

MANUAL
TÉCNICO
ROLE PLAYING GAME
“PUBG CISCO”

201830419 Brayan Panjoj

CONTENIDO

INTRODUCCION.....	3
OBJETIVOS DEL PROGRAMA.....	4
GENERALES	4
ESPECÍFICOS	4
PROCESO DE PUBG.....	5
DIAGRAMA DE CLASES	6
SOPORTES PARA EL DESARROLLO.....	7
SISTEMA OPERATIVO	7
IDE.....	7
REQUISITOS PARA SU DESARROLLO	8
COMPONENTES PARA SU DESARROLLO.....	9
(JAVA)	9
INGRESO A PUBG CISCO	10
DESCRIPCIÓN DE PUBG CISCO (ORIGEN CODIGO)	11
JAVADOC.....	11
CREDITO (COMPAÑIA).....	14
TERMINOS DESCONOCIDOS (WIKIPEDIA)	14

INTRODUCCION

COMO PARTE DE LA PRÁCTICA 2, DEL CURSO INTRODUCCIÓN A LA PROGRAMACIÓN 1, SE PRESENTA EL SIGUIENTE MANUAL TÉCNICO, DONDE SE PRETENDE COMPRENDER Y EJECUTAR LOS CONOCIMIENTOS ADQUIRIDOS EN EL CURSO.

BASICAMENTE EL JUEGO SE BASA EN UN ROLE PLAYING GAME, DONDE EL USUARIO CUENTA CON 3 VEHICULOS PARA ATACAR A ENEMIGOS SOBRE UN ESCENARIO CON DISTINTOS TIPOS DE TERRENO.

EL DESARROLLO O LA PROGRAMACION DEL JUEGO ESTA BASADO EN LENGUAJE JAVA, A TRAVEZ DE NETBEANS, UN IDE DE DESARROLLO INTEGRADO LIBRE, HECHO PRINCIPALMENTE PARA EL LENGUAJE DE PROGRAMACIÓN JAVA.

OBJETIVOS DEL PROGRAMA

GENERALES

ELABORAR LA LÓGICA PARA LA SOLUCIÓN DEL PROBLEMA PLANTEADO, A TRAVES DEL LENGUAJE JAVA. APLICANDO LOS CONOCIMIENTOS ADQUIRIDOS DURANTE EL CURSO DE INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN 1.

ESPECÍFICOS

- UTILIZAR CONCEPTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS.
- USO DE INTERFAZ GRÁFICA EN JAVA, UTILIZANDO JFRAMES, JBUTTONS, ENTRE OTROS.
- USO DE RECURSIVIDAD.
- AMPLIAR EL CONOCIMIENTO DEL LENGUAJE JAVA.
- DESARROLLAR DIAGRAMAS DE CLASE COMO PARTE DEL ANÁLISIS DEL PROBLEMA.

