

1.	Introdução .....	1
2.	Criação e Ativação .....	2
3.	Considerações Prévias.....	3
4.	Classes API .....	18
5.	Eventos.....	33
6.	Operações.....	54
7.	InfraPHP.....	92
8.	Padrão de Modelagem de Dados.....	106
9.	Padrão de Codificação PHP.....	109
10.	Gerador de Código .....	111

## 1. Introdução

O sistema possui uma API (Application Programming Interface) que permite o desenvolvimento de módulos sendo possível:

- adicionar botões e/ou ícones específicos da instituição em diversas funcionalidades como no Controle de Processos e na visualização da árvore de documentos;
- incluir itens de menu nas telas de usuários externos e pesquisa de publicações;
- interceptar eventos do sistema para tratamento específico como assinatura de documentos, envio de processos e cancelamento de documentos;
- realizar diversas operações no sistema como geração de processos, lançamento de andamentos no histórico e inclusão de documentos.

Os módulos podem ser desenvolvidos seguindo os padrões de cada instituição. Entretanto na seção InfraPHP consta uma descrição resumida do framework do sistema caso se deseje utilizá-lo. Algumas classes do framework são descritas ao longo deste documento pois a interação com elas será obrigatória devido a questões de segurança e integração com o sistema.

Para evitar conflitos com outros módulos recomendamos que ao nomear os artefatos como classes, recursos e tabelas seja utilizado um prefixo identificando a instituição. O documento contém também seções descrevendo o padrão de modelagem de dados e de codificação PHP utilizados no sistema.

## 2. Criação e Ativação

Os passos básicos para criação de um módulo são:

- a) adicionar um diretório dentro do diretório de módulos do sistema (sei/web/modulos) identificando a instituição e outro identificando o módulo:



- b) criar uma classe que estenda a classe SeiIntegracao e implemente os métodos getNome, getVersao e getInstituicao:

```
class AbcExemploIntegracao extends SeiIntegracao{

    public function getNome(){
        return 'Módulo de exemplos ABC';
    }

    public function getVersao() {
        return '1.0.0';
    }

    public function getInstituicao(){
        return 'TRF4 - Tribunal Regional Federal da 4ª Região';
    }
}
```

Esta classe deve ser salva em um arquivo com o mesmo nome (AbcExemploIntegracao.php).

- c) adicionar no arquivo de configuração do sistema na chave Modulos a referência para o nome da classe e o diretório onde ela se encontra:

```
'SEI' => array( ...
    'Modulos' => array( 'AbcExemploIntegracao' => 'abc/exemplo' )
),
```

- d) por fim verificar se o módulo foi carregado por meio do menu Infra/Módulos:

TRIBUNAL REGIONAL FEDERAL DA 4ª REGIÃO

sei Teste Para saber+ Menu Pesquisa [ ] [TESTE]

Administração ▸

Controle de Processos

Iniciar Processo

Retorno Programado

Pesquisa

Base de Conhecimento

Modelos Favoritos

Textos Padrão

### Módulos

Lista de Módulos Carregados (1 registro):

Nome	Versão	Instituição
Módulo de exemplos ABC	1.0.0	TRF4 - Tribunal Regional Federal da 4ª Região

### 3. Considerações Prévias

Antes de iniciar a construção do módulo é necessário conhecer o funcionamento básico do sistema em relação ao SIP e outras questões como o controle de transações e auditoria.

#### Perfis, Recursos e Menus

Todas as operações realizadas no sistema estão atreladas a recursos cadastrados no SIP (Sistema de Permissões). Os recursos podem ou não estar associados com itens de menu. Os recursos e itens de menu são agrupados em perfis que são então liberados para os usuários por meio das permissões. Os itens de menu serão montados automaticamente no login mas outros elementos associados com recursos (como botões e ícones) devem ser tratados via programação (ver seção Classes do Sistema, classe SessaoSEI, método verificarPermissao). A seguir são demonstrados os passos para adicionar um item de menu em um novo perfil.

1) criar o recurso no SIP (Recursos/Novo) sendo recomendado que recursos de módulos tenham o prefixo "md\_<instituição/módulo>":

2) adicionar item de menu associado com o recurso (Menus/Montar):

**Adicionar Item de Menu**

Órgão do Sistema: TRF4

Sistema: SEI

Menu: Principal

☒ Raiz

Recurso: md\_abc\_sala\_reservar

Rótulo: Reserva de Salas

Descrição:

Sequência: 9999

☐ Abrir em uma nova janela

3) criar um perfil específico da instituição ABC (Perfis/Novo) sendo recomendado que perfis de módulos tenham o prefixo "MD\_<instituição/módulo>":

**Novo Perfil**

Órgão do Sistema: TRF4

Sistema: SEI

Nome: MD\_ABC\_Básico

Descrição: Perfil específico da instituição ABC

☐ Disponível aos Coordenadores de Unidade

4) Adicionar o recurso e item de menu no perfil (Perfis/Montar):

**Montar Perfil**

Órgão do Sistema: TRF4

Sistema: SEI

Perfil: MD\_ABC\_Básico

Recurso: sala

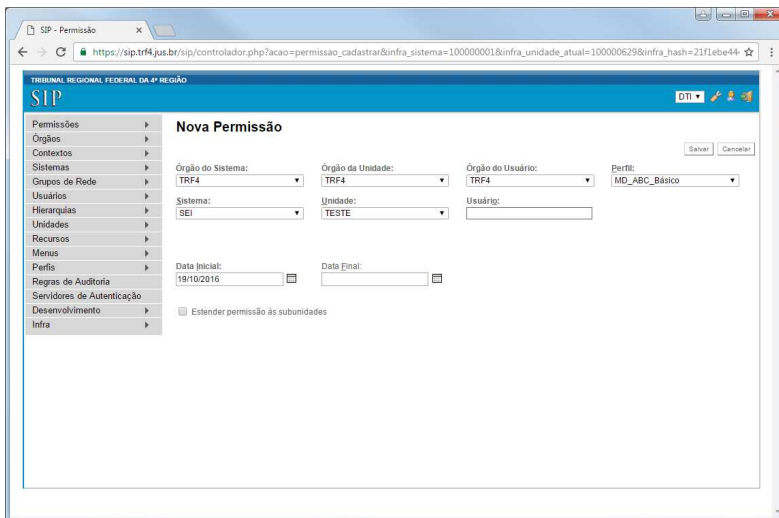
☐ Visualizar Somente Recursos do Perfil

☐ Visualizar Descrição do Recurso

Lista de Recursos (1 registro):

Nome	Perfil	Menu
md_abc_sala_reservar	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Reserva de Salas

5) atribuir permissão no perfil da instituição ABC:



6) O item de menu deve aparecer no SEI (necessário logar novamente se já estava logado):

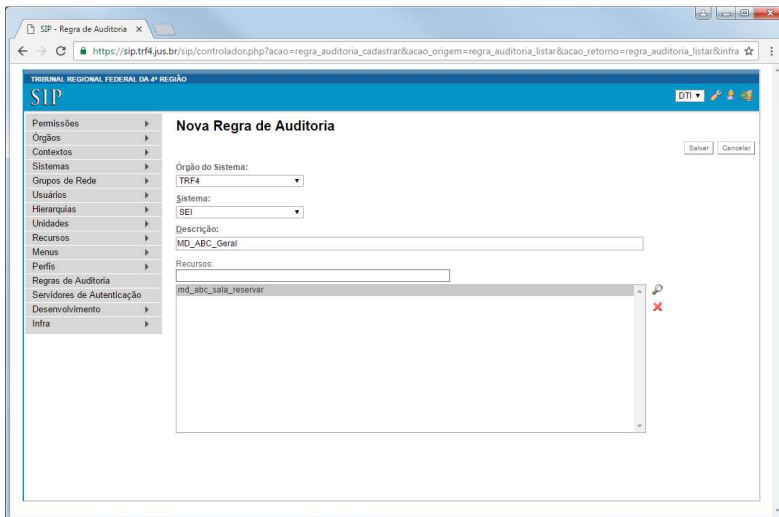


As telas de usuário externo e pesquisa de publicações não recebem itens de menu do SIP. Entretanto é possível adicionar itens através dos eventos `montarMenuUsuarioExterno` e `montarMenuPublicacoes` (ver seção Eventos).

## Auditoria

Para utilizar o mecanismo de auditoria é necessário:

a) criar uma regra de auditoria no SIP e adicionar o recurso sendo recomendado que as regras de módulos tenham o prefixo "MD\_<instituição/módulo>":



b) No método da camada de regra de negócio que realiza a operação incluir uma chamada para `validarAuditarPermissao` que recebe como parâmetros o nome do recurso, o nome do método e os valores associados que devem ser gravados na tabela de auditoria. Este método também lançará um erro de acesso negado ao recurso se o usuário não tiver permissão. Se o recurso não fizer parte de nenhuma regra de auditoria então apenas a validação de permissão será realizada.

```
protected function lancarAndamentosControlado($arrParametros){
    try{

        SessaoSEI::getInstance()->validarAuditarPermissao('md_abc_andamento_lancar', __METHOD__,
        $arrParametros);

        //processamento

    }catch(Exception $e){
        throw new InfraException('Erro lançando andamentos do módulo ABC.', $e);
    }
}
```

As operações auditadas poderão ser consultadas por meio do menu Infra/Auditoria do SEI.

## Conexões e Transações

O controle das conexões e transações no banco de dados pode ser feito manipulando diretamente o objeto `BancoSEI::getInstance()` ou utilizando os sufixos "Conectado" e "Controlado". Ver mais detalhes na seção Classes do Sistema, classe `BancoSEI` e na seção InfraPHP, classe `InfraRN`.

Se o módulo realizar processamento de operações no sistema como, por exemplo, geração de processos e inclusão de documentos então será obrigatório utilizar um dos controles descritos abaixo.

- Controle manual de conexões e transações

Neste caso basta chamar os métodos diretamente do objeto de banco. Ex.:

```

class MdAbcTesteRN {

    public function processarXXXXX($a, $b){
        try{

            BancoSEI::getInstance()->abrirConexao();
            BancoSEI::getInstance()->abrirTransacao();

            //executa comandos

            BancoSEI::getInstance()->confirmarTransacao();
            BancoSEI::getInstance()->fecharConexao();

        }catch(Exception $e){

            try{
                BancoSEI::getInstance()->cancelarTransacao();
            }catch(Exception $e2){}

            try{
                BancoSEI::getInstance()->fecharConexao();
            }catch(Exception $e2){}

            throw new InfraException('Erro processando operação XXXXX no módulo ABC.', $e);
        }
    }
}

```

Pode ser um problema quando existirem chamadas entre métodos que precisam abrir conexões/transações de forma isolada ou em conjunto. Neste caso é necessário implementar um mecanismo de controle auxiliar para saber se a conexão ou transação já foi aberta.

- Controle automático de conexões e transações

O controle automático pode ser feito utilizando os sufixos reservados:

Conectado - abre uma conexão se já não estiver aberta

Controlado - abre uma conexão e uma transação se já não estiverem abertas

A chamada entre métodos é transparente mantendo o escopo da conexão ou transação. Para utilizar este mecanismo:

- 1) criar uma classe filha de InfraRN;
- 2) implementar o método inicializarObjInfraIBanco retornando a instância do banco de dados;
- 3) criar um método protected com o sufixo Conectado ou Controlado;
- 4) este método não aceita múltiplos parâmetros então deve ser utilizado um objeto DTO ou um array encapsulador.

Utilizando este mecanismo após a primeira chamada a um método Conectado ou Controlado a conexão ficará aberta até o final da execução do script. Transações vão respeitar o mesmo escopo do método sendo confirmadas no final da execução do método ou canceladas caso uma exceção seja lançada.

```

class MdAbcTesteRN extends InfraRN {

    protected function inicializarObjInfraIBanco(){
        return BancoSEI::getInstance();
    }

    protected function processarXXXXXControlado($arrParametros){
        try{

            //executa comandos

        }catch(Exception $e){

```

```

        throw new InfraException('Erro processando operação XXXXX no módulo ABC.', $e);
    }
}

```

Atenção para o uso destes sufixos em construções como abaixo pois isso pode causar inconsistências em caso de erro e também problemas de desempenho:

```

protected function processarTudoConectado(){ //abre apenas conexão

    $arr = $this->buscarItensBanco();

    foreach($arr as $itens){
        $this->processarXXXXX($itens); //abre e fecha uma transação PARA CADA elemento do array
    }
}

```

Neste exemplo se houver um erro no processamento alguns itens já podem ter sido gravados no banco. Analisar se o método gerente processarTudo não deveria ser Controlado garantindo uma transação única.

## Andamentos

Os andamentos representam os registros exibidos no histórico de processo. No SEI os andamentos podem ter variáveis que serão substituídas por atributos informados no momento do lançamento. Cada andamento está associado com um registro de tarefa que indica o tipo do andamento. Tarefas com identificador menor que 1000 são reservadas do SEI. A única tarefa reservada que o módulo pode lançar é a de ID\_TAREFA=65 que representa uma atualização do andamento com texto livre (neste caso informar um atributo com NOME="DESCRICAO" e VALOR="texto do andamento").

Tarefas de módulos podem ser cadastradas com o comando abaixo:

```

INSERT INTO tarefa (
    id_tarefa,
    nome,
    sin_historico_resumido,
    sin_historico_completo,
    sin_lancar_andamento_fechado,
    sin_permite_processo_fechado,
    sin_fechar_andamentos_abertos,
    id_tarefa_modulo)
VALUES (
    ' . BancoSEI::getInstance()->getValorSequencia('seq_tarefa') . ',
    'Audiência agendada no prédio @PREDIO@ (sala @SALA@)',
    'S',
    'S',
    'N',
    'S',
    'S',
    'MD_ABC_AUDIENCIA_REALIZADA')

```

Onde:

- id\_tarefa

Deve ser maior ou igual a 1000, se a inserção for realizada sem a chamada ao método getValorSequencia('seq\_tarefa') então é necessário buscar o maior ID da tabela de tarefas:

```

select max(id_tarefa)+1 from tarefa

```



Se o valor retornado for maior ou igual a 1000 utilizar este valor, caso contrário utilizar 1000. Depois atualizar a sequência seq\_tarefa:

#### MySQL

```
alter table seq_tarefa AUTO_INCREMENT = [último valor utilizado];
```

#### SQL Server

```
DBCC CHECKIDENT ('seq_tarefa', RESEED, [último valor utilizado]);
```

#### Oracle

```
drop sequence seq_tarefa;  
CREATE SEQUENCE seq_tarefa START WITH [último valor utilizado] INCREMENT BY 1 NOCACHE NOCYCLE;
```

- nome

Representa o texto que será exibido no andamento. É possível utilizar variáveis no texto que devem ter o valor informado por meio de atributos ao lançar o andamento. Para o texto "Audiência agendada no prédio @PREDIO@ (sala @SALA@)" é necessário informar o valor de PREDIO E SALA, ex.:

```
$objEntradaLancarAndamentoAPI = new EntradaLancarAndamentoAPI();  
$objEntradaLancarAndamentoAPI->setIdProcedimento(10000000012057);  
$objEntradaLancarAndamentoAPI->setIdTarefaModulo('MD_ABC_AUDIENCIA_REALIZADA');  
  
$arrObjAtributoAndamentoAPI = array();  
  
$objAtributoAndamentoAPI = new AtributoAndamentoAPI();  
$objAtributoAndamentoAPI->setNome('PREDIO');  
$objAtributoAndamentoAPI->setValor('101');  
$objAtributoAndamentoAPI->setIdOrigem(54); //ID do prédio, pode ser null  
$arrObjAtributoAndamentoAPI[] = $objAtributoAndamentoAPI;  
  
$objAtributoAndamentoAPI = new AtributoAndamentoAPI();  
$objAtributoAndamentoAPI->setNome('SALA');  
$objAtributoAndamentoAPI->setValor('3A');  
$objAtributoAndamentoAPI->setIdOrigem(9334); //ID da sala, pode ser null  
$arrObjAtributoAndamentoAPI[] = $objAtributoAndamentoAPI;  
  
$objEntradaLancarAndamentoAPI->setAtributos($arrObjAtributoAndamentoAPI);  
  
$objSeiRN = new SeiRN();  
$objSeiRN->lançarAndamento($objEntradaLancarAndamentoAPI);
```

Algumas variáveis de andamento são reservadas do sistema mas podem ser utilizadas pelos módulos desde que as informações sejam devidamente preenchidas nos atributos:

Nome	Valor	IdOrigem	Observação
DOCUMENTO	Número do documento	ID do documento	Gera link para o documento
DOCUMENTOS	Null	null	Será substituída pelos links de todos os documentos informados nos atributos DOCUMENTO associados ao andamento separados por vírgula (agrupador)
NIVEL_ACESSO	null	0 - Público 1 - Restrito 2 - Sigiloso	
GRAU_SIGILO	null	U - Ultrassecreto	

		S - Secreto R - Reservado	
HIPOTESE_LEGAL	null	ID da hipótese legal	
VISUALIZACAO		I - Integral P - Parcial N - Nenhum	liberação de acesso externo
DATA_AUTUACAO	Data no formato dd/mm/aaaa	Null	
TIPO_CONFERENCIA	null	ID do tipo de conferência	
PROCESSO	Número do processo	ID do processo	Gera link para o processo
USUARIO	[sigla]¥[Nome]	ID do usuário	
USUARIOS	Null	null	Será substituída por todos os atributos USUARIO associados ao andamento separados por vírgula (agrupador)
UNIDADE	[sigla]¥[Descrição]	ID da unidade	
BLOCO	ID do bloco	ID do bloco	
DATA_HORA	Data/hora no formato dd/mm/aaaa H:M:S	ID da atividade	processos sigilosos
USUARIO_ANULACAO	[sigla]¥[nome]	ID da atividade	processos sigilosos
INTERESSADO	[sigla]¥[nome]	ID do participante	
LOCALIZADOR	Identificação do localizador	ID do localizador	
ANEXO	Nome do anexo	ID do anexo	

- **sin\_historico\_resumido**  
S/N - Indica se o andamento será exibido no histórico resumido.
- **sin\_historico\_completo**  
S/N - Indica se o andamento será exibido no histórico completo.
- **sin\_lancar\_andamento\_fechado**

S/N - Indica se o andamento deverá ser lançado como aberto ou concluído. Andamento aberto aparece em amarelo na consulta do histórico informando a última atividade realizada na unidade. Se o módulo tentar lançar um andamento aberto em processo concluído na unidade o sistema lançará um erro "Processo X não possui andamento aberto na unidade Y.". Para evitar este erro ver configuração do sinalizador **sin\_permite\_processo\_fechado**. Mas é importante verificar se não existe uma falha na lógica do módulo pois talvez seja necessário reabrir o processo automaticamente ao invés de permitir o lançamento concluído. Se o processo estiver concluído e o andamento for lançado como concluído então ninguém na unidade perceberá que ocorreu alteração no histórico.

- **sin\_permite\_processo\_fechado**

S/N - Indica se permite lançar o andamento em processo concluído na unidade. Se o módulo tentar lançar um andamento aberto e o processo estiver concluído na unidade então poderá lançar automaticamente como andamento concluído.

- **sin\_fechar\_andamentos\_abertos**

S/N - Indica se os registros de andamento em aberto na unidade devem ser concluídos. Se o processo estará aberto na unidade na hora do lançamento então utilizar o valor S. Desta forma apenas o novo andamento ficará em amarelo no histórico de processo indicando a última atividade.

- **id\_tarefa\_modulo**

Identificador da tarefa para o módulo com até 50 caracteres maiúsculos. Utilizar como prefixo MD\_ + sigla da instituição. Possibilita lançar/consultar andamentos do módulo sem a necessidade de conhecer o ID interno da tarefa que pode ser diferente em outra instalação. Não devem existir valores duplicados para este campo. Ex.: MD\_ABC\_AUDIENCIA\_REALIZADA

## Classes do Sistema

### SessaoSEI

Fornece acesso aos dados da sessão do usuário. Os atributos configurados pelo módulo podem ser consultados por meio do menu Infra/Atributos de Sessão (necessária permissão no perfil Informática). Esta classe implementa o padrão de projeto Singleton então na programação sempre acessar o objeto por meio do método estático getInstance().

Não é recomendado acessar diretamente o array \$\_SESSION do PHP.

Métodos de informação disponíveis:

<b>Método</b>	<b>Descrição</b>
getNumIdUsuario()	ID do usuário no SIP
getStrIdOrigemUsuario()	ID Origem associado com o usuário no SIP
getStrSiglaUsuario()	Sigla do Usuário
getStrNomeUsuario()	Nome do Usuário
getNumIdOrgaoUsuario()	ID do órgão do usuário
getStrSiglaOrgaoUsuario()	Sigla do órgão do usuário
getStrDescricaoOrgaoUsuario()	Descrição do órgão do usuário
getNumIdUnidadeAtual()	ID da unidade atual de trabalho
getStrIdOrigemUnidadeAtual()	ID Origem associado com a unidade atual de trabalho no SIP
getStrSiglaUnidadeAtual()	Sigla da unidade atual de trabalho
getStrDescricaoUnidadeAtual()	Descrição da unidade atual de trabalho
getNumIdOrgaoUnidadeAtual()	ID do órgão da unidade atual de trabalho
getStrSiglaOrgaoUnidadeAtual()	Sigla do órgão da unidade atual de trabalho
getStrDescricaoOrgaoUnidadeAtual()	Descrição do órgão da unidade atual de trabalho
getStrSinAcessibilidade()	S/N - indica se o usuário requer tratamento especial quanto a acessibilidade

Outros métodos:

❑ *assinarLink(\$strLink)*

Assina um link associado com a sessão do usuário evitando alteração nos parâmetros. Se ocorrer alteração nos parâmetros um erro de hash inválido será lançado redirecionando o usuário para a tela de login.

```
SessaoSEI::getInstance()->assinarLink('controlador.php?acao=...');
```

❑ *validarLink(\$strLink=null)*

Valida assinatura do link e se não for válido automaticamente lança um erro de "Hash inválido" e redireciona o usuário para a tela de login. Se o parâmetro não for informado então irá validar o link da página atual. Utilizado normalmente na entrada de páginas garantindo que o usuário acessou a tela por um link válido.

```
SessaoSEI::getInstance()->validarLink();
```

❑ *verificarLink(\$strLink=null)*

Realiza o mesmo comportamento do método validarLink retornando true/false ao invés de lançar um erro e redirecionar para o login.

❑ *validarPermissao(\$strNomeRecurso)*

Verifica se o usuário tem acesso ao recurso informado. Se o usuário não tiver acesso ao recurso será lançado o erro "Acesso negado a este recurso nesta unidade". Utilizado normalmente na entrada de páginas para validar se o usuário tem acesso ao recurso apontando pelo parâmetro "acao" da URL.

```
SessaoSEI::getInstance()->validarPermissao($_GET['acao']);
```

❑ *verificarPermissao(\$strNomeRecurso)*

Mesmo comportamento do método validarPermissao retornando true/false ao invés de lançar um erro. Utilizado para montar componentes na interface como botões e ícones e para validar a permissão nos métodos das classes de Regra de Negócio.

```
if (SessaoSEI::getInstance()->verificarPermissao('md_abc_andamento_lancar')) {  
    //monta botão correspondente  
}
```

❑ *setAtributo(\$strNome, \$strValor)*

Configura um atributo na sessão do usuário. Para evitar nomes duplicados utilizar o prefixo "MD\_<instituição/módulo>" no nome do atributo.

```
SessaoSEI::getInstance()->setAtributo('MD_ABC_ATRIBUTO_SESSAO',$strValor);
```

❑ *getAtributo(\$strNome)*

Recupera um atributo da sessão do usuário.

```
$strValor = SessaoSEI::getInstance()->getAtributo('MD_ABC_ATRIBUTO_SESSAO');
```

❑ *isSetAtributo(\$strNome)*

Verifica se um atributo consta na sessão do usuário retornando true/false.

```
if (SessaoSEI::getInstance()->isSetAtributo('MD_ABC_ATRIBUTO_SESSAO')){  
}
```

❑ *removerAtributo(\$strNome)*

Remove um atributo da sessão do usuário.

```
SessaoSEI::getInstance()->removerAtributo('MD_ABC_ATRIBUTO_SESSAO');
```

## CacheSEI

Fornece acesso ao servidor de cache em memória (memcache). Os atributos podem ser consultados por meio do menu Infra/Cache em Memória sendo necessária permissão no perfil Informática. Esta classe implementa o padrão de projeto Singleton então na programação sempre acessar o objeto por meio do método estático getInstance().

