Mermaid Class Diagram

```
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
```

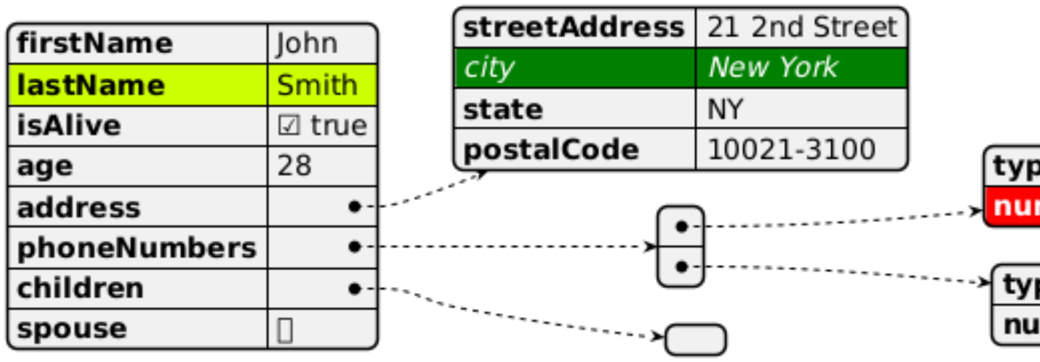
Mermaid Flowchart

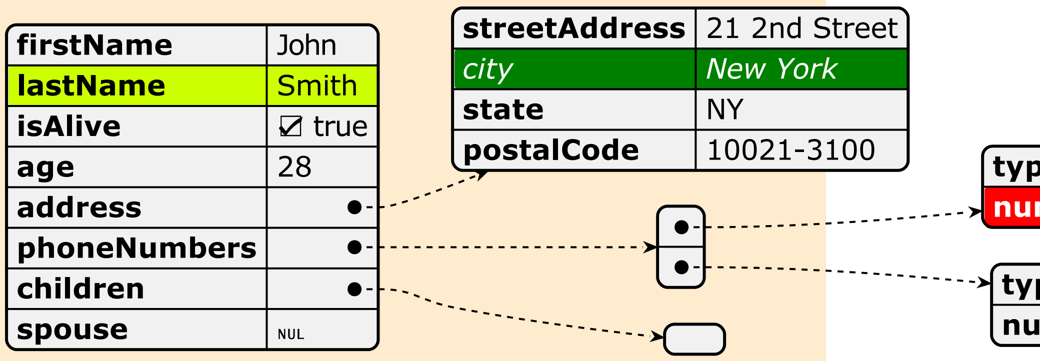
```
%%{init: {
"flowchart": {
"curve": "stepAfter",
"useMaxWidth": true
}
}}%%

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:3
classDef subcontainer fill:#f5f5f5,stroke:#78909c,stroke-widt
classDef default fill:white,stroke:#90a4ae,stroke-width:1px,c

class Component container
class Subtypes subcontainer

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

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
```

