Java Programming

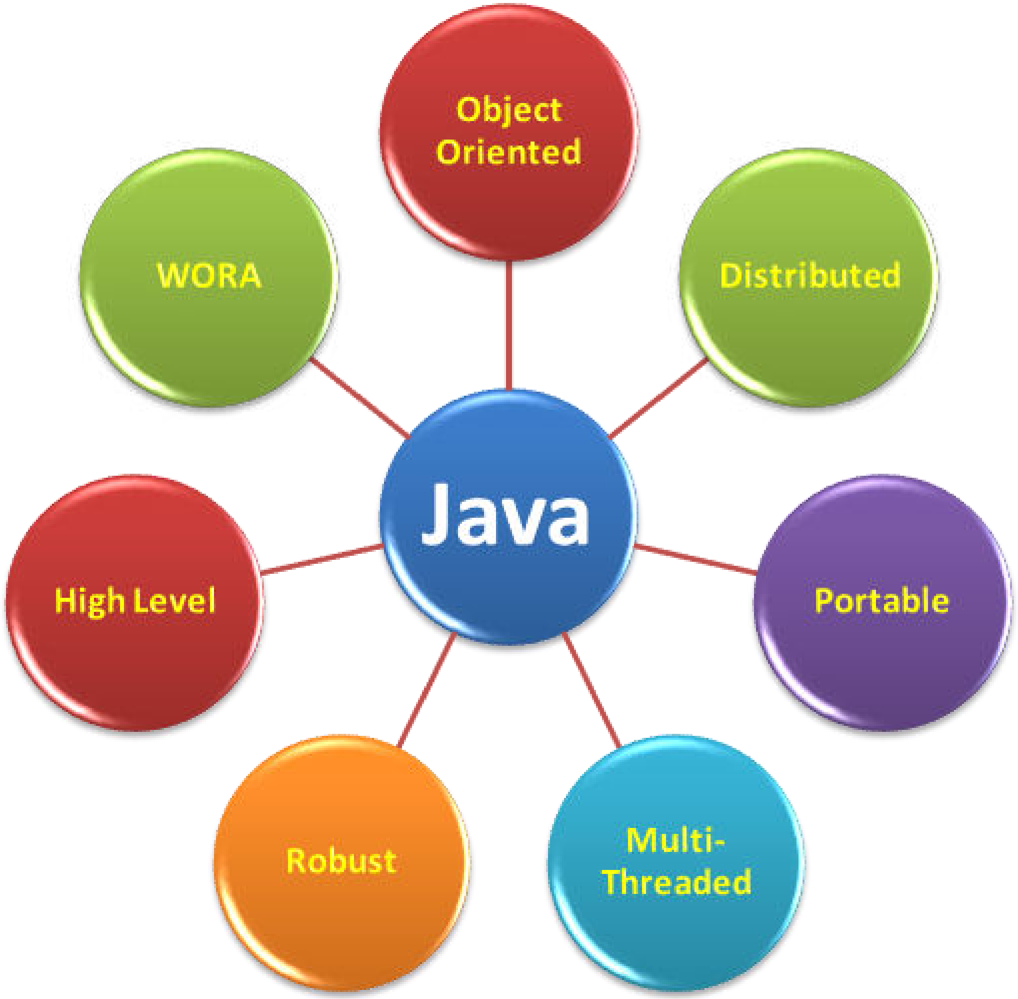
Basics

Jyoti Lakhani

JAVA BASICS

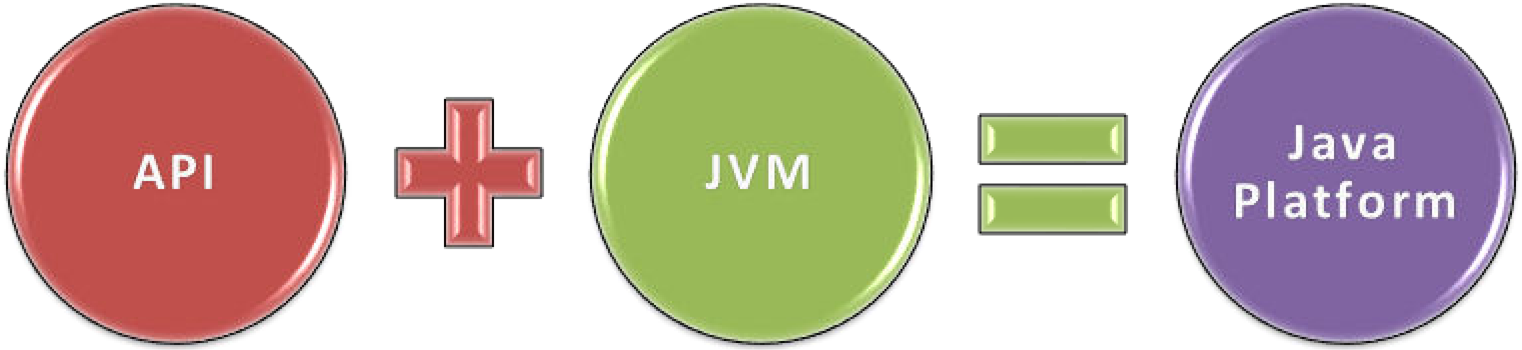
Part I

What Is Java ???



WORA- Write Once Run Everywhere

What Is Java ???

