

Começando com o pyHed.

1º Passo - Ambiente:

Para utilizar o pyHed é necessário configurar seu ambiente com:

- [Python 2.5](#): Interpretador python;
- [pyQt4](#): Biblioteca de interface gráfica;
- [sqlAlchemy 0.5.4](#): ORM;
 - [cx_Oracle](#): Driver para oracle;
 - [KinterbasDb](#): Driver para firebird;
 - **Importante**: Se você tiver problemas ao conectar com o KinterbasDb entre em contato conosco.
- [ReportLab](#): Biblioteca para geração de PDF.

Por fim adicione o diretório do pyHed ao PYTHONPATH. Digamos que o pyHed esteja no diretório X:\ a string do PYTHONPATH deve ser a seguinte "PYTHONPATH;X:\".

Pronto! Seu ambiente está configurado.

2º Passo – Banco de Dados:

O banco de dados de uma aplicação que utiliza o pyHed deve ter uma estrutura básica. Como tabela de ação, perfil, usuário e etc. Utilize o script ' BasicDatabase.sql ' que está disponível para download no site do pyHed para criar o seu banco de dados básico.

3º Passo - Estrutura de diretórios, classes e arquivos básicos:

Assumiremos como diretório padrão o X:\ adicionado ao PYTHONPATH anteriormente. Dentro do X: deve estar o pyHed e o seu projeto. Por padrão o seu projeto deve ter a seguinte estrutura de diretórios e alguns arquivos básico. Neste documento não vamos detalhar os arquivos básicos, você poderá vê-los no exemplo de aplicação contida nesta documentação. A seguir a estrutura básica.

- **novaApp**: Pasta do projeto
 - **dbTables**: Aqui o arquivo que conterà o mapeamento do banco de dados e algumas funções.
 - **appDbTables.py**: Neste deve conter a função de autenticação no sistema e gerencia de perfis do usuário. Não é necessário nomeá-lo desta forma.
 - **desktop**: Aqui os arquivos do projeto e a pasta de imagens.
 - **img**: Diretorio onde ficarão as imagens do projeto;
 - **novaApp.py** : Arquivo responsável pela execução;
 - **novaAppMain.py**: Classe principal do sistema, deve herdar o frameCustomMain, nela serão registrados os frames do sistema, montados os menus e demais funcionalidades da home do projeto.
 - **frameLogin** :Monta a tela de login e faz a validação. Deve herdar o frameCustomLogin.
 - **appConsts**: Arquivo que carregará constantes do sistema. Não é obrigatória para o funcionamento do mesmo. Porém torna mais fácil o acesso a constantes.
 - **config.ini**: Arquivo que contém as constantes do sistema.

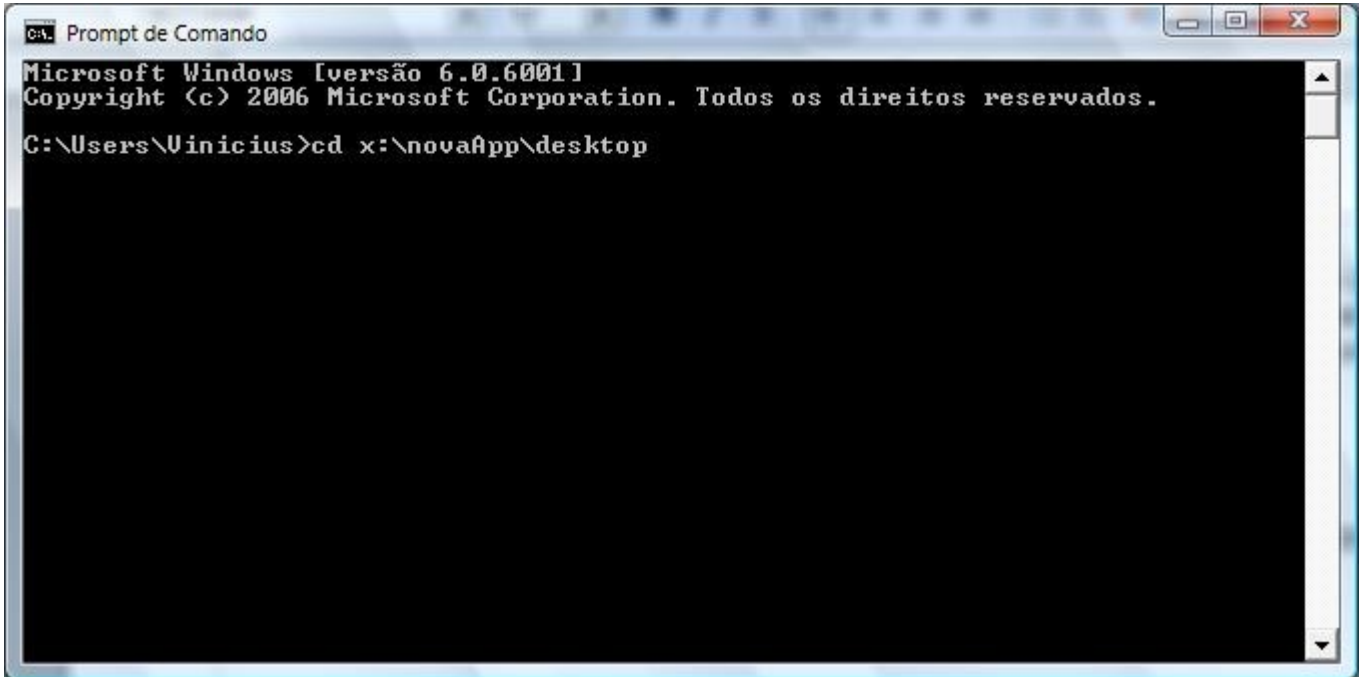
Esta estrutura é importante pois você pode criar uma versão web do seu projeto usando