❑ *setAtributo(\$strChave, \$strValor, \$numTempo)*

Adiciona um atributo no servidor memcache, onde:

\$strChave	Nome do atributo utilizando com o prefixo "MD_<instituição/módulo>" para evitar nomes duplicados.
\$strValor	Valor do atributo diferente de nulo.
\$numTempo	Tempo que o atributo deve ficar na cache. Utilizar CacheSEI::getInstance()->getNumTempo() para obter o valor configurado na chave CacheSEI/Tempo do arquivo de configuração (se a chave não for definida no arquivo de configuração então será retornado o valor padrão do sistema 3600 = 1 hora).

```
CacheSEI::getInstance()->setAtributo('MD_ABC_ATRIBUTO_MEMCACHE', $strValor,  
CacheSEI::getInstance()->getNumTempo());
```

❑ *getAtributo(\$strChave)*

Recupera um atributo no servidor memcache retornando nulo se não encontrar.

```
$strValor = CacheSEI::getInstance()->getAtributo('MD_ABC_ATRIBUTO_MEMCACHE');
```

## InfraParametro

Permite ler/gravar parâmetros na tabela infra\_parametro (menu Infra/Parâmetros). Ao criar o objeto deve ser passada a instância do banco de dados:

```
$objInfraParametro = new InfraParametro(BancoSEI::getInstance());
```

❑ *setValor(\$strNome, \$strValor)*

Configura o valor de um parâmetro. Se não existir será criado e se existir será atualizado. Para evitar nomes duplicados utilizar o prefixo "MD\_<instituição/módulo>".

```
$objInfraParametro->setValor('MD_ABC_ID_SERIE_TESTE', '601');
```

❑ *getValor(\$strNome, \$bolErroNaoEncontrado=true)*

Retorna o valor de um parâmetro ou nulo se não encontrar. O segundo parâmetro do método é opcional e indica se deve ser lançado um erro caso o parâmetro não exista na tabela.

```
$numIdSerieAbc = $objInfraParametro->getValor('MD_ABC_ID_SERIE_TESTE');
```

❑ *isSetValor(\$strNome)*

Verifica se um parâmetro existe na tabela retornando true/false. O valor do parâmetro não é analisado, ou seja, ele pode existir mas estar vazio.

```
if ($objInfraParametro->isSetValor('MD_ABC_ID_SERIE_TESTE')){  
}
```

❑ *listarValores(\$arrNomes, \$bolErroNaoEncontrado=true)*

Retorna conjunto de parâmetros informados no array de entrada. O segundo parâmetro do método é opcional e indica se deve ser lançado um erro caso algum parâmetro não exista na tabela. O array de retorno é indexado pelo nomes dos parâmetros.

```
$arrSeries = $objInfraParametro->listarValores(array('MD_ABC_PAR1', 'MD_ABC_PAR2'));
```

## LogSEI

Permite ler/gravar registros na tabela `infra_log` (menu `Infra/Log`). Esta classe implementa o padrão de projeto Singleton então na programação sempre acessar o objeto por meio do método estático `getInstance()`.

❑ *gravar(\$strTexto, \$strStaTipo='E')*

Grava o texto informado na tabela `infra_log` (não pode ser vazio). O segundo parâmetro é opcional e indica o tipo do registro. Utilizar uma das constantes abaixo:

```
InfraLog::$ERRO (default)  
InfraLog::$AVISO  
InfraLog::$INFORMACAO  
InfraLog::$DEBUG
```

```
LogSEI::getInstance()->gravar('abc teste log',InfraLog::$INFORMACAO);
```

## InfraDebug

Permite ler/gravar informações de debug armazenando os registros na sessão (sendo assim mantidos entre chamadas do sistema). Esta classe implementa o padrão de projeto Singleton então na programação sempre acessar o objeto por meio do método estático `getInstance()`.

Se utilizando a InfraPHP para realizar debug nas telas do sistema deve-se configurar no início da execução as opções desejadas:

```
InfraDebug::getInstance()->setBolLigado(true);  
InfraDebug::getInstance()->setBolDebugInfra(true);  
InfraDebug::getInstance()->limpar();
```

E no final retirar o comentário do método que monta a área de debug para visualização:

```
PaginaSEI::getInstance()->montarAreaDebug();
```

Atenção: não esquecer de desligar o debug após a sua utilização pois poderá comprometer o desempenho e a segurança do sistema.

❑ *gravar(\$strTexto)*

Grava um registro de debug na sessão.

```
InfraDebug::getInstance()->gravar('Valor encontrado: ' . $n);
```

❑ *limpar()*

Remove todos os registros que foram gravados no debug.

```
InfraDebug::getInstance()->limpar();
```

❑ *getStrDebug()*

Recupera todos os registros que foram gravados.

```
$strDebug = InfraDebug::getInstance()->getStrDebug();
```

❑ *setBolLigado(\$bolLigado)*

Recebe true/false indicando se as chamadas ao método "gravar" devem ser processadas ou ignoradas.

```
InfraDebug::getInstance()->setBolLigado(true);
```

❑ *setBolDebugInfra(\$bolDebugInfra)*

Recebe true/false indicando se deve registrar textos de debug gerados automaticamente pela InfraPHP contendo, por exemplo, os SQLs executados.

```
InfraDebug::getInstance()->setBolDebugInfra(true);
```

❑ *setBolEcho(\$bolEcho)*

Recebe true/false indicando se na chamada do método gravar deve ser feito um "echo" automaticamente do conteúdo.

```
InfraDebug::getInstance()->setBolEcho(true);
```

## ConfiguracaoSEI

Permite ler informações do arquivo de configurações do sistema "ConfiguracaoSEI.php". Esta classe implementa o padrão de projeto Singleton então na programação sempre acessar o objeto por meio do método estático getInstance().

- ❑ *getValor(\$strGrupo, \$strChave=null, \$bolErroNaoEncontrado=true, \$strValorPadrao=null)*

Recupera o valor associado com a chave no arquivo de configuração, onde:

\$strGrupo	Nome do grupo com o prefixo "MD_<instituição/módulo>" para evitar nomes duplicados.
\$strChave	Opcional. Nome da chave dentro do grupo.
bolErroNaoEncontrado	Opcional (valor padrão true). Indica se deve ser lançado um erro caso a configuração não exista na tabela.
strValorPadrao	Opcional. Informa o valor padrão para retorno se a configuração não existir.

```
$strServidor = ConfiguracaoSEI::getInstance()->getValor('MD_ABC_Banco','Servidor');
```

- ❑ *isSetValor(\$strGrupo, \$strChave)*

Retorna true/false indicando se uma configuração existe no arquivo de configuração. O valor da configuração não é analisado, ou seja, ela pode existir mas estar vazia.

```
if (ConfiguracaoSEI::getInstance()->isSetValor('MD_ABC_Banco','Servidor')){  
}
```

## BancoSEI

Fornece acesso ao banco de dados. Esta classe implementa o padrão de projeto Singleton então na programação sempre acessar o objeto por meio do método estático getInstance().

- ❑ *abrirConexao()*

Abre conexão com o banco de dados.

```
BancoSEI::getInstance()->abrirConexao();
```

- ❑ *fecharConexao()*

Fecha conexão com o banco de dados.

```
BancoSEI::getInstance()->fecharConexao();
```

- ❑ *getIdConexao()*



Retorna um identificador da conexão ou nulo. Este identificador não representa um recurso de conexão do PHP, ou seja, não pode ser utilizado em chamadas diretas da extensão do banco de dados utilizado. Por questões de segurança o recurso de conexão é privado e controlado pela classe de banco de dados do framework.

```
If (BancoSEI::getInstance()->getIdConexao()==null){  
    BancoSEI::getInstance()->abrirConexao();  
}
```

#### ❑ *abrirTransacao()*

Inicia uma transação.

```
BancoSEI::getInstance()->abrirTransacao();
```

#### ❑ *confirmarTransacao()*

Confirma uma transação (commit).

```
BancoSEI::getInstance()->confirmarTransacao();
```

#### ❑ *cancelarTransacao()*

Cancela uma transação (rollback).

```
BancoSEI::getInstance()->cancelarTransacao();
```

#### ❑ *consultarSql(\$strSql)*

Executa a consulta recebida no banco de dados retornando um array com os registros recuperados.

```
$rs = BancoSEI::getInstance()->consultarSql('select nome, valor from infra_parametro  
where nome like \'SEI_%\' order by nome asc');
```

```
Array  
(  
    [0] => Array  
        (  
            [nome] => SEI_HABILITAR_AUTENTICACAO_DOCUMENTO_EXTERNO  
            [valor] => 2  
        )  
    [1] => Array  
        (  
            [nome] => SEI_HABILITAR_GRAU_SIGILO  
            [valor] => 1  
        )  
    [2] => Array  
        (  
            [nome] => SEI_HABILITAR_HIPOTESE_LEGAL  
            [valor] => 1  
        )  
    ...  
)
```

#### ❑ *executarSql(\$strSql, \$arrCamposBind=null)*

Executa o comando recebido retornando o número de registros afetados. O parâmetro \$arrCamposBind é utilizado apenas em bases Oracle sendo um array indexado pelos nomes dos campos CLOB para bind.

```
BancoSEI::getInstance()->executarSql('update infra_parametro set valor =\'2\' where nome=\'SEI_HABILITAR_HIPOTESE_LEGAL\') ;
```

## 4. Classes API

A troca de informações do sistema com os módulos deve ser feita utilizando as classes específicas da API (diretório sei/web/api). Na classe de integração do módulo ao interceptar eventos o sistema disponibilizará objetos destas classes com atributos específicos preenchidos. Da mesma forma quando o módulo realizar uma operação por meio da classe SeiRN deverá criar objetos destas classes informando os atributos adequados. Ver mais detalhes nas seções Eventos e Operações.

**Não é recomendado o uso de outras classes ou objetos internos do sistema.**

### AcessoExternoAPI

IdAcessoExterno	Identificador do Acesso Externo
DataValidade	Data de validade do acesso externo
Procedimento	Ocorrência de ProcedimentoAPI
Documento	Ocorrência de DocumentoAPI
SinAcessoProcesso	S/N - indica se o acesso externo possibilita visualização integral do processo

### AndamentoAPI

IdAndamento	Identificador do Andamento
IdTarefa	Identificador da tarefa associada
IdTarefaModulo	Identificador da tarefa de módulo associada
Descricao	Descrição do andamento
DataHora	Data/hora do andamento
Usuario	Ocorrência de UsuarioAPI
Unidade	Ocorrência de UnidadeAPI
Atributos	Conjunto de ocorrências de AtributoAndamentoAPI
IdProtocolo	Identificador do protocolo do andamento

### AndamentoMarcadorAPI

IdAndamentoMarcador	Identificador do andamento de marcador
Texto	Texto do andamento
DataHora	Data/hora do andamento
Usuario	Ocorrência de UsuarioAPI
Marcador	Ocorrência de MarcadorAPI

## ArquivoExtensaoAPI

IdArquivoExtensao	Identificador do registro
Extensao	Extensão de arquivo
Descricao	Descrição da extensão

## ArvoreAcaoItemAPI

Tipo	Rótulo que serve como agrupador para nós do mesmo tipo (caso seja necessário varrer todos os nós posteriormente filtrando pelo tipo). Para agrupar os itens do módulo utilizar MD + instituição + tipo:  <code>\$objArvoreAcaoItemAPI-&gt;setTipo('MD_ABC_AUDIENCIAS');</code>
Id	Identificador do nó na árvore. Deve ser único sendo recomendado utilizar o MD + instituição + descrição + ID do protocolo, ex.:  <code>\$objArvoreAcaoItemAPI-&gt;setId('MD_ABC_AUDIENCIA_' . \$dblIdDocumento);</code>
IdPai	Identificador do nó pai na árvore. O SEI monta os nós de processo e documentos usando o identificador do protocolo como valor para o campo "Id", ex.:  <code>\$objArvoreAcaoItemAPI-&gt;setIdPai(\$dblIdDocumento);</code>
Href	Pode ser um link ou um código javascript, ex.:  <code>\$objArvoreAcaoItemAPI-&gt;setHref('...link...');</code> <code>\$objArvoreAcaoItemAPI-&gt;setHref('javascript:alert(\'Ícone Processo ABC\');');</code>
Target	Indica qual o destino para execução do link em href: _blank = nova janela ifrVisualizacao = área ao lado da árvore de processo onde é exibido o conteúdo do documento null = se o href é um javascript
Title	Texto que será exibido ao passar o mouse sobre o ícone
Icone	Caminho para a imagem sendo recomendado utilizar o formato PNG com tamanho 16x16, ex.:  <code>\$objArvoreAcaoItemAPI-&gt;setIcone('modulos/abc/exemplo/imagens/abc_pequeno.png');</code>
SinHabilitado	S/N - indica se o ícone será clicável ou não

## AssinaturaAPI

Nome	Nome do assinante
CargoFuncao	Cargo/função utilizado na assinatura
DataHora	Data/hora da assinatura

## AssuntoAPI

CodigoEstruturado	Código do assunto
-------------------	-------------------

Descricao	Descrição do assunto
-----------	----------------------

### AtributoAndamentoAPI

Nome	Nome do atributo, corresponde a variável que será substituída no texto, ex.: @SALA@
Valor	Valor do atributo, valor do atributo, ex.: 302
IdOrigem	Identificador opcional associado com o andamento (campo texto com até 50 posições), ex.: 76232 = identificador interno da sala

### CampoAPI

Nome	Nome do campo do formulário
Valor	Valor do campo do formulário

### Cargo

IdCargo	Identificador interno do cargo
ExpressaoCargo	Descrição do cargo (Ex.: Governador)
ExpressaoTratamento	Tratamento para o cargo (Ex.: A Sua Excelência o Senhor)
ExpressaoVocativo	Vocativo para o cargo (Ex.: Senhor Governador)

### CidadeAPI

IdCidade	Identificador da cidade
IdEstado	Identificador do estado
IdPais	Identificador do país
Nome	Nome da cidade
CodigoIbge	Código IBGE da cidade
SinCapital	S/N - indica se a cidade é capital do estado
Latitude	Latitude da cidade
Longitude	Longitude da cidade

### ContatoAPI

StaOperacao	A - Cadastrar/Alterar E - Excluir D - Desativar R - Reativar
IdContato	Identificador do contato
IdTipoContato	Tipo do contato
NomeTipoContato	Nome do tipo de contato
Sigla	Sigla do contato
Nome	Nome do contato
StaNatureza	F/J - Pessoa Física/Jurídica
IdContatoAssociado	Identificador do contato associado
NomeContatoAssociado	Nome do contato associado
SinEnderecoAssociado	S/N - indica se o contato utiliza ou não o endereço do contato associado

EnderecoAssociado	Dados de endereçamento do contato associado
ComplementoAssociado	
BairroAssociado	
IdCidadeAssociado	
NomeCidadeAssociado	
IdEstadoAssociado	
SiglaEstadoAssociado	
IdPaisAssociado	
NomePaisAssociado	
CepAssociado	
Endereco	Dados de endereçamento do contato
Complemento	
Bairro	
IdCidade	
NomeCidade	
IdEstado	
SiglaEstado	
IdPais	
NomePais	
Cep	
StaGenero	M/F - Masculino/Feminino
IdCargo	Identificador interno do cargo
ExpressaoCargo	Descrição do cargo
ExpressaoTratamento	Descrição do tratamento
ExpressaoVocativo	Descrição do vocativo
Cpf	Número do CPF sem formatação
Cnpj	Número do CNPJ sem formatação
Rg	Número do RG
OrgaoExpedidor	Órgão expedidor
Matricula	Número de matrícula
MatriculaOab	Matrícula OAB
TelefoneFixo	Telefone fixo
TelefoneCelular	Telefone celular
DataNascimento	Data de nascimento
Email	E-mail
SitioInternet	Sítio na internet
Observacao	Texto de observação associado
SinAtivo	S/N - Sinalizador indica se o registro está ativo

### DefinicaoMarcadorAPI

IdMarcador	Identificador do marcador
IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
Texto	Texto associado com a definição

### DestinatarioAPI

Sigla	Sigla do destinatário
-------	-----------------------

Nome	Nome do destinatário
------	----------------------

### DocumentoAPI

IdDocumento	Identificador do documento
Tipo	Tipo interno do documento: G - Gerado R - Recebido
SubTipo	Subtipos do tipo G (Gerado): E - Editor eDoc (descontinuado) I - Editor Interno HTML A - Formulário automático XML (email, ouvidoria) F - Formulário gerado  Subtipo do tipo R (Recebido): X - Externo
IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
IdSerie	Identificador do tipo do documento visível para o usuário
NomeSerie	Nome do tipo do documento visível para o usuário
Numero	Número do documento, ex: Portaria 123
Data	Data do documento
Descricao	Descrição associada com o documento
IdTipoConferencia	Identificador do tipo de conferência
Remetente	Instância de RemetenteAPI preenchida com Sigla e/ou Nome
Interessados	Lista de instâncias de InteressadoAPI preenchidas com Sigla e/ou Nome
Destinatarios	Lista de instâncias de DestinatarioAPI preenchidas com Sigla e/ou Nome
Observacao	Texto da observação
NomeArquivo	Nome do arquivo associado com o documento Tipo = R
NivelAcesso	0 - Público 1 - Restrito 2 - Sigiloso
IdHipoteseLegal	Identificador da hipótese legal associada
Conteudo	Conteúdo do documento externo codificado em Base64
ConteudoMTOM	Conteúdo do documento externo em formato binário
SinBloqueado	S/N - se o documento estiver bloqueado então seu conteúdo não poderá ser alterado
IdArquivo	Identificador do arquivo no repositório (ver método adicionar Arquivo)
Campos	Lista de instâncias de CampoAPI
IdUnidadeGeradora	Identificador da unidade geradora do documento
NumeroProtocolo	Número SEI do documento
SinAssinado	S/N - indica se o documento está assinado
CodigoAcesso	Valor numérico utilizado pelo sistema para indicar se o usuário tem acesso a determinado protocolo. O usuário tem acesso se este valor for MAIOR que zero (zero ou negativo indica falta de acesso).
IdOrgaoUnidadeGeradora	Identificador do órgão associado com a unidade geradora

### **EntradaAdicionarArquivoAPI**

Nome	Nome do arquivo
Tamanho	Tamanho em bytes do arquivo
Hash	MD5 do conteúdo total
Conteudo	Conteúdo total ou parcial do arquivo

### **EntradaAdicionarConteudoArquivoAPI**

IdArquivo	Identificador do arquivo no repositório
Conteudo	Conteúdo parcial do arquivo

### **EntradaAnexarProcessoAPI**

IdProcedimentoPrincipal	Identificador do processo que conterá o anexo
ProtocoloProcedimentoPrincipal	Número do processo que conterá o anexo visível para o usuário, ex: 12.1.000000077-4
IdProcedimentoAnexado	Identificador do processo que será anexado
ProtocoloProcedimentoAnexado	Número do processo que será anexado visível para o usuário, ex: 12.1.000001213-6

### **EntradaAtribuirProcessoAPI**

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
IdUsuario	Identificador do usuário
SinReabrir	S/N - indica se o processo deve ser reaberto caso esteja concluído

### **EntradaBloquearProcessoAPI**

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4

### **EntradaCancelarDisponibilizacaoBlocoAPI**

IdBloco	Identificador do bloco
---------	------------------------

### **EntradaCancelarDocumentoAPI**

IdDocumento	Identificador do documento
ProtocoloDocumento	Número SEI do documento
Motivo	Texto do motivo para cancelamento

### **EntradaConcluirProcessoAPI**

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4

### **EntradaConsultarBlocoAPI**

IdBloco	Identificador do bloco
---------	------------------------

SinRetornarProtocolos	S/N - indica se além dos dados do bloco também devem ser retornados os protocolos que ele contém
-----------------------	--

### EntradaConsultarDocumentoAPI

IdDocumento	Identificador do documento
ProtocoloDocumento	Número SEI do documento
SinRetornarAndamentoGeracao	S/N - indica se deve ser retornado o andamento de geração do documento
SinRetornarAssinaturas	S/N - indica se devem ser retornadas as assinaturas
SinRetornarPublicacao	S/N - indica se devem ser retornadas as informações de publicação
SinRetornarCampos	S/N - indica se devem ser retornados os campos (para formulários)

### EntradaConsultarProcedimentoAPI

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
SinRetornarAssuntos	S/N - indica se devem ser retornados os assuntos associados (conjunto de AssuntoAPI)
SinRetornarInteressados	S/N - indica se devem ser retornados os interessados (conjunto de InteressadoAPI)
SinRetornarObservacoes	S/N - indica se devem ser retornadas as observações (conjunto de ObservacaoAPI)
SinRetornarAndamentoGeracao	S/N - indica se deve ser retornado o andamento de geração (ocorrência de AndamentoAPI)
SinRetornarAndamentoConclusao	S/N - indica se deve ser retornado o andamento de conclusão na unidade (ocorrência de AndamentoAPI)
SinRetornarUltimoAndamento	S/N - indica se deve ser retornado o último andamento na unidade (ocorrência de AndamentoAPI)
SinRetornarUnidadesProcedimentoAberto	S/N - indica se devem ser retornadas as unidades onde o processo está aberto (conjunto de UnidadeProcedimentoAbertoAPI)
SinRetornarProcedimentosRelacionados	S/N - indica se deve ser retornados os processos relacionados (conjunto de ProcedimentoResumidoAPI)
SinRetornarProcedimentosAnexados	S/N - indica se deve ser retornados os processos anexados (conjunto de ProcedimentoResumidoAPI)

### EntradaDesanexarProcessoAPI

IdProcedimentoPrincipal	Identificador do processo que contém o anexo
ProtocoloProcedimentoPrincipal	Número do processo que contém o anexo visível para o usuário, ex: 12.1.000000077-4
IdProcedimentoAnexado	Identificador do processo anexado
ProtocoloProcedimentoAnexado	Número do processo anexado visível para o usuário, ex: 12.1.000001213-6



Motivo	Texto indicando o motivo da desanexação
--------	---

### **EntradaDesbloquearProcessoAPI**

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4

### **EntradaDisponibilizarBlocoAPI**

IdBloco	Identificador do bloco
---------	------------------------

### **EntradaEnviarProcessoAPI**

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
UnidadesDestino	Identificadores das unidades para envio
SinManterAbertoUnidade	S/N - indica se o processo deve ficar aberto na unidade origem
SinRemoverAnotacao	S/N - indica se a anotação deve ser removida do processo após o envio
SinEnviarEmailNotificacao	S/N - indica se deve ser enviado email de notificação para as unidades destino
DataRetornoProgramado	Data para retorno no formato dd/mm/aaaa
DiasRetornoProgramado	Número de dias para retorno
SinDiasUteisRetornoProgramado	S/N - indica se os dias informados são úteis
SinReabrir	S/N - indica se o processo deve ser reaberto automaticamente na unidade de origem antes do envio

### **EntradaExcluirBlocoAPI**

IdBloco	Identificador do bloco
---------	------------------------

### **EntradaGerarBlocoAPI**

Tipo	Tipo do bloco: A – Assinatura R – Reunião I - Interno
Descricao	Descrição do bloco
UnidadesDisponibilizacao	Identificadores internos das unidades para disponibilização. Passar um conjunto vazio caso o bloco não deva ser disponibilizado.
Documentos	Lista de protocolos de documentos (número visível para o usuário, ex.: 0003934)
IdDocumentos	Listas dos identificadores internos dos documentos
SinDisponibilizar	S/N - sinalizador indicando se o bloco deve ser automaticamente disponibilizado

### EntradaGerarProcedimentoAPI

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
Documentos	Informar os documentos que devem ser gerados em conjunto com o processo (conjunto de DocumentoAPI). Se nenhum documento for gerado informar um conjunto vazio.
ProcedimentosRelacionados	Identificadores dos processos que devem ser relacionados automaticamente com o novo processo
UnidadesEnvio	Identificadores das unidades para envio do processo após a geração. O processo ficará aberto na unidade geradora e nas unidades informadas neste parâmetro.
SinManterAbertoUnidade	S/N - sinalizador indica se o processo deve ser mantido aberto na unidade de origem
SinEnviarEmailNotificacao	S/N - sinalizador indicando se deve ser enviado email de aviso para as unidades destinatárias
DataRetornoProgramado	Data para definição de Retorno Programado (opcional)
DiasRetornoProgramado	Número de dias para o Retorno Programado (opcional)
SinDiasUteisRetornoProgramado	S/N - sinalizador indica se o valor passado no parâmetro DiasRetornoProgramado corresponde a dias úteis ou não (opcional)
IdMarcador	Identificador de um marcador da unidade para associação (opcional)
TextoMarcador	Texto do marcador (opcional)

### EntradaIncluirDocumentoBlocoAPI

IdBloco	Identificador do bloco
IdDocumento	Identificador interno do documento
ProtocoloDocumento	Número SEI do documento
Anotacao	Texto de anotação associado com o documento no bloco

### EntradaIncluirProcessoBlocoAPI

IdBloco	Identificador do bloco
IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
Anotacao	Texto de anotação associado com o documento no bloco

### EntradaLancarAndamentoAPI

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
IdTarefa	Identificador da tarefa
Atributos	Atributos associados com o andamento (conjunto de AtributoAndamentoAPI)

### EntradaListarAndamentosAPI

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
SinRetornarAtributos	S/N - indica se devem ser retornados os atributos associados com o andamento
Andamentos	Identificadores dos andamentos para filtro
Tarefas	Identificadores das tarefas para filtro

### EntradaListarAndamentosMarcadoresAPI

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4
Marcadores	Identificadores dos marcadores para filtro

### EntradaListarCargosAPI

IdCargo	Opcional. Identificador do cargo para filtro .
---------	--

### EntradaListarCidadesAPI

IdPais	Identificador do país para filtro
IdEstado	Identificador do estado para filtro

### EntradaListarContatosAPI

IdTipoContato	Filtra o tipo de contato
PaginaRegistros	Opcional. Informa o número máximo de registros que devem ser retornados por página de consulta (1 a 1000 com valor padrão 1).
PaginaAtual	Opcional. Informa o número da página atual (valor padrão 1).
Sigla	Opcional. Filtra contato pela sigla.
Nome	Opcional. Filtra contato pelo nome.
CPF	Opcional. Filtra contato pelo CPF.
CNPJ	Opcional. Filtra contato pelo CNPJ.
Matricula	Opcional. Filtra contato pelo número de matrícula.

