MK.Lab

Урок 3

«Интерфейс пользователя»

Как устроен пользовательский интерфейс? 1
Пользовательские формы Form View 1
Отступление про виджеты 2
Опции 3
Атрибуты 3
Стили 4
Домен 4
Задание для Form View 5
Списки Tree View 6
Поиск Search View 6

Как устроен пользовательский интерфейс?

Вы уже знаете, что по сути вся система odoo состоит из моделей данных или просто моделей.

Для каждой модели для создания пользовательского интерфейса имеются следующие объекты:

- Forms View
- Tree View
- Search View
- Kanban View
- Calendar View
- Action

Давайте рассмотрим основные объекты отдельно.

Пользовательские формы Form View

Это XML документ, который содержит в себе все элементы взаимодействия с пользователем: средства отображения и ввода данных в поля, кнопки, надписи и пр.

Форма всегда содержится внутри тега <form> в XML файлах.

Зайдите в режим Developer Mode и перейдите в модуль CRM, инициативы (воронки).

Создайте новый документ.

Откройте «жучок» - Edit Form View.

Давайте рассмотрим основные разделы этой формы.

<header> это заголовок. В нем удобно делать кнопки и выводить всякие предупреждающие данные.

<sheet> это тело формы. Если Вы сделаете форму без sheet, она будет некрасивой и в ней нельзя будет делать группировку полей.

<group> группировка полей. Форма имеет стандартную верстку Bootstrap (ранее Twitter Bootstrap). Поэтому имеются 4 колонки для вывода данных на стандартном экране. Если Вы выводите поля и названия полей, то сможете уместить две колонки полей, задав группировку.

В разных версиях odoo группировка работает немного по-разному, поэтому попробуйте на практике, как получается у Вас.

Совет: внутри группы может быть еще группа. А еще можно дать имя группе, указав опцию string= «имя группы»

Внутри формы:

<field name= «name» /> эта конструкция выводит поле из вашей модели. Виджет при этом автоматически соответствует типу данных.

Отступление про виджеты

Виджет — это средство пользовательского интерфейса, которое удобно отображает пользовательские данные.

Есть виджеты автоматические, которые система сама назначает полю в зависимости от его типа данных. В коде нигде этот виджет не виден, все происходит автоматически.

Но для некоторых типов данных Вы можете вручную указать виджет.

Например, есть тип данных Many2many. Отношение много-ко-многим. Например, в своей модели Вы в определенном поле хотите сделать связку со многими записями другой модели. Классический пример — какая-нибудь аналитика. Например, теги.

Есть список тегов и мы хотим назначить эти теги в нашу запись.

По умолчанию виджет many2many выведет вместо поля таблицу-список (Tree), в котором будет список привязанных объектов.

Но мы можем указать другой виджет many2many_tags, и тогда поле появится не с таблицей, а с красивыми тегами.

Как это устроено, посмотрите в карточке контакта. Там есть поле «Теги». Попробуйте заполнить.

MK.Lab

Для полей типа Many2many есть следующие виджеты:

many2many

many2many_tags

many2many_checkboxes

many2many_kanban

x2many_counter

many2many_binary

У любого поля есть важные параметры. Атрибуты, опции и домен.

Опции

Опции позволяют сделать настройки внешнего вида.

<field name="field_name" options="{'no_create': True}"/>

В этом случае для поля типа Many2many мы убираем показ пункта «Создать ЧТО_ТО» в выпадающем меню.

Есть варианты: no_quick_create, no_create_edit. Первый убирает «Создать и изменить», второй убирает оба пункта.

Очень полезная вещь.

Атрибуты

Атрибуты управляют отображением. Например, можно задать возможность появления поля на форме в зависимости от каких-либо значений других полей или в зависимости от прав пользователя.

Отдельно отметим атрибуты:

readonly=1 устанавливает поле в режим «Только чтение». Важное замечание: если Ваше поле в модели имеет атрибут «Только чтение», то этот атрибут появится сам. Если нет, а Вы в форме поставили **readonly=1** и, например, сделали бизнес-логику, которая срабатывает при изменении поля (onchange), то Ваше поле не будет изменяться. Хотя как будто бизнес-логика отрабатывает на стороне сервера, а форма на стороне клиента.

invisible=1 скрывает поле. Но оно по-прежнему на форме и содержит данные. Вы можете делать свои служебные вычисляемые поля для разных целей и скрывать их от пользователей.

required=1 делает поле обязательным для заполнения.