PROCESO DE PUBG

1. EJECUTAR INICIO

2. MENU

- JUGAR
- AYUDA
- CREDITOS
- SALIR



3. JUGAR

- INICIAR PARTIDA
- ESTADISTICA
- USUARIO
- CREAR V/A
- REGRESAR



4. INICIAR PARTIDA

- PVM
- PVP
- TIENDA
- REGRESAR



5. REGISTRAR

- NICKNAME
- VEHICULOS (1,2,3)
- GUARDAR
- JUGAR
- REGRESAR



6. ESCENARIO

- 4*4
- 6*4
- 9*8
- CANCELAR

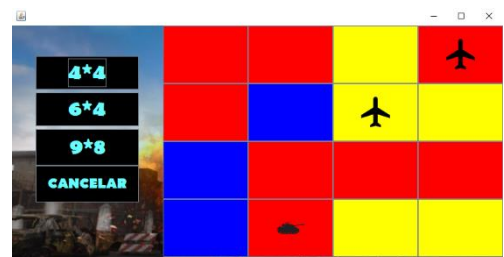
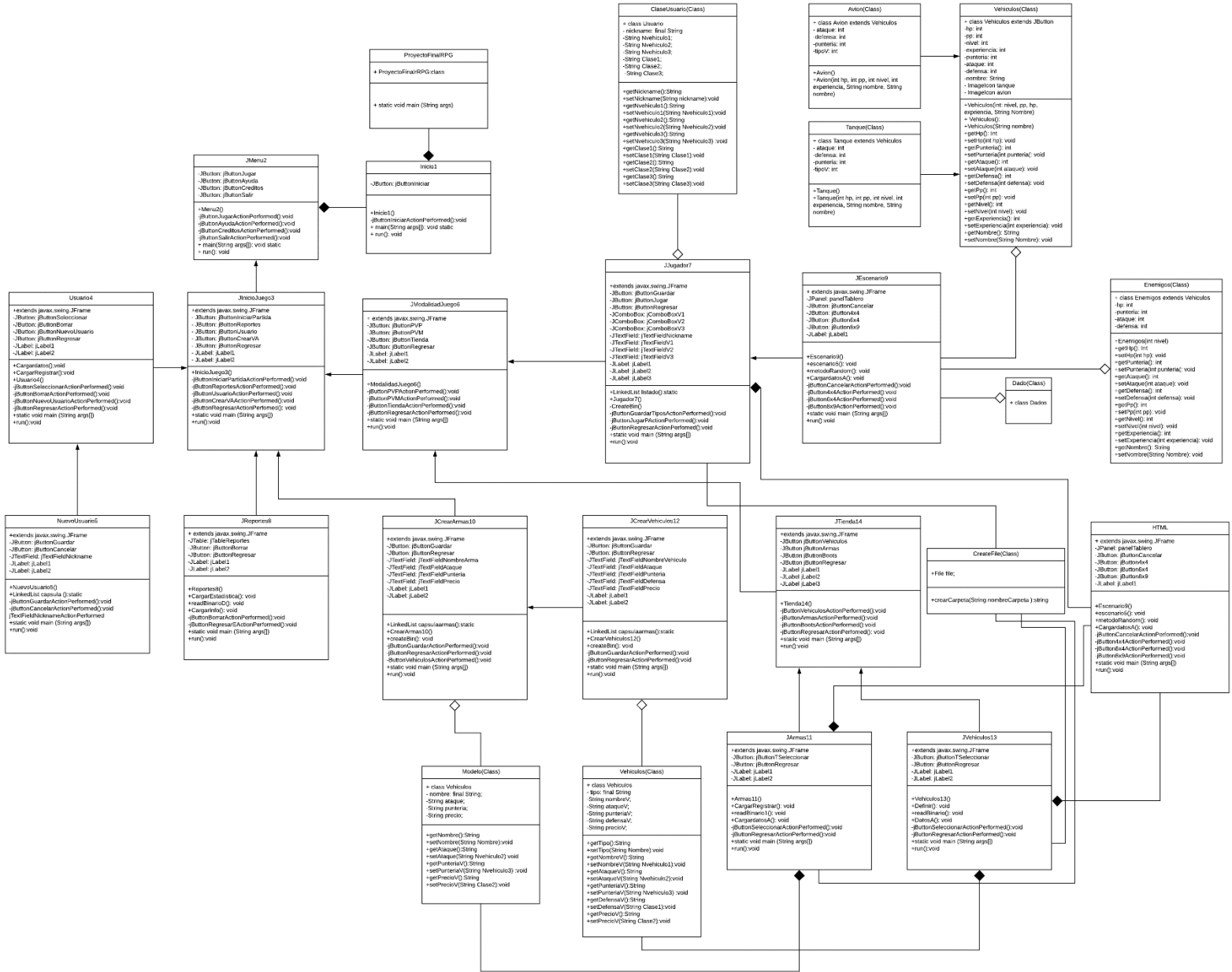


