Programa de gestión de una Base Aérea Militar

**AirBase**

Parte 2

Omar Hamza

Constantino Lapaz Giménez

Juan Mauricio González Arenas

Rafael Antonio Vázquez Flores

DAM Sector

Diagrama

Descripción generada automáticamente

Indice

[Punto de partida 1](#_Toc160736821)

[Trello 1](#_Toc160736822)

[Estructura del Proyecto 2](#_Toc160736823)

[Controladores 3](#_Toc160736824)

[Entidades 3](#_Toc160736825)

[Interfaces 3](#_Toc160736826)

[Main 4](#_Toc160736827)

[Servicios 4](#_Toc160736828)

[Menu.jsp y Test 4](#_Toc160736829)

[Diagrama proyecto 5](#_Toc160736830)

[Diagrama funcional del Proyecto v1 6](#_Toc160736831)

[Estructura de Código 9](#_Toc160736832)

[Esta sería la estructura del menú: 9](#_Toc160736833)

[Estructura de Interface 17](#_Toc160736834)

[Estructura del main. 22](#_Toc160736835)

[Estructura de Servicios 23](#_Toc160736836)

[Resumen 43](#_Toc160736837)

[3ºTrimestre 43](#_Toc160736838)

[Nota final 43](#_Toc160736839)

# Punto de partida

Nuestro objetivo principal en este segundo trimestre es unificar todas nuestras partes individuales (aunque ya estaba en un mismo proyecto, estaban en clases distintas sin utilizar realmente el código de los compañeros).

Ahora lo que deseamos hacer es trabajar como realmente lo hacen los grupos/equipos de trabajo, lo que implica que todos tenemos visibilidad constante del trabajo de todo el equipo, y todos podemos tocar todos los códigos.

Ahora el trabajo se estructurará mediante Clases y llamados a dichas clases para ejecutarse, nos olvidaremos completamente de la estructura y metodología anteriores para comenzar a generar un proyecto serio y acostumbrarse a como se trabaja realmente.

# Trello

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamentePara este punto del proyecto modificamos la estructura de trello simplificándola mucho, ya que todos tocamos el código, lo único que debemos saber es que tenemos echo, actualización de los commit realizado en git-Hub y la documentación que vamos generando.

# Texto Descripción generada automáticamenteEstructura del Proyecto

Usamos la estructura generada de manera estándar al crear un proyecto Spring Boot esto nos ayuda a trabajar con el estándar de nuestra profesión, acostumbrándonos a este tipo de proyectos.

Texto

Descripción generada automáticamenteGeneramos las también las estructuras necesarias para nuestras clases, ya que recordemos que ahora unificaremos todos nuestros proyectos individuales en uno solo.

Creamos los paquetes necesarios la estructura que necesitaremos si queremos usar tanto interfaces como conexiones web (que, aunque no es necesario experimentaremos con ello en este segundo trimestre) también generamos la carpeta donde realizaremos los test unitarios.

## Controladores

Empezando por *src/main/java* donde se alojará nuestra mayor parte del código funcional, aquí solo tenemos una única clase que es la gestión de los menus. Es en esta clase donde controlamos la visualización de nuestro código, todas las sysos necesarias.

## Texto, Chat o mensaje de texto Descripción generada automáticamenteEntidades

En el pakage entidades tenemos todos los constructores de cada una de las clases que tenemos.

## Texto Descripción generada automáticamenteInterfaces

Una interfaz es una especie de plantilla para la construcción de clases. Normalmente una interfaz se compone de un conjunto de declaraciones de cabeceras de métodos (sin implementar, de forma similar a un método abstracto) que especifican un protocolo de comportamiento para una o varias clases. Además, una clase puede implementar una o varias interfaces: en ese caso, la clase debe proporcionar la declaración y definición de todos los métodos de cada una de las interfaces o bien declarar la clase *abstract*. Por otro lado, una interfaz puede emplearse también para declarar constantes que luego puedan ser utilizadas por otras clases.

## Main

La clase main es la que permite ejecutar el código, es la cara visible del programa cuando damos a RUN

## Texto Descripción generada automáticamenteServicios

El pakage de servicios gestiona toda la funcionalidad de nuestro código cualquier funcion o código que necesitemos para que nuestro programa funcione respectivamente esta agrupada en estas clases, esto nos permite saber dónde buscar el código para poder solventar rápidamente nuestros problemas.

## Pantalla negra con letras blancas Descripción generada automáticamenteMenu.jsp y Test

Aquí se estructura en la parte HTML del proyecto referente a la parte web y en test, la clase java que nos permite hacer los test unitarios de nuestro proyecto.

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

# Diagrama Descripción generada automáticamenteDiagrama proyecto

## Diagrama Descripción generada automáticamenteDiagrama funcional del Proyecto v1

La idea de partida del proyecto es tener un menú donde accederemos a las distintas funcionalidades de nuestro código, en principio generaremos dos perfiles, usuarios Admin donde accederán a los registros tanto de Aviones de piloto, así como la gestión de incidencias, el otro menú será el de piloto, el cual permite ver una ficha del propio piloto donde podrá consultar sus misiones realizadas, las incidencias interpuestas o incluso los aviones pilotados.

Diagrama

Descripción generada automáticamente

El diagrama anteriormente presentado es como debería de funcionar el menú en una primera versión.

Accederemos a nuestro programa a través de un Código de piloto, si el piloto posee las credenciales de admin aparecerá un menú, que le preguntará si desea acceder como ADMIN o si dese acceder como piloto, según la elección accedera a un área del programa u a otra,

La sección admin:

* Podrá agregar o eliminar a Pilotos:
* Podra agregar o eliminar Aviones
* Cambiar o modificar las Incidencias
* Agregar las misiones que los pilotos podrán elegir
* Y un listado de informes de lo que desee para tener la información actualizada.

La sección de Piloto dispondrá:

* Ficha del Piloto
* Incidencias
* Aviones Pilotados
* Y Misiones, donde podrá consultar las realizadas y elegir la misión que va a realizara a continuación, lo importante aquí es que podrá elegir entre las que le haya asignado el Admin.

# Estructura de Código

A continuación, detallare las estructuras de código.

### Esta sería la estructura del menú:

package com.dam.airbase.controladores;

import com.dam.airbase.servicios.GestionAvionCRUD;

import com.dam.airbase.entidades.Avion;

import com.dam.airbase.entidades.Piloto;

import com.dam.airbase.entidades.Registro;

import com.dam.airbase.interfaces.Constantes;

import com.dam.airbase.servicios.GestionIncidencias;

import com.dam.airbase.servicios.GestionPilotoCRUD;

import com.dam.airbase.servicios.GestionRuta;

import com.dam.airbase.controladores.Menus;

public class Menus {

static int *opcion* = 0;

//Llamamos a los Servicios, y lo Instanciamos para poderlos inicializar

private static Registro *r*;

private static GestionRuta *gr* = new GestionRuta();

private static GestionPilotoCRUD *gpc* = new GestionPilotoCRUD();

private static GestionAvionCRUD *gac* = new GestionAvionCRUD();

private static GestionIncidencias *gi* = new GestionIncidencias();

private static Menus *m* = new Menus();

public void menu(String[] args) {

//Creamos las variables Hijo del Padre

Piloto p = (Piloto)*r*;

