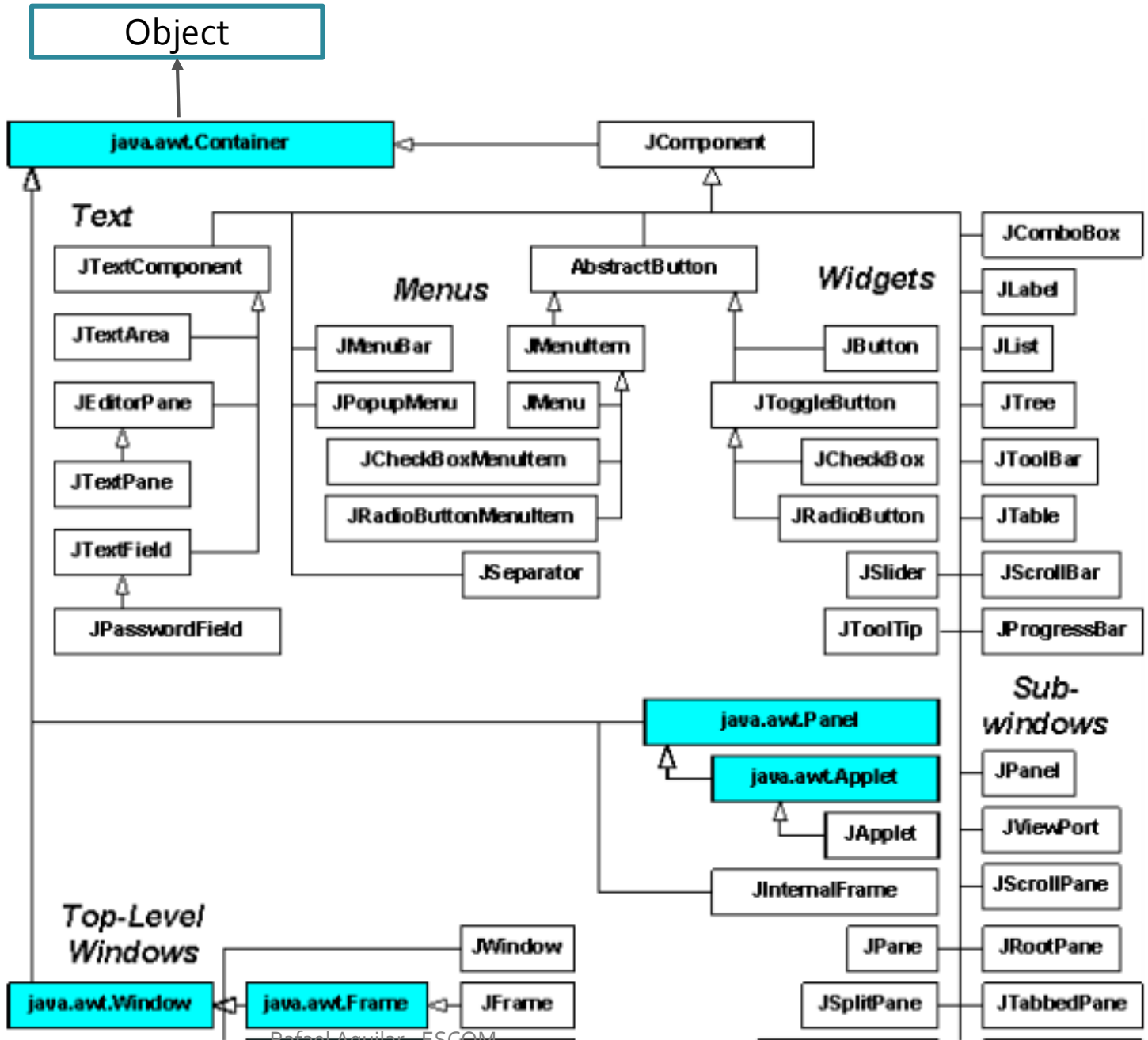
