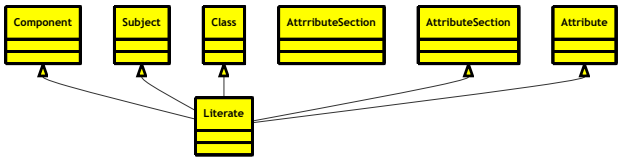
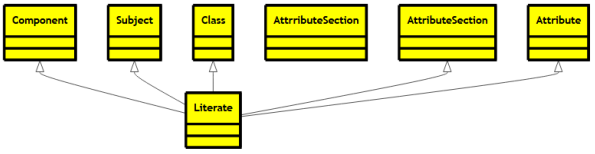
FIRST PAGE LEFT LEFT BLANK

**This is my first Mermaid test**

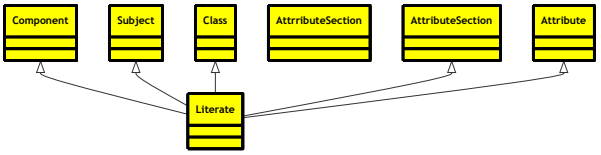
Mermaid Class Diagram



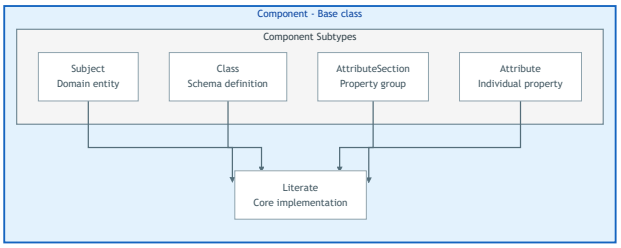

Mermaid PNG - mermaid_img14.png



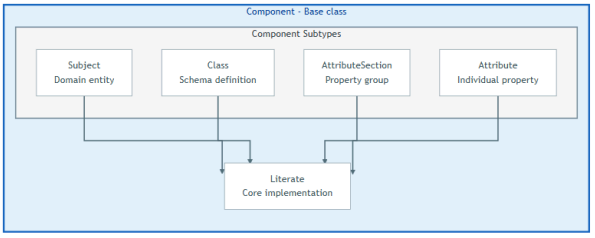

Mermaid SVG - mermaid_img14.svg



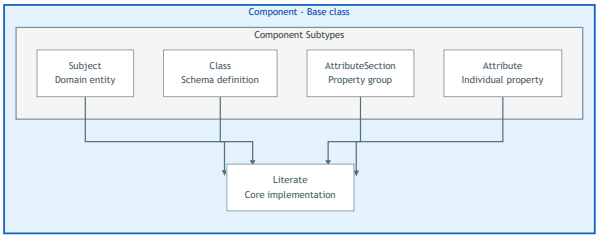
Mermaid Flowchart




Mermaid PNG - mermaid_img15.png

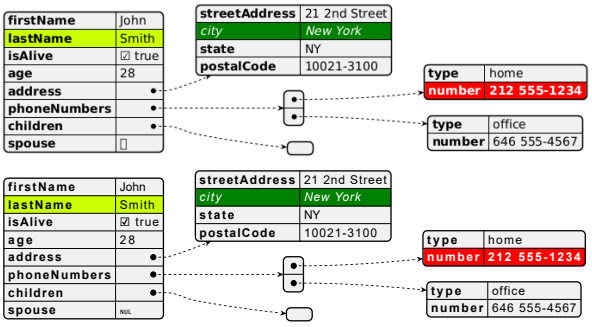



Mermaid SVG - mermaid_img15.svg



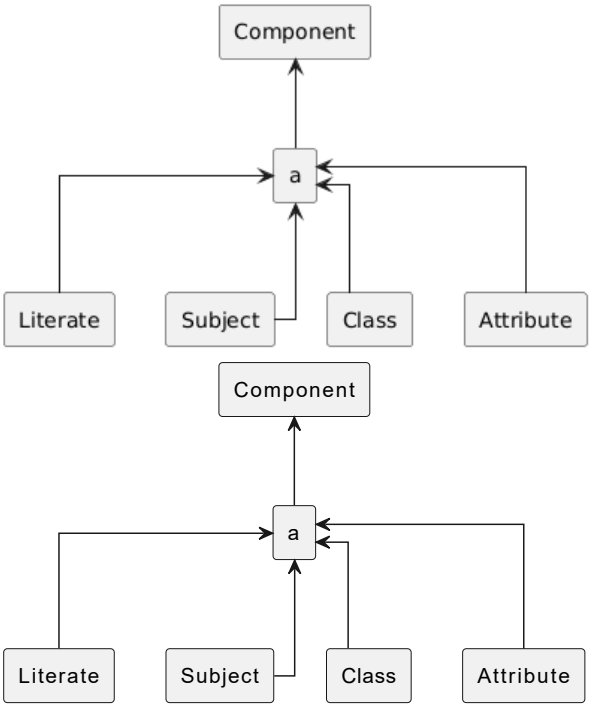
Plant UML jsondata

```
@startjson
<style>
.h1 {
BackGroundColor green
FontColor white
FontStyle italic
}
.h2 {
BackGroundColor red
FontColor white
FontStyle bold
}
</style>
#highlight "lastName"