Avion a = (Avion)*r*;

//Inicializamos arrayList

*gpc*.inicializar(p);

*gac*.inicializar(a);

*gi*.inicializarIncidencias();

*gr*.inicializarRutas();

cabecera();

//gi.incidenciaPiloto(gpc.pilotos);

int opcion;

do {

menuP();

opcion = Constantes.*sc*.nextInt();

switch (opcion) {

case 1:

menuGestionP(*gpc*, "Pilotos");

break;

case 2:

menuGestionA(*gac*, "Aviones");

break;

case 3:

menuGestionIncidencias(*gi*, *gpc*, *gac*);

break;

case 4:

menuGestionRuta(*gr*, "Rutas");

break;

case 5:

salir();

break;

default:

error();

}

} while (opcion != 5);

}

private static void menuGestionRuta(GestionRuta gr, String entidad) {

int opcion;

do {

*m*.gestionR();

opcion = Constantes.*sc*.nextInt();

switch (opcion) {

case 1:

gr.mostrarPaises();

String carga = *gpc*.elegirCarga(*gpc*.tipoCarga(Constantes.*sc*),Constantes.*sc*,*gpc*.tiposCargamento());

*gpc*.mostrarCarga(carga);

*gpc*.validarPiloto2();

*gac*.validarAvion2();

gr.mostrarRuta();

break;

case 2:

gr.mostrarPaises();

break;

case 3:

gr.verRutas(gr.obtenerRutas());

break;

case 4:

System.*out*.println(" --~{ Volviendo al Menú Principal.}~-- ");

break;

default:

*m*.error();

}

} while (opcion != 4);

}

private static void menuGestionA(GestionAvionCRUD gestion, String entidad) {

int opcion;

do {

*m*.menuGestion(entidad);

opcion = Constantes.*sc*.nextInt();

switch (opcion) {

case 1:

gestion.alta(null); // Puedes pasar un Registro si es necesario

break;

case 2:

gestion.baja(null);

break;

case 3:

gestion.modificar(null);

break;

case 4:

gestion.ver(null);

break;

case 5:

System.*out*.println(" --~{ Volviendo al Menú Principal.}~-- ");

break;

default:

*m*.error();

}

} while (opcion != 5);

}

// Método para mostrar menú y gestionar CRUD

private static void menuGestionP(GestionPilotoCRUD gestion, String entidad) {

int opcion;

do {

*m*.menuGestion(entidad);

opcion = Constantes.*sc*.nextInt();

switch (opcion) {

case 1:

gestion.alta(null); // Puedes pasar un Registro si es necesario

break;

case 2:

gestion.baja(null);

break;

case 3:

gestion.modificar(null);

break;

case 4:

gestion.ver(null);

break;

case 5:

System.*out*.println(" --~{ Volviendo al Menú Principal.}~-- ");

break;

default:

*m*.error();

}

} while (opcion != 5);

}

private static void menuGestionIncidencias(

GestionIncidencias gestionIncidencias, GestionPilotoCRUD gestionPilotoCRUD,

GestionAvionCRUD gestionAvionCRUD) {

int opcion;

do {

*m*.menuInciden();

opcion = Constantes.*sc*.nextInt();

switch (opcion) {

case 1:

gestionIncidencias.incidenciaPiloto(gestionPilotoCRUD.obtenerPilotos());

break;

case 2:

gestionIncidencias.incidenciaAvion(gestionAvionCRUD.obtenerAviones());

break;

case 3:

gestionIncidencias.verIncidencias();

break;

case 4:

System.*out*.println(" --~{ Volviendo al Menú Principal.}~-- ");

break;

default:

*m*.error();

}

} while (opcion != 4);

}

// Camisa de ONCE VARAS (complicamos el codigo Demasiado Funciones no utilizadas

public void menu() {

System.*out*.println(" Bienvenido Almirante ");

System.*out*.println(" ¿Como deseas acceder?");

System.*out*.println(" [1]. Administrador ");

System.*out*.println(" [2]. Piloto");

System.*out*.println(" [6]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

public void piloto() {

System.*out*.println(" Bienvenido Piloto ");

System.*out*.println(" [1]. Ficha ");

System.*out*.println(" [2]. Incidencias");

System.*out*.println(" [3]. Misiones ");

System.*out*.println(" [4]. Aviones Pilotados");

System.*out*.println(" [5]. Proxima Misión");

System.*out*.println(" [6]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

public void ficha() {

System.*out*.println(" Bienvenido Piloto ");

System.*out*.println(" [1]. Horas de Vuelo ");

System.*out*.println(" [2]. Misiones Realizadas");

System.*out*.println(" [3]. Incidencias ");

System.*out*.println(" [4]. volver");

System.*out*.println(" [5]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : " );

}

public void piloInciden() {

System.*out*.println(" Bienvenido Piloto "); //añadir Nombre del Piloto

System.*out*.println(" [1]. Registrar Incidencia ");

System.*out*.println(" [2]. Historico Incidencia ");

System.*out*.println(" [3]. Incidencias ");

System.*out*.println(" [4]. Volver");

System.*out*.println(" [5]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

public void piloAvion() {

System.*out*.println(" Bienvenido Piloto ");

System.*out*.println(" [1]. Aviones Pilotados ");

System.*out*.println(" [2]. Volver");

System.*out*.println(" [3]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

public void piloMision() {

System.*out*.println(" Bienvenido Piloto ");

System.*out*.println(" [1]. Misiones Realizadas ");

System.*out*.println(" [2]. Elegir Mision");

System.*out*.println(" [3]. Volver");

System.*out*.println(" [4]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

public void admin() {

System.*out*.println(" Bienvenido Admin ");

System.*out*.println(" [1]. Pilotos ");

System.*out*.println(" [2]. Aviones");

System.*out*.println(" [3]. Incidencias ");

System.*out*.println(" [4]. Generar Misiones");

System.*out*.println(" [5]. Hisotirco");

System.*out*.println(" [6]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

public void historial() {

System.*out*.println(" Bienvenido Admin ");

System.*out*.println(" [1]. Pilotos Agregados");

System.*out*.println(" [2]. Piolotos Eliminados ");

System.*out*.println(" [3]. Aviones Agregados ");

System.*out*.println(" [4]. Aviones Eliminados");

System.*out*.println(" [5]. Misiones Realizadas");

System.*out*.println(" [6]. Misiones Fallidas");

System.*out*.println(" [7]. Volver");

System.*out*.println(" [8]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

public void incidencia() {

System.*out*.println(" Bienvenido Admin ");

System.*out*.println(" [1]. Estado de Pilotos ");

System.*out*.println(" [2]. Estado de Aviones ");

System.*out*.println(" [3]. Incidencias ");

System.*out*.println(" [5]. Volver ");

System.*out*.println(" [6]. Salir Programa ");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

public void avion() {

System.*out*.println(" Bienvenido Admin ");

System.*out*.println(" [1]. Alta de Avión ");

System.*out*.println(" [2]. Baja de Avión");

System.*out*.println(" [3]. Incidencias ");

System.*out*.println(" [4]. Historial de un Avion");

System.*out*.println(" [5]. Volver");

System.*out*.println(" [6]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

public void pilo() {

System.*out*.println(" Bienvenido Admin ");