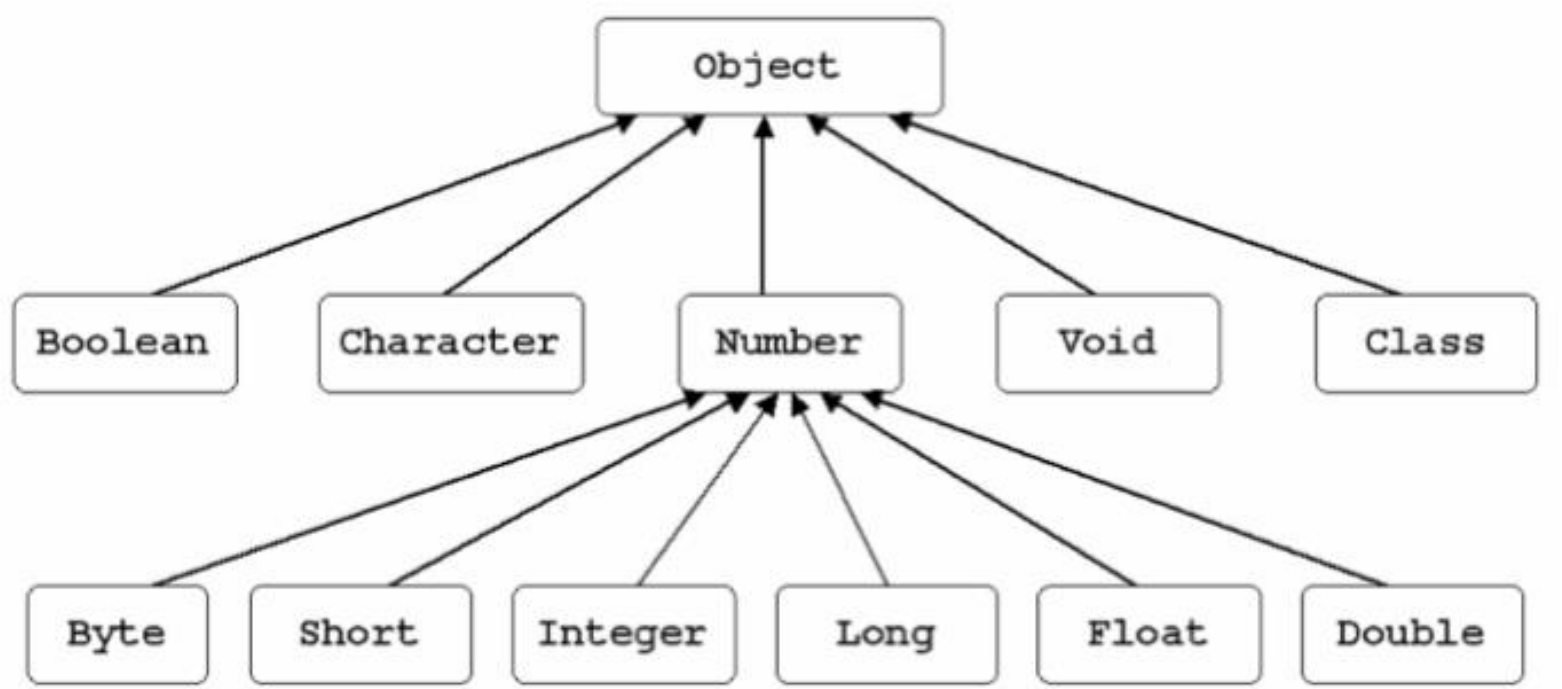
La clase *Object*

