Primeiros Passos com Pentaho Data Integration (Kettle)

Licença

Autor

María Carina Roldán
Membro da comunidade Pentaho
email: maria.carina.roldan@gmail.com

Informação do documento

Primeiros Passos com Pentaho Data Integration (Kettle)
Data de Publicação: Julho, 2008

Prefácio

Este documento é uma introdução ao PDI (Pentaho Data Integration) ou Kettle, a ferramenta de integração do suíte de Pentaho.
O texto é desenvolvido baseado no exemplo "Oi, Mundo", e explica detalhadamente os conceitos fundamentais da ferramenta.

O tutorial está destinado principalmente a aqueles que nunca utilizaram o PDI, porém também pode ser útil para quem já começou a explora-la, dado que esclarece alguns conceitos muito presentes no forum do PDI, como por exemplo o uso de variáveis.

O texto está organizado da seguinte maneira:
✓**Introdução** - Em primeiro lugar se introduz a ferramenta, e dão-se indicações para sua instalação.
✓**Oi, mundo** – Este é o primeiro exemplo do tutorial, com indicações passo a passo.
✓**Oi mundo mais uma vez** – É uma versão completa e melhorada do exemplo, onde se ampliam os conceitos vistos.

No final do documento você verá algumas dicas, como também endereços na internet para levar em conta quando trabalhar com o PDI.

Terminologia

Para as explicações do tutorial se utiliza a versão da ferramenta em inglês. Por tanto, para você não ficar confundido, a terminologia específica do PDI não está traduzida. Estas são as principais palavras que você verá em inglês sendo elas as mais utilizadas na ferramenta:

Transformation	transformação
Job	trabalho
Step	passo
Job entry	entrada de trabalho
Hop	conexão, ligação

Você verá as definições destas palavras à medida que você utiliza-las.

Conteúdos

Introdução

PDI (antes Kettle) é o componente de Pentaho responsável dos processos de extração, transformação e carregamento ou ETL (pela sigla em inglês). Embora o uso mais freqüente de toda ferramenta ETL, incluindo o PDI, é a carga de datawarehouses, PDI pode ser utilizado também para outros propósitos:

-Migração de dados entre bancos de dados ou aplicações
-Exportação de dados para arquivos texto
-Carga massiva de informação em bancos de dados
-Correção de erros e inconsistências nos dados
-Integração de aplicações
-etc.

PDI é fácil de utilizar. Todos os processos são criados num ambiente gráfico onde você especifica o quê fazer sem necessidade de escrever código para indicar como fazê-lo. É por isso que se diz que a solução está orientada a meta-dado.

PDI pode ser utilizado de forma independente, ou integrado com os outros componentes do suíte de Pentaho. Como ferramenta ETL, é a mais popular entre as de código aberto. PDI suporta ampla variedade de formatos de entradas e saídas, incluindo arquivos texto, planilhas, e conexão com gerenciadores de bancos de dados tanto comerciais quanto abertas. Além das entradas e saídas possui uma ampla gama de operações de transformação que permitem manipular dados sem limitações.

Neste tutorial você vai ver como é fácil trabalhar com PDI. Através da construção de um simples "Oi, Mundo" você vai conhecer as características mais utilizadas da ferramenta, e sem dúvida vai se animar a construir seu próprio projeto.

Instalação

O primeiro Step para trabalhar com PDI é instalar a ferramenta.
PDI pode ser baixado de http://community.pentaho.com/sourceforge/

No momento en que este tutorial foi criado, a versão liberada mais nova de PDI era a 3.0.3. O arquivo que você tem que baixar é `Kettle-3.0.3.GA-nnnn.zip`

PDI no precisa instalação (a não ser que você baixe a versão .exe).
O único pré-requisito para poder trabalhar com PDI é ter instalada a JRE 5.0 ou superior. A JRE pode ser baixada de http://www.javasoft.com/

Depois que você verificar este pré-requisito, simplesmente tem que descompactar o arquivo `zip` numa pasta a eleição. No caso de estar trabalhando em ambientes como Unix, Linux, Solaris, MacOS, é necessário tornar os scripts executáveis. Se este é seu caso, execute as instruções a seguir: (supondo que você descompactou na pasta `Kettle`)

```
cd Kettle
chmod +x *.sh
```

Você já tem tudo o que precisar para começar trabalhar.

Conhecendo a Spoon

Quando você vê as telas do **PDI**, na verdade o que você está vendo são telas da **Spoon**, a interface gráfica que permite desenhar e testar todos os processos de PDI. Os outros componentes de PDI são utilizados para executar os processos desenhados com a Spoon, e são executados de um prompt como verá mais adiante.

Catálogo e Arquivos

Na Spoon você cria **Jobs** e **Transformations**. Para salvar os Jobs e as Transformations, PDI permite escolher entre duas formas de armazenamento:

- Catálogo em Banco de Dados (Repository)
- Arquivos

Se você escolher o catálogo, você tem que criá-lo a primeira vez que executa a Spoon.
Se você escolher o método de armazenamento em arquivo, os Jobs são salvados em disco com extensão `kjb`, e as Transformations, com extensão `ktr`.

Neste tutorial você vai trabalhar com o segundo método.

Iniciando a Interface Gráfica

Para começar, inicie a Spoon executando:

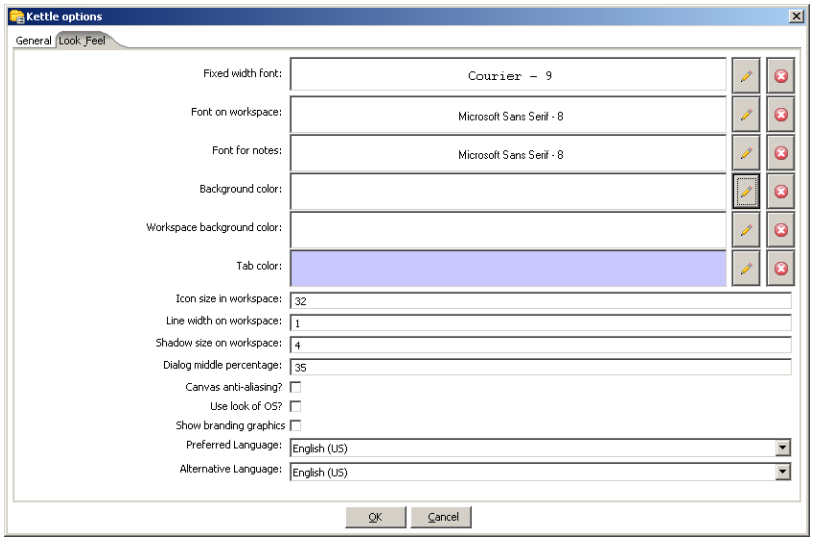
`Spoon.bat` em Windows

ou

`Spoon.sh` em outras plataformas como Unix, Linux, ...

Assim que a Spoon é iniciado, aparece uma caixa solicitando os dados de conexão do catálogo. Como ja foi dito, neste tutorial você trabalhará com arquivos. Por tanto, selecione o botão **`No Repository`**.

Inicialmente você verá uma tela de boas-vindas. Dado que é a primeira vez que você utiliza a Spoon, provavelmente você queira mudar algumas características visuais da interface. Para isso selecione `Options...` dentro do menu `Edit`. Aparecerá uma janela onde você poderá mudar vários aspectos gerais e de visualização, em particular o idioma de preferência. Para que a Spoon reconheça as mudanças, será necessário que você o reiniciar.
Até agora, infelizmente, a tradução ao português está muito pobre. Por isso o tutorial será explicado com a interface e a terminologia em inglês tal como foi dito no prefácio.



Agora sim você já está preparado para dar as boas-vindas ao "Oi, mundo".

Oi, Mundo!

"Oi, Mundo" será sua primeira experiência com PDI. Apesar de ser um exemplo simples você vai ver varias das características fundamentais de PDI:

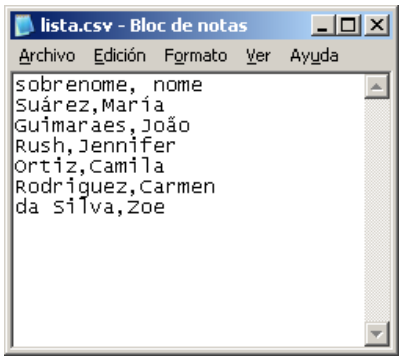
-Trabalho com a ferramenta Spoon
-Transformations
-Steps e Hops
-Variáveis predefinidas
-Pré-visualização e execução utilizando a interface gráfica
-Execução no prompt de comando com a ferramenta Pan.

Observe! Cada vez que você vir um quadro como este, tenha atenção, pois ele contem definições e conselhos importantes que serão úteis no dia a dia de trabalho com a ferramenta.

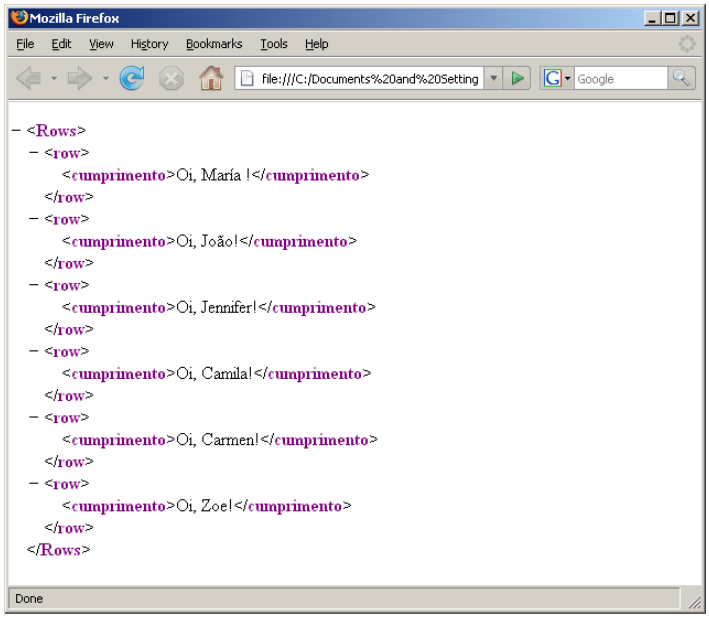
Exercício

Suponha que tem um arquivo `csv` contendo uma lista de pessoas, e quer gerar um arquivo `xml` que contenha saudações para todos os contatos da lista.

Se este fosse o conteúdo do arquivo de nomes de pessoas:



Esta seria a saída:



A geração do arquivo de saudações a partir do arquivo de nomes será a tarefa da sua primeira Transformation.

Uma **Transformation** é uma unidade de execução composta por ***Steps*** ligados por meio de ***Hops***. Estes Steps e Hops formam caminhos por onde os dados passam: entram, se transformam e saem. Por isso se diz que uma Transformation está orientada ao **fluxo de dados**.

Preparação do Ambiente

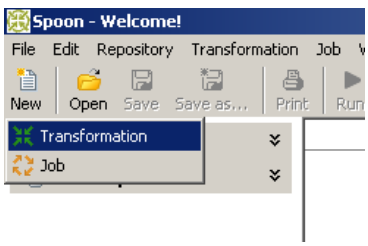
Antes de começar, crie a pasta `Tutorial` em um lugar a sua escolha. Aí você salvará todos os arquivos do tutorial.
Depois crie um arquivo como o visto na introdução deste exemplo, e salve-o com o nome `lista.csv`, na pasta criada.
Não é obrigatório que este arquivo exista, no entanto sua existência vai facilitar a configuração do primeiro Step da Transformation.

Passo a passo

Faça o exercício em três partes:

1)Crie a Transformation
2)Construa o esqueleto da Transformation utilizando Steps e Hops
3)Configure os Steps para especificar sua funcionalidade

1. Criação da Transformation



–Selecione **`New → Transformation`**. Uma alternativa é criar a Transformation desde o menu **`File → New → Transformation`** ou pressionar

<Ctrl-N>. Uma nova aba ficará disponível para trabalhar com a nova Transformation. Esta é a área de trabalho onde você colocará os Steps e os Hops.
–Selecione a opção **Transformation → Configuration**. Aparecerá uma janela onde você pode especificar propriedades gerais da nova Transformation. Neste caso só escreva um nome e uma descrição e pressione **OK**.

–Pressione o botão **Save**. Salve a Transformation na pasta `Tutorial` com o nome "`Oi`". Será criado o arquivo `Oi.ktr`.

2. Construção do Esqueleto da Transformation utilizando Steps e Hops

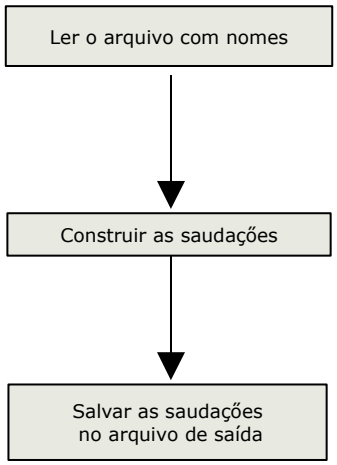
Um **Step** é a unidade mais atômica dentro de uma Transformation. Existe uma grande variedade de Steps disponível. Os Steps estam, agrupados em categorias como Input (entrada) o Output (saída), entre outras. Cada tipo de Step cumpre com uma funcionalidade específica, que vá desde ler um parâmetro hasta normalizar um conjunto de dados.

Um **Hop** é a representação gráfica do fluxo de dados entre dois Steps: um origem e um destino. A pilha de dados que passa por esse Hop constitui a Saída (**Output Data**) do Step origem e a Entrada (**Input Data**) do Step destino.

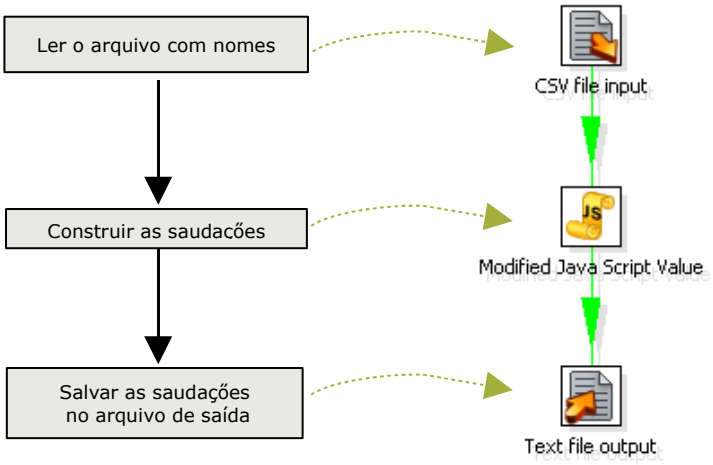
Um Hop tem só uma origem e um destino, porém de um Step pode sair mais de um Hop. Se esse é o caso, a Saída pode ser copiada a todos os Steps destino do Hop, ou pode ser distribuída entre esses Steps.

Também a um Step pode chegar mais de um Hop. Neste caso, o Step deve ter a capacidade de combinar as Entradas para gerar a Saída a partir delas.

Veja o que a Transformation tem que fazer:



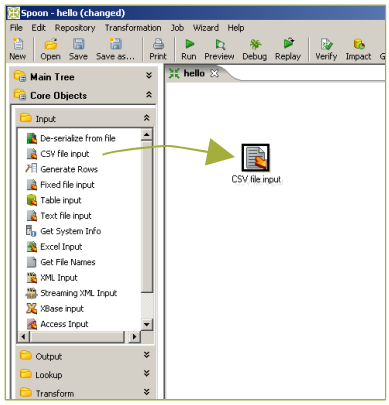
Para cada um destes itens você utilizará um Step diferente, como no seguinte diagrama:



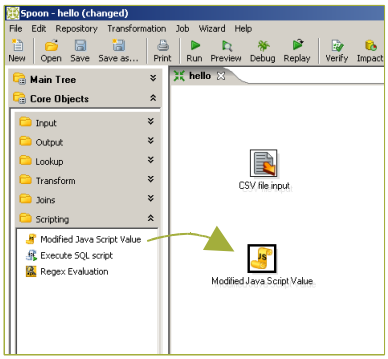
Leve em conta que a correspondência é um a um porque a Transformation é muito simples. Não sempre é assim!

Siga os seguintes passos:

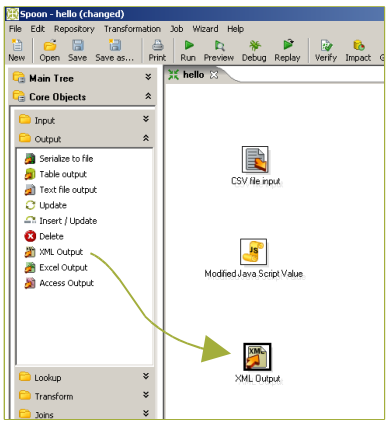
–Identifique dentro da paleta de Steps a categoria ***Input***. Clique no Step **CSV file input** e arraste-o à tela de trabalho.



–Identifique dentro da paleta a categoria ***Scripting***. Clique no Step **Modified Java Script Value** e arraste-o da mesma forma.



–Identifique dentro da paleta a categoria ***Output***. Clique no Step **XML Output** e arraste-o também à área de trabalho.



–Ligue **CSV file input** com **Modified Java Script Value** com um Hop: Clique no primeiro Step e com a tecla <Shift> pressionada, arraste o cursor em direção ao segundo Step. (o manual da Spoon explica outras maneiras para criar os Hops)
–Ligue **Modified Java Script Value** com **XML Output** da mesma maneira.

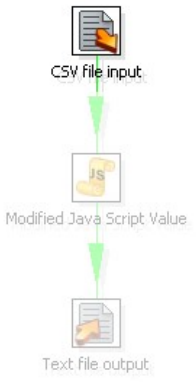
3. Configuração dos Steps para especificar sua funcionalidade

Todos os Steps têm uma **janela de configuração** associada. Estas janelas variam segundo a funcionalidade do Step e a categoria dentro da qual se encontra, entretanto todas têm em comum os seguintes dados:

Nome do Step (Step Name) é um nome representativo dentro da Transformation.
Descrição (Step Description) Permite acrescentar informação sobre o propósito do Step.

a) Configuração do Step CSV file input

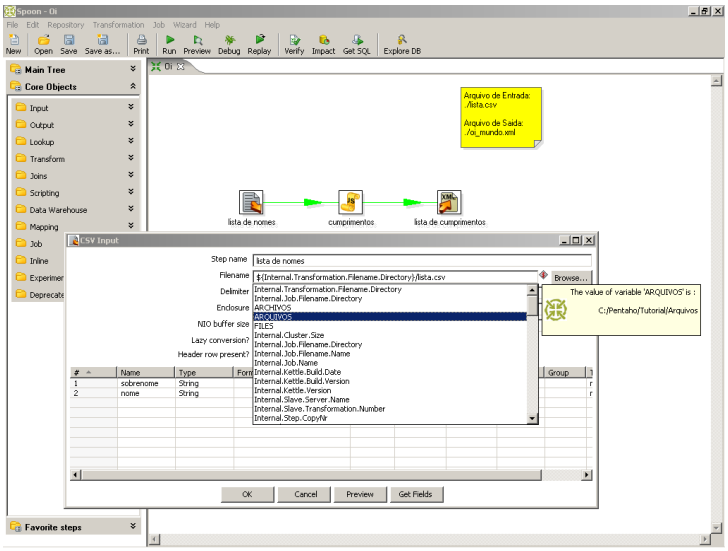
–Dê um duplo clique em cima do Step **CSV file input**. Aparecerá a janela de configuração deste tipo de Step. Aqui você vai indicar a localização, o formato e o conteúdo do arquivo de entrada.
–`Step name`: A primeira coisa que você deveria fazer *em qualquer Step que configure* é apagar o nome que é colocado pela ferramenta. Logo deveria colocar ao Step um nome mais representativo para a função que vá cumprir dentro da Transformation. Neste caso coloque "`lista de nomes`".
–`Filename:` Neste lugar você tem que indicar o nome e localização do arquivo de entrada. À direita do espaço destinado ao texto, você verá este símbolo: .



