Unidad 2
Fundamentos de Java

Sintáxis básica

Palabras reservadas

- En todo lenguaje de programación hay palabras con significado especial, que definen la gramática del lenguaje y que no se pueden utilizar para nombrar variables, métodos, clases, atributos, etc.
- Palabras reservadas en java:

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

- **true, false y null** son literales que también tienen significado especial, aunque no definen la gramática.

* Reservadas aunque no se pueden utilizar en el lenguaje

Variables y Tipos de Datos

Tipos primitivos

- Son tipos predefinidos con tamaños y rangos de valores conocidos. Las variables de tipos primitivos deben adecuar sus valores a dichos rangos.
- Tipos primitivos en java:

Tipo	Uso	Tamaño	Rango
byte	entero corto	8 bits	de −128 a 127
short	entero	16 bits	de −32.768 a 32.767
int	entero	32 bits	de −2.147.483.648 a 2.147.483.647
long	entero largo	64 bits	± 9.223.372.036.854.775.808
float	real precisión sencilla	32 bits	de −1032 a 1032
double	real precisión doble	64 bits	de −10300 a 10300
boolean	lógico	1 bit	true o false
char	texto	16 bits	cualquier carácter

Para evitar el **desbordamiento** de las variables, los rangos en Java funcionan de forma **circular**. El valor que sigue al máximo del rango es el mínimo de dicho rango.

Variables y Tipos de Datos

Declaración de variables y asignación de valor

- Para declarar una variable indicamos:

  tipo_variable nombreDeLaVariable ;

- Para asignar un valor a una variable utilizamos el =

  nombreDeLaVariable = nuevo_valor;

  donde nuevo_valor puede ser el resultado de una operación con números y/o valores, un valor concreto u otra variable

- Declaración y asignación a la vez:

  tipo_variable nombreDeLaVariable = valor ;

- Declaración de varias variables del mismo tipo en la misma instrucción:

  int a = 1, b=2, c=3 ;

**a, b y c son variables de tipo int con los valores asignados*

```
//declaración de una variable de tipo byte
byte varEjemplo;

//asignación de un valor a la variable varEjemplo
varEjemplo = 4;
```

