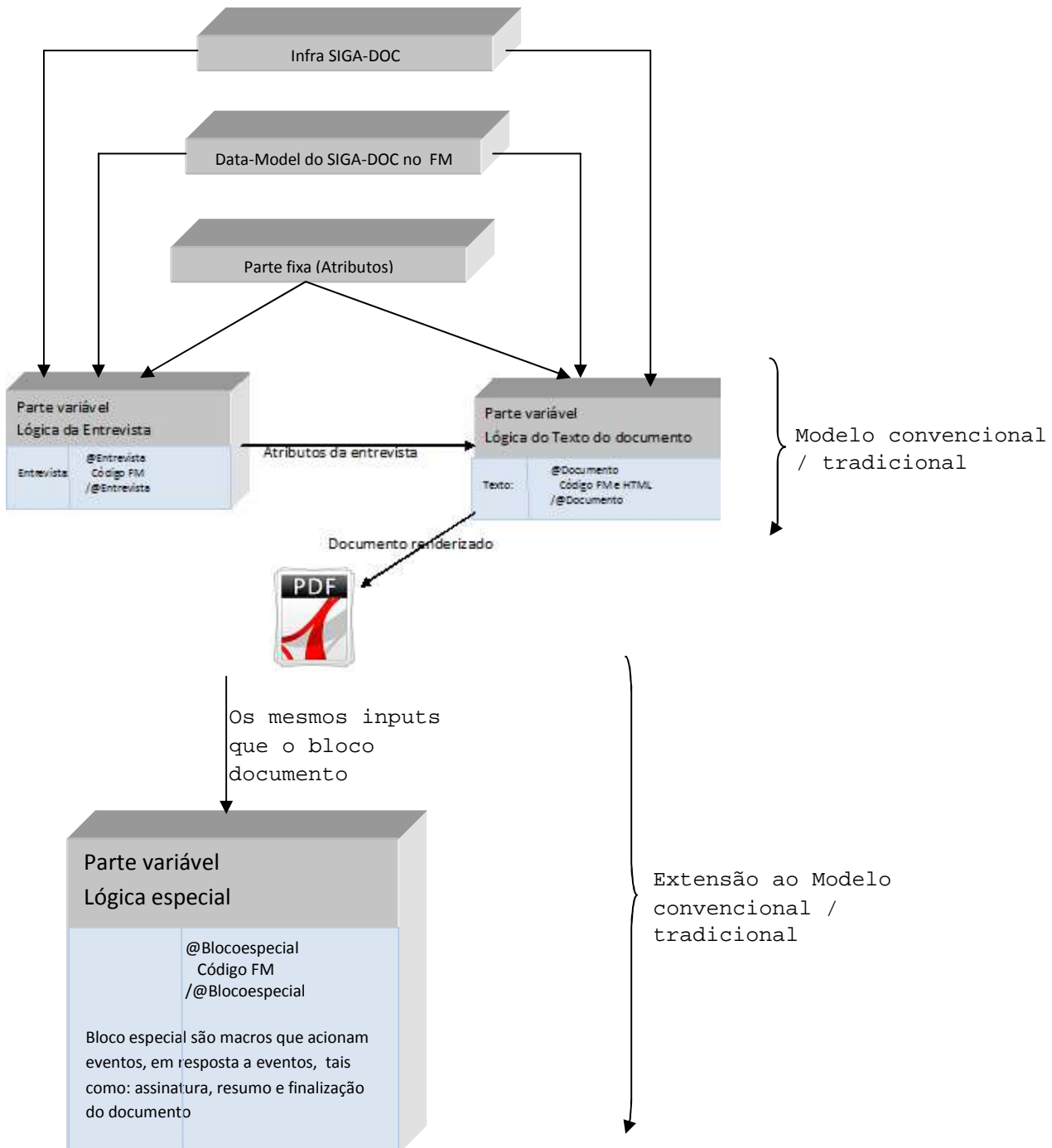


9.1 - @Entrevista e @Documento - Estrutura da aplicação FM para o SIGA-DOC

Basicamente a camada VIEW do modelo MVC deve, por definição, só apresentar os dados. Não deve possuir nenhuma lógica sofisticada de manipulação de dados, ou seja, os dados já devem ser apresentados a ela de forma mastigada pela camada CONTROLLER.

A estrutura da aplicação FM deve seguir o submodelo, já visto, e apresentado novamente aqui:



Estrutura de uma aplicação SIGA-DOC

```
[@entrevista]
Código da entrevista
[/@entrevista]
```

```
[@documento]
Código para geração do documento
[/@documento]
[@especial]
Código especial como resposta a um evento
[/@especial]
```

Ou seja, deve seguir os seguintes passos: coleta de informações (no bloco entrevista), crítica (no bloco entrevista ... no futuro implementaremos um bloco crítica) e geração do documento (no bloco documento). Se for necessário responder a algum evento especial, poderemos ter um bloco assinatura, finalização, resumo e etc.

9.1.1 - @Entrevista

Esta é a macro principal e obrigatória. Todo código/lógica da entrevista (coleta e crítica das informações) deve estar dentro deste bloco. Aqui teremos muito código FM e nenhum (ou muito pouco) código HTML. Basicamente todas as macros (e métodos que serão vistos posteriormente) descritas neste documento serão inseridas aqui.

