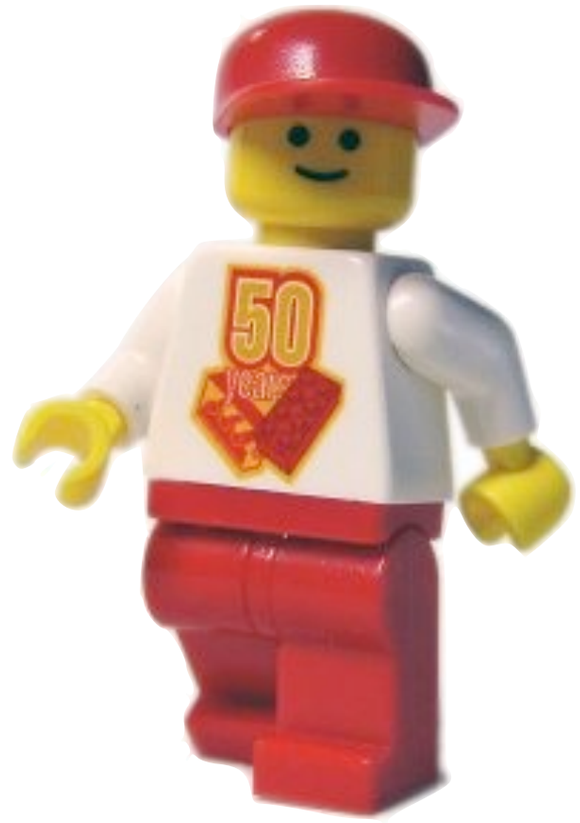
Java para Pythonistas



(Pythoneros?)

Material extra Taller de Proyecto Lego Mindstorms

Johan Fabry, Oficina 311, jfabry@dcc.uchile.cl, @johanfabry

¡¡ Cuidado !!

- Solo se trata de lo ultra-básico
  - Muy muy muy incompleto (muy)
  - Hay cosas ‘mágicas’
  - Hay cosas ‘por que si’
  - Hay cosas ‘feas’
- No esta enfocado en LeJos
  - Introducción de Java en general

Hola, Mundo

Python	`def main():`
	`  print "Hello world!"`
	
	`main()`

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

Ejemplo: F to C

```
def main():
    fahr = input("Enter the temperature in F: ")
    cel = (fahr - 32) * 5.0/9.0
    print "the temperature in C is: ", cel
```

Ejemplo: F to C

```
import java.util.Scanner;

public class TempConv {
  public static void main(String[] args) {
    Double fahr; Double cel; Scanner in;
    in = new Scanner(System.in);
    System.out.println("Enter temperature in F: ");
    fahr = in.nextDouble();

    cel = (fahr - 32) * 5.0/9.0;
    System.out.println("Temperature in C is: " + cel);
}}
```

Tipeo Estatico

```
import java.util.Scanner;

public class TempConv {
  public static void main(String[] args) {
    Double fahr; Double cel; Scanner in;

    in = new Scanner(System.in);
    System.out.println("Enter temperature in F: ");
    fahr = in.nextDouble();

    cel = (fahr - 32) * 5.0/9.0;
    System.out.println("Temperature in C is: " + cel);
}}
```

Tipos Comunes

'Primitivo'	Objeto	...
int	Integer	Ej: `42`
float	Float	Ej: `3.1415, 3.1415F`
double	Double	Ej: `1.618034D`
char	Char	`'x'`, `\t` tab, `\n` nueva linea
bool	Boolean	`true` y `false`
-	String	`"foo"`, se junta con `+`

Estructura del Codigo

```
import java.util.Scanner;

public class TempConv {
    public static void main(String[] args) {
        Double fahr; Double cel; Scanner in;

        in = new Scanner(System.in);
        System.out.println("Enter temperature in F: ");
        fahr = in.nextDouble();

        cel = (fahr - 32) * 5.0/9.0;
        System.out.println("Temperature in C is: " + cel);
    }
}
```

¡No Importa!

Separación de instrucciones

```
import java.util.Scanner;

public class TempConv {
  public static void main(String[] args) {
    Double fahr; Double cel; Scanner in;

    in = new Scanner(System.in);
    System.out.println("Enter temperature in F: ");
    fahr = in.nextDouble();

    cel = (fahr - 32) * 5.0/9.0;
    System.out.println("Temperature in C is: " + cel);
}}
```

Import = import (±)

```
import java.util.Scanner;

public class TempConv {
  public static void main(String[] args) {
    Double fahr; Double cel; Scanner in;

    in = new Scanner(System.in);
    System.out.println("Enter temperature in F: ");
    fahr = in.nextDouble();

    cel = (fahr - 32) * 5.0/9.0;
    System.out.println("Temperature in C is: " + cel);
}}
```

Import = import (±)

```
import java.util.Scanner;
```

```
import java.util.*;
```

```
public class TempConv {
  public static void main(String[] args) {
    Double fahr; Double cel; Scanner in;
    in = new Scanner(System.in);
    System.out.println("Enter temperature in F: ");
    fahr = in.nextDouble();

    cel = (fahr - 32) * 5.0/9.0;
    System.out.println("Temperature in C is: " + cel);
}}
```

Instanciacion

```
import java.util.Scanner;

public class TempConv {
  public static void main(String[] args) {
    Double fahr; Double cel; Scanner in;
    in = new Scanner(System.in);
    System.out.println("Enter temperature in F: ");
    fahr = in.nextDouble();

    cel = (fahr - 32) * 5.0/9.0;
    System.out.println("Temperature in C is: " + cel);
}}
```

Instanciacion

```
import java.util.Scanner;

public class TempConv {
  public static void main(String[] args) {
    Double fahr; Double cel; Scanner in;
    in = new Scanner(System.in);
    System.out.println("Enter temperature in F: ");
    fahr = in.nextDouble();

    cel = (fahr - 32) * 5.0/9.0;
    System.out.println("Temperature in C is: " + cel);
}}
```

XXX

Llamar métodos

```
import java.util.Scanner;

public class TempConv {
  public static void main(String[] args) {
    Double fahr; Double cel; Scanner in;
    in = new Scanner(System.in);
    System.out.println("Enter temperature in F: ");
    fahr = in.nextDouble();

    cel = (fahr - 32) * 5.0/9.0;
    System.out.println("Temperature in C is: " + cel);
}}
```

Llamar métodos

- Valor de retorno debe ser compatible con el tipo esperado.
- Ver documentación de Java de las clases ‘estándar’
  - ej: http://docs.oracle.com/javase/6/docs/api/java/util/Scanner.html

Ejemplo 2: Fracciones

```
public class Fraction {
  public Integer numerator;
  public Integer denominator;

  public Fraction(Integer top, Integer bottom){ XXX }

  public Fraction(Integer num) { XXX }

  public Fraction add(Fraction otherFrac) { XXX }

  public Fraction add(Integer other) { XXX }

  private Integer gcd(Integer m, Integer n) { XXX }

}
```

Clases

```
public class Fraction {
   public Integer numerator;
   public Integer denominator;

   public Fraction(Integer top, Integer bottom){    XXX    }

   public Fraction(Integer num) {    XXX    }

   public Fraction add(Fraction otherFrac) {    XXX    }

   public Fraction add(Integer other) {    XXX    }

   private Integer gcd(Integer m, Integer n) {    XXX    }

}
```

Constructores

```
public class Fraction {
  public Integer numerator;
  public Integer denominator;

  public Fraction(Integer top, Integer bottom){ XXX }

  public Fraction(Integer num) { XXX }

  public Fraction add(Fraction otherFrac) { XXX }

  public Fraction add(Integer other) { XXX }

  private Integer gcd(Integer m, Integer n) { XXX }

}
```

Constructores

```
public class Fraction {
  public Integer numerator;
  public Integer denominator;

  public Fraction(Integer top, Integer bottom){ XXX }

  public Fraction(Integer num) { XXX }

  pu                                rFrac) { XXX }

  pu                                ) { XXX }

  pu                                teger n) { XXX }

}
```

Overloading = Sobrecarga:
Se elige cual a base de
tipo y numero de argumentos

Metodos

```
public class Fraction {
  public Integer numerator;
  public Integer denominator;

  public Fraction(Integer top, Integer bottom){ XXX }

  public Fraction(Integer num) { XXX }

  public Fraction add(Fraction otherFrac) { XXX }

  public Fraction add(Integer other) { XXX }

  private Integer gcd(Integer m, Integer n) { XXX }

}
```

Metodos

Overloading = Sobrecarga:
Se elige cual a base de
tipo y numero de argumentos

```
pu                                                    eger bottom){ XXX }

  public Fraction(Integer num) { XXX }

  public Fraction add(Fraction otherFrac) { XXX }

  public Fraction add(Integer other) { XXX }

  private Integer gcd(Integer m, Integer n) { XXX }

}
```

Variables de Instancia

```
public class Fraction {
  public Integer numerator;
  public Integer denominator;

  public Fraction(Integer top, Integer bottom){ XXX }

  public Fraction(Integer num) { XXX }

  public Fraction add(Fraction otherFrac) { XXX }

  public Fraction add(Integer other) { XXX }

  private Integer gcd(Integer m, Integer n) { XXX }

}
```

Variables de Instancia

```
public class Fraction {
  public Integer numerator;
  public Integer denominator;

  public Fraction(Integer top, Integer bottom){ XXX }

  public Fraction() { XXX }

  public Fraction(Fraction otherFrac) { XXX }

  public Fraction add(Integer other) { XXX }

  private Integer gcd(Integer m, Integer n) { XXX }

}
```

Todo el mundo puede leer y escribir.

