**Hide Excel Sheet**

**Description**

We often have to download files in excel format, sometimes the excel file we download or the template we use to download the file in excel format has some worksheets attached to it. In some cases, according to the business requirement, the file which must be downloaded doesn’t want the other worksheets attached to it in the same excel file.

Now, to tackle this issue, I have created a component HideExcelSheets which will help you to hide the other worksheets easily.

**Typical usage scenario**

This component is used to hide excel sheets from an excel file. For example, suppose there is a list of data, and you want the download that data in an excel file and that excel file converts the data into a graph, either pie graph, bar graph or any other graph format by applying a template. And suppose you have created 5 worksheets in that template. Now if you don’t want to download 2nd and 3rd worksheets but you want the 1st, 4th and 5th worksheets, you won’t be able to do that easily. Now to tackle this problem we can use

this component.

**PreRequisites**

NA

**Features and limitations**

This component will help in downloading excel files according to your choice of worksheets. With the help of this component, you will save a lot of time which was wasted in doing this from scratch. This can be used in a single template without modifying the template.

**Dependencies**

Mendix Studio Pro 9.12.4.

Excel Export and MxModel Reflection Module from Marketplace.

**Installation**

1. **Download modules from marketplace**

* MxModel Reflection
* Excel Export

1. **Data Import Using API-Implementation**

API to make a REST call to retrieve data of products in a JSON format and display it in the application.

* 1. In your module, create a microflow to retrieve data from a link using API.
  2. For the microflow that will have the logic to make the REST call to retrieve the data is created, we need to create a JSON structure that will indicate how we are going to be receiving the data from the API.
  3. Now, in [JSON Placeholder page](https://jsonplaceholder.typicode.com/), look for /users in resources and open it in a new tab in your browser. Copy the JSON that is displayed and then paste it in the JSON structure content and Format the content.
  4. Now in the created microflow add activity to call Rest services and set the link of data to the location. In the Response tab, choose Import from mapping and provide the variable.
  5. The final microflow will look like this.