Estamos estudando a possibilidade de deslocar a crítica das informações para um bloco @Critica ([@critica] ... [/@critica]), desta forma, cada bloco terá uma função distinta. Porém, isto é para o futuro.

```
[@entrevista]
Código da entrevista
[/@entrevista]
```

9.1.2 - @Documento

Esta é a macro complemento e obrigatória. Todo código/lógica para gerar o documento deve estar dentro deste bloco. Aqui teremos pouco código FM e muito código HTML. As macros invocadas neste bloco são relacionadas a formatação do documento (brasão, cabeçalho, fecho ...).

```
[@documento]
Código para geração do documento
[/@documento]
```

9.1.3 - @Especial

São macros que acionam eventos (e-mail, workflow ...), em resposta a eventos, tais como: assinatura, resumo e finalização do documento. Desta forma, teremos, se for necessário, os blocos:

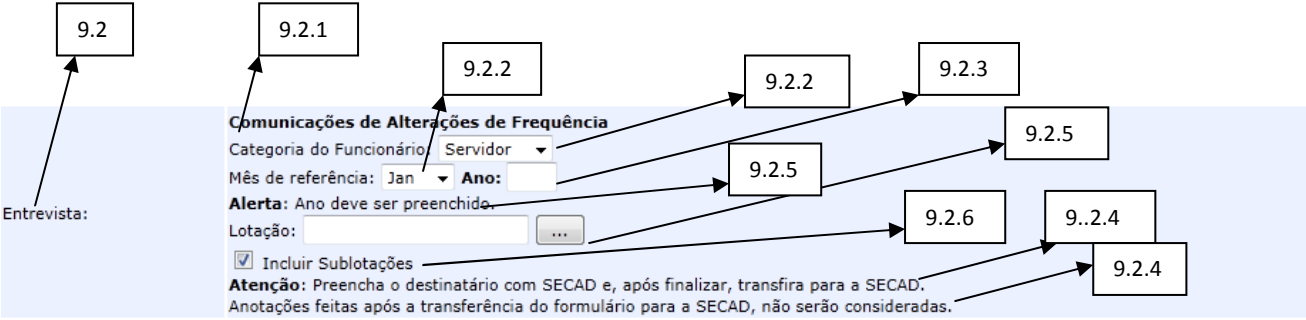
```
[@assinatura]
Código para quando o documento é assinado
[/@assinatura]

[@resumo]
Código para quando é gerado o resumo do documento
[/@resumo]

[@finalizacao]
Código para quando o documento é finalizado
[/@finalizacao]
```

9.2 - TRABALHANDO NO BLOCO @entrevista

As primeiras explicações sobre o funcionamento das macros serão baseadas na imagem da tela abaixo, de uma entrevista de uma aplicação recém desenvolvida em FM, cujo código será listado no anexo.



9.2.1 - @Grupo

O grupo tem por objetivo estruturar a entrevista em tabelas ou criar um bloco para a execução do AJAX. O @grupo promove uma quebra de linha, caso contrário as entrevistas seriam lado-alado, porém dentro do bloco @grupo as entrevistas ficam lado-a-lado.

Parâmetros:

Parâmetro	Descrição	Default
titulo	É o texto que aparecerá em negrito acima (linha anterior) da(s) pergunta(s) dentro do bloco	titulo=""
depende	Cria um bloco de código para o AJAX	depende=""
largura	Na verdade não estava nem funcionando, tive que mexer na macro na linha [#if grupoLarguraTotal]= 100> e alterá-la para [#if (grupoLarguraTotal >= 100)] Não esntendi O efeito deste parâmetro.	largura=0
esconder	Se esconder=true, a macro simplesmente desconsidera tudo (titulo, largura, depende ...) e trata o elemento que for definido dentro da macro como elemento inline Se esconder=false, a macro simplesmente considera tudo e trata o elemento que for definido dentro da macro como elemento block <div>	esconder=false

Exemplos:

@grupo sem código no seu interior.
[@grupo titulo="Comunicações de Alterações de Frequência"]

Código , geralmente uma seleção, texto e etc.O código não é obrigatório.

```
[/@grupo]
```

@Grupo com código em seu interior.

```
[@grupo titulo="Comunicações de Alterações de Frequência"]  
[@selecao var="catFuncionario" titulo="Categoria do Funcionário"  
  reler=true idAjax="catFuncionarioAjax" opcoes="Servidor; Estagiário"]  
[/@grupo]
```

O efeito da quebra de linha

```
[@grupo]  
  [@texto var="x" titulo="Nome" largura="30" maxcaracteres="40"]  
[/@grupo]  
[@grupo]  
  [@texto var="y" titulo="Matricula" largura="10" maxcaracteres="10"]  
[/@grupo]  
[@grupo]  
  [@data var="z" titulo="A partir de:"]  
[/@grupo]
```

Terá como resultado:

Nome: _____
Matrícula: _____
A partir de: _____

```
[@grupo]  
  [@texto var="x" titulo="Nome" largura="30" maxcaracteres="40"]  
  [@texto var="y" titulo="Matricula" largura="10" maxcaracteres="10"]  
  [@data var="z" titulo="A partir de:"]  
[/@grupo]
```

Terá como resultado:

Nome: _____ Matrícula: _____ A partir de: _____

⇒ Uma alternativa para se quebrar a linha dentro do @grupo é utilizar a macro @br que será vista adiante.

```
[@grupo]  
  [@texto var="x" titulo="Nome" largura="30" maxcaracteres="40"] [@br]  
  [@texto var="y" titulo="Matricula" largura="10" maxcaracteres="10"][@br]  
  [@data var="z" titulo="A partir de:"]  
[/@grupo]
```

Terá como resultado:

Nome: _____
Matrícula: _____
A partir de: _____

@Grupo com a opção **depende para execução de um código AJAX**.

Neste caso, se o ano não for fornecido será exibida uma mensagem. É importante observar que quando o código AJAX está sendo executado aparecerá a seguinte mensagem no canto direito superior do browser



```
[@texto titulo="Ano" var="ano" largura="4" maxcaracteres="4" obrigatorio="Sim"  
reler="ajax" idAjax="anoAjax"]
```

Observar que a identificação do AJAX do campo Ano será passada para o @Grupo depende, significando que este código é dependente deste campo.

```
[@grupo depende="anoAjax"]  
  [#if (ano!="") == ""]  
    [@mensagem titulo="Alerta" texto="Ano deve ser preenchido." vermelho=true/]  
  [/#if]  
[/@grupo]
```

Desta forma, a mensagem será exibida (ou retirada) sem a renderização de toda tela.

ESCLARECIMENTOS

Antes de prosseguirmos seria muito interessante uma palinha sobre o que representam os parâmetros: reler, relertab e obrigatório, já que eles sempre aparecerão nas macros abaixo, e com certeza estes geram muita confusão, seja pelo modo de funcionamento, seja pelas opções.