### EntradaListarEstadosAPI

IdPais	Identificador do país para filtro
--------	-----------------------------------

### EntradaListarExtensoesPermitidasAPI

IdArquivoExtensao	Identificador interno de uma extensão de arquivo
-------------------	--

### EntradaListarHipotesesLegaisAPI

NivelAcesso	1 - Restrito 2 - Sigiloso
-------------	------------------------------

### EntradaListarUsuariosAPI

IdUsuario	Identificador interno de um usuário para filtro
-----------	---

### EntradaReabrirProcessoAPI

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4

### EntradaRelacionarProcessoAPI

IdProcedimento1	Identificador do processo 1
ProtocoloProcedimento1	Número do processo 1 visível para o usuário, ex: 12.1.000000077-4
IdProcedimento2	Identificador do processo 2
ProtocoloProcedimento2	Número do processo 2 visível para o usuário, ex: 12.1.000003541-1

### EntradaRemoverRelacionamentoProcessoAPI

IdProcedimento1	Identificador do processo 1
ProtocoloProcedimento1	Número do processo 1 visível para o usuário, ex: 12.1.000000077-4
IdProcedimento2	Identificador do processo 2
ProtocoloProcedimento2	Número do processo 2 visível para o usuário, ex: 12.1.000003541-1

### EntradaRemoverSobrestamentoProcessoAPI

IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4

### EntradaRetirarDocumentoBlocoAPI

IdBloco	Identificador do bloco
IdDocumento	Identificador interno do documento
ProtocoloDocumento	Número SEI do documento

### EntradaRetirarProcessoBlocoAPI

IdBloco	Identificador do bloco
IdProcedimento	Identificador do processo
ProtocoloProcedimento	Número do processo visível para o usuário, ex: 12.1.000000077-4

### EntradaSobrestarProcessoAPI

IdProcedimento	Identificador do processo que deve ser sobrestado
ProtocoloProcedimento	Número do processo que deve ser sobrestado visível para o usuário, ex: 12.1.000000077-4
IdProcedimentoVinculado	Identificador do processo para vinculação
ProtocoloProcedimentoVinculado	Número do processo para vinculação visível para o usuário, ex: 12.1.000003541-1
Motivo	Texto indicando o motivo do sobrestamento

### EstadoAPI

IdEstado	Identificador interno do estado
IdPais	Identificador interno do país
Sigla	Sigla do estado
Nome	Nome do estado
CodigoIbge	Código IBGE do estado

### HipoteseLegalAPI

IdHipoteseLegal	Identificador da hipótese legal
Nome	Nome da hipótese legal
BaseLegal	Base legal associada
NivelAcesso	1 - Restrito 2 - Sigiloso

### InteressadoAPI

Sigla	Sigla do interessado
Nome	Nome do interessado

### MarcadorAPI

IdMarcador	Identificador interno do marcador
Nome	Nome do marcador
Icone	Imagem PNG em formato base64
SinAtivo	S/N - indica se o marcador está ativo

### ObservacaoAPI

Descricao	Texto da observação
Unidade	Unidade associada com a observação (ocorrência de UnidadeAPI)

### OrgaoAPI

IdOrgao	Identificador interno do órgão
Sigla	Sigla do órgão
Descricao	Descrição do órgão

### PaisAPI

IdPais	Identificador interno do país
Nome	Nome do país

### ProcedimentoAPI

IdProcedimento	Identificador interno do processo
IdTipoProcedimento	Identificador interno do tipo de processo
NomeTipoProcedimento	Nome do tipo de processo
NumeroProtocolo	Número do processo visível para o usuário, ex: 12.1.000003541-1
DataAutuacao	Data de autuação do processo

Especificacao	Texto da especificação associada
Assuntos	Assuntos do processo (ocorrências de AssuntoAPI)
Interessados	Interessados do processo (ocorrências de InteressadoAPI)
Observacao	Texto da observação da unidade
NivelAcesso	0 - Público 1 - Restrito 2 - Sigiloso
IdHipoteseLegal	Identificador interno da hipótese legal associada
CodigoAcesso	Valor numérico utilizado pelo sistema para indicar se o usuário tem acesso a determinado protocolo. O usuário tem acesso se este valor for MAIOR que zero (zero ou negativo indica falta de acesso)
SinAberto	S/N - indica se o processo está aberto na unidade
IdUnidadeGeradora	Identificador interno da unidade geradora
IdOrgaoUnidadeGeradora	Identificador interno do órgão da unidade geradora
GrauSigilo	U - Ultrassegredo S - Secreto R - Reservado

#### **ProcedimentoResumidoAPI**

IdProcedimento	Identificador do processo que deve ser sobrestado
ProtocoloProcedimento	Número do processo que deve ser sobrestado visível para o usuário, ex: 12.1.000000077-4
TipoProcedimento	Tipo do processo (ocorrência de TipoProcedimentoAPI)

#### **ProtocoloBlocoAPI**

ProtocoloFormatado	Número do processo ou documento visível para o usuário
Identificacao	Nome do tipo de processo ou do tipo documento (seguido da numeração associada se existir), ex.: Compra de Material e Contratação de Serviços , Portaria 35
Assinaturas	Assinaturas associadas com os documentos (conjunto de AssinaturaAPI)

#### **PublicacaoAPI**

NomeVeiculo	Nome do veículo cadastrado no SEI
Numero	Número da publicação
DataDisponibilizacao	Data da disponibilização
DataPublicacao	Data da publicação
Estado	A = Agendado P = Publicado
ImprensaNacional	Dados da Imprensa Nacional associados (ocorrência de PublicacaoImprensaNacionalAPI)

#### **PublicacaoImprensaNacionalAPI**

SiglaVeiculo	Sigla do veículo (ex.: DOU)
DescricaoVeiculo	Descrição do veículo (ex.: Diário Oficial da União)
Pagina	Página da publicação

Secao	Seção da publicação
Data	Data da publicação

### RemetenteAPI

Sigla	Sigla do interessado
Nome	Nome do interessado

### SaidaConsultarBlocoAPI

IdBloco	Identificador do bloco
Descricao	Descrição associada com o bloco
Tipo	A=Assinatura R=Reunião I=Interno
Estado	A=Aberto D=Disponibilizado R=Retornado C=Concluído
Unidade	Unidade que gerou o bloco (ocorrência de UnidadeAPI)
Usuario	Usuário que gerou o bloco (ocorrência de UsuarioAPI)
UnidadesDisponibilizacao	Unidades configuradas para disponibilização (conjunto de UnidadeAPI)
Protocolos	Protocolos do bloco (conjunto de ProtocoloBlocoAPI)

### SaidaConsultarDocumentoAPI

IdProcedimento	Id interno do processo no SEI, ex.: 1210000000774
ProcedimentoFormatado	Número do processo visível para o usuário, ex: 12.1.000000077-4
IdDocumento	Id interno do documento no SEI, ex.: 1140000000872
DocumentoFormatado	Número do documento visível para o usuário, ex.: 0003934
LinkAcesso	Link para acesso ao documento
Serie	Dados do tipo do documento (ocorrência de SerieAPI)
Numero	Número do documento
Data	Data de geração para documentos internos e para documentos externos é a data informada na tela de cadastro
UnidadeElaboradora	Dados da unidade que gerou o documento (ocorrência de UnidadeAPI)
AndamentoGeracao	Informações do andamento de geração (ocorrência de AndamentoAPI)
Assinaturas	Assinaturas do documento (conjunto de AssinaturaAPI)
Publicacao	Informações de publicação do documento (conjunto de PublicacaoAPI)
Campos	Campos do formulário (conjunto de CampoAPI)

### SaidaConsultarProcedimentoAPI

IdProcedimento	Id interno do processo no SEI, ex.: 1210000000774
ProcedimentoFormatado	Número do processo visível para o usuário, ex: 12.1.000000077-4
Especificacao	Especificação do processo

DataAutuacao	Data de autuação do processo
LinkAcesso	Link para acesso ao processo
TipoProcedimento	Dados do tipo do processo (ocorrência de TipoProcedimentoAPI)
AndamentoGeracao	Dados do andamento de geração (ocorrência de AndamentoAPI)
AndamentoConclusao	Dados do andamento de conclusão (ocorrência de AndamentoAPI)
UltimoAndamento	Dados do último andamento (ocorrência de AndamentoAPI)
UnidadesProcedimentoAberto	Conjunto de unidades onde o processo se encontra aberto (conjunto de UnidadeProcedimentoAbertoAPI)
Assuntos	Conjunto de assuntos do processo (conjunto de AssuntoAPI)
Interessados	Conjunto de interessados do processo (conjunto de InteressadoAPI)
Observacoes	Conjunto de observações das unidades (conjunto de ObservacaoAPI)
ProcedimentosRelacionados	Conjunto de processos relacionados (conjunto de ProcedimentoResumidoAPI)
ProcedimentosAnexados	Conjunto processos anexados (conjunto de ProcedimentoResumidoAPI)

### SaidaGerarProcedimentoAPI

IdProcedimento	Id interno do processo no SEI, ex.: 1210000000774
ProcedimentoFormatado	Número do processo visível para o usuário, ex: 12.1.000000077-4
LinkAcesso	Link para acesso ao processo
RetornoInclusaoDocumentos	Conjunto de ocorrências de RetornoInclusaoDocumentoAPI (com um item para cada documento informado na geração do processo)

### SaidaIncluirDocumentoAPI

IdDocumento	Id interno do documento no SEI, ex.: 1140000000872
DocumentoFormatado	Número do documento visível para o usuário, ex.: 0003934
LinkAcesso	Link para acesso ao documento

### SerieAPI

IdSerie	Identificador do tipo de documento
Nome	Nome do tipo de documento
Aplicabilidade	T = Documentos internos e externos I = documentos internos E = documentos externos F = formulários

### TipoConferenciaAPI

IdTipoConferencia	Identificador do tipo de conferência
Descricao	Descrição do tipo de conferência

### TipoProcedimentoAPI

IdTipoProcedimento	Identificador do tipo de processo
--------------------	-----------------------------------



Nome	Nome do tipo de processo
------	--------------------------

### UnidadeAPI

IdUnidade	Identificador da unidade
Sigla	Sigla da unidade
Descricao	Descrição da unidade
Orgao	Órgão associado com a unidade (ocorrência de OrgaoAPI)

### UnidadeProcedimentoAbertoAPI

Unidade	Dados da Unidade onde o processo está aberto (ocorrência de UnidadeAPI).
UsuarioAtribuicao	Dados do Usuário para o qual o processo está atribuído (ocorrência de UsuarioAPI)

### UsuarioAPI

IdUsuario	Identificador do usuário
Sigla	Sigla do usuário
Nome	Nome do usuário

## 5. Eventos


A interceptação de eventos ocorre na classe de integração do módulo através da sobrecarga de métodos da classe `SeiIntegracao`. Dependendo do evento interceptado o módulo pode realizar processamentos adicionais ou cancelar o processamento do sistema lançando validações ou exceções.

Nos eventos que recebem múltiplos objetos como parâmetros deve ser evitada a consulta individual de registros. Por exemplo, caso o Controle de Processos esteja com a paginação configurada em 100 isso significa que poderão ser exibidos até 200 processos (100 na coluna de gerados e 100 nos recebidos). Assim o evento `montarIconeControleProcessos` poderá receber um array com até 200 objetos `ProcedimentoAPI`. Se for realizada uma consulta para cada processo serão 200 acessos ao banco. Neste caso é melhor fazer uma única consulta utilizando "IN".

Exemplos para alguns dos métodos descritos nesta seção podem ser visualizados no código do módulo de exemplo ABC disponibilizado junto com o sistema.

### alterarIconeArvoreDocumento

Entrada	
<code>objProcedimentoAPI</code>	Ocorrência de <code>ProcedimentoAPI</code> preenchida com: <code>IdProcedimento</code> , <code>NumeroProtocolo</code> , <code>IdTipoProcedimento</code> , <code>NomeTipoProcedimento</code> , <code>NivelAcesso</code> , <code>GrauSigilo</code> , <code>IdHipoteseLegal</code> , <code>IdUnidadeGeradora</code> , <code>IdOrgaoUnidadeGeradora</code> , <code>CodigoAcesso</code> e <code>SinAberto</code>
<code>\$arrObjDocumentoAPI</code>	Conjunto de ocorrências de <code>DocumentoAPI</code> preenchidas com: <code>IdDocumento</code> , <code>NumeroProtocolo</code> , <code>IdSerie</code> , <code>NomeSerie</code> , <code>IdUnidadeGeradora</code> , <code>IdOrgaoUnidadeGeradora</code> , <code>SinAssinado</code> ,

	SinPublicado, SinBloqueado, CodigoAcesso, Tipo, SubTipo
Saída	
arrIcones	Um array PHP indexado pelos IDs dos documentos onde cada posição contém o caminho para a imagem. Documentos que não devem ter o ícone alterado pelo módulo não precisam estar no array. Para as imagens é recomendado utilizar o formato PNG com tamanho 16x16.
Exemplo	
<pre> public function alterarIconeArvoreDocumento(ProcedimentoAPI \$objProcedimentoAPI, \$arrObjDocumentoAPI){     \$arrIcones = array();      \$objInfraParametro = new InfraParametro(BancoSEI::getInstance());     \$numIdSerieAbc = \$objInfraParametro-&gt;getValor('MD_ABC_ID_SERIE_TESTE', false);      foreach (\$arrObjDocumentoAPI as \$objDocumentoAPI) {         if (\$objDocumentoAPI-&gt;getIdSerie()==\$numIdSerieAbc) {             \$arrIcones[\$objDocumentoAPI-&gt;getIdDocumento()] =             'modulos/abc/exemplo/imagens/abc_pequeno.png';         }     }      return \$arrIcones; } </pre>	
	

## anexarProcesso

Entrada	
\$objProcedimentoAPIPrincipal	Instância preenchida com IdProcedimento e NumeroProtocolo.
\$objProcedimentoAPIAnexado	Instância preenchida com IdProcedimento e NumeroProtocolo.

## assinarDocumento

Entrada	
\$arrObjDocumentoAPI	Instâncias preenchidas com IdDocumento, NumeroProtocolo, IdSerie, IdUnidadeGeradora, Tipo, SubTipo e NivelAcesso.

## atualizarConteudoDocumento

Entrada	
\$objDocumentoAPI	Instância preenchida com IdDocumento

**bloquearProcesso**

Entrada	
\$arrObjProcedimentoAPI	Instâncias preenchidas com IdProcedimento e NumeroProtocolo.

**cancelarDocumento**

Entrada	
\$objDocumentoAPI	Instância preenchida com IdDocumento, NumeroProtocolo, IdSerie, IdUnidadeGeradora, Tipo, SubTipo e NivelAcesso.

**concluirProcesso**

Entrada	
\$arrObjProcedimentoAPI	Instâncias preenchidas com IdProcedimento.

**desanexarProcesso**

Entrada	
\$objProcedimentoAPIPrincipal	Instância preenchida com IdProcedimento e NumeroProtocolo.
\$objProcedimentoAPIAnexo	Instância preenchida com IdProcedimento e NumeroProtocolo.

**desbloquearProcesso**

Entrada	
\$arrObjProcedimentoAPI	Instâncias preenchidas com IdProcedimento e NumeroProtocolo.

**enviarProcesso**

Entrada	
\$arrObjProcedimentoAPI	Instâncias preenchida com IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento e IdUnidadeGeradora.
\$arrObjUnidadeAPI	Instâncias preenchidas com IdUnidade, Sigla, Descricao e OrgaoAPI (IdOrgao, Sigla, Descricao).

**excluirDocumento**

Entrada	
\$objDocumentoAPI	Instância preenchida com IdDocumento.

**excluirProcesso**

Entrada	
\$objProcedimentoAPI	Instância preenchida com IdProcedimento.

### **excluirUnidade**

Entrada	
\$arrObjUnidadeAPI	Instâncias preenchidas com IdUnidade, Sigla e Descricao.

### **excluirUsuario**

Entrada	
\$arrObjUsuarioAPI	Instâncias preenchidas com IdUsuario, Sigla e Nome.

### **gerarDocumento**

Entrada	
\$objDocumentoAPI	Instância preenchida com IdDocumento, NumeroProtocolo, IdProcedimento, IdSerie, NivelAcesso e SubTipo.

### **gerarProcesso**

Entrada	
\$objProcedimentoAPI	Instância preenchida com IdProcedimento, NumeroProtocolo, IdTipoProcedimento e NivelAcesso.

### **inicializar**

Entrada	
strVersaoSEI	Número da versão do SEI (ex.: 3.0.0)
Observações	
Este método é chamado a cada acesso ao sistema e pode ser utilizado, por exemplo, para verificar a compatibilidade do módulo com a versão do SEI. Ele é executado em cada acesso ao sistema então se for necessário realizar um processamento complexo e demorado é recomendado o uso das classes SessaoSEI e CacheSEI para salvar o resultado na sessão do usuário ou no servidor memcache (garantindo que o processamento seja realizado apenas uma vez).	

### **montarAcaoControleAcessoExterno**

Entrada	
arrAcessoExternoAPI	Conjunto de ocorrências de AcessoExternoAPI preenchidas com: IdAcessoExterno, DataValidade, SinAcessoProcesso, Procedimento (ocorrência de ProcedimentoAPI preenchida com IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento e NivelAcesso), Documento (ocorrência de DocumentoAPI preenchida com IdDocumento, NumeroProtocolo, IdSerie, IdUnidadeGeradora, Tipo, SinAssinado, SinPublicado e NivelAcesso).
Saída	
arrIcones	Um array PHP indexado pelos IDs dos acessos externos onde cada posição contém outro array com o conjunto de ícones para exibição. Acessos externos que não devem exibir ícones do módulo não precisam estar no array. Para as imagens é recomendado utilizar o formato PNG com tamanho 16x16.
Exemplo	

```

public function montarAcaoControleAcessoExterno($arrObjAcessoExternoAPI){

    $arrIcones = array();

    foreach($arrObjAcessoExternoAPI as $objAcessoExternoAPI) {
        $arrIcones[$objAcessoExternoAPI->getIdAcessoExterno()][] = '<a
href="javascript:void(0);" '.PaginaSEI::montarTitleTooltip('Ícone Ação Acesso Externo
ABC', 'Módulo ABC').'></a>';
    }

    return $arrIcones;
}

```

TRIBUNAL REGIONAL FEDERAL DA 4ª REGIÃO

sei!

Teste

Menu



Controle de Acessos Externos

Alterar Senha

## Controle de Acessos Externos

Lista de Acessos Externos (39 registros)

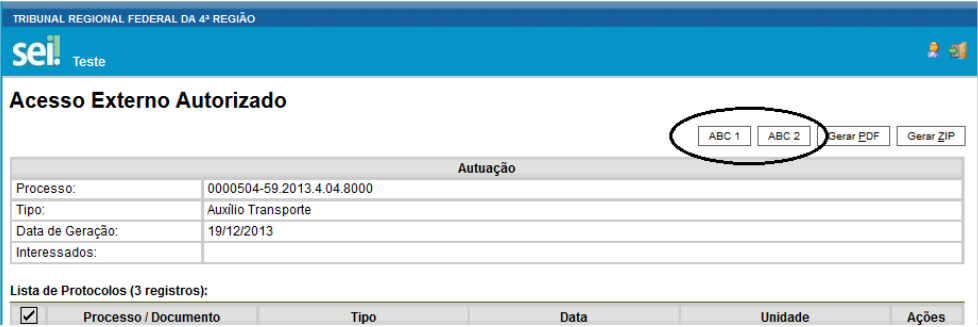
Processo	Documento	Tipo	Liberação	Validade	Ações
<a href="#">0000504-59.2013.4.04.8000</a>	<a href="#">0011703</a>	Ata	04/08/2016		  
<a href="#">0000504-59.2013.4.04.8000</a>	<a href="#">0011702</a>	Portaria	04/08/2016		  
<a href="#">0000504-59.2013.4.04.8000</a>	<a href="#">0027515</a>	Ato	04/08/2016		  
<a href="#">0000504-59.2013.4.04.8000</a>	<a href="#">0027472</a>	Ata	04/08/2016		  
<a href="#">0000504-59.2013.4.04.8000</a>	<a href="#">0011704</a>	Portaria	04/08/2016		  

## montarAcaoDocumentoAcessoExternoAutorizado


Entrada	
arrDocumentoAPI	Conjunto de ocorrências de DocumentoAPI preenchidas com: IdDocumento, NumeroProtocolo, IdSerie, IdUnidadeGeradora, Tipo, SinAssinado, SinPublicado e NivelAcesso.
Saída	
arrIcones	Um array PHP indexado pelos IDs dos documentos onde cada posição contém outro array com o conjunto de ícones para exibição. Documentos que não devem exibir ícones do módulo não precisam estar no array. Para as imagens é recomendado utilizar o formato PNG com tamanho 16x16.
Exemplo	
<pre> public function montarAcaoDocumentoAcessoExternoAutorizado(\$arrObjDocumentoAPI){     \$arrIcones = array();     foreach(\$arrObjDocumentoAPI as \$objDocumentoAPI) {         \$arrIcones[\$objDocumentoAPI-&gt;getIdDocumento()][] = '&lt;a href="javascript:void(0);" '.PaginaSEI::montarTitleTooltip('Ícone Ação Documento Acesso Externo Autorizado ABC', 'Módulo ABC').'&gt;&lt;img src="modulos/abc/exemplo/imagens/abc_pequeno.png" class="imagemStatus" /&gt;&lt;/a&gt;';     }     return \$arrIcones; } </pre>	



## montarBotaoAcessoExternoAutorizado

Entrada	
objProcedimentoAPI	Ocorrência de ProcedimentoAPI preenchida com: IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento e NivelAcesso.
Saída	
arrBotoes	Um array PHP onde cada item representa um botão.
Exemplo	
<pre> public function montarBotaoAcessoExternoAutorizado(ProcedimentoAPI \$objProcedimentoAPI){     \$arrBotoes = array();     \$arrBotoes[] = '&lt;button type="button" id="btnExemploABC1" name="btnExemploABC1" value="ABC 1" class="infraButton"&gt;ABC 1&lt;/button&gt;';     \$arrBotoes[] = '&lt;button type="button" id="btnExemploABC2" name="btnExemploABC2" value="ABC 2" class="infraButton"&gt;ABC 2&lt;/button&gt;';     return \$arrBotoes; } </pre>	
	

## montarBotaoControleAcessoExterno

Saída	
arrBotoes	Um array PHP onde cada item representa um botão.
Exemplo	
<pre> public function montarBotaoControleAcessoExterno(){     \$arrBotoes = array();     \$arrBotoes[] = '&lt;button type="button" id="btnExemploABC1" name="btnExemploABC1" value="ABC 1" class="infraButton"&gt;ABC 1&lt;/button&gt;';     \$arrBotoes[] = '&lt;button type="button" id="btnExemploABC2" name="btnExemploABC2" value="ABC 2" class="infraButton"&gt;ABC 2&lt;/button&gt;';     return \$arrBotoes; } </pre>	
	

## montarBotaoControleProcessos

Saída	
arrBotoes	Um array PHP onde cada item representa um botão. Para as imagens é recomendado utilizar o formato PNG com tamanho 40x40.
Exemplo	
<pre> public function montarBotaoControleProcessos(){      \$arrBotoes = array();      /*     Função javascript disponível na página de Controle de Processos:     acaoControleProcessos(link, requerSelecionado, aceitaSigiloso)      onde:     link - link para a página     requerSelecionado - true/false, indica necessidade ou não de selecionar processos     para executar a acao     aceitaSigiloso - true/false, indica se o usuario podera selecionar processos     sigilosos     */      if (SessaoSEI::getInstance()-&gt;verificarPermissao('md_abc_andamento_lancar')) {          \$arrBotoes[] = '&lt;a href="#" onclick="return acaoControleProcessos(\''.         SessaoSEI::getInstance()-         &gt;assinarLink('controlador.php?acao=md_abc_andamento_lancar&amp;acao_origem=procedimento_cont         rolar&amp;acao_retorno=procedimento_controlar') . '\', true, false);" tabindex="" .         PaginaSEI::getInstance()-&gt;getProxTabBarraComandosSuperior() . '" class="botaoSEI"&gt;&lt;img         class="infraCorBarraSistema" src="modulos/abc/exemplo/imagens/abc_grande.png" alt="Botão         Controle de Processos ABC" title="Botão Controle de Processos ABC" /&gt;&lt;/a&gt;';      }      return \$arrBotoes; } </pre>	
<p><b>Controle de Processos</b></p> 	

## montarBotaoDocumento

Entrada	
objProcedimentoAPI	Ocorrência de ProcedimentoAPI preenchida com: IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento, NivelAcesso, GrauSigilo, IdHipoteseLegal, IdUnidadeGeradora, IdOrgaoUnidadeGeradora, CodigoAcesso e SinAberto
\$arrObjDocumentoAPI	Conjunto de ocorrências de DocumentoAPI preenchidas com: IdDocumento, NumeroProtocolo, IdSerie, NomeSerie, IdUnidadeGeradora, IdOrgaoUnidadeGeradora, SinAssinado, SinPublicado, SinBloqueado, CodigoAcesso, Tipo, SubTipo



Saída	
arrBotoes	Um array PHP indexado pelos IDs dos documentos onde cada posição contém outro array com o conjunto de botões para exibição junto ao documento. Documentos que não devem exibir botões do módulo não precisam estar no array. Para as imagens é recomendado utilizar o formato PNG com tamanho 40x40.
Exemplo	
<pre> public function montarBotaoDocumento(ProcedimentoAPI \$objProcedimentoAPI, \$arrObjDocumentoAPI) {      \$arrBotoes = array();      if (\$objProcedimentoAPI-&gt;getCodigoAcesso() &gt; 0 &amp;&amp; \$objProcedimentoAPI-&gt;getSinAberto()=='S'){          \$dblIdProcedimento = \$objProcedimentoAPI-&gt;getIdProcedimento();          \$bolAcaoAbcDocumentoProcessar = SessaoSEI::getInstance()-&gt;verificarPermissao('md_abc_documento_processar');         //\$bolAcaoAbcDocumentoProcessar2 = SessaoSEI::getInstance()-&gt;verificarPermissao('md_abc_documento_processar2');         //\$bolAcaoAbcDocumentoProcessar3 = SessaoSEI::getInstance()-&gt;verificarPermissao('md_abc_documento_processar3');         //\$bolAcaoAbcDocumentoProcessarN = SessaoSEI::getInstance()-&gt;verificarPermissao('md_abc_documento_processarN');          foreach (\$arrObjDocumentoAPI as \$objDocumentoAPI) {              if (\$objDocumentoAPI-&gt;getCodigoAcesso() &gt; 0) {                  \$dblIdDocumento = \$objDocumentoAPI-&gt;getIdDocumento();                  \$arrBotoes[\$dblIdDocumento] = array();                  if (\$bolAcaoAbcDocumentoProcessar) {                     \$arrBotoes[\$dblIdDocumento][] = '&lt;a href="#" onclick="location.href=\\\'\' . SessaoSEI::getInstance()-&gt;assinarLink('controlador.php?acao=md_abc_documento_processar&amp;id_procedimento=' . \$dblIdProcedimento . '&amp;id_documento=' . \$dblIdDocumento . '&amp;arvore=1') . '\\\'\';" tabindex="\' . PaginaSEI::getInstance()-&gt;getProxTabBarraComandosSuperior() . \' class="botaoSEI"&gt;&lt;img class="infraCorBarraSistema" src="modulos/abc/exemplo/imagens/abc_grande.png" alt="Botão Documento ABC" title="Botão Documento ABC" /&gt;&lt;/a&gt;';                 }                  /*                 if (\$bolAcaoAbcDocumentoProcessar2) {                     \$arrBotoes[\$dblIdDocumento][] = '...';                 }                  if (\$bolAcaoAbcDocumentoProcessar3) {                     \$arrBotoes[\$dblIdDocumento][] = '...';                 }                  if (\$bolAcaoAbcDocumentoProcessarN) {                     \$arrBotoes[\$dblIdDocumento][] = '...';                 }                 */             }         }          return \$arrBotoes;     } } </pre>	



## montarBotaoProcesso

Entrada	
objProcedimentoAPI	Ocorrência de ProcedimentoAPI preenchida com: IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento, NivelAcesso, GrauSigilo, IdHipoteseLegal, IdUnidadeGeradora, IdOrgaoUnidadeGeradora, CodigoAcesso e SinAberto
Saída	
arrBotoes	Um array PHP onde cada item representa um botão. Para as imagens é recomendado utilizar o formato PNG com tamanho 40x40.

