FIRST PAGE LEFT LEFT BLANK

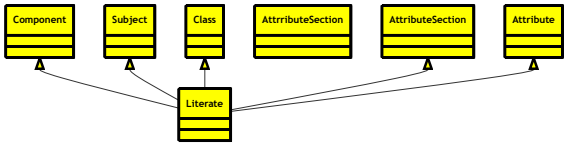
This is my first Mermaid test

Mermaid Class Diagram

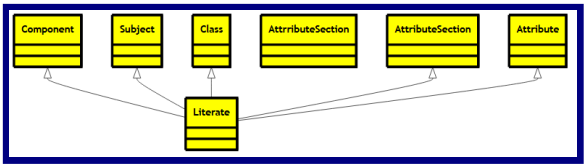
Mermaid Diagram - Inert

```
classDiagram
class Component
class Literate
class Subject
class Class
class Attrribute Section
class Attribute

Component  <|-- Literate
Subject  <|-- Literate
Class  <|-- Literate
AttributeSection  <|-- Literate
Attribute  <|-- Literate

classDef default fill:yellow,stroke:#000, color:black, stroke-width:4px
```

Mermaid Diagram - Live!



Mermaid Diagram - PNG for mermaid

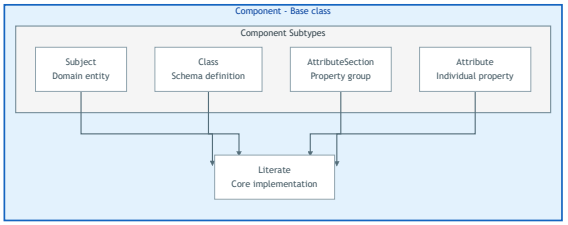


Mermaid Flowchart

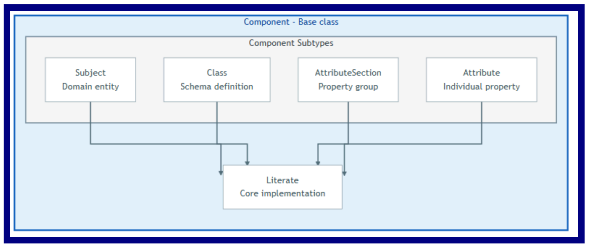
Mermaid Diagram - Inert

```
%%{init: {
"flowchart": {
"curve": "stepAfter",
"useMaxWidth": true
}
}}%%

flowchart TB
subgraph Component["Component - Base class"]
direction TB

Literate["Literate<br>Core implementation"]

subgraph Subtypes["Component Subtypes"]
direction LR
Subject["Subject<br>Domain entity"]
Class["Class<br>Schema definition"]
AttributeSection["AttributeSection<br>Property group"]
Attribute["Attribute<br>Individual property"]
end

Subject ==> Literate
Class ==> Literate
AttributeSection ==> Literate
Attribute ==> Literate
end

%% Styling with border-radius only
classDef container fill:#e3f2fd,stroke:#1565c0,stroke-width:3px,color:#0d47a1,border-radius:10px
classDef subcontainer fill:#f5f5f5,stroke:#78909c,stroke-width:2px,color:#37474f,border-radius:8px
classDef default fill:white,stroke:#90a4ae,stroke-width:1px,color:#455a64,border-radius:5px

class Component container
class Subtypes subcontainer

%% Edge styling
linkStyle default stroke:#546e7a,stroke-width:2px, border-radius: 20px
```

Mermaid Diagram - Live!



Mermaid Diagram - PNG for mermaid

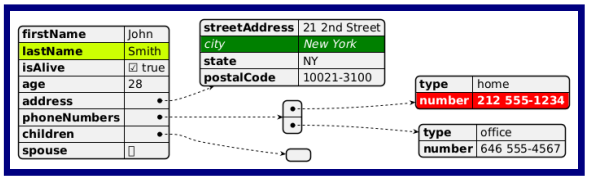

Component - Base class
Component Subtypes
Subject
Domain entity
Class
Schema definition
AttributeSection
Property group
Attribute
Individual property
Literate
Core implementation

Plant UML jsondata

PlantUML Diagram - Inert

```
@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
```

PlantUML Diagram - PNG for puml

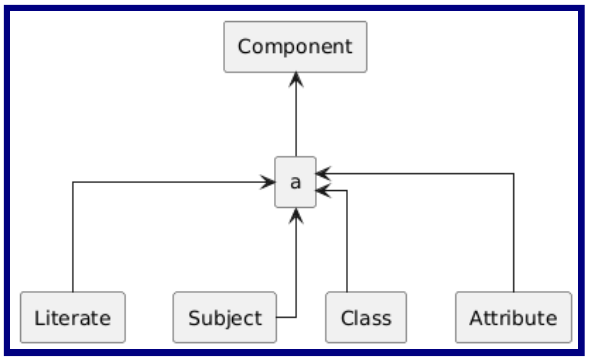


Plant UML UML

PlantUML Diagram - Inert

```
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
```