Reler: representa como a tela será lida (ida: cliente -> servidor) e consequentemente renderizada (retorno: servidor -> cliente). Devemos lembrar para que haja processamento no servidor os campos da tela devem ser enviados para o servidor, como descrito na seção [1.4 - Comportamento das aplicações WEB](#).

No sentido cliente -> servidor, os campos podem ir pela própria URL (método GET) ou via form (método POST). [Clique aqui para mais detalhes sobre GET e POST](#).

No retorno, pode voltar todo HTML da página (renderização total ou reload da página) ou somente parte do HTML, ou dados, via protocolo XML ou JSON, que chamaremos de renderização parcial. [Clique aqui para mais detalhes sobre XML e JSON](#). Para implementar a renderização parcial utilizamos a técnica AJAX que já foi bem discutida neste manual. [Clique aqui para mais informação sobre AJAX](#).

Opções do Reler

relertab =	idAjax fornecido?	Técnica de renderização	Evento que dispara a Renderização total ou AJAX?
"ajax"(*)	Sim	AJAX	onchange
"ajax"(*)	Não	AJAX, embora teremos um erro porque o idAjax não tenha sido passado	onchange
true ou "sim"(*)	Sim	AJAX	onchange
true ou "sim"(*)	Não	Total	onchange
false ou "nao"(*)	Sim	Não há renderização da tela	nenhum
false ou "nao"(*)	Não	Não há renderização da tela	nenhum
F5 (reload) no browser	Não importa	Total	F5, pois é o default dos browsers

(*) opções só para a macro @texto

Como podemos saber visualmente se estamos utilizando AJAX ou renderização total? Renderização total: depende do browser utilizado. No caso do Firefox,



Este ícone ficará girando

AJAX: vai depender da implementação. No nosso caso, implementamos este ícone,



Conforme visto no item 9.2.1

Outras formas de saber se estamos utilizando AJAX?

A forma mais apropriada é utilizando um depurador de cliente como mencionado no capítulo 10. No caso do FireBug, que é o depurador do FireFox, podemos utilizar a [aba Rede para verificar o tipo de request \(total ou Ajax\)](#).

No caso da renderização total perderei os dados que eu já tinha digitado. Teclei F5 sem querer, então vou perder os meus dados?

Não e sim. Não porque os dados são levados (cliente -> servidor) e depois retornados (servidor -> cliente). Sim, caso a aplicação servidora, por exemplo, inicialize alguns campos.

Evento onchange: observamos o que faz o AJAX ou a Renderização total ser acionada é o evento onchange, que traduzindo seria: na mudança. Ou seja, este evento é disparado quando um campo, com a opção onchange setado, é modificado.

Opções do Relertab

O tab faz referência a tecla que na prática leva o foco e retira o foco de um campo. O equivalente é clicar com o botão esquerdo (dar foco) e clicar no botão esquerdo fora do campo (retirar o foco).



que na prática leva o foco e retira o foco de um campo. O equivalente é clicar com o botão esquerdo (dar foco) e clicar no botão esquerdo fora do campo (retirar o foco).

Na linguagem de eventos, **ganhar o foco é onfocus** e **perder o foco é onblur**. Na verdade, estamos interessados aqui é no evento onblur. Este eventos são disparados sempre, mesmo que o conteúdo do campo não seja alterado.

relertab =	idAjax fornecido?	Técnica de renderização	Evento que dispara a Renderização total ou AJAX?
"sim"(*)	Não implementa AJAX	Total	onblur
"nao"(*)	Não implementa AJAX	nenhuma	nenhum

(*) aplicado somente a macro @texto. Embora as macros @pessoa, @lotacao e @funcao possuam o relertab como parâmetro, este não está implementado no código.

Podemos ter um campo com reler utilizando AJAX e relertab utilizando Renderização Total? Ehora não faça sentido, sim, visto que eles são implementados com eventos diferentes, porém se o usuário modificar um campo e sair dele, os dois eventos serão disparados, e dependendo do desenho de sua aplicação, o efeito pode não ser o esperado.

Exemplo de campo definido com reler e relertab.

Neste exemplo, o efeito final foi o mesmo.

```
[@grupo]
[@texto titulo="Ano Posse/Contratação" var="ano" largura="4" maxcaracteres="4" obrigatorio="sim"
reler="ajax" idAjax="anoAjax" relertab="sim"/]
```

```
[/@grupo]
[@grupo depende="anoAjax"]
[#if (ano!="") == ""]
  [@mensagem titulo="Alerta" texto="Ano deve ser preenchido."
    vermelho=true/]
[/#if]
[/@grupo]
```

obrigatorio é Obrigatório?



Sim e Não. Não durante a entrada de dados. Sim, caso o usuário clique em **OK** e saia da edição para elaboração (ver item 1.3 Transição de estado do documento), aí neste caso a crítica é realizada, fora do controle da aplicação, pelo SIGA-DOC. Como pelo SIGA-DOC?

Quando definimos um campo como obrigatório, passando o parâmetro `obrigatorio="Sim"` (na macro `@texto`) ou `obrigatorio=true` (nas outras macros), a macro cria um campo no formulário chamado `obrigatorios`, colocando todos os campos obrigatórios separados por vírgula, desta forma: `obrigatorios="campo1, campo2, campo3"`. Quando o usuário clica em **OK**, o SIGA-DOC assume o controle e acessa este campo. Para cada campo, ele verifica se o mesmo foi ou não preenchido, e em caso negativo, não deixa o usuário seguir para a elaboração. O SIGA-DOC coloca os campos obrigatórios em **negrito**.

Para uma tabela com as opções dos parâmetros `reler`, `relertab` e `obrigatorio`, [clique aqui](#).

9.2.2 - @Selecao

É um tipo de entrevista/pergunta que se exhibe um drop-down list para que o usuário selecione uma opção entre muitas.