System.*out*.println(" [1]. Alta de Piloto ");

System.*out*.println(" [2]. Baja de Piloto");

System.*out*.println(" [3]. Incidencias ");

System.*out*.println(" [4]. Generar Misiones");

System.*out*.println(" [5]. Volver");

System.*out*.println(" [6]. Salir Programa");

System.*out*.println("");

System.*out*.println(" Opción : ");

}

//---------------------- Realmente Los que Utilizo ------------------------------------

public void menuP() {

System.*out*.println("\n ====== Menú Principal ======");

System.*out*.println(" [1]. Gestión de Pilotos");

System.*out*.println(" [2]. Gestión de Aviones");

System.*out*.println(" [3]. Gestión de Incidencias");

System.*out*.println(" [4]. Gestión de Rutas");

System.*out*.println(" [5]. Salir");

System.*out*.print(" Seleccione una opción: ");

}

public void gestionR() {

System.*out*.println("\n ====== Menú de Gestión de Rutas ====== ");

System.*out*.println(" [1]. Crear Misión ");

System.*out*.println(" [2]. Mostrar Paises Disponibles ");

System.*out*.println(" [3]. Ver Rutas ");

System.*out*.println(" [4]. Volver al Menú Principal ");

System.*out*.print(" Seleccione una opción: ");

}

public void menuGestion(String entidad) {

System.*out*.println("\n ====== Menú de Gestión de " + entidad + " ====== ");

System.*out*.println(" [1]. Alta " + entidad);

System.*out*.println(" [2]. Baja " + entidad);

System.*out*.println(" [3]. Modificar " + entidad);

System.*out*.println(" [4]. Ver " + entidad + "s");

System.*out*.println(" [5]. Volver al Menú Principal");

System.*out*.print(" Seleccione una opción: ");

}

public void menuInciden() {

System.*out*.println("\n ====== Menú de Gestión de Incidencias ====== ");

System.*out*.println(" [1]. Incidencia de Piloto");

System.*out*.println(" [2]. Incidencia de Avión");

System.*out*.println(" [3]. Ver Incidencias");

System.*out*.println(" [4]. Volver al Menú Principal");

System.*out*.print(" Seleccione una opción: ");

}

public void volver() {

System.*out*.println("\n ====== Menú de Gestión de Incidencias ====== ");

System.*out*.println(" [1]. Incidencia de Piloto");

System.*out*.println(" [2]. Incidencia de Avión");

System.*out*.println(" [3]. Ver Incidencias");

System.*out*.println(" [4]. Volver al Menú Principal");

System.*out*.print(" Seleccione una opción: ");

}

public void salir() {

System.*out*.println("");

System.*out*.println("\r\n"

+ " █▀▀ █ █▄ █   █▀▄ █▀▀ █    █▀█ █▀█ █▀█ █▀▀ █▀█ ▄▀█ █▀▄▀█ █▀▄\r\n"

+ " █▀  █ █ ▀█   █▄▀ ██▄ █▄▄   █▀▀ █▀▄ █▄▀ █▄█ █▀▄ █▀█ █ ▀ █ █▀█");

System.*out*.println("");

System.exit(1);

}

public void cabecera() {

System.*out*.println("");

System.*out*.println("===============================================================================");

System.*out*.println(" ░█▀▄░█▀█░█▄█░░░█▀▀░█▀▀░█▀▀░▀█▀░█▀█░█▀▄░░░█▀▀░█░█░█▀▀░▀█▀░█▀▀░█▄█░░░▀█▀░▀█▀\r\n"

+ " ░█░█░█▀█░█░█░░░▀▀█░█▀▀░█░░░░█░░█░█░█▀▄░░░▀▀█░░█░░▀▀█░░█░░█▀▀░█░█░░░░█░░░█░\r\n"

+ " ░▀▀░░▀░▀░▀░▀░░░▀▀▀░▀▀▀░▀▀▀░░▀░░▀▀▀░▀░▀░░░▀▀▀░░▀░░▀▀▀░░▀░░▀▀▀░▀░▀░░░▀▀▀░▀▀▀ ");

System.*out*.println("===============================================================================");

System.*out*.println("");

System.*out*.println("");

System.*out*.println("\r\n"

+ " █ █▄ █ █ █▀▀ █ █▀█   █▀▄ █▀▀ █    █▀█ █▀█ █▀█ █▀▀ █▀█ █▀█ █▀▄▀█ █▀▄\r\n"

+ " █ █ ▀█ █ █▄▄ █ █▄▀   █▄▀ ██▄ █▄▄   █▀▀ █▀▄ █▄█ █▄█ █▀▄ █▀█ █ ▀ █ █▀█");

System.*out*.println("");

}

public void error() {

System.*out*.println("");

System.*out*.println(" --~{ Opción no válida. Intente de nuevo.}~-- ");

System.*out*.println("");

}

}

Texto

Descripción generada automáticamente

### Estructura de Interface

Para la gestión de este menú generaremos, tanto el controlador (donde se encuentran los sysos) como una interface donde colocaremos toda la funcionalidad del código y todos los atributos que crearemos para generar la funcionalidad del código, la estructura seguida será la siguiente.

package com.dam.airbase.interfaces;

import java.util.Scanner;

import com.dam.airbase.entidades.Registro;

public interface CRUDInterface {

public void alta(Registro r);

public void baja(Registro r);

public void modificar(Registro r);

public void ver(Registro r);

public void inicializar(Registro r);

public boolean ifExist(Registro r, String codigo, boolean ok);

public String elegirCarga(boolean carga,Scanner sc,String[][]t );

public void mostrarCarga(String carga);

public String[][] tiposCargamento();

public int cargaPiloto();

public boolean tipoCarga(Scanner sc);

public int sumarHoras(int horaRuta, int hora) ;

public void mostrarHorasActualizado(String nombre,int horas);

}

Siguiendo con la carpeta donde nos encontramos que es interfaz también hemos realizado la clase que es la de Constantes, donde tenemos todas las arrays y matrices que usamos en nuestro programa.

package com.dam.airbase.interfaces;

import java.util.Scanner;

public class Constantes {

public static final Scanner *sc* = new Scanner(System.*in*);

public final static String[] *NOMBRES\_PILOTO* = {"Faly", "Constantino", "Omar","Mauricio",

"Jose Manuel","Adrian","Luis","Pablo","Fausto","Mario","Jose Carlos"};

public final static String[][] *CONTINENTES\_PAISES*= {{"Africa", "Sudafrica","Nigeria", "Kenia",

"Marruecos","Ghana","Senegal", "Etiopia"},{"America", "Estados Unidos","Canada",

"Mexico", "Argentina","Brasil","Colombia", "Peru"},{"Antartida", "Orcadas",

"Carlini", "Marambio","Petrel","Bellingshausen","Palmer ", "Halley "},{"Asia",

"Japon","China", "India","Indionesia","Corea del Sur","Singapur", "Iran"},

{"Europa", "Italia","Alemania", "Francia","España","Inglaterra","Polonia",

"Ucrania"},{"Oceania", "Australia","Fiyi", "Samoa",

"Islas Salomon","Nauru","Tonga", "Palaos"}};

//CARGA

//COMBATE

//TRANSPORTE

public final static String [] *NOMBRE\_AVIONES* = {"Boeing 747","Boeing 737","Boeing 777",