Este símbolo à direita de uma caixa de texto implica que nessa caixa além de escrever texto você pode utilizar **variáveis**. A variável pode se escrever manualmente como **${nome_da_variável}** ou ser selecionada na janela de

variáveis disponíveis, à qual se acesa pressionando **<Ctrl-Espaço>.** A janela mostra tanto as variáveis predefinidas pelo PDI como aquelas definidas pelo usuário.

Como ainda você não criou nenhuma variável, só vai ver as variáveis predefinidas por PDI como se vê aqui:



Entre estas, selecione

```
${Internal.Transformation.Filename.Directory}
```

Ao lado do nome da variável escreva o nome do arquivo que você criou. O texto ficará assim:

```
${Internal.Transformation.Filename.Directory}/lista.csv
```

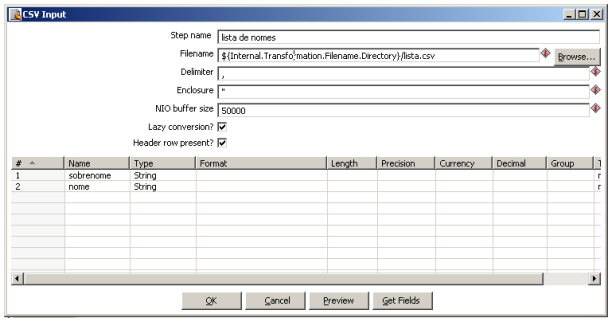
Na execução, a variável será substituída pelo seu valor, que será o caminho de armazenamento da Transformation no disco. A Transformation procurará o arquivo `lista.csv` nesse caminho.

–**Get Fields:** Pressione este botão para que a aplicação carregue à tabela com a lista dos nomes das colunas do arquivo. Observe que, se você não indicar o contrário, este Step considerará que o arquivo tem cabeçalhos (**Header row present** está selecionado).

O botão **Get Fields** se encontra na janela de configuração de muitos tipos de Steps. Sua função é carregar uma tabela com dados provenientes de fontes externas (por exemplo de um arquivo texto), o provenientes de Steps anteriores. Mesmo que os campos possam ser escritos manualmente, este botão serve como atalho quando você tem muitos campos disponíveis para utilizar.

A tela tem agora os nomes das colunas do arquivo de nomes: `sobrenome` e `nome`.

Já quase terminou com este Step. A tela agora deveria se ver assim:



–**Preview Rows:** Simplesmente para você se segurar que o arquivo será lido corretamente, pressione o botão de pré-visualização. Aparecerá uma janela mostrando dados do arquivo: tantos quantos você tenha solicitado (a não ser que a quantidade de dados do arquivo seja menor).

–**OK:** Pressione este botão, e já está configurado o Step **CSV file input**.

b) Configuração do Step Modified Java Script Value

–Dê um duplo clique em cima do Step **Modified Java Script Value**. Aparecerá a janela de configuração deste tipo de Step, certamente, muito diferente da tela que você viu no Step anterior. Neste caso, o Step permite criar código javascript, e você vai utilizá-lo para construir a mensagem "`Oi, `" concatenado com cada um dos nomes do arquivo.

–**Step name:** Coloque neste Step o nome "`Construção da mensagem`".

–**Código javascript:** Aqui você pode escrever código javascript.
À esquerda tem uma árvore com funções disponíveis que ajudam à codificação. Em particular, as últimas folhas da árvore correspondem aos campos de entrada (`Input Fields`) e de saída (`Output Fields`) disponíveis para utilizar no código. No exercício você tem dois campos: `sobrenome` e `nome`.

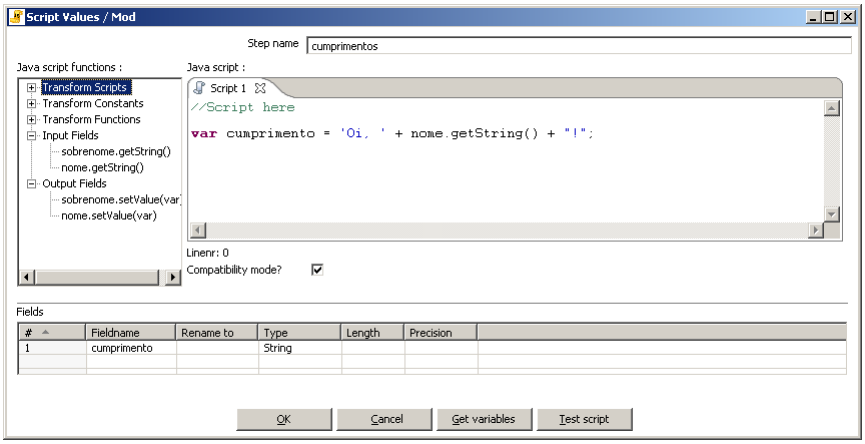
Escreva o seguinte código:

```
var saudação = 'Oi, ' + nome.getString() + "!";
```

O texto `nome.getString()` pode se escrever manualmente, ou fazendo duplo clique no mesmo texto dentro de "Input Fields" na árvore de funções.

–**Fields:** na parte inferior você pode colocar dados criados no código para serem agregados aos campos de saída. Neste caso, tem criado a variável de javascript `saudação`. (No se confunda com variáveis de PDI!) Dado que você precisa enviá-la ao arquivo de saída, agregue-a nesta tabela.

Já tem configurado o Step. Este é o resultado:



Para seu conhecimento! "modified" não é um adjetivo para "javascript", mas para o Step mesmo. Não temos aqui uma variante de javascript. O Step é o que está modificado: É uma versão melhorada do Step "Valor Javascript" que existia em versões anteriores de PDI.

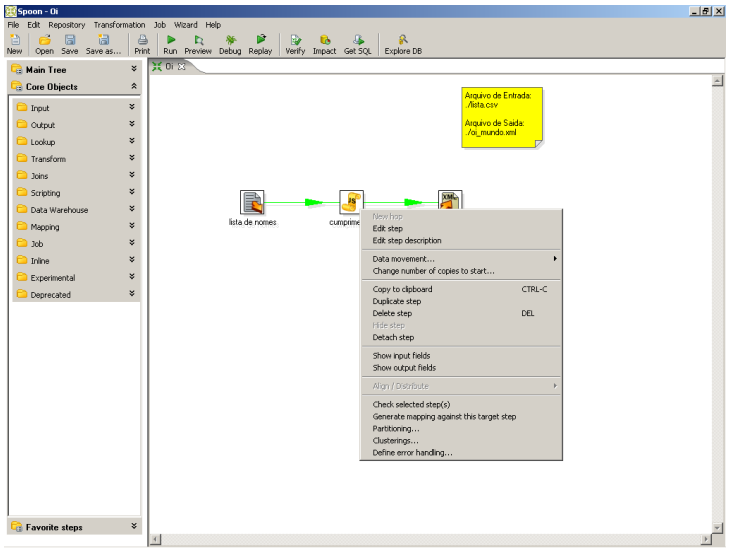
–**OK:** Pressione este botão, e já está configurado o Step **Modified Java Script Value**.

–Selecione o Step que você acaba de configurar. Para comprovar que o novo dado sairá deste Step, você vai ver agora os campos de entrada e de saída.

Campos de Entrada: são as colunas de dados que chegam a um Step.
Campos de Saída: são as colunas de dados que saem de um Step.

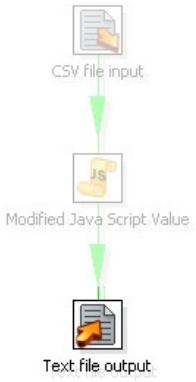
Tem Steps que simplesmente transformam os dados de entrada. Nestes casos os campos de entrada e de saída coincidem.
Mas não sempre é assim:
Tem Steps que acrescentam dados aos Campos de Saída. Exemplo: "Calculator"
Tem outros Steps que filtram o combinam dados, fazendo do que a saída tenha menos dados do que a entrada: Exemplo: "Group by"

Pressione o botão direito em cima do Step. Aparecerá o seguinte menu contextual:



–Selecione **`Show Input Fields`**. Você vai ver que os campos de entrada são os que vêm do Step **CSV file input**: `nome` e `sobrenome`.
–Selecione **`Show Output Fields`**. Você vai ver que esses dados foram acrescentados com o novo dado: `saudação`.

c) Configuração do Step XML Output



–Dê um duplo clique em cima do Step **XML Output**. Aparecerá a janela de configuração deste tipo de Step. Aqui você vai indicar o nome e localização do arquivo de saída, e quais os dados que você quer incluir no mesmo: poderiam ser tudos o só uma parte dos dados que chegam a este Step.

–**`Step name:`** Coloque para este Step o nome "`lista de saudações`".

–**`File:`** No lugar destinado ao nome do arquivo, escreva:

```
${Internal.Transformation.Filename.Directory}/Oi.xml
```

Lembre que você pode selecionar o nome da variável da janela de variáveis pressionando **`<Ctrl-Espaço>`**.

–**`Contents:`** Esta aba permite configurar o formato da saída. Deixe os dados sugeridos pela ferramenta.

–**`Fields:`** Pressione o botão **`Get Fields`**. A tabela é carregada com os três dados de entrada. No arquivo só se quer incluir a saudação. Por isso, apague o nome e o sobrenome.