Parâmetros:

Parâmetro	Descrição	Default
<code>var</code>	o nome da variável que reterá a opção selecionada pelo usuário	
<code>ttulo</code>	o texto que antecede (esquerda) o drop down list	
<code>idAjax</code>	se fornecido, representa a identificação do Ajax que será utilizado num bloco <code>@Grupo depende</code> visto anteriormente	<code>idAjax=""</code>
<code>reler</code>	<code>reler=true</code> <code>reler=false</code> Obs: se <code>reler=true</code> e <code>idajax</code> for passado, então assume-se AJAX	<code>reler=false</code>
<code>opcoes</code>	A lista contendo as opções que serão apresentadas no drop-down list	
<code>onclick</code>	Aplicação JS que pode ser passada quando a caixa de seleção é clicada. O default é ""	<code>onclick=""</code>

O que podemos passar para o onclick?

A - Emitir um alerta do JS com alguma explicação adicional

```
[@selecao var="catFuncionario" titulo="Categoria do Funcionário"
```

```
reler=true opcoes="Servidor; Estagiário" onclick=" Alert('aloaloalo')"/]
```

Onde temos aloaloalo, poderia ser \${variavel_da_aplicacao}

B - Abrindo uma janela e exibindo o conteúdo de uma string ou variável:

```
[@selecao var="catFuncionario" titulo="Categoria do Funcionário"
reler=true opcoes="Servidor; Estagiário" onclick="try {newwin.close();}
catch(err) {} newwin =
window.open('teste2',null,'height=160,width=400,status=no,toolbar=no,menubar=no,location=no'); newwin.document.write('aloaloalo')"/]
```

Onde temos aloaloalo, poderia ser \${variavel_da_aplicacao}

Exemplos:

```
[@selecao var="catFuncionario" titulo="Categoria do Funcionário"
reler=true opcoes="Servidor; Estagiário"/]
```

```
[@selecao var="mes" titulo="Mês de referência" reler=true
opcoes="Jan;Fev;Mar;Abr;Maio;Jun;Jul;Ago;Set;Out;Nov;Dez"/]
```

@Selecao com opção de idAjax.

```
[@selecao var="freq"+i titulo="Frequência" reler=true idAjax="ajax"+i opcoes="Sem
lançamentos;Com lançamentos"/]
```

```
[@grupo depende="ajax"+i]
  [#if .vars['freq'+i]?? && .vars['freq'+i] == "Com lançamentos"]
    [@comlançamentos/]
  [/#if]
[/@grupo]
```

9.2.3 - @Texto

É um tipo de entrevista/pergunta que se exhibe um campo texto de tamanho fixo para que o usuário digite qualquer coisa, números ou caracteres, sem crítica.

Parâmetros:

Parâmetro	Descrição	Default
var	o nome da variável que reterá a opção selecionada pelo usuário	
titulo	o texto que antecede (esquerda) o drop down list	titulo=""
idAjax	se fornecido, representa a identificação do Ajax que será utilizado num bloco @Grupo depende visto anteriormente	
reler	reler="ajax" reler="sim" reler="nao" Obs: se reler="ajax" então assume-se AJAX, mesmo que o idAjax seja "". Ajax: utiliza a técnica de renderização parcial da tela. Sim: promove a renderização de toda tela. Não: a tela não será renderizada. O evento que dispara o Ajax ou a Releitura total é o onchange.	reler="" Ou seja, nao. Se passarmos reler="abacaxi", ou reler="", ou qualquer outro string, será considerado como não.

largura	Largura da caixa de texto	largura=""
maxcaracteres	O número máximo de caracteres que o campo texto aceitará	maxcaracteres=""
obrigatorio	Poder ser: "Sim" ou "<u>nao</u>" Observação: Se um campo for definido como obrigatório o seu efeito não será imediato, ou seja, o usuário poderá deixá-lo em branco. Somente no momento em que for solicitado a geração do documento é que o sistema não permitirá que o mesmo fique em branco.	obrigatorio="nao" Obs: Observar no Sim, com maiúscula na primeira letra
relertab	relertab="sim" relertab="nao" sim: promove a renderização de toda tela Aqui não tem opção de AJAX. O evento que dispara é o onblur (quando o campo perde o foco) (efeito TAB)	relertab="" Ou seja, nao.
default	O valor que iniciará a variável passada ao parâmetro var , e que consequentemente será exibido no campo da tela inicialmente	default=""

Exemplos:

```
[@texto titulo="Ano" var="ano" largura="3" maxcaracteres="4" obrigatorio="Sim"
reler="ajax" idAjax="anoAjax"/]
```

9.2.4 - @Mensagem

É um tipo de entrevista/pergunta em que se exibe um campo texto de mensagem para alertar o usuário sobre algum erro no preenchimento da entrevista.

Parâmetros:

Parâmetro	Descrição	Default
texto	é a descrição da mensagem que será exibida abaixo do campo	
titulo	o texto que antecede (esquerda) a descrição da mensagem	titulo=""
vermelho	Pode ser: true ou false	vermelho=false

Exemplo:

```
[@mensagem titulo="Alerta" texto="Ano deve ser preenchido." vermelho=true/]
```

```
[@mensagem titulo="Atenção" texto="Preencha o destinatário com SECAD e, após finalizar, transfira para a SECAD."/]
```

```
[@mensagem titulo="" texto="Anotações feitas após a transferência do formulário para a SECAD, não serão consideradas." vermelho=true/]
```

Obs: temos a macro @mensagem2 [#macro mensagem2 texto titulo="" cor="black"]

9.2.5 - Macros de negócio

9.2.5.1 - @Lotacao


É um tipo de entrevista/pergunta em que se exibe um campo texto para que o usuário forneça a lotação (UO) desejada.