"F-22 Rapto","Eurofighter Typhoon","Dassault Rafale",

"CN-235","C-295","C-390"};

}

public final static String[][] *CONTINENTES\_PAISES*= {{"Africa", "Sudafrica","Nigeria", "Kenia",

"Marruecos","Ghana","Senegal", "Etiopia"},{"America", "Estados Unidos","Canada",

"Mexico", "Argentina","Brasil","Colombia", "Peru"},{"Antartida", "Orcadas",

"Carlini", "Marambio","Petrel","Bellingshausen","Palmer ", "Halley "},{"Asia",

"Japon","China", "India","Indionesia","Corea del Sur","Singapur", "Iran"},

{"Europa", "Italia","Alemania", "Francia","España","Inglaterra","Polonia",

"Ucrania"},{"Oceania", "Australia","Fiyi", "Samoa",

"Islas Salomon","Nauru","Tonga", "Palaos"}};

//CARGA

//COMBATE

//TRANSPORTE

public final static String [] *NOMBRE\_AVIONES* = {"Boeing 747","Boeing 737","Boeing 777",

"F-22 Rapto","Eurofighter Typhoon","Dassault Rafale",

"CN-235","C-295","C-390"};

}

Y para terminar con esta carpeta tenemos la clase de CRUDInterface

package com.dam.airbase.interfaces;

import java.util.Scanner;

import com.dam.airbase.entidades.Registro;

public interface CRUDInterface {

public void alta(Registro r);

public void baja(Registro r);

public void modificar(Registro r);

public void ver(Registro r);

public void inicializar(Registro r);

public boolean ifExist(Registro r, String codigo, boolean ok);

public String elegirCarga(boolean carga,Scanner sc,String[][]t );

public void mostrarCarga(String carga);

public String[][] tiposCargamento();

public int cargaPiloto();

public boolean tipoCarga(Scanner sc);

public int sumarHoras(int horaRuta, int hora) ;

public void mostrarHorasActualizado(String nombre,int horas);

}

El propósito de esta clase es:

* Definir un contrato para las operaciones de CRUD (Create, Read, Update, Delete) sobre objetos del tipo Registro.
* Establece los métodos que deben implementar las clases que la implementen para realizar estas operaciones.

En esta clase solo definimos los métodos que estamos usando, no necesitamos implementar nada.

Estructura de Entidades

En entidades tenemos todas las definiciones de los constructores de las clases que vamos a usar.

Su estructura es

package com.dam.airbase.entidades;

public class Avion extends Registro {

//CONSTRUCTOR de la clase Avion

public Avion (String codigo, String nombre, int horasVuelo, boolean operativo) {

super( codigo, nombre, horasVuelo, operativo);

}

public Avion () {

super();

}

//GETTERS Y SETTERS

/\*\*

\* **@return** the codigo

\*/

public String getCodigo() {

return codigo;

}

/\*\*

\* **@param** codigo the codigo to set

\*/

public void setCodigo(String codigo) {

this.codigo = codigo;

}

/\*\*

\* **@return** the nombre

\*/

public String getNombre() {

return nombre;

}

/\*\*

\* **@param** nombre the nombre to set

\*/

public void setNombre(String nombre) {

this.nombre = nombre;

}

/\*\*

\* **@return** the horasVuelo

\*/

public int getHorasVuelo() {

return horasVuelo;

}

/\*\*

\* **@param** horasVuelo the horasVuelo to set

\*/

public void setHorasVuelo(int horasVuelo) {

this.horasVuelo = horasVuelo;

}

/\*\*

\*

\* **@param** operativo

\*/

public void setOperativo(boolean operativo) {

this.operativo = operativo;

}

/\*\*

\*

\* **@return**

\*/

public boolean getOperativo() {

return operativo;

}

@Override

public String toString() {

return "Avion [codigo=" + codigo + ", nombre=" + nombre + ", horasVuelo=" + horasVuelo + ", operativo="

+ operativo + "]";

}

}

Prácticamente tenemos lo mismo para todas usaremos esta de ejemplo.

### Estructura del main.

package com.dam.airbase.main;

import com.dam.airbase.controladores.Menus;

public class Main {

static int *opcion* = 0;

//Llamamos a los Servicios, y lo Instanciamos para poderlos inicializar

private static Menus *m* = new Menus();

public static void main(String[] args) {

*m*.menu(args);

}

}

### Estructura de Servicios

Donde se encuentra el musculo de nuestro programa y la verdadera funcionalidad de cada una de las clases, tenemos una por cada una de las funcionalidades del menú.

Para GestionAvionCRUD

Básicamente lo que realizamos en esta clase es saber si el avión este operativo o no, damos de Alta a un avión mediante los datos que vienen de REGISTRO.

Modificamos datos de este, y elegimos el avión que ira a una misión, que dependiendo del tipo de avión podrá ir a unas misiones u a otras.

package com.dam.airbase.servicios;

import java.util.ArrayList;

import java.util.Random;

import java.util.Scanner;

import com.dam.airbase.entidades.Avion;

import com.dam.airbase.entidades.Registro;

import com.dam.airbase.interfaces.CRUDInterface;

import com.dam.airbase.interfaces.Constantes;

public class GestionAvionCRUD implements CRUDInterface {

public ArrayList<Avion>aviones;

private Random random = new Random();

public void inicializar(Registro r) {

aviones = new ArrayList <Avion>();

aviones.add(new Avion("A001","F22",100,true));

aviones.add(new Avion("A002","F15",250,true));

aviones.add(new Avion("A003","Typhoon",700,true));

aviones.add(new Avion("A004","Rafle e",400,true));

}

public ArrayList<Avion> obtenerAviones(){

return aviones;

}

public void alta(Registro r) {

Avion a = (Avion)r;

boolean ok =true;

String codigo;

do {

System.*out*.println("Introduce el codigo del avion");

codigo = Constantes.*sc*.next();

ok = !ifExist(a,codigo,ok);

}while(ok==true);

System.*out*.println("Introduce el nombre del avion");

String nombre = Constantes.*sc*.next();

int horas = random.nextInt(201);

boolean operativo = true;

aviones.add(new Avion (codigo,nombre,horas,operativo)); //esto casca y no sé por que

}

public void baja(Registro r) {

boolean ok = true;

Avion p = (Avion)r;

String codigo;

do {

System.*out*.println("Introduce el codigo del avion que quieres dar de baja");

codigo = Constantes.*sc*.next();

ok = ifExist(p,codigo,ok);

for(int i = 0; i<aviones.size();i++) {

if(codigo.equals(aviones.get(i).getCodigo())) {

aviones.remove(i);

}

}

} while(ok==true);

}

public void modificar(Registro r) {

boolean ok = true;

Avion p = (Avion)r;

String codigo;

String mod="";

String dato="";

do {

System.*out*.println("Introduce el codigo del avion que quieres modificar");

codigo = Constantes.*sc*.next();

ok = ifExist(p,codigo,ok);

for(int i = 0; i<aviones.size();i++) {

if(codigo.equals(aviones.get(i).getCodigo())) {

System.*out*.println(aviones.get(i));