Por último, não esqueça de salvar a Transformation com todas as mudanças realizadas.

Como funciona?

Quando uma Transformation é executada, quase todos os Steps são executados em forma simultânea. A execução é **assincrônica**. As filas de dados viajam através dos Steps a seu próprio ritmo. Cada fila de dados processada viaja em direção ao Step seguinte sem esperar às outras.
Na prática, se você esquecer desta propriedade das Transformations, poderia acontecer que os resultados não coincidissem com os esperados.

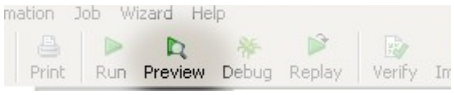
Já está tudo configurado. A Transformation lê o arquivo de entrada, p*ara cada uma das filas de dados* (neste caso para cada nome) se executa o código javascript que gera uma mensagem e finalmente essa mensagem é enviada ao arquivo de saída. Sendo este exemplo tão pequeno e tendo tão poucos dados, é muito difícil perceber que esta Transformation executa seus Steps de forma assincrônica, no entanto funciona sempre assim, por exemplo: pode acontecer que num momento a Transformation esteja enviando alguma mensagem de saudação ao arquivo de saída, enquanto algumas filas apenas estão saindo do Step que lê o arquivo.

Verificar, pré-visualizar e Executar!

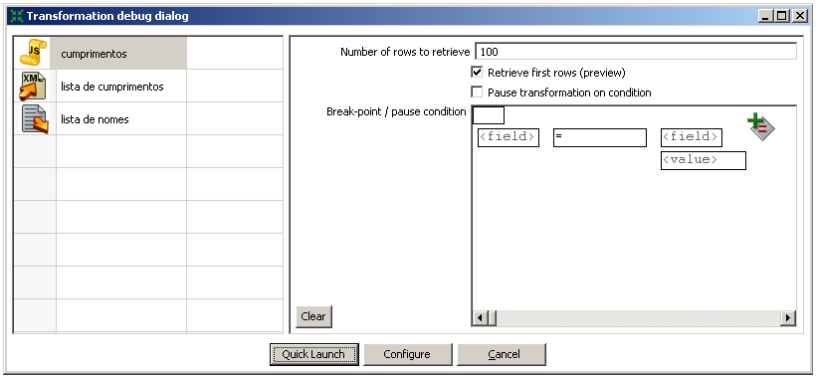
Antes de executar a Transformation, faça uma verificação.
Pressione o botão `Verify`.

A Spoon realizará uma verificação sintática da Transformation, detectando desta maneira erros de lógica como: Steps não atingíveis, falta de dados de conexão nos Steps relacionados com bancos de dados, entre outros.

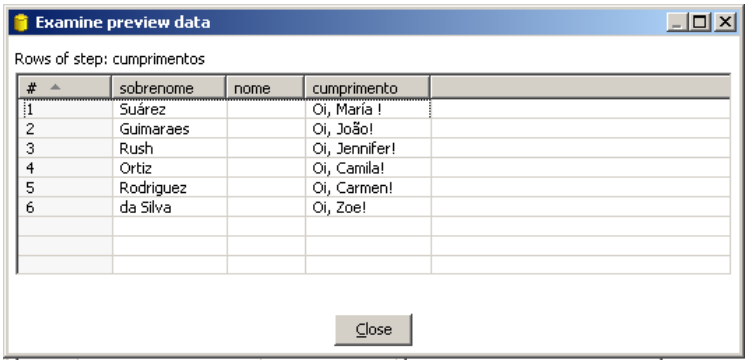
Se tudo estiver em ordem (deveria estar se você seguiu os Steps corretamente), você poderá pré-visualizar a saída.
Com o cursor posicionado no Step Javascript, selecione o botão `Preview`.



Aparecerá a seguinte janela:



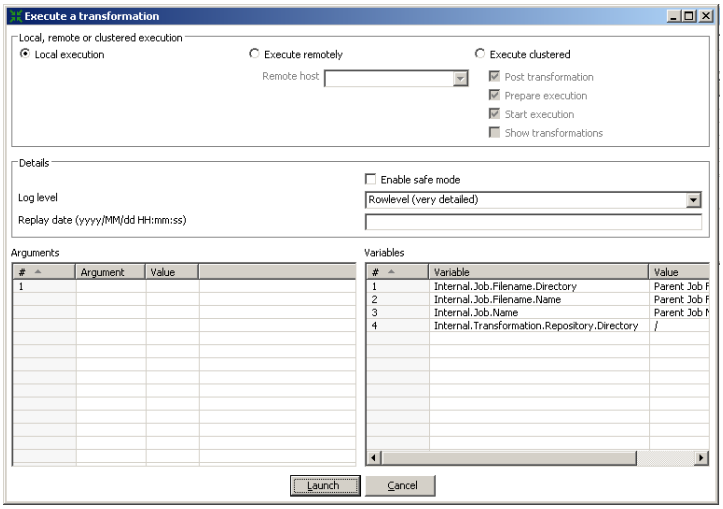
Como você pode ver, a Spoon sugire uma pré-visualização para o Step selecionado. Pressione **Quick Launch**. Aparece logo uma janela com uma mostra da saída do Step Javascript.



Você verá aqui as mensagens criadas para cada uma das pessoas da lista do arquivo de entrada.

Se a saída for a esperada, você estará preparado para executar a Transformation. Pressione o botão **Run**.

Em primeiro lugar aparecerá uma janela onde se podem indicar, entre outros dados, os parâmetros de execução para a Transformation, e o nível de log desejado. Neste caso não modifique nada e execute diretamente pressionando o botão **Launch** ao pé da janela.

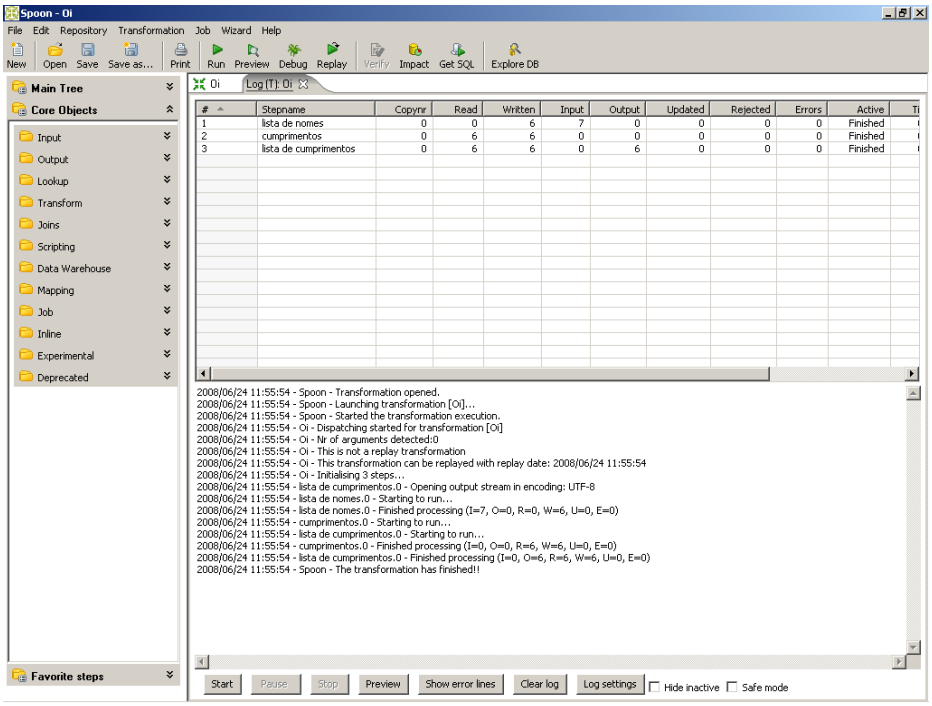


Imediatamente você vai ver uma nova aba na área de trabalho: É a **aba de log** correspondente ao log desta execução. (Você terá notado que a mesma aba apareceu quando fez a pré-visualização).

A aba de log tem duas seções:

Na parte superior, você verá, para cada Step da Transformation, um detalhe das operações realizadas. Em particular observe estas colunas:

- `Read`: corresponde ao número de filas que chegaram de Steps anteriores.
- `Written`: corresponde ao número de filas de saída em direção aos Steps seguintes.
- `Input`: número de linhas lidas de arquivo o base de dados.
- `Output`: número de linhas gravadas em arquivo o base de dados.
- `Errors`: indica se a execução deste Step teve erros. Nesse caso, a fila do log se verá em vermelho.



Na parte inferior poderá ver o passo a passo da execução. O detalhe mostrado dependerá do nível de log estabelecido. Se você tiver atenção neste detalhe, vai ver como a execução dos Steps é assincrônica.

Ao final do texto do log você vai ver:

```
Spoon - The transformation has finished!!
```

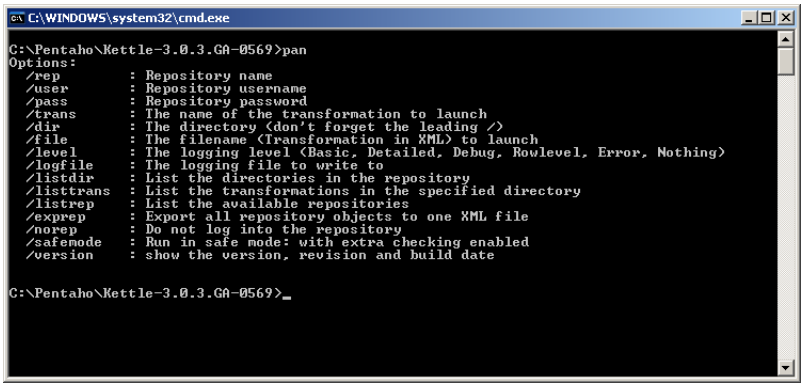
Se não tiver nenhum mensagem de erro, procure no disco o arquivo gerado `Oi.xml` e corrobore seu conteúdo.