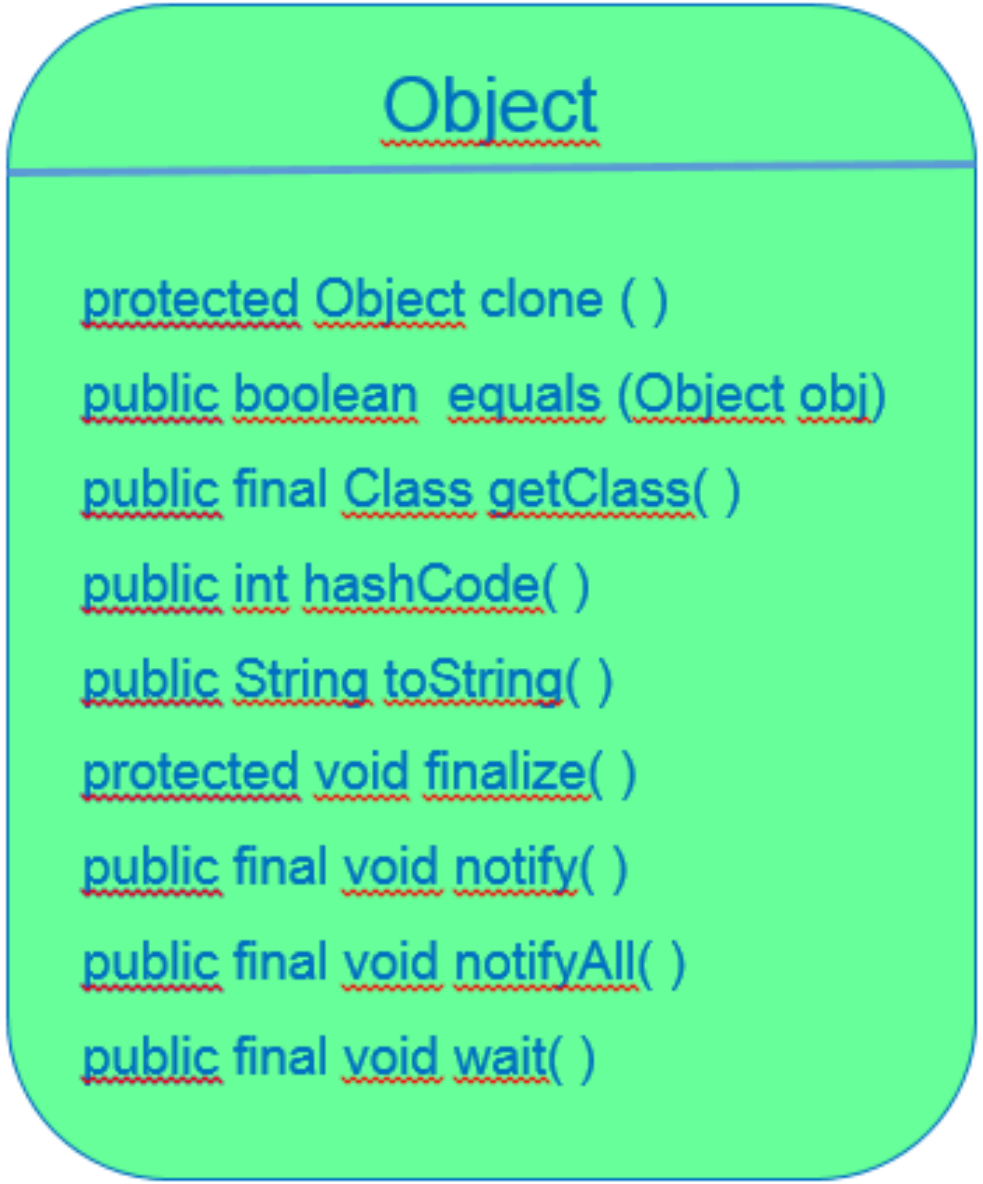

Object
java.awt.Container
JComponent
Text
JTextComponent
JTextArea
JEditorPane
JTextPane
JTextField
JPasswordField
Menus
JMenuBar
JPopupMenu
JCheckBoxMenuItem
JRadioButtonMenuItem
JSeparator
AbstractButton
JMenuItem
JMenu
Widgets
JButton
JToggleButton
JCheckBox
JRadioButton
JComboBox
JLabel
JList
JTree
JToolBar
JTable
JScrollBar
JProgressBar
JSlider
JToolTip
java.awt.Panel
java.awt.Applet
JApplet
JInternalFrame
Sub-windows
JPanel
JViewPort
JScrollPane
JRootPane
JTabbedPane
JPane
JSplitPane
Top-Level Windows
JWindow
java.awt.Window
java.awt.Frame
JFrame


