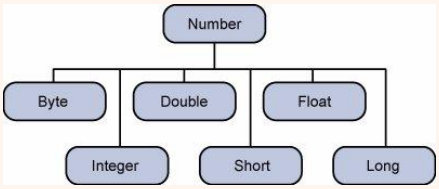
Java Cheat Sheet

by yunshu (xys) via cheatography.com/130138/cs/25835/

// import java.lang.Number;



Number Methods

Integer a = 20;	
a.byteValue(); a.shortValue(); a.intValue(); a.longValue(); a.floatValue(); a.doubleValue()	The method converts the value of the Number Object that invokes the method to the primitive data type that is returned from the method.
a.compareTopublic int compareTo(NumberSubClass same_type)	Compares this Number object to the argument.
a. equals(Object o)	Determines whether this number object is equal to the argument.
Number.valueOf(int i/String s/ String s, int radix)	The valueOf method returns the Integer holding the value of the argument passed.
a. toString()	string representation
Math.abs()	absolute value
Math.ceil()	The method ceil gives the smallest integer that is greater than or equal to the argument.
Math.rint()	Returns the integer that is closest in value to the argument. Returned as a double.
Math.round()	Returns the closest long or int, as indicated by the method's return type to the argument.
Math.min(a,b)	Returns the smaller of the two arguments.
Math.max(a,b)	Returns the larger of the two arguments.
Math.exp()	Returns the base of the natural logarithms, e, to the power of the argument.
Math.log()	Returns the natural logarithm of the argument.

Number Methods (cont)

Math.pow(a,b)	Returns the value of the first argument raised to the power of the second argument.
Math.sqrt()	Returns the square root of the argument.
Math.sin()	Returns the sine of the specified double value.
Math.cos(); Math.tan(); Math.asin(); Math.acos(); Math.atan(); Math.atan2();	
Math.toDegree()	Converts the argument to degrees.
Math.toRadians()	Converts the argument to radians.
Math.random()	Returns a random number.

StringBuilder

```
String Builder sb = new String Bui lder(capacity or
```

```
sb.app end (obj);
```

```
sb.cap aci ty();
```

```
sb.cha rAt (in dex);
```

```
sb.del ete (st art ,en d_e xcl usive);
```

```
sb.del ete Cha rAt (in dex);
```

```
getCha rs(int srcBegin, int srcEnd, char[] dst, int
;
```

```
sb.ind exO f(s tring,from);
```

```
sb.ins ert (in dex ,obj)
```

```
sb.rep lac e(int start, int end, String str);
```

Cheatography

Java Cheat Sheet

by yunshu (xys) via cheatography.com/130138/cs/25835/

StringBuilder (cont)

`sb.rev erse()`	Causes this character sequence to be replaced by the reverse of the sequence.
`sb.set Cha rAt(int index, char ch);`	The character at the specified index is set to ch.
`sb.sub Seq uen ce(int start, int end)`	Returns a new character sequence that is a subsequence of this sequence.
`sb.sub str ing(int start,end)`	Returns a new String that contains a subsequence of characters currently contained in this character sequence.
`sb.toS tring()`	Returns a string representing the data in this sequence.

Thread

Thread (cont)

```
    new Runnab le(){
public void run(){
System.ou t.p rin tln ("Ru nnable runnin g");}
}`
```

3. Java Lambda Implementation of Runnable

```
Runnable runnable = () -> { System.ou t.p rin tln ("
; };
```

Starting a Thread With a Runnable

```
Runnable runnable = new MyRunn able(); // or an anonymous
Thread thread = new Thread (ru nna ble);
thread.st art();
```

Synchronization

1. Synchronized Instance Methods

```
public synchr onized void synchr oni sed Cal cul a
te()
{setSu m(g etSum() + 1);}
```

2. Synchronized Static Methods

```
public static synchr onized void syncSt ati cCa lc
u late()
 {staticSum = staticSum + 1;}
```

