FACE

Data Types

Data Types

Primitive data-types

- byte
- int
- short
- long
- char
- Boolean
- float
- double

Non-primitive data-types

- Strings
- Arrays
- Classes ….

Primitive Data Types

Data – Type	Size	Description
byte	1 byte	Stores whole numbers from -128 to 127
int	4 bytes	Stores whole numbers from -2,147,483,648 to 2,147,483,647
short	2 bytes	Stores whole numbers from -32,768 to 32,767
long	8 bytes	Stores whole numbers from -9,223.372,036.854,775.808 to 9,223.372,036,854,775,808
char	2 bytes	Stores a single character/letter
boolean	1 byte	Stores true or false values
float	4 bytes	Stores fractional numbers from 3.4e−038 to 3.4e+038. Sufficient for storing 6 to 7 decimal digits
double	8 bytes	Stores fractional numbers from 1.7e−308 to 1.7e+038. Sufficient for storing 15 decimal digits

FACE

BYTE

- Byte stores from -128 and 127.
- Can be used instead of int or other integer types to save memory.

```
byte myNum = 100;
System.out.println(myNum);
```

SHORT

- short stores from -32768 to 32767:

```
short myNum = 5000;
System.out.println(myNum);
```

INT

- int stores from -2147483648 to 2147483647.
- preferred data type when we create variables with a numeric value.

```
int myNum = 100000;
System.out.println(myNum);
```

LONG

- long stores from -9223372036854775808 to 9223372036854775808.
- Used when int is not large enough to store the value.
- you should end the value with an "L":

```
long myNum = 15000000000L;
System.out.println(myNum);
```

Floating Point Types

FLOAT

- float can store fractional numbers from 3.4e−038 to 3.4e+038.
- should end the value with an "f":

```
float myNum = 5.75f;
System.out.println(myNum);
```

DOUBLE

- Double can store fractional numbers from 1.7e−308 to 1.7e+038.
- you should end the value with a "d":

```
double myNum = 19.99d;
System.out.println(myNum);
```

FACE

Use float or double?

FACE

BOOLEAN

- declared with the boolean keyword
- can only take the values true or false:

```
boolean isJavaFun = true;
boolean isFishTasty = false;
System.out.println(isJavaFun);
System.out.println(isFishTasty);
```

STRING

- The String data type is used to store a sequence of characters (text).
- String values must be surrounded by double quotes:

```
String greeting = "Hello World";
System.out.println(greeting);
```

CHARACTER

- The char data type is used to store a **single** character.
- A char value must be surrounded by single quotes, like 'A' or 'c':

```
char myGrade = 'B';
System.out.println(myGrade);
```

WHY CHAR REQUIRES 2 BYTES IN JAVA?

IS STRING A NINTH TYPE?

FACE

WHY DO WE NEED DATA-TYPES?

IS IT REQUIRED ?

VARIABLES

What is a variable ?

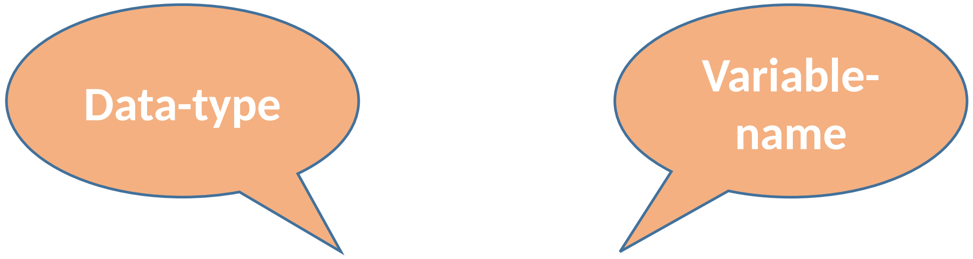
- A variable which holds value, during the life of a Java program.

- Every variable is assigned a **data type** which designates the type and quantity of value it can hold.