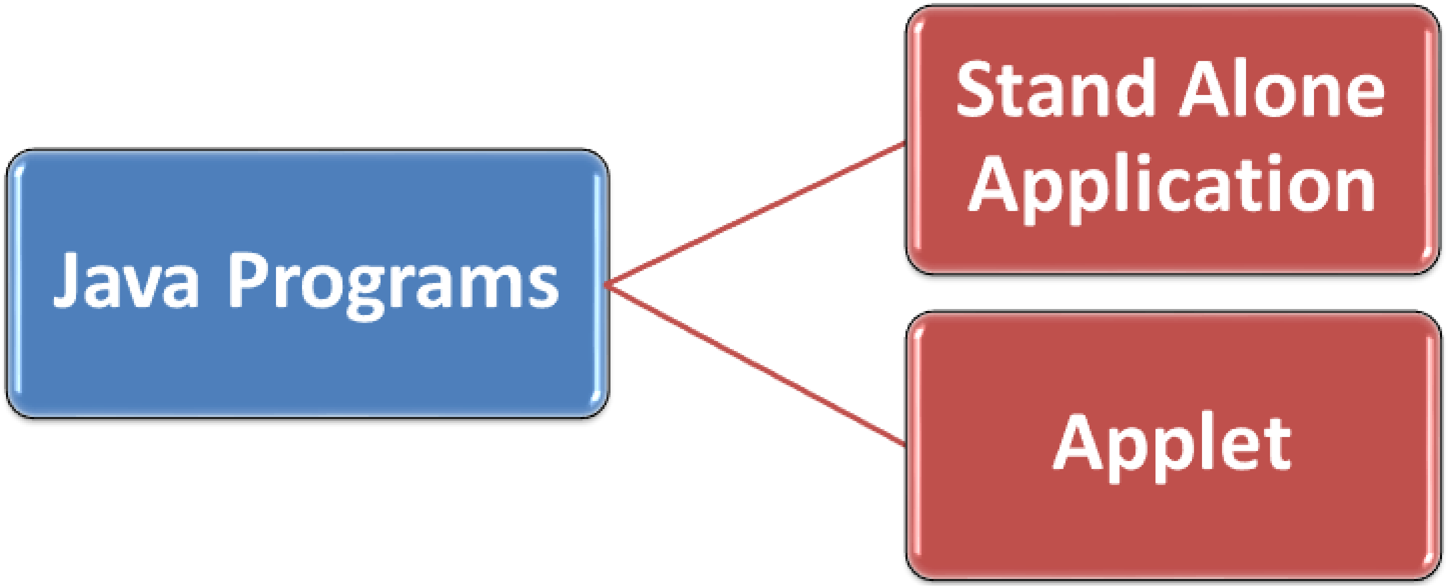


API

JVM

Java Platform

Java Programs



Java Programs

Stand Alone Application

Applet

Structure of Java Program

Documentation Section

- Includes the comments to improve the readability of the program

Package Statement

- Include package declarations

Import Statements

- Include statements used to referring classes and interfaces that are declared in other packages

Interface Section

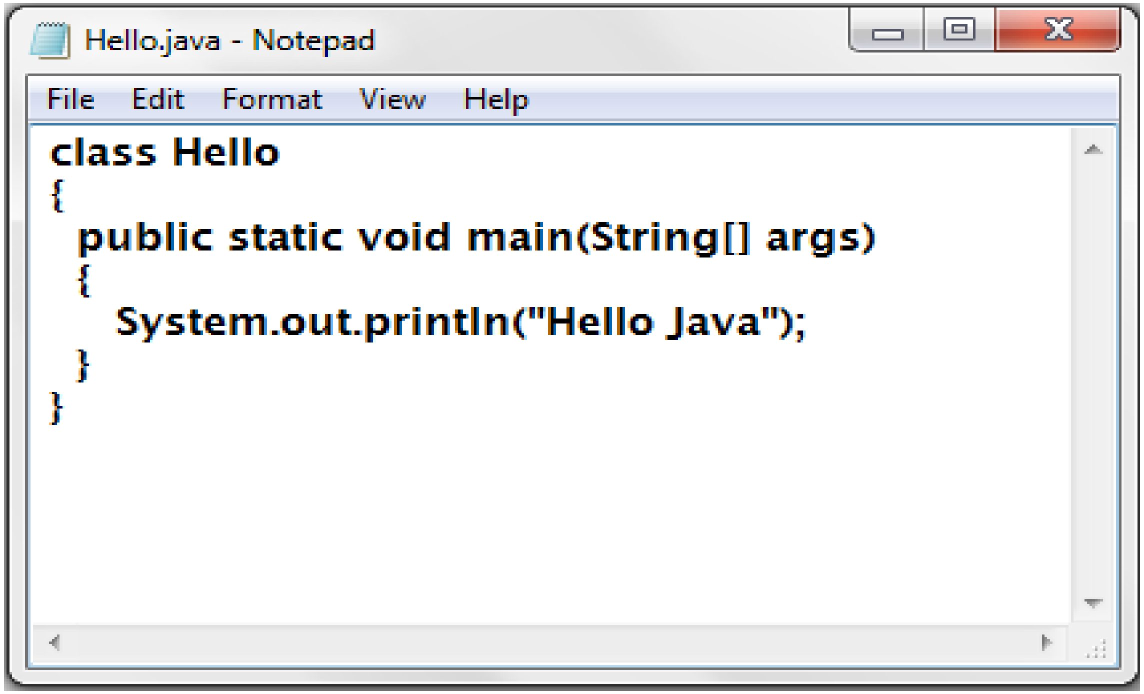
- Similar to class; but only include constants and method declarations

Class Section

- Information about user defined classes present in the program

First Java Program

Start Notepad and type following program-

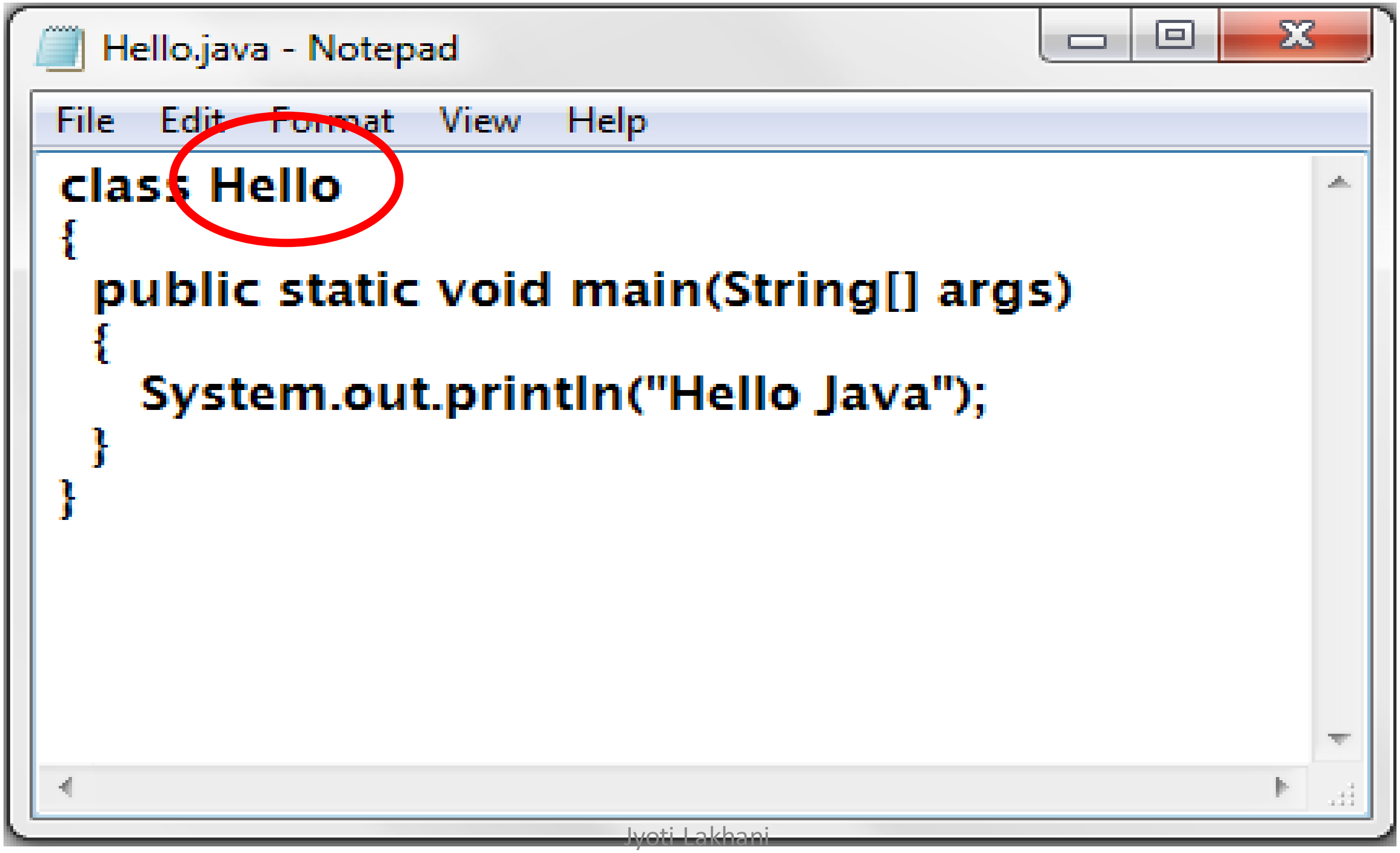


```
class Hello
{
  public static void main(String[] args)
  {
     System.out.println("Hello Java");
  }
}
```

