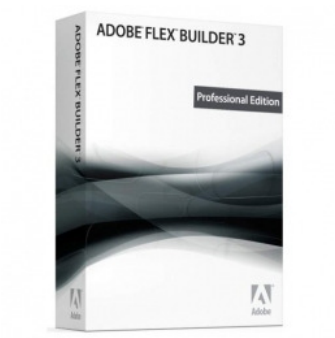
Salve salve!!!

Como havia prometido, começarei a postar alguns artigos mostrando como integrar Django e Flex! Antes de qualquer coisa, gostaria de dizer que, ainda existem vários pontos que são obscuros sobre este assunto, pois, a documentação do AMF que vamos utilizar não é muito boa, e erros que às vezes acontecem não são muito claros portanto, demora-se muito para compreender o que pode estar ocorrendo quando o "trem" não funciona. ;)

O que é Flex ?



O **Adobe Flex** (antes chamado de *Macromedia Flex* e depois rebatizado como *Adobe Flex* pela Adobe) é o nome de uma tecnologia lançada em Março de 2004 pela Macromedia, que suporta o desenvolvimento de aplicações ricas para a Internet, baseadas na plataforma do Macromedia Flash. A versão inicial possuia um SDK, um IDE uma integração com o J2EE também conhecido como Flex Data Services. Desde que a Adobe adquiriu a Macromedia em 2005, as versões subsequentes do Flex começaram a requerer uma licença para o Flex Data Services, que era inicialmente um produto separado e que posteriormente foi rebatizado como LiveCycle Data Services.

Em abril de 2007, a Adobe anuncia planos de abrir o código do Flex 3 SDK. O Adobe Flash Player, aplicativo pelo qual são visualizados as aplicações Flex, e o Flex Builder, a IDE utilizada para desenvolver aplicações Flex, continuam proprietárias e comerciais.

(Fonte: Wikipédia)

Bom, o Flex proporciona, de acordo com a criatividade do desenvolvedor ou designer, um resultado final fantástico!

Basta entrar no site da Adobe e procurar alguns exemplos para você ver que realmente, é "chick d +++" ;)

Também, para desenvolver, você irá precisar do Adobe Flex Builder. No site da Adobe, você pode pegar uma versão trial dele ;)

Bom, para utilizar o Django e o Flex, fizemos vários testes aqui e, passamos por diversos tipos de problemas e diversos tipos de dificuldades!

O primeiro, é parecido com um "problema" que temos no javascript.

Não conseguimos acesso a base de dados sem auxílio de um "intermediador".
Mas isto não é realmente um problema pois, se você pensar direitinho, tanto o javascript quanto o Flex (que exporta as aplicações como um arquivo .swf) rodam do lado do cliente.

Existem estes “intermediadores” para várias linguagens, mais, fãs do Django, queríamos utilizar a facilidade desta framework aliada a rica inteface que é gerada pelo Flex.

Outro ponto a se salientar é que a documentação também não ajuda muito!

Todos os amfs para python não são bem documentados, e a Adobe não tem uma documentação ou tutorial, ou até mesmo, alguma dica a respeito da dupla Flex + Python.

Tentamos, PyAMF, AMFast e DjangoAMF.

Com o último, obtivemos sucesso!! Apesar da documentação dele também ser meio “obscura”. Não quero ficar reclamando demais também pois, a solução é “di grátis”, e eu entendo perfeitamente que o autor tenha outras coisas pra fazer.

Para baixar a última versão do DjangoAMF acesse
http://djangoamf.sourceforge.jp/index.php?DjangoAMF_en .

Aqui, você encontra o manual em inglês... Também tem a versão em japonês (Salvo engano)... Fica a seu critério... rsrsrs
http://djangoamf.sourceforge.jp/index.php?UserManual_en

Feito o download, descompacte o conteúdo do arquivo em uma pasta e digite o comando:

“***python setup.py install***”

No arquivo settings.py, adicione o middleware que será o responsável pela tradução dos dados do python para flex.

