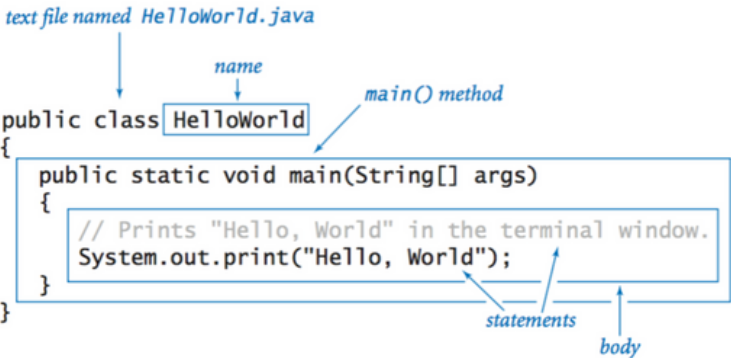
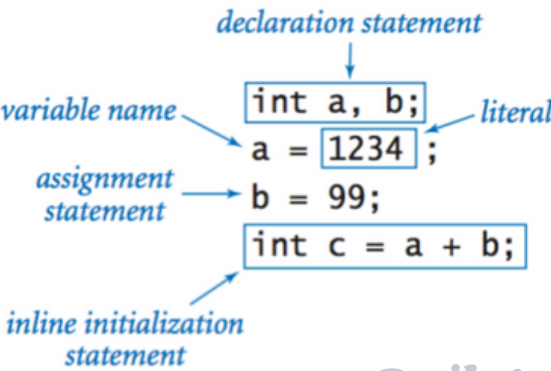


Beginners Java Cheatsheet

Writing your 1st Hello, World program



Declaring & Assigning Variables

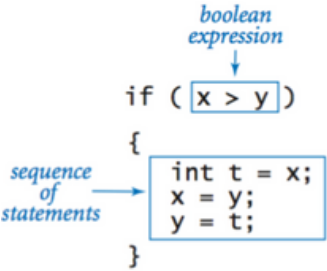


@pilatesdev

Printing to Console

```
void System.out.print(String s)    print s
void System.out.println(String s)  print s, followed by a newline
void System.out.println()          print a newline
```

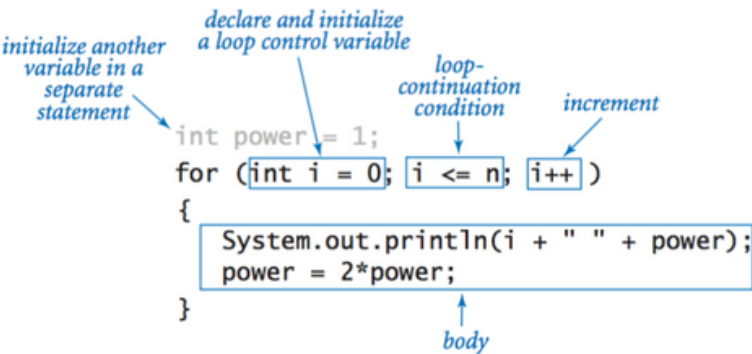
Writing your 1st If Statement



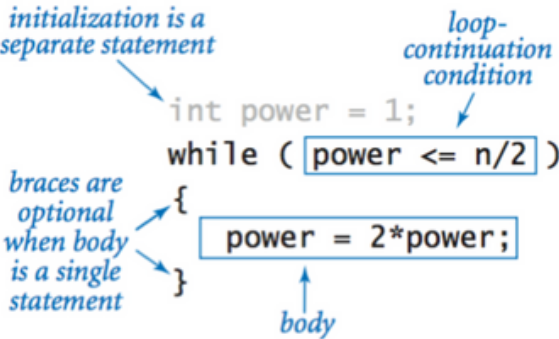
If/ Else Statement

```
if (income < 0) rate = 0.00;
else if (income < 8925) rate = 0.10;
else if (income < 36250) rate = 0.15;
else if (income < 87850) rate = 0.23;
else if (income < 183250) rate = 0.28;
else if (income < 398350) rate = 0.33;
else if (income < 400000) rate = 0.35;
else rate = 0.396;
```

For Loop



While Loop



Break Statement

```
int factor;
for (factor = 2; factor <= n/factor; factor++)
    if (n % factor == 0) break;

if (factor > n/factor)
    System.out.println(n + " is prime");
```