Save this file as Hello.java

Points to Remember...

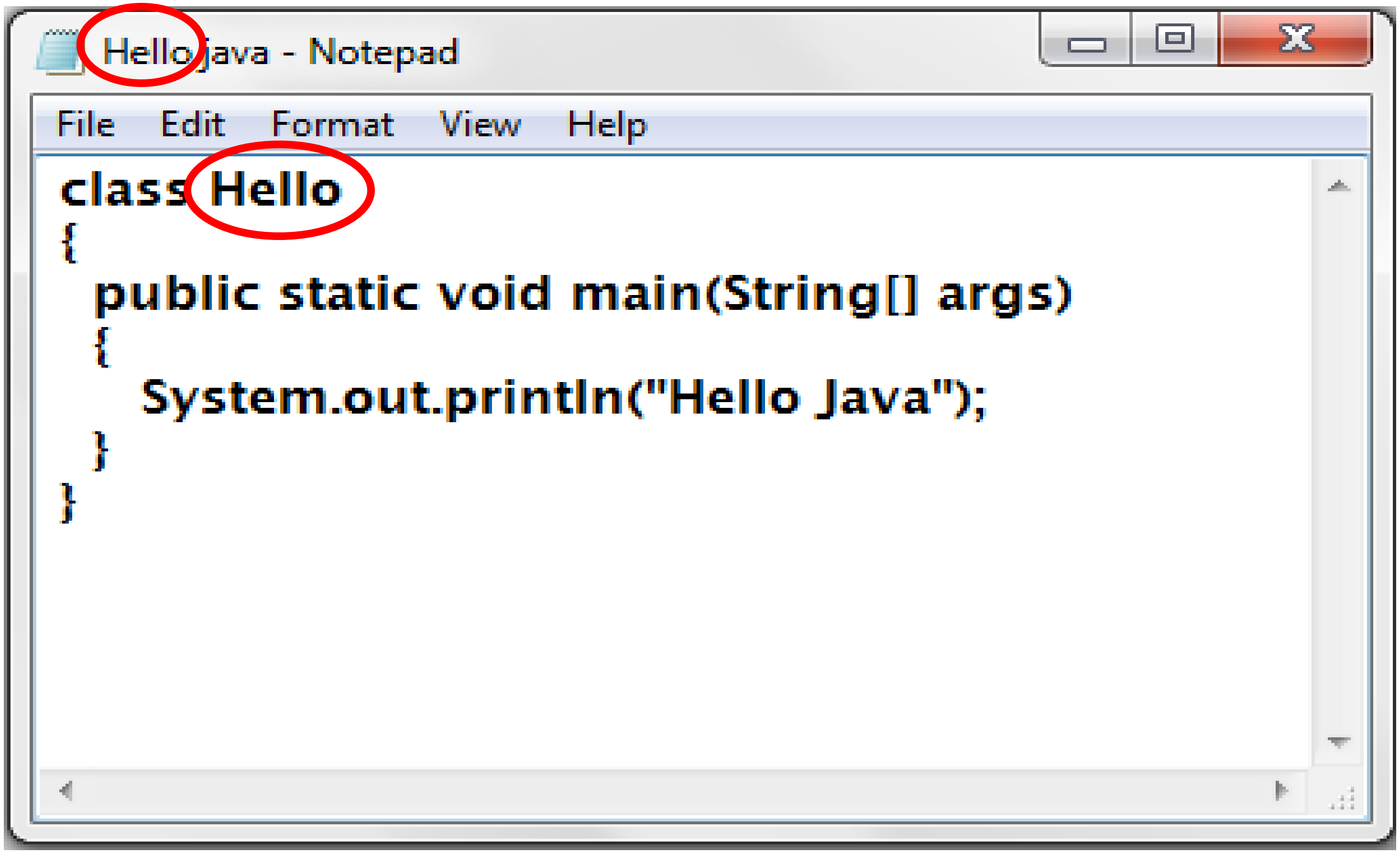
★ **The class name – always starts with capital letter**



```
class Hello
{
  public static void main(String[] args)
  {
    System.out.println("Hello Java");
  }
}
```