'***amf.django.middleware.AMFMiddleware***',

No fim do arquivo, adicione as variáveis:
AMF_GATEWAY_PATH - Geralmente é /gateway/ ;)

Agora, vamos criar um app básico Django, para que este retorne ao Flex os dados à serem exibidos:

“***manage.py startapp teste***”

No seu arquivo views.py crie a seguinte função:

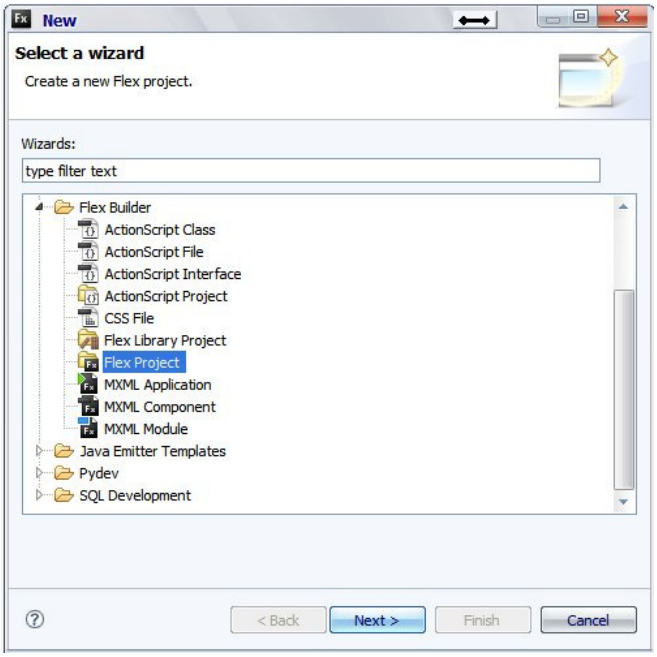
“***import amf, amf.django***
import datetime

def horaAgora(request):
    return datetime.datetime.now()”

Ao seu arquivo urls.py, adicione ao seu urlpatterns o trecho abaixo:
“***(r'^gateway/teste/(.*)', 'amf.django.views', {'views': 'teste.views'}),***”
Não se preocupe com as urls, pois, quem irá tratá-las será o middleware do DjangoAMF.

Agora, crie um novo projeto dentro do FlexBuilder ;)

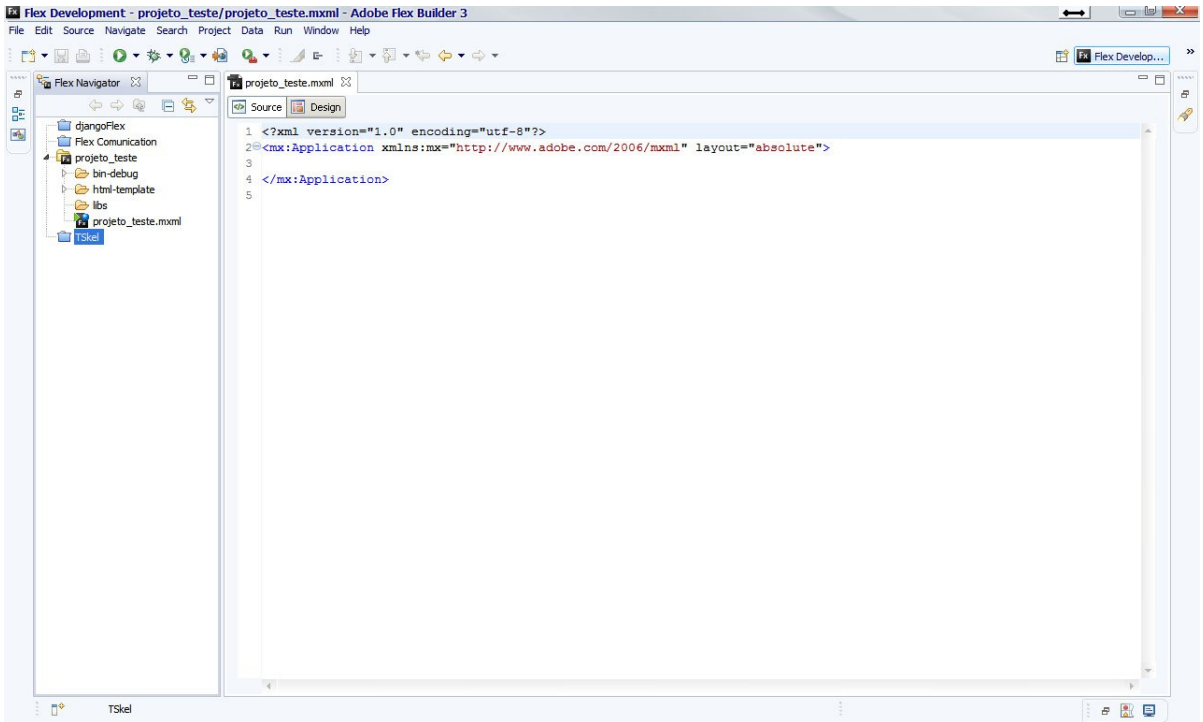
“File” -> “New” -> “Flex Project”



