onnet

2021
VIE

Odoo Models and Fields

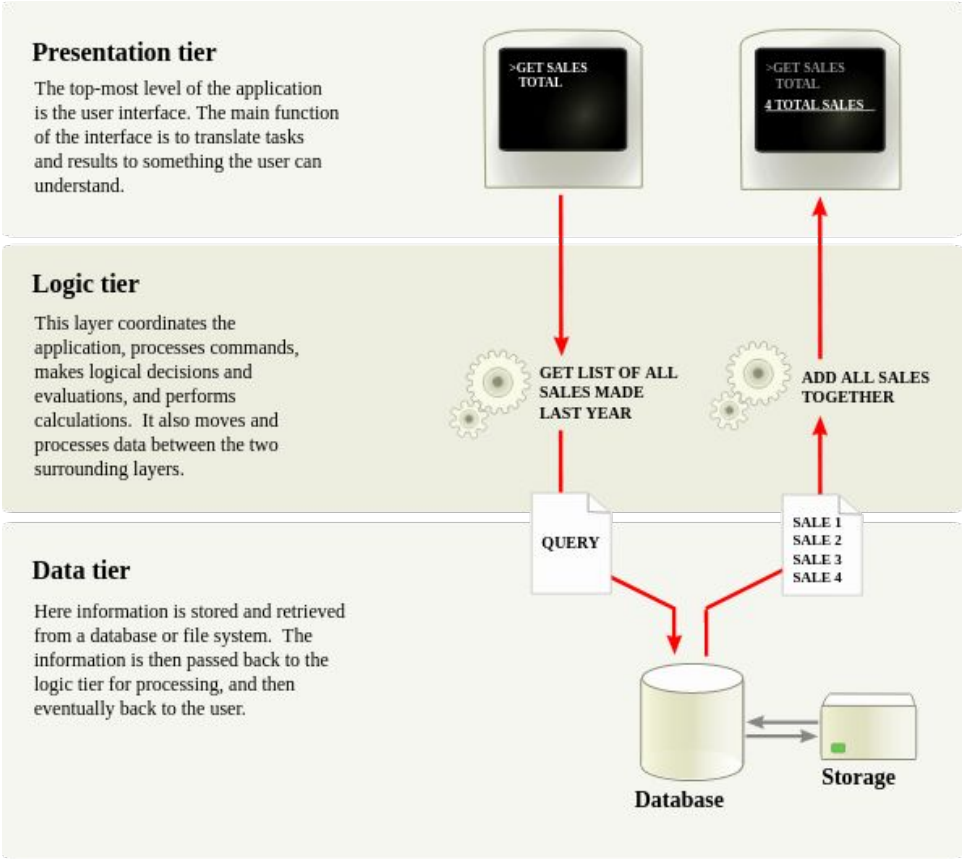
www.on.net.vn
www.arrowhitech.com

onnet
BY AHT TECH JSC

Table of contents

onnet
BY AHT TECH JSC

Architecture Overview



Odoo module (addon): Definition

- Everything in Odoo **starts and ends** with **modules**.

- Both server and client extensions are **packaged as modules** which are **optionally** loaded in a database. A module is a **collection of functions and data** that target a single purpose.

- **Odoo modules** can either add brand new **business logic** to an Odoo system or alter and extend existing business logic. One module can be created to add your country's accounting rules to Odoo's generic accounting support, while a different module can add support for real-time visualisation of a bus fleet.

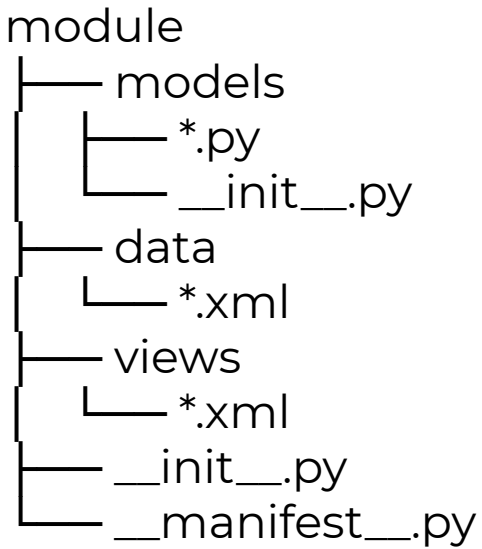
- Developers group their **business features** in **Odoo modules**. The main user-facing modules are flagged and exposed as **Apps**, but a majority of the modules aren't **Apps**. Modules may also be referred to as **addons** and the directories where the Odoo server finds them form the ***addons_path***.

Odoo module (addon): Composition of a module

- **Business objects**: A business object (e.g. an invoice) is declared as a **Python class**. The fields defined in these classes are automatically mapped to **database columns** thanks to the ORM layer.
- **Object views**: Define UI display
- **Data files**: XML or CSV files declaring the model data: views or reports, configuration data (modules parametrization, security rules), demonstration data and more
- **Web controllers**: Handle requests from web browsers

Odoo module structure

- Each module is a directory within a module directory. Module directories are specified by using the **--addons-path** option.
- An Odoo module is declared by its **manifest**.
- When an Odoo module includes business objects (i.e. Python files), they are organized as a Python package with a **__init__.py** file.
- This file contains import instructions for various Python files in the module.



Models: What is models ?

A **Model** determines the logical structure of a database and fundamentally determines in which manner data can be stored, organized, and manipulated.

Models can be configured by setting attributes in their definition. The most important attribute is ***_name***, which is required and defines the name for the model in the **Odoo** system.

Here is a minimum definition of a model:

```
from odoo import models

class TestModel(models.Model):
    _name = "test.model"
```

Model Types

Odoo models are created by inheriting one of the following:

- Model for regular database-persisted models
- TransientModel for temporary data, stored in the database but automatically vacuumed every so often
- AbstractModel for abstract super classes meant to be shared by multiple inheriting models

Fields

Fields are used to define what the model can store and where they are stored.

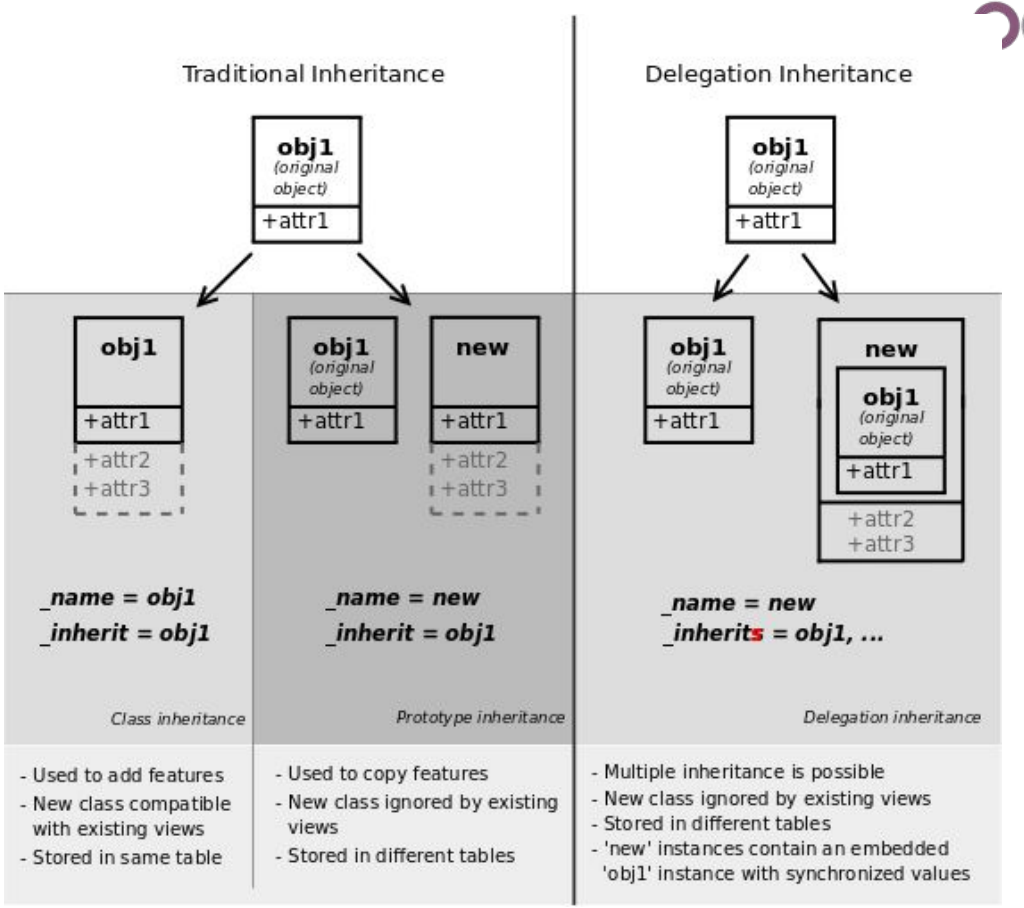
Fields are defined as attributes in the model class:

```
from odoo import fields, models

class TestModel(models.Model):
     _name = "test.model"
     _description = "Test Model"

     name = fields.Char()
```

Model Inheritance

Odoo provides three different mechanisms to extend models in a modular way:

- Classical Inheritance (_name, _inherit)
- Extension (_inherit)
- Delegation (_inherits)



Basic Fields

Boolean:	A boolean (true, false).
Integer:	An integer.
Float:	A floating point number.
Char:	A string of limited length. The required size parameter determines its size.
Text:	A text field with no limit in length.
Date:	A date.
Datetime:	Allows to store a date and the time of day in the same field.
Binary:	A binary chain
Selection:	A field which allows the user to make a selection between various predefined values

Relational Fields

Every model instance is a “recordset”, i.e., an ordered collection of records of the model.

- Many2one: The value of such a field is a recordset of size 0 (no record) or 1 (a single record).
- One2many
- Many2many
- Command

Automatic Fields

Odoo creates a few fields in all models. These fields are managed by the system and can’t be written to, but they can be read if useful or necessary:

- id (Id): The unique identifier for a record of the model.
- create_date (Datetime): Creation date of the record.
- create_uid (Many2one): User who created the record.
- write_date (Datetime): Last modification date of the record.
- write_uid (Many2one): User who last modified the record.

References

- Inheritance and Extension
- Models
- Fields

Practicals

Create a new module. In your module, you should:
- Create at least a new model
- Inherit at least one model and add a new field
- With some fields

You can follow the idea of academy module (which includes courses, books, sessions, students,..) or you can try with your own idea.

Your work is done when you can install your module successfully

Some ideal for new modules:
- academy: courses, books, sessions, attendees, host,...
- hospital: patient, room, bed,..
- library: books, members, ...

Q&A