Points to Remember…

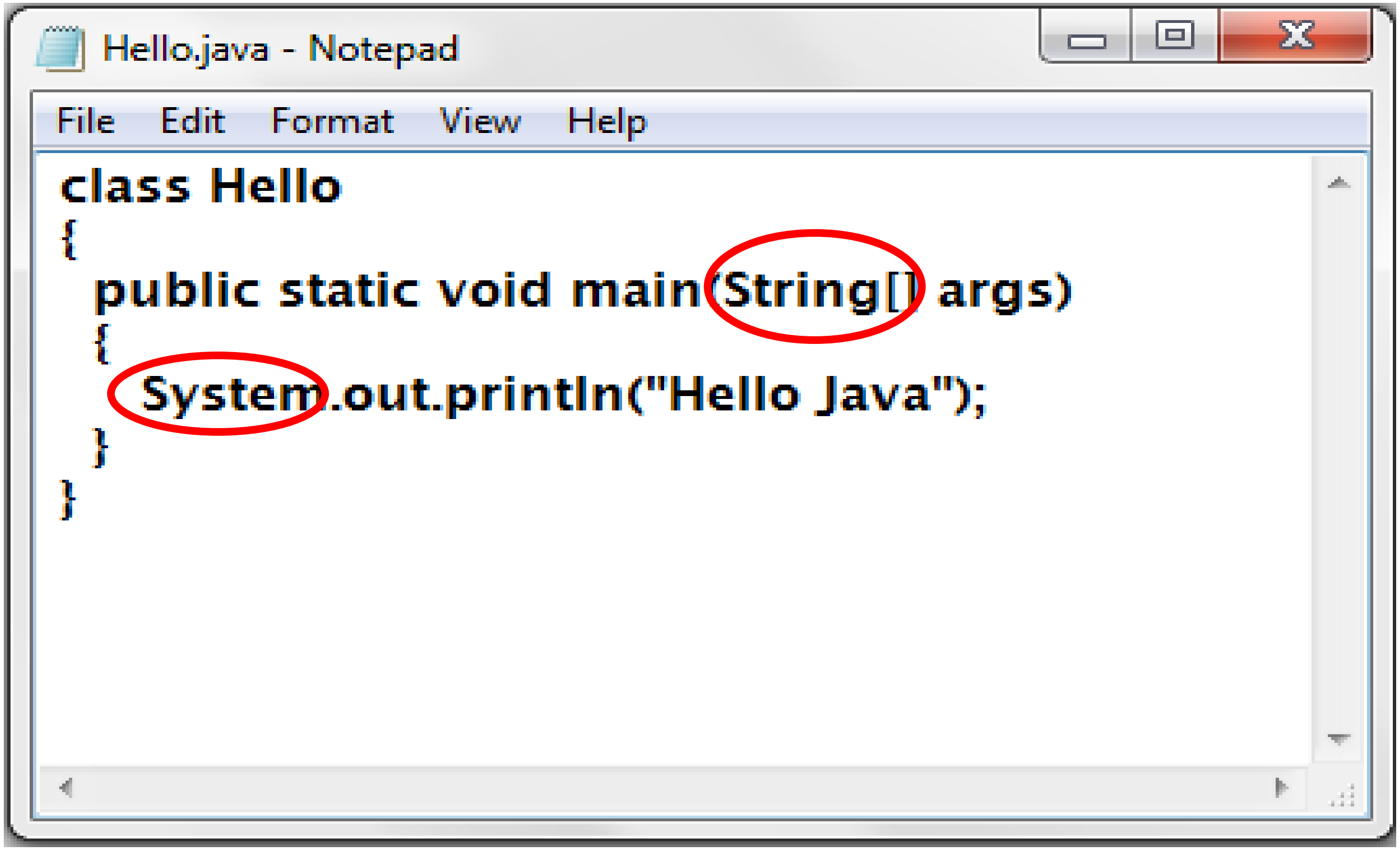
- File name should be exactly same of the class name in which main() function is exist



```
class Hello
{
  public static void main(String[] args)
  {
     System.out.println("Hello Java");
  }
}
```

Points to Remember...

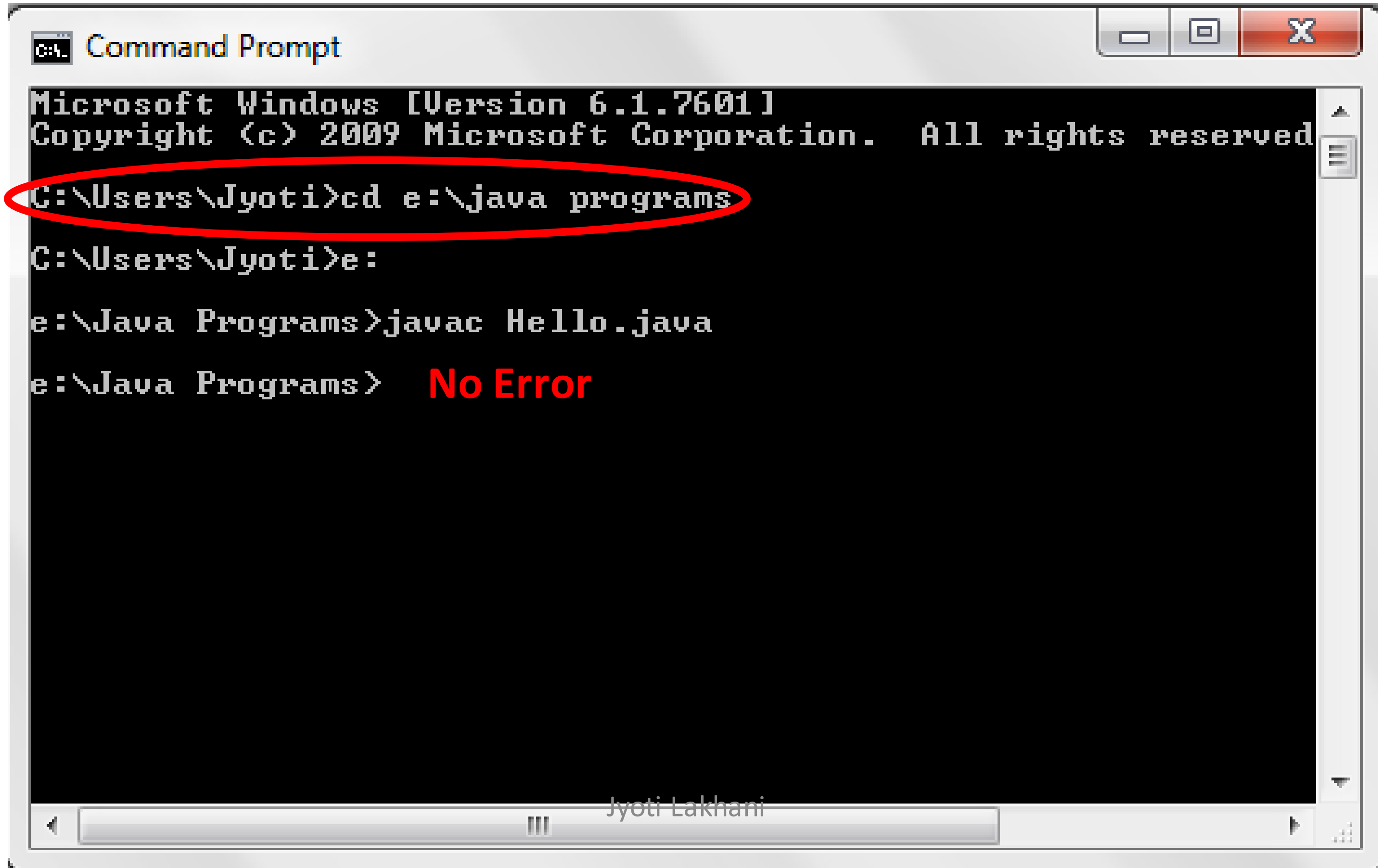
Always remember to capitalize the first letter of System and String keywords



```
class Hello
{
  public static void main(String[] args)
  {
    System.out.println("Hello Java");
  }
}
```

Run Java Program… Compilation

1. Go to the command prompt
2. go to the directory where your program is saved
3. type javac File_Name.java
4. If error is there in program, compiler will show error list with line numbers
5. If no error, it just show the prompt.



