

PlantUML

Customizar tus diagramas

1. Personalización

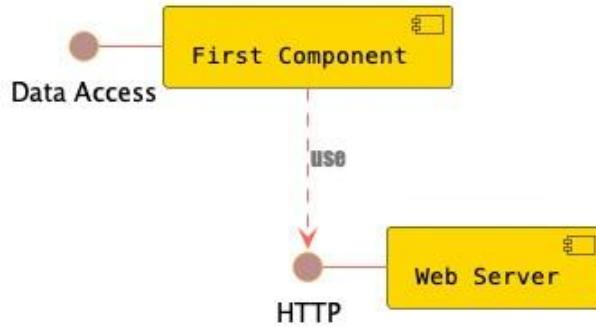
Se puede cambiar los colores y la fuente del dibujo usando el comando `skinparam`.

```
skinparam interface {
    backgroundColor RosyBrown
    borderColor orange
}

skinparam component {
    FontSize 13
    BackgroundColor<<Apache>> Red
    BorderColor<<Apache>> #FF6655
    FontName Courier
    BorderColor black
    BackgroundColor gold
    ArrowFontName Impact
    ArrowColor #FF6655
    ArrowFontColor #777777
}

() "Data Access" as DA

DA - [First Component]
[First Component] ...> () HTTP : use
HTTP - [Web Server] << Apache >>
```

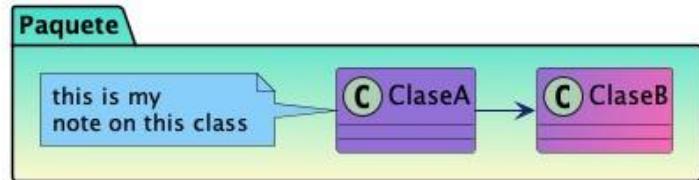


2. Colores

Las clases y los paquetes pueden tener distintos colores, se indica con `#` seguido del nombre del color que queremos usar. También puedes usar degradación de color en el fondo, con la siguiente sintaxis: dos nombres de colores separados por cualquier de los siguientes dependiendo de la dirección del degradado.

```
package Paquete #turquoise-lemonChiffon{
    note left of ClaseA #lightskyblue
        this is my
        note on this class
    end note

    class ClaseA #mediumPurple
    class ClaseB #mediumPurple/HotPink
}
ClaseA -[#midnightblue]right-> ClaseB
```

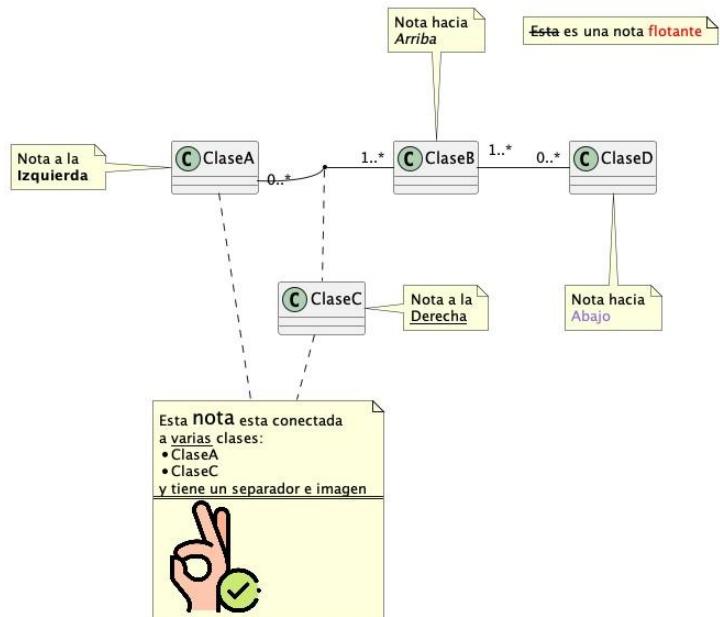


3. Notas

En PlantUML con “note” luego de un elemento se puede incorporar información suplementaria directamente en el diagrama para ayudar a la comprensión o para resaltar aspectos importantes.

En estas notas también es posible usar algunas etiquetas HTML como: ****, <u>, *<i>*, ~~<s>~~, ~~~~, ~~~~ o ~~~~, ~~<color:#AAAAAA>~~ or ~~<color:colorName>~~ para cambiar colores, ~~<size:nn>~~ para cambiar el tamaño de fuente, ~~~~ or ~~<img:file>~~ donde el archivo debe ser accesible por el sistema de archivos.

