

Accións Planificadas

Accións Planificadas

Podemos executar accións planificadas dende Odoo cada certo tempo. Para elo temos que crear un rexistro na táboa **ir_cron**.

Podemos ver a definición dese rexistro no arquivo **accion_planificada.xml** no cartafol **accions_planificadas**. Dito arquivo ten que estar referenciado no apartado 'data' do arquivo **"__manifest__.py"**

```
<odoo>
  <data>
    <!-- https://odoo-development.readthedocs.io/en/latest/odoo/models/ir.cron -->
    <record model="ir.cron" id="envio_facturas">
      <field name="name">Revisión de Facturas</field>
      <field name="active" eval="True" />
      <field name="user_id" ref="base.user_root" />
      <field name="interval_number">1</field>
      <field name="interval_type">hours</field>
      <field name="numbercall">-1</field> <!-- An integer value specifying the number of calls to execute -->
      <!--<field name="function">True</field>Odoo Version 12 -->
      <!--<field name="model_id" ref="model_account_invoice" /> Odoo Version 12 -->
      <field name="model_id" ref="model_account_move" /> <!-- Odoo Version 12 -->
      <field name="state">code</field>
      <field name="code">model.listado_facturas()</field>
      <field name="doall" eval="False" /> <!-- A boolean value indicating whether to execute the code immediately -->
    </record>
  </data>
</odoo>
```

Despois de actualizar o noso módulo podemos ver como foi dado de alta:



Acciones planificadas

Crear Importar

	Prioridad	Nombre de acción
<input type="checkbox"/>	5	Base: Limpieza automática de datos internos
<input type="checkbox"/>	5	Contabilidad: Generar asientos de reverso
<input type="checkbox"/>	1.000	Correo: Notificar a los moderadores de canales
<input type="checkbox"/>	5	Correo: servicio de Fetchmail
<input type="checkbox"/>	5	Correos de Digesto
<input type="checkbox"/>	5	Mail: Gestor de colas de email
<input type="checkbox"/>	5	Partner Autocomplete: Sincronización con la base de datos remota
<input type="checkbox"/>	5	Post process payment transactions
<input type="checkbox"/>	5	Revisión de Facturas
<input type="checkbox"/>	5	Snailmail: process letters queue

Acciones de ventana
Acciones de servidor
Asistentes de configuración
Valores por defecto del usuario
Interfaz de usuario
Elementos de menú
Vistas
Vistas personalizadas
Filtros de usuario
Recorridos
Estructura de la base de datos
Modelos
Campos
Restricciones del modelo
Relaciones ManyToMany
Adjuntos
Registro
Precisión decimal
Automatización
Acciones planificadas
Informes
Formato de papel
Informes
Secuencias e identificadores
Identificadores externos
Secuencias
Parámetros
Parámetros del sistema
Propiedades de la compañía
Seguridad
Reglas de registro
Permisos de acceso

Todos
Filtros

le ejecución

Se vemos a acción planificada, podemos ver que temos un botón de "Ejecutar Manualmente"(qu podemos utilizar para hacer as probas):

Ajustes Tablero Usuarios y compañías Traducciones Opciones Generales Técnico

Acciones planificadas / Revisión de Facturas

Editar Crear

Ejecutar Manualmente

Acción a realizar Ejecutar el código Python

Revisión de Facturas

Modelo Factura

Usuario del planificador OdooBot

Ejecutar cada 1Horas

Siguiente fecha de ejecución 28/04/2020 14:16:45

Número de ejecuciones -1

Prioridad 5

Repetir perdidos ☐

Código Python Ayuda

model.Listado_facturas()

pgAdmin 4 - Mozilla Firefox

pgAdmin 4

127.0.0.1:40881/browser/

pgAdmin

Query Editor Query History

SELECT * FROM public.ir_cron

Data Output Explain Messages Notifications

id	ir_actions_server_id	cron_name	user_id	active	interval_type	interval_number	numbercall	doall	nextcall
5	13	Revisión de Facturas	1	true	hours	1	-1	false	2020-04-28 12
6	11	Post process payment transactions	1	true	minutes	10	-1	false	2020-04-28 12
7	1	Base: Auto-vacuum internal data	1	true	days	1	-1	false	2020-04-28 12
8	3	Publisher: Update Notification	1	true	weeks	1	-1	false	2020-04-29 12
9	4	Mail: Notify channel moderators	1	true	days	1	-1	false	2020-04-28 12
10	8	141 Digest Emails	1	true	days	1	-1	false	2020-04-28 12
11	9	231 Account: Reverse entries	1	true	days	1	-1	false	2020-04-28 12

Cabe destacar nos campos que estamos definiendo:

- O usuario que executará a acción planificada `<field name="user_id" ref="base.user_root" />` definido a través do id externo(base.user_root).

