# Práctica 2 ED | 2º Ingeniería Informática David Rodriguez Aparicio

En esta práctica hemos creado un pequeño programa de Agenda con eventos.

Veremos explicaciones de algunos métodos y sus explicaciones, códigos y algunas imagenes del programa.

Además esta práctica ha sido documentada con Doxygen, creando con Doxyfile una web con toda la estructura de clase.

Para ejecutar el programa, primero hemos hecho "make" en la terminal, desde Practica 2, y a continuación hemos ejecutado agenda.exe, a partir de ahí se nos crea un menú con las distintas opciones del programa, y se carga la agenda de "datos/agenda.txt", que es la que usaremos.

Para esta práctica se podrían haber añadido muchas funcionalidades, pero con las que estan actualmente está bien para el desarrollo de la práctica.

En el código se han añadido los comentarios necesarios ( además de los necesarios para Doxygen ), hay partes de código muy claras que no necesitan y otras como algunos métodos que viene bien comentar algunas cosas de su funcionamiento.

### !!! Existe un pequeño fallo que no he conseguido solucionar con las tildes, por lo que lo mejor al crear eventos y guardar la agenda es no usar estas.

## EVENTO.CPP

```
#include "Evento.h"
#include <cassert>
#include <string>

using namespace std;

/* ------------------ CONSTRUCTOR ------------------ */

/* Provisional y revisar */
Evento::Evento(){
    this->nombre = "";
    this->dia = 1;
    this->hora_inicio = 0.0;
    this->hora_fin = 1.0;
}

Evento::Evento(string nombre, int dia, double hora_inicio, double hora_fin){
    this->nombre = nombre;
    this->dia = dia;
    this->hora_inicio = hora_inicio;
    this->hora_fin = hora_fin;
```

```
    assert(dia >= 1 && dia <= 7);
    assert(hora_inicio >= 0.0 && hora_inicio < 24.0);
    assert(hora_fin > hora_inicio && hora_fin <= 24.0);
}


/* ------------------ METODOS ------------------ */

/* Se podría añadir como static y ser un método de clase ?¿ */
bool Evento::eventoSolapado(Evento &otroEvento){
    /* No se si se tienen que cambiar los getters, en principio no, son publicos y
los atributos privados. */

    if(this->dia == otroEvento.getDia() ){

        if(this->hora_fin < otroEvento.getHoraInicio() || otroEvento.getHoraFin()
< this->hora_inicio ){

            return true;
            // return "CUIDADO: Hay Solapamiento de horario entre: "+ this->nombre
+ " y " + otroEvento.getNombre();
        }

        return false;
        // return "Se solapan los días pero no hay problema.";
    }

    return false;
    // return "No se solapand"; /* Temporal, se cambia */
}

string Evento::mostrarInfo(){
    string info = "Información de Evento \n\n";

    info += "Nombre: "+ this->nombre + "\n";
    info += "Dia: "+ to_string( this->dia ) + "\n";
    info += "Hora Inicio: "+ to_string( this->hora_inicio ) + "\n";
    info += "Hora Fin: "+ to_string( this->hora_fin ) + "\n\n";

    return info;

}

/* string static compararEventos(int cantidadEventos,Evento arrayEventos[]){
    for (int i = 0; i < cantidadEventos; i++){
        for (int j = 0; j < cantidadEventos; j++){

            if( arrayEventos[j].getNombre().compare(arrayEventos[j+1].getNombre())
> 0 ){

                Evento tmp = arrayEventos[j];
                arrayEventos[j] = arrayEventos[j+1];
                arrayEventos[j+1] = tmp;
            }
```

```
        }

    }

} */


/* ----------------- SETTERS / GETTERS ----------------- */

string Evento::getNombre() {
    return this->nombre;
}

int Evento::getDia() {
    return this->dia;
}

double Evento::getHoraInicio() {
    return this->hora_inicio;
}

double Evento::getHoraFin() {
    return this->hora_fin;
}




void Evento::setNombre(string nuevoNombre) {
    this->nombre = nuevoNombre;
}

void Evento::setDia(int nuevoDia) {
    assert(nuevoDia >= 1 && nuevoDia <= 7);
    this->dia = nuevoDia;
}

void Evento::setHoraInicio(double nuevaHoraInicio) {
    assert(nuevaHoraInicio >= 0.0 && nuevaHoraInicio < hora_fin);
    this->hora_inicio = nuevaHoraInicio;
}

void Evento::setHoraFin(double nuevaHoraFin) {
    assert(nuevaHoraFin > hora_inicio && nuevaHoraFin <= 24.0);
    this->hora_fin = nuevaHoraFin;
}
```