Do- While Loop

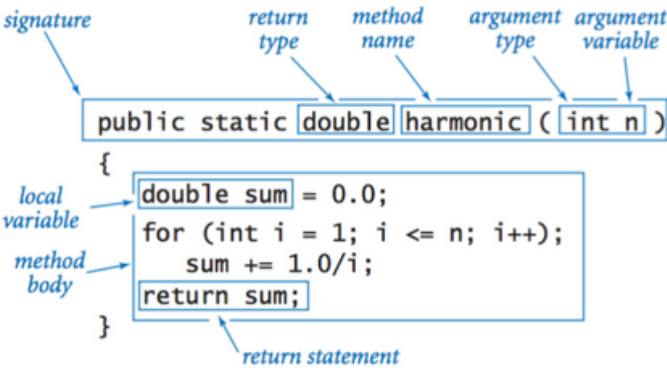
```
int i = 0;
do {
    System.out.println(i);
    i++;
} while (i < 5);
```

@pilatesdev

Writing your 1st Switch/Case Statement

```
switch (day) {
    case 0: System.out.println("Sun"); break;
    case 1: System.out.println("Mon"); break;
    case 2: System.out.println("Tue"); break;
    case 3: System.out.println("Wed"); break;
    case 4: System.out.println("Thu"); break;
    case 5: System.out.println("Fri"); break;
    case 6: System.out.println("Sat"); break;
}
```

Writing your 1st Function



Making your 1st Array

a[0]
a[1]
a[2]
a[3]
a[4]
a[5]
a[6]
a[7]



```
String[] SUITS = { "Clubs", "Diamonds", "Hearts", "Spades" };

String[] RANKS = {
    "2", "3", "4", "5", "6", "7", "8", "9", "10",
    "Jack", "Queen", "King", "Ace"
};
```

@pilatesdev

Beginners Java Cheatsheet

Creating & Using an Object

```
String s;  
s = new String("Hello, World");  
char c = s.charAt(4);
```

declare a variable (object name)

invoke a constructor to create an object

object name

invoke an instance method that operates on the object's value

Creating Instance Variables

```
public class Charge  
{  
    private final double rx, ry;  
    private final double q;  
    :  
    :  
}
```

instance variable declarations

access modifiers

Constructors

```
public Charge ( double x0 , double y0 , double q0 )  
{  
    rx = x0;  
    ry = y0;  
    q = q0;  
}
```

access modifier

no return type

constructor name (same as class name)

parameter variables

signature

instance variable names

body of constructor

@pilatesdev

Creating Instance Methods

```
public double potentialAt (double x, double y)  
{  
    double k = 8.99e09;  
    double dx = x - rx;  
    double dy = y - ry;  
    return k * q / Math.sqrt(dx*dx + dy*dy);  
}
```

access modifier

return type

method name

parameter variables

signature

local variables

paraameter variable name

instance variable name

call on a static method

local variable name

Creating your 1st Class

```
public class Charge  
{  
    private final double rx, ry;  
    private final double q;  
    public Charge(double x0, double y0, double q0)  
    { rx = x0; ry = y0; q = q0; }  
    public double potentialAt(double x, double y)  
    {  
        double k = 8.99e09;  
        double dx = x - rx;  
        double dy = y - ry;  
        return k * q / Math.sqrt(dx*dx + dy*dy);  
    }  
    public String toString()  
    { return q + " at " + "(" + rx + ", " + ry + ")"; }  
    public static void main(String[] args)  
    {  
        double x = Double.parseDouble(args[0]);  
        double y = Double.parseDouble(args[1]);  
        Charge c1 = new Charge(0.51, 0.63, 21.3);  
        Charge c2 = new Charge(0.13, 0.94, 81.9);  
        double v1 = c1.potentialAt(x, y);  
        double v2 = c2.potentialAt(x, y);  
        StdOut.printf("%.2e\n", (v1 + v2));  
    }  
}
```

instance variables

class name

constructor

instance methods

test client

create and initialize object

object

invoke

instance variable names

invoke constructor

OOP Breakdown

Creating an object

```
Charge c1 = new Charge(0.51, 0.63, 21.3);  
c1.potentialAt(x, y)
```

creates objects and invokes methods

Creating an Instance Variable

```
public class Charge  
{  
    Charge(double x0, double y0, double q0)  
    double potentialAt(double x, double y)  
    String toString()  
}
```

potential at (x, y) due to charge string representation

defines signatures and describes methods

Implementation

```
public class Charge  
{  
    private final double rx, ry;  
    private final double q;  
    public Charge(double x0, double y0, double q0)  
    { ... }  
    public double potentialAt(double x, double y)  
    { ... }  
    public String toString()  
    { ... }  
}
```

defines instance variables and implements methods

@pilatesdev

Made by @pilatesdev

Follow @pilatesdev on
Twitter for coding
cheatsheets, tips &
guided how-to's