System.*out*.print("Que dato quieres modificar: ");

mod= Constantes.*sc*.next();

if(mod.charAt(0)=='C' || mod.charAt(0)=='c') {

System.*out*.println("Introduce el nuevo codigo: ");

dato = Constantes.*sc*.next();

aviones.get(i).setCodigo(dato);

}else if(mod.charAt(0)=='N' || mod.charAt(0)=='n') {

System.*out*.println("Introduce el nuevo nombre: ");

dato = Constantes.*sc*.next();

aviones.get(i).setNombre(dato);

}else {

System.*out*.println("Introduce las nuevas horas de vuelo: ");

int d = Constantes.*sc*.nextInt();

aviones.get(i).setHorasVuelo(d);

}

}

}

} while(ok==true);

}

public void ver(Registro r) {

for(int i = 0;i < aviones.size(); i++) {

System.*out*.println(aviones.get(i).getCodigo()+ ", " + aviones.get(i).getNombre() + ", " + aviones.get(i).getHorasVuelo() + " Horas de vuelo" + ", Operativo: " + aviones.get(i).getOperativo());

}

}

public boolean ifExist(Registro r,String codigo, boolean ok) {

for (int i = 0;i<aviones.size() && ok==true;i++) {

if(codigo.equals(aviones.get(i).getCodigo())) {

System.*out*.println("Ya hay un avion así("+aviones.get(i).getNombre()+")");

ok = false;

} else {

ok = true;

}

}

return ok;

}

public String elegirCarga(boolean carga,Scanner sc,String[][]t ) {

/\*En esta funcion entra como parametro un booleano, Scanner sc y una matriz t.

boolean carga:

TRUE-->Provisiones-->ARMAMENTO MUNICION EXPLOSIVOS

FALSE-->Suministro-->TROPAS VEHICULOS GASOLINA

String [][] t:

Se recorre la matriz para verificar que la carga es correcta, si no es así se pide de nuevo

Devuelve el tipo de carga que va a transportar

\*/

boolean verificar=true;

String texto;

do {

if (carga ==true) {

System.*out*.println("¿Que tipo de provision va a transportar?");

System.*out*.println("[Armamento, Munición, Explosivos]");

texto=sc.next();

String c=texto;

for (int i = 0; i<t.length;i++) {

for(int j =0;j<t[i].length;j++) {

if(c.equals("Armamento")||c.equals("Municion")||c.equals("Explosivos")) {

verificar=true;

}

else {

verificar = false;

}

}

}

if(verificar ==false) {

System.*out*.println("ERROR");

}

} else {

System.*out*.println("¿Que tipo de suministro va a transportar?");

System.*out*.println("[Tropas, Vehiculos, Gasolina]");

texto=sc.next();

String c=texto;

for (int i = 0; i<t.length;i++) {

for(int j =0;j<t[i].length;j++) {

if(c.equals("Tropas")||c.equals("Vehiculos")||c.equals("Gasolina")) {

verificar=true;

}else {

verificar = false;

}

}

}

if(verificar ==false) {

System.*out*.println("ERROR");

}

}

}while(verificar==false);

return texto;

}

public void mostrarCarga(String carga) {

//Funcion que devuelve el mensaje de cual es la carga que se va a transportar

System.*out*.println("La carga que se llevará en el avión es: "+carga );

}

public String[][] tiposCargamento() {

//Matriz en la que se guarda los tipos de cargamento que puede llevar un avion

String [][] t= new String [3][3];

t[0][0] ="Armamento";

t[0][1] ="Municion";

t[0][2] ="Explosivos";

t[1][0] ="Tropas";

t[1][1] ="Vehiculos";

t[1][2] ="Gasolina";

return t;

}

public boolean tipoCarga(Scanner sc) {

//Que tipo de vuelo es [Provisiones o Suministro]

boolean vuelo = true; //true significa que es de Provisiones

boolean verificar = false; //para verificar si se ha escrito correctamente

while(verificar == false) {

System.*out*.println("Que tipo de vuelo es: [Provisiones o Suministros]");

char texto = sc.next().charAt(0);

if (texto=='S'|| texto=='s') {

vuelo=false;

verificar = true;

} else if(texto=='P'|| texto=='p') {

vuelo=true;

verificar = true;

}

if (verificar == false) {

System.*out*.println("ERROR"

+ "\n----------Debe ser Provisiones o Suministro----------");

}

}

return vuelo;

}

public void validarAvion2() {

System.*out*.println("\n == TIPOS DE RUTA QUE PUEDEN HACER LOS AVIONES ==\n");

for(int i = 0;i<aviones.size();i++) {

validarAvion(Constantes.*sc*,aviones.get(i).getNombre(),aviones.get(i).getHorasVuelo());

System.*out*.println();

}

}

public int cargaPiloto() {

int var= 0;

boolean t =tipoCarga(Constantes.*sc*);

if(t==true) {

var = 1;

}else if(t==false) {

var = 2;

}else {

var = 3;

}

return var;

}

public void validarAvion(Scanner sc, String string, int horas) {

//Que tipo de mision puede ir el avion

if (horas<=100) {

System.*out*.println(" " + string+" está cualificado para rutas de transporte");

} else if (horas>100 && horas<=300) {

System.*out*.println(" " + string+" está cualificado para transporte y bombarderos");

}

else {

System.*out*.println(" " + string+" puede ser usado para cualquier tipo de misión");

}

}

public int sumarHoras(int horaRuta, int hora) {

//Funcion que suma la duracion de la ruta y las horas de vuelo.

hora = horaRuta+hora;

return hora;

}

public void mostrarHorasActualizado(String nombre,int horas) {

//Se muestra las horas totales del piloto o avion

try{

int p=GestionRuta.*rutas*.get(0).getH();

horas=sumarHoras(aviones.get(aviones.size()-1).getHorasVuelo(),p);

System.*out*.println("ACTUALIZACION\n"

+ "\t"+nombre+" tiene "+ horas+" horas de vuelo ");

}catch(IndexOutOfBoundsException e){

}System.*out*.println("ACTUALIZACION\n"

+ "\t"+nombre+" tiene "+ horas+" horas de vuelo ");

}

public void mostrarHoras() {

mostrarHorasActualizado(aviones.get(aviones.size()-1).getNombre(),aviones.get(aviones.size()-1).getHorasVuelo());

}

}

Para GestionIncidencias

Toda la gestion de las incidencias, tanto añadir como modificar una.

package com.dam.airbase.servicios;

import java.util.ArrayList;

import java.util.Scanner;

import com.dam.airbase.entidades.Avion;

import com.dam.airbase.entidades.Incidencias;

import com.dam.airbase.entidades.Piloto;

public class GestionIncidencias {

Scanner dato = new Scanner(System.*in*);

private ArrayList<Incidencias> incidencia;

public void inicializarIncidencias() {

incidencia = new ArrayList <Incidencias>();

}

public ArrayList<Incidencias> obtenerIncidencias(){

return incidencia;

}

public void incidenciaPiloto(ArrayList<Piloto> pilotos) {

boolean ok = false;

int seleccion = 0;

String numIncidencia = "";

String codPil = "";

String descrip = "";

String op = "";

System.*out*.println("Introduce el numero de incidencia");

numIncidencia = dato.next();

while(ok == false) {

System.*out*.println("Introduce el código del piloto");

codPil = dato.next();

for(int i = 0;i<pilotos.size() && ok ==false;i++) {

if(codPil.equals(pilotos.get(i).getCodigo())) {

seleccion = i;

ok = true;

} else {ok = false;}

}

if(ok == false) {

System.*out*.println("Ningun piloto encontrado, vuelva a introducir");

}

}

System.*out*.println("Piloto seleccionado: " + pilotos.get(seleccion).getNombre());

System.*out*.println("Introduczca una descripción de la incidencia");

descrip = dato.nextLine();

descrip = dato.nextLine();

while (ok == true) {

System.*out*.println("¿Apto para el servicio?");

op = dato.next();

if(op.charAt(0) =='n'||op.charAt(0)=='N') {

pilotos.get(seleccion).setOperativo(false);

ok = false;

} else if(op.charAt(0) =='s'||op.charAt(0)=='S') {

pilotos.get(seleccion).setOperativo(true);

ok = false;

} else {

System.*out*.println("Error, respuesta no valida");

ok = true;

}

}

incidencia.add(new Incidencias(numIncidencia,codPil,descrip,op));

}

public void incidenciaAvion(ArrayList<Avion> aviones) {

boolean ok = false;

int seleccion = 0;

String numIncidencia = "";

String codAv = "";

String descrip ="";

String op = "";

System.*out*.println("Introduce el numero de incidencia");

numIncidencia = dato.next();

while(ok==false) {

System.*out*.println("Introduce el código del avión");

codAv = dato.next();

for(int i = 0;i<aviones.size() && ok == false;i++) {

if(codAv.equals(aviones.get(i).getCodigo())) {

seleccion = i;

ok = true;

} else {ok = false;}

}

if(ok == false) {

System.*out*.println("Ningún avión encontrado");

}

}

System.*out*.println("Avión seleccionado: " +aviones.get(seleccion).getNombre());

System.*out*.println("Introduzca una descripción de la incidencia");

descrip = dato.nextLine();

descrip = dato.nextLine();

while (ok == true) {

System.*out*.println("¿Apto para el servicio?");

op = dato.next();

if(op.charAt(0) =='n'||op.charAt(0)=='N') {

aviones.get(seleccion).setOperativo(false);

ok = false;

} else if(op.charAt(0) =='s'||op.charAt(0)=='S') {

aviones.get(seleccion).setOperativo(true);

ok = false;

} else {

System.*out*.println("Error, respuesta no valida");

ok = true;

}

}

incidencia.add(new Incidencias(numIncidencia,codAv,descrip,op));

}

public void verIncidencias() {

for(int i = 0;i<incidencia.size();i++) {

System.*out*.println("Numero de incidencia: "+ incidencia.get(i).getNumeroIncidencia() + " código ref: " + incidencia.get(i).getCodigoRef() + " Incidencia: " + incidencia.get(i).getDescripccion() + " operativo: "+ incidencia.get(i).getOperativo());

}

}

}

GestionPilotoCRUD

Al igual que para avión pero lo realizamos a los Pilotos.

package com.dam.airbase.servicios;

import java.util.ArrayList;

import java.util.Random;

import java.util.Scanner;

import com.dam.airbase.entidades.Piloto;

import com.dam.airbase.entidades.Registro;

import com.dam.airbase.interfaces.CRUDInterface;

import com.dam.airbase.interfaces.Constantes;

public class GestionPilotoCRUD implements CRUDInterface {

public ArrayList <Piloto> pilotos;

private Random random = new Random();

public void inicializar(Registro r) {

pilotos = new ArrayList <Piloto>();

pilotos.add(new Piloto("P001","Faly",200,true,true));

pilotos.add(new Piloto("P002","Constantino",200,true,false));

pilotos.add(new Piloto("P003","Omar",200,true, true));

pilotos.add(new Piloto("P004","Mauricio",200,true, false));

}

public ArrayList<Piloto> obtenerPilotos(){

return pilotos;

}

public void alta(Registro r) {

if (pilotos == null) {

// Inicia la lista de pilotos en caso de que esté null

pilotos = new ArrayList<Piloto>();

}

boolean ok = true;

Piloto p = (Piloto)r;

String codigo;

do {

System.*out*.println("Introduce el codigo del piloto");

codigo = Constantes.*sc*.next();

ok = !ifExist(p,codigo,ok); //Devuelve False y se cambia con '!' a true

} while(ok==true);

System.*out*.println("Introduce el nombre del piloto");

String nombre = Constantes.*sc*.next();

int horas = hVuelos();

boolean operativo = true;

pilotos.add(new Piloto (codigo, nombre, horas, operativo,false));

}

public void baja(Registro r) {

boolean ok = true;

Piloto p = (Piloto)r;

String codigo;

do {

System.*out*.println("Introduce el codigo del piloto que quieres dar de baja");

codigo = Constantes.*sc*.next();

ok = ifExist(p,codigo,ok);

for(int i = 0; i<pilotos.size();i++) {

if(codigo.equals(pilotos.get(i).getCodigo())) {

pilotos.remove(i);

}

}

} while(ok==true);

}

public void modificar(Registro r) {

boolean ok = true;

Piloto p = (Piloto)r;

String codigo;

String mod="";

String dato="";

do {

System.*out*.println("Introduce el codigo del piloto que quieres modificar");

codigo = Constantes.*sc*.next();

ok = ifExist(p,codigo,ok);

for(int i = 0; i<pilotos.size();i++) {

if(codigo.equals(pilotos.get(i).getCodigo())) {

System.*out*.println(pilotos.get(i));

System.*out*.print("Que dato quieres modificar: ");

mod= Constantes.*sc*.next();

if(mod.charAt(0)=='C' || mod.charAt(0)=='c') {

System.*out*.println("Introduce el nuevo codigo: ");

dato = Constantes.*sc*.next();

pilotos.get(i).setCodigo(dato);

}else if(mod.charAt(0)=='N' || mod.charAt(0)=='n') {

System.*out*.println("Introduce el nuevo nombre: ");

dato = Constantes.*sc*.next();

pilotos.get(i).setNombre(dato);

}else {

System.*out*.println("Introduce las nuevas horas de vuelo: ");

int d = Constantes.*sc*.nextInt();

pilotos.get(i).setHorasVuelo(d);

}

}

}

} while(ok==true);

}

public int hVuelos() {

int h=0;

h=random.nextInt(1000);

return h;

}

public void ver(Registro r) {

for(int i = 0;i < pilotos.size(); i++) {

System.*out*.println(pilotos.get(i).getCodigo()+ ", " + pilotos.get(i).getNombre()+ ", " + pilotos.get(i).getHorasVuelo()+" horas de vuelo" + ", Operativo: " + pilotos.get(i).getOperativo()+" Administrador: "+pilotos.get(i).getAdmin());

}

}

public boolean ifExist(Registro r, String codigo,boolean ok) {

for (int i = 0;i<pilotos.size() && ok==true;i++) {

if(codigo.equals(pilotos.get(i).getCodigo())) {

System.*out*.println("Ya hay un piloto así("+pilotos.get(i).getNombre()+")");

ok = false;

} else {

ok = true;

}

}

return ok;

}

public String elegirCarga(boolean carga,Scanner sc,String[][]t ) {

/\*En esta funcion entra como parametro un booleano, Scanner sc y una matriz t.

boolean carga:

TRUE-->Provisiones-->ARMAMENTO MUNICION EXPLOSIVOS

FALSE-->Suministro-->TROPAS VEHICULOS GASOLINA

String [][] t:

Se recorre la matriz para verificar que la carga es correcta, si no es así se pide de nuevo

Devuelve el tipo de carga que va a transportar

\*/

boolean verificar=true;

String texto;

do {

if (carga ==true) {

System.*out*.println("¿Que tipo de provision va a transportar?");

System.*out*.println("[Armamento, Munición, Explosivos]");

texto=sc.next();

String c=texto;

for (int i = 0; i<t.length;i++) {

for(int j =0;j<t[i].length;j++) {

if(c.equals("Armamento")||c.equals("Municion")||c.equals("Explosivos")) {

verificar=true;

}

else {

verificar = false;

}

}

}

if(verificar ==false) {

System.*out*.println("ERROR");

}

} else {

System.*out*.println("¿Que tipo de suministro va a transportar?");

System.*out*.println("[Tropas, Vehiculos, Gasolina]");

texto=sc.next();

String c=texto;

for (int i = 0; i<t.length;i++) {

for(int j =0;j<t[i].length;j++) {

if(c.equals("Tropas")||c.equals("Vehiculos")||c.equals("Gasolina")) {

verificar=true;

}else {

verificar = false;

}

}

}

if(verificar ==false) {

System.*out*.println("ERROR");

}

}

}while(verificar==false);

return texto;

}

public int cargaPiloto() {

int var= 0;

boolean t =tipoCarga(Constantes.*sc*);

if(t==true) {

var = 1;

}else if(t==false) {

var = 2;

}else {

var = 3;

}

return var;

}

public void mostrarCarga(String carga) {

//Funcion que devuelve el mensaje de cual es la carga que se va a transportar

System.*out*.println("La carga que se llevará en el avión es: "+carga );

}

public String[][] tiposCargamento() {

//Matriz en la que se guarda los tipos de cargamento que puede llevar un avion

String [][] t= new String [3][3];

t[0][0] ="Armamento";

t[0][1] ="Municion";

t[0][2] ="Explosivos";

t[1][0] ="Tropas";

t[1][1] ="Vehiculos";

t[1][2] ="Gasolina";

return t;

}

public boolean tipoCarga(Scanner sc) {

//Que tipo de vuelo es [Provisiones o Suministro]

boolean vuelo = true; //true significa que es de Provisiones

boolean verificar = false; //para verificar si se ha escrito correctamente

while(verificar == false) {

System.*out*.println("Que tipo de vuelo es: [Provisiones o Suministros]");

char texto = sc.next().charAt(0);

if (texto=='S'|| texto=='s') {

vuelo=false;

verificar = true;

} else if(texto=='P'|| texto=='p') {

vuelo=true;

verificar = true;

}

if (verificar == false) {

System.*out*.println("ERROR"

+ "\n----------Debe ser Provisiones o Suministro----------");

}

}

return vuelo;

}

public void validarPiloto2() {

System.*out*.println("\n == TIPOS DE RUTA QUE PUEDEN HACER LOS PILOTOS == \n");

for(int i = 0; i<pilotos.size();i++) {

validarPiloto(Constantes.*sc*,pilotos.get(i).getNombre(),pilotos.get(i).getHorasVuelo());

System.*out*.println();

}

}

public void validarPiloto(Scanner sc,String string, int exp) {

//En esta funcion vamos a ver que piloto esta cualificado para llevar el avion dependiendo del tipoVuelo

if (exp<100) {

//Si el piloto tiene menos de 100 horas podra pilotar solo aviones comerciales

System.*out*.println(" " + string +" solo puede pilotar aviones de transporte");

} else if(exp>=100 && exp<=500) {

System.*out*.println(" " + string +" puede pilotar aviones de transporte o bombarderos, "

+ "pero solo para misiones de suministro");

} else {

System.*out*.println(" " + string +" puede manejar cualquier tipo de avión sin depender de la misión"

+ ",es decir, aviones de transporte, bombarderos y aviones de combate");

}

}

public int sumarHoras(int horaRuta, int hora) {

//Funcion que suma la duracion de la ruta y las horas de vuelo.

hora = horaRuta+hora;

return hora;

}

public void mostrarHorasActualizado(String nombre,int horas) {

//Se muestra las horas totales del piloto o avion

horas=sumarHoras(pilotos.get(pilotos.size()-1).getHorasVuelo(),0);

System.*out*.println("ACTUALIZACION\n"

+ "\t"+nombre+" tiene "+ horas+" horas de vuelo ");

}

public void mostrarHoras() {

mostrarHorasActualizado(pilotos.get(pilotos.size()-1).getNombre(),pilotos.get(pilotos.size()-1).getHorasVuelo());

}

}

En GestionRegistro

Imprimimos un submenú para saber que desea realizar si dar de alta a un piloto o a un avión o modificar los datos de estos.

package com.dam.airbase.servicios;

import com.dam.airbase.interfaces.Constantes;

public class GestionRegistro {

public int mostrarOpcionesPiloto() {

System.*out*.println("1. Dar de Alta Piloto\n"

+ "2. Dar de Baja Piloto\n"

+ "3. Modificar Datos Piloto\n"

+ "4. Ver Pilotos");

int opcion = Constantes.*sc*.nextInt();

return opcion;

}

public int mostrarOpcionesAviones() {

System.*out*.println("1. Dar de Alta Avion\n"

+ "2. Dar de Baja Avion\n"

+ "3. Modificar Datos Avion\n"

+ "4. Ver Aviones");

int opcion = Constantes.*sc*.nextInt();

return opcion;

}

}

GestionRuta

Tenemos básicamente las rutas que realizaran nuestros pilotos con sus aviones añadiendo las horas de vuelo de la ruta al piloto y al avión.

package com.dam.airbase.servicios;

import java.util.ArrayList;

import java.util.Random;

import com.dam.airbase.entidades.Ruta;

import com.dam.airbase.interfaces.Constantes;

public class GestionRuta {

private String paises= "Africa: Sudafrica,Nigeria,Kenia,Marruecos,Ghana,Senegal,Etiopia\n"

+ "America:Estados Unidos,Canada,Mexico,Argentina,Brasil,Colombia,Peru\n"

+ "Antartida: Orcadas,Carlini,Marambio,Petrel,Bellingshausen,Palmer,Halley\n"

+ "Asia: Japon,China,India,Indionesia,Corea del Sur,Singapur,Iran\n"

+ "Europa: Italia,Alemania,Francia,España,Inglaterra,Polonia,Ucrania\n"

+ "Oceania: Australia,Fiyi,Samoa,Islas Salomon,Nauru,Tonga,Palaos";

//Atributos

public static ArrayList <Ruta> *rutas*;

//constructores

public GestionRuta() {

inicializarRutas();

}

//metodos

public void inicializarRutas() {

*rutas* = new ArrayList <Ruta>();

}

public ArrayList<Ruta> obtenerRutas(){

return *rutas*;

}

public void altaRuta( Ruta añadir, String paisOrigen, String paisDestino) {

añadir = new Ruta(paisOrigen,paisDestino);

*rutas*.add(añadir);

}

public void mostrarTodasRutas() {

System.*out*.println("\n"+paises+"\n");

