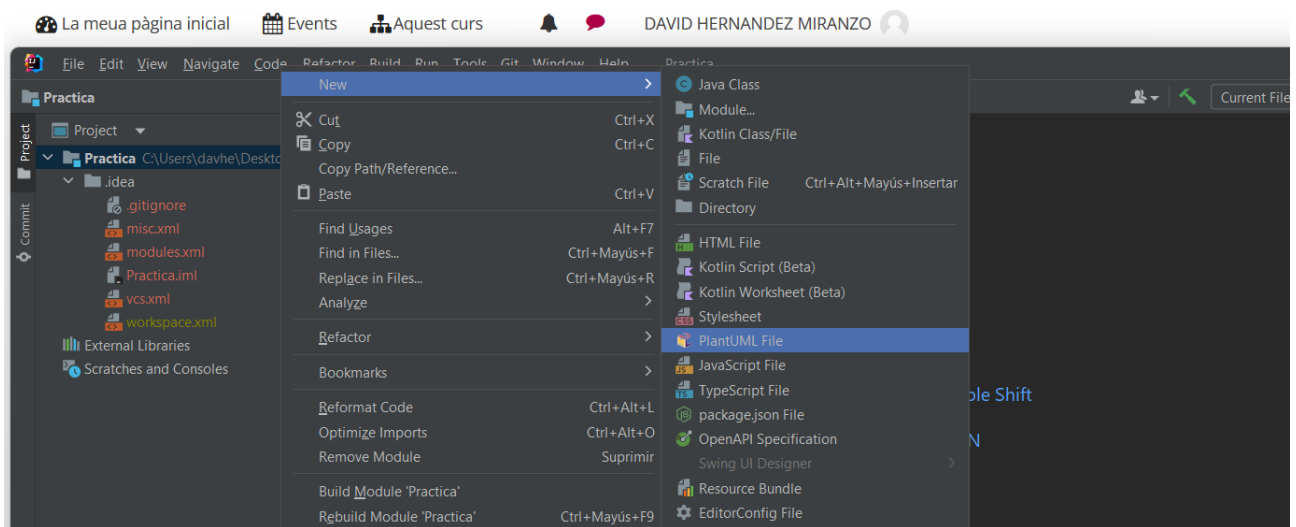


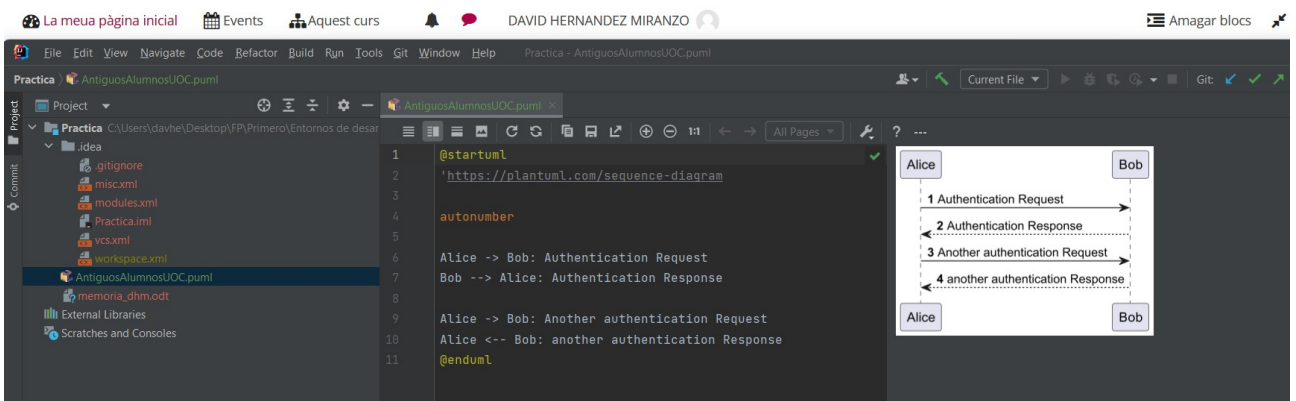
Memoria Práctica PlantUML

