Introducción a Java

Juan Manuel Fernández Peña

Curso 2005. Rev 2011

Tokens

- **Comentarios**:
  //comentario
  /* comentario
  fin de comentario */
  /** comentario para javadoc
  fin */

- **Palabras reservadas**:
  (como ejemplo)
  abstract, float, const, for, if, try, cast, int

Tokens

Identificadores:

- Secuencias de caracteres alfanuméricos y guión bajo;
- No debe comenzar con número;
- Se recomienda no comenzar con guión bajo.
- Importa si son mayúsculas o minúsculas
- Longitud máxima práctica: 31 caracteres
- Se usa mayúscula para inicio de nombres después del primero

- Se escribe tipo y luego nombre y opcionalmente valor inicial.
  int alfa;
  float ro = 1;
  Cuenta micta;
  String resp = "";

Tokens

Literales:

- Números: int, short, long float
- Caracteres entre apóstrofos 'a'
- Cadenas: entre comillas "cadena"
- Caracteres especiales: \b, \t, \n, \f, \r, \', \", \\
- Valores booleanos: true, false

Tokens

Operadores:

- Aritméticos: =, % (residuo), +, -, *, /, ++, --, +=, -=, %=, *=, /=
- Booleanos: ! (complemento), &, |, ^ (xor), &=, |=, ^=
- Condicionales <, >, <=, >=, ==, != &&, ||
- Op. Con bits: -, &, |, ^ <<, >>

Tokens

Variables

- Dos valores asociados:
  - Dirección(simbólica) y
  - valor

Referencias:

- El valor que almacenan es una dirección, permiten acceso indirecto; se usan para los objetos.

Proposiciones (statements)

Asignación

- <variable> = <expresión>
- Casos especiales <variable> <operador binario>= expresión
  - x += j;
    ++cont cont++

Servicios de objetos:

<nombre de objeto>.<servicio>(<parámetros>)

Bloques: {…} Se usa para clases, métodos, proposiciones compuestas en if, repeat, etc.

Los bloques se pueden anidar

Alcance de las variables

- La variable sólo puede usarse en el bloque donde se declara
- Ejemplos:
  - class xx{ int w; …}
  - public void met1(int a) { double z; …}
  - for (int jx=0; jx <= 10; jx++){ …}
  - if (cond) {boolean b; …}
- Las variables de clase pueden declararse
  - Públicas (todos los objetos las ven y pueden cambiar su valor)
  - Protegidas (las ven los objetos del mismo paquete (directorio)
  - Privadas (las ve solo la propia clase, ni siquiera sus subclases)

Modificadores aplicables a variables

- final: se declara la variable con un valor el cual será inalterable (es una constante)
  - final double pi = 3.1416;
  - Final String nom = "Sutanita de Tal"
- static: la variable será común a todos los objetos de la misma clase (peligroso, es como variable global)
  - Class Kla1 { static int ww1; …}
  - Class Ejem{final static double pi = 3.1416;

Proposiciones (statements)

Control de flujo: IF

- If (<expresión>) proposición>
- If (<expresión>) <proposición 1>; else <proposición 2>;
- **FOR**
- for (<proposición inicial>; <expresión 1>; <expresión 2>) <proposición>
- proposición inicial para definir índice, expresión 1: condición de parar; expresión 2 para cambiar índice
- for (int i=0; i<200; i++)

Proposiciones (statements)

while

while (<expresión booleana>) {proposición>}

do while

do <proposición>
while (<expresión booleana>);

switch

switch (<expresión entera>){
case <valor>:<proposiciones>; break;
default <proposiciones>;}

Proposiciones (statements)

Ramificaciones

- **Break**: rompe una secuencia de proposiciones dentro de un caso en un switch
  Rompe un for, while o do-while
  (el nivel más interno, si están anidados)
- **Continue**: termina el ciclo donde se encuentra y va al inicio (for, while o do-while
- **Return**: termina un método y puede regresar un valor:
  - return
  - return <expresión>