}

public static int horasVuelo(String a, String b ) {

/\*Esta funcion va a calcular aproximadamente la duracion de la ruta dependiendo del continente

origen y el continente del destino.\*/

String origen = a;

String destino = b;

int horas=0;

switch(origen) {

case "Europa":

//Europa--->Africa = [6,10]

//Europa--->Asia = [4,12]

//Europa--->Oceania = [10,14]

//Europa--->America = [8,13]

//Europa--->Antartida = [5,9]

switch(destino) {

case "Africa", "África":

horas=generarRandom(6,10);

break;

case "Asia":

horas=generarRandom(4,12);

break;

case "Oceania","Oceanía":

horas=generarRandom(10,14);

break;

case "America","América":

horas=generarRandom(8,13);

break;

case "Antartida","Antártida":

horas=generarRandom(5,9);

break;

default:

horas = 4;

}

case "Africa", "África":

//Africa--->Europa = [6,10]

//Africa--->Asia = [3,11]

//Africa--->Oceania = [7,12]

//Africa--->America = [6,13]

//Africa--->Antartida = [8,12]

switch(destino) {

case "Europa":

horas=generarRandom(6,10);

break;

case "Asia":

horas=generarRandom(3,11);

break;

case "Oceania","Oceanía":

horas=generarRandom(7,12);

break;

case "America","América":

horas=generarRandom(6,13);

break;

case "Antartida","Antártida":

horas=generarRandom(8,12);

break;

default:

horas = 4;

}

case "Asia":

//Asia--->Africa = [3,11]

//Asia--->Europa = [4,12]

//Asia--->Oceania = [2,8]

//Asia--->America = [3,9]

//Asia--->Antartida = [4,8]

switch(destino) {

case "Africa", "África":

horas=generarRandom(3,11);

break;

case "Europa":

horas=generarRandom(4,12);

break;

case "Oceania":

horas=generarRandom(2,8);

break;

case "America","América":

horas=generarRandom(3,9);

break;

case "Antartida","Antártida":

horas=generarRandom(4,8);

break;

default:

horas = 4;

}

case "Oceania", "Oceanía":

//Oceania--->Africa = [7,12]

//Oceania--->Asia = [2,8]

//Oceania--->Europa = [10,14]

//Oceania--->America = [4,10]

//Oceania--->Antartida = [6,12]

switch(destino) {

case "Africa", "África":

horas=generarRandom(7,12);

break;

case "Asia":

horas=generarRandom(2,8);

break;

case "Europa":

horas=generarRandom(10,14);

break;

case "America","América":

horas=generarRandom(4,10);

break;

case "Antartida","Antártida":

horas=generarRandom(6,12);

break;

default:

horas = 4;

}

case"America","América":

//America--->Africa = [6,13]

//America--->Asia = [3,9]

//America--->Oceania = [4,10]

//America--->Europa = [8,13]

//America--->Antartida = [2,12]

switch(destino) {

case "Africa", "África":

horas=generarRandom(6,13);

break;

case "Asia":

horas=generarRandom(3,9);

break;

case "Oceania","Oceanía":

horas=generarRandom(4,10);

break;

case "Europa":

horas=generarRandom(8,13);

break;

case "Antartida","Antártida":

horas=generarRandom(2,12);

break;

default:

horas =4;

}

case "Antartida","Antártida":

//Antartida--->Africa = [8,12]

//Antartida--->Asia = [4,8]

//Antartida--->Oceania = [6,12]

//Antartida--->America = [2,12]

//Antartida--->Europa = [5,9]

switch(destino) {

case "Africa", "África":

horas=generarRandom(8,12);

break;

case "Asia":

horas=generarRandom(4,8);

break;

case "Oceania","Oceanía":

horas=generarRandom(6,12);

break;

case "America","América":

horas=generarRandom(2,12);

break;

case "Europa":

horas=generarRandom(5,9);

break;

default:

horas = 4;

}

}

return horas;

}

public static int generarRandom(int min, int max) {

//Funcion que genera numeros aleatorios a partir de un maximo y un minimo

Random r = new Random();

int aleatorio = r.nextInt(max-min+1)+min;

return aleatorio;

}

public boolean validarRuta(String continente, String pais) {

boolean correct = false;

for(int i = 0; i<6;i++) {

for(int j=0;j<8;j++){

if(continente.equals(Constantes.*CONTINENTES\_PAISES*[i][0])) {

if(pais.equals(Constantes.*CONTINENTES\_PAISES*[i][j])) {

correct=true;

}

}

}

}

return correct;

}

public int altaRuta(ArrayList<Ruta> rutas) {

boolean ok=false;

String continenteOrigen="";

String paisOrigen="";

String continenteDestino="";

String paisDestino="";

while(ok==false) {

System.*out*.println(" Introduce el continente de origen");

continenteOrigen = Constantes.*sc*.next();

System.*out*.println(" Introduce el pais de origen");

paisOrigen = Constantes.*sc*.next();

if(validarRuta(continenteOrigen,paisOrigen)==false) {

System.*out*.println(" Error. Introducelos de nuevo");

System.*out*.println(paises+"\n");

ok=false;

}else if(validarRuta(continenteOrigen,paisOrigen)==true){

ok=true;

}

}

ok=false;

while(ok==false) {

System.*out*.println(" Introduce el continente de destino");

continenteDestino = Constantes.*sc*.next();

System.*out*.println(" Introduce el pais de destino");

paisDestino = Constantes.*sc*.next();

if(validarRuta(continenteDestino,paisDestino)==false) {

System.*out*.println(" Error. Introducelos de nuevo");

System.*out*.println(paises+"\n");

ok=false;

}else {

ok=true;

}

}

int horas= horasVuelo(continenteOrigen,continenteDestino);

rutas.add(new Ruta (paisOrigen, paisDestino));

return horas;

}

public void mostrarRuta(ArrayList<Ruta> rutas) {

int h = altaRuta(rutas);

System.*out*.println("La duracion de la ruta es: "+ h+" horas\n");

}

public void verRutas(ArrayList<Ruta> rutas) {

for(int i = 0;i < rutas.size(); i++) {

System.*out*.println(rutas.get(i).getPaisOrigen()+ " - " + rutas.get(i).getPaisDestino());

}

}

}

# Resumen

La programación orientada a objetos (POO) nos permite estructurar mejor el código y simplificar el código aligerando mucho la carga de proyecto, permite una rápida legibilidad del código al tener esta metodología.

# 3ºTrimestre

Propuesta de mejora para el tercer trimestre, y dado que veremos ficheros, esperamos implementar nuestro programa a una primera versión mucho más funcional, pudiéndolo conectar a una base de datos.

Aparte esperamos implementar un Login mas funcional permitiendo a reconocer que usuarios para acceder a uno u otro menú.

# Nota final

A la realización y entrega de este documento, las personas de este grupo que han realizado la mayor parte del código son:

* Para la mayor parte del código tanto en funcionalidad como en planteamiento a sido de Juan Mauricio González Arenas y Constantino Lapaz Giménez
* Para el Menu y su funcionalidad y limpieza del Main Rafael Antonio Vázquez Flores con la ayuda de la explicación de código de Mauricio y Constantino
* Omar genero las clases de Avion junto con su GestionAvión.