PlantUML Diagram - PNG for puml

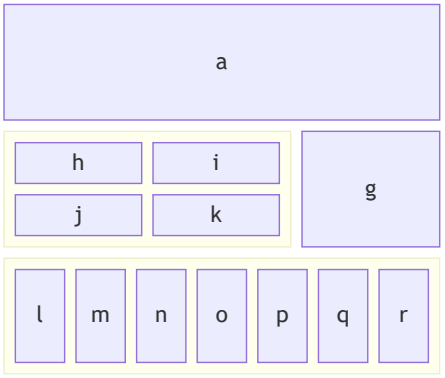


Mermaid block diagram

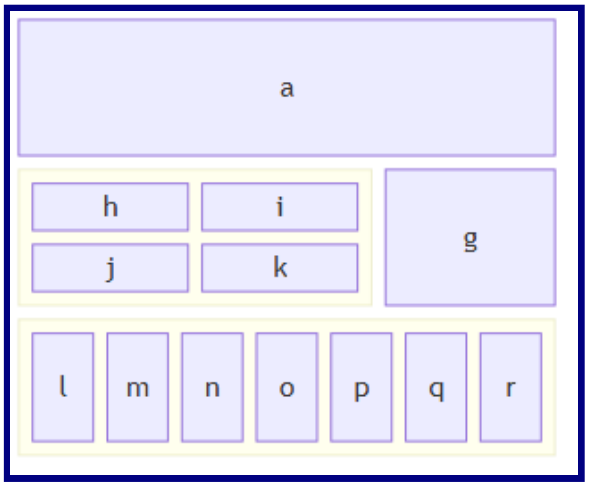
Mermaid Diagram - Inert

```
block-beta
columns 3
a:3
block:group1:2
columns 2
h i j k
end
g
block:group2:3
%% columns auto (default)
l m n o p q r
end
```

Mermaid Diagram - Live!



Mermaid Diagram - PNG for mermaid

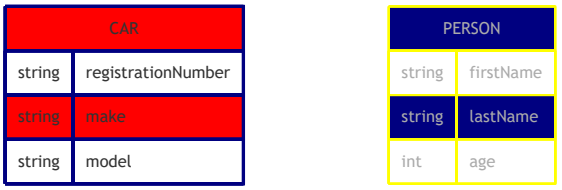


Mermaid ER Diagram

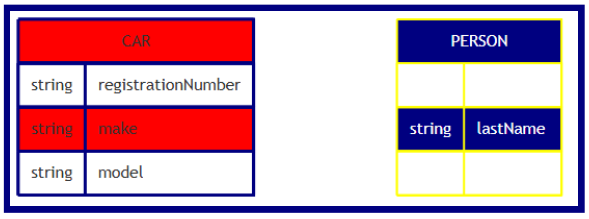
Mermaid Diagram - Inert

```
erDiagram
CAR {
string registrationNumber
string make
string model
}
PERSON {
string firstName
string lastName
int age
}
style CAR fill: red,stroke:navy,stroke-width:3px
style PERSON color: white, fill: navy,stroke:yellow ,stroke-width:2px
```

Mermaid Diagram - Live!