Manejo de arreglos

Declaración:

<tipo> [ ] nombre ;

Iniciación:

nombre = new <tipo> [tamaño];

<tipo> [ ] nombre = {valor, valor, ...}

Uso:

nombre[índice] = expresión; variable = nombre [índice];

Propiedad length:

nombre.length; devuelve # elementos

Inicio en cero

Excepciones

- Excepción cuando ocurre un error y el sistema trata de avisarnos; si no hacemos nada, el programa se detiene.
- Java tiene un mecanismo que se verá más adelante.
- A veces exige que se atrape (catch) o se listen los posibles errores.

Atrapar excepción:

try

{ <proposiciones que pueden fallar> **}**

catch (<nombre de excepción>**) {**<proposiciones para salvar el problema>**}**

Excepciones

Ejemplo:

```
try
{
   unalinea = ent.readLine();
   x = Integer.parseInt(unalinea);
   System.out.println("Mitad es "+(x/2));
} catch (Exception e)
   { System.out.println("Excepción: "+e); }
```

Espacio de nombres

- Para permitir carga dinámica de módulos, se tiene un alcance de cada identificador, para evitar conflictos (alcance: intervalo de vida)
- Cada variable es miembro (member) de una clase (class) y cada clase es parte de un paquete (package). El nombre calificado de una variable es:

  Paquete.clase.miembro
- Mientras está en un paquete, no se le menciona; si se está en una clase, no se menciona.
- Para otros paquetes, se importan y así no deben mencionarse
- Si no se especifica paquete, es el directorio donde está la clase.
- Si se especifica, es al principio de un archivo.
- Para importar: import <paquete>

Algunos paquetes de biblioteca

- **java.applet** clases para ejecutarse en browser
- **java.io** para maenjo de entrada y salida
- **java.lang** clases básicas para muchas operaciones
- **java.math** artimética de precisión arbitraria
- **java.util** tipos de datos muy usados
- **java.swing** elementos de interfaz

Entrada de datos

```
import java.io.*;
public class ensal1
{       //ejemplo de entrada y salida desde consola
  public static void main(String astrArgs[]) throws IOException
  {
    String lin;
    int c;
    System.out.println("*** C O M I E N Z A   E J E C U C I O N");
        //lectura de líneas como cuerdas
    BufferedReader ent = new BufferedReader(new InputStreamReader(
     System.in));
    lin = ent.readLine();
    System.out.println(lin);
    c = System.in.read();
    System.out.println("*** T E R M I N A   E J E C U C I O N");
  }     //fin del main
}
```

Entrada de datos

```
import java.io.*;
public class ensal2
{       //ejemplo de entrada y salida desde consola
  public static void main(String astrArgs[]) throws IOException
  {
    String lin;
    int c;
    int alfa, alfa2;
    float beta, beta2;
    Integer tt;
    Float  ff;
    Boolean bb;
    boolean gama, gama2;
    System.out.println("*** C O M I E N Z A   E J E C U C I O N");
```

Entrada de datos

```
    //lectura de líneas como cuerdas y luego conversión a números
    BufferedReader ent = new BufferedReader(new InputStreamReader(
     System.in));
    lin = ent.readLine();
    alfa = Integer.parseInt(lin);
    alfa2 = alfa + alfa;
    System.out.println("Resultado: alfa= "+alfa+" Doble: "+alfa2);
    lin = ent.readLine();
    beta = Float.parseFloat(lin);
    beta2 = beta + beta;
    System.out.println("Resultado: beta= "+beta+" Doble: "  +beta2);
    lin = ent.readLine();
    bb = Boolean.valueOf(lin);
    gama = bb.booleanValue();
    gama2 = !gama;
    System.out.println("Resultado: gama= "+gama+" Negado: "
      +gama2);
    c = System.in.read();
    System.out.println("*** T E R M I N A   E J E C U C I O N");
  }    //fin del main
}
```