Important Methods in Java

Important Methods of various Classes in Java with description.

NOTE : Methods mentioned below are only important ones, and not all methods.

Methods of Number Class :

Sr.No.	Method	Description
1	xxxValue()	Converts the value of *this* Number object to the xxx data type and returns it.
2	compareTo()	Compares *this* Number object to the argument.
3	equals()	Determines whether *this* number object is equal to the argument.
4	valueOf()	Returns an Integer object holding the value of the specified primitive.
5	toString()	Returns a String object representing the value of a specified int or Integer.
6	parseInt()	This method is used to get the primitive data type of a certain String.
7	abs()	Returns the absolute value of the argument.
8	ceil()	Returns the smallest integer that is greater than or equal to the argument. Returned as a double.
9	floor()	Returns the largest integer that is less than or equal to the argument. Returned as a double.
10	rint()	Returns the integer that is closest in value to the argument. Returned as a double.
11	round()	Returns the closest long or int, as indicated by the method's return type to the argument.
12	min()	Returns the smaller of the two arguments.
13	max()	Returns the larger of the two arguments.
14	exp()	Returns the base of the natural logarithms, e, to the power of the argument.
15	log()	Returns the natural logarithm of the argument.

16	pow()	Returns the value of the first argument raised to the power of the second argument.
17	sqrt()	Returns the square root of the argument.
18	sin()	Returns the sine of the specified double value.
19	cos()	Returns the cosine of the specified double value.
20	tan()	Returns the tangent of the specified double value.
21	asin()	Returns the arcsine of the specified double value.
22	acos()	Returns the arccosine of the specified double value.
23	atan()	Returns the arctangent of the specified double value.
24	atan2()	Converts rectangular coordinates (x, y) to polar coordinate (r, theta) and returns theta.
25	toDegrees()	Converts the argument to degrees.
26	toRadians()	Converts the argument to radians.
27	random()	Returns a random number.

NOTE : Most of above methods belongs to java.lang.Math and java.lang.Number

Methods of Character Class :

Sr.No.	Method	Description
1	isLetter()	Determines whether the specified char value is a letter.
2	isDigit()	Determines whether the specified char value is a digit.
3	isWhitespace()	Determines whether the specified char value is white space.
4	isUpperCase()	Determines whether the specified char value is uppercase.
5	isLowerCase()	Determines whether the specified char value is lowercase.
6	toUpperCase()	Returns the uppercase form of the specified char value.
7	toLowerCase()	Returns the lowercase form of the specified char value.
8	toString()	Returns a String object representing the specified character value that is, a 1-character string.

NOTE : All the above methods belongs to java.lang.Character

Methods of String Class :

No.	Method	Description
1	char charAt(int index)	returns char value for the particular index
2	int length()	returns string length
3	static String format(String format, Object... args)	returns formatted string
4	static String format(Locale l, String format, Object... args)	returns formatted string with given locale
5	String substring(int beginIndex)	returns substring for given begin index
6	String substring(int beginIndex, int endIndex)	returns substring for given begin index and end index
7	boolean contains(CharSeq s)	returns true or false after matching the sequence of char value
8	static String join(CharSeq delim, CharSeq... elem)	returns a joined string
9	static String join(CharSeq delim, Iterable<? extends CharSeq> elem)	returns a joined string
10	boolean equals(Object another)	checks the equality of string with object
11	boolean isEmpty()	checks if string is empty
12	String concat(String str)	concatinates specified string
13	String replace(char old, char new)	replaces all occurrences of specified char value
14	String replace(CharSeq old, CharSeq new)	replaces all occurrences of specified CharSeq
15	static String equalsIgnoreCase(String another)	compares another string. It doesn't check case.
16	String[] split(String regex)	returns splitted string matching regex
17	String[] split(String regex, int limit)	returns splitted string matching regex and limit

18	String intern()	returns interned string
19	int indexOf(int ch)	returns specified char value index
20	int indexOf(int ch, int fromIndex)	returns specified char value index starting with given index
21	int indexOf(String substring)	returns specified substring index
22	int indexOf(String substring, int fromIndex)	returns specified substring index starting with given index
23	String toLowerCase()	returns string in lowercase.
24	String toLowerCase(Locale l)	returns string in lowercase using specified locale.
25	String toUpperCase()	returns string in uppercase.
26	String toUpperCase(Locale l)	returns string in uppercase using specified locale.
27	String trim()	removes beginning and ending spaces of this string.
28	static String valueOf(int value)	converts given type into string. It is overloaded.

NOTE : CharSeq - CharSequence, delim - delimeter, elem - elements

Methods of Array Class :

Sr.No.	Methods & Description
1	**public static String toString(int[] a)** The string representation consists of a list of the array's elements, enclosed in square brackets ("[]"). Adjacent elements are separated by the characters a comma followed by a space. Elements are converted to strings as by String.valueOf(int). Returns "null" if a is null.
2	**public static int[] copyOf(int[] original, int newLength)** Copies the specified array and length. It truncates the array if provided length is smaller and if provided length is longer, it adds default value of particular type to extra elements.
3	**public static int[] copyOfRange(int[] original, int from, int to)** Copies the specified range of the specified array into a new array. The

	initial index of the range (from) must lie between zero and original.length, inclusive.
4	**public static void fill(int[] a, int val)** Assigns the specified int value to each element of the specified array of ints.
5	**public static void fill(int[] a, int fromIndex, int toIndex, int val)** Fills elements of the specified array with the specified value from the fromIndex element, but not including the toIndex element.
6	**public static boolean equals(int[] a, int[] a2)** Returns true if the two specified arrays of longs are equal to one another. Two arrays are considered equal if both arrays contain the same number of elements, and all corresponding pairs of elements in the two arrays are equal. This returns true if the two arrays are equal.
7	**public static void sort(int[] a)** Sorts the specified array of objects into an ascending order, according to the natural ordering of its elements.
8	**public static void sort(int[] a, int fromIndex, int toIndex)** If we wish to sort a specified range of the array into ascending order. we can use this. The range to be sorted extends from the index fromIndex, inclusive, to the index toIndex, exclusive. If fromIndex == toIndex, the range to be sorted is empty.
9	**public static int binarySearch(int[] a, int key)** Searches the array of ints for the specified value using the binary search algorithm. The array must be sorted prior to making this call. This returns index of the search key, if it is contained in the list; otherwise, it returns (– (insertion point + 1)).
10	**public static List asList(int[] a)** Takes an array and creates a wrapper (nothing is copied) that implements List, which makes the original array available as a list. Operations on the list wrapper are propagated to the original array. List operations like adding/removing elements aren't allowed, you can only read/overwrite the elements.
11	**static int hashCode(int[] a)** This method returns a hash code based on the contents of the specified array.

NOTE : All the above methods belongs to java.util.Array

All the methods mentioned above with int[] examples can be also used by all other primitive, wrapper and object data types (Eg - byte, short, long, Byte, Short, Int, etc)