Agora, digite o nome do projeto, e configure o workspace ;)

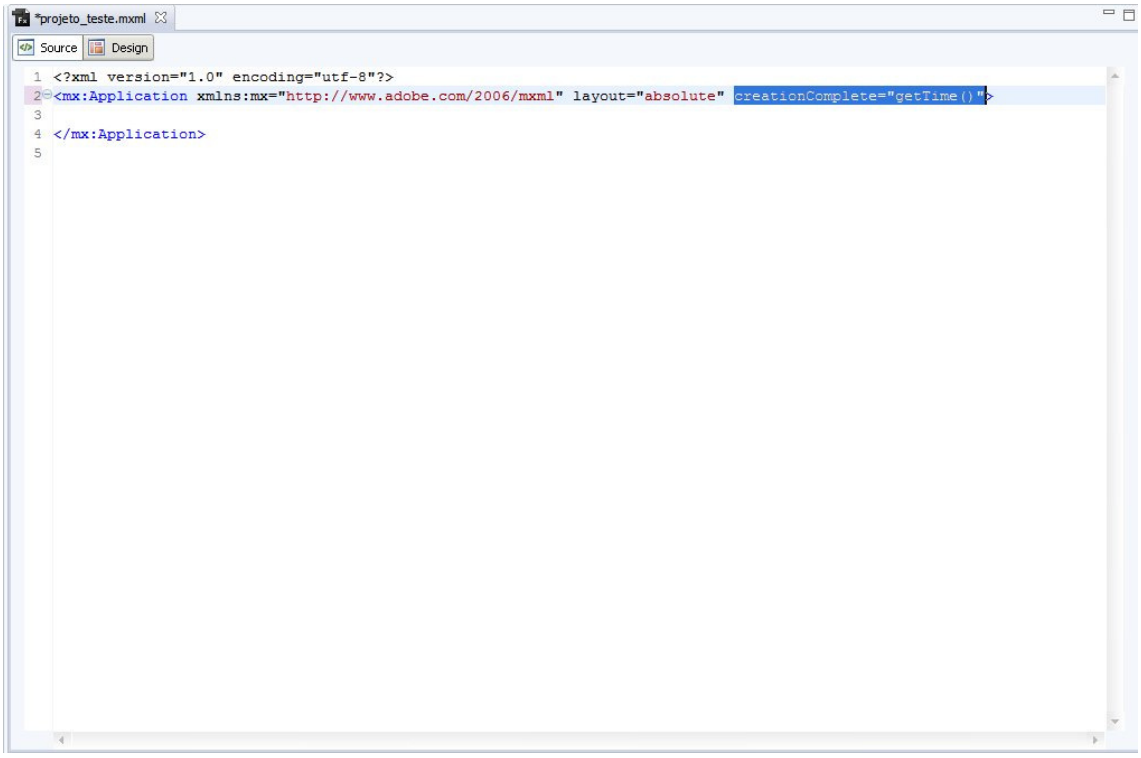
Depois é dar next... Na próxima imagem eu retiro o main src para que o arquivo MXML fique na raiz do projeto. Faça como você achar melhor ;)

Aqui está nosso projeto...



Agora, vamos começar a brincar ;)

Na nossa tag <mx: Application /> vamos adicionar uma função que será chamada assim que terminar o carregamento do Flash ...



Agora, vamos criar a bendita função para pegar a hora diretamente do Django...

*projeto_teste.mxml

Source | Design

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute" creationComplete="getTime()">
<mx:Script>
    <![CDATA[
        import mx.controls.Alert; //Para gerar o alertbox
        import mx.rpc.events.ResultEvent; // Gerenciar os resultados vindos do Django.
        import mx.rpc.events.FaultEvent; // Gerenciar os possíveis erros ;)
        import mx.rpc.remoting.mxml.RemoteObject; // Responsável pela comunicação...

        private function getTime():void
        {
            var ro: RemoteObject;
            ro = new RemoteObject('horaAgora');
            ro.endpoint = 'http://localhost:8000/gateway/';
            ro.addEventListener(FaultEvent.FAULT, getErro);
            ro.addEventListener(ResultEvent.RESULT, showResult);
            ro.getOperation('horaAgora').send();
        }

    ]]>
</mx:Script>
</mx:Application>
```

Mas ainda estão fatando as nossas funções para tratar os resultados e os possíveis erros...

Vamos lá!

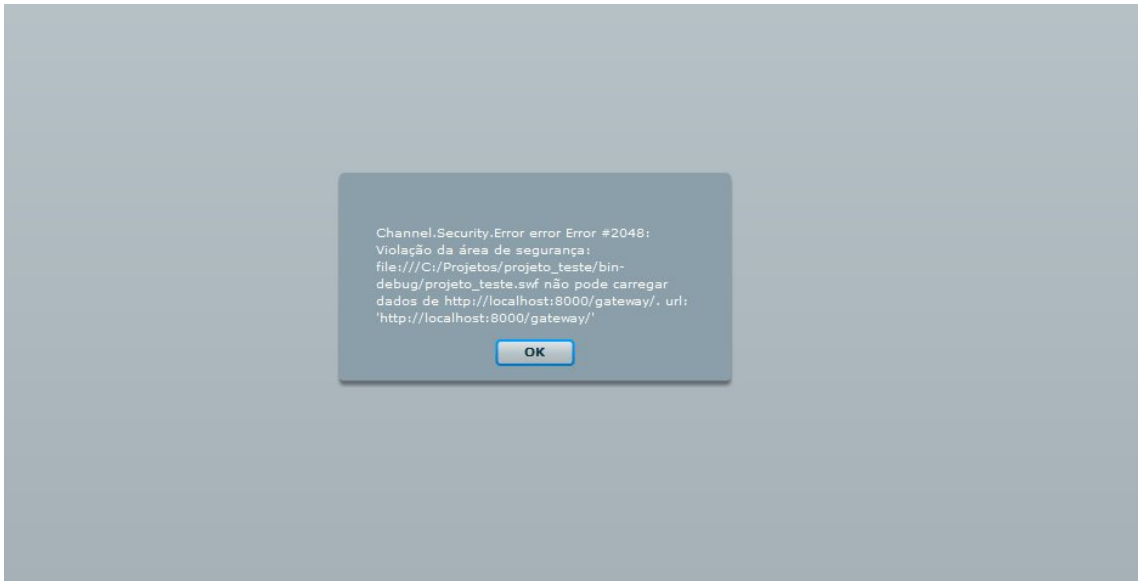
Faça as funções getErro e showResult ...

Source | Design

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute" creationComplete="getTime()">
<mx:Script>
    <![CDATA[
        import mx.controls.Alert; //Para gerar o alertbox
        import mx.rpc.events.ResultEvent; // Gerenciar os resultados vindos do Django.
        import mx.rpc.events.FaultEvent; // Gerenciar os possíveis erros ;)
        import mx.rpc.remoting.mxml.RemoteObject; // Responsável pela comunicação...


        private function getTime():void
        {
            var ro: RemoteObject;
            ro = new RemoteObject('teste'); //nome da aplicação
            ro.endpoint = 'http://localhost:8000/gateway/'; //url do server
            ro.addEventListener(FaultEvent.FAULT, getErro); //Caso ocorra algum erro...
            ro.addEventListener(ResultEvent.RESULT, showResult); //Chama a função para tratar o result ;)
            ro.getOperation('horaAgora').send(); //nome da função (view)
        }

        public function getErro(ro:FaultEvent):void
        {
            Alert.show(ro.fault.faultDetail);
        }

        public function showResult(ro: ResultEvent):void
        {
            Alert.show("A hora é " + ro.result);
        }

    ]]>
</mx:Script>
</mx:Application>
```

Salvando, agora é mandar rodar! Um detalhe... Enquanto eu redigia e testava o tutorial, experimentei um problema que não havia experimentado da primeira vez! E não encontrei solução (AINDA) quando era utilizado por padrão o browser Firefox...