DIAGRAMA DE CLASES



SOPORTES PARA EL DESARROLLO

SISTEMA OPERATIVO

- LINUX (UBUNTU)
- WINDOWS
- MAC_OS

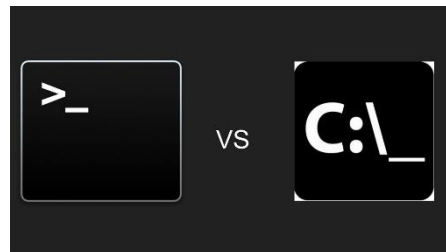


IDE

- NETBEANS (PREFERIBLE)
- INTELLIJ

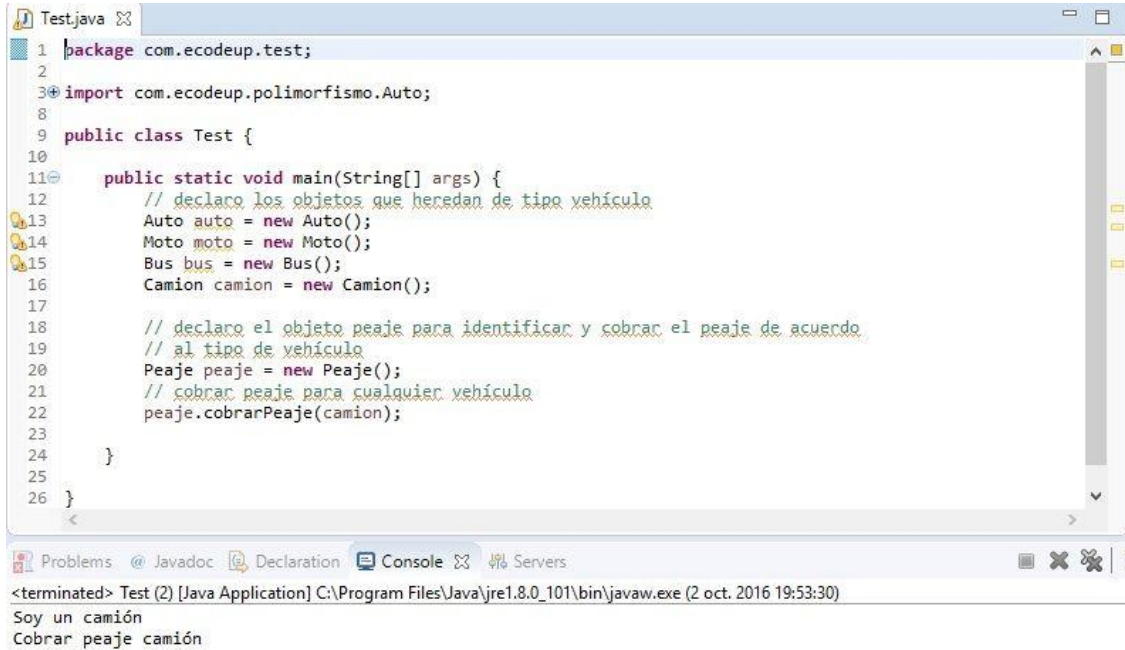


- CMD/Terminal
- EJECUTADOR (.JAR)



REQUISITOS PARA SU DESARROLLO

- COMPRENSION Y MANEJO DE CUALQUIER IDE, ESPECIALMENTE NETBEANS.
- COMPRENSION DE LENGUAJE JAVA (POO), RECURSIVIDAD, HILOS E INTERFAZ GRAFICA



The screenshot shows a Java IDE window titled 'Test.java'. The code defines a package, imports a class, and creates a 'Test' class with a 'main' method. The 'main' method creates instances of 'Auto', 'Moto', 'Bus', and 'Camion', and a 'Peaje' object. It then calls 'cobrarPeaje' on the 'Camion' object. The console output shows the execution of the program, displaying 'Soy un camión' and 'Cobrar peaje camión'.

```
1 package com.ecodeup.test;
2
3 import com.ecodeup.polimorfismo.Auto;
4
5 public class Test {
6
7     public static void main(String[] args) {
8         // declaro los objetos que heredan de tipo vehículo
9         Auto auto = new Auto();
10        Moto moto = new Moto();
11        Bus bus = new Bus();
12        Camion camion = new Camion();
13
14        // declaro el objeto peaje para identificar y cobrar el peaje de acuerdo
15        // al tipo de vehículo
16        Peaje peaje = new Peaje();
17        // cobrar peaje para cualquier vehículo
18        peaje.cobrarPeaje(camion);
19    }
20 }
21
22
23
24
25
26
```

Problems @ Javadoc Declaration Console Servers
<terminated> Test (2) [Java Application] C:\Program Files\Java\jre1.8.0_101\bin\javaw.exe (2 oct. 2016 19:53:30)
Soy un camión
Cobrar peaje camión