Parâmetros:

Parâmetro	Descrição	Default
titulo	O texto que antecede o campo de Lotação	
var	o nome da variável que reterá a opção digitada pelo usuário	
idAjax	se fornecido, representa a identificação do Ajax que será utilizado num bloco @Grupo depende visto anteriormente	idAjax=""
reler	reler=true reler=false Obs: se reler=true e idajax for passado, então assume-se AJAX	reler=false Ou seja, não.
relertab	O relertab não está implementado aqui, embora se tenha o parâmetro, até porque o evento que implementa o relertab é onblur, e este já está sendo usado para acionar a rotina AJAX como se pode observar na tabela abaixo: JavaScripts	relertab=""
buscarFechadas	Campos tipo: pessoa, lotação e lotação possuem dois estados distintos: Atual (aberto): é o estado normal, pois a data de encerramento está em aberto, é o que está em vigor, como por exemplo o nome atual de uma secretaria. Antigo(encerrado, fechado): seria, por exemplo, o(s) nome(s) antigo(s) de uma secretaria, a situação anterior de um servidor aposentado e etc. Este parâmetro permite que a busca seja efetuada também em estado(s) encerrado(s).	buscarFechadas=false Obs: no caso da macro @pessoa, caso buscarFechadas=true, então "buscarFechadas=true" é passado no paramList e todos outros parâmetros serão desconsiderados. [#if buscarFechadas] [@assign paramList = "buscarFechadas=true" /] [/#if]
default	O valor que iniciará a variável passada ao parâmetro var , e que consequentemente será exibido no campo da tela inicialmente	default=""
obrigatorio	Poder ser: true ou false . Observação: Se um campo for definido como obrigatório o seu efeito não será imediato, ou seja, o usuário poderá deixá-lo em branco. Somente no momento em que for solicitado a geração do documento é que o sistema não permitirá que o mesmo fique em	obrigatorio=false

	branco.	
paramList	Permite que parâmetros sejam passados, por exemplo, ao servlet que processará a ação desejada.	paramList="" Obs: os parâmetros devem ser passados na forma: "xpto=abacaxi"; "xpto2=banana"

JavaScripts:

Rotina	Evento que dispara	Descrição
ajax_\${var}\${tipoSel}() em editar.action Onde: tipoSel = {lotacao, pessoa, funcao} e var = nome da variável passada a macro @lotacao, @pessoa ou @funcao.	onblur	Executa a rotina Ajax conforme descrito em 3.4.1.1
return handleEnter() em static_javascript.js	onkeypress	Desconsidera a tecla <enter> quando pressionada pelo usuário
popitup_\${var}\${tipoSel}() em editar.action Onde: tipoSel = {lotacao, pessoa, funcao} e var = nome da variável passada a macro @lotacao, @pessoa ou @funcao.	onclick * Na verdade este é um evento do botão 	Abre janela de pesquisa: Se o campo sigla tiver preenchido, ele considera este conteúdo na string de pesquisa Senão, abre a janela listando todo o conteúdo

Exemplo:

```
[@lotacao titulo="Lotação" var="lotacao" reler=true idAjax="lotacaoAjax"/]
```

Será exibido

Lotação: 

Observação:


O nome da variável passado foi "lotacao", porém se analisarmos o HTML gerado não encontraremos este nome de variável, porque na verdade esta macro abre três campos:

nomedavariavelpassada_lotacaoSel.id (É o id da lotação como gravado no BD)
nomedavariavelpassada_lotacaoSel.sigla (É a sigla da lotação, tal como: STI)
nomedavariavelpassada_lotacaoSel.descricao (É a descrição da lotação, tal como: Subsecretaria de Tecnologia da Informação e de Comunicações)

Neste caso, teríamos:

lotacao_lotacaoSel.id
lotacao_lotacaoSel.sigla
lotacao_lotacaoSel.descricao

Se o usuário digitar STI, aparecerá:

Lotação: STI  Subsecretaria de Tecnologia da Informação e de Comunicações

Se o usuário clicar em ..., será exibida uma outra janela para que o usuário possa pesquisar as lotações.

Dados da Lotação	
Nome ou Sigla:	<input type="text"/>
Órgão:	Seção Judiciária do Rio de Janeiro
<input type="button" value="Pesquisar"/>	

Total: 413 Itens

Sigla	Nome	Fim de Vigência
10VF	10ª Vara Federal	
10VFCR	10ª Vara Federal Criminal	
10JEF	10º Juizado Especial Federal	
11VF	11ª Vara Federal	
12VF	12ª Vara Federal	
13VF	13ª Vara Federal	
14VF	14ª Vara Federal	
15VF	15ª Vara Federal	
16VF	16ª Vara Federal	
17VF	17ª Vara Federal	

[Primeira] 1 2 3 4 5 6 7 8 9 10 [Próxima> (2)] [Última]

9.2.5.2 - @Pessoa

É um tipo de entrevista/pergunta em que se exibe um campo texto para que o usuário forneça a matrícula / sigla do funcionário, nos mesmos moldes que a Lotação.

Variáveis criadas:

```
nomedavariavelpassada_pessoaSel.id
nomedavariavelpassada_pessoaSel.sigla
nomedavariavelpassada_pessoaSel.descricao
```

Exemplo:

```
[@pessoa titulo="matrícula" var="matricula" reler=true idAjax="pessoaAjax"/]
```

9.2.5.3 - @Funcao

É um tipo de entrevista/pergunta em que se exibe um campo texto para que o usuário forneça a função do funcionário, nos mesmos moldes que a Lotação.

Variáveis criadas:

```
nomedavariavelpassada_funcaoSel.id
nomedavariavelpassada_funcaoSel.sigla
nomedavariavelpassada_funcaoSel.descricao
```

Exemplo:

```
[@funcao titulo="Função" var="funcao" reler=true idAjax="funcaoAjax"/]
```

9.2.6 - @Checkbox

É um tipo de entrevista/pergunta em que se exibe um campo tipo checkbox.

