

Documentazione delle librerie Java

L'ultima versione della documentazione delle librerie Java può essere scaricata entrando nella sezione **Documentation** dell'area **Java SE**.

<https://www.oracle.com/technetwork/java/javase/documentation/jdk11-doc-downloads-5097203.html>

Dopo aver accettato il contratto di licenza, per eseguire il download della documentazione, si deve scegliere il file **jdk-11.0.1_doc-all.zip**.

La documentazione contenuta nell'archivio è in formato HTML e può essere letta aprendo il file **index.html** contenuto nella cartella **docs\api**,

The screenshot shows the Java SE 11 API Specification documentation page. The 'Overview' tab is selected, displaying the 'Java Platform, Standard Edition & Java Development Kit Version 11 API Specification'. A search bar on the right contains the text 'Math'. Below the search bar, a list of packages and types is shown, including 'java.lang.Math' and 'java.math.MathContext'. A table of modules is also visible on the left side of the page.

All Modules	Java SE	JDK	Other Modules
Module	Description		
java.base	Defines the foundational APIs of the Java SE Platform.		
java.compiler	Defines the Language Model, Annotation Processing, and Java Compiler API.		
java.datatransfer	Defines the API for transferring data between and within applications.		
java.desktop	Defines the AWT and Swing user interface toolkits, plus APIs for accessibility.		
java.instrument	Defines services that allow agents to instrument programs running on the JVM.		
java.logging	Defines the Java Logging API.		
java.management	Defines the Java Management Extensions (JMX) API.		
java.management.rmi	Defines the RMI connector for the Java Management Extensions (JMX) API.		
java.naming	Defines the Java Naming and Directory Interface (JNDI) API.		
java.net.http	Defines the HTTP Client and WebSocket APIs.		

La stessa documentazione è accessibile anche online all'indirizzo:

<https://docs.oracle.com/en/java/javase/11/docs/api/index.html>

Per cercare la descrizione di una classe, si può utilizzare il campo di ricerca in alto a destra.

Per esempio, per cercare la documentazione della classe **java.lang.Math**, si può scrivere la parola **Math** nel campo di ricerca e, nell'elenco dei risultati, si può fare clic sulla voce **java.lang.Math** presente nel gruppo **Types**.

La pagina di documentazione delle classi contiene le informazioni che possono servire al programmatore per capire le funzionalità offerte dalle classi del JDK e cioè, quali attributi e quali metodi sono accessibili e con quali parametri.

Per ogni classe del JDK, sono riportati i seguenti elementi:

- 1) una descrizione della classe,
- 2) un elenco degli attributi (**Field Summary**),
- 3) un elenco dei metodi (**Method Summary**),
- 4) una descrizione degli attributi (**Field Detail**),
- 5) una descrizione dei metodi (**Method Detail**).

Nel seguito vengono illustrati i precedenti cinque elementi della classe *java.lang.Math*.

Descrizione della classe

Oltre a riportare il nome del package, il nome della classe e la sua gerarchia delle classi, nella descrizione vengono solitamente spiegate le modalità di implementazione e di utilizzo della classe.

java.lang

Class Math

java.lang.Object
java.lang.Math

```
public final class Math
extends Object
```

The class `Math` contains methods for performing basic numeric operations such as the elementary exponential, logarithm, square root, and trigonometric functions.

Unlike some of the numeric methods of class `StrictMath`, all implementations of the equivalent functions of class `Math` are not defined to return the bit-for-bit same results. This relaxation permits better-performing implementations where strict reproducibility is not required.

By default many of the `Math` methods simply call the equivalent method in `StrictMath` for their implementation. Code generators are encouraged to use platform-specific native libraries or microprocessor instructions, where available, to provide higher-performance implementations of `Math` methods. Such higher-performance implementations still must conform to the specification for `Math`.