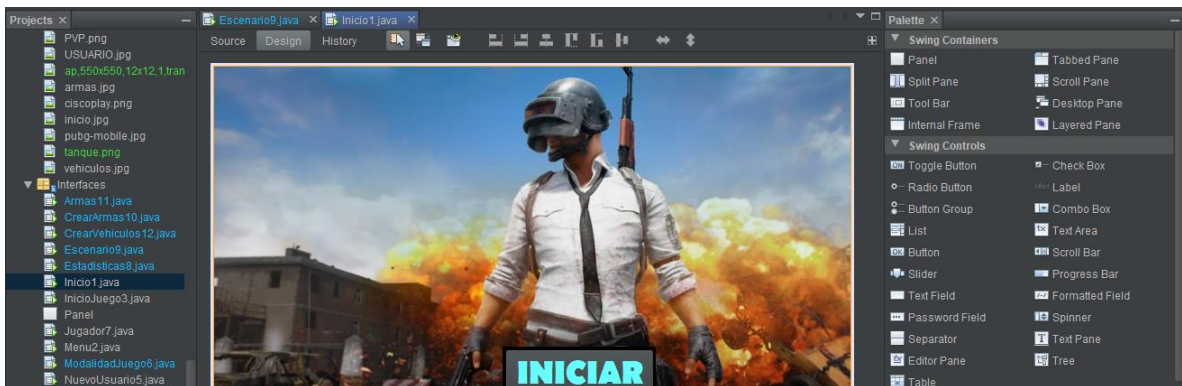
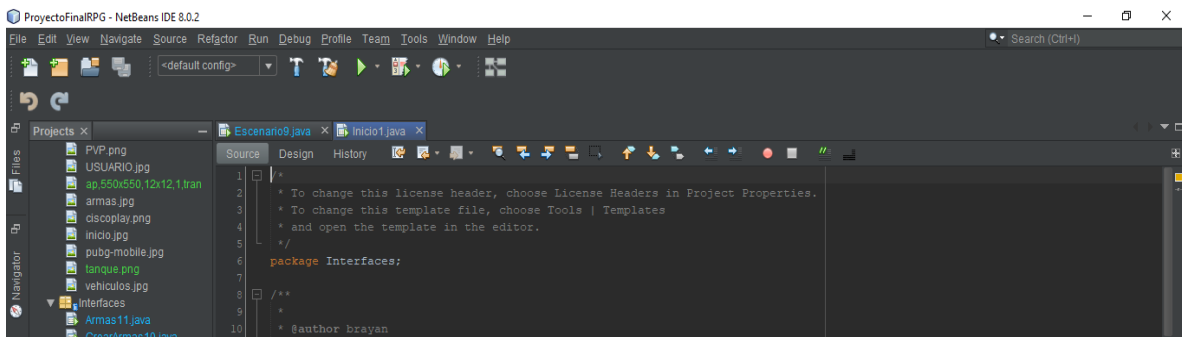


COMPONENTES PARA SU DESARROLLO (JAVA)

- **PACKAGE**
ES UN CONTENEDOR DE CLASES QUE PERMITE AGRUPAR LAS DISTINTAS PARTES DE UN PROGRAMA Y QUE POR LO GENERAL TIENE UNA FUNCIONALIDAD Y ELEMENTOS COMUNES, DEFINIENDO LA UBICACIÓN DE DICHAS CLASES EN UN DIRECTORIO DE ESTRUCTURA JERÁRQUICA.
- **CLASES**
SON PLANTILLAS PARA LA CREACIÓN DE OBJETOS, EN LO QUE SE CONOCE COMO PROGRAMACIÓN ORIENTADA A OBJETOS.
- **VARIABLES**
ES UN IDENTIFICADOR QUE REPRESENTA UNA PALABRA DE MEMORIA QUE CONTIENE INFORMACIÓN.
- **CONSTRUCTORES**
ES UN MÉTODO ESPECIAL DENTRO DE UNA CLASE, QUE SE LLAMA AUTOMÁTICAMENTE CADA VEZ QUE SE CREA UN OBJETO DE ESA CLASE.
- **METODOS**
UN MÉTODO EN JAVA ES UN CONJUNTO DE INSTRUCCIONES DEFINIDAS DENTRO DE UNA CLASE, QUE REALIZAN UNA DETERMINADA TAREA Y A LAS QUE PODEMOS INVOCAR MEDIANTE UN NOMBRE.
- **ARREGLOS**
ES UNA ESTRUCTURA DE DATOS QUE NOS PERMITE ALMACENAR UN CONJUNTO DE DATOS DE UN MISMO TIPO.

INGRESO A PUBG CISCO

EL PROYECTO COMPLETO LUEGO SE CARGA A LA TERMINAL O POR MEDIO DE NETBEANS IDEA O CUALQUIER IDEA CON LA CAPACIDAD DE EJECUTAR EL LENGUAJE JAVA (RECOMENDADO SI SE DESEA VER CAMBIOS EN CONSOLA). UNA VEZ CARGADO EL CÓDIGO CORRESPONDIENTE AL JUEGO SE UBICA LA CARPETA INICIO Y SE COMPILA EL CÓDIGO A TRAVÉS DE LA OPCIÓN DE NETBEANS PARA ACCEDER A LA CONSOLA Y A SU VEZ EJECUTAR LA INTERFAZ GRÁFICA Y SU CODIGO RESPECTIVAMENTE.