## Exemplo

```
public function montarBotaoProcesso(ProcedimentoAPI $objProcedimentoAPI){

    $arrBotoes = array();

    if (SessaoSEI::getInstance()->verificarPermissao('md_abc_processo_processar') &&
        $objProcedimentoAPI->getSinAberto()=='S' && $objProcedimentoAPI->getCodigoAcesso() > 0)
    {




        $arrBotoes[] = '<a href="#" onclick="location.href=\\\\"' .
            SessaoSEI::getInstance()-
            >assinarLink('controlador.php?acao=md_abc_processo_processar&id_procedimento=' .
            $objProcedimentoAPI->getIdProcedimento() . '&arvore=1') . '\\\\";" tabindex="' .
            PaginaSEI::getInstance()->getProxTabBarraComandosSuperior() . '" class="botaoSEI"></a>';

    }

    return $arrBotoes;
}
```



## montarIconeAcompanhamentoEspecial

Entrada																
arrObjProcedimentoAPI	Conjunto de ocorrências de ProcedimentoAPI preenchidas com: IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento, NivelAcesso, GrauSigilo, IdHipoteseLegal, IdUnidadeGeradora e IdOrgaoUnidadeGeradora															
Saída																
arrIcones	Um array PHP indexado pelos IDs dos processos onde cada posição contém outro array com o conjunto de ícones para exibição junto ao processo. Processos que não devem exibir ícones do módulo não precisam estar no array. Para as imagens é recomendado utilizar o formato PNG com tamanho 16x16.															
Exemplo																
<pre>public function montarIconeAcompanhamentoEspecial(\$arrObjProcedimentoAPI){      \$arrIcones = array();      foreach(\$arrObjProcedimentoAPI as \$objProcedimentoAPI) {          \$arrIcones[\$objProcedimentoAPI-&gt;getIdProcedimento()]] = '&lt;a href="javascript:void(0);" '.PaginaSEI::montarTitleTooltip('Ícone Acompanhamento Especial ABC', 'Módulo ABC').'&gt;&lt;img src="modulos/abc/exemplo/imagens/abc_pequeno.png" class="imagemStatus" /&gt;&lt;/a&gt;';      }      return \$arrIcones; }</pre>																
<h3>Acompanhamento Especial</h3> <p>Grupo:</p> <div><div>Todos</div><div></div></div> <table><tr><th><input checked="" type="checkbox"/></th><th>Processo</th><th>Usuário</th><th>Data</th><th>Grupo</th></tr><tr><td><input type="checkbox"/></td><td> 0000277-98.2015.4.04.8000</td><td>mga</td><td>13/05/2015 16:44:42</td><td></td></tr><tr><td><input type="checkbox"/></td><td>0000063-10.2015.4.04.8000</td><td>mga</td><td>24/03/2015 12:05:16</td><td></td></tr></table>		<input checked="" type="checkbox"/>	Processo	Usuário	Data	Grupo	<input type="checkbox"/>	 0000277-98.2015.4.04.8000	mga	13/05/2015 16:44:42		<input type="checkbox"/>	0000063-10.2015.4.04.8000	mga	24/03/2015 12:05:16	
<input checked="" type="checkbox"/>	Processo	Usuário	Data	Grupo												
<input type="checkbox"/>	 0000277-98.2015.4.04.8000	mga	13/05/2015 16:44:42													
<input type="checkbox"/>	0000063-10.2015.4.04.8000	mga	24/03/2015 12:05:16													

## montarIconeControleProcessos

Entrada	
arrObjProcedimentoAPI	Conjunto de ocorrências de ProcedimentoAPI preenchidas com: IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento, NivelAcesso, GrauSigilo, IdHipoteseLegal, IdUnidadeGeradora e IdOrgaoUnidadeGeradora
Saída	
arrIcones	Um array PHP indexado pelos IDs dos processos onde cada posição contém outro array com o conjunto de ícones para exibição junto ao processo. Processos que não devem exibir ícones do módulo não precisam estar no array. Para as imagens é

	recomendado utilizar o formato PNG com tamanho 16x16.										
Exemplo											
<pre> public function montarIconeControleProcessos(\$arrObjProcedimentoAPI){      \$arrIcones = array();      foreach(\$arrObjProcedimentoAPI as \$objProcedimentoAPI) {         \$arrIcones[\$objProcedimentoAPI-&gt;getIdProcedimento()][] = '&lt;a href="javascript:void(0);" '.PaginaSEI::montarTitleTooltip('Ícone Controle de Processos ABC','Módulo ABC').'&gt;&lt;img src="modulos/abc/exemplo/imagens/abc_pequeno.png" class="imagemStatus" /&gt;&lt;/a&gt;';     }      return \$arrIcones; } </pre>											
<b>Controle de Processos</b>  <div> Ver processos atribuídos a mim Ver por marcadores </div> <div> 1 ▶ ▶▶ </div> <div> 216 registros - 1 a 100: </div> <table border="1"> <thead> <tr> <th></th><th>Recebidos</th></tr> </thead> <tbody> <tr> <td><input checked="" type="checkbox"/></td><td></td></tr> <tr> <td><input type="checkbox"/></td><td>0000438-74.2016.4.04.8000</td></tr> <tr> <td><input type="checkbox"/></td><td>12.1.000002378-2</td></tr> <tr> <td><input type="checkbox"/></td><td>10.1.000002579-0</td></tr> </tbody> </table>			Recebidos	<input checked="" type="checkbox"/>		<input type="checkbox"/>	0000438-74.2016.4.04.8000	<input type="checkbox"/>	12.1.000002378-2	<input type="checkbox"/>	10.1.000002579-0
	Recebidos										
<input checked="" type="checkbox"/>											
<input type="checkbox"/>	0000438-74.2016.4.04.8000										
<input type="checkbox"/>	12.1.000002378-2										
<input type="checkbox"/>	10.1.000002579-0										

## montarIconeDocumento

Entrada	
objProcedimentoAPI	Ocorrência de ProcedimentoAPI preenchida com: IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento, NivelAcesso, GrauSigilo, IdHipoteseLegal, IdUnidadeGeradora, IdOrgaoUnidadeGeradora, CodigoAcesso e SinAberto
\$arrObjDocumentoAPI	Conjunto de ocorrências de DocumentoAPI preenchidas com: IdDocumento, NumeroProtocolo, IdSerie, NomeSerie, IdUnidadeGeradora, IdOrgaoUnidadeGeradora, SinAssinado, SinPublicado, SinBloqueado, CodigoAcesso, Tipo, SubTipo
Saída	
arrIcones	Um array PHP indexado pelos IDs dos documentos onde cada posição contém um array de ocorrências de ArvoreAcaoItemAPI com os ícones para exibição junto ao documento. Documentos que não devem exibir ícones do módulo não precisam estar no array.
Exemplo	
<pre> public function montarIconeDocumento(ProcedimentoAPI \$objProcedimentoAPI, \$arrObjDocumentoAPI){ </pre>	

```

    $arrIcones = array();

    if ($objProcedimentoAPI->getCodigoAcesso() > 0 && $objProcedimentoAPI->getSinAberto()=='S') {

        $bolAcaoAbcDocumentoProcessar = SessaoSEI::getInstance()->verificarPermissao('md_abc_documento_processar');

        foreach ($arrObjDocumentoAPI as $objDocumentoAPI) {

            if ($objDocumentoAPI->getCodigoAcesso() > 0) {

                $dblIdDocumento = $objDocumentoAPI->getIdDocumento();

                $arrIcones[$dblIdDocumento] = array();

                if ($bolAcaoAbcDocumentoProcessar) {
                    $objArvoreAcaoItemAPI = new ArvoreAcaoItemAPI();
                    $objArvoreAcaoItemAPI->setTipo('MD_ABC_DOCUMENTOS');
                    $objArvoreAcaoItemAPI->setId('MD_ABC_DOC_' . $dblIdDocumento);
                    $objArvoreAcaoItemAPI->setIdPai($dblIdDocumento);
                    $objArvoreAcaoItemAPI->setTitle('Ícone Documento ABC');
                    $objArvoreAcaoItemAPI->setIcône('modulos/abc/exemplo/imagens/abc_pequeno.png');

                    $objArvoreAcaoItemAPI->setTarget(null);
                    $objArvoreAcaoItemAPI->setHref('javascript:alert(\'Ícone Documento ABC\');');

                    // $objArvoreAcaoItemAPI->setTarget('_blank');
                    // $objArvoreAcaoItemAPI->setTarget('ifrVisualizacao');
                    // $objArvoreAcaoItemAPI->setHref(SessaoSEI::getInstance()->assinarLink('controlador.php?acao=md_abc_documento_processar&id_procedimento=' . $dblIdProcedimento . '&id_documento=' . $dblIdDocumento . '&arvore=1'));

                    $objArvoreAcaoItemAPI->setSinHabilitado('S');


                    $arrIcones[$dblIdDocumento][] = $objArvoreAcaoItemAPI;
                }
            }
        }
    }
    return $arrIcones;
}

```




## montarIcôneProcesso

Entrada	
objProcedimentoAPI	Ocorrência de ProcedimentoAPI preenchida com: IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento, NivelAcesso, GrauSigilo, IdHipoteseLegal, IdUnidadeGeradora, IdOrgaoUnidadeGeradora,

	CodigoAcesso e SinAberto
	Saída
arrObjArvoreAcaoItemAPI	Conjunto de ocorrências da estrutura ArvoreAcaoItemAPI
	Exemplo
<pre> public function montarIconeProcesso(ProcedimentoAPI \$objProcedimentoAPI){     \$arrObjArvoreAcaoItemAPI = array();      if (SessaoSEI::getInstance()-&gt;verificarPermissao('md_abc_processo_processar') &amp;&amp;         \$objProcedimentoAPI-&gt;getCodigoAcesso() &gt; 0 &amp;&amp; \$objProcedimentoAPI-&gt;getSinAberto()=='S')     {         \$dbIdProcedimento = \$objProcedimentoAPI-&gt;getIdProcedimento();          \$objArvoreAcaoItemAPI = new ArvoreAcaoItemAPI();         \$objArvoreAcaoItemAPI-&gt;setTipo('MD_ABC_PROCESSO');         \$objArvoreAcaoItemAPI-&gt;setId('MD_ABC_PROC_' . \$dbIdProcedimento);         \$objArvoreAcaoItemAPI-&gt;setIdPai(\$dbIdProcedimento);         \$objArvoreAcaoItemAPI-&gt;setTitle('Ícone Processo ABC');         \$objArvoreAcaoItemAPI-&gt;setIcone('modulos/abc/exemplo/imagens/abc_pequeno.png');          \$objArvoreAcaoItemAPI-&gt;setTarget(null);         \$objArvoreAcaoItemAPI-&gt;setHref('javascript:alert(\'Ícone Processo ABC\');');          // \$objArvoreAcaoItemAPI-&gt;setTarget('_blank');         // \$objArvoreAcaoItemAPI-&gt;setTarget('ifrVisualizacao');         // \$objArvoreAcaoItemAPI-&gt;setHref(SessaoSEI::getInstance()-         &gt;assinarLink('controlador.php?acao=md_abc_processo_processar&amp;id_procedimento=' .         \$dbIdProcedimento . '&amp;arvore=1'));          \$objArvoreAcaoItemAPI-&gt;setSinHabilitado('S');          \$arrObjArvoreAcaoItemAPI[] = \$objArvoreAcaoItemAPI;     }      return \$arrObjArvoreAcaoItemAPI; } </pre>	
	

## montarMensagemProcesso

	Entrada
objProcedimentoAPI	Ocorrência de ProcedimentoAPI preenchida com: IdProcedimento, NumeroProtocolo, IdTipoProcedimento, NomeTipoProcedimento, NivelAcesso, GrauSigilo, IdHipotesLegal, IdUnidadeGeradora, IdOrgaoUnidadeGeradora, CodigoAcesso e SinAberto

Saída	
strMsg	Texto da mensagem
Exemplo	
<pre> public function montarMensagemProcesso(ProcedimentoAPI \$objProcedimentoAPI){     \$strMsg = null;      \$objInfraParametro = new InfraParametro(BancoSEI::getInstance());     \$numIdTipoProcAbc = \$objInfraParametro-&gt;getValor('MD_ABC_ID_TIPO_PROCEDIM ENTO_TESTE', false);      if (\$objProcedimentoAPI-&gt;getSinAberto()=='S' &amp;&amp;         \$objProcedimentoAPI-&gt;getIdTipoProcedimento()==\$numIdTipoProcAbc) {          \$strMsg = 'Mensagem do módulo ABC...';      }      return \$strMsg; } </pre>	
	

## montarMenuPublicacoes

Saída	
\$arrItens	Um array de itens montados com a estrutura: [nível]^[url]^[título]^[rótulo]^[target (opcional)]  Onde o nível é representado pelo número de "hífens" informados.
Exemplo	
<pre> public function montarMenuPublicacoes(){      \$strURL = ConfiguracaoSEI::getInstance()-&gt;getValor('SEI','URL');      \$arrMenu = array();     \$arrMenu[] = '- ^'. \$strURL.'/publicacoes/controlador_publicacoes.php?acao=md_abc_publicacao_exemplo^For mulário Exemplo ABC - Publicações^Publicação ABC '; } </pre>	



```

    $arrMenu[] = '-^#^Sites de busca sugeridos^Sites de Busca';
    $arrMenu[] = '--^http://www.google.com^Página do Google^Google^_blank';
    $arrMenu[] = '--^http://br.search.yahoo.com^Página do Yahoo!^Yahoo!^_blank';
    return $arrMenu;
}

```

## montarMenuUsuarioExterno

Saída	
\$arrItens	Um array de itens montados com a estrutura: [nível]^[url]^[título]^[rótulo]^[target (opcional)]  Onde o nível é representado pelo número de "hífens" informados.
Exemplo	
<pre> public function montarMenuUsuarioExterno(){      \$strURL = ConfiguracaoSEI::getInstance()-&gt;getValor('SEI','URL');      \$arrMenu = array();     \$arrMenu[] = '-^'. \$strURL. '/controlador_externo.php?acao=md_abc_usuario_externo_exemplo^Formulário Exemplo ABC - Usuário Externo^Usuário Externo ABC';     \$arrMenu[] = '-^#^Sites de busca sugeridos^Sites de Busca';     \$arrMenu[] = '--^http://www.google.com^Página do Google^Google^_blank';     \$arrMenu[] = '--^http://br.search.yahoo.com^Página do Yahoo!^Yahoo!^_blank';     return \$arrMenu; } </pre>	

## moverDocumento

Entrada	
\$objDocumentoAPI	Instância preenchida com IdDocumento e NumeroProtocolo.
\$objProcedimentoAPIOrigem	Instância preenchida com IdProcedimento e



	NumeroProtocolo.
\$objProcedimentoAPIDestino	Instância preenchida com IdProcedimento e NumeroProtocolo.

### permitirAndamentoConcluido

Entrada	
\$objAndamentoAPI	Instância preenchida com IdProtocolo e IdTarefa.
Saída	
\$bolPermitir	Retornar true para permitir lançar o andamento concluído.
Observações	
Permite ignorar a configuração padrão de um andamento (ver seção Andamentos).	

### processarControlador

Entrada	
\$strAcao	Ação recebida na URL
Exemplo	
<pre> public function processarControlador(\$strAcao){     switch(\$strAcao) {         case 'md_abc_processo_processar':             require_once dirname(__FILE__).'/frm_processo.php';             return true;          case 'md_abc_documento_processar':             require_once dirname(__FILE__).'/frm_documento.php';             return true;      }      return false; } </pre>	

### processarControladorAjax

Entrada	
\$strAcaoAjax	Ação ajax recebida na URL
Exemplo	
<pre> public function processarControladorAjax(\$strAcao){     \$xml = null;      switch(\$strAcao) {         case 'md_abc_assunto_auto_completar':             \$arrObjAssuntoDTO = AssuntoINT::autoCompletarAssuntosRI1223(\$_POST['palavras_pesquisa']);             \$xml = InfraAjax::gerarXMLItensArrInfraDTO(\$arrObjAssuntoDTO, 'IdAssunto', 'CodigoEstruturado');             break;     }     return \$xml; } </pre>	

## processarControladorAjaxExterno

Entrada	
\$strAcaoAjax	Ação ajax recebida na URL
Exemplo	
<pre>public function processarControladorAjaxExterno(\$strAcao){     \$xml = null;      switch(\$strAcao) {         case 'md_abc_cargo_auto_completar':             \$xml = InfraAjax::gerarXMLItensArrInfraDTO(CargoINT::autoCompletarExpressao(\$_POST['pala vras_pesquisa']), 'IdCargo', 'Expressao');             break;         }      return \$xml; }</pre>	

## processarControladorExterno

Entrada	
\$strAcao	Ação recebida na URL
Exemplo	
<pre>public function processarControladorExterno(\$strAcao){      switch(\$strAcao) {          case 'md_abc_usuario_externo_exemplo':             require_once dirname(__FILE__) . '/usuario_externo_exemplo.php';             return true;         }      return false; }</pre>	

## processarControladorPublicacoes

Entrada	
\$strAcao	Ação recebida na URL
Exemplo	
<pre>public function processarControladorPublicacoes(\$strAcao){      switch(\$strAcao) {          case 'md_abc_publicacao_exemplo':             require_once dirname(__FILE__) . '/publicacao_exemplo.php';             return true;         }      return false; }</pre>	

## processarControladorWebServices

Entrada	
\$strServico	Serviço recebido na URL
Observação	
O WSDL poderá ser referenciado por: https://[servidor]/sei/controlador_ws.php?servico=[nome do serviço do módulo]	
Exemplo	
<pre>public function processarControladorWebServices(\$strServico){      \$strArq = null;      switch (\$strServico) {          case 'md_abc_servico_1':             \$strArq = 'servico1.wsdl';             break;          case 'md_abc_servico_2':             \$strArq = 'servico2.wsdl';             break;      }      if (\$strArq!=null){         \$strArq = dirname(__FILE__).'/ws/'.\$strArq;     }      return \$strArq; }</pre>	

## reabrirProcesso

Entrada	
\$objProcedimentoAPI	Instância preenchida com IdProcedimento.

## relacionarProcesso

Entrada	
\$objProcedimentoAPI1	Instância preenchida com IdProcedimento e NumeroProtocolo.
\$objProcedimentoAPI2	Instância preenchida com IdProcedimento e NumeroProtocolo.

## removerRelacionamentoProcesso

Entrada	
\$objProcedimentoAPI1	Instância preenchida com IdProcedimento e NumeroProtocolo.
\$objProcedimentoAPI2	Instância preenchida com IdProcedimento e NumeroProtocolo.

## removerSobrestamentoProcesso

Entrada	
\$objProcedimentoAPI	Instância preenchida com IdProcedimento e NumeroProtocolo.
\$objProcedimentoAPIVinculado	Instância preenchida com IdProcedimento e NumeroProtocolo (nulo se não estiver vinculado a outro)

	processo)
--	-----------

## sobrestarProcesso

Entrada	
\$objProcedimentoAPI	Instância preenchida com IdProcedimento e NumeroProtocolo.
\$objProcedimentoAPIVinculado	Instância preenchida com IdProcedimento e NumeroProtocolo (nulo se não estiver vinculado a outro processo)

## verificarAcessoProtocolo

Entrada	
\$arrObjProcedimentoAPI	Instâncias preenchidas com IdProcedimento, IdTipoProcedimento, IdUnidadeGeradora e NivelAcesso.
\$arrObjDocumentoAPI	Instâncias preenchidas com IdDocumento, IdProcedimento, IdSerie, IdUnidadeGeradora, Tipo, SubTipo e NivelAcesso.
Saída	
\$arrLiberacoes	Retornar um array indexado pelo número do protocolo onde cada item possui o valor P (Permitido) ou N (Negado) indicando o tipo de acesso ao processo ou documento. Também podem ser utilizadas as constantes abaixo da classe SeiIntegracao:  <pre>//TAM = Tipo Acesso Modulo public static \$TAM_PERMITIDO = 'P'; public static \$TAM_NEGADO = 'N';</pre>
Observações	
<p>Possibilita que o módulo permita ou negue acesso a determinados processos ou documentos. Se o acesso for exclusivamente devido ao módulo então ao visualizar a árvore de processo aparecerá ao lado do processo ou documento um cadeado aberto indicando o módulo que liberou o acesso. Da mesma forma se o acesso foi negado exclusivamente pelo módulo então será visualizado um cadeado fechado. Em situações conflitantes onde um módulo permite acesso e outro nega a negação terá prioridade.</p> <p>ATENÇÃO: Este evento requer cuidado redobrado na sua elaboração pois falhas na lógica de programação podem expor indevidamente informações restritas ou sigilosas. Todas as visualizações de processos ou documentos serão auditadas.</p>	
Exemplo	
<pre>//Libera rascunhos da serie de teste em processos que não sejam sigilosos.  public function verificarAcessoProtocolo(\$arrObjProcedimentoAPI, \$arrObjDocumentoAPI){      \$ret = null;      \$objInfraParametro = new InfraParametro(BancoSEI::getInstance());     \$numIdSerieAbc = \$objInfraParametro-&gt;getValor('MD_ABC_ID_SERIE_TESTE', false);      foreach(\$arrObjDocumentoAPI as \$objDocumentoAPI){         if (\$objDocumentoAPI-&gt;getIdSerie() == \$numIdSerieAbc &amp;&amp;             \$objDocumentoAPI-&gt;getSubTipo() == DocumentoRN::\$TD_EDITOR_INTERNO &amp;&amp;</pre>	

```

$ObjDocumentoAPI->getNivelAcesso() != ProtocoloRN::$NA_SIGILOSO){
    $ret[$ObjDocumentoAPI->getIdDocumento()] = SeiIntegracao::$TAM_PERMITIDO;
}
}

return $ret;
}

```



## verificarAcessoProtocoloExterno

Entrada	
\$arrObjProcedimentoAPI	Instâncias representando os processos anexados preenchidas com IdProcedimento, IdTipoProcedimento, IdUnidadeGeradora e NivelAcesso.
\$arrObjDocumentoAPI	Instâncias preenchidas com IdDocumento, IdProcedimento, IdSerie, IdUnidadeGeradora, SinAssinado, SinPublicado, Tipo, SubTipo e NivelAcesso.
Saída	
\$arrLiberacoes	Retornar um array indexado pelo número do protocolo onde cada item possui o valor P (Permitido) ou N (Negado) indicando o tipo de acesso ao processo ou documento. Também podem ser utilizadas as constantes abaixo da classe SeiIntegracao: <pre> //TAM = Tipo Acesso Modulo public static \$TAM_PERMITIDO = 'P'; public static \$TAM_NEGADO = 'N'; </pre>
Observações	
Possibilita que o módulo permita ou negue acesso a determinados processos anexados ou documentos na consulta de acesso externo. Em situações conflitantes onde um módulo permite acesso e outro nega a negação terá prioridade.	
ATENÇÃO: Este evento requer cuidado redobrado na sua elaboração pois falhas na lógica de programação podem expor indevidamente informações restritas ou sigilosas. Todas as visualizações de processos ou documentos serão auditadas.	
Exemplo	

```
//Bloqueia no acesso externo documentos da serie de teste que não estão publicados

public function verificarAcessoProtocoloExterno($arrObjProcedimentoAPI,
$arrObjDocumentoAPI){

    $ret = null;

    $objInfraParametro = new InfraParametro(BancoSEI::getInstance());
    $numIdSerieAbc = $objInfraParametro->getValor('MD_ABC_ID_SERIE_TESTE', false);

    foreach($arrObjDocumentoAPI as $objDocumentoAPI){
        if ($objDocumentoAPI->getIdSerie() == $numIdSerieAbc && $objDocumentoAPI->getSinPublicado()=='N'){
            $ret[$objDocumentoAPI->getIdDocumento()] = SeiIntegracao::$TAM_NEGADO;
        }
    }

    return $ret;
}
```

## 6. Operações

Os módulos podem realizar operações no sistema como geração de processos, inclusão de documentos e atualização de andamentos utilizando os objetos da API em conjunto com os métodos disponibilizados na classe SeiRN. A camada de Web Services do SEI também utiliza a API, sendo assim, todas as operações realizadas por meio de Web Services também podem ser realizadas diretamente pelos módulos.

