

PlantUML

ASW - Grupo SemEs1-01

Integrantes

— — —

Iván Rodríguez - UO265368

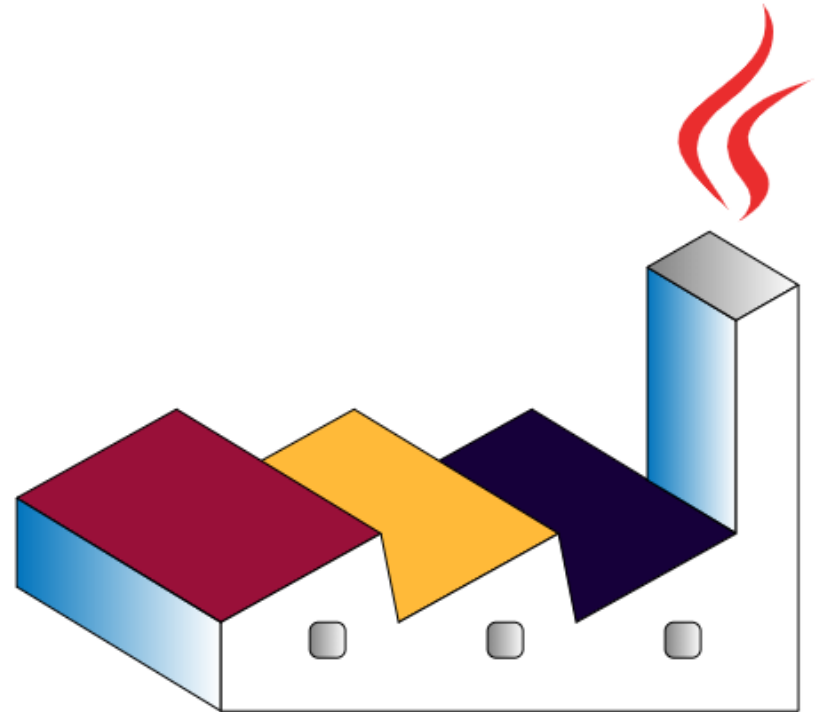
Israel Méndez Rodríguez - UO263845

Mario Miguel Blanco - UO264135

Daniel Pancho Cueto - UO245313

Gaspar Pisa Eyaralar - UO250825

PlantUML



¿Qué es PlantUML?

— — —

-herramienta de **código abierto** la cual permite **graficar diagramas UML**

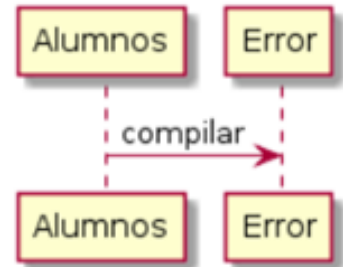
-lenguaje **simple**

-tener cuidado con la **inconsistencia**

-aplicación más de **dibujo** que de modelado

plantuml.com

```
@startuml
Alumnos -> Error : compilar
@enduml
```



Historia

plantUML fue desarrollado por Arnau Roques

— — —

- Arnau **re-desarrolló** un software capaz de **leer diagramas de secuencia** en formato de **texto** y de integrar las correspondientes **imágenes generadas en un doc. de Word**.

Lo dividió en **dos tareas**:

1-**software de Java** que genera imágenes PNG a partir de descripciones textuales.

2- **macro de Word** que añade un botón a Word y llama a este software de Java.

-Así fue el comienzo del proyecto en **2009**.

Características

PlantUML soporta varios de los diagramas más usados:

— — —

- Diagramas de Secuencia
- Diagramas de Clases
- Diagramas de Estados
- Diagramas de Actividad



Ventajas:

- Integración con una gran cantidad de aplicaciones
- Simplicidad e intuitividad de su sintaxis

Ej:

herencia en diagrama de clases: <|--

flecha en un diagrama de secuencia: ->

Objetivo

Objetivo

— — —

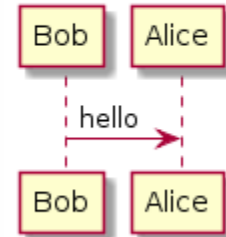
- PlantUML es usado para **graficar** diagramas UML, usando una descripción de texto simple y fácil de leer por humanos.
- Es más una herramienta de **dibujo** que una de modelado (Enlaces svg o png).