- In order to use a variable in a program you to need to perform 2 steps
--Variable Declaration
--Variable Initialization

Variable Declaration

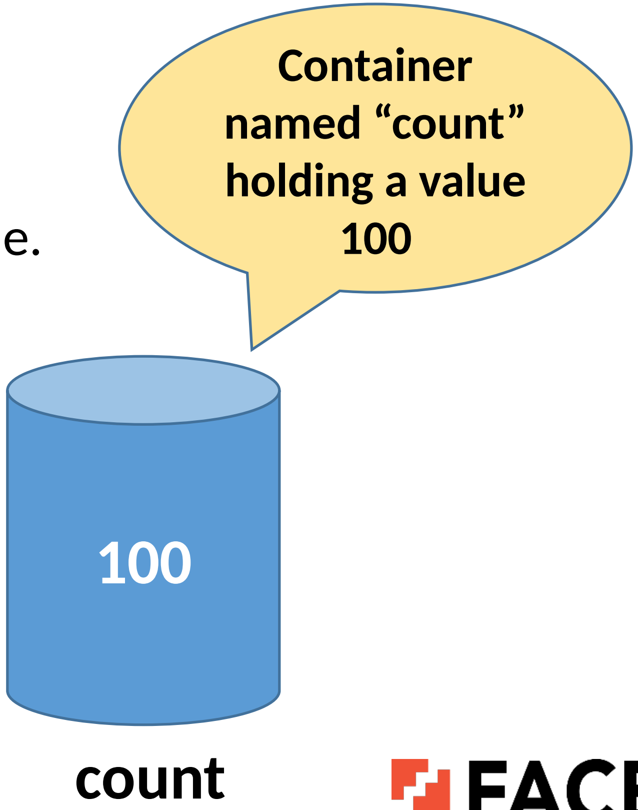
To declare a variable, you must specify the data type & give the variable a unique name.



int count;

To initialize a variable, you must assign it a valid value.

```
count=100;
```



You can combine variable declaration and initialization.

```
int count=100;
```

NAMING CONVENTION OF VARIABLES

- can start with underscore('_') but not with digits.
- Should be mnemonic i.e, designed to indicate to the casual observer the intent of its use.
- Can use _(underscore), digits and letters.
- Should not use any reserved word.

TYPES OF VARIABLES

Local variables

Instance variables

Static variables

Consider this code snippet

```
class Guru99
 {
            int data = 99; //instance variable
            static int a = 1; //static variable
            void method()
            {
                        int b = 90; //local variable
            }
 }
```

Non-static variable V/S Static variable

Non-static variable	Static variable
1. Memory is allocated multiple time whenever a new object is created.	1. Memory is allocated for these variable only once in the program.
2. Non-static variable also known as instance variable while because memory is allocated whenever instance is created.	2. Memory is allocated at the time of loading of class so that these are also known as class variable.
3. Non-static variable are specific to an object	3. Static variable are common for every object that means there memory location can be sharable by every object reference or same class.
4. Non-static variable can access with object reference.	4. Static variable can access with class reference.

FACE

WHAT IS WIDENING?

Consider this code snippet

```
public class Test
{
    public static void main(String[] args)
    {
        System.out.print("Y" + "O");
        System.out.print('L' + 'O');
    }
}
```

Can you predict the output?

YOLO ✗

YO155 ✓

FACE

Now, try to predict the output

```
public class Test
{
    public static void main(String[] args)
    {
        System.out.print("Y" + "O");
        System.out.print('L');
        System.out.print('O');
    }
}
```

YOLO

YO7679 ✗

✓

FACE

RULES FOR WIDENING PRIMITVE CONVERSION

- The result of adding Java chars, shorts or bytes is an **int.**
- If either operand is of type double, the other is converted to double.
- Otherwise, if either operand is of type float, the other is converted to float.
- Otherwise, if either operand is of type long, the other is converted to long.
- Otherwise, **both operands are converted to type int**

WHAT IS NARROWING?

NARROWING OR EXPLICIT TYPE-CASTING

If we want to assign a value of larger data type to a smaller data type we perform explicit type casting or narrowing.

- This is useful for incompatible data types where automatic conversion cannot be done.
- Here, target-type specifies the desired type to convert the specified value to.

Guess the output

```
public class Test
{
  public static void main(String[] argv)
  {
    char ch = 'c';
    int num = 88;
    ch = num;
  }
}
```

Error

Now, try to predict the output

Variable Initialization

10.5 ✓
10

THANK YOU

FACE