Enlace a proyecto: <https://github.com/Davhemir/UMLClasesDavidHM.git>

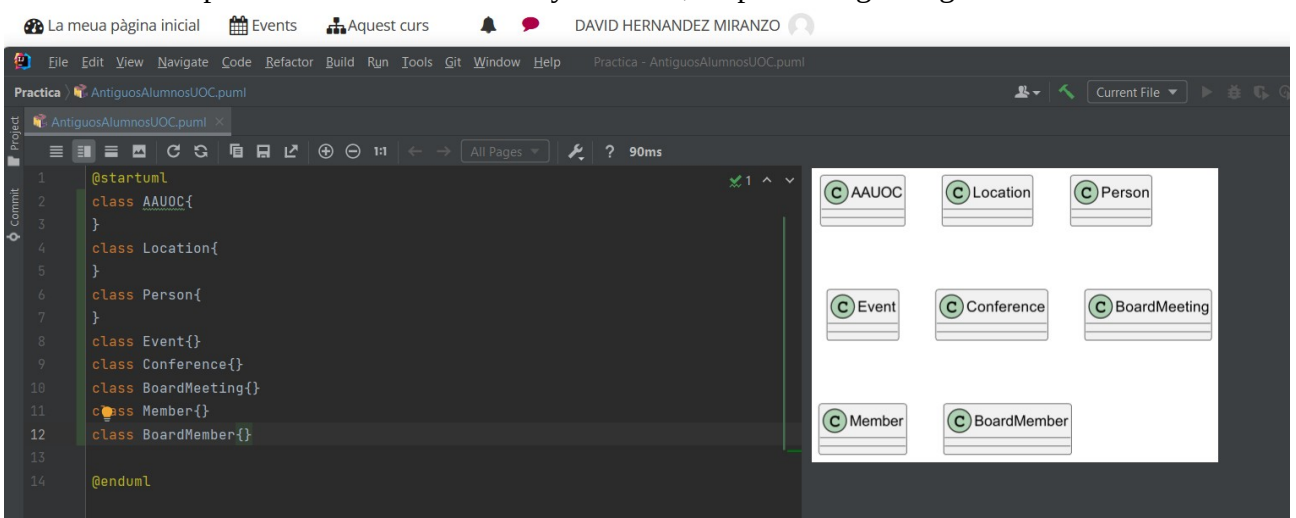
Creación archivo Uml en proyecto:



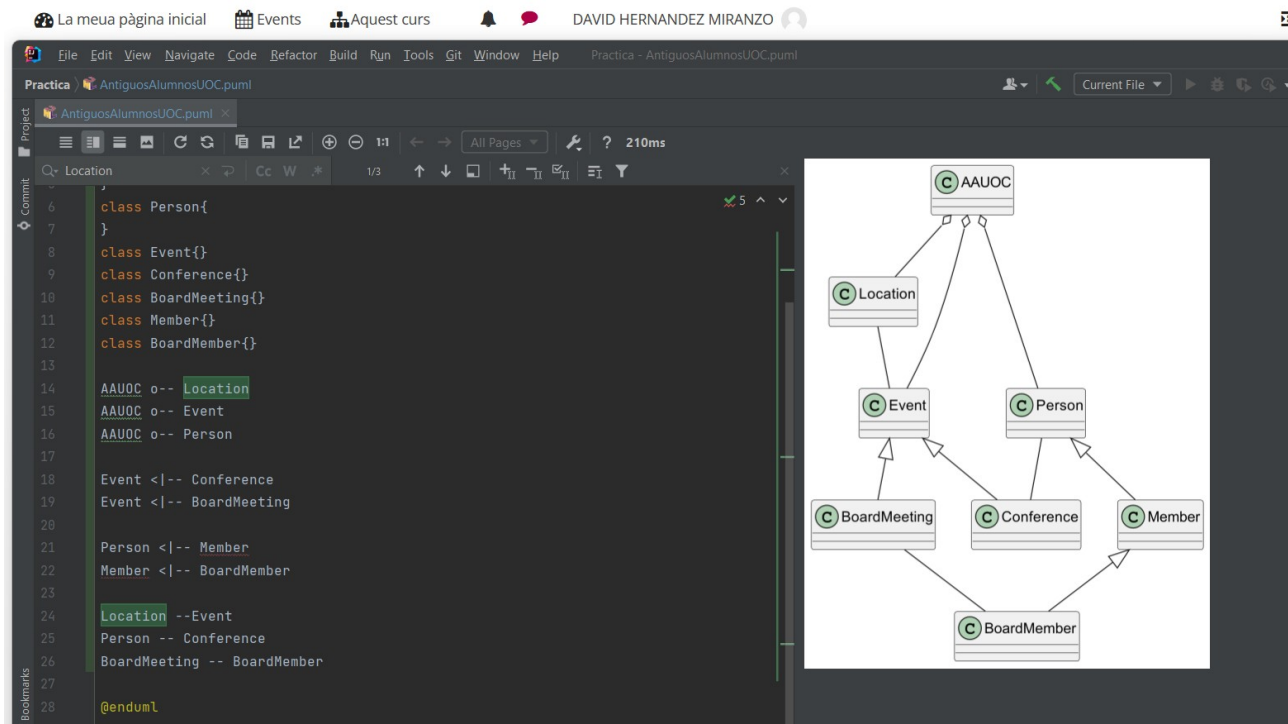
Archivo añadido y plugin funcionando:



Se añaden todas las clases que formarán el esquema UML precedidos de la palabra class y con llaves. No se especifican todavía atributos y métodos, en primer lugar se generaran las relaciones:



Una vez creadas las clases se establecen las relaciones:



Se generan diferentes tipos de relaciones:

Relaciones de Extensión entre superclases y subclases: <|--

Relaciones de Agregación: *--

Relaciones de Asociación: --

*Se ha añadido posteriormente la relación de asociación entre Member y Event

Inclusión de atributos y métodos a las clases:

```
class AAUOC{
    newLocation(l: Location): void
    newEvent(e: Event): void
    newPerson(p: Person): void
    informEvent(e: Event): void
    register(m: Member, e: Event): void
}
class Location{
    description: String
    address: String
}
class Person{
    name: String
}
class Member{
    e-mail: String
}
class BoardMember{}
class Event{
    date: Date
    description: String
    assign(l: Location): void
}
class Conference{
    max-attendees: Integer
}
class BoardMeeting{}
```

Los atributos se añaden con la sintaxis:

nombreAtributo: tipoAtributo

Los métodos se añaden con la sintaxis:

nombreMetodo(nombreParametro: tipoParametro): TipoVariableReturn

En este caso no se han añadido modificadores de acceso.

Inclusión de los nombres de las relaciones:

```
AAUOC o-- Location
AAUOC o-- Event
AAUOC o-- Person

Event <|-- Conference
Event <|-- BoardMeeting

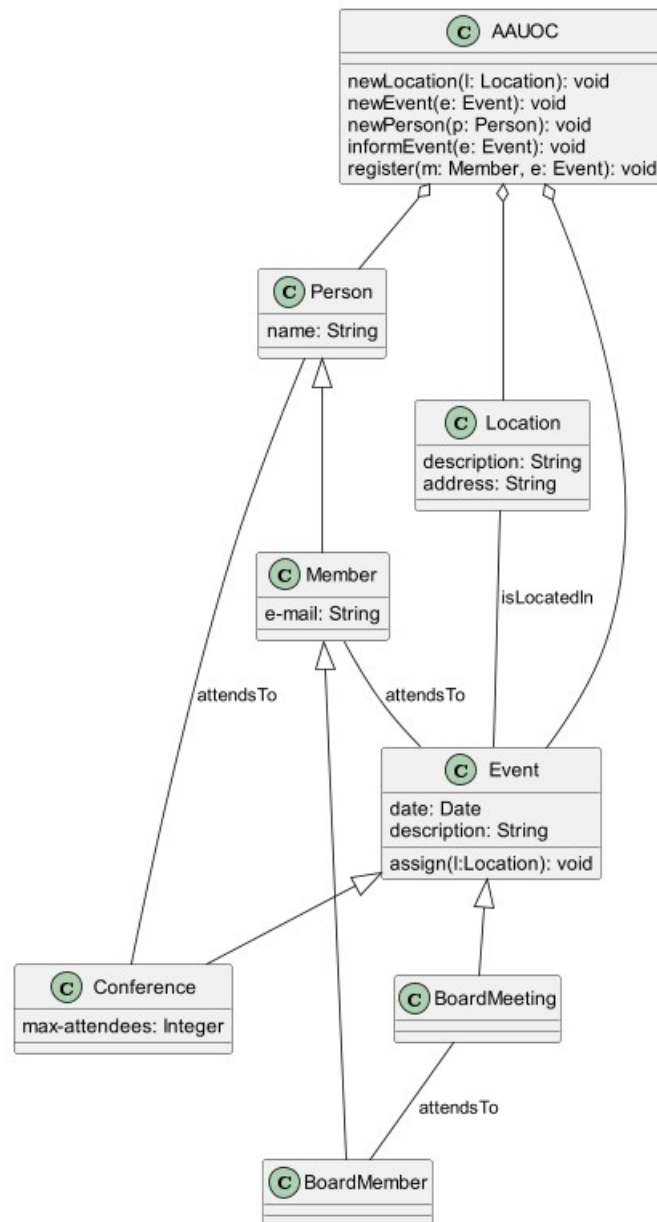
Person <|-- Member
Member <|-- BoardMember

Member --Event : attendsTo
Location --Event : isLocatedIn
Person -- Conference: attendsTo
BoardMeeting -- BoardMember : attendsTo
```

