

Now we will know the parameters:

- **params['Caption']**: Form title;
- **params['SearchSQL']**: This is the SQL that will be executed on the search frame class. It must contain the table Primary Key and its fields that you want show on the grid results. You MUST inform the clause %WHERECLAUSE% on the primary where. This substitution variable will be substituted on executing time by the where clause generated by the search frame class.
* NOTE: on future, pyHed will allow the developer to user a ORM class instead of a sql string.;
- **params['SearchColumnsTitle']**: Is a list with fields titles on search grid results. Note that the first is the table primary key and is preceded by *, Its means that it will be a hide column and will contains the primary key necessary to open data form by search form. IMPORTANT: The order of columns should be same of sql, be careful with it for your form work..
- **params['SearchFields']**: Is a list that contains the search filters. Each item should be a string separated by comma that should contain the follow order: *“label, FIELD, type”*. Where:
 - **label**: Text that is visible to user;
 - **FIELD**: The clause that will be inserted.
 Obs.: If your select contain any join statement and your tables have any alias for example “PESSOA P” the string value should be P.NOME as the example above. Its avoid problems as ambiguous columns.
 - **type**: Is the field type.
 Allowed types: string, integer, date;
- **params['SearchFields']**: Must be informed the primary Key of table or view.

OK. The “params” has all that needs. Now execute the inherits `__init__` follows example.

```
super(FrameGerenciaPessoas, self).__init__(parent, params, False, True)
```

OnPaint() Method

In this example we will rewrite it and calls inherits. But its can used for add buttons or another components that you needs. Follows the code:

```
def onPaint(self):
    super(FrameGerenciaPessoas, self).onPaint()
```

“evtGridResultDoubleClicked” Method

This is the method that is attributed on grid double click. If your form not execute any action on this event, rewrite using “pass”. Follow example:

```
.....  
def evtGridResultDoubleClicked(self, aRow, aCol):  
    pass  
.....
```

In our example we will open the register for from select grid result, then we will build the event like this:

```
.....  
def evtGridResultDoubleClicked(self, aRow, aCol):  
    self.setCursor(PyQt4.QtCore.Qt.WaitCursor)  
    try:  
        pyHedConsts.frmMain.showFrame(frameCadPessoa.FrameCadPessoa,  
                                     self,  
                                     ['GERENCIA_PESSOAS',  
                                      int(self.gridResult.item(aRow, 0).text())])  
    finally:  
        self.setCursor(PyQt4.QtCore.Qt.ArrowCursor)  
.....
```

OK. Your form is ready. Now we will create the menu and item of menu.

Add the menu like example bellow:

```

# Import dos Frames de Cadastro do Sistema
import frameCadPessoa
import frameGerenciaPessoas
import frameRelatorioPessoas

# Main class
class FrameNovaAppMain(frmCustomMain.FrmCustomMain):
    def __init__(self):
        # Login and start app
        super(FrameNovaAppMain, self).__init__(frameLogin.FrameLoginNovaApp,
                                                '%s/splashScreen.png' % appConsts.appImagePath,
                                                windowMaximize=False)

        super(FrameNovaAppMain, self).setIcon('%s/appIco.ico' % appConsts.appImagePath)

    def registerFrames(self):
        """ frames registration """
        # Cria o menu.
        menuCadastro = self.addMenu('&Cadastros')
        # Cria o Item de Menu e atribui a ação a self.actCadPessoa
        # self.addMenuItem(nome do menu, Classe, 'Nome da tela')
        self.actCadPessoa = self.addMenuItem(menuCadastro,
                                                frameCadPessoa.FrameCadPessoa,
                                                u'Cadastro de Pessoas')

        # Create menu
        menuGerencia = self.addMenu(u'&Gerências')
        # Create menu item
        self.actGerenciaPessoa = self.addMenuItem(menuGerencia,
                                                    frameGerenciaPessoas.FrameGerenciaPessoas,
                                                    u'Gerência de Pessoas')

```

Insert the action:

```

insert into ACAO (ID_ACAO, ID_ACAO_PAI, NOME, CAPTION,
                DATA_CADASTRO, NRO_EXECUCAO, ARQUIVO, ATALHO)
values (
    gen_id(ACAO_SEQ, 1), --- ID da ação, buscar da sequence
    1, -- ID da ação pai, neste caso a ação ROOT
    'FRAMEGERENCIAPESSOAS', -- O nome do frame em caixa alta
    'Gerência de Pessoas', -- Título para a tela.
    current_date, -- Data de cadastro da ação
    0, -- Nº de execuções
    null, -- Arquivo (Não utilizado neste exemplo)
    null) -- Atalho (Não utilizado neste exemplo)

```

Your form is ready. Must be like this:

Nova Aplicação Utilizando pyHed

Cadastros Gerências Relatórios

Gerência de Pessoas

Adicionar condição Remover condição Pesquisar Fechar

Campo Operação Condição

Nome Contendo a

Filtros informados

Exibindo todos os registros

Código Pessoa	Nome	Sexo	Data Nascimento
1 789	Vinicius Marconi	Masculino	4/3/1987
2 27	João da Silva Gulari	Masculino	2/5/1960
3 28	Rodrigo Jesuino da Silva	Masculino	2/8/2003
4 29	José da Silva	Masculino	30/6/2009

Congratulations! You build your form.