3. Synchronized Blocks Within Methods

```
public void perfor mSy nch ron ise dTask() {
//unsy nch ronized part
// ...
   synchr onized (this/obj) {
setCou nt( get Cou nt( )+1);}}
```

import java.time

Creating a thread in Java is done like this:

```
  Thread thread = new Thread();
```

To start the thread

```
  thread.st art();
```

To join the thread

```
thread.join()
```

Thread subclass

```
public class MyThread extends Thread {
public void run(){ Sys tem.ou t.p rin tln ("My Thread runni
n g"); }
}
MyThread myThread = new MyThre ad();
 myTrea d.s tart();
```

OR:

```
Thread thread = new Thread(){
public void run(){
 System.ou t.p rin tln ("Thread Runnin g");}
}`
thread.st art();
```

Runnable Interface Implementation

```
public interface Runnable() {public void run();}
```

To implement a Runnable

1. Java Class Implements Runnable

```
public class MyRunnable implements Runnable {
   public void run(){
 System.ou t.p rin tln ("My Run nable runnin g");  }
}
```

2. Anonymous Implementation of Runnable

```
Runnable myRunnable =
```

```
import java.t ime.Lo cal Date;
```

```
LocalDate myObj = LocalD ate.now();  System.ou t.p r
```

```
import java.t ime.Lo cal Time;
```

```
 LocalTime myObj = LocalT ime.now();     System.ou t
```

```
import java.t ime.Lo cal Dat eTime;
```

```
LocalD ateTime myDateObj = LocalD ate Tim e.n ow();
 Dat eObj);
```

source: https://www.w3schools.com/java/java_date.asp

```
import java.util.HashMap;
```

`public void clear()`	This method removes all of the mappings from this map.
`public boolean contai nsK ey( Object key)`	This method returns true if this map contains a mapping for the specified key.
`public boolean contai nsV alu e(O bject va lue)`	This method returns true if this map maps one or more keys to the specified value.
`public Set<Ma p.E ntr y<K ,V>> entrySet()`	This method returns a Set view of the mappings contained in this map.

import java.util.HashMap; (cont)

`public V get(Object key)`	This method returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.
`public boolean isEmpty()`	This method returns true if this map contains no key-value mapping.
`public Set<K> keySet()`	This method returns a Set view of the keys contained in this map.
`public V put(K key, V value)`	This method associates the specified value with the specified key in this map.

import java.util.PriorityQueue; (cont)

`public E peek()`	This method retrieves and removes the head of this qu
`public boolean remove (Object o)`	
`public int size()`	

`public void putAll (Ma p<? extends K,? extends V> m)`	This method copies all of the mappings from the specified map to this map.
`public V remove (Object key)`	This method removes the mapping for the specified key from this map if present.
`public int size()`	This method returns the number of key-value mappings in this map.
`public Collec tio n<V> values()`	This method returns a Collection view of the values contained in this map.

`public Object[] toArray()`
`public <T> T[] toArra y(T[] a)`

The java.util.PriorityQueue class is an unbounded priority queue based on a priority heap.Following are the important points about PriorityQueue

- The elements of the priority queue are ordered according to their natural ordering, or by a Comparator provided at queue construction time, depending on which constructor is used.

- A priority queue does not permit null elements.

- A priority queue relying on natural ordering also does not permit insertion of non-comparable objects.

import java.util.PriorityQueue;

import java.util.Stream;

https://developer.ibm.com/zh/languages/java/articles/j-lo-java8str-eamapi/

bye

Character class

Character c = '1';

Character.is-Letter()	The method determines whether the specified char value is a letter.

`public boolean add(E e)`	This method inserts the specified element into this priority queue.
`public boolean offer(E e)`	This method inserts the specified element into this priority queue.
`public void clear()`	This method removes all of the elements from this priority queue.
`public boolean contai ns( Object o)`	This method returns true if this queue contains the specified element.
`public Iterat or< E> iterator()`	This method returns an iterator over the elements in this queue.