Ejemplo:

(Puedes editar el texto que quieras y el diagrama se actualizará dinámicamente!)

```
Bob->Alice : hello
```

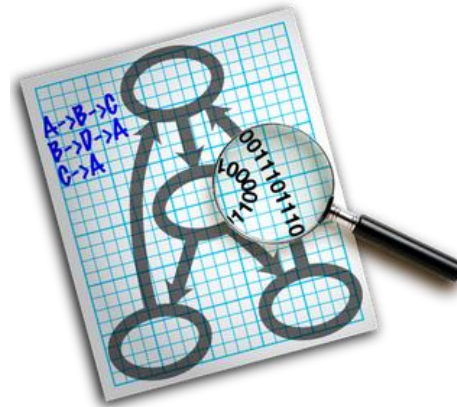
describe el siguiente diagrama :



Software

¿Qué se necesita?

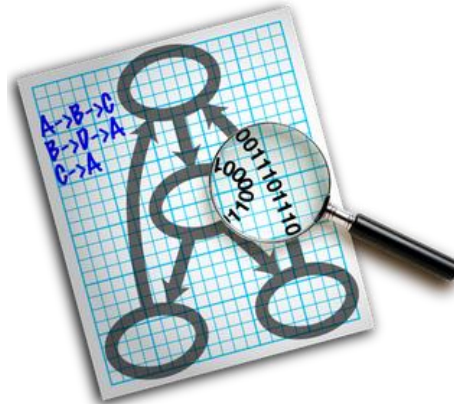
- Java
- Graphviz (opcional -> diagramas de secuencia y actividad)



Graphviz

— — —

- Open source **Graph Visualization** software.
- Es un conjunto de herramientas de **software** para el **diseño de diagramas** definido en el lenguaje descriptivo **DOT**. Fue desarrollado por *AT&T Labs* y liberado como **software libre** con licencia tipo **Eclipse**.



Arquitectura de Graphviz

— — —

- Graphviz consiste en un lenguaje de descripción de gráficos (en texto plano) llamado **DOT**, un conjunto de herramientas y librerías que pueden generar o procesar archivos DOT
- Herramientas:
 - dot
 - neato
 - sfdp
 - fdp
 - twopi
 - circo
 - dotty
 - lefty

DOT

— — —

- Define grafos, pero no provee un framework para generarlos. Existen varios programas para ello, entre los que se encuentra Graphviz, el cual a su vez es empleado para PlantUML.
- Sintaxis:
 - Gráficas indirectas (grafos no dirigidos).
 - Grafos dirigidos.
 - Atributos.
 - Comentarios.

Diagramas UML

Diagrama de secuencia

```
@startuml
Alice -> Bob: Authentication Request
Bob --> Alice: Authentication Response

Alice -> Bob: Another authentication Request
Alice <-- Bob: another authentication Response
@enduml
```

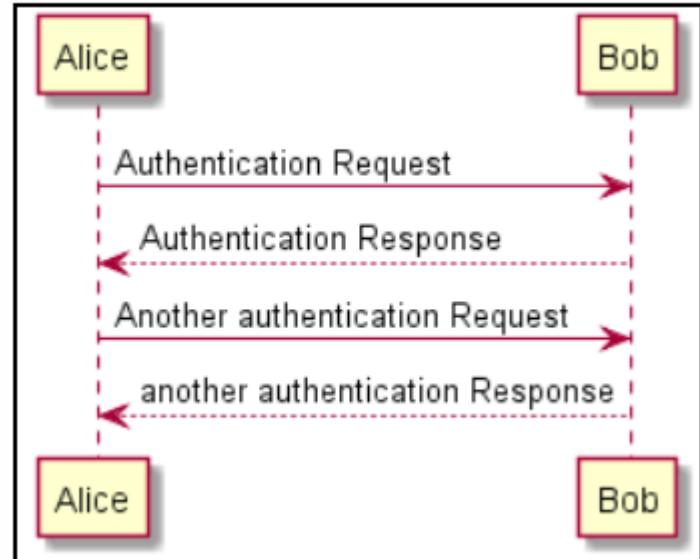


Diagrama de casos de uso

```
@startuml
left to right direction
skinparam packageStyle rectangle
actor customer
actor clerk
rectangle checkout {
  customer -- (checkout)
  (checkout) .> (payment) : include
  (help) .> (checkout) : extends
  (checkout) -- clerk
}
@enduml
```

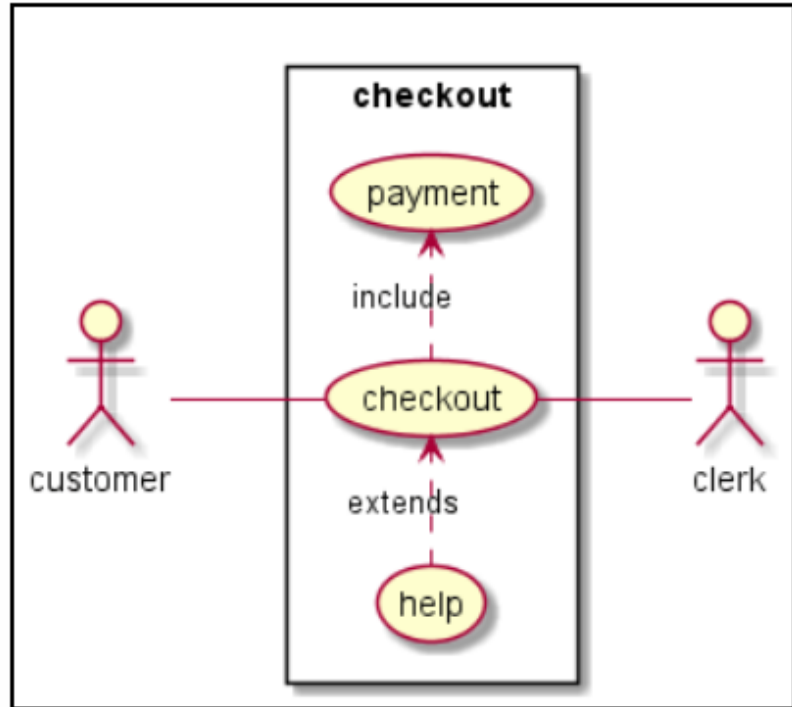
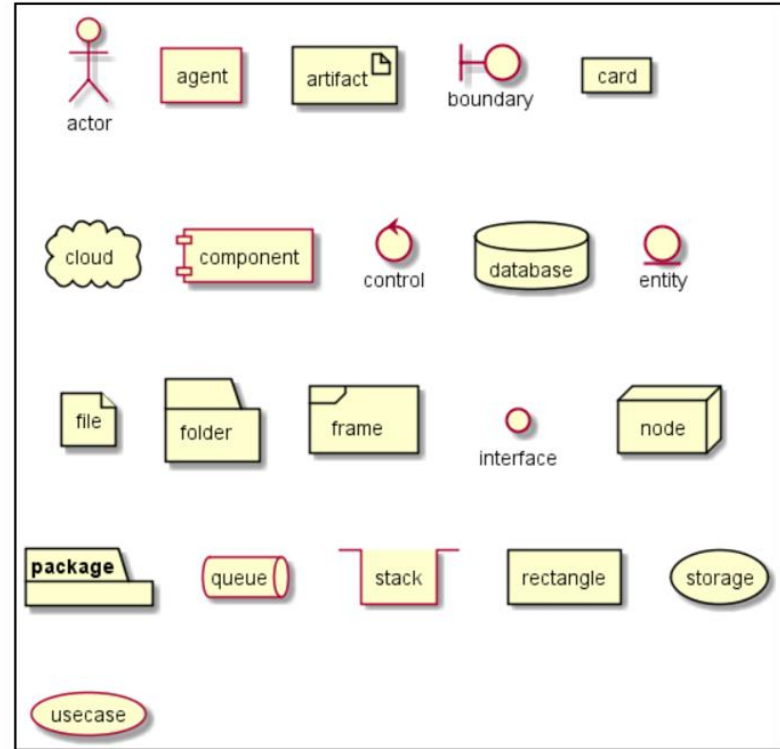


Diagrama de despliegue

```
@startuml
actor actor
agent agent
artifact artifact
boundary boundary
card card
cloud cloud
component component
control control
database database
entity entity
file file
folder folder
frame frame
interface interface
node node
package package
queue queue
stack stack
rectangle rectangle
storage storage
usecase usecase
@enduml
```



**UNIFIED
MODELING
LANGUAGE™**



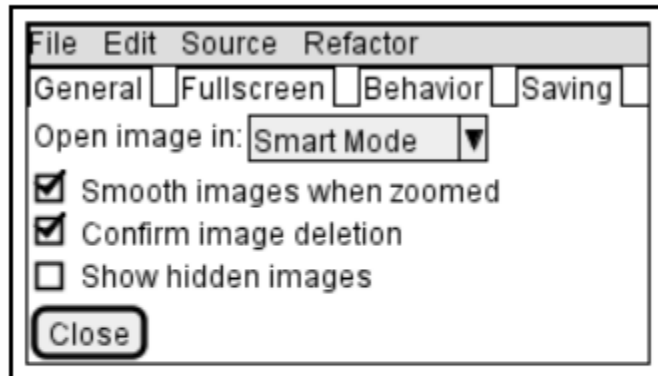
- Clases
- Objetos
- Componentes
- Estados
- Despliegue
- Timing

Diagramas no UML

Wireframe

— — —

```
@startsalt
{+
{* File | Edit | Source | Refactor }
{/ General | Fullscreen | Behavior | Saving }
{
    { Open image in: | ^Smart Mode^ }
    [X] Smooth images when zoomed
    [X] Confirm image deletion
    [ ] Show hidden images
}
[Close]
}
@endsalt
```



MindMap

— — —

```
@startmindmap
+ OS
++ Ubuntu
+++ Linux Mint
+++ Kubuntu
+++ Lubuntu
+++ KDE Neon
++ LMDE
++ SolydXK
++ SteamOS
++ Raspbian
-- Windows 95
-- Windows 98
-- Windows NT
--- Windows 8
--- Windows 10
@endmindmap
```

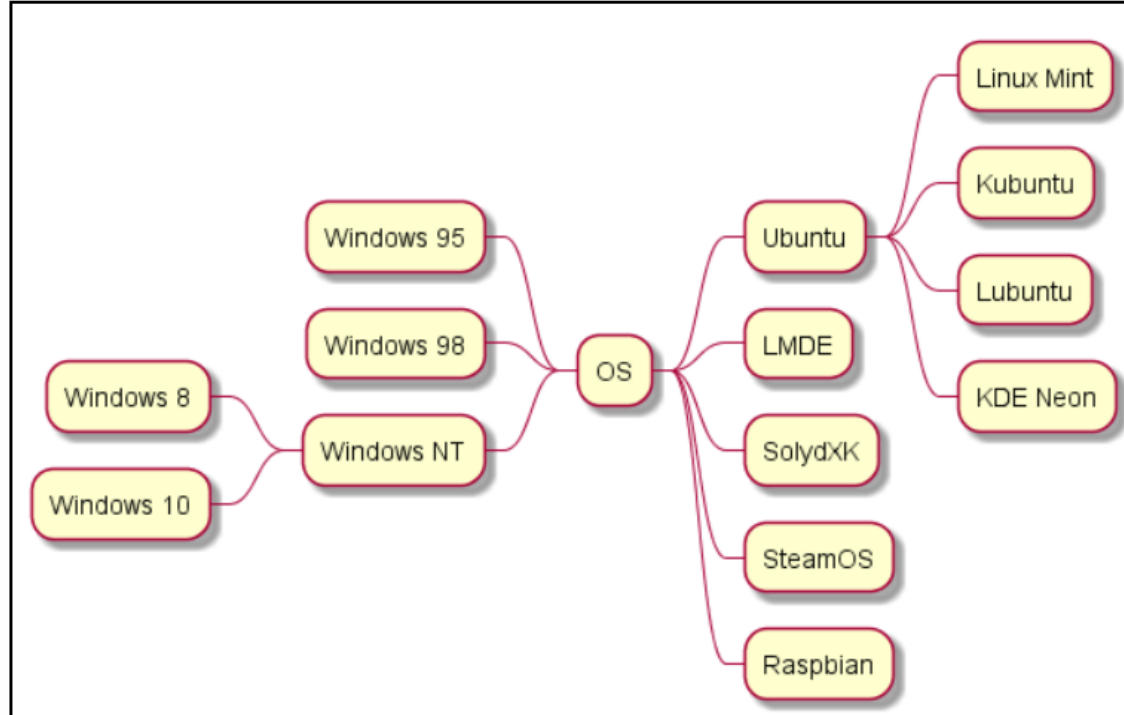
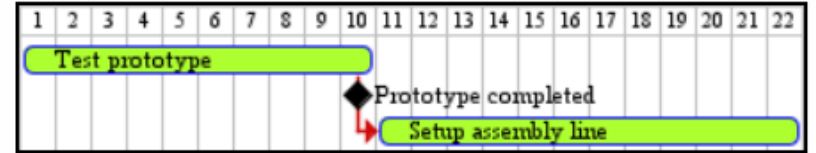