Parâmetros:

Parâmetro	Descrição	Default
titulo	O texto (label) que vai após ao campo checkbox	titulo=""
var	O nome da variável que reterá o texto	
default	Sim ou Não. Com sim o checkbox aparece checado	default="Nao"
idAjax	se fornecido, representa a identificação do Ajax que será utilizado num bloco @Grupo depende visto anteriormente	idAjax=""
Reler	reler=true reler=false Obs: se reler=true e idajax for passado, então assume-se AJAX	reler=false Ou seja, não.
onclick	Aplicação JS que pode ser passada	onclick=""

	quando o botão checkbox é clicado. O default é ""	
obrigatorio	Poder ser: true ou false . Observação: Se um campo for definido como obrigatório o seu efeito não será imediato, ou seja, o usuário poderá deixá-lo em branco. Somente no momento em que for solicitado a geração do documento é que o sistema não permitirá que o mesmo fique em branco.	obrigatorio=false

JavaScripts:

Rotina	Evento que dispara	Descrição
<pre>if (this.checked) document.getElementById('nomevar').value = 'Sim'; else document.getElementById('nomevar').value = 'Nao';</pre> <p>Onde <i>nomevar</i> é o nome da variável passada a macro @checkbox</p>	onclick	<p>Se o Box está ticado Então: a variável recebe o valor "sim" Senão: a variável recebe o valor "não"</p>

Exemplo:

```
[@checkbox var="sublotacoes" titulo="Incluir Sublotações" default="Sim"
reler=true idAjax="sublotacoesAjax"/]
```

```
[@grupo depende="sublotacoesAjax"]
[#if (sublotacoes!="") == "Sim"]
    [#assign buscarSublotacoes = true/]
[#else]
    [#assign buscarSublotacoes = false/]
[/#if]
```

O que podemos passar para o onclick?

A - Emitir um alerta do JS com alguma explicação adicional

```
[@checkbox titulo="teste" var="varcheck" default="Não" valor="sim" onclick="
Alert('aloaloalo')"/]
```

Onde temos aloaloalo, poderia ser \${variavel_da_aplicacao}

B - Abrindo uma janela e exibindo o conteúdo de uma string ou variável:

```
[@checkbox titulo="teste" var="varcheck" default="Não" valor="sim" onclick="try
{newwin.close();} catch(err) {} newwin =
window.open('teste2',null,'height=160,width=400,status=no,toolbar=no,menubar=no,location=no');
newwin.document.write('aloaloalo')"/]
```

Onde temos aloaloalo, poderia ser \${variavel_da_aplicacao}

9.2.7 - @Data

É um tipo de entrevista/pergunta em que se exibe um campo texto para que o usuário forneça uma data no formato dd/mm/aaaa.

Parâmetros:

Parâmetro	Descrição	Default
titulo	O texto que antecede o campo de data	
var	O nome da variável que reterá a data	
reler	reler=true reler=false Obs: se reler=true e idajax for passado, então assume-se AJAX	reler=false Ou seja, não.
idAjax		idAjax=""
default		default=""
alerta		alerta=false
obrigatorio	Poder ser: true ou false . Observação: Se um campo for definido como obrigatório o seu efeito não será imediato, ou seja, o usuário poderá deixá-lo em branco. Somente no momento em que for solicitado a geração do documento é que o sistema não permitirá que o mesmo fique em branco.	obrigatorio=false

JavaScripts:

Rotina	Evento que dispara	Descrição
verifica_data em static_javascript.js	onblur	Realiza a crítica da data, exibindo um alert caso a data seja inválida

A macro @data possui uma rotina em JS, chamada verifica_data, que é chamada no evento onchange. Esta rotina critica a data.

Exemplo:

```
[@data var="x" titulo="A partir de: "/]
```

9.2.8 - @Memo

É um tipo de entrevista/pergunta em que se exibe um campo texto tipo memo.

Parâmetros:

Parâmetro	Descrição	Default
titulo	O texto que antecede o campo memo	
var	O nome da variável que reterá o texto	
colunas	Número de colunas do memo	
linhas	Número de linhas do memo	
obrigatorio	Poder ser: true ou false . Observação: Se um campo for definido como obrigatório o seu efeito não será imediato, ou seja, o usuário poderá deixá-lo em branco. Somente no momento em que for solicitado a geração do documento é que o sistema não permitirá que o mesmo fique em	obrigatorio=false

	branco.	
default		default=""
reler	Só tem a opção false, pois não implementa AJAX	reler=false

Exemplo:

```
[@memo var="informacoes" titulo="Informações" colunas="63" linhas="3" /]
```

9.2.9 - @Oculto

É a forma de se manter / guardar variáveis no nível do formulário, sem que as mesmas sejam vistas pelo o usuário. Esta técnica é importante para se manter variáveis entre sessões, como no caso de um refresh de tela (F5), por exemplo.

Parâmetros:

Parâmetro	Descrição	Default
var	O nome da variável que reterá o valor	
valor	O valor a ser passado	valor=""
default		default=""

Exemplo:

```
[@oculto var="x" valor="{y}"/]
```

Neste caso, a variável x, oculta no formulário, conterá o valor da variável y. Pode-se passar também uma constante, tal como: valor="300".

9.2.10 - Macros de formatação de linha (sem parâmetros)

Parâmetros:

Macro	Descrição
@br	Causa a quebra de linha. O equivalente HTML tag é
@separador	Cria uma linha horizontal branca que é utilizada para dar destaque a títulos ou para gerar a sensação de quebra entre um item de informação e outro. O equivalente HTML tag é <hr/>

9.2.11 - Macros (Funções) utilitárias

9.2.11.1 - Formatar CPF (formatarCPF)

Descrição:

Esta função obtém uma string contendo o CPF a ser formatado e o devolve formatado com a seguinte apresentação: 999.999.999-99

Pré-condições:

A string passada como CPF pode conter quaisquer caracteres numéricos e não numéricos, visto que a função irá colher somente os dígitos numéricos. Se a quantidade de dígitos numéricos for menor que 11 isto indica que possivelmente não foram passados os zeros a esquerda e neste caso, a função introduzirá a quantidade necessária de zeros a esquerda.

Exemplos

String passada	String considerada	String retornada
4751084679	47510846749	475.108.467-49
475.108.467/49	47510846749	475.108.467-49

510846749
475xpto108bb46749

00510846748
47510846749

005.108.467-49
475.108.467-49

Condições de erro:

Se a string for: nula ou maior que 11 ou menor que 3 dígitos numéricos, será retornado "erro" pela função.