Character class (cont)

Character.isDigit()	The method determines whether the specified char value is a digit.
Character.isLetterOrDigit(char ch)	Determines if the specified character is a letter or digit.
Character.digit(char ch, int radix)	Returns the numeric value of the character ch in the specified radix.
Character.forDigit(int digit, int radix)	Determines the character representation for a specific digit in the specified radix.
Character.isWhitespace()	Determines whether the specified char value is white space.
Character.isUpperCase()	Determines whether the specified char value is uppercase.
Character.toUpperCase()	Determines whether the specified char value is lowercase.
Character.toLowerCase()	Returns the lowercase form of the specified char value.
Character.toString()	Returns a String object representing the specified character value that is, a one-character string.
Character.codePointAt(seq,index)	return the ASCII value of char at index in seq
c.charValue();	return the ASCII value of this char.
Character.getNumericValue(char ch)	Returns the int value that the specified Unicode character represents.
Character.isSpaceChar(char ch)	Determines if the specified character is a Unicode space character.
Character.isWhitespace(char ch)	Determines if the specified character is white space according to Java.

String class

String s = "123"

s.charAt()	This method returns the character located at the String's specified index. The string indexes start from zero.
s.compareTo()	This method compares two strings lexicographically.
s.compareToIgnoreCase()	Compares two strings lexicographically, ignoring case differences.
s.concat()	This method appends one String to the end of another. The method returns a String with the value of the String passed into the method, appended to the end of the String, used to invoke this method.
s.contentEqual(StringBuilder sb)	This method returns true if and only if this String represents the same sequence of characters as specified in StringBuffer.
s.endsWith()	This method tests if this string ends with the specified suffix.
s.equals()	This method compares this string to the specified object. The result is true if and only if the argument is not null and is a String object that represents the same sequence of characters as this object.
s.equalsIgnoreCase()	This method compares this String to another String, ignoring case considerations. Two strings are considered equal ignoring case, if they are of the same length, and corresponding characters in the two strings are equal ignoring case.

Cheatography

Java Cheat Sheet

by yunshu (xys) via cheatography.com/130138/cs/25835/

String class (cont)

s.getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin)	This method copies characters from this string into the destination character array.
s.indexOf(int ch)	Returns the index within this string of the first occurrence of the specified character.
s.indexOf(int ch, int fromIndex)	Returns the index within this string of the first occurrence of the specified character, starting the search at the specified index.
s.lastIndexOf(int ch)	Returns the index within this string of the last occurrence of the specified character.
s.lastIndexOf(int ch, int fromIndex)	Returns the index within this string of the last occurrence of the specified character, searching backward starting at the specified index.
s.length()	Returns the length of this string.
s.matches(regex)	Tells whether or not this string matches the given regular expression.
s.regionMatches(boolean ignoreCase,int this_offset,String other,int that_offset,int len))	This method has two variants which can be used to test if two string regions are equal.
s.replace(old,new)	Returns a new string resulting from replacing all occurrences of oldChar in this string with newChar.
s.split(del,*limit*)	split the string
s.subSequence(beg,end)	Returns a new character sequence that is a subsequence of this sequence.
s.substring(beg,*end*)	Returns a new string that is a substring of this string.
s.toCharArray()	Converts this string to a new character array.

String class (cont)

s.toLowerCase()	Converts all of the characters in this String to lower case using the rules of the default locale.
s.toUpperCase	Converts all of the characters in this String to upper case using the rules of the default locale.
s.trim()	Returns a copy of the string, with leading and trailing whitespace omitted.

import java.util.ArrayList;

```
ArrayL ist <St rin g> cars = new ArrayL ist <St r in g>();
cars.g et( index);
cars.s et(0, " Ope l");
cars.r emo ve(0);
cars.c lear();
cars.s ize();
import java.u til.Co lle ctions;
Collec tio ns.s or t(m yNu mbers);
```

import java.util.Arrays;

```
public static <T> List<T> asList (T... a)
```

```
public static int binary Sea rch (arr, fromInd,toInd,
```

```
public static <T> T[] copyOf(T[] original, int newLe
```

```
public static T[] copyOf Ran ge( ori gin alArr, int
 to)
```

```
public static boolean deepEq ual s(O bject[] a1, Obj
```

