Data Types

String	" type string here "
int	E.G: 0
boolean	true/false
char	'x'
double	E.G 0.00

Operators (Summary)

+	Addition / String concatenation
%	Modulus (remainder)
++ , --	Increment / Decrement by 1
!	Inverts boolean value
= , ==	Assigns, Equals to
!=	Does Not equal to
>=	Greater than OR Equal to
<=	Lesser than OR Equal to
&&	Conditional - AND
||	Conditional - OR

Methods

System.out.println(line)

System.out.printf(format, arguments)

Helper.read**DataType**(string)

// For printf formatting //
%d = integer
%s = String
%f = double (.2f = 2dp)
%b = boolean

Loops / Impt Statements

while(condition)	(option != 4) { code }
for(var,cond,incre)	(int 1=0;i<10;i++)
if / else if / else	if(condition) { code }

Loops / Impt Statements (cont)

switch(Expression)	switch (choice)

Switch syntax: E.g: Input is an int Choice

Switch (choice)
{
case 1:
< code >
break;
.....
default:
< code >
break;
}

Arrays

Declaring an Array

Syntax:
Datatype[ ] *nameOfArray* =
new *Datatype*[*No. of elements in array*];
Example:
int [ ] *randomValues* = new *int* [*7*];
// Creates an array called *randomValues* with *7 elements* (0-6)

Assigning Value to Array: (Using prev E.G)

randomValues[3] = 100;
// Assigns value of *100* to the *4th element [3]* of randomValues array

Declaring & Initializing @ Same Time

int[ ] randomValues = {5,12,51,23,12,24,21};
// Creates an array called *randomValues* and assigning *7 elements in it in a single line.*

Accessing Elements:

System.out.println(randomValue[3]);
// Prints out "23" (prev example)

Finding out Array Length

System.out.println(randomValue.length);
// Prints out 7

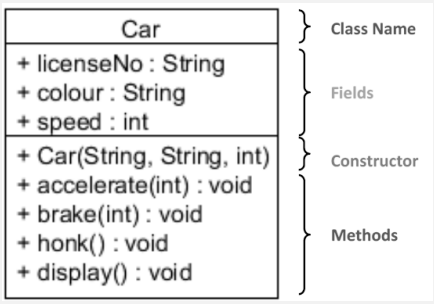
Values for Primitive Arrays

int	0
double	0.0
boolean	false
String	null

String Methods

charAt(index)	Returns char @ index
endsWith(suffix)	if ends w suffix
equalsIgnoreCase(string)	
length()	Returns length of string
startsWith(prefix)	if starts w prefix
toUpperCase()	Converts to upperCase
toLowerCase()	Converts to lowerCase

Class Diagram



'+' = Public , '-' = Private
underlined = Static

// Constructors have the same name as class.
// if return type is 'void' , no **return** statement is required.

Creation of Class

// Using the Class Diagram above //

Declare Fields First:

public String licenseNo, colour;

public int speed;

Constructor

[Right-click, Source, Create constructor using Fields, delete the super();]

Create Methods:

public void accelerate(int acc)

{

}

public void honk ()

{

}

// Creating Array/ Object in Main Class //

Array:

Syntax:

ClassName[ ]*arrayName* = new *ClassName*[x];

E.g:

Car[ ] testArray = new Car[5]

Objects

E.g:

Car newObject = new Car();

// Creates a new object, called 'newObject'.

Calling a method in Main class from another class

newObject.methodName();