The quality of implementation specifications concern two properties, accuracy of the returned result and monotonicity of the method. Accuracy of the floating-point `Math` methods is measured in terms of *ulps*, units in the last place. For a given floating-point format, an *ulp* of a specific real number value is the distance between the two floating-point values bracketing that numerical value. When discussing the accuracy of a method as a whole rather than at a specific argument, the number of *ulps* cited is for the worst-case error at any argument. If a method always has an error less than 0.5 *ulps*, the method always returns the floating-point number nearest the exact result; such a method is *correctly rounded*. A *correctly rounded* method is generally the best a floating-point approximation can be; however, it is impractical for many floating-point methods to be *correctly rounded*. Instead, for the `Math` class, a larger error bound of 1 or 2 *ulps* is allowed for certain methods. Informally, with a 1 *ulp* error bound, when the exact result is a representable number, the exact result should be returned as the computed result; otherwise, either of the two floating-point values which bracket the exact result may be returned. For exact results large in magnitude, one of the endpoints of the bracket may be infinite. Besides accuracy at individual arguments, maintaining proper relations between the method at different arguments is also important. Therefore, most methods with more than 0.5 *ulp* errors are required to be *semi-monotonic*: whenever the mathematical function is non-decreasing, so is the floating-point approximation, likewise, whenever the mathematical function is non-increasing, so is the floating-point approximation. Not all approximations that have 1 *ulp* accuracy will automatically meet the monotonicity requirements.

Since:

JDK1.0

Field Summary

Contiene l'elenco dei nomi degli attributi della classe, con l'indicazione del tipo e del livello di visibilità, oltre ad una breve descrizione.

Method Summary

Contiene l'elenco dei nomi dei metodi della classe, con il tipo di valore di ritorno e una descrizione. Le classi che prevedono i metodi costruttori hanno, nella documentazione delle API, un ulteriore riquadro apposito (*Constructor Summary*).

Field Summary

Fields

Modifier and Type	Field and Description
static double	E The double value that is closer than any other to e , the base of the natural logarithms.
static double	PI The double value that is closer than any other to π , the ratio of the circumference of a circle to its diameter.

Method Summary

Methods

Modifier and Type	Method and Description
static double	abs(double a) Returns the absolute value of a double value.
static float	abs(float a) Returns the absolute value of a float value.
static int	abs(int a) Returns the absolute value of an int value.
static long	abs(long a) Returns the absolute value of a long value.
static double	acos(double a) Returns the arc cosine of a value; the returned angle is in the range 0.0 through π .
static double	asin(double a) Returns the arc sine of a value; the returned angle is in the range $-\pi/2$ through $\pi/2$.
static double	atan(double a)

Field Detail

Rispetto al *Field Summary*, in questa zona viene riportata una descrizione più dettagliata degli attributi, che serve al programmatore per capire quale valore contengono.

Field Detail

E

```
public static final double E
```

The double value that is closer than any other to e , the base of the natural logarithms.

See Also:

[Constant Field Values](#)

PI

```
public static final double PI
```

The double value that is closer than any other to π , the ratio of the circumference of a circle to its diameter.

See Also:

[Constant Field Values](#)

Method Detail

Rispetto al *Method Summary*, in questa zona viene riportata una descrizione più dettagliata dei metodi, che illustra al programmatore il modo corretto di richiamare il metodo e quale valore di ritorno si può aspettare di ricevere. Ogni parametro da passare al metodo viene spiegato nel paragrafo **Parameters**. Il significato del valore di ritorno viene descritto nel paragrafo **Returns**. Se il metodo genera delle eccezioni, che devono essere gestite dal programmatore, il loro elenco è riportato nel paragrafo **Throws**.

Le descrizioni di dettaglio dei costruttori, se presenti, sono raggruppate nella zona *Constructor Detail*.

abs

```
public static int abs(int a)
```

Returns the absolute value of an `int` value. If the argument is not negative, the argument is returned. If the argument is negative, the negation of the argument is returned. Note that if the argument is equal to the value of `Integer.MIN_VALUE`, the most negative representable `int` value, the result is that same value, which is negative.

Parameters:

- `a` - the argument whose absolute value is to be determined

Returns:

- the absolute value of the argument.

abs

```
public static long abs(long a)
```

Returns the absolute value of a `long` value. If the argument is not negative, the argument is returned. If the argument is negative, the negation of the argument is returned. Note that if the argument is equal to the value of `Long.MIN_VALUE`, the most negative representable `long` value, the result is that same value, which is negative.

Parameters:

- `a` - the argument whose absolute value is to be determined

Returns:

- the absolute value of the argument.