Атрибуты могут быть динамическими. Т.е., включаться и выключаться в зависимости от значений полей.

default_focus=1

Указывает, что данное поле получит фокус при отображении формы.

attrs="{'invisible': [('extend_prop', '=', False)]}"

В данном случае поле буде невидимым, когда истино логическое выражение.

А именно: поле extend_prop имеет значение False

Начиная с 17 версии odoo свойство attrs является устаревшим, вместо этого будет использована конструкция:

invisible = "extend_prop == False"

Для указания нескольких условий:

invisible = "extend_prop == False and active==True"

Стили

В odoo частично работают стили HTML.

<field name="doctor" style="width: 50%" options="{'no_create': True}" default_focus="1" required="1" domain="[('fired','=',False)]"/>

Домен

Это фильтр для полей, связанных с другими моделями.

<field name="product_id" domain="[('type','=','service')]"/>

Такая конструкция означает, что в поиске будут участвовать только те записи, у которых поле type равно service. В данном случае связь с моделью product.product, т. е., с продуктами, а фильтр означает, что будут отображаться только услуги.

Про домены можно отметить следующее.

Домен всегда нужно писать в квадратных скобках

[]

Внутри домена записаны кортежи в круглых скобках. Они состоят из элементов:

(Поле, Операция, Значение)

Операции могут быть такие:

'=', '!=', '<=', '<', '>', '>=', '=?', '=like', '=ilike', 'like', 'not like', 'ilike', 'not ilike', 'in', 'not in', 'child_of'

В XML коде нужно помнить, по то, что не должно быть <, это признак тега. Нужно использовать замену: < (<), & (&), > (>), " ("), ' ('). В Python можно писать как есть.

Логические операторы внутри записи домена могут быть И или ИЛИ. Записываются так '&' и '|'

Домены записываются **в прямой польской записи**.

Рассмотрим пример перевода обычной записи логического выражения в прямую польскую:

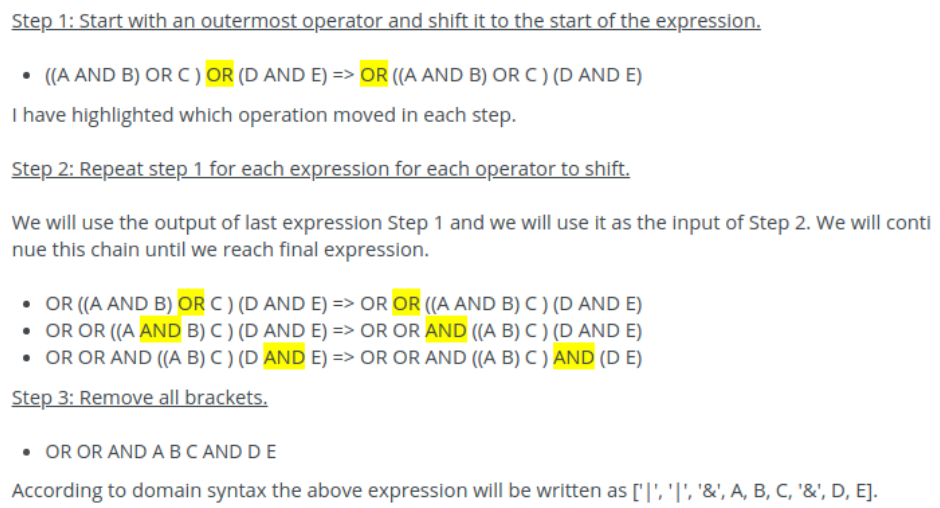
Пусть дано:

((A AND B) OR (C AND D) OR (F AND G)) AND E AND F

В прямой польской записи имеем:

AND AND OR OR AND A B AND C D AND F G E F

Про перевод в польскую запись посмотрите в Интернете. Очень хороший метод описан через вынос оператора OR за скобки. Приведем описание без изменений:



Step 1: Start with an outermost operator and shift it to the start of the expression.

- ((A AND B) OR C) OR (D AND E) => OR ((A AND B) OR C) (D AND E)

I have highlighted which operation moved in each step.

Step 2: Repeat step 1 for each expression for each operator to shift.

We will use the output of last expression Step 1 and we will use it as the input of Step 2. We will conti nue this chain until we reach final expression.

- OR ((A AND B) OR C) (D AND E) => OR OR ((A AND B) C) (D AND E)
- OR OR ((A AND B) C) (D AND E) => OR OR AND ((A B) C) (D AND E)
- OR OR AND ((A B) C) (D AND E) => OR OR AND ((A B) C) AND (D E)

Step 3: Remove all brackets.