**Graphical user interface, text, application, chat or text message

Description automatically generated**

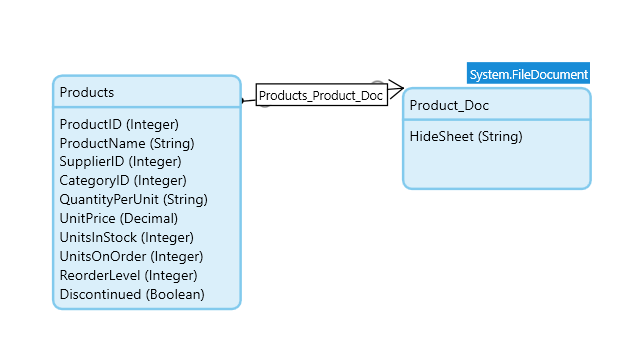
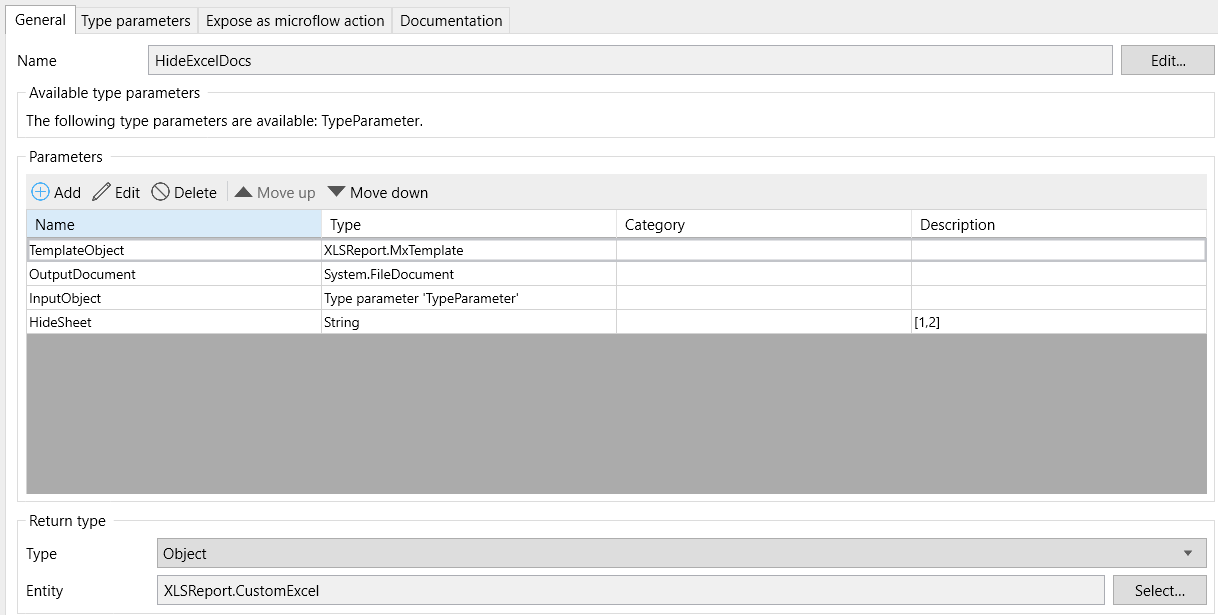
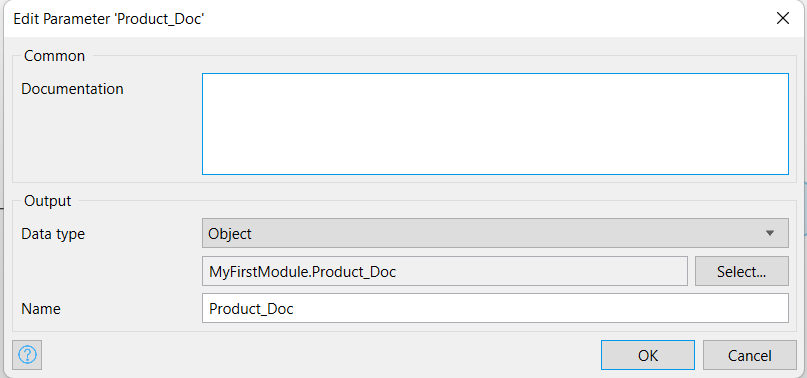
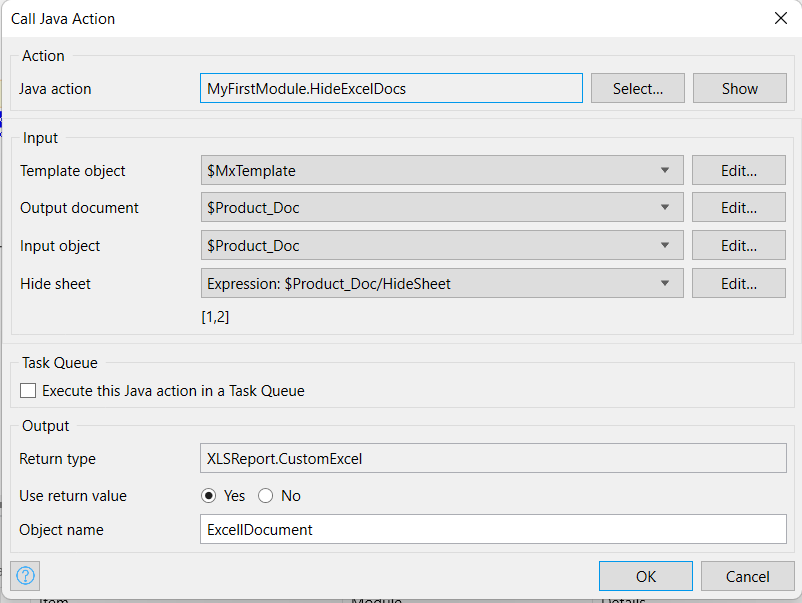
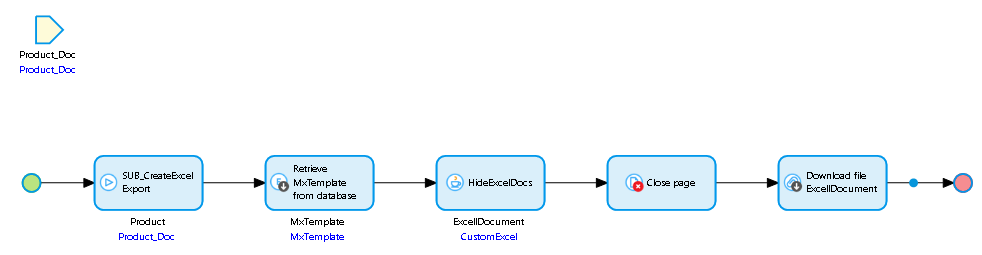
* 1. Go to “Home\_Web” page, add a Data Grid widget in the Layout grid.
  2. Click on the Data Grid and go to Data Source and select microflow, then select the microflow that we created.
  3. Select Yes to automatically fill data and for the server-side paging.
  4. Save files and run locally.
  5. The final page will look like this.

**A screenshot of a computer

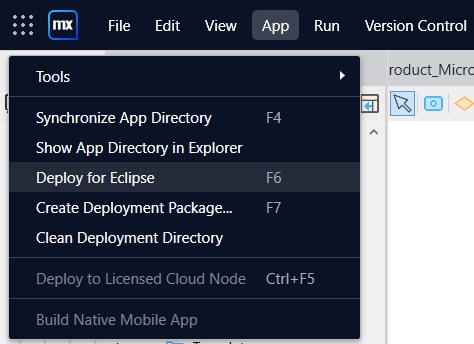
Description automatically generated with medium confidence**

1. **Adding Export button**

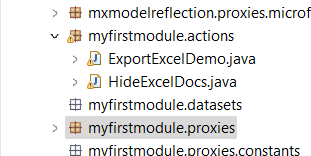
Reference : [Mendix Academy - 7.1 Using the Excel Exporter to Export Data](https://academy.mendix.com/link/modules/136/lectures/1252/7.1-Using-the-Excel-Exporter-to-Export-Data)

* 1. Click on the reference and do as per given. After doing this you need to do some modification.
  2. In the domain model add a attribute HideSheet as a string.
  3. Create a page name it Hidesheet\_Overview
  4. Copy GenerateExcelDoc from Marketplace module > XLSReport > Documents > Java Actions and paste it in the MyFirstModule and name it HideExcelDocs.
  5. Open HideExcelDocs java action and add a parameter HideSheet of type String.
  6. Now open microflow ACT\_Export\_To\_Excel add a parameter of object type Product\_Doc and click ok.
  7. Instead of GenerateExcelDoc java action replace it with the HideExcelDocs. In the output Document select Product\_Doc and in Input Object select Product\_Doc and in the HideSheet click edit and pass an argument : $Product\_Doc/HideSheet and click ok. After doing this it should like the below image
  8. Add a close page after this. Now the modified microflow will look like the below image, if you have followed the steps properly.

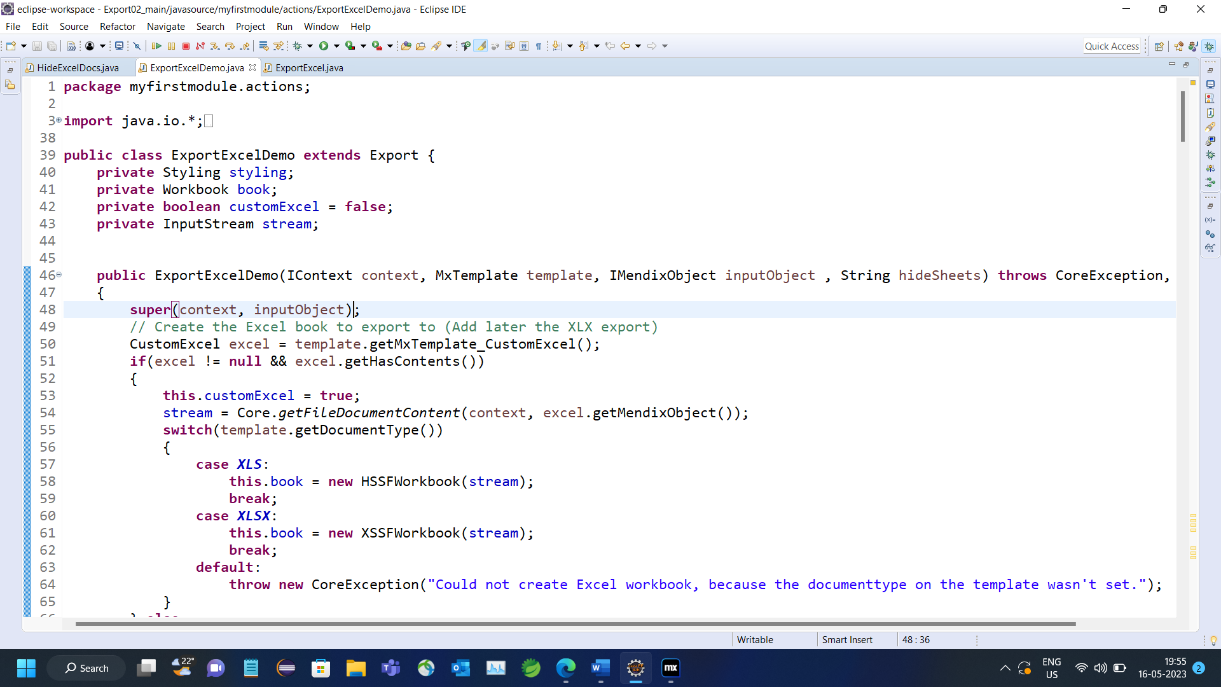
1. Deploy the code to eclipse.
   1. Go to App > Click on Deploy for Eclipse



* 1. Open eclipse and import it. I hope you know this. If you don’t know, you can search on internet you can easily get lot of details on how to import it.
  2. Go to JavaSources> xlsreport.report.export. Copy ExportExcel.java and paste it in myFirstModule.actions. Rename it to ExportExcelDemo.java or any name you want. After that it would look like the below image.



* 1. Go to ExportExcelDemo.java . Change the constructer name to ExportExcelDemo and add a attribute String hideSheets to it. It will look like the below image.



* 1. Now add a piece of code below line 80. You can apply your own code but I have provided the code below :

*this.book.setForceFormulaRecalculation(true);*

*if(hideSheets != null && hideSheets.length() > 0) {*

*String[] hideSheetNumbers = hideSheets.split(",");*

*for (String s: hideSheetNumbers) {*

*//Do your stuff here*

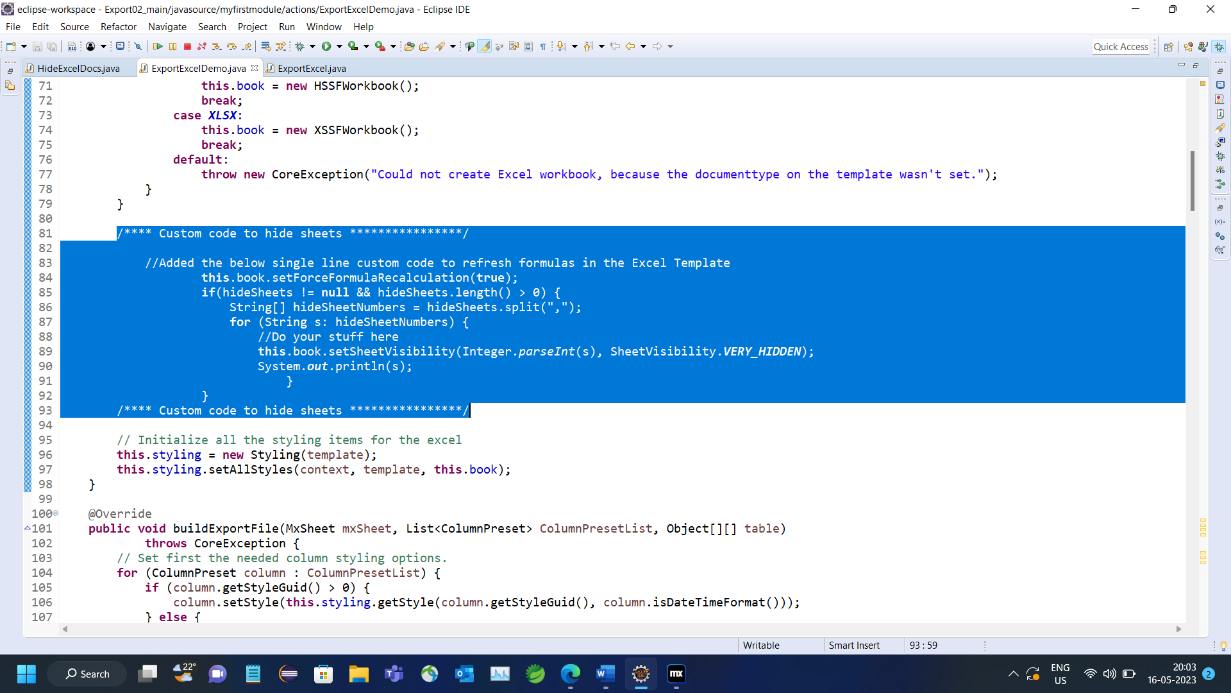
*this.book.setSheetVisibility(Integer.parseInt(s), SheetVisibility.VERY\_HIDDEN);*