O cal nos permite saber:

Ajustes | Tablero | Usuarios y compañías | Traducciones | Opciones Generales | Técnico

Identificadores externos / OdooBot

[Editar](#) [Crear](#) [Acción ▾](#)

base.user_root

Módulo: base

Identificador externo: user_root

No actualizable: ☒

Fecha de actualización: 08/04/2020 14:58:58

Fecha inicial:

Nombre mostrado: OdooBot

Nombre del modelo: res.users

ID de registro: 1

Registro: OdooBot

Que é o usuario con id=1 da táboa res_users.

PgAdmin | File | Object | Tools | Help

Browser: res_partner_res_partner_category_rel, res_partner_title, res_users

Columns (16): id, active, login, password, company_id, partner_id, create_date, signature, action_id, share, create_uid, write_uid, write_date, alias_id, notification_type, odooobot_state

Constraints (7): res_users_alias_id_fkey, res_users_company_id_fkey, res_users_create_uid_fkey, res_users_login_key, res_users_partner_id_fkey, res_users_pkey, res_users_write_uid_fkey

Indexes: , Rules:

Dashboard | Properties | SQL | Statistics | Dependencies | Dependents | public.res_user... | public.res_part... | public.res_users/d

Query Editor: public.res_users/documentacion/odoo@127.0.0.1

```
1 SELECT * FROM public.res_users
```

Data Output | Explain | Messages | Notifications

id	active	login	password	company_id	partner_id	create_date
1	3 false	default	[null]		5	2020-04-08 12:55:36.116836
2	2 true	odoo@odoo.com	\$pbkdf2-sha512\$2500...	1	3	2020-04-08 12:55:36.116836
3	4 false	public	\$pbkdf2-sha512\$2500...	1	4	2020-04-08 12:55:36.116836
4	5 false	portaltemplate	[null]	1	6	2020-04-08 12:55:36.116836
5	14 true	lectura@odoo.com	\$pbkdf2-sha512\$2500...	1	16	2020-04-17 12:28:20.93468
6	15 true	escritura@odoo.com	\$pbkdf2-sha512\$2500...	1	17	2020-04-17 12:28:20.93468
7	1 false	__system__	[null]	1	2	2020-04-08 12:55:35.092196

Vemos que é o usuario **__system__** que está relacionado coa táboa res_partner mediante partner_id = 2. Recordemos que res_users e res_partners teñen unha relación de Herdanza por Delegación.

PgAdmin | File | Object | Tools | Help

Browser: street, street2, zip, city, state_id, country_id, email, phone, mobile, is_company, industry_id

Dashboard | Properties | SQL | Statistics | Dependencies | Dependents | documentacion/odoo@127.0.0.1 *

Query Editor: documentacion/odoo@127.0.0.1

```
1 Select id,name,display_name,tz,active,email from res_partner order by id
```

</

Agora podemos ver que o rexistro de res_partner con id=2, ten como name=OdooBot, com tz=UTC e como email=odoobot@example.com

- O intervalo e tipo, no noso caso cada "1 hora". `<field name="interval_number">1</field>`
`<field name="interval_type">hours</field>`
- O modelo, no noso caso:
 - Versi3n 12 de Odoo **account_invoice** (usaremos a herdanza). Identifícase tam3n tr3v3s do id externo. `<field name="model_id" ref="model_account_invoice" />`
 - Versi3n 13 de Odoo **account_move** (usaremos a herdanza). Identifícase tam3n tr3v3s do id externo. `<field name="model_id" ref="model_account_move" />`
- O m3todo a executar que ser3 listado_facturas `<field name="code">model.listado_facturas()</field>`