**Não é recomendado o uso de outras classes ou objetos internos do sistema.** Eventuais alterações nos objetos da API e métodos da classe SeiRN serão documentadas em um roteiro de adaptação para os módulos. Alterações em outros componentes internos do sistema não serão documentadas.

Sempre que uma operação possuir opção de uso do ID interno ou do número de protocolo deve-se optar pelo ID, caso a informação esteja disponível, pois isso evitará um ou mais acessos ao banco.

### adicionarArquivo

Entrada
Instância do objeto EntradaAdicionarArquivoAPI preenchida com: <ul style="list-style-type: none"> <li>• Nome</li> <li>• Tamanho</li> <li>• Hash</li> <li>• Conteúdo</li> </ul>
Saída
Retornar o identificador do arquivo criado no repositório.
Exemplo
<pre>\$strArquivo = DIR_SEI_TEMP.'/exemplo.pdf'; \$fp = fopen(\$strArquivo, "rb");</pre>

```

$objEntradaAdicionarArquivoAPI = new EntradaAdicionarArquivoAPI();
$objEntradaAdicionarArquivoAPI->setNome('exemplo.pdf');
$objEntradaAdicionarArquivoAPI->setTamanho(filesize($strArquivo));
$objEntradaAdicionarArquivoAPI->setHash(md5_file($strArquivo));
$objEntradaAdicionarArquivoAPI->setConteudo(base64_encode(fread($fp, 1024*1024)));
//blocos de 1Mb

$objSeiRN = new SeiRN();
$numIdArquivo = $objSeiRN->adicionarArquivo($objEntradaAdicionarArquivoAPI);

while (!feof($fp)) {
    $objEntradaAdicionarConteudoArquivoAPI = new EntradaAdicionarConteudoArquivoAPI();
    $objEntradaAdicionarConteudoArquivoAPI->setIdArquivo($numIdArquivo);
    $objEntradaAdicionarConteudoArquivoAPI->setConteudo(base64_encode(fread($fp,
1024*1024))); //blocos de 1Mb
    $objSeiRN->adicionarConteudoArquivo($objEntradaAdicionarConteudoArquivoAPI);
}

fclose($fp);

```

## adicionarConteudoArquivo

Entrada
Instância do objeto EntradaAdicionarConteudoArquivoAPI preenchida com: <ul style="list-style-type: none"> <li>IdArquivo</li> <li>Conteudo</li> </ul>
Observação
Ver exemplo em adicionarArquivo.

## anexarProcesso

Entrada
Instância do objeto EntradaAnexarProcessoAPI preenchida com: <ul style="list-style-type: none"> <li>IdProcedimentoPrincipal ou ProtocoloProcedimentoPrincipal</li> <li>IdProcedimentoAnexado ou ProtocoloProcedimentoAnexado</li> </ul>
Exemplo
<pre> \$objEntradaAnexarProcessoAPI = new EntradaAnexarProcessoAPI(); // \$objEntradaAnexarProcessoAPI-&gt;setIdProcedimentoPrincipal(); \$objEntradaAnexarProcessoAPI-&gt;setProtocoloProcedimentoPrincipal('0000495- 63.2014.4.04.8000'); // \$objEntradaAnexarProcessoAPI-&gt;setIdProcedimentoAnexado(); \$objEntradaAnexarProcessoAPI-&gt;setProtocoloProcedimentoAnexado('0000504- 25.2014.4.04.8000');  \$objSeiRN = new SeiRN(); \$objSeiRN-&gt;anexarProcesso(\$objEntradaAnexarProcessoAPI); </pre>

## atribuirProcesso

Entrada
Instância do objeto EntradaAtribuirProcessoAPI preenchida com: <ul style="list-style-type: none"> <li>IdProcedimento ou ProtocoloProcedimento</li> <li>IdUsuario</li> <li>SinReabrir</li> </ul>
Exemplo

```

$objEntradaAtribuirProcessoAPI = new EntradaAtribuirProcessoAPI();
//$objEntradaAtribuirProcessoAPI->setIdProcedimento( );
$objEntradaAtribuirProcessoAPI->setProtocoloProcedimento('0000262-95.2016.4.04.8000');
$objEntradaAtribuirProcessoAPI->setIdUsuario(100002382);
$objEntradaAtribuirProcessoAPI->setSinReabrir('S');

$objSeiRN = new SeiRN();
$objSeiRN->atribuirProcesso($objEntradaAtribuirProcessoAPI);

```

## atualizarContatos

Entrada
<p>Conjunto de Instâncias do objeto ContatoAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• StaOperacao</li> <li>• IdContato</li> <li>• IdTipoContato</li> <li>• Sigla</li> <li>• Nome</li> <li>• StaNatureza</li> <li>• IdContatoAssociado</li> <li>• SinEnderecoAssociado</li> <li>• Endereco</li> <li>• Complemento</li> <li>• Bairro</li> <li>• IdCidade</li> <li>• IdEstado</li> <li>• IdPais</li> <li>• Cep</li> <li>• StaGenero</li> <li>• IdCargo</li> <li>• Cpf</li> <li>• Cnpj</li> <li>• Rg</li> <li>• OrgaoExpedidor</li> <li>• Matricula</li> <li>• MatriculaOab</li> <li>• TelefoneFixo</li> <li>• TelefoneCelular</li> <li>• DataNascimento</li> <li>• Email</li> <li>• SitioInternet</li> <li>• Observacao</li> <li>• SinAtivo</li> </ul>
Exemplo
<pre> \$objContatoAPI = new ContatoAPI(); \$objContatoAPI-&gt;setStaOperacao('A'); \$objContatoAPI-&gt;setIdContato(100003924); \$objContatoAPI-&gt;setIdTipoContato(1); \$objContatoAPI-&gt;setSigla('def'); </pre>



```

$objContatoAPI->setNome('Dddddd Eeeee Fffff');
$objContatoAPI->setStaNatureza('F');
$objContatoAPI->setIdContatoAssociado(null);
$objContatoAPI->setSinEnderecoAssociado('N');
$objContatoAPI->setEndereco('Rua Otávio Francisco Caruso da Rocha, 2000');
$objContatoAPI->setComplemento(null);
$objContatoAPI->setBairro('Praia de Belas');
$objContatoAPI->setIdCidade(4927);
$objContatoAPI->setIdEstado(23);
$objContatoAPI->setIdPais(76);
$objContatoAPI->setCep('90010-395');
$objContatoAPI->setStaGenero('M');
$objContatoAPI->setIdCargo(100001865);
$objContatoAPI->setCpf(11572070730);
$objContatoAPI->setCnpj(null);
$objContatoAPI->setRg(56584632);
$objContatoAPI->setOrgaoExpedidor('SSP/RS');
$objContatoAPI->setMatricula('54834');
$objContatoAPI->setMatriculaOab(null);
$objContatoAPI->setTelefoneFixo(null);
$objContatoAPI->setTelefoneCelular('(99)9999-9999');
$objContatoAPI->setDataNascimento('13/05/1971');
$objContatoAPI->setEmail('def@abc.gov.br');
$objContatoAPI->setSitioInternet(null);
$objContatoAPI->setObservacao('Exemplo alteração contato.');
```

```

$objSeiRN = new SeiRN();
$objSeiRN->atualizarContatos(array($objContatoAPI));

```

## bloquearProcesso

Entrada
<p>Instância do objeto EntradaBloquearProcessoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>IdProcedimento ou ProtocoloProcedimento</li> </ul>
Observações
<p>Quando um processo estiver bloqueado as operações abaixo NÃO serão permitidas:</p> <ul style="list-style-type: none"> <li>alterar cadastro do processo</li> <li>alterar cadastro do documento</li> <li>incluir documento no processo</li> <li>alterar conteúdo de documentos (internos/externos)</li> <li>enviar processo</li> <li>enviar email</li> <li>agendar publicação</li> <li>assinar documento</li> <li>mover documento</li> <li>cancelar documento</li> <li>gerar circular</li> <li>excluir documento</li> <li>excluir processo</li> <li>sobrestar processo</li> <li>anexar processo</li> </ul> <p>Operações permitidas em processos bloqueados:</p> <ul style="list-style-type: none"> <li>atribuir marcador da unidade (*)</li> </ul>

- atribuir processo (\*)
- atualizar andamento (\*)
- liberar/cancelar acesso externo (\*)
- incluir em bloco de reunião (\*)
- relacionar processo (\*)
- duplicar processo (\*)
- dar ciência (\*)
- registrar anotação
- cadastrar acompanhamento especial
- solicitar desarquivamento
- concluir processo
- reabrir processo

\* somente com processo aberto

Não é possível bloquear processos sigilosos.

#### Exemplo

```
$objEntradaBloquearProcessoAPI = new EntradaBloquearProcessoAPI();
// $objEntradaBloquearProcessoAPI->setIdProcedimento();
$objEntradaBloquearProcessoAPI->setProtocoloProcedimento('0000262-95.2016.4.04.8000');

$objSeiRN = new SeiRN();
$objSeiRN->bloquearProcesso($objEntradaBloquearProcessoAPI);
```

### cancelarDisponibilizacaoBloco

#### Entrada

Instância do objeto EntradaCancelarDisponibilizacaoBlocoAPI preenchida com:

- IdBloco

#### Exemplo

```
$objEntradaCancelarDisponibilizacaoBlocoAPI = new
EntradaCancelarDisponibilizacaoBlocoAPI();
$objEntradaCancelarDisponibilizacaoBlocoAPI->setIdBloco(932);

$objSeiRN = new SeiRN();
$objSeiRN->cancelarDisponibilizacaoBloco($objEntradaCancelarDisponibilizacaoBlocoAPI);
```

### cancelarDocumento

#### Entrada

Instância do objeto EntradaCancelarDocumentoAPI preenchida com:

- IdDocumento ou ProtocoloDocumento
- Motivo

#### Exemplo

```
$objEntradaCancelarDocumentoAPI = new EntradaCancelarDocumentoAPI();
// $objEntradaCancelarDocumentoAPI->setIdDocumento();
$objEntradaCancelarDocumentoAPI->setProtocoloDocumento('0031076');
$objEntradaCancelarDocumentoAPI->setMotivo('Exemplo cancelamento de documento módulo
ABC.');
```

```
$objSeiRN = new SeiRN();
$objSeiRN->cancelarDocumento($objEntradaCancelarDocumentoAPI);
```

## concluirProcesso

Entrada
<p>Instância do objeto EntradaConcluirProcessoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdProcedimento ou ProtocoloProcedimento</li> </ul>
Exemplo
<pre>\$objEntradaConcluirProcessoAPI = new EntradaConcluirProcessoAPI(); // \$objEntradaConcluirProcessoAPI-&gt;setIdProcedimento(); \$objEntradaConcluirProcessoAPI-&gt;setProtocoloProcedimento('0000262-95.2016.4.04.8000');  \$objSeiRN = new SeiRN(); \$objSeiRN-&gt;concluirProcesso(\$objEntradaConcluirProcessoAPI);</pre>

## consultarBloco

Entrada
<p>Instância do objeto EntradaConsultarBlocoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdBloco</li> <li>• SinRetornarProtocolos</li> </ul>
Saída
<p>Instância do objeto SaidaConsultarBlocoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdBloco</li> <li>• Descricao</li> <li>• Tipo</li> <li>• Estado</li> <li>• Unidade (UnidadeAPI) <ul style="list-style-type: none"> <li>○ IdUnidade</li> <li>○ Sigla</li> <li>○ Descricao</li> </ul> </li> <li>• Usuario (UsuarioAPI) <ul style="list-style-type: none"> <li>○ IdUsuario</li> <li>○ Sigla</li> <li>○ Nome</li> </ul> </li> <li>• UnidadesDisponibilizacao (array: UnidadeAPI) <ul style="list-style-type: none"> <li>○ IdUnidade</li> <li>○ Sigla</li> <li>○ Descricao</li> </ul> </li> <li>• Protocolos (array: ProtocoloBlocoAPI) <ul style="list-style-type: none"> <li>○ ProtocoloFormatado</li> <li>○ Identificacao</li> <li>○ Assinaturas (array: AssinaturaAPI) <ul style="list-style-type: none"> <li>▪ Nome</li> <li>▪ CargoFuncao</li> <li>▪ DataHora</li> </ul> </li> </ul> </li> </ul>
Exemplo

```

$objEntradaConsultarBlocoAPI = new EntradaConsultarBlocoAPI();
$objEntradaConsultarBlocoAPI->setIdBloco(911);
$objEntradaConsultarBlocoAPI->setSinRetornarProtocolos('S');

$objSeiRN = new SeiRN();
$objSaidaConsultarBlocoAPI = $objSeiRN->consultarBloco($objEntradaConsultarBlocoAPI);

SaidaConsultarBlocoAPI Object
(
    [IdBloco] => 911
    [Descricao] => Teste...
    [Tipo] => A
    [Estado] => D
    [Unidade] => UnidadeAPI Object
        (
            [IdUnidade] => 110000001
            [Sigla] => TESTE
            [Descricao] => Unidade de Teste
        )
    [Usuario] => UsuarioAPI Object
        (
            [IdUsuario] => 33645
            [Sigla] => abc
            [Nome] => Aaaaaa Bbbbbb Cccccc
        )
    [UnidadesDisponibilizacao] => Array
        (
            [0] => UnidadeAPI Object
                (
                    [IdUnidade] => 100000555
                    [Sigla] => DG
                    [Descricao] => Diretoria-geral
                )
            [1] => UnidadeAPI Object
                (
                    [IdUnidade] => 100000644
                    [Sigla] => VPRES
                    [Descricao] => Vice-presidência
                )
        )
    [Protocolos] => Array
        (
            [0] => ProtocoloBlocoAPI Object
                (
                    [ProtocoloFormatado] => 0011108
                    [Identificacao] => Ato 52
                    [Assinaturas] => Array
                        (
                            [0] => AssinaturaAPI Object
                                (
                                    [Nome] => Dddddd Eeeeeee Fffffff
                                    [CargoFuncao] => Supervisor
                                    [DataHora] => 21/19/2016 16:59:12
                                )
                            )
                )
            [1] => ProtocoloBlocoAPI Object
                (

```



- Assinaturas (array:AssinaturaAPI)
  - Nome
  - CargoFuncao
  - DataHora
- Publicacao (array:PublicacaoAPI)
  - NomeVeiculo
  - Numero
  - DataDisponibilizacao
  - DataPublicacao
  - Estado
  - ImprensaNacional (PublicacaoImprensaNacionalAPI)
    - SiglaVeiculo
    - DescricaoVeiculo
    - Pagina
    - Secao
  - Data
  - Campos (CampoAPI)
    - Nome
    - Valor
- LinkAcesso (link não assinado para montar a árvore de processo posicionando no documento)

#### Observações

Cada um dos sinalizadores do objeto de entrada implica em processamento adicional realizado pelo sistema, sendo assim, recomenda-se que seja solicitado o retorno somente para informações estritamente necessárias. Todos os sinalizadores de entrada são opcionais com valor padrão N.

#### Exemplo

```
$objEntradaConsultarDocumentoAPI = new EntradaConsultarDocumentoAPI();
//$objEntradaConsultarDocumentoAPI->setIdDocumento();
$objEntradaConsultarDocumentoAPI->setProtocoloDocumento('0023930');
$objEntradaConsultarDocumentoAPI->setSinRetornarAssinaturas('S');
$objEntradaConsultarDocumentoAPI->setSinRetornarPublicacao('S');

$objSeiRN = new SeiRN();
$objSaidaConsultarDocumentoAPI = $objSeiRN-
>consultarDocumento($objEntradaConsultarDocumentoAPI);

SaidaConsultarDocumentoAPI Object
(
    [IdProcedimento] => 42039220124056042
    [ProcedimentoFormatado] => 0000277-98.2015.4.04.8000
    [IdDocumento] => 42039220124060399
    [DocumentoFormatado] => 0023930
    [LinkAcesso] => https://sei-
desenv.trf4.jus.br/mga/controlador.php?acao=procedimento_trabalhar&id_procedimento=42039
220124056042&id_documento=42039220124060399
    [Serie] => SerieAPI Object
    (
        [IdSerie] => 371
        [Nome] => Ata
    )

    [Numero] =>
    [Data] => 03/06/2015
    [UnidadeElaboradora] => UnidadeAPI Object
```

```

    (
        [IdUnidade] => 110000001
        [Sigla] => TESTE
        [Descricao] => Unidade de Teste
    )

    [Assinaturas] => Array
    (
        [0] => AssinaturaAPI Object
        (
            [Nome] => Aaaaaa Bbbbbb Cccccc
            [CargoFuncao] => Diretor
            [DataHora] => 10/06/2015 15:31:00
        )
    )

    [Publicacao] => PublicacaoAPI Object
    (
        [NomeVeiculo] => Diário Eletrônico Administrativo
        [Numero] =>
        [DataDisponibilizacao] => 12/06/2015
        [DataPublicacao] =>
        [Estado] => A
        [ImprensaNacional] => PublicacaoImprensaNacionalAPI Object
        (
            [SiglaVeiculo] => DJU
            [DescricaoVeiculo] => Diário da Justiça
            [Pagina] => 122
            [Secao] => 2
            [Data] => 11/06/2015
        )
    )
)

```

## consultarProcedimento

Entrada
<p>Instância do objeto EntradaConsultarProcedimentoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdProcedimento ou ProtocoloProcedimento</li> <li>• SinRetornarAssuntos</li> <li>• SinRetornarInteressados</li> <li>• SinRetornarObservacoes</li> <li>• SinRetornarAndamentoGeracao</li> <li>• SinRetornarAndamentoConclusao</li> <li>• SinRetornarUltimoAndamento</li> <li>• SinRetornarUnidadesProcedimentoAberto</li> <li>• SinRetornarProcedimentosRelacionados</li> <li>• SinRetornarProcedimentosAnexados</li> </ul>
Saída
<p>Instância do objeto SaidaConsultarProcedimentoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdProcedimento</li> <li>• ProcedimentoFormatado</li> <li>• Especificacao</li> <li>• DataAutuacao</li> <li>• TipoProcedimento</li> </ul>

- IdTipoProcedimento
  - Nome
- Assuntos (array:AssuntoAPI)
  - CodigoEstruturado
  - Descricao
- Interessados (array:InteressadoAPI)
  - Sigla
  - Nome
- Observacoes (array:ObservacaoAPI)
  - Descricao
  - Unidade (UnidadeAPI)
    - IdUnidade
    - Sigla
    - Descricao
- AndamentoGeracao (array:AndamentoAPI)
  - Descricao
  - DataHora
  - Usuario (UsuarioAPI)
    - IdUsuario
    - Sigla
    - Nome
  - Unidade (UnidadeAPI)
    - IdUnidade
    - Sigla
    - Descricao
- AndamentoConclusao (array:AndamentoAPI)
  - Descricao
  - DataHora
  - Usuario (UsuarioAPI)
    - IdUsuario
    - Sigla
    - Nome
  - Unidade (UnidadeAPI)
    - IdUnidade
    - Sigla
    - Descricao
- UltimoAndamento (array:AndamentoAPI)
  - Descricao
  - DataHora
  - Usuario (UsuarioAPI)
    - IdUsuario
    - Sigla
    - Nome
  - Unidade (UnidadeAPI)
    - IdUnidade
    - Sigla
    - Descricao
- UnidadesProcedimentoAberto (array:UnidadeProcedimentoAbertoAPI)
  - Unidade (UnidadeAPI)



- IdUnidade
  - Sigla
  - Descricao
- UsuarioAtribuicao (UsuarioAPI)
  - IdUsuario
  - Sigla
  - Nome
- ProcedimentosRelacionados (array:ProcedimentoResumidoAPI)
  - IdProcedimento
  - ProcedimentoFormatado
  - TipoProcedimento
    - IdTipoProcedimento
    - Nome
- ProcedimentosAnexados (array:ProcedimentoResumidoAPI)
  - IdProcedimento
  - ProcedimentoFormatado
  - TipoProcedimento
    - IdTipoProcedimento
    - Nome
- LinkAcesso (link não assinado para montar a árvore de processo)

#### Observações

Cada um dos sinalizadores do objeto de entrada implica em processamento adicional realizado pelo sistema, sendo assim, recomenda-se que seja solicitado o retorno somente para informações estritamente necessárias. Todos os sinalizadores de entrada são opcionais com valor padrão N.

#### Exemplo

```
$objEntradaConsultarProcedimentoAPI = new EntradaConsultarProcedimentoAPI();
// $objEntradaConsultarProcedimentoAPI->setIdProcedimento();
$objEntradaConsultarProcedimentoAPI->setProtocoloProcedimento('0000486-33.2016.4.04.8000');
$objEntradaConsultarProcedimentoAPI->setSinRetornarAndamentoGeracao('S');
$objEntradaConsultarProcedimentoAPI->setSinRetornarUnidadesProcedimentoAberto('S');

$objSeiRN = new SeiRN();
$objSaidaConsultarProcedimentoAPI = $objSeiRN-
>consultarProcedimento($objEntradaConsultarProcedimentoAPI);

SaidaConsultarProcedimentoAPI Object
(
    [IdProcedimento] => 10000000012635
    [ProcedimentoFormatado] => 0000486-33.2016.4.04.8000
    [Especificacao] => Exemplo ABC com 2 documentos
    [DataAutuacao] => 04/10/2016
    [LinkAcesso] => https://sei-desenv.trf4.jus.br/mga/controlador.php?acao=procedimento_trabalhar&id_procedimento=10000000012635
    [TipoProcedimento] => TipoProcedimentoAPI Object
    (
        [IdTipoProcedimento] => 100000312
        [Nome] => Autorização
    )

    [AndamentoGeracao] => AndamentoAPI Object
    (
        [Descricao] => Processo público gerado
        [DataHora] => 04/10/2016 18:08:59
        [Usuario] => UsuarioAPI Object
    )
)
```

```

        (
            [IdUsuario] => 100002382
            [Sigla] => abc
            [Nome] => Aaaaaa Bbbbbb Cccccc
        )

        [Unidade] => UnidadeAPI Object
        (
            [IdUnidade] => 110000001
            [Sigla] => TESTE
            [Descricao] => Unidade de Teste
        )
    )
    [UnidadesProcedimentoAberto] => Array
    (
        [0] => UnidadeProcedimentoAbertoAPI Object
        (
            [Unidade] => UnidadeAPI Object
            (
                [IdUnidade] => 100000555
                [Sigla] => DG
                [Descricao] => Diretoria-geral
            )
            [UsuarioAtribuicao] =>
        )

        [1] => UnidadeProcedimentoAbertoAPI Object
        (
            [Unidade] => UnidadeAPI Object
            (
                [IdUnidade] => 100000644
                [Sigla] => VPRES
                [Descricao] => Vice-presidência
            )
            [UsuarioAtribuicao] =>
        )
    )
)

```

## definirMarcador

Saída
<p>Conjunto de Instâncias do objeto DefinicaoMarcadorAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>IdProcedimento ou ProtocoloProcedimento</li> <li>IdMarcador</li> <li>Texto</li> </ul>
Exemplo
<pre> \$arrObjDefinicaoMarcadorAPI = array();  \$objDefinicaoMarcadorAPI = new DefinicaoMarcadorAPI(); // \$objDefinicaoMarcadorAPI-&gt;setIdProcedimento(); \$objDefinicaoMarcadorAPI-&gt;setProtocoloProcedimento('0000488-03.2016.4.04.8000'); \$objDefinicaoMarcadorAPI-&gt;setIdMarcador(9); \$objDefinicaoMarcadorAPI-&gt;setTexto('Exemplo definição marcador processo módulo ABC.');</pre> <pre> \$arrObjDefinicaoMarcadorAPI[] = \$objDefinicaoMarcadorAPI;  \$objDefinicaoMarcadorAPI = new DefinicaoMarcadorAPI(); // \$objDefinicaoMarcadorAPI-&gt;setIdProcedimento(); \$objDefinicaoMarcadorAPI-&gt;setProtocoloProcedimento('0000442-14.2016.4.04.8000'); \$objDefinicaoMarcadorAPI-&gt;setIdMarcador(4); </pre>

```
$objDefinicaoMarcadorAPI->setTexto('Exemplo definição marcador processo módulo ABC.');
```

```
$arrObjDefinicaoMarcadorAPI[] = $objDefinicaoMarcadorAPI;
```

```
$objSeiRN = new SeiRN();
```

```
$objSeiRN->definirMarcador($arrObjDefinicaoMarcadorAPI);
```

## desanexarProcesso

Entrada
<p>Instância do objeto EntradaDesanexarProcessoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>IdProcedimentoPrincipal ou ProtocoloProcedimentoPrincipal</li> <li>IdProcedimentoAnexado ou ProtocoloProcedimentoAnexado</li> <li>Motivo</li> </ul>
Exemplo
<pre>\$objEntradaDesanexarProcessoAPI = new EntradaDesanexarProcessoAPI();</pre> <pre>// \$objEntradaDesanexarProcessoAPI-&gt;setIdProcedimentoPrincipal();</pre> <pre>\$objEntradaDesanexarProcessoAPI-&gt;setProtocoloProcedimentoPrincipal('0000495-63.2014.4.04.8000');</pre> <pre>// \$objEntradaDesanexarProcessoAPI-&gt;setIdProcedimentoAnexado();</pre> <pre>\$objEntradaDesanexarProcessoAPI-&gt;setProtocoloProcedimentoAnexado('0000504-25.2014.4.04.8000');</pre> <pre>\$objEntradaDesanexarProcessoAPI-&gt;setMotivo('Exemplo desanexação de processo módulo ABC.');</pre> <pre>\$objSeiRN = new SeiRN();</pre> <pre>\$objSeiRN-&gt;desanexarProcesso(\$objEntradaDesanexarProcessoAPI);</pre>

## desbloquearProcessoBloco

Entrada
<p>Instância do objeto EntradaBloquearProcessoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>IdProcedimento ou ProtocoloProcedimento</li> </ul>
Exemplo
<pre>\$objEntradaDesbloquearProcessoAPI = new EntradaDesbloquearProcessoAPI();</pre> <pre>// \$objEntradaDesbloquearProcessoAPI-&gt;setIdProcedimento();</pre> <pre>\$objEntradaDesbloquearProcessoAPI-&gt;setProtocoloProcedimento('0000262-95.2016.4.04.8000');</pre> <pre>\$objSeiRN = new SeiRN();</pre> <pre>\$objSeiRN-&gt;desbloquearProcesso(\$objEntradaDesbloquearProcessoAPI);</pre>

## disponibilizarBloco

Entrada
<p>Instância do objeto EntradaDisponibilizarBlocoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>IdBloco</li> </ul>
Exemplo
<pre>\$objEntradaDisponibilizarBlocoAPI = new EntradaDisponibilizarBlocoAPI();</pre> <pre>\$objEntradaDisponibilizarBlocoAPI-&gt;setIdBloco(932);</pre> <pre>\$objSeiRN = new SeiRN();</pre> <pre>\$objSeiRN-&gt;disponibilizarBloco(\$objEntradaDisponibilizarBlocoAPI);</pre>

## enviarProcesso

Entrada
Instância do objeto EntradaEnviarProcessoAPI preenchida com: <ul style="list-style-type: none"><li>• IdProcedimento ou ProtocoloProcedimento</li><li>• UnidadesDestino</li><li>• SinManterAbertoUnidade (valor padrão N)</li><li>• SinRemoverAnotacao (valor padrão N)</li><li>• SinEnviarEmailNotificacao (valor padrão N)</li><li>• DataRetornoProgramado (valor padrão nulo)</li><li>• DiasRetornoProgramado (valor padrão nulo)</li><li>• SinDiasUteisRetornoProgramado (valor padrão N)</li><li>• SinReabrir (valor padrão N)</li></ul>
Exemplo
<pre>\$objEntradaEnviarProcessoAPI = new EntradaEnviarProcessoAPI(); \$objEntradaEnviarProcessoAPI-&gt;setIdProcedimento(1000000012057); // \$objEntradaEnviarProcessoAPI-&gt;setProtocoloProcedimento( ); \$objEntradaEnviarProcessoAPI-&gt;setUnidadesDestino(array(10000555,10000644)); \$objEntradaEnviarProcessoAPI-&gt;setSinManterAbertoUnidade('S');  \$objSeiRN = new SeiRN(); \$objSeiRN-&gt;enviarProcesso(\$objEntradaEnviarProcessoAPI);</pre>

## excluirBloco

Entrada
Instância do objeto EntradaExcluirBlocoAPI preenchida com: <ul style="list-style-type: none"><li>• IdBloco</li></ul>
Exemplo
<pre>\$objEntradaExcluirBlocoAPI = new EntradaExcluirBlocoAPI(); \$objEntradaExcluirBlocoAPI-&gt;setIdBloco(933);  \$objSeiRN = new SeiRN(); \$objSeiRN-&gt;excluirBloco(\$objEntradaExcluirBlocoAPI);</pre>

## gerarBloco

Entrada
Instância do objeto EntradaGerarBlocoAPI preenchida com: <ul style="list-style-type: none"><li>• Tipo</li><li>• Descricao</li><li>• UnidadesDisponibilizacao</li><li>• Documentos ou IdDocumentos</li><li>• SinDisponibilizar</li></ul>
Saída
Retorna o ID do bloco gerado.
Exemplo
<pre>\$objEntradaGerarBlocoAPI = new EntradaGerarBlocoAPI(); \$objEntradaGerarBlocoAPI-&gt;setTipo('A'); \$objEntradaGerarBlocoAPI-&gt;setDescricao('Exemplo geração de bloco módulo ABC');</pre>

```

$objEntradaGerarBlocoAPI->setUnidadesDisponibilizacao(array('100000555','100000644'));

//se os identificadores estiverem disponíveis utilizar setIdDocumentos evitando
consulta ao banco
$objEntradaGerarBlocoAPI->setDocumentos(array('0031064','0030780'));
//$objEntradaGerarBlocoAPI->setIdDocumentos();

$objEntradaGerarBlocoAPI->setSinDisponibilizar('S');

$objSeiRN = new SeiRN();
$numIdBloco = $objSeiRN->gerarBloco($objEntradaGerarBlocoAPI);

```

## gerarProcedimento

Entrada
<p>Instância do objeto EntradaGerarProcedimentoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• Procedimento (ProcedimentoAPI) <ul style="list-style-type: none"> <li>○ IdTipoProcedimento</li> <li>○ NumeroProtocolo e DataAutuacao (opcionais)</li> <li>○ Especificacao</li> <li>○ Assuntos (array:AssuntoAPI preenchido comCodigoEstruturado, também adicionará automaticamente os assuntos sugeridos para o tipo de processo),</li> <li>○ Interessados (array:InteressadoAPI preenchido com Sigla/Nome),</li> <li>○ Observacao</li> <li>○ NivelAcesso (se não informado assumirá o nível padrão especificado para o tipo de processo)</li> <li>○ IdHipoteseLegal</li> </ul> </li> <li>• Documentos - array:DocumentoAPI <ul style="list-style-type: none"> <li>○ Tipo</li> <li>○ IdSerie</li> <li>○ Numero</li> <li>○ Data</li> <li>○ Descricao</li> <li>○ IdTipoConferencia</li> <li>○ Remetente (RemetenteAPI preenchido com Sigla/Nome)</li> <li>○ Interessados (array:InteressadoAPI preenchidos com Sigla/Nome)</li> <li>○ Destinatarios (array:DestinatarioAPI preenchidos com Sigla/Nome)</li> <li>○ Observacao</li> <li>○ NivelAcesso (se não informado assumirá o nível padrão especificado para o tipo de processo)</li> <li>○ IdHipoteseLegal</li> <li>○ NomeArquivo (obrigatório para documentos externos)</li> <li>○ IdArquivo ou Conteudo (Base64) ou ConteudoMTOM (Binário)</li> <li>○ Campos (array:CampoAPI preenchidos com Nome/Valor, somente para formulários)</li> </ul> </li> <li>• ProcedimentosRelacionados - array com os IDs dos processos</li> <li>• UnidadesEnvio - array com os IDs das unidades</li> <li>• SinManterAbertoUnidade</li> <li>• SinEnviarEmailNotificacao</li> <li>• DataRetornoProgramado</li> <li>• DiasRetornoProgramado</li> </ul>

- SinDiasUteisRetornoProgramado
- IdMarcador
- TextoMarcador

#### Saída

Instância do objeto SaidaGerarProcedimentoAPI preenchida com:

- IdProcedimento
- ProcedimentoFormatado
- LinkAcesso (link não assinado para montar a árvore de processo)
- RetornoInclusaoDocumentos - array:SaidaIncluirDocumentoAPI
  - IdDocumento
  - DocumentoFormatado
  - LinkAcesso (link não assinado para montar a árvore de processo posicionando no documento)

#### Exemplo

```
//Gera um processo
$objProcedimentoAPI = new ProcedimentoAPI();
$objProcedimentoAPI->setIdTipoProcedimento(100000312);
$objProcedimentoAPI->setEspecificacao('Exemplo ABC');

$objEntradaGerarProcedimentoAPI = new EntradaGerarProcedimentoAPI();
$objEntradaGerarProcedimentoAPI->setProcedimento($objProcedimentoAPI);

$objSeiRN = new SeiRN();
$objSeiRN->gerarProcedimento($objEntradaGerarProcedimentoAPI);

//Gera processo com 2 documentos enviando para 2 unidades mantendo aberto na unidade
atual
$objProcedimentoAPI = new ProcedimentoAPI();
$objProcedimentoAPI->setIdTipoProcedimento(100000312);
$objProcedimentoAPI->setEspecificacao('Exemplo ABC com 2 documentos');

$objDocumentoAPIGerado = new DocumentoAPI();
$objDocumentoAPIGerado->setTipo('G');
$objDocumentoAPIGerado->setIdSerie(371);
$objDocumentoAPIGerado->setConteudo(base64_encode('Texto do documento interno'));

$objDocumentoAPIExterno = new DocumentoAPI();
$objDocumentoAPIExterno->setTipo('R');
$objDocumentoAPIExterno->setIdSerie(293);
$objDocumentoAPIExterno->setData('26/03/2014');
$objDocumentoAPIExterno->setNomeArquivo('exemplo.pdf');
$objDocumentoAPIExterno->setConteudo(base64_encode(file_get_contents(DIR_SEI_TEMP.'/exemplo.pdf')));

$objEntradaGerarProcedimentoAPI = new EntradaGerarProcedimentoAPI();
$objEntradaGerarProcedimentoAPI->setProcedimento($objProcedimentoAPI);
$objEntradaGerarProcedimentoAPI->setDocumentos(array($objDocumentoAPIGerado,
$objDocumentoAPIExterno));
$objEntradaGerarProcedimentoAPI->setUnidadesEnvio(array(100000555,100000644));
$objEntradaGerarProcedimentoAPI->setSinManterAbertoUnidade('S');

$objSeiRN = new SeiRN();
$objSeiRN->gerarProcedimento($objEntradaGerarProcedimentoAPI);
```

### incluirDocumento

#### Entrada

Instância do objeto DocumentoAPI preenchida com:

- Tipo
- IdProcedimento ou ProtocoloProcedimento
- IdSerie
- Numero
- Data
- Descricao
- IdTipoConferencia
- Remetente (RemetenteAPI preenchido com Sigla/Nome)
- Interessados (array:InteressadoAPI preenchidos com Sigla/Nome)
- Destinatarios (array:DestinatarioAPI preenchidos com Sigla/Nome)
- Observacao
- NivelAcesso (se não informado assumirá o nível padrão especificado para o tipo de processo)
- IdHipoteseLegal
- NomeArquivo (obrigatório para documentos externos)
- IdArquivo ou Conteudo (Base64) ou ConteudoMTOM (Binário)
- Campos (array:CampoAPI preenchidos com Nome/Valor, somente para formulários)

#### Saída

Instância do objeto SaidaIncluirDocumentoAPI preenchida com:

- IdDocumento
- DocumentoFormatado
- LinkAcesso (link não assinado para montar a árvore de processo posicionando no documento)

#### Exemplo

```
//Incluir documento interno

$objDocumentoAPI = new DocumentoAPI();
//Se o ID do processo é conhecido utilizar setIdProcedimento no lugar de
setProtocoloProcedimento
//evitando uma consulta ao banco
$objDocumentoAPI->setProtocoloProcedimento('0000488-03.2016.4.04.8000');
//$objDocumentoAPI->setIdProcedimento();
$objDocumentoAPI->setTipo('G');
$objDocumentoAPI->setIdSerie(371);
$objDocumentoAPI->setConteudo(base64_encode('Texto do documento interno'));

$objSeiRN = new SeiRN();
$objSeiRN->incluirDocumento($objDocumentoAPI);

//Incluir documento externo

$objDocumentoAPI = new DocumentoAPI();
//Se o ID do processo é conhecido utilizar setIdProcedimento no lugar de
setProtocoloProcedimento
//evitando uma consulta ao banco
$objDocumentoAPI->setProtocoloProcedimento('0000488-03.2016.4.04.8000');
//$objDocumentoAPI->setIdProcedimento();
$objDocumentoAPI->setTipo('R');
$objDocumentoAPI->setIdSerie(293);
$objDocumentoAPI->setData('26/03/2014');
$objDocumentoAPI->setNomeArquivo('exemplo.pdf');
$objDocumentoAPI->setConteudo(base64_encode(file_get_contents(DIR_SEI_TEMP.'/exemplo.pdf')));
```

```
$objSeiRN = new SeiRN();
$objSeiRN->incluirDocumento($objDocumentoAPI);
```

## incluirDocumentoBloco

Entrada
<p>Instância do objeto EntradaIncluirDocumentoBlocoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdBloco</li> <li>• IdDocumento ou ProtocoloDocumento</li> <li>• Anotacao</li> </ul>
Exemplo
<pre>\$objEntradaIncluirDocumentoBlocoAPI = new EntradaIncluirDocumentoBlocoAPI(); \$objEntradaIncluirDocumentoBlocoAPI-&gt;setIdBloco(932); // \$objEntradaIncluirDocumentoBlocoAPI-&gt;setIdDocumento(); \$objEntradaIncluirDocumentoBlocoAPI-&gt;setProtocoloDocumento('0011323'); \$objEntradaIncluirDocumentoBlocoAPI-&gt;setAnotacao('Exemplo anotação em documento do bloco módulo ABC.');</pre> <pre>\$objSeiRN = new SeiRN(); \$objSeiRN-&gt;incluirDocumentoBloco(\$objEntradaIncluirDocumentoBlocoAPI);</pre>

## incluirProcessoBloco

Entrada
<p>Instância do objeto EntradaIncluirProcessoBlocoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdBloco</li> <li>• IdProcedimento ou ProtocoloProcedimento</li> <li>• Anotacao</li> </ul>
Exemplo
<pre>\$objEntradaIncluirProcessoBlocoAPI = new EntradaIncluirProcessoBlocoAPI(); \$objEntradaIncluirProcessoBlocoAPI-&gt;setIdBloco(777); // \$objEntradaIncluirProcessoBlocoAPI-&gt;setIdProcedimento(); \$objEntradaIncluirProcessoBlocoAPI-&gt;setProtocoloProcedimento('0001570- 06.2015.4.04.8000'); \$objEntradaIncluirProcessoBlocoAPI-&gt;setAnotacao('Exemplo anotação em processo do bloco módulo ABC.');</pre> <pre>\$objSeiRN = new SeiRN(); \$objSeiRN-&gt;incluirProcessoBloco(\$objEntradaIncluirProcessoBlocoAPI);</pre>

## lançarAndamento

Entrada
<p>Instância do objeto EntradaLancarAndamentoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdProcedimento ou ProtocoloProcedimento</li> <li>• IdTarefa ou IdTarefaModulo</li> <li>• Atributos (array:AtributoAndamentoAPI) <ul style="list-style-type: none"> <li>○ Nome</li> <li>○ Valor</li> <li>○ IdOrigem</li> </ul> </li> </ul>
Saída



Instância do objeto AndamentoAPI preenchida com:

- IdAndamento
- IdTarefa
- IdTarefaModulo
- Descricao
- DataHora
- Usuario (UsuarioAPI)
  - IdUsuario
  - Sigla
  - Nome
- Unidade (UnidadeAPI)
  - IdUnidade
  - Sigla
  - Descricao
- Atributos (array: AtributoAndamentoAPI)
  - Nome
  - Valor
  - IdOrigem

#### Observações

Se informado o atributo IdTarefa do objeto de entrada então ele deve ser um número maior ou igual a 1000 (identificadores abaixo deste valor são reservados do SEI) ou então 65 que equivale a tarefa de atualização de andamento (neste caso informar um atributo com Nome="DESCRICAO" e Valor="texto do andamento").

#### Exemplo

```
$objEntradaLancarAndamentoAPI = new EntradaLancarAndamentoAPI();
$objEntradaLancarAndamentoAPI->setIdProcedimento(10000000012057);
$objEntradaLancarAndamentoAPI->setIdTarefaModulo('MD_ABC_AUDIENCIA_REALIZADA');

$arrObjAtributoAndamentoAPI = array();

$objAtributoAndamentoAPI = new AtributoAndamentoAPI();
$objAtributoAndamentoAPI->setNome('PREDIO');
$objAtributoAndamentoAPI->setValor('101');
$objAtributoAndamentoAPI->setIdOrigem(54); //ID do prédio, pode ser null
$arrObjAtributoAndamentoAPI[] = $objAtributoAndamentoAPI;

$objAtributoAndamentoAPI = new AtributoAndamentoAPI();
$objAtributoAndamentoAPI->setNome('SALA');
$objAtributoAndamentoAPI->setValor('3A');
$objAtributoAndamentoAPI->setIdOrigem(9334); //ID da sala, pode ser null
$arrObjAtributoAndamentoAPI[] = $objAtributoAndamentoAPI;

$objEntradaLancarAndamentoAPI->setAtributos($arrObjAtributoAndamentoAPI);

$objSeiRN = new SeiRN();
$objAndamentoAPI = $objSeiRN->lancarAndamento($objEntradaLancarAndamentoAPI);
```

AndamentoAPI Object

```
(
    [IdAndamento] => 106159
    [IdTarefa] => 1016
    [IdTarefaModulo] => MD_ABC_AUDIENCIA_REALIZADA
    [Descricao] => Audiência agendada no prédio 101 (sala 3A)
    [DataHora] => 05/10/2016 16:12:40
    [Usuario] => UsuarioAPI Object
)
```

```

        [IdUsuario] => 100002382
        [Sigla] => abc
        [Nome] => Aaaaaa Bbbbbb Ccccc
    )

    [Unidade] => UnidadeAPI Object
    (
        [IdUnidade] => 110000001
        [Sigla] => TESTE
        [Descricao] => Unidade de Teste
    )

    [Atributos] => Array
    (
        [0] => AtributoAndamentoAPI Object
        (
            [Nome] => PREDIO
            [Valor] => 101
            [IdOrigem] => 54
        )

        [1] => AtributoAndamentoAPI Object
        (
            [Nome] => SALA
            [Valor] => 3A
            [IdOrigem] => 9334
        )
    )
)

```

## listarAndamentos

Entrada
<p>Instância do objeto EntradaListarAndamentosAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdProcedimento ou ProtocoloProcedimento</li> <li>• SinRetornarAtributos</li> <li>• Andamentos</li> <li>• Tarefas</li> <li>• TarefasModulos</li> </ul>
Saída
<p>Lista de instâncias do objeto AndamentoAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdAndamento</li> <li>• IdTarefa</li> <li>• IdTarefaModulo</li> <li>• Descricao</li> <li>• DataHora</li> <li>• Usuario (UsuarioAPI) <ul style="list-style-type: none"> <li>○ IdUsuario</li> <li>○ Sigla</li> <li>○ Nome</li> </ul> </li> <li>• Unidade (UnidadeAPI) <ul style="list-style-type: none"> <li>○ IdUnidade</li> <li>○ Sigla</li> <li>○ Descricao</li> </ul> </li> <li>• Atributos (array: AtributoAndamentoAPI) <ul style="list-style-type: none"> <li>○ Nome</li> <li>○ Valor</li> </ul> </li> </ul>

o IdOrigem	
Observações	
No objeto de entrada é necessário informar pelo menos um dos atributos Andamentos, Tarefas ou TarefasModulos. No atributo Tarefas é possível filtrar por qualquer tarefa do sistema (verificar os valores de tarefas internas nas constantes existentes no arquivo TarefaRN.php).	
Exemplo	
<pre> \$objEntradaListarAndamentosAPI = new EntradaListarAndamentosAPI(); \$objEntradaListarAndamentosAPI-&gt;setIdProcedimento(10000000012057); // \$objEntradaListarAndamentosAPI-&gt;setProtocoloProcedimento(); \$objEntradaListarAndamentosAPI-&gt;setTarefasModulos(array('MD_ABC_AUDIENCIA_REALIZADA')); \$objEntradaListarAndamentosAPI-&gt;setSinRetornarAtributos('S');  \$objSeiRN = new SeiRN(); \$arrObjAndamentoAPI = \$objSeiRN-&gt;listarAndamentos(\$objEntradaListarAndamentosAPI);  Array (     [0] =&gt;  AndamentoAPI Object (     [IdAndamento] =&gt; 106159     [IdTarefa] =&gt; 1016     [IdTarefaModulo] =&gt; MD_ABC_AUDIENCIA_REALIZADA     [Descricao] =&gt; Audiência agendada no prédio 101 (sala 3A)     [DataHora] =&gt; 05/10/2016 16:12:40     [Usuario] =&gt; UsuarioAPI Object         (             [IdUsuario] =&gt; 100002382             [Sigla] =&gt; abc             [Nome] =&gt; Aaaaaa Bbbbbb Cccccc         )      [Unidade] =&gt; UnidadeAPI Object         (             [IdUnidade] =&gt; 110000001             [Sigla] =&gt; TESTE             [Descricao] =&gt; Unidade de Teste         )      [Atributos] =&gt; Array         (             [0] =&gt; AtributoAndamentoAPI Object                 (                     [Nome] =&gt; PREDIO                     [Valor] =&gt; 101                     [IdOrigem] =&gt; 54                 )              [1] =&gt; AtributoAndamentoAPI Object                 (                     [Nome] =&gt; SALA                     [Valor] =&gt; 3A                     [IdOrigem] =&gt; 9334                 )         )     ) ) </pre>	

## listarAndamentosMarcadores

Entrada
<p>Conjunto de Instâncias do objeto EntradaListarAndamentosMarcadoresAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdProcedimento ou ProtocoloProcedimento</li> <li>• Marcadores</li> </ul>
Saída
<p>Conjunto de Instâncias do objeto AndamentoMarcadorAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdAndamentoMarcador</li> <li>• Texto</li> <li>• DataHora</li> <li>• Usuario (UsuarioAPI) <ul style="list-style-type: none"> <li>○ IdUsuario</li> <li>○ Sigla</li> <li>○ Nome</li> </ul> </li> <li>• Marcador (MarcadorAPI) <ul style="list-style-type: none"> <li>○ IdMarcador</li> <li>○ Nome</li> <li>○ Icone</li> <li>○ SinAtivo</li> </ul> </li> <li>• DataHora</li> </ul>
Exemplo
<pre> \$objEntradaListarAndamentosMarcadoresAPI = new EntradaListarAndamentosMarcadoresAPI(); //\$objEntradaListarAndamentosMarcadoresAPI-&gt;setIdProcedimento( ); \$objEntradaListarAndamentosMarcadoresAPI-&gt;setProtocoloProcedimento('0000488-03.2016.4.04.8000'); //\$objEntradaListarAndamentosMarcadoresAPI-&gt;setMarcadores(\$Marcadores);  \$objSeiRN = new SeiRN(); \$arrObjAndamentoMarcadorAPI = \$objSeiRN-&gt;listarAndamentosMarcadores(\$objEntradaListarAndamentosMarcadoresAPI);  Array (     [0] =&gt; AndamentoMarcadorAPI Object         (             [IdAndamentoMarcador] =&gt; 99             [Texto] =&gt; Texto do andamento             [DataHora] =&gt; 07/10/2016 11:30:15             [Usuario] =&gt; UsuarioAPI Object                 (                     [IdUsuario] =&gt; 56555                     [Sigla] =&gt; abc                     [Nome] =&gt; Aaaaaa Bbbbbb Cccccc                 )              [Marcador] =&gt; MarcadorAPI Object                 (                     [IdMarcador] =&gt; 12                     [Nome] =&gt; Teste 4                     [Icone] =&gt; data:image/png;base64,iVBORw0KGgoAAAANSUHEUgAAABAAAAQCAIAAACQkWg2AAAABnRSTlMA/wB/ACdeZ VobAAAACXBIWXMAAA7EAAAOxAGVKw4bAAAAgklEQVR4nJWSyxHAIaEkbGXNEFulkIfacRSvGFfycGMMUqIctL1 LR8Hdx4brIT/esghjiI1lg05xJ32TpQseoWWLhAAVAV/c1Pi9oo2DcMw3qZHBQ1aDZykJUsZph/apqcMLX0bKHH 9b5t+KqiekX61lHlUGrrVoMQUs9qk251vS8g0UT9daGY9AAAAABJRUS5ErkJggg==                     [SinAtivo] =&gt; S                 )         ) </pre>

```

    )

    [1] => AndamentoMarcadorAPI Object
    (
        [IdAndamentoMarcador] => 98
        [Texto] => Texto do andamento com marcador removido
        [DataHora] => 07/10/2016 11:29:56
        [Usuario] => UsuarioAPI Object
        (
            [IdUsuario] => 56555
            [Sigla] => abc
            [Nome] => Aaaaaa Bbbbbb Cccccc
        )

        [Marcador] =>
    )

    [2] => AndamentoMarcadorAPI Object
    (
        [IdAndamentoMarcador] => 97
        [Texto] => Texto do andamento
        [DataHora] => 06/10/2016 18:48:37
        [Usuario] => UsuarioAPI Object
        (
            [IdUsuario] => 37322
            [Sigla] => def
            [Nome] => Dddddd Eeeeeee Fffffff

            [Marcador] => MarcadorAPI Object
            (
                [IdMarcador] => 8
                [Nome] => Teste 3
                [Icone] =>
data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAABAAAAAQCAIAAAACQkWg2AAAABnRSTlMA/wB/ACdeZ
VobAAAAACXBIWXMAAA7EAAAOxAGVKw4bAAAAZklEQVR4nGP8X6/OQApGwSWRuaUEU3C6Tw8T8apx2oCserpPD5og
ugl4zMaigaBqFCdhVY0pyES82QgNRKqGBAD2YMWlmlgNcNVQDch8/KoRNUdSgynOhEcOqylMuFTgspORlOQNAPr
/JbzEDsOcAAAAAE1FTkSuQmCC
                [SinAtivo] => S
            )
        )

    [3] => AndamentoMarcadorAPI Object
    (
        [IdAndamentoMarcador] => 96
        [Texto] => Texto do andamento
        [DataHora] => 06/10/2016 18:48:26
        [Usuario] => UsuarioAPI Object
        (
            [IdUsuario] => 56555
            [Sigla] => abc
            [Nome] => Aaaaaa Bbbbbb Cccccc

            [Marcador] => MarcadorAPI Object
            (
                [IdMarcador] => 9
                [Nome] => Marcador1
                [Icone] =>
data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAABAAAAAQCAIAAAACQkWg2AAAABnRSTlMA/wB/ACdeZ
VobAAAAACXBIWXMAAA7EAAAOxAGVKw4bAAAAZklEQVR4nGP8X6/OQApGwSVx0HEmpqD9/nQm4lXjtAFZtf3+dDRB
dBvwmIlFA0HVKE7CqhpTkIl4sxEaiFQNCQDswYpLNbEa4KqhGpD5+FUjbMC1B1OcCY8cVlOYcKnAZScjqckbAEm
jJZ7CtTvqAAAAAE1FTkSuQmCC
                [SinAtivo] => S
            )
        )
    )

```

```

    )
)

```

## listarCargos

Entrada
<p>Instância do objeto EntradaListarCargosAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdCargo</li> </ul>
Saída
<p>Conjunto de Instâncias do objeto CargoAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdCargo</li> <li>• ExpressaoCargo</li> <li>• ExpressaoTratamento</li> <li>• ExpressaoVocativo</li> </ul>
Exemplo
<pre> \$objEntradaListarCargosAPI = new EntradaListarCargosAPI(); \$objEntradaListarCargosAPI-&gt;setIdCargo(null); //listar todos  \$objSeiRN = new SeiRN(); \$arrObjCargoAPI = \$objSeiRN-&gt;listarCargos(\$objEntradaListarCargosAPI);  Array (     [0] =&gt; CargoAPI Object         (             [IdCargo] =&gt; 100001843             [ExpressaoCargo] =&gt; Almirante da Marinha do Brasil             [ExpressaoTratamento] =&gt; A Sua Excelência o Senhor             [ExpressaoVocativo] =&gt; Senhor Almirante         )      [1] =&gt; CargoAPI Object         (             [IdCargo] =&gt; 100001845             [ExpressaoCargo] =&gt; Chefe de Gabinete             [ExpressaoTratamento] =&gt; Ao Senhor             [ExpressaoVocativo] =&gt; Senhor Chefe de Gabinete         )      [2] =&gt; CargoAPI Object         (             [IdCargo] =&gt; 100001846             [ExpressaoCargo] =&gt; Cidadão             [ExpressaoTratamento] =&gt; Ao Senhor             [ExpressaoVocativo] =&gt; Senhor      [3] =&gt; CargoAPI Object         (             [IdCargo] =&gt; 100001847             [ExpressaoCargo] =&gt; Cônsul             [ExpressaoTratamento] =&gt; A Sua Excelência o Senhor             [ExpressaoVocativo] =&gt; Senhor Cônsul             ...         ) ) </pre>

## listarCidades

Entrada
Conjunto de Instâncias do objeto EntradaListarCidadesAPI preenchidas com: <ul style="list-style-type: none"><li>• IdPais</li><li>• IdEstado</li></ul>
Saída
Conjunto de Instâncias do objeto EstadoAPI preenchidas com: <ul style="list-style-type: none"><li>• IdCidade</li><li>• IdEstado</li><li>• IdPais</li><li>• Nome</li><li>•CodigoIbge</li><li>• SinCapital</li><li>• Latitude</li><li>• Longitude</li></ul>
Exemplo
<pre>\$objEntradaListarCidadesAPI = new EntradaListarCidadesAPI(); \$objEntradaListarCidadesAPI-&gt;setIdPais(76); //Brasil \$objEntradaListarCidadesAPI-&gt;setIdEstado(2); //Acre  \$objSeiRN = new SeiRN(); \$arrObjCidadeAPI = \$objSeiRN-&gt;listarCidades(\$objEntradaListarCidadesAPI);  Array (     [0] =&gt; CidadeAPI Object         (             [IdCidade] =&gt; 53             [IdEstado] =&gt; 2             [IdPais] =&gt; 76             [Nome] =&gt; Acrelândia             [CodigoIbge] =&gt; 1200013             [SinCapital] =&gt; N             [Latitude] =&gt; -10,073794             [Longitude] =&gt; -67,052317         )      [1] =&gt; CidadeAPI Object         (             [IdCidade] =&gt; 54             [IdEstado] =&gt; 2             [IdPais] =&gt; 76             [Nome] =&gt; Assis Brasil             [CodigoIbge] =&gt; 1200054             [SinCapital] =&gt; N             [Latitude] =&gt; -10,942866             [Longitude] =&gt; -69,563459         )      [2] =&gt; CidadeAPI Object         (             [IdCidade] =&gt; 55             [IdEstado] =&gt; 2             [IdPais] =&gt; 76             [Nome] =&gt; Brasília             [CodigoIbge] =&gt; 1200104             [SinCapital] =&gt; N             [Latitude] =&gt; -11,016411</pre>

```

        [Longitude] => -68,747943
    )
    ....
)

```

## listarContatos

Entrada
<p>Conjunto de Instâncias do objeto EntradaListarContatosAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdTipoContato (obrigatório)</li> <li>• PaginaRegistros (opcional 1 a 1000 com valor padrão 1)</li> <li>• PaginaAtual (opcional começando em 1)</li> <li>• Sigla</li> <li>• Nome</li> <li>• Cpf</li> <li>• Cnpj</li> <li>• Matricula</li> </ul>
Saída
<p>Conjunto de Instâncias do objeto ContatoAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdContato</li> <li>• IdTipoContato</li> <li>• NomeTipoContato</li> <li>• Sigla</li> <li>• Nome</li> <li>• StaNatureza</li> <li>• IdContatoAssociado</li> <li>• NomeContatoAssociado</li> <li>• SinEnderecoAssociado</li> <li>• EnderecoAssociado</li> <li>• ComplementoAssociado</li> <li>• BairroAssociado</li> <li>• IdCidadeAssociado</li> <li>• NomeCidadeAssociado</li> <li>• IdEstadoAssociado</li> <li>• SiglaEstadoAssociado</li> <li>• IdPaisAssociado</li> <li>• NomePaisAssociado</li> <li>• CepAssociado</li> <li>• Endereco</li> <li>• Complemento</li> <li>• Bairro</li> <li>• IdCidade</li> <li>• NomeCidade</li> <li>• IdEstado</li> <li>• SiglaEstado</li> <li>• IdPais</li> <li>• NomePais</li> <li>• Cep</li> </ul>



- StaGenero
- IdCargo
- ExpressaoCargo
- ExpressaoTratamento
- ExpressaoVocativo
- Cpf
- Cnpj
- Rg
- OrgaoExpedidor
- Matricula
- MatriculaOab
- TelefoneFixo
- TelefoneCelular
- DataNascimento
- Email
- SitioInternet
- Observacao
- SinAtivo

### Exemplo

```
//Ler todos os contatos de um tipo em grupos de 100 registros
$numPaginaAtual = 1;
$objEntradaListarContatosAPI = new EntradaListarContatosAPI();
$objEntradaListarContatosAPI->setIdTipoContato(1);
$objEntradaListarContatosAPI->setPaginaRegistros(100);
$objEntradaListarContatosAPI->setPaginaAtual($numPaginaAtual++);

$objSeiRN = new SeiRN();
$arrObjContatoAPI = $objSeiRN->listarContatos($objEntradaListarContatosAPI);

while(count($arrObjContatoAPI)){

    //processar contatos

    $objEntradaListarContatosAPI->setPaginaAtual($numPaginaAtual++);
    $arrObjContatoAPI = $objSeiRN->listarContatos($objEntradaListarContatosAPI);
};

$objEntradaListarContatosAPI = new EntradaListarContatosAPI();
$objEntradaListarContatosAPI->setIdTipoContato(1);
$objEntradaListarContatosAPI->setCpf(11572070730);

$objSeiRN = new SeiRN();
$arrObjContatoAPI = $objSeiRN->listarContatos($objEntradaListarContatosAPI);

Array
(
    [0] => ContatoAPI Object
        (
            [IdContato] => 100003924
            [IdTipoContato] => 1
            [NomeTipoContato] => Autoridades
            [Sigla] => def
            [Nome] => Dddddd Eeeee Ffffff
            [StaNatureza] => F
            [IdContatoAssociado] =>
            [NomeContatoAssociado] =>
            [SinEnderecoAssociado] =>
        )
)
```

```

[EnderecoAssociado] =>
[ComplementoAssociado] =>
[BairroAssociado] =>
[IdCidadeAssociado] =>
[NomeCidadeAssociado] =>
[IdEstadoAssociado] =>
[SiglaEstadoAssociado] =>
[IdPaisAssociado] =>
[NomePaisAssociado] =>
[CepAssociado] =>
[Endereco] => Rua Otávio Francisco Caruso da Rocha, 2000
[Complemento] =>
[Bairro] => Praia de Belas
[IdCidade] => 4927
[NomeCidade] => Porto Alegre
[IdEstado] => 23
[SiglaEstado] => RS
[IdPais] => 76
[NomePais] => Brasil
[Cep] => 90010-395
[StaGenero] => M
[IdCargo] => 100001865
[ExpressaoCargo] => Diretor
[ExpressaoTratamento] => Ao Senhor
[ExpressaoVocativo] => Senhor Diretor
[Cpf] => 11572070730
[Cnpj] =>
[Rg] => 56584632
[OrgaoExpedidor] => SSP/RS
[Matricula] => 54834
[MatriculaOab] =>
[TelefoneFixo] =>
[TelefoneCelular] => (99)9999-9999
[DataNascimento] => 13/05/1971
[Email] => def@abc.gov.br
[SitioInternet] =>
[Observacao] =>
[SinAtivo] => S
)
)

```

## listarEstados

Entrada
<p>Conjunto de Instâncias do objeto EntradaListarEstadosAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>IdPais</li> </ul>
Saída
<p>Conjunto de Instâncias do objeto EstadoAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>IdEstado</li> <li>IdPais</li> <li>Sigla</li> <li>Nome</li> <li>CodigoIbge</li> </ul>
Exemplo
<pre> \$objEntradaListarEstadosAPI = new EntradaListarEstadosAPI(); \$objEntradaListarEstadosAPI-&gt;setIdPais(76); //Brasil  \$objSeiRN = new SeiRN(); \$arrObjEstadoAPI = \$objSeiRN-&gt;listarEstados(\$objEntradaListarEstadosAPI); </pre>

```

Array
(
    [0] => EstadoAPI Object
        (
            [IdEstado] => 2
            [IdPais] => 76
            [Sigla] => AC
            [Nome] => Acre
            [CodigoIbge] => 12
        )

    [1] => EstadoAPI Object
        (
            [IdEstado] => 14
            [IdPais] => 76
            [Sigla] => AL
            [Nome] => Alagoas
            [CodigoIbge] => 27
        )

    [2] => EstadoAPI Object
        (
            [IdEstado] => 3
            [IdPais] => 76
            [Sigla] => AM
            [Nome] => Amazonas
            [CodigoIbge] => 13
        )

    ...
)

```

## listarExtensoesPermitidas

Entrada
<p>Conjunto de Instâncias do objeto EntradaListarExtensoesPermitidasAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdArquivoExtensao</li> </ul>
Saída
<p>Conjunto de Instâncias do objeto ArquivoExtensaoAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdArquivoExtensao</li> <li>• Extensao</li> <li>• Descricao</li> </ul>
Exemplo
<pre> \$objEntradaListarExtensoesPermitidasAPI = new EntradaListarExtensoesPermitidasAPI(); \$objEntradaListarExtensoesPermitidasAPI-&gt;setIdArquivoExtensao(null); //lista todas  \$objSeiRN = new SeiRN(); \$arrObjArquivoExtensaoAPI = \$objSeiRN-&gt;listarExtensoesPermitidas(\$objEntradaListarExtensoesPermitidasAPI);  Array (     [0] =&gt; ArquivoExtensaoAPI Object         (             [IdArquivoExtensao] =&gt; 1             [Extensao] =&gt; html             [Descricao] =&gt; Linguagem de Marcação de Hipertexto utilizada em páginas da Web         ) ) </pre>

```

[1] => ArquivoExtensaoAPI Object
(
  [IdArquivoExtensao] => 14
  [Extensao] => jpg
  [Descricao] => Imagem
)

[2] => ArquivoExtensaoAPI Object
(
  [IdArquivoExtensao] => 6
  [Extensao] => odt
  [Descricao] => Texto com formatação (padrão aberto)
)

...
)

```

## listarHipotesesLegais

Entrada
<p>Conjunto de Instâncias do objeto EntradaListarHipotesesLegaisAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>NivelAcesso</li> </ul>
Saída
<p>Conjunto de Instâncias do objeto HipoteseLegalAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>IdHipoteseLegal</li> <li>Nome</li> <li>BaseLegal</li> <li>NivelAcesso</li> </ul>
Exemplo
<pre> \$objEntradaListarHipotesesLegaisAPI = new EntradaListarHipotesesLegaisAPI(); \$objEntradaListarHipotesesLegaisAPI-&gt;setNivelAcesso(null); //listar todas  \$objSeiRN = new SeiRN(); \$arrObjHipoteseLegalAPI = \$objSeiRN- &gt;listarHipotesesLegais(\$objEntradaListarHipotesesLegaisAPI);  Array (     [0] =&gt; HipoteseLegalAPI Object         (             [IdHipoteseLegal] =&gt; 1             [Nome] =&gt; Hipótese A             [BaseLegal] =&gt; Lei 1.000             [NivelAcesso] =&gt; 2         )      [1] =&gt; HipoteseLegalAPI Object         (             [IdHipoteseLegal] =&gt; 10             [Nome] =&gt; Hipótese B             [BaseLegal] =&gt; Lei 2.000             [NivelAcesso] =&gt; 1         )      [2] =&gt; HipoteseLegalAPI Object         (             [IdHipoteseLegal] =&gt; 2             [Nome] =&gt; Hipótese C             [BaseLegal] =&gt; Lei 3.000             [NivelAcesso] =&gt; 1         ) </pre>

```

    ...
)

```

## listarMarcadoresUnidade

Saída
<p>Conjunto de Instâncias do objeto MarcadorAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdMarcador</li> <li>• Nome</li> <li>• Icone</li> <li>• SinAtivo</li> </ul>
Exemplo
<pre> \$objSeiRN = new SeiRN(); \$arrObjMarcadorAPI = \$objSeiRN-&gt;listarMarcadoresUnidade();  Array (     [0] =&gt; MarcadorAPI Object         (             [IdMarcador] =&gt; 9             [Nome] =&gt; Marcador A             [Icone] =&gt; data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAABAAAAQCAIAAACQkWg2AAAABnRSTlMA/wB/ACdeZ VobAAAAACXBIWXMAAA7EAAAOxAGVKw4bAAAAZklEQVR4nGP8X6/OQApwSVx0HEmpqD9/nQm4lXjtAFZtf3+dDRB dBvwmI1FA0HVKE7CqhpTkI14sxEaiFQNCQDswYpLNbEa4KqhGpD5+FUjbmC1B1OcCY8cVlOYcKnAZScjqckbAEm jJZ7CtTvqAAAAAE1FTkSuQmCC             [SinAtivo] =&gt; S         )      [1] =&gt; MarcadorAPI Object         (             [IdMarcador] =&gt; 10             [Nome] =&gt; Marcador B             [Icone] =&gt; data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAABAAAAQCAIAAACQkWg2AAAABnRSTlMA/wB/ACdeZ VobAAAAACXBIWXMAAA7EAAAOxAGVKw4bAAAAhU1EQVR4nJWSvRGAIAYFA8culi4QO0bJHi7CKHswABNpgYcYYpRU 8PhefriYY19gJtzBQ/ZhFDGSbMg+4LpxsSS5Qk9XCACaYj9zy6T+anUahmGcTo+KVWgx7E8611SH4UPr9C9DT18 GjNT+W6fvCqJnpB8tMY9IAIsNjJQhtbPYpJld7xPPmkUafFnHsgAAAABJRU5ErkJggg==             [SinAtivo] =&gt; S         )      [2] =&gt; MarcadorAPI Object         (             [IdMarcador] =&gt; 4             [Nome] =&gt; Marcador C             [Icone] =&gt; data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAABAAAAQCAIAAACQkWg2AAAABnRSTlMA/wB/ACdeZ VobAAAAACXBIWXMAAA7EAAAOxAGVKw4bAAAAak1EQVR4nGP8X6/OQApwSXxbu5fTEGhZGYm4lXjtAFZtVAyM5og ug14zMaigaBqFCdhVY0pyES82QgNRKqGBAD2YMWlmlgNcNVQDch8/KoRNUdSgynOhEcOqylMuFTgshM9LeH3DwM DAwBgvyGtI+3dAAAAABJRU5ErkJggg==             [SinAtivo] =&gt; S         )     ... ) </pre>

## listarPaíses

Saída
-------

Conjunto de Instâncias do objeto PaisAPI preenchidas com:

- IdPais
- Nome

#### Exemplo

```
$objSeiRN = new SeiRN();  
$arrObjPaisAPI = $objSeiRN->listarPaises();
```

Array

```
(  
  [0] => PaisAPI Object  
    (  
      [IdPais] => 4  
      [Nome] => Afeganistão  
    )  
  [1] => PaisAPI Object  
    (  
      [IdPais] => 710  
      [Nome] => África do Sul  
    )  
  [2] => PaisAPI Object  
    (  
      [IdPais] => 8  
      [Nome] => Albânia  
    )  
  ...  
)
```

## listarSeries

#### Saída

Conjunto de Instâncias do objeto SerieAPI preenchidas com:

- IdSerie
- Nome
- Aplicabilidade

#### Exemplo

```
$objSeiRN = new SeiRN();  
$arrObjSerieAPI = $objSeiRN->listarSeries();
```

Array

```
(  
  [0] => SerieAPI Object  
    (  
      [IdSerie] => 349  
      [Nome] => Abono de Faltas  
      [Aplicabilidade] => T  
    )  
  [1] => SerieAPI Object  
    (  
      [IdSerie] => 475  
      [Nome] => Acórdão  
      [Aplicabilidade] => I  
    )  
)
```

```
[2] => SerieAPI Object
(
  [IdSerie] => 293
  [Nome] => Anexo
  [Aplicabilidade] => E
)

...
)
```

## listarTiposConferencia

Saída
<p>Conjunto de Instâncias do objeto TipoConferenciaAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdTipoConferencia</li> <li>• Descricao</li> </ul>
Exemplo
<pre>\$objSeiRN = new SeiRN(); \$arrObjTipoConferenciaAPI = \$objSeiRN-&gt;listarTiposConferencia();  Array (     [0] =&gt; TipoConferenciaAPI Object         (             [IdTipoConferencia] =&gt; 3             [Descricao] =&gt; Cópia autenticada administrativamente         )      [1] =&gt; TipoConferenciaAPI Object         (             [IdTipoConferencia] =&gt; 2             [Descricao] =&gt; Cópia autenticada por cartório         )      [2] =&gt; TipoConferenciaAPI Object         (             [IdTipoConferencia] =&gt; 4             [Descricao] =&gt; Cópia simples         )      [3] =&gt; TipoConferenciaAPI Object         (             [IdTipoConferencia] =&gt; 1             [Descricao] =&gt; Documento Original         ) )</pre>

## listarTiposProcedimento

Saída
<p>Conjunto de Instâncias do objeto TipoProcedimentoAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdTipoProcedimento</li> <li>• Nome</li> </ul>
Exemplo
<pre>\$objSeiRN = new SeiRN(); \$arrObjTipoProcedimentoAPI = \$objSeiRN-&gt;listarTiposProcedimento();</pre>

```

Array
(
    [0] => TipoProcedimentoAPI Object
        (
            [IdTipoProcedimento] => 100000311
            [Nome] => Abono de Faltas
        )

    [1] => TipoProcedimentoAPI Object
        (
            [IdTipoProcedimento] => 100000334
            [Nome] => Abono Permanência
        )

    [2] => TipoProcedimentoAPI Object
        (
            [IdTipoProcedimento] => 100000258
            [Nome] => Afastamentos
        )

    ...
)

```

## listarUnidades

Entrada
<p>Conjunto de Instâncias do objeto EntradaListarUnidadesAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdUnidade</li> <li>• IdOrgao</li> <li>• PalavrasPesquisa</li> </ul>
Saída
<p>Conjunto de Instâncias do objeto UnidadeAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>• IdUnidade</li> <li>• Sigla</li> <li>• Descricao</li> </ul>
Exemplo
<pre> \$objEntradaListarUnidadesAPI = new EntradaListarUnidadesAPI(); \$objEntradaListarUnidadesAPI-&gt;setIdOrgao(2); \$objEntradaListarUnidadesAPI-&gt;setPalavrasPesquisa('nucleo');  \$objSeiRN = new SeiRN(); \$arrObjUnidadeAPI = \$objSeiRN-&gt;listarUnidades(\$objEntradaListarUnidadesAPI);  Array (     [0] =&gt; UnidadeAPI Object         (             [IdUnidade] =&gt; 110000274             [Sigla] =&gt; RSPOANCI             [Descricao] =&gt; Núcleo de Controle Interno         )      [1] =&gt; UnidadeAPI Object         (             [IdUnidade] =&gt; 110000290             [Sigla] =&gt; RSPOANDOC             [Descricao] =&gt; Núcleo de Documentação         ) </pre>



```
[2] => UnidadeAPI Object
  (
    [IdUnidade] => 110000276
    [Sigla] => RSPOANGF
    [Descricao] => Núcleo de Gestão Funcional
  )
  ...
)
```

## listarUsuarios

Entrada
<p>Conjunto de Instâncias do objeto EntradaListarUsuariosAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>IdUsuario</li> </ul>
Saída
<p>Conjunto de Instâncias do objeto UsuarioAPI preenchidas com:</p> <ul style="list-style-type: none"> <li>IdUsuario</li> <li>Sigla</li> <li>Nome</li> </ul>
Exemplo
<pre>\$objEntradaListarUsuariosAPI = new EntradaListarUsuariosAPI(); \$objEntradaListarUsuariosAPI-&gt;setIdUsuario(null);  \$objSeiRN = new SeiRN(); \$arrObjUsuarioAPI = \$objSeiRN-&gt;listarUsuarios(\$objEntradaListarUsuariosAPI);  Array (     [0] =&gt; UsuarioAPI Object         (             [IdUsuario] =&gt; 100008510             [Sigla] =&gt; abc             [Nome] =&gt; Aaaaaa Bbbbbb Cccccc         )      [1] =&gt; UsuarioAPI Object         (             [IdUsuario] =&gt; 100011470             [Sigla] =&gt; def             [Nome] =&gt; Dddddd Eeeee Ffffff         )      [2] =&gt; UsuarioAPI Object         (             [IdUsuario] =&gt; 100008709             [Sigla] =&gt; ghi             [Nome] =&gt; Gggggg Hhhhhh Iiiiii         )     ... )</pre>

## reabrirProcesso

Entrada
<p>Instância do objeto EntradaReabrirProcessoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>IdProcedimento ou ProtocoloProcedimento</li> </ul>
Exemplo

```

$ObjEntradaReabrirProcessoAPI = new EntradaReabrirProcessoAPI();
//$ObjEntradaReabrirProcessoAPI->setIdProcedimento();
$ObjEntradaReabrirProcessoAPI->setProtocoloProcedimento('0000262-95.2016.4.04.8000');

$ObjSeiRN = new SeiRN();
$ObjSeiRN->reabrirProcesso($ObjEntradaReabrirProcessoAPI);

```

## relacionarProcesso

Entrada
<p>Instância do objeto EntradaRelacionarProcessoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>IdProcedimento1 ou ProtocoloProcedimento1</li> <li>IdProcedimento2 ou ProtocoloProcedimento2</li> </ul>
Exemplo
<pre> \$ObjEntradaRelacionarProcessoAPI = new EntradaRelacionarProcessoAPI(); //\$ObjEntradaRelacionarProcessoAPI-&gt;setIdProcedimento1(); \$ObjEntradaRelacionarProcessoAPI-&gt;setProtocoloProcedimento1('11.1.000000485-4'); //\$ObjEntradaRelacionarProcessoAPI-&gt;setIdProcedimento2(); \$ObjEntradaRelacionarProcessoAPI-&gt;setProtocoloProcedimento2('11.1.000000467-6');  \$ObjSeiRN = new SeiRN(); \$ObjSeiRN-&gt;relacionarProcesso(\$ObjEntradaRelacionarProcessoAPI); </pre>

## removerRelacionamentoProcesso

Entrada
<p>Instância do objeto EntradaRemoverRelacionamentoProcessoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>IdProcedimento1 ou ProtocoloProcedimento1</li> <li>IdProcedimento2 ou ProtocoloProcedimento2</li> </ul>
Exemplo
<pre> \$ObjEntradaRemoverRelacionamentoProcessoAPI = new EntradaRemoverRelacionamentoProcessoAPI(); //\$ObjEntradaRemoverRelacionamentoProcessoAPI-&gt;setIdProcedimento1(); \$ObjEntradaRemoverRelacionamentoProcessoAPI-&gt;setProtocoloProcedimento1('11.1.000000485-4'); //\$ObjEntradaRemoverRelacionamentoProcessoAPI-&gt;setIdProcedimento2(); \$ObjEntradaRemoverRelacionamentoProcessoAPI-&gt;setProtocoloProcedimento2('11.1.000000467-6');  \$ObjSeiRN = new SeiRN(); \$ObjSeiRN-&gt;removerRelacionamentoProcesso(\$ObjEntradaRemoverRelacionamentoProcessoAPI); </pre>

## removerSobrestamentoProcesso

Entrada
<p>Instância do objeto EntradaRemoverSobrestamentoProcessoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>IdProcedimento ou ProtocoloProcedimento</li> </ul>
Exemplo
<pre> \$ObjEntradaRemoverSobrestamentoProcessoAPI = new EntradaRemoverSobrestamentoProcessoAPI(); //\$ObjEntradaRemoverSobrestamentoProcessoAPI-&gt;setIdProcedimento(); \$ObjEntradaRemoverSobrestamentoProcessoAPI-&gt;setProtocoloProcedimento('0000495-63.2014.4.04.8000'); </pre>

```
$objSeiRN = new SeiRN();
$objSeiRN->removerSobrestamentoProcesso($objEntradaRemoverSobrestamentoProcessoAPI);
```

## retirarDocumentoBloco

Entrada
<p>Instância do objeto EntradaRetirarDocumentoBlocoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdBloco</li> <li>• IdDocumento ou ProtocoloDocumento</li> </ul>
Exemplo
<pre>\$objEntradaRetirarDocumentoBlocoAPI = new EntradaRetirarDocumentoBlocoAPI(); \$objEntradaRetirarDocumentoBlocoAPI-&gt;setIdBloco(932); // \$objEntradaRetirarDocumentoBlocoAPI-&gt;setIdDocumento(); \$objEntradaRetirarDocumentoBlocoAPI-&gt;setProtocoloDocumento('0011323');  \$objSeiRN = new SeiRN(); \$objSeiRN-&gt;retirarDocumentoBloco(\$objEntradaRetirarDocumentoBlocoAPI);</pre>

## retirarProcessoBloco

Entrada
<p>Instância do objeto EntradaRetirarProcessoBlocoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdBloco</li> <li>• IdProcedimento ou ProtocoloProcedimento</li> </ul>
Exemplo
<pre>\$objEntradaRetirarProcessoBlocoAPI = new EntradaRetirarProcessoBlocoAPI(); \$objEntradaRetirarProcessoBlocoAPI-&gt;setIdBloco(777); // \$objEntradaRetirarProcessoBlocoAPI-&gt;setIdProcedimento(); \$objEntradaRetirarProcessoBlocoAPI-&gt;setProtocoloProcedimento('0001570-06.2015.4.04.8000');  \$objSeiRN = new SeiRN(); \$objSeiRN-&gt;retirarProcessoBloco(\$objEntradaRetirarProcessoBlocoAPI);</pre>

## sobrestarProcesso

Entrada
<p>Instância do objeto EntradaSobrestarProcessoAPI preenchida com:</p> <ul style="list-style-type: none"> <li>• IdProcedimento ou ProtocoloProcedimento</li> <li>• IdProcedimentoVinculado ou ProtocoloProcedimentoVinculado (opcional)</li> <li>• Motivo</li> </ul>
Exemplo
<pre>\$objEntradaSobrestarProcessoAPI = new EntradaSobrestarProcessoAPI(); // \$objEntradaSobrestarProcessoAPI-&gt;setIdProcedimento(); \$objEntradaSobrestarProcessoAPI-&gt;setProtocoloProcedimento('0000495-63.2014.4.04.8000'); \$objEntradaSobrestarProcessoAPI-&gt;setMotivo('Exemplo sobrestamento de processo módulo ABC');</pre> <pre>\$objSeiRN = new SeiRN(); \$objSeiRN-&gt;sobrestarProcesso(\$objEntradaSobrestarProcessoAPI);</pre>

## 7. InfraPHP

O uso do framework InfraPHP é opcional e não tem como objetivo estabelecer métodos para a realização da análise do módulo que será implementado. Presume-se que as funcionalidades, regras de negócio, protótipos, diagrama E/R e outros artefatos necessários já tenham sido elaborados.