Parâmetros passados:

Parâmetro	Obrigatório	Tipo/Tam	Default	Descrição
	String	Sim		O CPF a ser formatado

Parâmetros devolvidos:

Parâmetro	Obrigatório	Tipo/Tam	Default	Descrição
	String	Sim		O CPF formatado ou "erro"

Chamada da function:

A chamada nas aplicações poderá ser algo tipo:

```
[#assign antigoCPF = "47510846749"  
[#assign novoCPF = formatarCPF(antigoCPF)/]  
[#if novoCPF == "erro"]  
    Tratar o erro, pois o CPF fornecido está inconsistente  
[/#if]
```

Se exibirmos o novoCPF `${novoCPF}` obteremos 475.108.467-49

9.2.11.2 - Validar CPF (*validarCPF*)

Descrição:

A validação do CPF se dá pelo confronto dos dígitos verificadores passado a função e calculados. O cálculo baseia-se em aplicar o módulo 11 nos primeiros 9 dígitos do CPF, obtendo-se assim o primeiro dígito verificador. Depois aplica-se o módulo 11 novamente nos primeiros 9 dígitos e no primeiro dígito verificador. O site <http://br.answers.yahoo.com/question/index?qid=20060829190814AAHOqY6> possui uma descrição detalhada de como o algoritmo deve funcionar.

Condições de erro:

Se a string for nula ou o dígito verificador calculado não bater com o fornecido a função retornará False.

Pré-condições:

O CPF deve estar no formato xxx.xxx.xxx-xx

Parâmetros passados:

Parâmetro	Obrigatório	Tipo/Tam	Default	Descrição
	String	Sim		O CPF a ser validado no formato xxx.xxx.xxx-xx

Parâmetros devolvidos:

Parâmetro	Obrigatório	Tipo/Tam	Default	Descrição
	Boolean	Sim		True ? CPF OK False ? CPF inválido

Chamada da function:

A chamada nas aplicações poderá ser algo tipo:

```
[#assign testeCPF = validarCPF(CPFaservalidado)/]
```



```
[#if !testeCPF]
    Tratar o erro, pois o CPF fornecido é inválido
[/#if]
```

9.2.11.3 - Formatar e Validar CPF (fmtvldCPF)

Descrição:

Esta função obtém uma string contendo o CPF a ser formatado e validado e o devolve formatado com a seguinte apresentação: 999.999.999-99

Pré-condições:

A string passada como CPF pode conter quaisquer caracteres numéricos e não numéricos, visto que a função irá colher somente os dígitos numéricos.

Se a quantidade de dígitos numéricos for menor que 11 isto indica que possivelmente não foram passados os zeros a esquerda e neste caso, a função introduzirá a quantidade necessária de zeros a esquerda.

Exemplos

String passada	String considerada	String retornada
4751084679	47510846749	475.108.467-49
475.108.467/49	47510846749	475.108.467-49
510846749	00510846748	005.108.467-49
475xpto108bb46749	47510846749	475.108.467-49

Condições de erro:

Se a string for: nula ou maior que 11 ou menor que 3 dígitos numéricos, será retornado "E1" pela função, indicando que o CPF passado não pode ser formatado.

Se o dígito verificador calculado não bater com dígito passado será retornado "E2" pela função, indicando que o CPF é inválido.

Parâmetros passados:

Parâmetro	Obrigatório	Tipo/Tam	Default	Descrição
	String	Sim		O CPF a ser formatado e validado no formato xxx.xxx.xxx-xx

Parâmetros devolvidos:

Parâmetro	Obrigatório	Tipo/Tam	Default	Descrição
	String	Sim		O CPF formatado ou os seguintes tipos de erro: E1 - O CPF passado não pode ser formatado E2 - O CPF passado é inválido

Chamada da function:

A chamada nas aplicações poderá ser algo do tipo:

```
[#assign antigoCPF = "47510846749"/]
[#assign novoCPF = fmtvldCPF(antigoCPF)/]
[#if novoCPF == "E1"]
    Tratar o erro, pois o CPF fornecido está inconsistente e não pode ser
    formatado
[#elseif novoCPF == "E2"]
    Tratar o erro, pois o CPF fornecido é inválido, ou seja, possui dígitos
    verificadores inválidos
[#else]
    Utilizar o novoCPF
[/#if]
```

Se exibirmos o novoCPF \${novoCPF} obteremos 475.108.467-49

9.2.12 - Obtendo dados do servidor/funcionário

Para tanto, o servidor deve estar instanciado, caso contrário não teremos acesso e poderemos receber uma exception do tipo undefined (ver capítulo "Lições Aprendidas").

Para testarmos se o servidor está instanciado, podemos usar as seguintes variáveis do SIGA-DOC (ver também item 16):

tipoDestinatario : Pode assumir os valores: 1, 2, 3 ou 4 dependendo da seleção: matrícula, órgão interno, órgão externo e campo livre.

Doc.destinatario: que conterá a matrícula, UO interna, UO externa ou qualquer coisa quando a opção for campo livre

Neste caso o servidor está instanciado

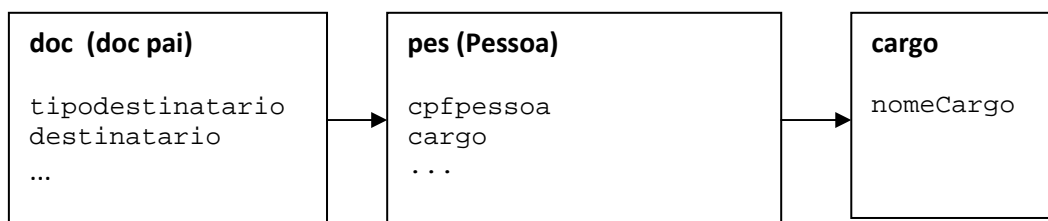
Como podemos saber via código se o servidor está instanciado?

```
[#if (tipoDestinatario! == "1") && (doc.destinatario??)]  
Então o servidor está instanciado  
[/#if]
```