#highlight "address" / "city" <<h1>>
#highlight "phoneNumbers" / "0" / "number" <<h2>>
{
"firstName": "John",
"lastName": "Smith",
"isAlive": true,
"age": 28,
"address": {
"streetAddress": "21 2nd Street",
"city": "New York",
"state": "NY",
"postalCode": "10021-3100"
},
"phoneNumbers": [
{
"type": "home",
"number": "212 555-1234"
},
{
"type": "office",
"number": "646 555-4567"
}
],
"children": [],
"spouse": null
}
@endjson
```

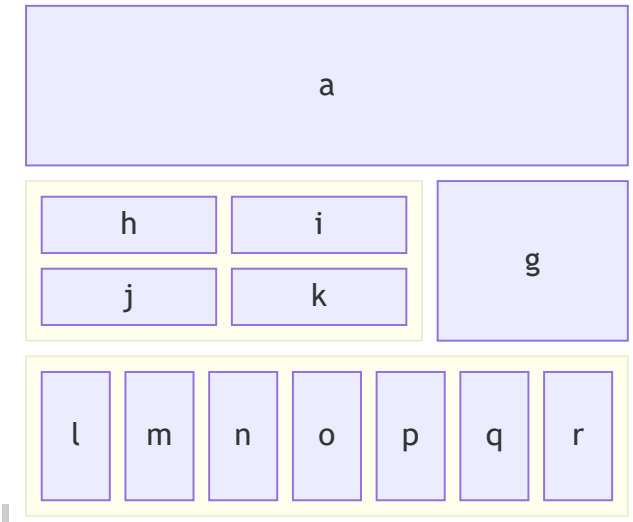


Plant UML UML

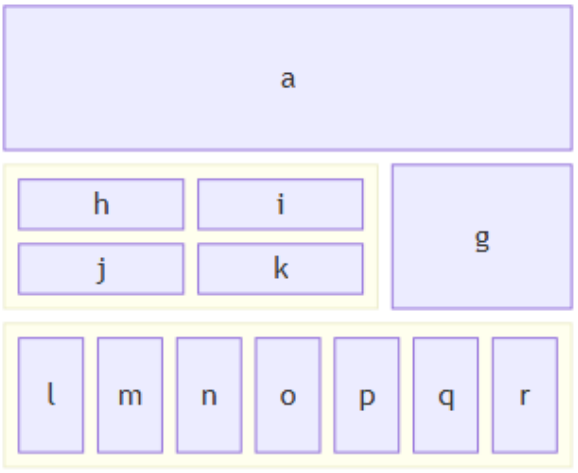
```
@startuml

rectangle Component
rectangle Literate
rectangle Subject
rectangle Class
rectangle Attribute
rectangle a

Literate -u->  a
Subject -u-> a
Class -u-> a
Attribute -u-> a
a -u-> Component
skinparam linetype ortho
@enduml
```