## AGENDAEVENTOS.CPP

```cpp
#include <iostream>
#include <string>
// #include "../include/agendaEventos.h"  /* No hace falta, se incluye en el
makefile */
#include "agendaEventos.h"

#include <limits>  /* Libreria útil para comprobaciones. */

using namespace std;



void mostrarMenu() {
    cout << "\n========= AGENDA DE EVENTOS =========\n";
    cout << "1. Mostrar todos los eventos\n";
    cout << "2. Buscar evento por nombre\n";
    cout << "3. Buscar eventos por dia\n";
    cout << "4. Agregar nuevo evento\n";
    cout << "5. Eliminar evento por nombre\n";
    cout << "6. Detectar franjas libres en un dia\n";
    cout << "7. Mostrar resumen semanal\n";
    cout << "8. Guardar Agenda\n";
    cout << "0. Salir\n\n";
    cout << "Seleccione una opcion: ";
}

int main(){

    AgendaEventos agenda("datos/agenda.txt");

    int opcion = -1;

    while (opcion != 0){
        mostrarMenu();

        cin >> opcion;

        /* REVISAR SWITCH PARA COMPROBACIONES (cin sobretodo) */
        switch (opcion){
            case 1: {
                cout << agenda.mostrarTodosLosEventos() << endl;
                break;
            }

            case 2: {

                string nombre;

                cout << "Introduce el nombre del evento a buscar: ";

                cin.ignore(numeric_limits<streamsize>::max(), '\n');
                getline(cin, nombre);
```

```
            cout << agenda.buscarEventoPorNombre(nombre) << endl;
            break;
        }

        case 3: {
            int dia;

            cout << "Introduce el dia a buscar (1-7): ";
            cin >> dia;

            string resultado = agenda.buscarEventoPorDia(dia);
            cout << resultado << endl;
            break;
        }

        case 4: {
            string nombre;
            int dia;
            double horaInicio, horaFin;

            cout << "Nombre del evento: ";
            cin >> nombre;

            cout << "Dia de la semana (1-7): ";
            cin >> dia;

            cout << "Hora de inicio (formato decimal, ej. 9.5 = 9:30): ";
            cin >> horaInicio;

            cout << "Hora de fin (formato decimal, ej. 11.0): ";
            cin >> horaFin;

            cin.ignore(numeric_limits<streamsize>::max(), '\n');


            Evento nuevo(nombre, dia, horaInicio, horaFin);

            if (agenda.agregarEvento(nuevo)) {
                cout << "Evento agregado correctamente.\n";
            } else {
                cout << "No se ha podido agregar el evento: hay solapamiento
con otro existente.\n";
            }


            break;
        }

        case 5: {
            string nombre;

            cout << "Introduce el nombre del evento a eliminar: ";
            cin >> nombre;
```

```
                if ( agenda.eliminarEvento(nombre) ) {
                    cout << "Evento eliminado correctamente.\n";
                } else {
                    cout << "No se ha encontrado ningun evento con ese nombre.\n";
                }

                break;
            }

            case 6: {
                int dia;

                cout << "Introduce el dia (1-7) para ver las franjas libres: ";
                cin >> dia;

                cout << agenda.detectarFranjasLibresPorDia(dia) << endl;

                break;
            }

            case 7: {
                cout << agenda.resumenSemanal() << endl;
                break;
            }

            case 8: {
                cout << "Guardando Agenda en 'agenda.txt' ";

                if (agenda.guardarArchivo()) {
                    cout << "Archivo guardado correctamente.\n";
                } else {
                    cout << "Error: no se pudo guardar el archivo.\n";
                }

                break;
            }

            case 0:
                agenda.~AgendaEventos();
                cout << "\n Saliendo del programa. \n";
                break;

        default:
            cout << "\n Opción no válida. \n";
            break;
        }
    }



}
```