Cheatography

Java Cheat Sheet

by yunshu (xys) via cheatography.com/130138/cs/25835/

import java.util.Arrays; (cont)

`public static boolean equals (Ob ject[] a, Object[] a2)`	This method returns true if the two specified arrays of Objects are equal to one another.
`public static void fill(O bject[] a, Object val)`	This method assigns the specified Object reference to each element of the specified array of Objects.
`public static void fill(o bject[] a, int fromIndex, int toIndex, object val )`	This method assigns the specified Object reference to each element of the specified range of the specified array of Objects.

import java.util.Collections; (cont)

`public static <T> void fill(L ist <? super T> list, T`	
`public static <T> void fill(L ist <? super T> list, T`	
`public static int indexO fSu bLi st( Lis t<?> source,`	

`public static void sort(O bject[] a)`	This method sorts the specified array of objects into ascending order, according to the natural ordering of its elements.
`public static void sort(O bject[] a, int fromIndex, int toIndex)`	This method sorts the specified range of the specified array of objects into ascending order, according to the natural ordering of its elements.
`public static String toStri ng( Obj ect[] a)`	This method returns a string represent-ation of the contents of the specified array of ints.

```
import java.util.Collections;
```

```
public static int lastIndex OfS ubL ist (Li st< ?> s
```

```
public static <T extends Object & Compar abl e<? supe
oll)
```

```
public static <T extends Object & Compar abl e<? supe
oll)
```

```
public static <T> boolean replac eAl l(L ist <T> list
```

`public static <T> boolean addAll (Co lle cti on<? super T> c, T... a)`	This method adds all of the specified elements to the specified collec-tion.
`public static <T> int binary Sea rch (Li st<? extends Compar abl e<? super T>> list, T key)`	This method searches the specified list for the specified object using the binary search algorithm.
`public static <T> void copy(L ist <? super T> dest,L ist <? extends T> src)`	This method copies all of the elements from one list into another.
`public static boolean disjoi nt( Col lec tio n<?> c1,Col lec tio n<?> c2)`	This method returns true if the two specified collec-tions have no elements in common.

public static void shuffl e(L ist <? list)
public static void rotate (Li st< ?> list, int distanc
public static <T extends Compar abl e<? super T>> voi

Java Cheat Sheet

by yunshu (xys) via cheatography.com/130138/cs/25835/

import java.util.Collections; (cont)

`public static void swap(L ist <?> list,int i,int j)`	This method swaps the elements at the specified positions in the specified list.

import java.util.HashSet;

`public boolean add(E e)`	This method adds the specified element to this set if it is not already present.
`public void clear()`	This method removes all of the elements from this set.
`public boolean contai ns( Object o)`	This method returns true if this set contains the specified element.
`public boolean isEmpty()`	This method returns true if this set contains no elements.
`public Iterat or< E> iterator()`	This method returns an iterator over the elements in this set.
`public boolean remove (Object o)`	This method removes the specified element from this set if it is present.
`public int size()`	This method returns returns the number of elements in this set(its cardinality).

import java.util.Stack

import java.util.Random

`protected int next(int bits)`	This method generates the next pseudorandom number.
`public boolean nextX()`	X = Boolean/Int/Double/Float/Long uniformly distributed between 0 and 1
`public double nextGa uss ian()`	This method returns the next pseudorandom, Gaussian ("normally") distributed double value with mean 0.0 and standard deviation 1.0 from this random number generator's sequence.
`public int nextIn t(int n)`	This method returns a pseudorandom, uniformly distributed int value between 0 (inclusive) and the specified value (exclusive), drawn from this random number generator's sequence.
`public void setSee d(long seed )`	This method sets the seed of this random number generator using a single long seed.

`public boolean empty()`	This method tests if this stack is empty.
`public Object peek()`	This method looks at the object at the top of this stack without removing it from the stack.
`public Object pop()`	This method removes the object at the top of this stack and returns that object as the value of this function.
`public Object push(O bject item )`	This method pushes an item onto the top of this stack.
`public int search (Object o)`	This method returns the 1-based position where an object is on this stack.