¡Mala practica en general !

Variables de instancia

```
public class Fraction {
   public Integer numerator;
   public Integer denominator;

   public Fraction add(Fraction otherFrac) {
     Integer newNum =
       otherFrac.denominator * numerator +
          denominator * otherFrac.numerator;

     Integer newDen = denominator*otherFrac.denominator;
     Integer common = gcd(newNum,newDen);
     return new Fraction(newNum/common, newDen/common );
}
```

XXX

XXX

Llamar métodos propios

```
public class Fraction {
  public Integer numerator;
  public Integer denominator;

    XXX

  public Fraction add(Fraction otherFrac) {
    Integer newNum =
      otherFrac.denominator * numerator +
        denominator * otherFrac.numerator;

    Integer newDen = denominator*otherFrac.denominator;
    Integer common = gcd(newNum,newDen);
    return new Fraction(newNum/common, newDen/common );
}
```

XXX

Return

```
public class Fraction {
  public Integer numerator;
  public Integer denominator;

  public Fraction add(Fraction otherFrac) {
    Integer newNum =
      otherFrac.denominator * numerator +
        denominator * otherFrac.numerator;

    Integer newDen = denominator*otherFrac.denominator;
    Integer common = gcd(newNum,newDen);
    return new Fraction(newNum/common, newDen/common );
  }
}
```

XXX

XXX

Return

```
public class Fraction {
  public Integer numerator;
  public Integer denom

    XXX

  public Fraction add(
    Integer newNum =
      otherFrac.denominator * numerator +
        denominator * otherFrac.numerator;

    Integer newDen = denominator*otherFrac.denominator;
    Integer common = gcd(newNum,newDen);
    return new Fraction(newNum/common, newDen/common );
}
```

Constructores nunca **`return`**.
Metodos sin return
tienen tipo **`void`**

XXX

Condicionales

Python	`if condition:`
	`  statement1`
	`  statement2`
	`  ...`

Java	`if (condition) {`
	`  statement1`
	`  statement2`
	`  ...`
	`}`

Python	`if condition:`
	`  statement1`
	`  ...`
	`else:`
	`  statement1`
	`  ...`

Java	`if (condition) {`
	`  statement1`
	`  ...`
	`} else {`
	`  statement1`
	`  ... }`

Condicionales II

Python

```
if grade < 60:
    print 'F'
elif grade < 70:
    print 'D'
elif grade < 80:
    print 'C'
elif grade < 90:
    print 'B'
else:
    print 'A'
```

Java

```
if (grade < 60) {
  System.out.println('F');
} else if (grade < 70) {
  System.out.println('D');
} else if (grade < 80) {
  System.out.println('C');
} else if (grade < 90) {
  System.out.println('B');
} else
  System.out.println('A');
```

Loops

Python	`for i in range(100,-1,-5):` `    print i`
Java	`for (Integer i = 100; i >= 0; i -= 5)` `    System.out.println(i);`

Python	`for i in range(100,-1,-5):`
	`    print i`
	`    ...`

Java	`for (Integer i = 100; i >= 0; i -= 5){`
	`    System.out.println(i);`
	`    ...`
	`}`

Loops

Python	`for i in range(100,-1,-5):` `    print i`
Java	`for (Integer i` `    System.out`

¡No poner {}
es mala practica en general !

Python	`for i in range(100,-1,-5):`
	`    print i`
	`    ...`

Java	`for (Integer i = 100; i >= 0; i -= 5){`
	`    System.out.println(i);`
	`    ...`
	`}`

Loops II

Python	`while condition:`
	`  statement1`
	`  statement2`
	`  ...`

Java	`while (condition) {`
	`  statement1`
	`  statement2`
	`  ...`
	`}`

Ejemplo: Arreglos


```
int[] ages = new int[10];
String[] names = new String[10];
```

```
Scanner input = new Scanner(System.in);
for (int i = 0; i < names.length; i++)
  names[i] = input.nextLine("Enter a name: ");

for (int i = names.length; i >= 0 ;i--)
   System.out.println(names[i]);
```

Lo mas importante

Comentarios (en serio)

Python	`# Ese codigo`
	`# se entiende`
	`# por que tiene`
	`# comentarios!`

Java	`// Ese codigo`
	`// se entiende`
	`// por que tiene`
	`// comentarios!`

Python	`"""`
	`Ese codigo`
	`se entiende`
	`"""`

Java	`/*`
	`Ese codigo`
	`se entiende`
	`*/`

Referencias

- Java for Python Programmers (fuente de mayoria de ese material)
  - http://knuth.luther.edu/~bmiller/Java4Python/index.html
- From Python to Java
  - http://home.wlu.edu/~lambertk/pythontojava/index.htm
- Java API Documentation
  - http://docs.oracle.com/javase/6/docs/api/