



**UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS**

Proyecto final Programación Orientada a Objetos

Blackjack en Java

Nicolás Alberto Rodríguez Delgado - 20202020019

Miguel Ángel Sabogal Maldonado - 20201020153

Julián Mateo Caicedo Ruiz - 20202020031

Facultad de Ingeniería

Ingeniería de Sistemas



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

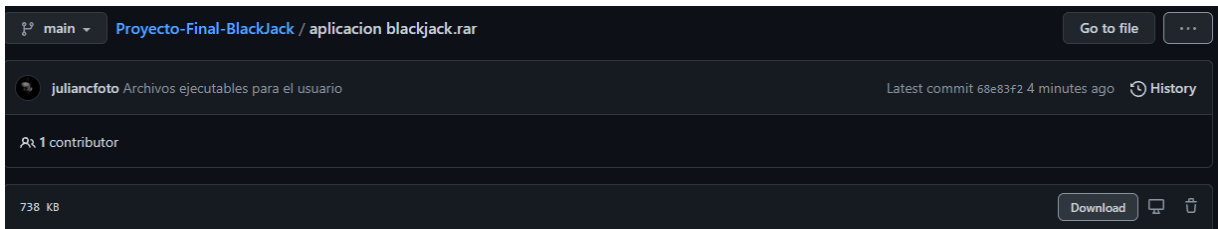
Manual de instalación del programa:

Para instalar el programa, debes entrar al siguiente enlace de GitHub: <https://github.com/juliancfoto/Proyecto-Final-BlackJack.git>, allí encontrarás un archivo llamado 'aplicacionblackjack.jar':

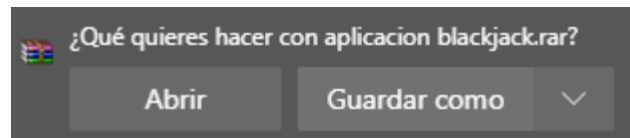
The screenshot shows the GitHub repository page for 'juliancfoto / Proyecto-Final-BlackJack'. The repository is public and has 4 branches and 0 tags. The file list includes:

File/Folder	Description	Last Update
Documentación	Última actualización	1 minute ago
aplicacion blackjack	Documentación completa (JavaDoc, jar, PDF)	9 minutes ago
img	Proyecto realizado, me falta el manejo de archivos	9 hours ago
src/proyectoFinal	Documentación completa (JavaDoc, jar, PDF)	9 minutes ago
.classpath	Primer commit del proyecto	4 days ago
.gitignore	Primer commit del proyecto	4 days ago
.project	Primer commit del proyecto	4 days ago
README.md	Initial commit	7 days ago
aplicacion blackjack.rar	Documentación completa (JavaDoc, jar, PDF)	9 minutes ago
diagrama_de_clases.uxf	Documentación completa (JavaDoc, jar, PDF)	9 minutes ago
manifest	Archivos ejecutables para el usuario	33 minutes ago

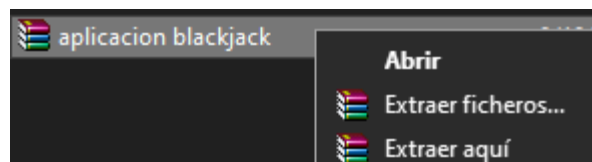
Abres el archivo y en la siguiente pantalla, deberás dar click en ‘Download’:



Lo guardas en donde desees:



Y le das click derecho -> extraer aquí:



Te aparecerá un archivo ejecutable .jar, el cual puedes abrir con click derecho -> Abrir o simplemente dando doble click:



Te aparecerá la siguiente ventana emergente y jeso es todo! ¡ya podrás jugar!:



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Manual del programador:

El programa consta de una clase 'Blackjack' creada con un entorno gráfico JFrame, encargada de ejecutar el programa y de albergar todos los ajustes gráficos y estéticos del programa, como lo son los botones, las layout, la posición de las cartas, la posición del jugador y de la banca y demás.

Consta de cuatro (4) clases que funcionan en unión con la clase 'Blackjack', las clases son:

- **'Carta'**: Es la clase padre. Contiene una variable 'valor', la cual se utilizará para asignar un valor a cada carta del programa y se utilizará de distintas maneras para saber cuál carta asignar al jugador y a la banca durante el juego.



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

```
public class Carta {  
  
    // Creamos variable de tipo private para poder retornar el valor,  
    private int valor;  
  
    public int getValor() {  
        return valor;  
    }  
    public void setValor(int valor) {  
        this.valor = valor;  
    }  
  
}
```

- **'Baraja'**: Hereda de la clase 'Carta'. Contiene cuatro métodos (llenarBaraja(), asigValor(), asigValor(String), retornarRandom(int)). En esta clase se harán todos los cambios y cálculos correspondientes con respecto a la baraja, ya sea para asignarle un valor numérico aleatorio y así saber cuál carta se utilizará, como también se le asignará un valor pero para posicionarlo en un punto de la pantalla, y también, un método para retornar un valor aleatorio que será mostrado al usuario como una carta.



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

```
public class Baraja extends Carta{

    // Declaramos variables que se van a utilizar.
    private String baraja[];
    private int valores[];

    // Creamos un método que llena la baraja de imágenes para después mostrarlas en pantalla.
    public void llenarBaraja() {
        baraja = new String [52];
        for (int i = 0; i < 13; i++) {
            baraja [i] = "img/Cartas/Negras/pica" + (i+1) + ".png";
        }
        for (int j = 13; j < 26; j++) {
            baraja [j] = "img/Cartas/Negras/trebol" + (j - 12) + ".png";
        }
        for (int k = 26; k < 39; k++) {
            baraja[k] = "img/Cartas/Rojas/corazon" + (k-25) + ".png";
        }
        for (int l = 39; l < 52; l++) {
            baraja[l] = "img/Cartas/Rojas/diamante" + (l-38) + ".png";
        }
    }

    // Creamos un método que le asigna a cada posición de la baraja el valor correspondiente.
    public void asigValor() {
        valores = new int [52];
        int temp = 0;
        for (int i = 1; i < 10; i++) {
            valores[i-1] = i;
        }
        for (int j = 14; j < 23; j++) {
            valores[j-1] = j-13;
        }
        for (int k = 27; k < 36; k++) {
            valores [k-1] = k-26;
        }
        for (int l = 40; l < 49; l++) {
            valores [l-1] = l-39;
        }
        for (int n = 9; n < 53;) {
            temp += n;
            for (int n2 = 0; n2 < 4; n2++) {
                valores [temp + n2] = 10;
            }
            n += 13;
        }
    }
}
```

- **‘ManoBanco’**: Hereda de la clase ‘Carta’. En esta clase se tiene un método que va sumando los valores de las cartas que van saliendo de la mano a la banca (sumaValB(int)).



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

```
public class ManoBanco extends Carta {  
    // Declaramos variables que se van a utilizar.  
    private int valorTotalBanca;  
  
    // Método que va sumando los valores de las cartas que van saliendo en la mano de la banca.  
    public int sumaValB(int valor) {  
        return valorTotalBanca += valor;  
    }  
  
    // Getters y Setters.  
    public int getValorTotal() {  
        return valorTotalBanca;  
    }  
  
    public void setValorTotal(int valorTotal) {  
        this.valorTotalBanca = valorTotal;  
    }  
}
```

- **‘ManoJugador’**: Hereda de la clase ‘Carta’. En esta clase se tiene un método que va sumando los valores de las cartas que van saliendo en la mano del jugador (sumaVal(int)).

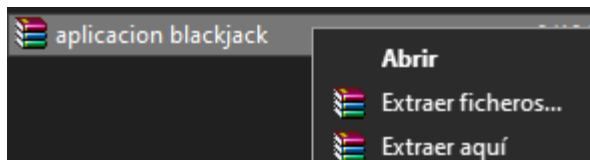


**UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS**

```
public class ManoJugador extends Carta{  
  
    // Declaramos variables que se van a utilizar.  
    private int valorTotal;  
  
    // Método que va sumando los valores de las cartas que van saliendo en la mano del jugador.  
    public int sumaVal(int valor) {  
        return valorTotal += valor;  
    }  
  
    // Getters y Setters.  
    public int getValorTotal() {  
        return valorTotal;  
    }  
  
    public void setValorTotal(int valorTotal) {  
        this.valorTotal = valorTotal;  
    }  
}
```


Manual de usuario:

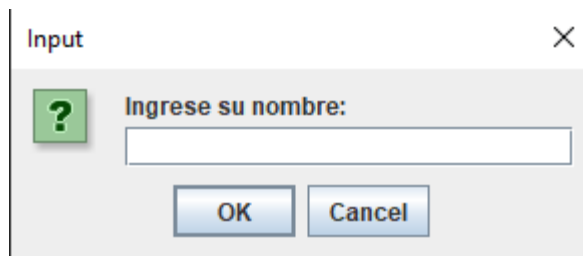
Primero deberás descargar y extraer el archivo aplicacionblackjack.zip:



Para ejecutar el programa, deberás abrir el archivo BlackJack.jar que se encuentra en la carpeta aplicacionblackjack.zip descargada:



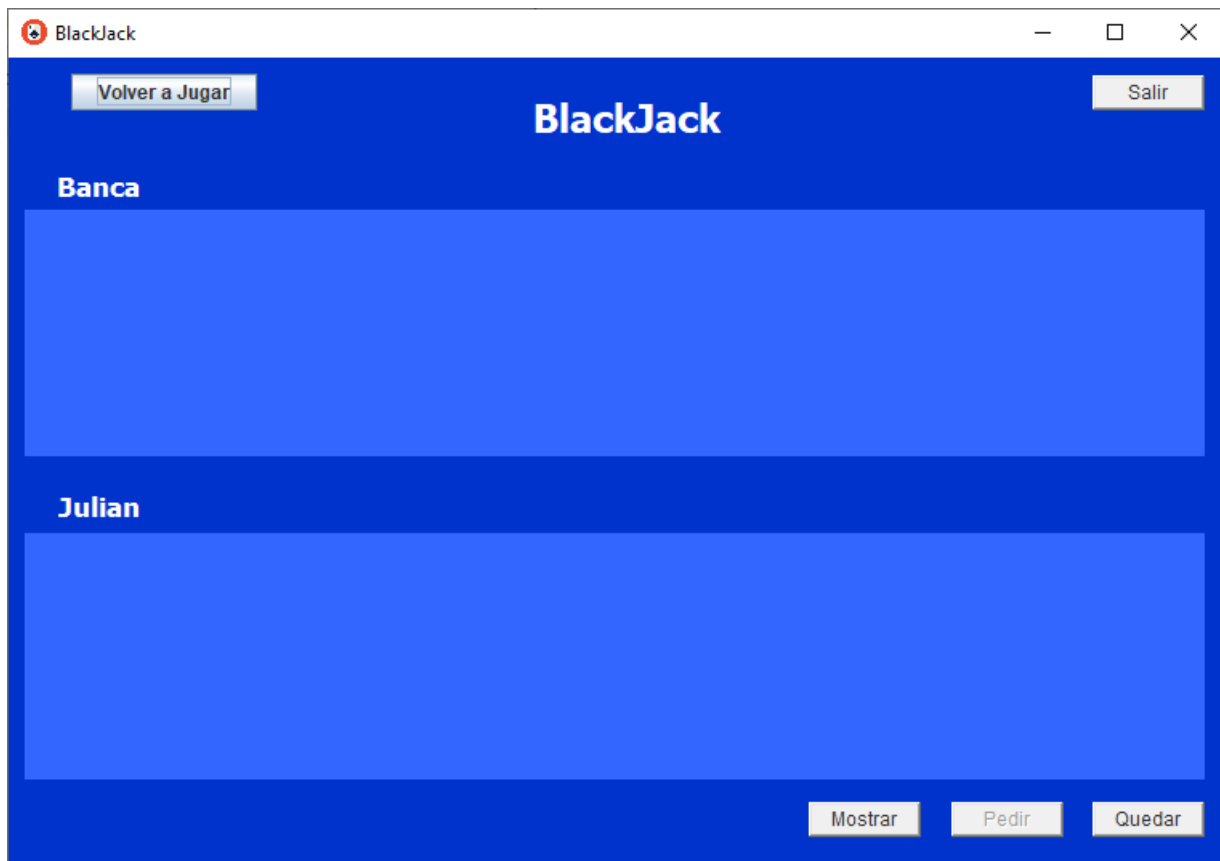
Una vez abierto el programa, te solicitará ingresar tu nombre como jugador:





UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

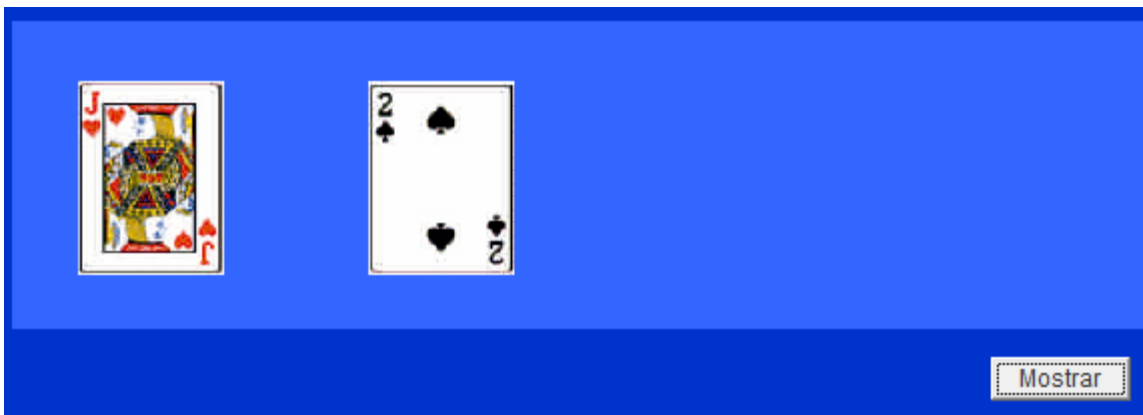
Te aparecerá la interfaz gráfica del programa, en la cual puedes interactuar con 5 botones: Volver a jugar, Salir, Mostrar, Pedir y Quedar.



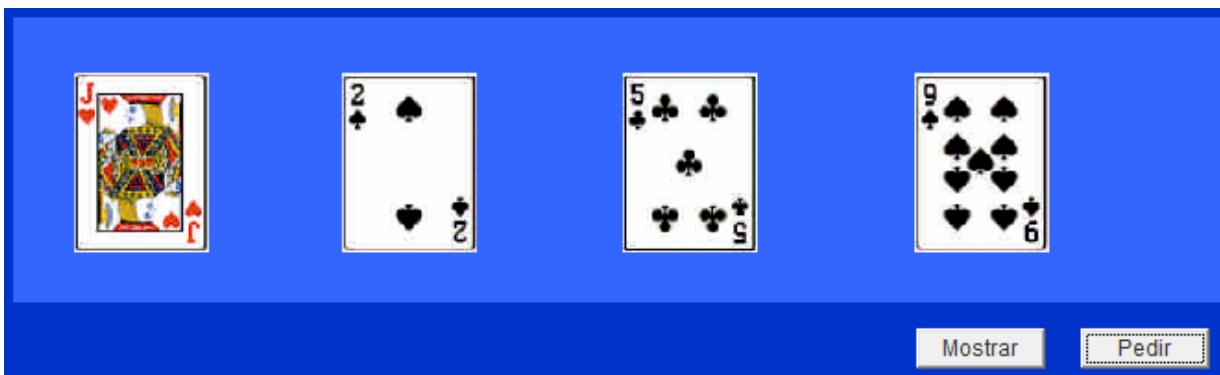


UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Ya podrás empezar a jugar. El primer paso es Mostrar las cartas:



Luego podrás Pedir cartas:











UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Y finalmente podrás Quedar, según sea conveniente para determinar si ganaste, perdiste o empataste:

Banca



Julian

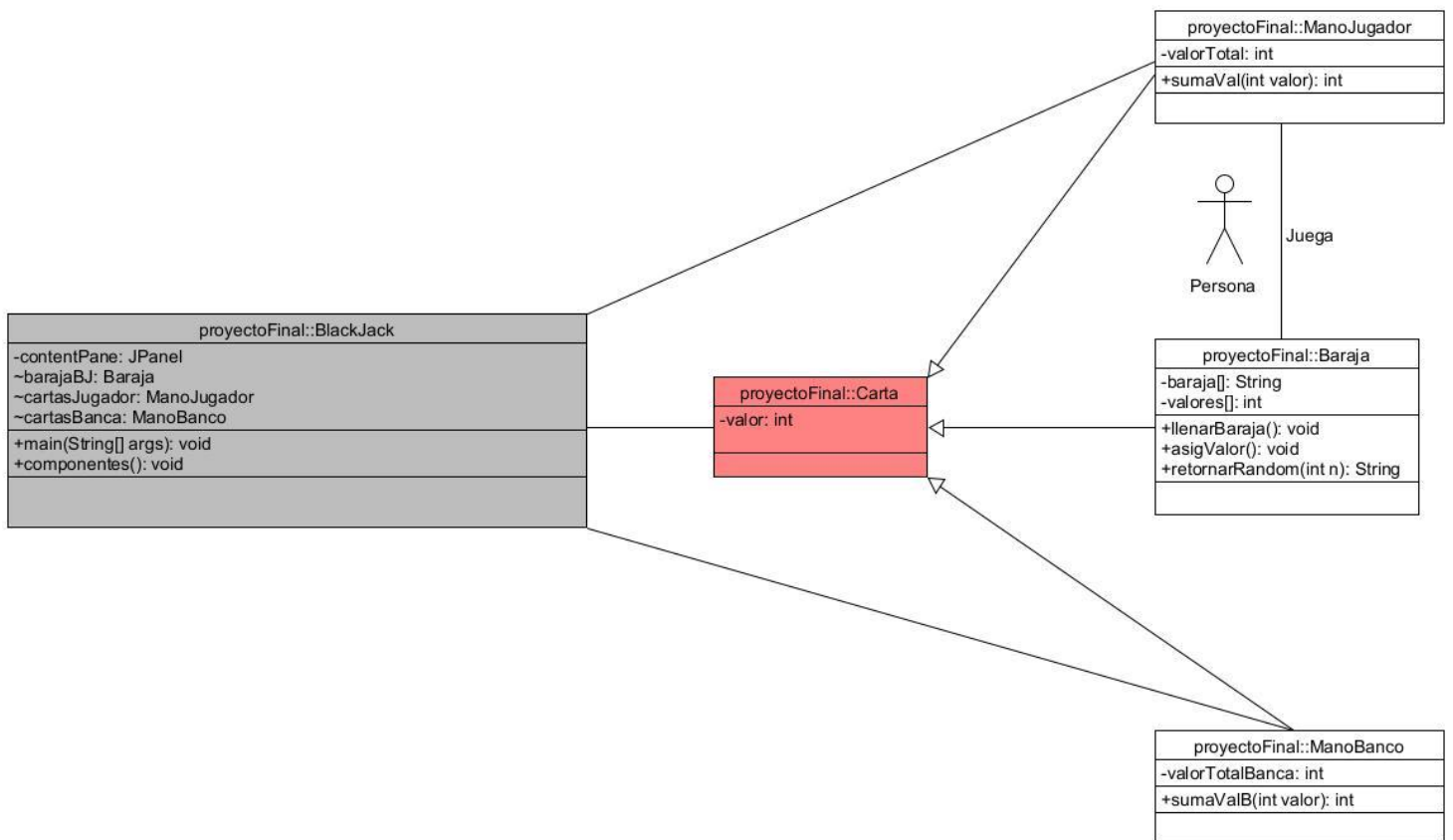




UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Diseño del programa:

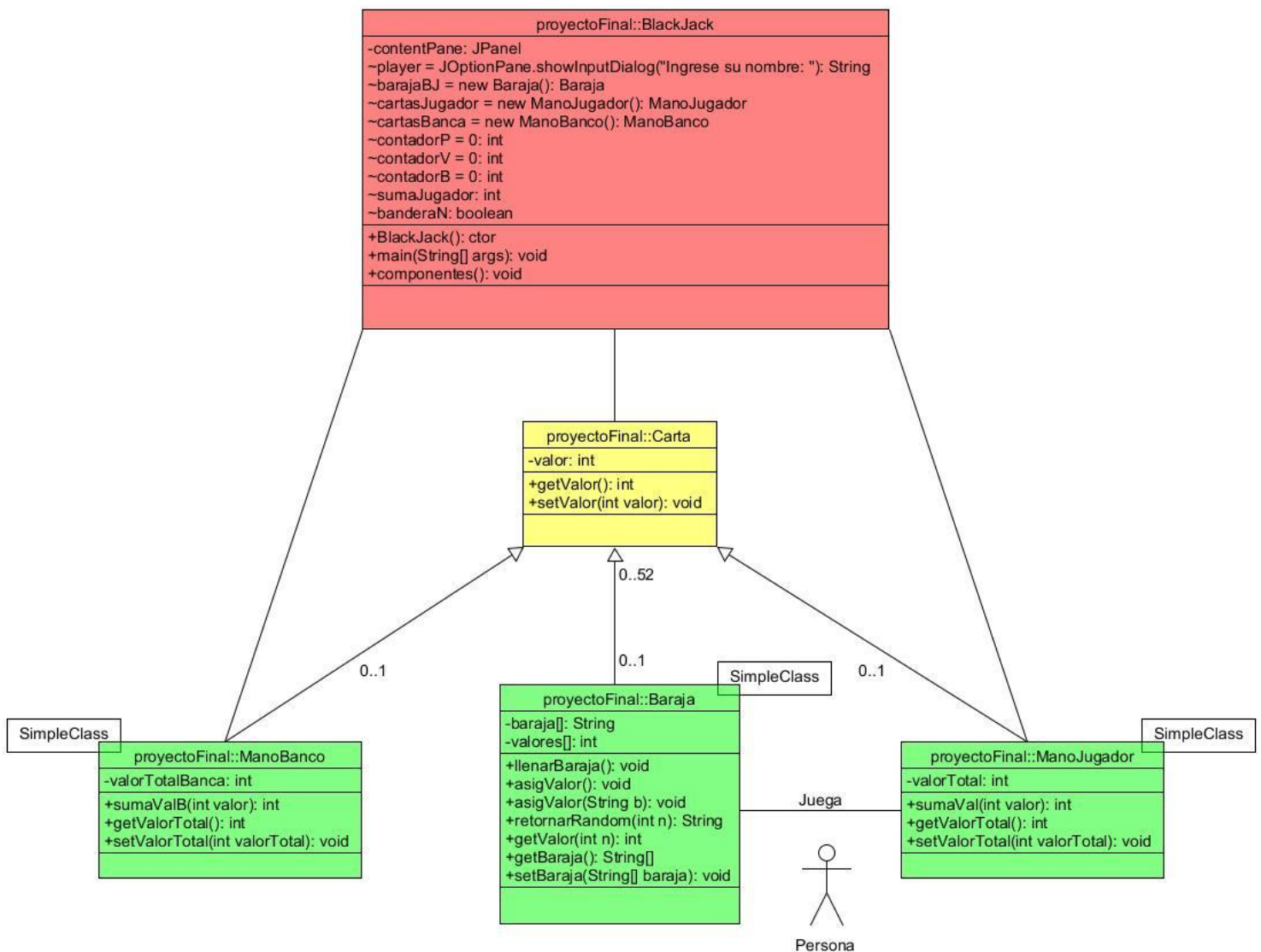
- Diagrama de clases PRE:





UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

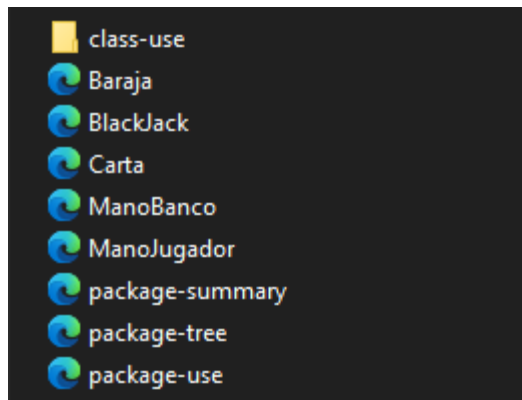
● Diagrama de clases POST:





UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Documentación JAVADOC:



En la carpeta general del programa, está la carpeta Documentación -> JavaDoc -> en donde se encuentra toda la documentación HTML

Ejemplo:

PACKAGE **CLASS** USE TREE INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD SEARCH:

Package proyectoFinal

Class BlackJack

java.lang.Object[Ⓢ]
 java.awt.Container[Ⓢ]
 java.awt.Window[Ⓢ]
 java.awt.Frame[Ⓢ]
 javax.swing.JFrame[Ⓢ]
 proyectoFinal.BlackJack

All Implemented Interfaces:
ImageObserver[Ⓢ], MenuContainer[Ⓢ], Serializable[Ⓢ], Accessible[Ⓢ], RootPaneContainer[Ⓢ], WindowConstants[Ⓢ]

public class BlackJack
extends JFrame[Ⓢ]

See Also:
Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class java.awt.Window[Ⓢ]

Window.Type[Ⓢ]

Nested classes/interfaces inherited from class java.awt.Component[Ⓢ]

Component.BaselineResizeBehavior[Ⓢ]