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved

C:\Users\Jyoti>cd e:\java programs

C:\Users\Jyoti>e:

e:\Java Programs>javac Hello.java

e:\Java Programs>
```

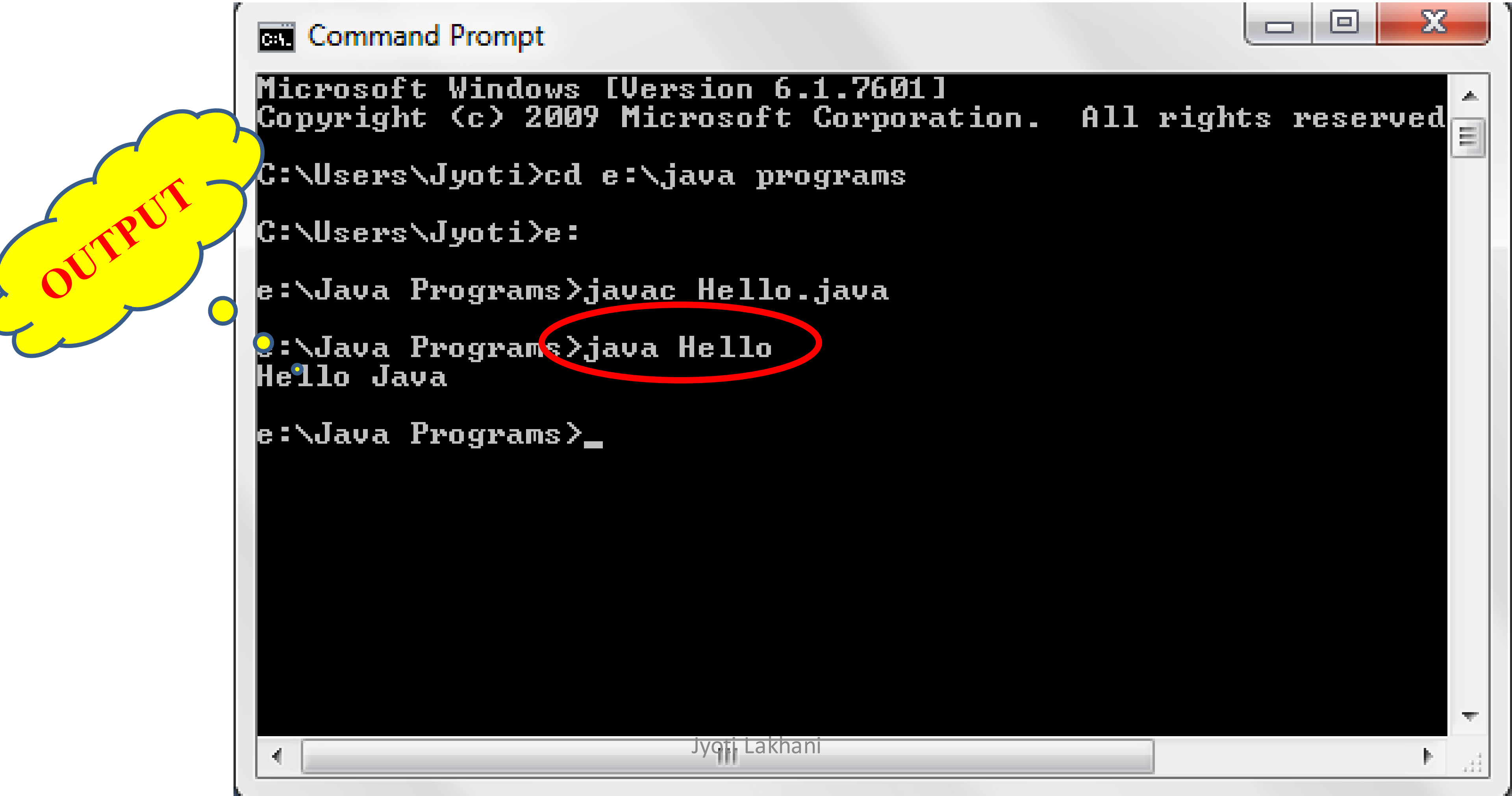
No Error

Run Java Program...

To run Java Program type-

java File_Name

The output will be appear on the screen

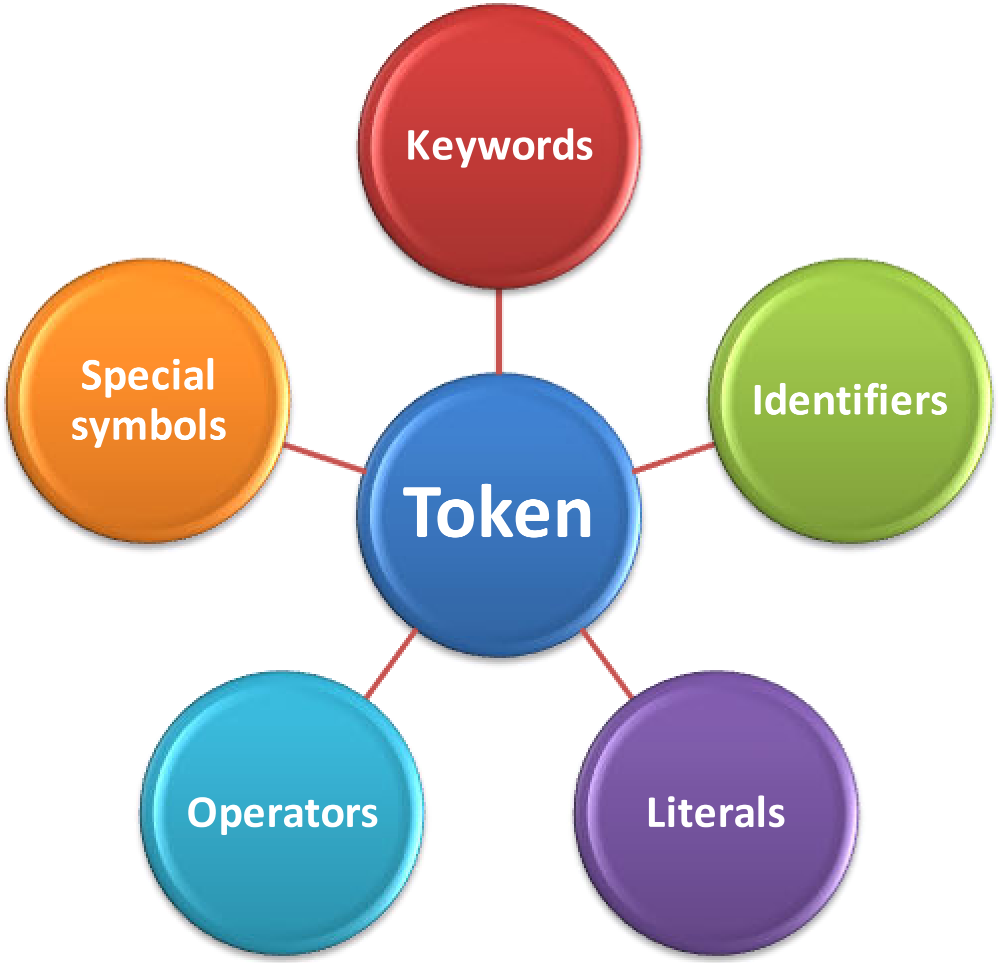


