Odoo 15 - Adding Custom Button in List Documentation

Contents

[Reference Article 1](#_Toc170810179)

[Prerequisite 1](#_Toc170810180)

[Step 1 – Creating a Custom Button 1](#_Toc170810181)

[Explanation: 2](#_Toc170810182)

[Step 2 – Add JavaScript Functionality to the Button 2](#_Toc170810183)

[Explanation 3](#_Toc170810184)

[Step 3 – Add the newly created files in \_\_manifest\_\_.py of the module 3](#_Toc170810185)

[Step 4 – Extend the Tree 3](#_Toc170810186)

[Step 5 – Now Reload the Backend and Upgrade the Respective App 4](#_Toc170810187)

Reference Article - <https://www.cybrosys.com/blog/how-to-add-a-create-button-near-tree-kanban-view-in-odoo-15>

Prerequisite –

* Odoo Project Structure
* Where and how to find the View code from frontend
* How to inherit views
* How to add Javascript functionality to our Tree/Kanban
* Xpath , attributes

## 

## Step 1 – Creating a Custom Button

Create a file named tree\_button.xml within the account module. The path for this file will be account/static/src/xml/tree\_button.xml.

<?xml version="1.0" encoding="UTF-8"?>

<templates>

   <t t-extend="ListView.buttons" t-name="button\_near\_create.buttons">

       <t t-jquery="button.o\_list\_button\_add" t-operation="after">

           <button type="button" class="btn btn-primary mr-2">

               Open Wizard

           </button>

       </t>

   </t>

</templates>

### Explanation:

* **t-extend**: Extends the ListView.buttons template.
* **t-name**: Defines the name of the extended template (button\_near\_create.buttons).
* **t-jquery**: Specifies the jQuery selector (button.o\_list\_button\_add) to locate the position after the "Create" button.
* **t-operation**: Indicates the position (after) where the new button should be inserted.
* **Button Configuration**: Configures a new button with the class btn btn-primary ms-2 and displays the text "Open Wizard".

## Step 2 – Add JavaScript Functionality to the Button

Create a file tree\_button.js within the account module. The path of this file will be account/static/src/js/tree\_button.js

odoo.define('button\_near\_create.tree\_button', function (require) {

    "use strict";

    var ListController = require('web.ListController');

    var ListView = require('web.ListView');

    var viewRegistry = require('web.view\_registry');

    var TreeButton = ListController.extend({

       buttons\_template: 'button\_near\_create.buttons',

    });

    var JournalItemListView = ListView.extend({

       config: \_.extend({}, ListView.prototype.config, {

           Controller: TreeButton,

       }),

    });

    viewRegistry.add('button\_in\_tree', JournalItemListView);

    });

Explanation**:**

* **odoo.define**: Registers a new Odoo module named button\_near\_create.tree\_button.
* **ListController and ListView**: Imports the base ListController and ListView modules from Odoo's web framework.
* **viewRegistry**: Imports the view registry to register custom views.
* **TreeButton**: Extends ListController to define custom behavior for managing buttons in the view.
  + **buttons\_template**: Specifies the XML template (button\_near\_create.buttons) to define custom buttons.
* **JournalItemListView**: Extends ListView to configure the view with custom settings.
  + **config**: Extends the default configuration of ListView to include the custom TreeButton controller.
* **viewRegistry.add**: Registers a new view type (button\_in\_tree) using the customized JournalItemListView.

## Step 3 – Add the newly created files in \_\_manifest\_\_.py of the module

'assets':

        'web.assets\_backend': [

            'account/static/src/js/tree\_button.js',

        ],

        'web.assets\_qweb': [

            'account/static/src/xml/tree\_button.xml',

        ],

    },

Adjust your path according to actual location

## 

## Step 4 – Extend the Tree

  <record id="view\_in\_invoice\_bill\_tree" model="ir.ui.view">

            <field name="name">account.out.invoice.tree</field>

            <field name="model">account.move</field>

            <field name="inherit\_id" ref="account.view\_in\_invoice\_tree"/>

            <field name="arch" type="xml">

                <tree string="Invoices">

                    <field name="currency\_id" position="attributes">

                        <attribute name="string">Bill Currency</attribute>

                    </field>

                </tree>

            </field>

        </record>

        <record id="view\_in\_invoice\_bill\_tree\_inherit" model="ir.ui.view">

           <field name="name">account.move.invoice.tree.inherit</field>

           <field name="model">account.move</field>

           <field name="inherit\_id" ref="account.view\_in\_invoice\_bill\_tree"/>

           <field name="arch" type="xml">

               <xpath expr="//tree" position="attributes">

                   <attribute name="js\_class">button\_in\_tree</attribute>

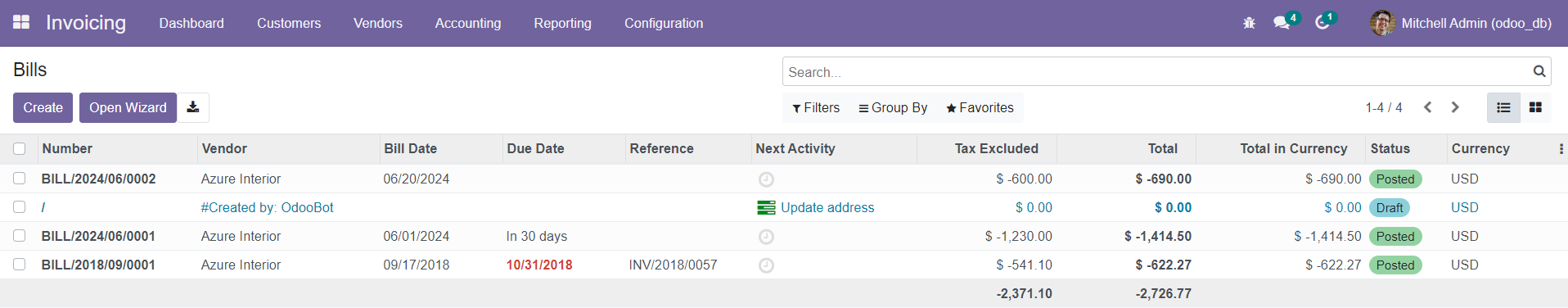
               </xpath>

           </field>

       </record>

* id="view\_in\_invoice\_bill\_tree" is the already implemented view
* id="view\_in\_invoice\_bill\_tree\_inherit” is the inherited view
* **inherit\_id**: Refers to the previously defined view\_in\_invoice\_bill\_tree to inherit and further customize the view.
* **arch**: Defines the view architecture.
* **xpath**: Locates the tree element and modifies its attributes.
  + **attribute[name='js\_class']**: Adds the js\_class attribute with the value button\_in\_tree to integrate the custom JavaScript functionality.

## Step 5 – Now Reload the Backend and Upgrade the Respective App



You can now see the Open Wizard Button besides Open Wizard