Pan

Pan é a ferramenta que permite executar Transformations no prompt. O script de execução é pan.bat (Windows) ou pan.sh (outras plataformas), e se encontra na mesma pasta onde você instalou o PDI.
Se você executar:

```
Pan
```

aparecerá uma descrição do comando e as opções disponíveis.



Para executar a Transformation que criou, execute o comando em sua forma mais simples:

```
Pan /file <caminho_da_pasta_de_trabalhos>/Oi.ktr /norep
```

`/norep` é um comando para indicar à ferramenta que no deve se conectar ao catálogo.

`/file` precede ao nome do arquivo correspondente à Transformation a executar.

`<caminho_da_pasta_de_Trabalhos>` é o caminho da pasta Tutorial, por exemplo:

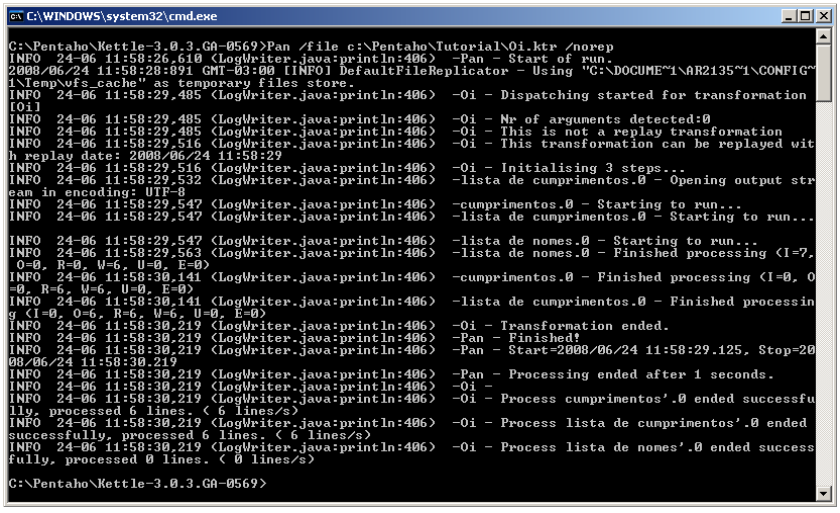
`c:/Pentaho/Tutorial` (Windows)

ou

`/home/PentahoUser/Tutorial`

Para as outras opções, como por exemplo o nível de log, a ferramenta utiliza valores predefinidos.

Assim que você ingressar este comando, a Transformation se executará da mesma maneira que com a Spoon, mas escreverá o log na mesma janela de execução, a não ser que você tiver direcionado o log a um arquivo. O formato do log varia um pouco, porém a informação é a mesma que via na interface gráfica.



Desta maneira terminou o exercício: Uma Transformation simples que permitiu conhecer os elementos de trabalho básicos do PDI.

Oi mundo, mais uma vez!

Agora que já criou e executou sua primeira Transformation, você vai acrescentar sua funcionalidade. Além de procurar um trabalho de mais qualidade (embora ele seja tão pequeno), a segunda parte do tutorial lhe permitirá conhecer alguns conceitos novos tão importantes como os vistos na primeira parte:

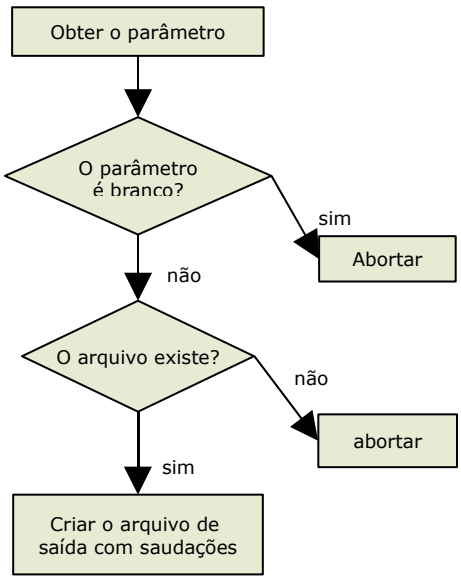
-Jobs
-Job Entries
-Hops
-Uso de Parâmetros e Informação do Sistema
-Variáveis de Usuário
-Arquivo de configuração kettle.properties
-Execução de Jobs no prompt com a ferramenta Kitchen.

Exercício

Estas são as melhoras que incorporará ao exercício:

-Procurará o arquivo de entrada numa pasta predefinida diferente à pasta onde estão as Transformations. O nome do arquivo não será fixo; será recebido como parâmetro.
-Validará que o arquivo exista (exercício: execute a Transformation anterior com um nome de arquivo inexistente e observe o que acontece!)
-Fará que o nome do arquivo de saída seja dependente do nome do arquivo de entrada.

Veja aqui um pequeno diagrama com as tarefas a realizar:



Você vai executar este diagrama com um Job.

Antes de começar, veja algumas definições novas:

Um **Job** é um componente composto por ***Job Entries*** ligadas por médio de ***Hops***. Estas Entradas e Hops estão arranjados de acordo à ordem de execução esperada. Por isso se diz que um Job está orientado ao **controle de fluxo**.

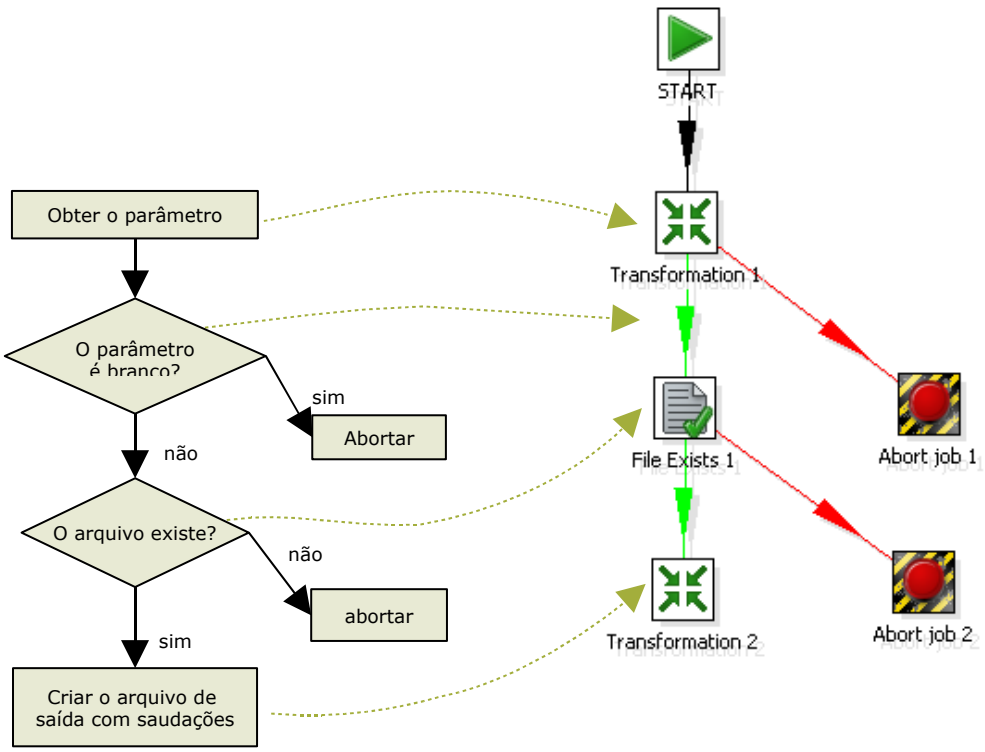
Una **Job Entry** é a unidade de execução dentro de um Job. Cada Job Entry cumpre uma funcionalidade específica que vá desde verificar a existência de uma tabela até fazer um envio de correio eletrônico.

Um Job pode executar uma Transformation o um outro Job, ou seja, "**Job**" e "**Transformation**" são dois casos particulares de Job Entries.

Um **Hop** é a representação gráfica que identifica a seqüência de execução entre duas Job Entries.

Mesmo que um Hop tenha só uma origem e um destino, uma Job Entry pode ser atingida por mais de um Hop e dela pode sair também mais de um Hop.

A seguinte é uma correspondência entre os elementos do diagrama, e as Job Entries e os Hops que você vai criar.



Desta maneira:

-A solicitude do parâmetro será resolvida com uma nova Transformation
-A verificação da existência do parâmetro será feita utilizando o resultado da Transformation anterior. A decisão de qual o Step a seguir, será feita utilizando execução condicional.
-A verificação da existência do arquivo será feita por uma Job Entry específica para essa funcionalidade.
-A execução da tarefa central do Job será feita por uma variante da Transformation da primeira parte do tutorial.

Preparação do Ambiente

Para esta parte do tutorial, você destinará, para os arquivos de entrada e saída, uma nova pasta. Crie então a pasta Arquivos no caminho de sua escolha.

Dentro desta pasta, copia o arquivo `lista.csv` o crie um diferente, respeitando o formato. Opcionalmente, cambie o nome dele.

Para não ter que escrever o caminho do arquivo cada vez que precisar fazer referência a ele, você criará uma variável que conterá esta informação, e que estará sempre disponível. Para isso edite o arquivo de configuração **kettle.properties**. Este arquivo se encontra na pasta

`C:\Documents and Settings\<username>\.kettle\` (Windows)

o

`$HOME/.kettle` (outras plataformas)

Y é criado junto com a pasta `.kettle` a primeira vez que você ingressa à Spoon.

Ao final do arquivo `kettle.properties`, acrescente esta linha:

```
ARQUIVOS=<caminho_da_pasta_criada>
```

Por exemplo:

```
ARQUIVOS=c:/Pentaho/Arquivos
```

ou

```
ARQUIVOS=/home/PentahoUser/Arquivos
```

Salve o arquivo.