Alguns exemplos de variáveis que podemos acessar:

```
[#assign matricula = doc.destinatario.sigla!/]  
[#assign nome = doc.destinatario.nomePessoa!/]  
[#assign data_Nascimento = doc.destinatario.dataNascimento!/]  
[#assign tipo_Sanguineo = doc.destinatario.tipoSanguineo!/]  
[#assign naturalidade = doc.destinatario.naturalidade!/]  
[#assign nacionalidade = doc.destinatario.nacionalidade!/]  
[#assign CPF_servidor = doc.destinatario.cpfPessoa?string/]  
[#assign natureza_AtoNomeação = doc.destinatario.atoNomeacao!/]  
[#assign data_Posse = doc.destinatario.dataPosse!/]  
[#assign data_Exercicio = doc.destinatario.dataExercicioPessoa!/]  
[#assignCodigo_cargo = doc.destinatario.cargo!/]  
[#assignDescricao_cargo = doc.destinatario.cargo.nomeCargo!/]  
[#assign padrao = doc.destinatario.padraoReferencia!/]  
[#assign data_PublicacaoPosse = doc.destinatario.dataPublicacao!/]
```

As Classes e seus relacionamentos



doc.destinatario.cpfPessoa
doc.destinatario.cargo

doc.destinatario.cargo.nomeCargo

Algumas variáveis mapeadas no **Hibernate**:

Nome do atributo	Nome da coluna no BD	Tipo de dados
Deverá ser prefixado com doc.destinatario. Exemplo: doc.destinatario.cpfPessoa		
idPessoaIni	ID_PESSOA_INICIAL	long
dataFimPessoa	DATA_FIM_PESSOA	java.util.Date
dataInicioPessoa	DATA_INI_PESSOA	java.util.Date
idePessoa	IDE_PESSOA	string
dataNascimento	DATA_NASC_PESSOA	date
nomePessoa	NOME_PESSOA	string
cpfPessoa	CPF_PESSOA	long
matricula	MATRICULA	long
sesbPessoa	SESB_PESSOA	string
emailPessoa	EMAIL_PESSOA	string
siglaPessoa	SIGLA_PESSOA	string
padraoReferencia	DSC_PADRAO_REFERENCIA_PESSOA	string
nomePessoaAI	REMOVE_ACENTO(NOME_PESSOA) *formula	string
situacaoFuncionalPessoa	SITUACAO_FUNCIONAL_PESSOA	string
dataExercicioPessoa	DATA_INICIO_EXERCICIO_PESSOA	date
atoNomeacao	ATO_NOMEACAO_PESSOA	string
dataNomeacao	DATA_NOMEACAO_PESSOA	date
dataPosse	DATA_POSSE_PESSOA	date
dataPublicacao	DATA_PUBLICACAO_PESSOA	date
grauInstrucao	GRAU_INSTRUCAO_PESSOA	string
idProvimento	ID_PROVIMENTO	integer
nacionalidade	NACIONALIDADE_PESSOA	string
naturalidade	NATURALIDADE_PESSOA	string
imprimeEndereco	FG_IMPRIME_END	string
sexo	SEXO_PESSOA	string
tipoServidor	TP_SERVIDOR_PESSOA	integer
tipoSanguineo	TP_SANGUINEO_PESSOA	string
endereco	ENDERECO_PESSOA	string
bairro	BAIRRO_PESSOA	string
cidade	CIDADE_PESSOA	string
cep	CEP_PESSOA	string
telefone	TELEFONE_PESSOA	string
Identidade	RG_PESSOA	string
orgaoIdentidade	RG_ORGAO_PESSOA	string
dataExpedicaoIdentidade	RG_DATA_EXPEDICAO_PESSOA	date
ufIdentidade	RG_UF_PESSOA	string
idEstadoCivil	ID_ESTADO_CIVIL	integer
nomeExibicao	NOME_EXIBICAO	string

9.2.13 - @Radio

É um tipo de entrevista/pergunta em que se exibe um campo tipo Radio button.

Parâmetros:

Parâmetro	Descrição	Default
titulo	O texto posterior ao campo radio	
var	O nome da variável que reterá a resposta, conforme o valor	

	estabelecido na variável valor	
valor	O valor	valor="Sim"
default	Sim ou não. Com sim o radio aparece checado O default é não.	default="Não"
idAjax	se fornecido, representa a identificação do Ajax que será utilizado num bloco <i>@Grupo depende</i> visto anteriormente	idAjax=""
reler	reler=true reler=false Obs: se reler=true e idajax for passado, então assume-se AJAX	reler=false Ou seja, não.
onclick	Aplicação JS que pode ser passada quando o botão radio é clicado. O default é "".	onclick=""

JavaScripts:


Rotina	Evento que dispara	Descrição
<pre>if (this.checked) document.getElementById('nomevar').value = 'valorvar';</pre> <p>Onde: <i>nomevar</i> é o nome da variável passada a macro @radio <i>valorvar</i> é o valor passado a macro @radio</p>	onclick	Se o Radio está ticado Então: a variável <i>nomevar</i> recebe o valor passado a macro @radio

Exemplo:

```
[@entrevista]
[@grupo titulo="A SEC está na Programação Anual?"]
[@radio titulo="Sim" var="radio_resp" valor="1" default="Sim" /]
[@radio titulo="Não" var="radio_resp" valor="0" /]
[/@grupo]
[#if (radio_resp=="0")]
[#assign varSN="Não" /]
[#else]
[#assign varSN=""]
[/#if]
[/@entrevista]
```

A SEC está na Programação Anual?

☒ Sim
 ☐ Não



Observações: As macros @radio devem estar dentro de um mesmo bloco @grupo, caso elas sejam do mesmo tipo, ou seja, possuem o mesmo nome de var (aqui, radio_resp).

O que podemos passar para o onclick?

A - Emitir um alerta do JS com alguma explicação adicional

```
[@radio titulo="teste" var="varradio" default="Não" valor="sim" onclick="Alert('aloaloalo')"/]
```

Onde temos aloaloalo, poderia ser \${variavel_da_aplicacao}

B - Abrindo uma janela e exibindo o conteúdo de uma string ou variável:

```
[@radio titulo="teste" var="varradio" default="Não" valor="sim" onclick="try  
{newwin.close();} catch(err) {} newwin =  
window.open('teste2',null,'height=160,width=400,status=no,toolbar=no,menubar=no,location=no'); newwin.document.write('aloaloalo')"/]
```

Onde temos aloaloalo, poderia ser \${variavel_da_aplicacao}