0.1 Die Nutzung des LSP

0.1.1 Installationsmöglichkeiten

Der LSP-Server benötigt keine zusätzlichen Strukturen und kann direkt als Datei ausgeführt werden. Deshalb ist die normale Vorgehensweise das Ablegen des LSP-Servers in einem Zentralen Server und das Hinzufügen des Pfades zu dem Umgebungsvariablen. Im Beispiel wurde die Datei zu 'dmflsp' umbenannt, damit der Name genauer das Programm beschreibt und nicht mit anderen LSP-Server kollidieren kann.

```
alexander@alexander-Ubuntu:~$ dmflsp --help
Usage of dmflsp:
  -characterInMarkdownLinks
        activates MarkdownLinks where the Character is transmitted with the schema #L<line>.<character>
  -disableSingleLineCommentsFolding
        do not generate folding ranges for single line comments
  -disabledLog
        Disable logging
  -port int
        switches communication to specified tcp port (default -1)
```

Abbildung 1: Aufruf des CLI des LSP-Servers

Es gibt Editoren die eine native Anbindung eines LSP-Servers ermöglichen. TODO Zed Bespiel

Intellij

Intellij unterstützt nur wenige LSP-Funktionen ohne zusätzliche Plugins. Mit von 'lsp4ij' können LSP-Server direkt in der Oberfläche konfiguriert werden.

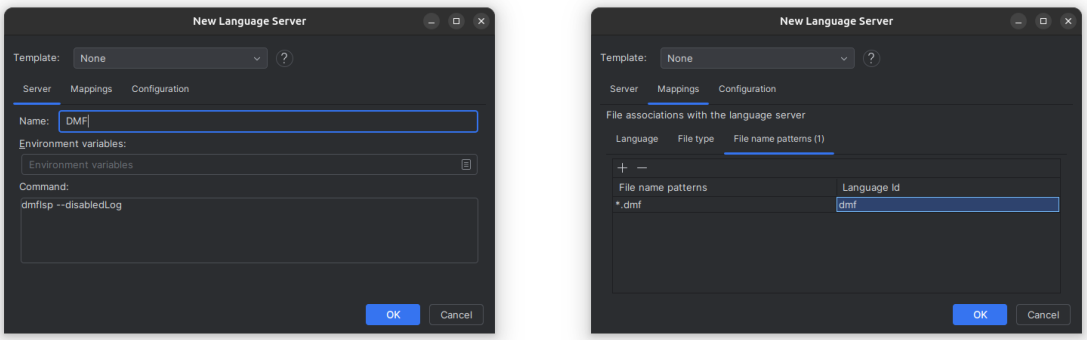

Abbildung 2: lsp4ij Konfiguration

Um diese Konfiguration automatisch anzulegen und den Dateien ein passendes Icon zu geben, kann das Intellij-Plugin für das DMF verwendet werden. Es enthält die verschiedenen Versionen des Servers und kann sie automatisch an den konfigurierten Pfad ablegen.

Der Pfad zum LSP-Server kann entweder in den Einstellungen des Plugins oder über die Umgebungsvariable ‘DMF_LSP’ konfiguriert werden.

Visual Studio Code

Um den LSP-Server in Visual Studio Code nutzen zu können, wird die Erweiterung für das DMF benötigt. Dieses enthält die Logik zum Verbinden zum Server und die verschiedenen Versionen. Die benötigte Version kann direkt ausgeführt werden und benötigt keine zusätzliche Konfiguration.
Im Gegensatz zu den bisherigen Konfigurationen nutzt die Visual Studio Code Erweiterung eine TCP-Verbindung.

0.1.2 Funktionen im Editor

Während des Bearbeitens der Modelle können die Funktionen des LSP-Servers genutzt werden. In diesem Abschnitt werden die Funktionen präsentiert.

Diagnosen

Wenn der Fehler in der Datei existieren werden diese automatisch im Editor markiert. Es wird auch eine Beschreibung und die detaillierte Beschreibung, welche auch im Generator ausgegeben wird, übertragen.
Die Darstellung wird von der IDE übernommen. In Intellij und Visual Studio Code wird die Hover-Beschreibung zusätzlich angezeigt.