## MAIN.CPP

```
#include <iostream>
#include <string>
// #include "../include/agendaEventos.h"  /* No hace falta, se incluye en el
makefile */
#include "agendaEventos.h"

#include <limits>  /* Libreria útil para comprobaciones. */

using namespace std;



void mostrarMenu() {
    cout << "\n========= AGENDA DE EVENTOS =========\n";
    cout << "1. Mostrar todos los eventos\n";
    cout << "2. Buscar evento por nombre\n";
    cout << "3. Buscar eventos por dia\n";
    cout << "4. Agregar nuevo evento\n";
    cout << "5. Eliminar evento por nombre\n";
    cout << "6. Detectar franjas libres en un dia\n";
    cout << "7. Mostrar resumen semanal\n";
    cout << "8. Guardar Agenda\n";
    cout << "0. Salir\n\n";
    cout << "Seleccione una opcion: ";
}

int main(){

    AgendaEventos agenda("datos/agenda.txt");

    int opcion = -1;

    while (opcion != 0){
        mostrarMenu();

        cin >> opcion;

        /* REVISAR SWITCH PARA COMPROBACIONES (cin sobretodo) */
        switch (opcion){
            case 1: {
                cout << agenda.mostrarTodosLosEventos() << endl;
                break;
            }

            case 2: {

                string nombre;
```

```
                cout << "Introduce el nombre del evento a buscar: ";

                cin.ignore(numeric_limits<streamsize>::max(), '\n');
                getline(cin, nombre);

                cout << agenda.buscarEventoPorNombre(nombre) << endl;
                break;
            }

            case 3: {
                int dia;

                cout << "Introduce el dia a buscar (1-7): ";
                cin >> dia;

                string resultado = agenda.buscarEventoPorDia(dia);
                cout << resultado << endl;
                break;
            }

            case 4: {
                string nombre;
                int dia;
                double horaInicio, horaFin;

                cout << "Nombre del evento: ";
                cin >> nombre;

                cout << "Dia de la semana (1-7): ";
                cin >> dia;

                cout << "Hora de inicio (formato decimal, ej. 9.5 = 9:30): ";
                cin >> horaInicio;

                cout << "Hora de fin (formato decimal, ej. 11.0): ";
                cin >> horaFin;

                cin.ignore(numeric_limits<streamsize>::max(), '\n');


                Evento nuevo(nombre, dia, horaInicio, horaFin);

                if (agenda.agregarEvento(nuevo)) {
                    cout << "Evento agregado correctamente.\n";
                } else {
                    cout << "No se ha podido agregar el evento: hay solapamiento
con otro existente.\n";
                }


                break;
            }

            case 5: {
```

```
                string nombre;

                cout << "Introduce el nombre del evento a eliminar: ";
                cin >> nombre;

                if ( agenda.eliminarEvento(nombre) ) {
                    cout << "Evento eliminado correctamente.\n";
                } else {
                    cout << "No se ha encontrado ningun evento con ese nombre.\n";
                }

                break;
            }

            case 6: {
                int dia;

                cout << "Introduce el dia (1-7) para ver las franjas libres: ";
                cin >> dia;

                cout << agenda.detectarFranjasLibresPorDia(dia) << endl;

                break;
            }

            case 7: {
                cout << agenda.resumenSemanal() << endl;
                break;
            }

            case 8: {
                cout << "Guardando Agenda en 'agenda.txt' ";

                if (agenda.guardarArchivo()) {
                    cout << "Archivo guardado correctamente.\n";
                } else {
                    cout << "Error: no se pudo guardar el archivo.\n";
                }

                break;
            }

            case 0:
                agenda.~AgendaEventos();
                cout << "\n Saliendo del programa. \n";
                break;

        default:
            cout << "\n Opción no válida. \n";
            break;
        }
    }
```

```
}
```

## OTROS ( MAKEFILE, AGENDA, ETC. )

```
# ==============================    Makefile Practica ED 2
==============================
CXX = g++
CXXFLAGS = -std=c++11 -Iinclude -Wall -Wextra

SRC = src
BUILD = build

OBJS = $(BUILD)/main.o \
       $(BUILD)/agendaEventos.o \
       $(BUILD)/evento.o

TARGET = agenda

all: $(TARGET)

$(TARGET): $(OBJS)
    $(CXX) $(OBJS) -o $(TARGET)

# --- reglas individuales ---

$(BUILD)/main.o: $(SRC)/main.cpp include/agendaEventos.h include/Evento.h
    @if not exist $(BUILD) mkdir $(BUILD)
    $(CXX) $(CXXFLAGS) -c $(SRC)/main.cpp -o $(BUILD)/main.o

$(BUILD)/agendaEventos.o: $(SRC)/agendaEventos.cpp include/agendaEventos.h
include/Evento.h
    @if not exist $(BUILD) mkdir $(BUILD)
    $(CXX) $(CXXFLAGS) -c $(SRC)/agendaEventos.cpp -o $(BUILD)/agendaEventos.o

$(BUILD)/evento.o: $(SRC)/evento.cpp include/Evento.h
    @if not exist $(BUILD) mkdir $(BUILD)
    $(CXX) $(CXXFLAGS) -c $(SRC)/evento.cpp -o $(BUILD)/evento.o
```

```
Clase de Matematicas,1,9,11
Laboratorio de Fisica,1,11,13
Tutoria con Maria,2,10,11
Clase de Programacion,3,9,11
Reunion de grupo,3,11,12
Evento1,7,10,12
```

```
Ev1,6,10,12
```

Y a continuación tenemos algunas imagenes del funcionamiento:

```
You, hace 27 segundos | 1 author (You)
1   Clase de Matemáticas,1,9,11        You, hace 19 horas • Casi Terminada
2   Laboratorio de Física,1,11,13
3   Tutoría con María,2,10,11
4   Clase de Programación,3,9,11
5   Reunión de grupo,3,11,12
6   Evento1,7,10,12
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN **TERMINAL** PUERTOS GITLENS