Object
Boolean
Character
Number
Void
Class
Byte
Short
Integer
Long
Float
Double

Clase *Object*

- Es la clase base de todas las clases en Java, tanto para la API como para los tipos creados por el programador.
- Todas las clases heredan los métodos de la clase ***Object***.

Los métodos de la clase *Object*



Método	Descripción
protected Object clone ( )	Devuelve un objeto duplicado del objeto que invoca el método.
public boolean equals (Object obj)	Determina si el objeto que invoca el método es igual al objeto que recibe como parámetro. Se comparan las referencias de ambos objetos.
public final Class getClass( )	Obtiene la clase de un objeto en tiempo de ejecución.
public int hashCode( )	Devuelve el código hash asociado con el objeto que invoca el método.
public String toString( )	Devuelve una cadena que es la descripción de un objeto.
protected void finalize( )	Es un método que se ejecuta cuando el recolector de basura lo ha elegido para ser reciclado (Obsoleto a partir de JDK 9)
public final void notify( ) public final void notifyAll( ) public final void wait( )	Son métodos utilizados en programación multitarea o ***multithread*** de Java. Se utilizan para sincronización de procesos en programación concurrente.

El método *equals*

- Cuando comparamos dos objetos, usamos el método ***equals( )***
- La implementación de Java para ***equals( )***, en la clase ***Object***, compara las variables referencias de ambos objetos; pero no compara el contenido de los campos de los objetos.
- El programador tendrá que redefinir el método equals( ) para llevar a cabo la comparación semántica de los objetos .
- El operador == compara dos objetos con base en sus variables referencia (direcciones de memoria).
- **NOTA:** Hay algunas clases de la API de Java que sí realizan la comparación semántica de los campos de los dos objetos en comparación, tal es el caso de clases como ***String***, ***Date***, ***Integer***, ***Double***, etc.
- Veamos un ejemplo con la clase ***String***:

El método *equals*

```
public class Prueba{
  public static void main(String [ ] args){
        String s1 = new String("Soy un String");
        String s2 = new String("Soy un String");

         boolean res1 = (s1 == s2);
         boolean res2 =  s1.equals(s2);

         System.out.printn("s1==s2:  " + res1);
         System.out.printn("s1.equals(s2):  " + res2);
  }
}
```

Salida:

```
s1 == s2: false
s1.equals(s2): true
```

El método *getClass*

- Este método no se puede redefinir (override) debido a que está declarado como ***final***:

public final Class getClass( )

- El método ***getClass ( )*** devuelve un objeto ***Class***, que tiene métodos que pueden usarse para obtener información sobre la clase, como su nombre (***getSimpleName ( )***), su superclase (***getSuperclass ( )***) y las interfaces que implementa (***getInterfaces ( )***). Por ejemplo, el siguiente método obtiene y muestra el nombre de clase de un objeto:

El método *getClass*

```
void printClassName(Object obj) {
    System.out.println("The object's" + " class is " +
        obj.getClass().getSimpleName());
}
```

- La clase ***Class***, en el paquete ***java.lang***, tiene una gran cantidad de métodos (más de 50). Por ejemplo, puede probar para ver si la clase es una anotación (***isAnnotation ( )***), una interfaz (***isInterface ( )***) o una enumeración (***isEnum ( )***). Puede ver cuáles son los campos del objeto (***getFields ( )***) o cuáles son sus métodos (***getMethods ( )***), y así sucesivamente.