```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved

C:\Users\Jyoti>cd e:\java programs

C:\Users\Jyoti>e:

e:\Java Programs>javac Hello.java

e:\Java Programs>java Hello
Hello Java

e:\Java Programs>_
```

Java Building Blocks

TOKENS

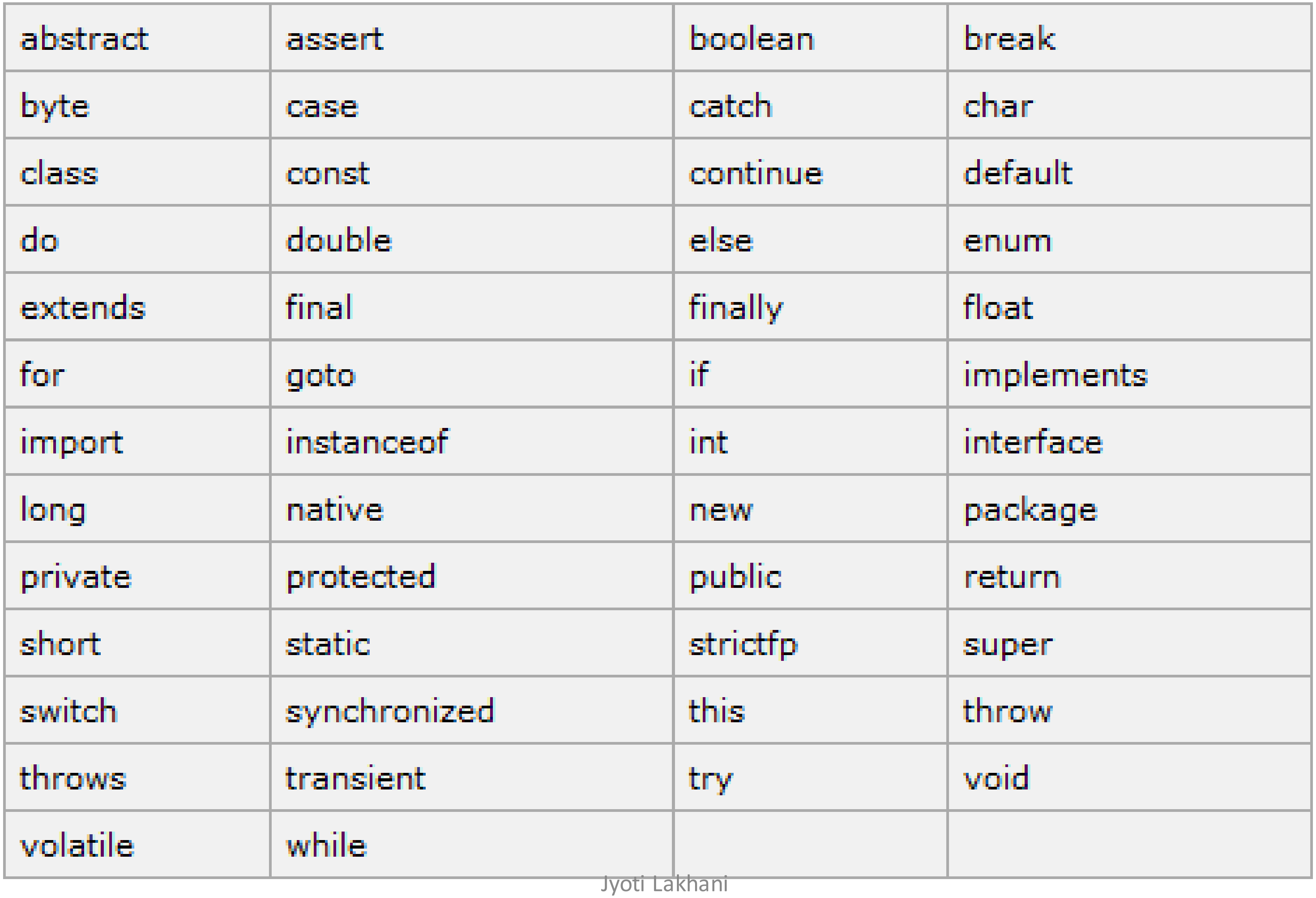
Java Building Blocks - TOKENS



Java Building Blocks - KEYWORDS

- Reserved Words
- Have special meaning
- Use for special purpose

Java Building Blocks - KEYWORDS



abstract	assert	boolean	break
byte	case	catch	char
class	const	continue	default
do	double	else	enum
extends	final	finally	float
for	goto	if	implements
import	instanceof	int	interface
long	native	new	package
private	protected	public	return
short	static	strictfp	super
switch	synchronized	this	throw
throws	transient	try	void
volatile	while		

Java Building Blocks - IDENIFIERS

IDENTIFIERS

Java Building Blocks - Identifiers

- A symbolic name

- Given by programmer

- To program elements viz. variable, constant, class, method, array, structures etc.

Java Building Blocks

Rules for Identifiers

- Identifires consists of A-Z, a-z, 0-9, _ and $.
- Can be several characters long
- Must start with a letter , _ or $
- Can not start with digit
- Must not contain tabs or spaces
- Must not be any java keyword
- Case sensitive
- Can not be true, false or null

Conventions for Identifiers

Class Name-

- nouns
- begin with Capital letter
- If class name contains more than one words, the first letter of each word should starts with capital letter.
- Method name- should begin with small letters

Conventions for Identifiers

Method name-

- verb
- should begin with small letters
- If contains multiple words, each subsequent word starts with Capital letter

Java Building Blocks

Conventions for Identifiers

Package name-

- should begin with small letters

Constant name-

- starts with Capital letter

Java Literals

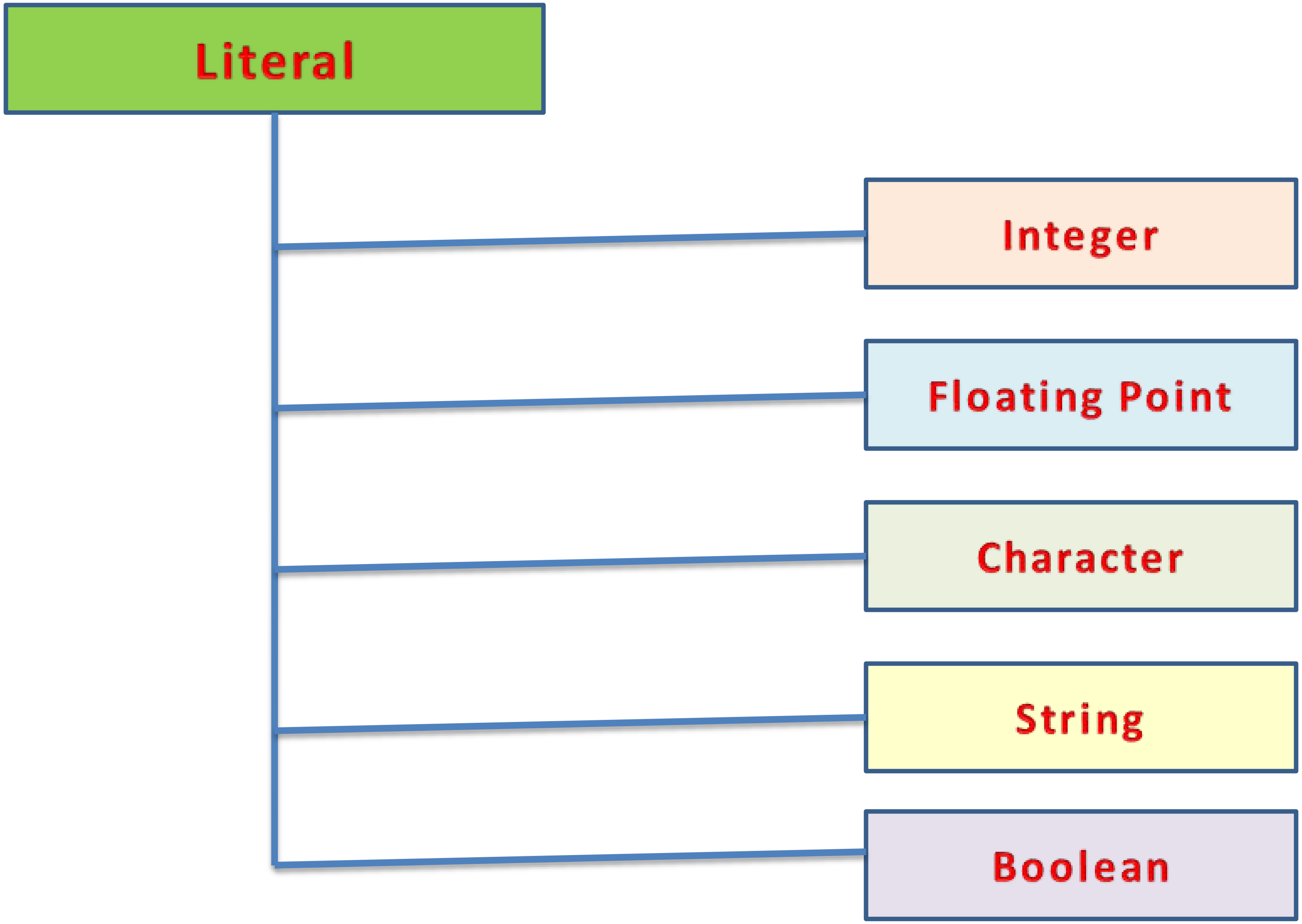
- A literal is a fixed value
- They are represented directly in the code without any computation
- Can be –
  - assigned to variables
  - passed to functions
  - used in expressions

- can be assigned to any primitive type variable

For example:

```
byte a = 68;
char a = 'A'
```

Java Literals



Literal

Integer

Floating Point

Character

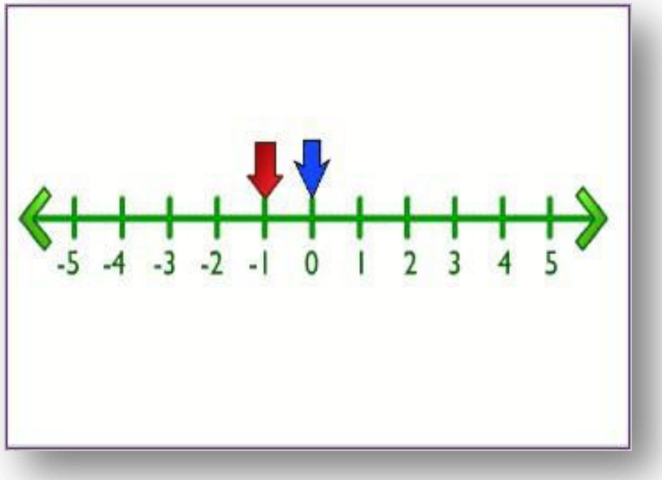
String

Boolean

Java Literals

Integer Literal

- whole number without decimal point
- Consists of a sequence of digits
- Must lie within the range of int data type
- We can use three number systems to represent integer literal
    - Decimal
    - Octal
    - Hexadecimal



Java Literals

Decimal Literals

- Any combination of digits from 0-9
- Consists of two or more digits
- First digit should be other than 0
- (if a decimal number is starts with 0, java compiler thinks that it is an octal number)

Example :

0
8
16
34565

Java Literals

Octal Literals

- Any combination of digits from 0 -7

- First digit must be 0

Example

0
010
050
020000

Java Literals

Hexa-Decimal Literals

- Any combination of digits from 0-9 or letters A-F or a-f
- Must starts with 0X or 0x
- It must have at least one digit

Example