Constantes

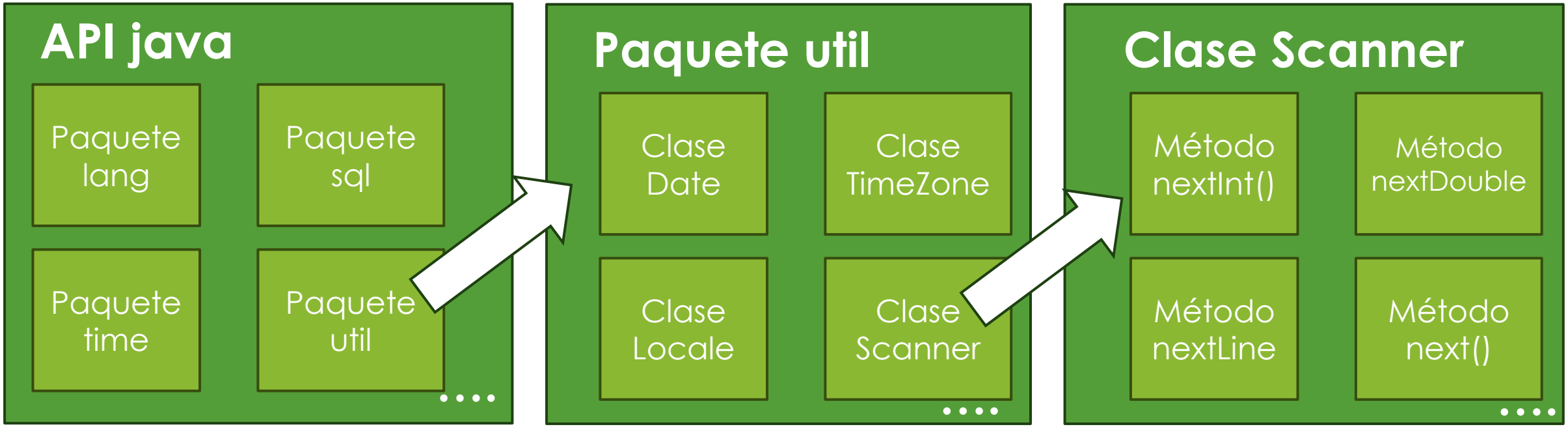
Constantes

- Son un tipo de variable especial cuyo valor permanece inmutable.
- El identificador de las constantes suele escribirse en mayúsculas
- Se declaran siguiendo esta sintaxis:

final *tipo_variable NOMBRE_CONST ;*

```
// declaración de constantes
final int RATIO_MAX_AULA = 32;
final double PI = 3.141592;
```

API de java



Para utilizar las clases y métodos de la API, hay que importar los paquetes/clases:

```
import java.util.* ; // importa todas las clases del paquete java.lang
import java.util.Date; // importa la clase 'Date', del paquete util, con todos sus métodos
```

Operadores aritméticos

- Para la asignación de un valor a una variable utilizamos el operador **=**
- Operadores aritméticos

Símbolo	Descripción
+	Suma
+	Más unario: positivo
–	Resta
–	Menor unario: negativo
*	Multiplicación
/	División
%	Módulo
++	Incremento en 1
––	Decremento en 1

PROMETEO
by thePower

Operadores relacionales

- Operadores relacionales

Símbolo	Descripción
==	Igual que
!=	Distinto que
<	Menor que
<=	Menor o igual que
>	Mayor que
>=	Mayor o igual que

- Operadores lógicos

Símbolo	Descripción
&&	Operador and: Y
||	Operador or: O
!	Operador not: negación

Operador ternario

- **Operador ternario**: Devuelve un valor entre dos posibles, la selección dependerá del resultado de la evaluación de una condición.
- Sintaxis:

  expresiónCondicional ? valor_si_verdadero : valor_si_falso;

  Se evalúa expresiónCondicional, y:

  -> Si es verdadero se devuelve el primer valor

  -> Si es falso, se devuelve el segundo valor

  El resultado del operador ternario se asigna a una variable cuyo tipo debe coincidir con el de los valores indicados en la expresión.

E12

Conversión de tipos

- Cuando se asigna un valor a una variable se debe asegurar que se mantiene la coherencia de los tipos de los datos.
  - A una variable de tipo entero se le asigna un valor entero, a una variable double se le asigna una variable o valor double, etc…
- En el caso de que no se mantenga esta coherencia, java hará la conversión automáticamente cuando el valor a asignar 'quepa' en la variable: **casting implícito**.
- Es decir, teniendo en cuenta la memoria (bits) que se reservan para los distintos tipos de variables.

Tipo	Uso	Tamaño
byte	entero corto	8 bits
short	entero	16 bits
int	entero	32 bits
long	entero largo	64 bits
float	real precisión sencilla	32 bits
double	real precisión doble	64 bits
boolean	lógico	1 bit
char	texto	16 bits

Un int 'cabe' en un double, por tanto, java hace la conversión automática si asignamos un int a una variable double.

Conversión de tipos

- Por el contrario, se da un error de compilación si se intenta asignar un valor que 'no cabe' a una variable, por ejemplo, asignar un valor double a una variable int.
- Si nos interesa por algún motivo forzar la conversión de tipos cuando java no la hace automáticamente, podemos aplicar un '**cast**' (int) al valor al que queremos cambiar el tipo: **casting explícito**

```
// se está haciendo un cast del valor 2.6, que es un double,
//para forzar el cambio a tipo entero y así
// poder asignárselo a la variable 'a', que es de tipo in
int a = (int) 2.6;
```