o coirmão pyHew e utilizar o mesmo controle de acesso e mapeamento de tabelas do desktop.

Para ver o conteúdo dos arquivos veja o exemplo de aplicação básica contido na documentação.

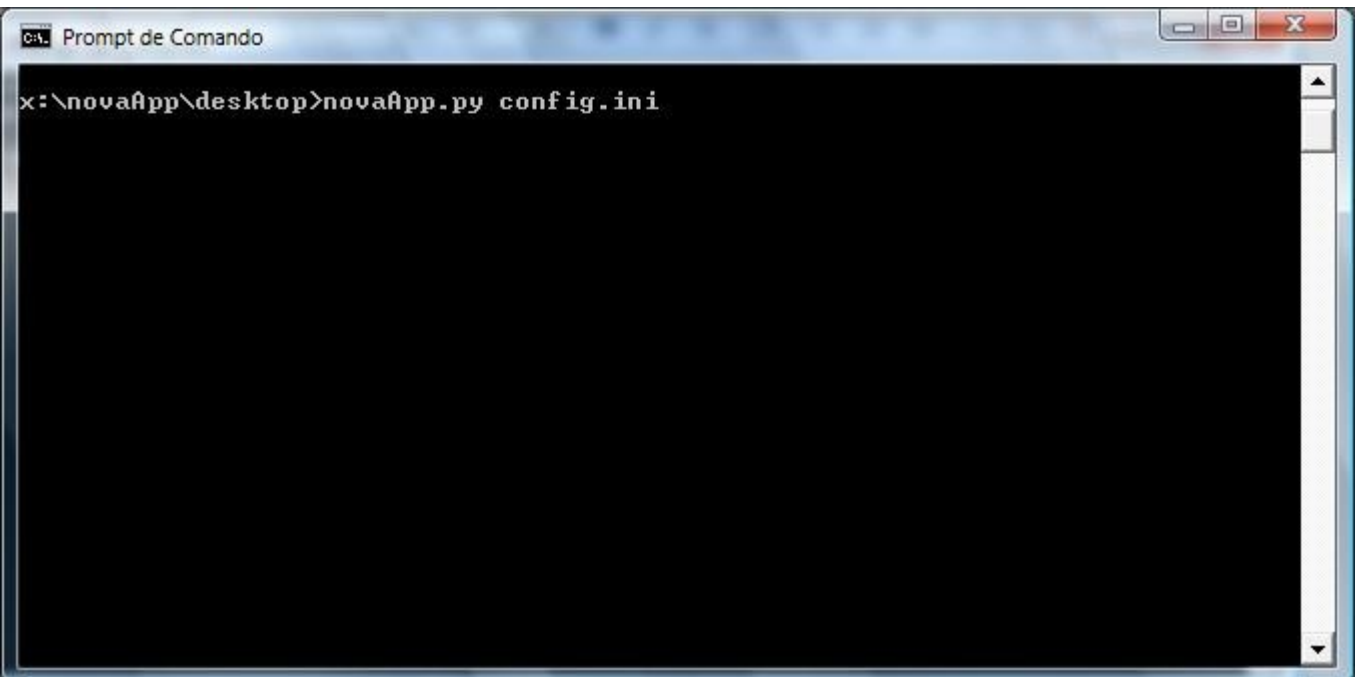
4° Passo - Executando a aplicação:

Para executar a aplicação abra o prompt de comando vá até o diretório que está o arquivo responsável pela execução, no nosso caso novaApp.py.



```
C:\Users\Unicius>cd x:\novaApp\desktop
```

A seguir execute >> novaApp.py config.ini



```
x:\novaApp\desktop>novaApp.py config.ini
```

novaApp.py: Arquivo responsável pela execução.

config.ini: Argumento necessário para execução.

Parabéns! Você aprendeu a usar o pyHed.