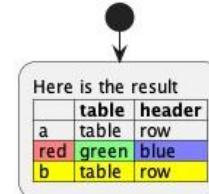
```
note as N1
    Esta <size:18>nota</size> esta conectada
    a <u>varias</u> clases:
    * ClaseA
    * ClaseC
    y tiene un separador e imagen
    ===
    <img:ok.png>
end note
ClaseA .. N1
ClaseC .. N1
class ClaseA
note left
    Nota a la
    <b>Izquierda</b>
end note
class ClaseB
note top
    Nota hacia
    <i>Arriba</i>
end note
class ClaseD
note bottom
    Nota hacia
    <font color=mediumpurple>Abajo</font>
end note
class ClaseC
note right
    Nota a la
    <u>Derecha</u>
end note
ClaseA "0..*" - "1..*" ClaseB
ClaseB "1..*" - "0..*" ClaseD
(ClaseA, ClaseB) .. ClaseC
note as N2
    <del>Esta</del> es una nota <color:#red>flotante</color>
end note
```



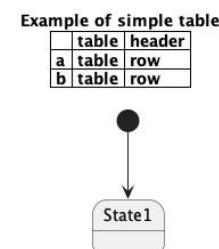
4. Tablas

Se pueden agregar tablas al diagrama, ya sea dentro de un estado como fuera del mismo usando un título (title).

```
start
:Here is the result
|= |= table |= header |
| a | table | row |
|<#FF8080> red |<#80FF80> green |<#8080FF> blue |
<#yellow>| b | table | row |;
```



```
skinparam titleFontSize 14
title
Example of simple table
|= |= table |= header |
| a | table | row |
| b | table | row |
end title
[*] --> State1
```



5. Títulos y leyendas

Se pueden agregar a los diagramas títulos, leyendas, pie de foto, encabezados y pie de diagrama.

```
@startuml "titleExample"
skinparam title {
    BorderRoundCorner 15
    BorderThickness 2
    BorderColor red
    BackgroundColor Aqua-CadetBlue
}

title Titulo\nen 2 líneas

header
    <font color=Aqua>IS</font>
    Ejemplo de encabezado
endheader

Alicia -> Juan: saluda()
Juan --> Alicia: Hola

caption Figura 1

legend top left
    Leyenda
end legend

center footer Ejemplo de pie de página centrado
@enduml
```

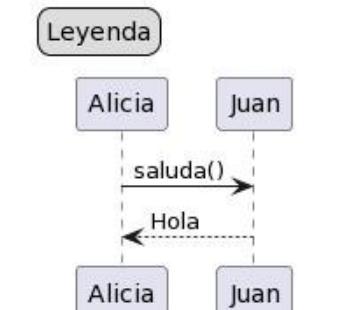


Figura 1
Ejemplo de pie de página centrado

6. Diagramas de Comunicación

Los diagramas de comunicación no existen como tal en PlantUML, pero los podemos hacer con diagrama de objetos y notas, y modificando algunas cosas que se dan por defecto como el color del fondo y del borde, así como esconder los atributos del objeto. Esto se hace con los siguientes comandos:

- `skinparam noteBackgroundColor transparent`
- `skinparam noteBorderColor transparent`
- `hide members` Por ejemplo:

```
@startuml "diagramaColaboracion"

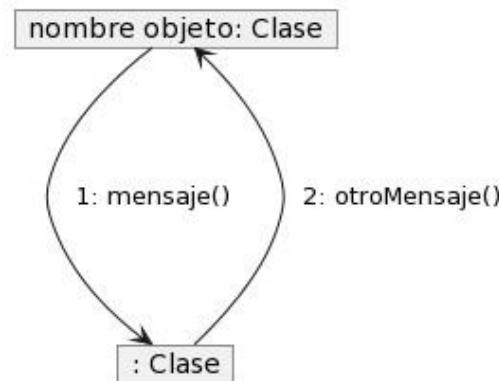
skinparam noteBackgroundColor transparent
skinparam noteBorderColor transparent
hide members

object "nombre objeto: Clase" as objeto1
object ": Clase" as objeto2

objeto1 -d--> objeto2
note on link
    1: mensaje()
end note

objeto2 -u--> objeto1
note on link
    2: otroMensaje()
end note

@enduml
```



Si desde Visual Code no le sale tal cual, por ejemplo, no esconde los atributos, puede cambiar en las preferencias de Visual Code para que renderice las imágenes desde el servidor de PlantUML:

