# samenvatting

### Voorbeeld

### Zie ook

* ZIE: SAMENVATTING X

C:\Users\marty\OneDrive\Bureaublad\repository voor novi\puml

### Te gebruiken bij:

-

### Eigenschappen:

-

### Info

#### instaleren

je kunt PlantUML ook gebruiken in IntelliJ IDEA, een populaire geïntegreerde ontwikkelomgeving (IDE) voor Java-ontwikkeling. IntelliJ IDEA biedt ondersteuning voor PlantUML via de PlantUML-integratieplug-in. Hier is hoe je PlantUML in IntelliJ IDEA kunt gebruiken:

1. Installeer de PlantUML-integratieplug-in: Open IntelliJ IDEA en ga naar "File" -> "Settings" (of "Preferences" op een Mac) om de instellingen te openen. Zoek naar "Plugins" en klik op "Marketplace". Zoek naar "PlantUML Integration" en installeer de plug-in.
2. Maak een nieuw PlantUML-bestand: Maak een nieuw bestand in je IntelliJ IDEA-project met de extensie ".puml" of ".pu" om je PlantUML-code op te slaan.
3. Schrijf je PlantUML-code: Typ je PlantUML-code in het nieuwe bestand. IntelliJ IDEA zal de syntax highlighting en autocompletie bieden om je te helpen bij het schrijven van correcte PlantUML-code.
4. Genereer het diagram: Klik met de rechtermuisknop in het PlantUML-bestand en selecteer "Diagrams" -> "Show Diagram" om het PlantUML-diagram te genereren. IntelliJ IDEA zal het diagram weergeven in een apart venster.
5. Blijf het diagram bijwerken: Je kunt je PlantUML-code blijven bewerken en IntelliJ IDEA zal automatisch het bijbehorende diagram bijwerken. Dit maakt het gemakkelijk om wijzigingen aan te brengen en de impact ervan direct te zien.

Met de PlantUML-integratieplug-in in IntelliJ IDEA kun je PlantUML-diagrammen maken en bijwerken binnen de vertrouwde ontwikkelomgeving. Het biedt een handige workflow voor ontwikkelaars die al met IntelliJ IDEA werken.

@startuml

#### voorbeeld

class ChatApp {

- chatRooms: List<ChatRoom>

- users: List<User>

+ addUser(user: User): void

+ createChatRoom(name: String): ChatRoom

}

class ChatRoom {

- name: String

- messages: List<Message>

- participants: List<User>

+ addMessage(message: Message): void

+ addParticipant(user: User): void

}

class User {

- username: String

- password: String

- email: String

+ sendMessage(chatRoom: ChatRoom, message: String): void

+ joinChatRoom(chatRoom: ChatRoom): void

}

class Message {

- content: String

- sender: User

- timestamp: Date

+ getContent(): String

+ getSender(): User

+ getTimestamp(): Date

}

ChatApp "1" --> "\*" ChatRoom

ChatApp "1" --> "\*" User

ChatRoom "\*" --> "\*" User

ChatRoom "\*" --> "\*" Message

Message "1" --> "1" User

@enduml

Natuurlijk! Hier is een uitleg van de code die in een PlantUML-bestand wordt gebruikt:

Klassedefinitie:

kotlin

Copy code

class ClassName {

// attributen

- privateAttribute: Type

+ publicAttribute: Type

// methoden

- privateMethod(): ReturnType

+ publicMethod(): ReturnType

}

class ClassName definieert een klasse met de naam "ClassName".

- privateAttribute: Type definieert een privaat attribuut met de naam "privateAttribute" en het gegevenstype "Type".

+ publicAttribute: Type definieert een publiek attribuut met de naam "publicAttribute" en het gegevenstype "Type".

- privateMethod(): ReturnType definieert een privé methode met de naam "privateMethod" en de retourtype "ReturnType".

+ publicMethod(): ReturnType definieert een publieke methode met de naam "publicMethod" en de retourtype "ReturnType".

Relaties:

kotlin

Copy code

class ClassA {

// ...

}

class ClassB {

// ...

}

ClassA --> ClassB // Associatie

ClassA --> "\*" ClassB // Compositie (1 ClassA heeft meerdere ClassB-objecten)

ClassA --> "1" ClassB // Aggregatie (1 ClassA heeft één ClassB-object)

ClassA --|> ClassB // Erfenis (ClassA erft van ClassB)

ClassA --> ClassB definieert een associatierelatie tussen de klassen ClassA en ClassB.

ClassA --> "\*" ClassB definieert een compositierelatie waarbij ClassA meerdere objecten van ClassB bezit (1-n relatie).

ClassA --> "1" ClassB definieert een aggregatierelatie waarbij ClassA één object van ClassB bezit (1-1 relatie).

ClassA --|> ClassB definieert een erfenisrelatie waarbij ClassA van ClassB erft (is een relatie).

Notaties voor attributen en methoden:

- privateAttribute: Type geeft een privaat attribuut aan met de naam "privateAttribute" en het gegevenstype "Type".

# protectedAttribute: Type geeft een beschermd attribuut aan met de naam "protectedAttribute" en het gegevenstype "Type".

+ publicAttribute: Type geeft een publiek attribuut aan met de naam "publicAttribute" en het gegevenstype "Type".

- privateMethod(): ReturnType geeft een privé methode aan met de naam "privateMethod" en het retourtype "ReturnType".

# protectedMethod(): ReturnType geeft een beschermd methode aan met de naam "protectedMethod" en het retourtype "ReturnType".

+ publicMethod(): ReturnType geeft een publieke methode aan met de naam "publicMethod" en het retourtype "ReturnType".

De PlantUML-code maakt gebruik van een specifieke syntax om UML-diagrammen te definiëren. Met behulp van deze code kun je klassen, attributen, methoden en relaties tussen klassen specificeren.

#### Thema veranderen

Om het thema van je klassediagram in IntelliJ IDEA te wijzigen, kun je de volgende stappen volgen:

Open IntelliJ IDEA en zorg ervoor dat je project geopend is.

1. Zorg ervoor dat je de PlantUML-integratie hebt geïnstalleerd en geactiveerd in IntelliJ IDEA. Als je dit nog niet hebt gedaan, kun je de plugin installeren via het "Settings" (of "Voorkeuren")-menu door naar "Plugins" te gaan en te zoeken naar "PlantUML Integration".
2. Open het PlantUML-bestand waarin je klassediagram is gedefinieerd.
3. Klik met de rechtermuisknop ergens in het PlantUML-bestand en selecteer "Diagrams" (of "Diagrammen") in het contextmenu. Kies vervolgens "Show Diagram" (of "Diagram weergeven") om het klassediagram te openen.
4. In het geopende klassediagramvenster zie je een toolbar bovenaan. Zoek naar het pictogram met een penseel (of een ander themagerelateerd pictogram) dat meestal wordt gebruikt om het thema te wijzigen.
5. Klik op het themapictogram en je zou een lijst met beschikbare thema's moeten zien. Kies het gewenste thema, bijvoorbeeld "Light" (licht) of "Dark" (donker).
6. Het klassediagram wordt automatisch bijgewerkt met het geselecteerde thema.

Merk op dat het beschikbare thema-aanbod afhankelijk kan zijn van de geïnstalleerde versie van IntelliJ IDEA en de gebruikte PlantUML-integratie. Zorg ervoor dat je de juiste plugins en updates hebt geïnstalleerd om de themawijzigingsopties te hebben.