As can be gleaned from the code, if the extension is null, the private attribute of the class of type String called extension is not changed and the default value of "vdmpp" is used.

The convert method is then called, with a file containing the path to the new directory which contain the generated VDM files.

```
uml2vdm.convert(new File(iFile.getProject().getLocation().
    toFile(), "uml_import"));
```

The New Uml2Vdm Handler* ??

6.2 Integrating the UML transformations into VS Code

Several of the plugins for the Eclipse version such as Java Code Generation and FMU Export/Import are already implemented on the VDM VSCode extension. They are implemented as commands that the user can execute using a drop-down menu when right clicking the workspace folder. The UML transformations are implemented ~~likewise~~ using a handler, UMLHandler.ts, which repurposes the code from the ? in a similar manner

In the extension.ts, the UML handler is imported and initialized together with the other handlers:

```
import { UMLHandler } from "./handlers/UMLHandler";

...

context.subscriptions.push(new UMLHandler(clientManager));
```

To allow the transformations to be used as commands in VS Code, changes have to be made to package.json. Specifically, under commands, commandPalette, menu and configurations, the UML to VDM function is added as vsccode.uml.import and the VDM to UML function is added as vscode.uml.export. Eg. the following code snippet shows how they are added under commands.