Pacote Policy

Exercício 1: Criando o pacote

(nos slides)

Utilize o esqueleto plone do ZopeSkel para criar o pacote treinamento.policy.

```
../bin/zopeskel plone treinamento.policy

Register Profile: True
Todas as outras perguntas aceite o padrão ([Enter])
```

Adicione ao pacote (eggs e develop de buildout.cfg)

Execute o buildout develop.cfg

Reinicie a instância e verifique se o pacote está disponível para ser instalado.

Exercício 3: Configurando dependências

No arquivo treinamento.policy/setup.py, adicione os pacotes treinamento.theme e treinamento.content à lista **install_requires**.

Deixe o arquivo **profiles/default/metadata.xml** da seguinte forma:

```
<?xml version="1.0"?>
<metadata>
     <version>1000</version>
     <dependencies>
         <dependency>profile-treinamento.theme:default</dependency>
         <dependency>profile-treinamento.content:default</dependency>
    </dependencies>
</metadata>
```

Reinicie a instância, instale o pacote **treinamento.policy** em um novo site e verifique o resultado.

** FALAR SOBRE GENERIC SETUP

** FALAR SOBRE O EXPORT DO PORTAL_SETUP

Exercício 4: Modificando o workflow padrão

Vamos colocar o intranet_workflow como workflow padrão do site.

Primeiro, faça essa modificação na ZMI, em portal_workflow (procure pelo campo Default).

Agora, na aba Export do portal_setup, exporte o Workflow Tool.

Pegue o arquivo workflows.xml e coloque em seu profiles/default.

Remova tudo que for desnecessário.

Exercício 5: Criando um GenericSetup Step

Vamos criar um GenericSetup Step que irá executar uma função chamada setupVarious dentro de um módulo chamado setuphandlers.py.

No **configure.zcml** do pacote treinamento.policy:

```
    ...

    <genericsetup:importStep
        name="_various"
         title="Projeto Alfa: miscellaneous import steps"
         description=" "
         handler="treinamento.policy.setuphandlers.setupVarious">
    </genericsetup:importStep>

</configure>
```

Agora criamos o módulo **setuphandlers.py** dentro de treinamento.policy/treinamento/policy:

```
# coding=utf-8

def setupVarious(context):

    if context.readDataFile('treinamento.policy-various.txt') is None:
        return

    import ipdb; ipdb.set_trace()
```

Crie o marker file **treinamento.policy-various.txt** dentro de profiles/default.

Reinicie sua instância, reinstale o pacote treinamento.policy e verifique se o console para no debugger.

Exercício 7: Removendo pastas

Neste exercício vamos remover as pastas **Usuários** e **Eventos** que são criadas por padrão pelo Plone.

Para isso, vamos utilizar um pacote chamado **plone.api**, vamos instalá-lo no buildout.cfg (opção eggs) e rodar o buildout (develop.cfg).

Crie uma função no setuphandlers.py chamada **remove_default_objects()**, ela utilizará o método **api.content.delete(objeto)**, que recebe o objeto a ser removido como parâmetro.

Criada a função, chame a mesma na função setupVarious.

Teste e verifique sua solução.

Obs: Tente usar o @@ipdb para sanar suas dúvidas, por exemplo, como consigo acessar as pastas Eventos (o id é events) e Usuários (id = Members)?

Obs1: Você precisa importar a api do plone (**from plone import api**)

Obs2: Para pegar a raiz do portal, utilize **portal = api.portal.get()**.

Exercício 8: Adicionando conteúdo

Neste exercício vamos criar a estrutura inicial de pastas do nosso projeto. Na raiz, teremos as seguintes pastas:

```
Notícias (já é criado por padrão)
Programação de Inverno
Programação de Verão
```

Portanto, precisamos criar dois objetos do tipo Programa.

Vamos utilizar o método api.content.create:

```
api.content.create(
    type='treinamento.content.programa',
    title=u'Programação de Inverno',
    container=portal)
```

Crie uma função chamada **create_programs()** e chame a mesma em setupVarious.

Reinicie a instância (ou utilize **@@reload**), reinstale o pacote e verfique que tudo está funcionando.

Exercício 9: Criando um grupo

Vamos criar um grupo da seguinte forma:

```
group = api.group.create(
    groupname='organizadores',
    title='Organizadores',
    description='Organizadores de Conferencias',
    roles=['Reader', 'Editor', 'Contributor'],
)
```

Crie uma função chamada **create_group** e a chame em setupVarious.

Exercício 10: Criando usuários

Defina a seguinte lista em setuphandlers.py:

```
USERS = [
    ['paulo', 'paulo@gmail.com', 'senha1'],
    ['maria', 'maria@gmail.com', 'senha2'],
    ['pedro', 'pedro@gmail.com', 'senha3'],
    ['jose', 'jose@gmail.com', 'senha4'],
]
```

Utilize o seguinte comando para criar um usuário:

```
api.user.create(
    username='noob',
    email='noob@plone.org',
    password='secret',
)
```

Crie uma função chamada **create_users()** que itera sobre a lista **USERS** para criar os usuários. Chame essa função em setupVarious.