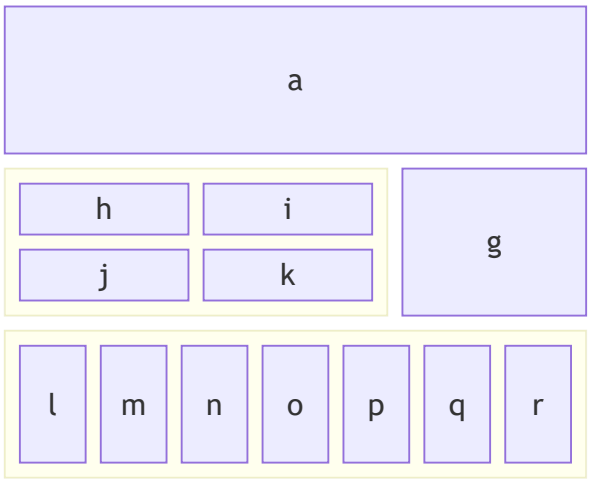
Mermaid block diagram



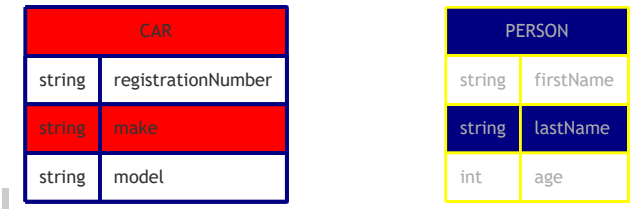

Mermaid PNG - mermaid_img18.png



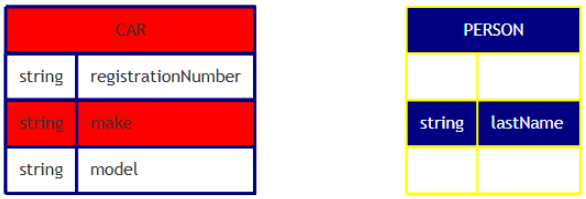
Mermaid SVG - mermaid_img18.svg



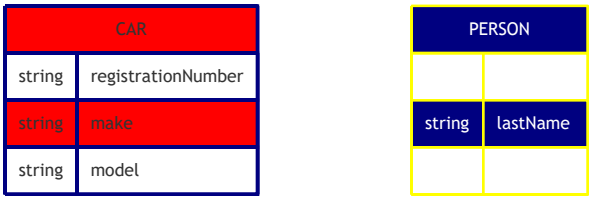
Mermaid ER Diagram



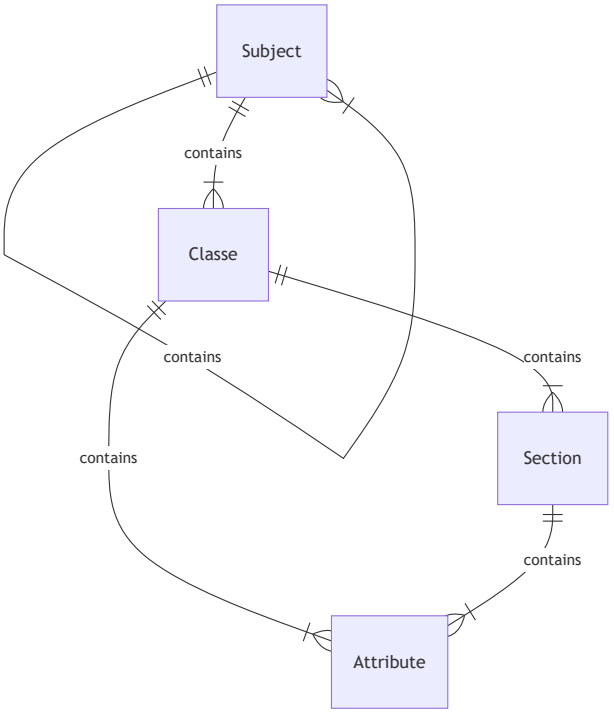

Mermaid PNG - mermaid_img19.png



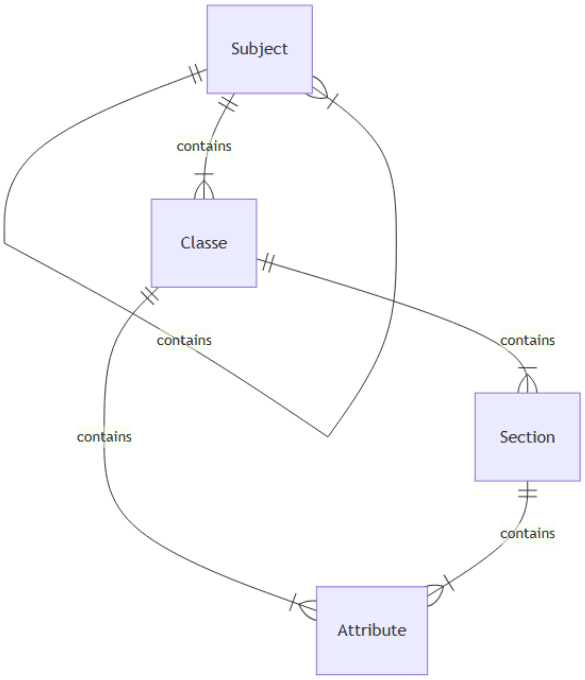

Mermaid SVG - mermaid_img19.svg



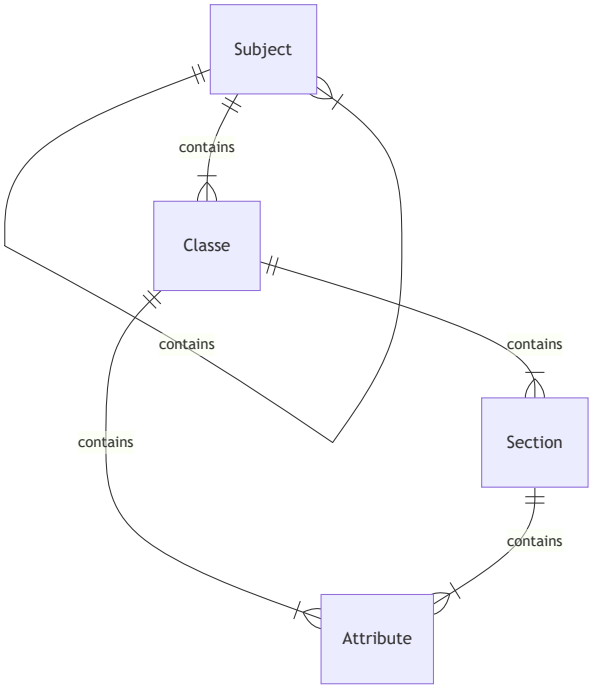
Mermaid ER Diagram


Subject
contains
Classe
contains
contains
contains
Section
contains
Attribute

Mermaid PNG - mermaid_img20.png




Mermaid SVG - mermaid_img20.svg



Captioned figure

+ 

+

*Fig.1 - Trulli, Puglia, Italy.*

+

And the same figure with figure/caption markup

**A Drivers License**

+  +

*My Non-Drivers License*

+

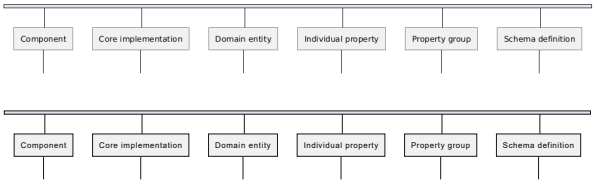
List of Codes

```
eFormat, Description
E-Book, 'Kindle or Apple books - etc'
PDF, formatted for printing and direct delivery
```

| | eFormat | Description |
|---|---|---|
| **0** | E-Book | 'Kindle or Apple books - etc' |
| **1** | PDF | formatted for printing and direct delivery |