*System.out.println(s);*

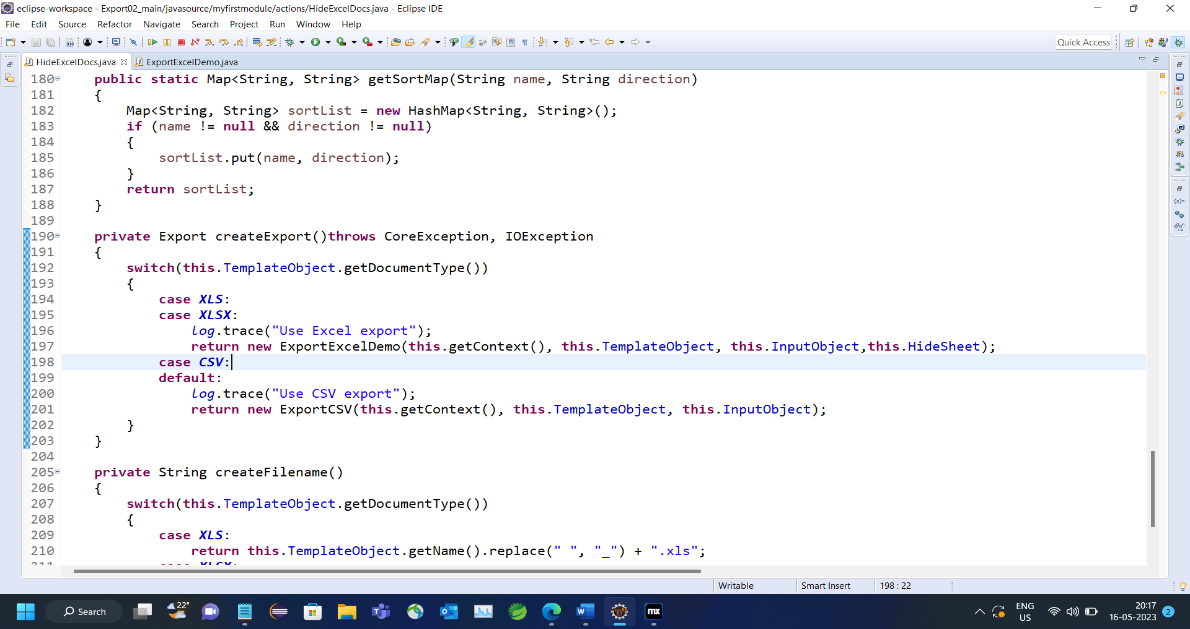
*}*

*}*

* 1. After doing this it will look like the below image:



* 1. Now open HideExcelDocs.java and go to method createExport(). You will find this in line 190. Change the object name from ExportExcel to ExportExcelDemo. See the below image for better understanding. Now run the application.



5) Complete.

**Configuration: NA**

**Category: Module**

**Subcategory: All**

**Visibility: Public Marketplace**

**Studio Pro Version: 9.12.4**