Diagrama de Grantt

```
@startgantt
[Test prototype] lasts 10 days
[Prototype completed] happens at [Test prototype]'s end
[Setup assembly line] lasts 12 days
[Setup assembly line] starts at [Test prototype]'s end
@endgantt
```



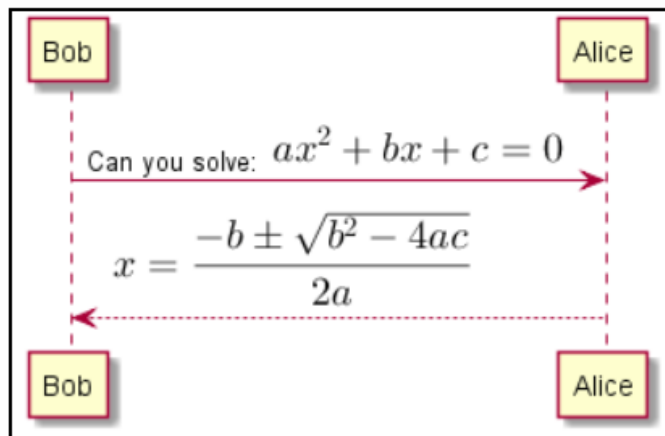
Maths

@startuml

Bob -> Alice : Can you solve: $ax^2+bx+c=0$

Alice --> Bob: $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$

@enduml



@startlatex

$\sum_{i=0}^{n-1} (a_i + b_i^2)$

@endlatex

$$\sum_{i=0}^{n-1} (a_i + b_i^2)$$

Usabilidad

Aplicaciones que utilizan PlantUML

— — —

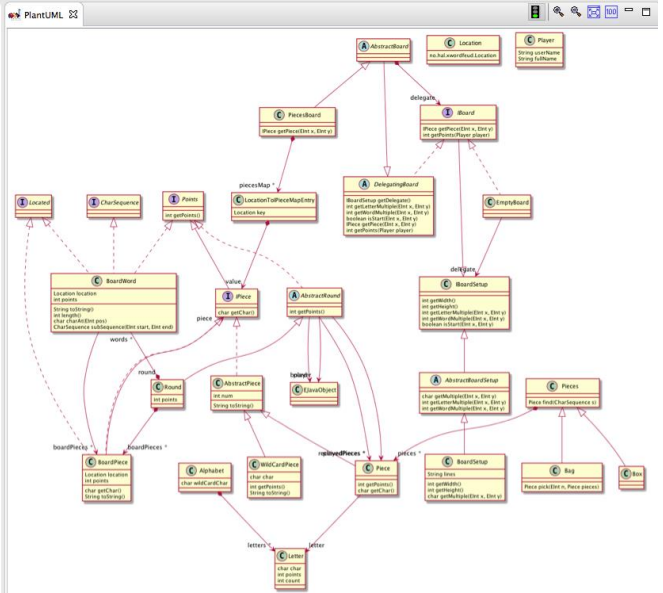
Las principales aplicaciones que utilizan PlantUML son las indicadas a continuación. Estas, lo utilizan incorporando diferentes Plugins o extensiones que se añaden al programa principal.

- Google Docs
- Eclipse
- Doxygen
- Confluence
- Atom
- Visual Studio Code
- Microsoft Word

- ```

1 xwordfeud.score 22
2
3 for (piece : boardPieces) {
4 var piecePoints = piece.points
5 if (round.boardPieces.contains(piece)) {
6 piecePoints = piecePoints * board.getLetterMultiple(piece.location.x,
7 wordMultiple = wordMultiple * board.getWordMultiple(piece.location.x,
8 }
9 points = points + piecePoints
10 }
11 points * wordMultiple
12 }
13
14 ap String toString() {
15 val StringBuilder buffer = new StringBuilder()
16 for (boardPiece : boardPieces) {
17 buffer.append(boardPiece.getChar())
18 }
19 buffer.append(boardPieces.get(0).location)
20 buffer.toString()
21 }
22
23 // from CharSequence
24
25 ap int length() {
26 boardPieces.size
27 }
28
29 ap char charAt(int pos) {
30 boardPieces.get(pos).getChar()
31 }
32
33 ap CharSequence subSequence(int start, int end) {
34 throw new UnsupportedOperationException(this.getClass() + " does not support
35 }
36 }
37
38 class Round extends AbstractRound {
39 contains BoardPiece[] boardPieces
40 contains BoardWord[] words opposite round
41
42 derived int points get {
43 var points = 0
44 for (word : words) {
45 points = points + word.points
46 }
47 points
48 }
49 }