Mermaid Diagram - PNG for mermaid

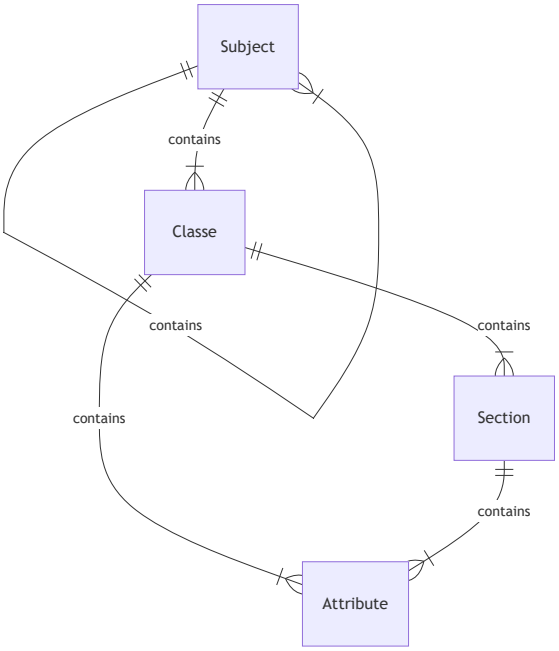


Mermaid ER Diagram

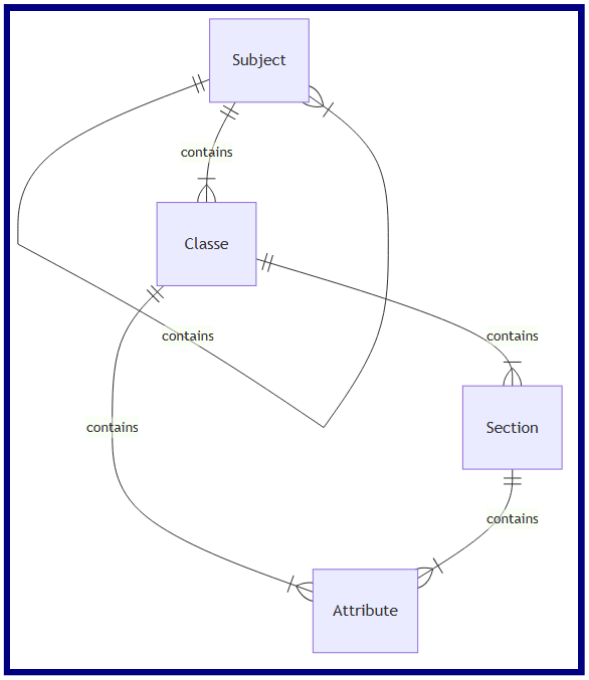
Mermaid Diagram - Inert

```
erDiagram

class Subject Component

class Section Component

class Attribute Component

class Classe Component

Subject ||--|{ Subject : contains

Subject ||--|{ Classe : contains


Classe ||--|{ Section : contains

Classe ||--|{ Attribute : contains

Section ||--|{ Attribute : contains
```

Mermaid Diagram - Live!



Mermaid Diagram - PNG for mermaid


Subject
contains
Classe
contains
contains
contains
Section
contains
Attribute

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
```

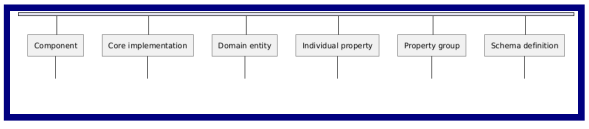
	eFormat	Description
0	E-Book	'Kindle or Apple books - etc'
1	PDF	formatted for printing and direct delivery

Plant UML UML

PlantUML Diagram - Inert

```
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"];

}
}
```

PlantUML Diagram - PNG for puml

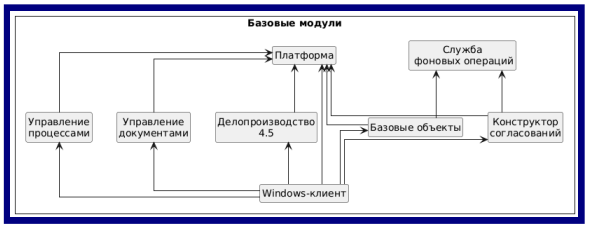


Russian UML

PlantUML Diagram - Inert

```
'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
```

PlantUML Diagram - PNG for puml



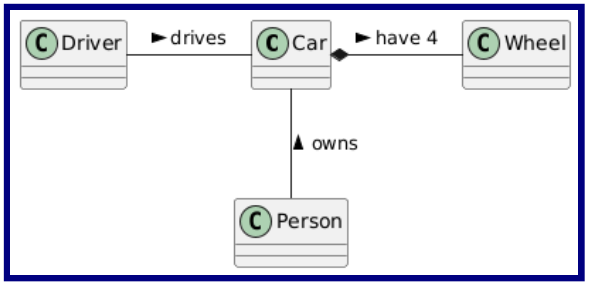
Car diagram

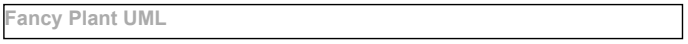
PlantUML Diagram - Inert

class Car

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

PlantUML Diagram - PNG for puml

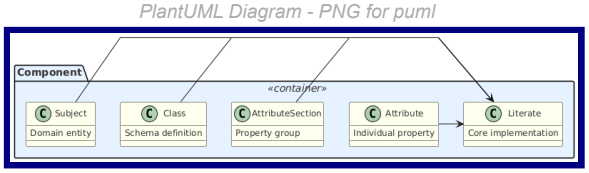



Fancy Plant UML

PlantUML Diagram - Inert

```
' 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
}
```

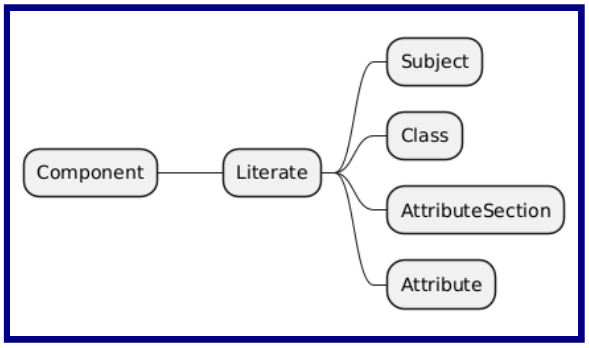


Mind Map
PlanUML

PlantUML Diagram - Inert

```
@startmindmap
* Component
** Literate
*** Subject
*** Class
*** AttributeSection
*** Attribute
@endmindmap

!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")


@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
```

PlantUML Diagram - PNG for puml



JSON for Components

PlantUML Diagram - Inert

```
@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
```

PlantUML Diagram - PNG for puml