```
0X101
0X080
0X10000
```

Java Literals

Rules for Integer Literals

No commas or blank spaces are allowed

Valid Integer Literals	Invalid Integer Literals
20 0x56 9978 09	5,45 0x 67 89 90

It can be either +ve or –ve. If no sign is there, it will consider +ve by default

Valid Integer Literals	Invalid Integer Literals
20 -56 -978 0999	5-45

It must not have a decimal integer

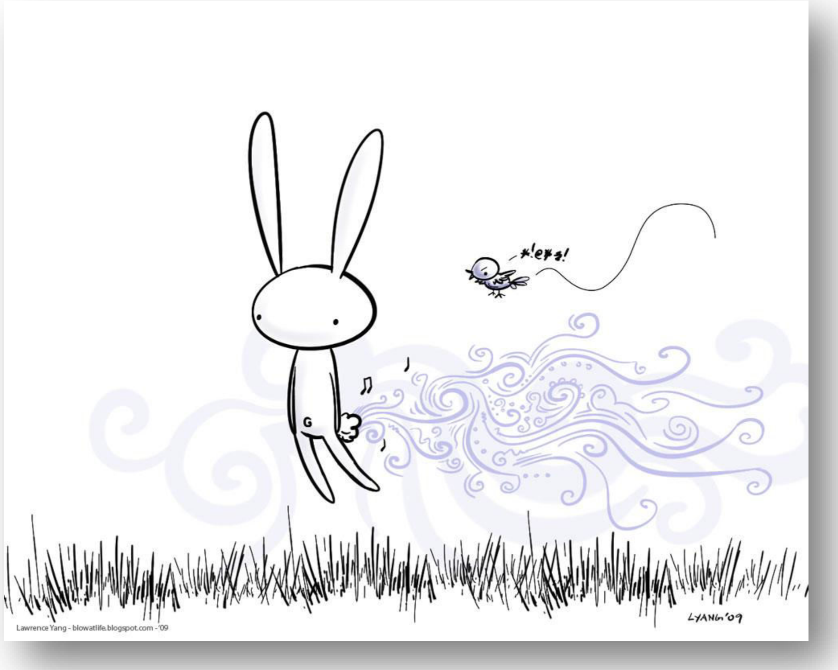
Valid Integer Literals	Invalid Integer Literals
20 -56 -978 0999	5.45 0x8.98 087.56

Jyoti Lakhani

Java Literals

Floating Point Literals

- Represent real numbers
- Consist of decimal point
- Two forms
  - Standard Notation
  - Scientific Notation



Lawrence Yang - blowatlife.blogspot.com - '09

Java Literals

Floating Point Literals- Standard Notation

Floating point numbers have two parts-

Integer part

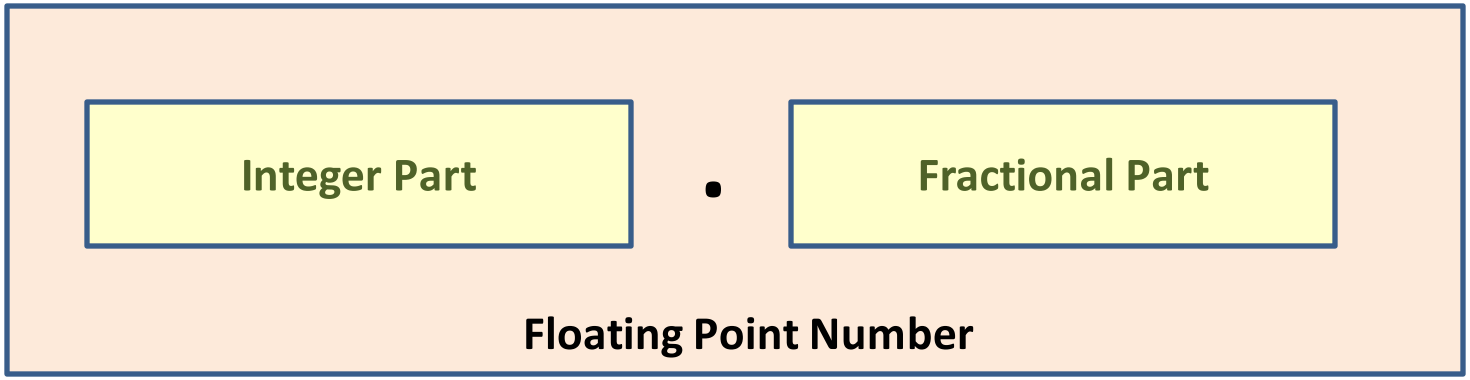
Decimal part

A decimal point between both parts

Java Literals

Floating Point Literals- Standard Notation- Rules

- A decimal point should be there
- No commas or Blanks



Integer Part . Fractional Part

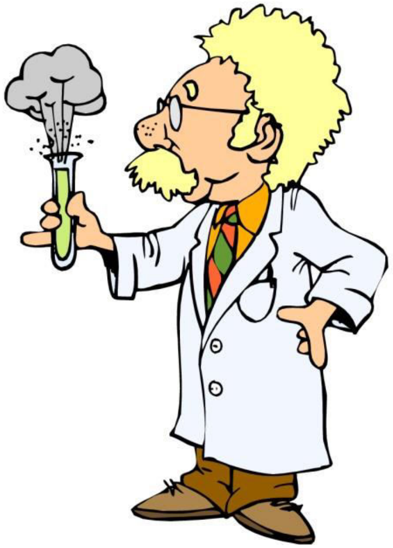
Floating Point Number

Example

12.90   345.89 67.87

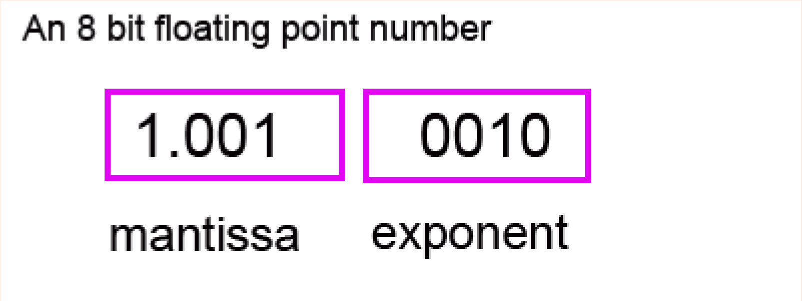
Java Literals

Floating Point Literals



SCIENTIFIC NOTATION

- Has two parts
  - mantissa
  - Exponent

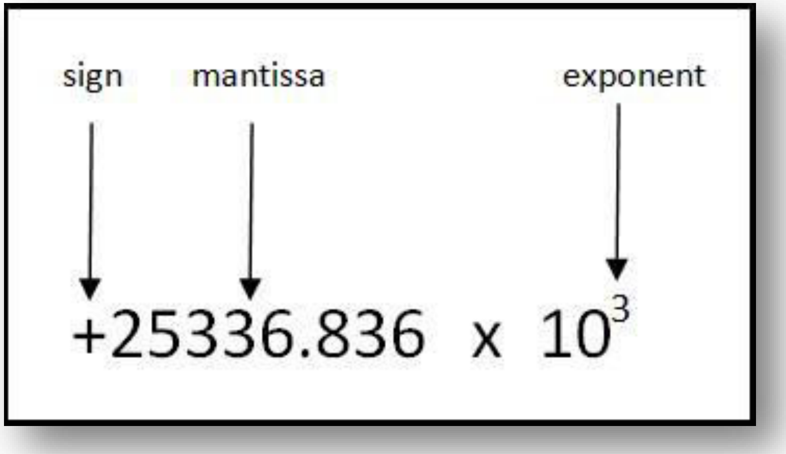


An 8 bit floating point number

Java Literals

Floating Point Literals- Scientific Notation

- Mantissa is a floating point number in standard notation
- Exponent denotes power of 10
- Mantissa and exponent parts are separated by letter E or e



sign

mantissa

exponent

$+25336.836 \times 10^{3}$

Example

231.54 == 2.3154e2

6000000 == 6.0e6

23.4 == 2.34e1 == 0.234e2 === 234e-1

Java Literals

Floating Point Literals- Scientific Notation- Rules

1. Mantissa part can be either integer or decimal form

2. It can be preceded by + or – sign

3. Exponent must have at least one digit

4. Spaces are not allowed in mantissa part as well exponent part

5. Letter e can be upper case or lower case

6. Decimal point can be ignored if e is included

Java Literals

Character Literals

- Represent a single Unicode characters

- Enclosed within a ‘ ’ mark

- Managed internally as integer and determined by Unicode table

- Some characters can not be shown by pressing the keyboard keys becoz they have some special meaning associated with them those can be shown by unicode

- Java provide escape sequences for that purpose

Visit unicode.org for detailed table

Java Literals

String Literals

- A collection of consecutive characters
- Enclosed within “ “
- Implemented by String class in java

Escape Sequences

Java language supports few special escape sequences for String and char literals as well. They are:

Notation	Character represented	Abbr	Action Performed
\n	Newline/ Line Feed (0x0a) Ascii - 10	NL/ LF	insert New Line
\r	Carriage return (0x0d) Ascii - 13	CR	return to the beginning of the current line
\f	Formfeed (0x0c) Ascii 12	FF	advance downward to the next "page"
\b	Backspace (0x08)		
\s	Space (0x20)		
\t	tab		
\"	Double quote		
\'	Single quote		
\\	backslash		
\ddd	Octal character (ddd)		
\uxxxx	Hexadecimal UNICODE character (xxxx)		