Vemos a definici3n do m3todo **listado_facturas** que temos definido no arquivo **accion_planificada.py**

```
class accion_planificada (models.Model):
    _inherit = "account.invoice" # Odoo Version 12
    _inherit = "account.move" # Odoo Version 13

    @api.model
    def listado_facturas(self):
        usuario_que_executa_o_metodo_que_e_o_definido_no_xml = self.env.user # usu
        self.env.user.tz = self.env['res.partner'].search([('id', '=', 3)])[0].tz
        agora = self.env['odoo_basico.informacion'].convirte_data_hora_de_utc_a_tir
        self.env.user.tz = 'UTC' # deixamolo como estaba con tz=UTC
        #facturas_ids = self.search([('state', '=', 'open')]) Version Odoo 12
        facturas_ids = self.search([('state', '=', 'posted')])
```

```

if facturas_ids:
    listado = ""
    for rexistro in facturas_ids:
        #listado = listado + "<br/>" + str(rexistro.number) + "-> " + str(rexistro.name)
        listado = listado + "<br/>" + str(rexistro.name) + "-> " + str(rexistro.number)
    # mail_de = self.env['res.partner'].search([('id', '=', 1)])[0].email
    # mail_relay_to = self.env['res.partner'].search([('id', '=', 1)])[0].email
    mail_reply_to = usuario_que_executa_o_metodo_que_e_o_definido_no_xml.py
    mail_para = self.env['res.partner'].search([('id', '=', 3)])[0].email
    mail_valores = {
        'subject': "Listaxe de facturas neste momento %s" % agora,
        'author_id': usuario_que_executa_o_metodo_que_e_o_definido_no_xml.py,
        'email_from': mail_reply_to,
        'email_to': mail_para,
        'message_type': 'email',
        'body_html': "Neste momento %s existen as seguintes facturas: %s" % (agora, listado)
    }
    mail_id = self.env['mail.mail'].create(mail_valores)
    mail_id.send()

```

Vemos que herdamos por Extensión o modelo **account.move** (na versión 13 de Odoo) e nesto caso engadimos un novo método "**listado_facturas**" para xerar e enviar un email.

O primeiro é determinar a data e hora actual para engadir no email. Mediante `fields.Datetime.now()` temos o datetime. Pero en hora UTC e queremos convertila ao timezone do usuario administrador.

Isto faremolo chamando ao método "`convirte_data_hora_de_utc_a_timezone_do_usuario`" que temos definido no módulo información.

```

agora = self.env['odoo_basico.informacion'].convirte_data_hora_de_utc_a_timezone_do_usuario()

```

Neste momento temos un problema xa que se recordamos o código que temos no módulo "informacion":

```

def convirte_data_hora_de_utc_a_timezone_do_usuario(self, data_hora_utc_object):
    usuario_timezone = pytz.timezone(self.env.user.tz or 'UTC') # obter a zona do usuario
    return pytz.UTC.localize(data_hora_utc_object).astimezone(usuario_timezone)
# para usar pytz temos que facer import pytz

```

Podemos ver que para determinar o tz usa o obxeto 'self' (`self.env.user.tz`) e como xa sabemos Python pasa automaticamente 'self' dende onde fixemos a chamada. No noso caso `self.env.user` far

referencia ao usuario que est executando a accion_planificada que non  o usuario administrado senon que  como xa vimos na definici3n de **accion_planificada.xml** o usuario **__system__** que ten como tz=**UTC**.

Por tanto, o que facemos  modificar temporalmente o valor de self.env.user.tz e **accion_planificada.py** para que cando chamemos "converte_data_hora_de_utc_a_timezone_do_usuario" tea o valor do usuario administrador. Is conseguimos mediante:

```
self.env.user.tz = self.env['res.partner'].search([('id', '=', 3)])[0].tz # como 1
```

Despois de facer a conversi3n volvemos deixar o obxeto self co seu valor orixinal:

```
self.env.user.tz = 'UTC' # deixamolo como estaba con tz=UTC
```

A continuaci3n seleccionamos todas as facturas que teen state= posted. Como estamos e **accion_planificada.py** e este modelo herda de **account.move** (na versi3n 13 de Odoo) o obxecto 'self' fai referencia aos valores das facturas.

```
#facturas_ids = self.search([('state', '=', 'open')]) Version Odoo 12
facturas_ids = self.search([('state', '=', 'posted')])
```

Se existe algunha factura inicializamos unha variable de texto (**listado**) para ir engadindo mensaxe que queremos enviar no body do email.