O erro é este:



Já procurei no amiGoogle e, das várias possíveis soluções encontradas, nenhuma resolveu o problema! Vale lembrar que no outro projeto que estou “brincando” não tenho problema algum e tecnicamente é a mesmíssima coisa !

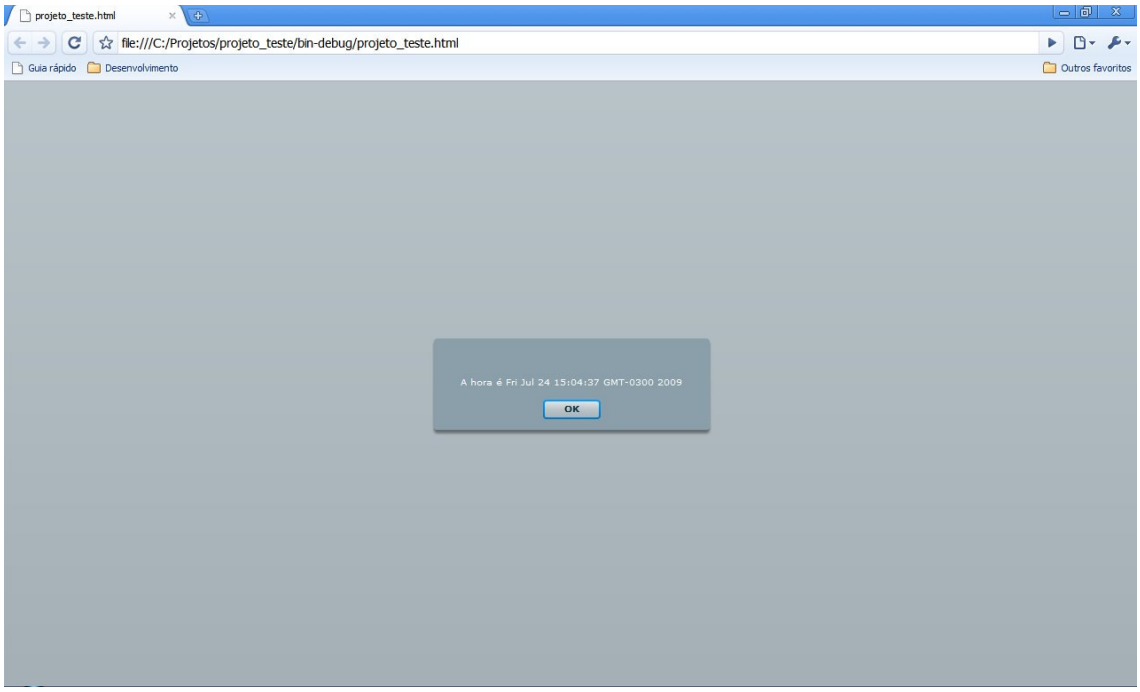
Bom, detalhes à parte, vamos prosseguir...

Testando no Chrome, Internet Explorer, Safari tudo funcionou como o esperado. ;)

Ainda vou descobrir o que foi que ocasionou o erro e em breve postarei para vocês!

O resultado é este: (Vale lembrar que não mechemos em nada nos estilos de botão, caixas de diálogos etc...)

Ficou assim:



Bom... Por enquanto é só... Agora você já pode começar a quebrar a cabeça!

Estou terminando de fazer um que utiliza um model, e que volte os dados do banco de dados...

Assim que eu terminar, estarei disponibilizando um novo tutorial...

Vale a pena ler a documentação, por mais fraca que ela seja, algumas coisas acabam de fato ajudando ;)

Também, estou disponibilizando o código fonte do projeto para que você importe para o seu Flex ;)

Abraços!!

Fred Chevitarese – GNU/Linux

http://chevitarese.wordpress.com