```
ResourceSet resourceSet = new ResourceSetImpl();
resourceSet.getResourceFactoryRegistry()
   .getExtensionToFactoryMap()
   .put("*",new XMIResourceFactoryImpl());
Resource resource = resourceSet
     .createResource(uri.appendFileExtension(UMLResource.
    FILE_EXTENSION));

resource.getContents().add(modelWorkingCopy);
```

Analysis of the Old Uml2Vdm Handler

`Uml2VdmCommand` is the name of the old handler for the transformation going in the direction of VDM to UML. The main purpose of the old Uml2Vdm handler, which the new handler mimics, is the acquire a path to the UML file, create an instance of the `Uml2Vdm` class and instantiate it with the contents of the UML file. Overall, the handler takes fever steps to facilitate the conversion, since only a single UML file needs to be handled as opposed to multiple VDM files with relations to each other.

In this direction the old handler depends on Eclipse specific resource classes as well. The classes in question are the `iFile` class which encapsulate the Eclipse notion of a file and `IVdmProject` which was also in the old Vdm2Uml handler.

To perform the conversion from UML to VDM, the `Uml2Vdm` class needs to be instantiated with the path to the UML file, and information about what dialect the to be generated VDM files should be in.

The path to the UML file is easily extracted from the `iFile` instance:

```
java.net.URI absolutePath = iFile.getLocationURI();
   final URI uri = URI.createFileURI(absolutePath.getPath());
```

To gain information about the dialect of the VDM model, an instance of the `IVdmProject` is casted from `iFile`, and the following code is executed: