

V 1.2

FREEMARKER NO SIGA-DOC

RESUMO

O FREEMARKER É UMA LINGUAGEM PARA CRIAÇÃO DE PÁGINAS WEB. ELA É BEM DOCUMENTADA E QUALQUER INFORMAÇÃO PODE SER OBTIDA NO SITE OFICIAL <http://freemarker.sourceforge.net/>. PORÉM, QUANDO O PROGRAMADOR PENSA QUE SOMENTE O CONHECIMENTO DA LINGUAGEM É SUFICIENTE PARA COMEÇAR A DESENVOLVER AS APLICAÇÕES PARA O SIGA-DOC, ELE SE FRUSTRA, VISTO QUE O AMBIENTE OU INFRAESTRUTURA DISPONIBILIZADO A ELE ESTÁ TOTALMENTE CUSTOMIZADO OU PARAMETRIZADO, E O PIOR, NÃO EXISTE MUITA DOCUMENTAÇÃO A RESPEITO. NORMALMENTE ELE NECESSITARÁ DE:

- SEGUIR AS NORMAS (BEST PRACTICES) DE PROGRAMAÇÃO;
- CONHECER AS MACROS E FUNCTIONS QUE ESTÃO DISPONÍVEIS;
- CONHECER O MODELO DE DADOS DO FREEMARKER;
- CONHECER OS MÉTODOS QUE ESTÃO DISPONÍVEIS;
- CONHECER O CONCEITO DO NEGÓCIO

Este pequeno manual tenta auxiliar o programador nos primeiros passos da programação do freemarker para o ambiente do SIGA-DOC.

Revisado por	Versão	Data
	1.0	17/04/2012
	1.1	21/08/2012
	1.2	20/04/2013

Ruben
SJRJ
13/04/2012



ÍNDICE

<u>Capítulo 1 - Introdução</u>	
1.1 - Conceitos Gerais	
1.2 - O Conceito de Documento	
1.3 - Transição de Estado do Documento	
1.4 - Comportamento de uma Aplicação WEB	
1.5 - Campos da Entrevista que provocam a leitura da Tela	
1.6 - Sintaxe do FM	
<u>Capítulo 2 - Macros do FreeMarker no SIGA-DOC</u>	
2.1 - Introdução	
2.2 - Classificação das macros	
2.3 - Componentes do bloco Entrevista e do bloco Documento	
2.4 - Categorização das macros	
2.5 - Lista de Macros e Functions (Posição em 26/04/2012)	
<u>3 - Programação Cliente - JS e outros</u>	
3.1 - Introdução	
3.2 - JavaScript- JS	
3.2.1 - Um método é pelo local onde elas estão armazenadas / definidas	
3.2.2 - Pelos depuradores de código (firebug, por exemplo)	
3.2.3 - Pela classificação funcional	
3.2.4 - Nonominal	
3.3 - Document Object Model (DOM)	
3.3.1 - Eventos HTML DOM	
3.3.2 - Elementos HTML DOM	
3.4 - AJAX	
3.4.1 - Modelo 1 - AJAX em ação no SIGA-DOC nos campos subscritor, titular, destinatário e classificação da parte fixa da entrevista	
3.4.2 - Modelo 2 - AJAX em ação no SIGA-DOC nos campos definidos Com idAjax e macro FM Grupo com a clausula depende nos campos da parte variável da entrevista	
3.4.3 - Modelo 3 - AJAX SIGA-DOC um caso específico ainda não atendido	
3.4.4 - Modelo 4 - AJAX SIGA-DOC Customizando a customização	
3.5 - CSS	
3.5.1 - Introdução	
3.5.2 - o SIGA-DOC e o CSS	
3.5.3 - Revisitando o código do CPF e customizando-o para colocar a mensagem ao lado do campo	
3.6 - Applet	
3.7 - Tendências	
3.7.1 - HTML 4.01, XHTML 1.1 e 2.0, HTML 5.0, XML, DHTML?	
3.7.2 - jQuery	
3.7.3 - JSON	
3.7.3.1 - JSON, JS e AJAX	

3.7.3.2 - Tipos de dados JSON	
3.7.3.3 - JSON, JQUERY e AJAX	
3.8 - Funções JS disponíveis ao programador (posição de 07/05/2012)	
3.8.1 - No arquivo static-javascript.js	
3.8.2 - Na página editar.action (tela da entrevista)	
3.8.3 - Na página buscar.action	
3.8.4 - No arquivo Ajax.js	
3.8.5 - No arquivo exibir.action	
3.9 - Mapeamento: Campos da parte fixa da entrevista (editar.action) e funções javaScript	
<u>4 - O DATA-MODEL DISPONIBILIZADO</u>	
4.1 - O que é um Data-Model	
4.2 - O Data-Model disponibilizado	
4.3 - Como acessar as variáveis do Data-Model	
4.4 - Como criar um Data-Model	
4.4.1 - Criando o hash pelo Java	
4.4.1.1 - Criando um MAP	
4.4.1.2 - Criando uma Classe e agregando ao hash	
4.5 - Mapa da Mina - Mapeamento dos campos da parte fixa da entrevista que estão no hash doc	
<u>5 - HASHES DO DATA-MODEL E SEUS MÉTODOS</u>	
5.1 - VARIÁVEIS DO HASH FUNC - CLASSE FuncoesEl.java	
5.2 - VARIÁVEIS DO HASH EXBL - Classe ExBL.java	
5.3 - VARIÁVEIS DO HASH DOC - Classe ExDocumento.java	
5.4 - VARIÁVEIS DO HASH PARAM	
<u>6 - LIÇÕES APRENDIDAS</u>	
6.1 - Erros mais comuns no FM pelo não entendimento do funcionamento do mesmo	
6.1.1 - Variáveis não iniciadas: por que tivemos que mudar o código?	
6.1.2 - Estamos sempre rodando do zero: então é impossível saber se é a primeira rodada ou não?	
6.1.3 - Outros problemas no processamento WEB Tradicional que observamos no SIGA-DOC	
6.2 - Manuseio de variáveis no FM: Existência	
6.3 - Visibilidade das variáveis no FM	
6.4 - Erros mais comuns no FM envolvendo nomes de variáveis	
6.4.1 - Macros com assign	
6.4.2 - Variável com o mesmo nome da macro ou function	
6.5 - Criando e manipulando Sequences no FM	
6.6 - Criando variáveis ("e Arrays") dinamicamente	
6.7 - Mantendo as variáveis entre sessões - Burlando o SATELESS	
6.8 - Geração do PDF - Problema com as Fontes	
6.9 - Geração HTML - Problema com Tabelas	
<u>7 - MELHORES PRÁTICAS</u>	

7.1 - Nome de Variáveis	
7.2 - Criação de variáveis Local X Assign	
7.3 - IMPORT X INCLUDE	
7.4 - Documentação	
7.5 - Estrutura Modular	
7.6 - Prefira Case a Ifs	
<u>8 - PALAVRAS RESERVADAS</u>	
8.1 - Palavras oriundas da biblioteca de macros do FM	
8.2 - Palavras oriundas dos nomes dos campos do formulário de entrevista do SIGA-DOC	
8.3 - IDs oriundos dos elementos HTML do formulário de entrevista do SIGA-DOC	
<u>9 - MACROS EM AÇÃO - PROGRAMAÇÃO POR EXEMPLOS</u>	
9.1 - @Entrevista e @Documento - Estrutura da aplicação FM para o SIGA-DOC	
9.1.1 - @Entrevista	
9.1.2 - @Documento	
9.1.3 - Especial	
9.2 - TRABALHANDO NO BLOCO @entrevista	
9.2.1 - @Grupo	
9.2.2 - @Selecao	
9.2.3 - @Texto	
9.2.4 - @Mensagem	
9.2.5 - Macros de negócio	
9.2.5.1 - @Lotacao	
9.2.5.2 - @Pessoa	
9.2.5.3 - @Funcao	
9.2.6 - @Checkbox	
9.2.7 - @Data	
9.2.8 - @Memo	
9.2.9 - @Oculto	
9.2.10 - Macros de formatação de linha (sem parâmetros)	
9.2.11 - Macros (Funções) utilitárias	
9.2.11.1 - Formatar CPF (formatarCPF)	
9.2.11.2 - Validar CPF (validarCPF)	
9.2.11.3 - Formatar e Validar CPF (fmtvldCPF)	
9.2.12 - Obtendo dados do servidor/funcionário	
9.2.13 - @Radio	
9.2.14 - Aplicação utilizando todas as macros da Entrevista	
9.2.14.1 - Aplicação FM	
9.2.14.2 - Formulário	
9.2.14.3 - Estrutura HTML da Aplicação	
9.2.14.4 - HTML da Aplicação	
9.2.14.5 - Como ficaria aplicação sem o [#grupo]	
9.3 - COMENTÁRIOS SOBRE A MACRO @ENTREVISTA - Responsável pela entrevista	
9.3.1 - Tabelas não possuem macros, como tratá-las?	
9.3.2 - Entrevista aninhada e variáveis criadas dinamicamente	
9.4 - TRABALHANDO NO BLOCO @documento	
9.4.1 - Textos e tabelas	
9.4.2 - Formatação: Cabeçalhos e Rodapés e outros	
9.4.2.1 - Utilizando a macro formulário (nível 4)	

9.4.2.2 - Utilizando a macro memorando (nível 4)	
9.4.2.3 - Utilizando a macro portaria(nível 4)	
9.4.2.4 - Estilos de Brasão	
9.4.2.5 - Montando uma macro (nível 4) para o documento	
A - Utilizando o estilo Brasão a Esquerda	
B - Utilizando o estilo Brasão Centralizado	

9.5 - FAZENDO MACROS

10 - DEBUGANDO A APLICAÇÃO

10.1 - Depuração no servidor

10.1.1- FM	
10.1.1.1 - A MACRO @dumpall (sem parâmetro)	
10.1.1.2 - Depurando o template (aplicação) FM utilizando o utilitário Integrador	
10.1.2- Depurando os métodos JAVA	

10.2 - Depuração no cliente

10.2.1 - Firebug	
10.2.2 - HTML / Estilo (CSS)	
10.2.3 - HTML / Exibição (Layout) O BOX MODEL	
10.2.4 - Script	
10.2.5 - DOM	
10.2.6 - Rede	
10.2.7 - Console	

11 - DISSECANDO O FORMULÁRIO DE ENTREVISTA

11.1 - Estrutura geral do formulário do SIGA-DOC

11.1.1 - Análise das tabelas	
------------------------------	--

11.2 - IDs por ordem de aparição e elementos HTML

11.3 - CLASSES CSS

11.4 - NOMES por ordem de aparição e elementos HTML

11.5 - CÓDIGO HTML

12 - AMBIENTE DE DESENVOLVIMENTO

12.1 - O Ambiente de Desenvolvimento

12.2 - O Ambiente de Desenvolvimento do FM

12.2.1 - Ambientes da SJRJ	
12.2.2 - Procedimentos para criar e manter uma aplicação FM	

12.3 - Criando uma aplicação FM

12.3.1 - Criando uma aplicação FM	
12.3.2 - Criando uma macro FM	

12.4 - Backup, Restore e comparando códigos FM

12.5 - Sintaxe HighLight para o FM no JBoss Developer Studio (JBDS)

12.5.1 - Criando um arquivo .ftl	
12.5.2 - Tela Principal do editor	
12.5.3 - Acusando o erro	
12.5.4 - Sintaxe dos comandos	

12.6 - INTEGRADOR

12.6.1 - Introdução	
12.6.2 - Setup do Ambiente	
12.6.3 - Funcionalidades do Aplicativo	
12.6.3.1 - Chamada do aplicativo	
12.6.3.2 - Tela Inicial	

12.6.3.3 - Ajuda	
12.6.3.4 - Opção / Executar	
12.6.3.5 - Opção / Registrar	
12.6.3.6 - Opção / Carregar	
12.6.3.7 - Opção / Editar	

13 – RODANDO O FREEMARKER NO JBDS – A ORIGEM

13.1 – INTRODUÇÃO

13.2 – Passo-a-Passo

13.2.1 - Criar um novo Workspace	
13.2.2 - Criar um projeto	
13.2.3 - Popular o projeto	
13.2.4 - Relacionamento, descrição dos arquivos e como funciona o projeto	
13.2.4.1 - O usuário digita no browser	
13.2.4.2 - O servelt do projeto entra em ação	
13.2.4.3 - O interpretador FM entra em ação	
13.2.4.4 - O usuário preenche os campos e submete (save) a página	
13.2.5 - Conteúdo dos arquivos utilizados	

13.3 – Entendendo HttpServlet, HttpServletRequest e HttpServletResponse

13.3.1 - Os métodos GET e POST com os respectivos request e response	
13.3.2 - A Classe abstrata HttpServlet	
13.3.3 - Os Objetos HttpServletRequest e HttpServletResponse	
13.3.4 - Escopo de Variáveis e Objetos	
16.3.5 - JAVA - Setando e Recuperando - Escopo de Variáveis e Objetos	
16.3.6 - FREEMARKER - Recuperando - Escopo de Variáveis e Objetos	

14 – CLASSE E DADOS

14.1 – Introdução

14.2 – ENTIDADES/TABELAS

Modelo ER – Sistema Dp	
Modelo ER – Sistema Cp	
Modelo ER – Sistema Ex	

14.3 – CLASSES

14.3.1 - Introdução	
Classe concreta x abstrata x interface	
Sobre a documentação das classes	
Sobre o javaDoc	
14.3.2 - Descrição das classes	
Modelo de Classes – Sistema Dp	
Modelo de Classes – Sistema Cp	
Modelo de Classes – Sistema Ex	

15 – ANEXOS

Anexo 1 – HTML do SIGA-DOC

Anexo 2 – Instalando o SQLDeveloper

Anexo 3 – Utilizando o JBOSS Developer Studio

3.1 - Pesquisando uma classe (Ctrl+Shift+R)	
3.2 - OpenOn - Navegando nas classes - (Cursor na classe e F3 ou Ctrl+click)	
3.3 - Ajuda Comando - (Ctrl+Space)	
3.4 - Encontrar os atributos / métodos dentro da classe-(Crtl+O)	

<p>3.5 - Hierarquia das classes (Open type Hierarchy - F4) 3.6 - Onde é referenciado (Open Call Hierarchy - Ctrl+Alt+H) 3.7 - Debugando um método JAVA 3.8 - Limpando Projetos Publicados 3.9 - Link do Editor com o Package Explorer (Tree)</p>	
<p>Anexo 4 - Disponibilizando uma função TLD para acesso pelo FM (ou JSP)</p> <p>4.1 - O que é TLD? 4.2 - Qual é a assinatura do método? 4.3 - Incluindo a nova function no .tld (no caso, Arquivo func.tld) 4.4 - Mapear o arquivo func.tld no WEB.XML, caso não esteja mapeado 4.5 - Utilização no FM 4.6 - Utilização no JSP 4.7 - Diagramaticamente</p>	
<p>Anexo 5 - Filtros no SIGA-DOC</p> <p>5.1 - Relação dos filtros no SIGA-DOC 5.2 - DEFINIÇÃO 5.2.1 - A interface Filter 5.2.2 - Um objeto FilterConfig 5.2.3 - Um objeto Filter Chain</p>	
<p>Anexo 6 - Listener no SIGA-DOC</p> <p>6.1 - Relação dos Listeners no SIGA-DOC 6.2 - Definição 6.3 - Os tipos de Listener 6.4 - Exemplo</p>	
<p>Anexo 7 - GIT - Conceitos</p> <p>7.1 - Introdução 7.2 - Estrutura do Git 7.3 - Breve comentário sobre os principais comando Git 7.3.1 - Trabalhando com repositório local 7.3.2 - Trabalhando com repositório remoto SITUAÇÃO 0 - Criando o Repositório Remoto SITUAÇÃO 1 - Clonando um Repositório Remoto com CLONE SITUAÇÃO 2 - Realizando mudanças no branch TB SITUAÇÃO 3 - Modificações no Repositório Remoto SITUAÇÃO 4 - Novas atualizações no Repositório Remoto com FETCH SITUAÇÃO 5 - Publicando atualizações no Repositório Remoto com PUSH 7.4 - O repositório do Git do disco local c:\ 7.5 - Integração Git e IDE (no caso, o Eclipse)</p>	
<p>Anexo 8 - GIT no Eclipse/JBDS - EGIT</p> <p>8.1 - Visão geral pelo menu do produto 8.2 - Projetos gerenciados e não gerenciados pelo GIT 8.3 - SHARE PROJECT (Criando um repositório) 8.4 - ADICIONANDO O PROJETO / ARQUIVOS PARA CONTROLE DE VERSÃO 8.5 - RETIRANDO ARQUIVOS DO CONTROLE DE VERSÃO 8.6 - COMMITANDO O PROJETO 8.7 - HISTÓRIA DO PROJETO 8.7.1 - Clicando no primeiro commit 8.7.2 - Clicando no segundo commit 8.7.3 - Outras Opções quando analisando o History de um projeto 8.7.3.1 - Show All Changes in Repository containing the selected resource 8.7.3.2 - Show All Changes in Project containing the selected resource 8.7.3.3 - Show All Changes in parent Folder of the selected resource 8.7.3.4 - Show All Changes of selected resource and its children 8.8 - Analisando Repositórios Git</p>	

8.9 -	Sincronização
8.10 -	Comparar Conteúdos – Compare with
8.10.1 -	Comparando a Working Tree com o último commit (HEAD Revision)
8.10.2 -	Comparando Working Tree com Git Index
8.10.3 -	Comparando Working Tree com um branch, uma tag ou uma reference
8.10.4 -	Comparando Working Tree com qualquer Commit
8.10.5 -	Comparando Dois Commits
8.10.6 -	Comparando Working Tree com Local History
8.10.7 -	Comparando Working Tree com History
8.10.8 -	Comparando Index com HEAD ou qualquer outro Commit

Anexo 9 – Instalando o plugin do Freemarker no Eclipse

9.1 -	Instalando manualmente
9.2 -	Instalando através do Eclipse installer/updater system
9.3 -	Atualizando a versão do plugin

Anexo 10 – Utilitários baseados no Freemarker

10.1 -	FMGENERATOR
10.2	RTF GENERATOR

Anexo 11 – Procedimento para atualizar o repositório Git com fontes JSPs

11.1 -	Obter os fontes JSP (ou JAVA) mais recentes via PULL
11.2 -	Criar um branch (ramificação, desvio ...) para atualizar os fontes JSP
11.3 -	Alterar o(s) fonte(s) JSP
11.4 -	Salvar (Commitar) as mudanças
11.5 -	Retornar (SWITCH) ao branch master
11.6 -	Aplicar as mudanças do branch de trabalho no branch máster via MERGE
11.7 -	Realizar o PULL novamente
11.8 -	Realizar as mudanças no repositório remoto (upstream) com o PUSH
11.9 -	Quando as mudanças terão efeito?
11.10 -	Posso excluir o branch de trabalho (JSP)?

Observações:

A -	O procedimento acima é para funcionários da SJRJ
B -	Configurando as credenciais e os endereços do repositório GIT
B.1 -	Utilize o procedimento abaixo para verificar os endereços Dos repositórios
B.2 -	Verificando as credenciais para acessar o repositório remoto
B.3 -	Autorização para acessar o projeto e Conta no GoogleCode
C -	Revertendo um código ao original

Anexo 12 – Instalando o DBVisualizer

12.1 -	Instalando o produto
12.2 -	Configurando a conexão

Anexo 13 – Instalando e configurando JBDS

Anexo 14 – Instalando e configurando o ObjectAid e UML Doclet (ex yDocs)

14.1 -	Instalando os produtos
14.1.1 -	ObjectAid
14.1.2 -	UML Doclet (ex yDoc)
14.2 -	Utilizando os produtos
14.2.1 -	ObjectAid
14.2.2 -	UML Doclet
14.2.2.1 -	Tipos de Diagrama
14.2.2.2 -	Destino da documentação

- | | |
|--|--|
| 14.2.2.3 - Formato da imagem | |
| 14.2.2.4 - Estilos de formatação | |
| 14.2.2.5 - Alterando o estilo | |
| 14.2.2.6 - Customizando um estilo | |
| 14.2.2.7 - Arquivo onde reside a configuração do UMLDoclet | |

O que é o SIGA-DOC? O que é FreeMarker? O que é um documento? Como podemos programar em freemarker para gerar documentos no ambiente do SIGA-DOC?

Este capítulo fala um pouco de tecnologia e de negócio com intuito de não entrarmos abruptamente na geração de código.

1.1 – CONCEITOS GERAIS

A conceituação que se segue é stricto sensu e visa a programação da interface do SIGA-DOC. Caso você deseje uma conceituação mais ampla, lato sensu, acesse <http://projeto-siga.googlecode.com/files/siga-visao.pdf>.

Todo documento no SIGA-DOC tem uma parte fixa (**Atributos** fixos do documento) e duas partes variáveis, uma chamada **Entrevista** e outra chamada **Texto** do documento (ou Documento). Na parte fixa, que serve a todos os documentos, caracteriza-se pela coleta de informações sobre a tramitação do documento (tipo, modelo, classificação documental e descrição), o subscritor (originador), destinatário e etc. A parte variável é onde serão colhidas (através de perguntas, chamada de entrevista) informações específicas sobre o tipo/modelo de documento e a geração (renderização) do mesmo, substituindo (interpolando) as variáveis colhidas na entrevista no texto padronizado do documento.

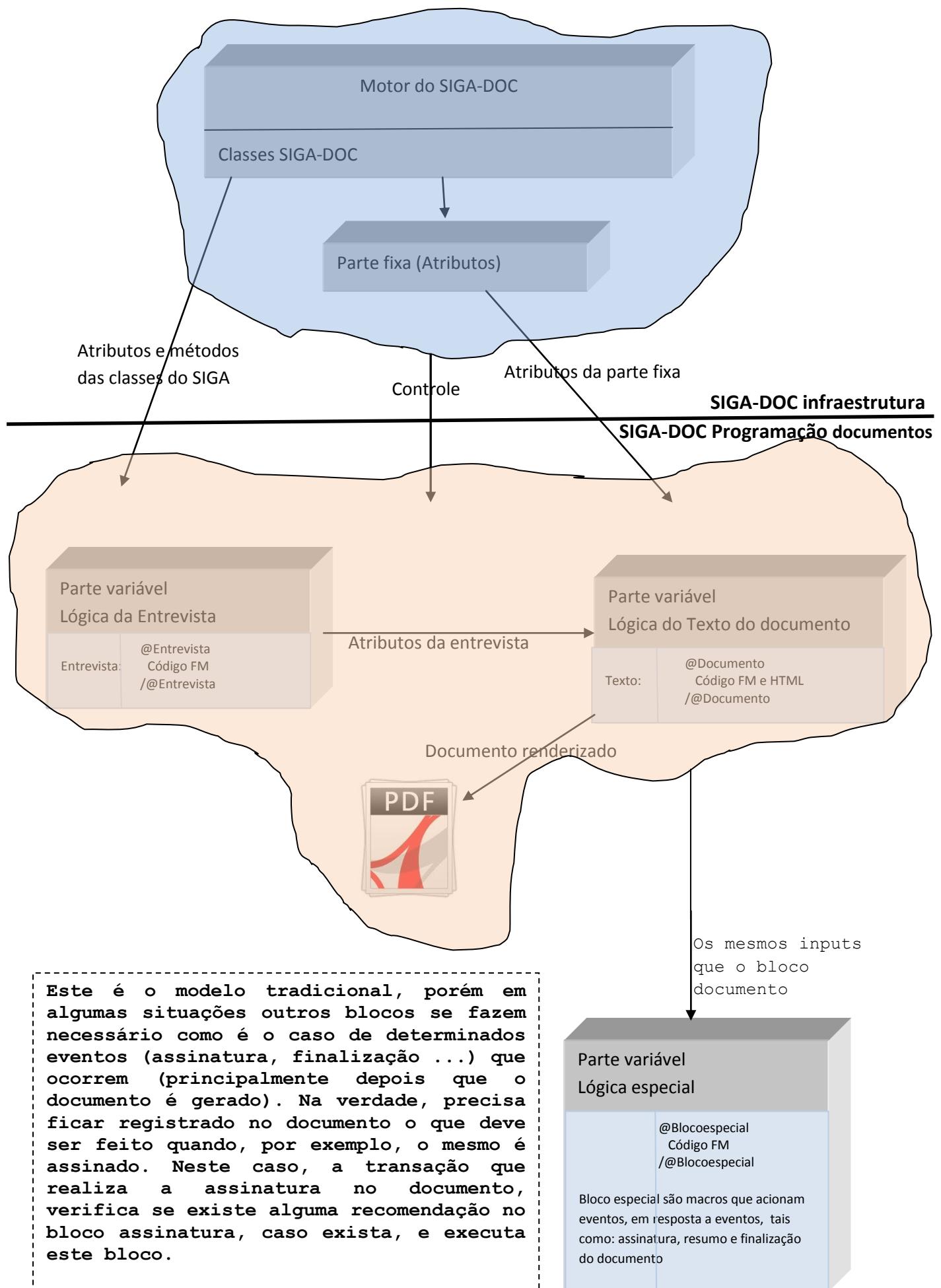
Parte fixa (Atributos)

Parte variável
Entrevista

Parte variável
Texto do documento

A programação de documentos no SIGA-DOC se restringe em realizar entrevistas (perguntas), para que se possa gerar um texto de documento customizado. Algumas entrevistas são muito simples e outras bem complexas, envolvendo acionamento de métodos em outras classes, como por exemplo, obter a lista de funcionários de um determinado setor. As entrevistas podem ser planas (flats) ou condicionais (aninhadas), quando a resposta a uma pergunta gera outras perguntas.

Também é utilizada técnica AJAX para evitar que toda tela seja totalmente renderizada ao se exibir um novo item de entrevista.

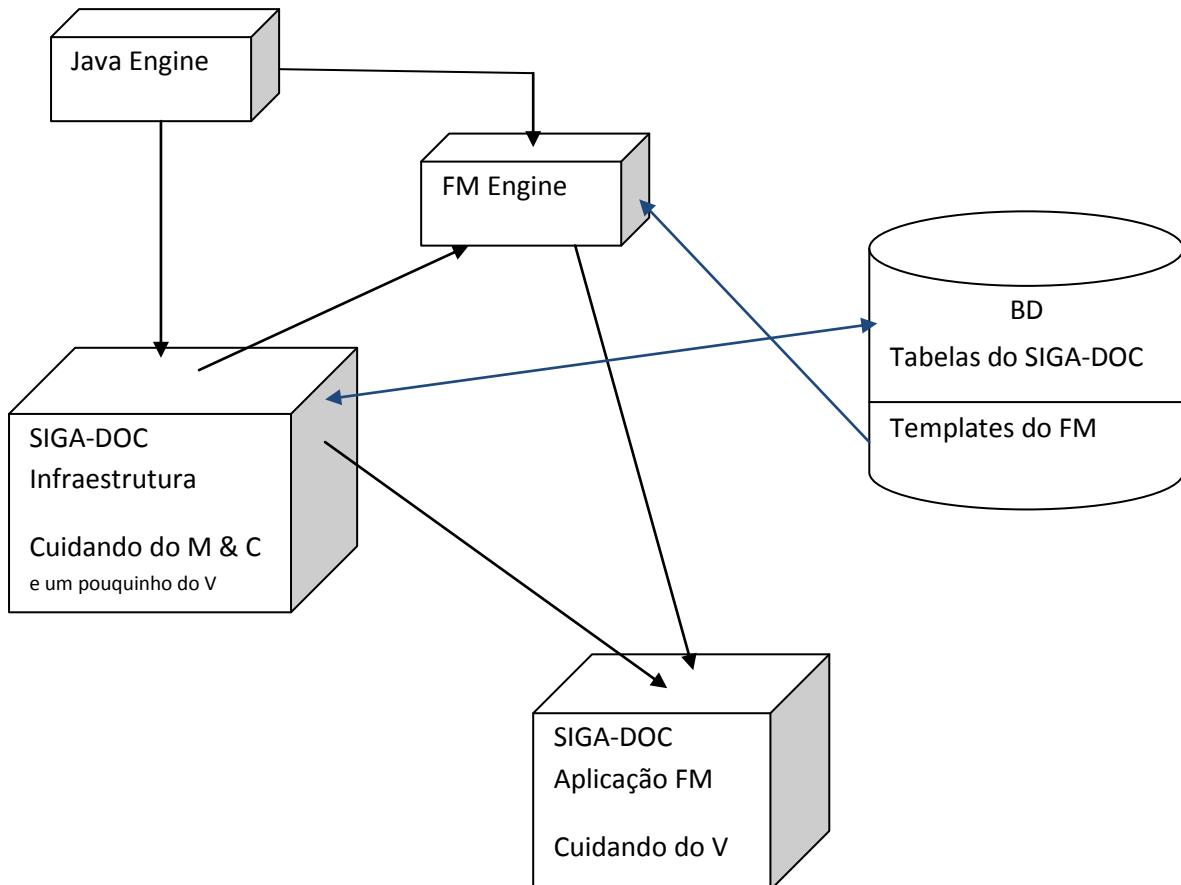


A programação pode ser feita em JSP (que está sendo descontinuada) ou em FreeMarker (FM). Para facilitar o desenvolvimento e a padronização, diversas macros foram preparadas, tanto em JSP quanto em FreeMarker. Este paper se resume em abordar as macros do FreeMarker.

No Design Pattern conhecido como MVC (Model, View, Controller), a programação JSP ou FM do SIGA corresponde a camada View.

A aplicação JSP é compilada e para ser testada precisa-se fazer um deploy de todo o sistema. A aplicação FM é interpretada e executada pelo FM Engine que roda sob o Java, e que aliado ao fato de poder ser armazenada num BD, permite que a execute sem necessidade de deploy, oferecendo um ambiente de desenvolvimento mais dinâmico e flexível. O JSP também necessita de extensões (taglibs, como o JSTL que fornece o core(c:) que inclui: c;IF, c:forEach e etc.) para sua biblioteca a fim de oferecer uma maior amplitude na sua linguagem. Já o FM possui um set de comandos bem completo.

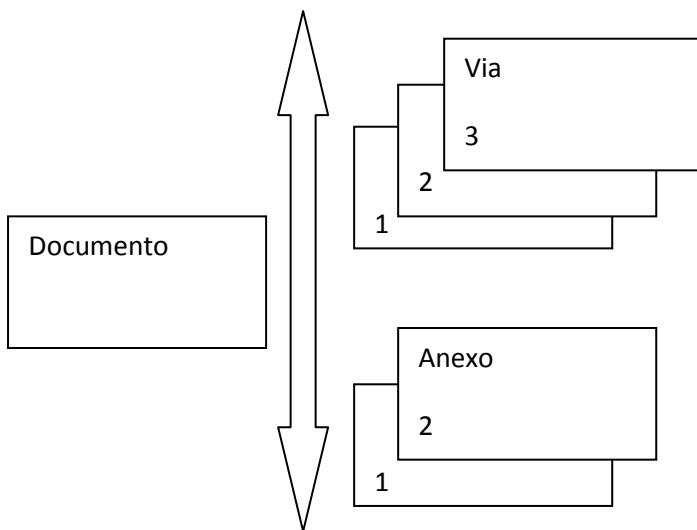
As aplicações FM podem estar no File System (FS), sob a extensão .ftl, ou armazenadas em um BD. No nosso ambiente, as aplicações são armazenadas no BD, o mesmo utilizado pelo SIGA-DOC.



[Veja em 12.2](#) as vantagens e desvantagens de se ter o FM residindo no FS (File System) ou no BD.

1.2 - O CONCEITO DE DOCUMENTO

Um documento pode ter várias vias e vários anexos. Existem ações que podem ser executadas na via, tais como: anexar arquivo, cancelar via e etc. ou para todos (móvel geral), tais como: assinar digitalmente, criar via, criar anexo e etc.



Exemplo: 1 documento 1 via com ações e log da via e do geral conforme figura abaixo:

RJ-MRU-2012/00004 - 1ª Via (Setor Competente) - Aguardando Andamento [RJ13635] - 16,3 kB

Ir para o documento:

Anexar Arquivo | Apensar | Arq. Corrente | Despachar/Transferir | Fazer Anotação | Juntar | Vincular | Visualizar Dossiê | Visualizar Impressão

Data	Evento	Cadastrante	Atendente	Descrição	Duração	
05/03/12	Criação	Lotação SESIE	Pessoa Renato	Lotação SESIE	Pessoa Renato	

Geral - 16,3 kB

Data	Evento	Cadastrante	Atendente	Descrição	Duração	
07/03/12	Assinatura	CDOC	Pessoa Anselmo	Lotação SESIE	Renato	Verificar
06/03/12	Assinatura	SESI	Ruben	SESI	Renato	Verificar
05/03/12	Assinatura	SESI	Renato	SESI	Renato	Verificar
	Assinatura	SECAD	Adriana	SESI	Renato	Verificar
	Inclusão de Co-signalário de Ordem	SESI	Renato	SESI	Renato	
	Inclusão de Co-signalário de Ordem	SESI	Renato	SESI	Renato	
	Inclusão de Co-signalário de Ordem	SESI	Renato	SESI	Renato	

Móbil Geral

Data	Evento	Cadastrante	Atendente	Descrição	Duração	
07/03/12	Assinatura	CDOC	Pessoa Anselmo	Lotação SESIE	Renato	Verificar
06/03/12	Assinatura	SESI	Ruben	SESI	Renato	Verificar
05/03/12	Assinatura	SESI	Renato	SESI	Renato	Verificar
	Inclusão de Co-signalário de Ordem	SESI	Adriana	SESI	Renato	Verificar
	Inclusão de Co-signalário de Ordem	SESI	Renato	SESI	Renato	
	Inclusão de Co-signalário de Ordem	SESI	Renato	SESI	Renato	

Documento Interno Produzido:RJ-MRU-2012/00004
Assinar Digitalmente | Criar Anexo | Criar Via | Definir Perfil | Duplicar | Exibir Informações Completas | Exibir Todas as Vias | Redefinir Nível de Acesso

1.2.1 - A linguagem documental

O SIGA-DOC é um sistema que fundiu dois outros sistemas mais antigos (de controle de expedientes e processos administrativos), trazendo inúmeras implementações importantes, sendo que a mais importante, sem dúvidas, é a capacidade de criar e anexar documentos eletrônicos.

Expediente - qualquer documento produzido ou cadastrado no Siga-Doc.

Processo Administrativo - conjunto de documentos (incluindo expedientes) que se ordenam cronologicamente, para materializar os atos **de um ou mais** procedimentos de um processo. Exemplo: PA-EOF-2012/0001: processo de contratação de serviço de digitalização de documentos. O termo processo neste caso equivale ao conceito de autos.

Subprocesso Administrativo - conjunto de documentos (incluindo expedientes) que se ordenam cronologicamente para materializar os atos **de um** determinado procedimento de um processo. Exemplo: PA-EOF-2012/0001.01: processo de contratação de serviço de digitalização de documentos - Destinado ao pagamento de notas fiscais. O termo subprocesso também equivale ao conceito de autos.

Observações importantes:

Pode-se dizer que processos e seus subprocessos são **independentes** do ponto de vista **formal**, da forma. Ou seja, são conjuntos de documentos que tramitam paralelamente, independentes uns dos outros.

Pode-se dizer também que processos e seus subprocessos são **dependentes** do ponto de vista **material**, da matéria de que tratam. Ou seja, tratam do mesmo processo (sentido jurídico), que no exemplo acima é a "contratação de serviço de digitalização de documentos". Porém, enquanto o subprocesso trata apenas do procedimento de pagamento das notas fiscais do processo de contratação de serviço de digitalização de documentos, o processo trata dos demais procedimentos, como licitação, homologação, assinatura contratual, termos aditivos e alterações contratuais diversas.

Vários são os motivos para abertura de subprocessos. O objetivo de separar, no exemplo citado, o procedimento de pagamento das notas fiscais dos demais procedimentos é evitar atrasos no pagamento do serviços prestados. Ou seja, graças ao subprocesso, o pagamento de uma nota fiscal pode ocorrer paralelamente, sem atrasos, à análise e produção de um termo aditivo no mesmo contrato.

Volume - parte de um processo ou subprocesso administrativo. Visando melhor manuseio, físico ou eletrônico, de um grande conjunto de documentos que formam um processo ou subprocesso, pode-se dividi-los em duas ou mais partes. Um processo ou subprocesso físico costuma ser dividido em volumes a cada 300 páginas; já um processo eletrônico, pode ser dividido em volumes a cada 2MB ou 3MB. Dessa forma, não é necessário tramitar a todo momento todo conjunto de documentos.

Encerrar volume - significa determinar o fim de uma parte do processo ou subprocesso e o início da parte seguinte.

Via - cópia clone (idêntica, e de mesmo valor do original) de um expediente; tem a mesma numeração do documento original. Tramitam de forma paralela e independente um do outro.

Cancelar via - cancela um clone de um expediente.

Apensar - ato de fazer dois ou mais documentos independentes tramitarem obrigatoriamente juntos. Ou seja, se os documentos A e B estão apensados um ao outro, ao se transferir o documento A para o setor X, o documento B será igualmente transferido para o setor X, automaticamente, junto com o documento A. Grosso modo, é o mesmo que amarrar fisicamente um documento ao outro, por meio de um barbante. O termo **Apensar** também é usado para "amarrar" entre si diversos volumes de um mesmo processo ou subprocesso. Inclusive, o Siga-Doc, a cada abertura e encerramento de volume, faz a apensação automática entre eles. Para "soltar" documentos apensados entre si é necessário fazer a **desapensação**.

Vincular - Equivale à expressão "ver também". Quando se faz, por exemplo, a vinculação do documento A no documento B, o sistema irá mostrar nos andamentos de ambos os processos, o movimento de "Vinculação de Ordem", com um link para acessar o(s) documento(s) vinculado(s), por meio da expressão "Ver também:". Documentos vinculados entre si continuam tramitando independentemente um do outro.

Juntar - ato de incluir um documento produzido ou cadastrado no Siga-Doc dentro de outro documento ou dentro de um processo ou de um subprocesso. Quando juntamos o documento A ao documento B, o documento A assume a classificação documental do documento B. Se, por exemplo, juntamos um memorando cuja classificação documental permite eliminação em 2 anos a um processo administrativo de guarda permanente, o

referido memorando passa a ser de guarda permanente junto com o processo. Para separar documento juntado a outro, é preciso fazer o **desentranhamento**.

Ao contrário da desapensação, que é um simples ato de "soltar" documentos e volumes apensados entre si, o desentranhamento irá gerar automaticamente uma certidão eletrônica informando que um documento foi retirado. Exemplo: foi juntado ao processo administrativo A, às fls. 15 a 20, o documento B, posteriormente desentranhado; logo, o sistema irá automaticamente substituir o documento B, de fls. 15 a 20, por uma certidão de desentranhamento. A numeração das páginas do processo administrativo não irá mudar, pois a certidão irá assumir a numeração das páginas do documento desentranhado.

Criar anexo - ato de criar um documento-filho, dependente de outro documento. Um documento criado por meio da opção Criar Anexo nasce automaticamente com a mesma classificação documental, e fica automático e permanentemente vinculado ao documento principal, por meio da expressão "Documento Filho:".

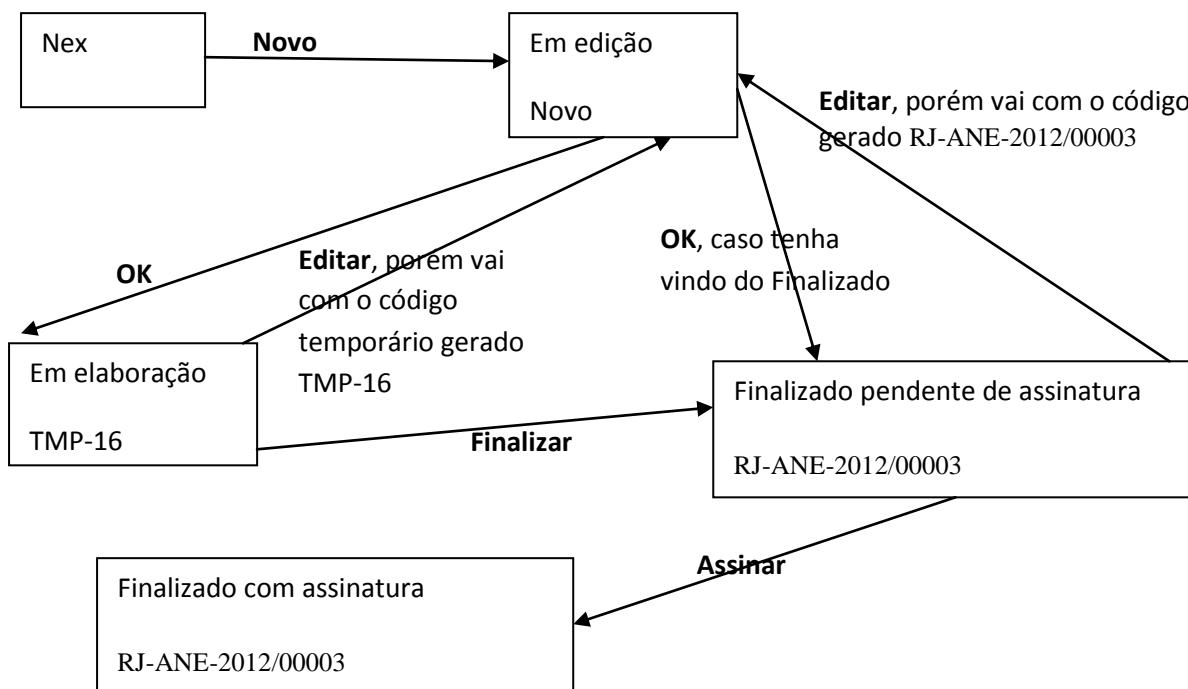
O anexo está para o expediente, assim como o subprocesso está para o processo. Ou seja, há **dependência material** entre eles. Um anexo e um subprocesso não fazem sentido sozinhos, sem seus respectivos expediente e processo. Porém, caso necessário, podem tramitar independentes uns dos outros, pois podem ser juntados/apensados ou desentranhados/desapensados.

Anexar arquivo - ato de juntar um documento em .PDF dentro de um documento criado ou cadastrado no Siga-Doc, ou dentro de um processo ou de um subprocesso. Quando anexamos um arquivo PDF qualquer ao documento A, esse arquivo PDF passa a tramitar junto com o documento A, como se fossem um único documento.

Antes de entrarmos na documentação das macros e métodos é importante analisar os tópicos: "transição de estado do documento" e "campos da entrevista que provocam a leitura (releitura) da tela", visto que isto influenciará no comportamento da aplicação FM e consequentemente no resultado nos métodos que são invocados.

1.3 - Transição de estado do documento

Em negrito, as ações do SIGA-DOC que provocam a transição de estado. Na parte inferior de cada caixa tem o código do documento que é gerado automaticamente pelo sistema.



- Os códigos são apenas para exemplificar.
- O OK, salva o documento. Neste momento, quando saímos da edição, uma crítica é efetuada no sentido que somente deixará seguir adiante se todos os campos obrigatórios forem preenchidos.
- Alguns métodos, que serão vistos adiante, só funcionarão em determinados estados. Por exemplo, a data de fechamento do documento só estará disponível após o mesmo ser finalizado (mesmo pendente de assinatura).
[#assign qqcoisa = doc.getDtFechamento()?string/
 \${qqcoisa}]
- Os métodos que trabalham com o móbil geral só funcionarão quando o documento for finalizado com assinatura, pois é neste momento que as linhas do móbil geral são persistidas.
[#assign qqcoisa = doc.getDtUltimaRemessaParaPublicacao()?string/
 \${qqcoisa}]

Observação: quando saímos de um ambiente (**Em edição**) para outro (**Em elaboração**) e voltamos para o anterior (**Em edição**) não perdemos o conteúdo da tela, para mais detalhes ver o capítulo “Lições Aprendidas, Burlando o Sateless ...”

1.4 – Comportamento das aplicações WEB

Quando recebemos a tela de um documento e começamos a preenchê-la notamos um comportamento diferente de uma aplicação cliente x servidor ou apenas cliente. O comportamento de uma aplicação WEB relembraria das aplicações que rodavam em mainframes e minis, que enviavam as famosas telas de texto para terminais burros (verdes), porém são distintos quanto à aplicação servidora, pois no mundo WEB, geralmente as aplicações são STATELESS (rodam, processam e encerram) e nos mainframes as aplicações são STATEFULL (rodam, enviam a tela, aguardam o usuário ...). A seguir, descreveremos alguns modelos de processamento para que possamos entender a filosofia (ou fisiologia) de uma aplicação WEB. Para um detalhamento de como o FM entra nesta equação da WEB favor consultar “Lições Aprendidas, na primeira seção”

Eu já passei por alguns modelos de processamento e senti muita dificuldade a me acostumar com o modelo WEB tradicional, que a princípio, pode parecer um retrocesso, como veremos a seguir. A grande questão é que o processamento WEB é STATELESS, enquanto os outros aos quais havia trabalhado eram essencialmente STATEFULL. Além disso, a tela (HTML) é criado no servidor e enviada ao cliente, gerando o que chamo de dissincronia entre o cliente e servidor, gerando uma certa confusão. No capítulo “Lições Aprendidas, na primeira seção” menciono os principais problemas com os quais me deparei com o modelo WEB.

Evolução dos modelos:

Nomenclatura:

SA – Servidor de aplicação, onde roda uma parte, ou toda, do código. Não vamos entrar no tecnicismo do conceito, e sim, definir o local onde roda a aplicação.

SBD – Servidor de Banco de Dados. Local onde roda e são armazenados os dados.

→ Conectado (STATEFULL)

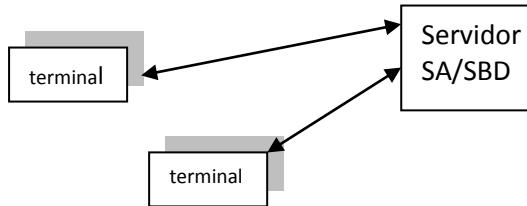
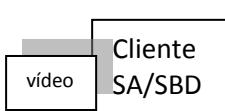
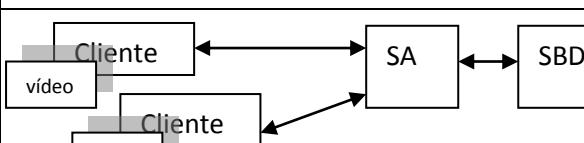
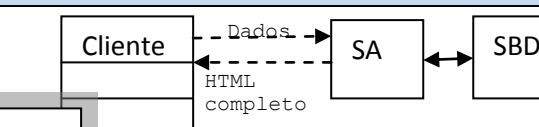
Statefull, ou seja, a aplicação está sempre esperando um comando do usuário para encerrar. Consequentemente, todas as variáveis criadas durante a execução da aplicação ficam disponíveis, não são perdidas. O SO está ciente da última instrução executada e da próxima

→ Desconectado (STATELESS)

Stateless, ou seja, a aplicação é chamada, executada e terminada. Ela não guarda o estado das variáveis, que são perdidas com o término da aplicação (de visualização). Já se fala em modelos statefull para web, porém não vem ao caso aqui.

STATELESS - Problemas a vista para os programadores!!!

Se a aplicação no servidor termina, então as minhas variáveis vão para o espaço. Como mantê-las entre várias sessões? Ver no capítulo “Lições Aprendidas” a solução deste problema.

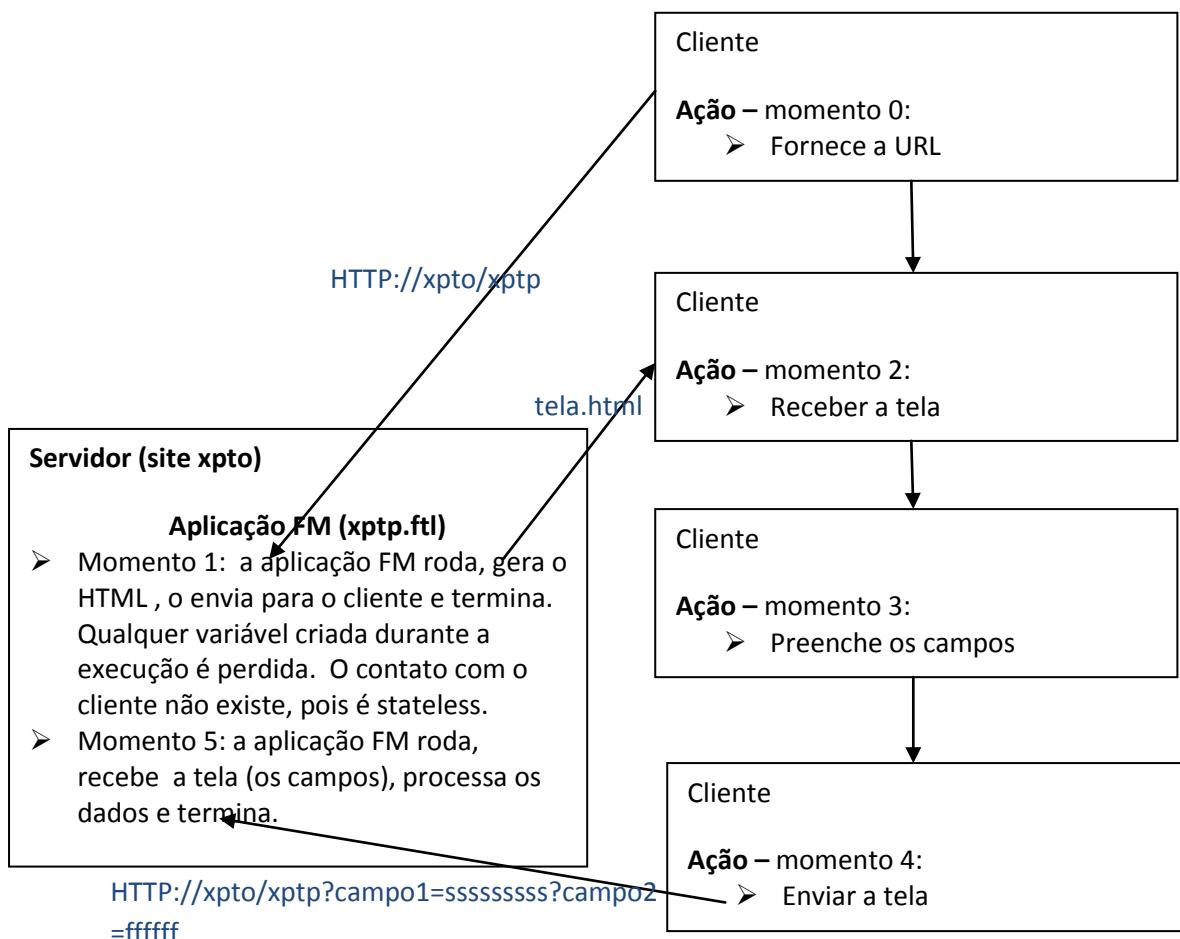
Modelo	Diagrama	Descrição	Camadas	Inteligência Cliente	Método
Servidor Puro O servidor pode ser uma Mainframe ou mini		Muitas transações de banco ainda utilizam este modelo, sendo que o PC simplesmente emula um terminal. Na justiça, as aplicações MUMPS utilizam este modelo.	1	Nenhuma, terminal burro. Enquanto o usuário estava digitando ou pensando (think time) a aplicação ficava em wait e não tinha a mínima noção do que estava acontecendo, até que ele pressionasse uma tecla de função que transmitia os campos da tela. A interação com o usuário era nula.	STATEFULL
Cliente Puro O cliente é um PC		Pequenas aplicações monousuárias.	1	Como o cliente é o próprio desktop ele fornece todo tipo de interação, já que o teclado está ligado a CPU. Aqui já podíamos ter algo como digitar um CPF e ele ser formatado em tempo de digitação. Uma evolução em termos de cliente. Começamos com interfaces tipo texto e depois passamos para gráficas.	STATEFULL
Cliente x Servidor O cliente é um PC e o servidor pode ser um mainframe, mini ou PC		Este é um modelo tradicional de rede, como o utilizado pelo sistema APOLO.	2	Idem a anterior	STATEFULL
Cliente x Servidor x Servidor (3 camadas) O cliente é um PC e os servidores podem ser Mainframes, minis ou PCs		Este modelo não pegou na prática e foi logo substituído pelo modelo WEB	3	Idem a anterior	STATEFULL
WEB					
WEB Tradicional Servidor (SA) x Servidor (SBD), e também, Servidor (SA) p/ o Cliente		No SA roda o FM (ou JSP) que gera o HTML. Também rodam servlets de controle e acesso a dados.	3, mas na verdade são 2	No modelo tradicional, sem JavaScript, a aplicação WEB se parece com o terminal burro da época dos mainframes, visto que não há uma interação síquera com o usuário. Embora seja um	STATELESS

				PC que esteja rodando o browser, não podemos dizer que a aplicação está conectada diretamente a CPU para obter os benefícios da interação total, visto que, na verdade, a aplicação está em outro servidor, possivelmente a quilômetros de distância do seu PC.	
Servidor (SA) x Servidor (SBD), e, Servidor (SA) p/ o cliente, e também, Cliente (JS)	<pre> graph LR Client[Cliente JS] -- Dados --> SA[SA] SA <-- HTML completo --> Client SA <--> SBD[SBD] </pre>	No SA roda o FM (ou JSP) que gera o HTML. Também rodam servletras de controle e acesso a dados.	3, mas na verdade são duas	Com JavaScript, a aplicação que roda no browser pode interagir com o usuário, permitindo que por exemplo: um CPF seja formatado em tempo de digitação ou ícones que se movam e etc.	STATELESS
Servidor (SA) x Servidor (SBD), e, Servidor (SA) p/ o cliente, e, Cliente (JS), e também, Cliente (AJAX) x Servidor	<pre> graph LR Client[Cliente JS] -- Dados --> SA[SA] SA <-- HTML completo --> Client SA <--> SBD[SBD] Client -.-> Assíncrono ou assíncrono SA </pre>	<p>Este é um dos modelos mais complexos. É o modelo que está disponibilizado no SIGA-DOC.</p> <p>No SA roda o FM (ou JSP) que gera o HTML. Também rodam servletras de controle e acesso a dados.</p>	3	O AJAX trouxe a possibilidade do cliente se conectar com o servidor por demanda, realizando requisições e obtendo respostas, reposta estas que podem conter só dados ou pedaços da tela (HTML) que são inseridos in loco no browser sem renderização da tela toda. Observação: o fato de renderizar ou não, não é uma propriedade do AJAX e sim do JS.	STATELESS

TENDÊNCIAS (NO MEU PONTO DE VISTA)

Cliente X Servidor (SA) x Servidor (SBD)	<pre> graph LR Client[Cliente JS / jQUERY] -- Carga Inicial --> SA[SA] Client -- Dados --> SA SA <--> SBD[SBD] Client -.-> Assíncrono ou assíncrono SA </pre>	<p>Com a evolução das bibliotecas JS, como JQUERY/JSON e seus companheiros JqueryUI, sliders, fancybox, Jquery plugins e outros, não haverá mais necessidade de uma aplicação intermediária (FM, JSP, ...) gerando o HTML para o cliente. O próprio cliente carregará a sua interface e fará acesso ao servidor, sob demanda, via AJAX.</p>	3	Voltaremos a ter controle total sobre a aplicação cliente, pois mais nada de HTML será gerado no servidor. Isto facilitará novamente o desenvolvimento de aplicações, sem o incômodo da dissincronia que tenho falado. Uma interface puramente gráfica orientada a eventos.	STATELESS
--	---	---	---	---	-----------

Esquema típico de uma aplicação WEB Tradicional com o botão <Enviar> e método GET (onde os campos da tela são enviados na requisição separados por ?)



Os métodos para enviar formulários via HTTP são o GET e POST, e podem ser analisados no link <http://www.comocriarsites.com/html/como-funciona-os-metodos-get-e-post-diferencias.>

Aqui: tela, HTML ou documento são sinônimos.

No momento 3, a aplicação FM não possui mais nenhum contato com o cliente. A única forma de interagir com o cliente neste momento seria através de javascript, pois ele vai incorporado no HTML e roda no cliente.

Ora, se por exemplo, no FM utilizamos o código abaixo (que invoca um método que será visto posteriormente) após o usuário ter digitado a data do documento na parte fixa da entrevista, nada será exibido, pois a tela não foi enviada ao servidor. Aliás, no momento 1 em que o HTML foi gerado, o método doc.getDtDocDDMMYY() deve ter retornado "".

```
[#assign qqcoisa = doc.getDtDocDDMMYY()?string/]
${qqcoisa}
```

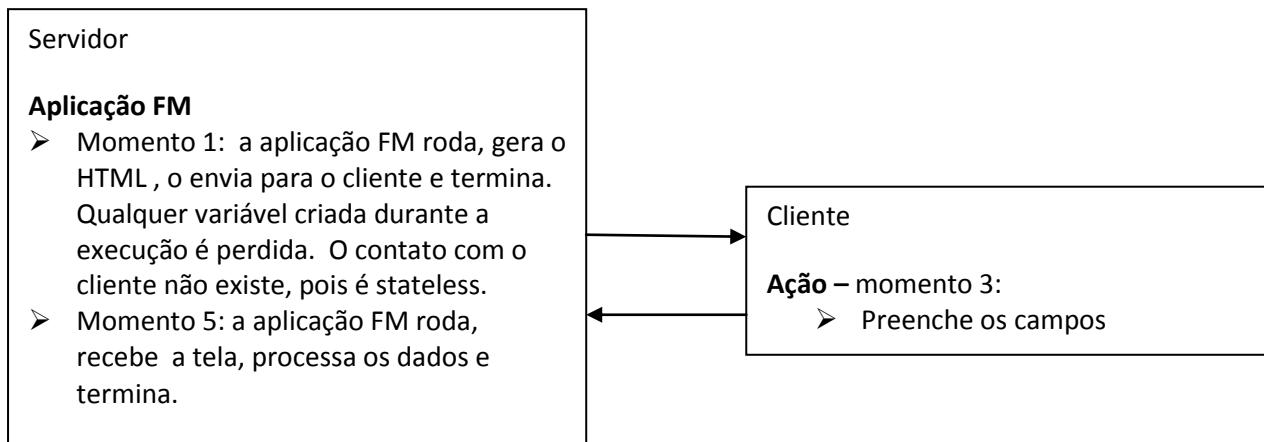
Porém ao usuário selecionar a matrícula como destinatário a data documento será exibida. Por quê? Ver o item seguinte.

1.5 - Campos da entrevista que provocam a leitura (releitura) da tela

Campos que provocam a leitura (releitura) da tela:

Na parte fixa da entrevista: Origem, Destinatário (transição entre um tipo e outro) ou o fornecimento de uma matrícula ou UO, o tipo e modelo do documento e qualquer outro campo na parte variável da entrevista que tenha sido definido com reler=sim.

Desta forma, quando o usuário está digitando no SIGA-DOC, os campos listados acima provocam a leitura da tela conforme o diagrama abaixo.



1.6 – Sintaxe do FM

Observação: Nós utilizamos a **sintaxe bracket (colchete)** [...] em substituição ao < ... > no FM aqui na SJRJ, porque isto evita confusão com os comandos HTML, que também utilizam < ... >.

Descrição	FM	JSP	HTML
Estrutura básica	<# ... /> OU [# ... /]	<% ... %>	< ... />
Comentário	<!-- ... --> OU [#-- ... --]	<% ... %>	<!-- ... -->
Interpolação de variável	`\${variavel}`	<%=variavel%>	

O SIGA-DOC fornece uma série de macros que atuam, principalmente, na parte variável da entrevista. Sem elas, teríamos que a todo o momento escrever código HTML para formatar campos na tela, implementar AJAX, criar Drop-Down List, Radio Button e etc. Por um lado, as macros engessam, porém, por outro, agilizam, além de fornecer uma boa padronização.

2.1 – Introdução

A tabela, a ser apresentada, lista todas as funções e macros existentes no ambiente do SIGA-DOC. Esta é a posição em 26/04/2012, visto que a tendência é só aumentar a quantidade de macros / functions.

Só pra lembrar, a diferença entre macro e function. A macro recebe (ou não) parâmetros e não retorna nada, já a function recebe (ou não) parâmetros e retorna sempre 1 parâmetro.

A macro é definida com:

```
[#macro nome_macro param1 param2 ...]  
    Escopo da macro. Comandos FM  
[/#macro]
```

E chamada com:

```
[@nome_macro param1 param2 ... /]
```

A function é definida com:

```
[#function nome_function param1 param2 ...]  
    Escopo da function. Comandos FM  
[/#function]
```

E chamada com:

```
[#assign varqualquer = nome_function(param1, param2, ...)/]
```

2.2 – Classificação das macros

As macros e functions foram classificadas da seguinte forma:

- **uso geral / utilitários:** são de propósito geral e praticamente constituída de aplicações utilitárias ou suporte / ajuda;
- **aplicação:** são macros / aplicações que fornecem funcionalidades padrão. A ideia, embora o FM não seja OO, é que estas macros formem as classes pai. Ao chamarmos uma destas macros numa aplicação(classe filha) podemos utilizar as funcionalidades padrão e adicionar funcionalidades específicas no escopo da aplicação;
- **associada ao bloco entrevista:** são macros que nos ajudam a realizar a entrevista com o usuário, ou seja, coletar informações. A única forma de realizar entrevista é através de formulários que possuem campos os quais sofrerão input por parte do usuário;
- **associada ao bloco documento:** são macros que nos ajudam a gerar o documento HTML, que por sua vez será convertido em PDF;
- **associada ao bloco especial:** são macros que acionam eventos, em resposta a eventos, tais como: assinatura, resumo e finalização do documento;
- **associada a funções avançadas do SIGA-DOC:** são macros específicas que praticamente não serão utilizadas pelo programador de documentos, e sim, pelo programador da infraestrutura do SIGA-DOC.

2.3 – Componentes do bloco Entrevista e do bloco Documento

É importante lembrar que a **entrevista (bloco entrevista)** é um formulário contendo campos (fields). Cada campo possui uma formatação específica dependendo do seu tipo: string, data, texto, memo e etc.

Parte Fixa
Campo1: <input type="text"/>
Campo2: <input type="text"/>
Entrevista - Parte Variável
Campo3: <input type="text"/> <small><-- macro @texto</small>
Campo4: <input type="text"/> <small><--macro @data</small>

As macros do bloco da entrevista, na verdade, geram tags HTML Input:

```
<form>
  Campo3: <input type="text" name="campo3" /><br />
  Campo4: <input type="text" name="campo4" />
</form>
```

É importante observar, por exemplo, que não existe em HTML um input type="date", porém existe uma macro @data que só permite inputs de data. É que a macro @data chama uma rotina em JavaScript que critica este tipo de input conforme o padrão de datas e limita o tamanho da variável "text"em 10 caracteres (dd/mm/aaaa).

Já o **documento (bloco documento)** possui a seguinte estrutura:

Página 1	Página 2
Cabeçalho: Só texto Poder Judiciário / Justiça Federal	
Primeiro Cabeçalho: aqui é inserido o brasão com o texto Poder Judiciário / Justiça Federal	
Corpo - Parte variável	
Abertura	
Texto: Aqui é onde serão interpoladas as variáveis da entrevista juntamente com o texto do documento.	
Blá blá blá blá \${campo3} blá bla'bla'blá blá blá \${campo4}	
Fecho	
Assinatura:	
Primeiro Rodapé: aqui é inserida a numeração e / ou classificação documental	

O documento pode possuir várias páginas, e por isso, possuir cabeçalhos e rodapés diferenciados para a primeira página e páginas subsequentes.

2.4 – Categorização das macros

As macros foram categorizadas da seguinte forma:

Nível	Descrição
1	Primitiva. Só possui o nested como comando FM, ou seja, o seu conteúdo será passado durante a invocação da macro.
2	Secundária. Possui comandos FM, porém não possui vida própria, só existe para atender outra macro.
3	O nível 3 e 4 não possuem diferença, e esta distinção só existe devido às macros associadas ao documento possuírem 4 níveis de hierarquia. Podem invocar as macros de nível 1 e 2 e geralmente são invocadas pelas aplicações Fm, ou macro de nível 4 ou 5.
4	O nível 3 e 4 não possuem diferença, e esta distinção só existe devido às macros associadas ao documento possuírem 4 níveis de hierarquia. Podem invocar as macros de nível 1, 2 e 3 e geralmente são invocadas pelas aplicações Fm ou macros de nível 5.
5	Aplicação. É uma aplicação completa, com bloco entrevista e bloco documento. Podem invocar as macros de nível 1, 2 e 3 e só pode ser invocada por uma aplicação FM.

2.5 – Lista de Macros e Functions (Posição em 26/04/2012)

Como podemos observar na tabela abaixo, cada macro pode possuir inúmeros parâmetros. Inicialmente não fornecerei a descrição de cada parâmetro nesta tabela, até porque não conheço todos. A descrição detalhada de algumas macros (principalmente os utilitários e as associadas à entrevista) e de alguns parâmetros será apresentada de forma contextual no capítulo “Macros em Ação – Programação por Exemplos”.

Macro / Function	Parâmetros	Descrição
CLASSIFICAÇÃO		
USO GERAL / UTILITÁRIOS		
function par	parameter	Não identifiquei o seu uso
function formatarCPF	fmtCPF_param	É uma function de nível 3 ou 4. Formata um CPF
function validarCPF	vldCPF_cpfrecebido	É uma function de nível 3 ou 4. Valida um CPF
function fmtvldCPF	fmtvldCPF_param	É uma function de nível 3 ou 4. Formata e Valida um CPF <i>Functions Chamadas:</i> formatarCPF validarCPF
macro dump		É macro de nível 3 (ou 4). Dump das variáveis da aplicação em tempo de execução. <i>Macros Chamadas:</i> vertipo
macro vertipo		Macro de nível 2 (secundária) chamada pela macro Dump
macro br		É uma macro de nível 3 (ou 4), simplíssima, pois só possui a tag HTML . Causa quebra de linha.
macro separador		É uma macro de nível 3 ou 4, simplíssima, pois só possui a tag HTML <hr/>. Causa uma linha separadora.
APLICAÇÃO		
macro oficio	_texto="" _tipo_autoridade="" _genero="" _vocativo="" _enderecamento_dest="" _nome_dest=""	Macro de nível 5. Embora pareça uma macro associada ao documento, esta macro é uma aplicação genérica para ofícios. Ela possui os blocos entrevista e documento.

ASSOCIADA A ENTREVISTA		
macro entrevista		Macro primitiva de nível 1. Responsável pela geração da entrevista. Esta macro é obrigatória em todas aplicações FM.
macro mensagem	Texto titulo="" vermelho=false	Macro de nível 3 (ou 4). Responsável por exibir uma mensagem de texto. Muito utilizada em críticas de informações e para instruir o usuário sobre o funcionamento da aplicação.
macro caixaSelecao	Titulo Var tipo="" idInicial="" siglaInicial="" descricaoInicial="" modulo="" desativar=false buscar=true ocultarDescricao=false reler=false idAjax="" default="" alerta=false obrigatorio=false relertab="" paramList="" grande=false	Macro secundária, nível 2, chamada pela macro selecionavel. Responsável por exibir uma cortina (dropdown list) de opções a qual o usuário fará uma seleção.
macro selecionavel	titulo var tipo reler=false idAjax="" default="" alerta=false obrigatorio=false relertab="" paramList=""	É uma macro secundária, nível 2, chamada pelas macros: pessoa, função e lotacao. <i>Macros Chamadas:</i> caixaSelecao
macro pessoa	titulo	Macro de nível 3 (ou 4). Responsável por recuperar o nome

	<pre>var reler=false relertab="" buscarFechadas=false idAjax="" default="" obrigatorio=false paramList=""</pre>	<p>do funcionário passando algum tipo de identificador, tal como: matrícula, sigla ou partes do nome.</p> <p><i>Macros Chamadas:</i> selecionavel</p>
macro funcao	<pre>titulo var reler=false relertab="" buscarFechadas=false idAjax="" default="" obrigatorio=false paramList=""</pre>	<p>Macro de nível 3 (ou 4). Responsável por recuperar o nome da função (auxiliar, chefe, supervisor ...) passando algum tipo de identificador, tal como: parte do nome da função.</p> <p><i>Macros Chamadas:</i> selecionavel</p>
macro lotacao	<pre>titulo var reler=false relertab="" buscarFechadas=false idAjax="" default="" obrigatorio=false paramList=""</pre>	<p>Macro de nível 3 (ou 4). Responsável por recuperar o nome da lotação (UOs Administrativas, varas...) passando algum tipo de identificador, tal como: sigla ou parte do nome da lotação.</p> <p><i>Macros Chamadas:</i> selecionavel</p>
macro data	<pre>titulo var reler=false idAjax="" default="" alerta=false obrigatorio=false</pre>	<p>Macro de nível 3 (ou 4). Responsável por exibir um campo que só aceita datas como input.</p>
macro grupo	<pre>titulo="" largura=0 depende="" esconder=false</pre>	<p>Macro nível 3 (ou 4). É a implementação da tag HTML <div> e atributo id=.</p> <p><i>Macros Chamadas:</i> Div - nível 2</p>

macro div	<pre>id="" depende="" suprimirIndependente=false</pre>	<p>É uma macro secundária, nível 2, chamada pela macro grupo.</p> <p>Ela cria uma tag <div> e um atributo HTML, <i>id</i>=, com o idAjax passado a ela pela macro @grupo. O <i>id</i>= é utilizado pelo AJAX.</p> <p>The <div> tag defines a division or a section in an HTML document. The <div> tag is used to group block-elements to format them with styles. The <div> element is very often used together with CSS, to layout a web page.</p> <p>Note: Browsers always place a line break before and after the <div> element.</p> <p>Ex: A section in a document that will be displayed in green:</p> <pre><div style="color:#00FF00"> <h3>This is a header</h3> <p>This is a paragraph.</p> </div></pre> <p>The id attribute is most used to point to a style in a style sheet, and by JavaScript (via the HTML DOM) to manipulate the element with the specific id.</p> <pre><html> <head> <script type="text/javascript"> function displayResult() { document.getElementById("myHeader").innerHTML="Have a nice day!"; } </script> </head> <body> <h1 id="myHeader">Hello World!</h1> <button onclick="displayResult()">Change text</button> </body> </html></pre>
-----------	--	---

macro texto	<pre>var titulo="" largura="" maxcaracteres="" idAjax="" reler="" relertab="" obrigatorio="nao" default=""</pre>	Macro de nível 3 (ou 4). Responsável por exibir um campo que só aceita texto como input.
macro oculto	<pre>var valor="" default=""</pre>	Macro de nível 3 (ou 4). Responsável por criar campos ocultos no formulário. Muito utilizado para fornecer persistência de variáveis de trabalho (ver capítulo "Lições Aprendidas").
macro checkbox	<pre>var titulo="" default="Nao" idAjax="" reler=false onclique="" obrigatorio=false</pre>	Macro de nível 3 (ou 4). Responsável por exibir um campo tipo checkbox.
macro radio	<pre>titulo var reler=false idAjax="" default="Não" valor="Sim" onclique=""</pre>	Macro de nível 3 (ou 4). Responsável por exibir um campo tipo radio Button.
macro identificacao	<pre>pessoa funcao="" nivelHierarquicoMaximoDaLot acao="" obs="" negrito="nao"</pre>	
macro selecao	<pre>var titulo opcoes</pre>	Macro de nível 3 (ou 4). Responsável por exibir uma cortina (drop-down list) de opções a qual o usuário fará uma seleção.

	<code>reler=false idAjax="" onclick=""</code>	
macro memo	<code>var titulo colunas linhas reler=false obrigatorio=false default=""</code>	Macro de nível 3 (ou 4). Responsável por exibir um campo texto com "N" linhas e "M" colunas. Muito usao para comentários, observações e etc.
ASSOCIADA A GERAÇÃO DO DOCUMENTO		
macro documento	<code>formato="A4" orientacao="retrato" margemEsquerda="3cm" margemDireita="2cm" margemSuperior="1cm" margemInferior="2cm"</code>	Responsável pela geração do documento. É uma macro primitiva, obrigatória nas aplicações, de nível 1, possuindo somente o nested.
<p>Macros de nível 1 (Primitivas, só possuem o nested definido) para cabeçalho e rodapé</p> <p>ACEITA O QUE FOR PASSADO AO NESTED, OU SEJA, UM TRECHO DE CÓDIGO FM QUE PODE INCLUIR: HTML E INTERPOLAÇÃO DE VARIÁVEIS PARA FORMAR O CABEÇALHO E/OU RODAPÉ.</p>		
macro primeiroCabecalho		Quando um documento possui muitas páginas, este é o cabeçalho da primeira página, mais completo (com mais informações) do que o cabeçalho das páginas subsequentes
macro cabecalho		Quando um documento possui muitas páginas, este é o cabeçalho das páginas subsequentes
macro primeiroRodape		Quando um documento possui muitas páginas, este é o rodapé da primeira página, mais completo (com mais informações) do que o rodapé das páginas subsequentes
macro rodape		Quando um documento possui muitas páginas, este é o rodapé das páginas subsequentes
<p>Macros de nível 1 (Primitivas, só possuem o nested) definido para marcação do Boletim Interno Eletrônico</p> <p>EXISTE UMA OUTRA APPLICAÇÃO QUE PEGARÁ ESTAS MARCAÇÕES NOS DOCUMENTOS PARA A GERAÇÃO DO BIE.</p> <p>ACEITA O QUE FOR PASSADO AO NESTED, OU SEJA, UM TRECHO DE CÓDIGO FM QUE PODE INCLUIR: HTML E INTERPOLAÇÃO DE VARIÁVEIS PARA FORMAR O CABEÇALHO E/OU RODAPÉ.</p>		
macro aberturaBIE		
macro corpoBIE		

macro fechoBIE		
macro assinaturaBIE		
	<p>Macros de nível 1 (Primitivas, só possuem o nested definido) para marcação do Diário da Justiça Eletrônico Existe uma outra aplicação que pegará estas marcações nos documentos para a geração do DJE. Aceita o que for passado ao nested, ou seja, um trecho de código FM que pode incluir: HTML e interpolação de variáveis para forma o cabeçalho e/ou rodapé.</p>	
macro numeroDJE		
macro mioloDJE		
macro tituloDJE		
Macros de nível 2		
macro cabecalhoCentralizadoPrimeiraPagina		Quando um documento possui muitas páginas, este é o cabeçalho da primeira página, mais completo. Responsável pelo brasão com a escrita Poder Judiciário / Justiça Federal
macro cabecalhoCentralizado		Quando um documento possui muitas páginas, este é o cabeçalho das páginas subsequentes. Responsável pela escrita Poder Judiciário / Justiça Federal
macro cabecalhoEsquerdaPrimeiraPagina		Quando um documento possui muitas páginas, este é o cabeçalho da primeira página, mais completo. Responsável pelo brasão com a escrita Poder Judiciário / Justiça Federal
macro cabecalhoEsquerda		Quando um documento possui muitas páginas, este é o cabeçalho das páginas subsequentes. Responsável pela escrita Poder Judiciário / Justiça Federal
macro rodapeClassificacaoDocumental	somenteTR=false	Insere a classificação documental no rodapé
macro rodapeNumeracaoADireita		Insere a numeração no rodapé
macro rodapeNumeracaoCentralizada		Insere a numeração no rodapé
macro assinaturaCentro	formatarOrgao=false	Insere Assinatura <i>Macros Chamadas:</i> assinaturaBIE

macro assinaturaMovCentro	formatarOrgao=false	Insere AssinaturaMov, ou seja, assinaturas posteriores na medida que o documento é movimentado / tramitado
Macros de nível 3 = Estilos de Documento		
macro estiloBrasaoAEsquerda	<p>Tipo tamanhoLetra="11pt" obs=""</p> <p>Expicação: obs: provoca inserção de comentário após a assinatura.</p>	<p>Estrutura o lay-out do documento.</p> <p><i>Macros Chamadas:</i></p> <ul style="list-style-type: none"> primeiroCabecalho - nível 1 cabecalhoEsquerdaPrimeiraPagina - nível 2 cabecalho - nível 1 cabecalhoEsquerda - nível 2 assinaturaCentro - nível 2 primeiroRodape - nível 1 rodapeClassificacaoDocumental - nível 2 rodape - nível 1 rodapeNumeracaoDireita - nível 2
macro estiloBrasaoCentralizado	<p>tipo tamanhoLetra="11pt" formatarOrgao=true numeracaoCentralizada=false dataAntesDaAssinatura=false</p> <p>Expicação: formatarOrgao=true: provoca a inserção da lotação abaixo da assinatura. numeracaoCentralizada=true: não funciona, ou seja, a numeração é sempre a direita. dataAntesDaAssinatura=true: provoca a inserção da data (Rio de janeiro) antes (acima) da assinatura.</p>	<p>Estrutura o lay-out do documento.</p> <p><i>Macros Chamadas:</i></p> <ul style="list-style-type: none"> primeiroCabecalho - nível 1 cabecalhoCentralizadoPrimeiraPagina - nível 2 cabecalho - nível 1 cabecalhoCentralizado - nível 2 numeroDJE tituloDJE assinaturaCentro ou assinaturaMovCentro - nível 2 primeiroRodape - nível 1 rodapeClassificacaoDocumental - nível 2 rodape - nível 1 rodapeNumeracaoDireita - nível 2
Macros de nível 4 = Tipo de documento		

macro assentamento_funcional	<pre>texto fecho="" tamanhoLetra="Normal" _tipo="ASSENTAMENTO FUNCIONAL"</pre>	Para documentos tipo Assentamento Funcional <i>Macros Chamadas</i> estiloBrasaoCentralizado - nível 3
macro formulario	<pre>Texto fecho="" tamanhoLetra="Normal" _tipo="FORMULÁRIO"</pre>	Para Documentos tipo Formulário <i>Macros Chamadas:</i> estiloBrasaoCentralizado - nível 3
macro memorando	<pre>texto fecho="Atenciosamente," tamanhoLetra="Normal" _tipo="MEMORANDO"</pre>	Para Documentos tipo Memorando <i>Macros Chamadas:</i> estiloBrasaoCentralizado - nível 3 DE / PARA / ASSUNTO
macro portaria	<pre>texto abertura="" tamanhoLetra="Normal" _tipo="PORTARIA" dispoe_sobre=""</pre>	Para Documentos tipo Portaria <i>Macros Chamadas:</i> estiloBrasaoCentralizado - nível 3 mioloDJE - nível 1 aberturaBIE - nível 1 corpoBIE - nível 1 fechoBIE - nível 1
macro oficio	<pre>_texto="" _tipo_autoridade="" _genero="" _vocativo="" _enderecamento_dest="" _nome_dest=""</pre>	Macro de nível 5. Embora pareça uma macro associada ao documento, esta macro é uma aplicação genérica para ofícios. Ela possui os blocos entrevista e documento.
Outras macros de uso geral utilizadas no documento		
macro letra	tamanho	Define o tamanho da letra. Esta macro chama o método <code>func.fontSize</code> , possivelmente para corrigir o bug (ver capítulo "Lições Aprendidas") no gerador de PDF atual NHEENGA, que está sendo substituído pelo PDF4ML. Esta macro é chamada em diversos pontos de outras macros, como as macros <code>estiloBrasao</code> .

macro fixcrlf	var=""	Faz um replace (Conserta,fix) de \r\f ou \n por (new line)
macro quebraPagina		Força a quebra de página

ASSOCIADA AO BLOCO ESPECIAL

Macros de nível 1 (Primitivas, só possuem o nested definido)

Estas macros permitem customizar o comportamento da aplicação na medida em que ocorre um evento. Estas macros podem aparecer após o bloco @entrevista e @documento

macro finalizacao		Se gerar_finalização é true, então aceita o que for passado ao nested, ou seja, um trecho de código FM que pode incluir: HTML, chamadas a macros, invocação de métodos e etc. Por exemplo, invocar um método do hash func para enviar um e-mail ou iniciar um workflow
macro assinatura		Se gerar_assinatura é true, então aceita o que for passado ao nested, ou seja, um trecho de código FM que pode incluir: HTML, chamadas a macros, invocação de métodos e etc. Por exemplo, invocar um método do hash func para enviar um e-mail ou iniciar um workflow, como a SEC, por exemplo faz, quando o documento é assinado
macro resumo	visivel=false	Se gerar_resumo é true, então aceita o que for passado ao nested, ou seja, um trecho de código FM que pode incluir: HTML, chamadas a macros, invocação de métodos e etc. Por exemplo, invocar um método do hash func para enviar um e-mail ou iniciar um workflow

ASSOCIADA A FUNÇÕES AVANÇADAS DO SIGA-DOC

macro editor	Var titulo semBotaoSalvar=false	Esta macro verifica pelo usuário qual editor ele poderá utilizar. Atualmente temos o editor padrão que é o FCK e o XStandard. Se for o padrão, ele chama o editor FCK, porém se o usuário pode utilizar outro, ele chama a macro extensaoEditor passando a extensão, que no nosso caso é o XStandard, porém pode ser qualquer um. <i>Macros Chamadas:</i> extensaoEditor
macro botoesExtensaoAssinador		Botões responsáveis pela assinatura do documento. Assinar e Configurar assinador da CertSign.
macro dadosComplementares		Não faz nada

macro extensaoBuscaTextual		Utilizada para a busca textual
macro extensaoEditor	nomeExtensao conteudoExtensao	Esta macro chama o editor não padrão, que no nosso caso é o Xstandard, desta forma, chama a macro Xstandard. <i>Macros Chamadas:</i> Xstandard
macro extensaoAssinador		Utilizado na assinatura
macro XStandard	nome="" conteudo=""	Chama o editor Xstandard.
macro topico	descricao valor	Não identifiquei a sua utilização.

O SIGA-DOC fornece uma série de funções JavaScript (JS). O JS é o verdadeiro motor por trás das interfaces do SIGA. Na verdade existe muito mais JS do que FM nas nossas páginas. Muitas macros FM possuem código JS embutido. Sem o JS seria impossível dotar as interfaces com mais interação e beleza, além de não permitir acessos aos dados do servidor sem que toda a página fosse renderizada, função esta exercida pelo AJAX, que é na verdade JS.

É necessário entender JS ou AJAX para desenvolver formulários (docs) para o SIGA-DOC? Não. Na verdade não é necessário nem saber da existência do JS, e se este é o seu caso, você pode pular este capítulo. Agora, se você pretende desenvolver interfaces mais apuradas, ou simplesmente deseja saber como as coisas funcionam por trás dos panos, você poderá ler este capítulo.

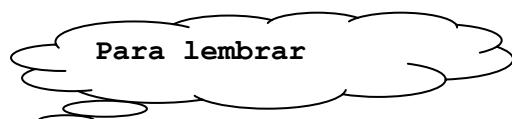
3.1 - Introdução

Novamente repito o que foi dito para o FM. Este não é um capítulo para ensinar JS. Longe disso. Se já não ensinamos o FM, imaginem o JS que é mais complexo e abrangente. Existem milhares de sites e centenas de livros que abordam o assunto. O que vamos comentar aqui são as funções que o SIGA-DOC disponibiliza ao desenvolvedor, sendo algumas para os desenvolvedores de formulários e outras para o desenvolvedor da infra do SIGA. No capítulo "Macros em ação - programação por exemplos" será mostrado como customizar uma macro FM com código JS.

Mas, o que podemos fazer com o JS para ajudar nos formulários? Por exemplo, poderíamos:

- construir campos funcionais, tipo CPF, CNPJ, hora e etc. que já iriam formatando e depois criticando as informações. A macro data visto no capítulo anterior é uma delas.
- campos especiais que não aceitam determinados caracteres tipo: só aceitar números, só aceitar caracter e etc.
- só permitir que determinadas teclas sejam aceitas e executar uma ação caso ela seja pressionada, como o campo subscriptor, por exemplo, que ao digitar e teclar <enter> ele processa uma seleção em busca do nome da pessoa;
- na medida (ou só no final) em que o usuário digita, converter maiúsculo para minúsculo e vice-versa;
- helps para campos, quando o usuário passa o mouse.

Ou seja, uma infinidade de coisas. Na nossa biblioteca JS e na WEB encontramos uma infinidade de aplicações úteis.



Devemos sempre ter em mente que:

JS roda no cliente

FM roda no servidor

AJAX roda no cliente e acessa o servidor

Onde estão as aplicações JS, pois não as vejo? Acredite, estão lá no seu browser rodando e são dezenas delas. Só com um depurador de browser elas podem ser identificadas (ver capítulo "Debugando a Aplicação").

3.2 - JavaScript- JS

Como tudo em JS gira em torno de função (chegam a chamar de programador funcional), vale a pena uma pequena categorização.

Observação: O JS não tem nada a ver com JAVA.

3.2.1 - Um método é pelo local onde elas estão armazenadas / definidas

3.2.1.1 - Em arquivo externo (bibliotecas)

Neste caso temos que definir a source indicando um arquivo.js de onde serão carregadas as funções:

```
<script type="text/javascript" src="/sigae/sigalibs/mensagensremotas.js">
</script>
```

3.2.1.2 - Inline, dentro da página

Neste caso devemos defini-la entre as tags do <body> do HTML. Observar que não existe a source.

```
...
<body>
<script type="text/javascript">
function noNumbers(e)
{
var keynum;
var keychar;
var numcheck;

if(window.event) // IE8 and earlier
{
keynum = e.keyCode;
}
else if(e.which) // IE9/Firefox/Chrome/Opera/Safari
{
keynum = e.which;
}
keychar = String.fromCharCode(keynum);
numcheck = /\d/;
return !numcheck.test(keychar);
}
</script>
...
```

3.2.1.3 - Inline, dentro dos eventos

Observar que aqui não tem source e nem a tag <script>.

```
<input type="text" value="digite aqui" onblur="if(this.value == '') {this.value =
'digite aqui';}" onfocus="if(this.value == 'digite aqui') {this.value = '';}"
/>
```

Observar que este campo possui duas funções definidas, um para cada evento. O onblur é quando o campo perde o foco, onfocus é quando o campo ganha o foco. Ver eventos na próxima seção.

3.2.2 - Pela classificação do depuradores de código (firebug, por exemplo)

3.2.2.1 - Estáticas

Seriam aquelas classificadas anteriormente como externas ou inline, dentro da página.

3.2.2.2 - Eval (de evaluation)

São aquelas criadas e executadas dinamicamente no código.

```
var dynamicScript = 'alert(\''Hello world!\');';
eval(dynamicScript);
```

dynamicScript sera avaliado (interpretado) e executado.

3.2.2.3 - Eventos

Seriam aquelas classificadas anteriormente como inline, dentro dos eventos.

3.2.3 - Pela classificação funcional

3.2.3.1 - Função tradicional

```
function hello(who) {  
    alert('Hello '+who); // outputs "Hello world"  
}  
  
hello('world');
```

3.2.3.2 - Função como variável

```
var hello = function (who) {  
    alert('Hello '+who); // outputs "Hello world"  
}  
  
hello('world');
```

3.2.3.3 - Passando uma função para uma função

Pode-se usar uma função lambda (passar uma função como argumento para outra função).

```
var say = function(what) {  
    what('say function');  
}  
  
var hello = function (who) {  
    alert('Hello '+who); // outputs "Hello say function"  
}  
  
say(hello); // função lambda
```

3.2.4 - Pela classificação nominal

3.2.4.1 - Inominada (Anônima)

```
var = function(who)  
{  
    alert('Hello '+who);  
}
```

3.2.4.2 - Nominada

```
function hello(who)  
{  
    alert('Hello '+who);  
}
```

Só de curiosidade:

```
var test = function () {  
    return "This is a String";  
}  
  
var testCopy = test; // testCopy is a pointer to the test function()  
var testStr = test(); // testStr contains "This is a string"  
var testing = testCopy(); // testing contains "This is a string"
```

3.3 – Document Object Model (DOM)

Outra buzzword. **DOM** (Document Object Model – Modelo de Objetos de Documentos) é uma especificação da W3C, independente de plataforma e linguagem, **onde se pode dinamicamente alterar e editar a estrutura, conteúdo e estilo de um documento.** A API DOM oferece uma maneira padrão de se acessar os elementos de um documento, além de se poder trabalhar com cada um desses elementos separadamente, e por esses motivos criar páginas altamente dinâmicas.
[\(http://pt.wikipedia.org/wiki/Modelo_de_Objeto_de_Documentos\)](http://pt.wikipedia.org/wiki/Modelo_de_Objeto_de_Documentos)

Desta forma, podemos concluir que o JS não seria (faria) nada sem o DOM. Todos os efeitos que vemos em sites, tal como aparecer um texto / gráfico / imagem (ou desaparecer) sem a renderização da página é feita pelo DOM. Os eventos de passagem de mouse, foco no campo, tecla pressionada, e outros como veremos baixo, também é capturado pelo DOM. O JS, como outras, é a linguagem que se utiliza das APIs do DOM.

Observação: temos HTML DOM e XML DOM.

Novamente repito a velha ladainha, o objetivo desta seção não é ensinar DOM e sim chamar atenção para esta tecnologia. A internet está cheia de material sobre o assunto.

3.3.1 – Eventos HTML DOM

Os eventos formam a base da programação orientada a eventos utilizada na programação cliente tradicional (visual basic, delphy e etc.).

Estes eventos podem ser capturados pelo próprio HTML ou pelo JS.

Exemplos:

Via HTML:
`<input id="lotacaoDestinatarioSelButton" type="button" theme="simple" onclick="popitup_lotacaoDestinatario();" value="...">`
Se clicarmos no botão a função popitup_lotacaoDestinatario() será executada.

Via JS: `document.FormName.ButtonName.onclick= function() {alert('Here is a pop up message');};`

Onde: formname é nome do formulário e buttonname é nome do botão.

Os eventos tratados pelo HTML são mais comuns.

Evento	Descrição	Onde utilizar
Abort (onAbort)	Este evento acontece quando o usuário interrompe o carregamento da imagem	Image
Blur (onBlur)	Ocorre quando o elemento perde o foco, ou seja, quando o usuário clica fora do elemento, este não será mais selecionado como estando ativo.	Button, Checkbox, FileUpload, Layer, Password, Radio, Reset, Select, Submit, Text, TextArea, window
Change (onChange)	Ocorre quando o usuário altera o conteúdo de um campo de dados.	FileUpload, Select, Submit, Text, TextArea
Click (onClick)	Ocorre quando o usuário clica no elemento associado ao evento.	Button, document, Checkbox, Link, Radio, Reset, Select, Submit
dragdrop (onDragdrop)	Ocorre quando o usuário efetua um arrastar-largar na janela do navegador. Este evento só é suportado pelas versões do JavaScript 1.2 e superior	window

error (onError)	É desencadeada quando ocorre um erro durante o carregamento da página. Este evento faz parte do JavaScript 1.1.	Image, window
dblclick (onDblclick)	Ocorre quando o usuário clica duas vezes no elemento associado ao evento (um hiperlink ou um elemento de formulário). Este evento só é suportado pelas versões do JavaScript 1.2 e superior	document, Link
Focus (onFocus)	Ocorre quando o usuário dá foco a um elemento, isso significa que este elemento foi selecionado como estando ativo	Button, Checkbox, FileUpload, Layer, Password, Radio, Reset, Select, Submit, Text, TextArea, window
keydown (onKeydown)	Ocorre quando o usuário pressiona uma tecla do seu teclado. Este evento só é suportado pelas versões do JavaScript 1.2 e superior	document, Image, Link, TextArea
keypress (onKeyPress)	Ocorre quando o usuário mantém pressionada uma tecla do seu teclado. Este evento só é suportado pelas versões do JavaScript 1.2 e superior	document, Image, Link, TextArea
keyup (onKeyup)	Ocorre quando o usuário solta uma tecla do seu teclado antepressionada anteriormente. Este evento só é suportado pelas versões do JavaScript 1.2 e superior	document, Image, Link, TextArea
Load (onLoad)	Ocorre quando o navegador do usuário carrega a página em curso	Image, Layer, window
MouseOver (onMouseOver)	Ocorre quando o usuário põe o cursor do mouse acima de um elemento	Area, Layer, Link
MouseOut (onMouseOut)	Ocorre quando o cursor do mouse deixa um elemento. Este evento faz parte do Javascript 1.1.	Layer, Link
Reset (onReset)	Ocorre quando o usuário apaga os dados de um formulário com o botão Reset.	form
Resize (onResize)	Ocorre quando o usuário redimensiona a janela do navegador	window
Select (onSelect)	Ocorre quando o usuário seleciona um texto (ou uma parte de um texto) em um campo do tipo "text" ou "textarea"	text, Textarea
Submit (onSubmit)	Ocorre quando o usuário clica no botão de submissão de um formulário (o botão que permite enviar o formulário)	Form
Unload (onUnload)	Ocorre quando o navegador do usuário sai da página em curso	window

Para uma lista de eventos do JS, consultar:
<http://www.mspc.eng.br/info/jscriptContrEv.shtml>

3.3.2 - Elementos HTML DOM

Um documento é composto por uma hierarquia de elementos. Esta hierarquia é chamada de árvore de nodos (nós). Os nodos podem ser classificados em:

- o Element
- o Text
- o Attributes

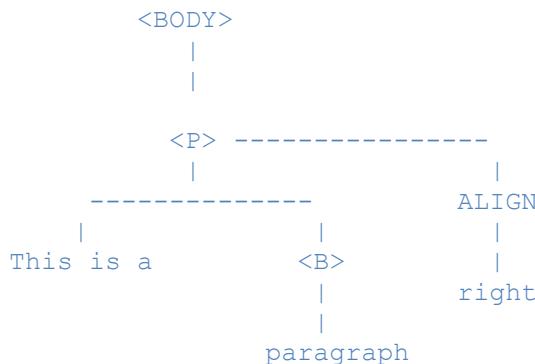
Existem outros tipos, porém a classificação acima atende a 99% dos casos.

Exemplo:

Esta pequena linha HTML ...

```
<P ALIGN="right">This is a <B>paragraph</B></P>
```

Gera a seguinte árvore ...



Obs: O <BODY> tem um pai que é o nodo HTML, que por sua vez, tem um nodo pai, raiz, chamado document, que todo documento possui.

NAVEGANDO NA ÁRVORE

Supor que x é uma variável que deve apontar para o elemento . Como fazer?

```
var x = document.getElementsByTagName('B')[0];
OU
var x = document.getElementsByTagName('P')[0].lastChild;
```

Como podemos navegar nesta árvore?

x.parentNode	Nos levará ao nodo element	<P>
x.childNodes[0]	Nos levará ao nodo text	paragraph
x.childNodes[1]	Nos levará ao nodo element	não existe

Existem comandos para alterar atributos (cor, fonte, tamanho ...) e criar, inserir, excluir nodos, permitindo que a página seja dinamicamente modificada.

ALTERANDO A ESTRUTURA DA ÁRVORE

Supor esta linha HTML:

```
<P ALIGN="right">This is a <B ID="hereweare">paragraph</B></P>
```

Criando um nodo Element ...

```
var x = document.createElement('HR');
```

Adicionando um nodo como filho ...

```
document.getElementById('inserthrhhere').appendChild(x);
```

Removendo um nodo ...

```
var x = document.getElementById('inserthrhhere')
x.removeChild(node.childNodes[0]); --- roda melhor no IE ----
```

OU

```
var Node1 = document.getElementById("Id_aqui"); ---- Roda melhor no NS ----
```

```

var len = Node1.childNodes.length;

for(var i = 0; i < len; i++)
{
    if(Node1.childNodes[i].id == "Child")
    {
        Node1.removeChild(Node1.childNodes[i]);
    }
}

```

Criando um nodo Text e adicionando-o como filho ...

```

var x = document.createTextNode(' A new text node has been appended!');
document.getElementById('hereweare').appendChild(x);

```

Observação: A MS disponibiliza as seguintes propriedades de um elemento (objeto): o innerHTML e outerHTML, que não é padrão DOM (e nem entendido por todos browsers), e que pode ser melhor entendido pelo exemplo a seguir.

Se fizermos `objectid.innerHTML = 'conteudo'`; estaremos jogando 'conteudo' no objectid.

O objectid pode ser obtido via `document.getElementById('id_do_elemento')`. Para conseguir o mesmo efeito com as APIs do DOM, teríamos que, no caso do inner por exemplo, varrer todos os filhos (e seus filhos ...) do elemento. Muito mais trabalhoso, pois o `nodevalue` do DOM só retorna texto puro.

Exemplo de inner e outerHTML.

The screenshot shows a developer tool interface with two alerts. On the left, under 'Edit and Click Me >', is the following HTML code:

```

<html>
<head>
</head>
<body>
<a href="#" onclick="alert(this.innerHTML)"> Exemplo de <b>innerHTML</b></a>
<a href="#" onclick="alert(this.outerHTML)">Exemplo de <b>outerHTML</b></a>
</body>
</html>

```

Two arrows point from the text 'innerHTML' and 'outerHTML' in the code to their respective alert boxes. The first alert, titled 'A página em www.w3schools.com says:', contains the text 'Exemplo de innerHTML' and an 'OK' button. The second alert, also titled 'A página em www.w3schools.com says:', contains the full HTML code from the 'innerHTML' example and an 'OK' button.

Observação:

`window.document.childNodes[0]` Seria o HTML elemento

Obs: o window. Não é necessário.

`document.childNodes[0].childNodes[0]` Seria o head tag.

`document.childNodes[0].childNodes[1]` Seria o body tag.

`document.head` Seria o shorcut para o head tag.

`document.body` Seria o shortcut para o body tag.

3.4 – AJAX

Caramba, como faço confusão com isto. Cada vez que leio um artigo na internet fico mais confuso ainda. Tentarei passar aqui a minha humilde e neófita opinião sobre o assunto.

AJAX é o acrônimo de Asynchronous Javascript and XML.

O que não é AJAX.

Muita gente pensa que AJAX é sinônimo fazer aparecer (ou desaparecer) elementos da página sem a necessidade de renderizar toda ela. No exemplo abaixo, existe um alerta entre o mês de referência e a lotação. Quando o usuário digita o ano, este alerta desaparece, com efeito suave, chamado de slide.

Quando o alerta entra existe um slide down daquele ponto em diante, ou seja, empurra a lotação para baixo. Quando ele sai, ocorre um slide up, ou seja, a lotação sobe.



Como já vimos na seção anterior, isto ocorre pela manipulação dos elementos DOM (**appendChild** e **removeChild**, por exemplo) somente, e o JS pode cuidar muito bem disto sem a necessidade do AJAX. Outra forma de se obter o mesmo efeito das APIs do Dom é utilizando uma característica do HTML que é: injetar conteúdo na div e injetar “nulo” na div, pois o browser não mostra `<div>` com conteúdo nulo (esta técnica é utilizada pelo SIGA-DOC na macro *Grupo depende*).

```
<div id="xpto">
  <b>Alerta</b>: Ano deve ser preenchido.
</div>
```

```
<div id="xpto">
</div>
```

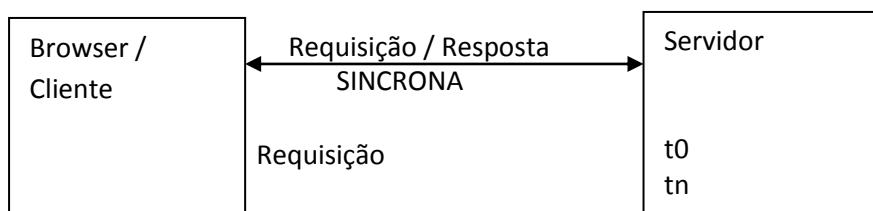
Faz sentido ir ao servidor verificar a ausência de uma informação e alertar o usuário? Agora, se o alerta tiver alguma informação que só o servidor pode obter (como por exemplo: a hora do servidor, um acesso ao BD ...), aí sim, temos o AJAX, pela informação, e não pelo efeito.

Obs: os trechos da tela acima são parte uma aplicação FM em produção. O campo ano foi definido com `idAjax` e o alerta numa macro `grupo com depende` (ver capítulo “FM em Ação”), e está implementando AJAX! O motivo é a estrutura do SIGA-DOC. Falaremos disso mais adiante com dois exemplos (itens 3.4.1 e 3.4.2).

O Que é o AJAX para mim. Sem aquele papo de um conjunto de tecnologias ...

A implementação do AJAX é feita com uma biblioteca JS, que disponibiliza um objeto chamado XMLHttpRequest que possui métodos e propriedades para abrir uma conexão Http entre o cliente (browser) e o servidor, e desta forma permitir que uma aplicação cliente converse com uma aplicação servidora. Baseado neste fato prefiro definir o AJAX como um protocolo de conversação de aplicações cliente (JS) / servidor, como o famoso RPC (Remote Procedure Call).

Quais são as características destas conversações:





t_n pode ser t₁, t₂ ..., ou seja, é defasado da requisição (t₀)

➤ **As conversações podem ser Síncronas ou Assíncronas. O default é assíncrono.**

Como saber se necessito uma conversação assíncrona ou síncrona?

Síncrona: o cliente fica esperando (e o usuário também) a resposta do servidor. Se o servidor estiver muito ocupado ou a aplicação servidora for muita pesada, o usuário poderá pensar que o browser congelou.

Assíncrona: o cliente faz a requisição e libera a página para que o usuário possa fazer outra operação. A resposta chega defasada em relação à requisição. Se o servidor estiver folgado, a resposta vem imediatamente.

E aí, qual escolher? Novamente vejo muita confusão sobre este assunto. O assíncrono libera o usuário para fazer outra operação, porém fazer o quê? Ficar clicando nos campos da página?

- Se eu (ou a aplicação) dependo da resposta para tomar uma decisão, então eu quero que seja síncrono;
- Se eu posso continuar e realizar algo de útil com a página liberada, então prefiro assíncrono.

Como o default é assíncrono, lembre-se do A do AJAX, então as pessoas acabam deixando assim. Dois bons exemplos de conversação VERDADEIRAMENTE assíncrona vêm do Google. No Gmail, quando você está compondo a mensagem, você pode anexar N arquivos. O processo de anexação é assíncrono e libera o usuário para continuar a edição da mensagem, por exemplo. No GoogleMap, enquanto você está passando a mão em alguma região e ele observa que as próximas regiões não estão carregadas, ele em modo silencioso e assíncrono vai no servidor buscar mais regiões. Imagine se ao iniciar o Googlemap ele trouxesse todo o globo terrestre! Pensaríamos que a aplicação tivesse congelado.

Falso Assíncrono, Falso AJAX. Existe uma aplicação de Harvard que exibe lindos gráficos e permite uma interação ótima com o usuário, selecionando os dados, tipos de gráficos e etc. Muitas pessoas pensam que tem AJAX, mas na verdade é puro JS, pois quando a aplicação é iniciada, ela demora um pouco, visto que está carregando os dados. Daí em diante, é só firula no cliente. Se os dados fossem carregados por demanda, a carga inicial da aplicação seria mais rápida, e aí teríamos AJAX.

➤ **A requisição pode ser pelo método GET ou POST**

Novamente confusão, pois devido aos nomes, dá a entender que GET é para ler do servidor e POST para enviar ao servidor.

Um bom esclarecimento está no site <http://www.comocriarsites.com/html/como-funciona-os-metodos-get-e-post-diferencias/>. Como resumo, transcrevo a dica do site.

A diferença é simples, sempre que for buscar ou apenas consultar alguma coisa, utilize GET e se for fazer alguma alteração com a requisição, envio de arquivo ou os dados forem muito (em quantidade ou tamanho), utilize POST.

Na requisição, deve-se indicar a URL. A URL indica a aplicação no servidor que processará a requisição e promoverá uma resposta.

➤ **A resposta pode vir em formato Texto ou HTML / XML**

É isso mesmo, você pode escolher texto (responseText) ou HTML / XML (responseXML), sendo texto a mais comum! Você pode testar o header da resposta para verificar o tipo de conteúdo, Texto ou XML. Ver trecho do programa abaixo.

```
var cType = this.getResponseHeader("Content-Type");  
if (cType == 'text/xml') {  
// XML response  
} else if (cType == 'text/plain') {  
// plain text response  
} else {  
alert('unknown content type');  
}
```

➤ **Deve-se indicar qual a função de retorno (callback)**

A biblioteca AJAX entregará a resposta a uma função JS que o programador desenvolveu. Desta forma, o programado na chamada AJAX deve passar o nome desta função como parâmetro.

Então, se a comunicação
pode ser síncrona e a
resposta pode ser texto o
AJAX pode virar JA, ou SJAT



O Objeto XMLHttpRequest disponibilizado ao programador pela biblioteca AJAX

Classe XMLHttpRequest



Agora, a variável http conterá um ponteiro para o objeto e poderá utilizar os métodos e propriedades, como por exemplo, `http.open(parâmetros a passar)`.

A partir da instanciação, os seguintes métodos e propriedades estarão disponíveis:

XMLHttpRequest Object Methods

Method	Description
abort()	Cancels the current request
getAllResponseHeaders()	Returns header information
getResponseHeader()	Returns specific header information
open(method,url,async,uname,pswd)	Specifies the type of request, the URL, if the request should be handled asynchronously or not, and other optional attributes of a request method: the type of request: GET or POST url: the location of the file on the server async: true (asynchronous) or false (synchronous)
send(string)	send(string) Sends the request off to the server. string: Only used for POST requests
setRequestHeader()	Adds a label/value pair to the header to be sent

XMLHttpRequest Object Properties

Property	Description
onreadystatechange	Stores a callback function (or the name of a function) to be called automatically each time the readyState property changes
readyState	Holds the status of the XMLHttpRequest. Changes from 0 to 4: 0: request not initialized 1: server connection established

	2: request received 3: processing request 4: request finished and response is ready
responseText	Returns the response data as a string
responseXML	Returns the response data as XML data
status	Returns the status-number (e.g. "404" for "Not Found" or "200" for "OK")
statusText	Returns the status-text (e.g. "Not Found" or "OK")

3.4.1 - Modelo 1 - AJAX em ação no SIGA-DOC nos campos subscritor, titular, destinatário e classificação da parte fixa da entrevista

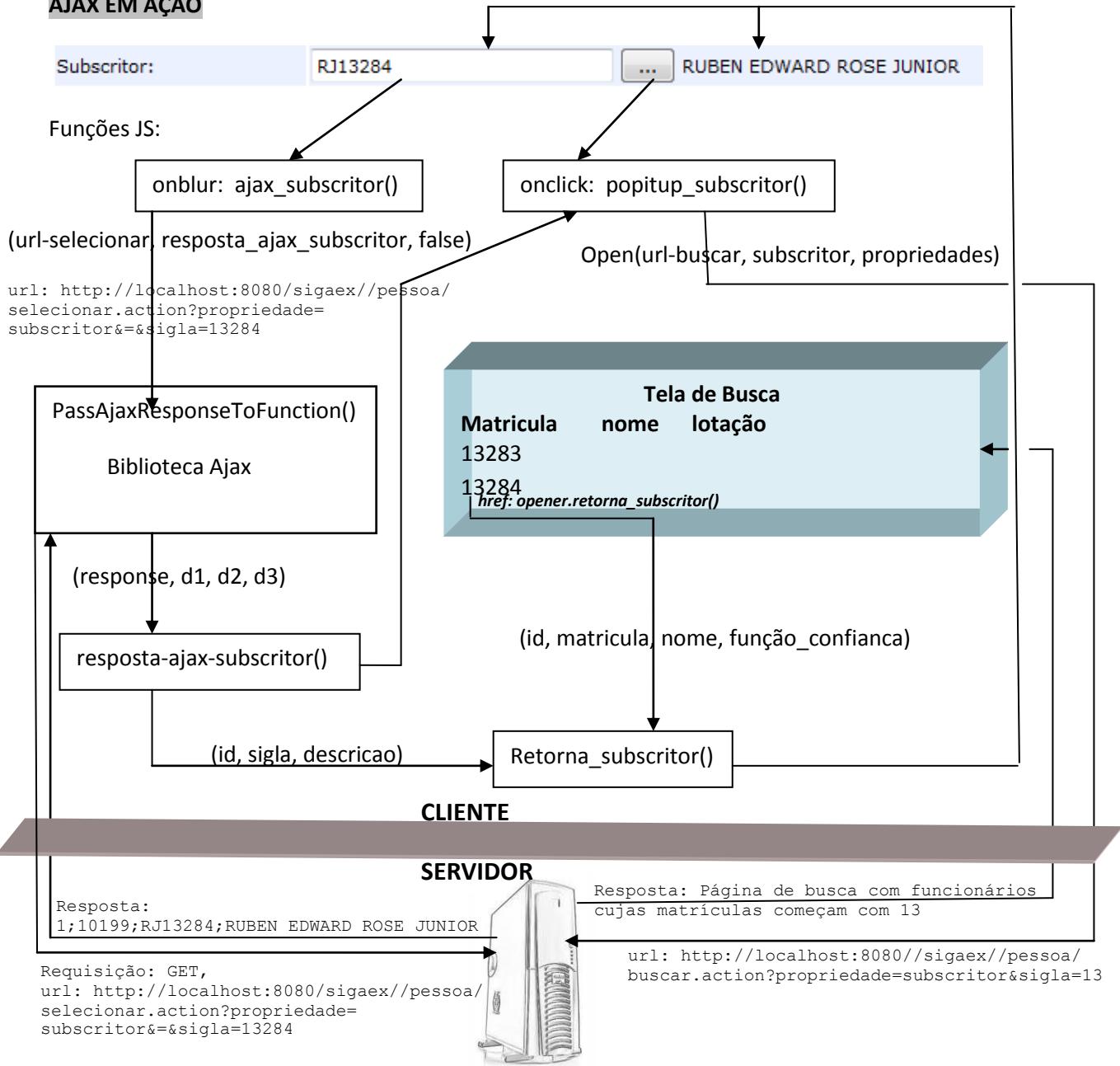
Como as funções JS / AJAX do subscritor, titular, destinatário e classificação funcionam e se relacionam? A ideia aqui é mostrar a complexidade por baixo dos panos, para que saibamos o que está acontecendo de verdade. Para tanto vamos nos basear no exemplo do subscritor.

O que acontece aqui:

- Quando o usuário digita a sigla (matricula) do subscritor e tecla <enter> ou deixa o campo (evento onblur) verifica-se se o mesmo existe. Em caso positivo (13284, por exemplo), retorna-se a descrição (nome) **via AJAX**. Caso contrário, abre-se a janela de busca com funcionários cujas matrículas possuem a string digitada no campo sigla (13, por exemplo);
- Quando o usuário tecla o botão ... é aberta uma janela de busca onde o usuário pode selecionar a pessoa desejada, e neste caso, a aplicação retorna a sigla e a descrição, **porém não utiliza AJAX**.

Observar que a função (resposta-ajax-subscritor()) passada como parâmetro na chamada da PassAjaxResponseToFunction() é a função de callback (ou retorno) utilizada pelo AJAX para devolver a resposta (o resultado) do servidor. A url também passada a esta função, aponta para a aplicação que vai produzir as informações.

AJAX EM AÇÃO



As funções declaradas acima estarão listadas nas tabelas do item 3.5.

As URLs apontam para as actions (aplicações) no servidor. Abaixo, as urls utilizadas para **buscar (tela de busca)** e **selecionar** informação:

REFERÊNCIAS DE URL DO BUSCAR.ACTION (Para tela de busca)

Aqui, ... significa, <http://localhost:8080>, ou seja, servidor de desenvolvimento.

- .../sigaex//expediente/buscar.action?propriedade=mobilPai&sigla=&
- .../sigaex//classificacao/buscar.action?propriedade=classificacao&sigla=&
- .../sigaex//pessoa/buscar.action?propriedade=destinatario&sigla=&
- .../sigaex//pessoa/buscar.action?propriedade=subscritor&sigla=&
- .../sigaex//pessoa/buscar.action?propriedade=titular&sigla=&
- .../sigaex//lotacao/buscar.action?propriedade=lotacaoDestinatario&sigla=&
- .../sigaex//orgao/buscar.action?propriedade=orgaoExternoDestinatario&sigla=&

REFERÉNCIAS DE URL DO SELECIONAR.ACTION (Para selecionar informações)

```
..../sigaex//expediente/selecionar.action?propriedade=mobilPai'+'&=
..../sigaex//classificacao/selecionar.action?propriedade=classificacao'+'&=
..../sigaex//pessoa/selecionar.action?propriedade=subscritor'+'&=
..../sigaex//pessoa/selecionar.action?propriedade=titular'+'&=
..../sigaex//pessoa/selecionar.action?propriedade=destinatario' '+'&=
..../sigaex//lotacao/selecionar.action?propriedade=lotacaoDestinatario'+'&=
..../sigaex//orgao/selecionar.action?propriedade=orgaoExternoDestinatario'+'&=
```

3.4.1.1 - Modelo 1 em @lotacao, @pessoa, @funcao - Meta Função AJAX

A mesma lógica é realizada na parte variável da entrevista com as macros do Freemarker @lotacao, @pessoa e @funcao. Estas macros chamam a macro @selecionavel que por sua vez, chama a macro @caixaSelecao que:

- Produz o seguinte código HTML

```
<input type="hidden" name="${var}${tipoSel}Sel.id" />
<input type="hidden" name="${var}${tipoSel}Sel.descricao" />
<input type="hidden" name="${var}${tipoSel}Sel.buscar" />
<input type="hidden" name="req${var}${tipoSel}Sel" />
<input type="hidden" name="alterouSel" value="" id="alterouSel" />
<input type="text" name="${var}${tipoSel}Sel.sigla"
onkeypress="return handleEnter(this, event)"
onblur="javascript: ajax_${var}${tipoSel}();" size="25"
${desativar?string('disabled="true','','')} />
[#if buscar]
<input type="button" id="${var}${tipoSel}SelButton" value="..." 
onclick="javascript: popup_${var}${tipoSel}('')"
${desativar?string("disabled","","")} theme="simple">
[/#if]
```

Onde: **tipoSel** = {lotacao, pessoa, funcao} e **var** = nome da variável passada a macro @lotacao, @pessoa ou @funcao.

Var é porque podemos ter vários macros, pessoa por exemplo, na entrevista.

- e os seguintes trechos de código JS

Função chamada no evento onblur

```
self.ajax_${var}${tipoSel} =
function() {
var sigla =
document.getElementsByName('${var}${tipoSel}Sel.sigla')[0].value;
if (sigla == '') {
return retorna_${var}${tipoSel}('', '', '');
}
var url =
'${acaoBusca}/selecionar.action?var=${var}
${tipoSel}&sigla=' + encodeURI(sigla) +'${sel
ecaoParams!}';
url = url + '&sigla=' + sigla;
PassAjaxResponseToFunction(url,
'reposta_ajax_${var}${tipoSel}', false);
}
```

Função de callback (ou retorno, ou resposta)

```
self.resposta_ajax_${var}${tipoSel} =
function(response, d1, d2, d3) {
var sigla =
document.getElementsByName('${var}${tipoSel}Se
l.sigla')[0].value;
var data = response.split(',');
if (data[0] == '1')
return retorna_${var}${tipoSel}(data[1],
data[2], data[3]);
retorna_${var}${tipoSel}('', '', '');
[#if buscar]
return popup_${var}${tipoSel}(sigla);
[#else]
return;
[/#if]
}
```

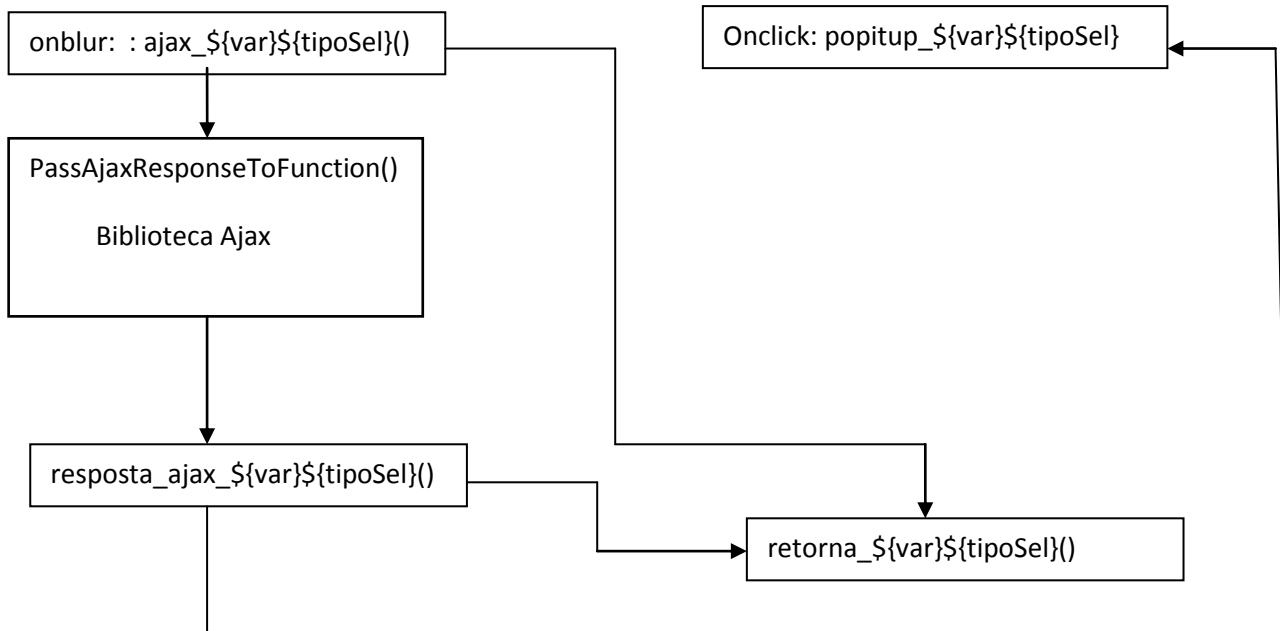
Função para retornar a sigla e descrição

```
self.retorna_${var}${tipoSel} =
function(id, sigla, descricao) {
try {
newwindow_${var}.close();
} catch (E) {
} finally {
}
document.getElementsByName('${var}${tipoSel}Sel.id')[0].value = id;
[#:if !ocultarDescricao]
try {
document.getElementsByName('${var}${tipoSel}Sel.descricao')[0].value = descricao;
document.getElementById('${var}${tipoSel}SelSpan').innerHTML = descricao;
} catch (E) {
}
[#:if]
document.getElementsByName('${var}${tipoSel}Sel.sigla')[0].value = sigla;
[#:if reler]
//window.alert("vou reler tudo!");
document.getElementsByName('req${var}${tipoSel}Sel')[0].value = "sim";
document.getElementById('alterouSel').value='${var}';
sbmt([#:if idAjax != ""] '${idAjax}' [#:if]);
[#:if]
}
self.newwindow_${var} = '';
}
```

Função para exibir a janela de pesquisa

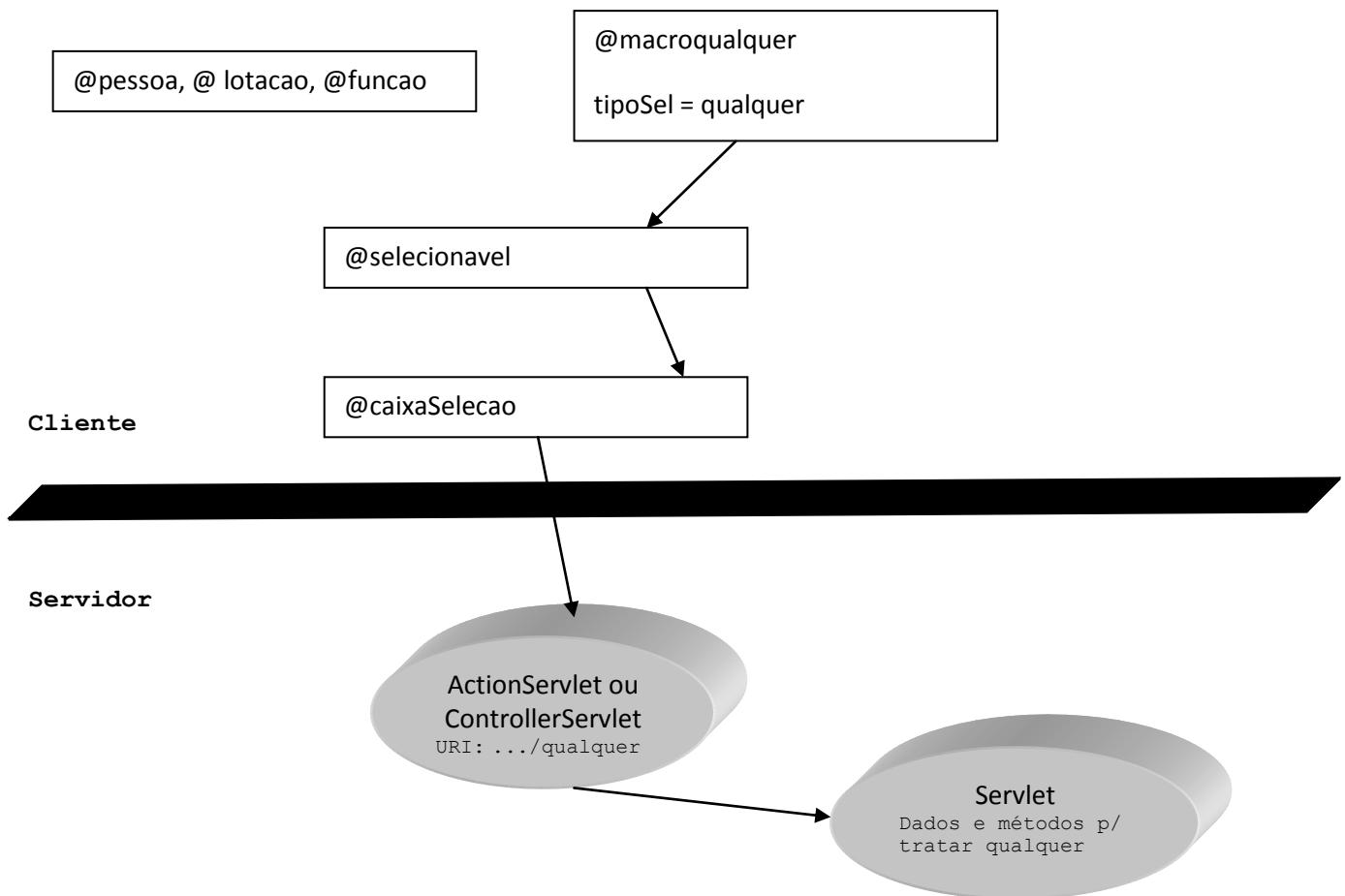
```
self.popitup_${var}${tipoSel} =
function(sigla) {
var url =
'${acaoBusca}/buscar.action?propriedade=${var}&${tipoSel}&sigla='+encodeURI(sigla)+'${selecaoParams!}';
if (!newwindow_${var}.closed && newwindow_${var}.location) {
newwindow_${var}.location.href = url;
} else {
var popW = ${larguraPopup};
var popH = ${alturaPopup};
[#:if grande]
popW = screen.width*0.75;
popH = screen.height*0.75;
[#:if]
var winleft = (screen.width - popW) / 2;
var winUp = (screen.height - popH) / 2;
winProp =
'width=' + popW + ',height=' + popH + ',left=' + winleft +
',top=' + winUp + ',scrollbars=yes,resizable';
newwindow_${var}=window.open(url,'${var}${tipoSel}',winProp);
}
newwindow_${var}.opener = self;
if (window.focus) {
newwindow_${var}.focus();
}
return false;
}
```

A partir do descrito, temos o seguinte diagrama que é idêntico ao do item 3.4.1



É importante observar que esta estrutura cria um framework com metas funções para criarmos macros que recuperem ID, SIGLA e DESCRIÇÃO de qualquer coisa.

Até as URLs estão parametrizadas, observar o destaque em cinza no código.



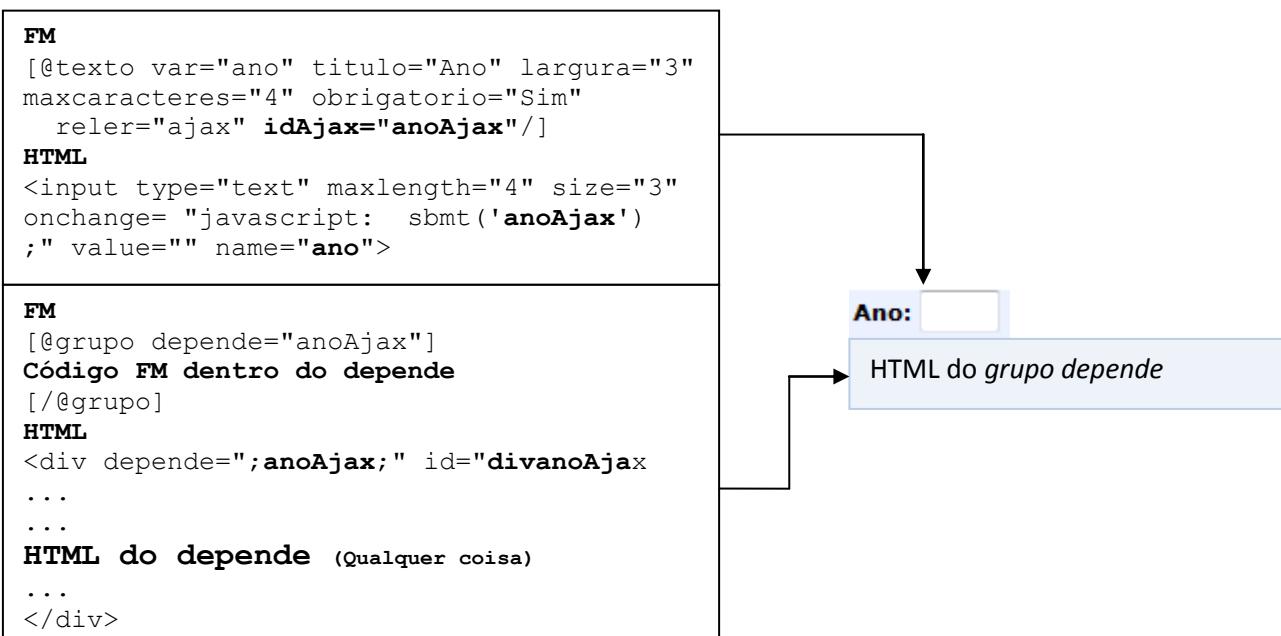
É óbvio que no lado servidor teremos que criar o servlet (action servlet) que será responsável por produzir as informações.

3.4.2 - Modelo 2 - AJAX em ação no SIGA-DOC nos campos definidos com idAjax e macro FM Grupo com a clausula depende nos campos da parte variável da entrevista

Como funciona o idAJAX. Quando você define um campo (supor, ano) com este *id* (supor, anoAjax) e define o bloco *grupo depende* vinculando a este campo são gerados os statements HTML abaixo. O vínculo entre o campo e o grupo depende é lógico e não físico como poderia se imaginar, ou seja, se entre a definição do campo ano e o grupo depende houver outro campo (por exemplo, uma mensagem), o HTML gerado pelo grupo depende vai ficar debaixo deste campo mensagem. Por isso, quando se cria uma campo com idAjax, a definição subsequente dever ser, por questão de lógica, do grupo depende, para que todo HTML entre debaixo do campo em questão.

Se observarmos abaixo, quando definimos um campo com idAjax é criado no HTML um evento *onchange* que, disparado, executa uma função JS *sbmt()* passando como parâmetro o *idAjax*, que inicia a conversão AJAX, como veremos daqui a pouco os detalhes.

Então quer dizer que esta função vai no servidor e só executa o *grupo depende* para obter o HTML gerado por ele? Não. Como veremos a aplicação FM é executada normalmente e vários statementes HTML são devolvidos como resposta, porém, as funções da biblioteca AJAX saberão separar o joio do trigo e só devolverão o HTML gerado pelo grupo depende. Lembre-se que passamos o *idAjax* como parâmetro.



Daremos um exemplo baseado num formulário real que estará disponível no capítulo “**Anexos**”. Não é ainda momento de vermos a programação do FM, por isso, caso você não entenda o código perfeitamente, não tem problema. Isto será explicado no capítulo “**MACROS EM AÇÃO - PROGRAMAÇÃO POR EXEMPLOS**”.

Trecho do código FM da aplicação “Comunicações de Alterações de Frequência”

```
[@entrevista]
[@grupo titulo="Comunicações de Alterações de Frequência"]
[@selecao var="mes" titulo="Mês de referência"
opcoes="Jan;Fev;Mar;Abr;Mai;Jun;Jul;Ago;Set;Out;Nov;Dez"]
[@texto var="ano" titulo="Ano" largura="3" maxcaracteres="4" obrigatorio="Sim"
reler="ajax" idAjax="anoAjax"]]
```

```

[@grupo]
[@grupo depende="anoAjax"]
[#if (ano "") == ""]
  [@mensagem titulo="Alerta" texto="Ano deve ser preenchido." vermelho=true/]
[/#if]
[@grupo]
[@lotacao var="lotacao" titulo="Lotação" reler=true idAjax="lotacaoAjax"][@br/]

```

E o código continua

Quando rodamos a primeira vez é isto que aparece:

O Alerta já aparece porque no inicio o ano está vazio.

The screenshot shows a user interface with a dropdown menu for 'Mês de referência' set to 'Jan'. Below it, there is a text input field labeled 'Ano:' which is empty. A red alert message 'Alerta: Ano deve ser preenchido.' is displayed above a button labeled 'Lotação:'. There is also a small '...' button next to the 'Lotação' input field.

Trecho do html gerado:

```

<input type="text" maxlength="4" size="3" onchange="javascript:
sbmt('anoAjax');" value="" name="ano">
<div depende=";anoAjax;" id="divanoAjax"><!--ajax:divanoAjax--><table
width="100%" class="entrevista">
<tbody><tr>
<td><span style="color:#ff0000"><b>Alerta</b>: Ano deve ser preenchido.</span>
</td>
</tr>
</tbody></table>
<!--/ajax:divanoAjax-->
</div>

```

Pois, a <div> "divanoAjax" possui um elemento span cujo conteúdo é a mensagem de alerta.

Quando fornecemos o ano, é isto que acontece, um slide up:

The screenshot shows the same interface as before, but now the 'Ano:' input field contains the value '2012'. The alert message has disappeared, indicating that the slide-up effect has occurred.

Trecho do HTML gerado:

```

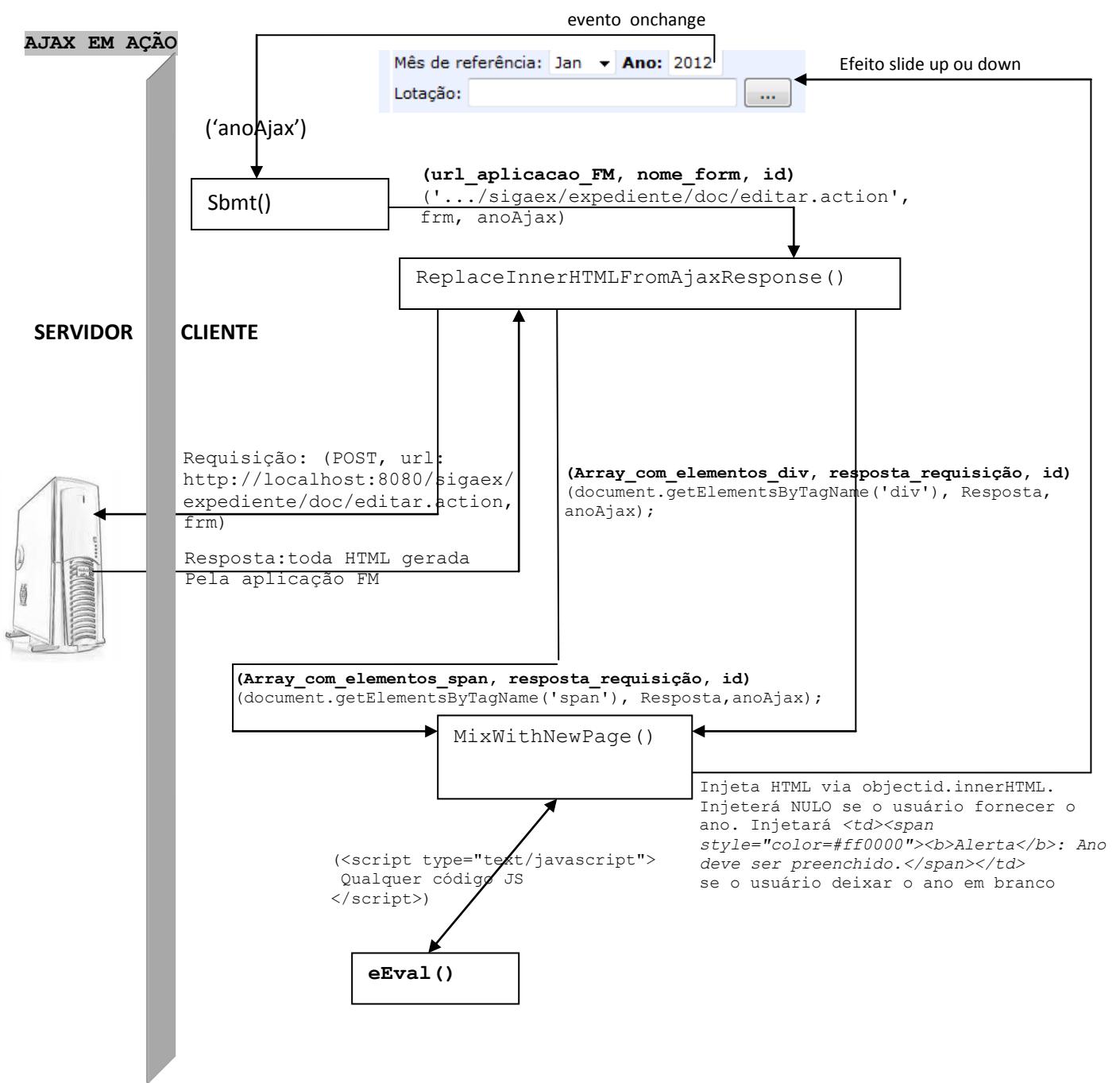
<input type="text" maxlength="4" size="3" onchange="javascript:
sbmt('anoAjax');" value="" name="ano">
<div depende=";anoAjax;" id="divanoAjax"><!--ajax:divanoAjax--><table
width="100%" class="entrevista">
<tbody><tr>
<td></td>
</tr>
</tbody></table>
</div>

```

Pois, a div "divanoAjax" possui conteúdo nulo.

Porém, o que ocorreu por debaixo dos panos, visto que o AJAX entrou em ação aqui!

Observar no HTML que o campo ano possui um evento (onchange) que dispara uma função JS sbmt() passando o idAjax, id este que foi criado na macro FM @texto que definiu o campo ano. E aqui começa a brincadeira:



OK, porém onde está a função de callback (retorno) necessária no AJAX? Neste modelo, diferente do modelo anterior, a função callback foi implementada como uma função anônima (`function ()`) dentro da própria função `ReplaceInnerHTMLFromAjaxResponse()`.

ReplaceInnerHTMLFromAjaxResponse(): Esta função realiza um conversaçāo assíncrona com o servidor via método POST e obtém como resposta todo o HTML gerado pela aplicāo FM. Depois faz duas chamadas a função `MixWithNewPage`, uma passando um array com elementos `<div>` e outra com elementos ``.

MixWithNewPage(): Responsável por separar o joio do trigo, ou seja, dentro da resposta (todo html gerado pela aplicāo FM) obter a `<div>` / `` com o id desejado e injetar o código HTML via `objectid.innerHTML`.

Existe uma outra opção no sentido que ele procura por código JS (procura por string `<script type="text/javascript">` e `</script>`) e o executa via `eval(substring)`. Pois existe o seguinte comentário na função:

// Caso seja necessário acrescentar algum script na pagina, ficou convencionado que // o script deverá ser marcado com `<script type="text/javascript">` e `</script>`

Desta forma, poderíamos, por exemplo, introduzir uma função para ser executada que seja pertinente ao campo em questão.

Exemplo: função alert "passei aqui"

```
[@grupo depende="anoAjax"]
<script type="text/javascript">
alert("passei aqui")
</script>
[if (ano! "") == ""]
  [@mensagem titulo="Alerta" texto="Ano deve ser preenchido." vermelho=true/]
[/if]
[@grupo]
```

```
<script type="text/javascript">
  function displaymessage()
  {
    alert("passei aqui");
  }
  displaymessage();
</script>
```

Qual foi o comportamento encontrado? Quando a aplicação executa a primeira vez, o JS acima foi rodado, ou seja, deve existir um comportamento no SIGA-DOC em executar qualquer JS que encontrar na página, talvez em algum evento de página tal como `onload`.

Isto pode não ser bom em alguns casos, pois se estamos vinculando o JS ao grupo depende de um determinado campo, pode ocorrer algum erro inesperado. Depois ele passou a ser rodado como esperado, ou seja, quando o AJAX é invocado devido a qualquer mudança no campo `ano` que possui o evento `onchange`.

Esta situação ao rodar o JS pela primeira vez pode ser contornada programaticamente, com o código abaixo por exemplo.

```
[@grupo depende="anoAjax"]
[if (flagx) == "1"]
  <script type="text/javascript">
    alert("passei aqui")
  </script>
[if (ano! "") == ""]
  [@mensagem titulo="Alerta" texto="Ano deve ser preenchido." vermelho=true/]
[/if]
[else]
  [-- na primeira vez flagx vem com zero e então é setado para 1 para que o then do if seja
  executado sempre --]
  [@atualizaoculto var="flagx" valor="1/"]
[/if]
[@grupo]
```

Porém, se o usuário definir algum campo que promova a releitura da tela, o problema voltará novamente, pois o `flagx` já estará setado para 1 e esta parte do código (`if then`) será executada novamente. COISAS DA PROGRAMAÇÃO STATELESS.

3.4.3 - Modelo 3 - AJAX SIGA-DOC um caso específico ainda não atendido

Tenho um caso particular em que os modelos anteriores do AJAX do SIGA não atendem, pois ele é uma mistura dos dois casos vistos anteriormente.

Tenho um campo de CPF, onde o usuário pode entrar com o mesmo formatado ou não. Após a entrada de dados, chamo uma macro FM que realiza a formatação do CPF e critica os dígitos verificadores. Se os dígitos estiverem errado, uma mensagem será exibida abaixo informando o erro (OK, o modelo de AJAX anterior atende este caso). Porém, se a crítica estiver OK, desejo que ele substitua o valor digitado pelo usuário (possivelmente não formatado) como o novo CPF formatado. A atualização do campo poderia usar o modelo visto para o destinatário, subscritor e etc, porém as rotinas AJAX estão prontas para este objetivo, e para me atender, teria que customizá-las ou criar novas. Prefiri, então, adotar o segundo modelo de AJAX, visto na seção anterior.

Solução de contorno para este problema. Utilizei uma pequena macro FM (atualizacampo) que carrega um JS que atualiza direto o campo em questão via APIs DOM. Esta macro é muito parecida com a macro vista na seção anterior para atualizar campo oculto.

CPF inválido e a mensagem de erro

CPF: 44444444
Alerta: CPF inconsistente

Ok Visualizar o modelo preenchido

CPF bom, porém não formatado

CPF: 47510846749

Ok Visualizar o modelo preenchido

Retorno com CPF bom formatado

CPF: 475.108.467-49

Ok Visualizar o modelo preenchido

```
[@grupo titulo="Teste Ajax CPF"]
  [@texto var="antigoCPF" titulo="CPF" largura="14" maxcaracteres="14"
reler="ajax" idAjax="cpfAjax"/>
[/@grupo]

[@grupo depende="cpfAjax"]
  [#if (antigoCPF!="")]
    [#assign novoCPF = fmtvldCPF(antigoCPF)/]
    [#if novoCPF == "E1" || novoCPF == "E2"]
      [@mensagem titulo="Alerta" texto="CPF inconsistente" vermelho=true/]
    [#else]
      [#assign campo = "antigoCPF" /]
      [@atualizacampo campo novoCPF/
    [/#if]
  [/#if]
[/@grupo]
```

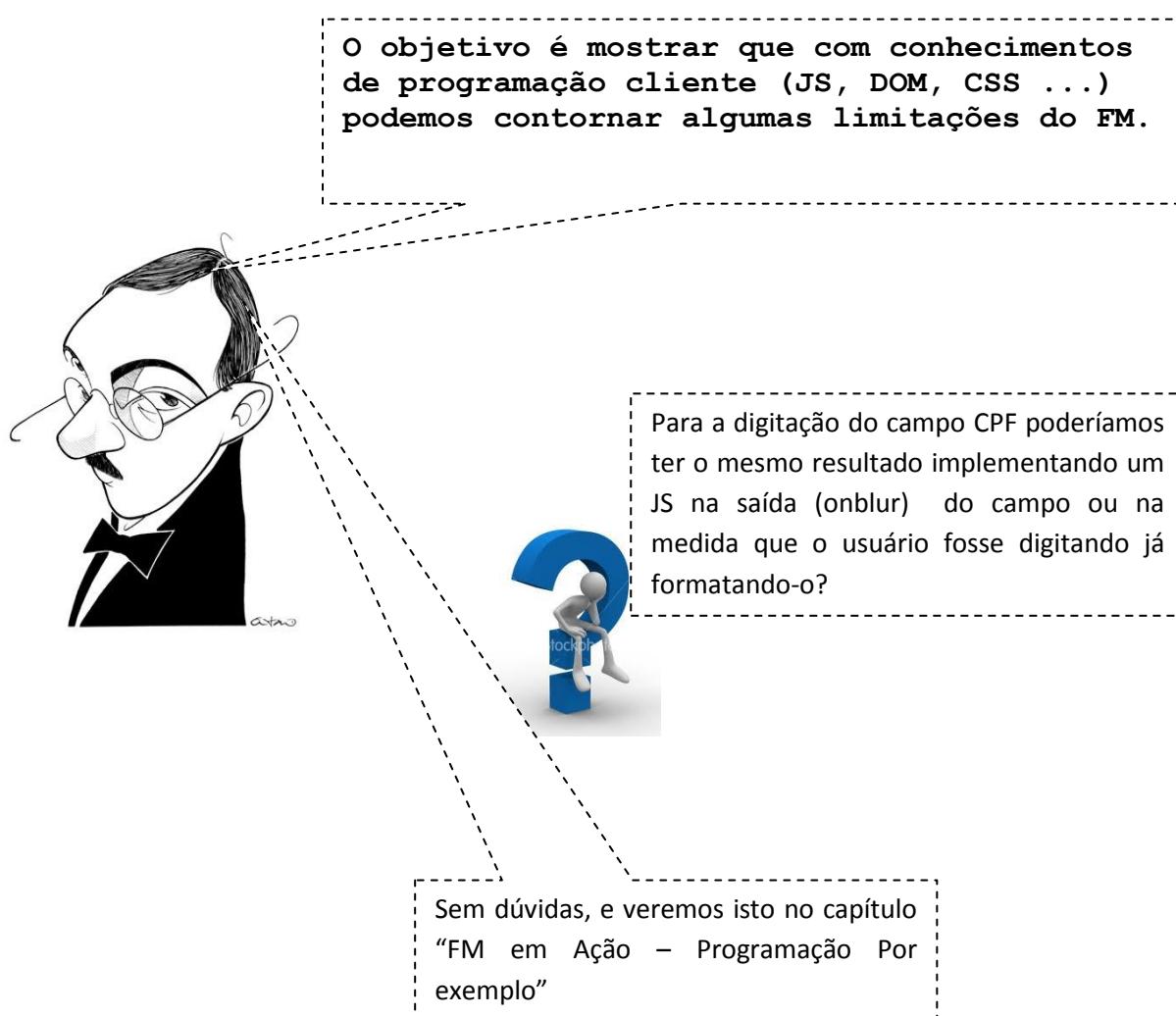
Esta macro inserirá o código JS que fará a chamada DOM

```
<script type="text/javascript">
  document.getElementsByName("antigoCPF")[0].value="${novoCPF}";
</script>
```

Desta forma, temos a utilização do AJAX conforme o segundo modelo, porém com algum requinte do primeiro modelo.

3.4.4 - Modelo 4 - AJAX SIGA-DOC Customizando a customização

No exemplo anterior notamos que a mensagem de erro aparece debaixo do campo CPF. Por quê? O bloco *grupo* depende cria uma `<div>`, e o grupo `<div>` gera uma quebra de linha. E se quisésemos que a mensagem fosse ao lado do campo CPF? Na próxima seção, sobre CSS, veremos como podemos contornar este tipo de problema e econtermos uma solução para contorná-lo.

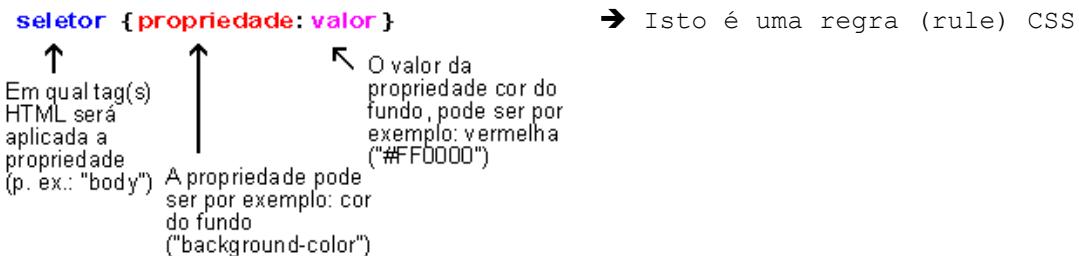


3.5 – CSS

3.5.1 – Introdução

O CSS permite a definição do estilo / lay-out do documento de uma forma mais eficiente e elegante do que se os estilos fossem definidos diretamente no elemento HTML como era na época pré-css. O HTML é para definir conteúdo, foi o propósito inicial, e depois foram adicionando penduricalhos de estilo (cor, fonte, tamanho e etc.). Definir atributos de estilo aos elementos HTML diretamente é coisa do passado.

Sintaxe básica do CSS:



O **seletor** pode ter a seguinte sintaxe: **[taghtml].[classe]**, sendo que um ou outro é opcional, porém pelo menos 1 é obrigatório.

Exemplo: `<table class="entrevista" width="100%">`, podemos ter regras csss para:

- `table` (se aplicará a todas `table`)
- `.entrevista` (se aplicará a qualquer elemento da classe `entrevista`)
- `table.entrevista` (só se aplicará ao elemento `table` da classe `entrevista`)

Comparação em definir estilo no HTML diretamente e via CSS.

HTML Example	CSS Example
<code><body bgcolor="black"></code> body é um statement HTML bgcolor é o atributo black é o valor do atributo	<code>body {background-color:black;}</code> body é um seletor CSS que se refere ao body do html background-color é uma propriedade black é o valor da propriedade

CSS é a abreviatura para **Cascading Style Sheets**. – Folha de Estilos em Cascata CSS é uma linguagem para estilos que define o layout de documentos HTML. Por exemplo, CSS controla fontes, cores, margens, linhas, alturas, larguras, imagens de fundo, posicionamentos e muito mais.

HTML pode ser (in)devidamente usado para definir o layout de websites. Contudo CSS proporciona mais opções e é mais preciso e sofisticado. CSS é suportado por todos os navegadores atuais.

Você pode aplicar CSS a um documento de três maneiras distintas. Os três métodos de aplicação estão exemplificados a seguir. Recomendamos que você foque no terceiro método, ou seja o método externo.

Método 1: In-line (o atributo style)

Uma maneira de aplicar CSS é pelo uso do atributo `style` do HTML. Tomando como base o exemplo mostrado anteriormente a cor vermelha para o fundo da página pode ser aplicada conforme mostrado a seguir:

```
<html>
  <head>
    <title>Exemplo</title>
  </head>
  <body style="background-color: #FF0000;">

    <p>Esta é uma página com fundo vermelho</p>
  </body>
</html>
```

Método 2: Interno (a tag style)

Uma outra maneira de aplicar CSS é pelo uso da tag `<style>` do HTML. Como mostrado a seguir:

```
<html>
  <head>
    <title>Exemplo</title>
    <style type="text/css">
      body {background-color: #FF0000;}
    </style>
  </head>
  <body>
    <p>Esta é uma página com fundo vermelho</p>
  </body>
</html>
```

Método 3: Externo (link para uma folha de estilos)

O método recomendado é o de lincar para uma folha de estilos externa. Usaremos este método nos exemplos deste tutorial.

Uma folha de estilos externa é um simples arquivo de texto com a extensão `.css`. Tal como com qualquer outro tipo de arquivo você pode colocar uma folha de estilos tanto no servidor como no disco rígido.

Vamos supor, por exemplo, que sua folha de estilos tenha sido nomeada de `style.css` e está localizada no diretório `style`. Tal situação está mostrada a seguir:



O "truque" é criar um link no documento HTML (`default.htm`) para a folha de estilos (`style.css`). O link é criado em uma simples linha de código HTML como mostrado a seguir:

```
<link rel="stylesheet" type="text/css" href="style/style.css" />
```

O efeito cascata

Que estilo será aplicado, quando há conflito de estilos especificados (por exemplo: uma regra de estilo determina que os parágrafos serão na cor preta e outra que serão na cor azul) para um mesmo elemento HTML?

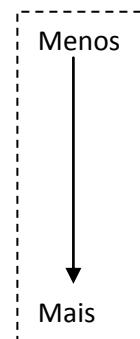
Aqui entra o **efeito cascata**, que nada mais é, do que o estabelecimento de uma prioridade para aplicação da regra de estilo ao elemento. Para determinar a prioridade são considerados diversos fatores, entre eles, o tipo de folha de estilo, o local físico da folha de estilo no seu todo, o local físico da regra de estilo na folha de estilo e a especificidade da regra de estilo.

A prioridade para o efeito cascata em ordem crescente (da MENOR para a MAIOR) é a seguinte:

RESOLUÇÃO DE CONFLITOS

a - Entre folhas

1. folha de estilo padrão do navegador do usuário;
2. folha de estilo do usuário;
3. folha de estilo do desenvolvedor;
 - o estilo externo (importado ou linkado);
 - o estilo incorporado (definido na seção **head** do documento);
 - o estilo inline (dentro de um elemento HTML);
4. declarações do desenvolvedor com !important;
5. declarações do usuário com !important;



Assim, uma declaração de estilo com !important definido pelo usuário prevalece sobre todas as demais, é a de mais alta prioridade. Entre as folhas de estilo definidas pelo desenvolvedor do site, os estilos inline (dentro de um elemento HTML) tem a prioridade mais elevada, isto é, prevalecerá sobre a folha de estilo definida na seção **head**, e, esta prevalecerá sobre uma folha de estilo externa. A prioridade mais baixa é para estilos padrão do navegador.

E este é o motivo de Cascata no nome desta tecnologia.

b - Na mesma folha

Na mesma folha, vale a mais específica. Em caso de empate, vale a que estiver mais próxima do fim da folha.

O poder do JQUERY (que falaremos adiante) integrado com CSS permite coisas fantásticas em uma única linha de comando.

3.5.2 - o SIGA-DOC e o CSS

Temos os seguintes arquivos CSS.

siga.css, **menu.css** e no editar.action temos também algumas regras css.

No código do CPF, listei um trecho do HTML gerado:

```
<table width="100%" class="entrevista">
<tbody><tr class="header">
```

```

<td>Teste Ajax CPF</td>
</tr>
<tr>
<td><input type="hidden" value="antigoCPF" name="vars">
<span style=",">CPF:</span>
<input type="text" maxlength="14" size="14" onchange="javascript: sbmt('cpfAjax');" value="" name="antigoCPF">
</td>
</tr>
</tbody>
</table>

```

No firebug coloquei em foco o campo CPF (input) e obtive as seguintes regras. As que estão riscadas é porque não estão valendo, pois perderam a prioridade para as três primeiras, que estão em vigor.

O efeito cascata em ação e a resolução de conflitos.

Regra	Referência
body, p, td, input , button, select { font-size: 11px;}	siga.css (** linha 109)
body, p, td, h1, h2, h3, input , button, select, newsarticle { font-family: Verdana,Arial,Helvetica,sans-serif;}	siga.css (**linha 105)
Herdado de td	
TABLE.entrevista TR TD { border-spacing: 0; }	siga.css (** linha 525)
body, p, td, input , button, select { font-size: 11px; }	siga.css (linha 109)
body, p, td, h1, h2, h3, input , button, select, newsarticle { font-family: Verdana,Arial,Helvetica,sans-serif; }	siga.css (linha 105)
Herdado de tr	
TABLE.entrevista TR { border-spacing: 0; +}	siga.css (linha 518)
Herdado de table.entrevista	
TABLE.entrevista { border-spacing: 0; +}	siga.css (linha 511)
TABLE { font-size: 10pt; +}	siga.css (linha 300)
... Continua	... continua

Conteúdo das regras em vigor para o campo CPF (no arquivo `siga.css`):

```
**Linha 105
body,p,td,h1,h2,h3,input,button,select,newssarticle {
    font-family: Verdana, Arial, Helvetica, sans-serif;
}

**Linha 109
body,p,td,input,button,select {
    font-size: 11px;
}

**Linha 525
TABLE.entrevista TR TD {
    padding: 0px;
    margin: 0px;
    border-width: 0px;
    border-spacing: 0px;
}
```

3.5.3 - Revisitando o código do CPF e customizando-o para colocar a mensagem ao lado do campo

Na seção sobre o AJAX deixamos o modelo 4 ([Modelo 4 - AJAX SIGA-DOC Customizando a customização](#)) em aberto, pois precisaríamos de conhecimentos de CSS para resolver o problema. Desta forma, veremos como:

a - Se o CPF estiver em branco, não tem problema. Não é obrigatório;

Teste Ajax CPF

CPF:

Ok Visualizar o modelo preenchido 

b - O usuário digita um CPF ruim, e a mensagem aparece sem renderização da tela

Teste Ajax CPF

CPF: 6767888888 <-- CPF inconsistente

Ok Visualizar o modelo preenchido 

c - O usuário fornece um CPF bom, sem formatação.

Teste Ajax CPF

CPF: 47510846749

Ok Visualizar o modelo preenchido 

d - O sistema avalia e retorna o CPF bom formatado.

Teste Ajax CPF

CPF: 475.108.467-49

Ok Visualizar o modelo preenchido 

Código principal

```
[@grupo titulo="Teste Ajax CPF"]
[@texto var="antigoCPF" titulo="CPF" largura="14" maxcaracteres="14"
reler="ajax" idAjax="cpfAjax"/>
[@mensagembyid idspan="xpto12" texto=" <== CPF inconsistente" vermelho=true]
[!--
esta macro cria um campo ao lado do CPF. Observe que a mensagem já vem com
conteúdo de erro. O programa ligará ou desligará esta mensagem de acordo com a
necessidade
--]
[/@grupo]

[@grupo depende="cpfAjax"]

[#if (antigoCPF!="") != ""]
[#assign novoCPF = fmtvldCPF(antigoCPF)/]
[#if novoCPF == "E1" || novoCPF == "E2"]
[@ligamsg idspan="xpto12"]
[#else]
[#assign campo = "antigoCPF" /]
[@atualizacampo campo novoCPF/]
[@desligamsg idspan="xpto12"]
[/#if]
[#else]
[@desligamsg idspan="xpto12"]  [!-- campo CPF em branco --]
[/#if]
[/@grupo]
```

As macros em azul foram desenvolvidas para solucionar o problema.

Por que criei a macro @mensagembyid se já existe a macro @mensagem? O problema é que a macro @mensagem não cria um id, dificultando o acesso ao elemento via DOM. Observe que a macro também manipula o CSS, colocando a mensagem em vermelho.

```
[#macro mensagembyid idspan texto vermelho]
[!--
Aplicação: Cria um campo de mensagem com id para que se possa mudar o seu
atributo dinamicamente
Autor: Ruben
Data: 13/05/2012
Descrição: Recebe o id e o texto da mensagem para que se possa alterar o
style.display para "none" ou "inline"
--]
<span id="${idspan}" style="#if vermelho]color:#ff0000[/#if]">
${texto!""}</span>
[/macro]
```

Esta macro faz o campo de mensagem aparecer. Observe o comando:
document.getElementById("\${idspan}").style.display="inline";
+-----DOM-----+-----CSS-----+
É a combinação de DOM com CSS manipulado por um pequeno código JS.

```
[#macro ligamsg idspan]
[!--
Aplicação: Liga um campo mensagem para exibir
Autor: Ruben
Data: 13/05/2012
Descrição: Recebe o id do campo
--]
```

```
<script type="text/javascript">
document.getElementById("${idspan}").style.display="inline";
</script>
[/#macro]
```

Esta macro faz o campo de mensagem desaparecer. Os comentários da macro anterior valem aqui.

```
[#macro desligamsg idspan]
[!--
Aplicação: Desliga um campo mensagem para exibir
Autor: Ruben
Data: 13/05/2012
Descrição: Recebe o id do campo
--]
<script type="text/javascript">
document.getElementById("${idspan}").style.display="none";
</script>
[/#macro]
```

3.6 – Applet

É, o SIGA-DOC também tem Applets. Applets são pequenas aplicações JAVA totalmente funcionais, porém roda no cliente e não no servidor (ela é carregada do servidor). As applets geralmente são usadas quando se exige uma segurança no cliente, visto que a aplicação é fechada.

A única applet encontrada foi para assinar eletronicamente o documento.

```
<applet id="oApplet"
codebase="${request_scheme}://${request_serverName}:${request_localPort}/siga-ext-
assinatura/applet"
code="br/com/esec/signapplet/DefaultSignApplet.class"
```

Esta applet está sendo substituída por uma aplicação jQuery.

3.7 – Tendências

3.7.1 – HTML 4.01, XHTML 1.1 e 2.0, HTML 5.0, XML, DHTML?



O SIGA-DOC utiliza o HTML 4.01.

HTML (Presente)

HTML é a língua mãe dos navegadores (browsers). É antiga, foi criada com um objetivo e tem sofrido alterações ao longo do tempo. Como é normal na TI, tudo que é antigo é questionado e são propostos novos padrões. **Enquanto os outros padrões não pegam, inchamos o que já existe para compensar a defasagem, em consequência, começamos a pesar, gerar incompatibilidades entre os browsers e assim por diante.**

HTML é uma linguagem que possibilita apresentar conteúdo na Internet. Aquilo que você vê quando abre uma página na Internet é a interpretação que seu navegador faz do HTML. Para visualizar o código HTML de uma página use o menu "View" (Ver) no topo do seu navegador e escolha a opção "Source" (Código fonte).

Cabeçalho – Empresa ACNE

Todo **documento** é composto por **conteúdo e lay-out (forma)**. Para ver um documento sem lay-out, abra-o no notepad, é puro conteúdo.

Rodapé - xxxx

As críticas ao HTML.

- Ele só deve se importar com o conteúdo e não com o lay-out (estilo), daí nasceu o CSS retirando parte do estilo do HTML;
- O conteúdo que ele apresenta não possui semântica. Não sabemos o que é produto, o que é preço e etc. Daí nasceu o XML para promover semântica ao conteúdo.

Ora se retirarem o conteúdo e o lay-out sobrará o que do HTML? Rsrssrsrsrs.

E o HTML 5.0? É uma resposta à observação de que o HTML e o XHTML, de uso comum na World Wide Web, é uma mistura de características introduzidas por várias especificações, juntamente com aquelas introduzidas por software, tais como os navegadores. Entre outras novidades, traz melhorias em termos de vídeo e som, sendo nativos, ou seja, não há necessidade de codecs, pois estão embutidos no browser. Traz melhorias no processamento de imagens, com efeitos de transição, a mesma técnica utilizada pelo PowerPoint. Ela também será semântica. Alguns dizem que o HTML matará o XHTML.

XML (Futuro)

É uma especificação técnica desenvolvida pela W3C (World Wide Web Consortium – entidade responsável pela definição da área gráfica da internet), para superar as limitações do HTML, que é o padrão das páginas da Web. A linguagem XML é definida como o formato universal para dados estruturados na Web. Esses dados consistem em tabelas, desenhos, parâmetros de configuração, etc. A linguagem então trata de definir regras que permitem escrever esses documentos de forma que sejam adequadamente visíveis ao computador. Para intercâmbio de dados entre computadores existem outras alternativas como o **JSON**, que veremos depois, que é muito mais leve, porque é menos verborrágico.

HTML (Liberal e gorda) X XML (Rígida e verborrágica)

A principal semelhança entre as duas é o fato de utilizarem tags (palavras-chaves e parâmetros). Em ambas as linguagens, cada tag consiste em duas partes, uma que inicia e outra que fecha o comando. No entanto, em muitos casos, se uma tag é aberta no HTML e não é fechada, a página é exibida mesmo assim.

Já no XML, se houver qualquer erro desse tipo, a aplicação simplesmente para. Percebe-se com esse exemplo, que o HTML é uma linguagem mais tolerante, enquanto o XML é altamente rígido.

Isso pode até parecer uma desvantagem, mas se for, é compensada pela extensibilidade do XML.

Para um melhor entendimento, veja o seguinte fato: no HTML, a tag `<p> </p>` indica o início e o fim de um parágrafo e pode conter qualquer coisa.

No XML, as tags são usadas para definir blocos de dados. O que isso quer dizer? O programador pode definir tags para `<peso>`, `<pessoa>`, `<nome>`, `<endereço>`, `<produto>`, `<carro>`, enfim, o que ele quiser que represente. Desta forma, estamos fornecendo semântica ao conteúdo. Por essa característica, o XML é até considerado por muitos uma linguagem capaz de gerar outras linguagens (metalinguagem), visto que quem define os comandos e suas funções é o programador. A praticidade é tanta que torna-se possível um usuário criar uma coleção própria de tags e aplicá-las nas páginas e documentos que desejar.

XHTML

Entre 8 e 80 deve existir um meio termo. Entre HTML, com problemas, e XML que ainda não vingou, deve existir um meio termo, que é ... o XHTML.

O XHTML está a disposição de projetistas e desenvolvedores web. XHTML é um código consistente que dispensa uso de "truques" e "hacks" para contornar "bugs". Editar um código XHTML existente é uma tarefa bem simples por se tratar de uma escrita limpa e consistente. O tempo de carregamento de uma página XHTML é mais rápido, pois os browsers interpretam uma página limpa sem ter que interpretar e decidir sobre renderização de erros de código. Uma página XHTML é mais acessível aos browsers permitindo a interoperabilidade e a portabilidade dos documentos web em diversos dispositivos existentes no mercado. Se você quer que seu aplicativos rode em celulares, TV, computadores e etc. o XHTML é a melhor pedida. Uma página XHTML é totalmente compatível com todas as aplicações HTML, sejam antigas e já ultrapassadas.

Tipos de XHTML existentes:

- XHTML Transitional - transição entre o HTML e o XHTML Strict. Permite a utilização de todos os elementos do HTML 4.01, possibilitando a compatibilidade com navegadores antigos.
- XHTML Strict - não inclui os elementos de formatação do HTML 4.01. Base para a XHTML 1.1 e XHTML 2.0.
- XHTML Frameset - deve ser usada quando se quer trabalhar com frames. Inclui todos os elementos da HTML 4.01.

DHTML

Dynamic HTML não é uma linguagem e nem um padrão WEB.

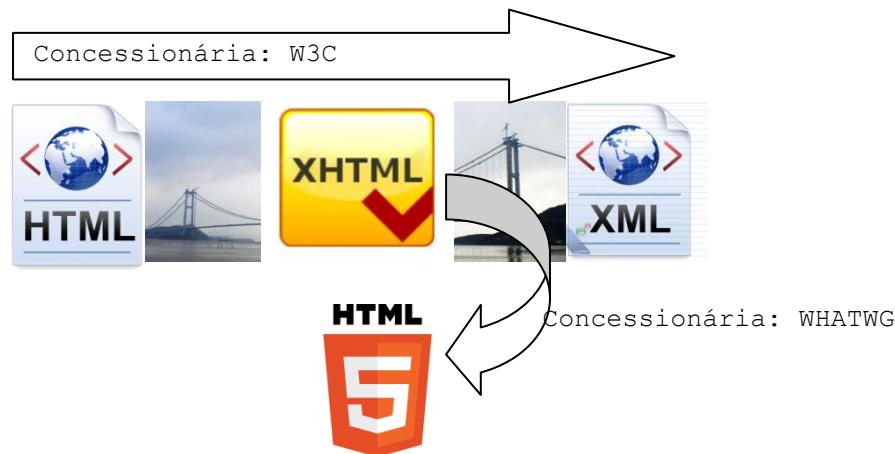
DHTML é a combinação de tecnologias para produzir efeitos dinâmicos nas páginas. Segundo o W3C:

"Dynamic HTML is a term used by some vendors to describe the combination of HTML, style sheets and scripts that allows documents to be animated."

Para muitos DHTML = HTML + JavaScript + DOM + CSS.

HTML5 e o SIGA-DOC

Como dito anteriormente, a ponte entre o presente (HTML 4.01) e o futuro (XML) é o XHTML. Bom, isto dito pela concessionária W3C, que vem há longo tempo pavimentando esta estrada.



Porém o pessoal do WHATWG (Web Hypertext Application Technology Working Group - grupo formado por desenvolvedores de diversas empresas, como Opera, Mozilla e Apple) andava muito insatisfeito com a lentidão da obra, promoveu um desvio e lançou o HTML5, que, segundo eles, vigorará até 2020, com atualizações frequentes. Segundo o W3School, o HTML5 incluirá o XHTML 5.0, quando esta especificação estiver pronta.

Com as novas features do HTML5, teremos:



Aliado ao jQuery, teremos:



É impressionante como muitas features que o SIGA-DOC (SDc) implementa, e o próprio jQuery ((veremos na sessão seguinte), já estão inseridas no HTML5, isto porque o HTML 4 ficou muito tempo estacionado, dando margem ao aparecimento de outros padrões, como o XHTML, XML e fazendo com que os desenvolvedores de site implementassem suas próprias features, como não deixou de acontecer com o SDc. Abaixo, farei uma breve explanação destas novidades e de como podemos tirar proveito para o SIGA-DOC:

3.7.1.1 - Formulários

3.7.1.1.1 - Novos campos, provendo mais semântica ao input / type

Estes novos campos, de negócio, podem substituir as macros FM.

Telefone - `<input name=tel type=tel>` - Utilizado para entrada de dados de telefones

Tel:

URL - `<input name=url type=url>` - Utilizado para entrada de variadas URLs. Por padrão o browser irá inserir o http:// como protocolo padrão

URL:

E-mail - `<input name=email type=email>` - Caso opte por validação, automaticamente o browser valida se o valor for um e-mail válido

E-mail:

Data e hora - `<input name=horario type=datetime>` - Utilizado para agendamento de eventos, reuniões, etc.

Data/hora:

Número - `<input name=numero type=number>` - Com os atributos "min" e "Max" é possível entrar com um intervalo de valores possíveis e com o "step" é possível definir o valor para cada incremento

Número:

Tempo - `<input name=tempo type=time>`

Tempo:

Semana - `<input name=semana type=week>`

Semana:

Slider - `<input name=slider type=range min=2 max=30 step=2 >`



3.7.1.1.2 - Validação

Não sera mais necessário a utilização de muito javascript para validação de formulários, pois o HTML5 já provê uma série de melhorias:

Obrigatório

O atributo "required", que dentro de um elemento input torna o campo obrigatório e transfere para o browser a tarefa da validação do campo.

`<p><label>Tel: <input name=tel type=tel required></label></p>`

Validação por expressão regular

Além de deixar o campo apenas como obrigatório, é possível colocar um padrão de entrada para aquele determinado campo, que será validado pelo browser. Por exemplo, em um campo em que só possa entrar valores numéricos com 3 dígitos:

`<input pattern="[0-9]{3}" name=dígito required title="Validação apenas para 3 dígitos"/>`

Observe que o atributo pattern aceita expressões regulares.

Validação por range de valores

No tipo de campo "numérico", é possível fazer uma validação dos valores possíveis de entrada utilizando os atributos min e max.

```
<input type="number" min=2 max=10 />
```

Auto completar

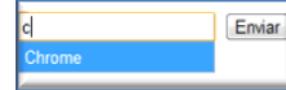
Este autocomplete é do tipo "LEMBRAR" o que já foi digitado no campo, o que facilita a digitação. É diferente do autocomplete do jQuery

```
<form action="demo_form.asp" method="get" autocomplete="on">
  First name:<input type="text" name="fname" /><br />
  E-mail: <input type="email" name="email" /><br />
  <input type="submit" />
</form>
```

Data List

Idêntico ao autocomplete do jQuery.

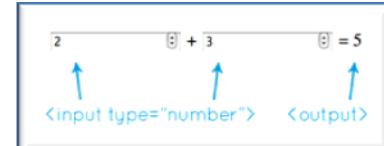
```
<form action="demo_form.asp" method="get">
<input list="browsers" name="browser" />
<datalist id="browsers">
  <option value="Internet Explorer">
  <option value="Firefox">
  <option value="Chrome">
  <option value="Opera">
  <option value="Safari">
</datalist>
<input type="submit" />
</form>
```



Output

Tempos o elemento input que exige uma entrada de dados. E se quisemos exibir uma campo de saída, como por exemplo, o resultado de uma conta?

```
<form onsubmit="return false" oninput="o.value =parseInt(a.value) +
  parseInt(b.value)">
  <input name="a" type="number" step="any"> +
  <input name="b" type="number" step="any"> =
  <output name="o"></output>
</form>
```



keyGen

The <keygen> tag specifies a key-pair generator field used for forms.

When the form is submitted, the private key is stored locally, and the public key is sent to the server.

```
<!DOCTYPE html>
<html>
<body>
<form action="demo_keygen.asp" method="get">
Username: <input type="text" name="usr_name">
Encryption: <keygen name="security" />
<input type="submit" />
</form>
</body>
</html>
```

Username: Encryption: 2048 (Nível alto)

Entrei com Ruben, e gerou

Input was received as:

usr_name=ruben&security=MIICQDCCASgwggEiMA0GCSqGSIb3DQEBAQUAA

/>

This page was returned to you from the server. The server has processed your input and returned this answer.

3.7.1.2 - Multimídia

O HTML5 fornece acesso nativo a video / audio, sem a necessidade de instalação de nenhum codec especial. Poderíamos no SIGA, dependendo do formulário, como o da reunião (MRU) por exemplo, anexar o vídeo ou audio da mesma.



Vídeos suportados:

- MP4 = MPEG 4 files with H264 video codec and AAC audio codec
- WebM = WebM files with VP8 video codec and Vorbis audio codec
- Ogg = Ogg files with Theora video codec and Vorbis audio codec

```
<video width="320" height="240" controls="controls">
  <source src="movie.mp4" type="video/mp4" />
  <source src="movie.ogg" type="video/ogg" />
  Your browser does not support the video tag.
</video>
```

Também teremos o Vídeo / DOM, ou seja, o elemento HTML5 <video> também possui: métodos, propriedades e eventos que poderão ser manipulados por JS / jQuery.

Audios suportados: MP3, WAV e OGG

```
<audio controls="controls">
  <source src="song.ogg" type="audio/ogg" />
  <source src="song.mp3" type="audio/mpeg" />
  Your browser does not support the audio element.
</audio>
```

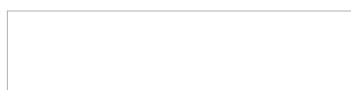
3.7.1.3 - Drag and Drop

Qualquer elemento em HTML5 poderá ser arrastado e soltado dentro da página.

```
<div id="div1" ondrop="drop(event)"  
ondragover="allowDrop(event)"></div>  

```

Drag the W3Schools image into the rectangle:



w3schools.com



Drag the W3Schools image into the rectangle:



JS que implementa os
eventos do Drag e Drop

```
<script type="text/javascript">  
function allowDrop(ev)  
{  
ev.preventDefault();  
}  
function drag(ev)  
{  
ev.dataTransfer.setData("Text",ev.target.id);  
}  
function drop(ev)  
{  
ev.preventDefault();  
var data=ev.dataTransfer.getData("Text");  
ev.target.appendChild(document.getElementById(data))  
};  
</script>
```

3.7.1.4 - Gráficos

SVG: Scalable Vector Graphics. São gráficos vetoriais, que diferente do bitmap (ou raster), pode ser ampliado infinitamente sem perda de definição (sem efeito serrilhamento).

```
<!DOCTYPE html>  
<html>  
<body>  
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="190">  
  <polygon points="100,10 40,180 190,60 10,60 160,180"  
    style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;" />  
</svg>  
</body>  
</html>
```

Producirá:



CANVAS: O elemento HTML5 `<canvas>` é usado para desenhar gráficos, on fly, via script. O desenho é renderizado pixel a pixel, sendo o tipo bitmap, raster.

```
<!DOCTYPE html>  
<html>  
<body>  
<canvas id="myCanvas" width="200" height="100" style="border:1px solid #c3c3c3;">
```

Your browser does not support the canvas element.

```
</canvas>
<script type="text/javascript">
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");
ctx.fillStyle="#FF0000";
ctx.fillRect(0,0,150,75);
</script>
</body>
</html>
```

Producirá:



Canvas	SVG
<ul style="list-style-type: none">• Resolution dependent• No support for event handlers• Poor text rendering capabilities• You can save the resulting image as .png or .jpg• Best suited for graphic-intensive games where many objects are redrawn frequently	<ul style="list-style-type: none">• Resolution independent• Support for event handlers• Best suited for applications with large rendering areas (Google Maps)• Slow rendering if complex (anything that uses the DOM a lot will be slow)• Not suited for game applicat

3.7.1.5 – Geo-Localização

É utilizado para obter a posição geográfica do usuário. É sensacional, pois cliquei no botão baixo e ele me retornou a posição correta de onde estava.

HTML5 Geolocation is used to locate a user's position [Try It](#)

HTML5 Geolocation is used to locate a user's position [Try It](#)



```
<!DOCTYPE html>
<html>
<body>
<p id="demo">Click the button to get your coordinates:</p>
<button onclick="getLocation()">Try It</button>
<script>
var x=document.getElementById("demo");
function getLocation()
{
if (navigator.geolocation)
{
navigator.geolocation.getCurrentPosition(showPosition);
}
else{x.innerHTML="Geolocation is not supported by this browser.";}
}
```

```
function showPosition(position)
{
  x.innerHTML="Latitude: " + position.coords.latitude +
  "<br />Longitude: " + position.coords.longitude;
}
</script>
</body>
</html>
```

Produzirá:

Latitude: -22.9090236
Longitude: -43.1081306

 Try It

3.7.1.6 - WEB-Storage (WS)

Com HTML5, as páginas WEB podem ser armazenadas localmente, dentro do browser do usuário. Isto poderia ser feito com cookies, porém WS é mais rápido e seguro. Pode-se armazenar grandes quantidades de dados por:

- localStorage - armazena dados sem data de expiração
- sessionStorage - armazena dados por uma sessão

Os dados são armazenados em pares de key / value, e somente a página que gravou os dados pode acessá-los. Exemplos:

Neste exemplo, a cada vez que o usuário aperta o botão ele é incrementado, mesmo que o usuário encerre a sessão, feche o browser ou desligue o computador.

```
<!DOCTYPE html>
<html>
<head>
<script>
function clickCounter()
{
if(typeof(Storage)!=="undefined")
{
  if (localStorage.clickcount)
  {

localStorage.clickcount=Number(localStorage.clickcount)+1;
  }
else
{
  localStorage.clickcount=1;
}

document.getElementById("result").innerHTML=
"You have clicked the button " +
localStorage.clickcount + " time(s).";
}
else
{

document.getElementById("result").innerHTML=
"Sorry, your browser does not support web
storage...";
}
}
</script>
</head>
<body>
<p><button onclick="clickCounter()" type="button">Click me!</button></p>
<div id="result"></div>
<p>Click the button to see the counter
increase.</p>
<p>Close the browser tab (or window), and
try again, and the counter will continue to
count (is not reset).</p>
</body>
</html>
```

Neste exemplo, o incremento só ocorre enquanto for a mesma sessão. Depois que a sessão é encerrada, tudo volta ao normal.

```
<!DOCTYPE html>
<html>
<head>
<script>
function clickCounter()
{
if(typeof(Storage)!=="undefined")
{
  if (sessionStorage.clickcount)
  {

sessionStorage.clickcount=Number(sessionStorage.clickcount)+1;
  }
else
{
  sessionStorage.clickcount=1;
}

document.getElementById("result").innerHTML=
"You have clicked the button " +
sessionStorage.clickcount + " time(s) in
this session.";
}
else
{

document.getElementById("result").innerHTML=
"Sorry, your browser does not support web
storage...";
}
}
</script>
</head>
<body>
<p><button onclick="clickCounter()" type="button">Click me!</button></p>
<div id="result"></div>
<p>Click the button to see the counter
increase.</p>
<p>Close the browser tab (or window), and
try again, and the counter is reset.</p>
</body>
</html>
```

You have clicked the button 2 time(s).

Click the button to see the counter increase.

Close the browser tab (or window), and try again, and the counter will continue to count (is not reset).

You have clicked the button 1 time(s) in this session.

Click the button to see the counter increase.

Close the browser tab (or window), and try again, and the counter is reset.

3.7.1.7 - Application Cache (appcache)

É possível navegar num site com a Internet fora do ar? Cenário: visito um site pelo notebook, vou embora para casa e no caminho, sem internet, continuo navegando no mesmo.

É possível com a declaração

```
<!DOCTYPE HTML>
<html manifest="demo.appcache">
...
</html>
```

E o arquivo de manifesto (demo.appcache)

```
CACHE MANIFEST
# 2012-02-21 v1.0.0
/theme.css
/logo.gif
/main.js

NETWORK:
login.asp

FALLBACK:
/html5/ /offline.html
```

Exemplo de uma aplicação sem o respectivo arquivo de manifesto:

```
<!DOCTYPE html>
<html manifest="demo_html.appcache">
<body>
<script type="text/javascript" src="demo_time.js">
</script>
<p id="timePara"><button onclick="getDateTime()">Get Date and Time</button></p>
<p></p>
<p>Try opening <a href="tryhtml5_html_manifest.htm" target="_blank">this page</a>, then go offline, and reload the page. The script and the image should still work.</p>
</body>
</html>
```

O que produzirá:

Mon Jul 09 2012 18:35:33 GMT-0300 (Hora oficial do Brasil)



Try opening [this page](#), then go offline, and reload the page. The script and the image should still work.

Como indica a mensagem, mesmo sem internet, você continua tendo acesso a imagem, ao texto e ao JavaScript.

3.7.1.8 - WEB Workers (WW)

Nome pomposo para uma coisa simples e genial. Que tal uma rotina JS rodando em background, sem parar, sem interferir na página (cliques, entrada de dados e etc.) e na performance da mesma.

Por quê? Quando uma rotina JS está rodando na página, esta torna-se inacessível (unresponsive) até a rotina acaba. Porém, com WW isto não acontece.

Quando um WW é criado ele fica ouvindo as mensagens, até que alguém decida desligá-lo.



Cenários: o WW pode ficar salvando o documento do SIGA-DOC de tempos em tempos, pode ficar checando se a sessão com o servidor está OK, realizando cálculos e etc.

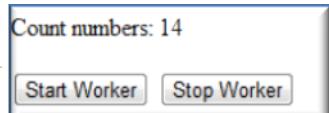
No exemplo abaixo uma rotina JS fica em background rodando, quando o usuário aperta start, enquanto a página fica disponível ao usuário.

Existe também a opção dele desligar o WW.

Existe uma aplicação de mapa do Google, que enquanto você está arrastando a imagem ele continua trabalhando, formatando-o.

```
<!DOCTYPE html>
<html>
<body>
<p>Count numbers: <output id="result"></output></p>
<button onclick="startWorker()">Start Worker</button>
<button onclick="stopWorker()">Stop Worker</button>
<br /><br />
<script>
var w;
function startWorker()
{
if(typeof(Worker)!=="undefined")
{
  if(typeof(w)=="undefined")
  {
    w=new Worker("demo_workers.js");
  }
  w.onmessage = function (event) {
    document.getElementById("result").innerHTML=event.data;
  };
}
else
{
  document.getElementById("result").innerHTML="Sorry, your browser does not
support Web Workers...";
}
}
function stopWorker()
{
w.terminate();
}
</script>
</body>
</html>
```

Producirá:



3.7.1.9 - Server Sent Events (SSE)

Você já imaginou se comunicar com o servidor sem enviar a página ou utilizar AJAX (pelo menos explicitamente)? Agora podemos, com Eventos enviados pelo Servidor.

No exemplo abaixo, a página fica obtendo a Hora do servidor e exibindo na página.

```
<!DOCTYPE html>
<html>
<body>
<h1>Getting server updates</h1>
<div id="result"></div>

<script>
if(typeof(EventSource) !=="undefined")
{
  var source=new EventSource ("demo_sse.php");
  source.onmessage=function(event)
  {
    document.getElementById("result").innerHTML+=event.data + "<br />";
  };
}
else
{
  document.getElementById("result").innerHTML="Sorry, your browser does not
support server-sent events...";
}
</script>

</body>
</html>
```

demo sse.php

```
<?php
header('Content-Type: text/event-
stream');
header('Cache-Control: no-cache');
$time = date('r');
echo "data: The server time is:
{$time}\n\n";
flush();
```

Producirá:

Getting server updates

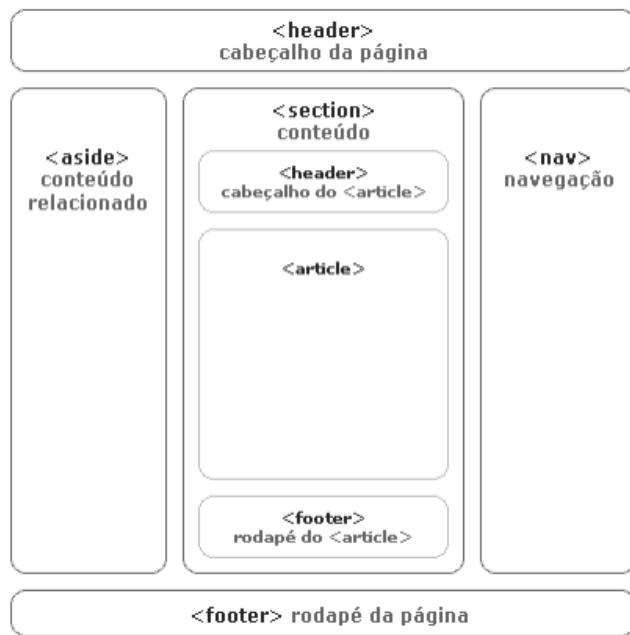
```
The server time is: Mon, 09 Jul 2012 18:36:05 -0400
The server time is: Mon, 09 Jul 2012 18:36:08 -0400
The server time is: Mon, 09 Jul 2012 18:36:11 -0400
```

3.7.1.10 - Novos elementos (TAGs)

3.7.1.10.1 - De estrutura

<header> - cabeçalho da página ou de uma seção (não confundir com a tag **<head>**);
<section> - cada seção do conteúdo;
<article> - um item do conteúdo dentro da página ou da seção;
<footer> - o rodapé da página ou de uma seção;
<nav> - o conjunto de links que formam a navegação, seja o menu principal do site ou links relacionados ao conteúdo da página;
<aside> - conteúdo relacionado ao artigo (como arquivos e posts relacionados em um blog, por exemplo)

A estrutura abaixo, de 3 colunas, pode ser criada sem tabelas e sem o uso de float left, ou right.



3.7.1.10.2 – Relação completa das <tag>. Novos (azul) e fora de uso (vermelho)

HTML5

Tag	Description
<!--...-->	Defines a comment
<!DOCTYPE>	Defines the document type
<a>	Defines a hyperlink
<abbr>	Defines an abbreviation
<acronym>	Not supported in HTML5
<address>	Defines contact information for the author/owner of a document/article
<applet>	Not supported in HTML5
<area>	Defines an area inside an image-map
<article> New	Defines an article
<aside> New	Defines content aside from the page content
<audio> New	Defines sound content
	Defines bold text
<base>	Specifies the base URL/target for all relative URLs in a document
<basefont>	Not supported in HTML5
<bdi> New	Isolates a part of text that might be formatted in a different direction from other text outside it
<bdo>	Overrides the current text direction
<big>	Not supported in HTML5
<blockquote>	Defines a section that is quoted from another source
<body>	Defines the document's body
 	Defines a single line break
<button>	Defines a clickable button
<canvas> New	Used to draw graphics, on the fly, via scripting (usually JavaScript)
<caption>	Defines a table caption
<center>	Not supported in HTML5
<cite>	Defines the title of a work
<code>	Defines a piece of computer code
<col>	Specifies column properties for each column within a <colgroup> element
<colgroup>	Specifies a group of one or more columns in a table for formatting
<command> New	Defines a command button that a user can invoke

<u><datalist></u> New	Specifies a list of pre-defined options for input controls
<u><dd></u>	Defines a description of an item in a definition list
<u></u>	Defines a text that has been deleted from a document
<u><details></u> New	Defines additional details that the user can view or hide
<u><dfn></u>	Defines a definition term
<u><dir></u>	Not supported in HTML5
<u><div></u>	Defines a section in a document
<u><dl></u>	Defines a definition list
<u><dt></u>	Defines a term (an item) in a definition list
<u></u>	Defines emphasized text
<u><embed></u> New	Defines a container for an external application or interactive content (a plug-in)
<u><fieldset></u>	Groups related elements in a form
<u><figcaption></u> New	Defines a caption for a <figure> element
<u><figure></u> New	Specifies self-contained content
<u></u>	Not supported in HTML5
<u><footer></u> New	Defines a footer for a document or section
<u><form></u>	Defines an HTML form for user input
<u><frame></u>	Not supported in HTML5
<u><frameset></u>	Not supported in HTML5
<u><h1></u> to <u><h6></u>	Defines HTML headings
<u><head></u>	Defines information about the document
<u><header></u> New	Defines a header for a document or section
<u><hgroup></u> New	Groups heading (<h1> to <h6>) elements
<u><hr></u>	Defines a thematic change in the content
<u><html></u>	Defines the root of an HTML document
<u><i></u>	Defines a part of text in an alternate voice or mood
<u><iframe></u>	Defines an inline frame
<u></u>	Defines an image
<u><input></u>	Defines an input control
<u><ins></u>	Defines a text that has been inserted into a document
<u><keygen></u>	Defines a key-pair generator field (for forms)
<u><kbd></u>	Defines keyboard input
<u><label></u>	Defines a label for an input element
<u><legend></u>	Defines a caption for a <fieldset>, <figure>, or <details> element
<u></u>	Defines a list item
<u><link></u>	Defines the relationship between a document and an external resource (most used to link to style sheets)
<u><map></u>	Defines a client-side image-map
<u><mark></u> New	Defines marked/highlighted text
<u><menu></u>	Defines a list/menu of commands
<u><meta></u>	Defines metadata about an HTML document
<u><meter></u> New	Defines a scalar measurement within a known range (a gauge)
<u><nav></u> New	Defines navigation links
<u><noframes></u>	Not supported in HTML5
<u><noscript></u>	Defines an alternate content for users that do not support client-side scripts
<u><object></u>	Defines an embedded object
<u></u>	Defines an ordered list
<u><optgroup></u>	Defines a group of related options in a drop-down list
<u><option></u>	Defines an option in a drop-down list
<u><output></u> New	Defines the result of a calculation
<u><p></u>	Defines a paragraph
<u><param></u>	Defines a parameter for an object
<u><pre></u>	Defines preformatted text
<u><progress></u> New	Represents the progress of a task
<u><q></u>	Defines a short quotation
<u><rp></u> New	Defines what to show in browsers that do not support ruby annotations
<u><rt></u> New	Defines an explanation/pronunciation of characters (for East Asian typography)

<u><ruby>New</u>	Defines a ruby annotation (for East Asian typography)
<u><s></u>	Defines text that is no longer correct
<u><samp></u>	Defines sample output from a computer program
<u><script></u>	Defines a client-side script
<u><section>New</u>	Defines a section in a document
<u><select></u>	Defines a drop-down list
<u><small></u>	Defines smaller text
<u><source>New</u>	Defines multiple media resources for media elements (<video> and <audio>)
<u></u>	Defines a section in a document
<u><strike></u>	Not supported in HTML5
<u></u>	Defines important text
<u><style></u>	Defines style information for a document
<u><sub></u>	Defines subscripted text
<u><summary>New</u>	Defines a visible heading for a <details> element
<u><sup></u>	Defines superscripted text
<u><table></u>	Defines a table
<u><tbody></u>	Groups the body content in a table
<u><td></u>	Defines a cell in a table
<u><textarea></u>	Defines a multiline input control (text area)
<u><tfoot></u>	Groups the footer content in a table
<u><th></u>	Defines a header cell in a table
<u><thead></u>	Groups the header content in a table
<u><time>New</u>	Defines a date/time
<u><title></u>	Defines a title for the document
<u><tr></u>	Defines a row in a table
<u><track>New</u>	Defines text tracks for media elements (<video> and <audio>)
<u><tt></u>	Not supported in HTML5
<u><u></u>	Defines text that should be stylistically different from normal text
<u></u>	Defines an unordered list
<u><var></u>	Defines a variable
<u><video>New</u>	Defines a video or movie
<u><wbr>New</u>	Defines a possible line-break

3.7.1.11 - Microdados

É a capacidade de fornecer mais semântica as páginas WEB, com intuito de facilitar a coleta de informações por WEBcrawlers. Para que cada um não crie o seu próprio vocabulário, já existem vocabulários que você pode utilizar. Os temas abordados pelo vocabulário são: Event, Organization, Person, Product, Review, Review-aggregate, Breadcrumb, Offer, Offer-aggregate, address e etc.

Visite: <http://data-vocabulary.org/>

O itemtype aponta para um dicionário, o itemscope cria um item e informa aos seus descebentes que eles são um sub-tipo dele e o itemprop (propertie) é o atributo ppd.

Trecho HTML sem semântica

```
<h1 Visie - Treinamento e Desenvolvimento Web</h1>
<p tel: 11 3477-3347</p>
<p Endereço: Alameda dos Ubiatans, 240 Planalto Paulista
São Paulo SP </p>
```

Trecho HTML com semântica

```
<div itemscope itemtype="http://data-vocabulary.org/Organization">
  <h1 itemprop="name">Visie - Treinamento e Desenvolvimento Web</h1>
  <p itemprop="tel">11 3477-3347</p>
```

```

<p itemprop="address" itemscope itemtype="http://data-
vocabulary.org/Address">
    <span itemprop="street-address">Alameda dos Ubiatans, 240 Planalto
Paulista</span>,
    <span itemprop="locality">São Paulo</span>,
    <span itemprop="region">SP</span>.
</p>
</div>

```

3.7.2 - jQuery

jQuery é um framework (biblioteca) simplesmente fantástica para desenvolvimento JavaScript. jQuery é capaz de aplicar um dinamismo incrível a qualquer site, com uma sintaxe muito mais simples, pequena e eficiente que a do JavaScript. Assim como o próprio slogan da linguagem já diz, **"Write Less, Do More (Escreva Menos, Faça Mais)"**, o jQuery é uma forma "compacta" do JavaScript, que permite que escrevamos muito menos código e façamos muito mais coisas.

Resumindo:

- É uma biblioteca de funções **Javascript**
- É uma biblioteca com o lema "Escrever menos e fazer mais"
- Contém os seguintes recursos
 - Seleções de elementos **HTML**
 - Manipulação de elementos **HTML**
 - Manipulação **CSS**
 - Eventos **HTML** DOM
 - Efeitos e animações **Javascript**
 - **HTML** DOM
 - AJAX

O **jQuery** está se tornando substituto do **Flash**, visto que existe muita crítica ao tempo de carregamento deste.

COMO ADICIONAR UMA BIBLIOTECA JQUERY

O **Jquery** não precisa ser instalado, só referenciado.

a - No próprio site

No seu site (depois do download)

```

<head>
<script type="text/javascript"
src="jquery.js">
</script>
</head>

```

A biblioteca pode ser baixada do site oficial <http://www.jquery.com> em dois formatos:

- Comprimida / minificada: para produção (todos os espaços e quebra de linhas são removidos do arquivo, ilegível para humanos). O nome do arquivo é jquery-versão.min.js
- Descomprimida: para desenvolvimento (código identado, legível para humanos). O nome do arquivo é jquery-versão.js

b - Site externo

Apontar p/ o Google

```
<head>
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.4.2/jquery.min.js">
</script>
</head>
```

Apontar p/ a Microsoft

```
<head>
<script type="text/javascript"
src="http://ajax.microsoft.com/ajax/jquery/jquery-1.4.2.min.js">
</script>
</head>
```

Existem As versões, em negrito, são várias.

Plugins e Companions

Já não bastasse a linguagem ser extremamente poderosa e pouco verborrágica, existem as seguintes opções para enriquecer o ambiente de desenvolvimento:

jQueryUI: Fornece animações, controles e efeitos especiais objetivando aumentar a interatividade da página WEB. Cabe destacar o novo controle Acordeão (<http://jqueryui.com/demos/accordion/>, muito interessante). Conferir em: <http://jqueryui.com/download>

jQueryUi Theming: permite que você selecione temas (skins) para a sua página. Conferir em <http://jqueryui.com/docs/Theming>

Plugins: São centenas de plugins desenvolvidos por terceiros. Tem de tudo. Conferir em <http://plugins.jquery.com/>

Companions: são outras bibliotecas que podem ser integradas ao jQuery:

- Fancybox: para exibir conteúdo HTML, multimedia e imagens no estilo Mac.
- Flowplayer: para tocar videos.
- Sliders.js: framework para criação de slides.

Exemplos do uso do jQuery.

a - O comando abaixo altera todos parágrafos do documento para um novo estilo em única tacada.

```
$( "p" ).css({ color: "#FFFFFF", background: "#000000" });
```

Efeito: todos <p> (PARAGRAFOS) ficarão assim

b - Como seria o jQuery e o SIGA-DOC?

O SIGA-DOC já está sendo reestilizado para o jQuery. Como ficaria uma entrevista utilizando a biblioteca jQuery UI. A seguir, um pequeno exemplo:

Código FM para execução do exemplo:

```
[@entrevista]
[@grupo titulo="Teste SIGA-DOC com jQuery"]
    [@texto var="dataemissao" titulo="Selecione a data do curso"]
[/@grupo]
[@grupo]
    [@texto var="ling" titulo="Selecione o curso"]
[/@grupo]
[@grupo]
```

```

[@botaoradio/]
[@grupo]
[@grupo]
[@botaocheck/]
[@grupo]
[@grupo]
[@slider/]
[@grupo]
[@grupo]
[@campodiv iddiv="switcher"]
[@grupo]
[@cargaelementos/]
[@entrevista]

```

Macro: APLICAÇÃO jQuery

```

[#macro cargaelementos]
<script type="text/javascript">
    jQuery(document).ready(function() {
        jQuery("#dataemissao").datepicker();
        jQuery("input#ling").autocomplete({ source: ["c++", "java", "php",
"freemarker", "javascript", "asp", "jsp"] });
        jQuery("#turno").buttonset();
        jQuery("#extra").buttonset();
        jQuery("#slide_me").slider({
            value: 8,
            min: 0,
            max: 10,
            step: 1,
            orientation: 'horizontal',
            slide: function( event, ui ) {
                jQuery("#my_value").val( ui.value );
            }
        });
        jQuery("#switcher").themeswitcher({loadTheme: 'Vader', cookieName:'', width:
160});
    });
</script>
[/#macro]

```

MACROS INLINE SECUNDÁRIAS

```

[#macro botaoradio]
<span style="color:black">Selecionar o Turno:</span>
<div id="turno">
<input type="radio" id="radio1" name="radio" />
<label for="radio1">Manhã</label>
<input type="radio" id="radio2" name="radio" />
<label for="radio2">Tarde</label>
<input type="radio" id="radio3" name="radio" />
<label for="radio3">Noite</label>
</div>
[/#macro]

[#macro botaocheck]
<span style="color:black">Selecionar Extras:</span>
<div id="extra">
    <input type="checkbox" id="check1" value="check1" />
    <label for="check1">Com Café</label>
    <input type="checkbox" id="check2" value="check2"/>

```

```

<label for="check2">Apostila Impressa</label>
<input type="checkbox" id="check3" value="check3" />
    <label for="check3">DVD com os fontes</label>
<input type="checkbox" id="check4" value="check4" />
    <label for="check4">DVD da Aula</label>
</div>
[/#macro]

[#macro slider]
<span style="color:black">Como você avalia o nosso contato?</span>
<input type="text" id="my_value" readonly="readonly"/>
<div id="slide_me"></div>
[/#macro]

```

[#macro campodiv iddiv]

```

<span style="color:black">Selecione o Tema(Skin) :</span>
<div id="${iddiv}"></div> [@br/]
[/#macro]
Tela do aplicativo:
```

Teste SIGA-DOC com jQuery

Selecionar a data do curso:

Selecionar o curso:

Selecionar o Turno:

Manhã	Tarde	Noite
-------	-------	-------

Selecionar Extras:

Com Café	Apostila Impressa	DVD com os fontes	DVD da Aula
----------	-------------------	-------------------	-------------

Como você avalia o nosso contato? 4

Selecionar o Tema(Skin):

O campo data utilizando a função datepicker():

```
jQuery("#dataemissao").datepicker();
```

Teste SIGA-DOC com jQuery

Selecionar a data do curso:

Selecionar o curso:

Selecionar o Turno:

Manhã	Tarde	Noite
-------	-------	-------

Selecionar Extras:

Com Café	Apostila Impressa	DVD com os fontes	DVD da Aula
----------	-------------------	-------------------	-------------

Como você avalia o nosso contato? 4

Selecionar o Tema(Skin):

A screenshot shows a date picker calendar for May 2012. The calendar grid is as follows:

Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Para os cursos foi utilizada a função autocomplete() que vai apresentando uma lista na medida que o usuário digita:

```
jQuery("input#ling").autocomplete({ source: ["c++", "java", "php", "freemarker", "javascript", "asp", "jsp"]});
```

Teste SIGA-DOC com jQuery

Selecione a data do curso: 05/31/2012

Selecione o curso: javascript

Selecionar o Turno:

Manhã Tarde Noite

java
javascript

Selecionar Extras:

Com Café Apostila Impressa DVD com os fontes DVD da Aula

jsp impressa

Como você avalia o nosso contato? 4

Selecionar o Tema(Skin):

Theme: Dot Luv

Ok Visualizar o modelo preenchido

Para o turno utilizamos a função `buttonset()` que aplicada a um grupo de radio buttons produz um novo estilo:

```
jQuery("#turno").buttonset();
```

Teste SIGA-DOC com jQuery

Selecione a data do curso: 05/31/2012

Selecione o curso: javascript

Selecionar o Turno:

Manhã Tarde Noite

Selecionar Extras:

Com Café Apostila Impressa DVD com os fontes DVD da Aula

Com Café Apostila Impressa DVD com os fontes DVD da Aula

Como você avalia o nosso contato? 4

Selecionar o Tema(Skin):

Theme: Dot Luv

Ok Visualizar o modelo preenchido

Para o extra utilizamos a função `buttonset()` que aplicada a um grupo de checkbox produz um novo estilo:

```
jQuery("#extra").buttonset();
```

Teste SIGA-DOC com jQuery

Selecione a data do curso: 05/31/2012

Selecione o curso: javascript

Selecionar o Turno:

Manhã Tarde Noite

Selecionar Extras:

Com Café Apostila Impressa DVD com os fontes DVD da Aula

Com Café Apostila Impressa DVD com os fontes DVD da Aula

Como você avalia o nosso contato? 4

Selecionar o Tema(Skin):

Theme: Dot Luv

Ok Visualizar o modelo preenchido

Para a avaliação utilizamos a barra slider, obviamente só para mostrar o potencial do jQuery:

```
jQuery("#slide_me").slider({
    value: 8,
    min: 0,
    max: 10,
    step: 1,
```

```
        orientation: 'horizontal',
        slide: function( event, ui ) {
            jQuery("#my_value").val( ui.value);
        }
    );
```

Teste SIGA-DOC com jQuery

Selecione a data do curso: 05/31/2012

Selecione o curso: javascript

Selecionar o Turno:

Manhã Tarde Noite

Selecionar Extras:

Com Café Apostila Impressa **DVD com os fontes** DVD da Aula

Como você avalia o nosso contato? 4

Selecionar o Tema(Skin):
Theme: Dot Luv

Ok Visualizar o modelo preenchido

Para mostrar os temas utilizamos a função themeswitcher():

```
jQuery("#switcher").themeswitcher({loadTheme: 'Vader', cookieName:'', width: 160});
```

Teste SIGA-DOC com jQuery

Selecione a data do curso: 05/31/2012

Selecione o curso: javascript

Selecionar o Turno:

Manhã Tarde Noite

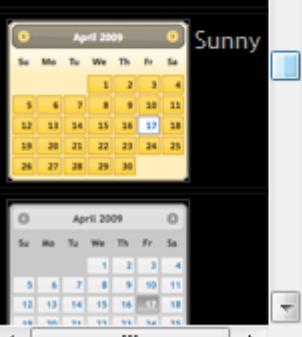
Selecionar Extras:

Com Café Apostila Impressa **DVD com os fontes** DVD da Aula

Como você avalia o nosso contato? 5

Selecionar o Tema(Skin):
Theme: Dot Luv

Redmond



Visualizar o modelo preenchido

3.7.3 - JSON

JSON ou JavaScript Object Notation é um formato de texto para serialização de estrutura de dados e intercâmbio de dados entre servidores, assim como o XML. JSON é uma estrutura de dados em javascript. Apesar de começarem a falar muito disso recentemente, JSON não é uma tecnologia nova, faz parte do javascript puro e não precisa de nenhum script pra trabalhar. O JSON pode substituir o XML, e faz isso muito bem na hora de trabalhar com respostas em AJAX. A estrutura de dados fica mais simples de trabalhar e o tempo de execução de um script lendo dados em JSON é dezenas de vezes mais rápido do que ler um conteúdo XML. Os dados JSON podem ser armazenados em arquivos .json.

As principais linguagens de programação server side tem suporte pra escrever dados em JSON.

```
var dados = {
    "aluno": [
        {"nome":"João", "provas": [ { "nota":8 }, {"nota":6}, {"nota":10 }, {"nota":2 } ] },
        {"nome":"Maria", "provas": [ { "nota":3}, {"nota":5}, {"nota":8 }, {"nota":1 } ] },
        {"nome":"Pedro", "provas": [ { "nota":7}, {"nota":6}, {"nota":6 }, {"nota":8 } ] },
    ]
}
```

```
var jsonData = eval(dados); OU (p/ evitar ambiguidades) var jsonData = eval("(" + dados + ")");
document.write(jsonData.aluno[1].provas.nota[3]);
// Escreve a nota da última prova da Pedro ;
```

Embora possa ser utilizado o método acima para ler os dados, o ideal é utilizar um parser (existente para várias linguagens além do JS). Isto devido aos seguintes fatos:

- segurança, pois o eval na verdade executa um JS e se alguém colocar um código malicioso no lugar JSON, já era;
- alguns caracteres unicode que são válidos em JSON não são válidos em JS.

Este parser não executa código JS, portanto é seguro: **JSON.parse()**.

3.7.3.1 - JSON, JS e AJAX

```
var my_JSON_object = {};
var http_request = new XMLHttpRequest();
http_request.open("GET", url, true);
http_request.onreadystatechange = function () {
    var done = 4, ok = 200;
    if (http_request.readyState == done && http_request.status == ok) {
        my_JSON_object = JSON.parse(http_request.responseText);
    }
};
http_request.send(null)
```

3.7.3.2 - Tipos de dados JSON

- **Number** (double precision floating-point format in JavaScript, generally depends on implementation)
- **String** (double-quoted Unicode (UTF-8 by default), with backslash escaping)
- **Boolean** (true or false)

- **Array** (an ordered sequence of values, comma-separated and enclosed in **square brackets**; the values do not need to be of the same type)
- **Object** (an unordered collection of key:value pairs with the ':' character separating the key and the value, comma-separated and enclosed in **curly braces**; the keys must be strings and should be distinct from each other)
- **null** (empty)

Observe que o JSON pode responder com um número (suponha que o servidor faça inúmeros cálculos e devolva um número), desta forma no código acima (JSON, JS e AJAX) poderíamos ter algo como:

```
my_JSON_object = JSON.parse(http_request.responseText) + 5,756;
```

3.7.3.3 - JSON, JQUERY e AJAX

Ejamos como é fácil utilizar o método **\$.getJSON()** do JQuery para manipular os dados, a sintaxe é:

```
jQuery.getJSON( url, [data], [callback] )
```

- **URL** -> A url que vai ser requisitada (obrigatório)
- **[data]** -> dados a serem enviados via POST (opcional)
- **[callback]** -> Função que vai ser executada quando os dados forem carregados com sucesso (opcional)

Exemplo: Supondo que temos 3 **INPUTs** do tipo **TEXT** no HTML da página e queremos colocar esses dados dentro deles, as IDs desses INPUT são respectivamente:

1. **tipoUdt**
2. **tituloUdt**
3. **descricaoUdt**

Supor que a página a ser requisitada se chama **categoria.php**, e que este programa gere a seguinte resposta JSON:

```
{"tipo": "PRODUTOS", "titulo": "COLECOES", "descricao": "MACIO TESTE"};
```

```
$.getJSON("../actions/admin/categoria.php", function(json) {
    $('#tipoUdt').val(json.tipo);
    $('#tituloUdt').val(json.titulo);
    $('#descricaoUdt').val(json.descricao);
} // fim do callback
); // fim do .getJSON()
```

val() é também um método **JQuery**, utilizado nesse caso para atribuir um valor ao **INPUT** do nosso **FORM HTML**.

Quanto código teríamos escrito em puro JS e XML?

3.8 - Seguem as funções JS disponíveis ao programador (posição de 07/05/2012)

3.8.1 - No arquivo static-javascript.js

Estas funções podem ser mais utilizadas para o desenvolvimento dos formulários, visto que muitas delas possuem propósito geral, são utilitárias, como por exemplo: valores monetários por extenso, verifica data ou hora e etc. Com certeza estas funções darão mais vida aos formulários do SIGA-DOC, e que seriam impossíveis, ou melhor, improdutivas de desenvolver em FM.

Função	Parâmetros	Descrição
function popup	url	
function NewWindow	mypage myname w h scroll size	
function tokenize	url	
function abreEnquete	clicou_voto form num_opcao	Abre um form para realizar enquetes: Se receber o nome do formulário por parâmetro, utiliza-o; caso contrário, usa o nome do formulário padrão (enquete) . Esta possibilidade está aberta para o caso de termos mais de uma pesquisa na mesma página ao mesmo tempo.
function displayTitular	thisElement	
function avisoColorido	msg cor tempo	
function escondeAvisoColorido		
function avisoVerde	msg tempo	
function avisoVermelho	msg tempo	
function avisoAzul	msg	

	tempo	
function isCarregando		
function carrega		
function descarrega		
function verifica_data	data naoObriga	<p>Verifica se o campo foi digitado e segue o formato de uma data.</p> <p>A data deve estar num dos seguintes formatos: DD/MM/AAAA ou DDMMAAAA. Também realiza uma crítica de dia e mês.</p> <p>Como Chamar: onblur="javascript:verifica_data(this, true);" Utilizada na data do documento, ver tabela de mapeamento abaixo</p>
function verifica_hora	hora naoObriga	<p>Verifica se o campo foi digitado e segue o formato de uma hora.</p> <p>A hora deve estar num dos seguintes formatos: HH:MM ou HHMM.</p> <p>Também critica a hora e os minutos.</p>
function sonumeroMesmo	e	<p>Só permite que o campo aceite dígitos (números). Exceto o ponto "..".</p> <p>Como chamar: onkeypress="return sonumeroMesmo(event)"</p>
function handleEnter	field event	<p>Se o usuário tecla enter após digitar, então é feito o processamento</p> <p>Como chamar: onkeypress="return handleEnter(this, event)" Utilizada no subscritor, ver tabela de mapeamento abaixo</p>
function formataReais	fld milSep decSep e	<p>Formatar valores monetários, passando o campo (this, quer dizer este campo), o separador de milhares ('.' ponto no caso), o separador de decimais (', ' vírgula no nosso caso) e o evento (tecla digitada)</p> <p>Como chamar: onkeypress="return formataReais(this, '.', ', ', event)"</p>

function extenso	label valor	Devolve o valor monetário por extenso. Como chamar: onBlur="extenso(\${var}); " \${var} é o display de uma variável do FM
function verificaNumero	e	Só permite a digitação de números (dígitos) no campo. Como chamar: onKeypress="return verificaNumero(event);"
function verificaCampos		Verifica se todos (no código só se considera 18 campos) os campos de um formulário foram digitados. Como chamar: onchange="verificaCampos();"
function dataMaiorIgual	dt1 dt2	Verifica se a data 1 é maior ou igual que a data 2. Como chamar: onchange="dataMaiorIgual(datafim,datainicio);"
function numeroExtenso	label valor	Retorna o valor literal de um numero Como chamar: onBlur="numeroExtenso(\${var}); " \${var} é o display de uma variável do FM
function replaceAll	string token newtoken	Substitui na string fornecida uma character/string por outro caracter/string Como chamar: s = replaceAll(s, "a", "b");
function toEntities	s	Substitui caracteres por suas entities correspondentes Como chamar: s = toEntities(s); Exemplos de entities relacionadas ao "A": "À", "À"

		"Á", "Á" "Â", "Â" "Ã", "Ã"
function setDisabled	el val	
function decodeMainValueRPC	p_nodValueRaiz	
function decodeMethodResponseRPC	p_domDocument	
function SimpleMethodRequestRPCGet		

3.8.2 - Na página editar.action (tela da entrevista)

Estas funções serão mais utilizadas pelos desenvolvedores do SIGA-DOC, porém serão listadas aqui como referência.

Função	Parâmetros	Descrição
function testaConexao		
function limitaDescricao	silencioso	
function naoCaiu	response param	
function avisaCaiu		'Atenção: O sistema parece não estar respondendo. As alterações feitas a partir de agora nesta tela não serão salvadas. Favor, abrir o Siga em uma outra janela.'
function autoSave		
function tryAgain		
function FCKeditor_OnComplete	editorInstance	
function salvaAjaxFCK	automatico	
function salvo	response	

function sbmt	id	<p>Provoca o submit(), ou seja, a leitura, releitura da tela. Se o idajax for passado, então o AJAX entrará em ação, e neste caso só o elemento HTML com o id especificado será inserido na tela</p> <p>Esta função chama:</p> <pre>ReplaceInnerHTMLFromAjaxResponse('/sigaex/expediente/doc/editar. action', frm, id); MixWithNewPage(document.getElementsByTagName('div'), r, obj_id);</pre>
window.customOnsubmit		
function customOnsubmit_frm		
function gravarDoc		<p>Quando se aperta o botão OK, passa-se para outra tela e o documento é gravado</p> <p>Como chamar: onclick="javascript: gravarDoc();" Utilizada no botão remover do preenchimento automático, ver tabela de mapeamento abaixo</p>
function validar	silencioso	
function removePreench		<p>Para remover o preenchimento automático</p> <p>Como chamar: onchange="javascript:removePreench();" Utilizada no botão remover do preenchimento automático, ver tabela de mapeamento abaixo</p>
function alteraPreench		<p>Para alterar o preenchimento automático</p> <p>Como chamar:</p>

		<code>onchange="javascript:alteraPreench()"</code> Utilizada no botão alterar do preenchimento automático, ver tabela de mapeamento abaixo
<code>function carregaPreench</code>		Para carregar o preenchimento automático Como chamar: <code>onchange="javascript:carregaPreench()"</code> Utilizada no preenchimento automático, ver tabela de mapeamento abaixo
<code>function adicionaPreench</code>		Para adicionarr o preenchimento automático Como chamar: <code>onchange="javascript:adicionarPreench()"</code> Utilizada no botão adicionar do preenchimento automático, ver tabela de mapeamento abaixo
<code>function popitup_documento</code>	true or false	Utilizada para gerar o document (HTML ou PDF) No caso do ícone do PDF <code>onclick="javascript: popitup_documento(true);"</code> No caso do botão "Visualizar Modelo preenchido" <code>onclick="javascript: popitup_documento(false);"</code>
<code>function checkBoxMsg</code>		As funções baixo foram definidas como self.nome_variavel = function() ...
<code>retorna_subscritor</code>	<code>id</code> <code>sigla</code> <code>descricao</code>	Retorna o campo sigla e descrição para a tela. É chamada pela função <code>resposta_ajax_subscritor</code> ou pela janela que é aberta pela função <code>popitup_subscritor</code>
<code>popitup_subscritor</code>	<code>sigla</code>	Responsável pela abertura da janela de pesquisa Como chamar: <code>onclick="javascript: popitup_subscritor('');</code> Utilizada no botão ... do subscritor, ver tabela de mapeamento abaixo

		É chamada pela função: resposta_ajax_subscritor
resposta_ajax_subscritor	response d1 d2 d3	<p>É a função de callback (retorno) da chamada AJAX da função ajax_subscritor</p> <p>Funções Chamadas:</p> <pre>return retorna_subscritor(data[1], data[2], data[3]); return popup_subscritor(sigla);</pre>
ajax_subscritor		<p>Responsável pela chamada AJAX referente ao campo campo subscritor</p> <p>Como chamar:</p> <pre>onblur="javascript: ajax_subscritor()"</pre> <p>Utilizada no campo subscritor, ver tabela de mapeamento abaixo</p> <p>Obs: esta função chama uma função da biblioteca Ajax: <code>PassAjaxResponseToFunction(url, 'resposta_ajax_subscritor', false);</code></p>
Idem (o que foi definido para subscritor) para Mobil Pai		
retorna_mobilPai	id sigla descricao	
popup_mobilPai	sigla	
resposta_ajax_mobilPai	response d1 d2 d3	
ajax_mobilPai		
Idem para Titular		
retorna_titular	id sigla descricao	
popup_titular	sigla	
resposta_ajax_titular	response d1	

	d2 d3	
ajax titular		
Idem para Destinatario		
retorna_destinatario	id sigla descricao	
popitup_destinatario	sigla	
resposta_ajax_destinatario	response d1 d2 d3	
ajax destinatario		
Idem para Classificação documental		
retorna_classificacao	id sigla descricao	
popitup_classificacao	sigla	
ajax classificacao		
resposta_ajax_classificacao	response d1 d2 d3	
Idem para Lotação		
retorna_lotacaoDestinatario	id sigla descricao	
popitup_lotacaoDestinatario	sigla	
ajax lotacaoDestinatario		
resposta_ajax_lotacaoDestinatario	response d1 d2 d3	
Idem para Órgão Externo		
retorna	id	

orgaoExternoDestinatario	sigla descricao	
popitup_ orgaoExternoDestinatario	sigla	
ajax_ orgaoExternoDestinatario		
resposta_ajax_ orgaoExternoDestinatario	response d1 d2 d3	

As funções baixo são criadas dinamicamente conforme descrito na [seção 3.4.1.1](#) e funcionam exatamente com as anteriores

Onde: **tipoSel** = {lotacao, pessoa, funcao} e **var** = nome da variável passada a macro @lotacao, @pessoa ou @funcao.

self.ajax_\${var}\${tipoSel} }		
self.resposta_ajax_\${var} \${tipoSel}	response d1 d2 d3	
self.retorna_\${var}\${tipo Sel}	id sigla descricao	
self.popitup_\${var}\${tipo Sel}	sigla	

3.8.3 - Na página buscar.action

Existem funções também aqui, visto que nesta página (tela de busca) existe, por exemplo, pesquisar lotação (no caso da propriedade subscritor ou titular), e consequentemente todas as funções relacionadas à lotação já vistas em editar.action.

3.8.4 - No arquivo Ajax.js

Função	Parâmetros	Descrição
function GetXmlHttp		Obtém uma instância do XMLHttpRequest
function mostraCarregando		Mostra a imagem do Ajax carregando
posiciona		Posiciona o carregando do Ajax no topo, direita da tela
function ocultaCarregando		Oculta a imagem do carregando do Ajax
function PassAjaxResponseToFunction	url callbackFunction params omitirCarregando postParams	Responsável pela requisição AJAX. Utilizada pelas funções ajax_subscritor , Ajax_lotacao e etc. (visto anteriormente)
functionToCall	callbackFunction + '(response,'+params +')'	Conterá a função de retorno (callback), como por exemplo: resposta_ajax_subscritor , reponse_ajax_lotacao e etc.
function MixWithNewPage	aElements sPage sIdObj	Função AJAX auxiliar, cagamada pela função ReplaceInnerHTMLFromAjaxResponse
function ReplaceInnerHTMLFromAjaxResponse	url frm obj_id	Responsável pela requisição AJAX. Utilizada pelas macros FM que utilizam a cláusula depende Esta função chama: MixWithNewPage(document.getElementsByTagName('div'), r, obj_id); MixWithNewPage(document.getElementsByTagName('span'), r, obj_id);
function SetInnerHTMLFromAjaxRespon	url, obj_id)	Responsável pela requisição AJAX.

se		Utilizada pelas macros FM tipo texto, data e etc. quando a opção AJAX é setada
function IsRunningAjaxRequest		Verifica se o AJAX está rodando ou não. Na verdade verifica se o id=carregando está none (desligado) ou block (ligado)

Existem outras pequenas funções que não foram listadas aqui.

3.8.5 - No arquivo **exibir.action**

Na tela de elaboração de documentos temos vários outros botões (ver abaixo) em que não foram implementados via JS e sim, via actions, portanto, não existem novas funções JS para o exibir.action.

Documento Interno Produzido: TMP-146

Anexar Arquivo | Definir Perfil | Duplicar | Editar | Excluir | Exibir Informações Completas | Finalizar | Incluir Co-signatário | Visualizar Dossiê | Visualizar Impressão

Seguem as actions associadas aos botões acima:

```

Anexar&nbsp;Arquivo
Definir&nbsp;Perfil
Duplicar
Editar
Excluir
Exibir&nbsp;Informações&nbsp;Completas
Finalizar
Incluir&nbsp;Co-signatário
Visualizar&nbsp;Dossiê
Visualizar&nbsp;Impressão

```

3.9 – Mapeamento: Campos da parte fixa da entrevista (editar.action) e funções javaScript

```

<select id="frm_idTpDoc" onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" name="idTpDoc">
<input id="frm_dtDocString" type="text" onblur="javascript:verifica_data(this, true);;" value="" size="10" name="dtDocString">
<select id="frm_nivelAcesso" name="nivelAcesso">
<input id="eletronicoCheck1" type="radio" value="1" name="eletronico">
<input id="eletronicoCheck2" type="radio" value="2" name="eletronico">
<input id="frm_subscritorSel_sigla" type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_subscritor();" value="" size="25" name="subscritorSel.sigla">
<input id="subscritorSelButton" type="button" theme="simple" onclick="javascript: popup_subscritor();" value="..." >
<span id="subscritorSelSpan">RUBEN EDWARD ROSE JUNIOR</span>
<input id="frm_substituicao" type="checkbox" onclick="javascript:displayTitular(this);;" value="true" name="substituicao">
<input id="frm_titularSel_sigla" type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_titular();" value="" size="25" name="titularSel.sigla">
<input id="titularSelButton" type="button" theme="simple" onclick="javascript: popup_titular();" value="..." >

```

Documento:	NOVO de: 07/05/12		
Origem:	Interno Produzido	Data:	Acesso Limitado ao órgão (padrão)
Subscritor:	RJ13284	...	RUBEN EDWARD ROSE JUNIOR <input checked="" type="checkbox"/> Substituto
Titular:	RJ13284	...	RUBEN EDWARD ROSE JUNIOR

```

<input id="frm_nmFuncaoSubscritor" type="text" value="" maxlength="128" size="50" name="nmFuncaoSubscritor">
<select id="frm_tipoDestinatario" onchange="javascript:sbmt();" name="tipoDestinatario">
PARA DESTINATÁRIO TIPO MATRÍCULA
<input id="frm_destinatarioSel_sigla" type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_destinatario();" value="RJ13284" size="25" name="destinatarioSel.sigla">
<input id="destinatarioSelButton" type="button" theme="simple" onclick="javascript: popup_destinatario();" value="..." >
<span id="destinatarioSelSpan"> RUBEN EDWARD ROSE JUNIOR </span>
PARA DESTINATÁRIO TIPO LOTAÇÃO
<input id="frm_lotacaoDestinatarioSel_sigla" type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_lotacaoDestinatario();" value="STI" size="25" name="lotacaoDestinatarioSel.sigla">
<input id="lotacaoDestinatarioSelButton" type="button" theme="simple" onclick="javascript: popup_lotacaoDestinatario();" value="..." >
<span id="lotacaoDestinatarioSelSpan"> Subsecretaria de Tecnologia da Informação e de Comunicações </span>
PARA DESTINATÁRIO TIPO ÓRGÃO EXTERNO
<input id="frm_orgaoExternoDestinatarioSel_sigla" type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_orgaoExternoDestinatario();" value="" size="25" name="orgaoExternoDestinatarioSel.sigla">
<input id="orgaoExternoDestinatarioSelButton" type="button" theme="simple" onclick="javascript: popup_orgaoExternoDestinatario();" value="..." >
PARA DESTINATÁRIO TIPO CAMPO LIVRE
<input id="frm_nmDestinatario" type="text" value="" maxlength="256" size="80" name="nmDestinatario">

```

Função;Lotação;Localidade:		(Opcionalmente informe a função e a lotação na forma: Função;Lotação;Localidade)	
Destinatário:	Matrícula	RJ13284	...
RUBEN EDWARD ROSE JUNIOR			

```

<select id="frm_idFormaDoc" onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" name="idFormaDoc">
<select id="frm_idMod" onchange="document.getElementById('alterouModelo').value='true';sbmt();" style="" name="idMod">
<select id="frm_preenchimento" onchange="javascript:carregaPreench()" name="preenchimento">
<input type="button" disabled="disabled" onclick="javascript:alteraPreench()" value="Alterar" name="btnAlterar">
<input type="button" disabled="disabled" onclick="javascript:removePreench()" value="Remover" name="btnRemover">
<input type="button" onclick="javascript:adicionaPreench()" name="btnAdicionar" value="Adicionar">

```

Tipo:	Anexo
Modelo:	RubenTeste
Preenchimento Automático:	[Em branco] <input type="button" value="Alterar"/> <input type="button" value="Remover"/> <input type="button" value="Adicionar"/>

```

<input id="frm_classificacaoSel_sigla" type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_classificacao();" value="" size="25" name="classificacaoSel.sigla">
<input id="classificacaoSelButton" type="button" theme="simple" onclick="javascript: popup_Classificacao();" value="...">
<span id="classificacaoSelSpan"> ORGANIZAÇÃO E FUNCIONAMENTO: ADMINISTRAÇÃO JUDICIÁRIA: ORGANIZAÇÃO ADMINISTRATIVA: Modernização Administrativa </span>
<textarea id="descrDocumento" rows="3" cols="80" name="descrDocumento"></textarea>

```

Classificação:	00.01.01.01	<input type="button" value="..."/>	ORGANIZAÇÃO E FUNCIONAMENTO: ADMINISTRAÇÃO JUDICIÁRIA: ORGANIZAÇÃO ADMINISTRATIVA: Modernização Administrativa
Descrição:	<p>(preencher o campo acima com palavras-chave, sempre usando substantivos, gênero masculino e singular)</p>		

```

<input type="button" value="Ok" name="gravar" onclick="javascript: gravarDoc();">
<input type="button" onclick="javascript: popup_documento(false); value="Visualizar o modelo preenchido" name="ver_doc">


```

Ok

Visualizar o modelo preenchido



Observar que os campos que possuem sbmt(), pois provoca o submit(), ou seja, a leitura (releitura) da tela. Como mencionado na seção 1.5 – Campos da entrevista que provocam a leitura (releitura) da tela.

O DATA-MODEL DISPONIBILIZADO

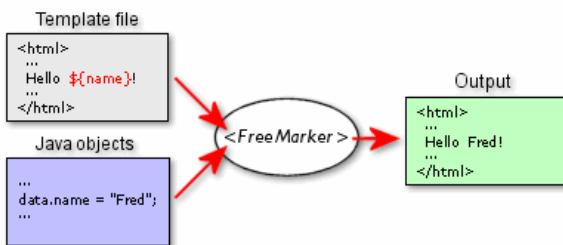
O FreeMarker utiliza HASH como data-model. Um hash pode conter escalares (string, booleano, número ...), sequências (sequences), coleções (collections), objetos e outros hashes. O próprio manual do Freemaker menciona que:

TEMPLATE + DATA-MODEL = OUTPUT

Sem o Data_Model o FM não teria comunicação com o meio externo.

4.1 – O que é um Data-Model

O próprio manual do FM, em sua primeira página, menciona o que eles chamam de equação fundamental, ou seja, **Template + Data-model = Output**, onde o template é formado pelas diretivas HTML e o Data-model formado pelos objetos Java disponibilizados (expostos) ao FM, conforme figura abaixo.



4.2 – O Data-Model disponibilizado

Segue o Data-model listado pela macro @dumpall. No momento da execução desta macro, os seguintes campos da entrevista do SIGA-DOC estavam preenchidos, conforme ilustrado pela figura abaixo.

Documento:	NOVO de: 17/04/12		
Origem:	Interno Produzido	Data:	12/04/2012
Subscritor:	RJ13284	RUBEN EDWARD ROSE JUNIOR	<input checked="" type="checkbox"/> Substituto
Titular:	RJ13285	EDSON SALES DA ROCHA	
Função;Lotação;Localidade:	Analista, CSIS, RJ (Opcionalmente informe a função e a lotação na forma: Função;Lotação;Localidade)		
Destinatário:	Matrícula	RJ11267	HERVAL LEMOS DO AMARAL JUNIOR
Tipo:	Anexo		
Modelo:	RubenTeste		
Preenchimento Automático:	[Em branco]	Alterar	Remover
Classificação:	00.01.01.06	ORGANIZAÇÃO E FUNCIONAMENTO: ADMINISTRAÇÃO JUDICIÁRIA: ORGANIZAÇÃO ADMINISTRATIVA: Documentos	
Descrição:	teste para rodar o dump		

Nome	Conteúdo / Tipo	Descrição
subscritorSel.sigla	"RJ13284"	Sigla (matrícula) do Subscritor
destinatarioSel.id	"9995"	
root	?? (hash)	Hash contendo informações do data-model, ou seja, exatamente o que está listado nesta tabela
classificacaoSel.descricao	"ORGANIZAÇÃO E FUNCIONAMENTO: ADMINISTRAÇÃO JUDICIÁRIA: ORGANIZAÇÃO ADMINISTRATIVA: Documentos operacionais referentes à estrutura organizacional "	Descrição da Classificação documental
mobilPaiSel.descricao	""	

reqmobilPaiSel	""	
classificacaoSel.sigla	"00.01.01.06"	Código da classificação documental
reqdestinatarioSel	"sim"	
reqsubscritorSel	""	
tipoDestinatario	"1"	1 - matrícula, 2 – UO, 3 – órgão externo 4 – campo livre
func	?? (hash)	Hash contendo informações de várias funções, da classe FuncoesEL.java (ver capítulo “Hashes e seus Métodos”)
exbl	?? (hash)	Hash contendo informações do SIGA-DOC , da classe ExBL.java (ver capítulo “Hashes e seus Métodos”)
preenchimento	"0"	
alterouModelo	""	
idFormaDoc	"60"	
mobilPaiSel.id	""	
eletronico	"1"	1 – é eletrônico 0 – é físico
mobilPaiSel.buscar	""	
webwork.token	"89CNO68A2N5G3QZT8UGNDZMX DBB48WRY"	
gerar_entrevista	true	
alterouSel	"destinatario,,,,"	
desativarDocPai	""	
nmArqMod	?? (enumerável)	
titularSel.buscar	""	
mobilPaiSel.sigla	""	
titularSel.descricao	"EDSON SALES DA ROCHA"	Titular
reqtitularSel	""	
idTpDoc	"1"	
subscritorSel.buscar	""	
nmMod	"RubenTeste"	Nome do template do FM
reqclassificacaoSel	""	
param	?? (hash)	Hash contendo parâmetros (ver capítulo “Hashes e seus Métodos”)
classificacaoSel.id	"1773"	
campos	"despachando,criandoAnexo,idTpDoc,dtDocString,nivelAcesso,eletronico,subscritorSel.id,substituicao,titularSel.id,nmFuncaoSubscritor,tipoDestinatario,destinatarioSel.id,preenchimento,classificacaoSel.id,descrDocumento"	
destinatarioSel.buscar	""	
nomePreenchimento	""	
titularSel.id	"10374"	ID do titular
subscritorSel.id	"10199"	ID do subscritor
classificacaoSel.buscar	""	
destinatarioSel.descricao	"HERVAL LEMOS DO AMARAL"	Destinatário

	JUNIOR"	
FALSE.substituicao	"false"	
subscritorSel.descricao	"RUBEN EDWARD ROSE JUNIOR"	Descrição(nome) do subscritor
entrevista	"1"	
idMod	"742"	
criandoAnexo	"false"	
descrDocumento	"teste para rodar o dump"	Descrição da aplicação FM
despachando	"false"	
destinatarioSel.sigla	"RJ11267"	Sigla (matrícula) do destinatário
postback	"1"	
sigla	""	
nivelAcesso	"1"	
titularSel.sigla	"RJ13285"	Sigla (matrícula) do titular
doc	?? (hash)	Hash contendo informações do documento, classe ExDocumento.java (ver capítulo "Hashes e seus Métodos")
nmFuncaoSubscritor	"Analista, CSIS, RJ"	Função, Lotação, Localidade
substituicao	"true"	
webwork.token.name	"webwork.token"	
dtDocString	"12/04/2012"	Data do documento
template	"[##- [#assign qqcoisa = doc.getNmSubscritor()?string/] \${qqcoisa}[@br/] --] [@dumpall/]"	A variável template contém o fonte da própria aplicação

Desta forma, podemos mencionar que o DATA-MODEL disponibilizado as aplicações FM segue a seguinte estrutura:

```

Root
  + Func  (hash)
  |
  + Exbl  (hash)
  |
  + Doc   (hash)
  |
  + Param (hash)
  |
  + Variáveis

```

4.3 – Como acessar as variáveis do Data-Model

```

Imprimindo / Exibindo
${.data_model["destinatarioSel.sigla"]!}      OU

${.vars["destinatarioSel.sigla"]}

Associando a uma variável
[#assign item = .data_model["destinatarioSel.sigla"]! /]      OU

[#assign item = .vars["destinatarioSel.sigla"]! /]

```

4.4 – Como criar um Data-Model

O programador não pode criar um Data-Model, sendo que o mesmo deve ser solicitado ao administrador do ambiente.

4.4.1 – Criando o hash pelo Java

4.4.1.1 – Criando um MAP

Este mapa seria criado na aplicação JAVA que carrega o FM Engine.

Trecho de código

```
...
// Create the root hash
Map root = new HashMap();
// Put string ``user'' into the root
root.put("user", "Big Joe");
// Create the hash for ``latestProduct''
Map latest = new HashMap();
// and put it into the root
root.put("latestProduct", latest);
// put ``url'' and ``name'' into latest
latest.put("url", "products/greenmouse.html");
latest.put("name", "green mouse");

...
```

O que geraria o seguinte **data-model (hash table)**:

```
(root)
|
+- user = "Big Joe"
|
+- latestProduct
    |
    +- url = "products/greenmouse.html"
    |
    +- name = "green mouse"
```

4.4.1.2 – Criando uma Classe e agregando ao hash

Cria-se a classe

```
public class TestObject {
    private String name;
    private int price;

    public TestObject(String name, int price) {
        this.name = name;
        this.price = price;
    }

    // JavaBean properties
    // Note that public fields are not visible directly;
    // you must write a getter method for them.
    public String getName() {return name;}
    public int getPrice() {return price;}

    // A method
```

```
    public double sin(double x) {  
        return Math.sin(x);  
    }  
}
```

Cria-se um hash e agraga-se o objeto

[trecho de código]

```
...  
SimpleHash root = new SimpleHash();  
// expose a "simple" java objects:  
root.put("theString", "wombat");  
// expose an "arbitrary" java objects:  
root.put("theObject", new TestObject("green mouse", 1200));  
...
```

Acessando as variáveis (e métodos) da Classe pelo Freemarker

```
 ${theObject.name}  
 ${theObject.price}  
 ${theObject.sin(123)}
```

O resultado seria:

```
green mouse  
1200  
-0,45990349068959124
```

4.5 - Mapa da Mina - Mapeamento dos campos da parte fixa da entrevista que estão no hash doc

tipoDestinatario = 1 tipoDestinatario = 2 tipoDestinatario = 3 tipoDestinatario = 4	destinatarioSel.sigla lotacaoDestinatarioSel.sigla orgaoExternoDestinatarioSel.sigla nmDestinatario	destinatarioSel.descricao lotacaoDestinatarioSel.descricao orgaoExternoDestinatarioSel.descricao
--	--	--

Documento: TMP-31 de: 13/04/12

Origem: idTpDoc

Subscritor: subscritorSel.sigla

Titular: titularSel.sigla

Função;Lotação;Localidade: nmFuncaoSubscritor

Destinatário: tipoDestinatario

Tipo: idFormaDoc

Modelo: idMod

Preenchimento Automático: preenchimento Alterar Remover Adicionar

Classificação: classificacaoSel.sigla

Descrição: descrDocumento

(preencher o campo acima com palavras-chave, sempre usando substantivos, gênero masculino e singular)

Lotação: lotacao_lotacaoSel.sigla

Servidor: lotacao_pessoaSel.sigla

lotacao_lotacaoSel.descricao

lotacao_pessoaSel.descricao

Muitos dos campos acima podem ser obtidos através das variáveis ou métodos do **hash doc** como será mostrado no capítulo seguinte "HASHES DO DATA-MODEL E SEUS MÉTODOS." Tabém no capítulo "DISSECANDO O FORMULÁRIO DE ENTREVISTA", seção "NOMES por ordem de aparição e elementos HTML", fornecemos uma lista com todos os [nomes dos campos da tela do formulário de entrevista do SIGA-DOC](#).

Como visto no capítulo 4, o Data_Model pode ser composto por classes (hashes) mapeadas (expostas) pela aplicação Java que carrega o FM Engine. Aqui detalharemos alguns métodos que são úteis às aplicações FM, tanto no acesso os dados dos documentos quanto em relação a funções utilitárias que podem ser invocadas. Por exemplo, nos capítulos anteriores vimos como ter acesso ao campo Função, Lotação, Localidade. Aqui veremos que existem métodos para retornar só a Função ou Lotação, por exemplo, ou que podemos adicionar outra informação (que não está documentada) após a Lotação que seria o Subscritor.

5.1 - VARIÁVEIS DO HASH FUNC - CLASSE FuncoesEl.java

Este hash contém inúmeras funções utilitárias que podem ser invocadas pelas aplicações FM.

Método	Descrição	Parâmetros	Utilização no FM
hashCode			
quebraLinhas	Quebra a linha quando inserido um texto longo. <i>Não identificamos bem como é realizado a quebra de linha</i>	Passados: String: frase Recebido: String: frase com quebra de linhas	[#assign suprid = func.quebraLinhas("O funcionário foi convocado")/ \${suprid} O funcionário foi convocado
contains			
has			
calculaData	Calcula a data somando o número de dias a uma data existente.	Passados: String: quantida_de_dias String: data_inicial Recebido: Date: data_final	[#assign suprid = func.calculaData("22","08/09/2012")/ \${suprid} 29/09/2012
formatarCPF	Formata o cpf com o formato de 11 caracteres determinando o número de zeros a esquerda.	Passados: String: cpf Recebido: String: cpf_formatado	[#assign cpf = func.formatarCPF("111112345678")/ \${cpf} 111.123.456-78
lotacao	Consulta a Lotação a partir do seu id.	Passados: Long: id da lotação Recebido DpLotacao	[#assign lota = func.lotacao(1005)/ \${lota.sigla} STI
fixFontSize			
reaisPorExtenso	O método realiza a operação onde o valor numérico digitado é mostrado por extenso.	Passados: String valor Recebido: String: valor_extenso	[#assign s= func.reaisPorExtenso("1000")/ \${s} Mil reais
monetarioParaFloat	Converte um valor monetário em Float	Passados: String: monetario Recebido: Float: monetario	[#assign s= func.monetarioParaFloat("3600,00")/ \${s} 3.600
cargoPessoa		Passados: Long: id da pessoa Recebido String: cargo da pessoa	[#assign cargo=cargoPessoa(10199)/ \${cargo} ANALISTA JUDICIÁRIO INFORMÁTICA
obterBlobMateriaLivre			
buscaDocumentosParaPublicar			
equals			
class ??(hash)			
destinacaoPorNumeroVia			
enxugaHtml			

obterTituloMateriaLivre			
removeAcento	Retira o acento das palavras.	Passados: String: texto Recebido: String: texto_sem_acento	[#assign s=func.removeAcento("léo")/] \$s leo
dao			
tratamento			
podeRemeterPorConfiguracao			
toString			
obterHierarquizadorBIE			
removeAcentoMaiusculas	Remove acento e converte as palavras em maiúsculas.	Passado: String: texto Recebido: String: texto_modificado	@entrevista] [#assign s=func.removeAcentoMaiusculas("grécia")/] \$s GRECIA
idadeEmAnos	Insere a Data de nascimento e retorna sua idade subtraindo da data do sistema. "Date().getTime()"	Passado: String:data Recebido: Float:data	[@entrevista] [#assign s=func.idadeEmAnos("12/02/1992")/] \$s 20
pessoasPorLotacao	Permite listar os Servidores de uma lotação passando o id da mesma. Pode-se passar também se inclui as sublotações ou não, visto que as lotações formam uma hierarquia. Esta função só lista servidores, portanto não lista: Juiz Federal, Juiz Substituto e Estagiários	Passados: Long: id da lotação Boolean: false ou true, se inclui ou não a sublotação Recebido: uma lista do tipo DpPessoa	[#assign buscarSublotacoes = true/] [#assign pessoas = func.pessoasPorLotacao(.vars['lotacao_lotacaoSel.id'])?number, buscarSublotacoes]/] [#list pessoas as pes] Matricula: \${pes.sigla} Nome: \${pes.descricao} [/#list] Matricula: xxxxx Nome: yyyyyy ... Matricula: aaaaaa Nome:bbbb
pessoasPorTipoPorLotacao	Permite listar as pessoas em função de um filtro combinado. Ver observação após esta tabela.	Passados: Long: id da lotação Boolean: false ou true, se inclui ou não as sublotações String: filtro Recebido: uma lista do tipo DpPessoa	Exemplo: Listar o servidores e magistrados. [#assign buscarSublotacoes = true/] [#assign pessoas = func.pessoasPorTipoPorLotacao(.vars['lotacao_lotacaoSel.id'])?number, buscarSublotacoes, '01+02']/] [#list pessoas as pes] Matricula: \${pes.sigla} Nome: \${pes.descricao} [/br] [/#list] <i>O resultado será:</i> Matricula: xxxxx Nome: yyyyyy ... Matricula: aaaaaa Nome:bbbb
mesModData	Converte a data (dd/mm/aaaa) em dd/mm	Passados: String: data_ddmmaaa Recebido:	[#assign s=func.mesModData("12/02/1992")/] \$s

		String: data_ddmm	12/02
podeDefinirPublicadoresPorConfiguracao			
maiusculasEMinusculas	Converte maiúscula em minúscula.	Passados: String: texto Recebido: String: texto-em_minusculo	[#assign s=func.maiusculasEMinusculas("LEONARDO")/] \${s?string} leonardo
stringParaMinuscula			
quebraString			
extraiCamposDescrAutomatica			
floatParaMonetario			
buscaDocumentosAssinadosAgrupadosPorMetadado			
buscaDocumentosAssinados			
monetarioParaLong			
verificaGenero			
getClass			
pessoa			
resource			
concat			
lotacaoPorNivelMaximo			
criarWorkflow			
primeiraMaiuscula			
calculaDiferencaDatas			
replace			
stringParaMinusculaNomes			
lotacaoPessoa			

Observação:

O novo método pessoasPorTipoPorLotacao()

Histórico:

O método pessoasPorLotacao() não atende as todas necessidades, pois só lista Servidores. Foi desenvolvido outro método, porém o mesmo se baseava na descrição do cargo. O Renato, então na época, solicitou para não dar continuidade, visto que: os nossos cargos não estão padronizados e não atenderia a demanda de outros órgãos / instituições.

O novo método:

Baseia-se somente no Tipo_Pessoa. Atualmente na SJRJ temos 4 tipos: 1-Magistrado, 2-Servidor, 3-Estagiário e 4-Terceiro.

Assinaturado método pessoasPorTipoPorLotacao(id_lotacao, inclui_sublotacoes, filtro)

Parâmetros Passados:

Long: id da lotação
Boolean: false ou true, se inclui ou não as sublotações
String: filtro

Parâmetro Recebido:

DpPessoa: uma lista do tipo DpPessoa ou null

A novidade é o filtro, o qual é uma string que o programador deve passar indicando os critérios para selecionar as pessoas.

A sintaxe do filtro:

Como não sabemos em que órgãos / instituições ele será utilizado, e consequentemente, não sabemos a quantidade de "Tipo_Pessoa" e as demandas que poderão surgir, resolvemos parametrizar o filtro com intuito de mitigar a possibilidade de duplicação de métodos e manutenções futuras.
A princípio, poderíamos permitir que somente 1 tipo de pessoa fosse passado como parâmetro, porém estendemos esta capacidade para permitir qualquer quantidade.

O filtro segue a lógica da teoria de conjuntos, no qual,

- o + assume o papel da união;
- o - o papel da diferença;
- Ø (todos os tipos de pessoa) assume o papel de conjunto universo;
- cada tipo de pessoa assume o papel de um conjunto.

Pode parecer um preciosismo, ou para alguns, inutilidade, porém ao observarmos que somente 4 tipos de pessoa (por enquanto, no caso da SJRJ) permitem 15 combinações distintas de filtros, ou seja, 15 possibilidades de consultas, observamos que as 4 consultas básicas (uma para cada tipo de pessoa) não atenderão as demandas futuras.

Se por exemplo, amanhã passarmos para 9 tipos (magistrado titular, magistrado substituto, analista, técnico, auxiliar, requisitado sem vínculo, requisitado com vínculo, estagiário e terceiro), isto permitirá 511 tipos distintos de filtros!!!!!! Só por curiosidade, a quantidade de filtros é dada por $2^n - 1$, onde n é a quantidade de tipos de pessoa.

Exemplos de filtros:

Filtros	Descrição
0-4	Todas pessoas, exceto os terceiros. Este filtro equivale ao filtro 1+2+3
1+2	Magistrados e servidores
1+2+3+4	Magistrados, servidores, estagiários e terceiros. Ou seja, todos, equivalente ao 0.

Exemplo de chamada e utilização do método no FM:

```
[#assign buscarSublotacoes = true/]
[#assign pessoas = func.pessoasPorTipoPorLotacao(1005, buscarSublotacoes, '1+2')/]
[!-- 1005 é o id da STI --]
[#list pessoas as pes]
Matricula: ${pes.sigla}
Nome: ${pes.descricao} [/br]
[/#list]
```

O resultado será:

```
Matricula: xxxxx Nome: YYYYYY
...
Matricula: aaaaaa Nome:bbbb
```

5.1.1 - Como Acessar uma classe e seus atributos pelo FM

Os métodos: **cargoPessoa**, **pessoasPorLotacao**, **pessoa**, **lotacaoPessoa** devolvem objetos armazenados no BD correspondentes a: Pessoa, Lotacao e Cargo. No caso de **pessoasPorLotacao**, é devolvido um array de pessoas. Porém quais são os atributos destas entidades / classes que podemos acessar e como acessá-los?

```
[#assign var = func.pessoa('12345')/] Onde 12345 é o id do funcionário
[#assign nome_Func = var.descricao/]
```

No FM, para acessar os atributos utiliza-se `var.nomeatributo`, que na verdade é `var.nomedometodo()`, visto que não podemos acessar o atributo de uma classe diretamente, e sim, via um método. O nome do atributo é o nome do método sem o get ou set. Se temos o atributo Matricula, temos os métodos `getMatricula` e `setMatricula`. No FM, podemos ignorar o get e referenciar como: `var.sigla`.

5.1.2 - Quais Atributos podemos acessar

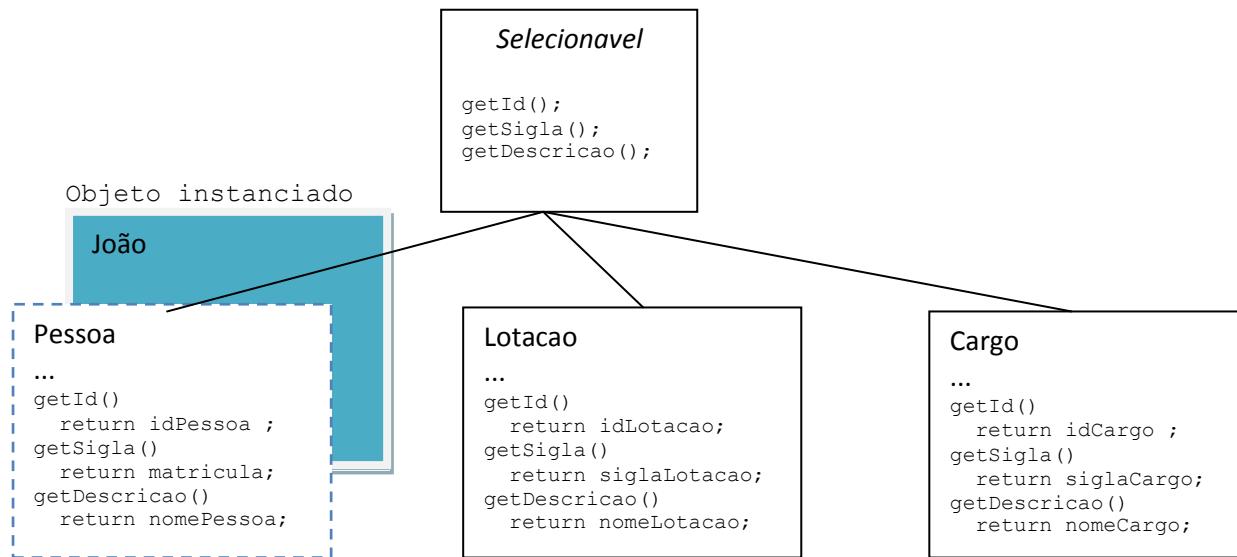
Alias em função do polimorfismo

As macros de negócio (ver seção 9.5.3) e e outras (destinatário, subscriptor ...) são parametrizadas, ou seja, em função de uma identificação (id) elas retornam uma sigla e uma descrição. Para tal, utilizam a interface selecionável, e dependendo do objeto instanciado (pessoa, lotacao, funcao ou cargo), as mesmas retornam as informações corretas sobre id, sigla e descrição (polimorfismo).

Exemplo: Macro Lotação

Lotação: STI ... Subsecretaria de Tecnologia da Informação e de Comunicações

Sigla **Descrição**



No exemplo acima, se for feita uma solicitação a classe selecionável em relação a `getDescricao()`, será retornado o nome do funcionário, no caso, João.

Alias	Classe Pessoa / Atributos principais	Classe Lotacao / Atributos principais	Classe Cargo / Atributos principais
<code>id</code>	<code>idPessoa</code>	<code>idLotacao</code>	<code>idCargo</code>
<code>sigla</code>	<code>matricula</code>	<code>siglaLotacao</code>	<code>siglaCargo</code>
<code>descricao</code>	<code>nomePessoa</code>	<code>nomeLotacao</code>	<code>nomeCargo</code>

Obs: Não fazer confusão, poi a classe pessoa também possui o atributo `siglaPessoa` que representa as três letras de acesso a rede.

Desta forma, tanto faz referenciar `var.descricao` como `var.nomePessoa`.

O Atributo é o nome do método, sem o get e com a primeira minúscula. Ex: método: `getNomeLotacao()` atributo: `nomeLotacao`

LISTA DE ATRIBUTOS QUE PODEM SER ACESSADOS

AbstractDpPessoa

- getTpPessoa(): CpTipoPessoa
- getNacionalidade(): String
- getNaturalidade(): String
- getImprimeEndereco(): String
- getIdEstadoCivil(): Integer
- getIdProvimento(): Integer
- getGrauInstrucao(): String
- getAtoNomeacao(): String
- getDataExercicioPessoa(): Date
- getIdPessoa(): String
- getCargo(): DpCargo
- getCpfPessoa(): Long
- getDataFimPessoa(): Date
- getDataInicioPessoa(): Date
- getDataNascimento(): Date
- getEmailPessoa(): String
- getFuncaoConfianca(): DpFuncaoConfianca
- getIdPessoa(): Long
- getIdPessoaIni(): Long
- getLotacao(): DpLotacao
- getMatricula(): Long
- getNomePessoa(): String
- getOrgaoUsuario(): CpOrgaoUsuario
- getSesbPessoa(): String
- getPadraoReferencia(): String
- getSituacaoFuncionalPessoa(): String
- getPessoalInicial(): DpPessoa
- getSexo(): String
- getSiglaPessoa(): String
- getSituacaoFuncional(): String
- getTipoServidor(): Integer
- getTipoSanguineo(): String
- getEndereco(): String
- getBairro(): String
- getCidade(): String
- getCep(): String
- getTelefone(): String
- getIdentidade(): String
- getOrgaoIdentidade(): String
- getDataExpedicaoIdentidade(): Date
- getUfIdentidade(): String
- getDataNomeacao(): Date
- getDataPosse(): Date
- getDataPublicacao(): Date
- getPessoasPosteriores(): Set<DpPessoa>
- getNomeExibicao(): String

AbstractDpLotacao

- getCpTipoLotacao(): CpTipoLotacao
- getIdLotacao(): String
- getDataFimLotacao(): Date
- getDataInicioLotacao(): Date
- getIdLotacao(): Long
- getIdLotacaoIni(): Long
- getLotacaoPai(): DpLotacao
- getNomeLotacao(): String
- getSiglaLotacao(): String
- getOrgaoUsuario(): CpOrgaoUsuario
- getLotacoesPosteriores(): Set<DpLotacao>
- getLotacaoInicial(): DpLotacao
- getDpLotacaoSubordinadosSet(): Set<DpLotacao>

AbstractDpCargo

- getIdCargo(): Long
- getNomeCargo(): String
- getDataFimCargo(): Date
- getDataInicioCargo(): Date
- getIdCargoIni(): Long
- getIdCargo(): String
- getOrgaoUsuario(): CpOrgaoUsuario
- getSiglaCargo(): String

AbstractDpFuncaoConfianca

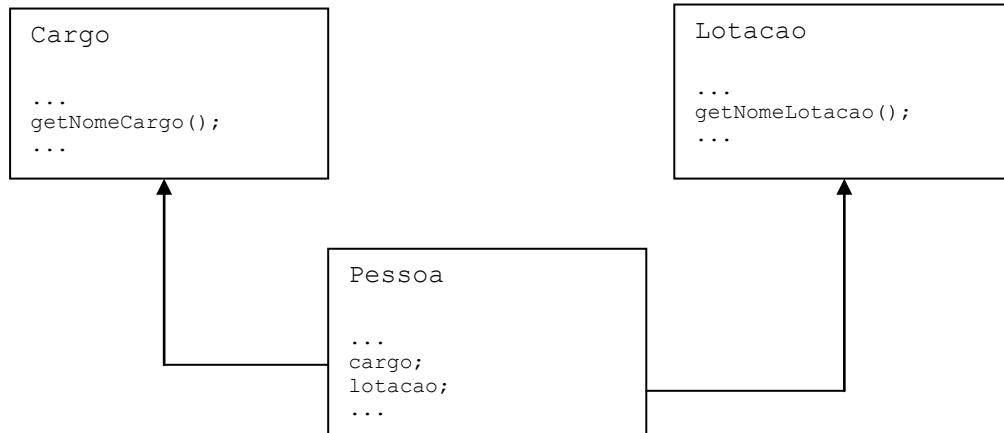
- getOrgaoUsuario(): CpOrgaoUsuario
- getCategoriaFuncao(): String
- getCodFolhaFuncao(): Integer
- getIdFuncao(): Long
- getIdFuncaoPai(): Long
- getNivelFuncao(): Integer
- getNomeFuncao(): String
- getDataFimFuncao(): Date
- getDataInicioFuncao(): Date
- getIdFuncaoIni(): Long
- getIdFuncao(): String
- getSiglaFuncaoConfianca(): String

AbstractDpSubstituicao

- getDtFimSubst(): Date
- getDtIniSubst(): Date
- getIdSubstituicao(): Long
- getLotaSubstituto(): DpLotacao
- getLotaTitular(): DpLotacao
- getSubstituto(): DpPessoa
- getTitular(): DpPessoa
- getIdRegistroInicial(): Long
- getDtFimRegistro(): Date
- getDtIniRegistro(): Date

5.1.3 - Navegando nas classes

Uma Pessoa possui Cargo e Lotação. Pessoa só contém o Id do Cargo e da Lotacao. Como obter o nome do Cargo e da Lotacao?



```
[#-- Instancia Pessoa --]
[#assign pes = func.pessoa(10199)]  [#-- Onde 12345 é o id do funcionário --]
[#assign nomeFunc = pes.nomePessoa /]
${nomeFunc} {@br/}

[#-- Obtem endereço Cargo --]
[#assign enderecoCargoFunc = pes.cargo /]
${enderecoCargoFunc} {@br/}

[#-- Navega para Cargo --]
[#assign idCargoFunc = pes.cargo.idCargo /]
${idCargoFunc} {@br/}

[#assign descricaoCargoFunc = pes.cargo.nomeCargo /]
${descricaoCargoFunc} {@br/}

[#-- Obtem endereço Lotação --]
[#assign enderecoLotaçãoFunc = pes.lotacao /]
${enderecoLotaçãoFunc} {@br/}

[#-- Navega para Lotação --]
[#assign idLotaçãoFunc = pes.lotacao.idLotação /]
${idLotaçãoFunc} {@br/}

[#assign descricaoLotaçãoFunc = pes.lotacao.nomeLotação /]
${descricaoLotaçãoFunc}
```

O resultado será:

RUBEN EDWARD ROSE JUNIOR
br.gov.jfrj.siga.dp.DpCargo@33fd30b5
74
ANALISTA JUDICIARIO/INFORMATICA
br.gov.jfrj.siga.dp.DpLotação@3b3fd3ae
1.122
Seção de Sistemas Especializados



Observação: queria fazer uma navegação, a partir de uma pessoa, descobrir quem é o chefe dela e exibir as informações do chefe. Pelo modelo hoje existente, isto não é possível.

Na seção 9.2.12 (Obtendo dados do Funcionário) temos mais exemplos, e mostra como navegar da classe Doc para Pessoa e depois Cargo (doc.destinatario.cargo.nomeCargo).

5.2 – VARIÁVEIS DO HASH EXBL – Classe ExBL.java

Este hash contém funções utilizadas pela infraestrutura do SIGA-DOC, ou seja, são métodos (funções) de negócio, e raramente, ou mesmo nunca, seriam utilizadas por uma aplicação FM, que está num nível mais abaixo na hierarquia de camadas do SIGA. Por exemplo, não faz sentido uma aplicação FM criar uma via. Este hash só está descrito aqui para fins de documentação.

Método	Descrição	Parâmetros	Utilização no FM
cancelarJuntada			
excluirMovimentacao			
receberEletronico			
processar			
pedirPublicacao			
urlEncodedFormFromMap			
vincularPapel			
hashCode			
cancelarPedidoBI			
reabrir			
podeAlterarSenha			
criarIdentidade			
marcarTudo			
registrarAcessoReservado			
publicarBoletim			
RegistrarAssinatura			
equals			
anexarArquivo			
buscarPorNumeroAssinatura			
bCorrigirCriacaoDupla			
atualizarWorkFlow			
excluirBI			
cancelar			
bloquearPessoa			
mostraDescricaoConfidencial			
atualizarPublicacao			
arquivarCorrente			
desapensarDocumento			
obterDocumentosNaoVaoBoletim			
im			
transferir			
conf??(hash)			
alimentaFilaIndexacao			
finalizarBI			
encerrar			
anotar			
obterDocumentosBoletim			
getClass			
comp??(hash)			
cancelarDocumento			
alterarCpModelo			
gravarBI			
criarVolume			
getConf			
desarquivar			
getComp			
descricaoConfidencialDoDocumento			
processarModelo			
mapFromUrlEncodedForm			
obterFormasDocumento			

definirSenhaDeIdentidade			
setDescricaoConfidencial			
criarVia			
obterMetodoPorString			
referenciarDocumento			
assinarMovimentacao			
numerarTudo			
mostraDescricaoConfidencialDoDocumento			
remeterParaPublicacao			
bCorrigirDataFimMov			
fechar			
canonicalizarHtml			
getProcessadorModeloJsp			
processadorModeloJsp??(hash)			
class??(hash)			
receber			
agendarPublicacaoBoletim			
cancelarMovimentacao			
redefinirNivelAcesso			
descricaoConfidencial			
toString			
apensarDocumento			
threadAlteracaoParcial??(hash)			
duplicar			
notificarPublicacao			
gravar			
obterPdfPorNumeroAssinatura			
assinarDocumento			
trocarSenhaDeIdentidade			
gravarModelo			
getThreadAlteracaoParcial			
configurarAcesso			
setComp			
obterListaModelos			
alterarIdentidade			
arquivarPermanente			
verificarAssinatura			
fixTableCols			
marcar			
main			
criarWorkflow			
cancelarIdentidade			
alterarSenhaDeIdentidade			
atualizarWorkflow			
incluirCosignatario			
bloquearIdentidade			
juntarDocumento			
refazerBi			
registrarDisponibilizacaoPublicacao			
setProcessadorModeloJsp			

5.3 - VARIÁVEIS DO HASH DOC - Classe ExDocumento.java

Este hash contém funções que acessam as informações dos documentos. São muito úteis as aplicações FM. Nos capítulos anteriores vimos como ter acesso (via data-model, macros e etc.) a dezenas de variáveis relativas aos documentos, e aqui, temos dezenas de funções que retornam dados e funções utilitárias que estendem as macros e o data-model do FM.

Método	Descrição	Parâmetros	Utilização no FM
conteudoBlobForm			
setExNivelAcesso			
hashCode			
getAnoEmissao			
isAssinadoPorTodosOsSignatarios			
getExMovimentacaoIndexacaoSet			
getLocalidadeStringMaiusculas	Retorna o nome da localidade em caixa alta, com base em getLocalidadeString()	Passados: Recebido: String	[#assign qqcoisa = doc. getLocalidadeStringMaiusculas ()?string/]\${qqcoisa} RIO DE JANEIRO
setExBoletimDocSet			
getNmDestinatario	Não foi encontrado na classe atual.		
numeracaoUnicaAutomatica	Não foi encontrado na classe atual.		
getSubscritorString			
setExTipoDocumento	Não foi encontrado na classe atual.		
setDescrDocumentoAI			
getDescrDocumentoAI			Erro no FM
orgaoUsuario			
setDestinatario			
dtRegDocDDMMYY			
getNumSequencia			
numUltimaVia			
getNumExtDoc			
getSiglaAssinatura			
setCadastrante			
numExtDoc			
nmDestinatario			
exNivelAcessoDoDocumento			
dtExtensoSemLocalidade			
getResumo			
editor			
setLotaCadastrante			
conteudoBlobPdf			
getForm			
getNumUltimaVia	Retorna o número da última via do documento	Passados: Recebido: Int	[#assign qqcoisa = doc.getNumUltimaVia()?string/] \${qqcoisa} 2
getEditor			
getFgEletronico			
isPublicacaoSolicitada			
dtDoc			
getTodosDocumentosFilhosSet			

idDoc			
setOrgaoUsuario			
setObsOrgao			
getNmLotacao	Retorna valor digitado para a lotação no campo ;Lotação;Localidade.	Passados: Recebido: String	[#assign qqcoisa = doc.getNmLotacao()?string/] \${qqcoisa} CSIS
setExMobilSet			
QRCode			
getReferencia			
data			
setNmDestinatario			
descrDocumentoAI			
dtDispUltimoAgendamento			
obsOrgao			
setNumExtDoc			
isPublicacaoBoletimSolicitada			
setExModelo			
conteudoBlobDoc2			
dtDocDDMMYYYY			
getNivelAcesso	Retorna a descrição do nível de acesso do documento definido no momento da criação do documento, desconsiderando as redefinições de nível.	Passados: Recebido: String	[#assign qqcoisa = doc.getNivelAcesso()?string/] \${qqcoisa} 20
setDtFechamento			
getByteCount			
siglaAssinatura			
dtExtenso			
getDtRegDocDDMMYY	Retorna a data de registro do documento no formato dd/mm/aa, por exemplo, 01/02/10	Passados: Recebido: String	[#assign qqcoisa = doc.getDtRegDocDDMMYY()?string/] \${qqcoisa} 12/04/2012
getClass			
conteudoBlobHtml			
setExClassificacao			
assinadoPorTodosOsSignatarios			
numSequencia			
getHtml			
getNmFuncaoSubscritor			
getLotaSubscritor			
dtYYYY			
expediente			
getNumUltimaViaNaoCancelada	Retorna o número da última via do documento	Passados: Recebido: Int	[#assign qqcoisa = doc.getNumUltimaViaNaoCancelada ()?string/] \${qqcoisa} 2
referencia			
getExNivelAcessoDoDocumento			
localidadeStringMaiusculas			
getDtFechamentoDDMMYY	Retorna a data de finalização do documento no formato dd/mm/aa	Passados: Recebido: String	[#assign qqcoisa = doc.getDtFechamentoDDMMYY ()?string/] \${qqcoisa}

			11/04/12
getUltimoVolume			
getExNivelAcesso			
sigla			
nmFuncaoSubscritor			
volume			
getConteudoBlobPdf			
setDescrDocumento			
numeroHtmlString			
setLotaDestinatario			
exMovimentacaoIndexacaoSet			
numAntigoDoc			
isVialnexistente			
getMobilGeral			
pdf			
conteudo			
getConteudoBlobPdfB64			
getConteudoBlobHtmlString			
numPaginas			
isExpediente			
getDtDoc			
isAssinado			
getNmFuncao	Retorna o valor digitado para a função no campo Função;Lotação;Localidade.	Passados: Recebido: String	[#assign qqcoisa = doc.getNmFuncao()?string/ \${qqcoisa}] Analista
dtRegDocDDMMYYHHMMSS			
getDtDocDDMMYY	Retorna a data do documento (campo da entrevista fixa) no formato dd/mm/aa	Passados: Recebido: String	[#assign qqcoisa = doc.getDtDocDDMMYY()?string/ \${qqcoisa}] 20/02/12
isExterno			
subscritorECosignatarios			
viasAdicionais			
setConteudoBlobHtml			
setOrgaoExterno			
titular			
getNmSubscritorExt			
getExModelo			
dtUltimaRemessaParaPublicacao			
conteudoTpDoc			
eletronico			
getTeste			
exMobilPai			
getConteudo			
getAssinaturaHtmlString			
numUltimoVolume			
setEletronico			
conteudoBlobResumo			
subscritor			
isBoletimPublicado			
getAssinantesCompleto			
todosDocumentosFilhosSet			
getLotaDestinatario			
exMobilSetInvertido			
setOrgaoExternoDestinatario			

getOrgaoExternoDestinatario			
setDtDoc			
nmMod			
getExFormaDocumento			
setExMobilPai			
todasAsAssinaturas			
orgaoExternoDestinatario			
exFormaDocumento			
getFechoHtmlString			
getDestinatarioString	Retorna a descrição do destinatário de um documento, conforme as seguintes regras, na seguinte ordem: Se foi definida uma pessoa destinatária, retorna a descrição dela com iniciais maiúsculas (getDescricaoIniciaisMaiusculas()) Ou então, se for definido um destinatário em campo livre, retorna o valor digitado Ou então, se for definida uma lotação destinatária, retorna a descrição dela Ou então, se for definido um órgão externo destinatário, retorna a descrição do órgão mais a observação sobre o órgão, se houver, ou apenas esta última, se não for selecionado órgão mas for definida descrição.	Passados: Recebido: String	[#assign qqcoisa = doc.getDestinatarioString()?string/] \${qqcoisa}
publicacaoSolicitada			
anoEmissao			
externo			
exTipoDocumento			
exDocAnterior			
getExMovimentacaoSet			
numUltimaViaNaoCancelada			
isIndexavel			
getConteudoBlobDoc2			
getNmMod			
aberturaHtmlString			
nmLotacao			
setAnoEmissao			
nmSubscritorExt			
getDescrFormaDoc			
setConteudoBlobPdf			
mensagem			
resumo			
pai			
getViasAdicionais			
assinaturasDigitais			
setNmArqDoc			
getExMobilSetInvertido			
lotaTitular			
getNumeroDePaginasParaInsercaoEmDossie			
getSetVias			
html			

assinado			
codigoCompacto			
conteudoBlobDoc			
dtD			
exClassificacao			
numExpediente			
via			
setConteudoBlobDoc			
codigoString			
nmLocalidade			
getLotaCadastrante			
getLocalidadeString	Retorna o nome da localidade (município) onde se encontra a lotação em que o documento foi produzido, caso não tenha sido digitado valor para a localidade no campo Função;Lotação;Localidade. A escolha da lotação para obtenção da localidade obedece à seguinte regra de precedência: >Lotação titular >Lotação subscritor; >Lotação cadastrante	Passados: Recebido: String	[#assign qqcoisa = doc.getLocalidadeString()?string/ \${qqcoisa}] Rio de Janeiro
destinatario			
getExDocAnterior			
setFgEletronico			
lotacao			
getDtExtenso	Retorna a data do documento por extenso no formato "Rio de Janeiro, 01 de fevereiro de 2010", por exemplo	Passados: Recebido: String	[#assign qqcoisa = doc.getDtExtenso()?string/ \${qqcoisa}] Rio de Janeiro, 20 de fevereiro de 2012.
descrCurta			
cadastrante			
nomeCompleto			
getTitular			
byteCount			
getDtRegDoc			
exMovimentacaoSet			
getDtDispUltimoAgendamento	Retorna a data de disponibilização da última movimentação do móbil geral, no formato dd/MM/yy. Obs.: não corresponde exatamente ao nome do método Parece estar em desuso, traz a data do sistema.	Passados: Recebido: String	[#assign qqcoisa = doc.getDtDispUltimoAgendamento()?string/ \${qqcoisa}] 12/04/12
getDtMMMM	Retorne o mês (da data) do documento por extenso.	Passados: Recebido: String	[#assign qqcoisa = doc.getDtMMMM()?string/ \${qqcoisa}] Fevereiro
getExDocumentoFilhoSet			
getConteudoBlobResumo			
getConteudoBlobDoc			
setNumExpediente			
setNumSequencia			
getPdf			

processo			
getAssinaturasDigitais			
publicacaoAgendada			
form			
fgEletronico			
getCadastrante			
equals			
assinantesCompleto			
descrClassifNovo			
setNumPaginas			
setTitular			
dtRegDoc			
getCodigoString	Retorna o código do documento. Se o documento for de origem externa, adiciona ao código do documento o código externo. Se for interno importado, adiciona ao código do documento o número antigo	Passados: Recebido: String	[#assign qqcoisa = doc.getCodigoString()?string/] \${qqcoisa} NOVO
setConteudoBlobResumo			
getNmLocalidade	Retorna valor digitado para a localidade no campo Função;Lotação;Localidade.		[#assign qqcoisa = doc.getNmLocalidade()?string/] \${qqcoisa} Niterói
getNumeroHtmlString			
contarNumeroDePaginas			
getDtExtensoSemLocalidade	Retorna a data do documento por extenso sem localidade, no formato "01 de fevereiro de 2010", por exemplo	Passados: Recebido: String	[#assign qqcoisa = doc.getDtExtensoSemLocalidade()?string/] \${qqcoisa} 20 de fevereiro de 2012.
getDtD			
getCodigo	Retorna o código do documento. Retorna com "/" e "-". Se o id do documento for nulo, então retorna NOVO	Passados: Recebido: String	[#assign qqcoisa = doc.getCodigo()?string/] \${qqcoisa} RJ-ANE-2012/00003
setConteudoBlob			
getIdDoc			
getExTipoDocumento			
publicacaoBoletimSolicitada			
getLotaTitular			
getSubscritor			
getConteudoTpDoc			
nmOrgaoExterno			
setNmOrgaoExterno			
getConteudoBlobHtmlB64			
getNome			
getSubscritorECosignatarios			
getQRCode			
setConteudoBlobDoc2			
getNumUltimoVolume			
getAberturaHtmlString			
setIdDoc			
setSubscritor			
descrFormaDoc			
conteudoBlobPdfB64			

isViaCancelada			
rascunho			
getCorpoHtmlString			
lotaCadastrante			
dtFechamento			
getNomeCompleto			
isRascunho			
getDescrCurta	Retorna a descrição (que aparece na entrevista fixa) do documento com no máximo 40 caracteres	Passados: Recebido: String	[#assign qqcoisa = doc.getDescrCurta()?string/ \${qqcoisa} Teste
setLotaTitular			
lotaSubscritorEfetiva			
mobilGeral			
getOrgaoExterno			
dtFechamentoDDMMYY			
getNumExpediente			
dtExtensoMaiusculasSemLocalidade			
numeroDePaginasParaInsercaoEmDossie			
getDtDocDDMMYYYY	Retorna a data do documento (campo da entrevista fixa) no formato dd/mm/aaaa	Passados: Recebido: String	[#assign qqcoisa = doc.getDtDocDDMMYYYY()?string/ \${qqcoisa} 20/02/2012
exModelo			
getReferenciaHtml			
setExDocAnterior			
subscritorString			
getExMobilSet			
isPublicacaoAgendada			
ultimoVolume			
getDestinatario			
getExBoletimDocSet			
getDescrClassifNovo			
hasPDF			
corpoHtmlString			
nivelAcesso			
lotaSubscritor			
isNumeracaoUnicaAutomatica			
getConteudoBlobForm			
getCodigoCompacto	Retorna o código do documento sem "-" ou "/"	Passados: Recebido: String	[#assign qqcoisa = doc.getCodigoCompacto()?string/ \${qqcoisa} RJANE201200003
setVias			
getDtFechamento	Retorna a data de finalização do documento no formato dd/mm/aaaa hh:mm:ss	Passados: Recebido: String	[#assign qqcoisa = doc.getDtFechamento()?string/ \${qqcoisa} 11/04/2012 18:46:22
getExMobilPai			
getDtYYYY	Retorna o ano que faz parte da data do documento no formato aaaa, por exemplo, 2010.		[#assign qqcoisa = doc.getDtYYYY()?string/ \${qqcoisa}]

			2012
getLotacao			
nmFuncao			
setExFormaDocumento			
setConteudoBlobHtmlString			
exDocumentoFilhoSet			
getConteudoBlob			
isProcesso			
codigo			
destinatarioString			
setConteudoTpDoc			
setConteudoBlobForm			
isCancelado			
localidadeString			
boletimPublicado			
setLotaSubscritor			
getNmSubscritor	Retorna o valor digitado para o nome do subscritor no campo Função;Lotação;Localidade;sub scritor.	Passados: Recebido: String	[#assign qqcoisa = doc.getNmSubscritor()?string/] \${qqcoisa} Ruben
conteudoBlobHtmlB64			
getLotaSubscritorEfetiva			
getVolume			
setNumAntigoDoc			
referenciaPDF			
class			
orgaoExterno			
nmArqDoc			
getContarNumeroDePaginas			
getReferenciaPDF			
getObsOrgao			
fechoHtmlString			
getDescrDocumento	Retorna a descrição (que aparece na entrevista fixa) completa do documento	Passados: Recebido: String	[#assign qqcoisa = doc.getDescrDocumento ()?string/] \${qqcoisa} Teste
getData	Obtém o timestamp do sistema	Passados: Recebido: Date no formato dd/mm/aaaa hh:mm:ss	[#assign datahoje = doc.getData()?substring(0,10)/] \${datahoje} [#assign horahoje = doc.getData()?substring(11,19)/] \${horahoje} 25/03/2012 17:50:30
setNmFuncaoSubscritor			
getDtExtensoMaiusculasSemLo calidade	Retorna a data do documento por extenso sem localidade e em maiúsculo no formato " 01 DE FEVEREIRO 2010", por exemplo	Passados: Recebido: String	[#assign qqcoisa = doc.getDtExtensoMaiusculasSemLo calidade ()?string/] \${qqcoisa} 20 DE FEVEREIRO DE 2012.
toString			
nome			
getTodasAsAssinaturas			
getDtAssinatura			
assinaturaHtmlString			
cancelado			

getNumPaginas			
getSigla	Retorna o código do documento	Passados: Recebido: String	[#assign qqcoisa = doc.getSigla()?string/] \${qqcoisa} NOVO
nmSubscritor			
getNumAntigoDoc			
getExClassificacao			
exMobilSet			
getConteudoBlobHtml			
isAssinadoDigitalmente			
conteudoBlobHtmlString			
exNivelAcesso			
setNmSubscritorExt			
dtDocDDMMYY			
exBoletimDocSet			
getNmOrgaoExterno			
getNmArqDoc			
setDescrClassifNovo			
descrDocumento			
referenciaHtml			
setDtRegDoc			
getDtUltimaRemessaParaPublicacao	Retorna a data da última movimentação do Mobil Geral, no formato dd/mm/yy, por exemplo, 12/10/10. Obs.: não está correspondendo exatamente ao nome do método Este método só pode ser testado fora do FM visto que para o móbil geral estar disponível o documento deve estar finalizado com assinatura, e o FM só tem controle durante a edição do documento.		[#assign qqcoisa = doc.getDtUltimaRemessaParaPublicacao()?string/] \${qqcoisa}
dtAssinatura			
getOrgaoUsuario			
getMensagem			
getPai	Retorna o documento pai, a partir do móbil pai		
isEletronico			
lotaDestinatario			
getDtRegDocDDMMYYHHMMS	Retorna a data de registro do documento no formato dd/mm/aa HH:mm:ss, por exemplo, 01/02/2010 16:10:00.		[#assign qqcoisa = doc.getDtRegDocDDMMYYHHMMSS()?string/] \${qqcoisa} 12/04/12 17:04:14
getArquivosNumerados			
assinadoDigitalmente			
dtMMMM			

5.4 – VARIÁVEIS DO HASH PARAM

Este hash não será utilizado pelas aplicações FM. Só está descrito aqui para fins de documentação.

Método	Descrição	Tipo	Utilização no FM
finalizacao		Enumerável	
processar_modelo		Enumerável	

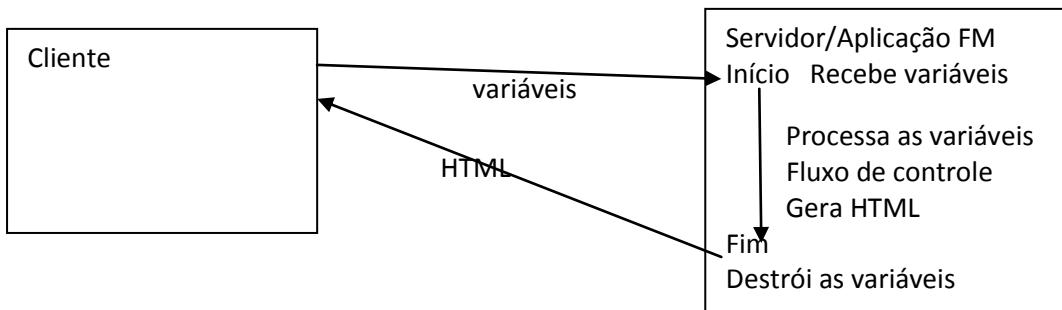
Aprendemos com o erro ... dos outros, porque o nosso é sofrimento. O fato de errar, achar uma solução, documentar e passá-la adiante é o que os americanos chamam de “Lessons Learned”.



*Nunca se esquecem as lições
aprendidas na dor*

6.1 Erros mais comuns no FM pelo não entendimento do funcionamento do mesmo

O grande problema de programar com FM (ou JSP, PHP ...) é que a aplicação roda no servidor, gera HTML (vamos chamar de tela), termina sua execução e envia a tela para o browser do cliente. Neste ponto não há mais comunicação entre a tela do usuário e a aplicação FM que rodou e terminou no servidor (Já falamos sobre isso, STATELESS). O ciclo se repete quando o usuário promove um submit, a tela é enviada ao servidor, o FM roda, gera ou não novo HTML, devolve ao cliente e termina. O FM é um mero gerador de HTML.



Darei um pequeno exemplo para entendermos como funciona uma aplicação FM e observarmos o que chamo de *dissincronia* entre o que você quer e o que acontece, ou seja, o que acontece no cliente e no servidor. Esta dissincronia não acontece na programação tradicional, seja cliente, servidor ou cliente x servidor, pois elas são STATEFULL, como mencionado na seção 1.4.

No exemplo vamos criar um campo oculto, que é uma técnica para manter uma variável entre sessões, visto que o FM inicia e termina, e portanto, não guarda os valores das variáveis criadas.

Pseudo-código

```
Se x não existe Então
    x = 0
Fim-se

Se x = 0 Então
    x = 1
    Passei aqui na primeira vez
Senão
    Passei aqui nas vezes subsequentes
Fim-se

Botão para submeter (... p/ testar)
```

FM

```
[#if (!x??)]
[@oculto valor="0" var="x"]
[/#if]

[#if (x) == "0"]
[#assign x = "1"]
Passei aqui na primeira vez
[#else]
Passei aqui nas vezes subsequentes
[/#if]

[@br/] <INPUT TYPE="submit" VALUE="Submeter">
```

Primeiro ponto, o código acima gerará um erro no segundo if, dizendo que x não está definido! Explicarei logo a seguir, porém, alterei o if para:

FM

```
...
[#if (x!="0") == "0"]
...
```

`(x != "0")`, quer dizer, se `x` não está definido assuma, NESTE PONTO, o default de 0. Comecei a rodar o código e as seguintes telas apareceram ...

Primeira rodada

Passei aqui na primeira vez

Submeter

Segunda rodada

Passei aqui na primeira vez

Submeter

Enésima rodada

Passei aqui na primeira vez

Submeter

Ou seja, não entrou nunca no else. Por quê? Nós setamos o `x` para 1!

Este diagrama tenta mostar a dissincronia.

	Primeira rodada		Segunda Rodada		Terceira rodada	
GET/POST (enviado)	nada existe		→ Get:url?X? 0		nada existe	
Processamento FM	HTML	Variáveis	HTML	Variaveis	HTML	Variáveis
	x não existe, então é criado o de statemente HTML pelo FM <input type=hidden name="x" value="0" /> É enviada a msg “Passei aqui na primeira vez”	x=nulo X continua a não existir, assume o default de 0, e então x é criado com valor de 1 x=0 (por default) x=1	x existe e nada é executado, ou seja, não temos mas o statemente HTML	X=0 x agora existe e possui o valor de 0, então atribui o valor de 1 a x x=1	x não existe, então é criado o statemente HTML pelo FM <input type=hidden name="x" value="0" /> envia a msg “Passei aqui na primeira vez”	X=nulo X continua a não existir, assume o default de 0 , então x é criado com valor de 1 x=0 (por default) x=1
Fim Processamento FM	HTML enviado	X foi destruído	HTML enviado	X foi destruído	HTML enviado	X foi destruído
Resultado (tela recebida)	<div style="border: 1px solid #ccc; padding: 5px; display: inline-block;"> x: 0 (escondido) </div> <div style="background-color: #f0f0ff; padding: 10px; border: 1px solid #ccc; border-radius: 5px; margin-top: 5px;"> Passei aqui na primeira vez <input type="button" value="Submeter"/> </div>		<div style="background-color: #f0f0ff; padding: 10px; border: 1px solid #ccc; border-radius: 5px; margin-top: 5px;"> Passei aqui na primeira vez <input type="button" value="Submeter"/> </div>		<div style="border: 1px solid #ccc; padding: 5px; display: inline-block;"> x: 0 (escondido) </div> <div style="background-color: #f0f0ff; padding: 10px; border: 1px solid #ccc; border-radius: 5px; margin-top: 5px;"> Passei aqui na primeira vez <input type="button" value="Submeter"/> </div>	
Interação do usuário	<div style="border: 1px dashed #ccc; padding: 5px; display: inline-block;"> x: 0 (escondido) </div> <div style="background-color: #f0f0ff; padding: 10px; border: 1px solid #ccc; border-radius: 5px; margin-top: 5px;"> Passei aqui na primeira vez <input type="button" value="Submeter"/> </div> Pressiona Submeter		<div style="background-color: #f0f0ff; padding: 10px; border: 1px solid #ccc; border-radius: 5px; margin-top: 5px;"> Passei aqui na primeira vez <input type="button" value="Submeter"/> </div> Pressiona Submeter		<div style="border: 1px dashed #ccc; padding: 5px; display: inline-block;"> x: 0 (escondido) </div> <div style="background-color: #f0f0ff; padding: 10px; border: 1px solid #ccc; border-radius: 5px; margin-top: 5px;"> Passei aqui na primeira vez <input type="button" value="Submeter"/> </div> Pressiona Submeter	

Observar que ora o input hidden é criado ora não, porque só colocamos a sua criação caso x não existisse.

Coisa tão simples, e não conseguimos resultado. Se fosse COBOL, c, pascal, vb, delphy, Java, não teríamos passado por isso.

6.1.1 - Variáveis não iniciadas: por que tivemos que mudar o código?

Mudamos De: [#if (x) == "0"] Para: [#if (x!="0") == "0"]

Na primeira rodada o FM gera o statement HTML <input type=hidden name="x" value="0" />, porém isto não quer dizer que a variável x tenha sido criada. Não, não foi, só o HTML foi criado.

Quando fazemos a referência [#if (x) == "0"] , o FM reclama (ERRO), informando que x não está definido.

Mais detalhes sobre como testar se uma variável existe ou não sobre default na seção a seguir "Manuseio de variáveis no FM: Existência".

6.1.2 - Estamos sempre rodando do zero: então é impossível saber se é a primeira rodada ou não?

A princípio sim, porém podemos criar condições de contorno:

Primeita tentativa:

Segunda tentativa:

<pre>[#if (!x??)] [@oculto var="x" valor="0"/] [#else] [@oculto var="x" valor="1"/] [/#if] [#if (x!="0") == "0"] [#assign x = "1"] Passei aqui na primeira vez [#else] Passei aqui nas vezes subsequentes [/#if] [@br/] <INPUT TYPE="submit" VALUE="Submeter"></pre>	<pre>[#if (!x??)] [@oculto var="x" valor="0"/] [#else] [@oculto var="x" valor="1"/] [/#if] [#if (x!="0") == "0"] [@atualizaoculto var="x" valor="1"/] Passei aqui na primeira vez [#else] Passei aqui nas vezes subsequentes [/#if]</pre>
--	--

Rodada	Não funcionou	Funcionou
1	Passei aqui na primeira vez <input type="button" value="Submeter"/>	Passei aqui na primeira vez <input type="button" value="Submeter"/>
2	Passei aqui na primeira vez <input type="button" value="Submeter"/>	Passei aqui nas vezes subsequentes <input type="button" value="Submeter"/>
3	Passei aqui nas vezes subsequentes <input type="button" value="Submeter"/>	Passei aqui nas vezes subsequentes <input type="button" value="Submeter"/>
N	Passei aqui nas vezes subsequentes <input type="button" value="Submeter"/>	Passei aqui nas vezes subsequentes <input type="button" value="Submeter"/>

A diferença nas duas tentativas, estúpida por sinal, está entre [#assign x = "1"] e [@atualizaoculto var="x" valor="1"/>].
[#assign x = "1"]
No FM, servidor, o <input type=hidden name="x" value="0"/> foi criado antes e depois ocorreu o assign x = "1". O FM enviou a tela e terminou, sendo que o novo conteúdo 1 foi para o espaço.

[@atualizaoculto var="x" valor="1"]
Esta é uma macro FM que roda um JS, obviamente NO CLIENTE. Este JS altera diretamente o valor de x em <input type=hidden name="x" value="0"/> para 1 via APIs DOM (apresentado no capítulo sobre "Programação Cliente - JS e outros"), ou seja, o HTML já existia no browser e foi aplicado um comando DOM do tipo:
document.getElementById("x").value="1"; alterando o seu valor em tempo real.

Este é o grande problema da dissincronia.

Observem que o código final FM ficou bem mais complexo que o idealizado e diferente do pseudo-código, sendo que este serviria como base para o processamento cliente, servidor ou cliente x servidor, porém não o WEB Tradicional - STATELESS

Digrama da segunda tentativa bem sucedida, incluindo o JS.

	Primeira rorada		Segunda Rodada		Terceira rodada	
GET/POST (enviado)	nada existe		Get:url?X? 1		Get:url?X? 1	
Processamento FM	HTML x não existe, então é criado o de statemente HTML pelo FM <code><input type=hidden name="x" value="0" /></code> E também é criado o statement HTML / JS <code><script type="text/javascript"> document.getElementById("x") .value="1"; </script></code> envia a msg <code>"Passei aqui na primeira vez"</code>	Variáveis x=nulo X continua a não existir, assume o default de 0, x=0 (por default)	HTML x existe e é criado o statement HTML <code>input type=hidden name="x" value="1" /></code> envia a msg <code>"Passei aqui nas vezes subsequentes"</code>	Variaveis x=1 x agora existe e possui o valor de 1,	HTML x existe e é criado o statement HTML <code>input type=hidden name="x" value="1" /></code> envia a msg <code>"Passei aqui nas vezes subsequentes"</code>	Variaveis x=1 x agora existe e possui o valor de 1,
Fim Processamento FM	HTML enviado	X foi destruído	HTML enviado	X foi destruído	HTML enviado	X foi destruído
Resultado (tela recebida)	x: 0 (escondido) <code>Passei aqui na primeira vez</code> <input type="button" value="Submeter"/>		x: 1 (escondido) <code>Passei aqui nas vezes subsequentes</code> <input type="button" value="Submeter"/>		x: 1 (escondido) <code>Passei aqui nas vezes subsequentes</code> <input type="button" value="Submeter"/>	
Processamento JS	O JS roda e atualiza o valor de x para 1, desta forma passamos a ter o seguinte statement HTML <code><input type=hidden name="x" value="1" /></code>					
Interação do usuário	<code>x: 1 (escondido)</code> <code>Passei aqui na primeira vez</code> <input type="button" value="Submeter"/> Pressiona Submeter		<code>x: 1 (escondido)</code> <code>Passei aqui nas vezes subsequentes</code> <input type="button" value="Submeter"/> Pressiona Submeter		<code>x: 1 (escondido)</code> <code>Passei aqui nas vezes subsequentes</code> <input type="button" value="Submeter"/> Pressiona Submeter	

6.1.3 - Outros problemas no processamento WEB Tradicional que observamos no SIGA-DOC

➤ Iniciar variáveis

Temos que ter muito cuidado, pois como era costume em outros ambientes de programação a estrutura clássica é:

No servidor (Main-frame e Minis)

Iniciar variáveis, apresentar tela, obter eventos (PFs), criticar e terminar

No ambiente cliente (PC) ou cliente (PC) x servidor (de Banco de Dados)

Iniciar variáveis, apresentar tela, criticar em loco, obter eventos (PFs) e terminar

Na programação FM (JSP, PHP ...), como ele sempre termina, ele volta a zero na próxima rodada, e consequentemente pode overridar um valor que o usuário digitou.

Iniciar variáveis, apresentar tela, obter eventos (PFs), criticar e terminar

Ex:

```
...
[#assign xpto = 2012/]
...
...
[@texto titulo="Ano" var="ano" largura="3" maxcaracteres="4" obrigatorio="Sim"
reler="sim" default="xpto"]
```

OK, na primeira vez que a tela for apresentada, o ano constará 2012. Porém, o usuário seta para 2010. O que acontecerá. No submit() o ano irá com 2010 (?ano?2010...), porém quando passar pelo assign, será substituído por 20120 novamente.

Contorno:

```
[#if (!xpto??)] // se não existe
    [#assign xpto = "2012"] // cria o default
[#else] // existe
    [#if xpto == "" || xpto == " " ] //é nulo ou branco
        [#assign xpto = "2012"] //mantém o default
    [/if]
[/if]
```

➤ Críticas

Outro ponto importante. As críticas no SIGA-DOC são passivas, no sentido que o usuário pode simplesmente seguir adiante. Lembre-se, não temos controle sobre a tela, mesmo utilizando algumas rotinas JS. Não tem como impedir o prosseguimento, a não ser que o campo seja definido como obrigatório, e neste caso, ao dar OK, o SIGA-DOC trava e não deixa passar.

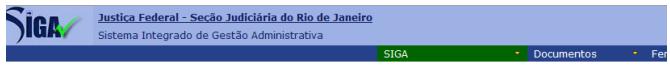
Soicitei ao Renato um bloco adicional, além do bloco da entrevista e do documento, um bloco de crítica. Como ele funcionaria?

```
[@critica]

[#assingn flg_critica = 0/]

[#if condicao1]
    [#assign flg_critica = 1/]
```

```
Msg1: qualquer coisa .....
[/#if]
[#if condicao2]
[#assign flg_critica = 1/]
Msg2: qualquer coisa .....
[/#if]
```



```
...
[msg1]
[msg2]
...
[/#if]

[/@critica]
```

O SIGA-DOC abriria uma nova janela, executaria o bloco critica, e se flg-critica fosse diferente de zero, então todas Msgs seriam exibidas nesta janela e o usuário não poderia seguir adiante. O usuário poderia ficar com a janela de erros e da entrevista abertas lado-a-lado.

Outro aspecto importante é que o código do bloco entrevista fica livre destes trechos de código, facilitando a documentação.

➤ ***Não temos controle sobre a tela, teclas digitadas ...***

Infelizmente não temos controle sobre estes eventos. Para contornar, poderíamos criar um JS que faria o papel de listener e nos avisaria sobre os eventos. Mas isso não é tão trivial, pois muitos campos já possuem eventos associados tais como: onblur, onchange e etc.

➤ ***Se o FM termina e destrói as variáveis, como podemos manter variáveis de trabalho (como contadores, por exemplo) entre as sessões?***

É outro ponto fraco do STATELESS. Imaginem, tenho um contador, de funcionários por exemplo, e desejo manter cont_func = 17. Se deixar ao encargo do FM, quando ele terminar, já era. Para isso, utilizamos a macro @oculto para guardar os valores destas variáveis de trabalho, ou seja, utilizar o próprio HTML como container de variáveis de trabalho. Ver a seção mais baixo intitulada "Mantendo as variáveis entre sessões - Burlando o SATELESS"

6.2 – Manuseio de variáveis no FM: Existência

No FM, diferentemente do JSP, se fizermos uma referência (via If, via \${} ...) a uma variável não iniciada ganharemos uma exception. Muitas vezes o usuário está preenchendo os campos da entrevista e pede para visualizar o modelo, para ver como está ficando o documento. Ora, se determinados campos da entrevista não foram ainda preenchidos, um erro do FM será exibido na geração do documento.

Como podemos nos livrar deste tipo de erro e ficarmos com um ambiente parecido com o JSP?

Utilizando o Default operator !

```
[#assign antigoCPF2 = cpfteste!"6666666666"?string/]
```

Se cpfteste não existir, será assumido o default e depois convertido para string. Podemos simplesmente escrever cpfteste!?string, visto que o FM assume ! como ! "", ou seja, empty (vazio). Observação, vazio aqui não é nulo. O FM odeia nulo.

Quando devemos usar parêntesis na variável?

Se a variável for oriunda de classes onde podemos ter qualificações tipo: **doc.destinatario** da classe doc, ou **doc.destinatario.cpfPessoa** da classe doc/pessoa ou **doc.destinatario.cargo.nomeCargo** das classes doc/pessoa/cargo.

Qual a diferença?

```
[#assign antigoCPF = doc.destinatario.cpfPessoa!"6666666666"?string/]
```

Aqui, se o cpfPessoa não existir a expressão funcionará OK, porém se destinatario não existir, então um erro de undefined será gerado. Ou seja, em variáveis qualificadas, somente se a última não existir é que um erro não será gerado.

X

```
[#assign antigoCPF = (doc.destinatario.cpfPessoa)!"6666666666"?string/]
```

Aqui, tanto faz, ou seja, se destinatário e/ ou cpfPessoa não existirem, um erro não será gerado.

Regra Geral: utilizar sempre o formato (variavel)!, ou seja, a variável entre parêntesis.

Vejamos o que diz o manual:

You can use this operator in two ways with non-top-level variables:

```
product.color!"red"
```

This will handle if color is missing inside product (and returns "red" if so), but will not handle if product is missing. That is, the product variable itself must exist, otherwise the template processing will die with error.

```
(product.color)!"red"
```

This will handle if product.color is missing. That is, if product is missing, or product exists but it does not contain color, the result will be "red", and no error will occur. The important difference between this and the previous example is that when surrounded with parentheses, it is allowed for any component of the expression to be undefined, while without parentheses only the last component of the expression is allowed to be undefined.

6.3 – Visibilidade das variáveis no FM

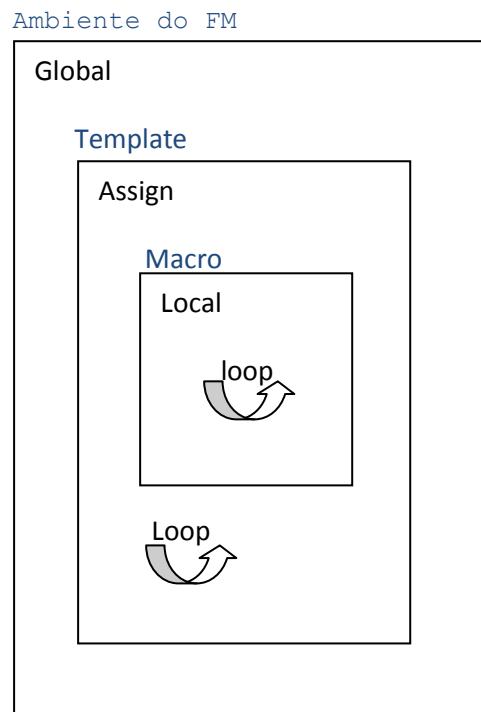
No FM as variáveis seguem a seguinte hierarquia:

- Variáveis de loop
- Variáveis criadas com local

- Variáveis criadas com assign
- Variáveis criadas com global

É importante ressaltar que uma variável de loop não sobrescreve (override) uma variável criada com assign, por exemplo, ela simplesmente esconde (hide) por um tempo, ou seja, pelo tempo de duração do loop.

É importante ressaltar que uma variável local não sobrescreve (override) uma variável criada com assign, por exemplo, ela simplesmente esconde (hide) por um tempo, ou seja, pelo tempo da execução da macro ou function.



O exemplo a seguir ilustra bem o conceito.

```

[#global x = "é global"]

[#assign x = "plain"]

1. ${x}  [##-- we see the plain var. here --]

[@test/]

6. ${x}  [##-- the value of plain var. was not changed --]

[#list ["loop"] as x]

7. ${x}  [##-- now the loop var. hides the plain var. --]
[#assign x = "plain2"] [##-- replace the plain var, hiding does not mater here --]

8. ${x}  [##-- it still hides the plain var. --]

[/#list]

9. ${x}  [##-- the new value of plain var. --]

10. ${.globals.x}
  
```

```

[#macro test]
2. ${x}  [##-- we still see the plain var. here --]

[#local x = "local"]

3. ${x}  [##-- now the local var. hides it --]

[#list ["loop"] as x]

4. ${x}  [##-- now the loop var. hides the local var. --]

[/#list]
5. ${x}  [##-- now we see the local var. again --]
[/macro]

```

Que produzirá o seguinte resultado:

```

1. plain
2. plain
3. local
4. loop
5. local
6. plain
7. loop
8. loop
9. plain2
10. é global

```

6.4 – Erros mais comuns no FM envolvendo nomes de variáveis

Cientes de: a visibilidade das variáveis no FM, visto no item anterior, e que as macros no nosso ambiente são incorporadas via include, e não via import (ver capítulo “Melhores Práticas - include x import”), podemos ocorrer nos seguintes erros.

6.4.1 – Macros com assign

Chamei uma macro, que não conheço a sua implementação (e nem deveria, pois ela está inserida na biblioteca de macros), e ela bugou a minha aplicação!!! Tenho que conhecer os nomes de todas as variáveis das macros que chamo?

```

[#assign x = 2/]
[@macroxpto/]
${x} {@br/>
${y!}

```

Que produzirá o seguinte resultado:

```

3
3

```

Biblioteca De Macros

```

[#macro macroxpto]
[#assign x = 3/]
[#assign y = 3/]
[/#macro]

```

Neste caso, a chamada da macro @macroxpto sobrescreveu a variável X e ainda deixou um legado, a variável Y.

No capítulo “[Palavras Reservadas](#)” existe uma relação de **nomes de variáveis tipo assign, dentro de macros ou functions, que não devem ser usadas para evitar**

problemas. Ver o capítulo sobre "Melhoras Práticas" para solucionar este problema.

6.4.2 - Variável com o mesmo nome da macro ou function

Para o FM as macros e functions são variáveis que desempenham um papel especial, desta forma, devemos evitar coincidência de nomes entre nomes das macros / functions e variáveis em geral.

Tenho que conhecer os nomes de todas macros???

```
[#assign texto = 'abc'/]
```

Algumas linhas depois ...

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

Que produzirá o seguinte resultado:

```
on line 12, column 1 in RubenTeste: texto is not a user-defined directive. It is a  
freemarker.template.SimpleScalar
```

```
-----  
==> user-directive texto [on line 12, column 1 in RubenTeste]
```

Neste caso, declaramos uma variável com o nome de texto e posteriormente no código invocamos a macro texto. Conflitos de tipos na variável. No capítulo "[Palavras Reservadas](#)" existe uma relação de **nomes de variáveis que não devem ser usadas** para evitar problemas. Ver o capítulo sobre "Melhoras Práticas" para solucionar este tipo de problema.

6.4.3 - Variável com o mesmo nome dos campos do formulário de entrevista do SIGA-DOC



ATENÇÃO

```
[@texto titulo="Email" var="eletronico" largura="30"  
maxcaracteres="30" obrigatorio="Sim" reler="ajax"  
idAjax="anoAjax"/]
```

eletronico é um nome de campo do formulário do SIGA-DOC. Este simples programa faz o browser voar (tela em branco), literalmente, sem qualquer mensagem de erro, quando qualquer campo que provoca a releitura da tela (destinatário, classificação ...) é preenchido.

Ver capítulo "DISSECANDO O FORMULÁRIO DE ENTREVISTA", seção "[NOMES por ordem de aparição e elementos HTML](#)". Ver o capítulo sobre "Melhoras Práticas" para solucionar este tipo de problema.

6.4.4 - Ids com o mesmo nome que os IDs do formulário de entrevista do SIGA-DOC

```
[@grupo titulo="Teste com Ids iguais"]  
[@texto var="emaildestino" titulo="Destinatário" largura="14"  
maxcaracteres="14" reler="ajax" idAjax="destinatario"]  
[/@grupo]
```

```

[@grupo depende="destinatario"]
[#if (emaildestino!="") = ""]
    [@mensagem titulo="Alerta" texto="Destinatario deve ser preenchido."
        vermelho=true/]
[/#if]
[/@grupo]

```

O ID que é passado no idAjax não deve conflitar com os Ids do formulário. Ver lista dos Ids no capítulo "DISSECANDO O FORMULÁRIO DE ENTREVISTA", seção "[IDs por ordem de aparição e elementos HTML](#)". Por quê? Porque não podemos ter 2 elementos com o mesmo ID no HTML, e se isto ocorrer, o AJAX não funcionará porque a API DOM emitida pelo JS do AJAX, `document.getElementById("id")`, espera receber somente 1 retorno, e se receber mais de 1, a API gera uma exception.

Por sorte, quando o SIGA-DOC gera o bloco grupo depende, verdade uma <div>, ele concatena o ID passado com a div. Ver o HTML gerado:

```
<div id="divdestinatario" depende=";destinatario;">.
forma, não temos que nos preocupar com esta questão,
fica o alerta.
```



6.5 – Criando e manipulando Sequences no FM

As sequences, como outros tipos de collections, não podem ser criadas (ou alteradas) dinamicamente. Se for necessário criar um vetor ou array dinamicamente, pode-se utilizar a técnica descrita no item 5.5 ou utilizar uma API Java.

```
<#assign pessoas = [
    {"name":"maria", "weight":60.45},
    {"name":"Barbara", "weight":53},
    {"name":"zeppelin", "weight":-200},
    {"name":"aardvark", "weight":30},
    {"name":"beetroot", "weight":0.3}
]>
```

O que geraria o seguinte **hash**:

```
(root)
|
|
+- pessoas
  |
  +- (1st)
  |   |
  |   +- name = "maria"
  |   |
  |   +- weight = 60.45
  +- (2nd)
  |
  ...
  |
  +- (3rd)
  ...
  ...
  ...
```

```
Desta forma, pessoas[0].name nos forneceria "maria"
```

Outro Exemplo:

```
<#assign membros = [
    {"name": {"first": "maria", "last": "silva"}, "age": 40},
    {"name": {"first": "Fred", "last": "Crooger"}, "age": 35},
    {"name": {"first": "Amanda", "last": "Fox"}, "age": 25}

]>
```

O que geraria o seguinte **hash**:

```
(root)
|
|
+- membros
  |
  +- (1st)
    |
    +- name
      |
      +- first = "maria"
      |
      +- last = "silva"
      |
      +- age = 40
  |
  +- (2nd)
  |
  ...
  |
  +- (3rd)
  ...
  ...
  ...
```

```
Desta forma, membros[0].name.first forneceria "maria"
```

Obs: Listando as Sequences com List

```
<#list sequence as item>
  ...
</#list>

<#assign seq = ["winter", "spring", "summer", "autumn"]>
<#list seq as x>
  ${x_index + 1}. ${x}<if x_has_next></if>
</#list>
```

O que produziria o seguinte resultado:

1. winter,
2. spring,
3. summer,
4. autumn

6.6 – Criando variáveis ("e Arrays") dinamicamente

Muitas vezes é necessário criar variáveis dinamicamente, em tempo de execução, para guardar valores que são digitados pelos usuários. Supor que para cada funcionário o usuário possa fornecer "N" motivos de faltas. As variáveis poderiam ser criadas conforme o seguinte padrão: matricula + i, onde o + aqui se refere à concatenação e i pode assumir valores de zero a N, desta forma rj1328437 representaria o campo contendo o 37º motivo de falta do funcionário rj13284.

Exemplo de código

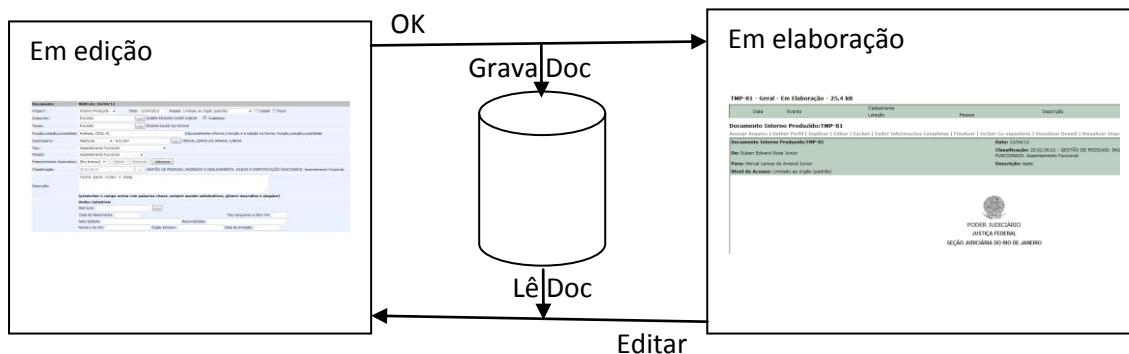
```
[#assign i = 1/]
[#assign var = "xpto"+i/]
[#assign inlineTemplate = "[#assign ${var} = 10/]"?interpret /]
[@inlineTemplate/]
${var} = ${.vars[var]}
```

O que produzirá:

```
xpto1 = 10
```

6.7 – Mantendo as variáveis entre sessões – Burlando o SATELESS

Quando saímos de um ambiente (Em edição) para outro (Em elaboração) e voltamos para o anterior (Em edição) não perdemos o conteúdo da tela, como seria o esperado. Isto se deve ao fato de que o SIGA-DOC persiste as informações da tela em BD, ou seja, salva o documento.



E as informações que não são campos da tela, porém exercem papel importante na minha aplicação, como contadores, por exemplo? O que fazer?

Armazenando-as no formulário HTML, via atributo Hide (ou macro @oculto, como visto anteriormente), ou seja, usamos o formulário como meio de persistência, porém as ocultamos para que o usuário não as veja.

```
[@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".

```
Aplicação FM
...
[# assign y = 200/]
[@oculto var="x" valor="100"]
...
```

A aplicação terminará e o valor

```
Página HTML
...
<input type="hidden" value="100" name="x" />
...
```

Porém o valor de x será preservado

De y será perdido.

6.8 – Geração do PDF – Problema com as Fontes

O gerador de PDF atual (antigo), NHEENGA, apresenta problemas na mudança da fonte do código HTML, como o tamanho por exemplo. O problema é que ele não respeita o que pedimos, mantendo muitas vezes o tamanho default.

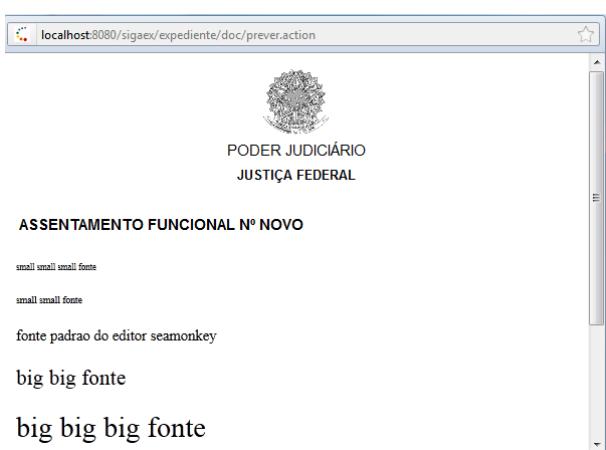
Este pequeno teste mostra a diferença entre a geração do HTML e do PDF.

```
[@documento]
[#assign texto_assentamento_funcional]
<html>
<head>
<meta content="text/html; charset=ISO-8859-1"
http-equiv="Content-Type">
<title></title>
</head>
<body>
<small><small><small>small small small fonte</small></small>&nbsp;
<br>
<br>
<small><small>small small fonte </small></small><br>
<br>
fonte padrao do editor seamonkey<br>
<br>
<big><big>big big fonte</big></big><br>
<br>
<big><big><big>big big big fonte</big></big></big><br>
<br>
<br>
</body>
</html>
[/#assign]
[@assentamento_funcional texto=texto_assentamento_funcional /]
[/@documento]
```

PDF gerado



HTML gerado



Ainda não testei o novo gerador de PDF, PDF4ML, e talvez estes problemas tenham sido sanados.

6.9 – Geração HTML – problemas com Tabelas

Comportamento esperado

Tabela 1		
Tabela 2		

Comportamento indesejado

Tabela 1			Tabela 2	

No Internet Explorer existe o seguinte bug: quando criamos tabelas, que se alinham uma debaixo da outra, por default, ele coloca a segunda tabela ao lado da primeira.

Para fugir deste bug, coloque a seguinte regra CSS no elemento (tabela2) que está com problema: `<table style="float:none; clear:both;" width="100%" ...>`

O float:none evita que o elemento flutue. Clear=both, both na verdade porque é left e right, faz com que o elemento não faça o overlap com qualquer elemento flutuante a sua esquerda ou direita. Acredito que o clear=both não seja necessário, a não ser que ele fosse definido na tabelal.

6.10 – Macros FM e a opção reler, relertab e obrigatorio

Um dos grandes problemas nas macros da entrevista é, referente aos parâmetros mencionados no título desta seção, que não estão ainda padronizados, pelo menos referente a macro @texto. Para tanto, segue uma colinha para utilizar esta opção.

Macro	Opções de reler	Opções de relertab	Opções obrigatorio
texto	reler="ajax" reler="sim" reler="nao" Obs: se reler="ajax" então assume-se AJAX, mesmo que que o idAjax for "". [#if reler == 'ajax'] [#local jreler = " onchange=\\"javascript: sbmt('"+ idAjax + "');\\\""] [/#if] [#if reler == 'sim'] [#local jreler = " onchange=\\"javascript: sbmt();\\\""] [/#if] -----DEFAULT----- reler="" Ou seja, não. Se passarmos	relertab="sim" relertab="nao" [#if relertab == 'sim'] [#local jrelertab = " onblur=\\"javascript: sbmt();\\\""] [/#if] -----DEFAULT----- relertab=""	obrigatorio="Sim" obrigatorio="nao" Observar o Sim com maiúscula na primeira letra. -----DEFAULT----- obrigatorio="nao"

	reler="abacaxi", ou reler="", ou qualquer outro string, será considerado como não.		
Checkbox	reler=true reler=false Obs: se reler=true e idajax for passado, então assume-se AJAX. <pre>[#if reler == true && idAjax != ""] [#local jreler = " sbmt('" + idAjax + "');\""] [#elseif reler == true] [#local jreler = " sbmt();\""] [/#if]</pre> -----DEFAULT----- reler=false Ou seja, não.		obrigatorio=true obrigatorio=false -----DEFAULT----- obrigatorio=false
radio	reler=true reler=false Obs: se reler=true e idajax for passado, então assume-se AJAX. <pre>[#if reler == true && idAjax != ""] [#local jreler = " sbmt('" + idAjax + "');\""] [#elseif reler == true] [#local jreler = " sbmt();\""] [/#if]</pre> -----DEFAULT----- reler=false Ou seja, não.		
data	reler=true reler=false Obs: se reler=true e idajax for passado, então assume-se AJAX. <pre>[#if reler == true && idAjax != ""] [#local jreler = " sbmt('" + idAjax + "');\""] [#elseif reler == true] [#local jreler = " sbmt();\""] [/#if]</pre> -----DEFAULT----- reler=false Ou seja, não.		obrigatorio=true obrigatorio=false -----DEFAULT----- obrigatorio=false
selecao	reler=true reler=false Obs: se reler=true e idajax for passado, então assume-se AJAX. <pre>[#if reler] onchange="javascript:</pre>		

	<pre>sbmt([#if idAjax != ""]'\${idAjax}'[/#if]);" [/#if] -----DEFAULT----- reler=false Ou seja, não.</pre>		
memo	<p>reler=true reler=false</p> <p>Obs: não tem opção AJAX</p> <pre>[#if reler == true] [#local jreler = " onchange=\"javascript: sbmt();\""] [/#if]</pre> <p>-----DEFAULT----- reler=false Ou seja, não.</p>	 NÃO IMPLEMENTADO	obrigatorio=true obrigatorio=false -----DEFAULT----- obrigatorio=false
caixaSel ecao As macros abaixo chamam a macro @seleciona vel que por sua vez chama a macro @caixaSele cao, e portanto seguem a mesma regra que ela: pessoa lotacao funcao	<p>reler=true reler=false</p> <p>Obs: se reler=true e idajax for passado, então assume-se AJAX.</p> <pre>[#if reler] //window.alert("vou reler tudo!"); document.getElementsByName('re q\${var}\${tipoSel}Sel')[0].valu e = "sim"; document.getElementById('alter ouSel').value='\${var}'; sbmt([#if idAjax != ""]'\${idAjax}'[/#if]); [/#if]</pre> <p>-----DEFAULT----- reler=false Ou seja, não.</p>	<p>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.</p> NÃO IMPLEMENTADO <p>-----DEFAULT----- relertab=""</p>	obrigatorio=true obrigatorio=false -----DEFAULT----- obrigatorio=false

Observação: Em breve vamos padronizar a macro @texto para que a mesma tenha os mesmos parâmetros das outras macros, porém mantendo a compatibilidade com o legado.

Macro	Opções de reler	Opções de relertab	Opções obligatorio
texto	<p>reler="ajax" reler="sim" ou true reler="nao" ou false</p> <p>Obs: se reler="ajax" e idajax for passado, então assume-se AJAX.</p> <pre>[#if reler == 'ajax' (reler == true && idAjax !=</pre>	<p>relertab="sim" ou true relertab="nao" ou false</p> <pre>[#if relertab == 'sim' relertab == true] [#local jrelertab = " onblur=\"javascript: sbmt();\""] [/#if]</pre> <p>-----DEFAULT-----</p>	<p>obrigatorio="sim" ou true obrigatorio="nao" ou false</p> -----DEFAULT----- <p>obrigatorio="nao"</p>

```

        """)]
[#local jreler = "
onchange=\"javascript:
sbmt('" + idAjax + "');\""]
[/*elseif reler == 'sim' ||
reler == true]
[#local jreler = "
onchange=\"javascript:
sbmt();\""]
[/*if]

-----DEFAULT-----

reler=false
Ou seja, não.

```

6.11 – Função JS com o mesmo nome de um campo HTML

Supor a seguinte tag HML:

```
<input type=text name="executar_APLICACAO" ...>
```

E o seguinte script JS:

```
<script ...
function executar_APLICACAO() ...
...
</script>
```

O que acontece? A função simplesmente não roda. Abrindo o console o JS diz que **executar_APLICACAO** não é uma função. Este erro só pode ser visto na console JS. Conclusão, deve-se mudar o nome da função ou do campo HTML.

6.12 – Retirando espaços em branco indesejados com #compress

Para que serve o **#compress**: ele é útil para remover supérfluos whitespaces (espaços em branco), ou seja, substitui “N” whitespaces por somente 1 whitespace.

Observação: o servlet JAVA que carrega o template FM inclui um **[#compress]** no início e um **[/#compress]** no final do template.

O exemplo abaixo mostrará o funcionamento:

```
<#assign x = "    moo  \n\n  ">

(<#compress>
 1 2 3      4      5
 ${x}
 test only

 I said, test only

</#compress>)
```

Que produzirá o seguinte efeito no texto entre o **#compress**: observar que os espaços entre os números também foram comprimidos para somente 1.

```
(1 2 3 4 5
moo
test only
I said, test only)
```

Exemplo aplicado:

Trecho de código do aplicativo Inclusão no Banco de Permutas: no exemplo abaixo, se não colocarmos o #compress a janela JS é aberta e nenhum texto é apresentado, sendo que nenhum erro é reportado. Qual é a lógica por trás deste problema? A única razão é que pode existir caracteres não visíveis na linha, o que prejudicará a exibição do campo, a outra pode ser mesmo o excesso de caracteres em branco.

```
...
[#assign conteudo11]
O JUIZ FEDERAL - DIRETOR DO FORO E CORREGEDOR PERMANENTE DOS SERVIÇOS AUXILIARES DA JUSTIÇA FEDERAL - SEÇÃO JUDICIÁRIA DO RIO DE JANEIRO, no uso de suas atribuições legais e com o objetivo de tornar o processo de lotação e remoção mais célere, RESOLVE: <br/>&lt;br/>Estabelecer os seguintes critérios a serem observados pelos servidores interessados em integrar o Banco de Permutas da Seção Judiciária do Rio de Janeiro:<br/>&lt;br/>
[/#assign]

[#assign conteudo12]
I- a inscrição do servidor implica a sua disponibilidade imediata para a realização da permuta pretendida. Caso o servidor não concorde em realizá-la à época em que surgir a possibilidade de atendimento, sua inscrição no Banco será excluída, automaticamente. Não haverá óbice para que o servidor faça uma nova inscrição, posteriormente.<br/>&lt;br/>
[/#assign]

[#assign conteudo13]
II- na hipótese de o servidor se inscrever para mais de uma unidade de lotação, quando houver o atendimento de uma delas, automaticamente, será excluído do Banco de Permutas, sem prejuízo para que proceda a uma nova inscrição.<br/>&lt;br/>
[/#assign]

[#assign conteudo14]
III- a movimentação dos servidores inscritos no Banco de Permutas dar-se-á de acordo com a ordem cronológica dos pedidos na Subsecretaria de Gestão de Pessoas.<br/>&lt;br/>
[/#assign]

[#assign conteudo15]
IV- Esta Portaria entra em vigor na data de sua publicação.<br/>&lt;br/>
[/#assign]

[#assign conteudo2]
Art. 12. Servidores ocupantes de cargos efetivos com especialidade somente poderão ser lotados em unidades organizacionais diretamente ligadas às atribuições dos cargos, exceto na hipótese de designação para função comissionada de chefia (FC-04 ou superior) ou para cargo em comissão de Direção ou Assessoramento.<br/>
[/#assign]

[#assign texto_ciencia]
<b>Estou ciente da
<span onmouseover="this.style.cursor='hand';" onclick="javascript: try {newwin.close();} catch(err) {}
newwin =
window.open('teste',null,'height=450,width=550,status=no,toolbar=no,menubar=no,location=no');
newwin.document.write('[#compress]${conteudo11}[/#compress]');
newwin.document.write('[#compress]${conteudo12}[/#compress]');
newwin.document.write('[#compress]${conteudo13}[/#compress]');
newwin.document.write('[#compress]${conteudo14}[/#compress]');
newwin.document.write('[#compress]${conteudo15}[/#compress]');
">
<u>Portaria nº 101/2010</u></span>, de 10/11/2010,      do

<span onmouseover="this.style.cursor='hand';" onclick="javascript: 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('[#compress]${conteudo2}[/#compress]');
">
<u>caput do art. 11 da portaria nº 32/2011</u> </span>, de 07/06/2011 e de que a desistência da inclusão no Banco de Permutas deverá ser comunicada à Subsecretaria de Gestão de Pessoas por meio de requerimento.
[@br/]
[/#assign]
...
```

Observação: no inicio tentamos colocar o #compress na variável, como demonstrado abaixo, porém também não funcionou, ou seja, o #compress tem que ser utilizado no LOCAL DO USO.

```
...
[#assign conteudo14]
[#compress]
III- a movimentação dos servidores inscritos no Banco de Permutas dar-se-á de acordo com a ordem
cronológica dos pedidos na Subsecretaria de Gestão de Pessoas.<br/><br/>
[/#compress]
[/#assign]
...
newwin.document.write('[#compress]${conteudo14}[/#compress]');
...
```



6.13 – Hora do sistema

A versão do FM que temos instalada não possui a hora do sistema, para tanto pode-se utilizar o método getData() do hash doc.

```
[#assign datahoje = doc.getData()?substring(0,10)]
${datahoje}
[#assign horahoje = doc.getData()?substring(11,19)]
${horahoje}
```

25/03/2012 17:50:30

Nas versões mais novas do FM temos o .now (que possui o timestamp, ou seja, data/hora)

```
 ${.now}
"Today is ${.now?date}"
"The current time is ${.now?time}"
```

6.14 – Try / Catch no FM

Será que temos o equivalente
Sim, temos o #attempt /

```
try {
// do something
} catch (error)
// about errors
}
```

ao try / catch do JAVA e JS no FM?
#recover.

Sintaxe:

```
<#attempt>
  attempt block
<#recover>
  recover block
</#attempt>
```

Onde:

#attempt block: bloco com conteúdo que será sempre executado, porém se algum erro ocorrer durante o processamento deste bloco, todo output sofrerá roll back e o recover block será executado.

#recover block: o recover é mandatório, e seu conteúdo será executado se ocorrer um erro no bloco attempt.

O blocos #attempt / #recover podem ser aninhados.

Exemplo:

```
... Conteúdo primário
<#attempt>
  Conteúdo principal: ${thisMayFails}
```

```
<#recover>
  Ops! Conteúdo opcional ou de exceção.
</#attempt>
... Conteúdo primário continuado
```

Se a variável thisMayFails não existe, então o output será:

```
... Conteúdo primário
  Ops! Conteúdo opcional ou de exceção.
... Conteúdo primário continuado
```

Se a variável thisMayFails existir seu valor é 123, então o output será:

```
... Conteúdo primário
  Conteúdo principal: 123
... Conteúdo primário continuado
```

6.15 - Não quero que o FM interprete certos trechos de código #noparse

Passei por uma situação em que não desejava que o FM interpretasse o código pois tinha uma rotina JS que escrevia código FM, e o FM EXECUTAVA O COMANDO FM!!!.

Este era o código:

```
...
Código FM
...
<script type="text/javascript" language="javascript">
...
function insereDump()
{
var blocoNotasJs = document.getElementById('fm_aplicacao').value;
var meuArray = blocoNotasJs.split('\n');
var tamBlocoNotas = meuArray.length;
// inicia variáveis
var meuArray2 = new Array();
var j = 0;
var flagAntes = "N";
var flagDepois = "N";
var linhaString = "";
var abremacro = "["
...
var dumpDepoisString = "[@dumpvardepois/]";
...
</script>
```

Esta macro era executada.

Este é o comportamento do FM, ou seja, achou instrução (seja dentro de FM, HTML, JS ...) ele executa

Na época em que não conhecia um comando para evitar este problema realizei o seguinte workaround:

```
var abremacro = "["
var dumpDepoisString = abremacro.concat('@').concat('dumpvardepois').concat('/');
```

Que na verdade poderia ser escrito de forma mais elegante:

```
var dumpDepoisString = "[#noparse] [@dumpvardepois/] [/#noparse]";
```

6.16 - Variáveis HTML

Devemos ter em mente que as aplicações que desenvolvemos não são feitas para nós, e sim, para os outros, ou seja, não adianta desenvolver uma aplicação que somente você saiba mantê-la. Aliás, mesmo quando desenvolvemos uma aplicação e levamos três ou mais meses para executar uma manutenção, sentimos dificuldades em saber a lógica por trás do código. Outro ponto importante é que as melhores práticas permite desenvolvermos um código mais robusto, menos sujeito a bugs.



7.1 – Nome de Variáveis

Primeira regra: utilize um prefixo na nomeação das variáveis das aplicações, macros ou functions, desta forma, será praticamente impossível ter colisões de nomes entre variáveis e nomes de macros / functions. É prolixidade? Sim, porém é mais seguro.

Se, por exemplo, a aplicação que está sendo desenvolvida é Assentamento Funcional Importado, pode-se utilizar o acrônimo afi_, como prefixo, para nomear as variáveis da aplicação.

Fora o prefixo, a nomeação das variáveis poderia seguir o recomendado pela SUN / ORACLE conforme o paper Code Conventions / Naming Conventions.

Variables	<p>Except for variables, all instance, class, and class constants are in mixed case with a lowercase first letter. Internal words start with capital letters. Variable names should not start with underscore _ or dollar sign \$ characters, even though both are allowed.</p> <p>Variable names should be short yet meaningful. The choice of a variable name should be mnemonic- that is, designed to indicate to the casual observer the intent of its use. One-character variable names should be avoided except for temporary "throwaway" variables. Common names for temporary variables are i, j, k, m, and n for integers; c, d, and e for characters.</p>	<pre>int i; char c; float myWidth;</pre>
Constants	<p>The names of variables declared class constants and of ANSI constants should be all uppercase with words separated by underscores ("_"). (ANSI constants should be avoided, for ease of debugging.)</p>	<pre>static final int MIN_WIDTH = 4; static final int MAX_WIDTH = 99; static final int GET_THE_CPU = 1;</pre>

7.2 – Criação de variáveis Local x Assign

Segunda regra: de preferência sempre a local nas macros e functions.

```
[#assign x = 2/]
[@macroxpto/]
${x} [@br/]
${y!}

[#macro macroxpto]
[#assign x = 3/]
[#assign y = 3/]
[/#macro]

Que produzirá o seguinte resultado:
```

3
3

```
[#assign x = 2/]
[@macroxpto/]
${x} [@br/]
${y!}

[#macro macroxpto]
[#local x = 3/]
[#local y = 3/]
[/#macro]
```

Que produzirá o seguinte resultado:

2

Obs: se o Y não estivesse com default, ou seja, Y!, uma exceção (variável não definida) seria gerada na aplicação.

Neste caso, a chamado da macro @macroxpto sobrescreveu a variável "x" e ainda deixou um legado, a variável "y".

Mas se o desejo do programador é realmente deixar uma legado da macro para a aplicação? O certo seria não usar local, e sim, assign, porém neste caso deveríamos respeitar a primeira regra visto anteriormente.

Supor que a aplicação se chame avaliação de desempenho, acrônimo: avd e a macro macroxpto, com acrônimo de xpt:

```
[#assign avd_x = 2/]
[@macroxpto/]
${avd_x} [@br/]
${xpt_y!}
[#macro macroxpto]
[#assign xpt_x = 3/]
[#assign xpt_y = 3/]
[/#macro]
```

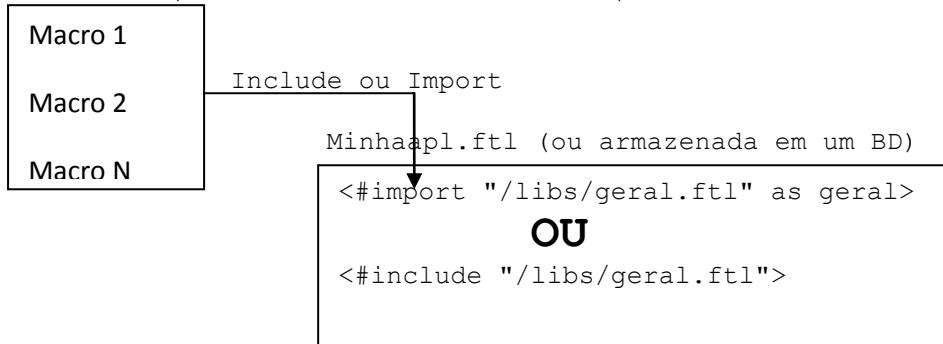
Que produzirá o seguinte resultado:

2
3

7.3 – IMPORT X INCLUDE

Como o arquivo de macros é incorporado a sua aplicação?

Geral.ftl (ou está armazenado em um BD)



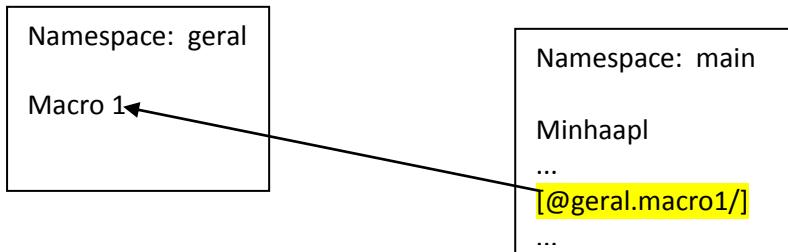
No Include, as macros são todas trazidas para a aplicação com os seguinte efeitos:

- Se uma macro possui erro de sintaxe, isto aparecerá na execução da sua aplicação;
- Os nomes das macros são trazidas para sua aplicação como variáveis;
- Se numa macro definimos uma variável como assign (ver item anterior), esta variável pode sobrepor a da sua aplicação.

No Import ocorre o seguinte efeito:

- Se observarmos o comando import, no final temos “**as geral**”, ou seja, o arquivo geral.ftl é importado “COMO GERAL”, onde “geral” define um namespace para rodar as macros. A aplicação roda no namespace padrão chamado “main”. Esta separação de namespaces elimina os problemas de nomeação de variáveis, pois tanto a aplicação quanto a macro rodarão em namespaces diferentes;

- As macros devem ser chamadas prefixando-as com o namespace, ou seja, geral.macrol, geral.macro2 e etc;
- Não pude testar o acesso a uma variável da macro. Se por exemplo a macrol possui um assign para a variável xpto, se devemos acessá-la via geral.xpto ou simplesmente xpto, porém tudo leva a crer que é via namespace.



No nosso ambiente as macros são incorporadas via include pelo programa Java que carrega o FM Engine. Tanto as macros quanto as aplicações estão em BD. Não podemos mudar a diretiva na aplicação.

7.4 – Documentação

Como dito na introdução do capítulo, não fazemos programas para nós, e sim, para os outros, no sentido que os outros possam realizar a manutenção com segurança. Devemos usar e abusar de comentários introdutórios (autor, data, versa, descrição, parâmetros ...) e de código.

Comentário introdutório: único, antes de escrevermos a aplicação.

Exemplo: Function formatarCPF

```

[#function formatarCPF fmtCPF_param]
[!-- Início do comentário

Aplicação: Função para formatar um CPF
Acrônimo: fmtCPF_
Autor: Ruben
Data: 13/03/2012
Descrição:
Esta função obtém uma string contendo o CPF a ser formatado e o devolve formatado
com a seguinte apresentação: 999.999.999-99

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

Parâmetros recebidos:
+-----+
| Parâmetro | Tipo/Tam | Obrigatório | Default | Descrição
+-----+
| fmtCPF_param | String | Sim | | O CPF a ser formatado
+-----+
Parâmetros devolvidos:
+-----+
| Parâmetro | Tipo/Tam | Obrigatório | 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]

Variáveis:
fmtCPF_param Variável local para o CPF recebido pela função
fmtCPF_novo Variável local para construir o novo CPF
fmtCPF_i Variável de looping usada como índice em ?substring
fmtCPF_j Variável de local usada como índice em ?substring
fmtCPF_dígito Variável local para cada caractere do CPF recebido
fmtCPF_retorno Variável local com o novo CPF formatado retornado pela função

Fim comentário --]

```

Comentários de código: para cada bloco de instruções em que a lógica é complexa.

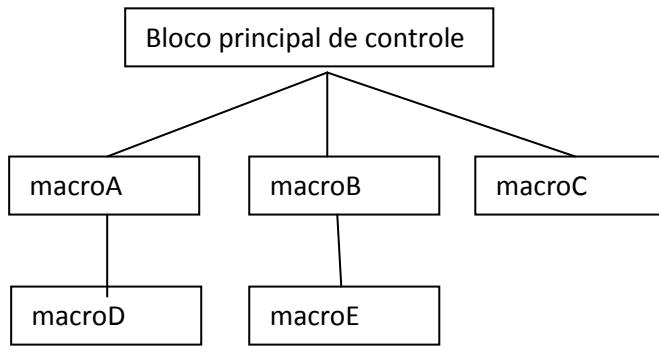
7.5 – Estrutura Modular

Escreva blocos de código que caibam no campo de visão da tela. Quem já teve que percorrer um If que começa numa tela e termina 5 ou 6 telas depois sabe o que estou dizendo. É um princípio antigo, muito antigo, da programação modular. Não é simplesmente cortar o código linear com a tesoura em tantas partes que você desejar. Cada módulo deve ter uma função bem definida e baixo acoplamento com os outros módulos.

Aplicação linear



Aplicação Modular



Como implementar no FM? Via execução de macro inline, ou seja, na própria aplicação da entrevista. **Não levar estas macros para o arquivo de macros!!!**
[@entrevista]

```
...
...
[@macroA/]
...
[/@entrevista]

[!-- Macros Inline --]
[#macro macroA]
...
[/#macro]
[#macro macroB]
...
[/#macro]
[#macro macroC]
...
[/#macro]
[#macro macroD]
...
[/#macro]
[#macro macroE]
...
[/#macro]
```

7.6 - Prefira Case a Ifs

Evitar o uso de Ifs, em detrimento ao uso de Case.

Estrutura

```
<#switch value>
  <#case refValue1>
    ...
    <#break>
  <#case refValue2>
    ...
    <#break>
  <#case refValueN>
    ...
    <#break>
  <#default>
    ...
</#switch>
```

Exemplo

```
<#switch being.size>
  <#case "small">
    This will be processed if it is small
    <#break>
  <#case "medium">
    This will be processed if it is medium
    <#break>
  <#case "large">
    This will be processed if it is large
    <#break>
  <#default>
    This will be processed if it is neither
</#switch>
```

Com certeza a aplicação fica muito mais clean e menos sujeita a bugs. Devemos sempre que possível fugir de ninhos de Ifs.

Combinar Case com Macros, o melhor dos dois mundos.

```
<@entrevista>
...
<#switch being.size>
  <#case "small">
    <@macrosmall/>
    <#break>
  <#case "medium">
    <@macromedium/>
    <#break>
  <#case "large">
    <@macrolarge/>
    <#break>
  <#default>
    <@macrodefault/>
</#switch>
...
</@entrevista>
```

```
<!-- Macros Inline -->
<#macro macrosmall>
  ...
</#macro>
<#macro macromedium>
  ...
</#macro>
<#macro macrolarge>
  ...
</#macro>
<#macro macrodefault>
  ...
</#macro>
```

Toda linguagem possui palavras reservadas. Este capítulo não trata das variáveis reservadas do FM ppd, e sim, das palavras (variáveis) que já são utilizadas nas macros FM do SIGA-DOC e campos da tela de entrevista , e portanto, não devem ser utilizadas nas aplicações, caso contrário, poderão provocar efeitos desastrosos como visto no capítulo 6.



8.1 – Palavras oriundas da biblioteca de macros do FM

Variáveis que devem ser evitadas nas aplicações FM.

Posição em 16/04/2012

Nome da variável	Onde foi definida
#assignenderecamentoPresidente	Assign dentro da macro ou da function
_presidente	Assign dentro da macro ou da function
_secretario_geral	Assign dentro da macro ou da function
_secretario_rh	Assign dentro da macro ou da function
aberturaBIE	Nome da macro
assentamento_funcional	Nome da macro
assinatura	Nome da macro
assinaturaBIE	Nome da macro
assinaturaCentro	Nome da macro
assinaturaMovCentro	Nome da macro
br	Nome da macro
cabecalho	Nome da macro
cabecalhoCentralizado	Nome da macro
cabecalhoCentralizadoPrimeiraPagina	Nome da macro
cabecalhoEsquerda	Nome da macro
cabecalhoEsquerdaPrimeiraPagina	Nome da macro
caixaSelecao	Nome da macro
checkbox	Nome da macro
corpoBIE	Nome da macro
dadosComplementares	Nome da macro
data	Nome da macro
div	Nome da macro
documento	Nome da macro
dump	Nome da macro
dump_item	Assign dentro da macro ou da function
dump_key	Assign dentro da macro ou da function
editor	Nome da macro
enderecamentoDiretorDeRH	Assign dentro da macro ou da function
enderecamentoDiretorGeral	Assign dentro da macro ou da function
entrevista	Nome da macro
estiloBrasaoAEsquerda	Nome da macro
estiloBrasaoCentralizado	Nome da macro
extensaoAssinador	Nome da macro
extensaoBuscaTextual	Nome da macro
extensaoEditor	Nome da macro
fechoBIE	Nome da macro
finalizacao	Nome da macro

fixcrlf	Nome da macro
fmtCPF_digito	Assign dentro da macro ou da function
fmtCPF_novo	Assign dentro da macro ou da function
fmtCPF_qtezeros	Assign dentro da macro ou da function
fmtCPF_retorno	Assign dentro da macro ou da function
fmtCPF_tam	Assign dentro da macro ou da function
formatarCPF	Function
formulario	Nome da macro
funcao	Nome da macro
genero	Assign dentro da macro ou da function
grupo	Nome da macro
grupoLarguraTotal	Assign dentro da macro ou da function
identificacao	Nome da macro
inTemplate	Assign dentro da macro ou da function
inTemplate	Assign dentro da macro ou da function
letra	Nome da macro
lotacao	Nome da macro
memo	Nome da macro
memorando	Nome da macro
mensagem	Nome da macro
mioloDJE	Nome da macro
numeroDJE	Nome da macro
oculto	Nome da macro
oficio	Nome da macro
orgaoAux	Assign dentro da macro ou da function
par	Function
pessoa	Nome da macro
portaria	Nome da macro
primeiroCabecalho	Nome da macro
primeiroRodape	Nome da macro
quebraPagina	Nome da macro
radio	Nome da macro
resumo	Nome da macro
rodape	Nome da macro
rodapeClassificacaoDocumental	Nome da macro
rodapeNumeracaoADireita	Nome da macro
rodapeNumeracaoCentralizada	Nome da macro
selecao	Nome da macro
selecionavel	Nome da macro
separador	Nome da macro
texto	Nome da macro
tipoAutoridade	Assign dentro da macro ou da function

tipoSel	Assign dentro da macro ou da function
tituloDJE	Nome da macro
tl	Assign dentro da macro ou da function
topico	Nome da macro
tratamento	Assign dentro da macro ou da function
tratamento	Assign dentro da macro ou da function
varName	Assign dentro da macro ou da function
vertipo	Nome da macro
virgula	Nome da macro
visivel	Assign dentro da macro ou da function
XStandard	Nome da macro

8.2 - Palavras oriundas dos nomes dos campos do formulário de entrevista do SIGA-DOC

Ver lista destes nomes na seção "[NOMES por ordem de aparição e elementos HTML](#)".

8.3 - IDs oriundos dos elementos HTML do formulário de entrevista do SIGA-DOC

Ver lista destes IDs na seção "[IDs por ordem de aparição e elementos HTML](#)"

Os efeitos do uso destes "Nomes" foram vistos em "[Erros mais comuns no FM envolvendo nomes de variáveis](#)"

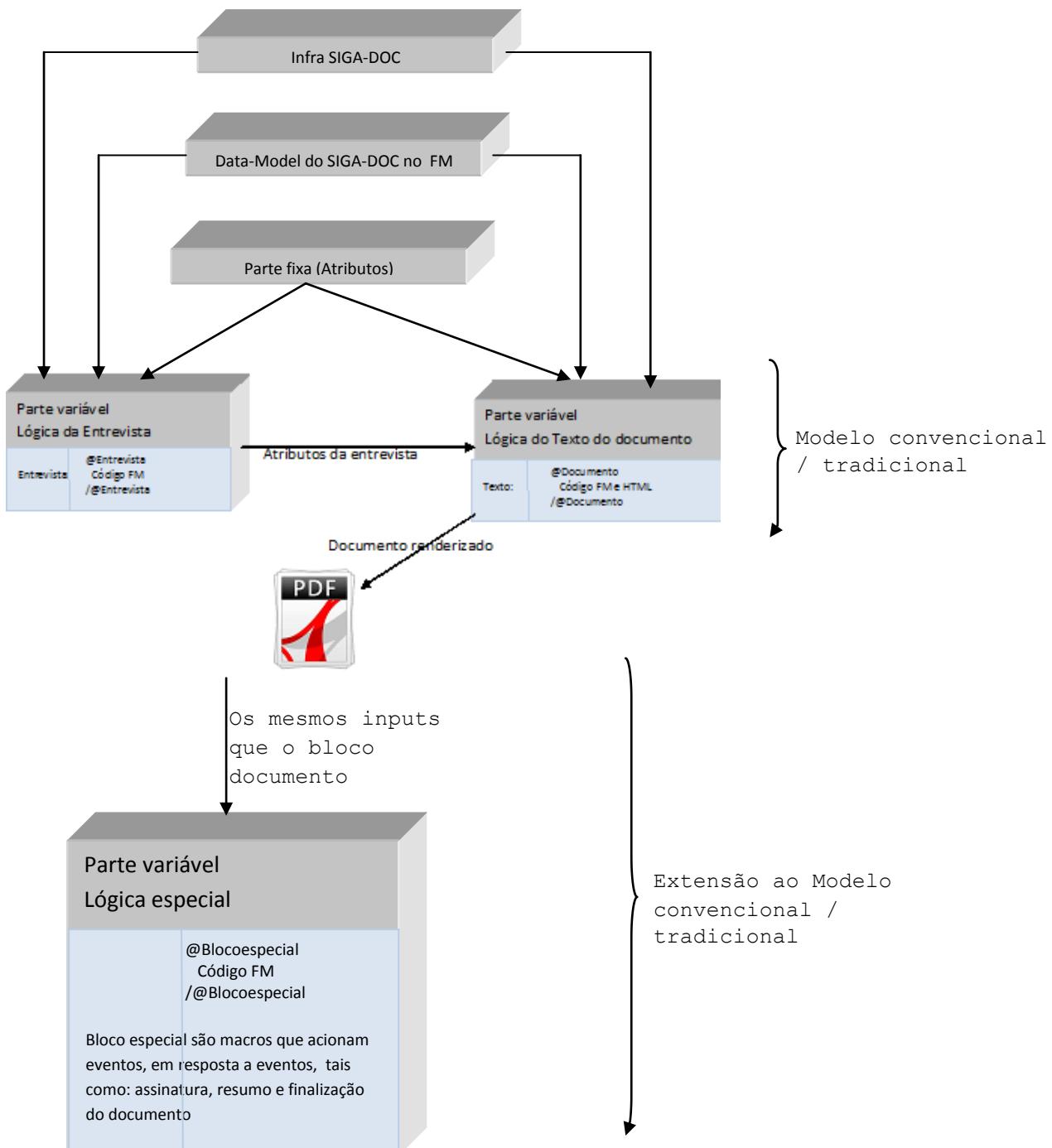
É hora de colocarmos a mão na massa. Seria muita pretensão de minha parte ensinar alguém a programar em um único capítulo, porém, finalizar o manual sem um exemplo, poderia deixar uma lacuna irrecuperável. Já vimos todas as macros e functions, o data-model, os hashes e seus métodos, lições aprendidas, boas práticas e palavras reservadas, e agora, como juntá-los numa aplicação real.



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 critica) 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. Dessa forma, teremos, se for necessário, os blocos:

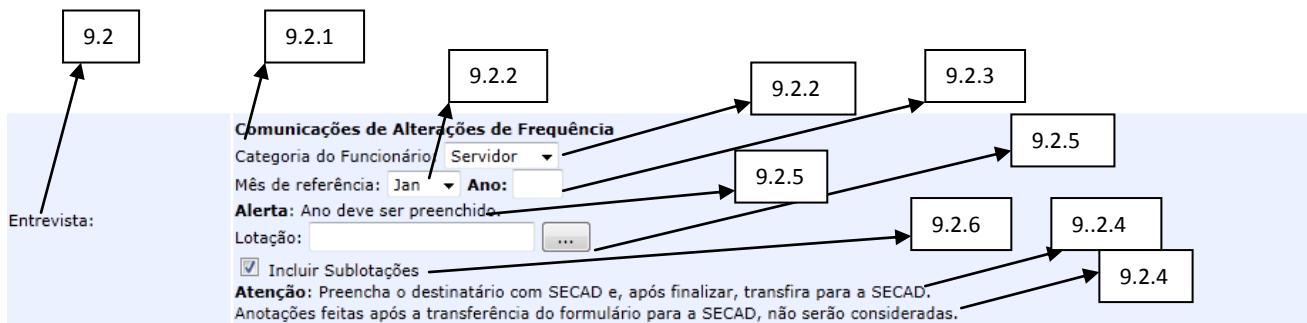
```
[@assinatura]
Código para quando o documento é assinado
[/@assinatura]

[@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 <code>[#if grupoLarguraTotal]=100></code> e alterá-la para <code>[#if (grupoLarguraTotal >= 100)]</code> Não entendi o efeito deste parâmetro.	largura=0
esconder	Se <code>esconder=true</code> , a macro simplesmente desconsidera tudo (titulo, largura, depende ...) e trata o elemento que for definido dentro da macro como elemento inline <code></code> Se <code>esconder=false</code> , a macro simplesmente considera tudo e trata o elemento que for definido dentro da macro como elemento block <code><div></code>	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.



Relet: 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 sa 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 informações sobre AJAX](#).

Opções do Relet

relet =	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, incialize 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 levar o ponteiro do mouse para um campo e 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? Embora 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 em [**\(6.10 – Macros FM e a opção reler, relertab e obrigatorio\)**](#).

9.2.2 - @Selecao

É um tipo de entrevista/pergunta que se exibe 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>	reler=true reler=false 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"]
    [@comlancamentos/]
  [/#if]
[/@grupo]
```

9.2.3 - @Texto

É um tipo de entrevista/pergunta que se exibe 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 " nao " 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

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:

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

Dados da Lotação		
Nome ou Sigla:		Pesquisar
Órgão:	Seção Judiciária do Rio de Janeiro	
Total: 413 Itens		
Sigla	Nome	Fim de Vigência
10VF	10ª Vara Federal	
10VFCF	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.
onclique	Aplicação JS que pode ser passada	onclique=""

	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'; Onde nomevar é o nome da variável passada a macro @checkbox</pre>	onclick	Se o Box está ticado Então: a variável recebe o valor "sim" Senão: a variável recebe o valor "não"

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]
[/@grupo]
```

O que podemos passar para o onclique?

A - Emitir um alerta do JS com alguma explicação adicional

```
[@checkbox titulo="teste" var="varcheck" default="Nao" onclique="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" onclique="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	obrigatorio=false

	permitirá que o mesmo fique em 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
4751084679

String considerada
47510846749

String retornada
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 "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 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.ooo-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:

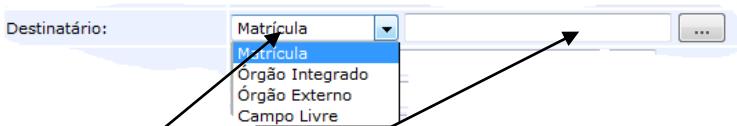
```
[#assign antigoCPF = "47510846749" /]
[#assign novoCPF = fmtvldCPF(antigoCPF) /]
[#if novoCPF == "E1"]
    Tratar o erro, pois o CPF fornecido está inconsistente e não pode ser formatado
[#elseif novoCPF == "E2"]
    Tratar o erro, pois o CPF fornecido é inválido, ou seja, possui dígitos verificadores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

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

9.2.12 – Obtendo dados do servidor/funcionário

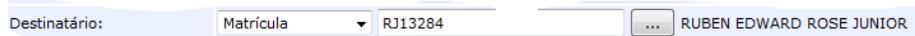
Para tanto, o servidor deve estar instanciado, caso contrário não teremos acesso e poderemos receber uma exception do tipo undefined (ver capítulo "Lições Aprendidas").

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

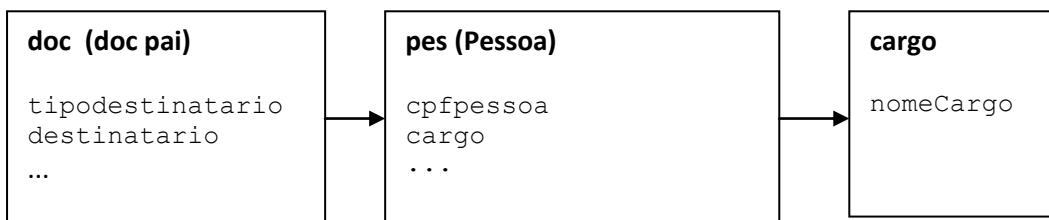
Como podemos saber via código se o servidor está instanciado?

```
[#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!/]  
[#assign codigo_Cargo = doc.destinatario.cargo!/]  
[#assign descricao_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 da classe pessoa 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	DATAINI_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 @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.
onclique	Aplicação JS que pode ser passada quando o botão radio é clicado. O default é "".	onclique=""

JavaScripts:

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

Ok **Visualizar o modelo preenchido**

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 onclique?

A - Emitir um alerta do JS com alguma explicação adicional

```
[@radio titulo="teste" var="varradio" default="Não" valor="sim" onclique="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" onclique="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]

9.2.14 - Aplicação utilizando todas as macros da Entrevista

9.2.14.1 - Aplicação FM

```
[@grupo titulo="Aplicação Teste para Checar o HTML Gerado"]
[@selecao var="catFuncionario" titulo="Categoria do Funcionário"
    reler=true opcoes="Servidor; Terceiro"]
[/@grupo]

[@grupo]
[@texto titulo="Ano Posse/Contratação" var="ano" largura="4" maxcaracteres="4"
obrigatorio="Sim" reler="ajax" idAjax="anoAjax"/>
[@grupo depende="anoAjax"]
[#if (ano!="") == ""]
    [@mensagem titulo="Alerta" texto="Ano deve ser preenchido." vermelho=true]
[/#if]
[/@grupo]
[/@grupo]

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

[@grupo]
[@data var="dataformat" titulo="Data de Formatura:"/>
[/@grupo]

[@grupo]
[@memo var="informacoes" titulo="Dados sobre a Formação" colunas="63" linhas="3"
/>
[/@grupo]

[@grupo]
[@checkbox var="fazertrein" titulo="Deseja fazer treinamento?" default="Sim"]
[/@grupo]

[@grupo titulo="Turno de Preferência"]
[@radio titulo="Manhã" var="radio_resp" valor="1" default="Manhã" /]
[@radio titulo="Tarde" var="radio_resp" valor="0" /]
[/@grupo]
```

9.2.14.2 - Formulário

The screenshot shows a web-based survey application. At the top, it displays the title "Aplicação Teste para Checar o HTML Gerado". Below this, there is a dropdown menu labeled "Categoria do Funcionário" with "Servidor" selected. There are two text input fields: one for "Ano Posse/Contratação" and another for "Lotação", which contains the placeholder "Alerta: Ano deve ser preenchido.". A "Data de Formatura" field is also present. A large "Dados sobre a Formação" memo area is available for additional information. A checkbox labeled "Deseja fazer treinamento?" is checked. At the bottom, there is a section titled "Turno de Preferência" with two radio buttons: "Manhã" (selected) and "Tarde".

9.2.14.3 - Estrutura HTML da Aplicação

Entrevista: SPAN

DIV TABLE TD Aplicação Teste para Checar o HTML Gerado

DIV TABLE TD Categoria do Funcionário: SELECT Servidor ▾

DIV TABLE TD SPAN Ano Posse/Contratação: INPUT

DIV TABLE TD SPAN B Alerta: Ano deve ser preenchido.

DIV TABLE TD SPAN Lotação: INPUT INPUT ... SPAN

DIV TABLE TD SPAN Data de Formatura: INPUT

DIV TABLE TD SPAN Dados sobre a Formação:

TEXTAREA DIV

DIV TABLE TD SPAN ?

DIV TABLE TD SPAN Deseja fazer treinamento?

DIV TABLE TD Turno de Preferência

DIV TABLE TD TABLE TD Manhã

DIV TABLE TD TABLE TD Tarde

```
<div><table><tr><td>Aplicação Teste para Checar o HTML Gerado</td></tr></table></div>
<div><table><tr><td>Categoria do Funcionário:</td><td>SELECT Servidor</td><td>▼</td></tr></table></div>
<div><table><tr><td>Ano Posse/Contratação:</td><td>INPUT</td></tr></table></div>
<div><table><tr><td>B Alerta: Ano deve ser preenchido.</td></tr></table></div>
<div><table><tr><td>Lotação:</td><td>INPUT</td><td>INPUT</td><td>...</td><td>SPAN</td></tr></table></div>
<div><table><tr><td>Data de Formatura:</td><td>INPUT</td></tr></table></div>
<div><table><tr><td>Dados sobre a Formação:</td><td>TEXTAREA</td><td>DIV</td></tr></table></div>
<div><table><tr><td>?</td></tr></table></div>
<div><table><tr><td>Deseja fazer treinamento? <input checked="" type="checkbox"></td></tr></table></div>
<div><table><tr><td>Turno de Preferência</td></tr></table></div>
<div><table><tr><td>Manhã</td></tr></table></div>
<div><table><tr><td>Tarde</td></tr></table></div>
```

Observando a estrutura e o HTML abaixo:

- Para cada macro {@grupo} é gerado uma <div> com uma <table>, incluindo o bloco {@grupo depende};
- As tabelas <table> estão associadas a classe CSS "**entrevista**"
- O título passado a macro {@grupo} fica em uma linha <tr> com a classe CSS "**header**";
- Geralmente os textos literais estão no bloco

9.2.14.4 - HTML da Aplicação

```
<td>Entrevista:</td>
<td colspan="3">
<span id="spanEntrevista" depende=";tipoDestinatario;destinatario;forma;modelo;">
<!-- CATEGORIA DO FUNCIONÁRIO -->
<!-- ANO POSSE / CONTRATAÇÃO -->
<div>
<table class="entrevista" width="100%">
<tbody>
<tr class="header">
<td>Aplicação Teste para Checar o HTML Gerado</td>
</tr>
<tr>
<td>
Categoria do Funcionário:
<input type="hidden" value="catFuncionario" name="vars">
<select onclick="" onchange="javascript: sbmt();" name="catFuncionario">
<option value="Servidor" selected="">Servidor</option>
<option value=" Terceiro"> Terceiro</option>
</select>
</td>
</tr>
</tbody>
</table>
</div>

<!-- ANO POSSE / CONTRATAÇÃO -->
<!-- ANO POSSE / CONTRATAÇÃO -->
<div>
<table class="entrevista" width="100%">
<tbody>
<tr>
<td>
<input type="hidden" value="ano" name="vars">
<input type="hidden" value="ano" name="obrigatorios">
<span style="font-weight:bold;">Ano Posse/Contratação:</span>
<input type="text" maxlength="4" size="4" onchange="javascript:
sbmt('anoAjax');" value="" name="ano">
<div id="divanoAjax" depende=";anoAjax;">
<table class="entrevista" width="100%">
<tbody>
<tr>
<td>
<span style="color:#ff0000">
<b>Alerta</b>
: Ano deve ser preenchido.
</span>
</td>
</tr>
</tbody>
</table>
</div>
</td>
</tr>
</tbody>
</table>
</div>
```

Quando definimos o `idAjax` é executado um JS `sbmt()` passando o `idajax`

Este é o bloco que corresponde ao `idajax`. Observar que o `id` passado é concatenado com `div` para criar o `id` do bloco. Quando o `ano` está vazio este bloco é gerado com a mensagem. Porém, se colocarmos o `ano`, o seguinte bloco será gerado:

```
<div depende=";anoAjax;">
<div id="divanoAjax"><!--ajax:divanoAjax-->
<table width="100%" class="entrevista">
<tbody>
<tr>
<td>
</td>
</tr>
</tbody>
</table>
</div>
```

Ou seja, vazio e nada será apresentado na tela.

```

<!-- LOTAÇÃO -->
<div>
<table class="entrevista" width="100%">
<tbody>
<tr>
<td>
<input type="hidden" value="lotacao_lotacaoSel.id" name="vars">
<input type="hidden" value="lotacao_lotacaoSel.sigla" name="vars">
<input type="hidden" value="lotacao_lotacaoSel.descricao" name="vars">
<span style="; ">Lotação:</span>
<input type="hidden" name="lotacao_lotacaoSel.id" value="">
<input type="hidden" name="lotacao_lotacaoSel.descricao" value="">
<input type="hidden" name="lotacao_lotacaoSel.buscar">
<input type="hidden" name="reqlotacao_lotacaoSel">
<input id="alterouSel" type="hidden" value="" name="alterouSel">
<input type="text" size="25" onblur="javascript: ajax_lotacao_lotacao();"
onkeypress="return handleEnter(this, event)" name="lotacao_lotacaoSel.sigla">
<input id="lotacao_lotacaoSelButton" type="button" theme="simple"
onclick="javascript: popup_lotacao_lotacao(''); " value="...">
<span id="lotacao_lotacaoSelSpan"></span>
</td>
</tr>
</tbody>
</table>
</div>
<!-- DATA FORMATURA -->
<div>
<table width="100%" class="entrevista">
<tbody><tr>
<td><span style="; ">Data de Formatura:</span>
<input type="hidden" value="dataformat" name="vars">
<input type="text" onblur="javascript:verifica_data(this); " maxlength="10"
size="10" value="" name="dataformat">
</td>
</tr>
</tbody>
</table>
</div>
<!-- DADOS S/ FORMAÇÃO-->
<div>
<table class="entrevista" width="100%">
<tbody>
<tr>
<td>
<input type="hidden" value="informacoes" name="vars">
<div style="padding-top:5px; ">
<span style="; ">
Dados sobre a Formação:
<br>
</span>
<textarea name="informacoes" rows="3" cols="63"></textarea>
</div>
</td>
</tr>

```

A macro @data cria um HTML "text", porém é executado um JS para verificar se o conteúdo é data ou não.

```

</tbody>
</table>
</div>

<!-- ----->
<!-- TREINAMENTO ----->
<!-- ----->
<div>
<table class="entrevista" width="100%">
<tbody>
<tr>
<td>
<input type="hidden" value="fazertrein" name="vars">
<input id="fazertrein" type="hidden" value="Sim" name="fazertrein">
<input type="checkbox" onclick="javascript: if (this.checked) document.getElementById('fazertrein').value = 'Sim'; else document.getElementById('fazertrein').value = 'Nao'; ;" checked="" value="Sim" name="fazertrein_chk">
<span style=";">Deseja fazer treinamento?</span>
</td>
</tr>
</tbody>
</table>
</div>

```

A macro
[@checkbox var="fazertrein"
titulo="Deseja fazer treinamento?"
default="Sim"/>]

JS é executado para colocar o valor "Sim" ou "Não"

```

<!-- ----->
<!-- TURNO ----->
<!-- ----->
<div>
<table class="entrevista" width="100%">
<tbody>
<tr class="header">
<td>Turno de Preferência</td>
</tr>
<tr>
<td>
<input type="hidden" value="radio_resp" name="vars">
<input id="radio_resp" type="hidden" value="1" name="radio_resp">
<table>
<tbody>
<tr>
<td>
<input type="radio" onclick="javascript: if (this.checked) document.getElementById('radio_resp').value = '1'; ;" value="1" name="radio_resp_chk">
</td>
<td>Manhã</td>
</tr>
</tbody>
</table>
<table>
<tbody>
<tr>
<td>
<input type="radio" onclick="javascript: if (this.checked) document.getElementById('radio_resp').value = '0'; ;" value="0" name="radio_resp_chk">
</td>

```

[@radio titulo="Manhã"
var="radio_resp" valor="1"
default="Manhã" /]
[@radio titulo="Tarde"
var="radio_resp" valor="0" /]

JS é executado para colocar o valor definido na macro @radio

```

<td>Tarde</td>
</tr>
</tbody>
</table>
</td>
</tr>
</tbody>
</table>
</div>
<!-- FIM -->

```

9.2.14.5 – Como ficaria aplicação sem o [#grupo]

A mesma aplicação sem o bloco [#grupo], visto que ele é opcional. Ele só não é opcional no [#grupo depende].

É importante lembrar que neste caso temos que nos preocupar com as quebras de linha.

Aplicação FM

```

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

[@br/]

[@texto titulo="Ano Posse/Contratação" var="ano" largura="4" maxcaracteres="4"
obrigatorio="Sim" reler="ajax" idAjax="anoAjax/"]
[@grupo depende="anoAjax"]
  [#if (ano!="") == ""]
    [@mensagem titulo="Alerta" texto="Ano deve ser preenchido."
      vermelho=true/]
  [/#if]
[/@grupo]

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

[@br/]

[@data var="dataformat" titulo="Data de Formatura:/"]

[@br/]

[@memo var="informacoes" titulo="Dados sobre a Formação" colunas="63" linhas="3"/>
]

[@checkbox var="fazertrein" titulo="Deseja fazer treinamento?" default="Sim/"]

[@br/]

[@grupo titulo="Turno de Preferência"]
  [@radio titulo="Manhã" var="radio_resp" valor="1" default="Manhã" /]
  [@radio titulo="Tarde" var="radio_resp" valor="0" /]
[/@grupo]

```

OBSERVAÇÃO: Se não utilizarmos o bloco [#grupo] o código HTML fica mais limpo como podemos observar na estrutura abaixo. O excesso de <div> e <table> (incluindo <tr>, <td> ...)

Estrutura HTML da Aplicação

SPAN Categoria do Funcionário: SELECT Servidor ▾

SPAN Ano Posse/Contratação: INPUT

DIV TABLE TD SPAN B Alerta: Ano deve ser preenchido.

SPAN Lotação: INPUT INPUT ... SPAN

SPAN Data de Formatura: INPUT

DIV SPAN Dados sobre a Formação:
TEXTAREA DIV ?

SPAN Deseja fazer treinamento?

DIV TABLE TD Turno de Preferência
TABLE TD TD TD Manhã
TABLE TD TD TD Tarde

9.3 - COMENTÁRIOS SOBRE A MACRO @ENTREVISTA - Responsável pela entrevista

9.3.1 - Tabelas não possuem macros, como tratá-las?

A tabela deve ser definida com código HTML.

No exemplo abaixo, continuação da tela que serviu de base para explicação das macros anteriores e cujo código será listado no anexo, existe uma tabela na entrevista, onde a primeira coluna é o nome do funcionário e a segunda uma macro @selecao com os valores "sem lançamento" e "com lançamento".

Comunicações de Alterações de Frequência

Mês de referência: Jan Ano: 2012

Lotação: STI Subsecretaria de Tecnologia da Informação e de Comunicações

Incluir Sublotações

Atenção: Preencha o destinatário com SECAD e, após finalizar, transfira para a SECAD. Anotações feitas após a transferência do formulário para a SECAD, não serão consideradas.

GUSTAVO MONTEIRO DE BARROS BARRETO	Frequência: Sem lançamentos
ILARA FUMO MARIANO DA SILVA	Frequência: Sem lançamentos
CARLOS FERNANDO DO NASCIMENTO	Frequência: Sem lançamentos
JULIO FERREIRA DOS SANTOS NETO	Frequência: Sem lançamentos
MARCELO RODRIGUES BASTOS	Frequência: Sem lançamentos
MARIA LUCIA GONCALVES COELHO CARNAVAL	Frequência: Sem lançamentos
MARISTELA DE SOUZA VICENTE	Frequência: Sem lançamentos
TEREZINHA REGINA FRYDMAN	Frequência: Sem lançamentos
ZELIA MARIA VASCONCELOS DE OLIVEIRA	Frequência: Sem lançamentos
ANDREIA MIRANDA DE LUNA	Frequência: Sem lançamentos
JOSE BERNARDO DE FIGUEIREDO CIRIACO	Frequência: Sem lançamentos
JULIO CESAR CHICRE DE OLIVEIRA	Frequência: Sem lançamentos
PAULO MARCOS MAGALHAES LIMA	Frequência: Sem lançamentos

No código abaixo, a cor azul é do FM e o vermelho é HTML e preto é comentário.

```
[#assign i = 0 /]
Inicializa a lista de servidores
[#assign pessoas = func.pessoasPorLotacao(.vars['lotacao_lotacaoSel.id'])?number,
    buscarSublotacoes)/]
Obtem uma lista (array) com todos os servidores de uma determinada lotação. Ver
descrição da função em 5.1
<table width="100%" border="0" cellspacing="0">
Inicia a definição da tabela
[#list pessoas as pes]
Inicia o loop do tipo for
[#assign i = i + 1 /]
Incrementa o contador de servidores
[@oculto var="pes_descricao"+i valor="${pes.descricao}"/]
[@oculto var="pes_sigla"+i valor="${pes.sigla}"/]
Guarda a sigla e a descrição (nome) do servidor
<tr>
Inicia a definição da linha da tabela
<td width="30%" align="left" valign="top">${pes.descricao}</td>
Define a primeira coluna com o nome do servidor
<td width="70%" align="left">
Inicia a definição da segunda coluna que conterá o @selecao
[@grupo]
[@selecao var="freq"+i titulo="Frequência" reler=true idAjax="ajax"+i
    opcoes="Sem lançamentos;Com lançamentos"/]
```

```

[@grupo]
Define a segunda coluna com o campo seleção e idAjax
[@grupo depende="ajax"+i]
Estabelece um depende AJAX para o campo seleção
[#if .vars['freq'+i]?? && .vars['freq'+i] == "Com lançamentos"]
Verifica se foi selecionado a opção "com lançamentos"
[@comlancamentos/]
Executa uma macro inline para tratar o lançamento
[/#if]
Fecha o if
[@grupo]
Fecha o grupo depende
</td>
Fecha a definição da segunda coluna da tabela
</tr>
Fecha a definição da linha da tabela
[/#list]
Fecha o loop for
</table>
Fecha a definição da tabela

```

Observação: não tem jeito, a tabela deve ser criada na mão com HTML.

9.3.2 – Entrevista aninhada e variáveis criadas dinamicamente

Continuando com exemplo anterior, que está norteando todo este capítulo, temos as seguintes condições:

- Podemos ter até N servidores por lotação, ou seja, não existe um número fixo;
- Cada servidor pode ter até 5 motivos de ausência;
- Se a ausência for por motivo de compensação de hora-extra, podemos ter até 20 lançamentos.

Comunicações de Alterações de Frequência

Mês de referência: Jan Ano: 2012

Lotação: STI Subsecretaria de Tecnologia da Informação e de Comunicações

Incluir Sublotações

Atenção: Preencha o destinatário com SECAD e, após finalizar, transfira para a SECAD. Anotações feitas após a transferência do formulário para a SECAD, não serão consideradas.

GUSTAVO MONTEIRO DE BARROS BARRETO	Frequência: Sem lançamentos	
ILARA FUMO MARIANO DA SILVA	Frequência: Com lançamentos	
	Ausência(s): 2	
Motivo (As tabelas da Base Legal encontram-se na página da SECAD na Intranet):		
Atraso/Saída Antecipada		
Data:	Tempo de Atraso/Saída Antecipada: hs	
Motivo (As tabelas da Base Legal encontram-se na página da SECAD na Intranet):		
Horas extras (trabalhadas sem remuneração)		
Nº de dias com horas extras: 4		
Data:	Quantidade de Horas:	Feriado ou Domingo ?: Sim
Data:	Quantidade de Horas:	Feriado ou Domingo ?: Sim
Data:	Quantidade de Horas:	Feriado ou Domingo ?: Sim
Data:	Quantidade de Horas:	Feriado ou Domingo ?: Sim
Obs: Máximo de duas horas em dias úteis e de dez em sábados, domingos e feriados.		

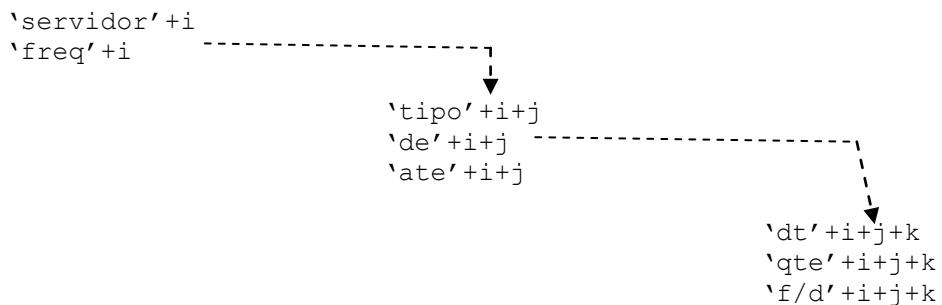
Isto se chama aninhamento de entrevista, neste caso em três níveis.

Problemas a vista: Supor que a maior diretoria tenha 70 servidores, neste caso teríamos que ter: $70 \times 75 = 5.250$ variáveis para guardar conforme a tabela abaixo:

Servidor	Aus. 1	...	Aus. 5	H.Extra1	...	H.Extra20
	Tipo De Até		Tipo De Até	Dt Qte F/D		Dt Qte F/D
Servidor1	Xx ddd aaa		Xy dd2 aa2	D1 3 f		D2 5 D
...
Servidor70	Xx ddd aaa		Xy dd2 aa2	D1 3 f		D2 5 D

Cada servidor pode ter até 75 variáveis. OK, vamos guardar tudo em um array no formato acima, correto? **Errado, pois o FM não trabalha com arrays.**

Então teremos que criar estas 5.250 variáveis na aplicação? Sim e Não. Sim, porque necessitam ser criadas, porém dinamicamente. Como? Simulando um array com ajuda de índices tipo: i, j, k. O + no FM indica concatenação de variáveis e/ou strings. Neste caso, 'freq'+i, indica a concatenação da string freq com a variável i.



Onde:

i (índice para manipular os servidores - até N)
j (índice para manipular os motivos de ausência dos servidores - até 5)
k (índice para manipular o motivo hora extra - até 20)

Desta forma, dt1043, seria a terceira data da H.extra, da quarta ocorrência da ausência, do décimo servidor.

E como criamos estas variáveis? Toda macro possui o var e neste caso utilizamos o seguinte artifício

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

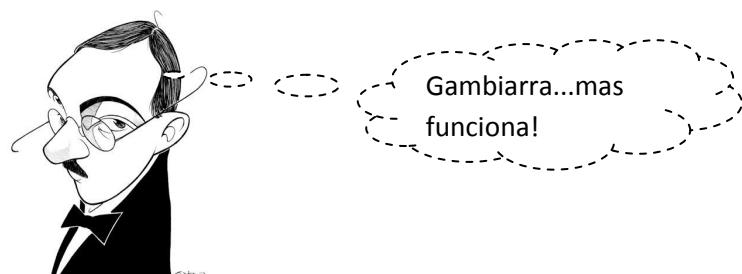
Sendo que o i, obviamente, já foi previamente carregado com valor desejado.

Como manipular o i, j e k? Com a diretiva list do FM.

Exemplo:

```
[#list 1..30 as i]
  [@selecao var="freq"+i titulo="Frequência" reler=true idAjax="ajax"+i
  opcoes="Sem lançamentos;Com lançamentos"]
[/#list]
```

Teremos 30 campos de seleção, com variáveis freq1, freq2...freq30.

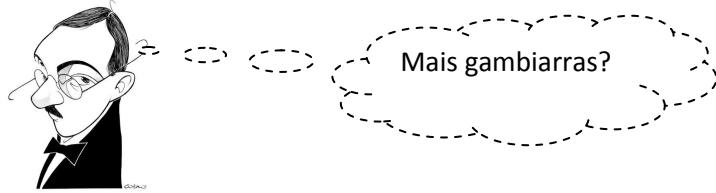


9.4 - TRABALHANDO NO BLOCO @documento

9.4.1 - Textos e tabelas

O bloco @documento é marcado por HTML e intrepolação de variáveis, seja em texto corrido, tabela de tamanho fixo ou tabela de tamanho variável (como no exemplo anterior).

Para quem não é versado em, e/ou não gosta de, HTML nós podemos utilizar o seguinte macete:



Desenhar o documento, texto e/ou tabela fixa, em um composer de HTML e colar no bloco @documento. Existe um composer free muito bom, o SeaMonkey (<http://www.seamonkey-project.org/>) .

Abaixo, a tela do composer do Seamonkey onde o documento (que os terceirizados devem assinar) foi produzido.

The screenshot shows the SeaMonkey Composer interface with the following content:

Processo: RJ-EOF-\${processo!upper_case}
Objeto: \${objeto!upper_case}
Nome da Empresa: \${nomeEmpresa!upper_case}
Nome do funcionário colaborador
\${nomeFuncColaborador!upper_case}

CONTRATO nº \${contrato!}
CNPJ: \${cnpj!}
Identidade
\${identidade!} CPF
\${cpf!}

Sistema/Serviço	S/N	OBS/JUSTIFICATIVAS
Acesso à rede da SJRJ (estação de trabalho: computador + impressora)	[#if (acessoRede!) == "Sim"] S [#else] N [/#if]	\${obsAcessoRede!}
Acesso à Intranet SJRJ	[#if (acessoIntranet!) == "Sim"] S [#else] N [/#if]	\${obsAcessoIntranet!}
Acesso à Internet	[#if (acessoInternet!) == "Sim"] S [#else] N [/#if]	\${obsAcessoInternet!}
Conta de E-mail corporativo	[#if (contaEmail!) == "Sim"] S [#else] N [/#if]	\${obsContaEmail!}
Acesso a Sistema Processual Apolo	[#if (acessoApolo!) == "Sim"] S [#else] N [/#if]	\${obsAcessoApolo!}
Acesso ao Sistema de Controle de Chamados: descrever em OBS	[#if (acessoCtrlChamados!) == "Sim"] S [#else]	\${obsAcessoCtrlCha!}

At the bottom, there are tabs for Normal, HTML Tags, HTML Source, Preview, and icons for mail, file, and search.

É importante observar que já inserimos no desenho as variáveis da aplicação do FM que serão interpoladas, e também pequenas regras do tipo `[#if (contaEmail!) == "Sim"] S [#else] N [/#if]`. Após o desenho, pedimos ao SeaMonkey para gerar o HTML e o introduzimos no bloco `@documento`. Obviamente o SeaMonkey gera o HTML completo (Head e Body), desta forma devemos selecionar só o que nos interessa, pois o SIGA-DOC já gera o HTML completo.

Trecho do HTML gerado:

```
<td style="vertical-align: top;"><font size="-1">Conta de E-mail corporativo<br>
</font> </td>
<td style="text-align: center; vertical-align: top;"><font
size="-1">[#if (contaEmail!) == "Sim"] S [#else] N [/#if]
</font></td>
```

Aqui, o documento completo:

Processo: RJ-EOF-\${processo!?"upper_case}	CONTRATO nº \${contrato!}	
Objeto: \${objeto!?"upper_case}		
Nome da Empresa: \${nomeEmpresa!?"upper_case}		CNPJ: \${cnpj!}
Nome do funcionário colaborador	Identidade	CPF
\${nomeFuncColaborador!?"upper_case}	\${identidade!}	\${cpf!}

Sistema/Serviço	S/N	OBS/JUSTIFICATIVAS
Acesso à rede da SJRJ (estação de trabalho: computador + impressora)	<code>[#if (acessoRede!) == "Sim"] S [#else] N [/#if]</code>	<code> \${obsAcessoRede!}</code>
Acesso à Intranet SJRJ	<code>[#if (acessoIntranet!) == "Sim"] S [#else] N [/#if]</code>	<code> \${obsAcessoIntranet!}</code>
Acesso à Internet	<code>[#if (acessoInternet!) == "Sim"] S [#else] N [/#if]</code>	<code> \${obsAcessoInternet!}</code>
Conta de E-mail corporativo	<code>[#if (contaEmail!) == "Sim"] S [#else] N [/#if]</code>	<code> \${obsContaEmail!}</code>
Acesso a Sistema Processual Apolo	<code>[#if (acessoApolo!) == "Sim"] S [#else] N [/#if]</code>	<code> \${obsAcessoApolo!}</code>

Acesso ao Sistema de Controle de Chamados: desecrever em OBS	[#if (acessoCtrlChamados!) == "Sim"] S [#else] N [/#if]	\${obsAcessoCtrlChal!}
Acesso a Sistemas SJRJ específicos: descrever em OBS	[#if (acessoSisSJRJ!) == "Sim"] S [#else] N [/#if]	\${obsAcessoSisSJRJ!}
Acesso a Softwares específicos: descrever em OBS	[#if (acessoSoftEspec!) == "Sim"] S [#else] N [/#if]	\${obsAcessoSoftEspec!}
Acesso a diretórios de rede específicos: descrever em OBS	[#if (acessoDirRedeEspec!) == "Sim"] S [#else] N [/#if]	\${obsAcessoDirRedeEspec!}
Outros: descrever em OBS	[#if (outros!) == "Sim"] S [#else] N [/#if]	\${obsOutros!}
Outros: descrever em OBS	[#if (outros1!) == "Sim"] S [#else] N [/#if]	\${obsOutros1!}
Outros: descrever em OBS	[#if (outros2!) == "Sim"] S [#else] N [/#if]	\${obsOutros2!}

Gestor do Contrato: \${gestorContrato!?upper_case}	Matrícula: \${matricula!}
Lotação \${lota!?upper_case}	Ramal/Email \${ramalEmail!?upper_case}

Como gestor do contrato, responsabilizo-me em manter atualizadas as informações acima prestadas, solicitando descredenciamento imediatamente assim que o funcionário/colaborador

acima descrito for desvinculado do serviço ou da empresa e/ou quando for extinta a relação contratual entre EMPRESA e a SJRJ e/ou sempre que o serviço prestado pelo funcionário/colaborador precise ser descontinuado por qualquer motivo. Comprometo-me também a manter controle atualizado das solicitações e atualizações efetuadas, instruindo devidamente o processo administrativo respectivo.

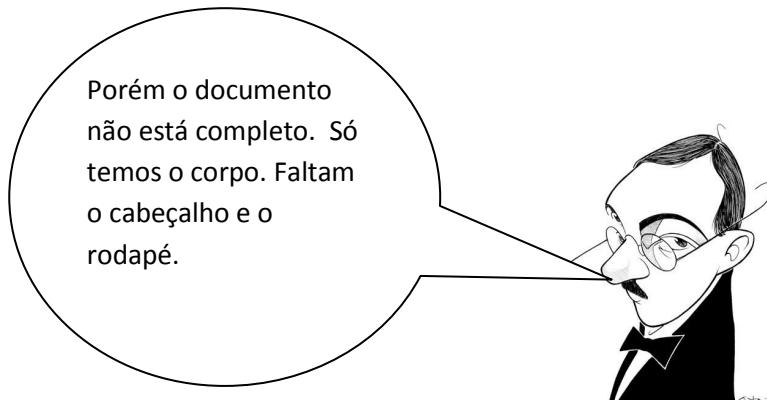
Assinatura do Gestor do Contrato

Assinatura da Direção da Subsecretaria

Observação: Se a tabela for variável, como no exemplo anterior, da entrevista, pode-se criar a tabela com a primeira linha no SeaMonkey e inserir o [#list ...] [/#list] para criar a estrutura de loop / for para popular a tabela.

Exemplo: O que está em vermelho foi criado pelo SeaMonkey e em azul é o código FM.

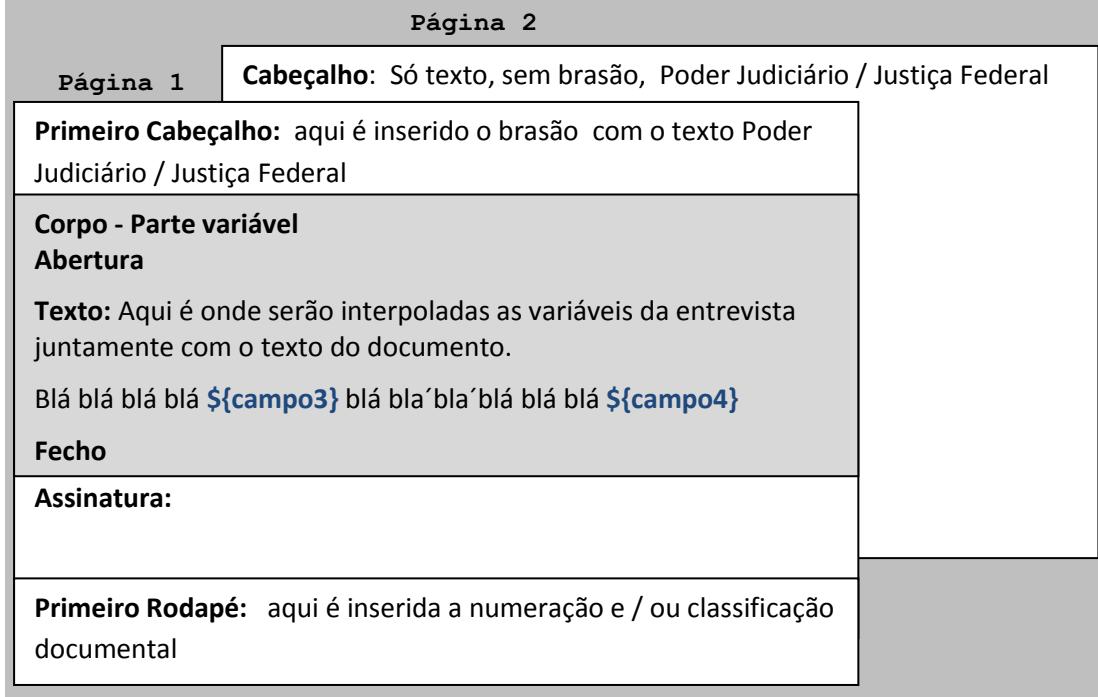
```
<table width="100%" border="0" cellspacing="0">
[#list pessoas as pes]
    Inicia o loop do tipo for
        <tr>
            Inicia a definição da linha da tabela
                <td width="30%" align="left" valign="top">${pes.descricao}</td>
                    Define a primeira coluna com o nome do servidor
            </tr>
            Fecha a definição da linha da tabela
        [/#list]
    Fecha o loop for
</table>
```



9.4.2 – Formatação: Cabeçalhos e Rodapés e outros

Vamos voltar a [seção 2.3](#) deste manual, cujo trecho principal está descrito abaixo por questões de comodidade.

Já o **documento (bloco documento)** possui a seguinte estrutura:



O documento pode possuir várias páginas, e por isso, possui cabeçalhos e rodapés diferenciados para a primeira página e páginas subsequentes.

Como mencionado na [seção 2.5](#), temos várias macros associadas a geração do documento, sejam elas de nível 1, 2, 3, 4 e 5.

Vamos tentar explicar as macros de geração de documento com um exemplo bem simples, uma entrevista com um só campo, gerando um documento com texto corrido, mais de 1 página, e uma interpolação de variáveis. O objetivo é de exibir os diversos lay-outs.

A entrevista coletará o nome de uma pessoa que será interpolado no final do texto. O texto, que não importa o conteúdo, será de três páginas deste manual.

Macro Documento

A macro @documento também aceita parâmetros

Chamada da macro documento com parâmetros

```
[#macro documento formato="A4" orientacao="retrato" margemEsquerda="3cm"
margemDireita="2cm" margemSuperior="1cm" margemInferior="2cm"]
```

<p>3 cm</p>	<p>1 cm</p> <p></p> <p>PODER JUDICIÁRIO JUSTIÇA FEDERAL SEÇÃO JUDICIÁRIA DO RIO DE JANEIRO</p> <p>XPTODOC Nº NOVO</p> <p>De:Seção de Sistemas Especializados Para: Subsecretaria de Tecnologia da Informação e de Comunicações Assunto: Documentos operacionais referentes à modernização administrativa</p> <p>Srs. aqui fazemos a abertura do DOM 3.3 ? Document Object Model (DOM)</p> <p>Outra buzzword. DOM (Document Object Model - Modelo de Objetos de Documentos) é uma especificação da W3C, independente de plataforma e linguagem, onde se pode dinamicamente alterar e editar a estrutura, conteúdo e estilo de um documento. A API DOM oferece uma maneira padrão de se acessar os elementos de um documento, além de se poder trabalhar com cada um desses elementos separadamente, e por esses motivos criar páginas altamente dinâmicas.</p> <p>(http://pt.wikipedia.org/wiki/Modelo_de_Objetos_de_Documentos)</p> <p>Desta forma, podemos concluir que o JS não seria (faria) nada sem o DOM. Todos os efeitos que vemos em sites, tal como aparecer um texto / gráfico / imagem (ou desaparecer) sem a renderização da página é feita pelo DOM. Os eventos de passagem de mouse, foco no campo, tecla pressionada, e outros como veremos baixo,também é capturado pelo DOM. O JS, como outras, é a linguagem que se utiliza das APIs do DOM.</p> <p>Observação: temos HTML DOM e XML DOM.</p> <p>Novamente repito a velha ladinha, o objetivo desta seção não é ensinar DOM e sim chamar atenção para esta tecnologia. A internet está cheia de material sobre o assunto.</p> <p>3.3.1 ? Eventos HTML DOM</p> <p>Os eventos formam a base da programação orientada a eventos utilizada na programação cliente tradicional (visual basic, delphy e etc.).</p> <p>Estes eventos podem ser capturados pelo próprio HTML ou pelo JS.</p> <p>Exemplos:</p> <p>Via HTML:<input id="lotacaoDestinatarioSelButton" type="button" theme="simple" onclick="popup_lotacaoDestinatario();" value="...">></p> <p>Se clicarmos no botão a função popup_lotacaoDestinatario() será executada.</p> <p>Elaborado por: Ruben Rose</p> <p>Atenciosamente,</p> <p>Rio de Janeiro, 13 de junho de 2012.</p> <p style="text-align: right;">[Classif. documental pp.01.01.02]</p>	<p>2 cm</p>
--------------------	--	--------------------

9.4.2.1 - Utilizando a macro formulário (nível 4)

Parâmetros e defaults: texto, fecho="", tamanhoLetra="Normal", _tipo="FORMULÁRIO"

```
[@entrevista]
[@texto var="nome" titulo="Nome" largura="20" maxcaracteres="20"]
[/@entrevista]

[@documento]
[#assign texto_formulario]
HTML
[/#assign]
[@formulario texto=texto_formulario fecho="Atenciosamente,"]
[/@documento]
```

The diagram illustrates the structure of a document page, divided into two main sections: 'Cabeçalho da primeira página com logo e descrição centralizado' (Header of the first page with logo and centered description) and 'Cabeçalho da página subsequente' (Header of the subsequent page).

Left Section (Header of the first page):

- Tipo: Formulário a esquerda**: A box pointing to the left margin of the page.
- Content**: Includes the logo of the Poder Judiciário, Justiça Federal, Seção Judiciária do Rio de Janeiro, the text 'FORMULÁRIO Nº NOVO', and the question '3.3 ? Document Object Model (DOM)'.
- Text below content**: 'Outra buzzword. DOM (Document Object Model - Modelo de Objetos de Documentos) é uma especificação da W3C, independente de plataforma e linguagem, onde se pode dinamicamente alterar e editar a estrutura, conteúdo e estilo de um documento. A API DOM oferece uma maneira padrão de se acessar os elementos de um documento, além de se poder trabalhar com cada um desses elementos separadamente, e por esses motivos criar páginas altamente dinâmicas.' (http://pt.wikipedia.org/wiki/Modelo_de_Objetos_de_Documentos)
- Text below footer**: 'Desta forma, podemos concluir que o JS não seria (faria) nada sem o DOM. Todos os efeitos que vemos em sites, tal como aparecer um texto / gráfico / imagem (ou desaparecer) sem a renderização da página é feita pelo DOM. Os eventos de passagem de mouse, foco no campo, tecla pressionada, e outros como veremos abaixo, também é capturado pelo DOM. O JS, como outras, é a linguagem que se utiliza das APIs do DOM.'
- Observação:** 'temos HTML DOM e XML DOM.'
- Text below observation**: 'Novamente repto a velha ladainha, o objetivo desta seção não é ensinar DOM e sim chamar atenção para esta tecnologia. A internet está cheia de material sobre o assunto.'
- Text below observation**: '3.3.1 ? Eventos HTML DOM'
- Text below footer**: 'Os eventos formam a base da programação orientada a eventos utilizada na programação cliente tradicional (visual basic, delphi e etc.).'
- Text below footer**: 'Estes eventos podem ser capturados pelo próprio HTML ou pelo JS.'
- Text below footer**: 'Clicar document [00.01.01.02]'

Right Section (Header of the subsequent page):

- Cabeçalho da página subsequente**: A box pointing to the top right of the page.
- Content**: Includes the logo of the Poder Judiciário, Justiça Federal, Seção Judiciária do Rio de Janeiro, the text 'Atenciosamente,' (with an arrow pointing to it), 'RUBEN EDWARD ROSE JUNIOR', 'ANALISTA JUDICIARIO/INFORMATICA', and the number '2'.
- Text below content**: 'Exemplos:'
 - Via HTML: <input id="lotacaoDestinatarioSelButton" type="button" theme="simple" onclick="popupup_lotacaoDestinatario()"; value="..."/>
 - Via JS: document.FormName.ButtonName.onclick= function() {alert('Here is a pop up message');}
- Text below footer**: 'Onde: formname é nome do formulário e buttonname é nome do botão.'
- Text below footer**: 'Os eventos tratados pelo HTML são mais comuns.'
- Text below footer**: 'Evento Descrição'
- Text below footer**: 'Elaborado por: Ruben Rose'
- Text below footer**: 'Assinatura'
- Text below footer**: 'Rodapé com número de página'

9.4.2.2 - Utilizando a macro memorando (nível 4)

Parâmetros e defaults: texto; fecho=""; tamanhoLetra="Normal"; _tipo="MEMORANDO

```
[@entrevista]
[@texto var="nome" titulo="Nome" largura="20" maxcaracteres="20"]
[/@entrevista]

[@documento]
[#assign texto_memorando]
HTML
[/#assign]
[@memorando texto=texto_memorando fecho="Atenciosamente, "/]
[/@documento]
```

A diferença para o anterior (formulário) está nos seguintes campos:

DE: (Subscritor)
PARA: (Destinatário)
ASSUNTO: (Descrição da classificação documental)

PODER JUDICIÁRIO
JUSTIÇA FEDERAL
SEÇÃO JUDICIÁRIA DO RIO DE JANEIRO

MEMORANDO Nº NOVO

De: Seção de Sistemas Especializados
Para: Coordenadoria de Sistemas de Informação
Assunto: Documentos operacionais referentes à modernização administrativa

3.3 ? Document Object Model (DOM)

Outra buzzword, DOM (Document Object Model - Modelo de Objetos de Documentos) é uma especificação da W3C, independente de plataforma e linguagem, onde se pode dinamicamente alterar e editar a estrutura, conteúdo e estilo de um documento. A API DOM oferece uma maneira padrão de se acessar os elementos de um documento, além de se poder trabalhar com cada um desses elementos separadamente, e por esses motivos criar páginas altamente dinâmicas.
(http://pt.wikipedia.org/wiki/Modelo_de_Objetos_de_Documentos)

Desta forma, podemos concluir que o JS não seria (faria) nada sem o DOM. Todos os efeitos que vemos em sites, tal como aparecer um texto / gráfico / imagem (ou desaparecer) sem a renderização da página é feita pelo DOM. Os eventos de passagem de mouse, foo no campo, tecla pressionada, e outros como veremos abaixo, também é capturado pelo DOM. O JS, como outras, é a linguagem que se utiliza das APIs do DOM.

Observação: temos HTML DOM e XML DOM.

Novamente repito a velha ladainha, o objetivo desta seção não é ensinar DOM e sim chamar atenção para esta tecnologia. A internet está cheia de material sobre o assunto.

3.3.1 ? Eventos HTML DOM

Clique no link para baixar o arquivo: [Clique aqui](#)

Poder Judiciário
Justiça Federal
Seção Judiciária do Rio de Janeiro

Os eventos formam a base da programação orientada a eventos utilizada na programação cliente tradicional (visual basic, delphi e etc.).

Estes eventos podem ser capturados pelo próprio HTML ou pelo JS.

Exemplos:

Via HTML: `<input id="lotacaoDestinatarioSelButton" type="button" theme="simple" onclick="popup_lotacaoDestinatario()"; value="... ">`

Se clicarmos no botão a função `popup_lotacaoDestinatario()` será executada.

Via JS: `document.FormName.ButtonName.onclick= function() {alert('Here is a pop up message');};`

Onde: `formname` é nome do formulário e `buttonname` é nome do botão.

Elaborado por: Ruben Rose

Atenciosamente,

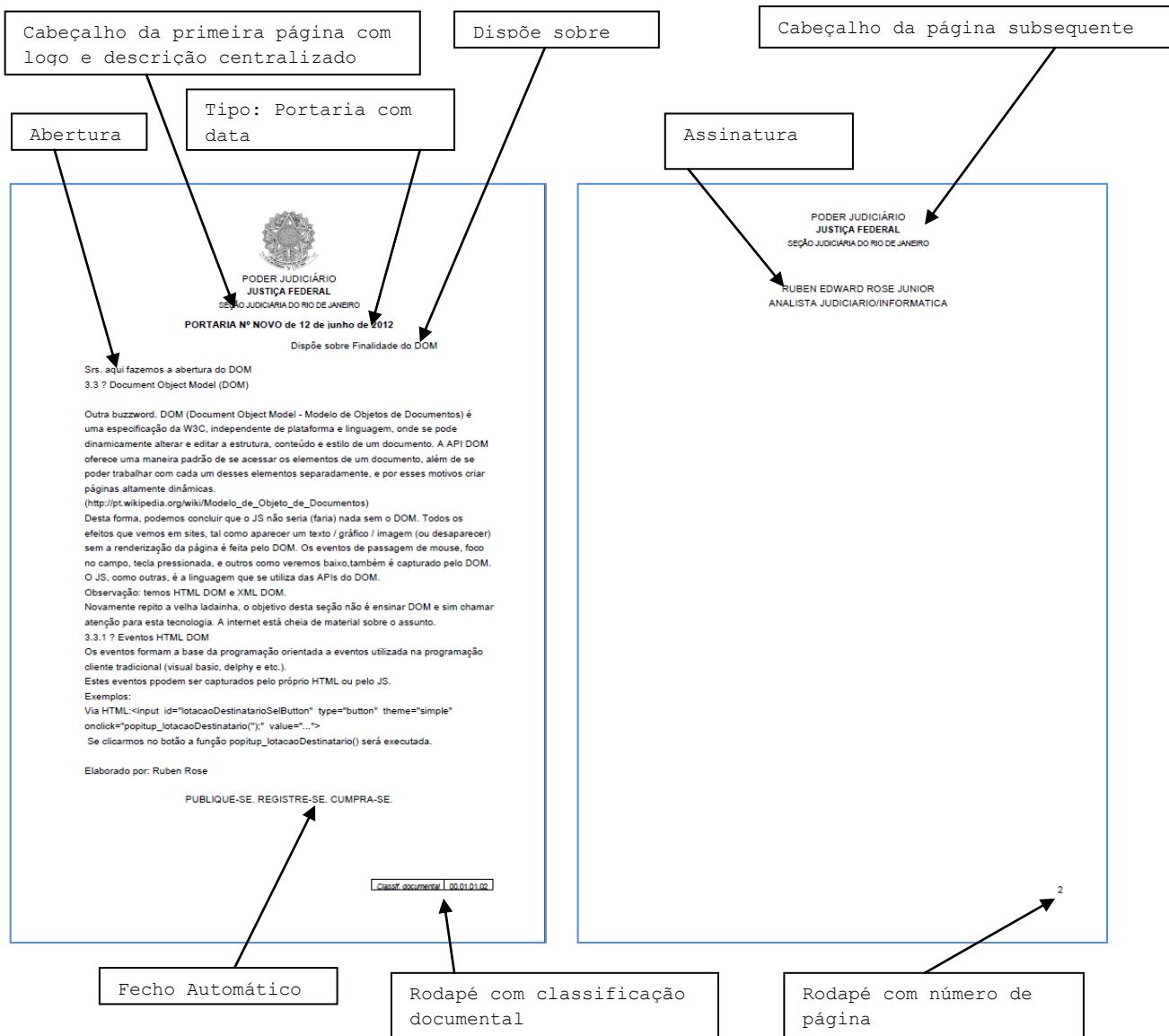
RUBEN EDWARD ROSE JUNIOR
ANALISTA JUDICIÁRIO/INFORMATICA

9.4.2.3 - Utilizando a macro portaria (nível 4)

Parâmetros e defaults: texto; abertura=""; tamanhoLetra="Normal"; _tipo="PORTARIA"; dispoe_sobre=""

```
[@entrevista]
[@texto var="nome" titulo="Nome" largura="20" maxcaracteres="20"]
[/@entrevista]

[@documento]
[#assign texto_portaria]
HTML
[/#assign]
[@portaria texto=texto_portaria abertura="Srs. aqui fazemos a abertura do DOM"
tamanhoLetra="Normal" _tipo="PORTARIA" dispoe_sobre="Finalidade do DOM"]
[/@documento]
```



A macro portaria também gera informações para outros sistemas como o Boletim Interno Eletrônico (BIE) e Diário da Justiça Eletrônico (DJE).

aberturaBIE: recebe o parâmetro abertura

corpoBIE: recebe o parâmetro texto

fechoBIE: é o famoso **PUBLIQUE-SE. REGISTRE-SE. CUMPRA-SE**

mioloDJE: recebe o parâmetro dispõe-sobre e engloba a aberturaBIE, corpoBIE e fechoBIE.

Trecho da macro portaria que menciona a BIE e DJE:

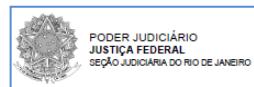
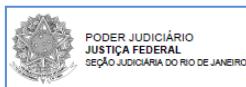
```
[@mioloDJE]
[#if dispoe_sobre != ""]


|                                                                                                                                                           |                                     |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------|
| </td> <td &gt;&lt;br="" &gt;<b="" align="left" style="font-family: Arial; font-size: \${tl};" width="50%">Dispõe sobre \${dispoe_sobre!}&lt;/td&gt; </td> | Dispõe sobre \${dispoe_sobre!}</td> |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------|


[/#if]
<div style="font-family: Arial; font-size: ${tl};">
[#if abertura != ""]
[@aberturaBIE]
${abertura!}
[@aberturaBIE]
[/#if]
[@corpoBIE]
${texto!}
[@corpoBIE]
<span style="font-family: Arial; font-size: ${tl}"><center>
[@fechoBIE]
PUBLIQUE-SE. REGISTRE-SE. CUMPRA-SE.</center></span></p>
[@fechoBIE]
</center></span></p>

```

9.4.2.4 - Estilos de Brasão



Qual a sua preferência:
Esquerda, Centro ou Direita?

Eu adoro Brasão ... Uso qq. um.

O estilo pode ser a esquerda ou centralizado. A direita não existe. O mais comum é o centralizado. As macros anteriores (Portaria, Memorando e Formulário) utilizam o estilo Brasão Centralizado, que é uma macro que pode ser customizada também:

O que é a Macro estiloBrasaoCentralizado?

A chamada da macro se dá na seguinte forma:

```
[@estiloBrasaoCentralizado tipo=_tipo tamanhoLetra=tl formatarOrgao=false
numeracaoCentralizada=true]
```

Formatação, incluindo Abertura, Texto, Fecho ...

```
[@estiloBrasaoCentralizado]
```

Macro estiloBrasaoCentralizado: parâmetros e estrutura

Parâmetros	Estrutura do documento
tipo tamanhoLetra="11pt" formatarOrgao=true numeracaoCentralizada=false dataAntesDaAssinatura=false Expicação: formatarOrgao=true: provoca a inserção da lotação abaixo da assinatura. numeracaoCentralizada=true: não funciona, ou seja, a numeração é sempre a direita. dataAntesDaAssinatura=true: provoca a inserção da data (Rio de janeiro) antes (acima) da assinatura.	<i>Macros Chamadas:</i> primeiroCabecalho - nível 1 cabecalhoCentralizadoPrimeiraPagina - nível 2 cabecalho - nível 1 cabecalhoCentralizado - nível 2 numeroDJE tituloDJE assinaturaCentro ou assinaturaMovCentro - nível 2 primeiroRodape - nível 1 rodapeClassificacaoDocumental - nível 2 rodape - nível 1 rodapeNumeracaoDireita - nível 2

Podemos observar que podemos customizar também se a numeração será centralizada ou não e se devemos ter a data antes da assinatura

O que é a Macro estiloBrasaoAEsquerda?

A chamada da macro se dá na seguinte forma:

```
[@estiloBrasaoAEsquerda tipo=_tipo tamanhoLetra=tl obs=""]
```

Formatação, incluindo Abertura, Texto, Fecho ...

```
[@estiloBrasaoAEsquerda]
```

Parâmetros	Estrutura do documento
tipo tamanhoLetra="11pt" obs="" Expicação: obs: provoca a inserção de um comentário após a assinatura.	<i>Macros Chamadas:</i> primeiroCabecalho - nível 1 cabecalhoEsquerdaPrimeirapagina - nível 2 cabecalho - nível 1 cabecalhoEsquerda - nível 2 assinaturaCentro - nível 2 primeiroRodape - nível 1 rodapeClassificacaoDocumental - nível 2 rodape - nível 1 rodapeNumeracaoDireita - nível 2

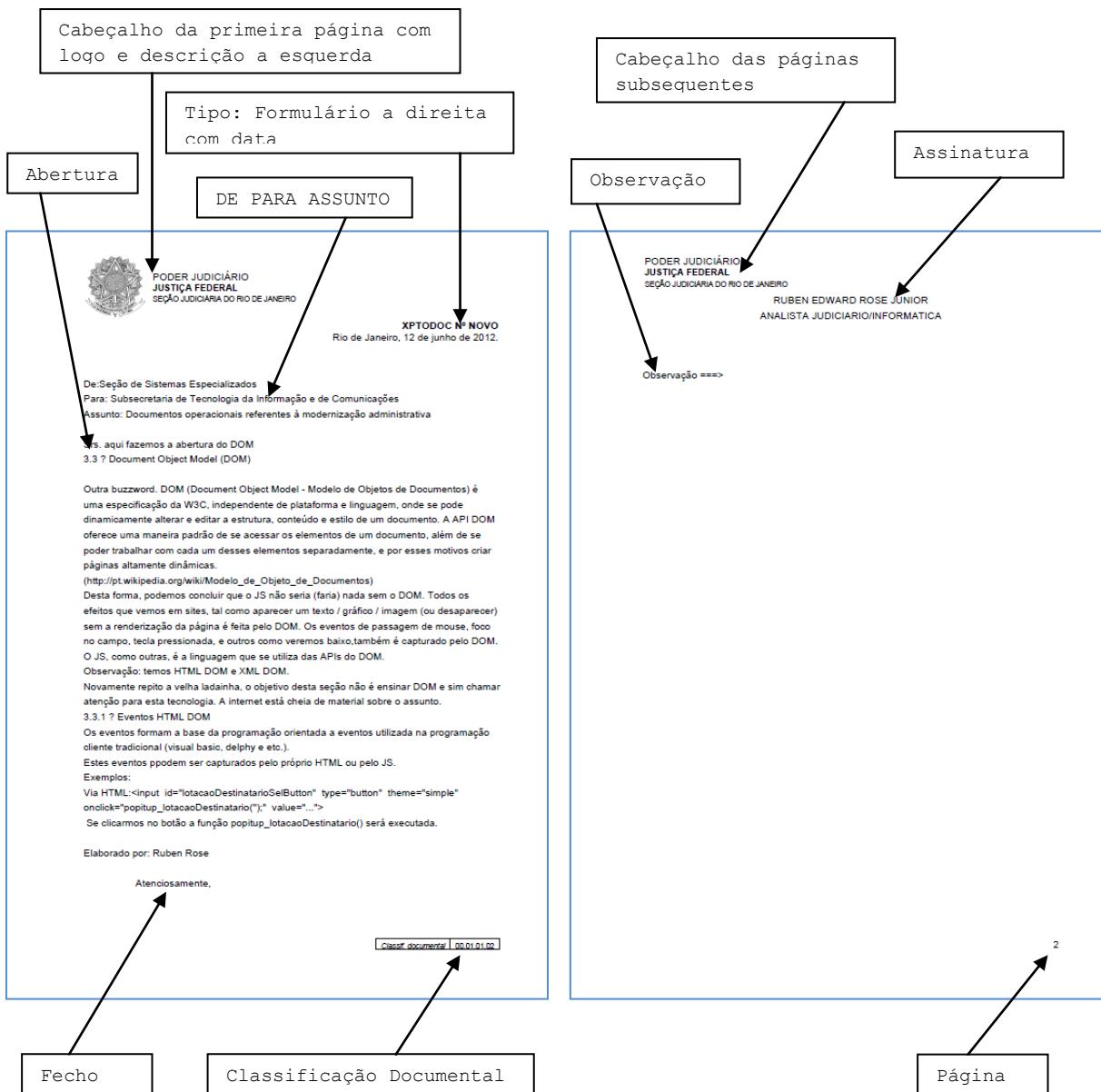
9.4.2.5 - Montando uma macro (nível 4) para o documento

Vamos criar uma Macro chamada xptodoc:

A - Utilizando o estilo Brasão a Esquerda

```
[#macro xptodoc texto abertura="Frase de abertura" fecho="Atenciosamente,"  
tamanhoLetra="Normal" _tipo="XPTODOC"]  
[!-- Definindo o tamanho da letra --]  
[#if tamanhoLetra! == "Normal"]  
[#assign tl = "11pt"/>]  
[#elseif tamanhoLetra! == "Pequeno"]  
[#assign tl = "9pt"/>]  
[#elseif tamanhoLetra! == "Grande"]  
[#assign tl = "13pt"/>]  
[#else]  
[#assign tl = "11pt"]  
[/#if]  
[!-- Cabeçalho --]  
[@estiloBrasaoAEsquerda tipo=_tipo tamanhoLetra=tl obs="Observação ===>"]  
[!-- DE PARA ASSUNTO --]  
<p align="left">  
De: [#if (doc.nmLotacao)??]${doc.nmLotacao} [#else]${(doc.titular.lotacao.nomeLotacao)!} [/#if]<br/>  
Para: ${(doc.destinatarioString)!}<br>  
Assunto: ${(doc.exClassificacao.descrClassificacao)!}</p>  
[!-- Abertura --]  
[#if abertura != ""]  
${abertura!}  
[/#if]  
[!-- Texto --]  
<span style="font-size: ${tl}"> ${texto!} </span>  
[!-- Fecho --]  
<p style="align: justify; TEXT-INDENT: 2cm">${fecho}</p>  
[/@estiloBrasaoAEsquerda]  
[/#macro]
```

Este foi o resultado da nossa macro xptodoc:



B - Utilizando o estilo Brasão Centralizado

```
[#macro xptodoc texto abertura="Frase de abertura" fecho="Atenciosamente," tamanhоЛetra="Normal" _tipo="XPTODOC"]
[!-- Definindo o tamanho da letra --]
[#if tamanhоЛetra! == "Normal"]
[#assign tl = "11pt" /]
[#elseif tamanhоЛetra! == "Pequeno"]
[#assign tl = "9pt" /]
[#elseif tamanhоЛetra! == "Grande"]
[#assign tl = "13pt" /]
[#else]
[#assign tl = "11pt"]
[/#if]
```

```

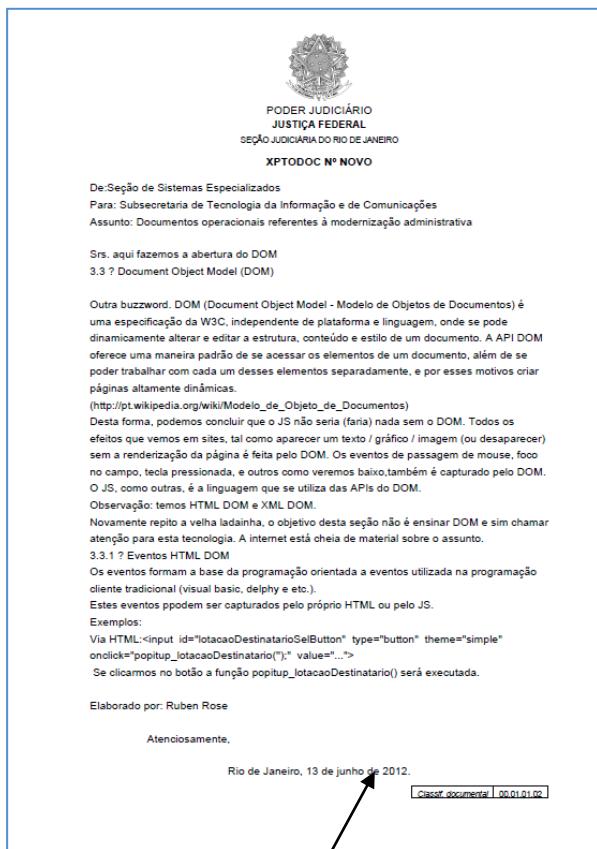
[--- Cabeçalho ---]
[@estiloBrasaoCentralizado tipo=_tipo tamanhoLetra=tl formatarOrgao=true
numeracaoCentralizada=true dataAntesDaAssinatura=true]
[--- DE PARA ASSUNTO --]
<p align="left">
De: ${doc.nmLotacao} ${doc.titular.lotacao.nomeLotacao}! [/if]<br>
Para: ${doc.destinatarioString}! <br>
Assunto: ${doc.exClassificacao.descrClassificacao}! </p>
[--- Abertura --]
[#if abertura != ""]
${abertura!}
[/if]
[--- Texto --]
<span style="font-size: ${tl}"> ${texto!} </span>
[--- Fecho --]
<p style="align: justify; TEXT-INDENT: 2cm">${fecho}</p>
[@estiloBrasaoCentralizado]
[/macro]

```

Observação:

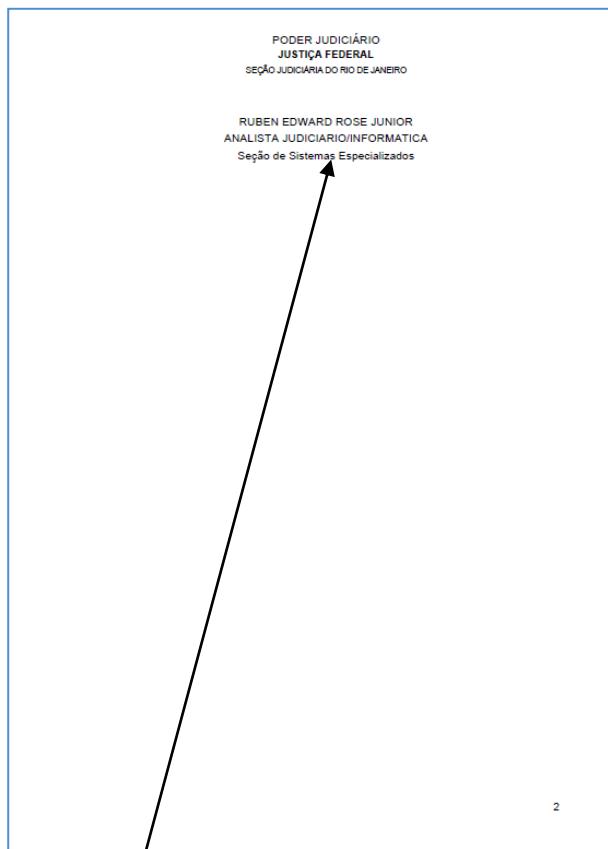
Utilizamos os seguintes parâmetros na chamada do estilo: `formatarOrgao=true`
`numeracaoCentralizada=true` `dataAntesDaAssinatura=true`

As diferenças em relação ao modelo anterior, além do cabeçalho centralizado, foi a inserção da data antes da assinatura e a inserção da lotação após (abaixo) a assinatura.



Rio de Janeiro, 13 de junho de 2012.

Cassoff_documental_00.01.01.02.



Assinatura

Lotação / Órgão

9.5 – FAZENDO MACROS

Bug! E agora?

Como fazer para rastrear o problema? Que ferramentas podem me auxiliar?



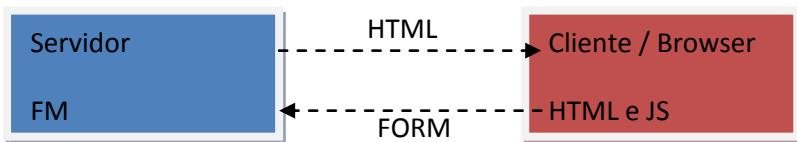
Inexoravelmente sua aplicação não rodará de plano.

Bom, isso pode verdade. Porém, ao ferranteas tem sido processo de

parecer pessimismo, mas é a pura longo dos anos diversas desenvolvidas para nos ajudar no depuração.



Como programadores de interface temos dois ambientes que devemos nos ater:



10.1 – Depuração no servidor

10.1.1- FM

Infelizmente, no FM não temos um ambiente de debug, como no JSP, o que torna o processo de resolução mais demorado, cansativo e até, frustrante.

A técnica, hoje, consiste de se colocar traps (do tipo: \${minhavar} ou "passei neste IF") no programa para exibir o conteúdo das variáveis que desejamos investigar ou sabermos por onde estamos passando (trace). Muitas vezes desejamos exibir várias variáveis em diversos pontos, o que é um trabalho maçante. Para amenizar o problema, desenvolvi a macro @dumpall, cuja descrição segue abaixo.

10.1.1.1 - A MACRO @dumpall (sem parâmetro)

Esta macro lista as variáveis (objetos), os respectivos tipos e os conteúdos (o conteúdo somente se a variável for escalar: numérica, string, date e boolean). É uma ferramenta útil para fins de debug, visto que não há necessidade de se ficar colocando traps do tipo \${varquequeropesquisar} no código.

Obs: existe a macro @dump, que é uma forma simplificada do @dumpall, pois só lista as variáveis do formulário (que é uma parte do data-model).

Tipificação (tipos de objetos listados pela macro):

```
is_method
is_enumerable
is_hash_ex    //hash extendido
is_number
is_string
is_boolean
is_date
is_transform
is_macro
is_hash
is_node
```

Objetos listados:

Data Model, o Hash Root, o hash (classe) func, o hash (classe) exbl, o hash (classe) doc, as variáveis (criadas via #assign) do template, as variáveis locais das macros / function criadas via #local

Diagramaticamente:

DATA MODEL	AMBIENTE DE EXECUÇÃO
root + Func (hash) + Exbl (hash) + Doc (hash) + Outras variáveis	+-----+ Global +-----+ +-----+ Template sendo executado Main +-----+ Macro ou Function Local +-----+ +-----+

- ⇒ Cada template roda no NAMESPACE default chamado Main, onde constam as variáveis definidas via diretiva Assign.
- ⇒ Nas macros e functions são permitidas a criação de variáveis locais via diretiva Local
- ⇒ As variáveis definidas via diretiva Global podem ser utilizadas em todo o ambiente do freemarker
- ⇒ O DATA MODEL e o Hash Root são idênticos

Aplicabilidade: Supor a aplicação abaixo. Tudo bem, ela é muito simples e conseguimos debugá-la no olho, porém serve ao propósito de exemplo. Supor que estamos com problema na variável dataformat. Para tanto, vamos introduzir ma macro @dumpall após a declaração da variável.

```
[@selecao var="catFuncionario" titulo="Categoria do Funcionário"  
reler=true opcoes="Servidor; Terceiro"]  
[@br]  
[@texto titulo="Ano Posse/Contratação" var="ano" largura="4" maxcaracteres="4"  
obrigatorio="Sim" reler="ajax" idAjax="anoAjax"]  
[@grupo depende="anoAjax"]  
[#if (ano!="") == ""]  
    [@mensagem titulo="Alerta" texto="Ano deve ser preenchido."  
    vermelho=true]  
[/#if]  
[@/grupo]  
[@lotacao titulo="Lotação" var="lotacao" reler=true idAjax="lotacaoAjax"]  
[@br]  
[@data var="dataformat" titulo="Data de Formatura:"]  
[@dumpall]  
[@br]  
[@memo var="informacoes" titulo="Dados sobre a Formação" colunas="63" linhas="3"]  
[@checkbox var="fazertrein" titulo="Deseja fazer treinamento?" default="Sim"]  
[@br]  
[@grupo titulo="Turno de Preferência"]  
[@radio titulo="Manhã" var="radio_resp" valor="1" default="Manhã" ]  
[@radio titulo="Tarde" var="radio_resp" valor="0" ]
```

[/@grupo]

Após rodar a aplicação, conseguiremos a tela abaixo, logo após ao campo dataformat da entrevista. Os campos em turquesa foram destacados por mim, e correspondem aos campos do formulário de entrevista. Os destaques em amarelo são para ser analisados com atenção:

Cada template roda no NAMESPACE default chamado Main, onde constam as variáveis definidas via diretiva Assign

Nas macros e functions são permitidas a criação de variáveis locais via diretiva Local
As variáveis definidas via diretiva Global podem ser utilizadas em todo o ambiente do freemarker

O DATA MODEL e o Hash Root são idênticos

===== VARIÁVEIS DO DATA MODEL =====

```
lotacao_lotacaoSel.id = ""
subscritorSel.sigla = ""
root = ?? (hash)
classificacaoSel.descricao = ""
lotacao_lotacaoSel.descricao = ""
mobilPaiSel.descricao = ""
reqmobilPaiSel = ""
fazertrein = "Sim"
classificacaoSel.sigla = ""
lotacaoDestinatarioSel.id = ""
ano = ""
reqsubscritorSel = ""
tipoDestinatario = "2"
func = ?? (hash)
exbl = ?? (hash)
preenchimento = "0"
alterouModelo = ""
template = "[@selecao var="catFuncionario" titulo="Categoria do Funcionário" reler=true
opcoes="Servidor; Terceiro"] [@br/] [@texto titulo="Ano Posse/Contratação" var="ano"
largura="4" maxcaracteres="4" obrigatorio="Sim" reler="ajax" idAjax="anoAjax"] [@grupo
depende="anoAjax"] [#if (ano!="") == ""] [@mensagem titulo="Alerta" texto="Ano deve ser
preenchido." vermelho=true/] [/#if] [@grupo] [@lotacao titulo="Lotação" var="lotacao"
reler=true idAjax="lotacaoAjax"] [@br/] [@data var="dataformat" titulo="Data de
Formatura:/"] [@br/] [@memo var="informacoes" titulo="Dados sobre a Formação" colunas="63"
linhas="3" /] [@checkbox var="fazertrein" titulo="Deseja fazer treinamento?"
default="Sim"] [@br/] [@grupo titulo="Turno de Preferência"] [@radio titulo="Manhã"
var="radio_resp" valor="1" default="Manhã"] [@radio titulo="Tarde" var="radio_resp"
valor="0"] [/@grupo] [@br/] [@dumpall/] "
idFormaDoc = "60"
reqlotacao_lotacaoSel = "sim"
fazertrein_chk = "Sim"
mobilPaiSel.id = ""
mobilPaiSel.buscar = ""
webwork.token = "NV5YPPID3TEUA02T3S3RHIUNT2K390V4"
gerar_entrevista = true
alterouSel = "lotacao,,,,,"
lotacaoDestinatarioSel.descricao = ""
lotacao_lotacaoSel.buscar = ""
desativarDocPai = ""
nmArqMod = ?? (enumerável)
titularSel.buscar = ""
vars =
"catFuncionario,ano,lotacao_lotacaoSel.id,lotacao_lotacaoSel.sigla,lotacao_lotacaoSel.desc
ricao,dataformat,informacoes,fazertrein,radio_resp"
mobilPaiSel.sigla = ""
serverAndPort = "localhost:8080"
informacoes = ""
titularSel.descricao = ""
reqtitularSel = ""
idTpDoc = "1"
lotacaoDestinatarioSel.sigla = ""
```

A variável **template** contém a aplicação FM

A variável **vars** contém o nome de todas as variáveis do formulário de entrevista da parte variável

```

radio_resp = ""
subscritorSel.buscar = ""
nmMod = "RubenTeste"
reqclassificacaoSel = ""
param = ?? (hash)
classificacaoSel.id = ""
campos =
"despachando,criandoAnexo,idTpDoc,dtDocString,nivelAcesso,eletronico,subscritorSel.id,subs
tituicao,titularSel.id,nmFuncaoSubscritor,tipoDestinatario,lotacaoDestinatarioSel.id,preen
chimento,classificacaoSel.id,descrDocumento"
lotacao_lotacaoSel.sigla = ""
nomePreenchimento = ""
obrigatorios = "ano"
titularSel.id = ""
subscritorSel.id = ""
classificacaoSel.buscar = ""
reqlotacaoDestinatarioSel = ""
_FALSE_.substituicao = "false"
subscritorSel.descricao = ""
catFuncionario = "Servidor"
entrevista = "1"
idMod = "742"
criandoAnexo = "false"
lotacaoDestinatarioSel.buscar = ""
descrDocumento = ""
despachando = "false"
postback = "1"
sigla = ""
lotaTitular = ?? (hash)
nivelAcesso = "1"
titularSel.sigla = ""
doc = ?? (hash)
nmFuncaoSubscritor = ""
webwork.token.name = "webwork.token"
dataformat = ""
dtDocString = ""

=====
===== TEMPLATE SENDO EXECUTADO (NAMESPACE MAIN) =====

```



Aqui serão listados os nomes de todas as macros e functions, assim como dos Assign (não é o assign dentro da macro ou function, e sim dos assign feitos diretamente no arquivo geral - Funcionando como constantes) definidos no arquivo geral que contém as macros.

ATENÇÃO !

```

extensaoAssinadorLote = ?? (macro)
rodape = ?? (macro)
assinatura = ?? (macro)
cabecalhoEsquerdaPrimeiraPagina = ?? (macro)
rodapeNumeracaoADireita = ?? (macro)
cabecalho = ?? (macro)
vertipo = ?? (macro)
extensaoBuscaTextual = ?? (macro)
estiloBrasaoAESquerda = ?? (macro)
numeroDJE = ?? (macro)
br = ?? (macro)
temRadio_radio_resp = true
tituloDJE = ?? (macro)
formatarCPF = ?? (macro)
lotacao = ?? (macro)
secretario_rh = ?? (hash)
formulario = ?? (macro)
rodapeNumeracaoCentralizada = ?? (macro)
enderecamentoDiretorGeral = "Ilmo(a). Sr(a). Diretor(a)-Geral"
portaria = ?? (macro)
assentamento_funcional = ?? (macro)
key = "key"

```

A variável **campos** contém o nome de todas as variáveis do formulário de entrevista da parte fixa

A variável **obrigatorios** contém o nome de todas as variáveis do formulário de entrevista que foram definidas como obrigatório

```

[#assign _secretario_rh = {
"genero":"F",
"vocativo":"Senhora",
"vocativo_carta":"Sra. Diretora da Subsecretaria de Gestão de Pessoas",
"enderecamento":"Sra. Dra.",
"nome":<DEFINIR_NOME>,
"cargo":<DEFINIR_CARGO>,
"orgao":<DEFINIR_ORGAO>,
"endereco":"Avenida Almirante Barroso, 78 - Centro - Rio de Janeiro/RJ - CEP: 20031-004" } /]

```

Obs: Se quisermos acessar um dado deste hash utilizamos a notação: `_secretario_rh.vocatico`, por exemplo, para obter "Senhora".

Quando a macro lotacao foi invocada, esta chama a macro selecionavel que define estas duas variáveis com assign, que conterá dados da última macro chamada (lotacao ou função ou pessoa)

```

finalizacao = ?? (macro)
tipoSel = "lotacao"
varName = "lotacao_lotacaoSel.descricao"
fmtvldCPF = ?? (macro)
grupo = ?? (macro)
msgid = ?? (macro)
texto = ?? (macro)
atualizaoculto = ?? (macro)
estiloBrasaoCentralizado = ?? (macro)
presidente = ?? (hash) ----->
assinaturaCentro = ?? (macro)
editor = ?? (macro)
corpoBIE = ?? (macro)
dadosComplementares = ?? (macro)
quebraPagina = ?? (macro)
primeiroCabecalho = ?? (macro)
assinaturaBIE = ?? (macro)
cabecalhoEsquerda = ?? (macro)
mioloDJE = ?? (macro)
assinaturaMovCentro = ?? (macro)
inlineTemplate = ?? (transform)
separador = ?? (macro)
dump = ?? (macro)
memo = ?? (macro)
oficio = ?? (macro)
validarCPF = ?? (macro)
rodapeClassificacaoDocumental = ?? (macro)
div = ?? (macro)
data = ?? (macro)
fixcrlf = ?? (macro)
caixaSelecao = ?? (macro)
botoesExtensaoAssinador = ?? (macro)
par = ?? (macro)
primeiroRodape = ?? (macro)
obrigatorios = ?? (macro)
funcao = ?? (macro)
requerimento2 = ?? (macro)
enderecamentoPresidente = "Exmo. Sr. Juiz Federal - Diretor de Foro"
memorando = ?? (macro)
oculto = ?? (macro)
aberturaBIE = ?? (macro)
cabecalhoCentralizadoPrimeiraPagina = ?? (macro)
entrevista = ?? (macro)
letra = ?? (macro)
mensagem2 = ?? (macro)
atualizacampo = ?? (macro)
requerimento = ?? (macro)
mensagem = ?? (macro)
selecionavel = ?? (macro)
fechoBIE = ?? (macro)
pessoa = ?? (macro)
enderecamentoDiretorDeRH = "Ilma. Sra. Diretora da Subsecretaria de Gestão de Pessoas"
documento = ?? (macro)
resumo = ?? (macro)
secretario_geral = ?? (hash) ----->
cabecalhoCentralizado = ?? (macro)
identificacao = ?? (macro)
selecao = ?? (macro)
topico = ?? (macro)
sec = ?? (macro)
item = ?? (macro)
checkbox = ?? (macro)
estiloBrasaoCentralizadoSEC = ?? (macro)
radio = ?? (macro)
extensaoEditor = ?? (macro)
extensaoAssinador = ?? (macro)
XStandard = ?? (macro)

```

```

[#assign _presidente = {
"genero":"M",
"vocativo":"Excelentíssimo Senhor",
"vocativo_carta":"Exmo. Sr. Juiz Federal
- Diretor do Foro",
"enderecamento":"Exmo. Sr. Dr.",
"nome": "<DEFINIR_NOME>",
"cargo": "<DEFINIR_CARGO>",
"orgao": "<DEFINIR_ORGAO>",
"endereco": "Avenida Almirante Barroso,
78 / 13º andar - Centro - Rio de Janeiro/RJ - CEP: 20031-004"}]

```

Esta variável é utilizada pelas macros checkbox, radio e seleção. Ela conterá trechos de código FM que são interpretados e executados dinamicamente

```

[#assign _secretario_geral = {
"genero":"F",
"vocativo":"Senhora",
"vocativo_carta":"Sra. Diretora da Secretaria Geral",
"enderecamento":"Sra. Dra.",
"nome": "<DEFINIR_NOME>",
"cargo": "<DEFINIR_CARGO>",
"orgao": "<DEFINIR_ORGAO>",
"endereco": "Avenida Almirante Barroso,
78 - Centro - Rio de Janeiro/RJ - CEP: 20031-004"}]

```

===== VARIÁVEIS DO HASH ROOT (Idêntica do DATA_MODEL) =====

... Não listarei aqui

===== VARIÁVEIS DO HASH (DA CLASSE) FUNC =====

```
hashCode = ?? (method)
quebraLinhas = ?? (method)
contains = ?? (method)
has = ?? (method)
calculaData = ?? (method)
formatarCPF = ?? (method)
lotacaoPessoa = ?? (method)
... aqui neste documento não listarei todos métodos
```

===== VARIÁVEIS DO HASH (DA CLASSE) EXBL =====

```
cancelarJuntada = ?? (method)
excluirMovimentacao = ?? (method)
receberEletronico = ?? (method)
processar = ?? (method)
pedirPublicacao = ?? (method)
... aqui neste documento não listarei todos métodos
```

===== VARIÁVEIS DO HASH (DA CLASSE) DOC =====

```
conteudoBlobForm = "?? (method)"
setExNivelAcesso = "?? (method)"
isAssinadoEletronicoPorTodosOsSignatarios = "?? (method)"
hashCode = "?? (method)"
getAnoEmissao = "?? (method)"
... aqui neste documento não listarei todos métodos
```

===== VARIÁVEIS DEFINIDAS COMO LOCAL =====

dump_var = "variável local criada para não gerar erro na rotina de listar as variáveis locais"

10.1.1.2 - Depurando o template (aplicação) FM utilizando o utilitário Integrador

Ver seção “12.6 – Integrador” para obter todos os detalhes de como depurar um template FM utilizando este utilitário.

10.1.2- Depurando os métodos JAVA

Como já vimos em outras seções, um template FM pode chamar métodos JAVA. Desta forma, podemos utilizar o ambiente da IDE (JBDS, Eclipse ...) para depurar estes métodos.

Supor que tenhamos a seguinte aplicação FM:

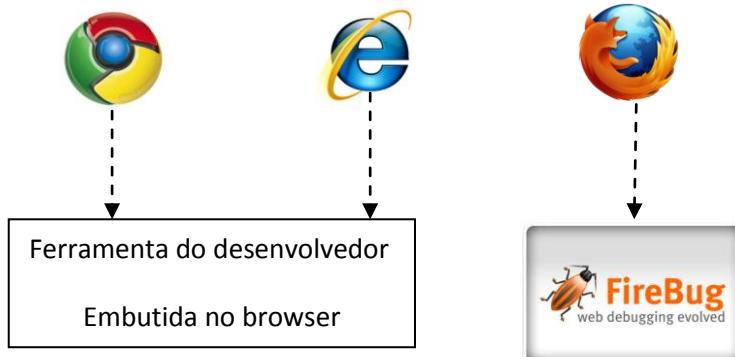
```
[#assign suprid = func.calculaData("22","08/09/2012")]
${suprid}
```

A aplicação FM acima executa o método calculaData() na classe func (FuncoesEl.java). Para mais detalhes sobre este método e outros, consultar a seção “5.1 – VARIÁVEIS DO HASH FUNC – CLASSE FuncoesEl.java”.

Ver “Anexo 3 – Item 7 – Debugando um método JAVA” para obter mais detalhes de como depurar um método na IDE JBOSS Developer Studio.

10.2 – Depuração no cliente

Aqui nós temos boas, ou melhor, excelentes ferramentas para análise / depuração do cliente. Estas ferramentas estão embutidas no próprio browser. No Chrome e no IE, temos a ferramenta do desenvolvedor que já vem instalada. No FF (FireFox) nós temos o Firebug, que temos que baixar e instalar no browser.



10.2.1 – Firebug

A minha preferência é o Firebug, que pode ser obtido em: (<http://getfirebug.com>).

O que podemos analisar no cliente com estas ferramentas?

Para responder a esta pergunta é interessante exibirmos a tela do depurador, que se abre abaixo da página WEB que está sendo depurada. Está longe deste manual explicar o pleno funcionamento desta ferramenta que é extremamente complexo. Farei somente um destaque da coisas importantes.

Tela do entrevista do exemplo anterior, visto em 10.1

Tela do depurador

Clicando no campo da entrevista, o Firebug (FB) já exibe o HTML correspondente, no caso, o cursor está no campo dataformat (Data de Formatura). Reciprocamente, se clicarmos no HTML ele exibe o campo da entrevista correspondente.

Na aba principal do FB temos:

10.2.2 - HTML / Estilo (CSS)

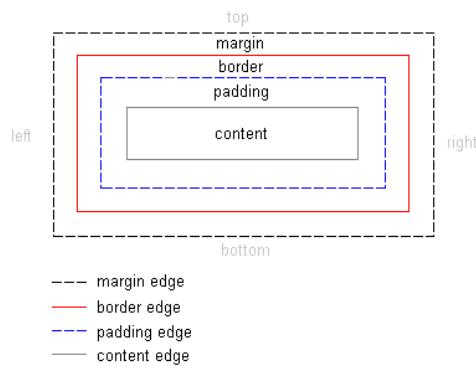
Com o cursor no Ano da Posse, podemos ver na esquerda o HTML correspondente e na direira o CSS (cascateado e herdado) que vigora para o campo em questão.

The screenshot shows the browser's developer tools with the 'HTML' tab selected. On the left, the HTML structure is displayed, including the input field for 'Ano Posse/Contratação'. On the right, the 'Estilo' (Style) panel shows the cascading CSS rules applied to this element. The 'Herdado de td' section indicates that the input inherits styles from its parent table cell. Specific styles like font-size: 11px; and font-family: Verdana, Arial, Helvetica, sans-serif; are listed under 'sigacss.css'.

10.2.3 - HTML / Exibição (Layout) O BOX MODEL

O box model (modelo das caixas) em CSS, descreve os boxes (as caixas) geradas pelos elementos HTML. O box model, detalha ainda, as opções de ajuste de margens, bordas, padding e conteúdo para cada elemento. Abaixo um diagrama representando a estrutura de construção do box model:

O box model em CSS



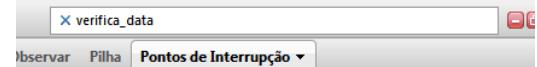
Com o cursor no Ano da Posse, podemos ver na esquerda o HTML correspondente e na direira o Box Model do campo. Observar como o campo foi enquadrado na régua.

This screenshot shows the developer tools with the 'Exibição' (Display) tab selected. It displays both the HTML structure and a detailed representation of the Box Model for the 'Ano Posse/Contratação' input field. The Box Model diagram shows the element's position (static), dimensions (41x15 pixels), and the values for margin (0), border (1 pixel), and padding (2 pixels). The overall box-sizing is set to 'border-box'.

10.2.4 - Script

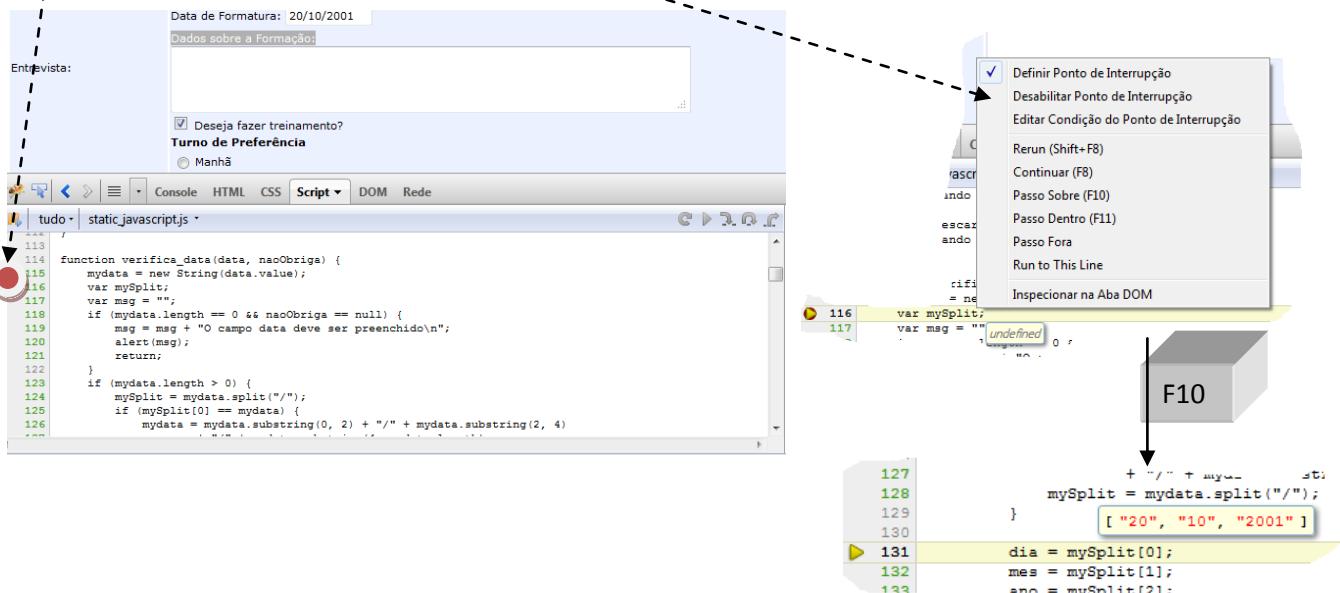
O campo Data de Formatura possui um JS associado a ele, e o mesmo é executado no evento **onchange**. É só observar no HTML da tela do item 10.2.1. A rotina se chama **verifica_data**.

Na aba Script, selecionei o arquivo de script JS chamado **static_javascript.js**. Por quê? Porque é nele que está definido a função que desejamos depurar. Existem 2 formas de descobrir onde a função reside: uma é neste manual, na seção 3.8 e a outra é consultando no próprio FB, selecionando os arquivos .JS na aba Script (ou na aba DOM) e pesquisando via área de pesquisa.



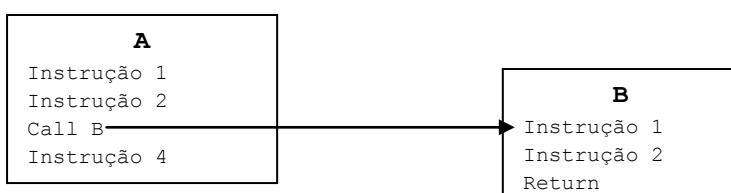
Defini um ponto de interrupção (breakpoint) para analisar a variável **mySplit** (linha 116).

Na entrevista preenchi o campo com a data 20/010/2001. Após o preenchimento a tela abaixo é apresentada, com a parada no local previsto. Com o botão direito do mouse na linha, surgem as seguintes opções:



Na linha 116 o conteúdo de mySplit é indefinido, pois ela está sendo definida. Na linha 128, já conseguimos analisar o seu conteúdo (20, 10, 2001).

Funções do depurador JS:



F8 (continue) – Continuar: continua a executar a aplicação até o próximo breakpoint, caso exista, ou até ao final do código.

Shift + F8 (Rerun) - Volta a executar a função do início.

Terminate - Interrompe o servidor em modo Debug

Resume - Executa o código até o próximo breakpoint, caso exista, ou até o final do código.

Step over - **Passo sobre:** vai executando linha-a-linha, porém se o seu código chama outro método / função (incluindo os métodos / funções da linguagem, tais como: parse.int, string ...) ele não entra. Se na figura acima o debug (em A) está na instrução Call B, ele executará B, porém não entrará no código de B.

Step **over** proceeds to the next line in your current scope (i.e. it goes to the next line), without descending into any method calls on the way. This is generally used for following the logic through a particular method without worrying about the details of its collaborators, and can be useful for finding at what point in a method the expected conditions are violated.

Step into - Passo Dentro. Idêntico ao step over, porém ele entrará nos métodos / funções chamados. Se na figura acima o debug (em A) está na instrução Call B, ele entrará em B, e continuará executando linha-a-linha. A única forma de sair de B é chamar o Step out / Step return.

Step **into** will cause the debugger to descend into any method calls on the current line. If there are multiple method calls, they'll be visited in order of execution; if there are no method calls, this is same as step over. This is broadly equivalent to following every individual line of execution as would be seen by the interpreter.

Step Out / Step Return - termina a execução do método / função e retorna ao chamador na instrução de chamada. Se na figura acima o debug (em A) está na instrução Call B, e o usuário está modo Step into, ele entrará em B, e continuará executando linha-a-linha. A única forma de sair de B diretamente (sem executá-lo) é chamar o Step out / Step return, que fará o debug retornar para A na instrução Call B. Se usuário fornecer Step over, ele executará sem entrar em B, se der Step into, entrará novamente em B, e executará linha-a-linha.

Step out proceeds until the next "return" or equivalent - i.e. until control has returned to the preceding stack frame. This is generally used when you've seen all you need to at thispoint/method, and want to bubble up the stack a few layers to where the value is actually used.

Imagine the following code, which entered through main() and is now on the first line of bar:

Exemplo:

```
function main() {
    val s = foo();
    bar(s);
}

function foo() {
    return "hi";
}

function bar(s) {
```

```

val t = s + foo(); // <== Debugger is currently here
return t;
}

```

Then:

- **Step into (F11)** will proceed into the foo call, and the current line will then become the return "hi"; line within foo.
- **Step over** will ignore the fact that another method is being invoked, and will proceed to the return t; line (which lets you quickly see what t is evaluated as).
- **Step out** will finish the execution of the rest of the bar method, and control will return to the last line of the main method.

Resumo:

Step Into	Step Over	Step Out
<ul style="list-style-type: none"> ➤ Step Into will cause the debugger to go into the next function call and break there. ➤ Go into the subroutine and wait for next action. ➤ Change the debugger context to run into the function the code is stopped on. If the code cannot step into the function, this is the same as Step Over. 	<ul style="list-style-type: none"> ➤ Step Over will tell the debugger to execute the next function and break afterwards. ➤ Jump over the subroutine without waiting again. ➤ Execute the code the debugger is stopped on, but stay within the current function. 	<ul style="list-style-type: none"> ➤ Step Out will tell the debugger to finish the current function and break after it. ➤ If you are in the subroutine, you will leave it without waiting again ➤ Execute code until the end of the current function, and resume debugging once it has returned.

10.2.5 - DOM

Existem dois tipos de objetos e funções:

- Aqueles que são parte do padrão DOM, e
- Aqueles definidos pelo usuário através de JS / DOM.

Firebug exibe os objetos e funções criados pelos usuários, via scripts, em negrito, no topo da lista dos elementos DOM.

O Sistema de cores da Aba DOM:

Black are properties and green are methods. Bold means the member was declared "by the user" meaning the members aren't from the default

Numbers are blue, strings are red. Objects appear as a "instance preview" in which the type name and the member names are green and the member values

```

window
  a1
    ajax
      get
      gets
      post
      send
      serialize
      submit
      update
      x
    ap1
    atag
    b1
    carregando
    cm_obj
    conexaoTimer
    counter
    dd

```

undefined
Object { x=function(), serialize=function(), send=function(), more... }
function()
<TextNode textContent=" ">

Navegando na Hierarquia: Document/Head ou Document/Body:

```

document
  currentScript
  head
    mozFullScreen
    mozFullScreenElement
    mozFullScreenEnabled
    mozHidden
    mozSyntheticDocument
    mozVisibilityState
    onmouseenter
    onmouseleave
    onmozfullscreenerror
    parentElement
  scripts
    contains
    mozCancelFullScreen
    mozSetImageElement
    onmouseover
    releaseCapture
  URL
  activeElement
    alinkColor
  anchors
  applets
    attributes
  baseURI
    bgColor
  body
    characterSet

```

Document editar.action
null
head
false
null
true
false
false
false
"visible"
null
null
null
null
[script /sigaex/..ajax.js, script /sigaex/..cript.js, script, 13 more...]
contains()
mozCancelFullScreen()
mozSetImageElement()
function()
releaseCapture()
"http://localhost:8080/s...iente/doc/editar.action"
body
""
[]
[]
null
"http://localhost:8080/s...iente/doc/editar.action"
""
body
"UTF-8"

Vamos a um exemplo. Com o cursor no Ano da Posse, na esquerda temos o HTML e na direita os atributos DOM, incluindo: o nodo pai, nodos filhos, o array de atributos do elemento e etc.

The screenshot shows a browser's developer tools with the DOM tab selected. On the left, the HTML structure of a form is visible, including various input fields and a text area. On the right, the DOM object for the selected input field is displayed with its properties and methods. Arrows point from specific parts of the UI to the corresponding code and DOM object properties.

```

Categoría do Funcionario: Servidor
Ano Posse/Contratação: [ ] 
Alerta: Ano deve ser preenchido.


Ano Posse/Contratação:

```

DOM Tab Content:

- outerHTML:** null
- parentElement:** selectionDirection
- get validationMessage:** "Ano deve ser preenchido."
- get validity:** ValidityState { constructor=ValidityState, valueMissing=false, typeMismatch=false }
- get willValidate:** true
- checkValidity:** checkValidity()
- contains:** contains()
- insertAdjacentHTML:** insertAdjacentHTML()
- mozIsTextField:** mozIsTextField()
- mozRequestFullScreen:** mozRequestFullScreen()
- onchange:** onchange(event)
- setCustomValidity:** setCustomValidity()
- accept:** ""
- accessKey:** ""
- align:** ""
- alt:** ""
- attributes:**
 - 0:** type="text", maxlength="4", 3 more...
 - 1:** type="text"
 - 2:** maxlength="4"
 - 3:** size="4"
 - 4:** onchange="javascript: sbmt('anoAjax'); value=''"
 - 5:** name="ano"

Podemos alterar em tempo real as propriedades do elemento. Por exemplo: vamos colocar um title com o conteúdo "Ano da Posse do Funcionário". Podemos clicar no outerHTML da aba DOM, ou na aba HTML, clicar no elemento e com o botão direito selecionar "Editar HTML". Podemos então incluir: title="Ano da Posse do Funcionário", clicar em Editar e PRONTO. Se colocarmos o cursor no campo Ano, já veremos a caixa com o título.Tudo em tempo real, sem reload da página.

Através da aba HTML ou DOM, pode-se:

- Alterar o elemento;
- Incluir um novo elemento;
- Excluir um elemento;

Tudo dinamicamente, sem reload da página.

Como saber a estrutura DOM de um elemento?

Às vezes, é importante sabermos qual é o DOM de um elemento para trabalharmos com o JS.

Exemplo: na tela de entrevista, para desenvolver o aplicativo Integrador (Capítulo 12 - Ambiente de Desenvolvimento), necessitei saber o DOM associado ao elemento <td>Entrevista:</td>.

- Primeiro accesei a aba HTML e realizei uma pesquisa com Entrevista;
- Ao achar, cliquei no elemento para obter o XPath (/html/body/table/tbody/tr/td/center/table/tbody/tr/td/form/table/tbody/**tr[13]**/td). Observem o tr[13].;

Pesquisa

```
<form#frm < td < tr < tbody < table < center < td < tr < tbody < table < body < html
+ <tr>
+ <tr>
+ <tr>
+ <tr>
    <td>Entrevista:</td>
    <td colspan="3">
        + /html/body/table/tr/td[13]/td[1] (http://www.w3.org/1999/xhtml)
    </td>
</tr>
+ <tr>
```

- Então tentei acessar o elemento emitindo o seguinte comando no console:
`document.getElementsByTagName("tr") [13];`

```
Limpar Persistente Perfil Tudo Erros Avisos Informações Informação de Depuração Cookies | jQueryify
>>> document.getElementsByTagName("tr") [13];
<tr>
```

Clicando em <TR>, no levará ao elemento na tela

```
<tr < tbody < table < center < td < tr < tbody < table < body < html
+ <tr>
+ <tr>
+ <tr>
+ <tr>
+ <tr>
    <td>Entrevista:</td>
    <td colspan="3">
        + /html/body/table/tr/td[14]/td[1] (http://www.w3.org/1999/xhtml)
    </td>
</tr>
+ <tr>
```

- Observei que o elemento é anterior ao que desejo acessar, então tento o seguinte comando: `document.getElementsByTagName("tr") [14]`, o que me leva ao <TR> Correto. Porém desejo acessar o <td>, e emito o seguinte comando: `document.getElementsByTagName("tr") [14].childNodes[0]`, supondo que ele fosse o primeiro elemento do <TR>;
- Não obtive sucesso, e tentei o comando: `document.getElementsByTagName("tr") [14].childNodes[1]`, obtendo sucesso. Para me garantir, solicitei o innerHTML e o outerHTML.

```
document.getElementsByTagName("tr") [14].childNodes[0].innerHTML
"Entrevista:"
```

```
document.getElementsByTagName("tr") [14].childNodes[1].outerHTML
"<td>Entrevista:</td>"
```

10.2.6 - Rede

Aqui pode-se observar todo o fluxo de dados entre o cliente e servidor. Os métodos, GET ou POST, o HTML recebido pelo cliente, os dados passados ao servidor, o tempo de cada requisição, gargalos e etc.

CASO 1: Requisições durante um AJAX

Vamos ver o que acontece quando o Ano da Posse é preenchido, visto que neste caso, o AJAX será ativado.

Neste exemplo, vamos preencher o campo Ano da Posse:

The screenshot shows the Firebug Network tab with one request listed: POST editar.action. The status is 200 OK, the domain is localhost:8080, and the response size is 80.5 KB. The response content is shown in the HTML tab, displaying the updated form with the year 2011 filled in. The Post tab shows the parameters sent: _FALSE_substituicao: false, alterouModelo, alterouSel, alterouSel, alterouSel, alterouSel, alterouSel, alterouSel, ano: 2011, btnAdicionar: Adicionar, and btnAlterar: Alterar.

O que foi postado para o servidor. Observar as variáveis enviadas, incluindo a data da posse

HTML recebido pelo cliente

Observar que a data foi fornecida, logo não há a mensagem de erro

No AJAX, temos somente 1 Request

The screenshot shows the Firebug Network tab with one request listed: POST editar.action. The status is 200 OK, the domain is localhost:8080, and the response size is 80.5 KB. The response content is shown in the HTML tab, displaying the updated form with the year 2011 filled in. The Post tab shows the parameters sent: _FALSE_substituicao: false, alterouModelo, alterouSel, alterouSel, alterouSel, alterouSel, alterouSel, alterouSel, ano: 2011, btnAdicionar: Adicionar, and btnAlterar: Alterar.

STOP no XMLHttpRequest (XHR)

Observe que a aba mudou para Script automaticamente. Neste ponto pode se feito uma análise do código, variáveis, etc.

The screenshot shows the Firebug Script tab with the file ajax.js open. The code is as follows:

```
342 http://localhost:8080/sigaex/expediente/doc/editar.action
343 You can disable/enable break notifications in panel's tab menu.
344
345     xmlhttp.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
346     xmlhttp.send.ajax.serialize(frm));
347 }
348 }
```

Se voltarmos a preencher o campo Ano da Posse, com o XHR habilitado, a aplicação parará no xmlhttp.send, como mostrado na tela abaixo.

CASO 2: O que ocorre durante um F5 (Reload de página) ou com campo que possua `reler=true`?

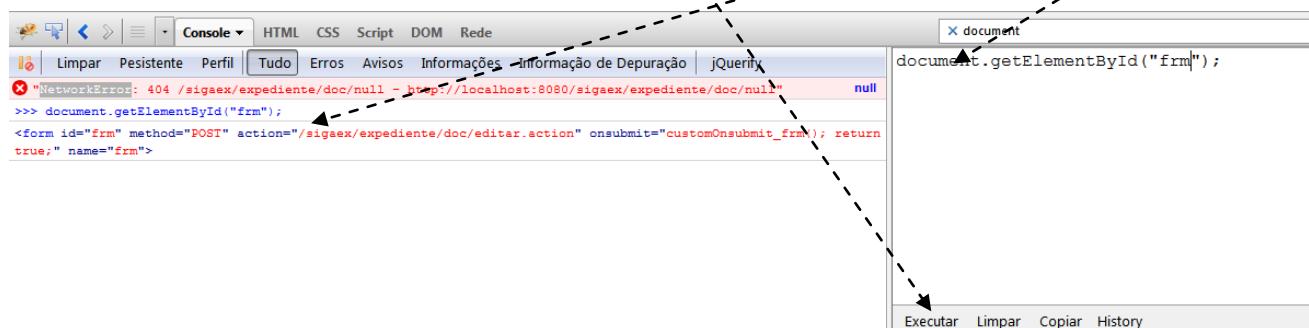
URL	Status	Domínio	Tamanho	Remote IP	Linha de tempo
POST editar.action	200 OK	localhost:8080	80.6 KB	::1:8080	
GET siga.css	304 Not Modified	localhost:8080	14.5 KB	::1:8080	
GET ajax.js	304 Not Modified	localhost:8080	11.2 KB	::1:8080	
GET static_javascript.js	304 Not Modified	localhost:8080	31.4 KB	::1:8080	
GET menu.css	304 Not Modified	localhost:8080	8.7 KB	::1:8080	
GET toplogo1_70.png	304 Not Modified	localhost:8080	10.4 KB	::1:8080	
GET toplogo2_70.png	304 Not Modified	localhost:8080	23.6 KB	::1:8080	
GET null	404 /sigaex/expediente/doc/null	localhost:8080	1 KB	::1:8080	
GET sample3_main_arrow_0.gif	304 Not Modified	localhost:8080	827 B	::1:8080	
GET sample3_sub_arrow_0.gif	304 Not Modified	localhost:8080	827 B	::1:8080	
GET pdf.gif	304 Not Modified	localhost:8080	185 B	::1:8080	
GET base1.gif	304 Not Modified	localhost:8080	3.4 KB	::1:8080	
GET sinlogo.gif	304 Not Modified	localhost:8080	1.1 KB	::1:8080	
GET base2.gif	304 Not Modified	localhost:8080	5.3 KB	::1:8080	
14 requests		192.9 KB (191.9 KB do cache)			

14 Requests, entre eles: Gifs, Pngs, JS , CSS são carregados novamente.

10.2.7 - Console

A console apresenta os erros de HTML e JS. Pode-se também entrar com comandos na console, ou seja, comandos JS. Pode-se carregar uma aplicação JS a partir da console.

Exemplo: Entrar com o comando: `document.getElementById("frm")`. O comando foi entrado na janela da direita, depois executar, e o resultado com o elemento HTML cujo id é "frm" aparece na janela da esquerda.



É importante saber como o formulário de entrevista foi projetado e codificado. A identificação dos IDs, NAMEs e CLASSES (CSS) dos elementos HTML são úteis para observação do comportamento, identificação de bugs e implantação de melhorias.



11.1 - Estrutura geral do formulário do SIGA-DOC

Tabela1: 4 coluna, 3 linhas

Formulário			
Tabela2: 4 colunas, 1 linha			
Tabela3: 4 colunas. 14 linhas (2 linhas são exibidas inicialmente com a opção none)			

Documento			colspan="3">
Origem			
Subscritor			colspan="3">
Função / Lotação			colspan="3">
Destinatário			colspan="3">
Tipo			colspan="3">
Modelo			colspan="3">
Preenchimento			colspan="3">
Classificação			colspan="3">
Descrição			colspan="3">
Entrevista			colspan="3">
	Botões		

Tabela4: 4 colunas, 1 linha

colspan="4" colspan="4">

BODY
DIV
DIV
IMG
DIV
P

Justica Federal - Seção Judiciária do Rio de Janeiro

DIV
IMG

Sistema Integrado de Gestão Administrativa

Documento:	NOVO de: 13/06/12
Origem:	Interno Produzido
Data:	<input type="text" value="13/06/2012"/>
Acesso:	Limitado ao órgão (padrão)
Subscritor:	<input type="text" value="RJ13284"/> RUBEN EDWARD ROSE JUNIOR Substituto
Funcão:Lotação:Localidade:	<input type="text"/> (Opcionalmente informe a função e a lotação na forma: Funcão:Lotação:Localidade)
Destinatário:	Órgão Integrado STI Subsecretaria de Tecnologia da Informação e de Comunicações
Tipo:	Anexo
Modelo:	RubenTesteBrasao
Preenchimento Automático:	[Em branco] <input type="button" value="Alterar"/> <input type="button" value="Remover"/> <input type="button" value="Adicionar"/>
Classificação:	00.01.01.02 ORGANIZAÇÃO E FUNCIONAMENTO: ADMINISTRAÇÃO JUDICIÁRIA: ORGANIZAÇÃO ADMINISTRATIVA: Documentos operacionais referentes à modernização administrativa
Descrição:	<input type="text"/> (preencher o campo acima com palavras-chave, sempre usando substantivos, gênero masculino e singular)
Entrevista:	Nome: <input type="text" value="Ruben Rose"/>
	<input type="button" value="Ok"/> <input type="button" value="Visualizar o modelo preenchido"/>



RUBEN EDWARD ROSE JUNIOR - Seção de Sistemas Especializados

As duas linhas ocultas, que entram com "display = none". Documento Pai e Titular;

The screenshot shows a user interface for document management. At the top, it says 'NOVO de: 02/07/12'. Below that, there are fields for 'Documento' (Document), 'Origem' (Origin), 'Data' (Date), 'Acesso' (Access), and 'Digital' or 'Físico' (Digital or Physical). A dashed arrow points from the 'Documento' field to a 'Documento Pai' (Parent Document) field. Another dashed arrow points from the 'Titular' (Holder) field to another 'Documento Pai' field. There are also fields for 'Subscritor' (Subscriber) and 'Substituto' (Substitute).

11.1.1 - Análise das tabelas

```
<table width="100%" height="100%" cellspacing="0" cellpadding="0" border="0">
  <table width="100%" border="0">
    <form method="POST" action="http://localhost:8080/sigaex/expediente/doc/editar.action"
      onsubmit="customOnsubmit_frm(); return true;" name="frm" id="frm" target="">
      <table width="100%" class="form">
        <table width="100%" cellspacing="0" cellpadding="0" border="0" name="rodapeSuspenso">
          </tbody>
        </table>
      </table>
    </form>
  </table>
</table>
```

Line 65: <table width="100%" height="100%" cellspacing="0" cellpadding="0" border="0">
Line 69: <table width="100%" border="0">
Line 72: <form method="POST" action="http://localhost:8080/sigaex/expediente/doc/editar.action"
 onsubmit="customOnsubmit_frm(); return true;" name="frm" id="frm" target="">
Line 84: <table width="100%" class="form">
Line 352: </tbody></table>
Line 353: </form>
Line 356: </tbody></table>
Line 366: <table width="100%" cellspacing="0" cellpadding="0" border="0" name="rodapeSuspenso">
Line 375: </table>
Line 379: </table>

DEFINIÇÃO DAS LINHAS:

As linhas são referentes ao código HTML listado em 11.5.

Line 66	É a definição da linha da tabela <table width="100%" height="100%" cellspacing="0" cellpadding="0" border="0">	<tbody><tr>
Line 70	É a definição da linha da tabela <table width="100%" border="0">	<tbody><tr>
Line 85	Documento:	<tbody><tr class="header">
Line 91	Origem:	<tr>
Line 117	Documento Pai:	<tr style="display:none;">
Line 132	Subscritor:	<tr>
Line 152	Titular:	<tr style="display:none" id="tr_titular">
Line 168	Função, Lotação, Localidade:	<tr>
Line 175	Destinatário:	<tr>
Line 202	Tipo:	<tr>
Line 276	Modelo:	<tr>
Line 300	Preenchimento Automático:	<tr>
Line 309	Classificação:	<tr>
Line 328	Descrição:	<tr>
Line 336	Entrevista:	<tr>
Line 345	Botões:	<tr>
Line 361	src="/sigaex/imagens/base1.gif">	<tr>
Line 364	background="/sigaex/imagens/base2.gif"	<tr>
Line 367	Nome / Setor e src="/sigaex/imagens/sinlogo.gif"	<tbody><tr>

Observação: background="/sigaex/imagens/base2.gif" (imagem de fundo) (linha 364) é a mesma linha do Nome / Setor e src="/sigaex/imagens/sinlogo.gif" (linha 367)

No total, na tela da entrevista, temos 19 linhas:

Tabela1: 4 coluna, 3 linhas

Tabela2: 4 colunas, 1 linha

Tabela3: 4 colunas, 14 linhas (2 linhas não são exibidas inicialmente, pois possuem a opção display:none)

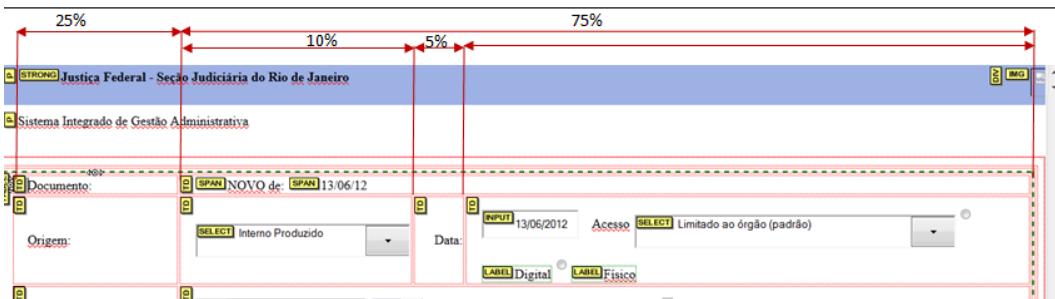
Tabela4: 4 colunas, 1 linha

DEFINIÇÃO DAS COLUNAS colspan=3 e colspan=4:

Tamanho das colunas:

As colunas possuem os seguintes percentuais: 25% 10% 5% 60%
Como a maioria faz colspan=3, teremos 25% 75%

Observação: não está em escala, porém os % estão corretos



COLSPAN

Line 65: <table width="100%" height="100%" cellspacing="0" cellpadding="0" border="0">

Line 67: <td valign="top" style="padding-left: 7; padding-top: 7; padding-right: 7; padding-bottom: 7; colspan="4">

Line 69: <table width="100%" border="0">

Line 84: <table width="100%" class="form">

Documento	Line 87: <td colspan="3">NOVO		
Origem	Line 93: <td width="10%"><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idTpDoc" name="idTpDoc">	Line 100: <td width="5%" align="right">Data:</td>	
Subscritor	Line 119: <td colspan="3">		
Titular	Line 136: <td colspan="3">		
Função, Lotação, Loc.	Line 155: <td colspan="3">		
	Line 170: <td colspan="3"><input type="hidden" value="nmFuncaoSubscritor" name="campos"> <input type="text" id="frm_nmFuncaoSubscritor" value="" maxlength="128" size="50"		

	name="nmFuncaoSubscritor">
Destinatário	Line 178: <td colspan="3" ><select onchange="javascript:sbmt();" id="frm_tipoDestinatario" name="tipoDestinatario">
Tipo	line 204: <td colspan="3" ><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idFormaDoc" name="idFormaDoc">
Modelo	Line 278: <td colspan="3" >
Preenchimento	Line 303: <td colspan="3" ><select onchange="javascript:carregaPreench()" id="frm_preenchimento" name="preenchimento">
Classificação	Line 312: <td colspan="3" >
Descrição	Line 331: <td colspan="3" ><textarea id="descrDocumento" rows="3" cols="80" name="descrDocumento"></textarea>
Entrevista	Line 338: <td colspan="3" >
Botões	Line 347: <td colspan="3" ><input type="button" value="Ok" name="gravar" onclick="javascript:gravarDoc()"

Line 362: <td height="27" **colspan="4"**></td>

Line 365: <td height="22" background="/sigaex/imagens/base2.gif" **colspan="4"**>

Line 366: <**table** width="100%" cellspacing="0" cellpadding="0" border="0" name="rodapeSuspenso">

↓
Observação:

O bloco entrevista, dependendo das macros utilizadas, pode conter mais tabelas, uma para cada campo definido. Se o campo for definido dentro do bloco @grupo, então uma tabela será gerada ([ver campos da entrevista implementados como tabela](#)), caso contrário não ([ver campos da entrevista implementados sem tabela](#)). O programador também pode definir uma tabela na mão na entrevista.

11.2 – IDs por ordem de aparição e elementos HTML

Linha	Elemento HTML	ID
41	<div style="position: absolute; top: 0px; right: 0px; background-color: red; font-weight: bold; padding: 4px; color: white; display: none" id="carregando">Carregando...</div>	id="carregando"
42	<div style="position: absolute; font-weight: bold; padding: 4px; color: white; visibility: hidden" id="quadroAviso">-</div>	id="quadroAviso"
44	<div id="mensagens-remotas"></div>	id="mensagens-remotas"
72	<form method="POST" action="http://localhost:8080/sigaex/expediente/doc/editar.action" onsubmit="customOnsubmit frm(); return true;" name="frm" id="frm" target="">	id="frm"
76	<input type="hidden" name="alterouModelo" id="alterouModelo">	id="alterouModelo"
77	<input type="hidden" id="frm_postback" value="1" name="postback">	id="frm_postback"
78	<input type="hidden" id="sigla" value="" name="sigla">	id="sigla"
79	<input type="hidden" id="frm_nomePreenchimento" value="" name="nomePreenchimento">	id="frm_nomePreenchimento"
81	<input type="hidden" id="frm_despachando" value="false" name="despachando">	id="frm_despachando"
83	<input type="hidden" id="frm_criandoAnexo" value="false" name="criandoAnexo">	id="frm_criandoAnexo"
87	<td colspan="3">NOVO	id="codigoDoc"
87	13/06/12</td>	id="dataDoc"
93	<td width="10%"><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idTpDoc" name="idTpDoc">	id="frm_idTpDoc"
102	<td><input type="text" onblur="javascript:verifica_data(this, true); " id="frm_dtDocString" value="13/06/2012" size="10" name="dtDocString">	id="frm_dtDocString"
103	<input type="hidden" value="nivelAcesso" name="campos">Acesso <select id="frm_nivelAcesso" name="nivelAcesso">	id="frm_nivelAcesso"
112	<input type="radio" value="1" id="eletronicoCheck1" name="eletronico"><label for="eletronicoCheck1">Digital</label>	id="eletronicoCheck1"
113	<input type="radio" value="2" id="eletronicoCheck2" name="eletronico"><label for="eletronicoCheck2">Físico</label>	id="eletronicoCheck2"
121	<input type="hidden" id="frm_mobilPaiSel_id" value="" name="mobilPaiSel.id">	id="frm_mobilPaiSel_id"
122	<input type="hidden" id="frm_mobilPaiSel_descricao" value="" name="mobilPaiSel.descricao">	id="frm_mobilPaiSel_descricao"
123	<input type="hidden" id="frm_mobilPaiSel_buscar" value="" name="mobilPaiSel.buscar">	id="frm_mobilPaiSel_buscar"
124	<input type="hidden" id="frm_reqmobilPaiSel" value="" name="reqmobilPaiSel">	id="frm_reqmobilPaiSel"
125	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"

126	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_mobilPai();" id="frm_mobilPaiSel_sigla" value="" size="25" name="mobilPaiSel.sigla">	id="frm_mobilPaiSel_sigla"
127	<input type="button" theme="simple" onclick="javascript: popupup_mobilPai('');" value="..." id="mobilPaiSelButton">	id="mobilPaiSelButton"
128		id="mobilPaiSelSpan"
138	<input type="hidden" id="frm_subscritorSel_id" value="10199" name="subscritorSel.id">	id="frm_subscritorSel_id"
139	<input type="hidden" id="frm_subscritorSel_descricao" value="RUBEN EDWARD ROSE JUNIOR" name="subscritorSel.descricao">	id="frm_subscritorSel_descricao"
140	<input type="hidden" id="frm_subscritorSel_buscar" value="" name="subscritorSel.buscar">	id="frm_subscritorSel_buscar"
141	<input type="hidden" id="frm_reqsubscritorSel" value="" name="reqsubscritorSel">	id="frm_reqsubscritorSel"
142	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"
143	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_subscritor();" id="frm_subscritorSel_sigla" value="RJ13284" size="25" name="subscritorSel.sigla">	id="frm_subscritorSel_sigla"
144	<input type="button" theme="simple" onclick="javascript: popupup_subscritor('');" value="..." id="subscritorSelButton">	id="subscritorSelButton"
145		id="subscritorSelSpan"
149	<input type="checkbox" onclick="javascript:displayTitular(this);;" id="frm_substituicao" value="true" name="substituicao">	id="frm_substituicao"
152	<tr style="display: none" id="tr_titular">	id="tr_titular"
157	<input type="hidden" id="frm_titularSel_id" value="" name="titularSel.id">	id="frm_titularSel_id"
158	<input type="hidden" id="frm_titularSel_descricao" value="" name="titularSel.descricao">	id="frm_titularSel_descricao"
159	<input type="hidden" id="frm_titularSel_buscar" value="" name="titularSel.buscar">	id="frm_titularSel_buscar"
160	<input type="hidden" id="frm_reqtitularSel" value="" name="reqtitularSel">	id="frm_reqtitularSel"
161	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"
162	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_titular();" id="frm_titularSel_sigla" value="" size="25" name="titularSel.sigla">	id="frm_titularSel_sigla"
163	<input type="button" theme="simple" onclick="javascript: popupup_titular('');" value="..." id="titularSelButton">	id="titularSelButton"
164		id="titularSelSpan"
170	<td colspan="3"><input type="hidden" value="nmFuncaoSubscritor" name="campos"><input type="text" id="frm_nmFuncaoSubscritor" value="" maxlength="128" size="50" name="nmFuncaoSubscritor">	id="frm_nmFuncaoSubscritor"
178	<td colspan="3"><select onchange="javascript:sbmt();" id="frm_tipoDestinatario" name="tipoDestinatario">	id="frm_tipoDestinatario"

185		id="destinatario"
188	<input type="hidden" id="frm_lotacaoDestinatarioSel_id" value="1005" name="lotacaoDestinatarioSel.id">	id="frm_lotacaoDestinatarioSel_id"
189	<input type="hidden" id="frm_lotacaoDestinatarioSel_descricao" value="Subsecretaria de Tecnologia da Informação e de Comunicações" name="lotacaoDestinatarioSel.descricao">	id="frm_lotacaoDestinatarioSel_descricao"
190	<input type="hidden" id="frm_lotacaoDestinatarioSel_buscar" value="" name="lotacaoDestinatarioSel.buscar">	id="frm_lotacaoDestinatarioSel_buscar"
191	<input type="hidden" id="frm_reqlotacaoDestinatarioSel" value="" name="reqlotacaoDestinatarioSel">	id="frm_reqlotacaoDestinatarioSel"
192	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"
193	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_lotacaoDestinatario();" id="frm_lotacaoDestinatarioSel_sigla" value="STI" size="25" name="lotacaoDestinatarioSel.sigla">	id="frm_lotacaoDestinatarioSel_sigla"
194	<input type="button" theme="simple" onclick="javascript: popitup_lotacaoDestinatario('');" value="..." id="lotacaoDestinatarioSelButton">	id="lotacaoDestinariosSelButton"
195		id="lotacaoDestinatariosSelSpan"
204	<td colspan="3"><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idFormaDoc" name="idFormaDoc">	id="frm_idFormaDoc"
279	<div depende=";forma;" id="modelo">	id="modelo"
280	<select onchange="document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idMod" name="idMod">	id="frm_idMod"
303	<td colspan="3"><select onchange="javascript:carregaPreench()" id="frm_preenchimento" name="preenchimento">	id="frm_preenchimento"
313		id="classificacao"
315	<input type="hidden" id="frm_classificacaoSel_id" value="1769" name="classificacaoSel.id">	id="frm_classificacaoSel_id"
316	<input type="hidden" id="frm_classificacaoSel_descricao" value="ORGANIZA ãO E FUNCIONAMENTO: ADMINISTRA ãO JUDICIÁRIA: ORGANIZA ãO ADMINISTRATIVA: Documentos operacionais referentes à modernização administrativa " name="classificacaoSel.descricao">	id="frm_classificacaoSel_descricao"
317	<input type="hidden" id="frm_classificacaoSel_buscar" value="" name="classificacaoSel.buscar">	id="frm_classificacaoSel_buscar"
318	<input type="hidden" id="frm_reqclassificacaoSel" value="" name="reqclassificacaoSel">	id="frm_reqclassificacaoSel"
319	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"
320	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_classificacao();" id="frm_classificacaoSel_sigla" value="00.01.01.02" size="25" name="classificacaoSel.sigla">	id="frm_classificacaoSel_sigla"

321	<input type="button" theme="simple" onclick="javascript:popitup_classificacao(''); value='...' id="classificacaoSelButton">	id="classificacaoSelButton"
322		id="classificacaoSelSpan"
331	<td colspan="3"><textarea id="descrDocumento" rows="3" cols="80" name="descrDocumento"></textarea>	id="descrDocumento"
339	<!--ajax:spanEntrevista-->	id="spanEntrevista"
368	<td class="base">&nbsp&nbsp	id="spanMenuSuspensao"
380	<div onmouseout="javascript:this.style.visibility='hidden';" onmouseover="javascript:this.style.visibility='visible';" style="position: absolute; border: 1px solid #000000; background: #3B437B; height: auto; z-index: 1; left: 30px; bottom: 21px; visibility: hidden; font-color: #FFFFFF;" id="menuSuspensao" > </div>	id="menuSuspensao"

IDs classificado por ID

Linha	Elemento HTML	ID
76	<input type="hidden" name="alterouModelo" id="alterouModelo">	id="alterouModelo"
125	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"
142	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"
161	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"
192	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"
319	<input type="hidden" id="alterouSel" value="" name="alterouSel">	id="alterouSel"
41	<div style="position: absolute; top: 0px; right: 0px; background-color: red; font-weight: bold; padding: 4px; color: white; display: none" id="carregando">Carregando...</div>	id="carregando"
313		id="classificacao"
321	<input type="button" theme="simple" onclick="javascript:popitup_classificacao(''); value='...' id="classificacaoSelButton">	id="classificacaoSelButton"
322		id="classificacaoSelSpan"
87	<td colspan="3">NOVO	id="codigoDoc"
87	13/06/12</td>	id="dataDoc"

331	<td colspan="3"><textarea id="descrDocumento" rows="3" cols="80" name="descrDocumento"></textarea>	id="descrDocumento"		
185		id="destinatario"		
112	<input type="radio" value="1" id="eletronicoCheck1" name="eletronico"><label for="eletronicoCheck1">Digital</label>	id="eletronicoCheck1"		
113	<input type="radio" value="2" id="eletronicoCheck2" name="eletronico"><label for="eletronicoCheck2">Físico</label>	id="eletronicoCheck2"		
72	<form method="POST" action="http://localhost:8080/sigaex/expediente/doc/editar.action" onsubmit="customOnsubmit_frm(); return true;" name="frm" id="frm" target="">	id="frm"		
317	<input type="hidden" id="frm_classificacaoSel_buscar" value="" name="classificacaoSel.buscar">	id="frm_classificacaoSel_buscar"		
316	<input type="hidden" id="frm_classificacaoSel_descricao" value="ORGANIZA ãO E FUNCIONAMENTO: ADMINISTRA ãO JUDICIÁRIA: ORGANIZA ãO ADMINISTRATIVA: Documentos operacionais referentes à modernização administrativa " name="classificacaoSel.descricao">	id="frm_classificacaoSel_descricao"		
315	<input type="hidden" id="frm_classificacaoSel_id" value="1769" name="classificacaoSel.id">	id="frm_classificacaoSel_id"		
320	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_classificacao();" id="frm_classificacaoSel_sigla" value="00.01.01.02" size="25" name="classificacaoSel.sigla">	id="frm_classificacaoSel_sigla"		
83	<input type="hidden" id="frm_criandoAnexo" value="false" name="criandoAnexo">	id="frm_criandoAnexo"		
81	<input type="hidden" id="frm_despachando" value="false" name="despachando">	id="frm_despachando"		
102	<td><input type="text" onblur="javascript:verifica_data(this, true);;" id="frm_dtDocString" value="13/06/2012" size="10" name="dtDocString">	id="frm_dtDocString"		
204	<td colspan="3"><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idFormaDoc" name="idFormaDoc">	id="frm_idFormaDoc"		
280	<select onchange="document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idMod" name="idMod">	id="frm_idMod"		
93	<td width="10%"><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idTpDoc" name="idTpDoc">	id="frm_idTpDoc"		
190	<input type="hidden" id="frm_lotacaoDestinatarioSel_buscar" value="" name="lotacaoDestinatarioSel.buscar">	id="frm_lotacaoDestinatarioSel_buscar"		
189	<input type="hidden" id="frm_lotacaoDestinatarioSel_descricao"	id="frm_lotacaoDestinatarioSel_descricao"		

	<code>value="Subsecretaria de Tecnologia da Informação e de Comunicações" name="lotacaoDestinatarioSel.descricao"></code>	<code>rioSel_descricao"</code>
188	<code><input type="hidden" id="frm_lotacaoDestinatarioSel_id" value="1005" name="lotacaoDestinatarioSel.id"></code>	<code>id="frm_lotacaoDestinata rioSel_id"</code>
193	<code><input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_lotacaoDestinatario();" id="frm_lotacaoDestinatarioSel_sigla" value="STI" size="25" name="lotacaoDestinatarioSel.sigla"></code>	<code>id="frm_lotacaoDestinata rioSel_sigla"</code>
123	<code><input type="hidden" id="frm_mobilPaiSel_buscar" value="" name="mobilPaiSel.buscar"></code>	<code>id="frm_mobilPaiSel_busc ar"</code>
122	<code><input type="hidden" id="frm_mobilPaiSel_descricao" value="" name="mobilPaiSel.descricao"></code>	<code>id="frm_mobilPaiSel_desc ricao"</code>
121	<code><input type="hidden" id="frm_mobilPaiSel_id" value="" name="mobilPaiSel.id"></code>	<code>id="frm_mobilPaiSel_id"</code>
126	<code><input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_mobilPai();" id="frm_mobilPaiSel_sigla" value="" size="25" name="mobilPaiSel.sigla"></code>	<code>id="frm_mobilPaiSel_sigl a"</code>
103	<code><input type="hidden" value="nivelAcesso" name="campos">Acesso <select id="frm_nivelAcesso" name="nivelAcesso"></code>	<code>id="frm_nivelAcesso"</code>
170	<code><td colspan="3"><input type="hidden" value="nmFuncaoSubscritor" name="campos"> <input type="text" id="frm_nmFuncaoSubscritor" value="" maxlength="128" size="50" name="nmFuncaoSubscritor"></code>	<code>id="frm_nmFuncaoSubscrit or"</code>
79	<code><input type="hidden" id="frm_nomePreenchimento" value="" name="nomePreenchimento"></code>	<code>id="frm_nomePreenchiment o"</code>
77	<code><input type="hidden" id="frm_postback" value="1" name="postback"></code>	<code>id="frm_postback"</code>
303	<code><td colspan="3"><select onchange="javascript:carregaPreench()" id="frm_preenchimento" name="preenchimento"></code>	<code>id="frm_preenchimento"</code>
318	<code><input type="hidden" id="frm_reqclassificacaoSel" value="" name="reqclassificacaoSel"></code>	<code>id="frm_reqclassificacao Sel"</code>
191	<code><input type="hidden" id="frm_reqlotacaoDestinatarioSel" value="" name="reqlotacaoDestinatarioSel"></code>	<code>id="frm_reqlotacaoDestin atarioSel"</code>
124	<code><input type="hidden" id="frm_reqmobilPaiSel" value="" name="reqmobilPaiSel"></code>	<code>id="frm_reqmobilPaiSel"</code>
141	<code><input type="hidden" id="frm_reqsubscritorSel" value="" name="reqsubscritorSel"></code>	<code>id="frm_reqsubscritorSel "</code>
160	<code><input type="hidden" id="frm_reqtitularSel" value="" name="reqtitularSel"></code>	<code>id="frm_reqtitularSel"</code>
140	<code><input type="hidden" id="frm_subscritorSel_buscar" value="" name="subscritorSel.buscar"></code>	<code>id="frm_subscritorSel_bu scar"</code>

139	<input type="hidden" id="frm_subscritorSel_descricao" value="RUBEN EDWARD ROSE JUNIOR" name="subscritorSel.descricao">	id="frm_subscritorSel_descriao"
138	<input type="hidden" id="frm_subscritorSel_id" value="10199" name="subscritorSel.id">	id="frm_subscritorSel_id"
143	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_subscritor();" id="frm_subscritorSel_sigla" value="RJ13284" size="25" name="subscritorSel.sigla">	id="frm_subscritorSel_sigla"
149	<input type="checkbox" onclick="javascript:displayTitular(this);;" id="frm_substituicao" value="true" name="substituicao">	id="frm_substituicao"
178	<td colspan="3"><select onchange="javascript:sbmt();" id="frm_tipoDestinatario" name="tipoDestinatario">	id="frm_tipoDestinatario"
159	<input type="hidden" id="frm_titularSel_buscar" value="" name="titularSel.buscar">	id="frm_titularSel_buscar"
158	<input type="hidden" id="frm_titularSel_descricao" value="" name="titularSel.descricao">	id="frm_titularSel_descricao"
157	<input type="hidden" id="frm_titularSel_id" value="" name="titularSel.id">	id="frm_titularSel_id"
162	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_titular();" id="frm_titularSel_sigla" value="" size="25" name="titularSel.sigla">	id="frm_titularSel_sigla"
194	<input type="button" theme="simple" onclick="javascript: popitup_lotacaoDestinatario(''); value='...' id="lotacaoDestinatarioSelButton">	id="lotacaoDestinatariosSelButton"
195		id="lotacaoDestinatariosSelSpan"
44	<div id="mensagens-remotas"></div>	id="mensagens-remotas"
380	<div onmouseout="javascript:this.style.visibility='hidden';" onmouseover="javascript:this.style.visibility='visible';" style="position: absolute; border: 1px solid #000000; background: #3B437B; height: auto; z-index: 1; left: 30px; bottom: 21px; visibility: hidden; font-color: #FFFFFF;" id="menuSuspensao"> </div>	id="menuSuspensao"
127	<input type="button" theme="simple" onclick="javascript: popitup_mobilPai(''); value='...' id="mobilPaiSelButton">	id="mobilPaiSelButton"
128		id="mobilPaiSelSpan"
279	<div depende=";forma;" id="modelo">	id="modelo"
42	<div style="position: absolute; font-weight: bold; padding: 4px; color: white; visibility: hidden" id="quadroAviso">-</div>	id="quadroAviso"
78	<input type="hidden" id="sigla" value="" name="sigla">	id="sigla"

339	<!--ajax:spanEntrevista-->	id="spanEntrevista"
368	<td class="base">&nbsp&nbsp	id="spanMenuSuspensao"
144	<input type="button" theme="simple" onclick="javascript: popitup_subscritor('');" value="..." id="subscritorSelButton">	id="subscritorSelButton"
145		id="subscritorSelSpan"
163	<input type="button" theme="simple" onclick="javascript: popitup_titular('');" value="..." id="titularSelButton">	id="titularSelButton"
164		id="titularSelSpan"
152	<tr style="display: none" id="tr_titular">	id="tr_titular"
76	<input type="hidden" name="alterouModelo" id="alterouModelo">	id="alterouModelo"

11.3 – CLASSES CSS

27	<p class="cabecalho-title">	class="cabecalho-title"
32	<p class="cabecalho-subtitle">	class="cabecalho-subtitle"
84	<table width="100%" class="form">	class="form"
85	<tbody><tr class="header">	class="header"
368	<td class="base">	class="base"

11.4 - NOMES por ordem de aparição e elementos HTML

Linha	Elemento HTML	NOME
72	<form method="POST" action="http://localhost:8080/sigaex/expediente/doc/editar.action" onsubmit="customOnsubmit_frm(); return true;" name="frm" id="frm" target="">	name="frm"
74	<input type="hidden" value="webwork.token" name="webwork.token.name">	name="webwork.token.name"
75	<input type="hidden" value="3IITNY6F1GDJUY1LFYMH637UR1JXLDX8" name="webwork.token">	name="webwork.token"
76	<input type="hidden" name="alterouModelo" id="alterouModelo">	name="alterouModelo"
77	<input type="hidden" id="frm_postback" value="1" name="postback">	name="postback"
78	<input type="hidden" id="sigla" value="" name="sigla">	name="sigla"
79	<input type="hidden" id="frm_nomePreenchimento" value="" name="nomePreenchimento">	name="nomePreenchimento"
80	<input type="hidden" value="despachando" name="campos">	name="campos"
81	<input type="hidden" id="frm_despachando" value="false" name="despachando">	name="despachando"
82	<input type="hidden" value="criandoAnexo" name="campos">	name="campos"
83	<input type="hidden" id="frm_criandoAnexo" value="false" name="criandoAnexo">	name="criandoAnexo"
90	<input type="hidden" value="idTpDoc" name="campos">	name="campos"
93	<td width="10%><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idTpDoc" name="idTpDoc">	name="idTpDoc"
101	<input type="hidden" value="dtDocString" name="campos">	name="campos"
102	<td><input type="text" onblur="javascript:verifica_data(this, true);" id="frm_dtDocString" value="13/06/2012" size="10" name="dtDocString">	name="dtDocString"
103	 <input type="hidden" value="nivelAcesso" name="campos">Acesso <select id="frm_nivelAcesso" name="nivelAcesso">	name="nivelAcesso"
111	<input type="hidden" value="eletronico" name="campos">	name="campos"
112	<input type="radio" value="1" id="eletronicoCheck1" name="eletronico"><label for="eletronicoCheck1">Digital</label>	name="eletronico"
113	<input type="radio" value="2" id="eletronicoCheck2" name="eletronico"><label for="eletronicoCheck2">Físico</label>	name="eletronico"
116	<input type="hidden" value="" name="desativarDocPai">	name="desativarDocPai"

121	<input type="hidden" id="frm_mobilPaiSel_id" value="" name="mobilPaiSel.id">	name="mobilPaiSel.id"
122	<input type="hidden" id="frm_mobilPaiSel_descricao" value="" name="mobilPaiSel.descricao">	name="mobilPaiSel.descricao"
123	<input type="hidden" id="frm_mobilPaiSel_buscar" value="" name="mobilPaiSel.buscar">	name="mobilPaiSel.buscar"
124	<input type="hidden" id="frm_reqmobilPaiSel" value="" name="reqmobilPaiSel">	name="reqmobilPaiSel"
125	<input type="hidden" id="alterouSel" value="" name="alterouSel">	name="alterouSel"
126	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_mobilPai();" id="frm_mobilPaiSel_sigla" value="" size="25" name="mobilPaiSel.sigla">	name="mobilPaiSel.sigla"
134	<input type="hidden" value="subscritorSel.id" name="campos">	name="campos"
135	<input type="hidden" value="substituicao" name="campos">	name="campos"
138	<input type="hidden" id="frm_subscritorSel_id" value="10199" name="subscritorSel.id">	name="subscritorSel.id"
139	<input type="hidden" id="frm_subscritorSel_descricao" value="RUBEN EDWARD ROSE JUNIOR" name="subscritorSel.descricao">	name="subscritorSel.descricao"
140	<input type="hidden" id="frm_subscritorSel_buscar" value="" name="subscritorSel.buscar">	name="subscritorSel.buscar"
141	<input type="hidden" id="frm_reqsubscritorSel" value="" name="reqsubscritorSel">	name="reqsubscritorSel"
142	<input type="hidden" id="alterouSel" value="" name="alterouSel">	name="alterouSel"
143	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_subscritor();" id="frm_subscritorSel_sigla" value="RJ13284" size="25" name="subscritorSel.sigla">	name="subscritorSel.sigla"
148	&nbsp&nbsp<input type="hidden" value="false" name="_FALSE_.substituicao">	name="_FALSE_.substituicao"
149	<input type="checkbox" onclick="javascript:displayTitular(this); id="frm_substituicao" value="true" name="substituicao">	name="substituicao"
154	<input type="hidden" value="titularSel.id" name="campos">	name="campos"
157	<input type="hidden" id="frm_titularSel_id" value="" name="titularSel.id">	name="titularSel.id"
158	<input type="hidden" id="frm_titularSel_descricao" value="" name="titularSel.descricao">	name="titularSel.descricao"
159	<input type="hidden" id="frm_titularSel_buscar" value="">	name="titularSel.buscar"

	<code>name="titularSel.buscar"></code>	
160	<code><input type="hidden" id="frm_reqtitularSel" value="" name="reqtitularSel"></code>	<code>name="reqtitularSel"</code>
161	<code><input type="hidden" id="alterouSel" value="" name="alterouSel"></code>	<code>name="alterouSel"</code>
162	<code><input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_titular();" id="frm_titularSel_sigla" value="" size="25" name="titularSel.sigla"></code>	<code>name="titularSel.sigla"</code>
170	<code><td colspan="3"><input type="hidden" value="nmFuncaoSubscritor" name="campos" > <input type="text" id="frm_nmFuncaoSubscritor" value="" maxlength="128" size="50" name="nmFuncaoSubscritor"></code>	<code>name="nmFuncaoSubscritor"</code>
177	<code><input type="hidden" value="tipoDestinatario" name="campos"></code>	<code>name="campos"</code>
178	<code><td colspan="3"><select onchange="javascript:sbmt();" id="frm_tipoDestinatario" name="tipoDestinatario"></code>	<code>name="tipoDestinatario"</code>
186	<code><input type="hidden" value="lotacaoDestinatarioSel.id" name="campos"></code>	<code>name="campos"</code>
188	<code><input type="hidden" id="frm_lotacaoDestinatarioSel_id" value="1005" name="lotacaoDestinatarioSel.id"></code>	<code>name="lotacaoDestinatarioSel.id"</code>
189	<code><input type="hidden" id="frm_lotacaoDestinatarioSel_descricao" value="Subsecretaria de Tecnologia da Informação e de Comunicações" name="lotacaoDestinatarioSel.descricao"></code>	<code>name="lotacaoDestinatarioSel.descricao"</code>
190	<code><input type="hidden" id="frm_lotacaoDestinatarioSel_buscar" value="" name="lotacaoDestinatarioSel.buscar"></code>	<code>name="lotacaoDestinatarioSel.buscar"</code>
191	<code><input type="hidden" id="frm_reqlotacaoDestinatarioSel" value="" name="reqlotacaoDestinatarioSel"></code>	<code>name="reqlotacaoDestinatarioSel"</code>
192	<code><input type="hidden" id="alterouSel" value="" name="alterouSel"></code>	<code>name="alterouSel"</code>
193	<code><input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_lotacaoDestinatario();" id="frm_lotacaoDestinatarioSel_sigla" value="STI" size="25" name="lotacaoDestinatarioSel.sigla"></code>	<code>name="lotacaoDestinatarioSel.sigla"</code>
204	<code><td colspan="3"><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idFormaDoc" name="idFormaDoc"></code>	<code>name="idFormaDoc"</code>
280	<code><select onchange="document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idMod" name="idMod"></code>	<code>name="idMod"</code>
302	<code><input type="hidden" value="preenchimento" name="campos"></code>	<code>name="campos"</code>
303	<code><td colspan="3"><select onchange="javascript:carregaPreench()"</code>	<code>name="preenchimento"</code>

	<pre>id="frm_preenchimento" name="preenchimento"></pre>	
307	<pre><input type="button" disabled="disabled" onclick="javascript:alteraPreench()" value="Alterar" name="btnAlterar">&nbsp;<input type="button" disabled="disabled" onclick="javascript:removePreench()" value="Remover" name="btnRemover">&nbsp;<input type="button" onclick="javascript:adicionaPreench()" name="btnAdicionar" value="Adicionar"></td></pre>	<pre>name="btnAdicionar"</pre>
311	<pre><input type="hidden" value="classificacaoSel.id" name="campos"></pre>	<pre>name="campos"</pre>
315	<pre><input type="hidden" id="frm_classificacaoSel_id" value="1769" name="classificacaoSel.id"></pre>	<pre>name="classificacaoSel.id"</pre>
316	<pre><input type="hidden" id="frm_classificacaoSel_descricao" value="ORGANIZA ÃO E FUNCIONAMENTO: ADMINISTRA ÃO JUDICIÁRIA: ORGANIZA ÃO ADMINISTRATIVA: Documentos operacionais referentes à modernização administrativa " name="classificacaoSel.descricao"></pre>	<pre>name="classificacaoSel.descricao"</pre>
317	<pre><input type="hidden" id="frm_classificacaoSel_buscar" value="" name="classificacaoSel.buscar"></pre>	<pre>name="classificacaoSel.buscar"</pre>
318	<pre><input type="hidden" id="frm_reqclassificacaoSel" value="" name="reqclassificacaoSel"></pre>	<pre>name="reqclassificacaoSel"</pre>
319	<pre><input type="hidden" id="alterouSel" value="" name="alterouSel"></pre>	<pre>name="alterouSel"</pre>
320	<pre><input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_classificacao();" id="frm_classificacaoSel_sigla" value="00.01.01.02" size="25" name="classificacaoSel.sigla"></pre>	<pre>name="classificacaoSel.sigla"</pre>
329	<pre><input type="hidden" value="descrDocumento" name="campos"></pre>	<pre>name="campos"</pre>
331	<pre><td colspan="3"><textarea id="descrDocumento" rows="3" cols="80" name="descrDocumento"></textarea></pre>	<pre>name="descrDocumento"</pre>
340	<pre><input type="hidden" value="nome" name="vars"></pre>	<pre>name="vars"</pre>
347	<pre><td colspan="3"><input type="button" value="Ok" name="gravar" onclick="javascript: gravarDoc();"></pre>	<pre>name="gravar"</pre>
348	<pre><input type="button" onclick="javascript: popitup_documento(false);" value="Visualizar o modelo preenchido" name="ver_doc"></pre>	<pre>name="ver_doc"</pre>
349	<pre></pre>	<pre>name="ver_doc_pdf"</pre>
366	<pre><table width="100%" cellspacing="0" cellpadding="0" border="0"</pre>	<pre>name="rodapeSuspenso"</pre>

	<code>name="rodapeSuspenso"></code>	
--	--	--

NOMES classificados por nome

Linha	Elemento HTML	Nome
148	 <input type="hidden" value="false" name="_FALSE_.substituicao">	name="_FALSE_.substituicao"
76	<input type="hidden" name="alterouModelo" id="alterouModelo">	name="alterouModelo"
125	<input type="hidden" id="alterouSel" value="" name="alterouSel">	name="alterouSel"
142	<input type="hidden" id="alterouSel" value="" name="alterouSel">	name="alterouSel"
161	<input type="hidden" id="alterouSel" value="" name="alterouSel">	name="alterouSel"
192	<input type="hidden" id="alterouSel" value="" name="alterouSel">	name="alterouSel"
319	<input type="hidden" id="alterouSel" value="" name="alterouSel">	name="alterouSel"
307	<input type="button" disabled="disabled" onclick="javascript:alteraPreench()" value="Alterar" name="btnAlterar"> <input type="button" disabled="disabled" onclick="javascript:removePreench()" value="Remover" name="btnRemover"> <input type="button" onclick="javascript:adicionaPreench()" name="btnAdicionar" value="Adicionar"></td>	name="btnAdicionar"
80	<input type="hidden" value="despachando" name="campos">	name="campos"
82	<input type="hidden" value="criandoAnexo" name="campos">	name="campos"
90	<input type="hidden" value="idTpDoc" name="campos">	name="campos"
101	<input type="hidden" value="dtDocString" name="campos">	name="campos"
111	<input type="hidden" value="eletronico" name="campos">	name="campos"
134	<input type="hidden" value="subscritorSel.id" name="campos">	name="campos"
135	<input type="hidden" value="substituicao" name="campos">	name="campos"
154	<input type="hidden" value="titularSel.id" name="campos">	name="campos"
177	<input type="hidden" value="tipoDestinatario" name="campos">	name="campos"
186	<input type="hidden" value="lotacaoDestinatarioSel.id" name="campos">	name="campos"
302	<input type="hidden" value="preenchimento" name="campos">	name="campos"
311	<input type="hidden" value="classificacaoSel.id" name="campos">	name="campos"
329	<input type="hidden" value="descrDocumento" name="campos">	name="campos"
317	<input type="hidden" id="frm_classificacaoSel_buscar" value="" name="classificacaoSel.buscar">	name="classificacaoSel.buscar"

316	<input type="hidden" id="frm_classificacaoSel_descricao" value="ORGANIZA ÃO E FUNCIONAMENTO: ADMINISTRA ãO JUDICIÁRIA: ORGANIZA ãO ADMINISTRATIVA: Documentos operacionais referentes à modernização administrativa " name="classificacaoSel.descricao">	name="classificacaoSel.descricao"
315	<input type="hidden" id="frm_classificacaoSel_id" value="1769" name="classificacaoSel.id">	name="classificacaoSel.id"
320	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_classificacao();" id="frm_classificacaoSel_sigla" value="00.01.01.02" size="25" name="classificacaoSel.sigla">	name="classificacaoSel.sigla"
83	<input type="hidden" id="frm_criandoAnexo" value="false" name="criandoAnexo">	name="criandoAnexo"
116	<input type="hidden" value="" name="desativarDocPai">	name="desativarDocPai"
331	<td colspan="3"><textarea id="descrDocumento" rows="3" cols="80" name="descrDocumento"></textarea>	name="descrDocumento"
81	<input type="hidden" id="frm_despachando" value="false" name="despachando">	name="despachando"
102	<td><input type="text" onblur="javascript:verifica_data(this, true);" id="frm_dtDocString" value="13/06/2012" size="10" name="dtDocString">	name="dtDocString"
112	<input type="radio" value="1" id="eletronicoCheck1" name="eletronico"><label for="eletronicoCheck1">Digital</label>	name="eletronico"
113	<input type="radio" value="2" id="eletronicoCheck2" name="eletronico"><label for="eletronicoCheck2">Físico</label>	name="eletronico"
72	<form method="POST" action="http://localhost:8080/sigaex/expediente/doc/editar.action" onsubmit="customOnsubmit_frm(); return true;" name="frm" id="frm" target="">	name="frm"
347	<td colspan="3"><input type="button" value="Ok" name="gravar" onclick="javascript: gravarDoc();">	name="gravar"
204	<td colspan="3"><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idFormaDoc" name="idFormaDoc">	name="idFormaDoc"
280	<select onchange="document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idMod" name="idMod">	name="idMod"
93	<td width="10%">><select	name="idTpDoc"

	<pre>onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idTpDoc" name="idTpDoc"></pre>	
190	<pre><input type="hidden" id="frm_lotacaoDestinatarioSel_buscar" value="" name="lotacaoDestinatarioSel.buscar"></pre>	name="lotacaoDestinatarioSel.buscar"
189	<pre><input type="hidden" id="frm_lotacaoDestinatarioSel_descricao" value="Subsecretaria de Tecnologia da Informação e de Comunicações" name="lotacaoDestinatarioSel.descricao"></pre>	name="lotacaoDestinatarioSel.descricao"
188	<pre><input type="hidden" id="frm_lotacaoDestinatarioSel_id" value="1005" name="lotacaoDestinatarioSel.id"></pre>	name="lotacaoDestinatarioSel.id"
193	<pre><input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_lotacaoDestinatario();" id="frm_lotacaoDestinatarioSel_sigla" value="STI" size="25" name="lotacaoDestinatarioSel.sigla"></pre>	name="lotacaoDestinatarioSel.sigla"
123	<pre><input type="hidden" id="frm_mobilPaiSel_buscar" value="" name="mobilPaiSel.buscar"></pre>	name="mobilPaiSel.buscar"
122	<pre><input type="hidden" id="frm_mobilPaiSel_descricao" value="" name="mobilPaiSel.descricao"></pre>	name="mobilPaiSel.descricao"
121	<pre><input type="hidden" id="frm_mobilPaiSel_id" value="" name="mobilPaiSel.id"></pre>	name="mobilPaiSel.id"
126	<pre><input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_mobilPai();" id="frm_mobilPaiSel_sigla" value="" size="25" name="mobilPaiSel.sigla"></pre>	name="mobilPaiSel.sigla"
103	<pre>&nbsp;&nbsp; <input type="hidden" value="nivelAcesso" name="campos">Acesso <select id="frm_nivelAcesso" name="nivelAcesso"></pre>	name="nivelAcesso"
170	<pre><td colspan="3"><input type="hidden" value="nmFuncaoSubscritor" name="campos"> <input type="text" id="frm_nmFuncaoSubscritor" value="" maxlength="128" size="50" name="nmFuncaoSubscritor"></pre>	name="nmFuncaoSubscritor"
79	<pre><input type="hidden" id="frm_nomePreenchimento" value="" name="nomePreenchimento"></pre>	name="nomePreenchimento"
77	<pre><input type="hidden" id="frm_postback" value="1" name="postback"></pre>	name="postback"
303	<pre><td colspan="3"><select onchange="javascript:carregaPreench()" id="frm_preenchimento" name="preenchimento"></pre>	name="preenchimento"
318	<pre><input type="hidden" id="frm_reqclassificacaoSel" value="" name="reqclassificacaoSel"></pre>	name="reqclassificacaoSel"
191	<pre><input type="hidden" id="frm_reqlotacaoDestinatarioSel" value="" name="reqlotacaoDestinatarioSel"></pre>	name="reqlotacaoDestinatarioSel"

124	<input type="hidden" id="frm_reqmobilPaiSel" value="" name="reqmobilPaiSel">	name="reqmobilPaiSel"
141	<input type="hidden" id="frm_reqsubscritorSel" value="" name="reqsubscritorSel">	name="reqsubscritorSel"
160	<input type="hidden" id="frm_reqtitularSel" value="" name="reqtitularSel">	name="reqtitularSel"
366	<table width="100%" cellspacing="0" cellpadding="0" border="0" name="rodapeSuspensos">	name="rodapeSuspensos"
78	<input type="hidden" id="sigla" value="" name="sigla">	name="sigla"
140	<input type="hidden" id="frm_subscritorSel_buscar" value="" name="subscritorSel.buscar">	name="subscritorSel.buscar"
139	<input type="hidden" id="frm_subscritorSel_descricao" value="RUBEN EDWARD ROSE JUNIOR" name="subscritorSel.descricao">	name="subscritorSel.descricao"
138	<input type="hidden" id="frm_subscritorSel_id" value="10199" name="subscritorSel.id">	name="subscritorSel.id"
143	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_subscritor();" id="frm_subscritorSel_sigla" value="RJ13284" size="25" name="subscritorSel.sigla">	name="subscritorSel.sigla"
149	<input type="checkbox" onclick="javascript:displayTitular(this);;" id="frm_substituicao" value="true" name="substituicao">	name="substituicao"
178	<td colspan="3"><select onchange="javascript:sbmt();;" id="frm_tipoDestinatario" name="tipoDestinatario">	name="tipoDestinatario"
159	<input type="hidden" id="frm_titularSel_buscar" value="" name="titularSel.buscar">	name="titularSel.buscar"
158	<input type="hidden" id="frm_titularSel_descricao" value="" name="titularSel.descricao">	name="titularSel.descricao"
157	<input type="hidden" id="frm_titularSel_id" value="" name="titularSel.id">	name="titularSel.id"
162	<input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_titular();" id="frm_titularSel_sigla" value="" size="25" name="titularSel.sigla">	name="titularSel.sigla"
340	<input type="hidden" value="nome" name="vars">	name="vars"
348	<input type="button" onclick="javascript: popitup_documento(false);;" value="Visualizar o modelo preenchido" name="ver_doc">	name="ver_doc"
349	<img onclick="javascript: popitup_documento(true);;" onmouseover="javascript:this.style.cursor='hand'" name="ver_doc_pdf"	name="ver_doc_pdf"

	<code>src="/sigaex/imagens/pdf.gif" valign="center" style=""></code>	
75	<code><input type="hidden" value="3IITNY6F1GDJUY1LFYMH637UR1JXLDX8" name="webwork.token"></code>	<code>name="webwork.token"</code>
74	<code><input type="hidden" value="webwork.token" name="webwork.token.name"></code>	<code>name="webwork.token.name"</code>

11.5 - CÓDIGO HTML

No [Anexo 1](#) temos o código em formato texto.

```
1 <html>
2   <head>
3     <title>SIGA - Novo Documento</title>
4     <meta content="0" http-equiv="Expires">
5     <meta content="no-cache" http-equiv="Pragma">
6     <meta content="no-cache" http-equiv="Cache-Control">
7     <meta content="text/html; charset=UTF-8" http-equiv="content-type">
8
9
10
11    <link media="screen" title="SIGA Estilos" type="text/css" href="/sigaex/signalibs/siga.css" rel="StyleSheet">
12    <script type="text/javascript" language="JavaScript1.1" src="/sigaex/signalibs/ajax.js"></script>
13    <script charset="utf-8" type="text/javascript" language="JavaScript1.1" src="/sigaex/signalibs/static_javascript.js"></script>
14
15    <link type="text/css" rel="stylesheet" href="/sigaex/signalibs/menu.css">
16
17    <link href="/signalibs/siga.ico" rel="shortcut icon">
18  </head>
19
20  <body marginwidth="0" marginheight="0" style="background: url(null) fixed no-repeat;" onload="" topmargin="0" leftmargin="0">
21
22    <div style="background-color: #9DB1E5; height: 49px; width: 100%;>
23      <div style="float: left">
24        
25      </div>
26      <div style="float: left">
27        <p class="cabecalho-title">
28          <strong>Justiça Federal
29          - Seção Judiciária do Rio de Janeiro
30          - </strong>
31        </p>
32        <p class="cabecalho-subtitle">
33          Sistema Integrado de Gestão Administrativa
34        </p>
35      </div>
36      <div style="float: right">
37        
38      </div>
39    </div>
40
41    <div style="position: absolute; top: 0px; right: 0px; background-color: red; font-weight: bold; padding: 4px; color: white; display: none" id="carregando">Carregando...
42    <div style="position: absolute; font-weight: bold; padding: 4px; color: white; visibility: hidden" id="quadroAviso">--</div>
43  <!-- Mensagens remotas
44  <div id="mensagens-remotas"></div>
45
46  <script type="text/javascript" src="/sigaex/signalibs/mensagensremotas.js"></script>
47  <script type="text/javascript" >
48    exibirMensagensRemotas("arquivos/mensagens/mensagens_remotas.xml"
49    , "mensagens-remotas"
50    , "color: yellow; display: inline-block; font-weight: bolder; font-size:medium; position: relative; width: 100%; text-align: center; background-color: #9DB1E5; border: 1px solid black; border-radius: 5px; padding: 2px; margin: 5px 0; </script>
51  );
52 </script> -->
53 <!-- Fim das mensagens remotas -->
54
55 <!--|**START IMENUS*|imenu0,inline-->
56 <!--[if IE]><style type="text/css">.imcm .imea span(position:absolute).imcm .imclear,.imclear(display:none).imcm(zoom:1;) .imcm li(curosr:hand;) .imcm ul(zoom:1).imcm
57 <!--[if gte IE 7]><style type="text/css">.imcm .imsubc(background-image:url(ie_css_fix));</style><![endif]-->
58
59 <!--|**START IMENUS*|imenu1,inline-->
60 <!--[if IE]><style type="text/css">.imcm .imea span(position:absolute).imcm .imclear,.imclear(display:none).imcm(zoom:1;) .imcm li(curosr:hand;) .imcm ul(zoom:1).imcm
61 <!--[if gte IE 7]><style type="text/css">.imcm .imsubc(background-image:url(ie_css_fix));</style><![endif]-->
62
63 <!-- <body style="padding: 0px 0px 0px 0px; margin: 0px 0px 0px 0px;" -->
64
65    <table width="100%" height="100%" cellspacing="0" cellpadding="0" border="0">
66      <tbody><tr>
67        <td valign="top" style="padding-left: 7; padding-top: 7; padding-right: 7; padding-bottom: 7; colspan="4">
68          <center>
69            <table width="100%" border="0">
70              <tbody><tr>
71                <td>
72                  <form method="POST" action="http://localhost:8080/sigaex/expediente/doc/editar.action" onsubmit="customOnsubmit_frm(); return true;" name="frm" id="frm" target="">>
73
74                    <input type="hidden" value="webwork.token" name="webwork.token.name">
75                    <input type="hidden" value="3LTNY6f10DJUV1LFYMH637URL1XLDX8" name="webwork.token">
76                    <input type="hidden" name="alterouModelo" id="alterouModelo">
77                    <input type="hidden" id="frm_postback" value="1" name="postback">
78                    <input type="hidden" id="sigla" value="" name="sigla">
79                    <input type="hidden" id="frm_nomePreenchimento" value="" name="nomePreenchimento">
80                    <input type="hidden" value="despachando" name="campos">
81                    <input type="hidden" id="frm_despachando" value="false" name="despachando">
82                    <input type="hidden" value="criandoAnexo" name="campos">
83                    <input type="hidden" id="frm_criandoAnexo" value="false" name="criandoAnexo">
84                    <table width="100%" class="form">
85                      <tbody><tr class="header">
86                        <td>Documento:</td>
87                        <td colspan="3"><span id="codigoDoc">NOVO</span>
88                        <td>Data:<span id="dataDoc">13/06/12</span></td>
89                      </tr>
90                      <input type="hidden" value="idTpDoc" name="campos">
91                      <tr>
92                        <td width="10%">Origen:</td>
93                        <td width="10%"><select onchange="javascript:document.getElementById('alterouModelo').value='true';submit()" style="" id="frm_idTpDoc" name="idTpD
94                          <option value="3">Externo</option>
95                          <option selected="" value="1">Interno Produzido</option>
96                          <option value="2">Interno Importado</option>
97                        </select>
98                        <span style="display: none">Interno Produzido</span>
99                        </td>
100                       <td width="5%" align="right">Data:</td>
101                       <input type="hidden" value="dtDocString" name="campos">
102                       <td><input type="text" onblur="javascript:verifica_data(this, true); id="frm_dtDocString" value="13/06/2012" size="10" name="dtDocString">
103                         &nbsp;&nbsp; <input type="hidden" value="nivelAcesso" name="campos">Acesso <select id="frm_nivelAcesso" name="nivelAcesso">
104                           <option value="">Público</option>
105                           <option selected="" value="1">Limitado ao órgão (padrão)</option>
106                           <option value="2">Limitado de pessoa para subsecretaria</option>
107                           <option value="2">Limitado de subsecretaria para pessoa</option>
108                           <option value="3">Limitado entre lotações</option>
109                           <option value="5">Limitado entre pessoas</option>
110                         </select>

```

```

111 <input type="hidden" value="eletronico" name="campos">
112 <input type="radio" value="1" id="eletronicoCheck1" name="eletronico"><label for="eletronicoCheck1">Digital</label>
113 <input type="radio" value="2" id="eletronicoCheck2" name="eletronico"><label for="eletronicoCheck2">Físico</label>
114 </td>
115 </tr>
116 <input type="hidden" value="" name="desativarDocPai">
117 <tr style="display: none;">
118 <td>Documento Pai</td>
119 <td colspan="3">
120 <!-- A lista de par -->
121 <input type="hidden" id="frm_mobilPaiSel_id" value="" name="mobilPaiSel.id">
122 <input type="hidden" id="frm_mobilPaiSel_descricao" value="" name="mobilPaiSel.descricao">
123 <input type="hidden" id="frm_mobilPaiSel_buscar" value="" name="mobilPaiSel.buscar">
124 <input type="hidden" id="frm_recomobilPaiSel" value="" name="recomobilPaiSel">
125 <input type="hidden" id="alterouSel" value="" name="alterouSel">
126 <input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_mobilPai(); id="frm_mobilPaiSel_sigla" value="" size="25" name="mobilPaiSel.sigla">
127 <input type="button" theme="simple" onclick="javascript: popup_mobilPai(''); value='...' id="mobilPaiSelButton">
128 <span id="mobilPaiSelSpan">
129 </span>
130 </td>
131 </tr>
132 <tr>
133 <td>Subscritor:</td>
134 <input type="hidden" value="subscritorSel.id" name="campos">
135 <input type="hidden" value="substituicao" name="campos">
136 <td colspan="3">
137 <!-- A lista de par -->
138 <input type="hidden" id="frm_subscritorSel_id" value="10199" name="subscritorSel.id">
139 <input type="hidden" id="frm_subscritorSel_descricao" value="RUBEN EDWARD ROSE JUNIOR" name="subscritorSel.descricao">
140 <input type="hidden" id="frm_subscritorSel_buscar" value="" name="subscritorSel.buscar">
141 <input type="hidden" id="frm_regsubscritorSel" value="" name="regsubscritorSel">
142 <input type="hidden" id="alterouSel" value="" name="alterouSel">
143 <input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_subscritor(); id="frm_subscritorSel_sigla" value="RJ13284" size="25" name="subscritorSel.sigla">
144 <input type="button" theme="simple" onclick="javascript: popup_subscritor(''); value='...' id="subscritorSelButton">
145 <span id="subscritorSelSpan">
146 RUBEN EDWARD ROSE JUNIOR
147 &ampnbsp&ampnbsp<input type="checkbox" onclick="javascript:displayTitular(this); id="frm_substituicao" value="true" name="substituicao">
148 Substituto</td>
149 <tr>
150 <td>Titular:</td>
151 <input type="hidden" value="titularSel.id" name="campos">
152 <td colspan="3">
153 <!-- A lista de par -->
154 <input type="hidden" id="frm_titularSel_id" value="" name="titularSel.id">
155 <input type="hidden" id="frm_titularSel_descricao" value="" name="titularSel.descricao">
156 <input type="hidden" id="frm_titularSel_buscar" value="" name="titularSel.buscar">
157 <input type="hidden" id="frm_reqtitularSel" value="" name="reqtitularSel">
158 <input type="hidden" id="alterouSel" value="" name="alterouSel">
159 <input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_titular(); id="frm_titularSel_sigla" value="" size="25" name="titularSel.sigla">
160 <input type="button" theme="simple" onclick="javascript: popup_titular(''); value='...' id="titularSelButton">
161 <span id="titularSelSpan">
162 </span>
163 </td>
164 </tr>
165 <tr>
166 <td>Função:Lotação:Localidade:</td>
167 <td colspan="3"><input type="hidden" value="nmFuncaoSubscritor" name="campos"> <input type="text" id="frm_nmFuncaoSubscritor" value="" maxlength="100" name="nmFuncaoSubscritor">
168 (Opcionalmente informe a função e a lotação na forma:
169 <input type="text" value="nmFuncaoSubscritor" name="nmFuncaoSubscritor">
170 <td>Destinatário:</td>
171 <input type="hidden" value="tipoDestinatario" name="campos">
172 <td colspan="3"><select onchange="javascript:sbm(); id="frm_tipoDestinatario" name="tipoDestinatario">
173 <option value="1">Matrícula</option>
174 <option selected="selected" value="2">Órgão Integrado</option>
175 <option value="3">Órgão Externo</option>
176 <option value="4">Campo Livre</option>
177 </select>
178 <!-- sbm('tipoDestinatario') -->
179 <span depende="tipoDestinatario" id="destinatario"><!--ajax:destinatario-->
180 <input type="hidden" value="lotacaoDestinatarioSel.id" name="campos">
181 <!-- A lista de par -->
182 <input type="hidden" id="frm_lotacaoDestinatarioSel_id" value="1005" name="lotacaoDestinatarioSel.id">
183 <input type="hidden" id="frm_lotacaoDestinatarioSel_descricao" value="Subsecretaria de Tecnologia da Informação e de Comunicações" name="lotacaoDestinatarioSel.descricao">
184 <input type="hidden" id="frm_lotacaoDestinatarioSel_buscar" value="" name="lotacaoDestinatarioSel.buscar">
185 <input type="hidden" id="frm_reglotacaoDestinatarioSel" value="" name="reqlotacaoDestinatarioSel">
186 <input type="hidden" id="alterouSel" value="" name="alterouSel">
187 <input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_lotacaoDestinatario(); id="frm_lotacaoDestinatarioSel_sigla" value="STI" size="25" name="lotacaoDestinatarioSel.sigla">
188 <input type="button" theme="simple" onclick="javascript: popup_lotacaoDestinatario(''); value='...' id="lotacaoDestinatarioSelButton">
189 <span id="lotacaoDestinatarioSelSpan">
190 Subsecretaria de Tecnologia da Informação e de Comunicações
191 </span>
192 </td>
193 <!-- idAjax="destinatario" -->
194 <!--/ajax:destinatario-->
195 </tr>
196 <tr>
197 <td>Tipo:</td>
198 <td colspan="3"><select onchange="javascript:document.getElementById('alterouModelo').value='true';sbmt(); style="" id="frm_idFormaDoc" name="idFormaDoc">
199 <option selected="selected" value="60">Anexo</option>
200 <option value="80">Assentamento Funcional</option>
201 <option value="10">Ato</option>
202 <option value="67">Ato da Corregedoria</option>
203 <option value="70">Aviso (DIRPO)</option>
204 <option value="54">Boletim Interno</option>
205 <option value="11">Carta</option>
206 <option value="15">Certidão</option>
207 <option value="61">Certidão de Tempo de Contribuição</option>
208 <option value="87">Comunicado de Inventário</option>
209 <option value="9">Contrato</option>
210 <option value="62">Declaração</option>
211 <option value="82">Despacho</option>
212 <option value="59">Edital (Teor Administrativo)</option>
213 <option value="51">Exposição de Motivos</option>
214 <option value="3">Formulário</option>
215 <option value="4">Informação</option>
216 <option value="21">Memorando</option>
217 <option value="43">Memorando Circular</option>
218 <option value="50">Memorando Circular (SG)</option>
219 <option value="97">Memória de Reunião</option>
220 <option value="1">Ofício</option>
221 <option value="42">Ofício Circular</option>

```

```

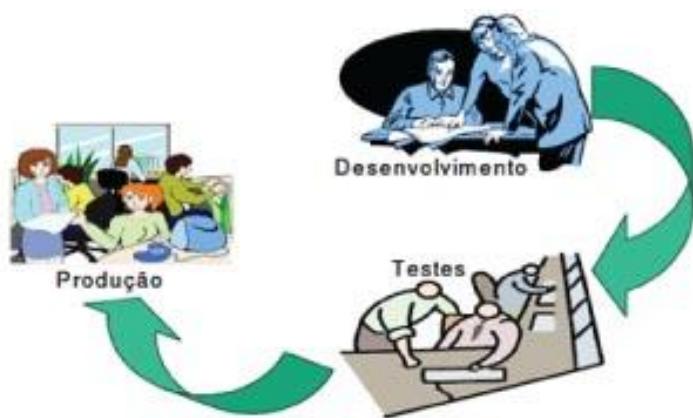
228 <option value="48">Ofício Circular (DIRPO)</option>
229 <option value="7">Ordem de Serviço</option>
230 <option value="49">Ordem de Serviço (DIRPO)</option>
231 <option value="75">Ordem de Serviço da Corregedoria</option>
232 <option value="14">Parecer</option>
233 <option value="63">Pauta</option>
234 <option value="6">Portaria</option>
235 <option value="74">Portaria (DIRPO-GP)</option>
236 <option value="73">Portaria (NPG)</option>
237 <option value="47">Portaria (PGD)</option>
238 <option value="46">Portaria (SG)</option>
239 <option value="41">Portaria (SRH)</option>
240 <option value="68">Portaria da Corregedoria</option>
241 <option value="88">Portarias da EMARF</option>
242 <option value="66">Processo Administrativo Disciplinar</option>
243 <option value="81">Processo de Acompanhamento de Projetos</option>
244 <option value="76">Processo de Comunicação</option>
245 <option value="94">Processo de Corregedoria</option>
246 <option value="96">Processo de Emendas Regimentais</option>
247 <option value="57">Processo de Execução Orçamentária e Financeira</option>
248 <option value="55">Processo de Outros Assuntos Administrativos</option>
249 <option value="95">Processo de Pedido de Provídências</option>
250 <option value="64">Processo de Pessoal</option>
251 <option value="77">Processo de Petição</option>
252 <option value="93">Processo de Procedimento Normativo</option>
253 <option value="65">Processo de Sindicância</option>
254 <option value="79">Processo de Vitaliciamento</option>
255 <option value="58">Processo do Conselho Consultivo</option>
256 <option value="69">Provimento da Corregedoria</option>
257 <option value="98">Relatório</option>
258 <option value="12">>Requerimento</option>
259 <option value="92">>Resolução</option>
260 <option value="13">>Solicitação</option>
261 <option value="78">>Solicitação Eletrônica de Contratação</option>
262 <option value="99">>TRF - CDEFS Decisão</option>
263 <option value="90">>TRF - SCA Nota de Auditoria</option>
264 <option value="86">>TRF-PRES Ato da Presidência</option>
265 <option value="82">>TRF-PRES Edital da Presidência</option>
266 <option value="83">>TRF-PRES Ordem de Serviço da Presidência</option>
267 <option value="85">>TRF-PRES Portaria da Presidência</option>
268 <option value="84">>TRF-PRES Resolução da Presidência</option>
269 <option value="89">>TRF-Proposta e Concessão de Diárias</option>
270 <option value="16">>Termo</option>
271 </select>
272 <!-- sbmt('forma') -->
273 <!-- sbmt('forma') --> <span style="display: none">Anexo</span>
274 </td>
275 </tr>
276 <tr>
277 <td>Modelo:</td>
278 <td colspan="3">
279 <div depende=";forma;" id="modelo"><!--ajax:modelo-->
280 <select onchange="document.getElementById('alterouModelo').value='true';sbmt();" style="" id="frm_idMod" name="idMod">
281 <option value="800">(Fremarker) SOP: Ambiente de Saúde: Encaminhamento de recibo(s)</option>
282 <option value="744">Ambiente de Teste - André Vitorino</option>
283 <option value="907">Anexo</option>
284 <option value="665">Despacho Automático</option>
285 <option value="780">Ruben teste para o manual</option>
286 <option value="742">RubenTeste</option>
287 <option selected="" value="880">RubenTesteBrasao</option>
288 <option value="803">Solicitação Eletrônica de Contratação 01</option>
289 <option value="806">Teste JS/Jquery2</option>
290 <option value="807">Teste Priscila</option>
291 <option value="805">TesteJquery</option>
292 <option value="804">teste TJA&lt;/option>
293 <option value="862">testeruben3 final</option>
294 </select>
295 <!-- sbmt('modelo') -->
296 <!-- ajax:modelo--></div>
297 </td>
298 </tr>
299 <tr>
300 <td>Preenchimento Automático:</td>
301 <td><input type="hidden" value="preenchimento" name="campos">
302 <td colspan="3"><select onchange="javascript:carregaPreench()" id="frm_preenchimento" name="preenchimento">
303 <option selected="" value="0"><span style="display: none">RubenTesteBrasao</span>
304 <!-- sbmt('modelo') -->
305 <!-- ajax:modelo--></div>
306 &nbsp;
307 <td><input type="button" disabled="disabled" onclick="javascript:alteraPreench()" value="Alterar" name="btnAlterar">&nbsp;<input type="button" d
308 </td>
309 <tr>
310 <td>Classificação:</td>
311 <td><input type="hidden" value="classificacaoSel.id" name="campos">
312 <td colspan="3">
313 <span depende=";forma;modelo;" id="classificacao"><!--ajax:classificacao-->
314 <!-- A lista de par -->
315 <input type="hidden" id="frm_classificacaoSel_id" value="1769" name="classificacaoSel.id">
316 <input type="hidden" id="frm_classificacaoSel_descricao" value="ORGANIZAÇÃO E FUNCIONAMENTO: ADMINISTRAÇÃO JUDICIÁRIA: ORGANIZAÇÃO ADMINISTRATIVA: Documentos operaciona
317 <input type="hidden" id="frm_classificacaoSel_buscar" value="" name="classificacaoSel.buscar">
318 <input type="hidden" id="frm_reclassificacaoSel" value="" name="reclassificacaoSel">
319 <input type="hidden" id="alteronSel" value="" name="alterouSel">
320 <input type="text" onkeypress="return handleEnter(this, event)" onblur="javascript: ajax_classificacao(); id="frm_classificacaoSel_sigla" value="00.01.01.02" size="25" :
321 <input type="button" theme="simple" onclick="javascript: popup_ classificacao(''); value='...' id="classificacaoSelButton">
322 <span id="classificacaoSelSpan">
323 ORGANIZAÇÃO E FUNCIONAMENTO: ADMINISTRAÇÃO JUDICIÁRIA: ORGANIZAÇÃO ADMINISTRATIVA: Documentos operacionais referentes à modernização administrativa
324 </span>
325 <!-- idAjax="classificacao" -->
326 <!--ajax:classificacao--></span></td>
327 </tr>
328 <tr>
329 <td><input type="hidden" value="descrDocumento" name="campos">
330 <td>Descrição:</td>
331 <td colspan="3"><textarea id="descrDocumento" rows="3" cols="80" name="descrDocumento"></textarea>
332 <br>
333 <span><b>(preencher o campo acima com palavras-chave, sempre usando substantivos, gênero masculino e singular)</b></span></td>
334 </tr>
335 <tr>
336 <td>Entrevista:</td>
337 <td colspan="3">
338 <span depende=";tipoDestinatario:destinatario:forma:modelo;" id="spanEntrevista"><!--ajax:spanEntrevista-->
339 <input type="hidden" value="name" name="vars">
340 <span style=";">>Name:</span>
341 <input type="text" maxlength="20" size="20" value="Ruben Rose" name="nome">
342 <!-- ajax:spanEntrevista--></span></td>
343 </tr>

```

```
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
```

		<tr>	
345		<td></td>	
346		<td colspan="3"><input type="button" value="Ok" name="gravar" onclick="javascript: gravarDoc();">	
347		<input type="button" onclick="javascript: popup_documento(false);" value="Visualizar o modelo preenchido" name="ver_doc">	
348		<img width="100%" height="27" alt="" onmouseout="javascript:document.getElementById('menuSuspenso').style.visibility='hidden'"	
363		</tr>	
364		<tr>	
365		<td height="22" background="/sigaex/imagens/base2.gif" colspan="4">	
366		<table width="100%" cellspacing="0" cellpadding="0" border="0" name="rodapeSuspenso">	
367		<tbody><tr>	
368		<td class="base"><span style="cursor: pointer" onmouseover="javascript:document.getElementById('menuSuspenso').style.visibility='visible';" id="s	
369		RUBEN EDWARD ROSE JUNIOR	
370		- Seção de Sistemas Especializados	
371		</td>	
372		<td align="right">&nbsp&nbsp</td>	
373		</tr>	
374		</tbody>	
375		</table>	
376		</td>	
377		</tr>	
378		</tbody>	
379		</table>	
380		<div onmouseout="javascript:this.style.visibility='hidden';" onmouseover="javascript:this.style.visibility='visible';" style="position: absolute; border: 1px sol	
381			
382		</div>	

Qual é o ambiente de desenvolvimento disponibilizado para o FM? Qual é o editor? Temos CVS? Perdi o meu código, como recuperá-lo?



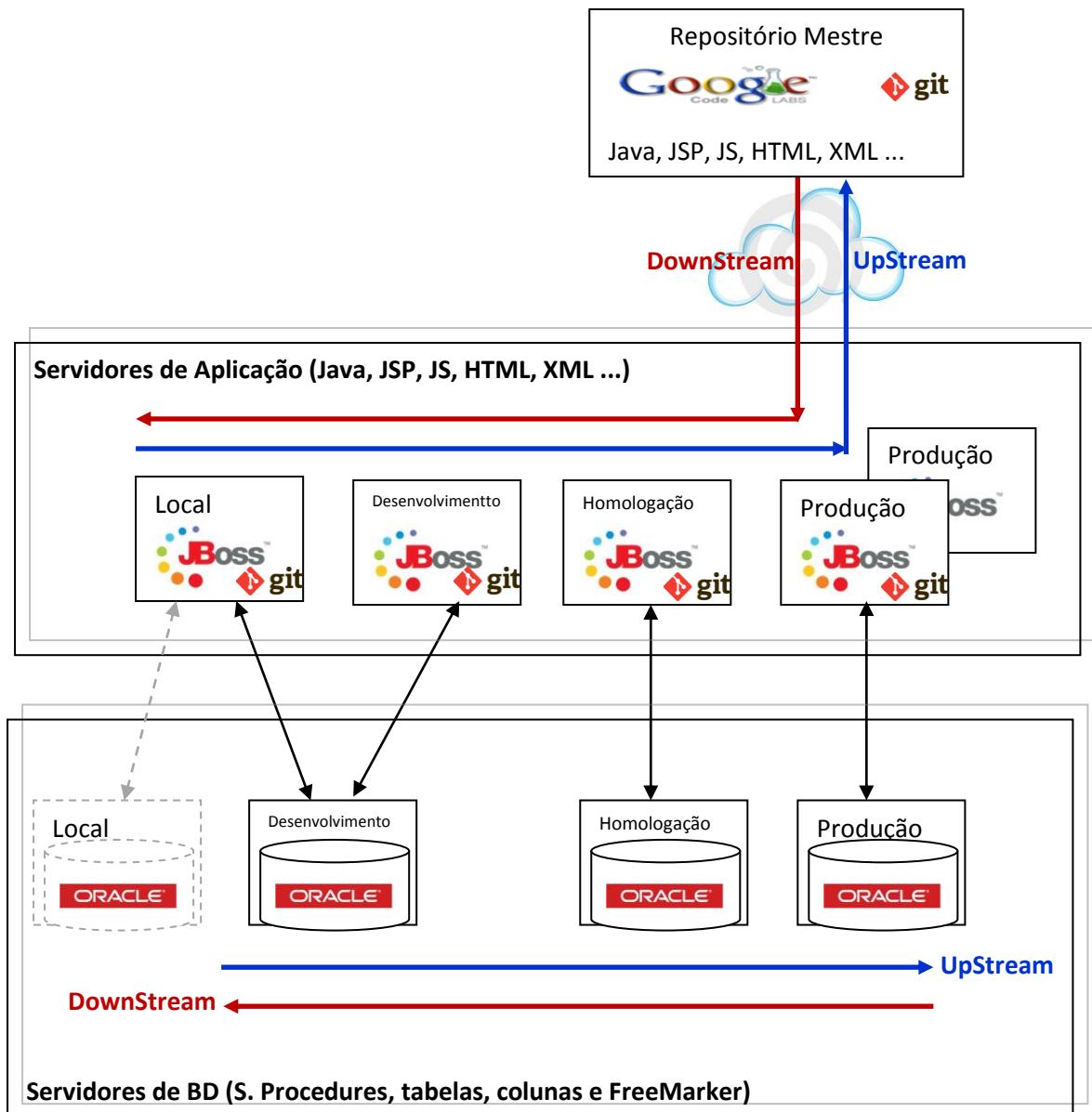
12.1 – O Ambiente de Desenvolvimento

Ambiente clássico:

- Local (Work Directory)
 - Desenvolvimento
 - Homologação e/ou Treinamento
 - Produção
- } Específica para cada programador.
- } Compartilhada por todos programadores.

O ambiente local é composto por uma IDE que possui, além de outros, de um editor de código que verifica a sintaxe em tempo de codificação (syntax highlighting), fornece dicas para comandos, fornece uma ferramenta para depurar (debug) o código e está integrado a uma ferramenta de controle de versão de código (CVS).

Visão geral para atualização dos servidores de aplicação e BD



AS IDEs atendem as atualizações nos servidores de aplicação, porém nos servidores de BD, temos que realizar tudo manualmente.

Existem duas correntes de atualizações:

- **Upstream:** deveria ser sempre a única e principal abordagem. Ela começa na máquina local do desenvolvedor e vai subindo (escalando) até chegar a produção e ao Repositório Mestre. Esta abordagem é sempre utilizada no desenvolvimento de produtos novos e melhorias, porém nem sempre respeitada nas correções, principalmente nas urgentes, visto que muitos programadores preferem atacar o problema direto na produção e, depois(?????????), realizar o downstream.
- **Downstream:** é a mão oposta do upstream, como explicado anteriormente. É utilizada no caso de correções urgentes aplicadas diretamente no ambiente de produção (que devem ser realizadas nos ambientes inferiores) e no caso de se perder o controle dos códigos do ambiente de homologação e desenvolvimento, sendo que neste último caso faz-se um downstream forçado da produção para estes ambientes, e às vezes, criando alguns problemas se não tivermos um ambiente local (work directory).

Para o código do SIGA-DOC utilizamos a IDE JBoss Developer Studio e o GIT como ferramenta de controle de versão, sendo o repositório do Google Code o repositório Master. Porém, só abordaremos a atualização do código FM neste manual e o FM reside no BD.

12.2 – O Ambiente de Desenvolvimento do FM

Para o FM NÃO temos uma IDE e nem uma ferramenta de controle de versão!!! Por quê?

É o problema do tradeoff, Ônus e Bônus, da nossa escolha.

Os dois modos de trabalho do FM. A nossa escolha recaiu sobre o BD.

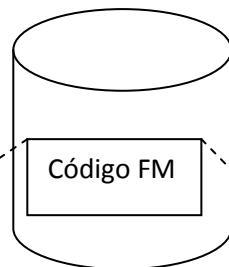
FM residindo no File System (FS)

Neste caso, os arquivos FM residem no diretório do projeto, como o JSP, com a extensão .ftl

```
└─ sigaex [git_google master]
  ├─ src
  ├─ JBoss 4.2 Runtime [JBoss-4.2.2.GA Runtime]
  ├─ JRE System Library [jdk1.6.0]
  ├─ Java EE 5 libraries (JBoss Tools)
  ├─ Referenced Libraries
  ├─ Web App Libraries
  └─ WebContent
    ├─ arquivos
    ├─ imagens
    ├─ META-INF
    └─ paginas
      ├─ configuracao
      ├─ despacho
      ├─ entradas
      └─ expediente
        ├─ formas
        ├─ modelos
        ├─ relatorios
        │  ├─ agenda_publicacao.jsp
        │  ├─ anexa.jsp
        │  ├─ testa_html2pdf.jsp
        │  ├─ teste.ftl
        │  └─ teste2.ftl
      └─ transfere_lote.jsp
```

FM residindo no Banco de Dados (BD)

Neste caso, os arquivos FM residem no BD.



Nome da aplicação	Código
Teste2	<pre>... [@selecao var="catFuncionario" titulo="Categoria do Funcionário" reler=true opcoes="Servidor; Terceiro"] ...</pre>

TRADEOFFS

Local onde reside o FM	Vantagens	Desvantagens
No FS		Deployment, que segue o

	Pode ser assistida por uma IDE, CVS e debug.	mesmo modus operandi de um JSP, por exemplo. Lento para realizar pequenas correções e para testar o código.
NO BD	Deployment ZERO. Uma correção é colocada no ar instantaneamente. Isto também facilita o desenvolvimento, visto que podemos testar em tempo real.	Sem IDE, sem CVS e sem Debug. Os controles são todos manuais. Atualização de código sujeito a erros. Dificuldade em depurar código.

É importante deixar claro que em um futuro próximo poderemos ter IDE, CVS e Debug para FM residindo em BD. Enquanto este tempo não chegar, podemos utilizar o Integrador V1.0 que veremos na seção 12.6.

12.2.1 - Ambientes da SJRJ

Ambientes	Endereço da aplicação SIGA no Servidor de Aplicação	Qual BD é apontado automaticamente pelo servidor de aplicação
Desenvolvimento no ambiente local * A princípio, o desenvolvedor não utilizará o ambiente local para o FM	http://localhost:8080/siga	Teste (ou Desenvolvimento) * Caso o desenvolvedor tenha o BD instalado na sua máquina ele pode apontar o servidor de aplicação para o BD Local
Desenvolvimento no ambiente de testes	http://versailles:8080/siga	Teste (ou Desenvolvimento)
Homologação	http://sigat.jfrj.jus.br	Homologação (ou Treinamento)
Produção	http://siga.jfrj.jus.br	Produção

12.2.2 - Procedimentos para criar e manter uma aplicação FM

Criando uma nova aplicação FM:

- Desenvolver no ambiente local, se for o caso;
- Desenvolver (ou migrar o código do ambiente local) no ambiente de Desenvolvimento (também conhecido como ambiente de testes);
- Migrar o código do ambiente de Desenvolvimento para o ambiente de Homologação (também conhecido como ambiente de Treinamento). Solicitar que o usuário homologue a aplicação. Se for ok, passar para a atividade seguinte, caso contrário, retornar a atividade anterior;
- Migrar o código do ambiente de Homologação para o ambiente de Produção.

Realizando manutenção em uma aplicação FM existente:

- Obter o código mais atual da aplicação que sofrerá manutenção. Teoricamente, os ambientes de Desenvolvimento, Homologação e Produção DEVEM possuir o mesmo código, porém não há nada que garanta isto. Nesta situação recomenda-se obter o código do ambiente de Produção;
- Se a manutenção é urgente deve-se seguir o downstream como mencionado em 12.1, ou seja, atualizar o código no ambiente de Produção e replicá-lo para os ambientes de Homologação e Desenvolvimento;
- Se a manutenção não for urgente, seguir o upstream (sempre preferido)
 - Atualizar o código no ambiente de Desenvolvimento;

- Migrar o código do ambiente de Desenvolvimento para o ambiente de Homologação. Solicitar que o usuário homologue a aplicação. Se for ok, passar para a atividade seguinte, caso contrário, retornar a atividade anterior;
- Migrar o código do ambiente de Homologação para o ambiente de Produção.

12.3 – Criando uma aplicação FM

Como o FM não deixa de ser um documento, utilizamos o próprio ambiente do SIGA-DOC para criá-lo e armazená-lo.

12.3.1 – Criando uma aplicação FM

No SIGA-DOC, acessar **Ferramentas / Cadastro de Modelos / Novo**.

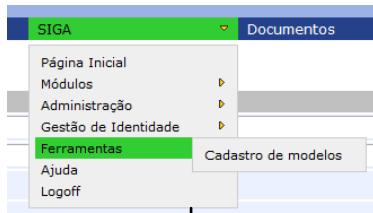
Edição de Modelo	
Modelo	
Nome:	Teste para manual
Descrição:	Teste para manual
Classificação:	...
Classificação para criação de vias:	...
Forma:	Anexo
Nível de acesso:	Público
Tipo do Modelo:	Freemarker <pre> 4 Este é o editor 5 [6 @grupo] 7 [@texto titulo="Ano Posse/Contratação" var="ano" largura="". 8 @/grupo] 9 [@grupo depende="anoAjax"] 10 [#if (ano!="")] 11 [@mensagem titulo="Alerta" texto="Ano deve ser preenchido em vermelho=true/"] 12 [/#if] 13 [/@grupo] 14]/@grupo]</pre>
<input type="button" value="Ok"/> <input type="button" value="Aplicar"/> <input type="button" value="Cancelar"/>	

Editor FM

Observação: No desenvolvimento, geralmente escolhemos “**Anexo**” como Forma, apenas por convenção, desta forma, todas aplicações sendo desenvolvidas residem neste tipo de documento, até serem migrados para o tipo correto.

12.3.2 - Criando uma macro FM

A - No SIGA-DOC, acessar **SIGA / Ferramentas / Cadastro de Modelos / Novo**



```

2202 }
2203 checkAppletStarted();
2204 </script>
2205 [/#macro]
2206
2207
  
```

Salvar

B - Escrever o código da macro

```

Modelos:
GERAL
1 [function par 'parâmetro']
2 [if param[parâmetro1?]]
3 [else]
4 [else]
5 [selecionar]
6 [return ""]
7 [else]
8 [assign inlineTemplate = ["#{assign default:$var = true;}", "assignInlineTemplate"]?interpret /]
9 [inlineTemplate/]
10 [/function]
11
12 [function formatarCPF fmtCPF_param]
13 [!início do comentário]
14 [Aplicação: Função para formatar um CPF]
15 [Acionador: fmtCPF]
16 [Autor: Ruben]
17 [Data: 13/03/2012]
18 [Descrição: Esta função obtém uma string contendo o CPF a ser formatado e o devolve formatado
19 com a seguinte apresentação: 999.999.999-99
20
21
22
  
```

C - Salvar

Orgão Usuário: Conselho da Justiça Federal

Salvar

Orgão Usuário: Seção Judiciária do Espírito Santo

Salvar

Orgão Usuário: Seção Judiciária do Rio de Janeiro

Salvar

Orgão Usuário: Tribunal Regional Federal - 2ª Região

Salvar

Orgão Usuário: ORGÃO TESTE ZZ

Salvar

É importante observar no arquivo de macros, a seção correspondente a sua instituição. A primeira seção é para todos, chamada de Geral, e é a que os desenvolvedores na SJRJ utilizam.

12.4 - Backup, Restore e comparando códigos FM

A forma de se levar um produto (código) seguindo a corrente upstream, do desenvolvimento para homologação e da homologação para produção é através de Copy e Paste, em cada ambiente do SIGA instalado.



É prática comum dos desenvolvedores da infraestrutura do SIGA-DOC realizar o downstream forçado, da produção para a homologação e desenvolvimento, porque como dito, perde-se o controle das atualizações que são realizadas diretamente na produção. Como não temos um BD local (ver diagrama em 12.1 ... e até poderíamos ter, porém o Oracle pesa muito a máquina), **todos os nossos desenvolvimentos são perdidos**. Para minimizar este problema, e por questões de segurança, podemos realizar um **backup** de todas as nossas aplicações e das macros. No fundo da tela, próximo ao botão |NOVO|, temos o botão |Exportar XML|, que gera um arquivo XML com todas as aplicações ou macros.

BACKUP

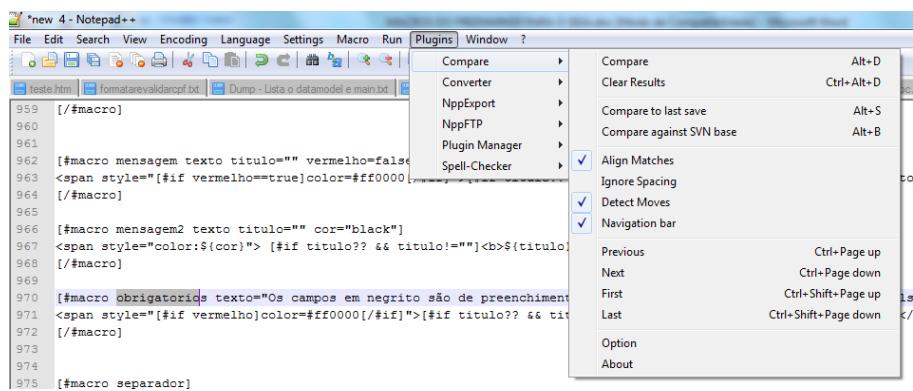
Exemplo, no caso das aplicações:

TRF-PRES Portaria da Presidência	TRF-PRES-PORTARIA Licença Saúde para Desembargador
TRF-PRES Resolução da Presidência	TRF-PRES Resolução da Presidência
TRF-Proposta e Concessão de Diárias	TRF-Proposta e Concessão de Diárias
Termo	SGP: Recebimento da Segunda Via de Crachá
Termo	SGP: Recebimento de Crachá
Termo	Termo - Modelo Livre

Novo Exportar XML



Podemos também utilizar uma ferramente de comparação de códigos (Diff), para comparar o arquivo XML do desenvolvimento com o da produção, por exemplo. Pode-se utilizar a ferramenta free Notepad++ com o plugin Compare.

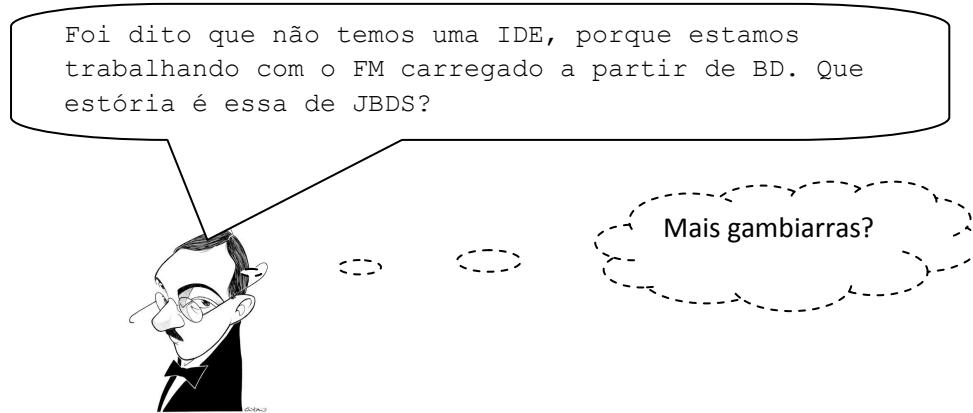


Notepad++ e o plugin Compare, permite que se compare os arquivos, por exemplo, desenv20062012.xml e producao20062012.xml. Ele abrirá 2 janelas e apresentará as diferenças, desta forma você pode tomar uma decisão.

RESTORE

Infelizmente não temos uma ferramenta, tipo importar. O restore tem que ser feito como Copy e Paste, a partir do arquivo XML. Pode-se copiar e colar todas as macros ao mesmo tempo, porém no caso das aplicações, temos que fazer uma a uma.

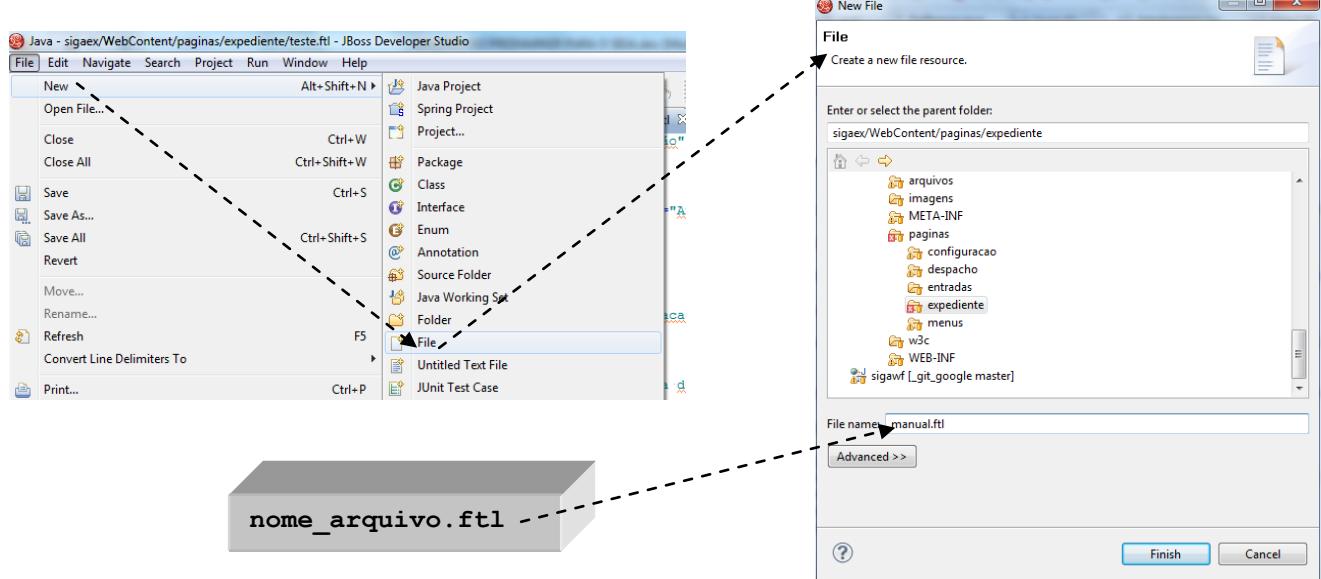
12.5 – Sintaxe HighLight para o FM no JBoss Developer Studio (JBDS)



Isto é verdade, porém o IDE JBDS trabalha com arquivos .ftl. Neste caso podemos escrever a aplicação neste ambiente, usufruir do seu editor com intuito de obtermos uma aplicação sintaticamente correta, e depois, copiá-la e colá-la no editor do SIGA-DOC. No futuro, tentaremos configurar o JBOSS local para que ele rode também a aplicação, enquanto isso, poderemos utilizar o Integrador que será visto na próxima seção.

Para obter detalhes de como instalar o plugin Freemaker no Eclipse visite o capítulo [Anexo 9 – Instalando o plugin do Freemaker no Eclipse](#)

12.5.1 - Criando um arquivo .ftl



12.5.2 - Tela Principal do editor

Área do código

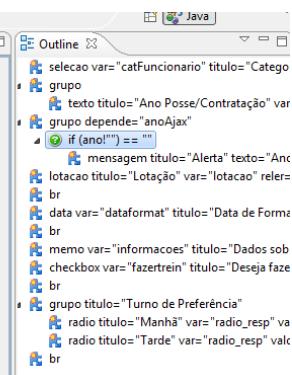
```

1 {@selecao var="catFuncionario" titulo="Categoria do Funcionário"
2   reler=true opcoes="Servidor; Terceiro"/>}
3
4
5
6 {@grupo}
7 {@texto titulo="Ano Posse/Contratação" var="ano" largura="4" maxcaracterest=}
8 [/@grupo]
9 {@grupo depende="anoAjax"}
10 ...{if (ano!="") == ""}{}
11 ...{@mensagem titulo="Alerta" texto="Ano deve ser preenchido." vermelho=true}
12 ...
13 ...{/#if}
14 [/@grupo]
15 ...
16 ...
17 {@lotacao titulo="Lotação" var="lotacao" reler=true idAjax="lotacaoAjax"/>
18 ...
19 {@br/}
20 ...
21 {@data var="dataformat" titulo="Data de Formatura:" obrigatorio=true/}
22 ...
23 {@br/}
```

Clicando no **[/#IF]**, em amarelo, que é um IF de fechamento, ele passa para o IF de abertura e vice-versa, facilitando entender os blocos de código.

O outliner estrutura o código e para cada comando, IF, CASE etc., coloca um ícone específico, como o  para o IF.

Outliner



12.5.3 - Acusando o erro

Observar o erro, falta o [, e o apontamento

```
8 [/@grupo]
9 {@grupo depende="anoAjax"}@|
10 . . . [if (ano!="") != "" ]@|
11 . . . {@mensagem titulo="Alerta" texto="Ano deve ser preenchido."}@|
12 . . . vermelho=true/]@|
13 . . . [/@grupo]@|
14 . . . @|
```

Se clicar aqui, teremos

12.5.4 - Sintaxe dos comandos

```
37 @|
38 [#]
  assign
attempt
break
break
case
compress
default
else
elseif
escape
fallback
flush
```

```
6 [@grupo]@|
Encountered "[\"@grupo]" at line 14, column 1.
Was expecting one of:
<ATTEMPT> ...
<IF> ...
<ELSE_IF> ...
<LIST> ...
<FOREACH> ...
<SWITCH> ...
<ASSIGN> ...
<GLOBAL_ASSIGN> ...
<LOCAL_ASSIGN> ...
<INCLUDE> ...
<IMPORT> ...
<FUNCTION> ...
<MACRO> ...
<TRANSFORM> ...
<VISIT> ...
<STOP> ...
<RETURN> ...
<CALL> ...
<SETTING> ...
<COMPRESS> ...
<COMMENT> ...
<TERSE_COMMENT> ...
<KNOPARSE> ...
<END_IF> ...
<ELSE> ...
<BREAK> ...
<SIMPLE_RETURN> ...
<HALT> ...
<FLUSH> ...
<TRIM> ...
<LTRIM> ...
<NOTRIM> ...
<NESTED> ...
<SIMPLE_RECURSE> ...
<RECURSE> ...
<FALLBACK> ...
<ESCAPE> ...
<NOESCAPE> ...
<UNIFIED_CALL> ...
<WHITESPACE> ...
<PRINTABLE_CHARS> ...
<FALSE_ALERT> ...
"$" ...
"#" ...
```

12.6.1 – Introdução

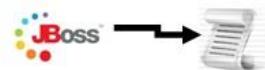
Os modelos freemarker na base de dados do SIGA-DOC promovem zero deployment, porém o desenvolvimento é prejudicado. Por outro lado, o desenvolvimento no JBDS é excelente, porém necessita de deployment. É aí que entra o Integrador, combinando o melhor dos dois mundos: desenvolvimento no JBDS com zero deployment.

O Integrador é um aplicativo (modelo do SIGA-DOC) desenvolvido com os seguintes objetivos em mente:

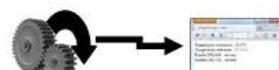
INTEGRADOR

CARGA (Sem a necessidade de copy / paste)

A carga se dá a partir do JBOSS, do projeto/pasta /sigaex/freemarker/



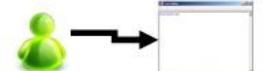
- ❖ Por default, template.ftl é carregado para o editor. Caso não exista, o editor aparecerá vazio;
- ❖ Posteriormente, pode-se informar outro modelo a ser carregado;

EXECUÇÃO (On the fly. Sem a necessidade de cadastrar o modelo)

- ❖ O modelo pode ser executado de forma normal, como em produção;
- ❖ Porém, pode ser executado em modo DUMP. Neste caso, um dump do data-model e variáveis do template é realizado no início e outro no final da execução do modelo freemarker. Dois botões serão inseridos na aplicação para visualização dos dumps. Útil como ferramenta de debug.

REGISTRO NO SIGA-DOC (Sem sair do ambiente, sem copy / paste)

- ❖ O Modelo pode ser cadastrado (registrado) na base de desenvolvimento do SIGA-DOC diretamente.

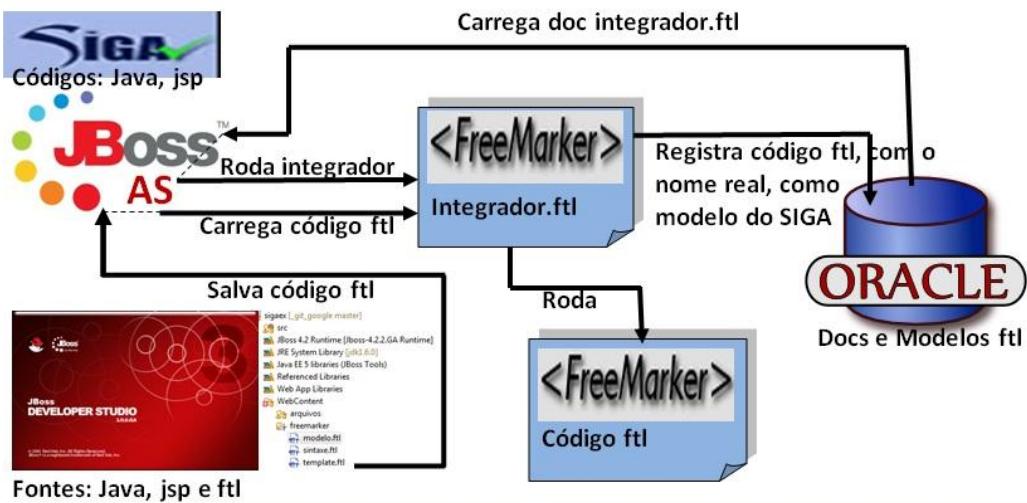
EDIÇÃO (Sem sair do ambiente, sem copy / paste)

- ❖ Pode-se colar ou escrever um código ou comandos diretamente no editor. Otimo para testar comandos, funções, pequenas aplicações e etc. Pode-se colocar traps na aplicação sem afetar o código original. Muito útil para debug.

Mas como funciona e por que estamos privilegiando o desenvolvimento no JBDS?

Pelos motivos expostos nas seções anteriores, o qual faço um resumo abaixo, tanto do funcionamento quanto das vantagens em utilizar o JBDS.

FREEMARKER NO JBOSS DEVELOPER STUDIO

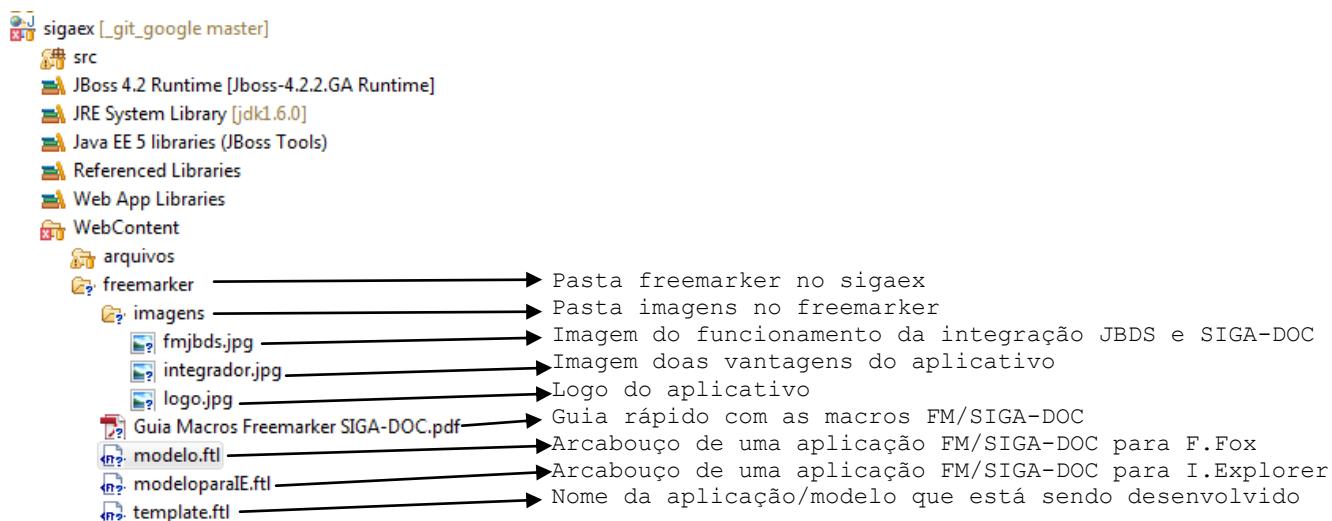


Vantagens

- Editor ftl no JBDS é infinitamente melhor;
- Os templates ficam em work área local. Nunca são substituídos;
- Controle de versão dos templates pelo Git;
- Desenvolvimento offline, sem timeout. Não depende do JBAS, SIGA-DOC (SDc) e Oracle;
- Ambiente voltado ao desenvolvimento de código, facilitando e acelerando o mesmo.

Mas só funciona com a IDE JBDS e o Servidor de Aplicação JBOSS AS? Embora não testado, acredito que funcione com qualquer IDE e com qualquer Servidor de Aplicação, visto que o acomplamento com estes produtos é muito fraco, ou melhor, inexistente.

12.6.2 - Setup do Ambiente



Explicação dos objetos:

Pasta freemarker: é necessário criá-la debaixo do projeto sigaex.

Pasta imagens: é necessário criá-la debaixo da pasta freemarker. Conterá as imagens do aplicativo que serão carregadas quando o mesmo for iniciado.

Guia da macros feemarker / SIGA-DOC: um resumo do manual contendo apenas as macros, com intuito de servir de guia ao desenvolvimento. Aqui, o desenvolvedor obterá as respostas que necessita, exemplos e etc.

Modelo.ftl: um esqueleto (V0) de uma aplicação freemarker para o SIGA-DOC. Deve-se utilizar este modelo para o Fire Fox.

ModeloparaIE.ftl: idem ao anterior, porém para o Internet Explorer.

Template.ftl: este deve ser o nome do aplicativo que está sendo desenvolvido. Não é obrigatório utilizar este procedimento, porém o aplicativo Integrador já carrega este aplicativo na área de edição.

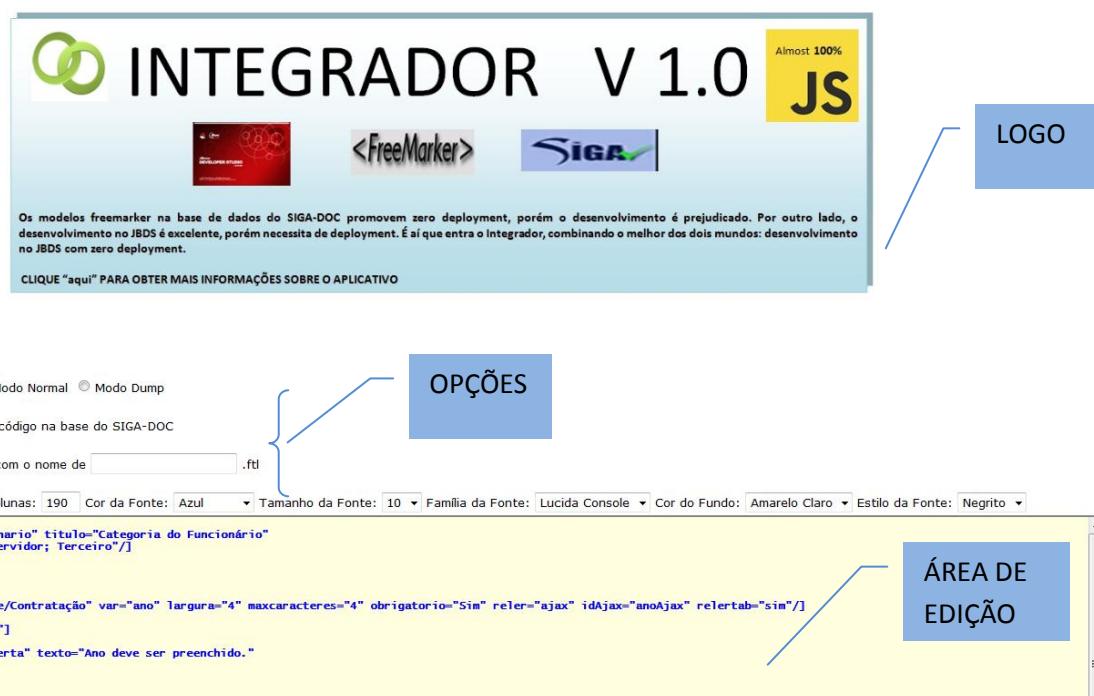
12.6.3 - Funcionalidades do Aplicativo

12.6.3.1. - Chamada do aplicativo

Tipo: Anexo

Tipo: IntegraJBDS-SIGADOC Versão 1.0.0.0

12.6.3.2 - Tela Inicial



É importante observar que, por default, o aplicativo já carrega o modelo chamado template.ftl do projeto/sigaex/freemarker na área de edição. Desta forma, é interessante que o aplicativo atual, que está sendo desenvolvido, possua este nome. Depois que for testado e carregado na base do SIGA-DOC, pode-se renomeá-lo com o nome real. Isto não é uma obrigação, visto que podemos após a inicialização do aplicativo indicar outro modelo a ser carregado.

12.6.3.3 - Ajuda

Ao clicar na imagem / logo do aplicativo (da tela inicial) será apresentado uma ajuda conforme apresentado abaixo.



Ao se clicar na imagem acima, voltamos para a tela inicial.

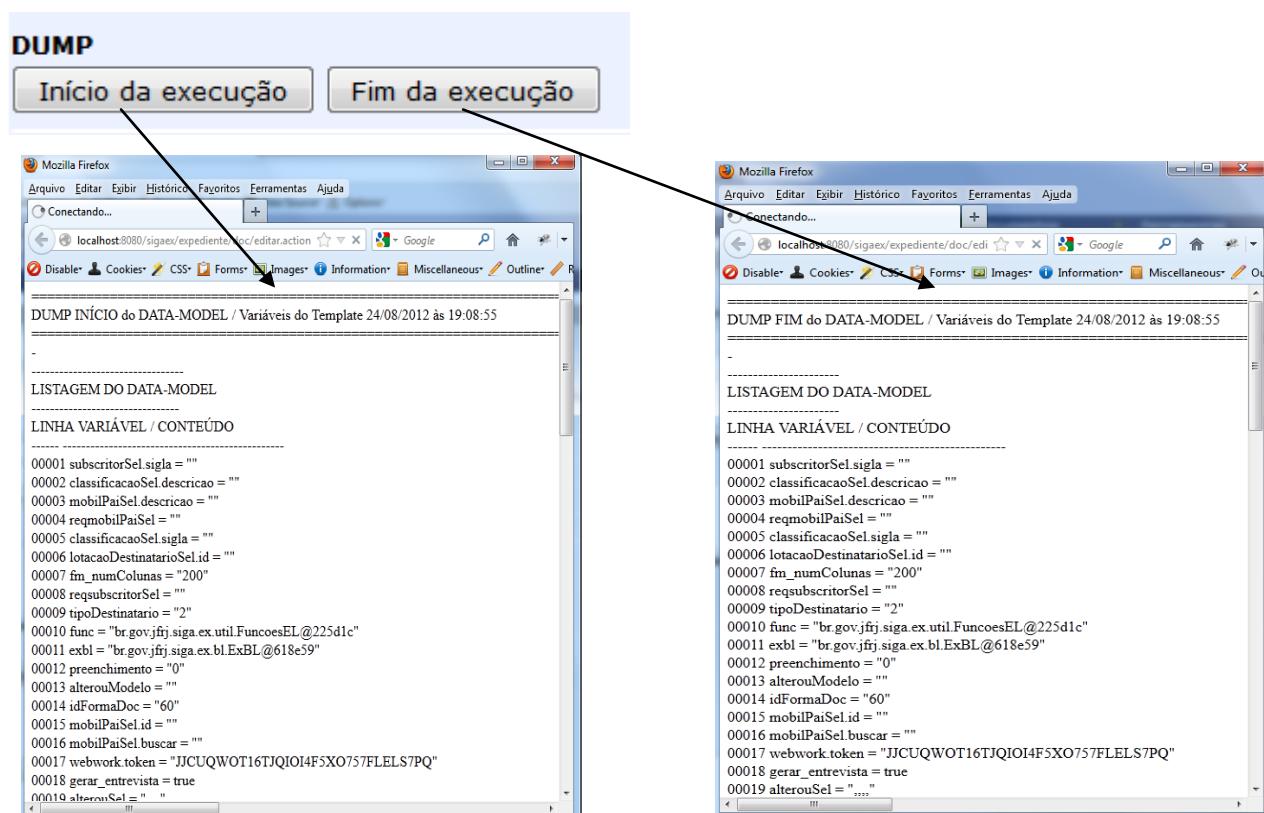
12.6.3.4 - Opção / Executar

Executar o código Modo Normal Modo Dump

Pemite que se execute o código que está na área de edição. Pode-se rodar em modo:

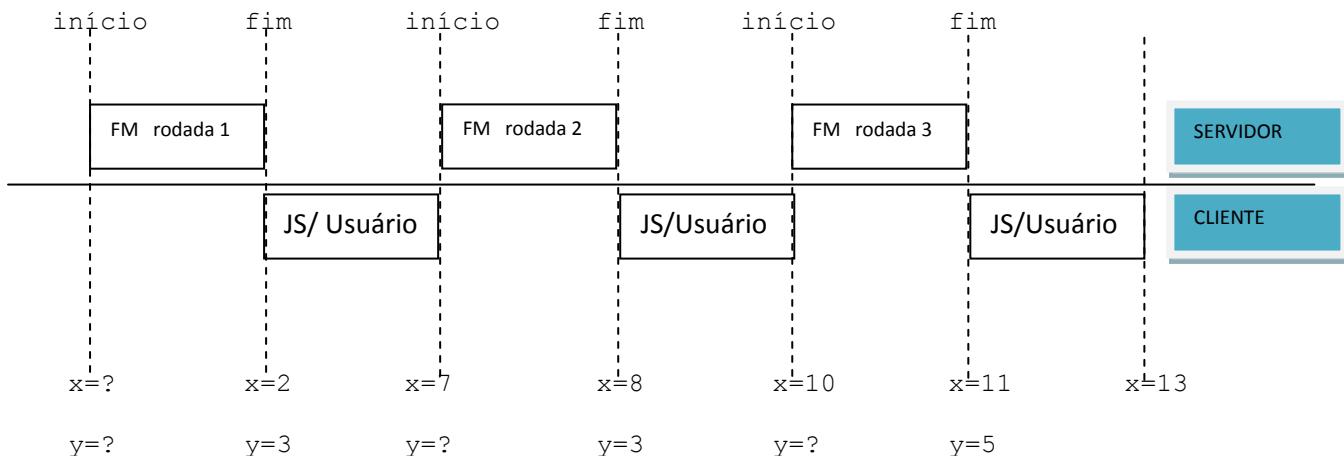
Normal: ou seja, rodá-lo como se em produção estivesse.;

Dump: neste caso, são inseridos dois comandos freemarker (duas chamadas de macros) no código, uma no início (`[@dumpvarantes/]`) e outra no final (`[@dumpvardepois/]`). Estas macros tirarão uma foto do data-model + variáveis do template e as armazenarão para posterior consulta na aplicação sendo executada. Será automaticamente inserido os seguintes botões na aplicação;



Para que servem estes dumps?

Para analisar o estado (conteúdo) das variáveis ao longo do processamento. Pode-se guardar os conteúdos entre cada rodada do FM no servidor, ou seja, a cada vez que a tela é enviada ou relida (reler=true), com intuito a identificar quem / o que está promovendo tais mudanças nas variáveis.



Supor que x seja uma variável de um form HTML e y uma variável freemarker.

Pode-se observar a evolução da variável x.

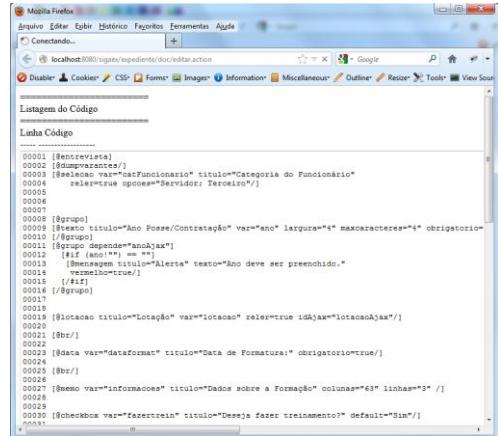
- No início não existia, e depois o FM associou o valor 2 e enviou a tela;
- O usuário ou algum aplicativo JS alterou o valor para 7;
- O FM alterou o valor para 8 e enviou a tela;
- e assim por diante ...

As variáveis freemarker, no início da execução do template ainda não existem (?), são criadas durante a execução (via assign) e destruídas quando o freemarker acaba a execução do template.

As variáveis do data-model só podem ser lidas pelo FM, nunca alteradas. Quem altera o valor das variáveis do data-model é o aplicativo JAVA que executa o FM e carrega o template.

Quando executamos um código a seguinte janela (popup) aparece, exibindo o código com a numeração. Por Quê? Já não temos a numeração no editor JBDS?

O problema é que o editor do Integrador é aberto, como veremos mais abaixo. Você pode desenvolver um código ad-hoc, copiar e colar um código qualquer ou alterar o código na área de edição, para inserir traps (tipo: passei por aqui, conteúdo de x é ...), por exemplo. Desta forma, seria perdida a referência da numeração, e caso ocorra um erro (o freemarker indica a linha que ocorreu) seria difícil identificar a linha que ocorreu o mesmo. Esta janela permite você identificar a linha.



12.6.3.5 - Opção / Registrar

Registrar (cadastrar) o código na base do SIGA-DOC

Permite que o código seja registrado (cadsatrado) no na base do SIGA-DOC. A seguinte tela será apresentada:

INTEGRADOR - Registrar (Cadastrar) o template na base do SIGA-DOC Desenvolvimento

Edição de Modelo

Modelo	
Nome:	<input type="text"/>
Descrição:	<input type="text"/>
Forma:	Anexo <input type="button" value="▼"/>
Nível de acesso:	Público <input type="button" value="▼"/>
Tipo do Modelo:	Freemarker <input type="button" value="▼"/>
<input type="button" value="Clique aqui para cadastrar o template"/>	

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

O nome é o que aparecerá como Modelo e a forma como Tipo na tela Documento do SIGA-DOC

Destinatário:	Órgão Integrado <input type="button" value="▼"/> <input type="button" value="..."/>
Tipo:	Memorando <input type="button" value="▼"/>
Modelo:	Memorando <input type="button" value="▼"/>

Após a inclusão, será exibida a tela com a lista dos modelos.

12.6.3.6 - Opção / Carregar

Carregar outro modelo,com o nome de .ftl

Permite que se carregue outro modelo / template da localização /sigae/freemarker. Lembre-se que o template.ftl é carregado como default na incialização do Integrador. A partir daqui podemos carregar outro modelo para a área de edição.

12.6.3.7 - Opção / Editar

Editar											
Linhas:	30	Colunas:	200	Cor da Fonte:	Azul	<input type="button" value="▼"/>					
Tamanho da Fonte:	10	<input type="button" value="▼"/>	Família da Fonte:	Courier New	<input type="button" value="▼"/>	Cor do Fundo:	Branco	<input type="button" value="▼"/>	Estilo da Fonte:	Negrito	<input type="button" value="▼"/>

Por default, a área de edição é readonly. Clicando em Editar, abre-se a área de edição para alteração. Pode-se customizar o editor em relação a cor, tamanho da fonte e etc.

Cenário de utilização:

Você está desenvolvendo o template no JBDS e carrega-o no editor. Porém antes de rodar, você deseja adicionar traps, tipo: 'passei no Then do If do matric=="1212"', 'passei no case xyz', 'Valor da variável x =' \${x} e etc. Ou seja, adicionar código de depuração, que só serve para a rodada. Qualquer alteração no código na área de edição não afeta o original no JBDS, deixando-o intacto.



Você até pode registrar no SIGA-DOC o código que está na área de edição, mesmo que ele seja diferente do código no JBDS, porém não é uma boa prática. Deve-se sempre consertar os bugs no código original no JBDS e carregá-lo novamente no Integrador.



Será que o meu código no JBDS está igual ao registrado na base do SIGA-DOC?

Utilize o Notepad++, plugin Compare.

The screenshot shows the Notepad++ interface with two code snippets highlighted in red boxes. The left snippet is from a file named 'Dump - Lista variaveis enumeraveis.btp'. The right snippet is from a file named 'FORMULARIOsigadoc.btp'. Both snippets contain similar code related to parameters and lists. A black arrow points from the left red box to the 'Compare' submenu in the Plugins menu. Another black arrow points from the right red box to the same submenu, indicating the two files being compared.

Códigos a serem comparados

Rodar FM no JBDS para que?

No ponto de vista de treino é perfeitamente aceitável. Como já dito algumas vezes, programar macros FM no SIGA-DOC é como programar circundado por quatro paredes, com muitas limitações. Este capítulo mostra como instalar o FM, criar um pequeno projeto, e em adição, fala um pouco a respeito de: *HttpServlet, Request Response e Contexto (Solicitação, Sessão e Aplicação)*, visto que estas foram as fundações do SIGA-DOC.



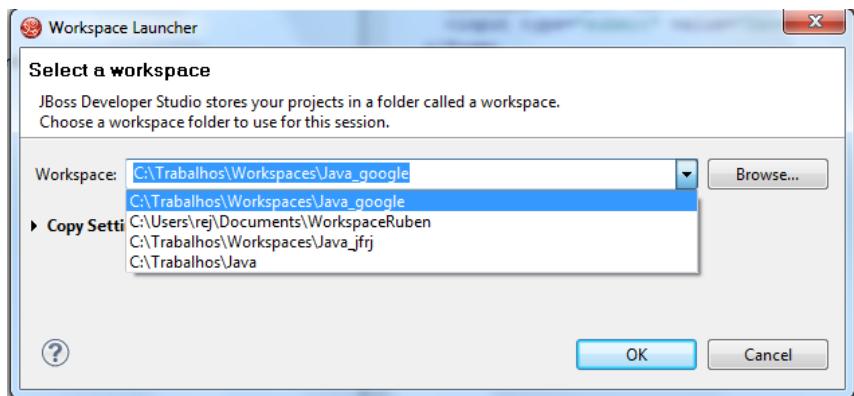
13.1 – INTRODUÇÃO

O objetivo é instalar o Freemarker de forma standalone e tradicional (carregando a partir do File System) no JBoss Developer Studio (JBDS) para que se possa testar a ferramenta, independente do SIGA-DOC, com intuito de treinar, conhecer a linguagem e seu relacionamento com o Java (servlets).

13.2 – Passo-a-Passo

13.2.1 – Criar um novo Workspace

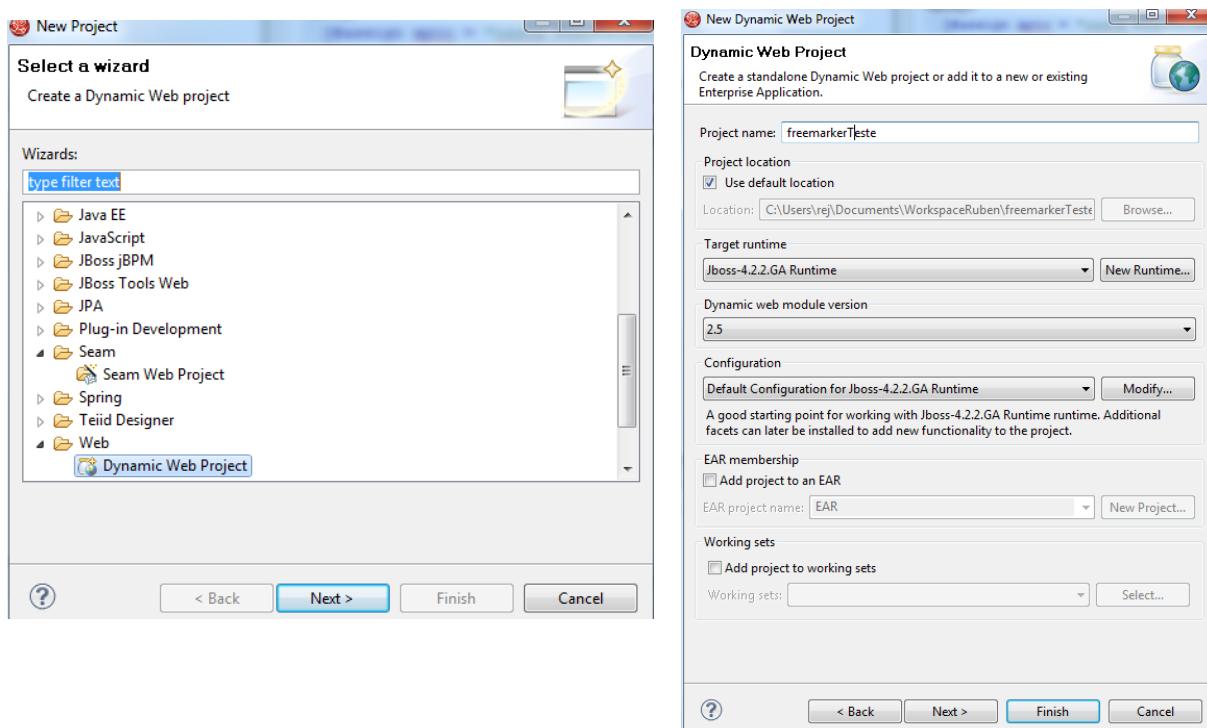
No JBDS, (File / Switch WorkSpace / Other) para se ter um ambiente totalmente limpo, independente do SIGA.



Eu criei no meu usuário: c:/Users/rej/Documents/WorkspaceRuben

13.2.2 – Criar um projeto

No JBDS, (File / New / Project). Selecionar Dynamic Web Project

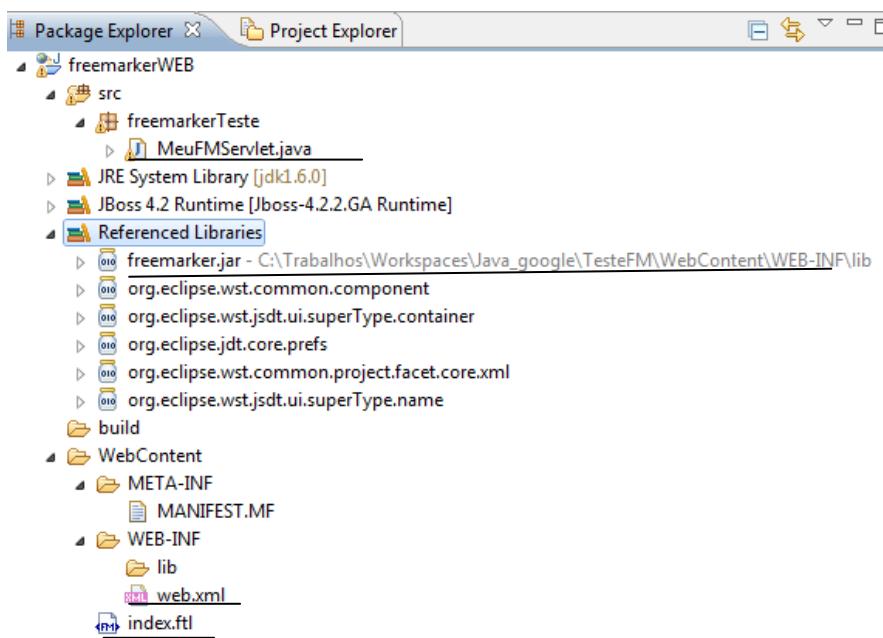


13.2.3 - Popular o projeto

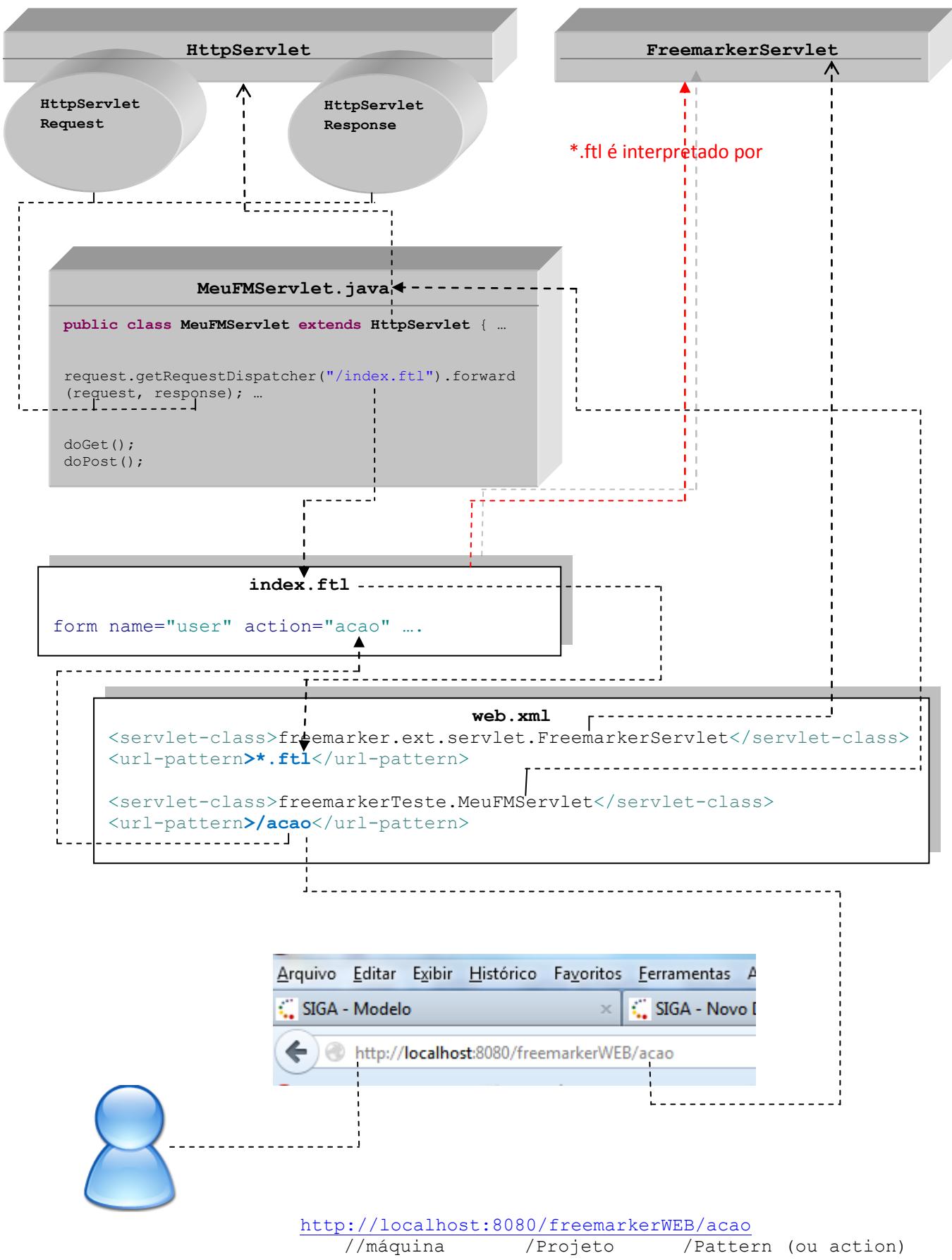
Neste projeto utilizei 4 arquivos a saber:

Arquivo	Descrição
MeuFMServlet.java	Aplicação Java única que processará a página WEB
web.xml (Deployment Descriptor)	Arquivo de configuração da aplicação WEB onde são descritos os servlets utilizados, os parâmetros, URL pattern e outros. É melhor entrar com esta configuração utilizando a janela do JBDS, e não criar o arquivo por fora e colar, pois geralmente dá erro
index.ftl	Template FM que será utilizado para criar a página WEB
freemarker.jar	Biblioteca do FM que deve ser baixada do site da empresa.

Estrutura do Projeto:



13.2.4 - Relacionamento, descrição dos arquivos e como funciona o projeto



13.2.4.1 - O usuário digita no browser
<http://localhost:8080/freemarkerWEB/acao>
// máquina / Projeto / Pattern (ou action)

O arquivo web.xml, indica que o /acao deve ser processado pelo servlet MeuFMServlet

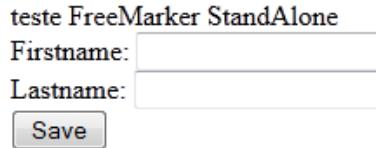
13.2.4.2 - O servlet do projeto entra em ação

A primeira vez que o request é feito o mesmo é executado pelo método (default) GET, e neste caso, o servlet MeuFMServlet passa o controle para o método doGet, que promove um forward para index.ftl.

É importante observar que o servlet **MeuFMServlet**, estende a classe **HttpServlet** (que é uma classe abstrata com os métodos doGet e doPost que devem ser overridden na classe estendida). **MeuFMServlet** também recebe como parâmetros os ponteiros para os objetos **HttpServletRequest** e **HttpServletResponse** que são criados toda vez pelo Server Container (JBoss, Tomcat ...) quando um cliente envia um request.

13.2.4.3 - O interpretador FM entra em ação

É para ser apresentada a página index.ftl, porém um regra no arquivo web.xml indica que todas páginas *.ftl devem ser processadas pela classe FreeMarkerServlet, da biblioteca do FM, que interpreta o template, gerando o HTML. A página é processada e enviada ao browser cliente.



teste FreeMarker StandAlone

Firstname:

Lastname:

13.2.4.4 - O usuário preenche os campos e submete (save) a página

Neste caso, quando a página é submetida, o método POST entra em ação (vide form com método post) e o servlet MeuFMServlet passa o controle ao método doPost, que executa o método doGet para reenvio da página.

Numa aplicação real, aqui no método doPost, obteríamos os campos digitados, criticaríamos e persistiríamos a informação, como no exemplo abaixo:

```
String firstname = request.getParameter("firstname");
String lastname = request.getParameter("lastname");

if(null != firstname && null != lastname
    && !firstname.isEmpty() && !lastname.isEmpty()) {

    /* Poderíamos criticar e persistir a informação */
}
```

getParameter() é um método do objeto HttpServletRequest.

Observações sobre o projeto:

- Teoricamente, utilizando o método getWriter() do HttpServletResponse poderíamos criar a nossa página HTML, sem a necessidade de utilizarmos FTL ou JSP, ou seja, o nosso servlet montaria o HTML e processaria as informações, porém isto quebraria o conceito de camadas do MVC, como visto capítulo 1, ou seja, um servlet controller não pode ser responsável pelo view.

Exemplo: trecho escrevendo código HTML
PrintWriter out = response.getWriter();

```
out.print("<h1>Hello world </h1>");
```

- Outro ponto, no mundo real, é que poderíamos ter um servlet que processaria todos os tipos de ações (action servlet) e que chamaria a página (FTL ou JSP), o outro servlet, responsável para processar a respectiva ação.

```
http://localhost:8080/meuprojeto/Controller?action={  
    salvarProduto  
    listarProduto  
    ...}
```

Onde o servlet Controller está definido no web.xml,

```
<servlet-name>Controller</servlet-name>  
<servlet-class>meusservlets.ActionControllerServlet</servlet-class>
```

Exemplo: trecho de código de um Action Servlet

```
String action = request.getParameter("action");  
...  
String dispatchUrl = null;  
if (action.equals("listarProduto")) {  
    dispatchUrl = "/freemarker/listaproduto.ftl";  
}  
else if (action.equals("salvarProduto")) {  
    dispatchUrl = "/freemarker/salvaproduto.ftl";  
}  
if (dispatchUrl != null) {  
    RequestDispatcher rd = request.getRequestDispatcher(dispatchUrl);  
    rd.forward(request, response);  
}
```

Outra abordagem, em vez de utilizar Action Servlet, é empregar Action Manager e View manager, principalmente com Struts, onde temos o Action Class.

- Poderíamos passar um Data-Model (passar func, doc, exbl como no SIGA-DOC) para todos os templates FM, como o index.ftl. Para isto, deveríamos criar uma subclasse da classe **FreemarkerServlet** e fazer um override do método preTemplateProcess(), inserindo o Data-Model conforme visto na seção 4.4 – Como criar um Data-Model, antes do processamento do template.

O FM coloca 4 hashes disponíveis, como default, no Data-Model. As variáveis hash são : Request, Session e Application e uma outra adicional chamada de RequestParameters (que fornece acesso aos parâmetros do HTTP Request).

13.2.5 - Conteúdo dos arquivos utilizados

MeuFMServlet.java

```
package freemarkerTeste;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
import javax.servlet.ServletException;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
public class MeuFMServlet extends HttpServlet {  
    private static final long serialVersionUID = 1L;
```

```

@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
request.getRequestDispatcher("/index.ftl").forward(request, response);
}
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    doGet(request, response);
}
/*
* O doPost é chamado quando o form é submetido. Observe que o doGet() é chamado
dentro do doPost(), porque queremos enviar o mesmo formulário sempre */

```

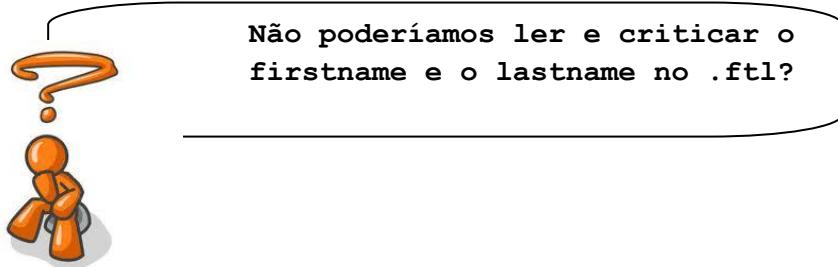
index.ftl

```

<html>
<head><title>FreeMarker Hello World</title>
<body>
    [#assign xpto = "Teste FreeMarker StandAlone"]
    ${xpto}
    <br/>
    <form name="user" action="acao" method="post">
        Firstname: <input type="text" name="firstname"/>
        <br/>
        Lastname: <input type="text" name="lastname"/>
        <br/>
        <input type="submit" value="Save" />
    </form>
</body>
</html>

```

Obs: neste projeto, o template .ftl é apenas um condutor de form. Diferente dos formulários do SIGA-DOC em que os campos da tela são lidos e criticados pelo template, aqui tudo é passado para a aplicação Java MeuFMServlet.



Tipo: [#if firstname == ""]
 Atenção, o primeiro nome deve ser fornecido
 [#else]
 [/#if]

Se fizermos isto diretamente não funcionará. Teremos que acessar o Data-Model, o hash **RequestParameters** para obter os campos da tela. Faremos isto mais adiante.

Por que no SIGA-DOC podemos? Porque os campos da tela são gravados no Data-Model pela aplicação Java que carrega o template e invoca o parser FM.

web.xml (Deployment Descriptor)

É importante observar a configuração do servlet do freemarker, onde utilizei o tag_syntax = square_bracket para configurar o [# ...], em vez de <# ...>.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
  http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
  <display-name>SIGA - Gestão Documental</display-name>
  <servlet>
    <servlet-name>freemarker</servlet-name>
    <servlet-class>freemarker.ext.servlet.FreemarkerServlet</servlet-class>
    <init-param>
      <param-name>TemplatePath</param-name>
      <param-value>/</param-value>
    </init-param>
    <init-param>
      <param-name>NoCache</param-name>
      <param-value>true</param-value>
    </init-param>
    <init-param>
      <param-name>ContentType</param-name>
      <param-value>text/html; charset=UTF-8</param-value>
    </init-param>
    <init-param>
      <param-name>template_update_delay</param-name>
      <param-value>0</param-value>
    </init-param>
    <init-param>
      <param-name>default_encoding</param-name>
      <param-value>ISO-8859-1</param-value>
    </init-param>
    <init-param>
      <param-name>tag_syntax</param-name>
      <param-value>square_bracket</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet>
    <servlet-name>meufm_servlet</servlet-name>
    <servlet-class>freemarkerTeste.MeuFMServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>freemarker</servlet-name>
    <url-pattern>*.ftl</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
    <servlet-name>meufm_servlet</servlet-name>
    <url-pattern>/acao</url-pattern>
  </servlet-mapping>
  <session-config>
    <session-timeout>120</session-timeout>
  </session-config>
  <!-- The Usual Welcome File List -->
  <welcome-file-list>
    <welcome-file>/index.html</welcome-file>
```

```

</welcome-file-list>
<security-constraint>
<display-name>Freemarker MVC Views</display-name>
<web-resource-collection>
<web-resource-name>Freemarker MVC Views</web-resource-name>
<url-pattern>*.ftl</url-pattern>
</web-resource-collection>
</security-constraint>
</web-app>

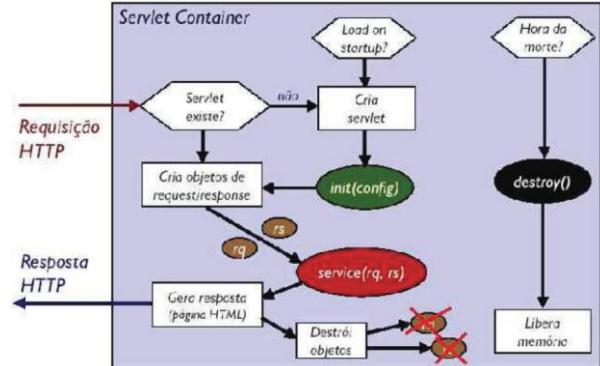
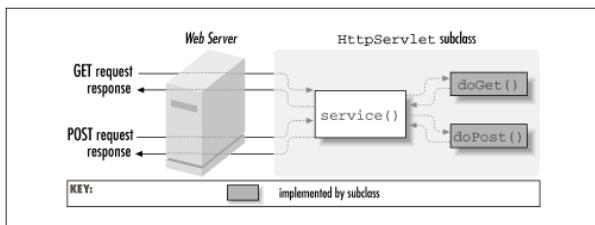
```

13.3 – Entendendo HttpServlet, HttpServletRequest e HttpServletResponse

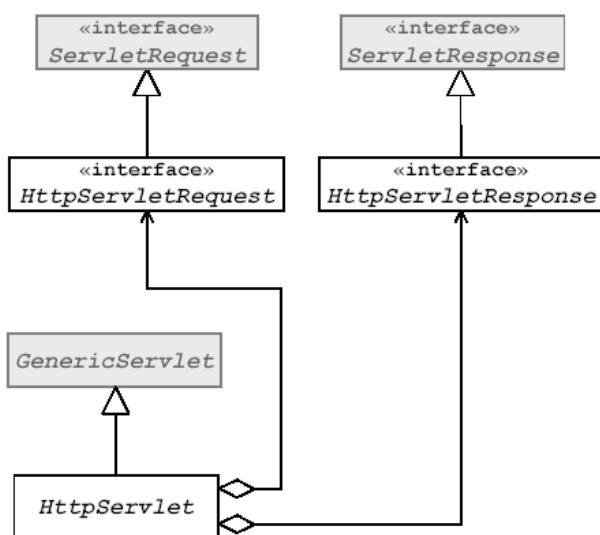
13.3.1 - Os métodos GET e POST com os respectivos request e response

Request: vai campos (parâmetros) do formulário, do browser cliente, para o servidor.

Response: vai HTML montado no servidor, para o cliente.



13.3.2 - A Classe abstrata HttpServlet



HttpServlet: Esta classe estende a classe GenericServlet. Possui basicamente seis métodos que são chamados automaticamente de acordo com os métodos HTTP que são

requisitados. Por exemplo se a solicitação do seu browser for feita pelo método GET, no servlet será chamado o método doGet().

Os seis métodos são:

- **doPost();**
- **doPut();**
- **doGet();**
- **doDelete();**
- **doOption();**
- **doTrave();**

13.3.3 - Os Objetos HttpServletRequest e HttpServletResponse

HttpServletRequest:

As solicitações HTTP que o browser envia pelo cliente ao servidor com informações importantes, tais como cookies e dados do formulário são tratadas a partir deste objeto, que é um dos dois argumentos dos métodos doGet ou doPost.

Principal função: recuperar dados enviados ao servidor.

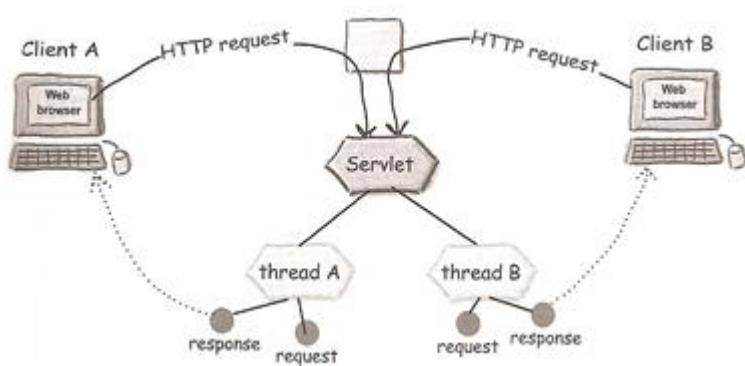
- **getHeaderNames();** - pega todos os nomes dos cabeçalhos .
- **getHeader();** - pega todos os valores do cabeçalho .
- **getQueryString();** - pega a Query String completa.
- **getParameterNames();** - pega todos os nomes dos parâmetros passados.
- **getParameterValues();** - recuperação de parâmetros de múltiplos valores.
- **getParameter();** - recuperação de parâmetros de acordo com o nome passado.

HttpServletResponse:

Responsável por manipular a resposta dado a requisição, permitindo a escrita de conteúdo, seja HTML ou qualquer outro MIME. Esta interface é o outro argumento dos métodos doGet e doPost.

- **addHeader(String nome, String valor)** - adiciona cabeçalho HTTP
- **setContentType(tipo MIME)** - define o tipo MIME que será usado
- **sendRedirect(String location)** - envia informação de redirecionamento
- **Writer getWriter()** - obtém um Writer para gerar a saída.
- **OutputStream getOutputStream()** - obtém um OutputStream geralmente usada para gerar formatos diferentes de texto (imagens, etc.)
- **addCookie(Cookie c)** - adiciona um novo cookie

Servlet atende cada requisição abrindo um thread. MultiThread



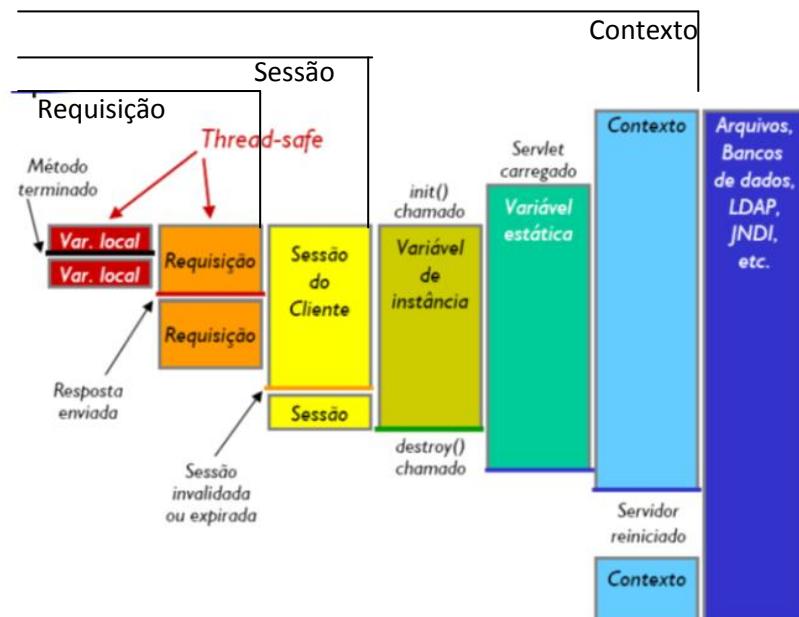
13.3.4 - Escopo de Variáveis e Objetos

Escopo variável	Descrição
-----------------	-----------

Solicitação (Request)	Eles duram apenas o tempo de uma requisição; ao término da mesma todos os dados setados no objeto que representa o escopo terão sido apagados.
Sessão (Session)	Quando um usuário acessar o sistema web, ele estabelece com o servidor uma sessão. Os dados setados no objeto que representa este escopo existem desde o instante inicial, quando o usuário acessa a aplicação, até que essa expire por inatividade, seja voluntariamente ou finalizada pela aplicação.
Aplicação (Application)	Os objetos vivem desde a inicialização do servidor de aplicação até que ele seja finalizado. Podem ser compartilhados entre diversos servlets.
Página (page)	Se aplica somente ao ciclo de vida das JSP's da sua aplicação. Nele os objetos são definidos em e para cada página e existem apenas para elas mesmas.

Escopo objeto	Descrição
Requisição (HttpServletRequest)	<p>É thread-safe!</p> <p>Acessível por: recurso da requisição</p> <p>Por quanto tempo existe: enquanto a requisição existir</p> <p>Exemplo de uso: dados de um determinado item a venda</p>
Sessão (HttpSession)	<p>Não é thread-safe!</p> <p>Acessível por: recursos de uma sessão</p> <p>Por quanto tempo existe: enquanto a sessão existir</p> <p>Exemplo de uso: carrinho de compras</p>
Contexto (HttpContext)	<p>Não é thread-safe!</p> <p>Acessível por: qualquer recurso da aplicação</p> <p>Por quanto tempo existe: enquanto existir a aplicação</p> <p>Exemplo de uso: número de usuários ativos</p>

Quadro geral de persistência das variáveis



Sessão – Manutenção de estado (`HttpSession session = request.getSession();`)

- HTTP é um protocolo que não mantém estado;

Na ausência do HttpSession as seguintes técnicas são utilizadas:

- Solução mais utilizada: Cookies, informações que são armazenadas no browser cliente, enviadas pelo servidor durante uma resposta de uma requisição;
- Solução alternativa: "Hidden Fields", campos <input> que são passados tela a tela. Aumenta o trabalho da manutenção e o tempo de transferência da página;
- Reescrita da URL: You can append some extra data on the end of each URL that identifies the session, and the server can associate that session identifier with data it has stored about that session. For example, with `http://tutorialspoint.com/file.htm;sessionid=12345`, the session identifier is attached as `sessionid=12345` which can be accessed at the web server to identify the client. URL rewriting is a better way to maintain sessions and works for the browsers when they don't support cookies but here drawback is that you would have generate every URL dynamically to assign a session ID though page is simple static HTML page.

ServletContext

- Permite ao Servlet buscar informações do seu servidor/container;
- Existe um único ServletContext para cada WebApplication;
- Utilizado para armazenar dados comuns a varios Servlets;
- Está contido em ServletConfig, que também permite ao servlet obter valores de inicialização;

13.3.5 – JAVA – Setando e Recuperando – Escopo de Variáveis e Objetos

No método doGet

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
// Setando atributos...
request.setAttribute("nome", "valor"); // Setando no escopo de requisição
request.getSession().setAttribute("nome", "valor"); // Setando no escopo de sessão.
getServletContext().setAttribute("nome", "valor"); // Setando no escopo de aplicação.
// Recuperando valores
Object valorRequest = request.getAttribute("nome"); //Pegando no escopo de requisição
Object valorSession = request.getSession().getAttribute("nome"); //Pegando no escopo de sessão.
Object valorApplication = getServletContext().getAttribute("nome"); //Pegando no escopo de aplicação.
}
```

No método doPost

```
public void doPost...{  
    resp.setContentType("text/html");  
    PrintWriter out = resp.getWriter();  
    out.print("Os dados armazenados como atributos são: <BR><BR>");  
  
    if(req.getAttribute("nome") == null) {  
        req.setAttribute("nome", req.getParameter("nome"));  
        req.setAttribute("idade", req.getParameter("idade"));  
        req.setAttribute("cidade", req.getParameter("cidade"));  
        req.setAttribute("estado", req.getParameter("estado"));  
    }  
  
    if(req.getSession().getAttribute("qtdade") == null) {  
        req.getSession().setAttribute("qtdade", 1);  
    } else {  
        req.getSession().setAttribute("qtdade",  
            (Integer)req.getSession().getAttribute("qtdade")+1);  
    }  
  
    out.print("Nome: " + req.getAttribute("nome") + "<BR>");  
    out.print("Idade: " + req.getAttribute("idade") + "<BR>");  
    out.print("Cidade: " + req.getAttribute("cidade") + "<BR>");  
    out.print("Estado: " + req.getAttribute("estado") + "<BR><BR>");  
    out.print("Quantidade de Cadastros: " + req.getSession().getAttribute("qtdade") +  
        "<BR><BR>");  
    out.print("Muito obrigado!!!");  
}
```

Atributos de
requisição

Atributos de
sessão

13.3.6 – FREEMARKER – Recuperando – Escopo de Variáveis e Objetos

Application Scope Attribute

Assuming there's an attribute with name myApplicationAttribute in the Application scope.

```
<#if Application.myApplicationAttribute?exists>  
    ${Application.myApplicationAttribute}  
</#if>
```

or

```
<@s.property value="#${application.myApplicationAttribute}" />
```

Session Scope Attribute

Assuming there's an attribute with name mySessionAttribute in the Session scope.

```
<#if Session.mySessionAttribute?exists>  
    ${Session.mySessionAttribute}  
</#if>
```

or

```
<@s.property value="#${session.mySessionAttribute}" />
```

Observação: para se ter acesso aos dados da Session, deve-se configurar o FreeMarker View Resolver para expo-los:

```
<property name="exposeSessionAttributes"><value>true</value></property>
```

Request Scope Attribute

Assuming there's an attribute with name 'myRequestAttribute' in the Request scope.

```
<#if Request.myRequestAttribute?exists>
    ${Request.myRequestAttribute}
</#if>
```

or

```
<@s.property value="#request.myRequestAttribute" />
```

Observação: para se ter acesso aos dados do Request, deve-se configurar o Freemarker View Resolver para expo-los:

Request Parameter

Assuming there's a request parameter myParameter
(eg. <http://host/myApp/myAction.action?myParameter=one>).

```
<#if Parameters.myParameter?exists>
    ${Parameters.myParameter}
</#if>
```

Listando todos os parâmetros

```
<#list RequestParameters?keys as key>
    ${Key} = ${RequestParameters[key]}
</#list>
```

or

```
<@s.property value="#parameters.myParameter" />
```

Observação: Sintaxe alternativa

To get access to **Request Parameters** use one of the following syntaxes where test is the request property

```
 ${RequestParameters['test']}  
 or  
 ${RequestParameters.test}
```

To get access to **Request Attributes** use one of the following where test is the request attribute

```
 ${Request['test']}  
 or  
 ${Request.test}
```

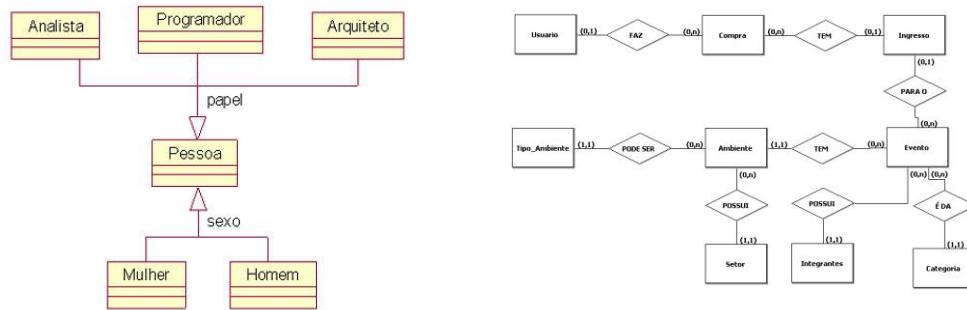
Context parameter

Assuming there's a parameter with the name myContextParam in framework context.

```
 ${stack.findValue('#myContextParam') }  
  
 or  
  
 <@s.property value="#{ #myContextParam}" />
```

Classes e Modelo de dados

A planta baixa de um sistema, na minha humilde opinião, é dada pelo modelo de dados e o modelo de classes. Programar, sem eles, é como furar uma parede sem saber o que está por trás da mesma. É óbvio que para um programador FM não existe a necessidade de se conhecer o modelo de dados, e somente algumas classes estão disponíveis a navegação. De qualquer forma, mesmo que estejamos pregando tachinhas na parede, é interessante saber o que está por detrás.

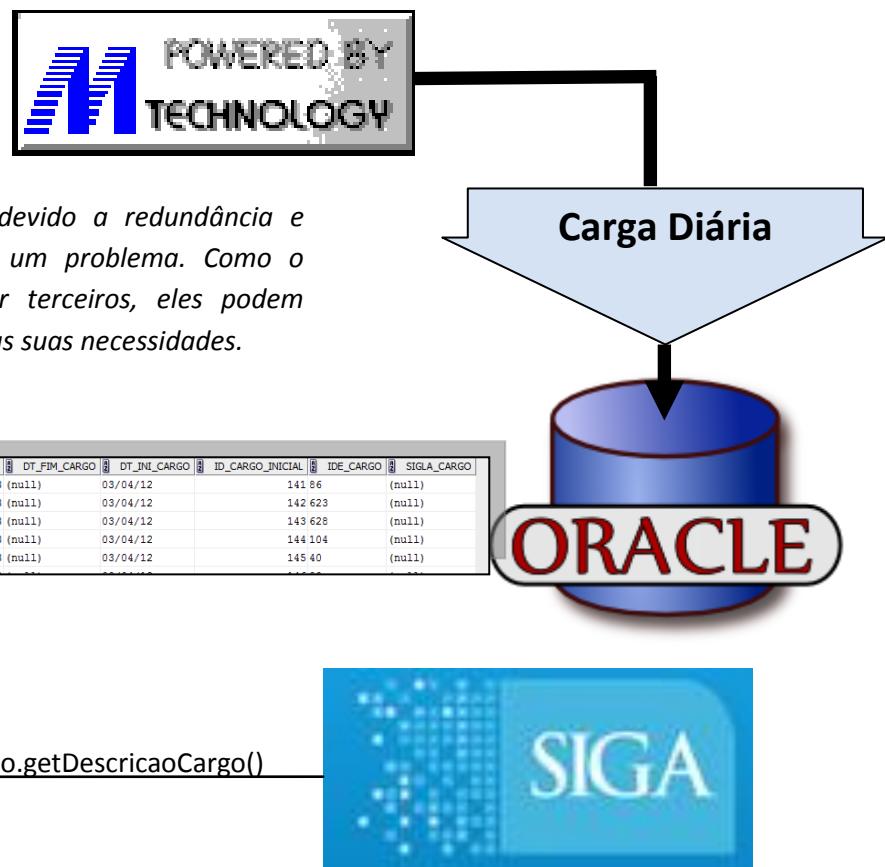


14.1 – Introdução

Sistemas:

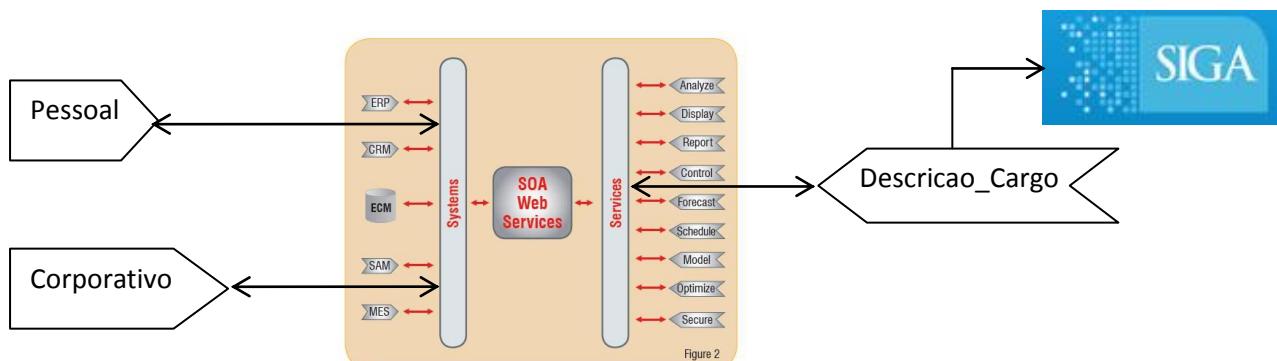
Sigla	Descrição
Dp	Pessoal (Servidores e Magistrados)
Cp	Corporativo (Tabelas de uso geral)
Ex	Expedientes (Uso no SIGA-DOC)

Por que o SIGA-DOC possui tabelas e classes relacionadas ao sistema de Pessoal e Corporativo? Bom, estas bases residem originalmente no MUMPS (também conhecida como sistema M). Como o SIGA-DOC, feito em JAVA/WEB/BD Relacional, teria acesso a estas bases? O resultado foi criar estas tabelas no SIGA-DOC e populá-las a partir do MUMPS, através de rotinas batch diária.



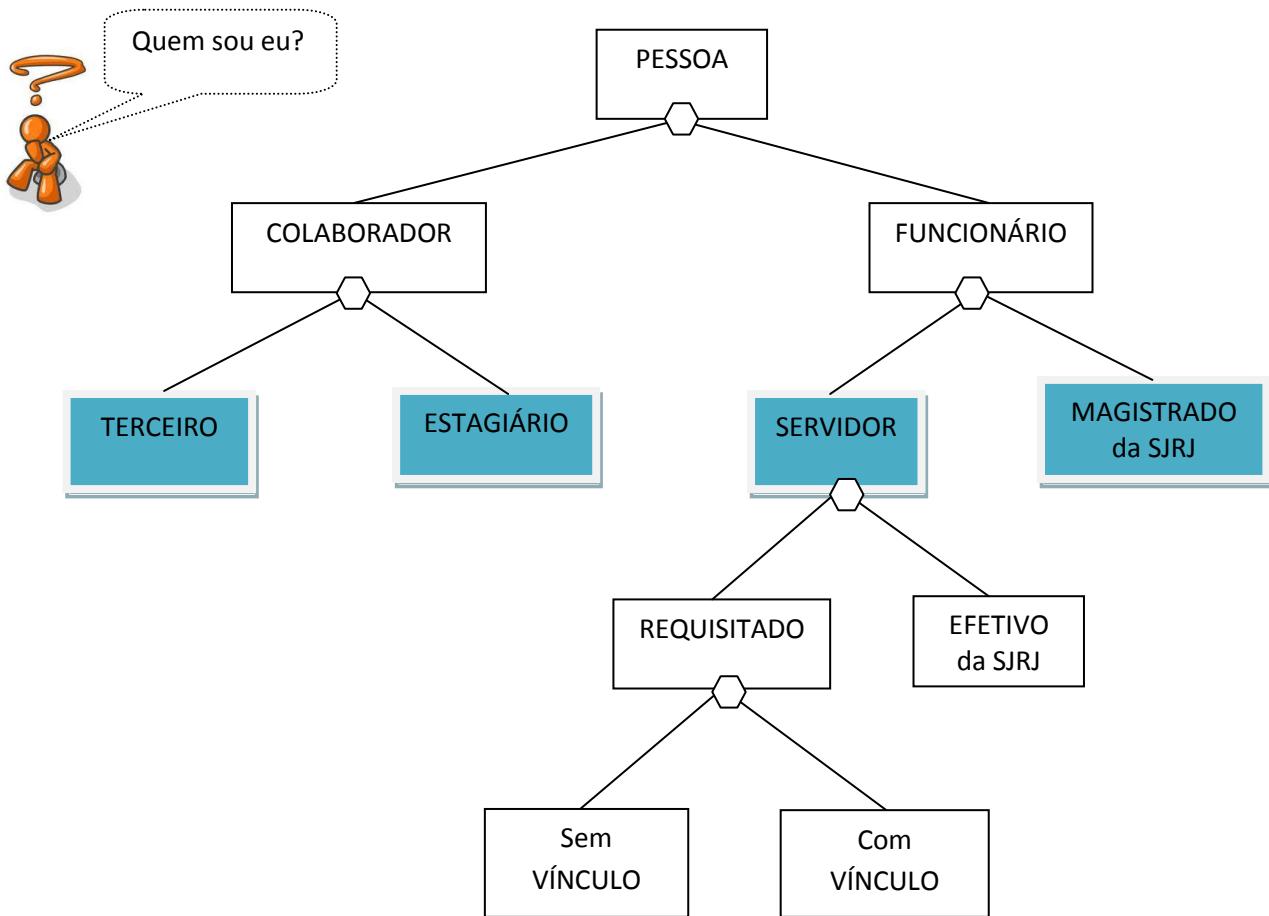
Para onde o futuro aponta?

Para uma estrutura de serviços, nos moldes SOA / WEB Services, no qual os sistemas disponibilizarão os seus serviços de forma padronizada, independente da implementação dos mesmos (classes, executáveis, queries ...).



Pessoa, Funcionário, Servidor, Magistrado?

Muitas vezes fazemos confusão com estes termos, talvez por falta de uma padronização. Uma possível generalização / especialização para Pessoa:



Colaborador: é externo, e não é considerado funcionário público, pois não exerce cargo ou função pública. Estamos falando dos estagiários e dos terceirizados.

Funcionário: é funcionário público, pois exerce cargo ou função pública. Se uma pessoa, mesmo sem emprego, é requisitada para trabalhar na SJRJ em regime de cargo em confiança ela estará exercendo uma função pública.

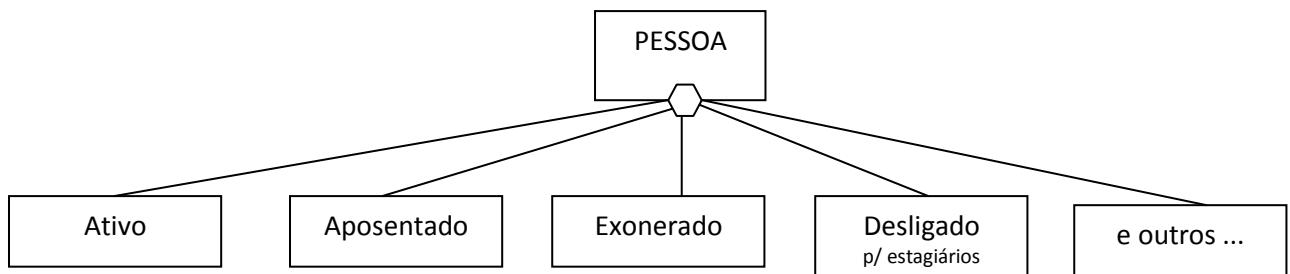
Servidor: são todos que exercem cargo ou função pública na SJRJ, excetuando-se os magistrados.

O efetivo é aquele que prestou concurso, tomou posse e exerce um cargo na SJRJ. O efetivo também pode adicionalmente possuir uma função comissionada (de confiança).

O requisitado é aquele que vem de fora prestar os seus serviços na SJRJ, podendo ter algum vínculo com outro órgão público (ou seja, é funcionário público de outro órgão no âmbito federal, estadual ou municipal) ou não.

O SIGA-DOC só reconhece as categorias **que estão hachuriadas em azul**. Quem fornece essa categorização é a entidade / classe Tipo_Pessoa (do sistema corporativo), que no caso da SJRJ lista somente os quatro tipos. Se precisarmos, por exemplo, listar somente os efetivos de um setor / diretoria, não temos como realizá-lo. É possível que com a implantação do novo sistema de benefícios, estas lacunas sejam preenchidas.

Situação Funcional do Funcionário



A situação é obtida através da entidade / classe, SIT_FUNCIONAL, e do atributo data-fim (que é uma idiosincrasia nossa, da SJRJ)

Ativo: Situação funcional = 1 e data_fim_pessoa = nulo

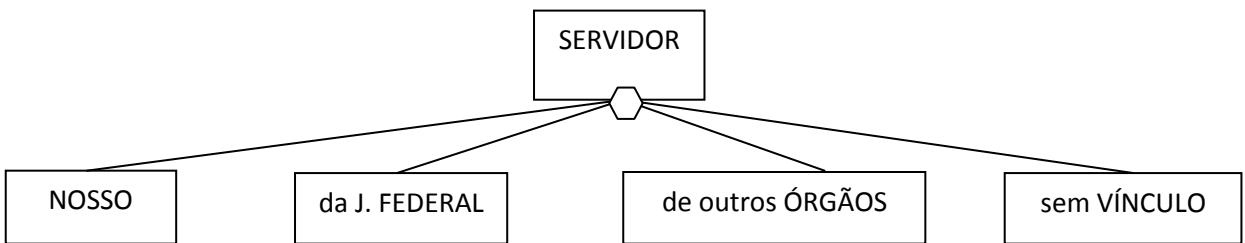
Inativo: (Aposentado): Situação funcional = 4 e data_fim_pessoa != nulo

Para nós, data-fim != nulo representa uma exclusão lógica do funcionário, no sentido que ele não presta mais serviços para a SJRJ.

Workarounds

Tentar categorizar um servidor em analista / técnico pela descrição do cargo não é uma boa, pois a descrição não está padronizada. Premissas do tipo, todo requisitado possui o cargo nulo, ou categorizar a pessoa por range de matrícula também não é uma boa prática.

Outra abordagem para Servidores:



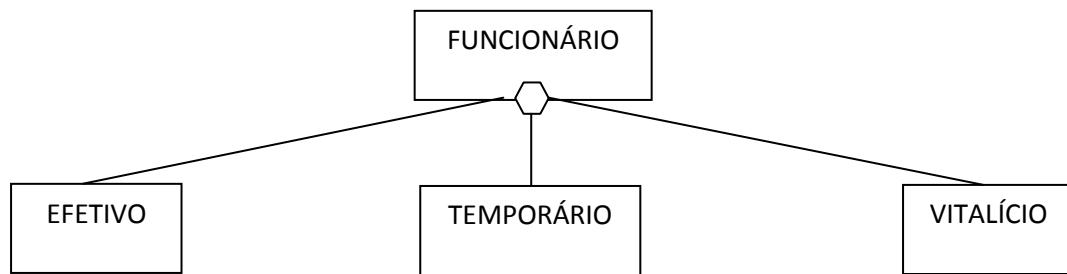
Nosso: pessoa que prestou concurso e tomou posse na SJRJ.

Da Justiça Federal: pessoa requisitada de outras Seções Judiciárias e/ou TRFs.

De Outros Órgãos: pessoa requisitada de outros órgãos públicos, tanto no âmbito federal, estadual ou municipal.

Sem Vínculo: pessoa requisitada que não possui nenhum vínculo com órgão público.

Abordagem Legal (lei) para funcionário:



A nomeação, em função da natureza do cargo a ser provido, será feita:

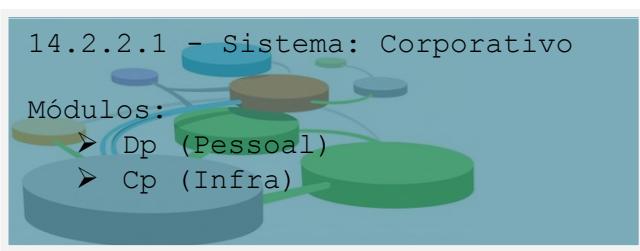
- Em caráter efetivo (permanente), através de concurso público;
- Em caráter temporário, cargos ou funções de direção, chefia, assistência e assessoramento superior e intermediário;
- Em caráter vitalício, magistrados, os membros do Ministério Público e os Conselheiros dos Tribunais de Contas.

14.2 – ENTIDADES/TABELAS

14.2.1 – Introdução

É notório que as tabelas e os modelos de dados (ER) ficam em segundo plano quando se trata de desenvolvimento orientado a objetos, visto que o desenvolvedor acessa e manipula classes e objetos. Mas não devemos nos esquecer, enquanto não tivermos um BD orientado a objetos, que o BD relacional e suas tabelas são as fundações de qualquer sistema, até porque quando vamos desenvolver orientado a objetos, muitas vezes, estas tabelas já existem. Mais adiante, no mapeamento objeto x relacional, veremos que as tabelas possuem associação direta com as classes de negócio, praticamente de 1 : 1, como por exemplo, citando apenas uma, a tabela DP_CARGO e as classes DpCargo e AbstractDpCargo.

14.2.2 – Relação / Descrição das entidades / tabelas



O sistema corporativo engloba as tabelas oriundas de outros sistemas da SJRJ. Estas tabelas não são mantidas pelo SIGA-DOC, porém são utilizadas. Temos as tabelas do módulo Dp, pessoal, e Cp, corporativo ppd, que provê tabelas de infraestrutura, como feriados, UF e etc.

Módulo: DP Pessoal

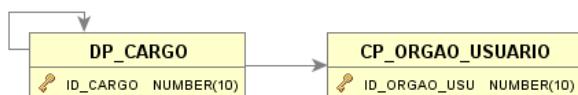
Relação das Tabelas
CAD_SIT_FUNCIONAL
DP_CARGO
DP_ESTADO_CIVIL
DP_FUNCAO_CONFIANCA
DP_LOTACAO
DP_PADRAO_REFERENCIA
DP_PESSOA
DP_PROVIDIMENTO
DP_SUBSTITUICAO

Tabela	Descrição / Observação	
CAD_SIT_FUNCIONAL	Situação Funcional: 1 - ativo, 4 - aposentado, 5 - exonerado ...	
Atributos	PK / FK	Descrição
ID_CAD_SIT_FUNCIONAL	PK	
DSC_SIT_FUNCIONAL		
ID_MUMPS		
Imported keys (para quem esta tabela aponta)		

Exported keys (quais tabelas apontam para esta tabela)

Tabela	Descrição / Observação	
DP_CARGO	Cargo do servidor: analista, técnico... / Existe IDE_CARGO que é o ID do cargo no Mumps	
Atributos	PK / FK	Descrição
ID_CARGO	PK	
NOME_CARGO		
ID_ORGAO_USU	FK	Chave estrangeira que aponta para a tabela de Órgão Usuário.
DT_FIM_CARGO		Data de fim de vigência do cargo.
DT_INI_CARGO		Data de inicio de vigência do cargo.
ID_CARGO_INICIAL	FK	
IDE_CARGO		Identificador de cargo importado de tabela externa.
SIGLA_CARGO		

Imported Keys (para quem esta tabela aponta)



Exported Keys (quais tabelas apontam para esta tabela)

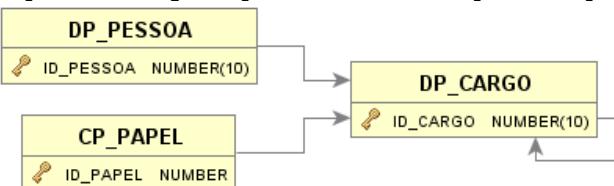
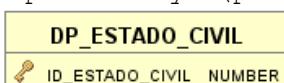


Tabela	Descrição / Observação	
DP_ESTADO_CIVIL	Estado Civil: 1-casado, 2-solteiro ...	
Atributos	PK / FK	Descrição
ID_ESTADO_CIVIL	PK	Representa o código identificador do estado civil, é sequencial e transparente para o usuário.
NM_ESTADO_CIVIL		Nome do estado civil, campo livre.

Imported Keys (para quem esta tabela aponta)



Exported Keys (quais tabelas apontam para esta tabela)

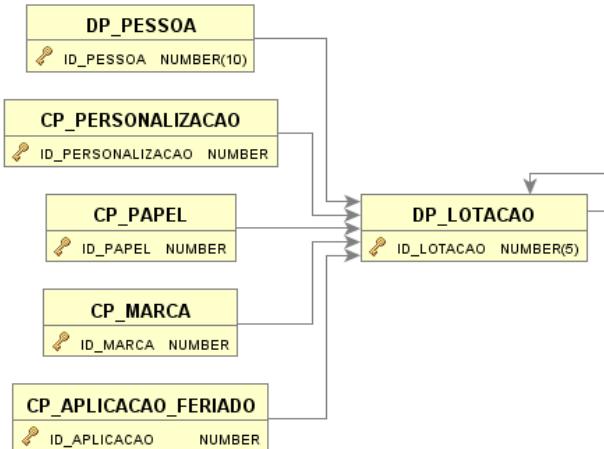


Tabela	Descrição / Observação	
DP_FUNCAO_CONFIANCA	Função de confiança: São as FCs (Funções Comissionadas) e CCs (Cargos em Comissão). Assistente, assessor, diretor, supervisor ... / IDE_FUNCAO_CONFIANCA que é o ID da função no Mumps	
Atributos	PK / FK	Descrição
ID_FUNCAO_CONFIANCA	PK	
NOME_FUNCAO_CONFIANCA		

NIVEL_FUNCAO_CONFIANCA		
COD_FOLHA_FUNCAO_CONFIANCA		
DT_INI_FUNCAO_CONFIANCA		
DT_FIM_FUNCAO_CONFIANCA		
ID_FUNCAO_CONFIANCA_PAIS	FK	
CATEGORIA_FUNCAO_CONFIANCA		
ID_ORGAO_USUARIOS	FK	Chave estrangeira que aponta para CP_ORGAO_USUARIO
IDE_FUNCAO_CONFIANCA		Chave da função originária de
ID_FUN_CONF_INI	FK	Chave da função originária desta função.
SIGLA_FUNCAO_CONFIANCA		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
DP_LOTACAO	Lotação: Lotação do servidor em termos de SUB (STI, por exemplo), seção (SESIE, por exemplo), setor, divisão, coordenação (CSIS, por exemplo) ...	
Atributos	PK / FK	Descrição
ID_LOTACAO	PK	
DATA_INI_LOT		Data de criação ou alteração da lotação.
DATA_FIM_LOT		Data de extinção ou fim de determinada situação da lotação.
NOME_LOTACAO		
ID_LOTACAO_PAIS	FK	Identificador da Lotação superior em hierarquia.
SIGLA_LOTACAO		
ID_ORGAO_USUARIOS	FK	Sigla da lotação.
IDE_LOTACAO	FK	
ID_LOTACAO_INI	FK	
ID_TP_LOTACAO		
Imported keys (para quem esta tabela aponta)		

Exported keys (quais tabelas apontam para esta tabela)



Tabela

DP_PADRAO_REFERENCIA

Descrição / Observação

Padrão de Referência: indica a posição do servidor no plano de carreira (Cargo/Classe/Padrão). Exemplo: NS-B8 (Cargo - S - Superior, Classe - B e Padrão - 8) *** (atualmente vazia)

Atributos

ID_PADRAO_REFERENCIA

PK / FK

Descrição

ID_PADRAO_REFERENCIA_PAIS

PK

DSC_PADRAO

DSC_CLASSE

DSC_NIVEL

PADRAO_REFERENCIA_DT_FIM

ID_ORGAO_USU

FK

ID_ORGAO_USU

FK

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

Tabela

DP_PESSOA

Descrição / Observação

Contem os dados de todos os funcionários (ativos e inativos) que prestam ou prestaram serviços a SJRJ. / Existe IDE_PESSOA que é o ID da pessoa no Mumps

Atributos

ID_PESSOA

PK / FK

Descrição

DATA_INI_PESSOA

PK

DATA_FIM_PESSOA

CPF_PESSOA

NOME_PESSOA

DATA_NASC_PESSOA

MATRICULA

ID_LOTACAO

FK

ID_CARGO

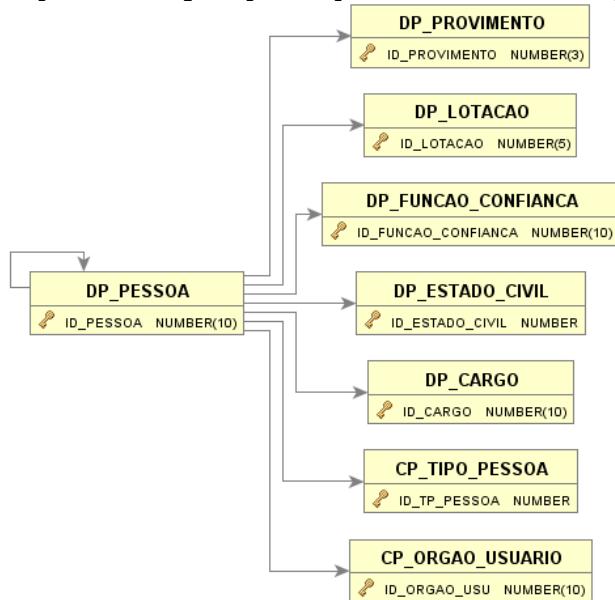
FK

ID_FUNCAO_CONFIANCA

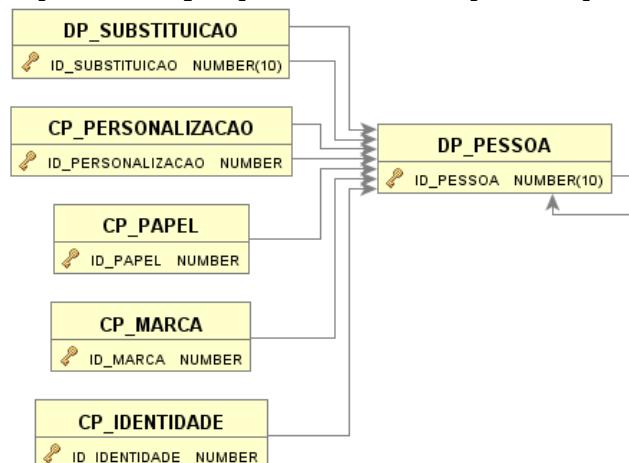
FK

SESB_PESSOA		
EMAIL_PESSOA		
TP_SERVIDOR_PESSOA		
SIGLA_PESSOA		
SEXO_PESSOA		
GRAU_INSTRUCAO_PESSOA		
TP_SANGUINEO_PESSOA		
NACIONALIDADE_PESSOA		
DATA_POSSE_PESSOA		
DATA_NOMEACAO_PESSOA		
DATA_PUBLICACAO_PESSOA		
DATA_INICIO_EXERCICIO_PESSOA		
ATO_NOMEACAO_PESSOA		
SITUACAO_FUNCIONAL_PESSOA		
ID_PROVIMENTO	FK	
NATURALIDADE_PESSOA		
FG_IMPRIME_END		Flag que indica se o servidor optou por imprimir o endereço no contracheque.
DSC_PADRAO_REFERENCIA_PESSOA		
ID_ORGAO_USU	FK	
IDE_PESSOA		
ID_PESSOA_INICIAL	FK	
ENDERECO_PESSOA		
BAIRRO_PESSOA		
CIDADE_PESSOA		
CEP_PESSOA		
TELEFONE_PESSOA		
RG_PESSOA		
RG_ORGAO_PESSOA		
RG_DATA_EXPEDICAO_PESSOA		
RG_UF_PESSOA		
ID_ESTADO_CIVIL	FK	
ID_TP_PESSOA	FK	
NOME_EXIBICAO		

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)



Tabela

Descrição / Observação

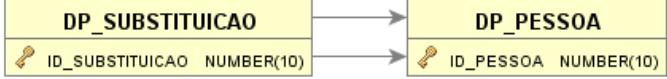
DP_PROVIMENTO	Tabela que armazena as formas de provimento da pessoa. Ato de preencher cargo ou ofício público por nomeação, promoção, transferência, reintegração, readmissão, aproveitamento ou reversão. *** (atualmente vazia)	
Atributos	PK / FK	Descrição
ID_PROVIMENTO	PK	

DSC_PROVIMENTO

Imported keys (para quem esta tabela aponta)

Exported keys (quais tabelas apontam para esta tabela)



Tabela	Descrição / Observação	
DP_SUBSTITUICAO	Cadastro de substituto. O próprio titular cadastra seus substitutos, ou seja, pessoas que responderão em seu nome nas ausências (férias, licença e etc.)	
Atributos	PK / FK	Descrição
ID_SUBSTITUICAO	PK	código sequencial gerado automaticamente pelo sistema.
ID_TITULAR	FK	chave estrangeira do titular, aponta para DP_PESSOA
ID_LOTA_TITULAR	FK	refere-se a lotação do titular, no momento da substituição
ID_SUBSTITUTO	FK	chave estrangeira do substituto, refere-se à tabela DP_PESSOA
ID_LOTA_SUBSTITUTO	FK	refere-se a lotação do substituto, no momento da substituição
DT_INI_SUBST		data inicial da substituição
DT_FIM_SUBST		data final da substituição. pode ser nula caso não haja previsão do fim da substituição.
DT_INI_REG		Data de inicio do registro. Refere-se à data de inclusão do registro.
DT_FIM_REG		Data de fim do registro. Refere-se à data que foi informada o fim da substituição.
ID_REG_INI	FK	Identifica o primeiro registro de substituição de este titular.
Imported keys (para quem esta tabela aponta)		
		
Exported keys (quais tabelas apontam para esta tabela)		

Módulo: CP Corporativo

Relação das Tabelas
CP_APPLICACAO_FERIADO
CP_CONFIGURACAO
CP_FERIADO
CP_GRUPO
CP_IDENTIDADE
CP_LOCALIDADE
CP_MARCA
CP_MARCADOR
CP_MODELO
CP_OCORRENCIA_FERIADO
CP_ORGAO
CP_ORGAO_USUARIO
CP_PAPEL
CP_PERSONALIZACAO
CP_SEDE
CP_SERVICO
CP_SITUACAO_CONFIGURACAO
CP_TIPO_CONFIGURACAO

CP TIPO GRUPO
CP TIPO IDENTIDADE
CP TIPO LOTACAO
CP TIPO MARCA
CP TIPO MARCADOR
CP TIPO PAPEL
CP TIPO PESSOA
CP TIPO SERVICO
CP TIPO SERVICO SITUACAO
CP UF

Tabela	Descrição / Observação	
CP_APPLICACAO_FERIADO		Cadastro de feriados por aplicação, dependente do órgão / localidade e lotação *** (atualmente vazia)
Atributos	PK / FK	Descrição
ID_APPLICACAO	PK	
ID_ORGAO_USU	FK	
ID_LOTACAO	FK	
ID_LOCALIDADE	FK	
ID_OCORRENCIA_FERIADO	FK	
FERIADO		
Imported keys (para quem esta tabela aponta)		
CP_APPLICACAO_FERIADO ID_APPLICACAO NUMBER	DP_LOTACAO ID_LOTACAO NUMBER(6)	
	CP_ORGAO_USUARIO ID_ORGAO_USU NUMBER(10)	
	CP_OCORRENCIA_FERIADO ID_OCORRENCIA NUMBER	
	CP_LOCALIDADE ID_LOCALIDADE NUMBER	
Powered by yFiles		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
CP_COMPLEXO		
Atributos	PK / FK	Descrição
ID_COMPLEXO	PK	
NOME_COMPLEXO		
ID_LOCALIDADE	FK	
Imported keys (para quem esta tabela aponta)		
CP_COMPLEXO	CP_LOCALIDADE ID_LOCALIDADE NUMBER	
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
CP_CONFIGURACAO		
Atributos	PK / FK	Descrição
ID_CONFIGURACAO	PK	
DT_INI_VIG_CONFIGURACAO		

DT_FIM_VIG_CONFIGURACAO		
HIS_DT_INI		
ID_ORGAO_USU	FK	
ID_LOTACAO	FK	
ID_CARGO	FK	
ID_FUNCAO_CONFIANCA	FK	
ID_PESSOA	FK	
ID_SIT_CONFIGURACAO	FK	
ID_TP_CONFIGURACAO	FK	
ID_SERVICO	FK	
ID_GRUPO	FK	
NM_EMAIL		
DESC_FORMULA		
ID_TP_LOTACAO	FK	
ID_IDENTIDADE	FK	
HIS_IDC_INI		
HIS_IDC_FIM		
HIS_DT_FIM		
HIS_ID_INI		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
CP_FERIADO	TABELA DE FERIADOS REGISTRA FERIADOS FIXOS E MÓVEIS PARA CADA LOCALIDADE *** (atualmente vazia)	
Atributos	PK / FK	Descrição
ID_FERIADO	PK	Código de identificação do feriado
DSC_FERIADO		Descrição do feriado
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
CP_GRUPO	São grupos de acesso ao sistema, tais como: Administrador do Sistema de Benefícios, Administrador do Siga Treinamento etc.	
Atributos	PK / FK	Descrição
ID_GRUPO	PK	
ID_TP_GRUPO		
ID_ORGAO_USU	FK	
ID_GRUPO_PAII	FK	
SIGLA_GRUPO		
DESC_GRUPO		
HIS_ID_INI		
HIS_DT_INI		
HIS_IDC_INI		

HIS_DT_FIM		
HIS_IDC_FIM		
HIS_ATIVO		
Imported keys (para quem esta tabela aponta)		
	<pre> graph LR CP_GRUPO[CP_GRUPO ID_GRUPO NUMBER] --> CP_TIPO_GRUPO[CP_TIPO_GRUPO ID_TP_GRUPO NUMBER] CP_GRUPO --> CP_IDENTIDADE[CP_IDENTIDADE ID_IDENTIDADE NUMBER] CP_GRUPO --> CP_GRUPO </pre>	
Exported keys (quais tabelas apontam para esta tabela)		
	<pre> graph LR CP_IDENTIDADE[CP_IDENTIDADE ID_IDENTIDADE NUMBER] --> CP_GRUPO[CP_GRUPO ID_GRUPO NUMBER] </pre>	

Tabela	Descrição / Observação	
CP_IDENTIDADE		É um cadastro com matrículas e senhas dos funcionários Obs.: LOGIN_IDENTIDADE é a matrícula do funcionário RJ13284, ...Treinamento etc.
Atributos	PK / FK	Descrição
ID_IDENTIDADE	PK	
ID_TP_IDENTIDADE	FK	
ID_PESSOA	FK	
DATA_CRIACAO_IDENTIDADE		
DATA_EXPIRACAO_IDENTIDADE		
DATA_CANCELAMENTO_IDENTIDADE		
ID_ORGAO_USU	FK	
LOGIN_IDENTIDADE		
SENHA_IDENTIDADE		
SENHA_IDENTIDADE_CRIPTO		
SENHA_IDENTIDADE_CRIPTO_SINC		
HIS_ID_INI		
HIS_DT_INI		
HIS_IDC_INI		
HIS_DT_FIM		
HIS_IDC_FIM		
HIS_ATIVO		
Imported keys (para quem esta tabela aponta)		
	<pre> graph LR CP_IDENTIDADE[CP_IDENTIDADE ID_IDENTIDADE NUMBER] --> DP_PESSOA[DP_PESSOA ID_PESSOA NUMBER(10)] CP_IDENTIDADE --> CP_TIPO_IDENTIDADE[CP_TIPO_IDENTIDADE ID_TP_IDENTIDADE NUMBER] CP_IDENTIDADE --> CP_ORGAO_USUARIO[CP_ORGAO_USUARIO ID_ORGAO_USU NUMBER(10)] </pre>	

Exported keys (quais tabelas apontam para esta tabela)

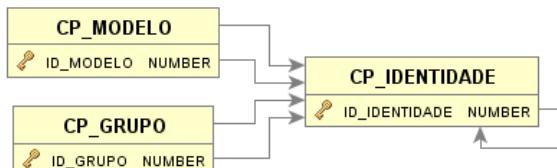
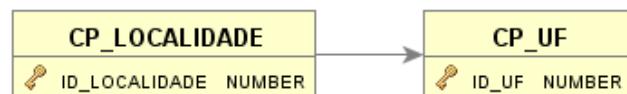


Tabela	Descrição / Observação	
CP_LOCALIDADE	Cadastro de localidades, preferencialmente municípios: Niterói, Angra do Reis ...	
Atributos	PK / FK	Descrição
ID_LOCALIDADE	PK	Código sequencia de gerado pelo sistema
NM_LOCALIDADE		Nome do município (localidade) do órgão, ou pessoa.
ID_UF	FK	Chave estrangeira que identifica a unidade federal a que a localidade pertence.

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

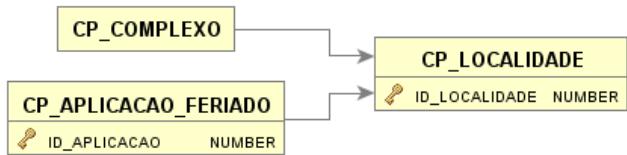
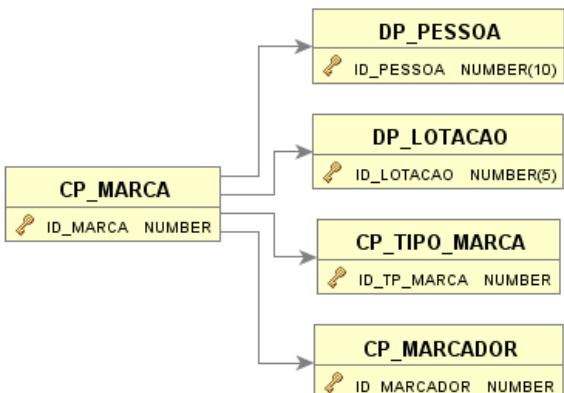


Tabela	Descrição / Observação	
CP MARCA		
Atributos	PK / FK	Descrição
ID_MARCA	PK	
DT_INI_MARCA		
DT_FIM_MARCA		
ID_MARCADOR	FK	
ID_PESSOA_INI	FK	
ID_LOTACAO_INI	FK	
ID_MOBIL	FK	
ID_TP_MARCA	FK	

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

Tabela	Descrição / Observação	
CP_MARCADOR	É o status do documento. Alguns exemplos de marcador: Pendente, Cancelado, Em Elaboração, Aguardando Andamento e etc.	
Atributos	PK / FK	Descrição
ID_MARCADOR	PK	
DESCR_MARCADOR		
ID_TP_MARCADOR	FK	

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)



Tabela	Descrição / Observação	
CP_MODELO	Versionamento do modelo geral de macros FM. Para cada atualização do modelo geral é gravado uma linha no BD. Obs.: O atributo CONTEUDO_BLOB_MOD contém o fonte FM do modelo geral das macros	
Atributos	PK / FK	Descrição
ID_MODELO	PK	
ID_ORGAO_USU	FK	
CONTEUDO_BLOB_MOD		
HIS_ID_INI		
HIS_DT_INI		
HIS_IDC_INI		
HIS_DT_FIM		
HIS_IDC_FIM		
HIS_ATIVO		

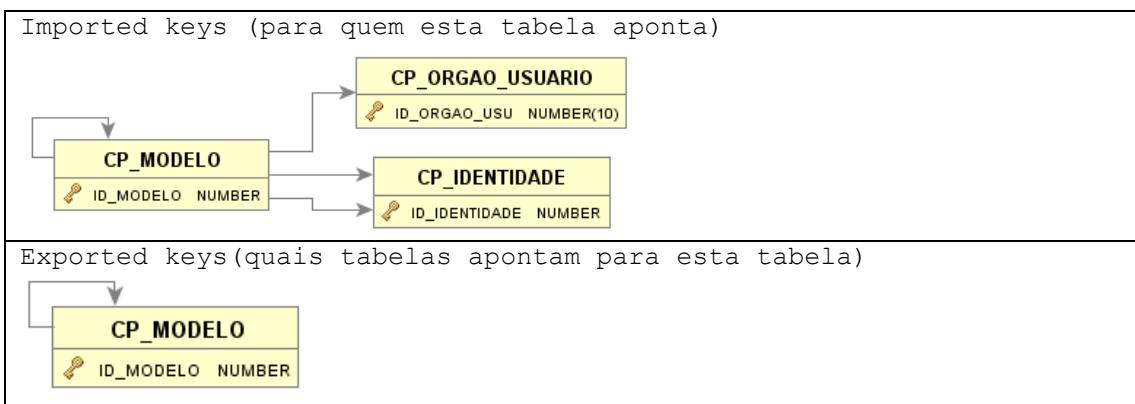


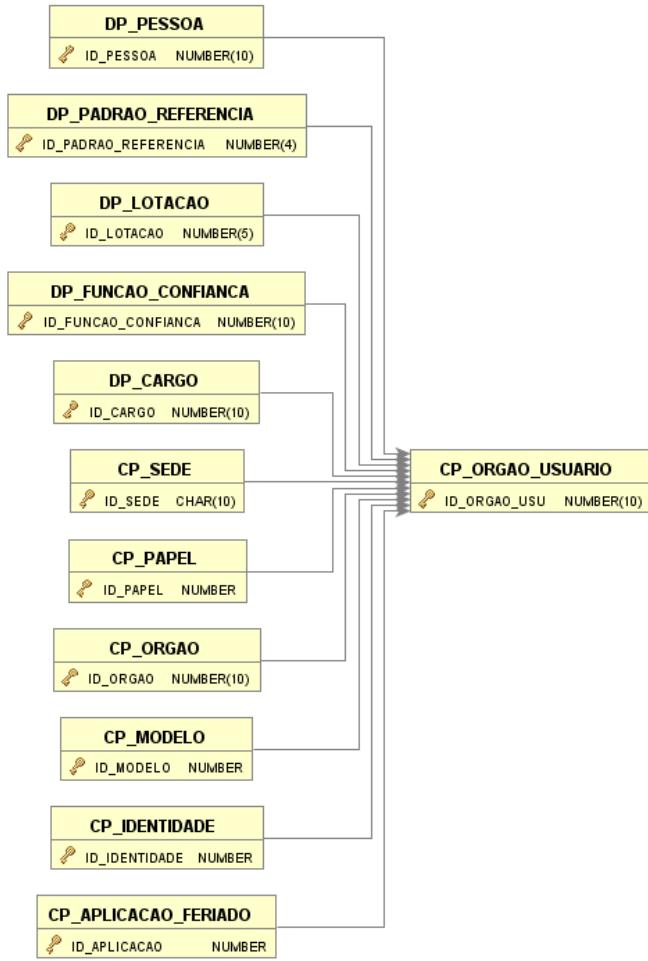
Tabela	Descrição / Observação	
CP_OCURRENCIA_FERIADO	São os feriados propriamente ditos. *** (atualmente vazia)	
Atributos	PK / FK	Descrição
ID_OCURRENCIA	PK	
DT_INI_FERIADO		
DT_FIM_FERIADO		
ID_FERIADO	FK	
Imported keys (para quem esta tabela aponta)		
CP_OCURRENCIA_FERIADO	<pre> graph LR CP_OCURRENCIA_FERIADO[CP_OCURRENCIA_FERIADO] --> CP_FERIADO[CP_FERIADO] CP_OCURRENCIA_FERIADO CP_FERIADO </pre> <p>CP_FERIADO ID_FERIADO NUMBER</p>	
Exported keys (quais tabelas apontam para esta tabela)	<pre> graph LR CP_APPLICACAO_FERIADO[CP_APPLICACAO_FERIADO] --> CP_OCURRENCIA_FERIADO[CP_OCURRENCIA_FERIADO] CP_APPLICACAO_FERIADO CP_OCURRENCIA_FERIADO </pre> <p>CP_APPLICACAO_FERIADO ID_APPLICACAO NUMBER</p> <p>CP_OCURRENCIA_FERIADO ID_OCURRENCIA NUMBER</p>	

Tabela	Descrição / Observação	
CP_ORGAO	Tabela de Órgãos (de destino). Por exemplo: Presidência da República, CNJ e etc. Quando enviamos um documento, este campo representa o órgão de destino, indicando que o documento foi enviado para lá, porém não terá retorno pelo sistema.	
Atributos	PK / FK	Descrição
ID_ORGAO	PK	Numero sequencial identificador do registro. É interno e gerado automaticamente pelo sistema
NM_ORGAO		Nome do Órgão Externo, campo livre.
CGC_ORGAO		CGC do Órgão Externo.
RAZAO_SOCIAL_ORGAO		Razão social do Órgão Externo.
END_ORGAO		Endereço do Órgão Externo, campo livre.
BAIRRO_ORGAO		
MUNICIPIO_ORGAO		
CEP_ORGAO		
DSC_TIPO_ORGAO		Descrição do Tipo de Órgão Externo.
NOME_RESPONSABEL_ORGAO		

EMAIL_RESPONSABEL_ORGAO		
NOME_CONTATO_ORGAO		
EMAIL_CONTATO_ORGAO		
TEL_CONTATO_ORGAO		
SIGLA_ORGAO		
UF_ORGAO		
ID_ORGAO_USU	FK	Órgão responsável pelo cadastramento e manutenção dos dados do registro. Só este órgão terá acesso a este registro.
FG_ATIVO		
HIS_ID_INI		
HIS_IDE		
HIS_DT_INI		
HIS_DT_FIM		
HIS_ATIVO		
Imported keys (para quem esta tabela aponta)		
		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
CP_ORGAO_USUARIO	Tabela que armazena os órgãos usuários capazes de criar, cadastrar, ou alterar documentos no sistema./ Observação: MUNICIPIO_ORGAO_USU e UF_ORGAO_USU fazem parte dos atributos, portanto não utiliza chave estrangeira.	
Atributos	PK / FK	Descrição
ID_ORGAO_USU	PK	
NM_ORGAO_USU		
CGC_ORGAO_USU		
RAZAO_SOCIAL_ORGAO_USU		
END_ORGAO_USU		
BAIRRO_ORGAO_USU		
MUNICIPIO_ORGAO_USU		
CEP_ORGAO_USU		
NM_RESP_ORGAO_USU		
TEL_ORGAO_USU		
SIGLA_ORGAO_USU		
UF_ORGAO_USU		
COD_ORGAO_USU		Campo livre destinado ao cadastramento dos códigos de órgãos de forma a haver correspondência com sistemas do TRF e JFES
ACRONIMO_ORGAO_USU		
Imported keys (para quem esta tabela aponta)		

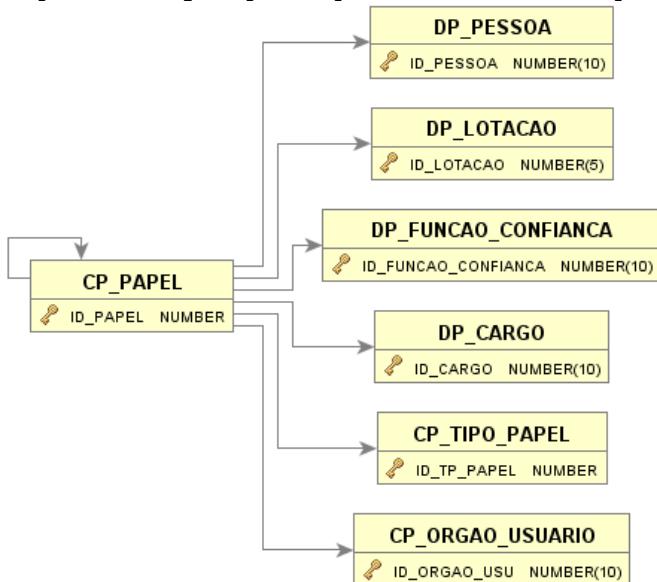
Exported keys (quais tabelas apontam para esta tabela)



iles

Tabela	Descrição / Observação	
CP_PAPEL		
Atributos	PK / FK	Descrição
ID_PAPEL	PK	
ID_TP_PAPEL	FK	
ID_PESSOA	FK	
ID_LOTACAO	FK	
ID_FUNCAO_CONFIANCA	FK	
ID_CARGO	FK	
ID_ORGAO_USU	FK	
HIS_ID_INI		
HIS_IDE		
HIS_DT_INI		
HIS_DT_FIM		
HIS_ATIVO		

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

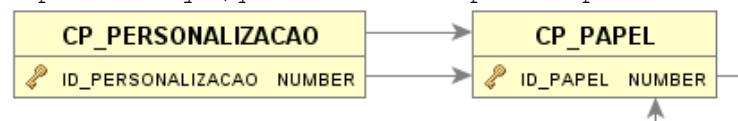
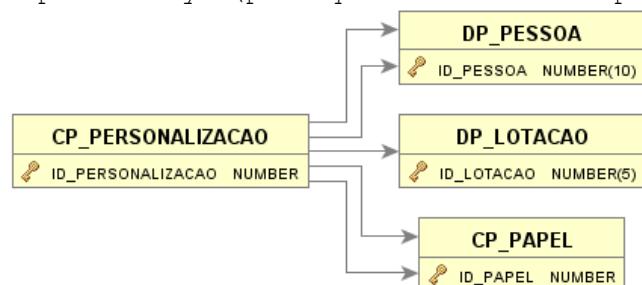


Tabela	Descrição / Observação	
CP_PERSONALIZACAO		
Atributos	PK / FK	Descrição
ID_PESSOA	PK	
ID_PAPEL_ATIVO	FK	
ID_SUBSTITUINDO_PESSOA	FK	
ID_SUBSTITUINDO_LOTACAO	FK	
ID_SUBSTITUINDO_PAPEL	FK	
NM_SIMULANDO_USUARIO		
ID_PERSONALIZACAO	FK	

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

Tabela	Descrição / Observação	
CP_SEDE	Um Órgão pode ter várias sedes. Por exemplo: SJRJ, possui a sede R. Branco e A. Barroso *** (atualmente vazia)	
Atributos	PK / FK	Descrição
ID_SEDE	PK	
NM_SEDE		
DSC_SEDE		
ID_ORGAO_USU	FK	
Imported keys (para quem esta tabela aponta)		
CP_SEDE		CP_ORGAO_USUARIO
ID_SEDE CHAR(10)		ID_ORGAO_USU NUMBER(10)
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
CP_SERVICO	Relação dos serviços dos sistemas. Os Sistemas possuem uma série de serviços: Assinatura digital, Editar modelos, Relação de formulários e etc. Uma regra de permissão associará serviços a pessoas.	
Atributos	PK / FK	Descrição
ID_SERVICO	PK	
SIGLA_SERVICO		
DESC_SERVICO		
ID_SERVICO_PAIS	FK	
ID_TP_SERVICO	FK	
Imported keys (para quem esta tabela aponta)		
CP_SERVICO		CP_TIPO_SERVICO
ID_SERVICO NUMBER		ID_TP_SERVICO NUMBER
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
CP_SITUACAO_CONFIGURACAO	Descreve o status da configuração. Exemplos: Ignorar configuração anterior, Pode, Não Pode, Obrigatório, Opcional e etc.	
Atributos	PK / FK	Descrição
ID_SIT_CONFIGURACAO	PK	
DSC_SIT_CONFIGURACAO		
RESTRITIVIDADE_SIT_CONF		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
CP_TIPO_SERVICO_SITUACAO		CP_SITUACAO_CONFIGURACAO
ID_TP_CONFIGURACAO NUMBER(19)		ID_SIT_CONFIGURACAO NUMBER(19)

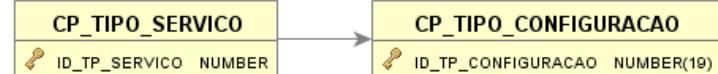
Tabela	Descrição / Observação	
CP_TIPO_CONFIGURACAO	Descreve os tipos de configuração de serviços. Exemplos: Utilizar Serviço de Outra Lotação, Despachável, Destinatário, Utilizar XStandard. Utilizar PD4ML, Movimentar, Criar e etc.	
Atributos	PK / FK	Descrição
ID_TP_CONFIGURACAO	PK	
DSC_TP_CONFIGURACAO		
ID_SIT_CONFIGURACAO	FK	
Imported keys (para quem esta tabela aponta)		
		
Exported keys (quais tabelas apontam para esta tabela)		
		

Tabela	Descrição / Observação	
CP_TIPO_GRUPO	Descreve o tipo de grupo. Exemplos: Perfil de Acesso, Grupo de Email, Perfil de Acesso do JEE	
Atributos	PK / FK	Descrição
ID_TP_GRUPO	PK	
DESC_TP_GRUPO		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
		

Tabela	Descrição / Observação	
CP_TIPO_IDENTIDADE	Descreve os tipos de identidade para assinar os documentos. Exemplos: Login e Senha, Certidão Digital ICP-Brasil	
Atributos	PK / FK	Descrição
ID_TP_IDENTIDADE	PK	
DESC_TP_IDENTIDADE		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
		

Tabela	Descrição / Observação	
CP_TIPO_LOTACAO	Descreve os tipos de lotação. Exemplos: Unidade da Administração, Unidade Judicial, Vara Federal, Vara Federal Criminal	
Atributos	PK / FK	Descrição
ID_TP_LOTACAO	PK	
SIGLA_TP_LOTACAO		

DESC_TP_LOTACAO		
ID_TP_LOTACAO_PAIS	FK	
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
DP_LOTACAO		CP_TIPO_LOTACAO
ID_LOTACAO NUMBER(5)		ID_TP_LOTACAO NUMBER

Tabela	Descrição / Observação	
CP_TIPO_MARCA	Exemplos: SIGA-EX, SIGA-SR	
Atributos	PK / FK	Descrição
ID_TP_MARCA	PK	
DESCR_TP_MARCA		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
CP_MARCA		CP_TIPO_MARCA
ID_MARCA NUMBER		ID_TP_MARCA NUMBER

Tabela	Descrição / Observação	
CP_TIPO_MARCADOR	Exemplos: Sistema Geral, Lotação e sublotações, Lotação, Pessoa	
Atributos	PK / FK	Descrição
ID_TP_MARCADOR	PK	
DESCR_TIPO_MARCADOR		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
CP_MARCADOR		CP_TIPO_MARCADOR
ID_MARCADOR NUMBER		ID_TP_MARCADOR NUMBER

Tabela	Descrição / Observação	
CP_TIPO_PAPEL	Exemplos: Principal, Funcional	
Atributos	PK / FK	Descrição
ID_TP_PAPEL	PK	
DESCR_TP_PAPEL		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
CP_PAPEL		CP_TIPO_PAPEL
ID_PAPEL NUMBER		ID_TP_PAPEL NUMBER

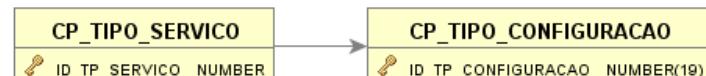
Tabela	Descrição / Observação	
CP_TIPO_PESSOA	Descreve os tipos/categorias de pessoas. Exemplos: Magistrado, Servidor, Estagiário, Terceirizado	
Atributos	PK / FK	Descrição
ID_TP_PESSOA	PK	
DESCR_TP_PESSOA		
Imported keys (para quem esta tabela aponta)		

Exported keys (quais tabelas apontam para esta tabela)



Tabela	Descrição / Observação	
CP_TIPO_SERVICO	Exemplos: Diretório, Sistema	
Atributos	PK / FK	Descrição
ID_TP_SERVICO	PK	
DESC_TP_SERVICO		
ID_SIT_CONFIGURACAO	FK	

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

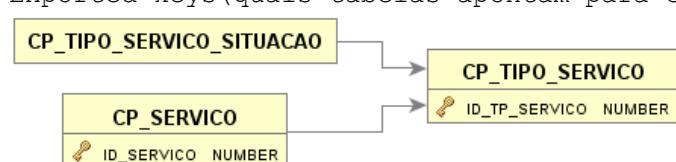
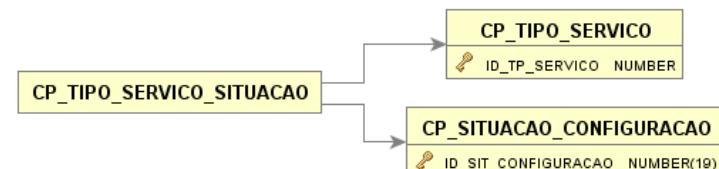


Tabela	Descrição / Observação	
CP_TIPO_SERVICO_SITUACAO	Exemplos: Diretório, Sistema	
Atributos	PK / FK	Descrição
ID_TP_SERVICO	PK	
ID_SIT_CONFIGURACAO	FK	

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

Tabela	Descrição / Observação	
CP_UF	Unidades da federação (estados): RJ, SP e etc.	
Atributos	PK / FK	Descrição
ID_UF	PK	
NM_UF		

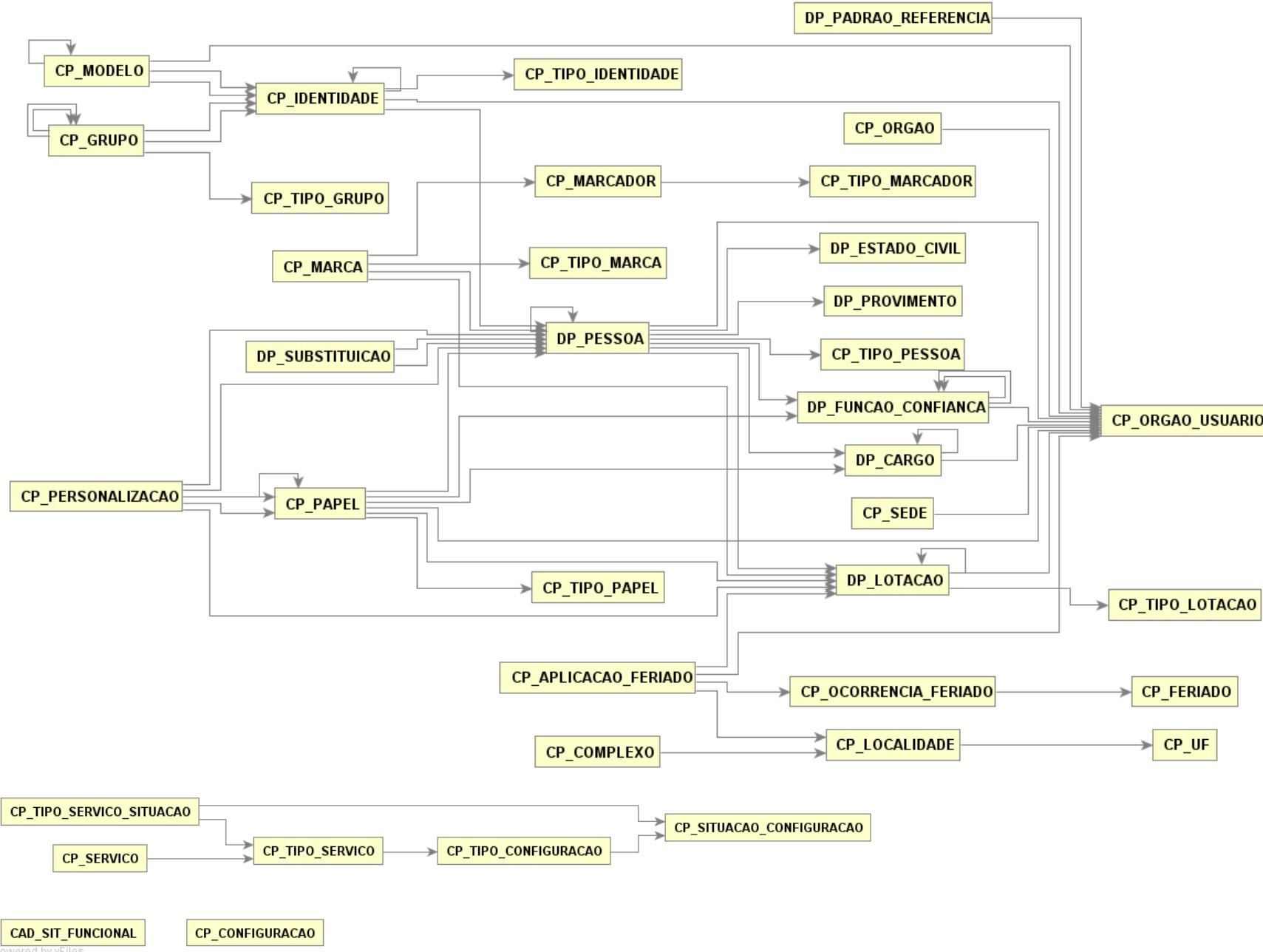
Imported keys (para quem esta tabela aponta)

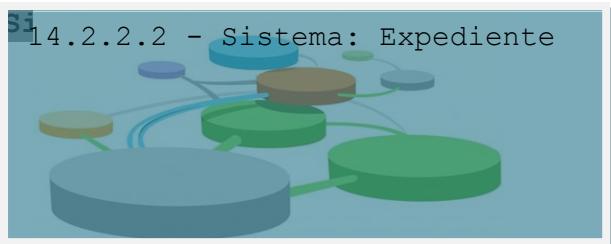
Exported keys (quais tabelas apontam para esta tabela)

```

graph LR
    CP_LOCALIDADE[CP_LOCALIDADE  
ID_LOCALIDADE NUMBER] --> CP_UF[CP_UF  
ID_UF NUMBER]
  
```

Modelo ER Sistema Corporativo - Nódulos Dp e Cp





O sistema expediente deu origem ao atual SIGA-DOC, desta forma, as tabelas do antigo sistema expediente foram reaproveitadas e modificadas para o novo sistema.

Relação das Tabelas
EX_BOLETIM_DOC
EX_CLASSIFICAÇÃO
EX_COMPETENCIA
EX_CONFIGURACAO
EX_DOCUMENTO
EX_EMAIL_NOTIFICAÇÃO
EX_ESTADO_DOC
EX_ESTADO_TP_MOV
EX_FORMA_DOCUMENTO
EX_MOBIL
EX_MODELO
EX_MODELO_TP_DOC_PUBLICACAO
EX_MOVIMENTACAO
EX_NIVEL_ACESSO
EX_NUMERACAO
EX_PAPEL
EX_PREENCHIMENTO
EX_SITUACAO_CONFIGURACAO
EX_TEMPORALIDADE
EX_TIPO_DESPACHO
EX_TIPO_DESTINACAO
EX_TIPO_DOCUMENTO
EX_TIPO_FORMA_DOCUMENTO
EX_TIPO_MOBIL
EX_TIPO_MOVIMENTACAO
EX_TP_DOC_PUBLICACAO
EX_TP_FORMA_DOC
EX_TP_MOV_ESTADO
EX_VIA

Tabela	Descrição / Observação	
EX_BOLETIM_DOC		
Atributos	PK / FK	Descrição
ID_BOLETIM_DOC	PK	
ID_DOC	FK	
ID_BOLETIM	FK	

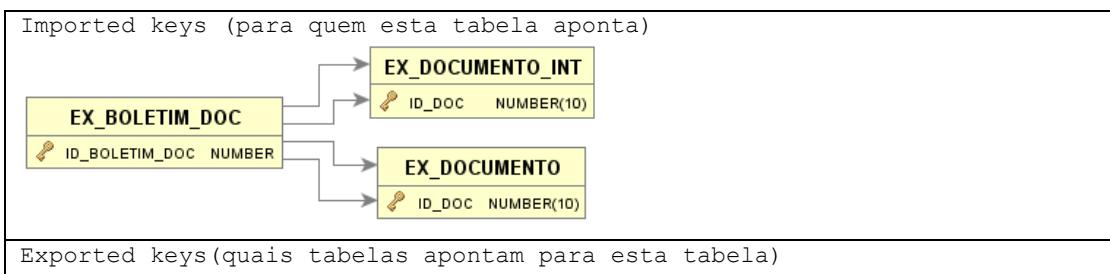


Tabela	Descrição / Observação	
EX_CLASSIFICACAO		
Atributos	PK / FK	Descrição
ID_CLASSIFICACAO	PK	Identificador interno do assunto no sistema é um número sequencial gerado automaticamente.
COD_ASSUNTO_PRINCIPAL		Código de classificação do assunto principal.
COD_ASSUNTO_SECUNDARIO		Código de classificação do código secundário.
COD_CLASSE		Código da Classe funcional do documento. São dez classes principais, de acordo com o foco administrativo.
COD_SUBCLASSE		Código da subclasse do documento ou processo.
COD_ATIVIDADE		Código da atividade funcional do documento ou processo.
DESCR_CLASSIFICACAO		Descrição do escopo do assunto a que se refere o documento ou processo.
FACILITADOR_CLASS		Texto explicando as regras pré-estabelecidas para a criação deste tipo de documento.
ID_REG_INI	FK	
DTINI_REG		
DTFIM_REG		
COD_ASSUNTO		

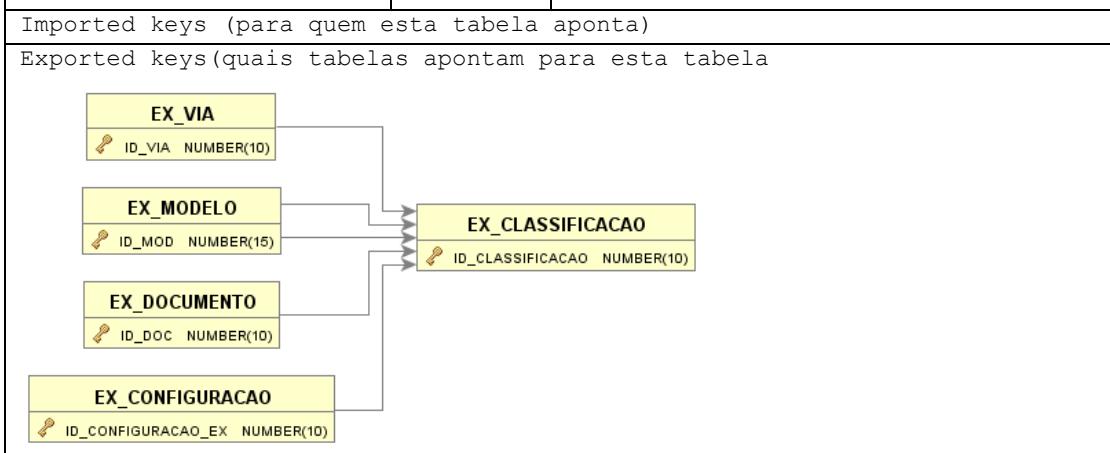


Tabela	Descrição / Observação	
--------	------------------------	--

EX_COMPETENCIA		
Atributos	PK / FK	Descrição
FG_COMPETENCIA		
ID_PESSOA	PK	
ID_CARGO	FK	
ID_LOTACAO	FK	
DT_INI_VIG_COMPETENCIA		Data de início da competência tem a função de preservar o histórico.
DT_FIM_VIG_COMPETENCIA		Data de fim da competência tem a função de preservar o histórico.
ID_COMPETENCIA	FK	Número sequencial gerado automaticamente que identifica internamente a competência.
ID_FUNCAO_CONFIANCA		
ID_FORMA_DOC	FK	
Imported keys (para quem esta tabela aponta)		
EX_COMPETENCIA ID_COMPETENCIA NUMBER(10)	DP_PESSOA	
	DP_LOTACAO	
	DP_FUNCAO_CONFIANCA	
	DP_CARGO	
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
EX_CONFIGURACAO		
Atributos	PK / FK	Descrição
ID_CONFIGURACAO_EX	PK	
ID_TP_MOV	FK	
ID_TP_DOC	FK	
ID_TP_FORMA_DOC	FK	
ID_FORMA_DOC	FK	Data de início da competência tem a função de preservar o histórico.
ID_MOD	FK	Data de fim da competência tem a função de preservar o histórico.
ID_CLASSIFICACAO	FK	Número sequencial gerado automaticamente que identifica internamente a competência.
ID_VIA	FK	
ID_NIVEL_ACESSO	FK	
ID_PAPEL	FK	

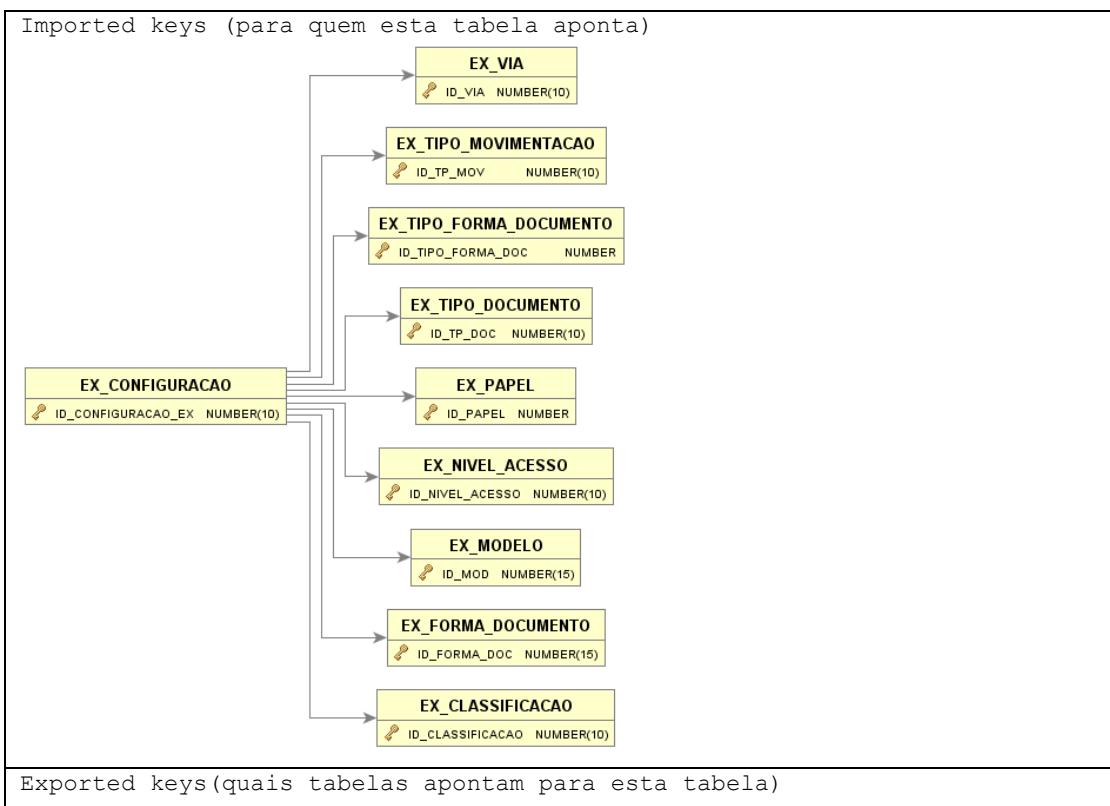
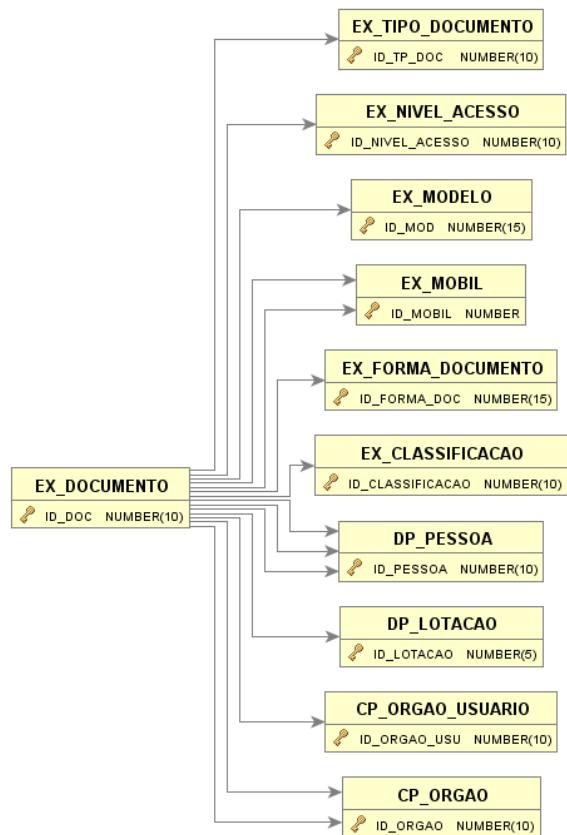


Tabela	Descrição / Observação	
EX_DOCUMENTO		
Atributos	PK / FK	Descrição
ID_DOC	PF	Numero sequencial de identificação interna do documento. É gerado automaticamente.
NUM_EXPEDIENTE		Número de identificação do expediente para a consulta pelos usuários (sequencial zerado na mudança do ano)
ANO_EMISSAO		Ano de emissão do documento.
ID_TP_DOC	FK	
ID_CADASTRANTE	FK	
ID_LOTA_CADASTRANTE	FK	
ID_SUBSCRITOR	FK	
ID_LOTA_SUBSCRITOR	FK	
DESCR_DOCUMENTO		
DT_DOC		
DT_REG_DOC		
NM_SUBSCRITOR_EXT		
NUM_EXT_DOC		
CONTEUDO_BLOB_DOC		
NM_ARQ_DOC		
CONTEUDO_TP_DOC		
ID_DESTINATARIO	FK	
ID_LOTA_DESTINATARIO	FK	
NM_DESTINATARIO		

DT_FECHAMENTO		
ASSINATURA_BLOB_DOC		
ID_MOD	FK	
ID_ORGAO_USU	FK	
ID_CLASSIFICACAO	FK	
ID_FORMA_DOC	FK	
FG_PESSOAL		
ID_ORGAO_DESTINATARIO	FK	
ID_ORGAO	FK	
OBS_ORGAO_DOC		
NM_ORGAO_DESTINATARIO		
FG_SIGILOSO		
NM_FUNCAO_SUBSCRITOR		
FG_ELETRONICO		
NUM_ANTIGO_DOC		
ID_LOTA_TITULAR	FK	
ID_TITULAR	FK	
NUM_AUX_DOC		
DSC_CLASS_DOC		
ID_NIVEL_ACESSO	FK	
ID_DOC_PAIS	FK	
NUM_VIA_DOC_PAIS		
ID_DOC_ANTERIOR	FK	
ID_MOB_PAIS	FK	
NUM_SEQUENCIA		
NUM_PAGINAS		
DT_DOC_ORIGINAL		
ID_MOB_AUTUADO	FK	

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

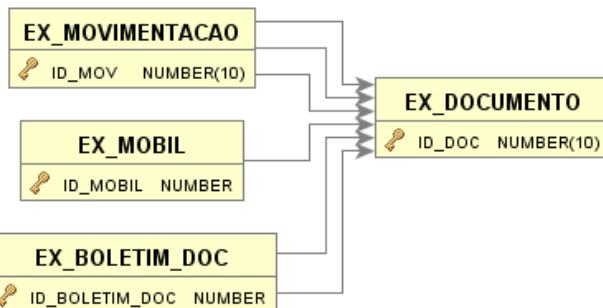


Tabela	Descrição / Observação	
EX_EMAIL_NOTIFICAÇÃO		
Atributos	PK / FK	Descrição
ID_EMAIL_NOTIFICACAO	PK	
ID_LOTACAO	FK	
EMAIL		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
EX_ESTADO_DOC		
Atributos	PK / FK	Descrição
ID_ESTADO_DOC	PK	Número sequencial que identifica o estado do documento internamente no sistema. (Gerado Automaticamente)
DESC_ESTADO_DOC		Descrição do estado do documento.
ORDEM_ESTADO_DOC		
Imported keys (para quem esta tabela aponta)		
Exported keys(quaies tabelas apontam para esta tabela)		
<pre> graph LR A[EX_TP_MOV_ESTADO] --> C[EX_ESTADO_DOC] B[EX_MOVIMENTACAO] --> C D[EX_ESTADO_TP_MOV] --> C E[EX_ESTADO_TP_MOV] --> A </pre>		

Tabela	Descrição / Observação	
EX_ESTADO_TP_MOV		
Atributos	PK / FK	Descrição
ID_ESTADO_DOC	PK	Código de identificação do estado de um documento é chave estrangeira, refere-se a da tela de estados (EX ESTADO DOC)
ID_TP_MOV	FK	Código do tipo de movimentação que o documento pode sofrer, encontrando-se no estado apresentado na mesma dupla.
Imported keys (para quem esta tabela aponta)		
<pre> graph LR A[EX_ESTADO_TP_MOV] --> B[EX_TIPO_MOVIMENTACAO] A --> C[EX_ESTADO_DOC] </pre>		
Exported keys(quaies tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
EX_FORMA_DOCUMENTO		
Atributos	PK / FK	Descrição
ID_FORMA_DOC	PK	Numero que identifica internamente a forma no sistema. Gerado automaticamente.

DESCR_FORMA_DOC		Descrição da forma que um documento pode ser apresentado ou veiculado.
SIGLA_FORMA_DOC		Sigla da forma do documento.
ID_TIPO_FORMA_DOC	FK	

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

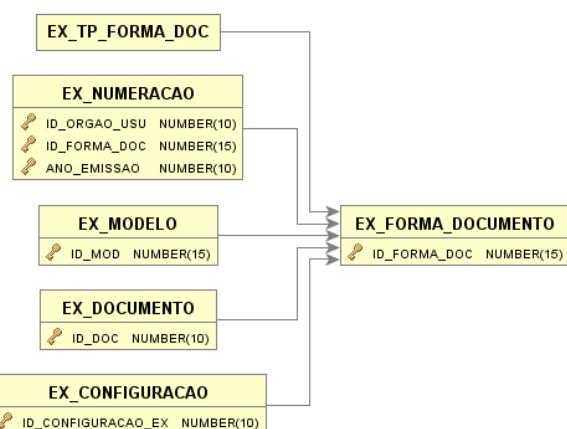
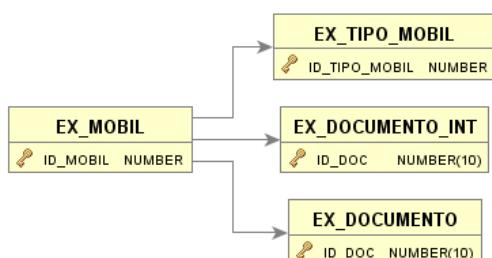


Tabela	Descrição / Observação	
EX_MOBIL		
Atributos	PK / FK	Descrição
ID_MOBIL	PF	
ID_DOC	FK	
ID_TIPO_MOBIL	FK	
NUM_SEQUENCIA		

Imported keys (para quem esta tabela aponta)

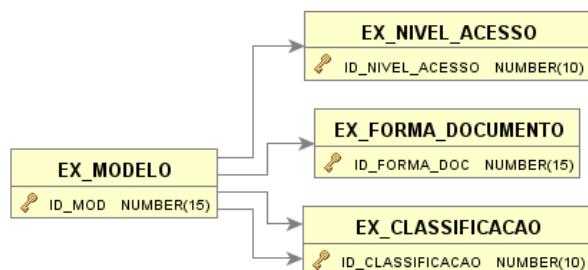


Exported keys (quais tabelas apontam para esta tabela)



Tabela	Descrição / Observação	
EX_MODELO		
Atributos	PK / FK	Descrição
ID_MOD	PF	Numero de identificação interna do modelo no sistema. Gerado
NM_MOD		Nome do modelo é aquele que será apresentado ao usuário.
DESC_MOD		Descrição do modelo apresenta a utilização e características do modelo
CONTEUDO_BLOB_MOD		É o modelo em si, o arquivo que será apresentado ao usuário.
CONTEUDO_TP_BLOB		Descrição do CONTENT TYPE do arquivo armazenado, que é uma string independente do tipo de arquivo declarado na extensão (ex.: *.DOC, *.RTF, etc) utilizado como
NM_ARQ_MOD		Nome do arquivo eletrônico do modelo.
ID_CLASSIFICACAO	FK	
ID_FORMA_DOC	FK	
ID_CLASS_CRIACAO_VIA	FK	
ID_NIVEL_ACESSO	FK	identificador do tipo de sigilo que um modelo pode ser associado

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)

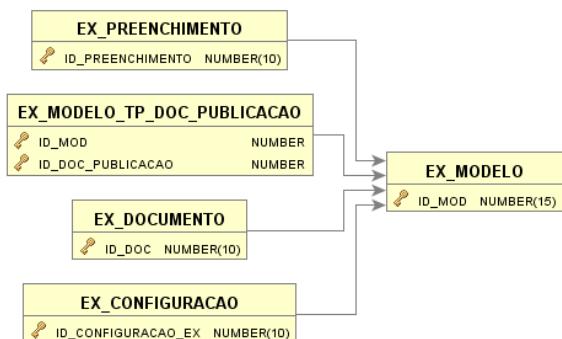
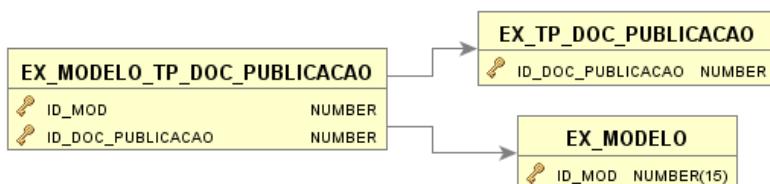


Tabela	Descrição / Observação	
Atributos	PK / FK	Descrição
ID_MOD	PF	
ID_DOC_PUBLICACAO	FK	

Imported keys (para quem esta tabela aponta)

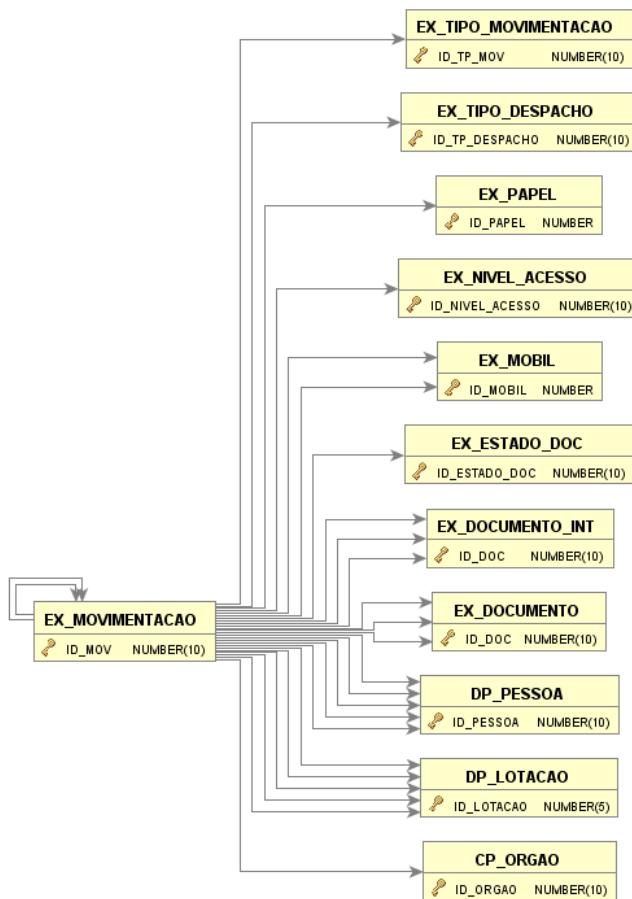


Exported keys (quais tabelas apontam para esta tabela)

Tabela	Descrição / Observação	
Atributos	PK / FK	Descrição
ID_MOV	PF	Código sequencial interno, gerado automaticamente pelo sistema.
ID_DOC	FK	Código identificador do documento a que se refere à movimentação.
ID_DOC_PAIS	FK	Código do documento principal, caso o documento mencionado por ID_DOC seja anexado, ou juntado a outro documento
ID_TP_MOV	FK	
ID_ESTADO_DOC	FK	
ID_TP_DESPACHO	FK	
ID_CADASTRANTE	FK	
ID_LOTA_CADASTRANTE	FK	
ID_SUBSCRITOR	FK	
ID_LOTA_SUBSCRITOR	FK	
DT_MOV		
DT_INI_MOV		

NUM_VIA		Número da via do documento mencionado por ID_DOC a que se refere à movimentação.
CONTEUDO_BLOB_MOV		
ID_MOV_CANCELADORA	FK	Código da movimentação que cancela esta movimentação.
NM_ARQ_MOV		
CONTEUDO_TP_MOV		
DT_FIM_MOV		
ID_LOTA_RESP	FK	
ID_RESP	FK	
DESCR_MOV		
ASSINATURA_BLOB_MOV		
ID_DESTINO_FINAL	FK	
ID_LOTA_DESTINO_FINAL	FK	
NUM_VIA_DOC_PAIS		Número da via do documento principal a que o documento mencionado por ID_DOC foi anexado
ID_DOC_REF	FK	Código do documento que serve de referência para o doc mencionado por ID_DOC
NUM_VIA_DOC_REF		Número da via do documento que serve de referência para o doc mencionado por ID_DOC
OBS_ORGAO_MOV		
ID_ORGAO	FK	
ID_MOV_REF	FK	
ID_LOTA_TITULAR	FK	identifica a lotação que está em substituição
ID_TITULAR	FK	identifica se a movimentação foi efetuada por sub. de função
NM_FUNCAO_SUBSCRITOR		Campo livre onde o usuário descreve uma função (pode ser informal ou temporária) do subscritor
NUM_PROC_ADM		
ID_NIVEL_ACESSO	FK	identificador do sigilo a ser redefinido para um documento ou
DT_DISP_PUBLICACAO		será usada na movimentação de agendamento de publicação de
DT_EFETIVA_PUBLICACAO		
DT_EFETIVA_DISP_PUBLICACAO		
PAG_PUBLICACAO		
NUM_TRF_PUBLICACAO		
CADERNO_PUBLICACAO_DJE		
ID_MOBIL	FK	
ID_MOB_REF	FK	
NUM_PAGINAS		
NUM_PAGINAS_ORI		
ID_PAPEL	FK	

Imported keys (para quem esta tabela aponta)



Exported keys (quais tabelas apontam para esta tabela)



Tabela	Descrição / Observação	
EX_NIVEL_ACESSO		
Atributos	PK / FK	Descrição
ID_NIVEL_ACESSO	PF	
NM_NIVEL_ACESSO		
DSC_NIVEL_ACESSO		
GRAU_NIVEL_ACESSO		
Imported keys (para quem esta tabela aponta)		

Exported keys (quais tabelas apontam para esta tabela)

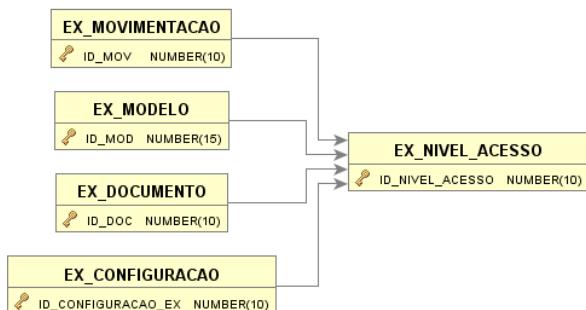


Tabela	Descrição / Observação	
EX_NUMERACAO		
Atributos	PK / FK	Descrição
ID_ORGAO_USU	PF	
ID_FORMA_DOC	FK	
ANO_EMISSAO		
NUM_EXPEDIENTE		
Imported keys (para quem esta tabela aponta)		
<pre> graph LR EX_NUMERACAO[EX_NUMERACAO] --> EX_FORMA_DOCUMENTO[EX_FORMA_DOCUMENTO] EX_NUMERACAO --> CP_ORGAO_USUARIO[CP_ORGAO_USUARIO] </pre> <p>The diagram illustrates the imported keys for the table EX_NUMERACAO. It shows two other tables (EX_FORMA_DOCUMENTO and CP_ORGAO_USUARIO) that have foreign key relationships pointing to the primary key of EX_NUMERACAO.</p>		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
EX_PAPEL		
Atributos	PK / FK	Descrição
ID_PAPEL	PF	
DESC_PAPEL		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
<pre> graph LR EX_MOVIMENTACAO[EX_MOVIMENTACAO] --> EX_PAPEL[EX_PAPEL] EX_CONFIGURACAO[EX_CONFIGURACAO] --> EX_PAPEL </pre> <p>The diagram illustrates the exported keys for the table EX_PAPEL. It shows two other tables (EX_MOVIMENTACAO and EX_CONFIGURACAO) that have foreign key relationships pointing to the primary key of EX_PAPEL.</p>		

Tabela	Descrição / Observação	
EX_PREENCHIMENTO		
Atributos	PK / FK	Descrição

ID_PREENCHIMENTO	PF	Campo sequencial de identificação interna do preenchimento. Número gerado automaticamente pelo sistema
ID_LOTACAO	FK	
ID_MOD	FK	Nome ou descrição do preenchimento, tem a finalidade de facilitar a escolha do preenchimento pelo usuário.
EX_NOME_PREENCHIMENTO		
PREENCHIMENTO_BLOB		conteúdo do preenchimento a ser preservado
Imported keys (para quem esta tabela aponta)		
<pre> graph LR EP[EX_PREENCHIMENTO] --> EM[EX_MODELO] EP --> DL[DP_LOTACAO] EM -.-> IDMOD[PK ID_MOD NUMBER(15)] DL -.-> IDLOTACAO[PK ID_LOTACAO NUMBER(5)] </pre>		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
EX_SITUACAO_CONFIGURACAO		
Atributos	PK / FK	Descrição
ID_SIT_CONFIGURACAO	PF	
DSC_SIT_CONFIGURACAO	FK	
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
EX_TEMPORALIDADE		
Atributos	PK / FK	Descrição
ID_TEMPORALIDADE	PF	
DESC_TEMPORALIDADE		
PERMANENCIA_ARQUIVO		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		

Tabela	Descrição / Observação	
EX_TIPO_DESPACHO		
Atributos	PK / FK	Descrição
ID_TP_DESPACHO	PF	
DESC_TP_DESPACHO		
FG_ATIVO_TP_DESPACHO		
Imported keys (para quem esta tabela aponta)		

Exported keys (quais tabelas apontam para esta tabela)

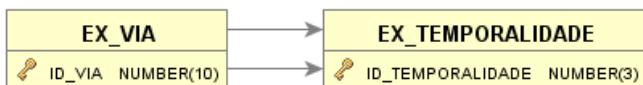


Tabela	Descrição / Observação	
EX_TIPO_DESPACHO		
Atributos	PK / FK	Descrição
ID_TP_DESPACHO	PF	
DESC_TP_DESPACHO		
FG_ATIVO_TP_DESPACHO		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
EX_MOVIMENTACAO	<pre> graph LR EX_MOVIMENTACAO[EX_MOVIMENTACAO] --> EX_TIPO_DESPACHO[EX_TIPO_DESPACHO] EX_MOVIMENTACAO --> EX_TIPO_DESPACHO EX_MOVIMENTACAO["ID_MOV NUMBER(10)"] EX_TIPO_DESPACHO["ID_TP_DESPACHO NUMBER(10)"] </pre>	

Tabela	Descrição / Observação	
EX_TIPO_DESTINACAO		
Atributos	PK / FK	Descrição
ID_TP_DESTINACAO	PF	Numero sequencial de identificação interna. Gerado automaticamente pelo sistema
DESCR_TIPO_DESTINACAO		Descrição do tipo de destinação final da via do documento (ex.: Eliminação, Guarda Permanente)
FACILITADOR_DEST		Texto que esclarece o significado do tipo de destinação. Campo livre.
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)	<pre> graph LR EX_VIA[EX_VIA] --> EX_TIPO_DESTINACAO[EX_TIPO_DESTINACAO] EX_VIA --> EX_TIPO_DESTINACAO EX_VIA["ID_VIA NUMBER(10)"] EX_TIPO_DESTINACAO["ID_TP_DESTINACAO NUMBER(10)"] </pre>	

Tabela	Descrição / Observação	
EX_TIPO_DOCUMENTO		
Atributos	PK / FK	Descrição
ID_TP_DOC	PF	Numero sequencial do tipo de documento, é gerado automaticamente.
DESCR_TIPO_DOCUMENTO		Descrição do tipo de documento.
Imported keys (para quem esta tabela aponta)		

Exported keys (quais tabelas apontam para esta tabela)

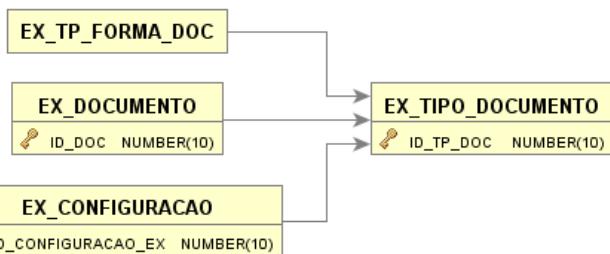


Tabela	Descrição / Observação	
EX_TIPO_FORMA_DOCUMENTO		
Atributos	PK / FK	Descrição
ID_TIPO_FORMA_DOC	PF	
DESC_TIPO_FORMA_DOC		
NUMERACAO_UNICA		

Imported keys (para quem esta tabela aponta)

Exported keys (quais tabelas apontam para esta tabela)

```

    graph LR
        EX_FORMA_DOCUMENTO[EX_FORMA_DOCUMENTO] --> EX_TIPO_FORMA_DOCUMENTO[EX_TIPO_FORMA_DOCUMENTO]
        EX_CONFIGURACAO[EX_CONFIGURACAO] --> EX_TIPO_FORMA_DOCUMENTO
    
```

The diagram shows two tables: EX_FORMA_DOCUMENTO and EX_CONFIGURACAO. Both have relationships pointing to EX_TIPO_FORMA_DOCUMENTO.

Tabela	Descrição / Observação	
EX_TIPO_MOBIL		
Atributos	PK / FK	Descrição
ID_TIPO_MOBIL	PF	
DESC_TIPO_MOBIL		

Imported keys (para quem esta tabela aponta)

Exported keys (quais tabelas apontam para esta tabela)

```

    graph LR
        EX_MOBIL[EX_MOBIL] --> EX_TIPO_MOBIL[EX_TIPO_MOBIL]
    
```

The diagram shows one table: EX_MOBIL, which has a relationship pointing to EX_TIPO_MOBIL.

Tabela	Descrição / Observação	
EX_TIPO_MOVIMENTACAO		
Atributos	PK / FK	Descrição
ID_TP_MOV	PF	
DESCR_TIPO_MOVIMENTACAO		

Imported keys (para quem esta tabela aponta)

Exported keys (quais tabelas apontam para esta tabela)

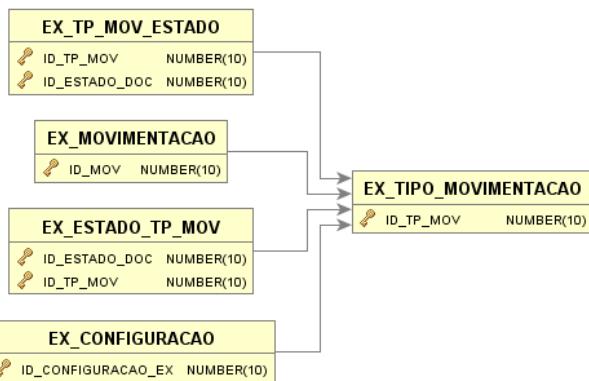


Tabela	Descrição / Observação	
EX_TP_DOC_PUBLICACAO		
Atributos	PK / FK	Descrição
ID_DOC_PUBLICACAO	PF	
NM_DOC_PUBLICACAO		
CARATER		
Imported keys (para quem esta tabela aponta)		
Exported keys (quais tabelas apontam para esta tabela)		
EX_MODELO_TP_DOC_PUBLICACAO	→ EX_TP_DOC_PUBLICACAO	
ID_MOD		
ID_DOC_PUBLICACAO		

Tabela	Descrição / Observação	
EX_TP_FORMA_DOC		
Atributos	PK / FK	Descrição
ID_FORMA_DOC	PF	
ID_TP_DOC		
Imported keys (para quem esta tabela aponta)		
EX_TP_FORMA_DOC	→ EX_TIPO_DOCUMENTO → EX_FORMA_DOCUMENTO	
ID_TP_DOC		
ID_FORMA_DOC		
Exported keys (quais tabelas apontam para esta tabela)		

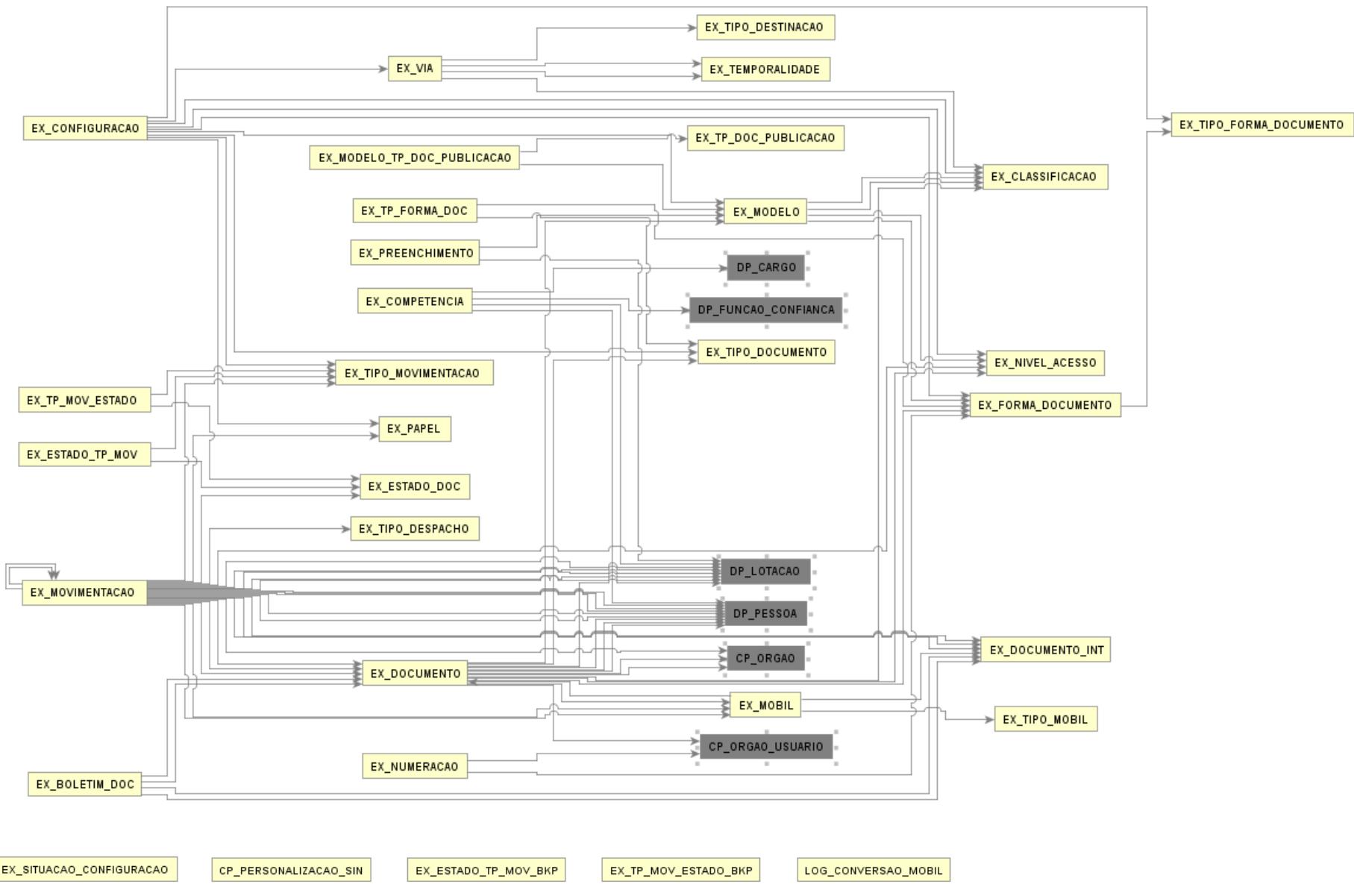
Tabela	Descrição / Observação	
EX_TP_MOV_ESTADO		
Atributos	PK / FK	Descrição
ID_FORMA_DOC	PF	

ID_TP_DOC	FK	
Imported keys (para quem esta tabela aponta)		
EX_TP_MOV_ESTADO		EX_TIPO_MOVIMENTACAO ID_TP_MOV NUMBER(10)
ID_TP_MOV NUMBER(10) ID_ESTADO_DOC NUMBER(10)		EX_ESTADO_DOC ID_ESTADO_DOC NUMBER(10)

Exported keys (quais tabelas apontam para esta tabela)

Tabela	Descrição / Observação	
EX_VIA		
Atributos	PK / FK	Descrição
ID_VIA	PF	
ID_CLASSIFICACAO	FK	
ID_TP_DESTINACAO	FK	
COD_VIA		
ID_TMP_CORRENTE	FK	
ID_TMP_INTERMEDIARIO	FK	
OBS		
FG_MAIOR		
ID_DESTINACAO_FINAL	FK	
ID_REG_INI	FK	
DT_INI_REG		
DT_FIM_REG		
Imported keys (para quem esta tabela aponta)		
EX_VIA		EX_TIPO_DESTINACAO ID_TP_DESTINACAO NUMBER(10)
ID_VIA NUMBER(10)		EX_TEMPORALIDADE ID_TEMPORALIDADE NUMBER(3)
		EX_CLASSIFICACAO ID_CLASSIFICACAO NUMBER(10)
Exported keys (quais tabelas apontam para esta tabela)		
EX_CONFIGURACAO		EX_VIA ID_VIA NUMBER(10)
ID_CONFIGURACAO_EX NUMBER(10)		

Modelo ER Sistema Expediente (Ex) integrado ao DP e CP



14.3 - CLASSES

14.3.1 - Introdução

Classificação das classes:

Quanto ao tipo (Java): concretas, abstratas e interface.

Quanto a funcionalidade: negócio e implementação.

Como veremos posteriormente, as classes concretas/negócio são aquelas mais próximas das entidades do modelo E-R. As classes abstratas e interfaces estão mais ligadas a implementação/projeto.

Classe Concreta: É uma classe que possui atributos, métodos construtores e outros e pode ser instanciada, ou seja, permite a criação de objetos a partir dela. Classes concretas podem ser herdadas por outras classes.

Classe Abstrata: Possui a mesma estrutura de uma classe concreta com a diferença que tem um modificador abstract em sua definição. Não podem ser instanciadas, ou seja, não se obtém objetos através delas. Classes abstratas podem ser herdadas por outras classes abstratas ou concretas possibilitando o polimorfismo. Uma classe é dita abstrata porque ela não está pronta e portanto não pode ser realizada. Uma classe abstrata pode ter 0..N métodos abstratos e 0..N métodos concretos. Não é obrigatório ter pelo menos um método abstrato. Qual é o objetivo de uma classe abstrata? Servir de MODELO.

Uma classe abstrata pura é aquela em que todos os seus métodos são abstratos. Desta forma, qual seria a diferença entre a classe abstrata pura e uma interface? A diferença é que a classe abstrata permite implementação (comportamento padrão / default) e as classes que a herdam não são OBRIGADAS a implementar todos os métodos dela.

Interface: São utilizadas para definição de um modelo através da assinatura de métodos que DEVERÃO ser implementados pelas classes que a herdarem. O uso de interfaces possibilita o polimorfismo. Não existe implementação nas interfaces e estas também não podem ser instanciadas. No fundo, interface não é uma classe, e sim, uma entidade.

A interface tem o objetivo de se definir um MODELO de forma que você possa injetar a implementação que você deseja usar. Elas existem para modelar o domínio de um problema sem implementação de código, mas sim definição, com o objetivo de tornar o sistema flexível, extensível e de fácil manutenção.

Exemplo:

Você precisa especificar um padrão de projeto mas quem vai implementar esse padrão é outra pessoa conforme o contexto da aplicação. Assim você vai definir todos os métodos das interfaces de modo a atender suas responsabilidades no padrão de projeto.

Quando alguém for utilizar essas interfaces, poderá seguir o seu modelo e implementar conforme desejado, herdando por implementação (implements). A interface te diz o que fazer para resolver o problema, mas como ele será resolvido será definido nas classes que a herdarem, sejam concretas ou abstratas.

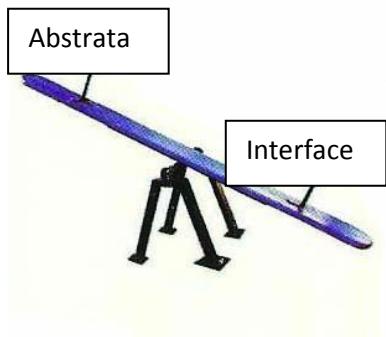
Abaixo um quadro comparativo para tornar mais fácil a compreensão entre as diferenças e similaridades entre Classes Abstratas e Interfaces.

Característica	Interface	Classe Abstrata
Herança múltipla	Uma classe pode implementar diversas interfaces	Uma classe pode herdar somente uma classe
Implementação Padrão	Uma interface não pode conter qualquer tipo de código, muito menos código padrão.	Uma classe abstrata pode fornecer código completo, código padrão ou ter apenas a declaração de seu esqueleto para ser posteriormente sobreescrita.
Constantes	Supõe somente constantes do tipo estática.	Pode conter constantes estáticas e de instância.
Componentes de	Uma implementação de uma interface	Uma classe de terceiros precisa ser

terceiros	pode ser incluída a qualquer classe de terceiros.	reescrita para estender somente a partir da classe abstrata.
Homogeneidade	Se todas as diversas implementações compartilham a assinatura do método então a interface funciona melhor.	Se as várias implementações são todas do tipo e compartilham um comportamento e status comum , então a classe abstrata funciona melhor.
Manutenção	Se o código do seu cliente conversa somente em termos de uma interface, você pode facilmente alterar a implementação concreta usando um método factory.	Idêntico
Velocidade	Lento, requer trabalho extra para encontrar o método correspondente na classe atual.	Rápido
Clareza	Todas as declarações de constantes em uma interface são presumidamente públicas ou estáticas.	Você pode por código compartilhado em uma classe abstrata. Você pode usar código para computar o valor inicial de suas constantes e variáveis de instância ou estáticas.
Funcionalidades Adicionais	Se você incluir um novo método em uma interface você precisa ajustar todas as implementações da interface.	Se você incluir um novo método em uma classe abstrata você tem a opção de fornecer uma implementação padrão para ele.

Ou seja, existe tradeoff entre as duas:

An abstract class has an advantage that you can attach some default behavior to it (e.g. an implementation). The downside is that any given object can only subclass one class in Java, so using an abstract class exclusively for typing is severely limiting your users.



On the flip side, classes can implement multiple interfaces, so you're giving the implementor more options. But - interfaces have no behavior attached to them (e.g. there's no default implementation). This means each person who implements the interface needs to implement the whole thing.

For complex functionality, it's common to use both - first, an interface that defines a component's functionality, and then a base class that implements this interface in a default manner.

Sobre a documentação das classes

Devido a importância dos Modelos (Diagramas) de Classe para o desenvolvedor, foi desenvolvido um estudo com intuito de analisar ferramentas que façam engenharia reversa nos projetos, pacotes e classes, isto porque é praticamente impossível manter uma documentação de 15 projetos, 96 pacotes e aproximadamente 1000 classes, ATUALIZADA, manualmente. Os modelos (Cp e Dp) que serão vistos a seguir servem apenas de entendimento para quem está se familiarizando com o SIGA-DOC e para firmar o conceito de padrão de projeto do SIGA-DOC que se utiliza de classes abstratas e concretas para uma mesma entidade (como por exemplo a entidade DpPessoa), pois são sistemas de apoio, com poucas classes, puramente de negócio. Por esta razão, não nos atreveríamos a documentar / desenhar o sistema Ex (SIGA-DOC ppd) e suas dependências de forma manual, pois a mesma se tornaria obsoleta várias vezes até o seu "térmico".

O estudo apontou para duas ferramentas o ObjectAid para documentar / desenhar o Diagrama de Classes (no nível do pacote, por exemplo) e o UML Doclet para documentar os pacotes, classes, propriedades e métodos de forma detalhada, visto que este último é um plug-in do JavaDoc, ferramenta já conhecida dos desenvolvedores. Para obter mais detalhes sobre estas ferramentas consultar o capítulo 15, Anexo, item 14.

Para consultar a documentação extra, completa, acessar
<file:///K:/ADMINISTRACAO/STI/CSIS/PUBLICA/DocumentacaoSigadoc/index.html>

Sobre o javaDoc

É importante que os sistemas sejam documentados com o JavaDoc, e para tanto, que os desenvolvedores se conscientizem da importância desta documentação. A documentação default do JavaDoc já é muito boa, e caso o desenvolvedor insira documentação adicional, ao nível da classe e dos métodos, aí sim teremos uma documentação excepcional e extremamente útil.

Anotação (Tag)	Uso	Aplica-se a
@author John Smith	Describes an author.	Class, Interface, Enum
@version version	Provides software version entry. Max one per Class or Interface.	Class, Interface, Enum
@since since-text	Describes when this functionality has first existed.	Class, Interface, Enum, Field, Method
@see reference	Provides a link to other element of documentation.	Class, Interface, Enum, Field, Method
@param name description	Describes a method parameter.	Method
@return description	Describes the return value.	Method
@exception classname description @throws classname description	Describes an exception that may be thrown from this method.	Method
@deprecated description	Describes an outdated method.	Method
{@inheritDoc}	Copies the description from the overridden method.	Overriding Method
{@link reference}	Link to other symbol.	Class, Interface, Enum, Field, Method
{@value #STATIC_FIELD}	Return the value of a static field.	Static Field

Exemplo de documentação com o javadoc:

A estrutura básica para documentar em javadoc é colocar a documentação dentro dos seguintes marcadores `/** ... */`.

No nível da classe:

```
/**  
 * @author Firstname Lastname <address @ example.com>  
 * @version 1.6 (current version number of program)  
 * @since 2010-03-31 (the version of the package this class was first added to)  
 */  
public class Test {  
    // class body  
}
```

No nível da variável:

```
/**  
 * Description of the variable here.  
 */  
private int debug = 0;
```

No nível do método:

```
/**  
 * Short one line description. (1)  
 *  
 * Longer description. If there were any, it would be [2]  
 * here.  
 * <p>  
 * And even more explanations to follow in consecutive  
 * paragraphs separated by HTML paragraph breaks.  
 *
```

```
* @param variable Description text text text.          (3)
* @return Description text text text.
*/
public int methodName (...) {
    // method body with a return statement
}
```

Obs: como a documentação do javadoc é em HTML, pode-se introduzir tags HTML dentro da documentação javadoc.

14.3.2 – Descrição das classes

CONCRETAS

Classes	Descrição
<code>public class DpPessoa extends AbstractDpPessoa implements Serializable, Seletionavel, Historico, Sincronizavel</code>	Pessoa: principal classe com os atributos das pessoas / funcionários (servidores e magistrados)
<code>public class DpLotacao extends AbstractDpLotacao implements Serializable, Seletionavel, Historico, Sincronizavel</code>	Lotação: STI, SESIE ...
<code>public class DpCargo extends AbstractDpCargo implements Serializable, Seletionavel, Sincronizavel</code>	Cargo: Analista, técnico, auxiliar ...
<code>public class DpFuncaoConfianca extends AbstractDpFuncaoConfianca implements Serializable, Seletionavel, Sincronizavel</code>	Função de confiança: indica as FCs e CCs das pessoas
<code>public class DpSubstituicao extends AbstractDpSubstituicao implements Serializable</code>	Substituição: indica quem é o substituto do servidor / chefe.
<code>public class CpTipoPessoa extends AbstractCpTipoPessoa</code>	Tipo de pessoa: classifica a pessoa em: Servidores, magistrados, estagiários e terceirizados na SJRJ
<code>public class CpOrgaoUsuario extends AbstractCpOrgaoUsuario implements Serializable, Seletionavel, Assemelhavel</code>	Indica onde a pessoa trabalha: TRF2, SJRJ ...
<code>public class CpTipoLotacao extends AbstractCpTipoLotacao</code>	Tipo de Lotação: indica se é administrativa, judiciária, vara ...
<code>public class ExDocumento extends AbstractExDocumento implements Serializable</code>	

ABSTRATAS

```

public abstract class AbstractDpPessoa extends DpResponsavel implements Serializable
public abstract class AbstractDpCargo implements Serializable
public abstract class AbstractDpLotacao extends DpResponsavel implements Serializable
public abstract class AbstractDpFuncaoConfianca implements Serializable
public abstract class AbstractDpSubstituicao implements Serializable
public abstract class DpResponsavel
public abstract class AbstractCpTipoPessoa
public abstract class AbstractCpOrgaoUsuario implements Serializable
public class AbstractCpTipoLotacao
public abstract class AbstractExDocumento extends ExArquivo implements Serializable
public abstract class ExArquivo

```

INTERFACES

```

public interface Seletionavel
public interface Historico extends Assemelhavel
public interface Sincronizavel extends Assemelhavel
public interface Assemelhavel

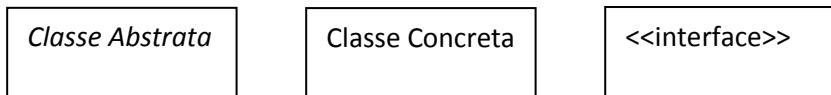
```

Nomenclatura utilizada nos diagramas

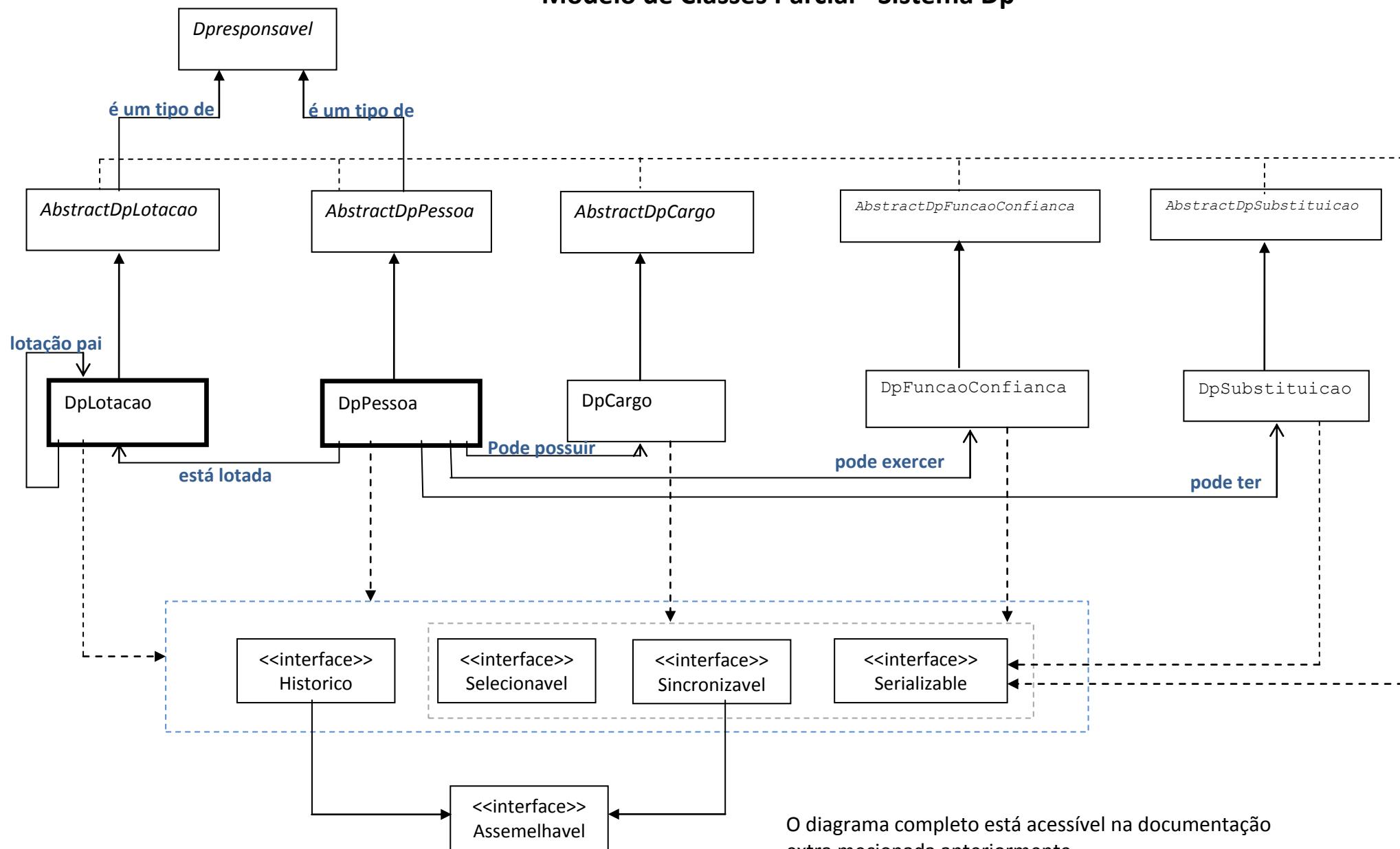
Relacionamentos

↑Implementa ↑Estende ↑Associação

Classes



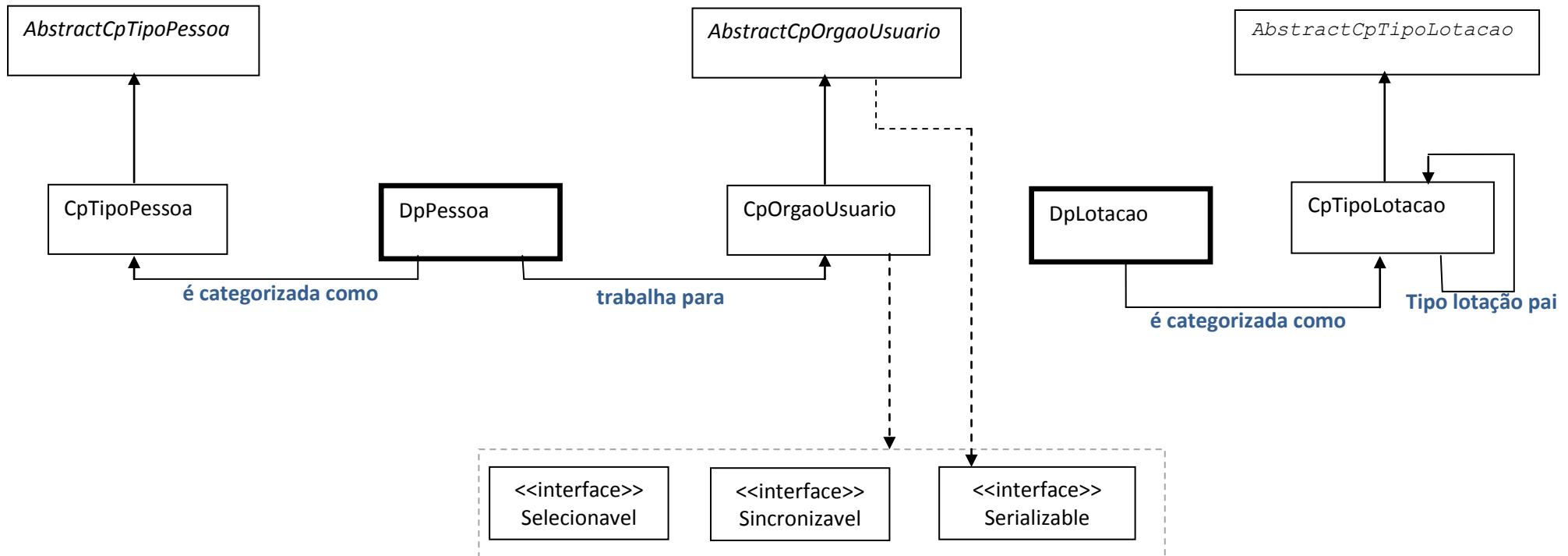
Modelo de Classes Parcial - Sistema Dp



O diagrama completo está acessível na documentação extra mencionada anteriormente

<file:///K:/ADMINISTRACAO/STI/CSIS/PUBLICA/DocumentacaoSigadoc/index.html>

Modelo de Classes Parcial - Sistema Cp



O diagrama completo está acessível na documentação extra mencionada anteriormente

<file:///K:/ADMINISTRACAO/STI/CSIS/PUBLICA/DocumentacaoSigadoc/index.html>

Modelo de Classes - Sistema Ex

O diagrama completo está acessível na documentação extra mencionada anteriormente.

<file:///K:/ADMINISTRACAO/STI/CSIS/PUBLICA/DocumentacaoSigadoc/index.html>

ANEXOS

- Anexo 1 - HTML do SIGA-DOC
- Anexo 2 - Instalando o SQLDeveloper
- Anexo 3 - Utilizando o JBOSS Developer Studio
- Anexo 4 - Disponibilizando uma função TLD para acesso pelo FM (ou JSP)
- Anexo 5 - Filtros no SIGA-DOC
- Anexo 6 - Listener no SIGA-DOC
- Anexo 7 - GIT - Conceitos
- Anexo 8 - GIT no Eclipse/JBDS - EGit
- Anexo 9 - Instalando o plugin do Freemarker no Eclipse
- Anexo 10 - Utilitários baseados no Freemarker
- Anexo 11 - Procedimento para atualizar o repositório Git com fontes JSPs
- Anexo 12 - Instalando o DBVisualizer
- Anexo 13 - Instalando e configurando JBDS
- Anexo 14 - Instalando e configurando o ObjectAid e UML Doclet (ex yDocs)

Anexo 1 - HTML do SIGA-DOC

```
1 <html>
2 <head>
3 <title>SIGA - Novo Documento</title>
4 <meta content="0" http-equiv="Expires">
5 <meta content="no-cache" http-equiv="Pragma">
6 <meta content="no-cache" http-equiv="Cache-Control">
7 <meta content="text/html; charset=UTF-8" http-equiv="content-type">
8
9
10
11 <link media="screen" title="SIGA Estilos" type="text/css" href=
"/sigaex/sigalibs/siga.css" rel="StyleSheet">
12 <script type="text/javascript" language="JavaScript1.1" src=
"/sigaex/sigalibs/ajax.js"></script>
13 <script charset="utf-8" type="text/javascript" language="JavaScript1.1" src=
"/sigaex/sigalibs/static_javascript.js"></script>
14
15 <link type="text/css" rel="stylesheet" href="/sigaex/sigalibs/menu.css">
16
17 <link href="/sigalibs/siga.ico" rel="shortcut icon">
18 </head>
19
20 <body marginwidth="0" marginheight="0" style="background: url(null) fixed no-
repeat;">
onload="" topmargin="0" leftmargin="0">
21
22 <div style="background-color: #9DB1E5; height: 49px; width: 100%;">
23 <div style="float: left">
24 
25 </div>
26 <div style="float: left">
27 <p class="cabecalho-title">
28 <strong>Justiça Federal
29 - Seção Judiciária do Rio de Janeiro
30 </strong>
31 </p>
32 <p class="cabecalho-subtitle">
33 Sistema Integrado de Gestão Administrativa
34 </p>
35 </div>
36 <div style="float: right">
37 
38 </div>
39 </div>
40
41 <div style="position: absolute; top: 0px; right: 0px; background-color: red;
font-weight: bold; padding: 4px; color: white; display: none" id=
"carregando">Carregando...</div>
42 <div style="position: absolute; font-weight: bold; padding: 4px; color:
white; visibility: hidden" id="quadroAviso">-</div>
43 <!-- Mensagens remotas
44 <div id="mensagens-remotas"></div>
45
46 <script type="text/javascript"
src="/sigaex/sigalibs/mensagensremotas.js"></script>
47 <script type="text/javascript" >
48 exibirMensagensRemotas("arquivos/mensagens/mensagens_remotas.xml"
49 , "mensagens-remotas"
-1-
C:\Users\rej\Documents\SIGA\FORMULARIOsigadoc2.txt sexta-feira, 27 de julho de 2012 16:41
50 , "color: yellow; display: inline-block;
font-weight: bolder; font-size:medium; position:
relative; width: 100%; text-align: center;
background-color: black; vertical-align: middle;
border-width: 1px; border-color: #254189 ;
border-style: solid;"
```

```

51 );
52 </script> -->
53 <!-- Fim das mensagens remotas -->
54
55 <!--|**START IMENUS**|imenus0,inline-->
56 <!--[if IE]><style type="text/css">.imcm .imea span{position:absolute;}.imcm
57 .imclear,.imclear{display:none;}.imcm{zoom:1;} .imcm li{curosr:hand;} .imcm
58 ul{zoom:1}.imcm a{zoom:1;}</style><! [endif]-->
59 <!--[if gte IE 7]><style type="text/css">.imcm
60 .imsubc{background-image:url(ie_css_fix);}</style><! [endif]-->
61
62 <!--|**START IMENUS**|imenus1,inline-->
63 <!--[if IE]><style type="text/css">.imcm .imea span{position:absolute;}.imcm
64 .imclear,.imclear{display:none;}.imcm{zoom:1;} .imcm li{curosr:hand;} .imcm
65 ul{zoom:1}.imcm a{zoom:1;}</style><! [endif]-->
66 <!--[if gte IE 7]><style type="text/css">.imcm
67 .imsubc{background-image:url(ie_css_fix);}</style><! [endif]-->
68
69 <!-- <body style="padding: 0px 0px 0px 0px; margin: 0px 0px 0px 0px;"> -->
70
71 <table width="100%" height="100%" cellspacing="0" cellpadding="0" border="0">
72 <tbody><tr>
73 <td valign="top" style="padding-left: 7; padding-top: 7;
74 padding-right: 7; padding-bottom: 7;" colspan="4">
75 <center>
76 <table width="100%" border="0">
77 <tbody><tr>
78 <td>
79 <form method="POST" action=
79 "http://localhost:8080/sigaex/expediente/doc/editar.action" onsubmit=
79 "customOnsubmit_frm(); return true;" name="frm" id="frm" target="">
80 <input type="hidden" value="webwork.token" name="webwork.token.name">
81 <input type="hidden" value="3IITNY6F1GDJUY1LFYMH637UR1JXLDX8" name=
81 "webwork.token">
82 <input type="hidden" name="alterouModelo" id="alterouModelo">
83 <input type="hidden" id="frm_postback" value="1" name="postback">
84 <input type="hidden" id="sigla" value="" name="sigla">
85 <input type="hidden" id="frm_nomePreenchimento" value="" name=
85 "nomePreenchimento">
86 <input type="hidden" value="despachando" name="campos">
87 <input type="hidden" id="frm_despachando" value="false" name=
87 "despachando">
88 <input type="hidden" value="criandoAnexo" name="campos">
89 <input type="hidden" id="frm_criandoAnexo" value="false" name=
89 "criandoAnexo">
90 <table width="100%" class="form">
91 <tbody><tr class="header">
92 <td>Documento:</td>
93 <td colspan="3"><span id="codigoDoc">NOVO</span>
94 <td> de: <span id="dataDoc">13/06/12</span></td>
95
96 -2-
C:\Users\rej\Documents\SIGA\FORMULARIOsigadoc2.txt sexta-feira, 27 de julho de 2012 16:41
97 </tr>
98 <input type="hidden" value="idTpDoc" name="campos">
99 <tr>
100 <td width="10%">Origem:</td>
101 <td width="10%"><select onchange=
101 "javascript:document.getElementById('alterouModelo').value='t
101 rue';sbmt();" style="" id="frm_idTpDoc" name="idTpDoc">
102 <option value="3">Externo</option>
103 <option selected="selected" value="1">Interno Produzido</option>
104 <option value="2">Interno Importado</option>
105 </select>
106 <span style="display: none">Interno Produzido</span>
107 </td>
108 <td width="5%" align="right">Data:</td>
109 <input type="hidden" value="dtDocString" name="campos">
110 <td><input type="text" onblur=
110 "javascript:verifica_data(this, true);" id="frm_dtDocString"

```

```

value="13/06/2012" size="10" name="dtDocString">
103 &nbsp;&nbsp; <input type="hidden" value="nivelAcesso" name="campos">Acesso
<select
id="frm_nivelAcesso" name="nivelAcesso">
104 <option value="6">Público</option>
105 <option selected="selected" value="1">Limitado ao órgão (padrão)</option>
106 <option value="7">Limitado de pessoa para subsecretaria</option>
107 <option value="2">Limitado de subsecretaria para pessoa</option>
108 <option value="3">Limitado entre lotações</option>
109 <option value="5">Limitado entre pessoas</option>
110 </select>
111 <input type="hidden" value="eletronico" name="campos">
112 <input type="radio" value="1" id="eletronicoCheck1"
name="eletronico"><label for="eletronicoCheck1">
Digital</label>
113 <input type="radio" value="2" id="eletronicoCheck2"
name="eletronico"><label for="eletronicoCheck2">
Físico</label>
114 </td>
115 </tr>
116 <input type="hidden" value="" name="desativarDocPai">
117 <tr style="display: none;">
118 <td>Documento Pai:</td>
119 <td colspan="3">
120 <!-- A lista de par -->
121 <input type="hidden" id="frm_mobilPaiSel_id" value="" name="mobilPaiSel.id">
122 <input type="hidden" id="frm_mobilPaiSel_descricao" value="" name=
"mobilPaiSel.descricao">
123 <input type="hidden" id="frm_mobilPaiSel_buscar" value=""
name="mobilPaiSel.buscar">
124 <input type="hidden" id="frm_reqmobilPaiSel" value="" name="reqmobilPaiSel">
125 <input type="hidden" id="alterouSel" value="" name="alterouSel">
126 <input type="text" onkeypress="return handleEnter(this, event)"
onblur="javascript:
ajax_mobilPai();" id="frm_mobilPaiSel_sigla" value="" size="25" name=
"mobilPaiSel.sigla">
127 <input type="button" theme="simple" onclick="javascript:
popitup_mobilPai('');" value
="..." id="mobilPaiSelButton">
128 <span id="mobilPaiSelSpan">
129 </span>
130 </td>
131 </tr>
-3-
C:\Users\rej\Documents\SIGA\FORMULARIOsigadoc2.txt sexta-feira, 27 de julho de 2012 16:41
132 <tr>
133 <td>Subscritor:</td>
134 <input type="hidden" value="subscritorSel.id" name=
"campos">
135 <input type="hidden" value="substituicao" name=
"campos">
136 <td colspan="3">
137 <!-- A lista de par -->
138 <input type="hidden" id="frm_subscritorSel_id" value="10199"
name="subscritorSel.id">
139 <input type="hidden" id="frm_subscritorSel_descricao" value="RUBEN EDWARD
ROSE
JUNIOR" name="subscritorSel.descricao">
140 <input type="hidden" id="frm_subscritorSel_buscar" value="" name=
"subscritorSel.buscar">
141 <input type="hidden" id="frm_reqsubscritorSel" value=""
name="reqsubscritorSel">
142 <input type="hidden" id="alterouSel" value="" name="alterouSel">
143 <input type="text" onkeypress="return handleEnter(this, event)"
onblur="javascript:
ajax_subscritor();" id="frm_subscritorSel_sigla" value="RJ13284" size="25" name=
"subscritorSel.sigla">
144 <input type="button" theme="simple" onclick="javascript:
popitup_subscritor('');"
value="..." id="subscritorSelButton">

```

```

145 <span id="subscritorSelSpan">
146 RUBEN EDWARD ROSE JUNIOR
147 </span>
148 &nbsp;&nbsp;<input type="hidden" value="false" name=" FALSE .substituicao">
149 <input type="checkbox" onclick="javascript:displayTitular(this); " id=
"frm_substituicao" value="true" name="substituicao">
150 Substituto</td>
151 </tr>
152 <tr style="display: none" id="tr_titular">
153 <td>Titular:</td>
154 <input type="hidden" value="titularSel.id" name="campos">
155 <td colspan="3">
156 <!-- A lista de par -->
157 <input type="hidden" id="frm_titularSel_id" value="" name="titularSel.id">
158 <input type="hidden" id="frm_titularSel_descricao" value="" name=
"titularSel.descricao">
159 <input type="hidden" id="frm_titularSel_buscar" value=""
name="titularSel.buscar">
160 <input type="hidden" id="frm_reqtitularSel" value="" name="reqtitularSel">
161 <input type="hidden" id="alterouSel" value="" name="alterouSel">
162 <input type="text" onkeypress="return handleEnter(this, event)" 
onblur="javascript:
ajax_titular(); " id="frm_titularSel_sigla" value="" size="25" name=
"titularSel.sigla">
163 <input type="button" theme="simple" onclick="javascript:
popitup_titular(''); " value=
"..." id="titularSelButton">
164 <span id="titularSelSpan">
165 </span>
166 </td>
167 </tr>
168 <tr>
169 <td>Função;Lotação;Localidade:</td>
170 <td colspan="3"><input type="hidden" value=
"nmFuncaoSubscritor" name="campos"> <input type="text" id=
"frm_nmFuncaoSubscritor" value="" maxlength="128" size="50"
name="nmFuncaoSubscritor">
171 (Opcionalmente informe a função e a lotação na forma:
172 Função;Lotação;Localidade)</td>
-4-
C:\Users\rej\Documents\SIGA\FORMULARIOsigadoc2.txt sexta-feira, 27 de julho de 2012 16:41
174 </tr>
175 <tr>
176 <td>Destinatário:</td>
177 <input type="hidden" value="tipoDestinatario" name="campos">
178 <td colspan="3"><select onchange="javascript:sbmt(); " id=
"frm_tipoDestinatario" name="tipoDestinatario">
179 <option value="1">Matrícula</option>
180 <option selected="selected" value="2">Órgão Integrado</option>
181 <option value="3">Órgão Externo</option>
182 <option value="4">Campo Livre</option>
183 </select>
184 <!-- sbmt('tipoDestinatario') -->
185 <span depende=";tipoDestinatario;" id="destinatario"><!--ajax:destinatario-->
186 <input type="hidden" value=
"lotacaoDestinatarioSel.id" name="campos">
187 <!-- A lista de par -->
188 <input type="hidden" id="frm_lotacaoDestinatarioSel_id" value="1005" name=
"lotacaoDestinatarioSel.id">
189 <input type="hidden" id="frm_lotacaoDestinatarioSel_descricao"
value="Subsecretaria
de Tecnologia da Informação e de Comunicações" name=
"lotacaoDestinatarioSel.descricao">
190 <input type="hidden" id="frm_lotacaoDestinatarioSel_buscar" value="" name=
"lotacaoDestinatarioSel.buscar">
191 <input type="hidden" id="frm_reqlotacaoDestinatarioSel" value="" name=
"reqlotacaoDestinatarioSel">
192 <input type="hidden" id="alterouSel" value="" name="alterouSel">

```

```

193 <input type="text" onkeypress="return handleEnter(this, event)"  

onblur="javascript:  

ajax_lotacaoDestinatario();" id="frm_lotacaoDestinatarioSel_sigla" value="STI"  

size=  

"25" name="lotacaoDestinatarioSel.sigla">  

194 <input type="button" theme="simple" onclick="javascript:  

popup_it_lotacaoDestinatario('');" value="..." id="lotacaoDestinatarioSelButton">  

195 <span id="lotacaoDestinatarioSelSpan">  

196 Subsecretaria de Tecnologia da Informação e de Comunicações  

197 </span>  

198 </span></td>  

199 <!-- idAjax="destinatario" -->  

200 <!--/ajax:destinatario-->  

201 </tr>  

202 <tr>  

203 <td>Tipo:</td>  

204 <td colspan="3"><select onchange=  

"javascript:document.getElementById('alterouModelo').valu  

e='true';sbmt();" style="" id="frm_idFormaDoc" name=  

"idFormaDoc">  

205 <option selected="selected" value="60">Anexo</option>  

206 <option value="80">Assentamento Funcional</option>  

207 <option value="10">Ata</option>  

208 <option value="67">Ato da Corregedoria</option>  

209 <option value="70">Aviso (DIRFO)</option>  

210 <option value="54">Boletim Interno</option>  

211 <option value="11">Carta</option>  

212 <option value="15">Certidão</option>  

213 <option value="61">Certidão de Tempo de Contribuição</option>  

214 <option value="87">Comunicado de Inventário</option>  

215 <option value="9">Contrato</option>  

216 <option value="62">Declaração</option>  

-5-  

C:\Users\rej\Documents\SIGA\FORMULARIOsigadoc2.txt sexta-feira, 27 de julho de 2012 16:41  

217 <option value="8">Despacho</option>  

218 <option value="59">Edital (Teor Administrativo)</option>  

219 <option value="51">Exposição de Motivos</option>  

220 <option value="3">Formulário</option>  

221 <option value="4">Informação</option>  

222 <option value="2">Memorando</option>  

223 <option value="43">Memorando Circular</option>  

224 <option value="50">Memorando Circular (SG)</option>  

225 <option value="97">Memória de Reunião</option>  

226 <option value="1">Ofício</option>  

227 <option value="42">Ofício Circular</option>  

228 <option value="48">Ofício Circular (DIRFO)</option>  

229 <option value="7">Ordem de Serviço</option>  

230 <option value="49">Ordem de Serviço (DIRFO)</option>  

231 <option value="75">Ordem de Serviço da Corregedoria</option>  

232 <option value="14">Parecer</option>  

233 <option value="63">Pauta</option>  

234 <option value="6">Portaria</option>  

235 <option value="74">Portaria (DIRFO-GP)</option>  

236 <option value="73">Portaria (NGP)</option>  

237 <option value="47">Portaria (PGD)</option>  

238 <option value="46">Portaria (SG)</option>  

239 <option value="41">Portaria (SRH)</option>  

240 <option value="68">Portaria da Corregedoria</option>  

241 <option value="88">Portarias da EMARF</option>  

242 <option value="66">Processo Administrativo Disciplinar</option>  

243 <option value="81">Processo de Acompanhamento de Projetos</option>  

244 <option value="76">Processo de Comunicação</option>  

245 <option value="94">Processo de Corregedoria</option>  

246 <option value="96">Processo de Emendas Regimentais</option>  

247 <option value="57">Processo de Execução Orçamentária e Financeira</option>  

248 <option value="55">Processo de Outros Assuntos Administrativos</option>  

249 <option value="95">Processo de Pedido de Providências</option>  

250 <option value="64">Processo de Pessoal</option>  

251 <option value="77">Processo de Petição</option>  

252 <option value="93">Processo de Procedimento Normativo</option>

```

```

253 <option value="65">Processo de Sindicância</option>
254 <option value="79">Processo de Vitaliciamento</option>
255 <option value="58">Processo do Conselho Consultivo</option>
256 <option value="69">Provimento da Corregedoria</option>
257 <option value="98">Relatório</option>
258 <option value="12">Requerimento</option>
259 <option value="92">Resolução</option>
260 <option value="13">Solicitação</option>
261 <option value="78">Solicitação Eletrônica de Contratação</option>
262 <option value="99">TRF - CJEFs Decisão</option>
263 <option value="90">TRF - SCI Nota de Auditoria</option>
264 <option value="86">TRF-PRES Ato da Presidência</option>
265 <option value="82">TRF-PRES Edital da Presidência</option>
266 <option value="83">TRF-PRES Ordem de Serviço da Presidência</option>
267 <option value="85">TRF-PRES Portaria da Presidência</option>
268 <option value="84">TRF-PRES Resolução da Presidência</option>
269 <option value="89">TRF-Proposta e Concessão de Diárias</option>
270 <option value="16">Termo</option>
271 </select>
272 <!-- sbmt('forma') -->
273 <span style="display: none">Anexo</span>
-6-
C:\Users\rei\Documents\SIGA\FORMULARIOSigadoc2.txt sexta-feira, 27 de julho de 2012 16:41
274 </td>
275 </tr>
276 <tr>
277 <td>Modelo:</td>
278 <td colspan="3">
279 <div depende=";forma;" id="modelo"><!--ajax:modelo-->
280 <select
onchange="document.getElementById('alterouModelo').value='true';sbmt();"
style="" id="frm_idMod" name="idMod">
281 <option value="800">(Freemarker) SGP: Auxílio-saúde: Encaminhamento de
recibo(s)
</option>
282 <option value="744">Ambiente de Teste - André Vitorino</option>
283 <option value="507">Anexo</option>
284 <option value="665">Despacho Automático</option>
285 <option value="780">Ruben teste para o manual</option>
286 <option value="742">RubenTeste</option>
287 <option selected="selected" value="880">RubenTesteBrasao</option>
288 <option value="803">Solicitação Eletrônica de Contratação 01</option>
289 <option value="806">Teste JS/Jquery2</option>
290 <option value="807">Teste Priscila</option>
291 <option value="805">TesteJquery</option>
292 <option value="804">teste TJPA</option>
293 <option value="862">testeruben3 final</option>
294 </select>
295 <span style="display: none">RubenTesteBrasao
</span>
296 <!-- sbmt('modelo') -->
297 <!--/ajax:modelo--></div>
298 </td>
299 </tr>
300 <tr>
301 <td>Preenchimento Automático:</td>
302 <input type="hidden" value="preenchimento" name="campos">
303 <td colspan="3"><select onchange=
"javascript:carregaPreench()" id="frm_preenchimento"
name="preenchimento">
304 <option selected="selected" value="0"> [Em branco] </option>
305 </select>
306 &nbsp;
307 <input type="button" disabled="disabled" onclick=
"javascript:alteraPreench()" value="Alterar" name=
"btnAlterar">&nbsp;<input type="button" disabled=
"disabled" onclick="javascript:removePreench()" value=
"Remover" name="btnRemover">&nbsp;<input type="button"
onclick="javascript:adicionaPreench()" name=
"btnAdicionar" value="Adicionar"></td>

```

```

308 </tr>
309 <tr>
310 <td>Classificação:</td>
311 <input type="hidden" value="classificacaoSel.id" name=
"campos">
312 <td colspan="3">
313 <span depende=";forma;modelo;" id="classificacao"><!--ajax:classificacao-->
314 <!-- A lista de par -->
315 <input type="hidden" id="frm_classificacaoSel_id" value="1769" name=
"classificacaoSel.id">
316 <input type="hidden" id="frm_classificacaoSel_descricao" value="ORGANIZAÇÃO E
-7-
C:\Users\rej\Documents\SIGA\FORMULARIOsigadoc2.txt sexta-feira, 27 de julho de 2012 16:41
FUNCIONAMENTO: ADMINISTRAÇÃO JUDICIÁRIA: ORGANIZAÇÃO ADMINISTRATIVA: Documentos
operacionais referentes à modernização administrativa " name=
"classificacaoSel.descricao">
317 <input type="hidden" id="frm_classificacaoSel_buscar" value="" name=
"classificacaoSel.buscar">
318 <input type="hidden" id="frm_reqclassificacaoSel" value=""
name="reqclassificacaoSel">
319 <input type="hidden" id="alterouSel" value="" name="alterouSel">
320 <input type="text" onkeypress="return handleEnter(this, event)"
onblur="javascript:
ajax_classificacao();" id="frm_classificacaoSel_sigla" value="00.01.01.02"
size="25"
name="classificacaoSel.sigla">
321 <input type="button" theme="simple" onclick="javascript:
popitup_classificacao('');"
value="..." id="classificacaoSelButton">
322 <span id="classificacaoSelSpan">
323 ORGANIZAÇÃO E FUNCIONAMENTO: ADMINISTRAÇÃO JUDICIÁRIA: ORGANIZAÇÃO
ADMINISTRATIVA: Documentos operacionais referentes à modernização administrativa
324 </span>
325 <!-- idAjax="classificacao" -->
326 <!--/ajax:classificacao--></span></td>
327 </tr>
328 <tr>
329 <input type="hidden" value="descrDocumento" name="campos">
330 <td>Descrição:</td>
331 <td colspan="3"><textarea id="descrDocumento" rows="3" cols=
"80" name="descrDocumento"></textarea>
332 <br>
333 <span><b>(preencher o campo acima com palavras-chave,
334 sempre usando substantivos, gênero masculino e singular)
</b></span></td>
335 </tr>
336 <tr>
337 <td>Entrevista:</td>
338 <td colspan="3">
339 <span depende=";tipoDestinatario;destinatario;forma;modelo;" id="spanEntrevista">
<!--ajax:spanEntrevista-->
340 <input type="hidden" value="nome" name="vars">
341 <span style=";">Nome:</span>
342 <input type="text" maxlength="20" size="20" value="Ruben Rose" name="nome">
343 <!--/ajax:spanEntrevista--></span></td>
344 </tr>
345 <tr>
346 <td></td>
347 <td colspan="3"><input type="button" value="Ok" name="gravar"
onclick="javascript: gravarDoc();">
348 <input type="button" onclick="javascript:
popitup_documento(false); value="Visualizar o
modelo preenchido" name="ver_doc">
349 
350 </td>
351 </tr>

```

```
352 </tbody></table>
353 </form>
354 </td>
-8-
C:\Users\rej\Documents\SIGA\FORMULARIOsigadoc2.txt sexta-feira, 27 de julho de 2012 16:41
355 </tr>
356 </tbody></table>
357 </center>
358 <!-- tabela do rodapé -->
359 </td>
360 </tr>
361 <tr>
362 <td height="27" colspan="4"></td>
363 </tr>
364 <tr>
365 <td height="22" background="/sigaex/imagens/base2.gif" colspan="4">
366 <table width="100%" cellspacing="0" cellpadding="0" border="0" name="rodapeSuspens">
367 <tbody><tr>
368 <td class="base"><span style="cursor: pointer" onmouseover="javascript:document.getElementById('menuSuspens').style.visibility='visible';" id="spanMenuSuspens">&ampnbsp&ampnbsp
369 RUBEN EDWARD ROSE JUNIOR
370 - Seção de Sistemas Especializados
371 </span></td>
372 <td align="right">&ampnbsp&ampnbsp</td>
373 </tr>
374 </tbody>
375 </table>
376 </td>
377 </tr>
378 </tbody>
379 </table>
380 <div onmouseout="javascript:this.style.visibility='hidden';" onmouseover="javascript:this.style.visibility='visible';" style="position: absolute; border: 1px solid #000000; background: #3B437B; height: auto; z-index: 1; left: 30px; bottom: 21px; visibility: hidden; font-color: #FFFFFF;" id="menuSuspens"> </div>
381 </body>
382 </html>
-9-
```

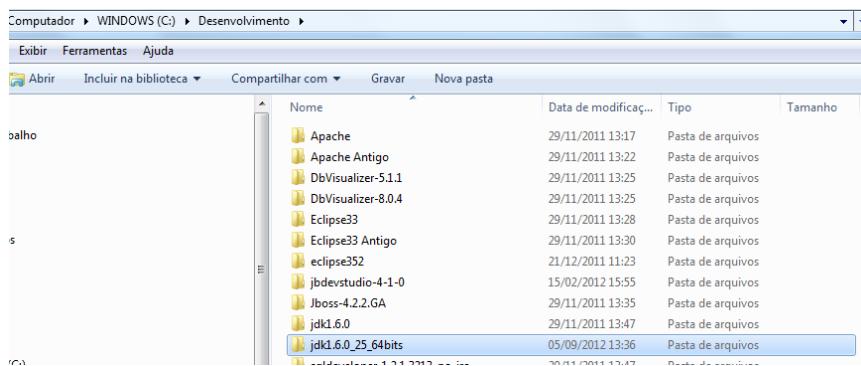
Anexo 2 - Instalando o SQLDeveloper

Download do produto:

<http://www.oracle.com/technetwork/developer-tools/sql-developer/downloads/index.html>

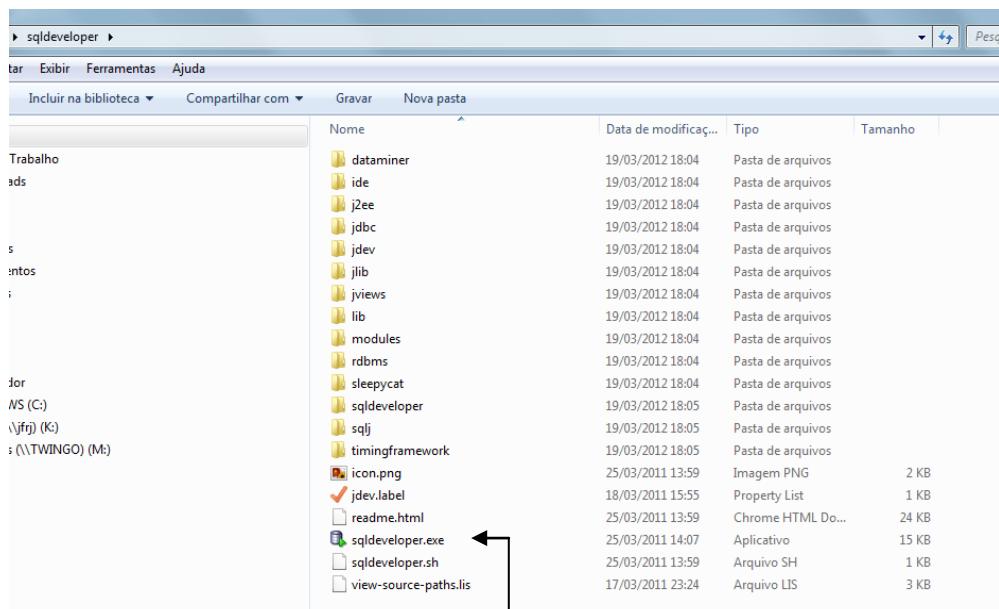
Pré-Requisitos:

jdk1.6.0_25_64bits



O Software:

SQLDeveloper

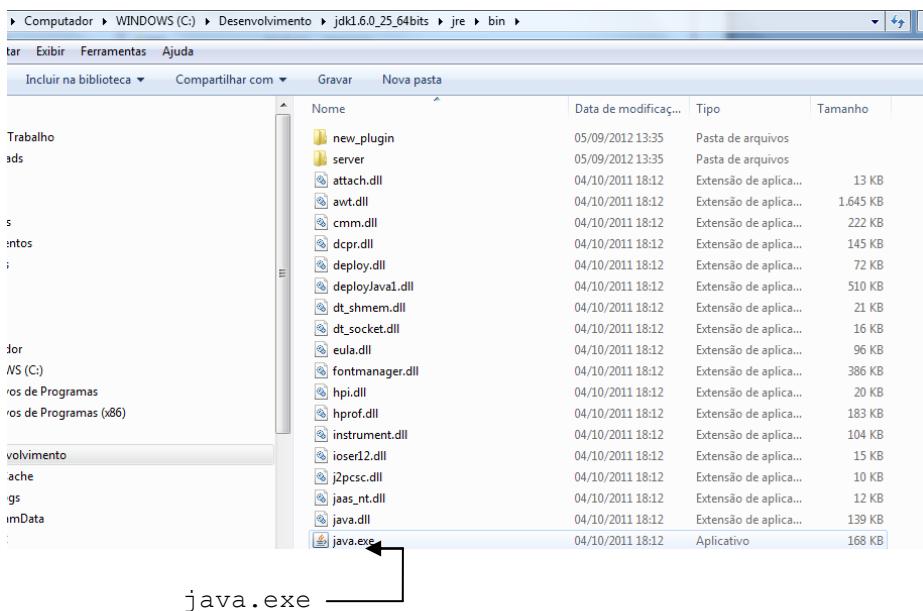


Instalando:

Rodar sqldeveloper.exe

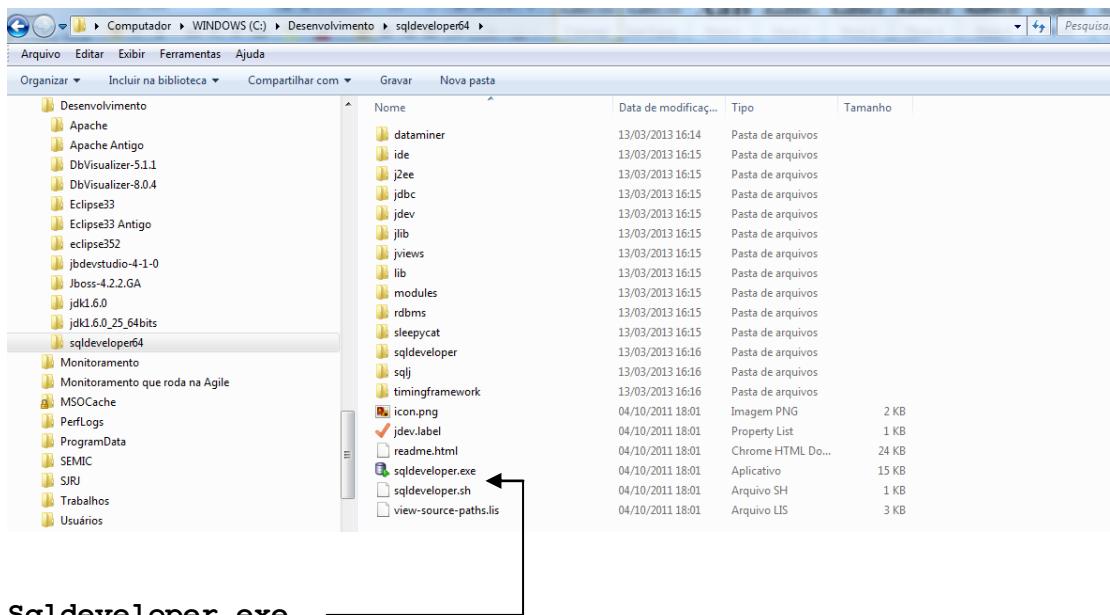
Será pedido o path para o Java.exe, que está no jdk

C:\Desenvolvimento\jdk1.6.0_25_64bits\jre\bin



Executando a primeira vez:

C:\Desenvolvimento\sqldeveloper64

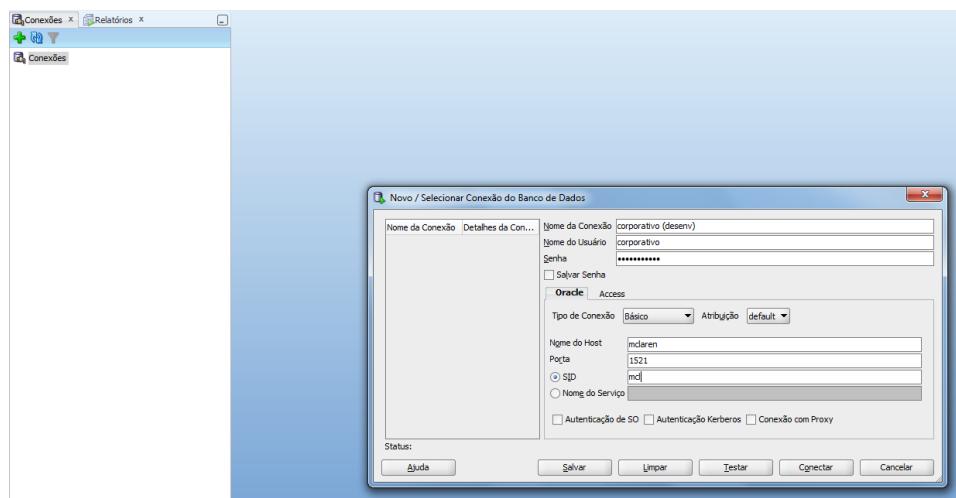


Sqldeveloper.exe

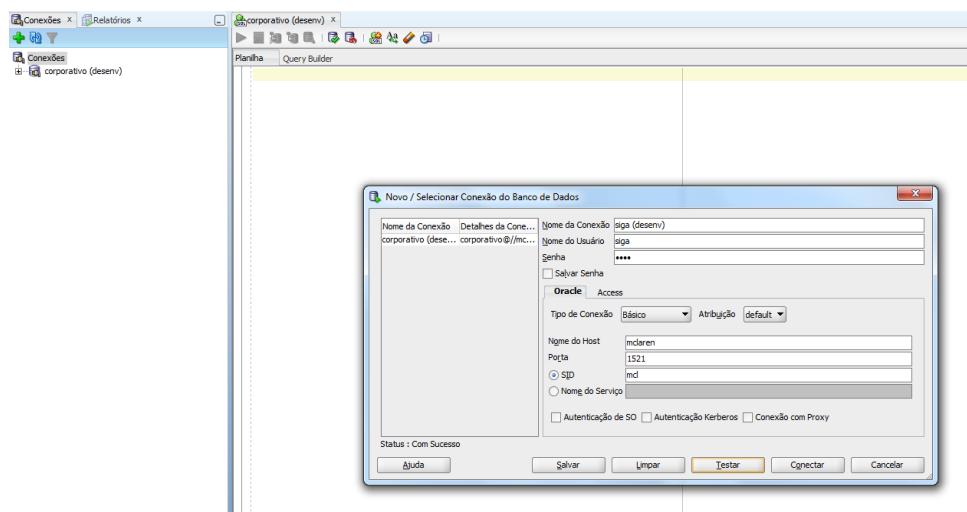
Será solicitado a conexão:

O Siga-doc trabalha com duas bases, uma do corporativo e outra do próprio siga (sigaex).

Configuração da conexão da Base do Corporativo



Configuração da conexão da Base do Siga



Configurações:

Corporativo Desenv	SIGAEX Desenv	SIGAEX Homo
corporativo	siga	siga
corporativo	siga	siga
sid	sid	sid
mclaren	McLaren	mclaren
1521	1521	1521
mcl	mcl	homolo

Agora é só usar:

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Conexões' (Connections) and 'Relatórios' (Reports) panes are visible. The main area displays the results of a query on the 'DP_PESSOA' table from the 'corporativo (desenv)' schema. The table has 27 columns and 27 rows of data, showing various personal information like ID, name, birthdate, and matriculation number.

ID_PESSOA	DATA_INI_PESSOA	DATA_FIM_PESSOA	CPF_PESSOA	NOME_PESSOA	DATA_NASC_PESSOA	MATRICULA	ID_LOTACAO
7	06/03/2012	04/04/12	5301555758	CAROLINA DE BRITO EMILIO E FERNANDES	13/09/85	14166	37
8	06/03/2012	04/04/12	210240733	LUCIANE TEIXEIRA DE OLIVEIRA	04/10/68	11184	53
9	06/03/2012	04/04/12	70883211734	YGOR VLADIMIR SA TOBIAS DA COSTA	17/07/61	13020	64
10	06/03/2012	04/04/12	13154776702	MARCELA COPINI FARO	02/08/89	43457	145
11	06/03/2012	04/04/12	7283985751	SILVIA MONTEIRO BARCELOS	09/09/76	15389	105
12	06/03/2012	04/04/12	18655661791	GERALDO DOS SANTOS LEAL	13/05/33	20087	58
13	06/03/2012	04/04/12	13016444712	PEDRO LUIZ TAVARES MARZANO SOUZA	10/05/88	44233	55
14	06/03/2012	04/04/12	5772030701	RAFAEL HENZE PIMENTEL	05/09/85	14040	5
15	06/03/2012	04/04/12	90845439715	MARIA GORETE PAULINO NOVO	05/08/65	12359	58
16	06/03/2012	04/04/12	5396364793	DÉBORA VANESSA BARROS DE ALMEIDA	23/06/78	14471	21
17	06/03/2012	04/04/12	2595534769	DANIELA MILANEZ	23/07/73	17101	13
18	06/03/2012	04/04/12	4167238705	JULIANA FERRAZ DE OLIVEIRA CARREIRA	16/04/77	13530	60
19	06/03/2012	04/04/12	3539286748	ANA REGINA PEREIRA LIMA	09/01/74	43763	67
20	06/03/2012	04/04/12	7435795771	ALINE SENRA PIRES	27/07/77	13847	141
21	06/03/2012	04/04/12	12488415731	NATACHA GOMES DA COSTA MARINS	23/03/88	44089	122
22	06/03/2012	04/04/12	4755982732	MARCOS PAULO MONTEIRO PEREIRA	06/04/75	12588	139
23	06/03/2012	04/04/12	6521860763	LUIZ MOREIRA LEITE	15/01/24	20140	58
24	06/03/2012	04/04/12	664987049	LUCY DE AZEVEDO BARBOSA	29/05/11	20047	58
25	06/03/2012	04/04/12	47816945604	JOSE WILSON DA SILVA	02/02/67	12599	59
26	06/03/2012	04/04/12	7878020779	LUCIANO PIRES DE MORAIS	29/05/79	13533	88
27	06/03/2012	04/04/12	81172931704	ROGÉRIO CESAR COSTA DE AZEVEDO	27/01/65	13060	102

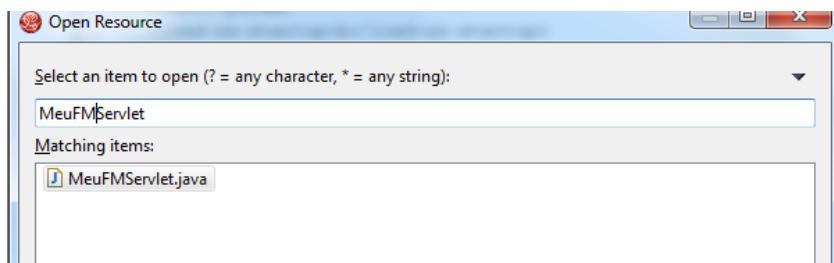
The screenshot shows the schema comparison feature in Oracle SQL Developer. It compares the schemas of two databases: 'corporativo (desenv)' on the left and 'sigia (desenv)' on the right. Both schemas contain numerous tables, such as 'EX_BOLETIM_DOC', 'EX_CLASSIFICACAO', 'EX_COMPETENCIA', etc., which are highlighted in blue, indicating they are present in both schemas.

- corporativo (desenv) Tables (Filtrado):**
 - CAD_SIT_FUNCIONAL
 - CP_APPLICACAO_FERIADO
 - CP_CONFIGURACAO
 - CP_FERIADO
 - CP_GRUPO
 - CP_IDENTIDADE
 - CP_LOCALIDADE
 - CP_MARCA
 - CP_MARCADOR
 - CP_MODELO
 - CP_OCURRENCIA_FERIADO
 - CP_ORGAO
 - CP_ORGAO_USUARIO
 - CP_PAPEL
 - CP_PERSONALIZACAO
 - CP_SEDE
 - CP_SERVICO
 - CP_SITUACAO_CONFIGURACAO
 - CP_TIPO_CONFIGURACAO
 - CP_TIPO_GRUPO
 - CP_TIPO_IDENTIDADE
 - CP_TIPO_LOTACAO
 - CP_TIPO_MARCA
 - CP_TIPO_MARCADOR
 - CP_TIPO_PAPEL
 - CP_TIPO_PESSOA
 - CP_TIPO_SERVICO
 - CP_TIPO_SERVICO_SITUACAO
 - CP_UF
 - DP_CARGO
 - DP_ESTADO_CIVIL
 - DP_FUNCAO_CONFIANCA
 - DP_LOTACAO
 - DP_PADRAO_REFERENCIA
 - DP_PESSOA
 - DP_Provimento
 - DP_SUBSTITUICAO
 - PLAN_TABLE
 - SCHEMA_VERSION
- sigia (desenv) Tables (Filtrado):**
 - EX_BOLETIM_DOC
 - EX_CLASSIFICACAO
 - EX_COMPETENCIA
 - EX_CONFIGURACAO
 - EX_DOCUMENTO
 - EX_DOCUMENTO_INT
 - EX_EMAIL_NOTIFICACAO
 - EX_ESTADO_DOC
 - EX_ESTADO_TP_MOV
 - EX_FORMA_DOCUMENTO
 - EX_MOBIL
 - EX_MODELO
 - EX_MODELO_TP_DOC_PUBLICACAO
 - EX_MOVIMENTACAO
 - EX_NIVEL_ACESSO
 - EX_NUMERACAO
 - EX_PAPEL
 - EX_PREENCHIMENTO
 - EX_SITUACAO_CONFIGURACAO
 - EX_TEMPORALIDADE
 - EX_TIPO_DESPACHO
 - EX_TIPO_DESTINACAO
 - EX_TIPO_DOCUMENTO
 - EX_TIPO_FORMA_DOCUMENTO
 - EX_TIPO_MOBIL
 - EX_TIPO_MOVIMENTACAO
 - EX_TP_DOC_PUBLICACAO
 - EX_TP_FORMA_DOC
 - EX_TP_MOV_ESTADO
 - EX_VIA
 - SCHEMA_VERSION

Esta versão do SQLDeveloper não realiza engenharia reversa, porém traz informações adicionais ao DBVisualizer (que realiza engenharia reversa) que será visto mais adiante.

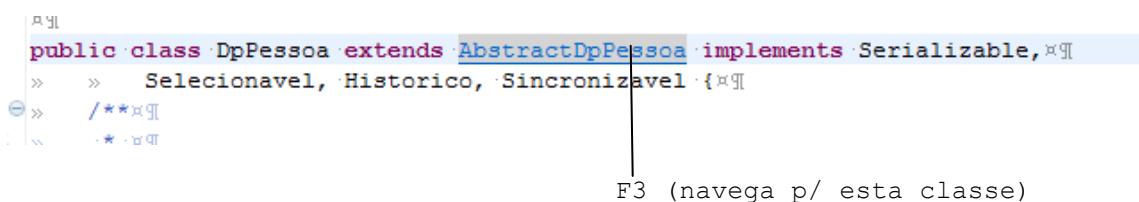
Anexo 3 - Utilizando o JBOSS Developer Studio

3.1 – Pesquisando uma classe (Ctrl+Shift+R)



Clicando em MeuFMServlet.java abre o mesmo no editor. Porém onde ele está na árvore do Package Explorer? Ver no item 3.9 (Link Editor).

3.2 – OpenOn – Navegando nas classes – (Cursor na classe e F3 ou Ctrl+click)



OpenOn provides an easy method for switching directly from one project resource to another without navigating through the Package Explorer view. Pressing F3 or Ctrl+click when a reference to another file is highlighted will open the file in the editor. Refer to the Editors chapter of the Visual Web Tools Reference Guide for more details.

3.3 – Ajuda Comando – (Ctrl+Space)

Content Assist displays context-specific code completion suggestions while typing, speeding up development and reducing typing errors. Content Assist is supported in the following contexts: The suggestion list can be displayed by pressing Ctrl+Space, and the highlighted entry can be selected and inserted by pressing Enter.

Note

3.4 – Encontrar os atributos / métodos dentro da classe – (Ctrl+O)

Com uma classe no editor, o Ctrl+O promove a lista dos métodos e atributos que podem ser filtrados.



3.5 – Hierarquia das classes (Open type Hierarchy – F4)

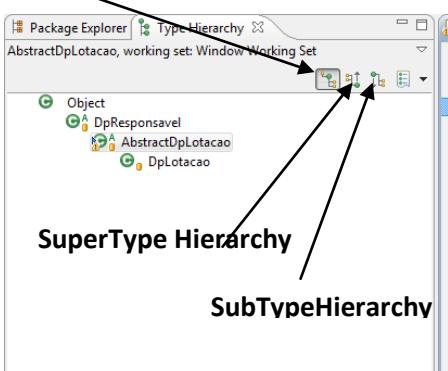
```

63  ...
64  >> @Desconsiderar<%
65  >> private Set<DpLotacao> dpLotacaoSubordinadosSet;<%
66  >> private CpTipoLotacao cpTipoLotacao;<%
67  >> /**
68  >> * @return the cpTipoLotacao<%
69  >> */
70  >> public CpTipoLotacao getCpTipoLotacao() {<%
71  >>     >> return cpTipoLotacao;<%
72  >> }<%
73  >> /**
74  >> */

```

Undo Ctrl+Z
Revert File
Save Ctrl+S
Open Declaration F3
Open Type Hierarchy F4
Open Call Hierarchy Ctrl+Alt+H
Show in Breadcrumb Alt+Shift+B
Quick Outline Ctrl+O

Type Hierarchy p/ AbstractDpLotacao (Dpresponsavel -> AbstractDpLoatacao -> DpLotacao)



3.6 – Onde é referenciado (Open Call Hierarchy – Ctrl+Alt+H)

Ex: onde o método getCpTipoLotacao() é utilizado

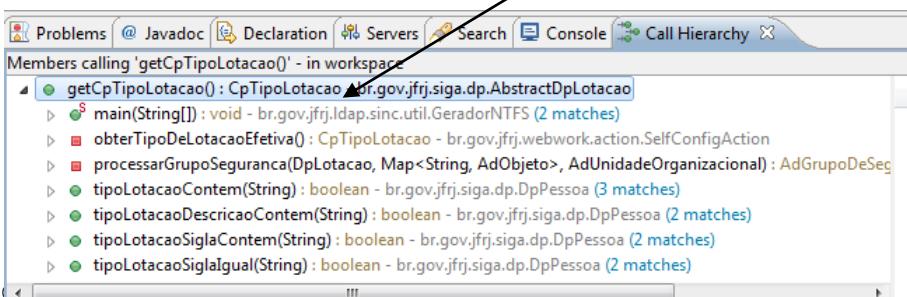
```

63  ...
64  >> @Desconsiderar<%
65  >> private Set<DpLotacao> dpLotacaoSubordinadosSet;<%
66  >> private CpTipoLotacao cpTipoLotacao;<%
67  >> /**
68  >> * @return the cpTipoLotacao<%
69  >> */
70  >> public CpTipoLotacao getCpTipoLotacao() {<%
71  >>     >> return cpTipoLotacao;<%
72  >> }<%
73  >> /**
74  >> */

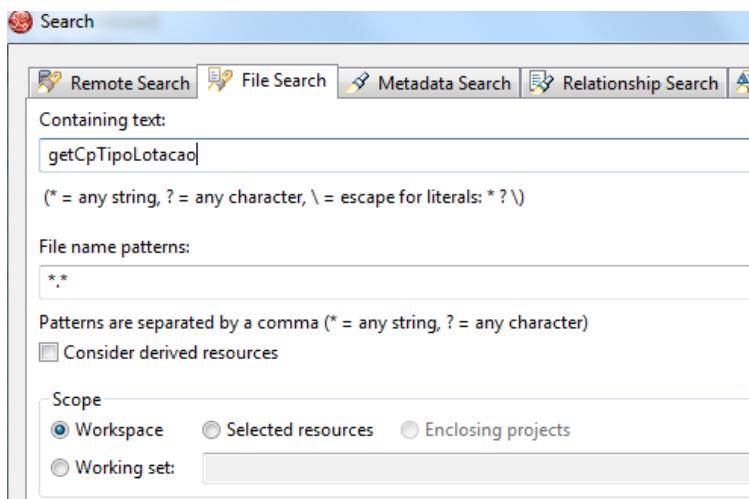
```

Undo Ctrl+Z
Revert File
Save Ctrl+S
Open Declaration F3
Open Type Hierarchy F4
Open Call Hierarchy Ctrl+Alt+H
Show in Breadcrumb Alt+Shift+B
Quick Outline Ctrl+O

Call Hierarchy p/ o getCpTipoLotacao()



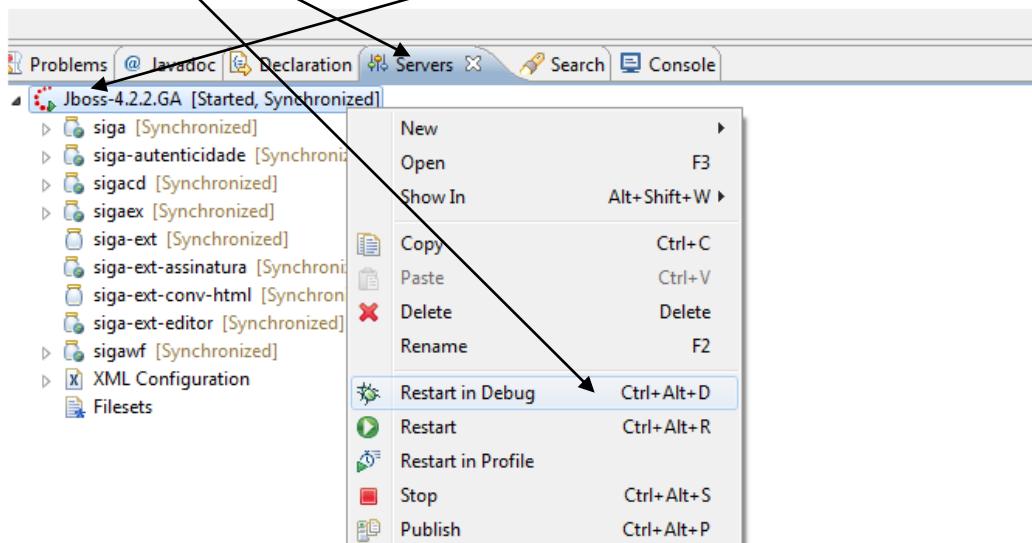
Esta forma de descobrir quem chama getCpTipoLotacao() não é 100% confiável, visto que as chamadas em código JSP ou FM não aparecerão no Call Hierarchy. Neste caso devemos utilizar o Search / File...



3.7 – Debugando um método JAVA

Colocando o servidor em debug Mode:

Na aba Servers, clicar com o botão direito no servidor (jboss-4.2.2.GA) e executar a opção Restart in Debug



Uma vez que o servidor esteja em debug mode, selecione a classe/ método desejado.

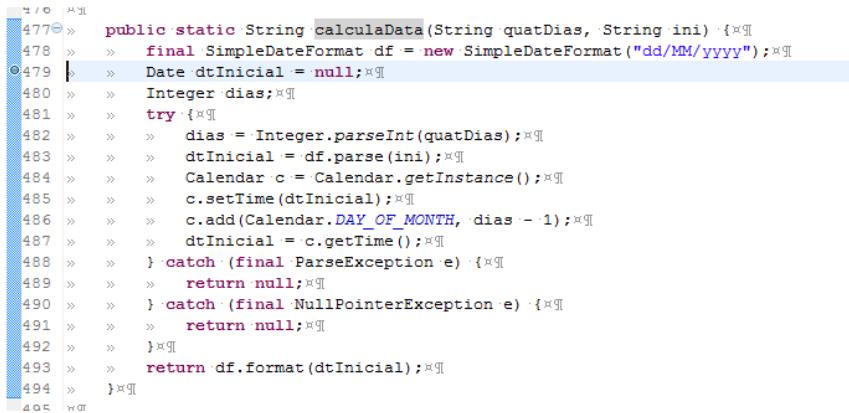
```
[#assign suprid = func.calculaData("22","08/09/2012")]
${suprid}
A aplicação FM acima executa o método calculaData() na classe
FuncoesEl.java.
```

Encontrando a classe e método:

Utilizando Ctrl+Shift+R (item 1) fornecemos o nome da classe.
Utilizando Ctrl+O (item 4) fornecemos o nome do método.

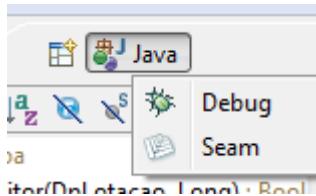
Selecione a linha e insira um breakpoint com duplo click.

Dê um duplo clique na linha desejada. No caso, 479. Observe o círculo cheio



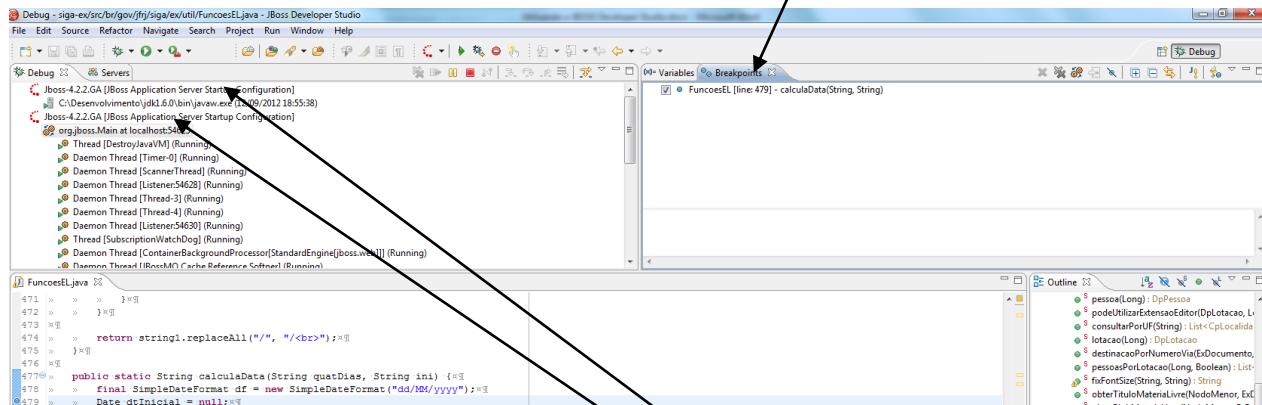
```
477 >     public static String calculaData(String quatDias, String ini) {  
478 >         final SimpleDateFormat df = new SimpleDateFormat("dd/MM/yyyy");  
479 >         Date dtInicial = null;  
480 >         Integer dias;  
481 >         try {  
482 >             dias = Integer.parseInt(quatDias);  
483 >             dtInicial = df.parse(ini);  
484 >             Calendar c = Calendar.getInstance();  
485 >             c.setTime(dtInicial);  
486 >             c.add(Calendar.DAY_OF_MONTH, dias - 1);  
487 >             dtInicial = c.getTime();  
488 >         } catch (final ParseException e) {  
489 >             return null;  
490 >         } catch (final NullPointerException e) {  
491 >             return null;  
492 >         }  
493 >         return df.format(dtInicial);  
494 >     }  
495 >
```

Mude a perspectiva de JAVA para Debug:



Verifique se não existem outros breakpoints indesejados na aba Breakpoints.

Verifique se não existem outras instâncias do jboss rodando.



OBSERVAÇÃO: No caso acima existem 2 instâncias, e quando rodei a aplicação FM o debug não entrou. Coloquei o cursor nas instâncias, botão direito e executei a opção Terminate and Remove. Depois startei o servidor novamente em modo debug.

Executei novamente a aplicação FM, e o JBDS acusou (ícone piscando) o debug / breakpoint.



Código parado do breakpoint:

```
FuncoesEl.java
```

```
471 >   >   >   }<br>
472 >   >   }<br>
473 > 
474 >   >   return string1.replaceAll("/", "<br>");<br>
475 >   > 
476 > 
477 >   public static String calculaData(String quatDias, String ini) {<br>
478 >       final SimpleDateFormat df = new SimpleDateFormat("dd/MM/yyyy");<br>
479 >       Date dtInicial = null; <br>
480 >       Integer dias; <br>
481 >       try {<br>
482 >           dias = Integer.parseInt(quatDias);<br>
```

A screenshot of the Java code editor in JBDS. The code is in 'FuncoesEl.java'. A line number 479 is highlighted with a blue background, and the word 'null' is highlighted in green. A black arrow points from the text above to this line.

Opções:

Resume(F8)

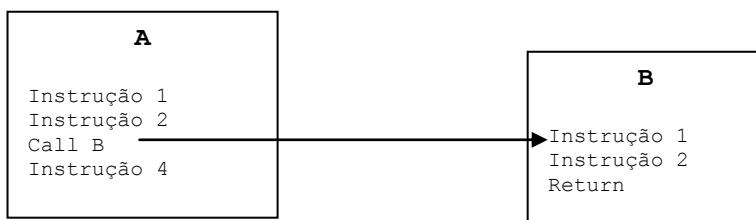
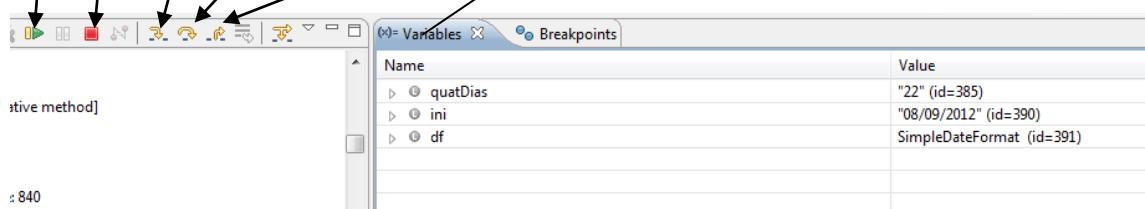
Terminate(Ctrl+F2)

Step Into(F5)

Step Over F6)

Step Return (F7)

Conteúdo das variáveis



Terminate - Interrompe o servidor em modo Debug

Resume - Executa o código até o próximo breakpoint, caso exista, ou até o final do código.

Step over - Passo sobre: vai executando linha-a-linha, porém se o seu código chama outro método / função (incluindo os métodos / funções da linguagem, tais

como: parse.int, string ...) ele não entra. Se na figura acima o debug (em A) está na instrução Call B, ele executará B, porém não entrará no código de B.

Step **over** proceeds to the next line in your current scope (i.e. it goes to the next line), without descending into any method calls on the way. This is generally used for following the logic through a particular method without worrying about the details of its collaborators, and can be useful for finding at what point in a method the expected conditions are violated.

Step into - Passo Dentro. Idêntico ao step over, porém ele entrará nos métodos / funções chamados. Se na figura acima o debug (em A) está na instrução Call B, ele entrará em B, e continuará executando linha-a-linha. A única forma de sair de B é chamar o Step out / Step return.

Step **into** will cause the debugger to descend into any method calls on the current line. If there are multiple method calls, they'll be visited in order of execution; if there are no method calls, this is same as step over. This is broadly equivalent to following every individual line of execution as would be seen by the interpreter.

Step Out / Step Return - termina a execução do método / função e retorna ao chamador na instrução de chamada. Se na figura acima o debug (em A) está na instrução Call B, e o usuário está modo Step into, ele entrará em B, e continuará executando linha-a-linha. A única forma de sair de B diretamente (sem executá-lo) é chamar o Step out / Step return, que fará o debug retornar para A na instrução Call B. Se usuário fornecer Step over, ele executará sem entrar em B, se der Step into, entrará novamente em B, e executará linha-a-linha.

Step out proceeds until the next "return" or equivalent - i.e. until control has returned to the preceding stack frame. This is generally used when you've seen all you need to at thispoint/method, and want to bubble up the stack a few layers to where the value is actually used.

Step Into	Step Over	Step Out / Step Return
<ul style="list-style-type: none">➤ Step Into will cause the debugger to go into the next function call and break there.➤ Go into the subroutine and wait for next action.➤ Change the debugger context to run into the function the code is stopped on. If the code cannot step into the function, this is the same as Step Over.	<ul style="list-style-type: none">➤ Step Over will tell the debugger to execute the next function and break afterwards.➤ Jump over the subroutine without waiting again.➤ Execute the code the debugger is stopped on, but stay within the current function.	<ul style="list-style-type: none">➤ Step Out will tell the debugger to finish the current function and break after it.➤ If you are in the subroutine, you will leave it without waiting again➤ Execute code until the end of the current function, and resume debugging once it has returned.

3.8 – Limpando Projetos Publicados

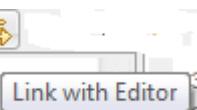
Clean projects published on the server.

You can use the clean option available in the Servers view to remove any invalid resources from the server before doing a full republish. This helps remove by-products generated as a result of the publishing process. When you find there is old code or an invalid state of code running on the server, try using this clean option to see if this helps remove these invalid states from the server.

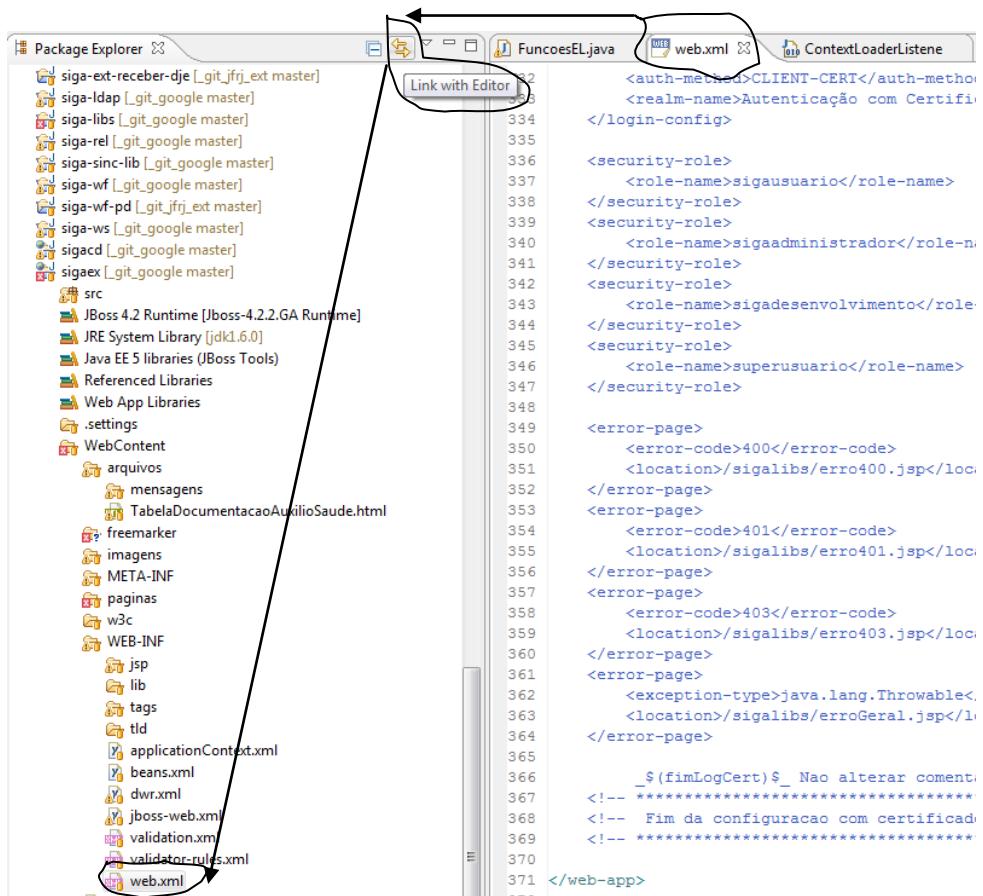
To clean projects published on the server, complete the following steps:

1. In the Servers view, right-click a server.
2. In the pop-up menu, select Clean.
3. The following message dialog box displays:
4. Clean will discard all publish state and republish from scratch. Are you sure you want to clean all published projects?
5. Select OK.

3.9 – Link do Editor com o Package Explorer (Tree)



Com um aplicativo aberto no Editor, pode-se rapidamente visitá-lo na árvore do projeto (Package Explorer) utilizando a opção Link with Editor.



Anexo 4 – Disponibilizando uma função TLD para acesso pelo FM (ou JSP)

Para que uma função TLD(Tag Library Descriptor) funcione no FM (ou JSP), além, é óbvio, de incluí-la na classe respectiva como um método, devemos escrever a sua assinatura em um arquivo `.tld` correspondente. No FM podemos expor diretamente uma classe / método sem o artifício do TLD (é o caso dos métodos nas classes `doc` e `exbl`).

4.1 - O que é TLD?

TLD – Tag Library Descriptor

Uma Taglib é nada mais nada menos que uma biblioteca de tags customizadas que são utilizadas na composição de páginas JSP. Em um passo adiante, podemos dizer que uma Taglib é uma biblioteca de “classes Java” que são utilizadas na forma de tags para auxiliar na geração de conteúdo dinâmico em uma página JSP. Tags Customizadas nos ajudam a eliminar consideravelmente a utilização de scriptlets e redundância de código em páginas JSP. *“Escrevemos uma tag customizada que gere o conteúdo dinâmico que precisamos e a utilizamos em quantas páginas JSP desejarmos”.*

Um arquivo `.tld` é um documento XML contendo dados que definem uma Custom Tag e / ou EL Function.

- **Custom Tag:** é um termo genérico para um código o qual simplifica a escrita de código JAVA (scriptlets, classes ...). Geralmente os arquivos `.tag` são escritos em JSP com extensão `.tag` (hora.tag, texto.tag ...). É a forma de desenvolvedores, sem conhecimento de JAVA, escreverem tags customizadas sem a necessidade de escrever-las em JAVA. É a possibilidade de se criar macros e são equivalentes as macros FM. Pode se passar parâmetros, porém sem retorno.

Exemplo: `:<myTagLibrary:texto/>`

- **Custom EL (Expression language) Functions:** é um método JAVA que se torna uma função customizada a qual pode ser invocada utilizando JSP (ou FM). Na verdade, um arquivo TLD de funções é como se fosse uma <<interface>> Java, com os métodos e as respectivas assinaturas.

Exemplo: `${myTagLibrary:calculaData() }`

Relação dos TLDs no SIGA-DOC:

Arquivos <code>.tld</code> em (WEB-INF)	Custom Function ou Custom TAG	Classe base correspondente
Func.tld	Function	FuncoesEl.java Obs: um arquivo <code>.tld</code> pode se referenciar a várias classes
Modelos.tld	TAG	
pdf4ml.tld	TAG	
Tags.tld	TAG	

Exemplo:

Supor que queiramos disponibilizar um método chamado `calculaData`, da classe `FuncoesEl.java` no FM.

4.2 - Qual é a assinatura do método?

Procurar na classe base os tipos de dados passados e recebido.

```
public class FuncoesEL {...  
public static String calculaData(String quatDias, String ini) {  
...}
```

Neste exemplo, passamos dois strings (quantidade de dias e data inicial) e recebemos um string (data final).

4.3 - Incluindo a nova function no `.tld` (no caso, Arquivo `func.tld`)

...

```
<function>
  <name>
    calculaData
  </name>
  <function-class>
    br.gov.jfrj.siga.ex.util.FuncoesEL
  </function-class>
  <function-signature>
    java.lang.String calculaData(java.lang.String,java.lang.String)
  </function-signature>
</function>
...

```

4.4 - Mapear o arquivo func.tld no WEB.XML, caso não esteja mapeado

```
<taglib>
  <taglib-uri>http://localhost/modelostag</taglib-uri>
  <taglib-location>/WEB-INF/tld/func.tld</taglib-location>
</taglib>
```

4.5 - Utilização no FM

Supor que func.tld tenha sido exposto ao hash do FM como func.

```
[#assign suprid = func.calculaData("22","08/09/2012")]
```

```
A data é ${suprid}
```

Resultará em : 29/09/2012

4.6 - Utilização no JSP

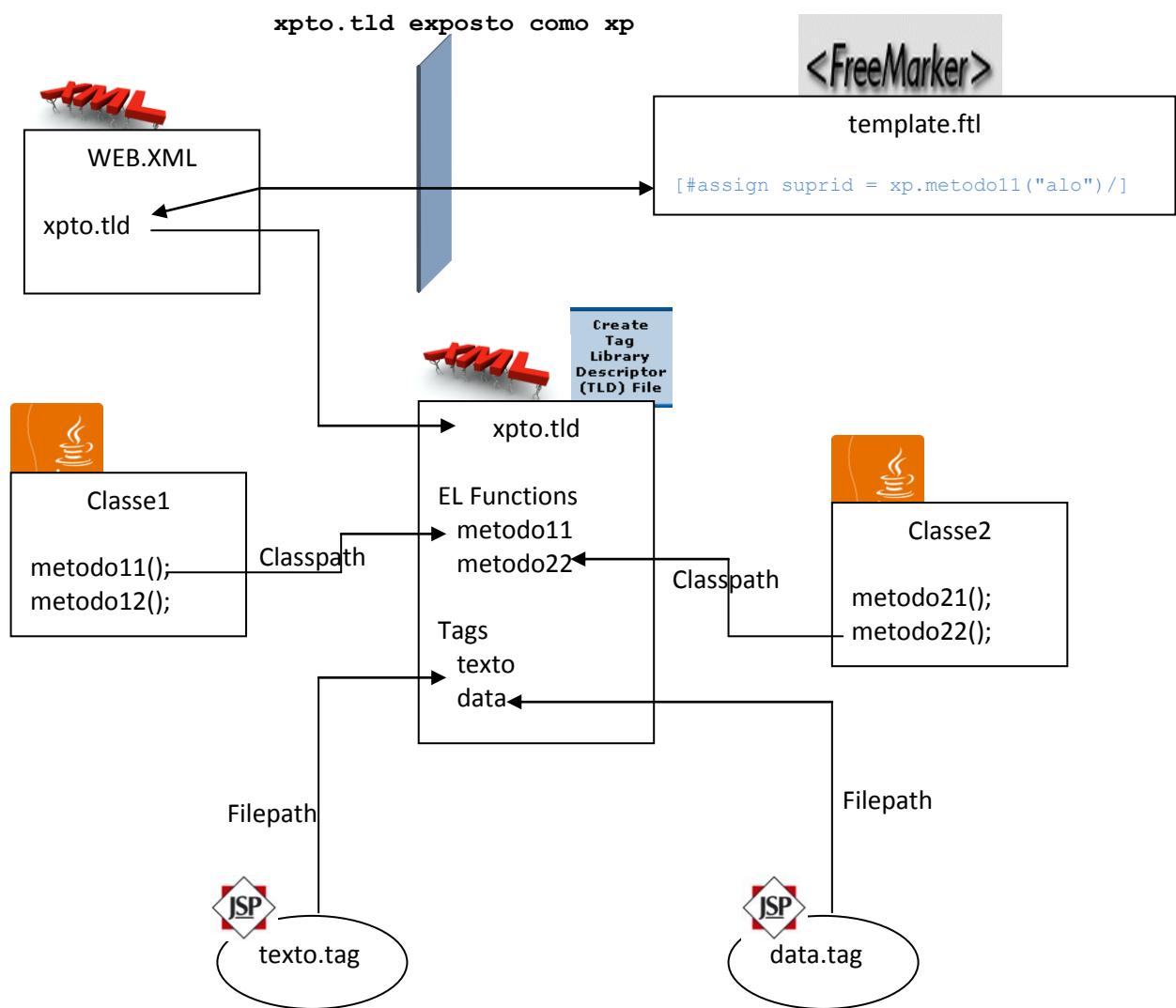
Supor que func.tld tenha sido exposto ao JSP com prefixo f.

```
<%@ taglib uri ="http://localhost/modelostag" prefix ="f"%>
```

```
...
```

```
A data é ${f:calculaData("22","08/09/2012")}
```

4.7 – Diagramaticamente



`texto.tag` e `data.tag` são equivalentes as macros FM, `texto` e `data`.

Anexo 5 – Filtros no SIGA-DOC

5.1 – Relação dos filtros no SIGA-DOC

Quando um formulário FM é exibido, não é só o servlet que o carrega e o interpreta que entra em ação, antes dele, como podemos notar na tabela abaixo para o pattern *.action, quatro filtros (classes JAVA) entram em ação.

Nome	ExFilter	SigaLog Filter	Salvamento Automatico Filter	webwork	ResponseHeader FilterNoCache	ResponseHeader Filter
Pattern						
*.action	x	x		x	x	
*.pdf	x					
*.html	x					
/dwr/*	x					
/anexo/*	x					
/servicos/*	x					
/expediente/doc/gravar_ajax.action			x			
/imagens/*						x
/recursos/*						x
/sigalibs/*						x

O que cada filtro faz:

ExFilter (classe ExThreadFilter): configura a sessionFactory do hibernate. Acusa exceções do tipo: Erro: sessão do Hibernate está fechada.; "Não foi possível configurar o hibernate".

SigaLogFilter(classe LogThreadFilter): realiza o log(em sigaex.log) dos acessos ao SIGA-DOC.

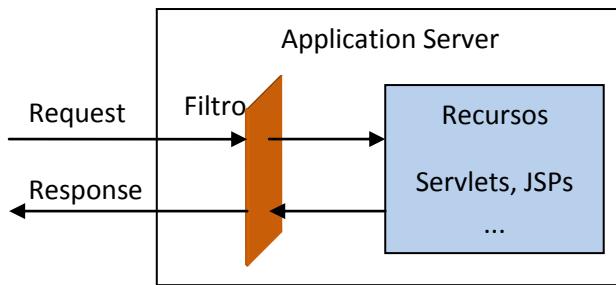
SalvamentoAutomaticoFilter(classe SalvamentoAutomaticoThreadFilter): cria uma lista de requisições para salvar automaticamente o conteúdo da página.

Webwork(classe FilterDispatcher): não encontrei a classe correspondente.

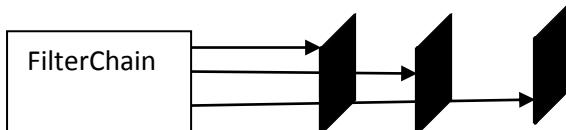
ResponseHeaderFilterNoCache(classe ResponseHeaderFilter): configura o header http com o parâmetro Cache-Control="no-cache" no envio da página (response).

ResponseHeaderFilter(classe ResponseHeaderFilter): configura o header http com o parâmetro Cache-Control="max-age=3600" no envio da página (response).

5.2 - DEFINIÇÃO



Os filtros podem ser encadeados (filterchain):



Os filtros permitem que código seja adicionado a aplicação sem interferir nos recursos (servlets, jsp ...) existentes.

Para que utilizar filtros?

- Para logar acesso aos recursos (quem, quando ...);
- Para gerar relatório de acesso, estatísticas etc.;
- Para impedir que certas pessoas tenham acesso a alguns recursos;
- Para comprimir o conteúdo da página durante o envio da mesma;
- Para setar parâmetros no http header tipo cache, charset e etc;
- Para substituir (replace) certo conteúdo por outro, como por exemplo: o nome da instituição, setores e etc.

Os filtros são recursos da especificação Servlet 2.3, possibilitando que você intercepte uma solicitação antes dela atingir um recurso. Um filtro dá acesso aos objetos HttpServletRequest e HttpServletResponse antes de serem passados a um servlet.

Para um filtro interceptar uma solicitação a um servlet, você precisa declarar o filtro com um elemento <filter> no Deploy Descriptor (DD) (web.xml) e mapear o filtro para um servlet, usando o elemento <filter-mapping>. Você pode também usar um filtro que trabalhe em múltiplos servlets. Isso pode ser feito mapeando um filtro a um URL padrão, para que qualquer solicitação que combine com aquela URL padrão seja filtrada.

Ao escrever um Servlet Filter, basicamente você lida com três interfaces (e os respectivos métodos) no pacote javax.servlet:

- 5.2.1 - **Filter**
 - **init**
 - **doFilter**
 - **destroy**
- 5.2.2 - **FilterConfig**
- 5.2.3 - **FilterChain**
 - **doFilter**

5.2.1 - A interface Filter, está localizada no pacote javax.servlet.Filter. É uma interface que você precisa implementar ao escrever um filtro. O ciclo de vida de um filtro é representado por três métodos da interface Filter, o **init**, **doFilter** e **destroy**.

Um filtro inicia a sua vida quando o seu método **init** é chamado pelo Server Container (por exemplo Apache Tomcat) de servlet. O container chama um método **init** do filtro apenas uma vez, quando ele termina de executar o filtro. O container

passará no objeto **FilterConfig**, que representa a configuração do filtro. O método **init** pode ser comparado ao método init da interface Servlet.

O método **doFilter** é onde a filtragem é feita. O container chama o método **doFilter** sempre que um usuário solicita um recurso, tal como um servlet, ao qual o filtro está mapeado. Quando o **doFilter** é chamado, o container passa o objeto HttpServletRequest, o HttpServletResponse e um objeto FilterChain. Os objetos HttpServletRequest e HttpServletResponse são os mesmos objetos que serão passados a um servlet.

```
public void init(FilterConfig filterConfig)
public void doFilter(HttpServletRequest request, HttpServletResponse response, FilterChain chain)
public void destroy()
```

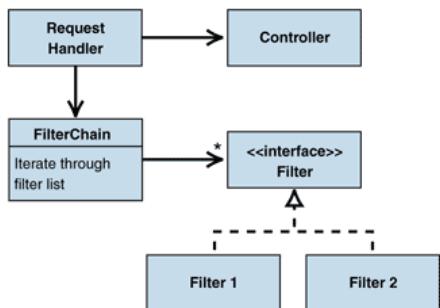
5.2.2 - Um objeto **FilterConfig** representa a configuração para o filtro. Esse objeto permite que você obtenha o objeto ServletContext e passe valores de iniciação ao filtro através de seus parâmetros iniciais, que são definidos na DD (WEB.XML), ao declarar o filtro.

```
public String getFilterName()
public String getInitParameter(String parameterName)
public java.util.Enumeration getInitParameterNames()
public ServletContext getServletContext()
```

5.2.3 - Um objeto **Filter Chain** é passado pelo container ao método **doFilter**, da classe do filtro. Os filtros usam o objeto FilterChain para chamar o próximo filtro na cadeia ou, se o filtro for o último na cadeia, para chamar o próximo recurso(servlet). A interface FilterChain tem apenas um método doFilter listado abaixo:

```
public void doFilter(HttpServletRequest request, HttpServletResponse response)
```

Esquematicamente:



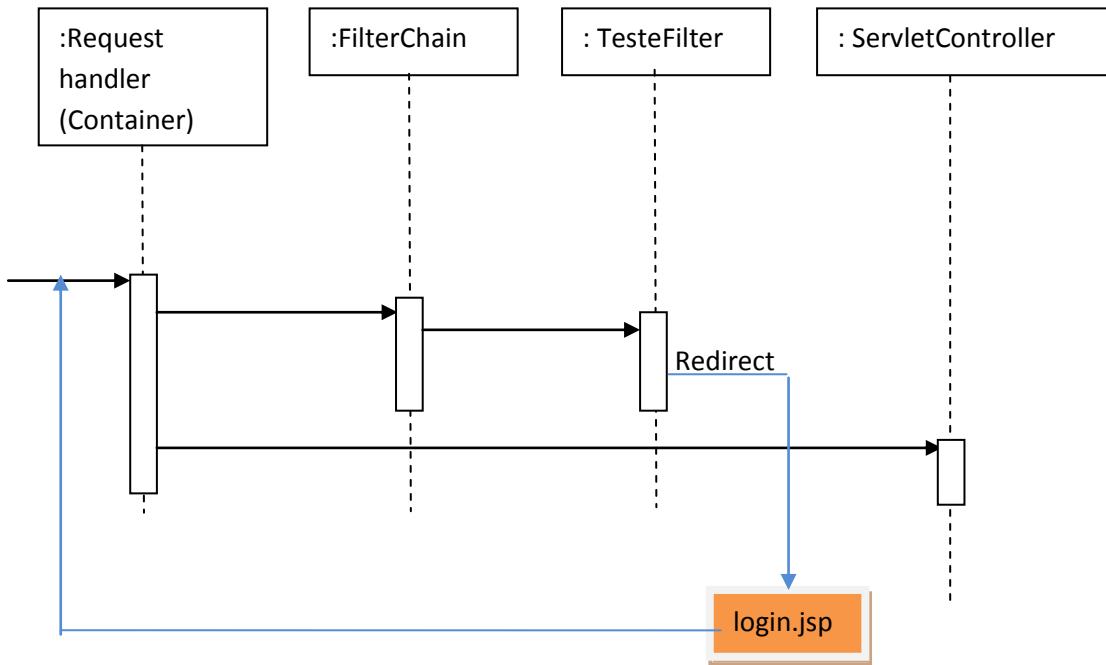
```
1. import java.io.IOException;
2. import javax.servlet.Filter;
3. import javax.servlet.FilterChain;
4. import javax.servlet.FilterConfig;
5. import javax.servlet.ServletException;
6. import javax.servlet.ServletRequest;
7. import javax.servlet.ServletResponse;
8. import javax.servlet.http.HttpServletRequest;
9. import javax.servlet.http.HttpServletResponse;
10. import javax.servlet.http.HttpSession;
11.
12.
13. public class TesteFilter implements Filter{
14.
15.     public void destroy() {
16.     }
17.
```

```
18.     public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException {
19.         ServletException ion, ServletException {
20.             HttpServletRequest req = (HttpServletRequest) request;
21.             HttpServletResponse res = (HttpServletResponse) response;
22.             HttpSession session = req.getSession();
23.             String user = (String) session.getAttribute("user");
24.             if(user!=null && user.trim().length()>0){
25.                 chain.doFilter(req,res); // OK tudo bem, siga em diante
26.             }
27.             res.sendRedirect("login.jsp"); // Pede ao usuário para se logar
28.         }
29.
30.     public void init(FilterConfig arg0) throws ServletException {
31.     }
32. }
```

WEB.XML

```
1.      <filter>
2.          <filter-name>validator</filter-name>
3.          <filter-class>TesteFilter</filter-class>
4.      </filter>
5.
6.      <filter-mapping>
7.          <filter-name>validator</filter-name>
8.      <url-pattern>/servlet/*</url-pattern>
9.      </filter-mapping>
```

Diagrama de Sequência do exemplo



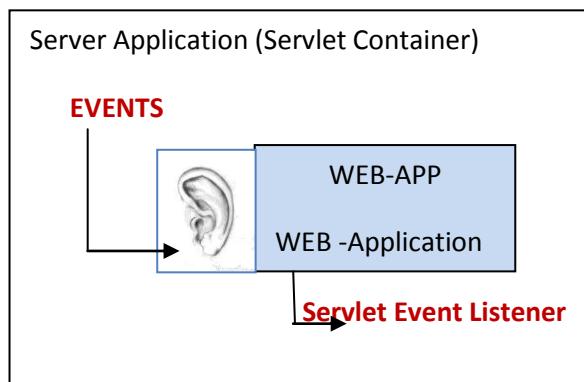
Anexo 6 – Listener no SIGA-DOC

6.1 – Relação dos Listeners no SIGA-DOC

Descrito em WEB.XML (DD)

```
<listener>
<listener-class>org.springframework.web.context.ContextLoaderListener</listener-
class>
</listener>
```

6.2 – Definição



O container pode ser utilizado para notificar a aplicação WEB de importantes eventos, e os Servlets Event Listener são os mecanismos.

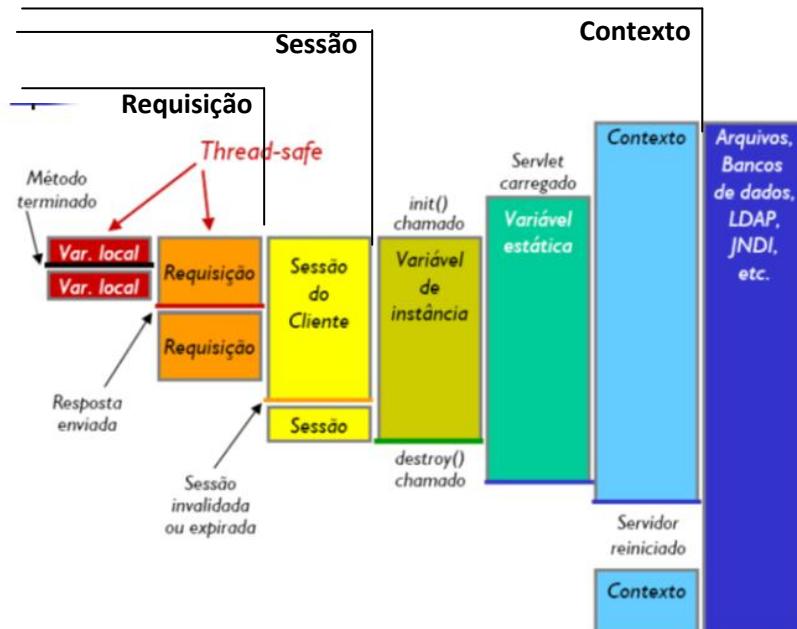
Os listeners permitem que código seja adicionado a aplicação sem interferir nos recursos (servlets, jsp ...) existentes.

Para que usar Listener(s)?

- Saber quando o container inicializa uma aplicação WEB ou quando a aplicação WEB é removida;
- Quando é necessário carregar um database em tempo de startup do container ou salvá-lo quando se realiza o shutdown do container;
- Manter trilha do número de usuários concorrentes . Embora isto possa ser feito de outras maneiras, é muito simples utilizar um listener que espera um cliente startar uma nova sessão.

6.3 – Os tipos de Listener

Antes é importante lembrar os tipos de estado / persistência de uma aplicação WEB.

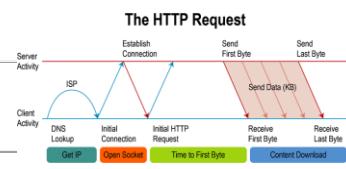


<<Interfaces>> para os Listener Events

REQUEST

`javax.servlet.ServletRequestListener`

void	<code>requestDestroyed(ServletRequestEvent sre)</code> Receives notification that a ServletRequest is about to go out of scope of the web application.
void	<code>requestInitialized(ServletRequestEvent sre)</code> Receives notification that a ServletRequest is about to come into scope of the web application.



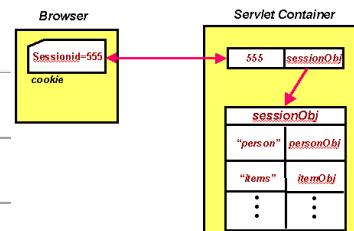
`javax.servlet.ServletRequestAttributeListener`

void	<code>attributeAdded(ServletRequestAttributeEvent srae)</code> Receives notification that an attribute has been added to the ServletRequest.
void	<code>attributeRemoved(ServletRequestAttributeEvent srae)</code> Receives notification that an attribute has been removed from the ServletRequest.
void	<code>attributeReplaced(ServletRequestAttributeEvent srae)</code> Receives notification that an attribute has been replaced on the ServletRequest.

SESSION

`javax.servlet.http.HttpSessionListener`

void	<code>sessionCreated(HttpSessionEvent se)</code> Receives notification that a session has been created.
void	<code>sessionDestroyed(HttpSessionEvent se)</code> Receives notification that a session is about to be invalidated.



`javax.servlet.http.HttpSessionAttributeListener`

void	<code>attributeAdded(HttpSessionBindingEvent event)</code> Receives notification that an attribute has been added to a session.
void	<code>attributeRemoved(HttpSessionBindingEvent event)</code> Receives notification that an attribute has been removed from a session.
void	<code>attributeReplaced(HttpSessionBindingEvent event)</code> Receives notification that an attribute has been replaced in a session.

`javax.servlet.http.HttpSessionActivationListener Methods`

void	<code>sessionDidActivate(HttpSessionEvent se)</code> Notification that the session has just been activated.
void	<code>sessionWillPassivate(HttpSessionEvent se)</code> Notification that the session is about to be passivated.

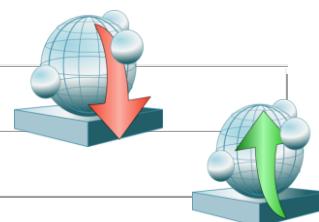
`javax.servlet.http.HttpSessionBindingListener Methods`

void	<code>valueBound(HttpSessionBindingEvent event)</code> Notifies the object that it is being bound to a session and identifies the session.
void	<code>valueUnbound(HttpSessionBindingEvent event)</code> Notifies the object that it is being unbound from a session and identifies the session.

CONTEXT

`javax.servlet.ServletContextListener`

void	<code>contextDestroyed(ServletContextEvent sce)</code> Receives notification that the ServletContext is about to be shut down.
void	<code>contextInitialized(ServletContextEvent sce)</code> Receives notification that the web application initialization process is starting.



`javax.servlet.ServletContextAttributeListener`

void	<code>attributeAdded(ServletContextAttributeEvent scab)</code> Notification that a new attribute was added to the servlet context.
void	<code>attributeRemoved(ServletContextAttributeEvent scab)</code>

	Notification that an existing attribute has been removed from the servlet context.
void	attributeReplaced(ServletContextAttributeEvent scab) Notification that an attribute on the servlet context has been replaced.

6.4- Exemplo

Listener: Implementação para visualizar Usuários Concorrentes no sistema

ConcurrenteuserTracker.java

```

33. package com.jspbook;
34.
1. import javax.servlet.*;
2. import javax.servlet.http.*;
3.
4. public class ConcurrentUserTracker implements HttpSessionListener {
5.     static int users = 0;
6.
7.     public void sessionCreated(HttpSessionEvent e) {
8.         users++;
9.     }
10.    public void sessionDestroyed(HttpSessionEvent e) {
11.        users--;
12.    }
13.    public static int getConcurrentUsers() {
14.        return users;
15.    }
16. }
```

WEB.XML (DD)

```

1. <listener>
2.   <listener-class>
3.     com.jspbook.ConcurrentUserTracker
4.   </listener-class>
5. </listener>
```

Observar que o DD não especifica que tipo de interface do listener está sendo usada. Este tipo de informação não é necessária porque o container pode descobrir.

Servlet para fornecer o display dos usuários concorrentes:

DisplayUsers.java

```

1. package com.jspbook;
2.
3. import java.io.*;
4. import javax.servlet.*;
5. import javax.servlet.http.*;
6.
7. public class DisplayUsers extends HttpServlet {
8.     public void doGet(HttpServletRequest request,
9.                         HttpServletResponse response)
10.        throws IOException, ServletException {
11.
12.     request.getSession();
13.
14.     response.setContentType("text/html");
15.     PrintWriter out = response.getWriter();
16.     out.println("<html>");
17.     out.print("Users:");
18.     out.print(ConcurrentUserTracker.getConcurrentUsers());
19.     out.println("</html>");
20.   }
21. }
```

Anexo 7 - GIT - Conceitos

7.1 - Introdução



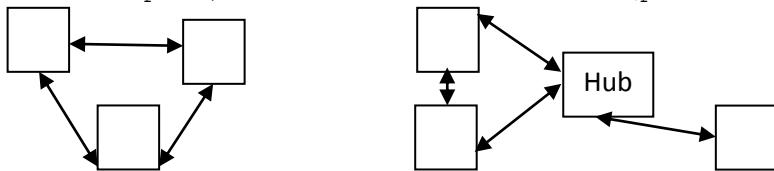
O Git é um sw de controle de versões distribuído. Ele foi desenvolvido tendo-se em mente vários grupos colaborando de forma independente / autônoma, porém sem perder o controle do que está sendo feito. O Git veio a atender os seguintes problemas:

- Independência de um servidor centralizado;
- Foco na aplicação e não nos arquivos de forma individual;
- Permitir que vários grupos colaborem e compartilhem o código, incluindo o desenvolvimento de várias versões da aplicação em paralelo.

Bom, a melhor forma de entender o Git é compará-lo com o CVS que todos conhecem.

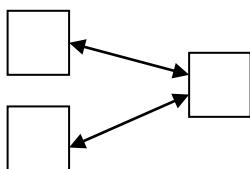
Quanto a topologia:

Git: distribuído puro, ou distribuído com hubs (parciais e / ou final).



Como exemplo de hubs final temos o GitHub e o Google que hospedam o código mais recente de um determinado projeto. O objetivo é todo mundo atualizar o hub final e obter o código mais recente do mesmo. Os hubs parciais podem ser configurados, por exemplo, para receber código de desenvolvedores externos, depois, um desenvolvedor interno atualizará o hub final a partir do hub parcial.

CVS: estrela (centralizado).



Unidade de Commit:

Git: Tree (árvore podendo conter vários arquivos).

CVS: arquivo.

Unidade de Versão:

Git: aplicação.

CVS: arquivo.

Vantagens / Desvantagens:

Grande Vantagem do CVS Quando você pega um arquivo para trabalhar, você tem garantias que ele não será alterado por outra pessoa.	Grande Vantagem do Git Como a unidade de versionamento é a aplicação, fica muito fácil voltar o estado da mesma a qualquer tempo / versão.
Grande problema do CVS Como a unidade de versionamento é o arquivo, fica difícil, ou mesmo impossível, retornar uma aplicação a	Grande problema do Git Supor que você pegue um arquivo para alterar em t0. Em t1 um outro usuário altera (ou até mesmo apaga) o arquivo

estados / versões anteriores.

e realiza o commit / merge. Em t2 você commita o arquivo e quando for realizar o merge o Git indica que existe uma inconsistência. Ou seja, no Git você não tem garantias que está trabalhando com a versão mais atual de um arquivo. Isto se chama no Git de CONFLITOS.

Conclusão: Se você está disponibilizando código para que outras equipes (de fora) contribuam ou se o sistema é de médio ou grande porte e o versionamento do mesmo é importante, sem dúvidas o Git é a melhor opção.

7.2 - Estrutura do Git

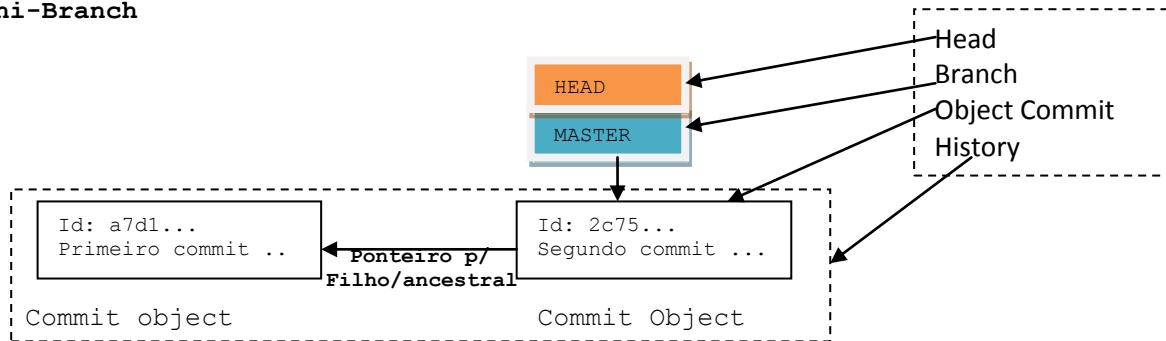
Head: é o ponteiro que aponta para um determinado branch.

Branch: é uma linha de desenvolvimento ou manutenção (esta será mergeada depois para um determinado branch). Podemos ter 1 ou N linhas de desenvolvimento e/ou manutenção .

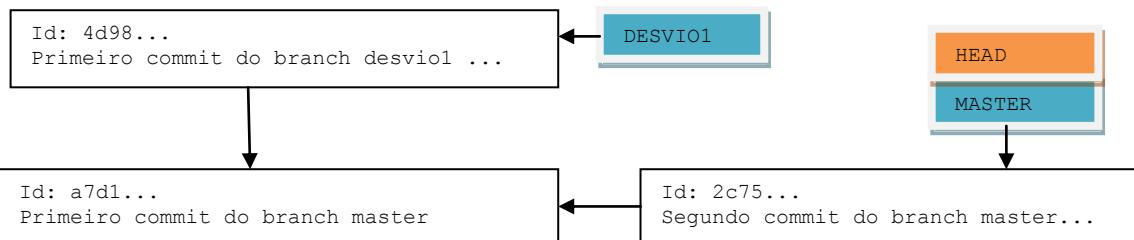
Commit object: é um commit específico contendo atributos tais como: mensagem (texto) e id (hash sha-1).

History: é o conjunto de todos object commit, realizando a história do projeto.

Uni-Branch



Multi-Branch

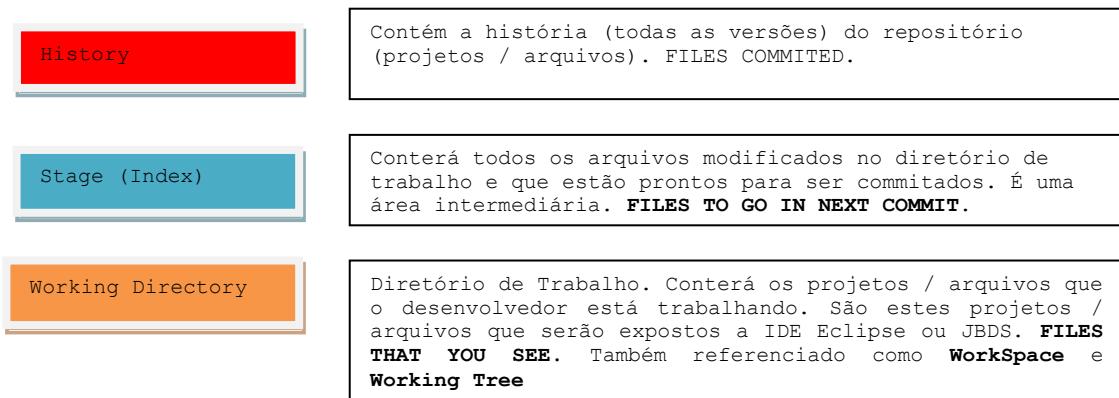


DESVIO1 pode ser um branch definitivo, ou seja, outra linha (em paralelo) de desenvolvimento da aplicação, ou temporário, podendo conter uma série de modificações que serão mergeadas (aplicadas) para o branch principal (master).

O HEAD pode estar apontando para o branch master ou desviol, dependendo da opção do desenvolvedor (switch branch).

Esquema Git em 3 camadas:

É muito comum a apresentação de esquema Git, principalmente para mostrar a aplicação de comandos, em forma de 3 camadas.



7.3 – Breve comentário sobre os principais comando Git

7.3.1 – Trabalhando com repositório local

Comando	Comentário
add files <code>git add -p</code> , instead of (or in addition to) specifying particular files to interactively choose which hunks copy	copies files (at their current state) to the stage.
commit <code>git commit -a</code> is equivalent to running git add on all filenames that existed in the latest commit, and then running git commit. <code>git commit files</code> creates a new commit containing the contents of the latest commit, plus a snapshot of files taken from the working directory. Additionally, files are copied to the stage.	saves a snapshot of the stage as a commit.
checkout -- files <code>git checkout HEAD -- files</code> copies files from the latest commit to both the stage and the working directory. <code>git checkout -p</code> , instead of (or in addition to) specifying particular files to interactively choose which hunks copy	copies files from the stage to the working directory. Use this to throw away local changes.
reset -- files <code>reset -p</code> , instead of (or in addition to) specifying particular files to interactively choose which hunks copy	unstages files; that is, it copies files from the latest commit to the stage. Use this command to "undo" a git add files. You can also git reset to unstage everything.
merge Obs: em múltiplos branches um merge cria um simples commit com dois pais, deixando a	creates a new commit that incorporates changes from other commits. Before merging, the stage must match the

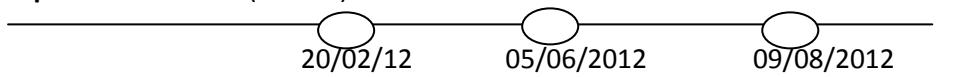
<p>história não linear. Para deixar a história linear deve-se usar o rebase no lugar.</p> <p>Obs: o merge está sujeito a conflitos</p> 	<p>current commit. The trivial case is if the other commit is an ancestor of the current commit, in which case nothing is done. The next most simple is if the current commit is an ancestor of the other commit. This results in a <i>fast-forward</i> merge. The reference is simply moved, and then the new commit is checked out.</p>
<p>cherry-pick</p>	<p>"copies" a commit, creating a new commit on the current branch with the same message and patch as another commit.</p>
<p>Rebase</p>	<p>is an alternative to a merge for combining multiple branches. Whereas a merge creates a single commit with two parents, leaving a non-linear history, a rebase replays the commits from the current branch onto another, leaving a linear history. In essence, this is an automated way of performing several cherry-picks in a row.</p>



Que tipo de conflito?

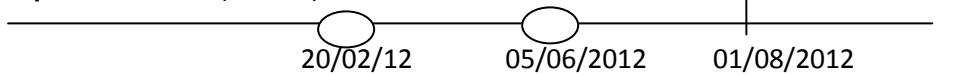
Supor o seguinte exemplo:

Repositório remoto (master)



Outro usuário alterou o fonte xpto.jsp e atualizou (push) o repositório remoto

Repositório Local (master)

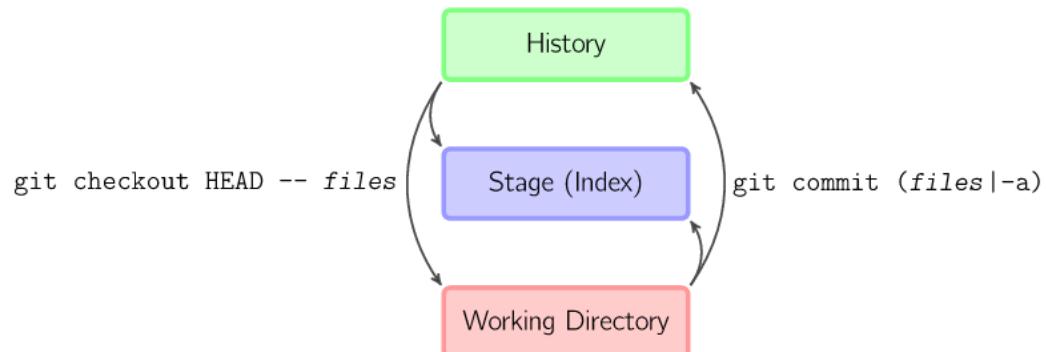
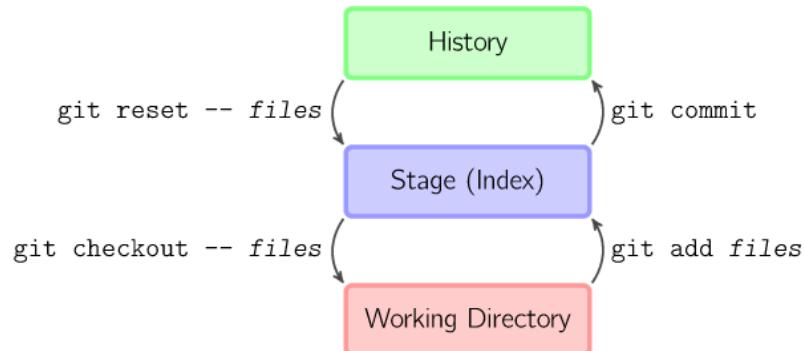


o usuário alterou o fonte xpto.jsp

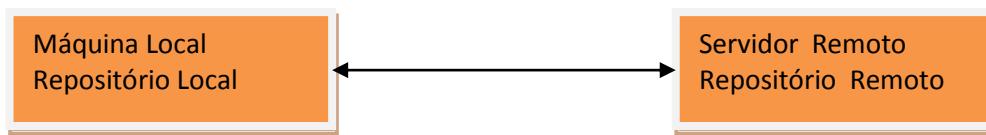
Se o usuário na máquina local realizar um PULL (fetch + merge), ou um MERGE (após o fetch no repositório remoto), o Git acusará um conflito no fonte xpto.jsp, conflito este que deverá ser administrado pelo usuário da máquina local, ou seja, receber o fonte mais novo do repositório remoto e aplicar as mudanças que ele realizou em 01/08/2012.

Um ótimo site para entender o funcionamento dos comandos locais do Git é o Visual Git Reference (<http://marklodato.github.com/visual-git-guide/index-en.html>), visto que utiliza o esquema de 3 camadas.

Abaixo, algumas ilustrações do site:



7.3.2 - Trabalhando com repositório remoto



Existem três tipos de branches no repositório máquina local:

- **Local branch (LB)**: que é o branch tradicional criado pelo git branch;
- **Remote Tracking Branch (RTB)**: são criados automaticamente quando **cloning** ou **fetching** repositórios remotos. Um RTB no repositório local sempre corresponde a um (local) branch no repositório remoto. O RTB aponta para o mesmo commit que o branch correspondente no repositório remoto, no momento do fetch / clone. RTBs podem ser usados para criação automatizada de "Upstream Configuration" dos branches locais;
- **Tracking Branch (TB)**: vamos assumir por agora que o TB é idêntico ao RTB, porém o TB pode ser modificado pelo usuário, pois é read-write e o RTB não, pois é read-only. Algumas literaturas assumem o TB como Local TB ou LTB.

Bom, para diferenciar o **RTB do TB** é melhor apresentar uma tabela e um esquema visual, visto que esta distinção não é fácil de explicar somente no modo textual.

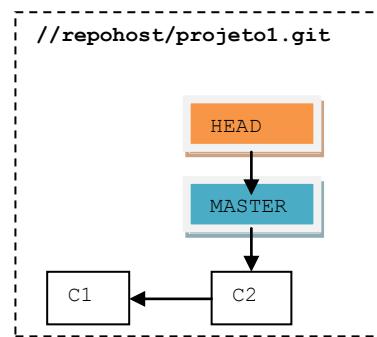
Tabela comparando **RTB e TB**

Tipo de Branch	RTB	TB
Diferenças		
Consegue seu conteúdo via	clone	clone
É atualizado via	fetch e pull. Ou seja, somente pelo repositório remoto	merge e pull. Ou seja, pelo repositório remoto eo pelo usuário (repositório local)
Client access	Read-only	Read-write
Pode-se publicar as mudanças via	Não pode	push

Esquemas visuais para ilustrar como o **RTB** e **TB** são criados e atualizados.

SITUAÇÃO 0 - Criando o Repositório Remoto

Supor que tenhamos o seguinte repositório remoto.



As situações

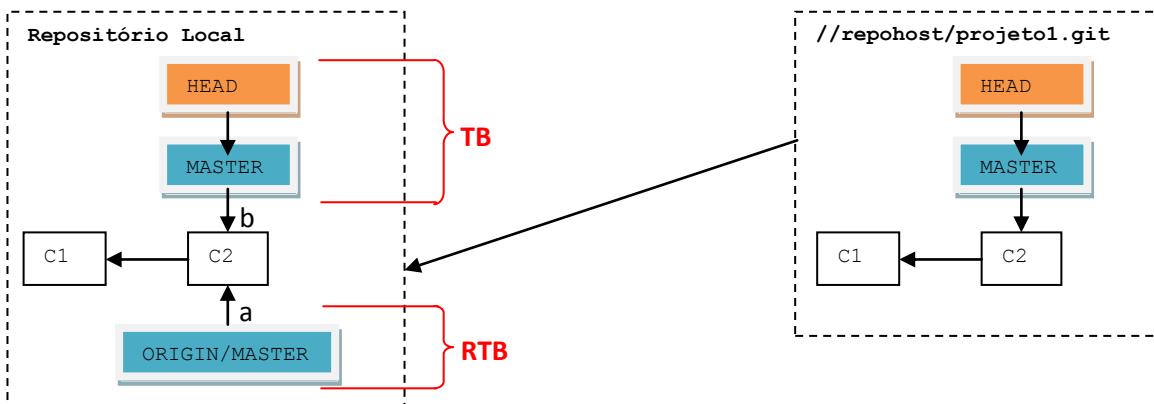
SITUAÇÃO 1 - Clonando um Repositório Remoto com CLONE

Um usuário (diferente do usuário da situação 0) em sua máquina local, limpa, só com o Git instalado emite o seguinte comando

> `git clone git://repohost/projeto1.git`

Neste momento o comando clone fará uma série de coisas:

- a - uma cópia do repositório remoto para o **RTB criado e nomeado de origin/master**. Este branch é basicamente read-only para o usuário local. Ele é atualizado somente a partir do repositório remoto.
- b - **um novo TB é criado e nomeado de master**. O conteúdo do RTB origin/master é copiado para o TB máster. O usuário pode fazer modificações neste branch.

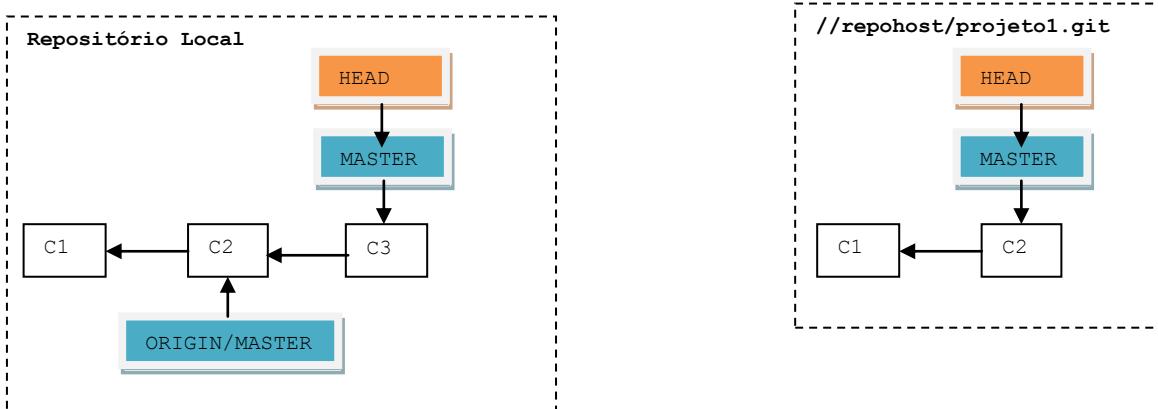


É interessante observar que a partir de agora temos dois branches (master e origin/master) que andam de forma independente.

SITUAÇÃO 2 – Realizando mudanças no branch TB

Supor que o usuário faça o seguinte na máquina local:

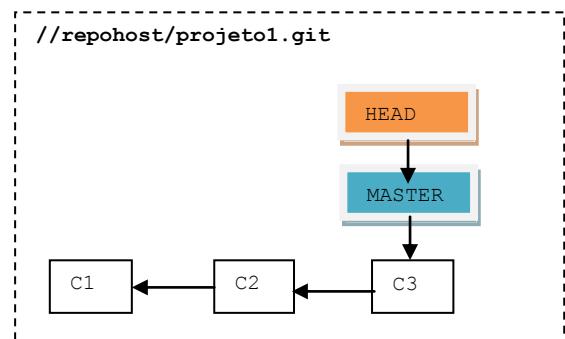
```
> altere algum arquivo  
> git add  
> git commit -m "minha primeira modificação"
```



Só o branch master se movimentou.

SITUAÇÃO 3 – Modificações no Repositório Remoto

Retornando a situação 1, supor agora que alguém tenha alterado o repositório remoto:

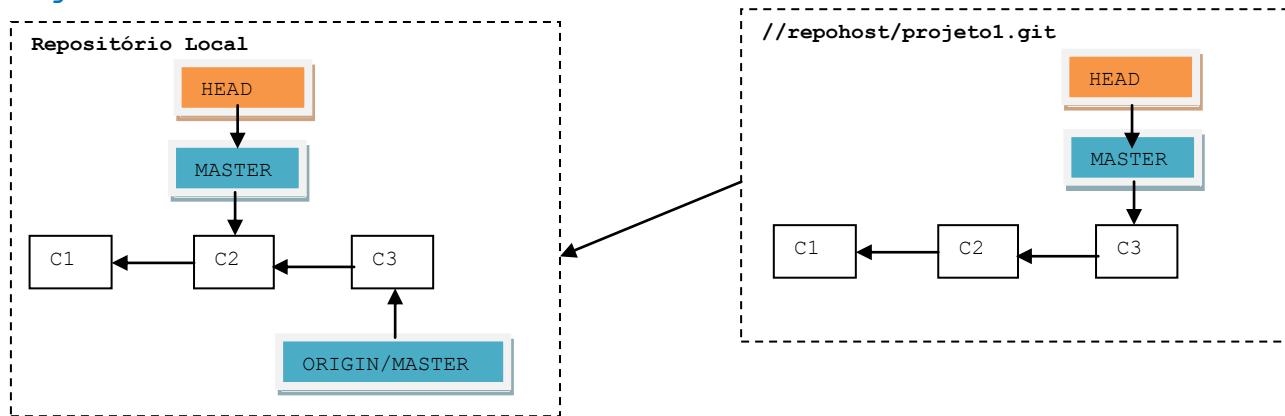


SITUAÇÃO 4 – Buscando novas atualizações no Repositório Remoto com FETCH

Retornando a situação 1.

Supor que o usuário faça o seguinte na máquina local:

> `git fetch`



Só o branch origin/master se movimentou.

E se quiséssemos atualizar o master (TB) no repositório local para apontar para o novo commit C3?

Neste caso, poderíamos emitir o seguinte comando:

> `git merge origin/master`

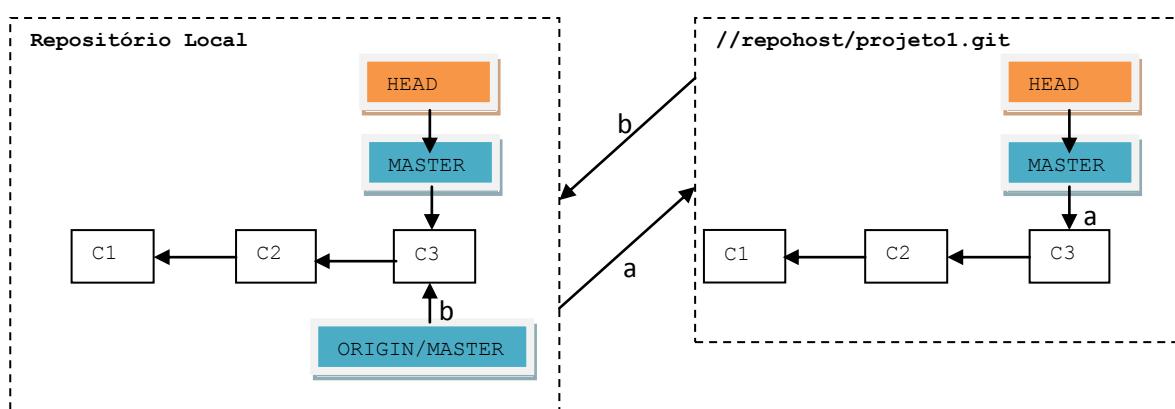
A sintaxe do merge é: `merge <branch-to-merge-from>`. Uma alternativa para o `git fetch e git merge` é o `git pull`.

SITUAÇÃO 5 – Publicando atualizações no Repositório Remoto com PUSH

Retornando a situação 2, supor que o usuário queira atualizar o repositório remoto.

Supor que o usuário faça o seguinte na máquina local:

> `git push`



Neste momento o commando push fará uma série de coisas:

a - o branch local, master, é publicado para o repositório remoto. Neste caso específico, o commit C3 é publicado no repositório remoto;

b - o RTB é também atualizado com novos updates a partir do repositório remoto.

Neste caso específico, o branch origin/máster também aponta para o novo commit C3.

Breve resumo dos principais comandos remotos do Git

Comando	Comentário
clone	O clone foi apresentado na situação 1. Como o próprio nome diz, ele clona um repositório. Ele cria o TB e O RTB.
push	A definição do manual é: Update remote refs along with associated objects , porém prefiro uma definição mais representativa tal como, Upload changes from your local repository into a remote repository. Ver situação 5.
pull	Pull = Fetch + Merge. Git pull" is exactly equivalent to "git fetch" followed by "git merge". Fetch a partir do repositório remoto e merge com o branch local corrente. O PULL atualiza o TB e o RTB.
fetch	Download novos branches e dados a partir do repositório remote. O FETCH só atualiza o RTB.
remote	Lista, adiciona e deleta os alias de repositórios remotos

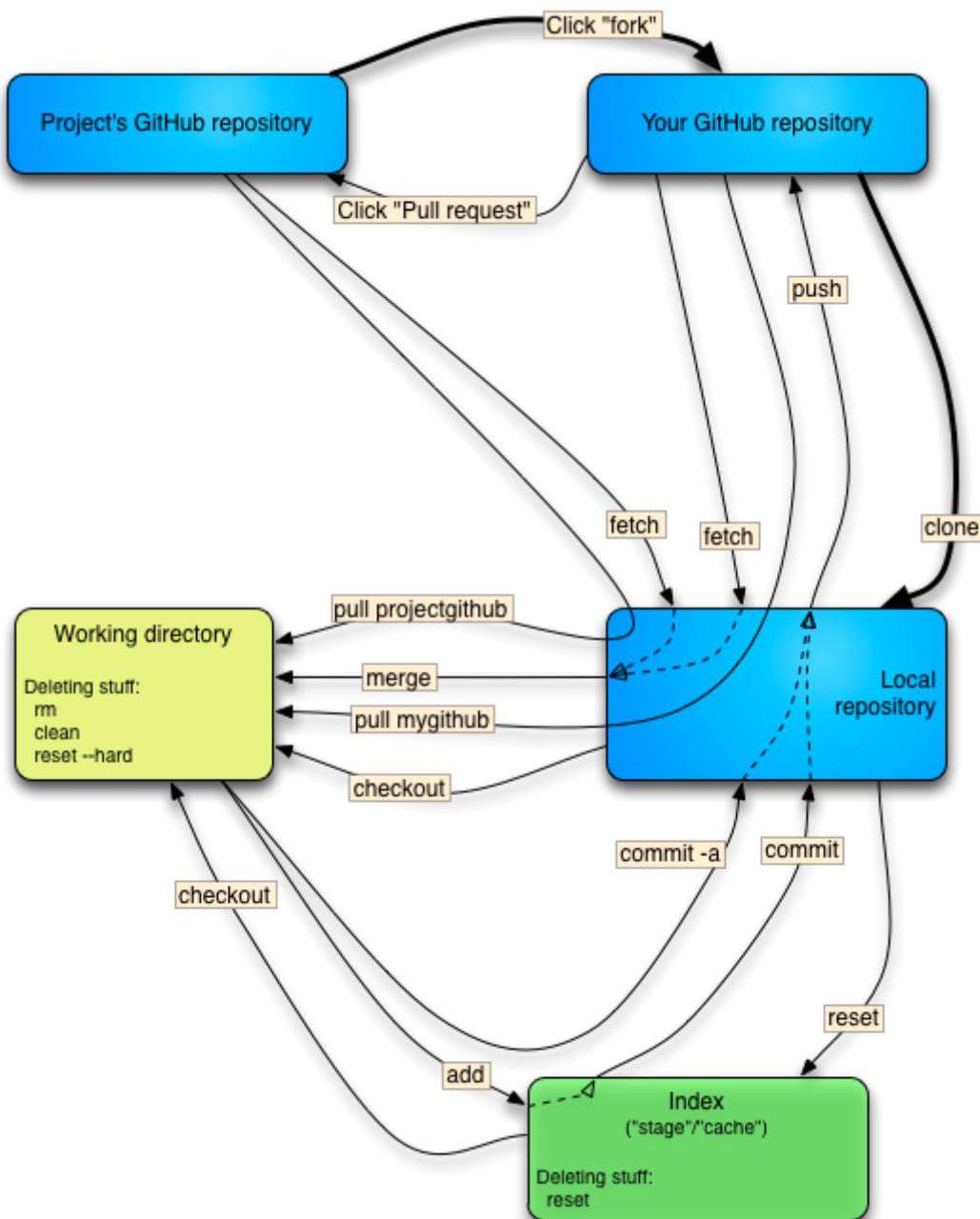
Um ótimo site para entender o funcionamento dos comandos remotos do Git de forma visual é o "Tracking Branches" And "Remote-Tracking Branches"

(<http://www.gitguys.com/topics/tracking-branches-and-remote-tracking-branches/>)

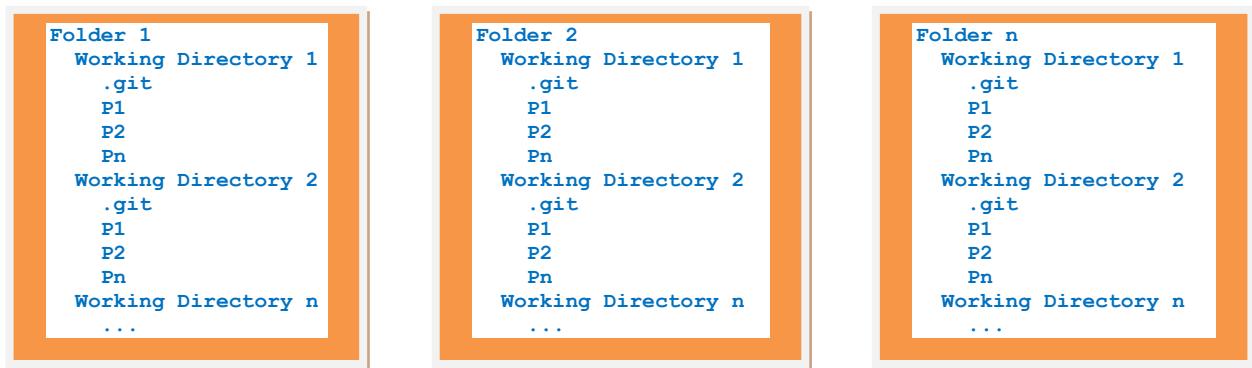
Uma referência visual completa incluindo comandos locais e remotos:

- Reppositório local;
- Um repositório remoto, hub do usuário. É a partir deste hub que ele atualiza o hub do projeto;
- Um repositório remoto, hub do projeto.

Neste site (<http://steveko.wordpress.com/2012/02/24/10-things-i-hate-about-git/>), além do belíssimo resumo abaixo, o autor critica alguns aspectos do Git, principalmente a complexidade de se fazer tarefas simples.



7.4 - O repósitório do Git do disco local c:\



Folder (ou Pasta): localização a partir da qual a estrutura Git será criada. Podemos ter 1 ou mais Folders criados / instalados. O Folder pode ter qualquer nome.

Working Directory: é o diretório de trabalho do Git. Ele contém a pasta .git e os projetos. Podemos ter 1 ou mais Working Directory por Folder. O Working Directory pode ter qualquer nome.

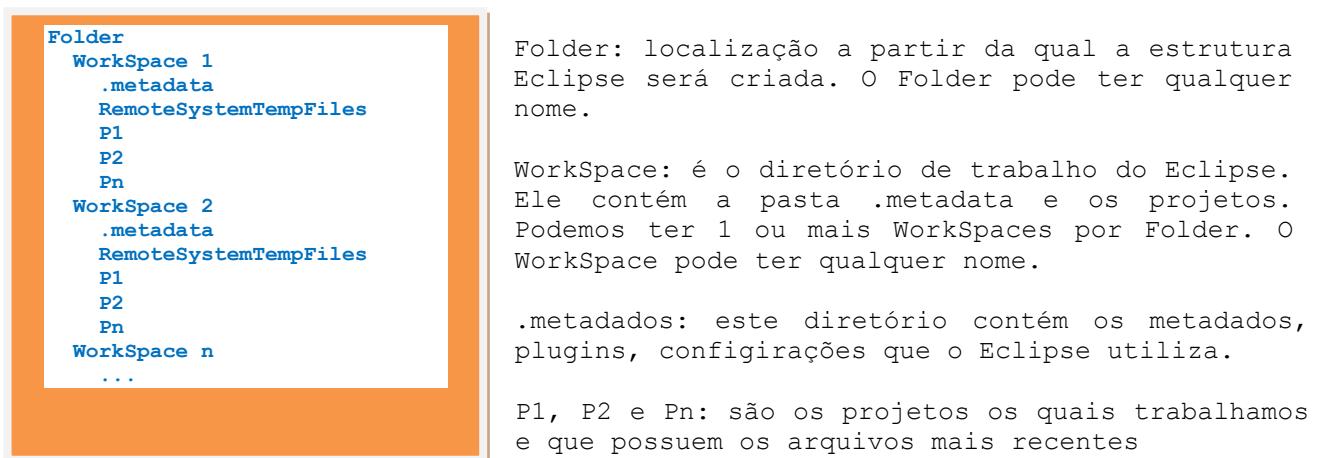
.git: este diretório contém os metadados, tabelas e índices que o Git utiliza.

P1, P2 e Pn: são os projetos os quais trabalhamos e que possuem os arquivos mais recentes. Equivale ao WorkSpace do Eclipse.

7.5 - Integração Git e IDE (no caso, o Eclipse)

Como já vimos antes, o JBDS foi desenvolvido sobre o Eclipse. Na literatura encontraremos referência para o Eclipse Git, ou EGit. No anexo 8 falaremos sobre o EGit, porém antes é bom diferenciar os dois ambientes no que tange aos seus repositórios. O JBDS / Eclipse trabalha com o conceito de WorkSpace e o Git, como visto no item anterior, com o conceito de Working Directory.

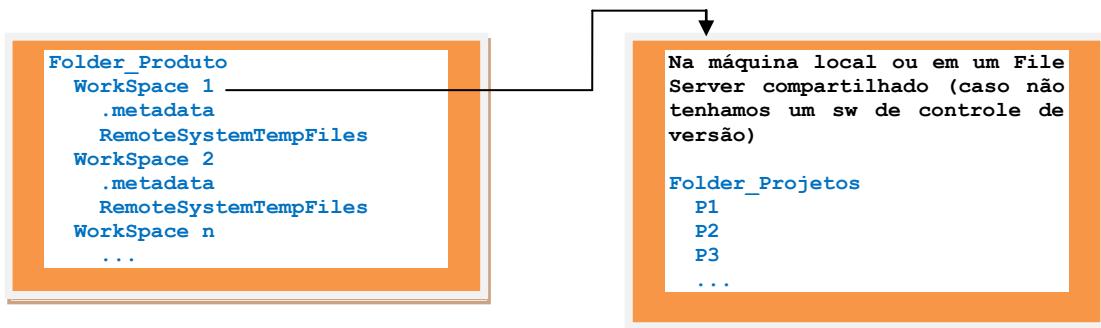
Estrutura do WorkSpace



Esta é a estrutura default e não é aconselhável por diversos motivos:

- Nesta forma os projeto não estão compartilhados;
- Se quisermos ter versões diferentes do eclipse instaladas;
- Os metadados são muito acessado e realizam concorrência com os projetos.

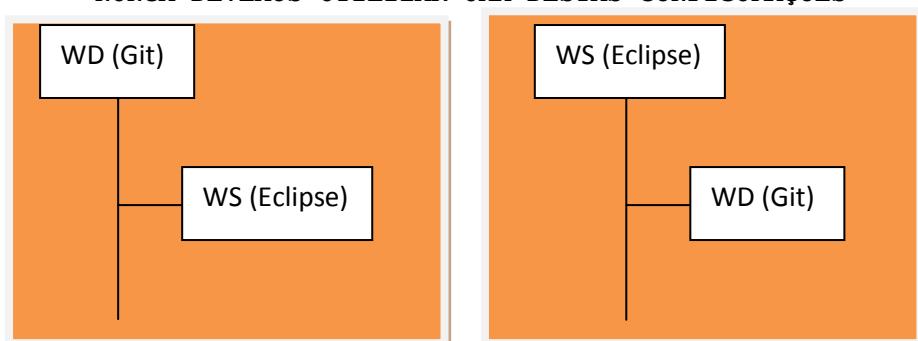
Desta forma, dá-se preferência para este tipo de estrutura:



Caso tenhamos também o Git, esta é a melhor estrutura:

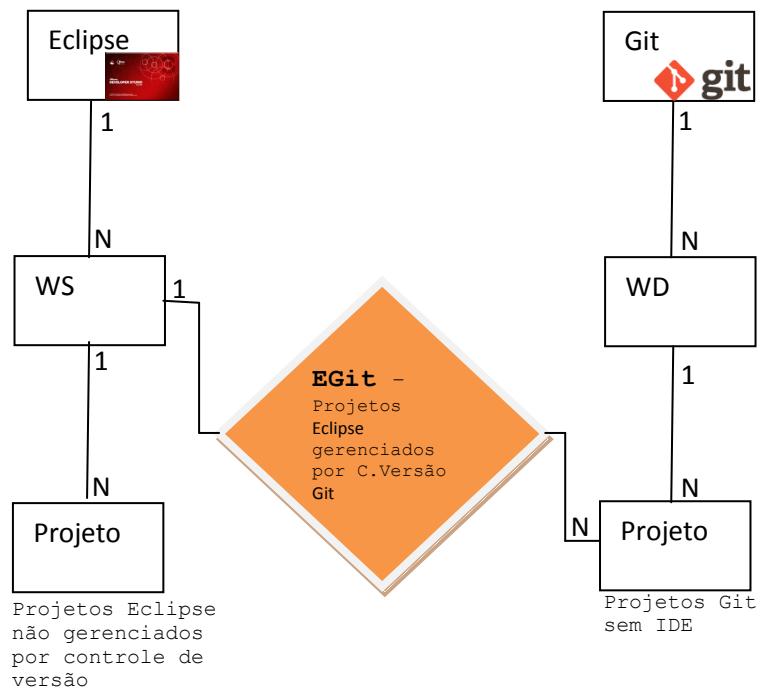


NUNCA DEVEMOS UTILIZAR UMA DESTAS CONFIGURAÇÕES



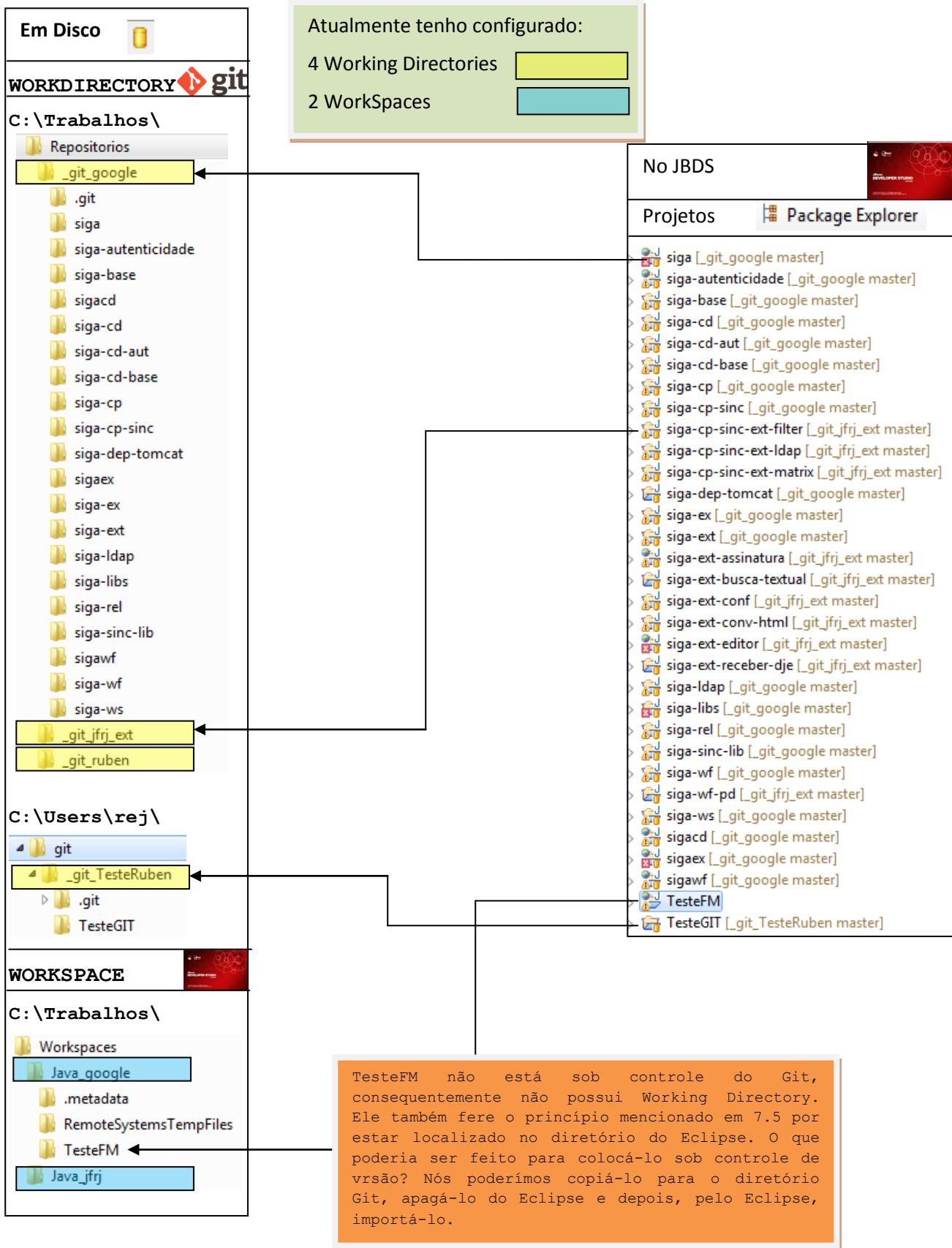
São inúmeros os motivos que vão desde a baixa performance a inconsistências diversas.

A partir destas informações, podemos desenhar o seguinte modelo para os dois produtos Eclipse e Git:



Para ilustrar os conceitos anteriores listei os repositórios Git e Eclipse da minha máquina, drive c:\.

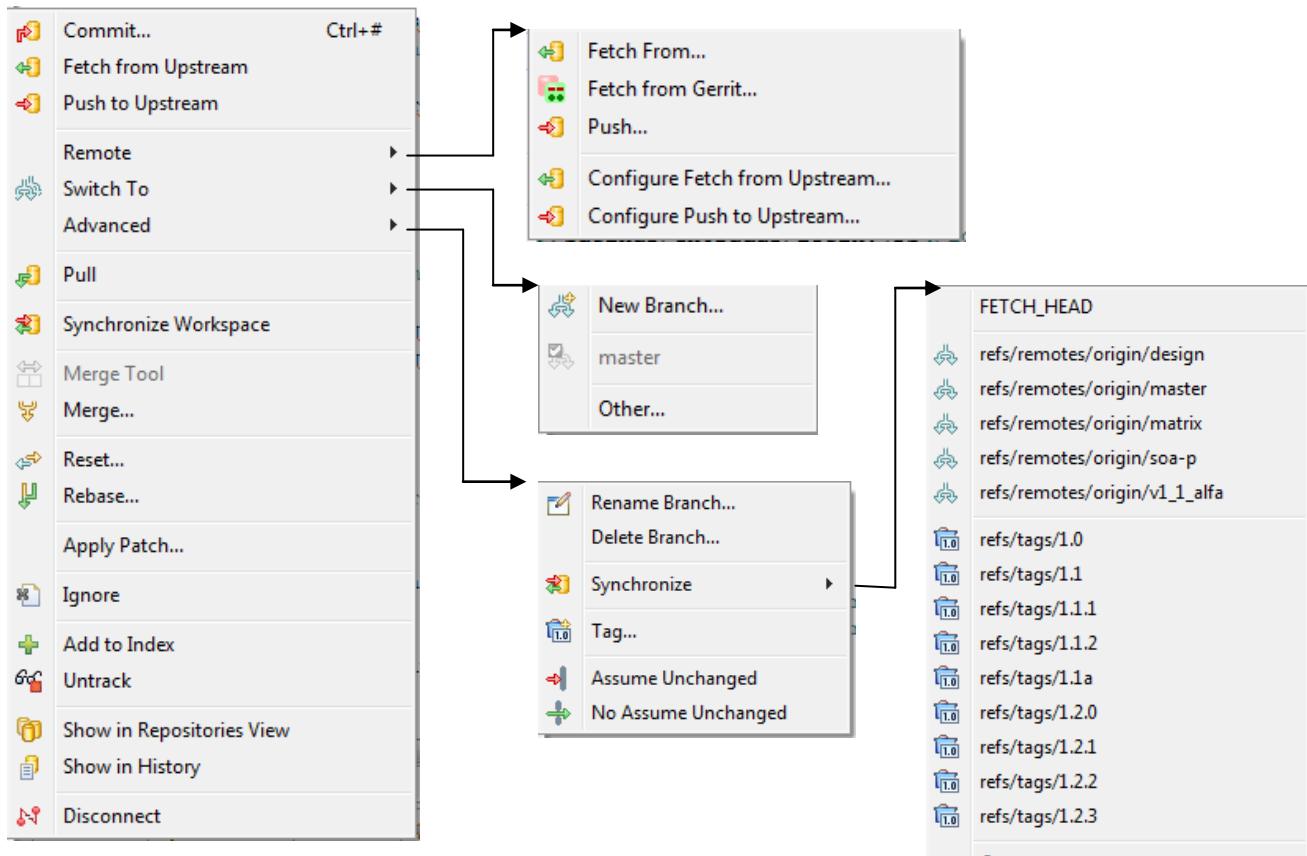
Caso Real: Relacionamento dos Repositórios Git X Eclipse



Anexo 8 – GIT no Eclipse/JBDS – EGit

8.1 – Visão geral pelo menu do produto

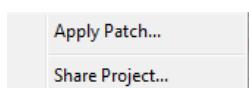
Quando se clica em qualquer projeto gerenciado pelo GIT, a seguinte árvore de menus é aberta.



8.2 – Projetos gerenciados e não gerenciados pelo GIT

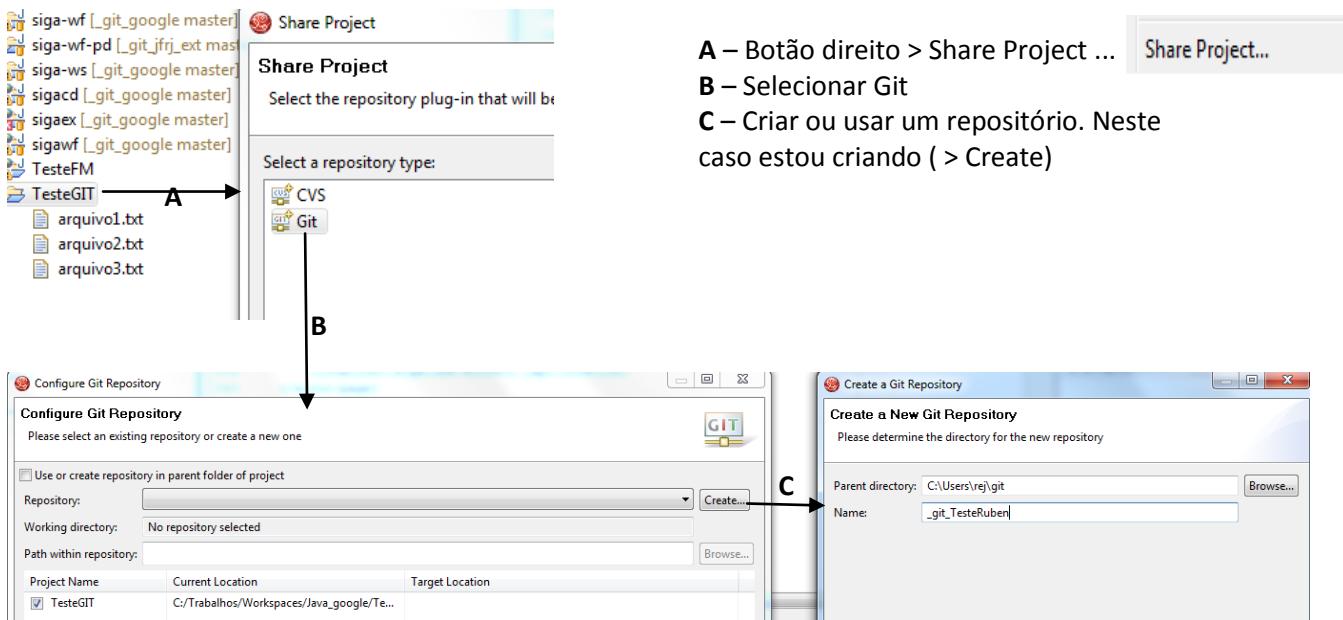
- ▷ `siga-wt [git_google master]`
 - ▷ `siga-wf-pd [git_jfrj_ext master]`
 - ▷ `siga-ws [git_google master]`
 - ▷ `sigacd [git_google master]`
 - ▷ `sigaex [git_google master]`
 - ▷ `sigawf [git_google master]`
 - ▷ `TesteFM`
 - ▷ `TesteGIT`
- }
- Gerenciado, informando o repositório e o branch entre [], ou seja, [repositório branch]. O head do Git está apontando para o branch master.
- }
- Não gerenciado, sem repositório

Se colocarmos o cursor em TesteGIT, as seguintes opções serão fornecidas, visto que o projeto não está ainda gerenciado.

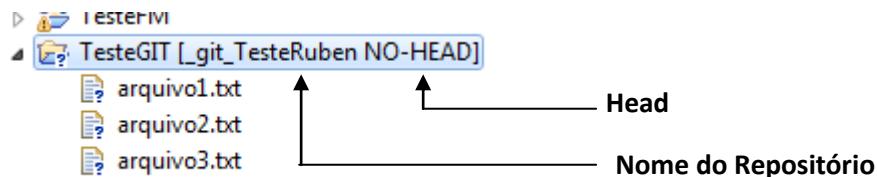


8.3 - SHARE PROJECT (Criando um repositório)

Vamos colocar o projeto testeGIT gerenciado pelo GIT.

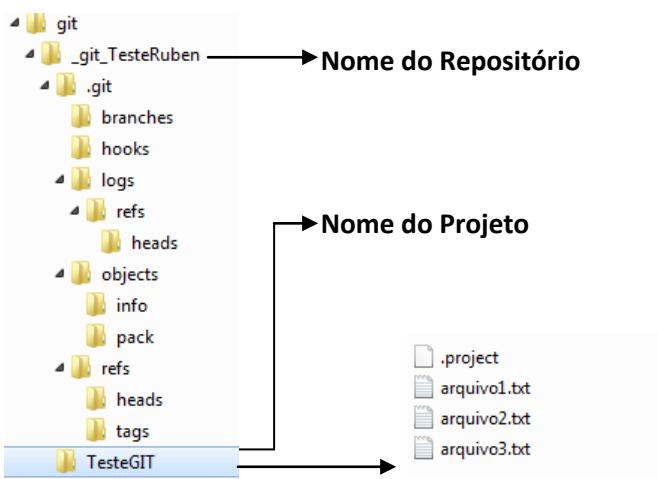


Após isso, o projeto passa a ser gerenciado:



Observar que:

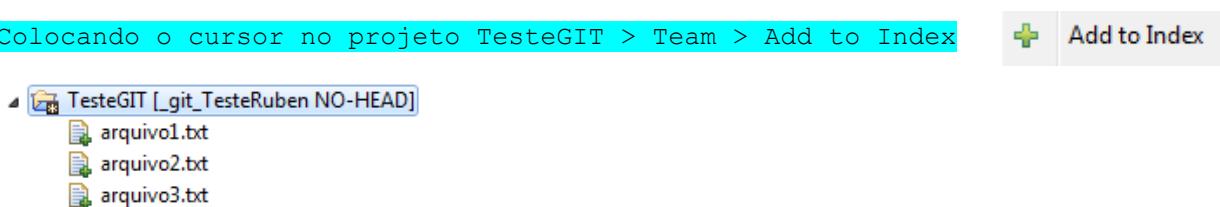
- a - não temos ainda um head porque não fizemos ainda nenhum commit e não temos um branch;
- b - ? nos arquivos, informando que os mesmos não estão ainda sob controle de versão;
- c - como está tudo fisicamente no disco (em `c:\users\rej\git`)



8.4 – ADICIONANDO O PROJETO / ARQUIVOS PARA CONTROLE DE VERSÃO

Como vimos anteriormente, os arquivos do projeto TesteGIT não estão sob controle de versão, e desta forma, devemos adicioná-lo.

Colocando o cursor no projeto TesteGIT > Team > Add to Index

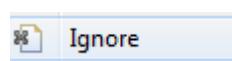


Observar que o ícone mudou de para informando que o projeto está sob controle de versão.

8.5 – RETIRANDO ARQUIVOS DO CONTROLE DE VERSÃO

E se não quisermos que um arquivo / diretório (como o /bin por exemplo) fique sob o controle de versão do GIT. É possível? Sim, utilizando-se a opção Ignore ...

Coloque o cursor no arquivo / diretório > Team > Ignore



Ou, crie um arquivo chamado **.gitignore** na raiz do projeto e coloque o nome do diretório (ex: /bin) ou arquivo (ex: /bin/xpto.class) a ser ignorado



OK, porém não consigo ver o **.gitignore** ... o que fazer? O problema é que o eclipse / JBDs deve estar configurado para não exibir arquivos começando com **..**, e nesta caso proceda conforme ilustrado abaixo:

The image shows two windows from the Eclipse IDE. On the left is the 'Package Explorer' view, which displays a project named 'TesteGIT [_git_Testeruben NO-HEAD]' containing files 'arquivo1.txt', 'arquivo2.txt', and 'arquivo3.txt'. A black arrow labeled 'A' points from the text 'Selecionar view' to the 'Filters...' option in the 'View' menu of the Package Explorer. Another black arrow labeled 'B' points from the 'Filters...' option to the 'Java Element Filters' dialog window on the right. This dialog has a list of checkboxes under 'Select the elements to exclude from the view:' with several items checked, including '.resources', 'Empty library containers', 'Empty packages', 'Empty parent packages', 'External JavaScript Source', 'External plug-in libraries project', 'Fields', 'Import declarations', and 'Inner class files'. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

A – No **Package Explorer > view > Filters**

B – Desmarcar **.*resources**

C – Observar que o **.project** apareceu na árvore do projeto

8.6 – COMMITANDO O PROJETO

Vamos agora criar o primeiro commit do projeto.

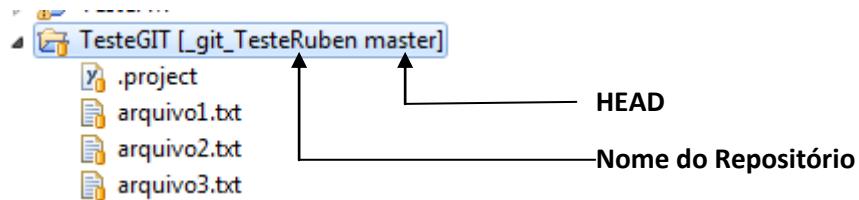
Cursor no projeto > team > Commit



Observar o Add Signed-off-by, que clicado inclui na mensagem do commit.

Autor: pode-se mudar o nome do autor, informando que o committer está fazendo o commit em nome de terceiro

Como resultado, teremos:

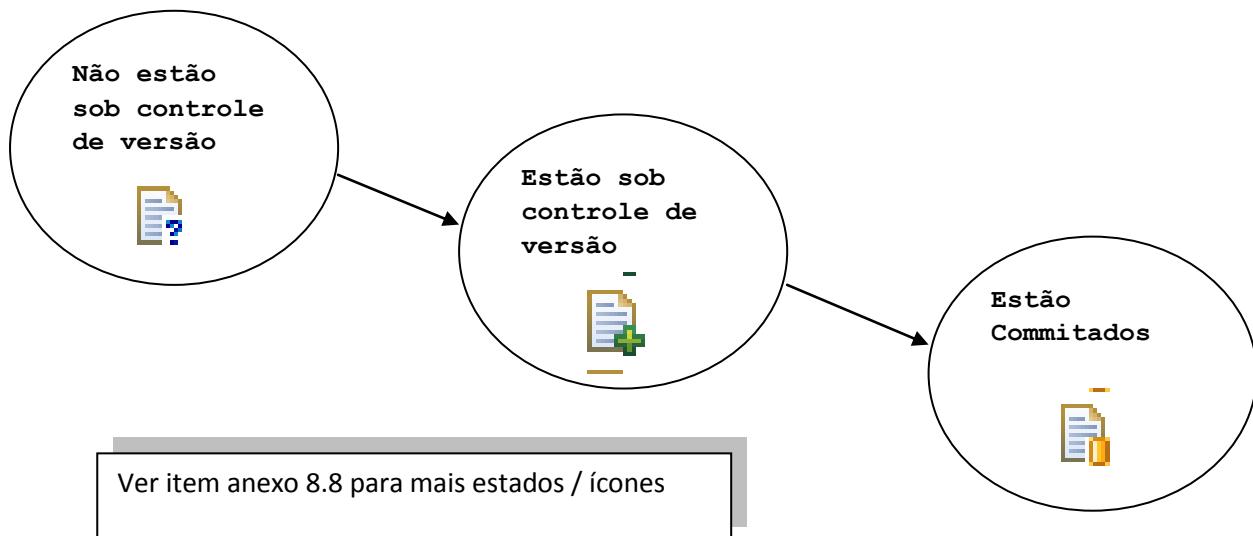


Observar que:

- a - após o commit, o ícone do projeto mudou de o projeto já foi committed. O commit representa incluindo seu conteúdo e sua história;
- b - já temos um HEAD apontando para o branch default master.

para informando que uma revisão de um projeto,

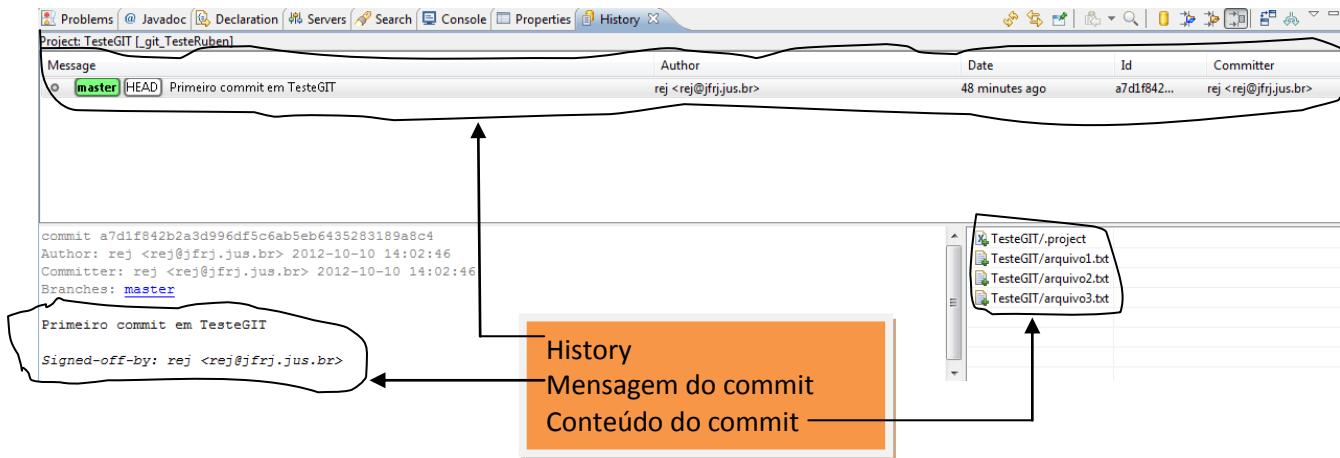
Transição de estado dos recursos do projeto conforme o GIT:



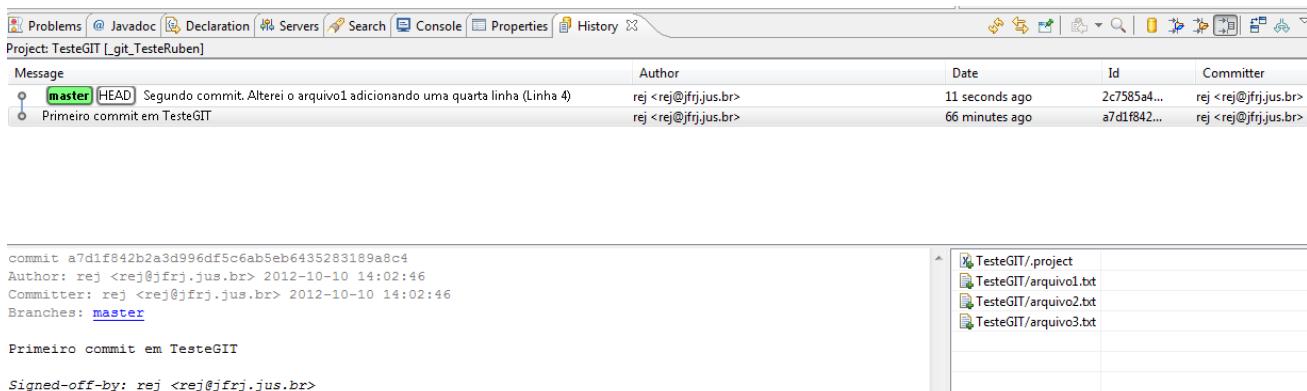
8.7 – HISTÓRIA DO PROJETO

Cursor no projeto > Team > Show in History

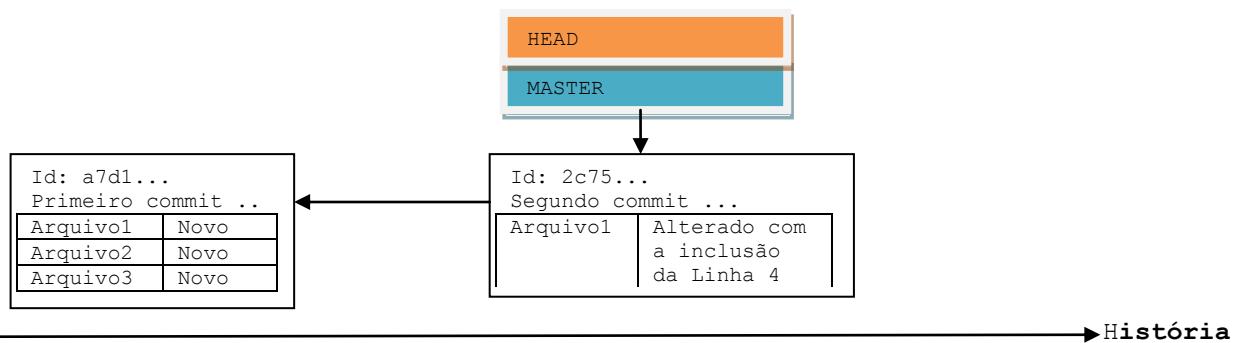
Show in History



Farei uma modificação no arquivo1.txt, adicionando uma quarta linha (Linha 4) e commitarei as mudanças.



Assim está, diagramaticamente, o projeto para o GIT:



O que podemos fazer agora que temos 2 versões do projeto?

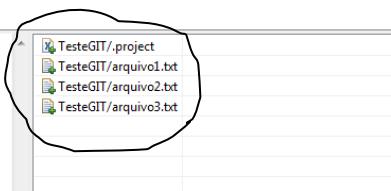
8.7.1 - Clicando no primeiro commit

Message	Author	Date	Id	Committer
master [HEAD] Segundo commit. Alterei o arquivo1 adicionando uma quarta linha (Linha 4)	rej <rej@jfrj.jus.br>	79 minutes ago	2c7585a4...	rej <rej@jfrj.jus.br>
Primo commit em TesteGIT	rej <rej@jfrj.jus.br>	2 hours ago	a7d1f842...	rej <rej@jfrj.jus.br>

```
commit a7d1f842b2a3d996df5c6ab5eb6435283189a8c4
Author: rej <rej@jfrj.jus.br> 2012-10-10 14:02:46
Committer: rej <rej@jfrj.jus.br> 2012-10-10 14:02:46
Branches: master

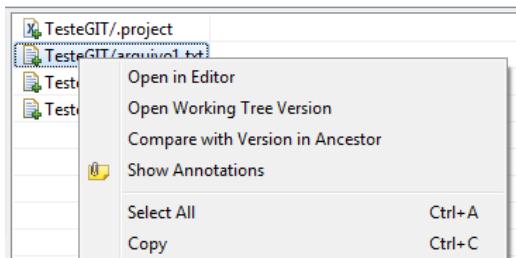
Primeiro commit em TesteGIT

Signed-off-by: rej <rej@jfrj.jus.br>
```

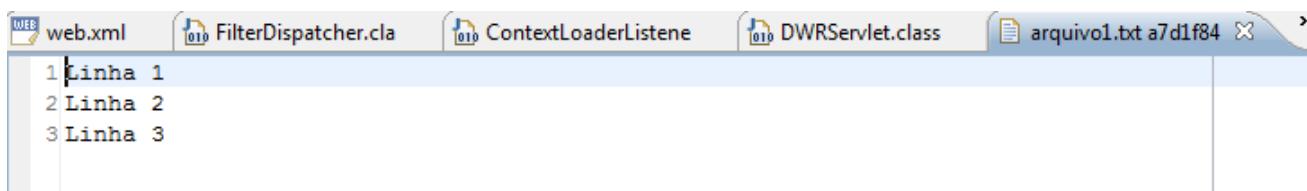


Observamos todos os arquivos que estão nesta versão.

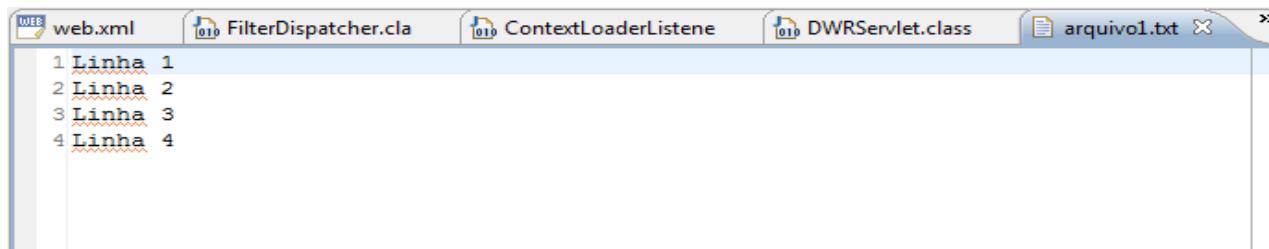
Ao clicarmos no arquivo1.txt teremos as seguintes opções:



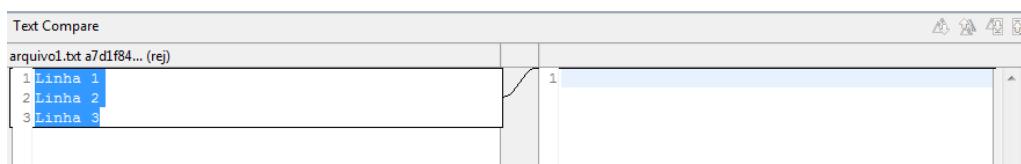
Open in Editor: aparecerá a versão antiga e no editor aparecerá arquivo1.txt at a7d1... (onde a7d1... é o hash sha-1 do primeiro commit).



Open Working Tree Version: aparecerá a versão mais nova e no editor aparecerá arquivo1.txt.



Compare with Version in Ancestor: compara com a versão do ancestral, porém como não tínhamos nenhum commit antes, não temos o ancestral e a janela da versão anterior aparecerá vazia.



8.7.2 – Clicando no segundo commit

Project: TesteGIT [.git_ TesteRuben]

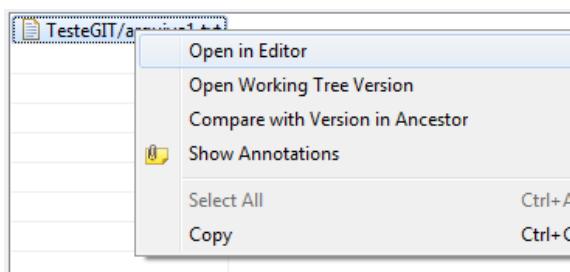
Message	Author	Date	Id	Committer
master [HEAD] Segundo commit. Alterei o arquivo1 adicionando uma quarta linha (Linha 4)	rej <rej@jfrj.jus.br>	2 hours ago	2c7585a4...	rej <rej@jfrj.jus.br>
Primeiro commit em TesteGIT	rej <rej@jfrj.jus.br>	4 hours ago	a7d1f842...	rej <rej@jfrj.jus.br>

```
commit 2c7585a4ee7c7478add62d6b1110c8adf76f621b
Author: rej <rej@jfrj.jus.br> 2012-10-10 15:08:42
Committer: rej <rej@jfrj.jus.br> 2012-10-10 15:08:42
Parent: a7d1f842b2a3d996df5c6ab5eb6435289189a8c4 (Primeiro commit em TesteGIT)
Branches: master

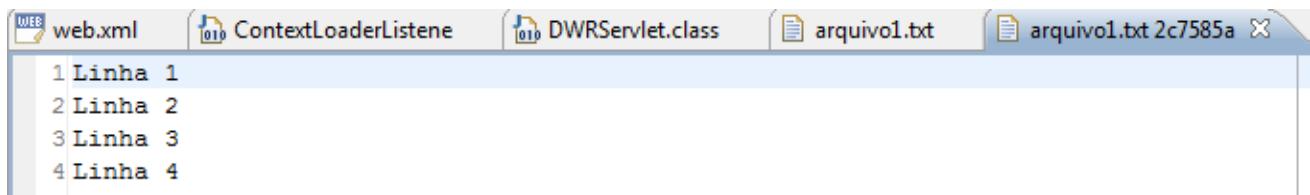
Segundo commit. Alterei o arquivo1 adicionando uma quarta linha (Linha
4)
```

Observamos todos os arquivos que estão nesta versão.

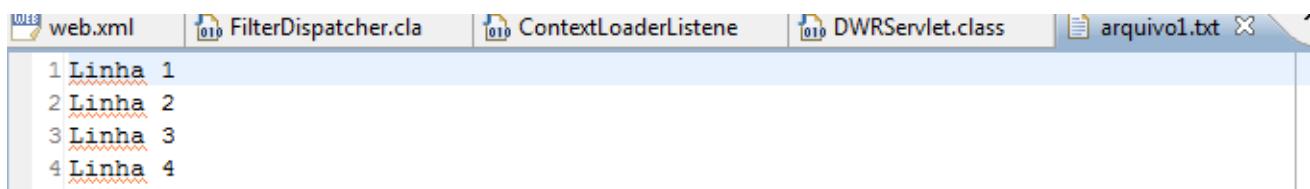
Ao clicarmos no arquivo1.txt teremos as seguintes opções:



Open in Editor: aparecerá a versão nova e no editor aparecerá arquivo1.txt at 2c75... (onde 2c75... é o hash sha-1 do segundo commit).



Open Working Tree Version: aparecerá a versão mais nova e no editor aparecerá arquivo1.txt.



Compare with Version in Ancestor: compara com a versão do ancestral.

Versão mais atual (2c75...)

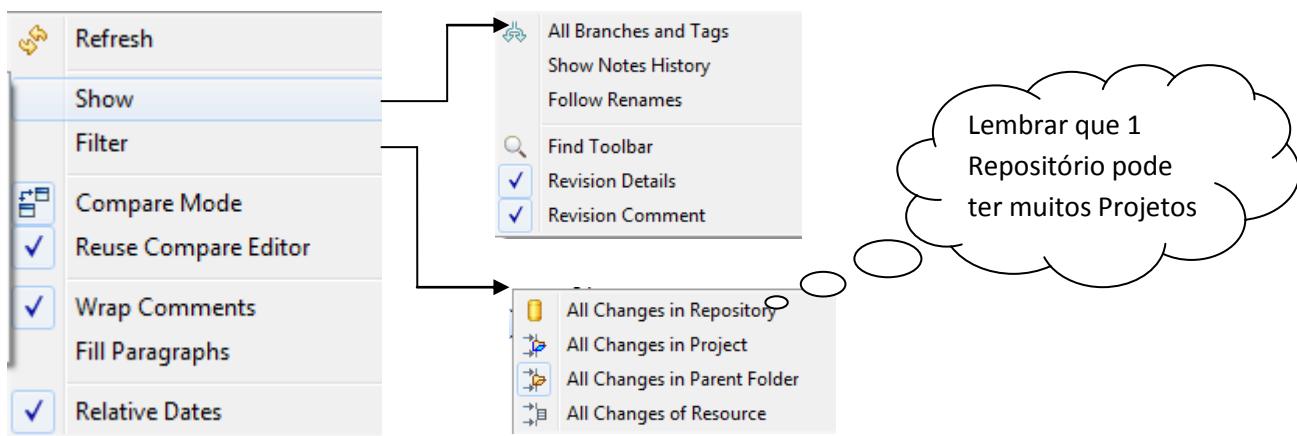
Text Compare

arquivo1.txt 2c7585a... (rej)	arquivo1.txt a7d1f84... (rej)
1 Linha 1	1 Linha 1
2 Linha 2	2 Linha 2
3 Linha 3	3 Linha 3
4 Linha 4	

Versão anterior (a7d1...)

△ □ ▲ ▲

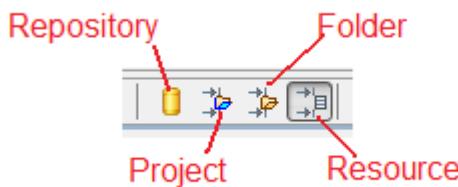
8.7.3 – Outras Opções quando analisando o History de um projeto



Para estes exemplos utilizarei os projetos do SIGA como referência, visto que eles possuem uma história mais complexa, e consequentemente mais interessante, para fins de consulta do que o nosso minúsculo projeto testeGIT.

FILTER:

Vamos supor que queremos analisar a classe FuncoesEL.java. Quantas vezes foi alterada? Quando? Por quem?



Selecionando um recurso (neste caso, FuncoesEL.java):

Com o cursor no recurso (neste caso FuncoesEL.java) > Team > Show in History > View Menu >

All Changes in Repository

8.7.3.1 – Show All Changes in Repository containing the selected resource

If the "Repository" button is down, the commit log is not filtered and shows all commits reachable from the currently checked out branch (or all commits, see below about the "All Branches" action)

All Changes in Project

8.7.3.2 – Show All Changes in Project containing the selected resource

If the "Project" button is down, the commit log is filtered to show all commits which **affected any of the resources in the project containing the current input**.

All Changes in Parent Folder

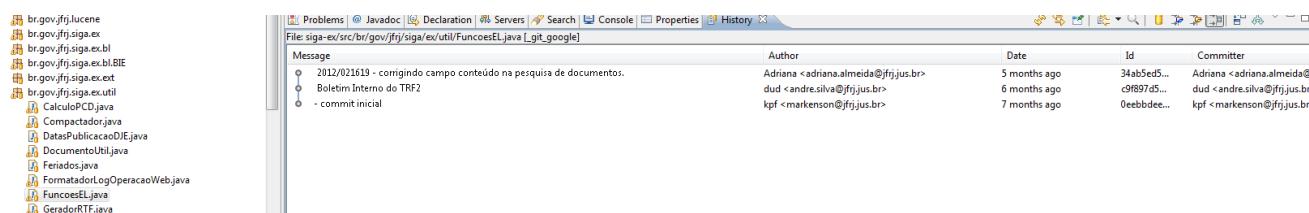
8.7.3.3 – Show All Changes in parent Folder of the selected resource

If the "Folder" toggle is down, the commit log is filtered to show all commits which **affected any of the resources in the parent folder of the current input**. Quem é o parent folder do FuncoesEL.java? É siga-ex/src/BR/gov/jfrj/siga/ex/util. Neste caso, serão listadas todas as mudanças que ocorreram em qualquer objeto deste folder, seja em FuncoesEL.java, seja em Notificador.java e etc.

All Changes of Resource

8.7.3.4 - Show All Changes of selected resource and its children

If the "Resource" button is down, the commit log is filtered to show only commits which **affected the current input**; the view menu item Show > Follow Renames allows to toggle whether renames of the selected ressource should be followed by this filter.



Observamos que:

- O recurso, FuncoesEL.java, foi alterado 3 vezes, com os autores e data acima exibidos;
- Caso o recurso existisse em mais de um projeto no repositório, seriam exibidas todas ocorrências.

SHOW:

Show Find Toolbar

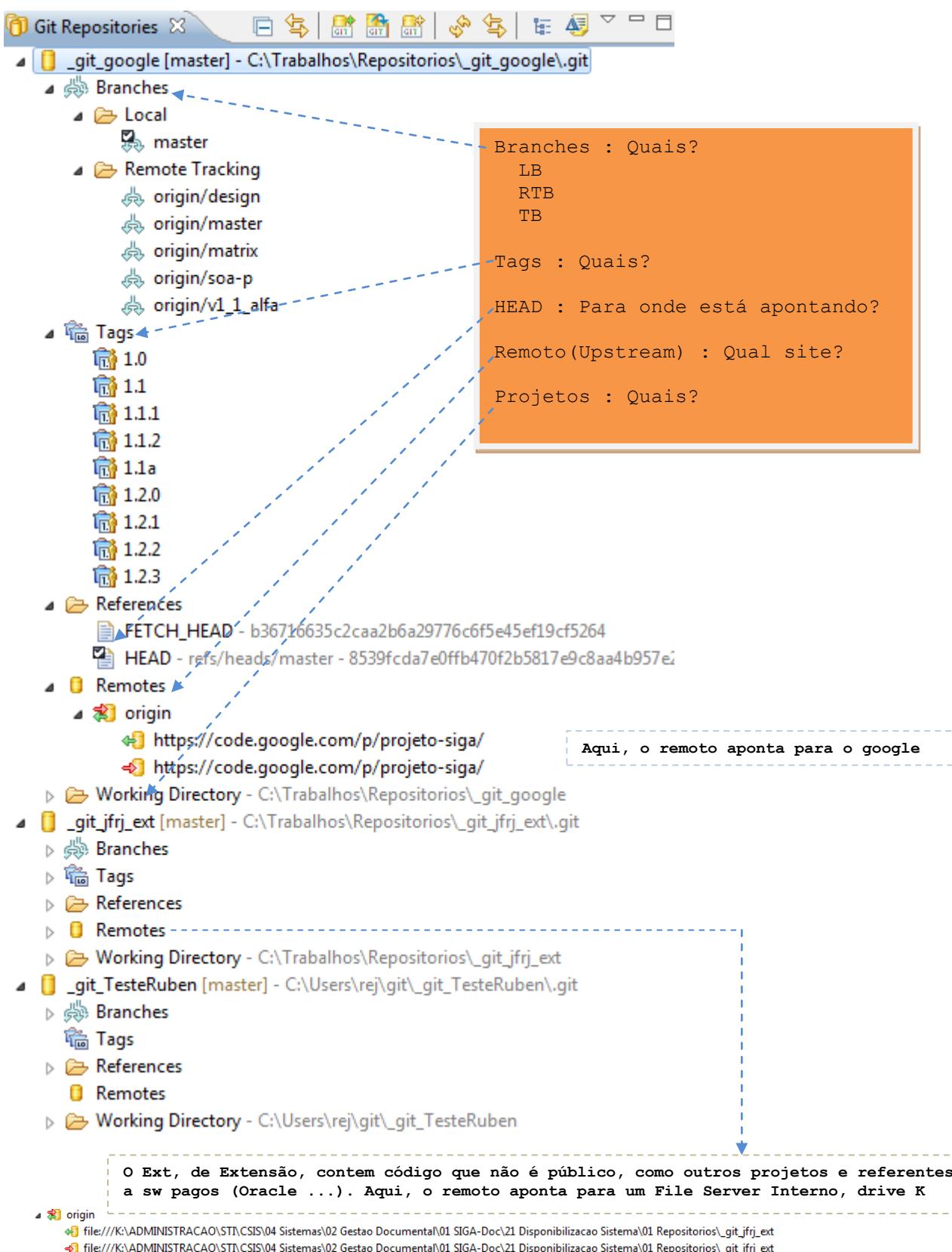
Permite pesquisar por autor, commiter, mensagem do commit e etc.



8.8 – Analisando Repositórios Git

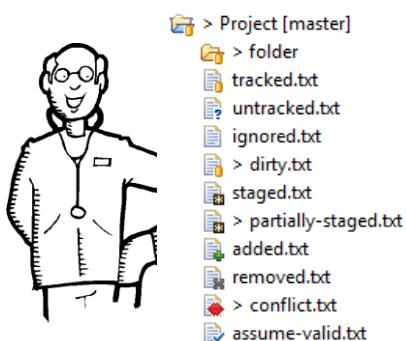
Temos uma View do Git por Re却itórios. Esta View abre outro menu referente aos Re却itórios. É interessante porque temos acesso a todos dados como branches, tags e enderecos dos sites remotos.

Cursor no Projeto > Team > Show In Repositories View



Inspecionando o estado do repositório:

O que significam todos aqueles ícones?



Observação:

Não é só o ícone que acusa problema.

> É sinal de problema



Mesmos ícones, porém o sinal de > indica problema.

dirty (folder) > folder	At least one file below the folder is dirty; that means that it has changes in the working tree that are neither in the index nor in the repository.
tracked tracked.txt	The resource is known to the Git repository and hence under version control.
untracked untracked.txt	The resource is not known to the Git repository and will not be version controlled until it is explicitly added.
ignored ignored.txt	The resource is ignored by the Git team provider. The preference settings under Team > Ignored Resources , "derived" flag and settings from .gitignore files are taken into account.
dirty > dirty.txt	The resource has changes in the working tree that are neither in the index nor in the repository.
staged staged.txt	The resource has changes which have been added to the index. Note that adding changes to the index is currently possible only in the commit dialog via the context menu of a resource.
Partially-staged > partially-staged.txt	The resource has changes which are added to the index and additional changes in the working tree that neither reached the index nor have been committed to the repository.
added added.txt	The resource has not yet reached any commit in the repository but has been freshly added to the Git repository in order to be tracked in future.
removed removed.txt	The resource is staged for removal from the Git repository.
conflict > conflict.txt	A merge conflict exists for the file.
Assume-valid assume-valid.txt	The resource has the "assume unchanged" flag. This means that Git stops checking the working tree files for possible modifications, so you need to manually unset the bit to tell Git when you change the working tree file. Also see Assume unchanged action

Lembrando que:

History

Contém a história (todas as versões) do repositório (projetos / arquivos). FILES COMMITED.

Stage (Index)

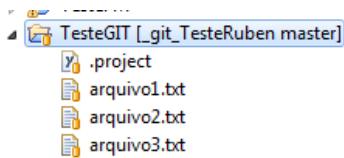
Conterá todos os arquivos modificados no diretório de trabalho e que estão prontos para ser commitados. É uma área intermediária. FILES TO GO IN NEXT COMMIT.

Working Directory

Diretório de Trabalho. Conterá os projetos / arquivos que o desenvolvedor está trabalhando. São estes projetos / arquivos que serão expostos a IDE Eclipse ou JBDs. FILES THAT YOU SEE. Também referenciado como WorkSpace e Working Tree

Exemplo:

No nosso projeto de teste, TesteGIT



Alterei o arquivo2.txt, adicionando uma quarta linha



A imagem informa que o arquivo2.txt está sujo, ou seja, foi alterado no WorkSpace (Working Tree ou Working Directory), e está diferente da versão do Stage (ou Index) e/ou do repositório (História)

8.9 – Sincronização

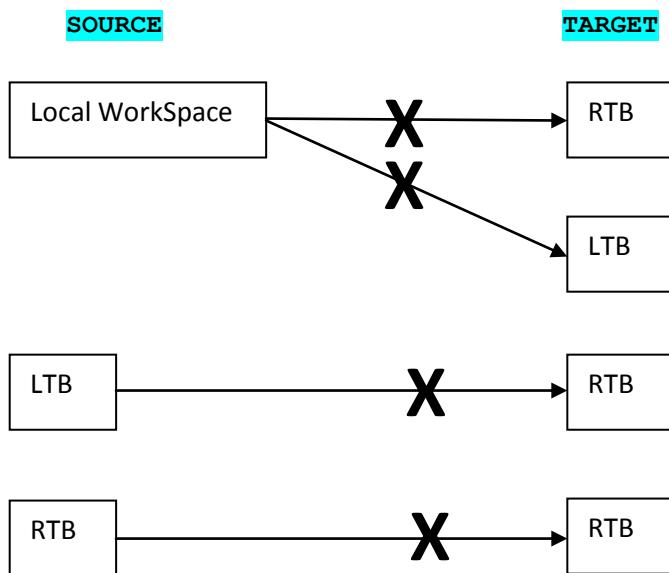
Permite Comparar os recursos entre:

Local WorkSpace

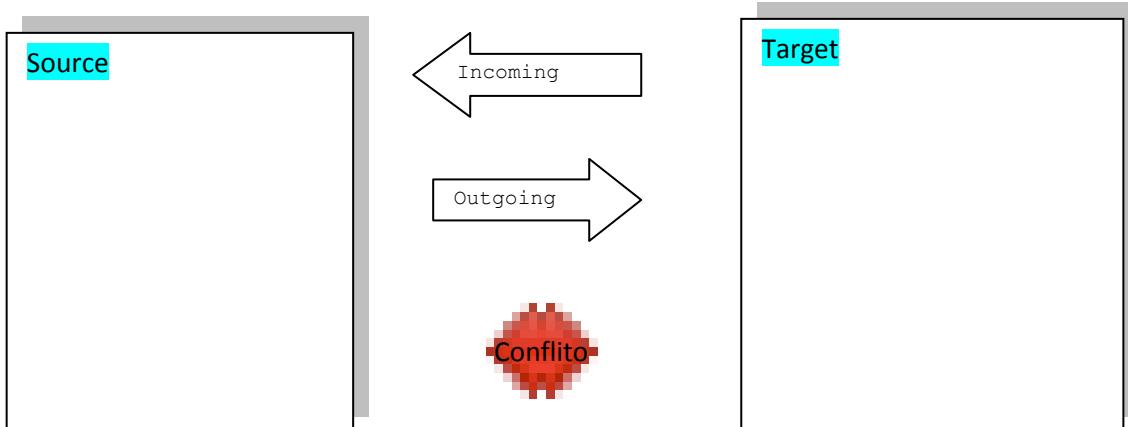
RTB: Remote Tracking Branch

LTB: Local Tracking Branch ou TB, Tracking Branch

Quais combinações são possíveis?



Que tipo de informações são geradas?



Em que granularidade?

- + Adição do recurso
- Exclusão de recurso
- Alteração de recurso

Exemplo: Significa que um recurso existe (foi adicionado) no Target e Não existe no Source

Tabela completa de inconsistências

Ícone	Descrição
	INCOMING: Existe no target e não no source An incoming addition means that a resource has been added to the target branch.
	An incoming change means that the file has changed in the target branch.
	An incoming deletion means that a resource was deleted from the target branch.
	OUTGOING: Existe no source e não no target An outgoing addition means that the file was added to your workspace or source branch and is not yet in the target branch.
	An outgoing change means that the file was changed in your workspace or source branch.
	An outgoing deletion is a resource that has been deleted in your workspace or source branch.
	CONFLITO: Existe no target e no source A conflicting addition means that the resource has been added in your workspace or source branch and in the target branch. A conflicting change means that the file has been changed in your workspace or local branch and in the target branch. A manual or automatic merge will be required. Also, any entries in the view that contain children that are conflicts will also be decorated with the conflict icon. This is done to make conflicts easy to find. A conflicting deletion means that the resource was deleted in your workspace or source branch and in the target branch.

Exemplo:

No final do item anexo 8.8 fizemos um exemplo. Agora vamos realizar o synchronize:

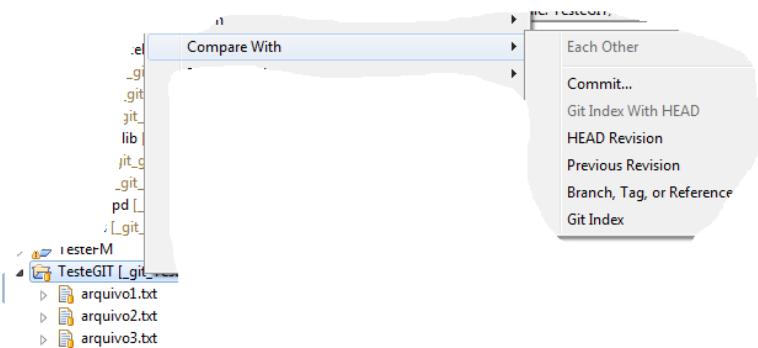
The screenshot shows the Eclipse IDE interface. At the top, there's a toolbar with a 'Synchronize Workspace' button. Below it, the 'Git (TesteGIT)' perspective is active, indicated by a tab bar. In the main workspace, there's a tree view showing a folder named 'TesteGIT [git_Testeruben master]' containing an 'arquivo2.txt' file. The 'arquivo2.txt' node is highlighted with a yellow icon, indicating it has been modified in the workspace but not yet pushed to the target branch.

A figura indica que o arquivo2.txt foi mudado no WorkSpace e não foi mudado no target (branch master).

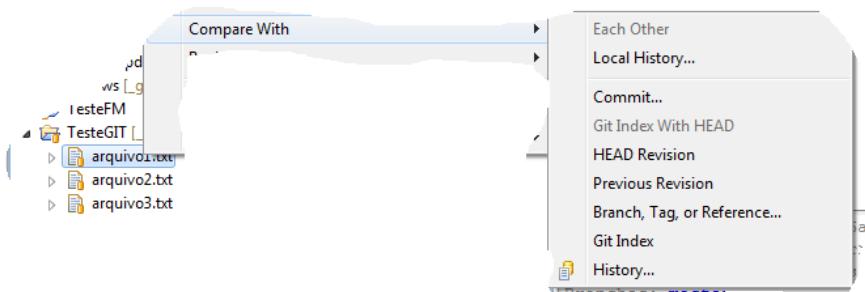
8.10 – Comparar Conteúdos – Compare with

Comparar versões (conteúdo) de recursos, tais como: arquivos, pastas ou projetos.

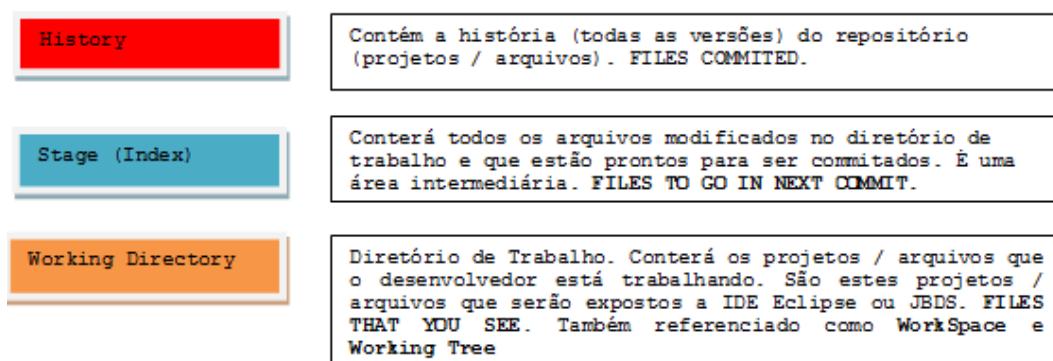
Com o cursor em um projeto:



Como o cursor em um arquivo:

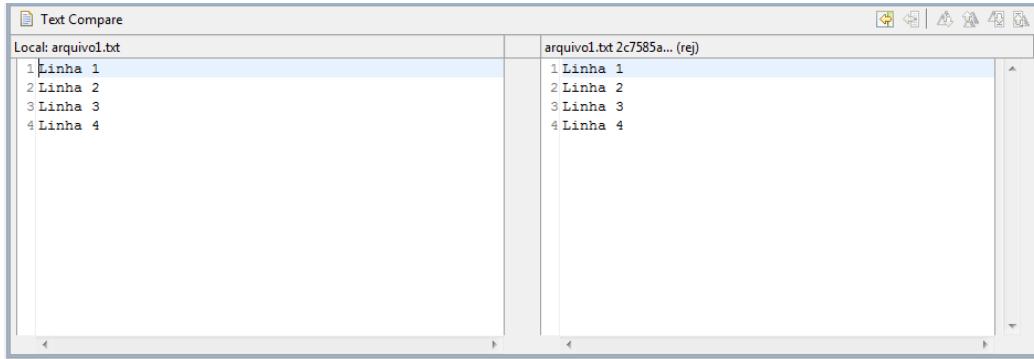


Permite comparar recursos pelas três camadas do Git:



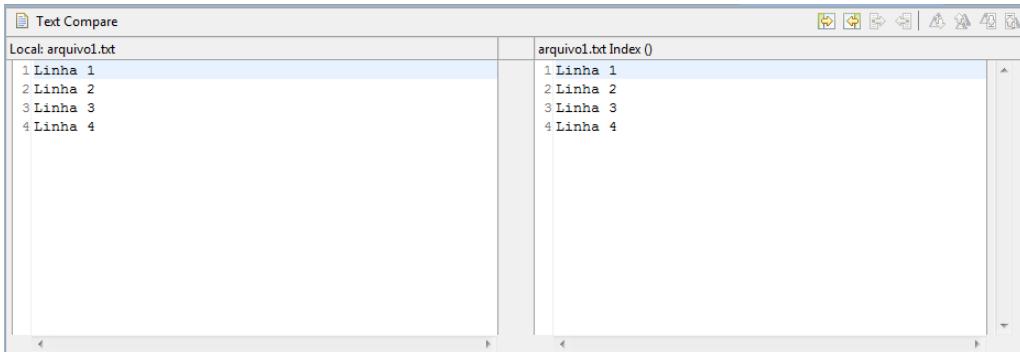
8.10.1 - Comparando a Working Tree com o último commit (HEAD Revision)

The difference between a resource in the current working directory and in the last commit in the current branch can be viewed from the context menu **Compare With > HEAD revision**. This feature is also available in the Commit dialog. Double clicking on an entry in the Commit dialog opens a compare dialog.



8.10.2 - Comparando Working Tree com Git Index

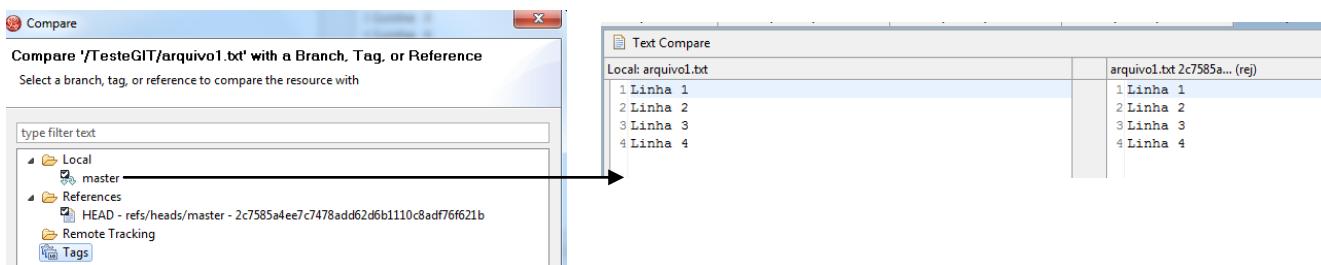
The differences between the current working tree and the index (based on the currently selected resource) can be viewed from the context menu **Compare With > Git Index**.



8.10.3 - Comparando Working Tree com um branch, uma tag ou uma reference

- Select a resource;
- right-click **Compare With > Branch, Tag, or Reference...**;
- select a branch, tag or reference.

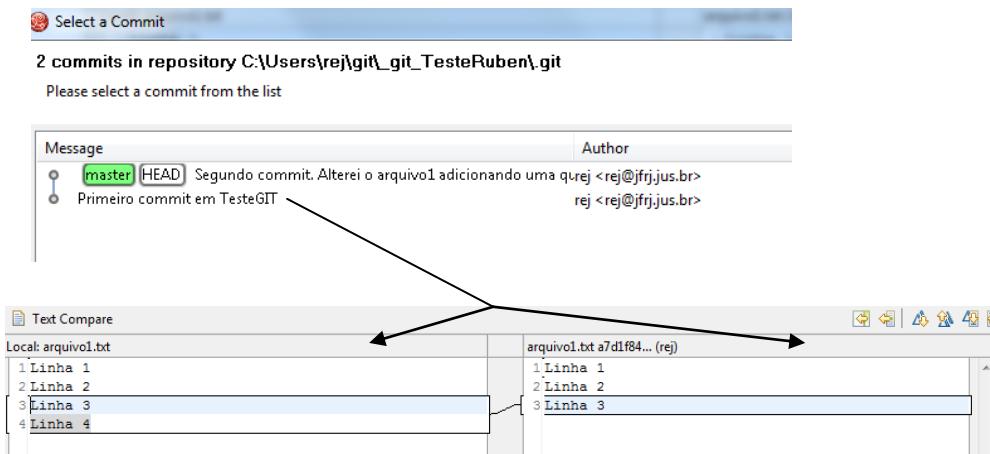
Será apresentada uma janela para você selecionar o Branch, Tag ou Reference



8.10.4 – Comparando Working Tree com qualquer Commit

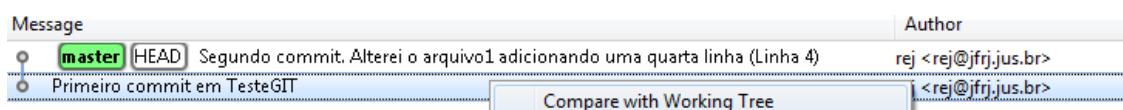
A partir do project explorer:

- Select a resource;
- right-click **Compare With > Commit...**;
- select a commit from the commit graph.



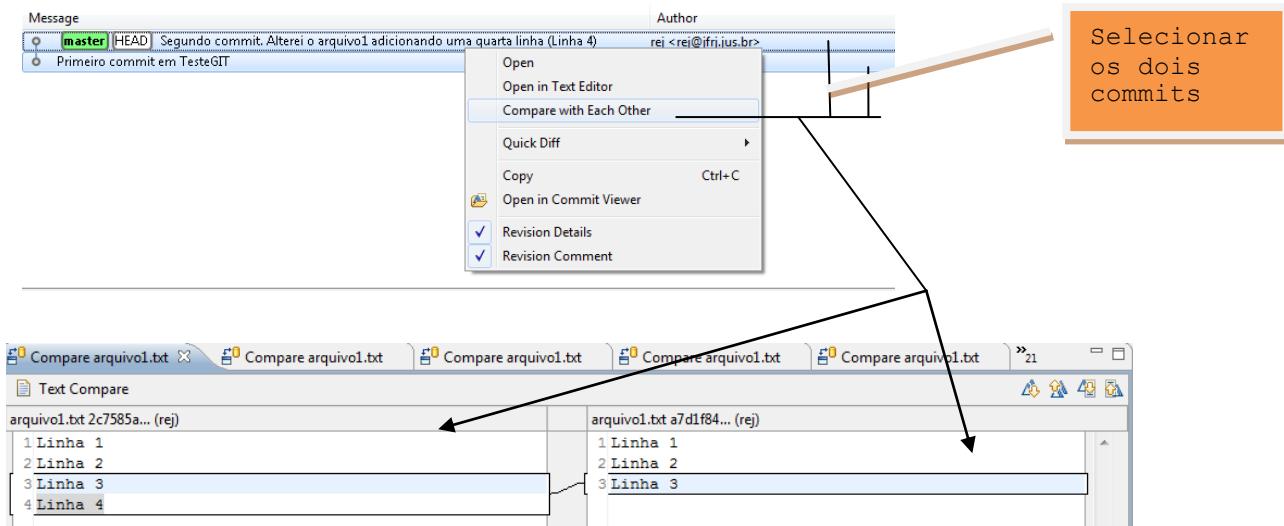
A partir da history view (arquivos somente):

- Select a file in the package explorer;
- right-click **Team > Show in History** or **Compare With > History...**;
- in the commit graph select a commit;
- from the context menu select **Compare with working tree**;
- this will open a compare dialog showing the changes between the selected commit and the current working tree.



8.10.5 – Comparando Dois Commits

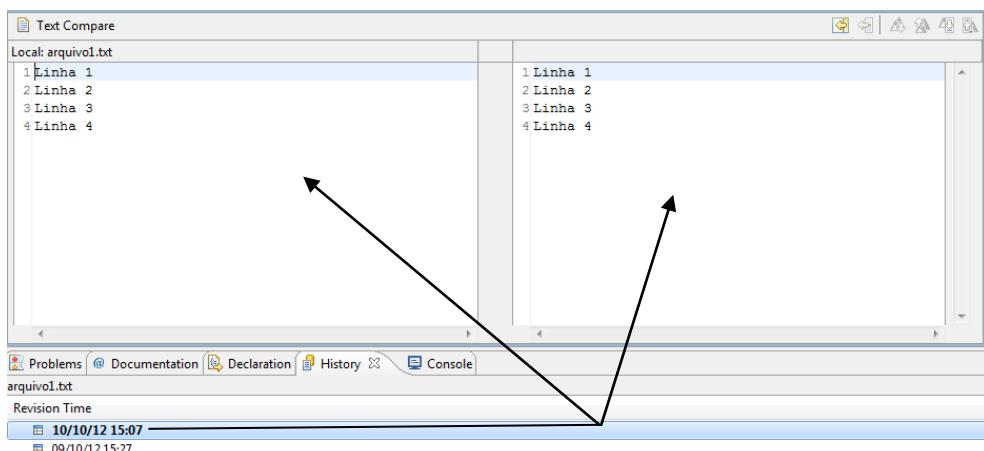
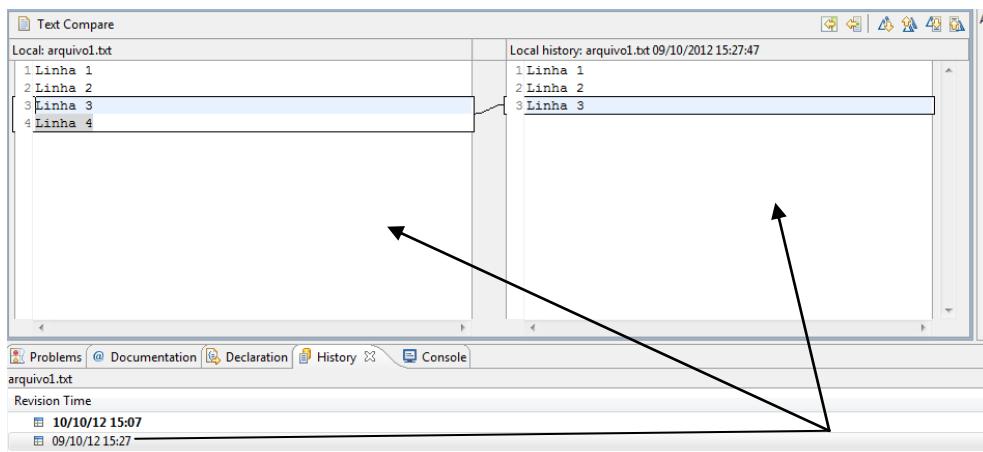
- Select a resource in the Package Explorer;
- click **Team > Show in History** or **Compare With > History...** (the latter for files only);
- in the commit graph select two commits;
- right-click **Compare with each other**;
- this will open a compare dialog showing the changes between the two selected commits;
- you can also open a Git Tree Compare view by right-clicking **Compare with each other in Tree**.



8.10.6 - Comparando Working Tree com Local History

- Select a file in the package explorer;
- **Compare With > Local History...**;

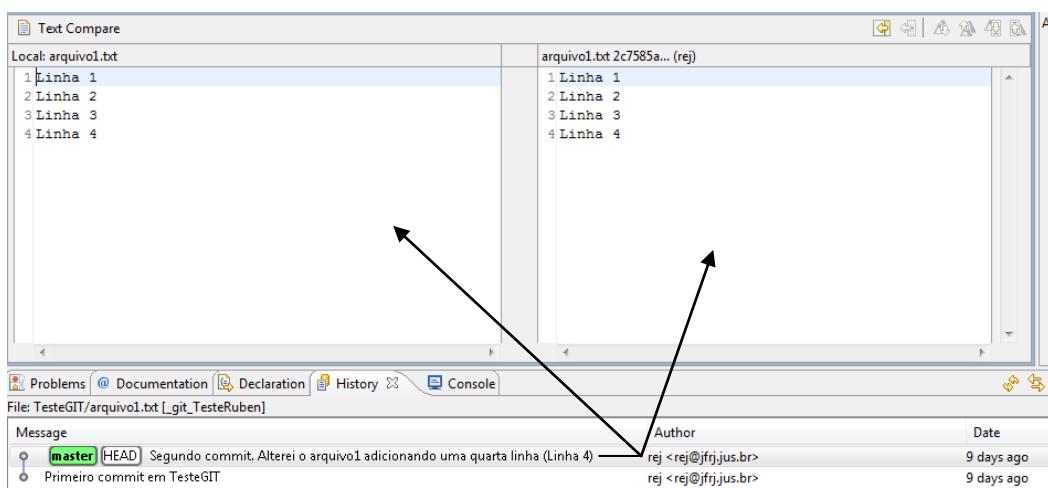
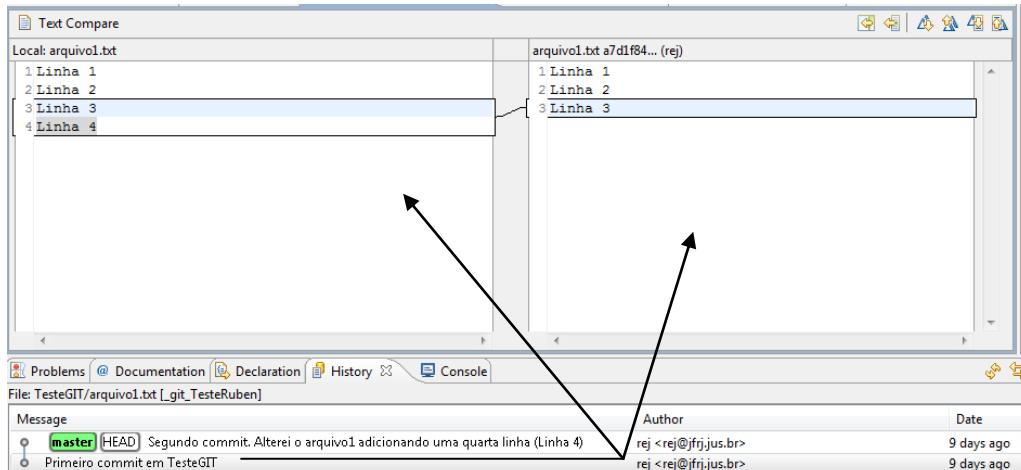
Será apresentado um Revision Time para que o usuário selecione uma revisão.



8.10.7 - Comparando Working Tree com History

- Select a file in the package explorer;
- Compare With > History...;

Será apresentado um Commit History para que o usuário selecione um commit.



8.10.8 - Comparando Index com HEAD ou qualquer outro Commit

Esta feature não foi ainda implementada.

Anexo 9 – Instalando o plugin do Freemarker no Eclipse

Note that new Eclipse releases can have compatibility problems. The most recent version we have tried this plugin with is: Eclipse 3.0. If you find that the plugin doesn't work on a later release, inform us.

You can install the eclipse plug-in either manually, or with the installer/updater system of Eclipse.

9.1 – Instalando manualmente

Visit <http://sourceforge.net/projects/freemarker>, and download the latest release of package eclipse-plugin. Extract the downloaded tar.gz into the plugins directory of Eclipse (so there will be 2 org.visigoths.freemarker* subdirectories), and then restart Eclipse.

To update the already installed version, just stop Eclipse, delete the two directories that were made when you have extracted the earlier tar.gz, and install the latest version as was explained above.

Once after the installation or after the update it's maybe good idea to start Eclipse from the command-line with the "clean" option: `eclipse -clean`

9.2 – Instalando através do Eclipse installer/updater system

Eclipse has its own on-line plug-in installer/updater system. To install the FreeMarker plug-in, be connected to the Internet, start Eclipse and:

If you use Eclipse 2.x:

1. Open the Window menu, then Open Perspective -> Install/Update
2. Click with the right mouse button on the Feature Updates view, then select New -> Site Bookmark
3. In the displayed dialog box, type "FreeMarker" for Name and "<http://www.freemarker.org/eclipse/update>" for URL. Leave the "Bookmark type" radio buttons on "Eclipse update site".
4. Click Finish
5. Open the tree node under the newly created update site named "FreeMarker", select the "FreeMarker X.Y.Z" feature, and install it using the Install now button in the preview pane.

If you use Eclipse 3.x:

1. Help -> Software updates -> Find and install....
2. Choose "Search for new features to install".
3. Click Add Update Site..., and type "FreeMarker" for Name and "<http://www.freemarker.org/eclipse/update>" for URL.
4. Check the box of the "FreeMarker" feature.
5. "Next"-s until it is installed...

From now on, you have **syntax highlight, syntax error reporting, and outline view for .ftl files.** (If for some reason Eclipse doesn't open *.ftl files with FreeMarker plugin, you have to associate *.ftl with the Freemarker Editor manually. For this use Window -> Preferences -> General -> Editors -> File Associations in Eclipse. Also in the tree views like the Resource Navigator or Package Explorer you can right-click on a file and choose "Open with..." and then "Freemarker Editor", however this decision will apply to that single file only.)

9.3 – Atualizando a versão do plugin

If you already have a previous version installed, do the following:

If you use Eclipse 2.x:

1. Open the Window menu, then Open Perspective -> Install/Update
2. In the Feature Updates view, open the tree node under the already existing update site named "FreeMarker", select the "FreeMarker X.Y.Z" feature, and install it using the Update Now button in the preview pane.

If you use Eclipse 3.x:

1. Help -> Software updates -> Manage Configuration....
2. Find the FreeMarker feature in the tree, and click Scan for Updates...

Important note regarding false errors in templates

The included freemarker.jar is often outdated; overwrite it with a newer version manually if you get error messages when using newer template language constructs in the edited template files.

Observação:

Alguns usuários na internet reclamaram que o procedimento acima não está muito correto.

How can I install Freemaker Plug-In in eclipse?

I googled it and found this site

<https://sites.google.com/site/hcao2008site/development/leareclipse/install-freemarker-plugin-in-eclipse> but it did not work for me. The URL "<http://www.freemarker.org/eclipse/update>" as it has mentioned in #3 **does not exist**. Then I tried the URL "<http://www.freemarker.org/eclipse/>" then **it worked** but after installation, I added *.ftl as file types inside Windows -> Preference -> General -> Editors -> File Associations but I could not find the Freemaker editor inside "Associated editors".
I am using Eclipse of Version: Indigo Service Release 1 in windows 7.

Any suggestions please???

E sugeriram:

You could try the **JBoss Tools** plugins available from <http://download.jboss.org/jbosstools/updates/development/indigo/>, they contain the "FreeMarker IDE" which provides a nice freemarker editor.
I didn't find a screenshot of the editor right now, so I will show you how it looks like in my installation (Indigo on Ubuntu 11.10), so that you can decide if this is what you want:

No site acima temos as instruções para instalação e os plugins:



JBoss Tools

Eclipse Plugins for JBoss technology.

JBoss Tools - Core - Stable Release Update Site

Latest Build: 3.3.2.v20130118-1803-H229-Final

This is the **Stable Release** Update Site for JBoss Tools - Core.

1. To [install](#) from this site, start up Eclipse 3.7, then do:
Help > Install New Software... >
2. Copy this site's URL into Eclipse, and hit Enter.
3. When the site loads, select the features to install, or click the Select All button.
4. To properly resolve all dependencies, check
[x] Contact all update sites during install to find required software
5. Click Next, agree to the license terms, and install.

You can also download JBoss Tools as individual zips for offline installation. See [JBoss Tools Downloads](#).
If you downloaded this site as a zip, see [Installation README](#). See also [Installation methods](#).

Feature	Version	Feature Category(ies)
org.jboss.ide.eclipse.archive.feature	3.3.1.v20130102-1730-H94-Final	AbridgedTools GeneralTools
org.jboss.ide.eclipse.archive.feature.source	3.3.1.v20130102-1730-H94-Final	AllSources
org.jboss.ide.eclipse.as.feature	2.3.2.v20130112-0201-H153-Final	AbridgedTools WebTools
org.jboss.ide.eclipse.as.feature.source	2.3.2.v20130112-0201-H153-Final	AllSources
org.jboss.ide.eclipse.freemarker.feature	1.2.0.v20130102-1731-H97-Final	AbridgedTools GeneralTools
org.jboss.ide.eclipse.freemarker.feature.source	1.2.0.v20130102-1731-H97-Final	AllSources

Anexo 10 – Utilitários baseados no Freemarker

10.1 – FMGENERATOR

É um Eclipse Builder. O Freemarker processa um arquivo *.fmg* gerando arquivos alvo (target) da seguinte forma: o path (do eclipse) do arquivo é obtido na macro *fmg_path* e o conteúdo do arquivo é obtido a partir da macro *fmg-content*.

Mais detalhes em <http://sourceforge.net/p/fmgenerator/wiki/Home/>.

10.2 RTF GENERATOR

É uma combinação de aplicativos JAVA e templates (macros) Freemarker. O objetivo é de se gerar um arquivo *.rtf* a partir de um template Freemarker utilizando chamadas as macros *rtf.document*, *rtf.newline*, *rtf.italic*, e etc.

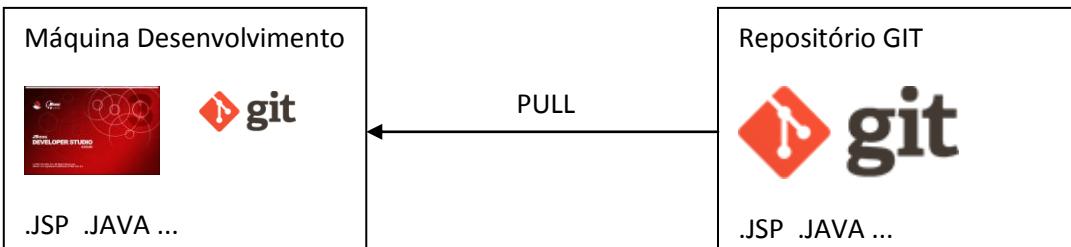
Download e mais detalhes em: <http://freemarker.sourceforge.net/libraries.html>

e <http://forum.springsource.org/showthread.php?35696-output-rtf-document-with-freemarker>

Anexo 11 – Procedimento para atualizar o repositório Git com fontes JSPs

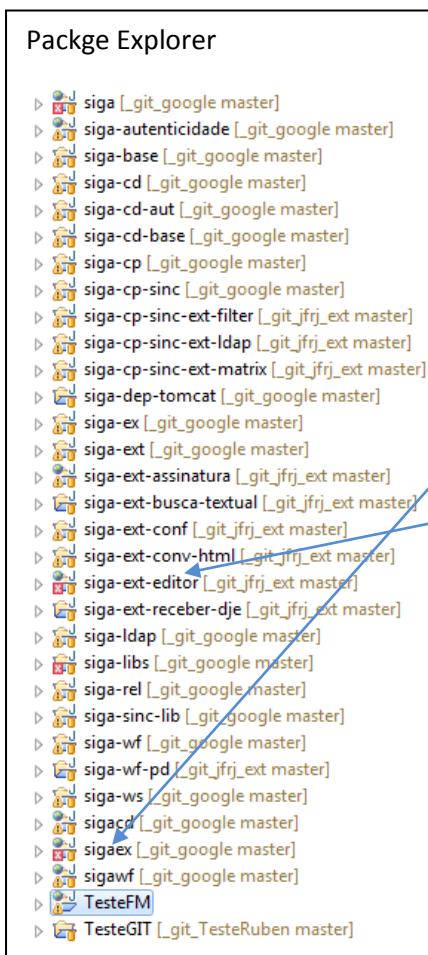
Estes procedimentos servem também para atualizar aplicações Java.

11.1 – Obter os fontes JSP (ou JAVA) mais recentes via PULL



O objetivo é pegar os fontes mais novos possíveis do JSP que serão alterados, visto que os fontes da máquina de desenvolvimento podem estar obsoletos.

Se observarmos o Package Explorer, nós temos o nome do projeto, e ao lado, o repositório onde ele se econtra. Um repositório pode, e deve, conter N projetos. O Pull é por repositório e não por projeto, ou seja, ao pegarmos qualquer projeto na árvore e clicarmos em **Team > Pull** estaremos atualizando também TODOS os projetos que temos naquele repositório.



Atualmente temos no SIGA dois repositórios, um remoto hospedado no Google (**_git_google**) e outro local (**_git_jfrj_ext**) hospedado no drive K.

Clicar em um projeto que contenha o repositório remoto, sigaex, por exemplo, e selecionar **team > pull**

Clicar em um projeto que contenha o repositório local, siga-ext-editor, por exemplo, e selecionar **team > pull**

Caso o PULL não funcione pode ser porque: o endereço do repositório mudou, as suas credenciais para acessar o repositório remoto estão erradas e/ou você não tem acesso ao repositório. Ver observações, item B.



PULL = FETCH + MERGE

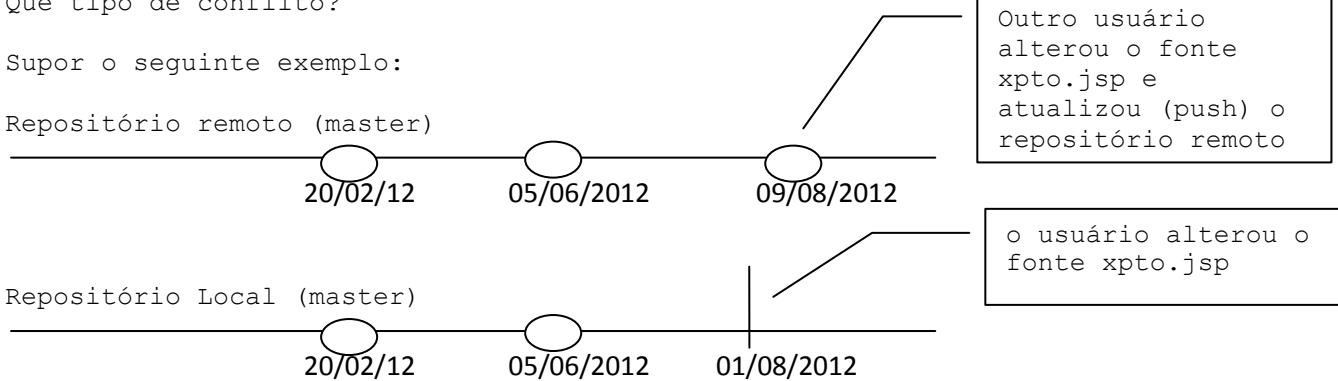
O MERGE está sujeito a conflitos como descrito em Anexos, item 7.3.1, que está copiado abaixo.



Que tipo de conflito?

Supor o seguinte exemplo:

Repositório remoto (master)



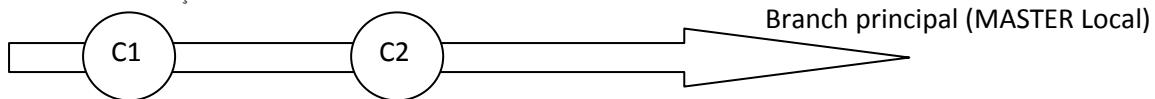
Se o usuário na máquina local realizar um PULL (fetch + merge), ou um MERGE (após o fetch no repositório remoto), o Git acusará um conflito no fonte xpto.jsp, conflito este que deverá ser administrado pelo usuário da máquina local, ou seja, receber o fonte mais novo do repositório remoto e aplicar as mudanças que ele realizou em 01/08/2012.

Observação: O PULL só afeta o repositório local.

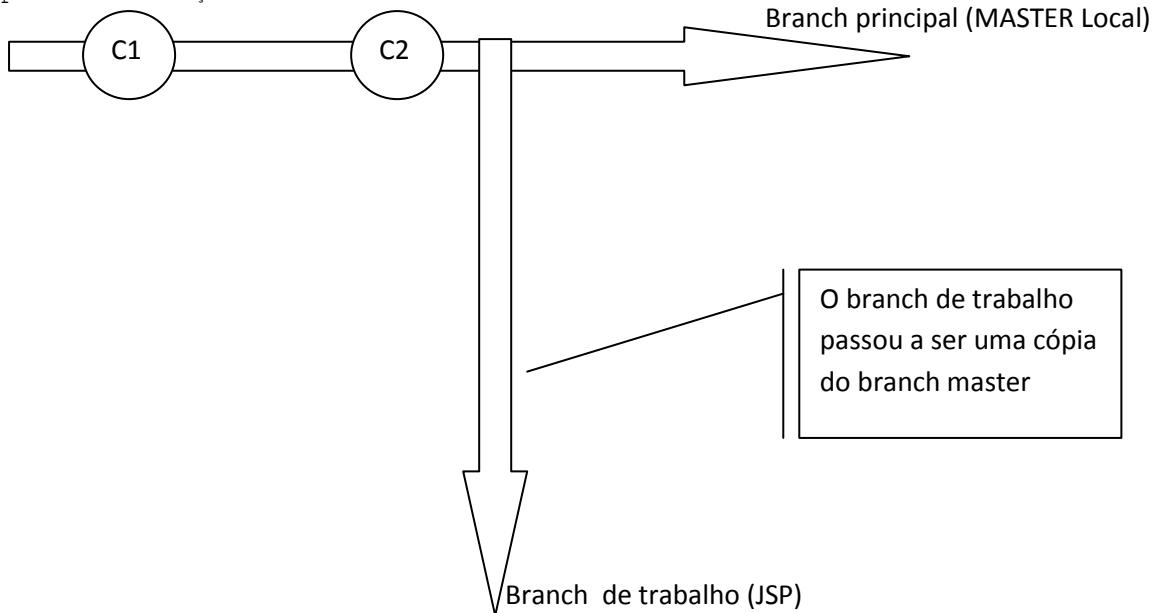
11.2 – Criar um branch (ramificação, desvio ...) para atualizar os fontes JSP

No exemplo abaixo, C1 e C2 são pontos de commit.

Antes da criação do branch

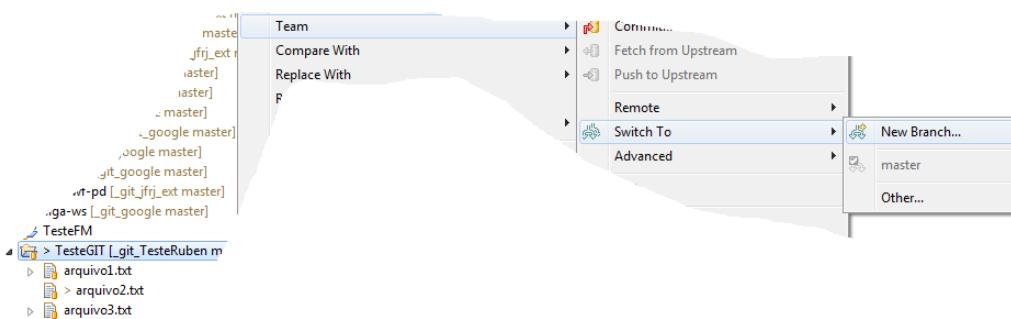


Depois da criação do branch

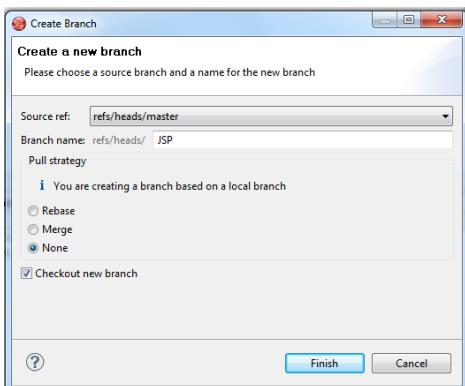


O objetivo é realizar as modificações dos fontes JSPs em uma área de trabalho (branch de trabalho), preservando os fontes originais na área principal (branch master)

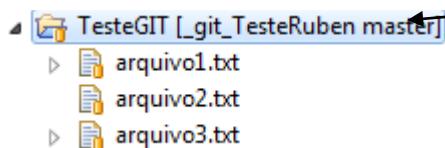
Utilizar o comando **Team > Switch to > New Branch** no mesmo projeto utilizado no item anterior (11.1). Observar que o New Branch afetará todos os projetos que estão no mesmo repositório.



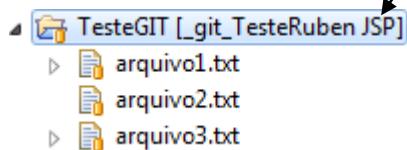
Criar o **branch de trabalho com nome semântico** (número do chamado, alteração principal ...). Neste exemplo chamaremos o novo branch simplesmente de JSP.



Observar que antes você estava trabalhando no branch master



e agora está no branch JSP



Observação: O BRANCH só afeta o repositório local.

11.3 – Alterar o(s) fonte(s) JSP

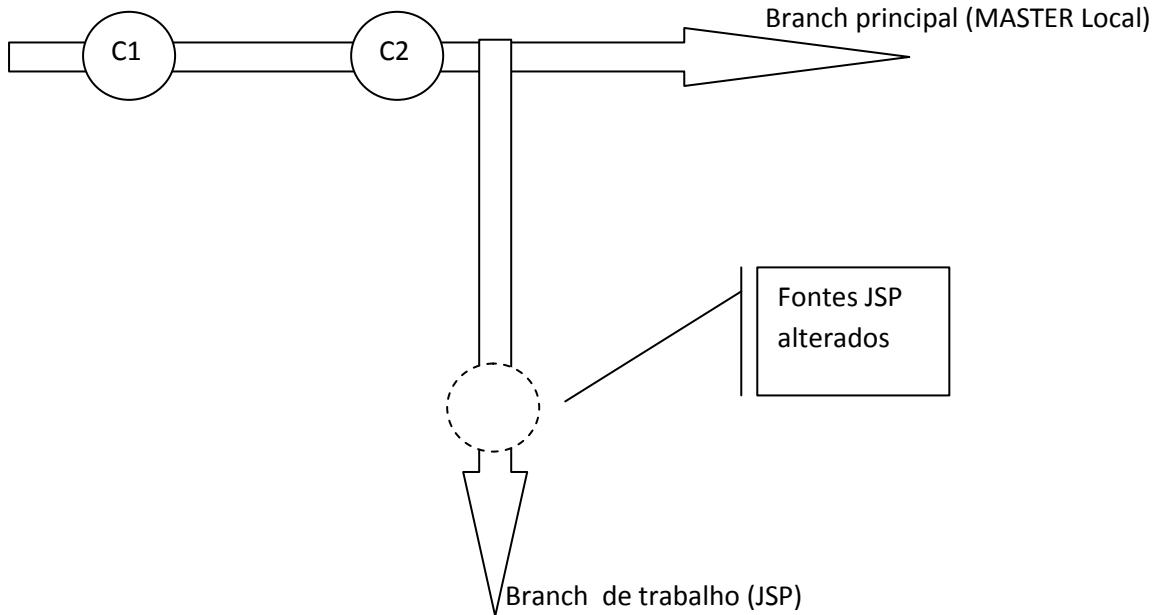
Agora que temos uma nova área de trabalho (Branch JSP) podemos alterar os nossos fontes sem preocupação.

A cada alteração, a aplicação JSP pode ser testada no ambiente local de desenvolvimento, ou seja, na IDE JBDS rodando o JBOSS (endereço: <http://localhost:8080/siga>)

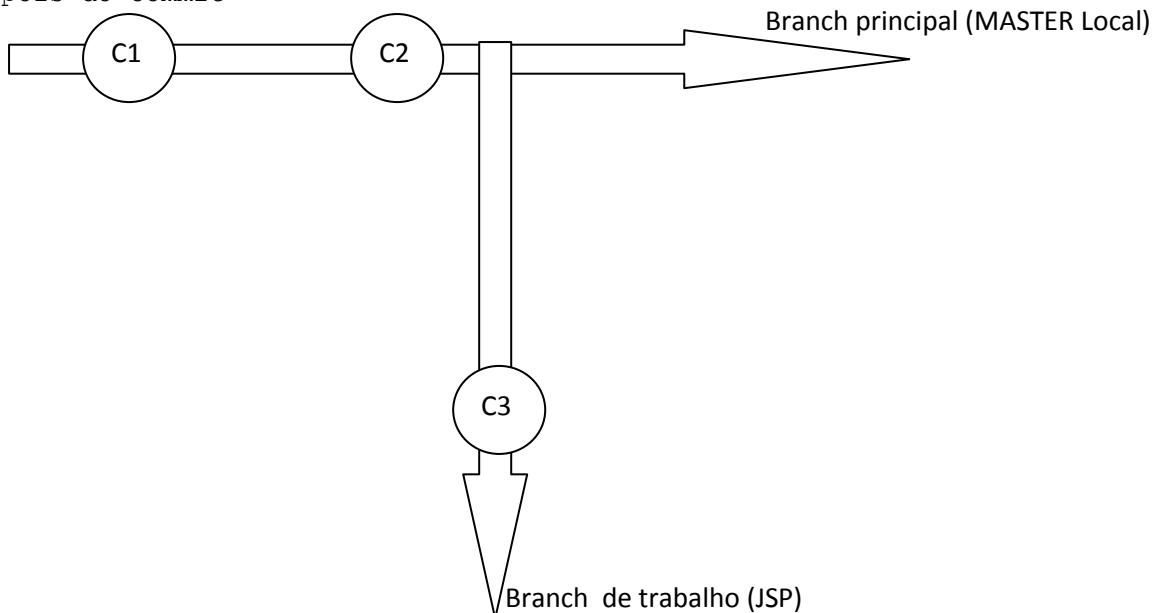
11.4 – Salvar (Commitar) as mudanças

O Commit “salva” (teoricamente é criar um ponto de mudanças) as mudanças no branch onde o comando é emitido, no nosso caso, branch de trabalho (JSP no caso).

Antes do Commit

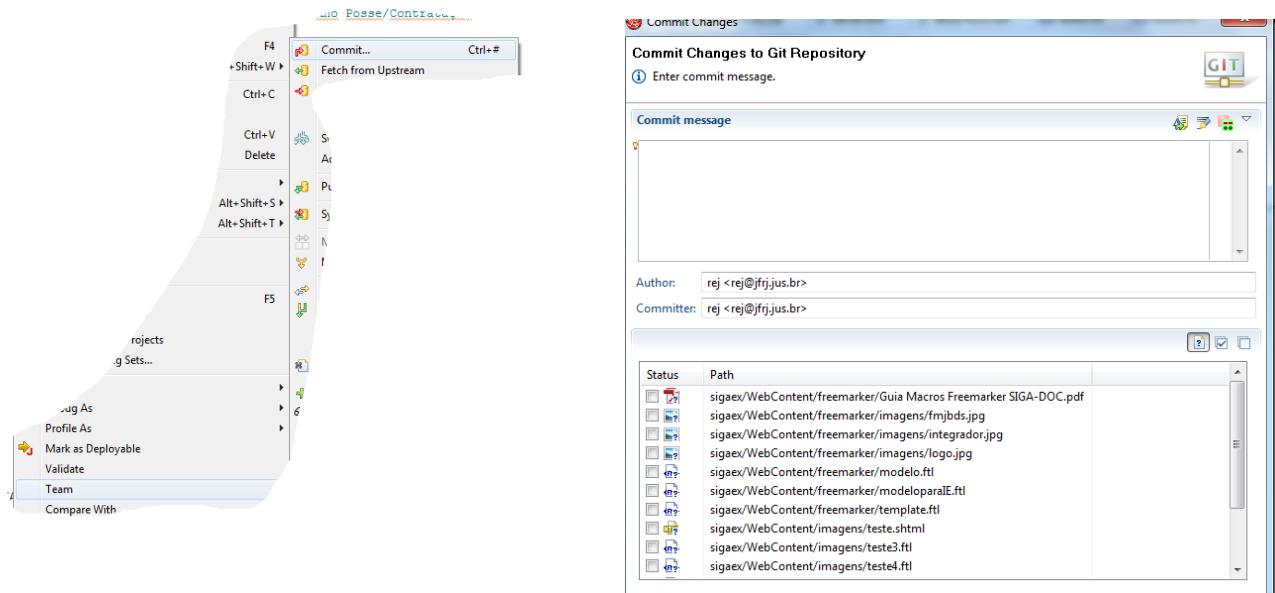


Depois do Commit



Observar que este novo ponto de commit (C3) não foi ainda replicado para o branch master local.

Utilizar o comando **Team > Commit** no mesmo projeto utilizado no item 11.1. Observar que o commit afetará todos os projetos que estão no mesmo repositório.

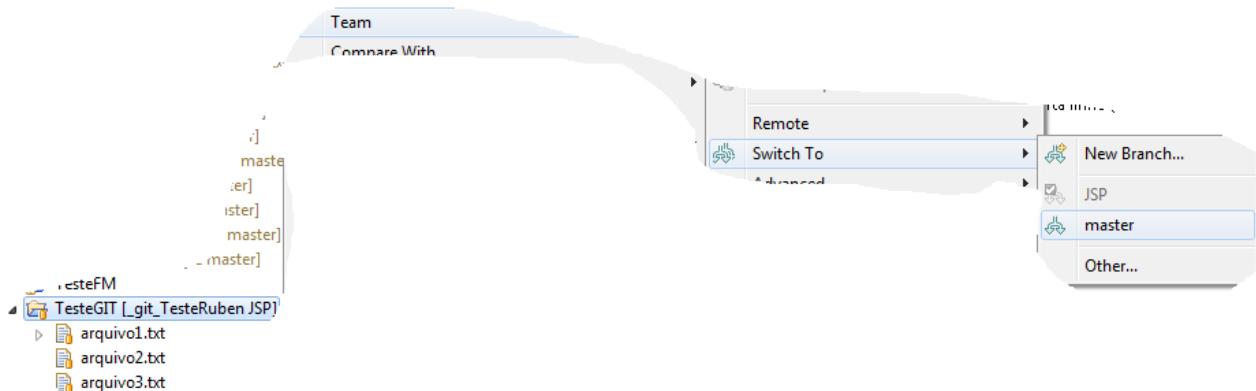


Colocar na mensagem do commit: uma breve descrição semântica, números do chamado e etc. Selecionar na parte inferior os arquivos que serão commitados.

Observação: O COMMIT só afeta o repositório local. Para que as mudanças se reflitam no repositório remoto elas devem ser publicadas via PUSH, como veremos a seguir

11.5 – Retornar (SWITCH) ao branch master

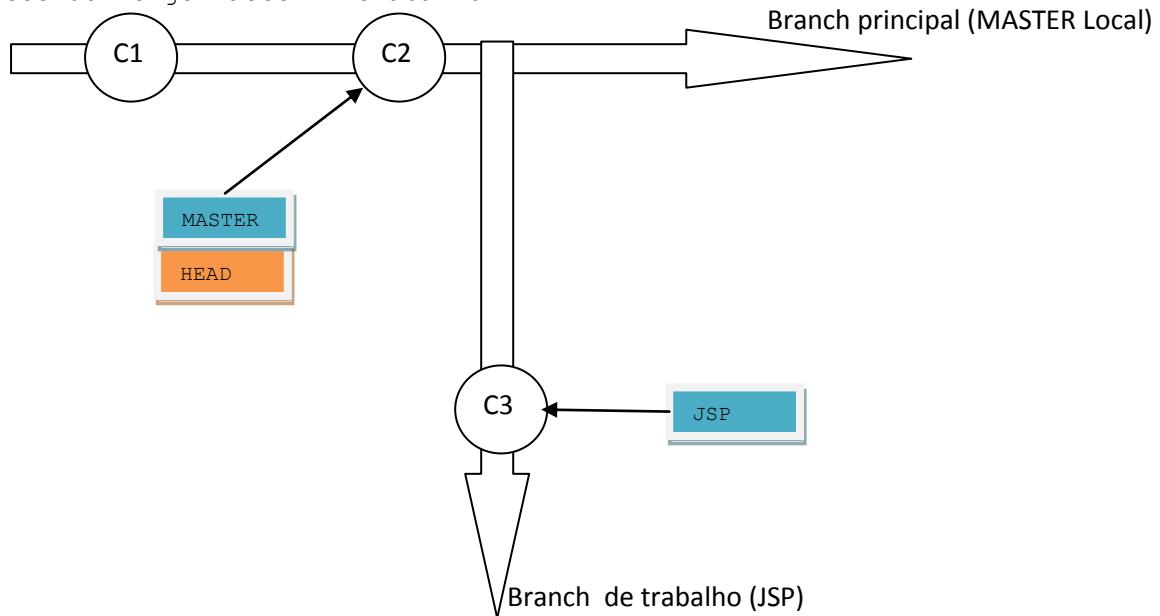
Team > Switch To > master



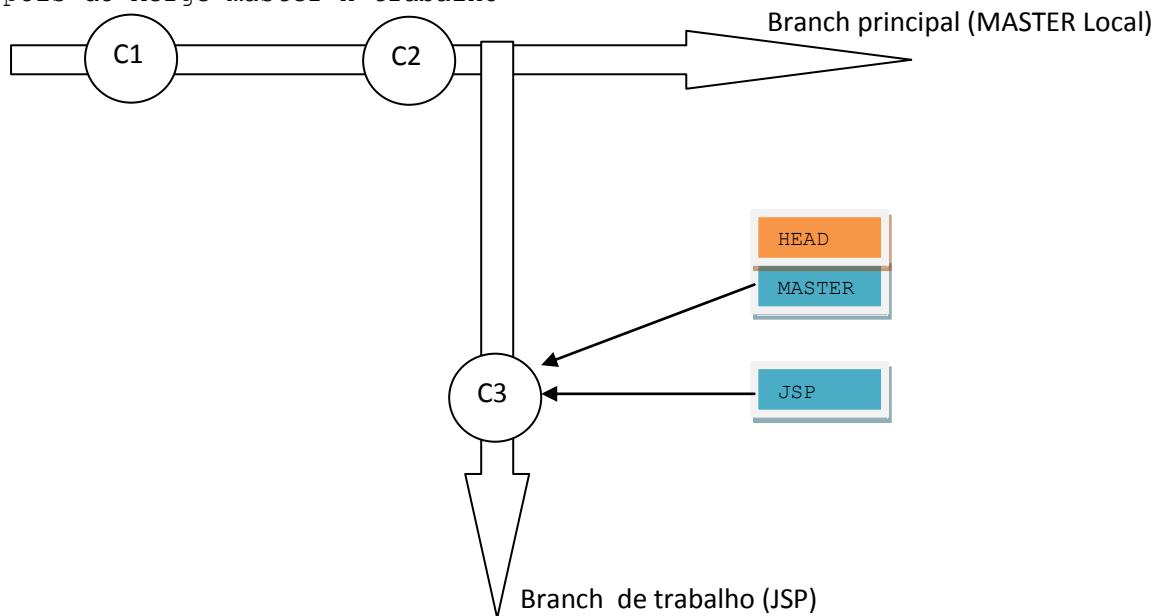
11.6 – Aplicar as mudanças do branch de trabalho no branch master via MERGE

O Merge cria um novo commit que incorpora as mudanças de outro commits. O exemplo abaixo é um exemplo de Fast Forward Merge.

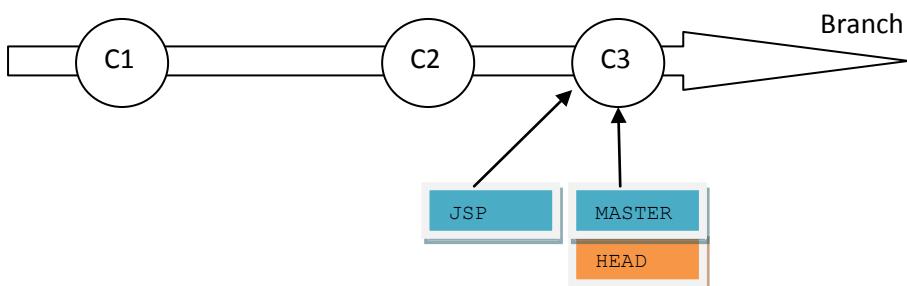
Antes do merge master x trabalho



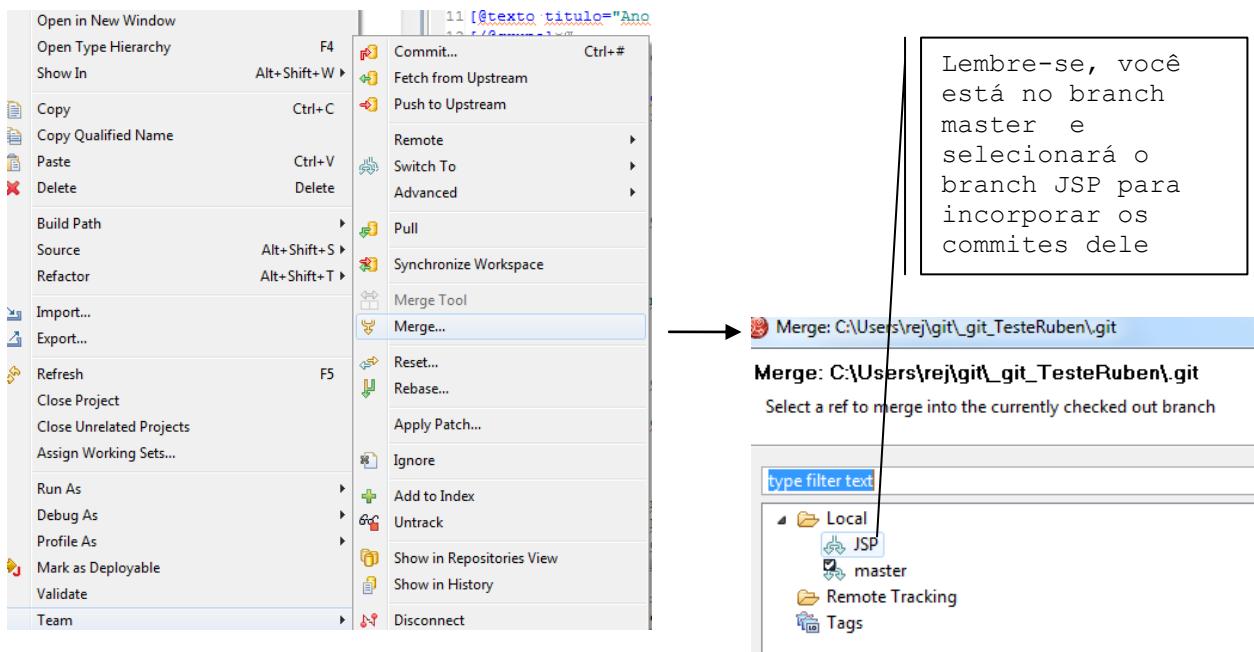
Depois do Merge master x trabalho



Ou simplificando,



Team > Merge



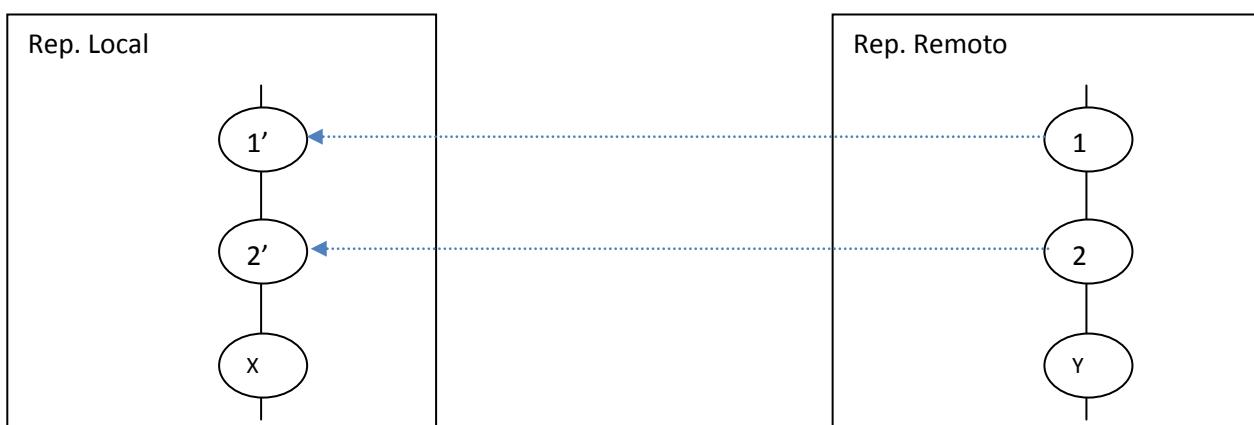
Observação: O MERGE só afeta o repositório local.

11.7 – Realizar o PULL novamente

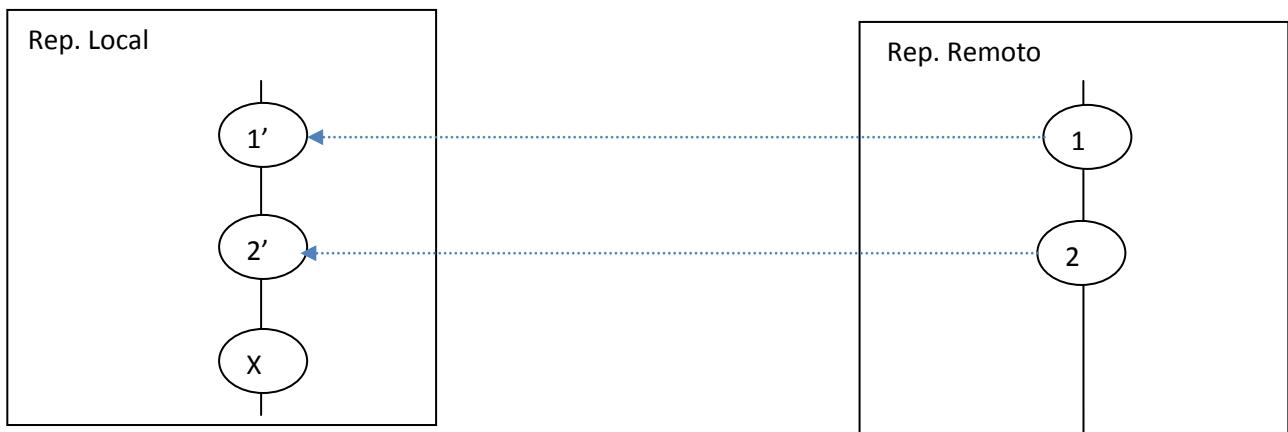
De novo? Isto já foi feito no item 11.1!

O problema é que antes de realizarmos o PUSH necessitamos verificar se os níveis de commit do repositório local (master) e o remoto (msater) estão equalizados.

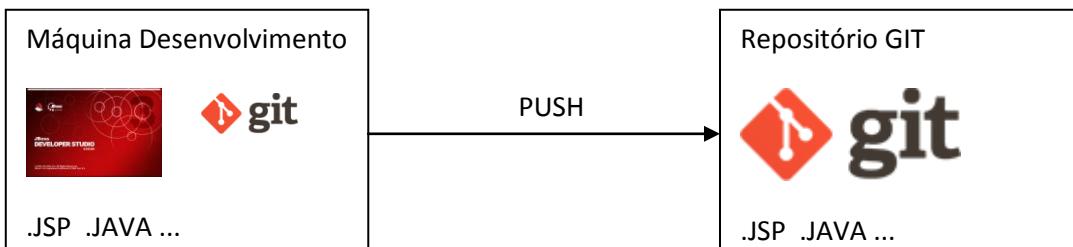
Neste exemplo, o repositório remoto possui o commit Y que não foi ainda aplicado ao local. O commit Y pode conter atualizações em alguns fontes que o usuário local tenha também feito. Aqui temos um conflito. O usuário local deve então realizar um PULL (fetch + merge), administrar os conflitos, commitar e aí sim, realizar o PUSH.



Neste exemplo, os níveis de commit estão OK, e o usuário local pode realizar um PUSH. Nem é necessário realizar um PULL antes.

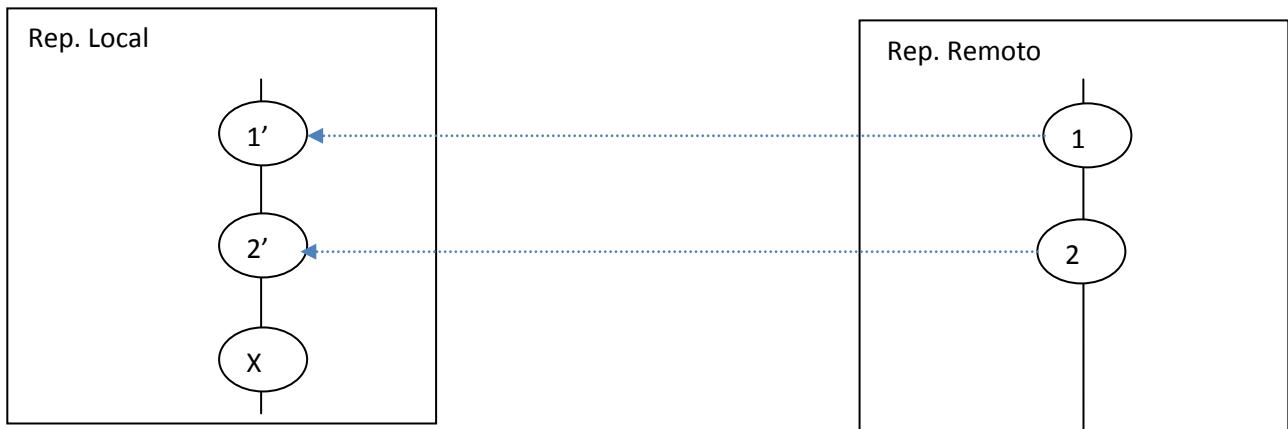


11.8 – Realizar a mudanças no repositório remoto (upstream) com o PUSH

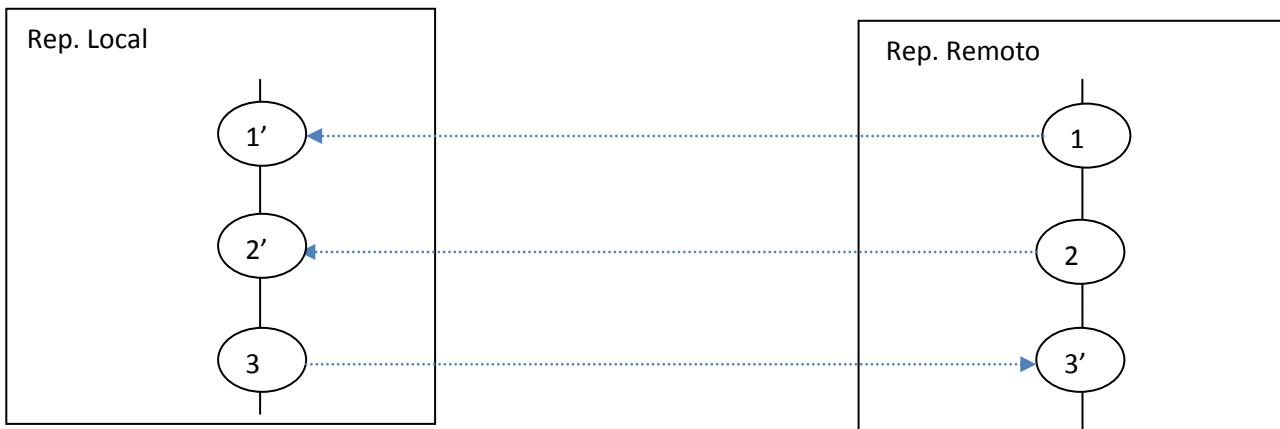


O PUSH atualiza o repositório remoto a partir do repositório local.

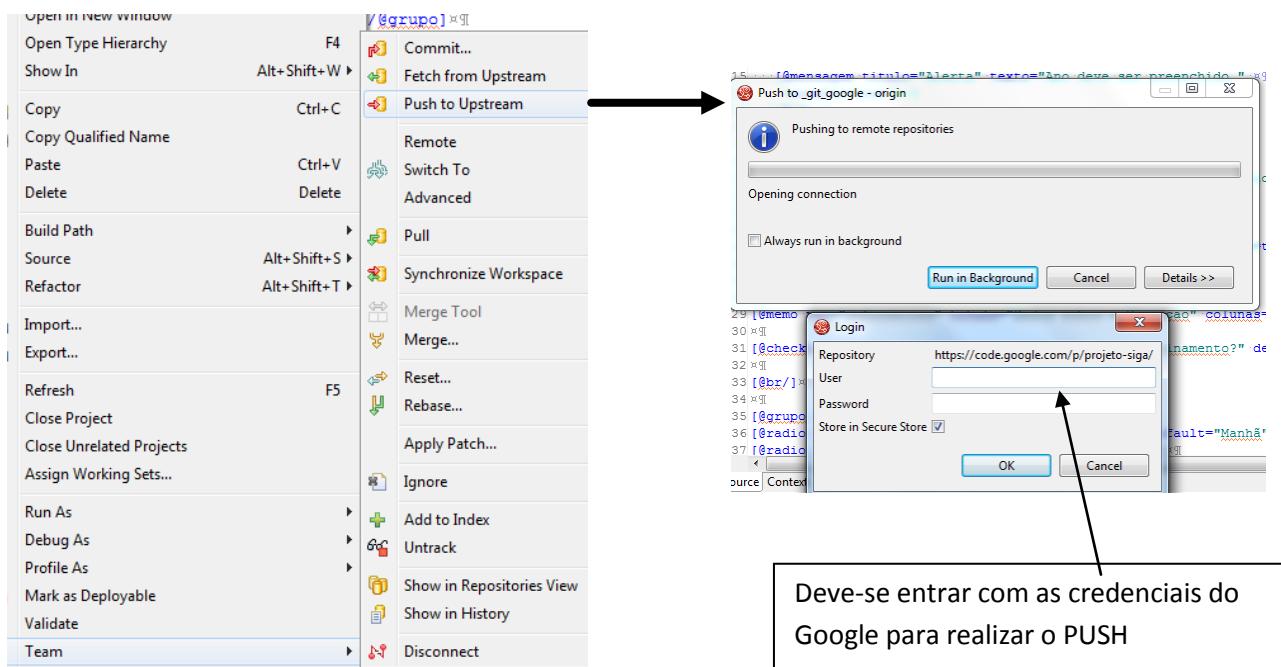
Antes do PUSH



Depois do PUSH



Team > Push to Upstream



Observação: O PUSH afeta o repositório remoto. Somente usuários autorizados podem atualizar o repositório remoto com a conta do Google.

Caso o PUSH não funcione pode ser porque: o endereço do repositório mudou, as suas credenciais para acessar o repositório remoto estão erradas e/ou você não tem acesso ao repositório. Ver observações, item B.

11.9 – Quando as mudanças terão efeito?

O fato de ter atualizado o repositório remoto não quer dizer que as alterações já estão em produção e que o usuário possa utilizá-las. Só depois que o pessoal da infraestrutura do SIGA atualizar o servidor de aplicação é que as mudanças tomarão efeito.

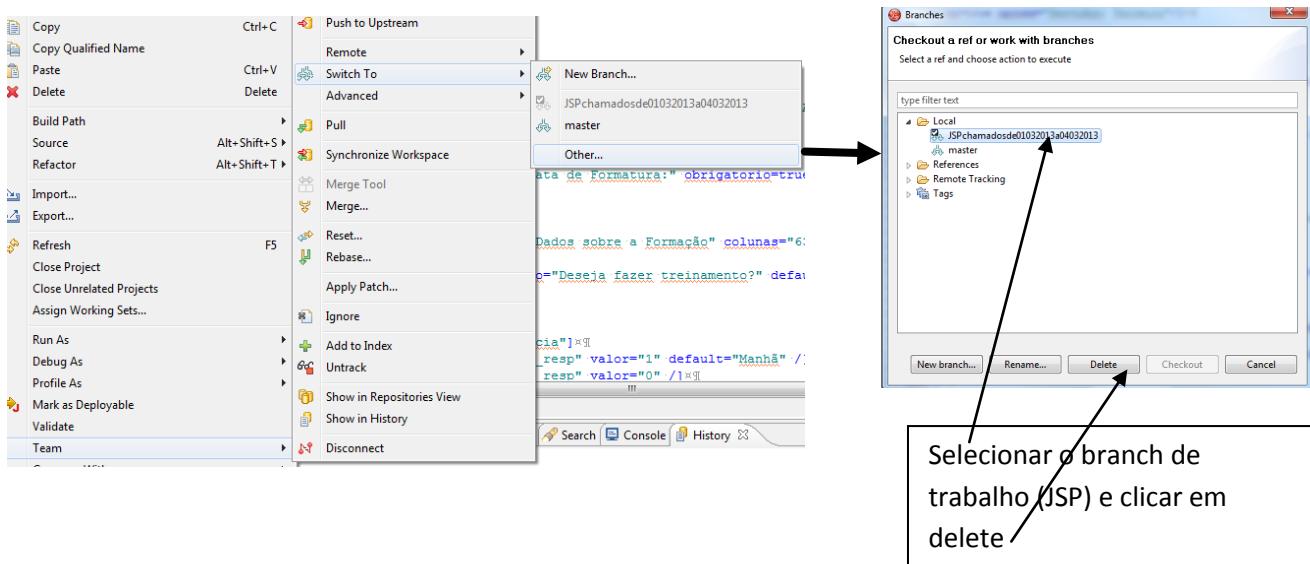
E se precisarmos colocar um JSP em produção COM URGÊNCIA?

Existe a hipótese de mover diretamente o arquivo .jsp para um diretório específico (.../paginas/expediente/modelos) do servidor de aplicação (JBOSS). É importante que esta operação seja apoiada pelo pessoal de infra do SIGA.

11.10 – Posso excluir o branch de trabalho (JSP)?

Depois que as alterações já tenham sido aplicadas ao servidor de aplicação e o usuário testado, aí sim podemos excluir o branch de trabalho.

Team > Switch to > Other ...



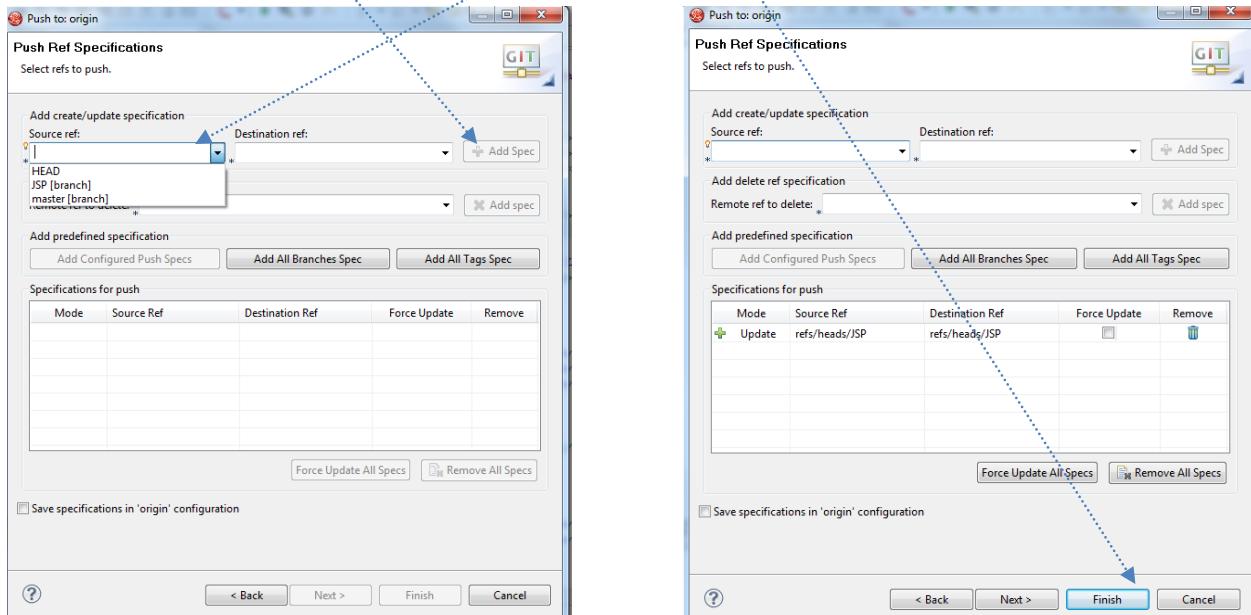
Observações:

A - O procedimento acima é para funcionários da SJRJ

Para estagiários e terceiros temos outra política. Estes funcionários não podem atualizar o branch master remoto diretamente, e sim, salvar nele um branch contendo as modificações. Depois, um funcionário habilitado aplica as mudanças no branch master.

Estagiário / Terceiro:

- Realizar os passos de 1(11.1) a 4(11.4) exatamente como descrito acima;
- Realizar o pull do passo 7 (11.7), embora neste caso não seja necessário;
- Realizar o push, passo 8(11.8).
No Push Ref Specifications, selecionar o branch criado contendo as modificações. Clicar em Add Spec depois Finish.



Funcionário:

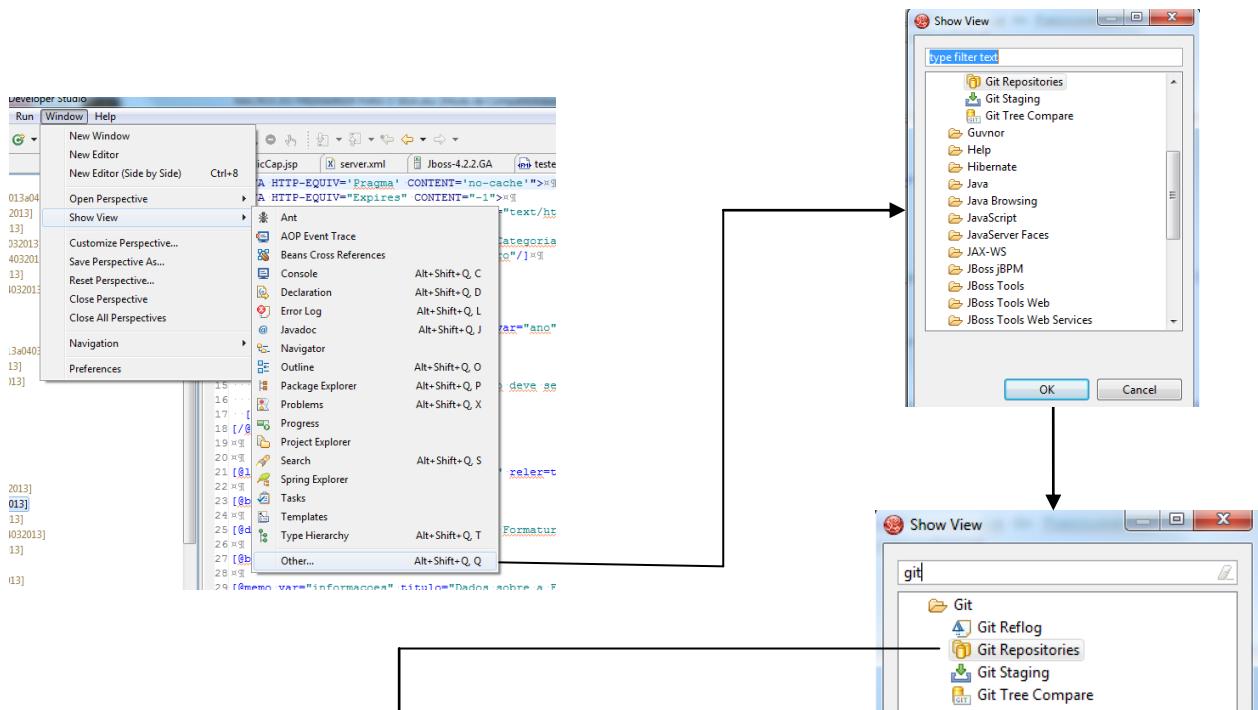
- Realizar o pull, passo 7 (11.7) para buscar o branch remoto com as modificações;
- Estar ciente que está no branch master local. Caso contrário, realizar um switch, como no passo 5 (11.5);
- Realizar um merge, passo 6 (11.6) entre o branch atual (master) e o branch contendo as modificações (que deve ser selecionado);
- Realizar o push, passo 8 (11.8). Neste caso, no Push Ref Specifications selecionar o master [branch].

B - Configurando as credenciais e os endereços do repositório GIT

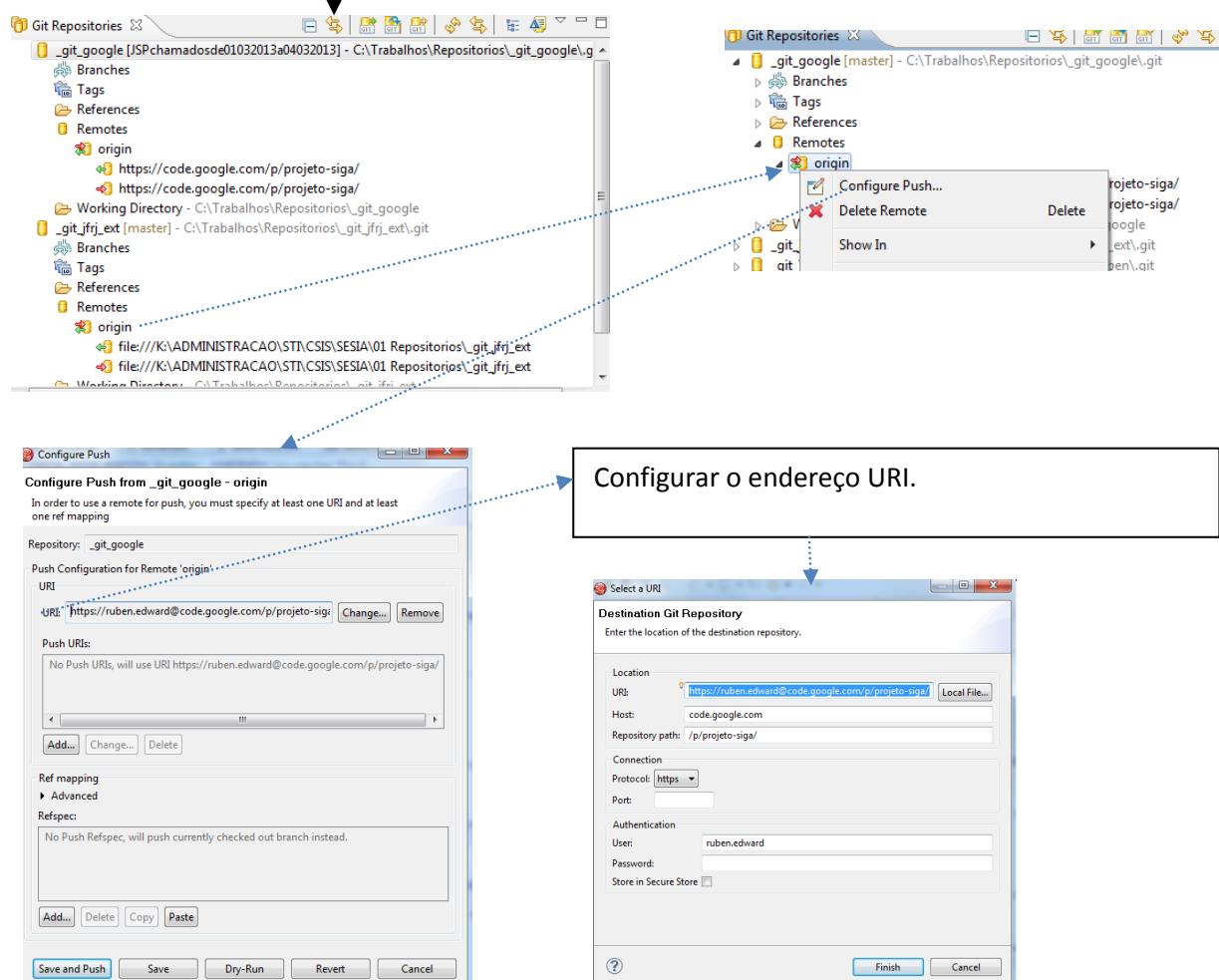
B.1 - Utilize o procedimento abaixo para verificar os endereços dos repositórios

Acessar a View dos repositórios Git

View > Show View > Other ... Git Repositories



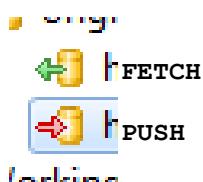
CURSOR EM origin > Configure Push



Atualmente são os endereços abaixo que estão valendo:
Repositório Local: K:\ADMINISTRACAO\STI\CSIS\SESSIA\01 Repositorios_git_jfrj_ext
Repositório Remoto: <https://code.google.com/p/projeto-siga/>

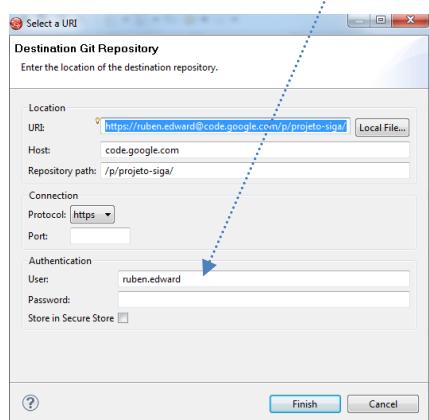
Observação: No caso do Google o endereço que fica registrado é:
<https://usuario@code.google.com/p/projeto-siga/>, sendo que no meu caso, <https://ruben.edward@code.google.com/p/projeto-siga/>

O mesmo procedimento serve para configurar o FETCH (PULL)



B.2 – Verificando as credenciais para acessar o repositório remoto

No item anterior vimos como configura a URI, e nesta mesma tela, temos como configurar as credenciais, user e password.



B.3 – Autorização para acessar o projeto e Conta no GoogleCode

Para que um usuário faça atualizações no repositório remoto (googlecode) via PUSH é necessário que: ele possua uma conta no google e que o administrador do repositório o habilite.

Após a habilitação o usuário ganhará uma password para o googlecode:



Caso o usuário deseje utilizar a password do Google / Gmail ele deve marcar a opção abaixo:

Security

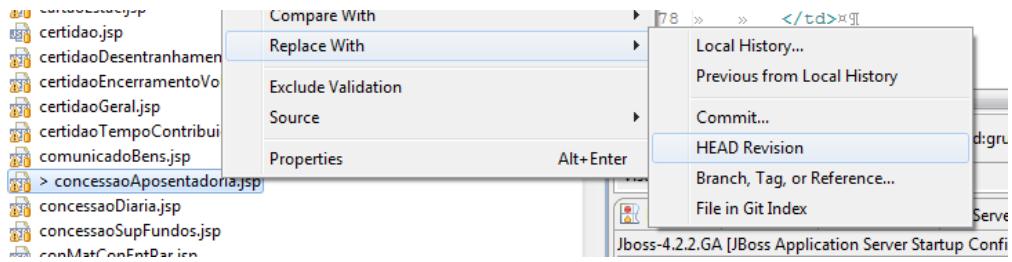
Accept Ruben.Edward@gmail.com Google Account password when using a Git or Mercurial client. To make sure your password is safe, always use the latest client from:

- <http://git-scm.com/downloads>
- <http://mercurial.selenic.com/downloads/>

C – Revertendo um código ao original

Suponha que você já tenha atualizado o arquivo concessaoAposentadoria.jsp, porém deseja que ele volte ao original.

Selecionar o jsp > Replace With > HEAD Revision



Anexo 12 – Instalando o DBVisualizer

12.1 – Instalando o produto

O DBVisualizer nos permite realizar a engenharia reversa das tabelas nos esquemas Oracle.

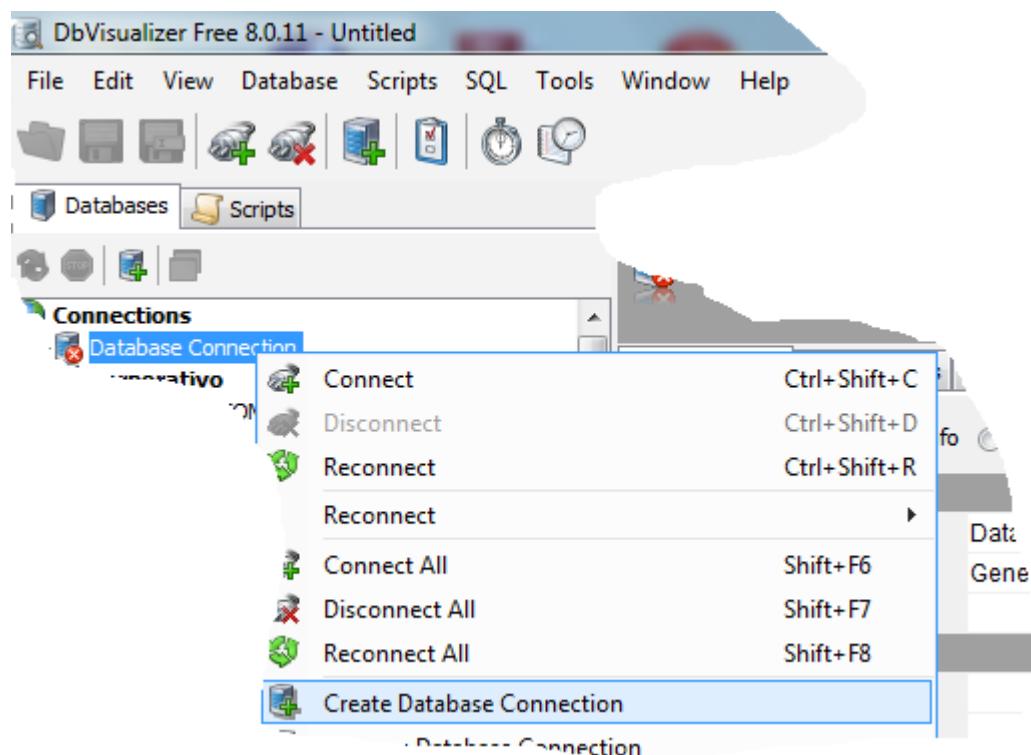
O DBVisualizer já está na versão 9.0.6 (<http://www.dbvis.com/download/>) , existindo uma versão free e uma paga (pro). Estamos utilizando a versão free.

Aqui na SJRJ já temos no diretório desenvolvimento a versão 8.0.4. Basta copiar (ou baixar) o diretório (DbVisualizer-8.0.4) e executar o **dbvis.exe**, **não precisa de instalação**.

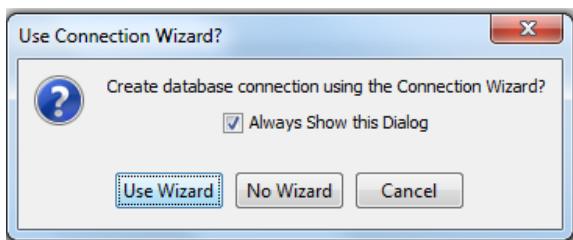
A vantagem dele sobre o SQLDeveloper é que ele faz engenharia reversa das tabelas e cria um modelo ER físico.

12.2 – Configurando a conexão

Database Connection > Create Database Connection



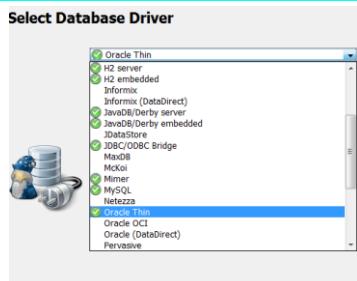
Use Wizard



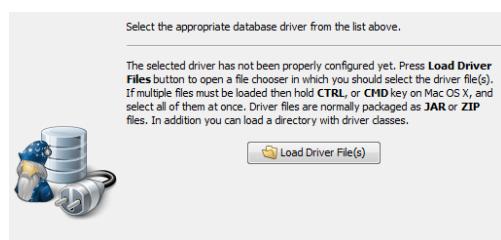
Entre com o nome (qualquer) da conexão



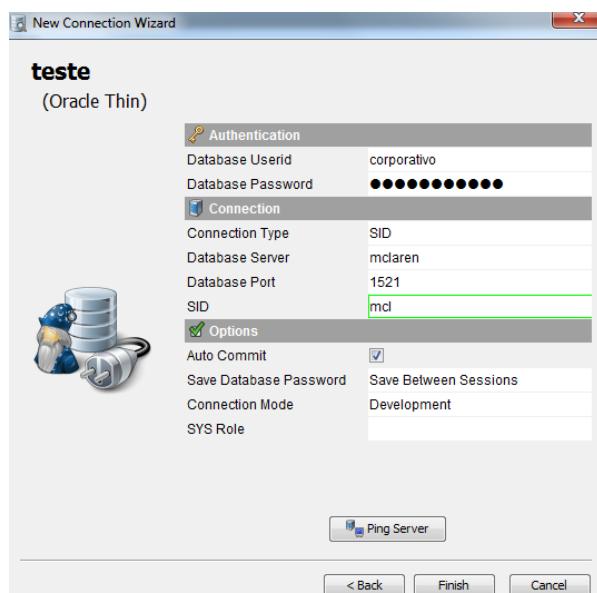
Selecione Oracle Thin Driver



Se o Oracle Thin já estiver marcado com é porque o driver já foi resolvido. Caso contrário, temos que carregá-lo. O driver é o **ojdbc14.jar**. Geralmente ele está na pasta Java ou JDK. No meu caso está em **C:\Desenvolvimento\jdk1.6.0\lib**.



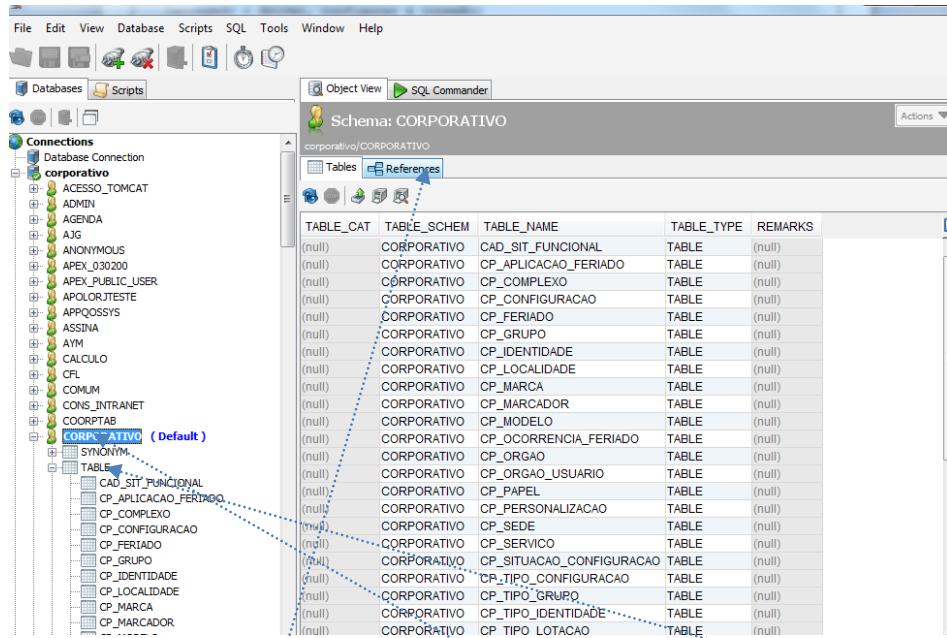
Carregado o driver, configurar a conexão:



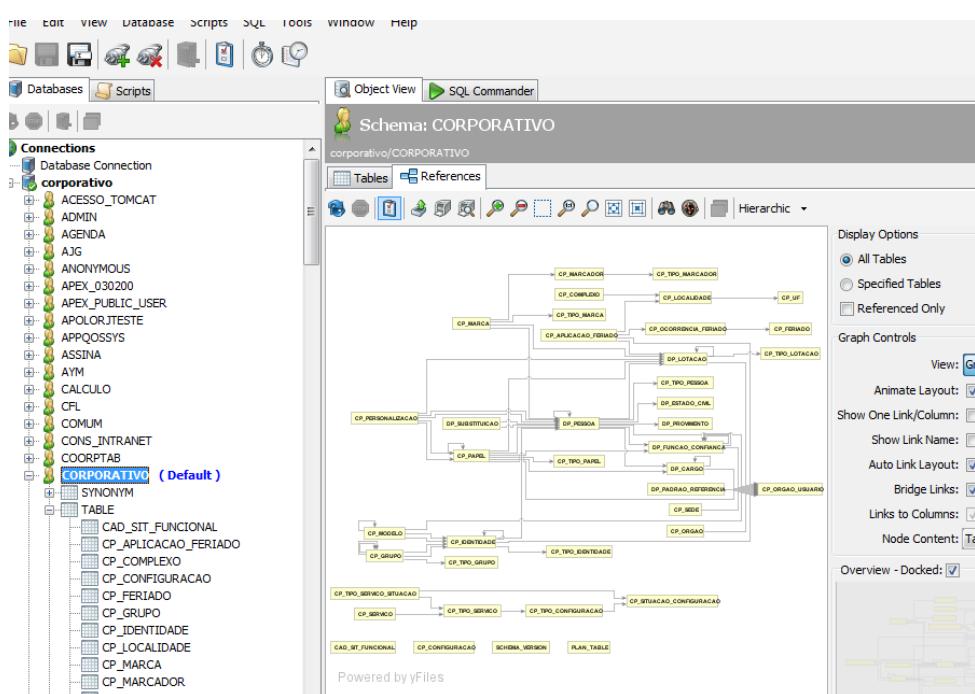
Corporativo	SIGAEX Desenv	SIGAEX Homo
corporativo	siga	siga
corporativo	siga	siga
sid	sid	sid
mclaren	McLaren	mclaren
1521	1521	1521
mcl	mcl	homolo

12.3 - Utilizando o produto

Depois de configurada e estabelecida uma conexão, clicar na mesma para que seja exibido todos os esquemas do BD. Obviamente, somente os esquemas que o usuário (fornecido na conexão) possui autorização serão listados.

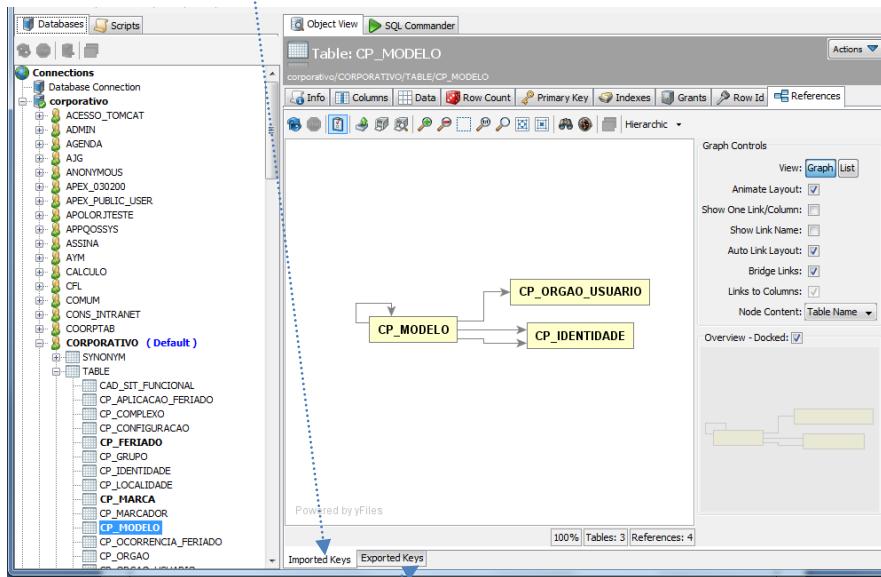


No DB explorer, selecionar um esquema e clicar em table. Na janela da direita clicar em References.

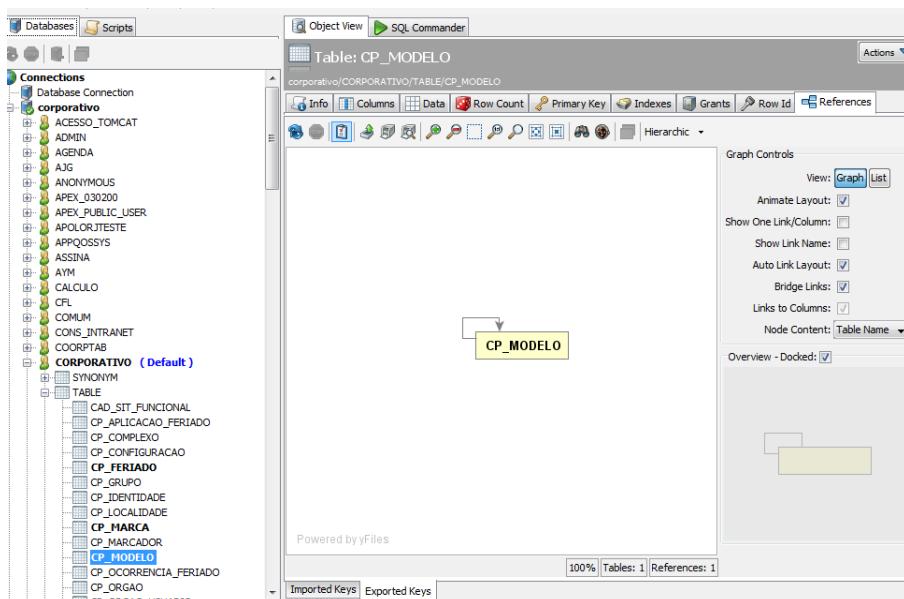


O mesmo processo pode ser feito no nível da tabela.

Selecionar uma tabela específica e clicar em reference.
O default é Imported Keys (para quais tabelas esta tabela aponta - Integridade referencial)



Mudar para Exported Keys (quais tabelas apontam para esta tabela)



Observação: Existe um bug no DBVisualizer em relação as constraints. Mesmo que elas estejam com status de disable, o produto desenha as constraints. Para verificar se as constraints estão enable ou disable nós utilizamos o SQLDeveloper, pois não encontramos esta opção no DBVisualizer.

Anexo 13 – Instalando e configurando JBDS

Copiar os diretórios **Desenvolvimento** (binários das aplicações) e **Trabalhos** (workspaces, repositório dos fontes) para a máquina de destino. O Git também está incluído.

Colocar um link, na área de trabalho, para a chamada do JBDS que é dado por jbdevstudio.bat, <C:\Desenvolvimento\jbdevstudio-4-1-0\jbdevstudio.bat>.

Ao copiar, toda a configuração da máquina de origem é clonada, inclusive as credenciais (usuário e password) do repositório remoto no googlecode. Para tanto, é importante reconfigurar estas credenciais conforme mencionado no anexo 11, observação, B.2.

Anexo 14 – Instalando e configurando o ObjectAid e UML Doclet (ex yDocs)

Gerando o modelo de Classes via engenharia reversa.

Para este trabalho estaremos utilizando duas ferramentas distintas:

- Para gerar o modelo global do projeto: utilizaremos um plugin do Eclipse, o ObjectAid (<http://www.objectaid.com/class-diagram>)
- Para documentar as classes: utilizaremos um plugin do JavaDoc, o UML Doclet (ex yDoc) da yWorks (http://www.yworks.com/en/products_ydoc.html).

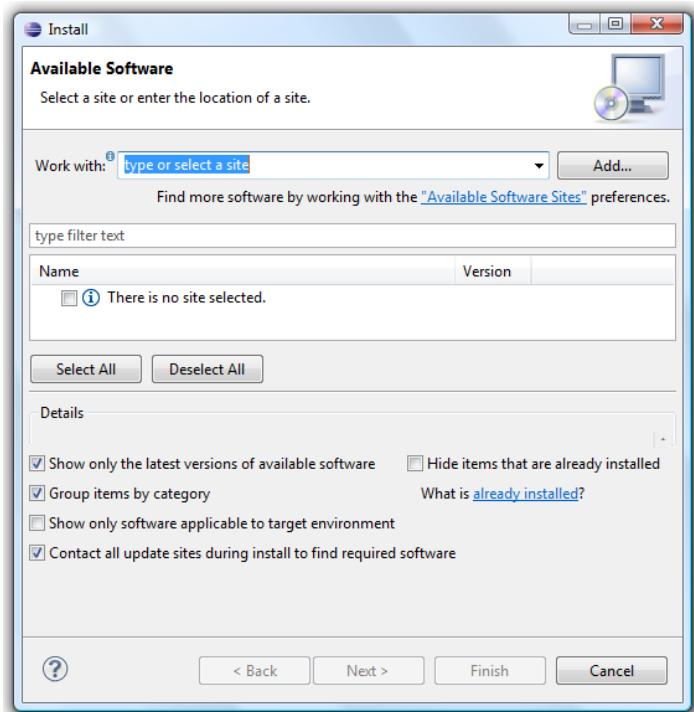
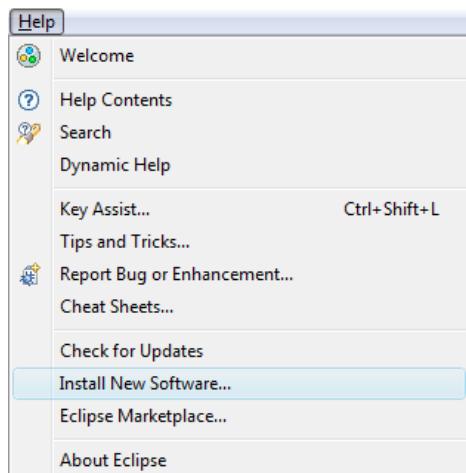
14.1 – Instalando os produtos

14.1.1 – ObjectAid

A instalação se dá pelo eclipse.

Help > Install New Software ...

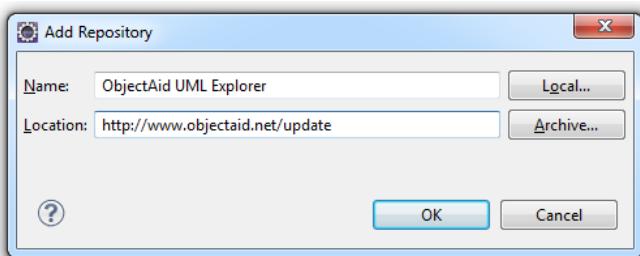
Na página "Available Software" pressione Add ...



Na página "Add Repository" entre com as informações abaixo e pressione OK

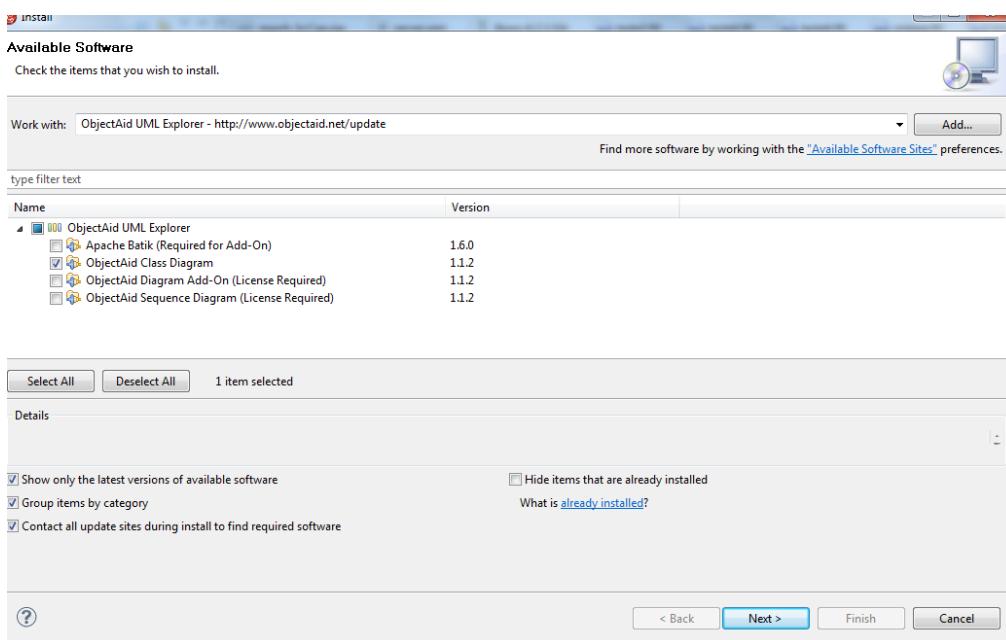
Name: ObjectAid UML Explorer

URL: <http://www.objectaid.net/update>



Serão mostrados todos plug-ins do objectAid.

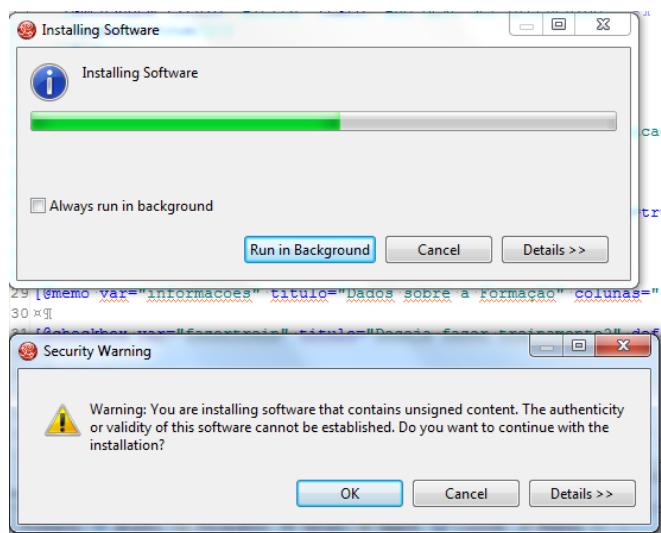
Selecione ObjectAid Class Diagram e pressione Next.



Aceite o License Agreement.

O processo de instalação dará início.

Um aviso (`... contains unsigned content ...`) aparecerá, simplesmente pressione OK. O Plug-in será instalado e solicitará que o IDE seja restartado.



14.1.2 - UML Doclet (ex yDoc)

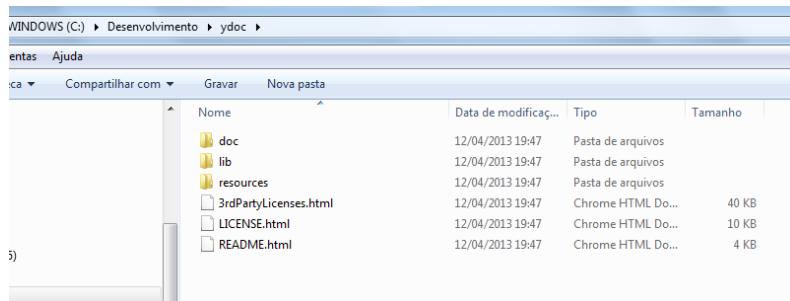
O UML Doclet é um plug-in para o javadoc da empresa yWorks. Existe uma versão paga e uma free (edição comunitária) que será utilizada.

Realizar o download do plug-in em http://www.yworks.com/en/products_ydoc.html

Download
Download the yWorks UML Doclet Community Edition:
yWorks UML Doclet Community Edition (JDK 1.5.0 / 1.6.0 / 1.7.0)
yWorks UML Doclet Community Edition (J2SDK 1.4.x)

Buy
Buy the yWorks UML Doclet Professional Edition (licenses are sold by our sales partner share-it!):
yWorks UML Doclet Professional Edition

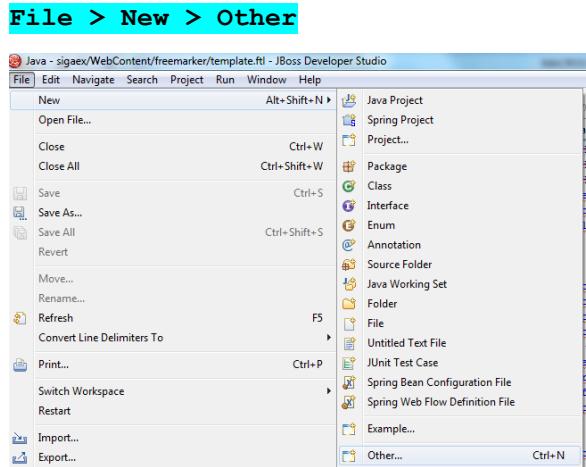
loc e unzipar o download para lá.



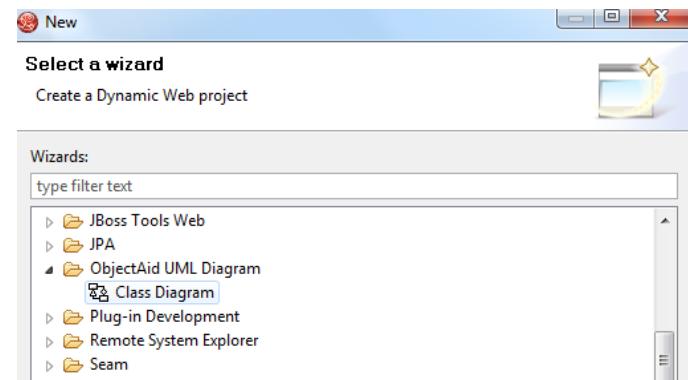
O UML Doclet será rodado pelo javadoc na interface da IDE como veremos a seguir.

14.2 - Utilizando os produtos

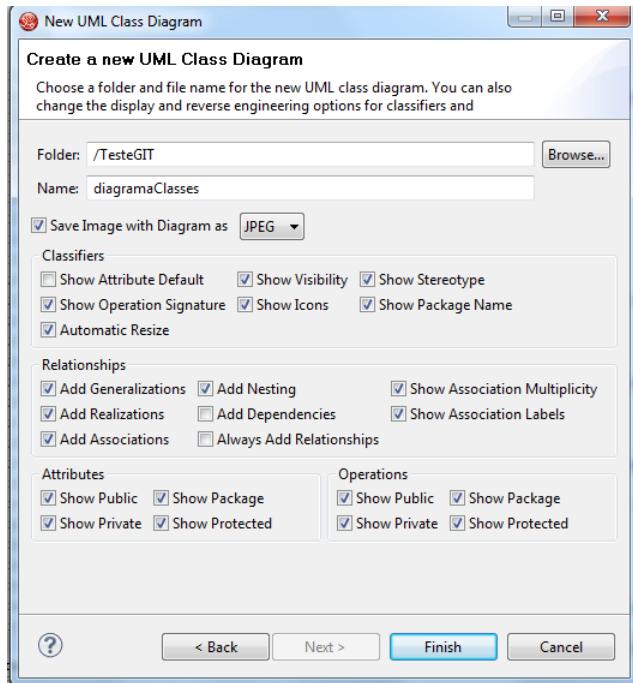
14.2.1 - ObjectAid



Selecionar ObjectAid / Class Diagram

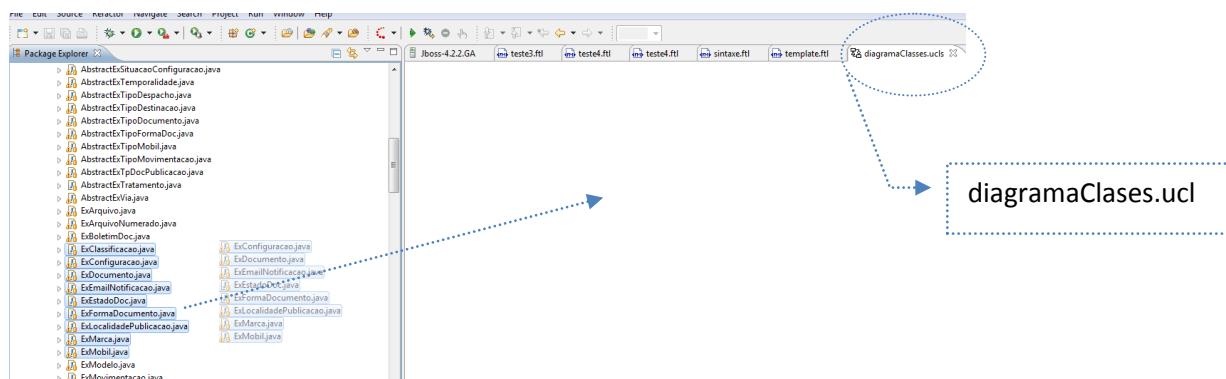


Selecionar o folder (caminho) onde residirá o diagrama a ser gerado. Fornecer um nome para o diagrama. Selecionar o formato da imagem e marcar as opções desejadas.



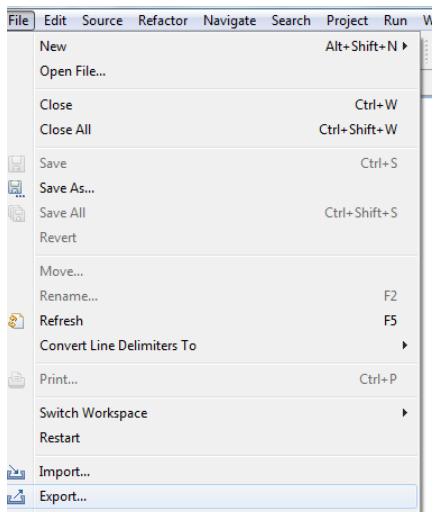
Será criada uma aba na IDE com o nome_do_diagrama_fornecido.ucl (no caso, diagramaClasses.ucl).

Agora, navegue nos projetos / pacotes, selecione as classes e arraste-as para a aba recém criada. O diagrama será gerado.

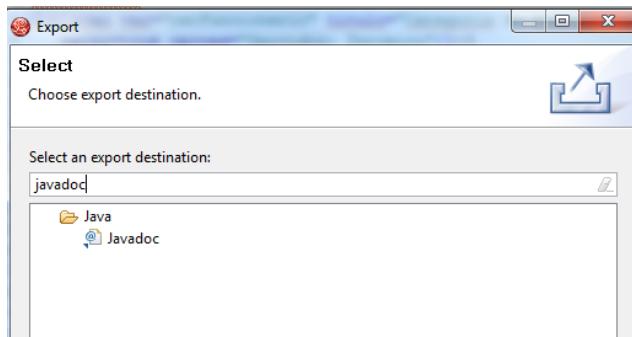


14.2.2 – UML Doclet

File > Export



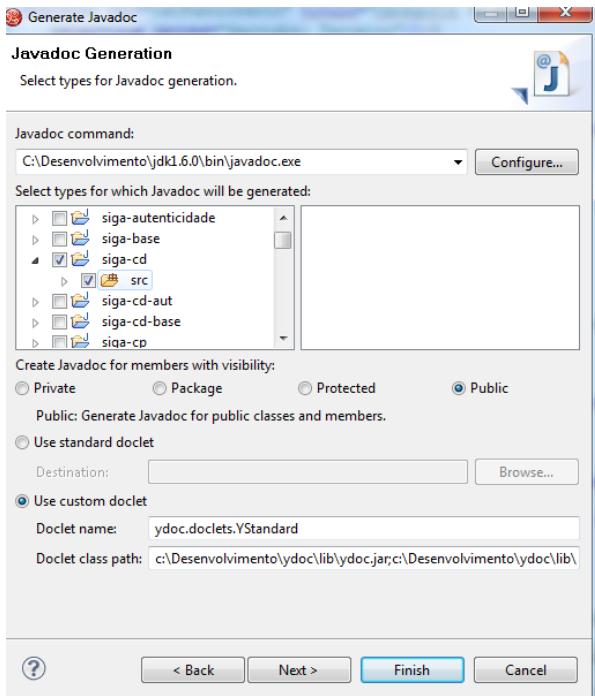
Selecionar Javadoc



Selecionar o(s) projeto(s) e/ou pacote(s). Marcar "Use custom doclet" e informar também:

- **Doclet name:** ydoc.doclets.YStandard
- **Doclet class path:** c:\Desenvolvimento\ydoc\lib\ydoc.jar;
c:\Desenvolvimento\ydoc\lib\styleed.jar; c:\Desenvolvimento\ydoc\resources;

Clicar em Next



Na tela seguinte, fornecer as seguintes informações:

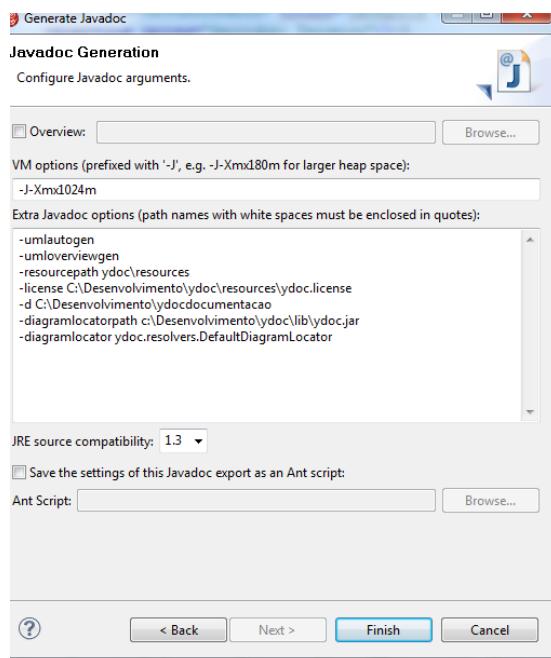
VM Options: -J-Xmx1024m

Extra Javadoc options:

```

-umlautogen
-umloverviewgen
-resourcepath ydoc\resources
-license C:\Desenvolvimento\ydoc\resources\ydoc.license
-d C:\Desenvolvimento\ydocdocumentacao
-diagramlocatorpath c:\Desenvolvimento\ydoc\lib\ydoc.jar
-diagramlocator ydoc.resolvers.DefaultDiagramLocator

```



Clicar em Finish.

Na console será exibido o progresso da documentação.

```

Source Context
Problems @ Javadoc Declaration Servers Search Console History
<terminated> Javadoc Generation
Loading source files for package br.gov.jfrj.siga.cd.service.impl.test...
Loading source files for package br.gov.jfrj.siga.cd...
Loading source files for package br.gov.jfrj.siga.cd.service.impl...
Loading source files for package br.gov.jfrj.siga.cd.old...
Constructing Javadoc information...
C:\Trabalhos\Repositories\ git google\siga-cd\src\br\gov\jfrj\siga\cd\service
import com.sun.org.apache.xerces.internal.impl.dv.util.Base64;
^

```

Explicação de alguns parâmetros que podem ser passados ao UMLDoclet

14.2.2.1 - Tipos de Diagrama

-umlgen

UML diagrams will be created and embedded for all documented files with an @y.uml tag. @y.uml may be used in type, package, and overview documentation.

O @y.uml é uma anotação UML Doclet. Por exemplo, se uma classe A possui esta anotação e uma classe B não, ao pedirmos o tipo -umlgen, somente a classe A será documentada.

-umloverviewgen

An UML **overview diagram** will be created and embedded, even if there is no @y.uml tag in overview.html.

-umlpackagegen

UML diagrams will be created and embedded for all documented **packages**, not only for those with an @y.uml tag.

-umltypegen

UML diagrams will be created and embedded for all documented **classes** and **interfaces**, not only for those with an @y.uml tag.

Segundo o manual, **-umlautogen** é o mesmo que utilizar **-umltypegen**, **-umlpackagegen**, e **-umloverviewgen** em combinação.

14.2.2.2 - Destino da documentação

Local onde o javadoc gerará a documentação. É dado pelo parâmetro **-d**. Neste exemplo, **-d C:\Desenvolvimento\ydocdocumentacao**.

14.2.2.3 - Formato da imagem

-umlfileformat formatname

Overrides the `uml_file_format` property in `resources/ydoc.cfg`.

Formatos suportados: GIF, JPG, PNG, SVG (Scalable Vector Graphics, a XML-based vector graphics

Format), SVGZ (Compressed SVG), SWF (Shockwave Flash, a popular binary vector graphicsformat).

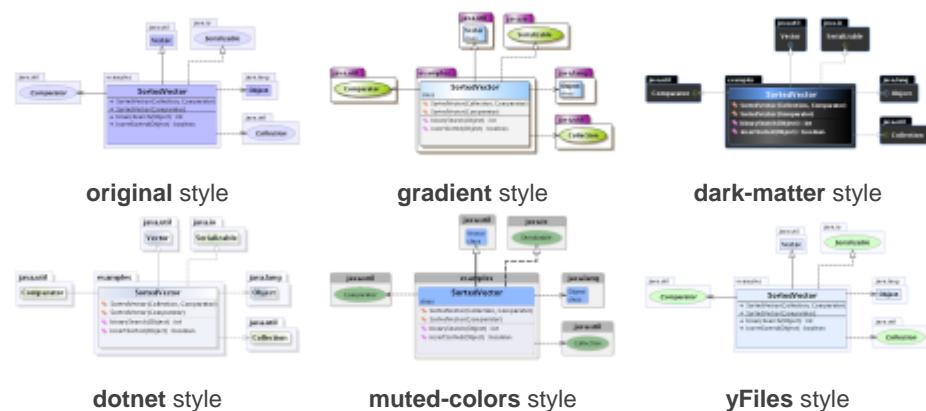
14.2.2.4 - Estilos de formatação

O estilo default é o `original-style.xml`.

Os estilos reside no seguinte diretório: `C:\Desenvolvimento\ydoc\resources\styles`

	dark-matter-style.xml	12/04/2013 19:47	Documento XML	5 KB
	default-style.xml	12/04/2013 19:47	Documento XML	4 KB
	dotnet-style.xml	12/04/2013 19:47	Documento XML	5 KB
	gradient-style.xml	12/04/2013 19:47	Documento XML	5 KB
	muted-colors-style.xml	12/04/2013 19:47	Documento XML	6 KB
	original-style.xml	12/04/2013 19:47	Documento XML	6 KB
	theBlues-style.xml	12/04/2013 19:47	Documento XML	7 KB
	ydocstyle.xsd	12/04/2013 19:47	Arquivo XSD	11 KB
	yFiles-style.xml	12/04/2013 19:47	Documento XML	6 KB

Exemplos de estilos (skins) :



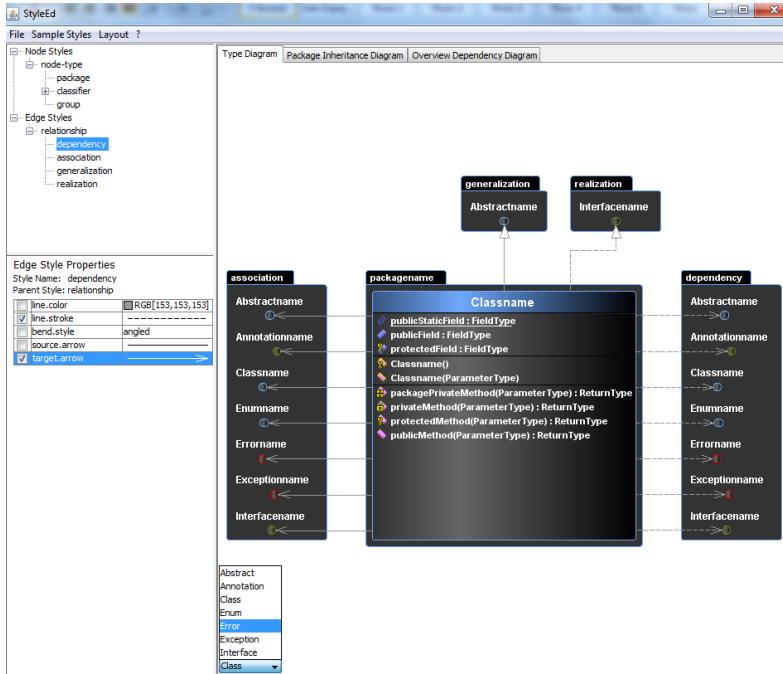
14.2.2.5 - Alterando o estilo

Acessar e alterar o arquivo de configuração do UMLDoclet (ver abaixo), informando o estilo css desejado.

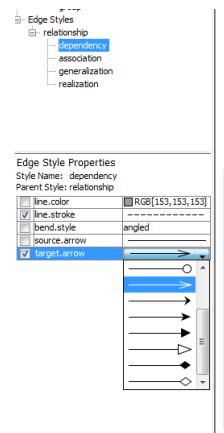
14.2.2.6 - Customizando um estilo

Além dos estilos que já vem com o produto, pode-se customizar um estilo. Existem dezenas de opções, como por exemplo: tipo de seta para os relacionamentos, cor, tamanho e etc.

Executar a aplicação: C:\Desenvolvimento\ydoc\lib\styleed.jar



Alterando a seta para um relacionamento de dependência.



14.2.2.7 – Arquivo onde reside a configuração do UMLDoclet:

Os parâmetros passados em runtime ao UMLDoclet fazem um override nos parâmetros no arquivo de configuração.

Arquivo de configuração: C:\Desenvolvimento\ydoc\resources\ydoc.cfg

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration xmlns="http://www.yworks.com/xml/ydoc">
  - <group name="diagrams">
    - <group name="overview">
      - <group name="diagram">
        <property name="style" value=".styles/original-style.xml"/>
        <property name="type" value="dependency"/>
        <property name="id" value="0"/>
        + <group name="layout">
        + <group name="insets">
        + <group name="include">
      </group>
    - <group name="diagram">
      <property name="style" value=".styles/original-style.xml"/>
      <property name="type" value="inheritance"/>
      <property name="id" value="1"/>
      + <group name="layout">
      + <group name="insets">
      + <group name="include">
    </group>
  - <group name="package">
    - <group name="diagram">
      <property name="style" value=".styles/original-style.xml"/>
      + <group name="layout">
      + <group name="insets">
      + <group name="include">
    </group>
  - <group name="type">
    - <group name="diagram">
      <property name="style" value=".styles/original-style.xml"/>
      + <group name="include">
        <!-- Sample exclude group. All classes in java.* packages are excluded
        <!-- <group name="exclude"> <group name="pattern"> <property na
        <property name="realizations" value="javax.*"/> </group> </group>
      + <group name="insets">
    - <group name="order">
      <property name="fields" value="mod-lex"/>
      <property name="constructors" value="lex-ic"/>
      <property name="methods" value="lex-ic"/>
    </group>
    + <group name="layout">
  </group>
  - <group name="formats">
    <property name="fileformat" value="PNG"/>
    + <group name="vectorgraphics">
    + <group name="image">
  </group>
+ <group name="misc">
</configuration>
```

Estilos CSS

Formato Imagem