- OR OR AND A B C AND D E

According to domain syntax the above expression will be written as ['|', '|', '&', A, B, C, '&', D, E].

Это довольно сложный момент для новичков, а также для тех, кто не изучал математику в университете. Но стоит потратить время, чтобы этому научиться, потому что использование домена иногда порой нечем заменить, домен работает значительно быстрее, т.к. непосредственно формирует запрос к базе данных.

Если в Вашей базе много записей, нельзя в целях упрощения загонять в свой recordset все данные, а потом в цикле их перебирать, накладывать фильтры, писать кучу if и т.д. Нужно поручить выборку СУБД, а у себя в python коде уже работать с нужным набором.

Задание для Form View

В Вашем модуле, который умеет менять тип поля в модуле crm, добавьте в класс еще одно поле. Сделаем связь с продуктом.

product_id= fields.Many2one('product.product', string='Услуга')

Обновите модуль.

Добавьте на форму новое поле.

Сделайте так, чтобы выбор был только из товаров.

Сделайте так, чтобы поле пропадало, если вероятность сделки менее 10.

Сделайте так, чтобы поле было красного цвета.

Списки Tree View

Эта структура служит для отображения списка записей.

Структура повторяет общую структуру любого View.

Т.е. список полей и атрибуты, которые мы рассмотрели в разделе про формы.

```
<record model="ir.ui.view" id="main_contract.list">
 <field name="name">contract header list</field>
 <field name="model">contract.header</field>
 <field name="arch" type="xml">
  <tree>
   <field name="name"/>
   <field name="date_start"/>
   <field name="date_end"/>
   <field name="date_signed"/>

   <field name="state"/>
   <field name="store_id"/>
  </tree>
 </field>
</record>
```

Как видите, ничего необычного тут нет.

Поиск Search View

В версиях Odoo13 и старше Search View называется Control Panel View.

Для того чтобы настроить пользовательский поиск, указав системе, какие поля будут участвовать в поиске, какие сделать заранее заданные фильтры и группировки, существует специальный пользовательский View.

Давайте рассмотрим пример:

```
<?xml version="1.0"?>
<search>
<filter string="Сегодня"
    domain="[('appointment_date', '&gt;=', datetime.datetime.now().strftime('%Y-%m-%d 00:00:00')),
    ('appointment_date', '&lt;=',datetime.datetime.now().strftime('%Y-%m-%d 23:23:59'))]"/>
<filter string="Созданы сегодня"
    domain="[('create_date', '&gt;=', datetime.datetime.now().strftime('%Y-%m-%d 00:00:00'))]"/>
<separator />
<filter string="Повторный" domain="[('new_pat1', '=', '3')]"/>
<filter string="Новый" domain="[('new_pat1', '=', '2')]"/>

<separator />
<filter string="Троицкая, 7" domain="[('institution_partner_id','=','Троицкая, 7/1')]" />
<filter string="Палиха, 13" domain="[('institution_partner_id','=','Палиха, 13')]" />

<separator/>
<filter string="Счет не выставлен" domain="[('is_invoiced','=',False)]"/>

<field name="doctor_id"/>
<field name="patient_id"/>
<field name="name"/>
<field name="day"/>

</search>
```

Строки типа **<field name="day"/>** указывают, что по этому полю можно делать поиск.

<filter string="Счет не выставлен" domain="[('is_invoiced','=',False)]"/> указывает на то, что мы создали фильтр с указанным доменом.

<separator /> нужен для того, чтобы фильтры сработали в режиме логического И или ИЛИ.

Поиграйте с фильтрами в системе, посмотрите как они работают.

Начиная с 13-ой версии у нас есть еще один объект. Это панель слева.

```
<searchpanel>
    <field name="company_id" groups="base.group_multi_company" icon="fa-building" enable_counters="1"/>
    <field name="department_id" icon="fa-users" enable_counters="1"/>
</searchpanel>
```

Элементы панели поиска могут иметь следующие атрибуты:

name: имя поля для фильтрации

one (по умолчанию): мы можем выбрать только одно значение за раз (для полей типа many2one и selection).

multi: в multi мы можем выбрать несколько значений с помощью флажков. Поддерживаются типы полей many2one, many2many и selection.

icon: возможность выбора иконки, которая будет отображаться вместе со строкой.

enable_counters: значение по умолчанию равно нулю. Если включено, мы можем увидеть нет. записей в каждом значении.

limit: определяет максимальное количество записей для выборки для поля.

Итоговое задание:

Сделайте задание из раздела FormView.