Mediante un bucle imonos movendo por todas as facturas e acumulando na variable de texto **listado** :

- Unha factura por lia, mediante a etiqueta HTML **br**
- Por exemplo seleccionamos para cada factura: Nmero de factura, Nome do cliente
Importe total da factura
- Como vimos no apartado de "Envio de Email" xogamos con 3 enderezos de email.
 - From: ser a conta de gmail que temos configurado en Odoo. Non temos que indicala o fai Odoo.
 - Reply_to: No noso caso ser o email do usuario que executa a acci3n planificada (**__system__**).

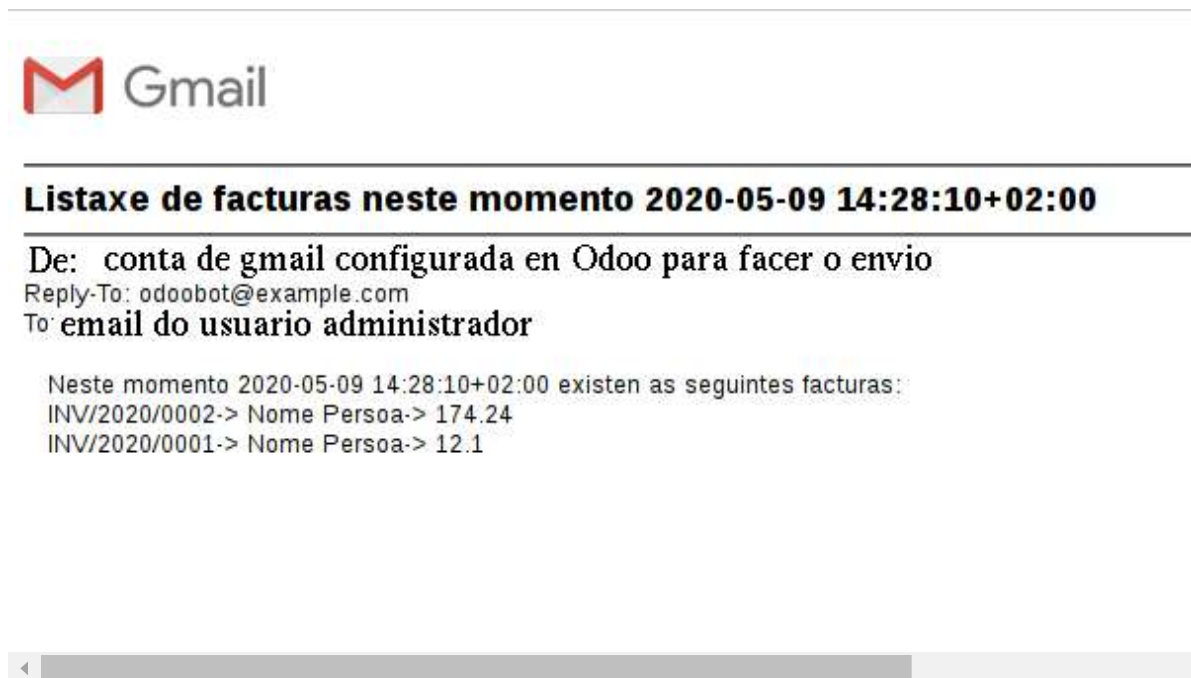
(__system__):

- mail_reply_to = usuario_que_executa_o_metodo_que_e_o_definido_no_xml



- To: O email do usuario administrador. (Se nos fixamos nas capturas anteriores figur odoo@odoo.com, pero modifiquei a configuración do email do usuario administrador unha conta miña onde poder recibir os emails)
- E finalmente como xa vimos no apartado de "Envío de Email" cargamos os valores necesarios para o envío do email na variable mail_valores, creamos o email e enviamolo. Neste caso non temos que utilizar sudo porque o usuario que envía o email é __system__ ten os permisos necesarios para facelo. Caso distinto de cando enviamos o email dende módulo informacion xa que nese caso o usuario era o administrador e necesitamos utilizar **sudo** para que tivese os permisos necesarios.

Vemos o email recibido:



Obra publicada con [Licencia Creative Commons Reconocimiento Compartir igual 4.0](https://creativecommons.org/licenses/by-sa/4.0/)