```
1. Mostrar todos los eventos
2. Buscar evento por nombre
3. Buscar eventos por dia
4. Agregar nuevo evento
5. Eliminar evento por nombre
6. Detectar franjas libres en un dia
7. Mostrar resumen semanal
8. Guardar Agenda
0. Salir

Seleccione una opcion: 2
Introduce el nombre del evento a buscar: Evento1

 Evento Encontrado:

 Evento1 | 7 | 10.000000 - 12.000000

========= AGENDA DE EVENTOS =========
1. Mostrar todos los eventos
2. Buscar evento por nombre
3. Buscar eventos por dia
4. Agregar nuevo evento
5. Eliminar evento por nombre
6. Detectar franjas libres en un dia
7. Mostrar resumen semanal
8. Guardar Agenda
0. Salir

Seleccione una opcion: 2
Introduce el nombre del evento a buscar: Clase de Matemáticas
Evento no encontrado.
```

```
========= AGENDA DE EVENTOS =========
1. Mostrar todos los eventos
2. Buscar evento por nombre
3. Buscar eventos por dia
4. Agregar nuevo evento
5. Eliminar evento por nombre
6. Detectar franjas libres en un dia
7. Mostrar resumen semanal
8. Guardar Agenda
0. Salir

Seleccione una opcion: 1

 Lista de eventos (ordenados por día y hora):
```

```
Clase de Matematicas | 1 | 9.000000 - 11.000000
Laboratorio de Fisica | 1 | 11.000000 - 13.000000
Tutoria con Maria | 2 | 10.000000 - 11.000000
Clase de Programacion | 3 | 9.000000 - 11.000000
Reunion de grupo | 3 | 11.000000 - 12.000000
Evento1 | 7 | 10.000000 - 12.000000
```

```
========= AGENDA DE EVENTOS =========
1. Mostrar todos los eventos
2. Buscar evento por nombre
3. Buscar eventos por dia
4. Agregar nuevo evento
5. Eliminar evento por nombre
6. Detectar franjas libres en un dia
7. Mostrar resumen semanal
8. Guardar Agenda
0. Salir

Seleccione una opcion: 3
Introduce el dia a buscar (1-7): 2

 Evento Encontrado:

 Tutoria con Maria | 2 | 10.000000 - 11.000000


========= AGENDA DE EVENTOS =========
1. Mostrar todos los eventos
2. Buscar evento por nombre
3. Buscar eventos por dia
4. Agregar nuevo evento
5. Eliminar evento por nombre
6. Detectar franjas libres en un dia
7. Mostrar resumen semanal
8. Guardar Agenda
0. Salir

Seleccione una opcion: 3
Introduce el dia a buscar (1-7): 1

 Evento Encontrado:

 Clase de Matematicas | 1 | 9.000000 - 11.000000
Laboratorio de Fisica | 1 | 11.000000 - 13.000000
```

```
3. Buscar eventos por dia
4. Agregar nuevo evento
5. Eliminar evento por nombre
6. Detectar franjas libres en un dia
7. Mostrar resumen semanal
8. Guardar Agenda
0. Salir

Seleccione una opcion: 4
Nombre del evento: Ev1
Dia de la semana (1-7): 6
Hora de inicio (formato decimal, ej. 9.5 = 9:30): 10
Hora de fin (formato decimal, ej. 11.0): 12
Evento agregado correctamente.

========= AGENDA DE EVENTOS =========
1. Mostrar todos los eventos
2. Buscar evento por nombre
3. Buscar eventos por dia
4. Agregar nuevo evento
5. Eliminar evento por nombre
6. Detectar franjas libres en un dia
7. Mostrar resumen semanal
8. Guardar Agenda
0. Salir

Seleccione una opcion: 1

 Lista de eventos (ordenados por día y hora):

Clase de Matematicas | 1 | 9.000000 - 11.000000
Laboratorio de Fisica | 1 | 11.000000 - 13.000000
Tutoria con Maria | 2 | 10.000000 - 11.000000
Clase de Programacion | 3 | 9.000000 - 11.000000
Reunion de grupo | 3 | 11.000000 - 12.000000
Ev1 | 6 | 10.000000 - 12.000000
Evento1 | 7 | 10.000000 - 12.000000
```

```
========= AGENDA DE EVENTOS =========
1. Mostrar todos los eventos
2. Buscar evento por nombre
3. Buscar eventos por dia
4. Agregar nuevo evento
5. Eliminar evento por nombre
6. Detectar franjas libres en un dia
7. Mostrar resumen semanal
8. Guardar Agenda
0. Salir

Seleccione una opcion: 8
Guardando Agenda en 'agenda.txt' Archivo guardado correctamente.
```