated` is one that programmers are discouraged from using.
Double	The `Double` class wraps a value of the primitive type `double` in an object.
Enum<E extends **Enum**<E>>	This is the common base class of all Java language enumeration classes.
Enum.EnumDesc<E extends **Enum**<E>>	A nominal descriptor for an `enum` constant.
EnumConstantNotPresentException	Thrown when an application tries to access an enum constant by name and the enum type contains no constant with the specified name.
Error	An `Error` is a subclass of `Throwable` that indicates serious problems that a reasonable application should not try to catch.
Exception	The class `Exception` and its subclasses are a form of `Throwable` that indicates conditions that a reasonable application might want to catch.
ExceptionInInitializerError	Signals that an unexpected exception has occurred in a static initializer.
Float	The `Float` class wraps a value of primitive type `float` in an object.
FunctionalInterface	An informative annotation type used to indicate that an interface type declaration is intended to be a *functional interface* as defined by the Java Language Specification.
IllegalAccessError	Thrown if an application attempts to access or modify a field, or to call a method that it does not have access to.
IllegalAccessException	An IllegalAccessException is thrown when an application tries to reflectively create an instance (other than an array), set or get a field, or invoke a method, but the currently executing method does not have access to the definition of the specified class, field, method or constructor.

IllegalArgumentException	Thrown to indicate that a method has been passed an illegal or inappropriate argument.
IllegalCallerException	Thrown to indicate that a method has been called by an inappropriate caller.
IllegalMonitorStateException	Thrown to indicate that a thread has attempted to wait on an object's monitor or to notify other threads waiting on an object's monitor without owning the specified monitor.
IllegalStateException	Signals that a method has been invoked at an illegal or inappropriate time.
IllegalThreadStateException	Thrown to indicate that a thread is not in an appropriate state for the requested operation.
IncompatibleClassChangeError	Thrown when an incompatible class change has occurred to some class definition.
IndexOutOfBoundsException	Thrown to indicate that an index of some sort (such as to an array, to a string, or to a vector) is out of range.
InheritableThreadLocal<T>	This class extends `ThreadLocal` to provide inheritance of values from parent thread to child thread: when a child thread is created, the child receives initial values for all inheritable thread-local variables for which the parent has values.
InstantiationError	Thrown when an application tries to use the Java `new` construct to instantiate an abstract class or an interface.
InstantiationException	Thrown when an application tries to create an instance of a class using the `newInstance` method in class `Class`, but the specified class object cannot be instantiated.
Integer	The `Integer` class wraps a value of the primitive type `int` in an object.
InternalError	Thrown to indicate some unexpected internal error has occurred in the Java Virtual Machine.
InterruptedException	Thrown when a thread is waiting, sleeping, or otherwise occupied, and the thread is interrupted, either before or during the activity.
Iterable<T>	Implementing this interface allows an object to be the target of the enhanced `for` statement (sometimes called the "for-each loop" statement).
LayerInstantiationException	Thrown when creating a module layer fails.
LinkageError	Subclasses of `LinkageError` indicate that a class has some dependency on another class; however, the latter class has incompatibly changed after the compilation of the former class.
Long	The `Long` class wraps a value of the primitive type `long` in an object.
Math	The class `Math` contains methods for performing basic numeric operations such as the elementary exponential, logarithm, square root, and trigonometric functions.
Module	Represents a run-time module, either `named` or unnamed.
ModuleLayer	A layer of modules in the Java virtual machine.
ModuleLayer.Controller	Controls a module layer.
NegativeArraySizeException	Thrown if an application tries to create an array with negative size.
NoClassDefFoundError	Thrown if the Java Virtual Machine or a `ClassLoader` instance tries to load in the definition of a class (as part of a normal method call or as part of creating a new instance using the `new` expression) and no definition of the class could be found.
NoSuchFieldError	Thrown if an application tries to access or modify a specified field of an object, and that object no longer has that field.
NoSuchFieldException	Signals that the class doesn't have a field of a specified name.
NoSuchMethodError	Thrown if an application tries to call a specified method of a class (either static or instance), and that class no longer has a definition of that method.
NoSuchMethodException	Thrown when a particular method cannot be found.
NullPointerException	Thrown when an application attempts to use `null` in a case where an object is required.
Number	The abstract class `Number` is the superclass of platform classes representing numeric values that are convertible to the primitive types `byte`, `double`, `float`, `int`, `long`, and `short`.
NumberFormatException	Thrown to indicate that the application has attempted to convert a string to one of the numeric types, but that the string does not have the appropriate format.
Object	Class `Object` is the root of the class hierarchy.

OutOfMemoryError	Thrown when the Java Virtual Machine cannot allocate an object because it is out of memory, and no more memory could be made available by the garbage collector.
Override	Indicates that a method declaration is intended to override a method declaration in a supertype.
Package	Represents metadata about a run-time package associated with a class loader.
Process	`Process` provides control of native processes started by ProcessBuilder.start and Runtime.exec.
ProcessBuilder	This class is used to create operating system processes.
ProcessBuilder.Redirect	Represents a source of subprocess input or a destination of subprocess output.
ProcessBuilder.Redirect.Type	The type of a `ProcessBuilder.Redirect`.
ProcessHandle	ProcessHandle identifies and provides control of native processes.
ProcessHandle.Info	Information snapshot about the process.
Readable	A `Readable` is a source of characters.
Record	This is the common base class of all Java language record classes.
ReflectiveOperationException	Common superclass of exceptions thrown by reflective operations in core reflection.
Runnable	The `Runnable` interface should be implemented by any class whose instances are intended to be executed by a thread.
Runtime	Every Java application has a single instance of class `Runtime` that allows the application to interface with the environment in which the application is running.
Runtime.Version	A representation of a version string for an implementation of the Java SE Platform.
RuntimeException	`RuntimeException` is the superclass of those exceptions that can be thrown during the normal operation of the Java Virtual Machine.
RuntimePermission	This class is for runtime permissions.
SafeVarargs	A programmer assertion that the body of the annotated method or constructor does not perform potentially unsafe operations on its varargs parameter.
SecurityException	Thrown by the security manager to indicate a security violation.
SecurityManager	Deprecated, for removal: This API element is subject to removal in a future version. *The Security Manager is deprecated and subject to removal in a future release.*
Short	The `Short` class wraps a value of primitive type `short` in an object.
StackOverflowError	Thrown when a stack overflow occurs because an application recurses too deeply.
StackTraceElement	An element in a stack trace, as returned by `Throwable.getStackTrace()`.
StackWalker	A stack walker.
StackWalker.Option	Stack walker option to configure the stack frame information obtained by a `StackWalker`.
StackWalker.StackFrame	A `StackFrame` object represents a method invocation returned by `StackWalker`.
StrictMath	The class `StrictMath` contains methods for performing basic numeric operations such as the elementary exponential, logarithm, square root, and trigonometric functions.
String	The `String` class represents character strings.
StringBuffer	A thread-safe, mutable sequence of characters.
StringBuilder	A mutable sequence of characters.
StringIndexOutOfBoundsException	Thrown by `String` methods to indicate that an index is either negative or greater than the size of the string.
SuppressWarnings	Indicates that the named compiler warnings should be suppressed in the annotated element (and in all program elements contained in the annotated element).
System	The `System` class contains several useful class fields and methods.
System.Logger	`System.Logger` instances log messages that will be routed to the underlying logging framework the `LoggerFinder` uses.
System.Logger.Level	System loggers levels.
System.LoggerFinder	The `LoggerFinder` service is responsible for creating, managing, and configuring loggers to the underlying framework it uses.

Thread	A *thread* is a thread of execution in a program.
Thread.State	A thread state.
Thread.UncaughtExceptionHandler	Interface for handlers invoked when a `Thread` abruptly terminates due to an uncaught exception.
ThreadDeath	An instance of `ThreadDeath` is thrown in the victim thread when the (deprecated) `Thread.stop()` method is invoked.
ThreadGroup	A thread group represents a set of threads.
ThreadLocal<T>	This class provides thread-local variables.
Throwable	The `Throwable` class is the superclass of all errors and exceptions in the Java language.
TypeNotPresentException	Thrown when an application tries to access a type using a string representing the type's name, but no definition for the type with the specified name can be found.
UnknownError	Thrown when an unknown but serious exception has occurred in the Java Virtual Machine.
UnsatisfiedLinkError	Thrown if the Java Virtual Machine cannot find an appropriate native-language definition of a method declared `native`.
UnsupportedClassVersionError	Thrown when the Java Virtual Machine attempts to read a class file and determines that the major and minor version numbers in the file are not supported.
UnsupportedOperationException	Thrown to indicate that the requested operation is not supported.
VerifyError	Thrown when the "verifier" detects that a class file, though well formed, contains some sort of internal inconsistency or security problem.
VirtualMachineError	Thrown to indicate that the Java Virtual Machine is broken or has run out of resources necessary for it to continue operating.
Void	The `Void` class is an uninstantiable placeholder class to hold a reference to the `Class` object representing the Java keyword void.