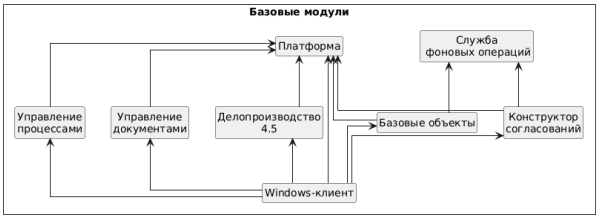
Plant UML UML

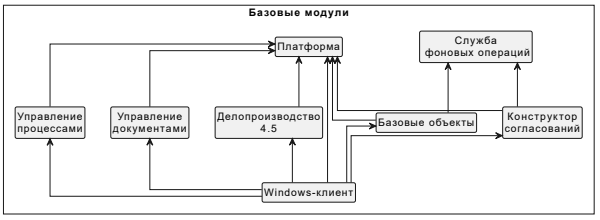
```
@startuml

nwdiag {

network {

Component;

Literate;

Subject;

Attribute;

AttributeSection;

Class;

Component -- Literate;

Component -- Subject;

Component -- Class;

Component -- AttributeSection;

Component -- Attribute;


Subject [description = "Domain entity"];

Literate [description = "Core implementation"];

AttributeSection [description = "Property group"];

Attribute [description = "Individual property"];

Class [description = "Schema definition"];



}

}

@enduml
```



Russian UML

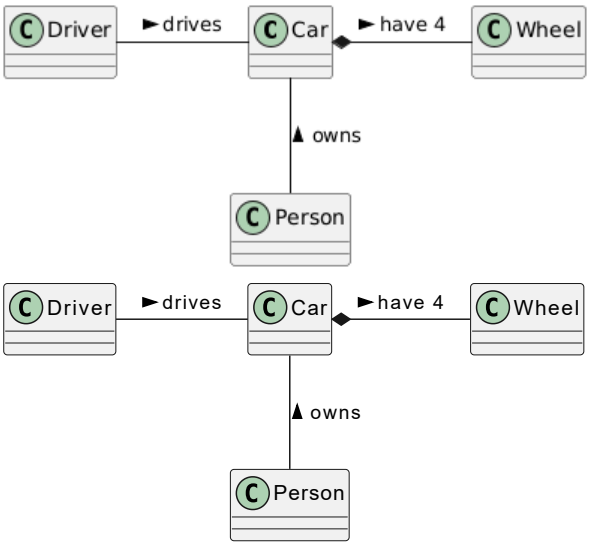
```
@startuml
'hide empty description
'!pragma layout elk
skinparam rectangleBorderThickness 1
skinparam defaultTextAlignment center
skinparam lifelineStrategy solid
skinparam monochrome false
skinparam style strictuml
hide empty members
skinparam Linetype ortho

rectangle "Базовые модули" as base {

class "Базовые объекты" as baseobjects
class "Делопроизводство\n4.5" as takeoffice
class "Управление\nпроцессами" as workflow
class "Windows-клиент" as windowsclient

class "Управление\nдокументами" as documentmanagement
class "Конструктор\nсогласований" as approvaldesigner

class "Платформа" as platform
class "Служба\n фоновых операций" as worker

}

platform <-- baseobjects
platform <-- workflow
platform <-- takeoffice
platform <-- windowsclient
platform <-- documentmanagement
platform <-- approvaldesigner

windowsclient -up-> approvaldesigner
windowsclient -up-> documentmanagement
windowsclient -up-> baseobjects
windowsclient -up-> takeoffice
windowsclient -up-> workflow

worker <-- approvaldesigner
worker <-- baseobjects
@enduml
```




Базовые модули
Платформа
Служба фоновых операций
Управление процессами
Управление документами
Делопроизводство 4.5
Базовые объекты
Конструктор согласований
Windows-клиент

Car diagram

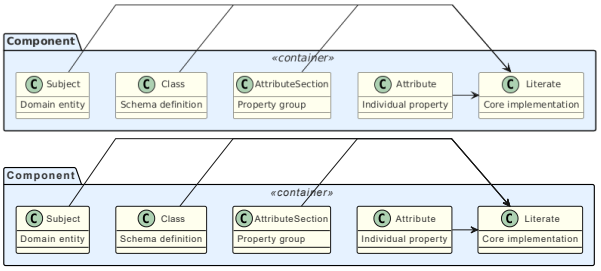
```
@startuml
class Car

Driver - Car : drives >
Car *- Wheel : have 4 >
Car -- Person : < owns
@enduml
```