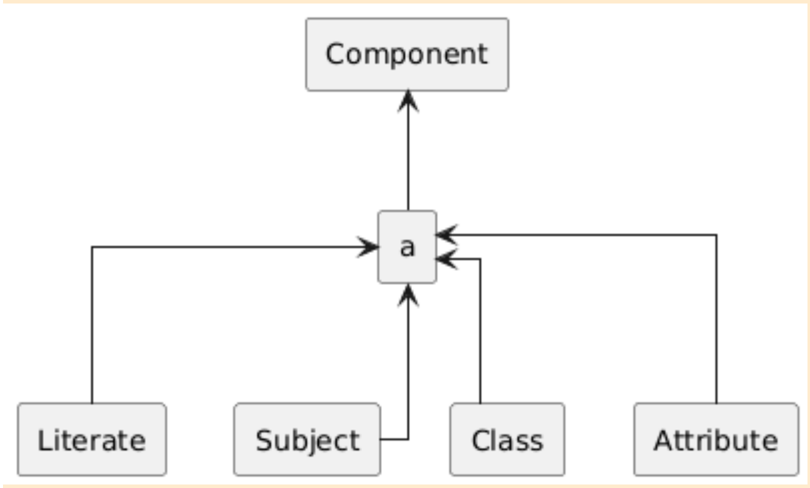


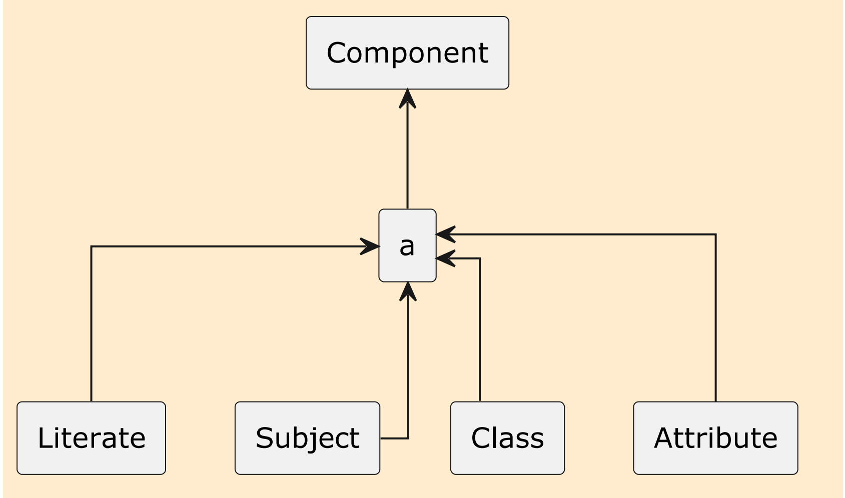


Plant UML UML

```
@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
```





```
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 ER Diagram

```
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-w
```

Mermaid ER Diagram

```
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
```

Captioned figure figure img src FMKNYID Front IM jpg alt Trulli style width figcaption Fig

Trulli, Puglia, Italy. And the same figure with figure/caption markup

+ +
My Non-Drivers License
+

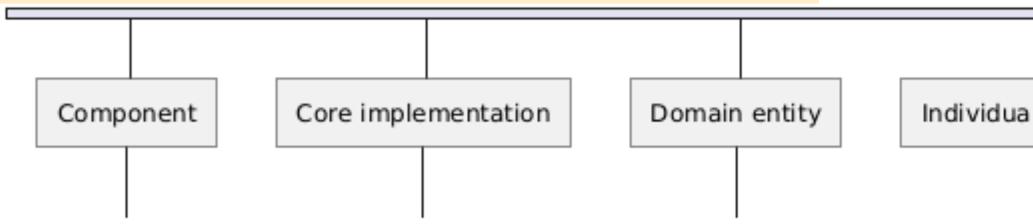
List of Codes

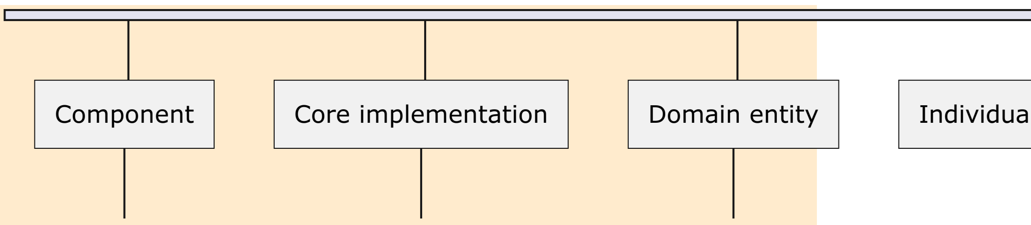
```
eFormat, Description
E-Book, 'Kindle or Apple books - etc'
PDF, formatted for printing and direct delivery
```

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

UML

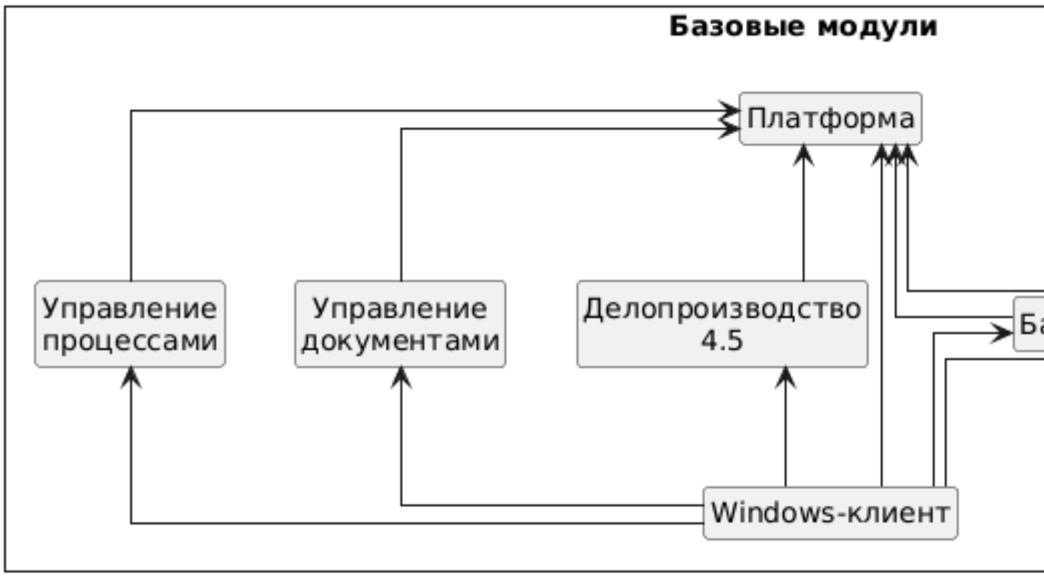
```
@startuml