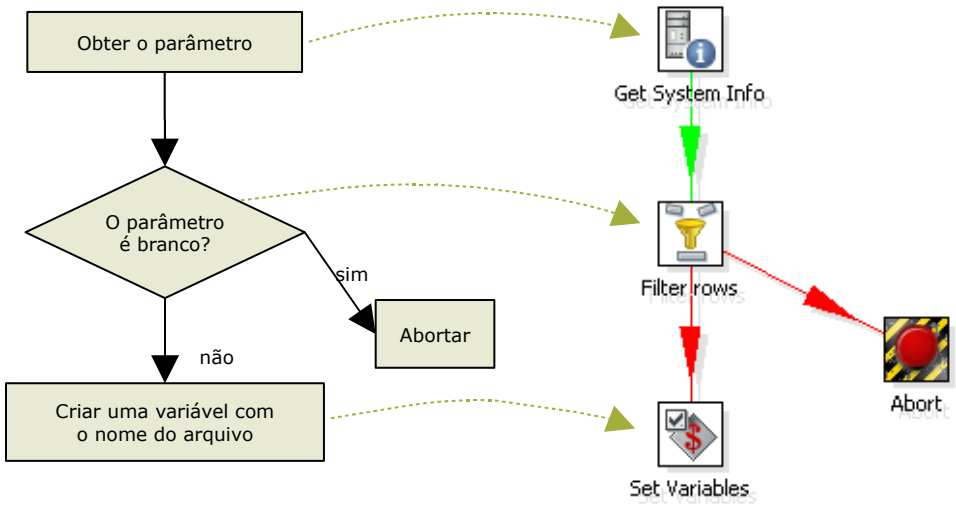
A Spoon lê o arquivo de configuração no início. Por isso, para que sua variável `ARQUIVOS` seja reconhecida, você terá que reiniciar a Spoon.

Agora você está preparado para começar. Você fará o exercício em três partes:

A.Criará a Transformation que pede o parâmetro
B.Modificará a Transformation `Oi.ktr` para parametrizar os nomes dos arquivos de entrada e de saída.
C.Construirá o Job segundo o diagrama anterior.

A. Criação da Transformation que pede o parâmetro

Esta nova Transformation é muito simples. Este diagrama mostra as tarefas envolvidas e sua correspondência com Steps do PDI:



Passo a passo

1.Criação da Transformation: Crie uma nova Transformation, da mesma maneira que fez na primeira parte do tutorial. Chame a esta Transformation de "`obter_nome_de_arquivo`".
2.Leve à área de trabalho estes Steps, coloque o nome e una-os com Hops segundo o diagrama.

 Get System Info (categoria Input)

 Filter Rows (categoria Transform)

 Abort (categoria Transform)

 Set Variable (categoria Job)

3.Configure os Steps como se explica abaixo.

a) Configuração do Step Get System Info (categoria Input)

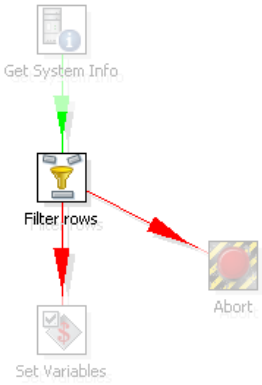
Este Step permite capturar informação externa à Transformation, tanto informação do sistema como por exemplo fecha atual, como parâmetros ingressados no prompt de comando. Neste caso você vai utilizar o Step para tomar o primeiro e único parâmetro e guardá-lo no campo "arquivo".

A janela de configuração do Step tem uma tabela. Nesta tabela, cada fila que preencher gerará uma nova coluna e seu conteúdo corresponderá a um dado proveniente do sistema.

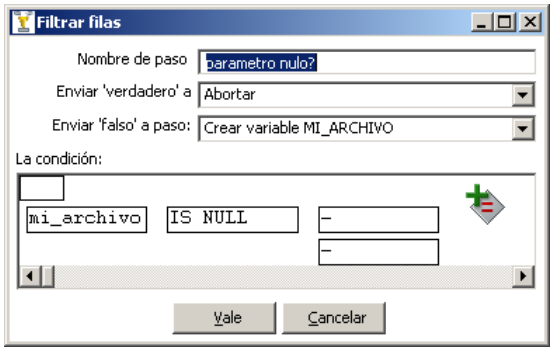
-Selecione o Step com duplo clique.
-Na primeira cela escreva sob a coluna **Name**, a palavra "meu_arquivo".
-Quando você der clique na cela da mesma fila, sob a coluna "**Type**", aparecerá uma janela com as opções disponíveis. Selecione "command line argument 1".
-Clique em **OK**.

b) Configuração do Step Filter Rows (categoria Transform)

Este Step divide a saída em dois, segundo uma condição. Aqueles dados que cumprem a condição seguem um caminho, e os que não a cumprem, seguem outro. No caso que deste Step saia só um Hop, unicamente passarão ao seguinte Step os dados que cumprirem a condição.



-Selecione o Step com duplo clique
-Escreva a condição: no quadro **Field** selecione "meu_arquivo" e substitua o signo "=" por "IS NULL".
-Na lista ao lado do "**Send 'true' data to Step**" selecione o Step **Abort**.
-Na lista ao lado do "**Send 'false' data to Step**" selecione o Step **Set Variable**
-Clique em **OK**.

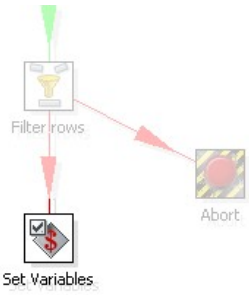


Desta maneira, um parâmetro em branco chegará ao Step **Abort**. Um parâmetro que não for branco chegará ao Step **Set Variable**.

c) Configuração do Step Abort (categoria Transform)

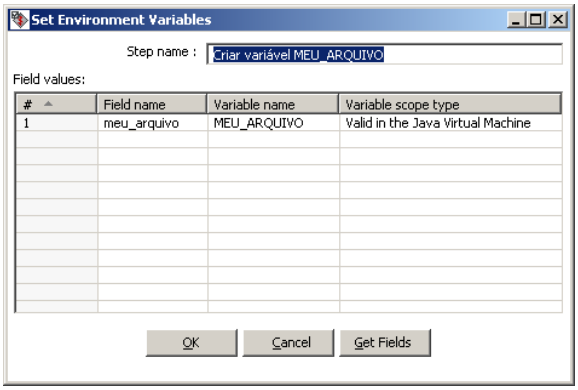
Na verdade, não tem nada para configurar neste Step. Se um dado chega até este Step, a Transformation aborta. Isto ocasiona que a mesma termine como mal-sucedida, resultado que será utilizado no controle da seqüência do Job.

d) Configuração do Step "Set Variable" (categoria "Job")



Este Step permite criar variáveis e lhes colocar o valor de alguns dos dados de entrada do Step.
A janela de configuração do Step tem uma tabela. Cada fila desta tabela se corresponde com uma nova variável.

-Selecione o Step com duplo clique.
-Pressione o botão **Get Fields**. Aparecerá o único campo que existe: `meu_arquivo`. Como nome para a variável que está criando se sugere o nome do campo, em maiúscula: `MEU_ARQUIVO`. Deixe os dados sugeridos, tal como se mostra nesta tela:



-Clique em **OK**.

Desta maneira você tem criado a variável `MEU_ARQUIVO`, que utilizará depois.

Execução

Para verificar o correto funcionamento desta Transformation, pressione **Executar**.
Na primeira janela observe a tabela **Parameters**. Esta tabela serve para escrever nela os mesmos dados que colocaria como parâmetro se executasse no prompt. Na primeira fila, sob a coluna **Value**, escreva `lista`.
Pressione o botão **Launch**.

Se você observar o log, vai encontrar uma linha como esta:

```
Set Variables.0 - Set variable MEU_ARQUIVO to value [lista]
```

Faça mais um teste. Novamente pressione **Executar**, no entanto esta vez não ingresse parâmetros. No log você vai encontrar esta saída:

```
Abort.0 - Row nr 1 causing abort : []
Abort.0 - Aborting after having seen 1 rows.
```

Além disso, o Step **Abort** na parte superior da aba de log aparecerá marcando um erro. Isto indica que a Transformation finalizou com erro, como você já tinha previsto.

B. Parametrização da Transformation Oi.ktr

Aqui você vai modificar a Transformation `Oi` para parametrizar os nomes dos arquivos de entrada e saída. Se o parâmetro é `xxx`, a Transformation lerá o arquivo `xxx.csv` e gerará o arquivo `xxx_con_saudações.xml`.
E como último cambio, agregará um filtro para descartar las filas do arquivo que vier em branco.

Passo a passo

-Abra a Transformation `Oi.ktr`
-Abra a janela de configuração do Step **CSV File Input**
-**Filename:** Apague o conteúdo desta caixa de texto, e pressione **<Ctrl-Espaço>** para ver a lista de variáveis. Você verá, na lista, a variável que criou no arquivo de configuração: `ARQUIVOS`. Selecione-la, e ao lado escreva o nome da variável que criou na Transformation anterior, de modo que o texto completo fique assim:

```
${ARQUIVOS}/${MEU_ARQUIVO}.csv
```

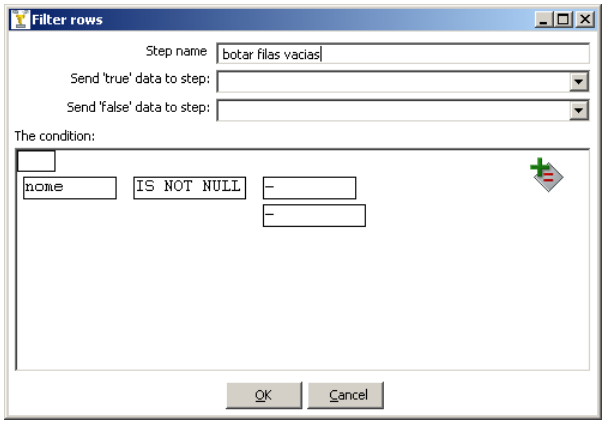
-Pressione **OK**.

