**THE DOCUMENTATION**

# IN THIS DOCUMENT: :

1. WRITING THE BASE OF A MODULE
2. WRITING A MODEL
3. WRITING A MENU ITEM
4. WRITING A VIEW

# WRITING THE BASE OF A MODULE:

To learn on how I would be able to learn to write a module I sought the needed [documentation](https://www.odoo.com/documentation/16.0/developer/tutorials/getting_started/01_architecture.html). With this, I went through the given tutorial.  
  
eventually you will end up on [this page](https://www.odoo.com/documentation/16.0/developer/tutorials/getting_started/03_newapp.html).

On this page we learn on how to create a new module:

The steps are:

1. Creating a new folder.
2. Creating 2 new files (“**manifest.py**”, “**\_\_init\_\_py**”)

in the **manifest.py** file, we will define our module, you can add as much information into the **manifest.py** file as you want (as seen in this **[example](https://github.com/odoo/odoo/blob/fc92728fb2aa306bf0e01a7f9ae1cfa3c1df0e10/addons/crm/__manifest__.py" \l "L1-L67)**). but the minimal requirements are:

* **name**
* **depends**
* **data**
* **application**

**Don’t forget to define ‘base’ in the depends variable!**

By defining these variables as shown in the example and in the tutorial, we will be able to see, and download our module (although it is an empty shell).

* Documentation of this part:

Documentation:

* + <https://www.odoo.com/documentation/16.0/developer/tutorials/getting_started/03_newapp.html>

Examples:

* + [https://github.com/odoo/odoo/blob/fc92728fb2aa306bf0e01a7f9ae1cfa3c1df0e10/addons/crm/\_\_manifest\_\_.py#L1-L67](https://github.com/odoo/odoo/blob/fc92728fb2aa306bf0e01a7f9ae1cfa3c1df0e10/addons/crm/__manifest__.py" \l "L1-L67)

# WRITING A NEW MODEL:

Writing a new module in ODOO is not that hard. If you already have experience in Python Django, then you might see some (not all) resemblance.

Now how do we write a new module:

1. Create a new folder called “**models**” in the same directory as the **\_\_init\_\_.py** and **manifest.py**.  
   now that we created a new folder called “**models**”, add a new file into it called: “**\_\_init\_\_.py**”.  
   this will be the file where we will be importing our models from.
2. Create a new file with the name of the model. e.g., “**apple.py**”.  
     
   to make it “easier to remember” you can choose to name your model with your module name combined. e.g., “**trees\_apples.py**”.
3. Now that we created our model file, we will need to fill it, lets start with the basics:

from **odoo** import models

class **AppleModel**(models.Model):

\_name = "apple.model"

\_description = "an Apple is not a Pear"

1. Now that we made our model, we will need to define it in the \_\_init\_\_.py in the same folder.  
   example:

from **.** import apple

don’t forget to put in the name in front of the .model!! (apple.model )

**TIP:** name your model file and your model with the same name, to not get confused**!!**

**e.g.:** “apple.py”, “apple.model”

now we go to the \_\_init\_\_.py in the main folder, here we define:  
from **models** import \*

with this we have successfully made the base of over time expanding model. We also have defined the create model within the \_**\_init\_\_.py** of main directory and the **models** folder. so that ODOO knows where to find the model.

* Documentation of this part:

Documentation:   
<https://www.odoo.com/documentation/16.0/developer/tutorials/getting_started/04_basicmodel.html>

# ADDING A MENU ITEM:

After having added a new module we will need to have a way to navigate into the Module.  
  
This is done by adding **Menu’s**.

To add a menu to the module we do the following:

1. Create a folder named views.
2. Add a new file into our newly created folder named: \*modulename\*\_menus.xml
3. Add this code to the file and change it according to your model:

**<menuitem** id="apple\_menu\_root" name="Test"**>**

**<menuitem** id="apple\_first\_level\_menu" name="First Level"**>**

**<menuitem** id="apple\_model\_menu\_action" action="test\_model\_action"**/>**

**</menuitem>**

**</menuitem>**with this piece of code we add the menus in order  
  
The “apple\_menu\_root” will be our button where we enter the module from on the homescreen.  
The "apple\_first\_level\_menu" will be the row of buttons that will appear as a header inside our module screen.  
The “apple\_model\_menu\_action” will appear as dropdown options inside the header buttons mentioned above!^

1. Add the folder to the \_\_**manifest\_\_.py**, **don’t forget** to put it in the **data** [] variable, (and keep it on the **bottom** of the list inside it when adding new views!).

After having done all this. ODOO will be able to show us some default views for our model by navigating through the menus.

# WRITING A VIEW:

We have seen in the **previous chapter** that Odoo is able to generate default views for a given model. In practice, the default view is **never** acceptable for a business application. Instead, we should at least organize the various fields in a logical manner.  
In ODOO we have a various type of views, some examples are: **Lists**, **Forms**, **Search** and **Kanban**

## List View

List views, also called tree views, display records in a tabular form.  
Their root element is <tree>. The most basic version of this view simply lists all the fields to display in the table

To create a list view we do the following steps:

1. We create a new file in the views folder with the name of our model. (e.g. tree\_apple\_views.xml)
2. We write code that will define our list view. E.g.:  
     
   **<tree** string="Tests"**>**

**<field** name="name"**/>**

**<field** name="last\_seen"**/>**

**</tree>**

With this we can now control the list view shown in our module.

## Form View

Forms are used to create and edit single records.

Their root element is <form>. They are composed of high-level structure elements (groups and notebooks) and interactive elements (buttons and fields).

To create a custom form view, we simply open our folder, select our views file for the model we want to add the view to, and specify the form and the fields we want to show, e.g.:

**<form** string="Test"**>**

**<sheet>**

**<group>**

**<group>**

**<field** name="name"**/>**

**</group>**

**<group>**

**<field** name="last\_seen"**/>**

**</group>**

**<notebook>**

**<page** string="Description"**>**

**<field** name="description"**/>**

**</page>**

**</notebook>**

**</group>**

**</sheet>**

**</form>**

# Search:

Search views are slightly different from the list and form views since they don’t display *content*. Although they apply to a specific model, they are used to filter other views’ content (generally aggregated views such as List). Beyond the difference in use case, they are defined the same way.

Their root element is <search>. The most basic version of this view simply lists all the fields for which a shortcut is desired:

**<search** string="Tests"**>**

**<field** name="name"**/>**

**<field** name="last\_seen"**/>**

**</search>**