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
```

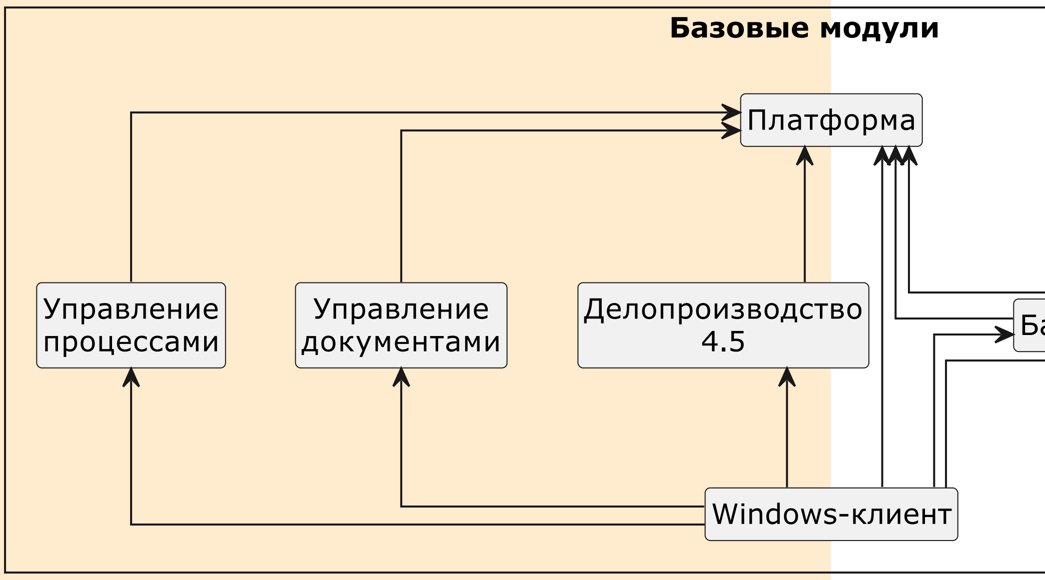




Russian UML

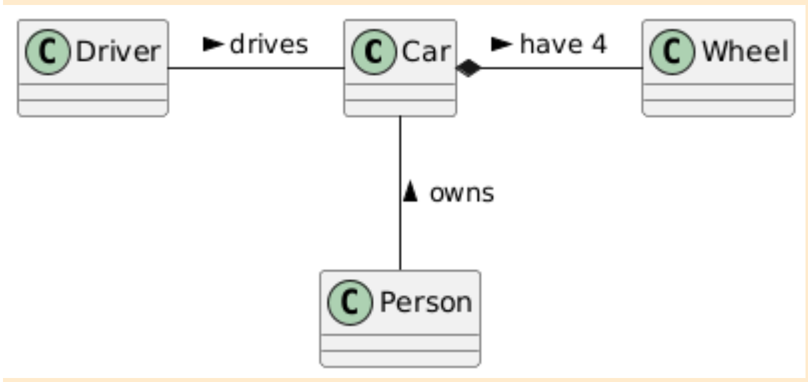
```
@startuml
'hide empty description
'!pragma layout elk
skinparam rectangleBorderThickness 1
skinparam defaultTextAlignment center
skinparam lifelineStrategy solid
skinparam monochrome false
skinparam style strictuml
hide empty members
skinparam Linetype ortho

