Clase de objeto en Java

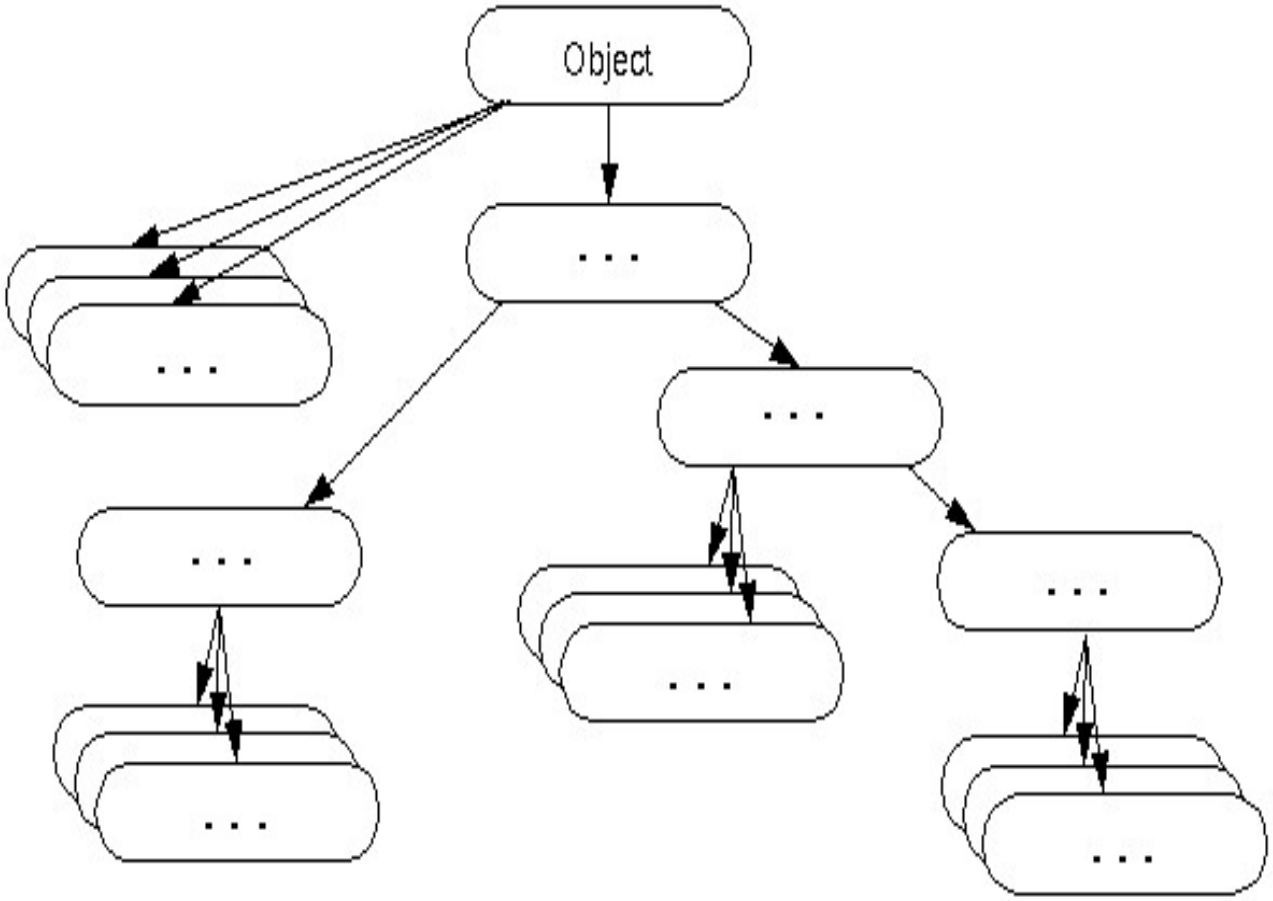
La **clase Object** es la clase padre de todas las clases en java por defecto. En otras palabras, es la clase más alta de Java.

La clase Object es beneficiosa si desea hacer referencia a cualquier objeto cuyo tipo no conoce. Tenga en cuenta que la variable de referencia de la clase principal puede hacer referencia al objeto de la clase secundaria, conocido como upcasting.

Tomemos un ejemplo, hay un método getObject () que devuelve un objeto pero puede ser de cualquier tipo como Empleado, Estudiante, etc., podemos usar la referencia de clase de Objeto para referir ese objeto. Por ejemplo:

```
1.  Objeto obj = getObject (); // no sabemos qué objeto se devolverá de este método
```

La clase Object proporciona algunos comportamientos comunes a todos los objetos, por ejemplo, el objeto se puede comparar, el objeto se puede clonar, el objeto se puede notificar, etc.



Métodos de clase de objeto

La clase Object proporciona muchos métodos. Son los siguientes:

Método	Descripción
public final Class getClass()	devuelve el objeto de clase Class de este objeto. La

	clase Class se puede usar para obtener los metadatos de esta clase.
public int hashCode()	devuelve el número de código hash para este objeto.
public boolean equals(Object obj)	compara el objeto dado con este objeto.
protected Object clone() throws CloneNotSupportedException	crea y devuelve la copia exacta (clon) de este objeto.
public String toString()	devuelve la representación de cadena de este objeto.
public final void notify()	despierta un solo hilo, esperando en el monitor de este objeto.
public final void notifyAll()	despierta todos los hilos, esperando en el monitor de este objeto.
public final void wait(long timeout)throws InterruptedException	hace que el subproceso actual espere los milisegundos especificados, hasta que otro subproceso notifique (invoca el método notify () o notifyAll ()).
public final void wait(long timeout,int nanos)throws InterruptedException	hace que el subproceso actual espere los milisegundos y nanosegundos especificados, hasta que otro subproceso notifique (invoca el método notify () o notifyAll ()).
public final void wait()throws InterruptedException	hace que el hilo actual espere, hasta que otro hilo notifique (invoca el método notify () o notifyAll ()).
protected void finalize()throws Throwable	es invocado por el recolector de basura antes de que el objeto sea recolectado.