DESCRIPCIÓN DE PUBG CISCO (ORIGEN CODIGO)

DATO IMPORTANTE : LA DESCRIPCIÓN DEL JUEGO EJECUTABLE
LA ENCONTRARÁ EN EL MANUAL DE USUARIO.

JAVADOC

<file:///C:/Users/braya/Documents/NetBeansProjects/ProyectoFinalRPG/dist/javadoc/index.html>

Packages

Package	Description
---------	-------------

Interfaces

Class Summary

Class	Description
-------	-------------

Escenario

Inicio

JListas

Menu

Mesa

Interfaces

Class Inicio

- java.lang.Object
 - java.awt.Component
 - java.awt.Container
 - java.awt.Window
 - java.awt.Frame
 - javax.swing.JFrame
 -

- Interfaces.Inicio

- **All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer,
java.io.Serializable, javax.accessibility.Accessible,
javax.swing.RootPaneContainer, javax.swing.WindowConstants

```
public class Inicio  
extends javax.swing.JFrame
```

See Also:

[Serialized Form](#)

-

- ***Nested Class Summary***

- **Nested classes/interfaces inherited from class javax.swing.JFrame**

```
javax.swing.JFrame.AccessibleJFrame
```

- **Nested classes/interfaces inherited from class java.awt.Frame**

```
java.awt.Frame.AccessibleAWTFrame
```

- **Nested classes/interfaces inherited from class java.awt.Window**

```
java.awt.Window.AccessibleAWTWindow, java.awt.Window.Type
```

- **Nested classes/interfaces inherited from class java.awt.Container**

```
java.awt.Container.AccessibleAWTContainer
```

- **Nested classes/interfaces inherited from class java.awt.Component**

```
java.awt.Component.AccessibleAWTComponent,  
java.awt.Component.BaselineResizeBehavior,  
java.awt.Component.BltBufferStrategy,  
java.awt.Component.FlipBufferStrategy
```

- ***Field Summary***

- **Fields inherited from class javax.swing.JFrame**

```
accessibleContext, EXIT_ON_CLOSE, rootPane, rootPaneCheckingEnabled
```

- **Fields inherited from class java.awt.Frame**

CROSSHAIR_CURSOR, DEFAULT_CURSOR, E_RESIZE_CURSOR, HAND_CURSOR, ICONIFIED, MAXIMIZED_BOTH, MAXIMIZED_HORIZ, MAXIMIZED_VERT, MOVE_CURSOR, N_RESIZE_CURSOR, NE_RESIZE_CURSOR, NORMAL, NW_RESIZE_CURSOR, S_RESIZE_CURSOR, SE_RESIZE_CURSOR, SW_RESIZE_CURSOR, TEXT_CURSOR, W_RESIZE_CURSOR, WAIT_CURSOR

- **Fields inherited from class java.awt.Component**

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

- **Fields inherited from interface javax.swing.WindowConstants**

DISPOSE_ON_CLOSE, DO_NOTHING_ON_CLOSE, HIDE_ON_CLOSE

- **Fields inherited from interface java.awt.image.ImageObserver**

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

- **Constructor Summary**

Constructors

Constructor and Description

`Inicio()`

Creates new form Inicio

-

CREDITO (COMPAÑIA)



TERMINOS DESCONOCIDOS (WIKIPEDIA)

Java.Util: Clases de utilidades para el lenguaje, como acceso a recursos del sistema.

JPanel: Los JPanel en Java son objetos contenedores, la finalidad de estos objetos es la agrupación de otros objetos tales como botones, campos de texto, etiquetas, selectores, etc;

JFrame: JFrame es una clase utilizada en Swing para generar ventanas sobre las cuales añadir distintos objetos con los que podrá interactuar o no el usuario.

JLabel: Un Label o etiqueta puede mostrar texto plano, una imagen o una imagen con un texto.

JTextField: Un JTextField o campo de texto es un componente utilizado para la captura de datos, estos son casi indispensables en una aplicación.

JButton: En informática un botón es una metáfora común, utilizada en interfaces gráficas con objetivo similar al de un botón corriente. Los botones suelen ser representados como rectángulos con una leyenda o icono dentro.