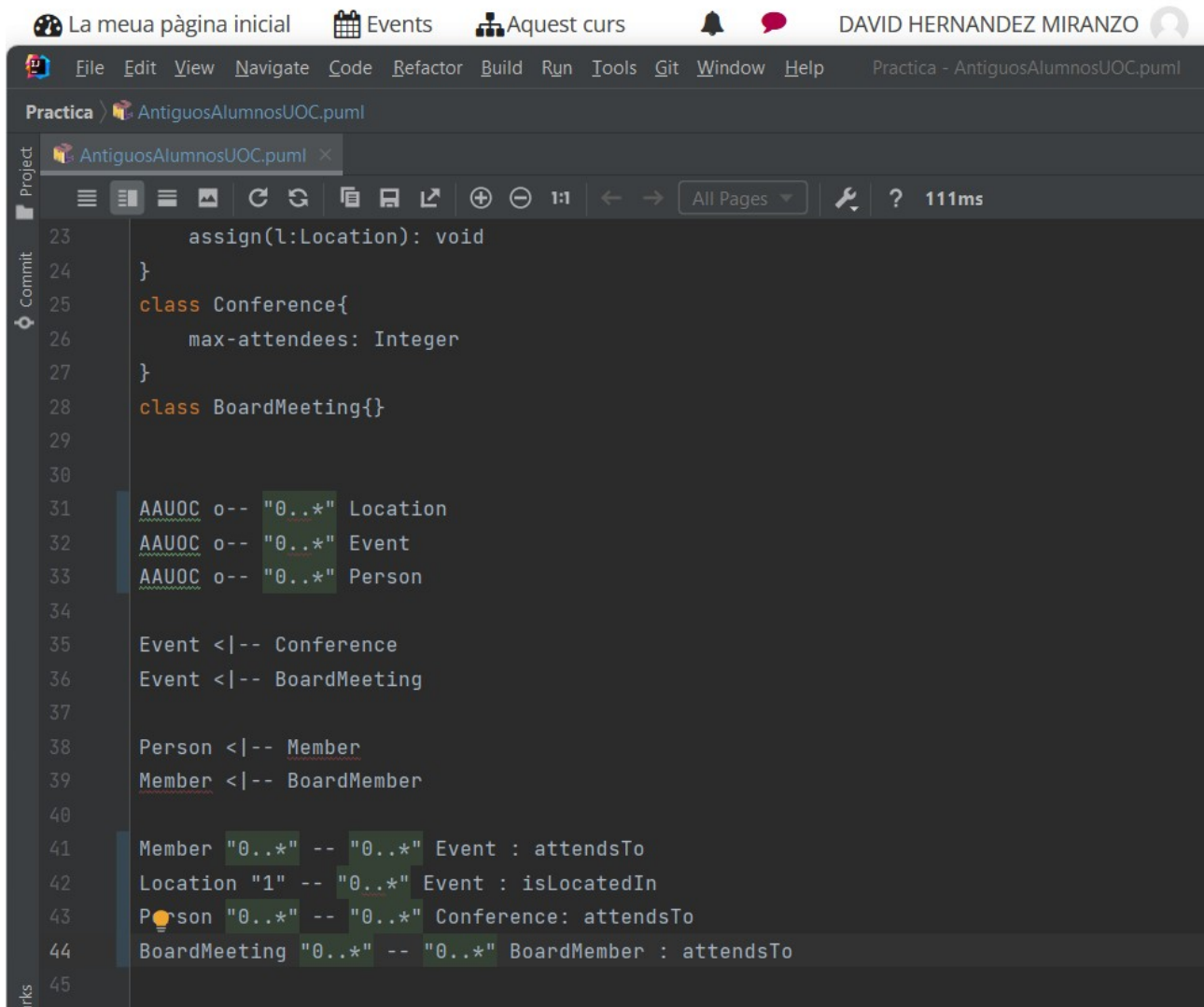
Los nombres de las relaciones se añaden tras : al final de la relación.

PrimerElemento – SegundoElemento : NombreRelación

Visualización actual del esquema:



Cardinalidad de las relaciones:



The screenshot shows an IDE window titled 'Practica - AntiguosAlumnosUOC.puml'. The code defines a class diagram with the following elements:

```
23     assign(l:Location): void
24 }
25 class Conference{
26     max-attendees: Integer
27 }
28 class BoardMeeting{}
29
30
31 AAUOC o-- "0..*" Location
32 AAUOC o-- "0..*" Event
33 AAUOC o-- "0..*" Person
34
35 Event <|-- Conference
36 Event <|-- BoardMeeting
37
38 Person <|-- Member
39 Member <|-- BoardMember
40
41 Member "0..*" -- "0..*" Event : attendsTo
42 Location "1" -- "0..*" Event : isLocatedIn
43 Person "0..*" -- "0..*" Conference: attendsTo
44 BoardMeeting "0..*" -- "0..*" BoardMember : attendsTo
45
```

The diagram includes a sidebar with 'Project', 'Commit', and 'Marks' views. The top bar shows navigation and editing tools, and the bottom status bar indicates '111ms'.

Las cardinalidades se añaden a cada parte de la relación entre comillas dobles:

Elemento1 “cardinalidad1” – “cardinalidad2” Elemento2

En este caso se utilizan dos tipos de cardinalidades

-”0..*” → Indica que puede ser 0 o cualquier número de elementos

-”1” → Indica que solamente puede haber un elemento en la relación

Resultado final del esquema UML de clases