-Abra a janela de configuração do Step **XML Output**
-**Filename:** Apague o conteúdo desta caixa de texto, e substitua-lo por este:

```
${ARQUIVOS}/${MEU_ARQUIVO}_com_saudações.xml
```

-Pressione **OK**.

-Arraste à área de trabalho um Step **Filter Rows**, até que o Step estiver em cima do Hop que sai de **CSV Input**. Quando você vir que a linha do Hop vira más larga, solte o botão do mouse. Você terá ligado o novo Step à seqüência da Transformation. Escreva a condição: No quadro **Field** selecione "`nome`" e na condição selecione "`IS NOT NULL`".
Deixe os dados "**Send 'true' data to Step**" e "**Send 'true' data to Step**" em branco. Desta maneira, vão seguir ao seguinte Step unicamente os nomes que cumprirem a condição, ou seja, aqueles que não forem nulos. Esto é una variante na forma de utilizar o Step **Filter Rows**, respeito de como foi usada anteriormente.

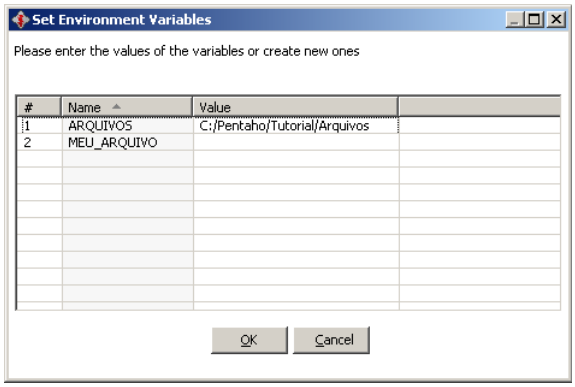


-Pressione **OK**.

-Para não perder a Transformation original, pressione "**Save As**" e coloque o nome "`Oi_con_parametros`". Se criará o arquivo `Oi_con_parametros.ktr`

Execução

Para verificar as mudanças realizadas na Transformation, você precisa que a variável `MEU_ARQUIVO` exista e tenha um valor. Como esta Transformation é independente da Transformation que cria a variável, para poder executá-la você terá que criar a variável manualmente. Para isso, selecione "**Set Environment Variables**" no menu "**Edit**". Aparecerão as variáveis existentes. Ao final da lista, coloque a variável `MEU_ARQUIVO`, e coloque como conteúdo o nome do arquivo que você criou, ou `lista` caso você tenha deixado o existente. Não coloque a extensão `.csv`.



Agora pressione **Executar**.
Você vai ver que na tela de configuração de execução sua variável aparece na lista de variáveis existentes.
Pressione o botão **Launch**, e deixe que a Transformation faça sua tarefa.
Depois verifique que o arquivo

```
<meu_arquivo>_com_saudações.xml
```

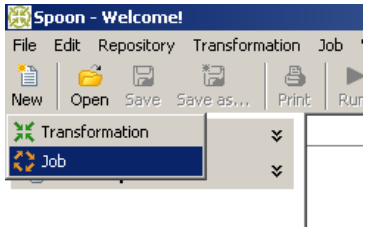
tenha sido criado na pasta de arquivos, onde `<meu_arquivo>` corresponde ao nome do arquivo que utilizou como entrada. Também verifique que o conteúdo seja o esperado.

C. Construção do Job principal

O último passo desta parte do tutorial é a construção do Job principal.

Passo a passo

1)Criação do Job:



a)Selecione **New → Job**. Também você pode criar um Job desde o menu **Files → New → Job** ou pressionando **<Ctrl-Alt-N>**. Uma nova aba ficará disponível para trabalhar com o novo Job. Aqui você colocará as Job Entries e os Hops.
b)Selecione a opção **Job → Configuração**. Aparecerá uma janela onde você poderá especificar propriedades gerais do Job. Neste caso só coloque um nome e uma descrição.
c)Pressione o botão **Save**. Salve o Job na pasta `Tutorial` com o nome "`Oi`". Será criado o arquivo `Oi.jkb`.

2)Armado do esqueleto do Job com Job Entries e Hops: Tal como para as Transformations você tinha uma paleta de Steps, aqui tem uma paleta de Job Entries. Uma diferença que você notará ao início é que estas Entradas no estão agrupadas em Categorias[1].

Crie o Job:

a)Leve à área de trabalho estas Job Entries, coloque o nome e una-las com Hops segundo o diagrama.

 Start

 Transformation x 2

 File Exists

b)Leve à área de trabalho as seguintes Job Entries, coloque o nome e una-las ao diagrama.

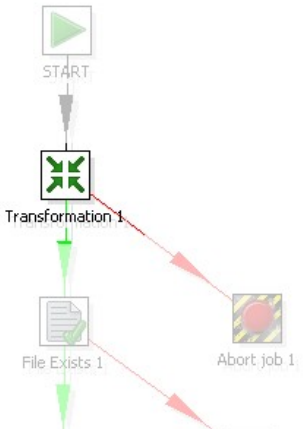
 Abort Job x 2

Você vai ver que os Hops ficaram pintados de vermelho. Depois verá o motivo.

c)Configure os Steps como se explica abaixo.

a) Configuração da Primeira Job Entry Transformation

A entrada **Transformation** tem o objetivo de executar uma Transformation. Você vai configurar esta entrada para ela executar a Transformation que obtém o parâmetro.



-Dê um duplo clique em cima da Job Entry para selecioná-la. Aparece a janela de configuração para este tipo de Job Entries.

-**Transformation filename:** Aqui deve indicar o nome do arquivo da Transformation "`obter_nome_de_arquivo.ktr`". Como as Transformations e os Jobs se encontram na mesma pasta, vai usar como referência o caminho onde está salvado o Job. Ingressa este texto:

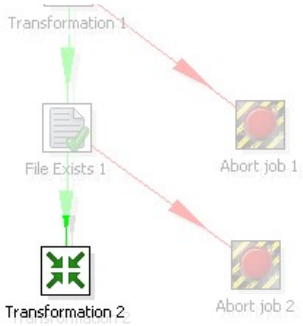
```
${Internal.Job.Filename.Directory}/obter_nome_de_arquivo.ktr
```

A variável pode ser escrita manualmente, ou selecionada da lista de variáveis. Da mesma forma, o nome do arquivo pode ser escrito, ou procurado com o botão "**Browse**".

-Pressione **OK**.

[1] Na versão 3.1.0 – não liberada ainda - as Job Entries aparecem agrupadas em Categorias como General, Mail, File management, entre outras.

b) Configuração da Segunda Job Entry Transformation



Você vai configurar esta entrada para ela executar a Transformation que faz a principal função deste Job.

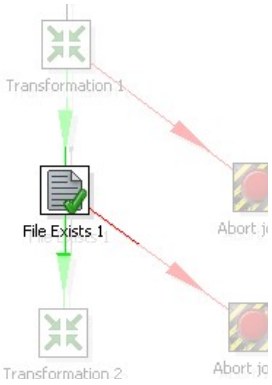
-Dê um duplo clique em cima da Job Entry para selecioná-la. Aparece a janela de configuração para este tipo de Job Entries.

-**Transformation filename:** Neste caso, ingresse o nome da outra Transformation:

```
${Internal.Job.Filename.Directory}/Oi_con_parametros.ktr
```

-Pressione **OK**

c) Configuração de File Exists



A Entrada **File Exists** verifica a existência de um arquivo. Só se o arquivo existe a Entrada tem um resultado bem-sucedido.

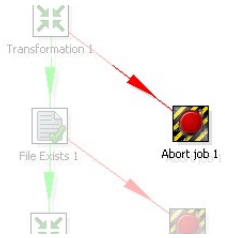
-Dê um duplo clique em cima da Job Entry para selecioná-la. Aparece a janela de configuração para este tipo de Job Entries.

-**Filename:** Aqui deve indicar o nome completo do arquivo cuja existência quer verificar. É o mesmo que escreveu na Transformation `Oi` modificada:

```
${ARQUIVOS}/${MEU_ARQUIVO}.csv
```

Lembre que a variável `${ARQUIVOS}` está definida no arquivo `kettle.properties`, e a variável `${MEU_ARQUIVO}` é criada na Job Entry anterior a esta.

d) Configuração da Primeira Job Entry Abort Job

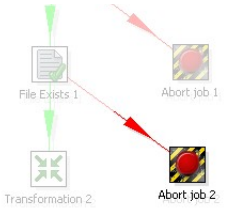


A primeira Entrada **Abort Job** será executada quando não tiver indicado um parâmetro. O que você vai configurar aqui é a mensagem com o motivo pelo qual o Job é abortado.

-Em "**Message**" escreva este texto:

```
No se indicou o parâmetro com o nome do arquivo
```

e) Configuração da Segunda Job Entry Abort Job



Esta Entrada **Abort Job** será executada quando o arquivo não existir.