Fancy Plant UML

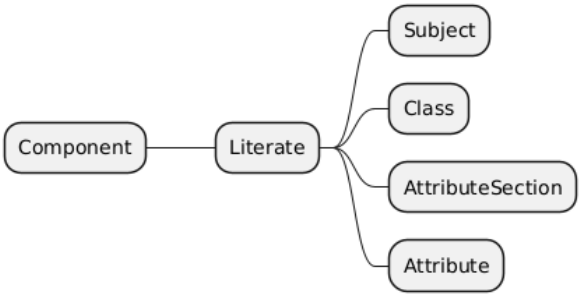
```
@startuml
' Configure the modern style approach with CSS
' Try polyline instead of ortho
skinparam linetype polyline

<style>
/* Global settings */
diagram {
backgroundColor: white;
}

/* Class styling */
class {
BackgroundColor: #FFFFEE;
BorderColor: #333333;
BorderThickness: 1;
BorderRadius: 8;
FontColor: #333333;
FontSize: 12;
}

/* Arrow styling */
arrow {
Color: #333333;
Thickness: 1.5;
}

/* Package styling */
package {
BackgroundColor: #E6F2FF;
BorderColor: #336699;
BorderThickness: 3;
FontColor: #333333;
}

/* Custom style for Component */
.container {
BackgroundColor: #E6F2FF;
BorderColor: #336699;
BorderThickness: 3;
BorderRadius: 10;
}
</style>

package "Component" <<container>> {
class Literate {
Core implementation
}

class Subject {
Domain entity
}
class Class {
Schema definition
}
class AttributeSection {
Property group
}
class Attribute {
Individual property
```

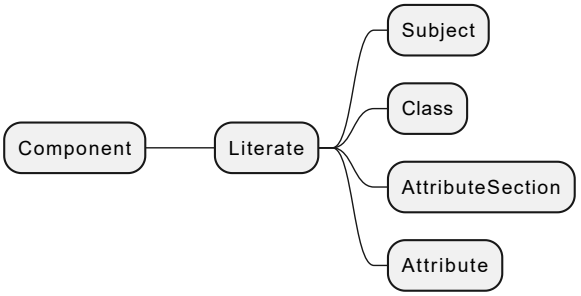
```
}

' Relationships
Subject -> Literate
Class -> Literate
AttributeSection -> Literate
Attribute -> Literate
}

@enduml
```



**Mind Map**
PlanUML

```
@startmindmap
* Component
** Literate
*** Subject
*** Class
*** AttributeSection
*** Attribute
@endmindmap
@startuml
!include <C4/C4_Component>

Person(user, "User")
Container_Boundary(component, "Component") {
Component(literate, "Literate", "Core implementation")
Component(subject, "Subject", "Domain entity")
Component(class, "Class", "Schema definition")
Component(attributeSection, "AttributeSection", "Property group")
Component(attribute, "Attribute", "Individual property")
}

Rel(subject, literate, "extends")
Rel(class, literate, "extends")
Rel(attributeSection, literate, "extends")
Rel(attribute, literate, "extends")
@enduml

@startjson
{
"Component": {
"Literate": {"description": "Core implementation"},
"Subject": {"description": "Domain entity", "extends": "Literate"},
"Class": {"description": "Schema definition", "extends": "Literate"},
"AttributeSection": {"description": "Property group", "extends": "Literate"},
"Attribute": {"description": "Individual property", "extends": "Literate"}
}
}
@endjson
```




Component
Literate
Subject
Class
AttributeSection
Attribute

JSON for Components

```
@startjson
{
"Component": {
"Literate": ["description", "Core implementation"],
"Subject": {"description": "Domain entity", "extends": "Literate"},
"Class": {"description": "Schema definition", "extends": "Literate"},
"AttributeSection": {"description": "Property group", "extends": "Literate"},
"Attribute": {"description": "Individual property", "extends": "Literate"}
}
}
@endjson
```