```



The screenshot displays the PlantUML IDE interface. On the left, a Java code editor window titled 'Foo.java' contains the following code:

```
1 package demo;
2
3
4 public class Foo {
5
6 private int dummy;
7
8 public int getDummy() {
9 return 0;
10 }
11
12 }
13
```

On the right, a UML class diagram window titled 'PlantUML' shows the generated diagram for the provided Java code. The diagram consists of a single class box for 'Foo'. The box has a yellow background and a red border. It contains a circle with the letter 'C' in the top-left corner, indicating a class. Below the class name, there are two attributes listed: 'int dummy' and 'int getDummy()'. The 'int getDummy()' attribute is highlighted with a green circle, indicating it is the selected element.

# PlantUML Stakeholders

— — —

- Los stakeholders, o interesados en PlantUML son en general todos aquellos que deseen modificar sus complejos diagramas en una descripción de texto simple y fácil de leer por humanos.

Usos frecuentes:

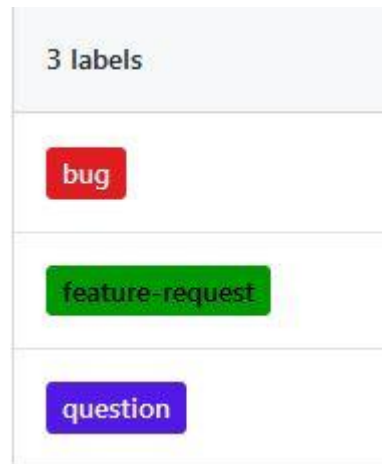
- Puede ser utilizado por alumnos, para desarrollar sus trabajos de forma legible para todo el mundo, por profesores que quieren hacer más fácil sus explicaciones para los alumnos o para cualquiera que quiera graficar sus diagramas UML y hacerlos más sencillos

Desarrollo

# Gestión de Issues

— — —

- Git
- FAQ
- Foro de la comunidad



|                        |       |
|------------------------|-------|
| All categories         |       |
| Question / help        | (405) |
| Bug                    | (964) |
| Wanted features        | (871) |
| Closed question / help | (87)  |
| Closed bug             | (51)  |
| Closed feature request | (124) |
| Won't fix              | (1)   |
| Won't implement        | (9)   |
| Can't help             | (9)   |
| To be deleted          | (8)   |
| To be sorted           | (419) |

\_\_\_\_\_

- [illegible]

The screenshot shows a development environment with a code editor on the left and a browser window on the right. The code editor displays a Jupyter Notebook cell with the following content:

```

01. Introduction and Goals
02. You should know all parties involved in development of
 the system or affected by the system.
03. Otherwise, you may get nasty surprises later in the
 development process.
04. These stakeholders determine the extent and the level of
 detail of your work and its results.
05.
06. Form
07. Table with role names, person names, and their
 expectations with respect to the architecture and its
 documentation.
08. ****
09.
10. [options="header",cols="1,2,2"]
11. |==
12. |Role|Name|Contact|Expectations
13. |_-Role-1_-|_-Contact-1_-|_-Expectation-1_-|
14. |_-Role-2_-|_-Contact-2_-|_-Expectation-2_-|
15. |==
16.
17. [plantuml, diagram, png]
18. ----
19. class AA
20. class DiagramBlock
21. class Ditaablock
22. class PlantUmlBlock
23.
24. AA <|-- DiagramBlock
25. DiagramBlock <|-- Ditaablock
26. DiagramBlock <|-- PlantUmlBlock
27. ----

```

The browser window shows the rendered output of the code, including the same table of contents, the rendered table, and the rendered class diagram.

# Conclusiones

- Software simple de usar y aprender
- Al alcance de todo el mundo
- Útil para prácticas de la asignatura





# Bibliografía

- <https://en.wikipedia.org/wiki/PlantUML>
- <https://plantuml.com/es/>
- <https://www.graphviz.org/>
- <https://es.wikipedia.org/wiki/DOT>
- [http://pdf.plantuml.net/PlantUML\\_Language\\_Reference\\_Guide\\_es.pdf](http://pdf.plantuml.net/PlantUML_Language_Reference_Guide_es.pdf)
- <https://www.adictosaltrabajo.com/2015/12/18/tutorial-plantuml-dibujar-diagramas-de-forma-sencilla/>
- <https://sourceforge.net/blog/july-2019-community-choice-project-month-plantuml/>

# Preguntas