rectangle "Ð‘Ð°Ð·Ð¾Ð²Ñ‹Ðµ Ð¼Ð¾Ð´ÑƒÐ»Ð¸" as base {

class "Ð‘Ð°Ð·Ð¾Ð²Ñ‹Ðµ Ð¾Ð±ÑŠÐµÐºÑ‚Ñ‹" as baseobjects
class "Ð”ÐµÐ»Ð¾Ð¿Ñ€Ð¾Ð¸Ð·Ð²Ð¾Ð´Ñ�Ñ‚Ð²Ð¾\n4.5" as takeoffice
class "Ð£Ð¿Ñ€Ð°Ð²Ð»ÐµÐ½Ð¸Ðµ\nÐ¿Ñ€Ð¾Ñ†ÐµÑ�Ñ�Ð°Ð¼Ð¸" as workfl
class "Windows-ÐºÐ»Ð¸ÐµÐ½Ñ‚" as windowsclient

class "Ð£Ð¿Ñ€Ð°Ð²Ð»ÐµÐ½Ð¸Ðµ\nÐ´Ð¾ÐºÑƒÐ¼ÐµÐ½Ñ‚Ð°Ð¼Ð¸" as docum
class "ÐšÐ¾Ð½Ñ�Ñ‚Ñ€ÑƒÐºÑ‚Ð¾Ñ€\nÑ�Ð¾Ð³Ð»Ð°Ñ�Ð¾Ð²Ð°Ð½Ð¸Ð¹" as

class "ÐŸÐ»Ð°Ñ‚Ñ„Ð¾Ñ€Ð¼Ð°" as platform
class "Ð¡Ð»ÑƒÐ¶Ð±Ð°\n Ñ„Ð¾Ð½Ð¾Ð²Ñ‹Ñ… Ð¾Ð¿ÐµÑ€Ð°Ñ†Ð¸Ð¹" as wor

}

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
```

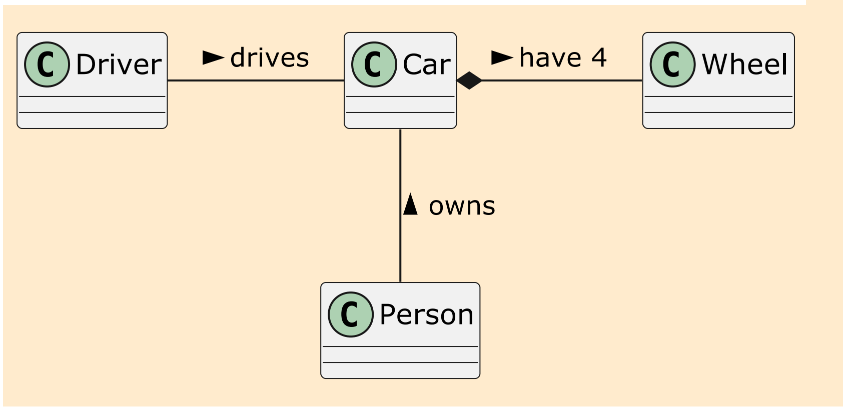




Car diagram

```
@startuml
class Car

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





and a dummy section