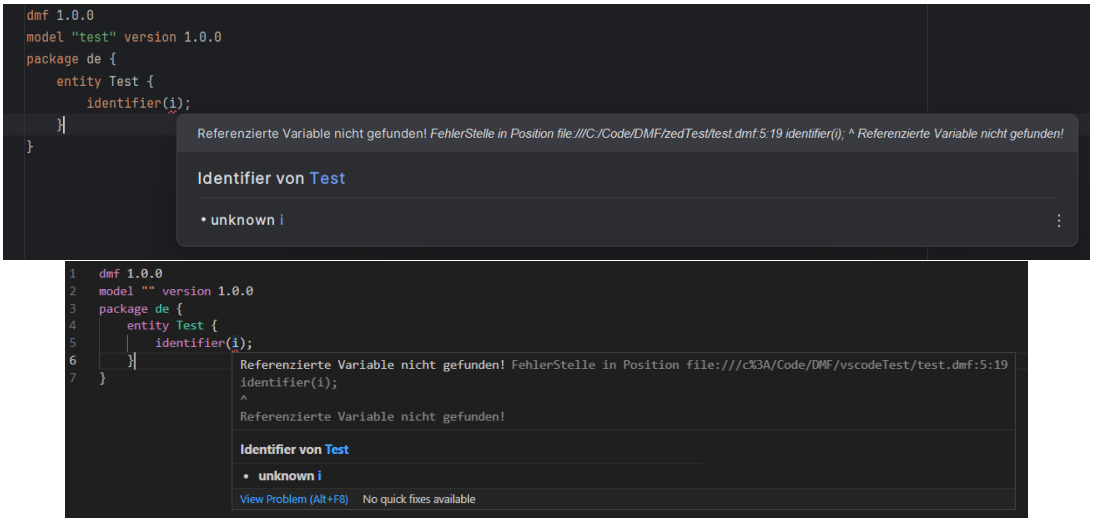

Abbildung 3: Beispiele aus Intellij und Visual Studio Code

Hover-Beschreibungen

Um Informationen über ein Element bereitzustellen, kann mit dem Mauszeiger über einem Element gehovered werden. Für alle PackageElemente, EntityIdentifier, Argumente, Referenzen, MultiReferenzen und Kommentare können Beschreibungen angezeigt werden. Mithilfe von Links in den Beschreibungen kann direkt zum erwähnten Element navigiert werden.

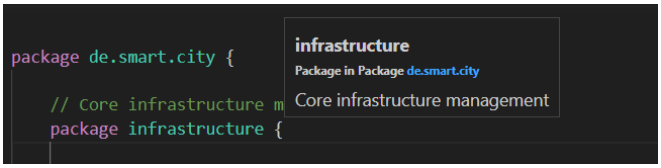

Abbildung 4: Beschreibung eines Package Elements

Bei PackageElementen enthält die Beschreibung den Kommentar des Elements sowie das Package, in dem es liegt.

Abbildung 5: Beschreibung einer Enum Konstante ohne Kommentar

Die Beschreibung von Enum Konstanten enthält das Enum, falls vorhanden den Kommentar und die Argumente des Enum mit den Werten der Konstante. Der erste Wert ist der Index, welcher in der Datenbank gespeichert wird.

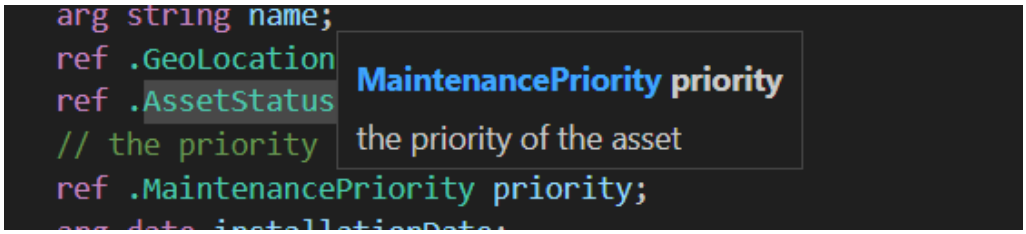

Abbildung 6: Beschreibung einer Referenz

Referenzen enthalten den Kommentar sowie den Typen und Namen der Variable. Argumente und MultiReferenzen verhalten sich gleich.

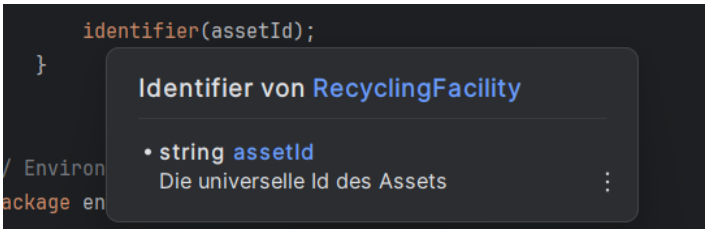

Abbildung 7: Beschreibung eines Entity Identifiers

Bei einem Entity Identifier werden die referenzierten Variablen ihren Kommentaren angezeigt.