-Em "**Message**" escreva este texto:

```
O arquivo indicado no existe (${ARQUIVOS}/${MEU_ARQUIVO}.csv)
```

Durante a execução, a ferramenta substituirá as variáveis pelo seu valor, mostrando, por exemplo, esta mensagem:

```
O arquivo indicado no existe (c:/Pentaho/Arquivos/lista.csv)
```

f) Configuração dos Hops

A execução de uma Job Entry pode ser realizada:
- incondicionalmente: se executa sempre
- só quando a execução da Job Entry anterior foi bem-sucedida
- só quando a execução da Job Entry anterior foi mal-sucedida

Esta execução se indica graficamente por meio de cores nos Hops:

- um Hop em **preto** indica que a seguinte Job Entry se executa incondicionalmente
- um Hop em **verde** indica que a seguinte Job Entry se executa quando a anterior foi bem-sucedida
- um Hop em **vermelho** indica que a seguinte Job Entry se executa quando a anterior foi mal-sucedida

Por causa da ordem em que criou e ligou as Job Entries, todos os Hops quedaram configurados tal como você precisa, isto é:

-A primeira Transformation é executada sempre (O Hop que sai de **Start** em direção a ela está em **preto**)
-Se a Transformation que toma o parâmetro não encontrar o parâmetro, o seja, for mal-sucedida, o controle passará pelo Hop **vermelho** em direção à Job Entry **Abort Job**.
Se a Transformation for bem-sucedida, o controle passará pelo Hop **verde** que conduz à Job Entry **File Exists**.
-Se o arquivo não existir, o seja, a verificação da existência do arquivo for mal-sucedida, o controle passará pelo Hop **vermelho** em direção à segunda Job Entry **Abort Job**.
-Se a verificação for bem-sucedida, o controle passará pelo Hop **verde** que conduz à segunda Job Entry **Transformation**.

Se quisesse mudar a condição para a execução de uma Job Entry, estes seriam os Steps:
-Selecionar o Hop que chega a esta Job Entry
-Pressionar botão direito, com o qual aparecerá um menu contextual.
-No menu selecionar `Evaluation` seguido de uma das três condições que se vêem.

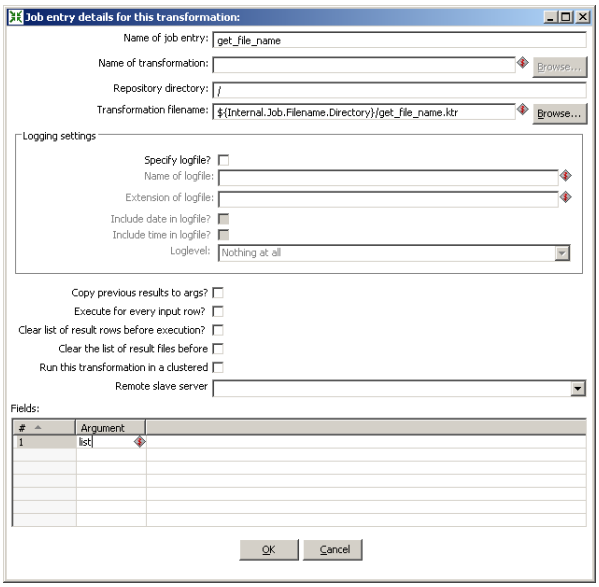
Como funciona?

Quando um Job é executado, a execução está conformada pela ordem das Job Entries, o sentido dos Hops, e a condição sob a qual se executa ou não uma Entrada. A execução é **seqüencial**. Uma Job Entry deve terminar seu execução para que a execução das Job Entries que lhe seguem no diagrama possam ser iniciadas.

Na prática, um Job pode ser um salva-vidas para corrigir problemas de seqüencialidade nas Transformations. Se você precisar que uma parte de uma Transformation terminar completamente antes que o resto começar, uma forma de resolvê-lo pode ser desmembrando a Transformation em duas Transformations independentes, e invocando-as desde um Job em forma seqüencial.

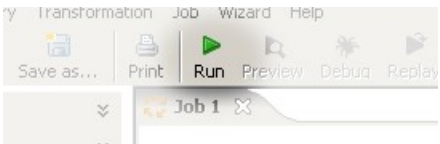
Execução

Para executar o Job, em primeiro lugar você precisa indicar o parâmetro. Dado que o único lugar onde o utiliza é na Transformation "obter_nome_de_arquivo" (depois só utiliza a variável que contem esse dato) indique o parâmetro da seguinte forma:
–Selecione com duplo clique a Transformation "`obter_nome_de_arquivo`"
–A janela que aparece tem uma tabela titulada **`Fields`**. Na primeira fila da tabela escreva o nome do arquivo criado na pasta "`Arquivos`" (sem extensão).



–Pressione **`OK`**.
Fica assim estabelecido o parâmetro.

Agora pressione o botão **`Executar`**.



Aparece uma janela com dados gerais relacionados com a execução do Job.
Pressione o botão **Executar**.

De imediato é aberta a **aba de log de Jobs**.

A aba de log dos Jobs, como a das Transformations, se encontra dividida em duas partes:

A parte superior mostra as Job Entries. Para cada Job Entry executada, se vê, entre outros dados, o resultado da execução. Como já foi dito, a execução das Entradas é seqüencial, por tanto, se uma Entrada terminar mal-sucedida, não se verão no log as seguintes Entradas, dado que as mesmas nem começam a se executar.

Na parte inferior é mostrado o detalhe do log indicando o início e fim das Job Entries. No caso das Job Entries correspondentes a Transformations (das quais tem duas) se inclui também o detalhe de log das Transformations mesmas.

Quando você vir ao final do log o texto:

```
Spoon - Job has ended.
```

quer dizer que já se encontra disponível o arquivo gerado. Se o arquivo de entrada fosse: `lista.csv`
o arquivo de saída deveria se chamar de: `lista_con_saudações.xml`
e estar localizado na mesma pasta.
Procure-o na pasta Arquivos e corrobore seu conteúdo.

Mude o nome do parâmetro: Substitua-lo por um nome que não se corresponda com um arquivo da pasta Arquivos. Execute o Job. Você verá como o Job aborta, e aparece no log a mensagem

```
Abort - o arquivo indicado não existe
```

seguido do nome do arquivo que você colocou como parâmetro.

Por último apague o conteúdo do parâmetro, e execute pela terceira vez. Neste caso o Job também aborta, e no log você poderá ver a seguinte mensagem:

```
Abort - No se indicou o parâmetro com o nome do arquivo
```

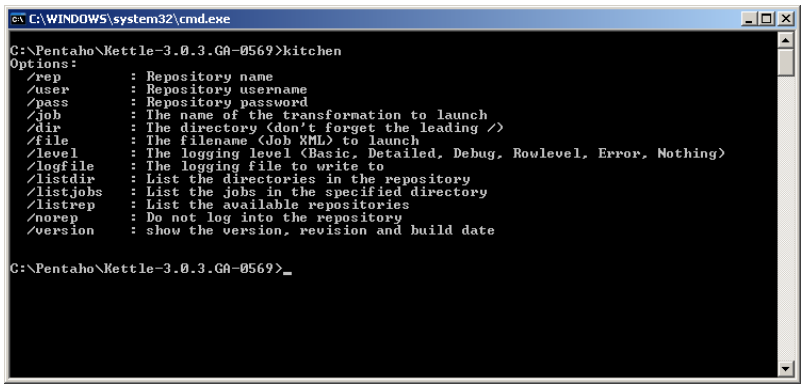
que é o comportamento que esperava neste caso.

Kitchen

Kitchen é a ferramenta que permite executar Jobs no prompt. O script de execução é `Kitchen.bat` (Windows) ou `Kitchen.sh` (outras plataformas), e se encontra na pasta de instalação do PDI.
Se você executar:

```
Kitchen
```

aparecerá uma descrição do comando e as opções disponíveis.



Para executar o Job criado, execute o comando em sua forma mais simples:

`<Kitchen /file <caminho_da_pasta_de_Jobs>/Oi.kjb <par> /norep`

`/norep` é um comando para indicar à ferramenta que no deve se conectar ao catálogo.

`/file` precede ao nome do arquivo correspondente ao Job a executar.

`<caminho_da_pasta_de_Jobs>` é o caminho da pasta Tutorial, por exemplo:

`c:/Pentaho/Tutorial` (Windows)
ou
`/home/PentahoUser/Tutorial`

`<par>` é o parâmetro que o Job espera. Lembre que corresponde ao nome do arquivo sem a extensão `csv`.

Para as outras opções, como por exemplo o nível de log, a ferramenta utiliza valores predefinidos.

Assim que você ingressar este comando, o Job se executará da mesma maneira que com a Spoon, escrevendo o log na mesma janela de execução, a não ser que você tiver direcionado o log a um arquivo. O formato do log varia um pouco mas basicamente a informação é a mesma que via na interface gráfica.

Tente executar o Job:
–sem parâmetros
–com um parâmetro inválido (que não se corresponda com um arquivo existente)
–com um parâmetro válido
e verifique que tudo dê certo.

Chegou o final desta segunda parte. Construiu um Job que aplicou melhoras à Transformation da primeira parte, e lhe permitiu conhecer mais elementos de trabalho do PDI, imprescindíveis para o dia a dia dos que trabalham com esta ferramenta.

Conclusão

Agora que você já viu um panorama do PDI, você está preparado para aproveitar toda a capacidade da ferramenta em seus próprios projetos.

Sem dúvida, a partir de agora recorrer às seguintes fontes será de muita utilidade para você:

- **Manuais de Usuário**
  Não há (ainda) manuais de usuário em português. Os manuais em inglês estão na pasta `\docs\English` dentro da pasta de instalação do PDI.

- **Exemplos**
  A pasta Samples dentro da pasta de instalação de PDI contem bastantes exemplos. Os mesmos podem servir para entender como funcionam alguns Steps, o para você se basear neles para criar seus próprios Jobs e Transformations.

- **Forum**
  O forum do Pentaho é uma fonte interessante de conhecimento. Aí você pode ver os problemas com os quais têm se enfrentado outros usuários do PDI e a forma em que podem ser resolvidos.
  Também pode ser você mesmo quem exponha dúvidas que tiver, o ajude a outros a solucionar seus dificuldades depois que você tiver adquirido experiência com a ferramenta.
  O forum do PDI se acesa diretamente de:
  http://forums.pentaho.org/forumdisplay.php?f=69

Além dos mencionados aqui, na tela de boas-vindas da Spoon você vai encontrar algumas fontes adicionais.

Tomara que este tutorial tenha sido útil para você começar a trabalhar com o PDI.

Comentários, críticas e sugestões são bem-vindos,
Sorte com PDI!