### Visão Geral

A arquitetura propõe a separação em múltiplas camadas denominadas: Cliente, RN e BD. Cada camada possui um conjunto definido de responsabilidades:

#### Cliente

- Interação com o usuário ou outros sistemas
- Validações simples que não necessitem acesso à dados

#### RN

- Controle de transação
- Validação de permissões
- Auditoria
- Validação das regras de negócio

#### BD

- Persistência
- Recuperação de dados

Além disso devem ser respeitadas as seguintes regras:

- Toda a comunicação ocorre somente entre RNs garantindo que sempre as permissões de acesso, regras de negócio e auditoria serão executadas;
- Uma RN não pode chamar uma BD de outra classe;
- Não deve ocorrer comunicação direta entre BDs;
- Uma BD não pode persistir ou atualizar dados em mais de uma tabela.

#### InfraDTO

Para comunicação entre as camadas foi adotado o padrão de projeto DTO (Data Transfer Object) que consiste basicamente em passar apenas um objeto nas chamadas de métodos. Assim, por exemplo, na inserção de um documento, ao invés de passar todos os valores necessários como parâmetros individuais para o método de inserção, é criada uma classe para transporte denominada DocumentoDTO. Esta classe contém atributos para todos os valores envolvidos. Para inserir um documento deve ser instanciado um objeto desta classe sendo preenchidos os atributos com os valores correspondentes. Após esta etapa o objeto de transporte deve ser passado para o método de inserção.

Todo o tráfego entre as camadas deverá ser feito utilizando DTOs.

O framework estendeu o conceito original para permitir que o DTO seja uma fonte de informações do modelo de dados que possibilite a geração automática de instruções SQL. Para isso todos os DTOs deverão herdar da classe `InfraDTO` implementando alguns métodos abstratos. Ao realizar este procedimento a classe `InfraDTO` passará a fornecer automaticamente o seguinte conjunto de métodos:

Método	Descrição
<code>set&lt;atributo&gt;</code>	Configura o valor do atributo. A configuração do valor do atributo terá reflexos na operação que será realizada com o DTO: Inserções – serão considerados na montagem do INSERT Alterações – serão adicionados ao bloco SET do UPDATE Consultas – serão adicionados a cláusula WHERE, neste caso aceita um segundo parâmetro informado o critério de filtro: <code>InfraDTO::\$OPER_IN</code> <code>InfraDTO::\$OPER_NOT_IN</code> <code>InfraDTO::\$OPER_IGUAL</code> (valor padrão) <code>InfraDTO::\$OPER_DIFERENTE</code> <code>InfraDTO::\$OPER_LIKE</code> <code>InfraDTO::\$OPER_NOT_LIKE</code> <code>InfraDTO::\$OPER_MAIOR</code> <code>InfraDTO::\$OPER_MENOR</code> <code>InfraDTO::\$OPER_MAIOR_IGUAL</code> <code>InfraDTO::\$OPER_MENOR_IGUAL</code>
<code>get&lt;atributo&gt;</code>	Retorna o valor do atributo.
<code>unSet&lt;atributo&gt;</code>	Passa a indicar que o atributo não possui mais valor configurado.
<code>isSet&lt;atributo&gt;</code>	Verifica se o atributo possui valor configurado (utilizado na montagem do SQL).
<code>ret&lt;atributo&gt;</code>	Indica que o atributo deve ser retornado em consultas.
<code>unRet&lt;atributo&gt;</code>	Passa a indicar que o atributo não deve mais retornar na consulta.
<code>isRet&lt;atributo&gt;</code>	Verifica se o atributo deve retornar (utilizado na montagem do SQL).
<code>setOrd&lt;atributo&gt;</code>	Indica que o atributo deve ser utilizado para ordenação, recebe um parâmetro informando o tipo de ordenação (ascendente ou descendente): <code>InfraDTO::\$TIPO_ORDENACAO_ASC</code> <code>InfraDTO::\$TIPO_ORDENACAO_DESC</code>  Este método obedece a ordem de configuração dos atributos (para ordenação por múltiplos atributos).
<code>getOrd&lt;atributo&gt;</code>	Retorna o tipo de ordenação configurado atualmente para o atributo (utilizado na montagem do SQL).
<code>unOrd&lt;atributo&gt;</code>	Passa a indicar que o atributo não deve mais ser utilizado para ordenação.
<code>isOrd&lt;atributo&gt;</code>	Verifica se o atributo deve ser utilizado para ordenação (utilizado na montagem do SQL).
<code>retTodos</code>	Indica que todos os atributos devem ser retornados em uma consulta. Recebe um parâmetro true/false para indicar se os campos das tabelas relacionadas também devem ser retornados (valor padrão false).
<code>unSetTodos</code>	Indica que nenhum atributo possui valor configurado.
<code>unRetTodos</code>	Indica que nenhum atributo deve retornar em uma consulta.
<code>unOrdTodos</code>	Indica que nenhum atributo deve ser utilizado para ordenação.

Constantes de prefixos de tipos:

Nome	Descrição	Formato
InfraDTO::\$PREFIXO_NUM	Numérico sem decimais	
InfraDTO::\$PREFIXO_DIN	Valor monetário	999.999,99
InfraDTO::\$PREFIXO_DBL	Numérico com decimais	999999,99
InfraDTO::\$PREFIXO_STR	Texto	
InfraDTO::\$PREFIXO_DTA	Data	dd/mm/aaaa
InfraDTO::\$PREFIXO_DTH	Data e hora	dd/mm/aaaa hh:mm:ss
InfraDTO::\$PREFIXO_BOL	Booleano	
InfraDTO::\$PREFIXO_ARR	Array	
InfraDTO::\$PREFIXO_OBJ	Objeto	
InfraDTO::\$PREFIXO_BIN	Dados binários	

Para montagem do DTO utilizar os métodos de InfraDTO descritos abaixo:

❑ *getStrNomeTabela*

Método abstrato, deve retornar o nome da tabela correspondente no banco de dados (retornar *null* se o DTO não é persistido em uma tabela). Esta tabela será a principal na geração de SQLs com inner e left joins montados em relação a ela.

❑ *montarDTO*

Método abstrato, usado para adicionar e configurar os atributos do DTO.

❑ *adicionarAtributo (\$strPrefixo, \$strNome)*

Adiciona um atributo que não é persistido no DTO.

strPrefixo      Utilizar uma das constantes de prefixo de tipo da classe.  
strNome        Nome do atributo.

❑ *adicionarAtributoTabela(\$strPrefixo, \$strNome, \$strCampoSql)*

Adiciona um atributo da tabela principal no DTO.

strPrefixo      Utilizar uma das constantes de prefixo de tipo da classe.  
strNome        Nome do atributo.  
strCampoSql    Nome do campo correspondente na tabela do banco de dados.

❑ *adicionarAtributoTabelaRelacionada(\$strPrefixo, \$strNome, \$strCampoSqlRelacionado, \$strTabelaRelacionada)*

Adiciona um atributo de uma tabela relacionada com a tabela principal do DTO. O uso deste método requer a configuração da chave-estrangeira correspondente através do método *configurarFK*.

strPrefixo      Utilizar uma das constantes de prefixo de tipo da classe.  
strNome        Nome do atributo no formato <campo><tabela relacionada> (Ex.: NomeSerie, SiglaOrgao).

strCampoSql Relacionado	Nome do campo correspondente na tabela relacionada do banco de dados.
----------------------------	---

❑ *configurarPK(\$strAtributo, \$numTipoPK)*

Indica que o atributo é chave-primária.

strAtributo	Nome do atributo, deve ser igual ao nome de um atributo já adicionado ao DTO através do método <i>adicionarAtributo</i> .
-------------	---

numTipoPK	Informa o tipo da chave-primária, utilizar uma das constantes:
-----------	--

- `InfraDTO::$TIPO_PK_INFORMADO` - ao inserir um registro o valor do ID deve ser informado através da chamada do método `set<atributo>` correspondente).
- `InfraDTO::$TIPO_PK_SEQUENCIAL` - na inserção a classe de banco buscará o próximo ID através da classe de infra-estrutura `InfraSequencia`.
- `InfraDTO::$TIPO_PK_NATIVA` - na inserção a classe de banco buscará o próximo ID através em um objeto de sequência, ver seção Padrão de Modelagem de Dados.

❑ *configurarFK(\$strAtributo, \$strTabelaRelacionada, \$strCampoRelacionado, \$numTipoFK, \$numFiltroFK)*

Indica que o campo é chave-estrangeira.

strAtributo	Nome do atributo, deve ser igual ao nome de um atributo já adicionado ao DTO através dos métodos <i>adicionarAtributo</i> ou <i>adicionarAtributoTabelaRelacionada</i> .
-------------	--

strTabelaRelacionada	Nome da tabela que contém o atributo chave-primária correspondente a esta chave-estrangeira.
----------------------	--

strCampoRelacionado	Nome do campo chave-primária na tabela relacionada.
---------------------	---

\$numTipoFK	Informa o tipo da chave-estrangeira, utilizar uma das constantes:
-------------	---

- `InfraDTO::$TIPO_FK_OBRIGATORIA` (default) - fará um INNER JOIN com a tabela principal.
- `InfraDTO::$TIPO_FK_OPCIONAL` - fará um LEFT JOIN com a tabela principal.

\$numFiltroFK	Indica a forma como os registros relacionados serão filtrados:
---------------	--

- `InfraDTO::$FILTRO_FK_ON` (default) - montará filtros da tabela relacionada diretamente no inner ou left join através

da cláusula ON.

- `InfraDTO::$FILTRO_FK_WHERE` - aplicará o filtro sobre o resultado final da consulta na cláusula WHERE.

□ *configurarExclusaoLogica(\$strAtributo, \$valorDesativado)*

Sinaliza qual atributo deve ser atualizado no caso de uma exclusão lógica (método desativar).

`strAtributo` Nome do atributo, deve ser igual ao nome de um atributo já adicionado ao DTO através do método *adicionarAtributo*.

`valorDesativado` Informa qual valor representa um registro desativado.

Exemplo de código de um DTO:

```
class DocumentoDTO extends InfraDTO {
    public function getStrNomeTabela() {
        return "documento";
    }

    public function montarDTO() {

        $this->adicionarAtributoTabela(InfraDTO::$PREFIXO_DBL, "IdDocumento",
        "id_documento");

        $this->adicionarAtributoTabela(InfraDTO::$PREFIXO_NUM, "IdSerie", "id_serie");

        $this->adicionarAtributoTabela(InfraDTO::$PREFIXO_STR, "Numero", "numero");

        //... demais campos da tabela documento ...

        $this->adicionarAtributoTabelaRelacionada(InfraDTO::$PREFIXO_STR, "NomeSerie",
        "nome", "serie");

        $this->configurarPK("IdDocumento", InfraDTO::$TIPO_PK_NATIVA);

        $this->configurarFK("IdSerie", "Serie", "id_serie");

    }
}
```

Exemplo de uma consulta utilizando o DTO anterior:

- 1) Retornar IdDocumento, Tipo do Documento e Número;
- 2) Somente para tipos de documento com ID igual 18 e 20 (Atos e Portarias);
- 3) Ordenadas pelo tipo de documento ascendente e pelo número descendente

```
$objDocumentoDTO = new DocumentoDTO();
$objDocumentoDTO->retDblIdDocumento();
$objDocumentoDTO->retStrNomeSerie();
$objDocumentoDTO->retStrNumero();
$objDocumentoDTO->setNumIdSerie(array(18,20),InfraDTO::$OPER_IN);
$objDocumentoDTO->setOrdStrNomeSerie(InfraDTO::$TIPO_ORDENACAO_ASC);
$objDocumentoDTO->setOrdStrNumero(InfraDTO::$TIPO_ORDENACAO_DESC);

$objDocumentoRN = new DocumentoRN();
$arrObjDocumentoDTO = $objDocumentoRN->listarRN0008($objDocumentoDTO);
```



Consulta SQL gerada automaticamente:

```
SELECT documento.id_documento AS iddocumento,serie.nome AS nomeserie,documento.numero
FROM documento INNER JOIN serie ON documento.id_serie=serie.id_serie
WHERE documento.id_serie IN (18,20)
ORDER BY serie.nome ASC, documento.numero DESC
```

## InfraPagina (1ª Camada)

A interface é composta por arquivos PHP que acessam as classes RN e geram páginas HTML. Para a comunicação com a 2ª camada os dados devem ser encapsulados em DTOs e repassados para os métodos adequados. A comunicação entre elementos da interface pode ser feita através GET, POST, ou variáveis de sessão, considerando que:

- A URL de uma página deve conter pelo menos um parâmetro obrigatório denominado *acao*, cujo valor deve ser composto por <entidade>\_<acao> (documento\_assinar, serie\_excluir, cidade\_listar). Normalmente estes valores correspondem a recursos no SIP;
- Deve existir um módulo controlador denominado “controlador.php” que estabeleça a navegação entre páginas diferentes através da análise do parâmetro *acao*;
- Toda página deve submeter os seus dados para ela mesma ou para o módulo controlador;

Os principais arquivos de interface para cada entidade do sistema são:

- <nome da entidade>\_lista.php - listagem ou pesquisa de registros, nesta tela também normalmente são realizadas operações sobre vários registros simultaneamente como exclusão e desativação;
- <nome da entidade>\_cadastro.php - suporta as operações de cadastro, alteração e consulta;
- <nome da entidade>INT.php - elementos de interface compartilhados (combos, checkboxes, formatação de dados para exibição, etc...)

Para facilitar o desenvolvimento da interface foi implementada uma classe, denominada InfraPagina que monta todo o esqueleto das páginas adicionando barras superiores, menu, barra de comandos, etc. Esta classe pode ter outras especializações montando páginas com layouts diferentes: InfraPaginaEsquema, InfraPaginaEsquema2,...

Exemplo para montagem de página com a classe InfraPagina:

```
<?
try {
    require_once dirname(__FILE__).'../../SEI.php';

    session_start();

    //////////////////////////////////////
    //InfraDebug::getInstance()->setBolLigado(false);
```

```

//InfraDebug::getInstance()->setBolDebugInfra(false);
//InfraDebug::getInstance()->limpar();
////////////////////////////////////

SessaoSEI::getInstance()->validarLink();

SessaoSEI::getInstance()->validarPermissao($_GET['acao']);

$arrComandos = array();

switch($_GET['acao']){

    case 'md_abc_exemplo1':

        $strTitulo = 'Exemplo 1 ABC';
        //processamento
        break;

    case 'md_abc_exemplo2':

        $strTitulo = 'Exemplo 2 ABC';
        //processamento
        break;

    default:
        throw new InfraException("Ação '".$_GET['acao']."' não reconhecida.");
}

//monta tabela se for página de listagem
}catch(Exception $e){
    PaginaSEI::getInstance()->processarExcecao($e);
}
PaginaSEI::getInstance()->montarDocType();
PaginaSEI::getInstance()->abrirHtml();
PaginaSEI::getInstance()->abrirHead();
PaginaSEI::getInstance()->montarMeta();
PaginaSEI::getInstance()->montarTitle(PaginaSEI::getInstance()->getStrNomeSistema().'
- '.$strTitulo);
PaginaSEI::getInstance()->montarStyle();
PaginaSEI::getInstance()->abrirStyle();
?>
/* CSS */
<?
PaginaSEI::getInstance()->fecharStyle();
PaginaSEI::getInstance()->montarJavaScript();
PaginaSEI::getInstance()->abrirJavaScript();
?>
/* Javascript */
<?
PaginaSEI::getInstance()->fecharJavaScript();
PaginaSEI::getInstance()->fecharHead();
PaginaSEI::getInstance()->abrirBody($strTitulo,'onload="inicializar();"');
?>
<form ....>
<?
PaginaSEI::getInstance()->montarBarraComandosSuperior($arrComandos);
//PaginaSEI::getInstance()->montarAreaValidacao();
PaginaSEI::getInstance()->abrirAreaDados('5em');
?>
<!-- HTML -->
<?
PaginaSEI::getInstance()->fecharAreaDados();
PaginaSEI::getInstance()->montarAreaTabela($strResultado, $numRegistros);
//PaginaSEI::getInstance()->montarAreaDebug();
PaginaSEI::getInstance()->montarBarraComandosInferior($arrComandos);
?>

```

```

</form>
<?
PaginaSEI::getInstance()->fecharBody();
PaginaSEI::getInstance()->fecharHtml();
?>

```

Abaixo a relação de classes CSS disponíveis:

Controle HTML	Classe CSS	Observação
Input type = [button, reset, submit]	infraButton	
label	infraLabelOpcional	Descrição de campos opcionais
label	infraLabelObrigatorio	Descrição de campos obrigatórios
table	infraTable	
tr	infraTrClara	Linha clara em tabelas de resultado
tr	infraTrEscula	Linha escura em tabelas de resultado
th	infraTh	Linha de título para tabelas
caption	infraCaption	Linha de descrição da tabela
Input type = checkbox	infraCheckbox	
Input type = radio	infraRadio	
fieldset	infraFieldset	
legend	infraLegendOpcional	Grupo de campos opcional
legend	infraLegendObrigatorio	Grupo de campos obrigatório
span	infraTeclaAtalho	Visualização da tecla de atalho em um label
form	infraForm	
Input type = text	infraText	
Input type = password	infraPassword	
textarea	infraTextarea	
select	infraSelect	

## InfraRN (2ª Camada)

A segunda camada implementa o padrão de projeto estrutural Façade gerando uma fachada para acesso aos conceitos do sistema. Nela todas as regras de negócio levantadas devem ser implementadas incluindo regras simples que já tenham sido validadas na interface como, por exemplo, tamanho máximo de campos e validação de datas. Desta forma garantimos que qualquer cliente (browser, web service, app, ...) que acesse as classes da 2ª camada realizará as validações de regras, permissões e auditorias necessárias.

A classe InfraRN implementa um mecanismo para controle automático do contexto de conexão ou transação. Para utilização da classe é necessário que:

- 1) a classe de 2ª camada herde de InfraRN;
- 2) seja implementado o método *inicializarObjInfraBanco*;
- 3) para cada método que se deseja controlar criar um método protegido com o sufixo *Controlado* ou *Conectado* recebendo como parâmetro apenas o DTO da entidade ou um array de parâmetros. Ao invocar métodos com o sufixo Conectado a classe abrirá uma conexão se já não estiver aberta e com o sufixo Controlado abrirá uma conexão e uma transação se já não estiverem abertas.

Exemplo de método da camada de regras de negócio:

```
protected function processarControlado($objDocumentoDTO) {  
  
    try{  
        //Validação de Permissão e Auditoria  
  
        //Validação de Regras de Negócio  
  
        //Executa métodos de outras RNs e da BD de documento  
  
    }catch(Exception $e){  
        throw new InfraException("Erro processando documento.", $e);  
    }  
}
```

Utilizando este mecanismo, após a primeira chamada a um método Conectado ou Controlado, a conexão ficará aberta até o final da execução do script. Transações vão respeitar o mesmo escopo do método sendo confirmadas no final da execução do método ou canceladas caso uma exceção seja lançada.

### InfraBD (3ª camada)

A classe BD isola a forma como os dados são acessados e foi desenvolvida com o objetivo de automatizar os métodos básicos: cadastrar, alterar, consultar, listar, contar, excluir, desativar, reativar e bloquear. Para sua utilização basta que a classe de 3ª camada herde de InfraBD e repasse para seu construtor a instância do banco recebida da classe de 2ª camada:

```
class DocumentoBD extends InfraBD {  
    public function __construct($objInfraIBanco){  
        parent::__construct($objInfraIBanco);  
    }  
}
```

Após este procedimento as chamadas aos métodos automatizados serão respondidas pela classe InfraBD que dinamicamente montará o comando SQL de acordo com a configuração do DTO recebido. A classe também disponibiliza o método getObjInfraIBanco para recuperação do objeto passado para o seu construtor.

Se for necessário implementar algo específico na BD é recomendado utilizar os métodos abaixo:

- formatarGravacao [Dta | Dth | Str | Bol | Num | Din | Dbl | Bin]  
Para converter o formato para gravação no banco.
- formatarLeitura [Dta | Dth | Str | Bol | Num | Din | Dbl | Bin]  
Para converter o formato lido do banco.
- formatarSelecao [Dta | Dth | Str | Bol | Num | Din | Dbl | Bin]  
Para informar ao banco como trazer o campo (cast).

Exemplo de método de alteração na camada de banco:

```

public function processarXXXXX($objXXXXXDTO){
    try {

        $objBanco = $this->getObjInfraIBanco();

        $sql = "UPDATE xxxxx SET campo_e=". $objBanco->formatarGravacaoStr('T')." WHERE
campo_d <= ". $objBanco->formatarGravacaoDta('01/01/2016');

        $objBanco->executarSql($sql);

    }catch(Exception $e){
        throw new InfraException("Erro processando XXXXX.", $e);
    }
}

```

Exemplo de método de alteração na camada de banco:

```

public function listarXXXXX() {
    try {

        $objBanco = $this->getObjInfraIBanco();

        $sql = "SELECT ".
            $objBanco->formatarSelecaoNum('xxxxx','campo_a','CampoA').",".
            $objBanco->formatarSelecaoStr('xxxxx','campo_b','CampoB').",".
            $objBanco->formatarSelecaoDta('xxxxx','campo_c','CampoC').
            " FROM xxxxx ".
            " WHERE campo_d > ". $objBanco->formatarGravacaoDta('01/01/2016').
            " AND campo_e = ". $objBanco->formatarGravacaoStr('A');

        $rs = $objBanco->consultarSql($sql);

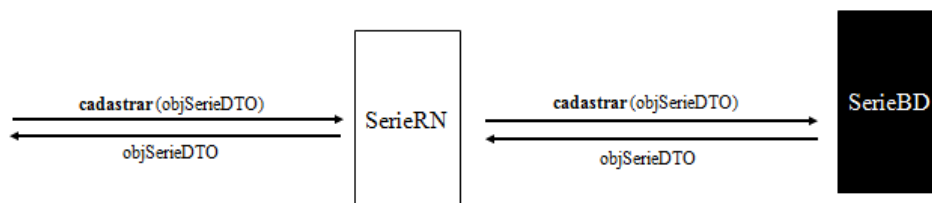
        $ret = array();
        foreach($rs as $item){
            $objXXXXXDTO = new XXXXXDTO();
            $objXXXXXDTO->setNumCampoA($objBanco->formatarLeituraNum($item['CampoA']));
            $objXXXXXDTO->setStrCampoB($objBanco->formatarLeituraStr($item['CampoB']));
            $objXXXXXDTO->setDtaCampoC($objBanco->formatarLeituraDta($item['CampoC']));
            $ret[] = $objXXXXXDTO;
        }

        return $ret;
    }catch(Exception $e){
        throw new InfraException("Erro listando XXXXX.", $e);
    }
}

```

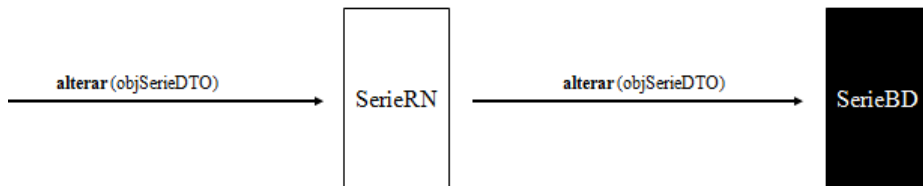
## Métodos Padronizados

### Cadastrar (Controlado)



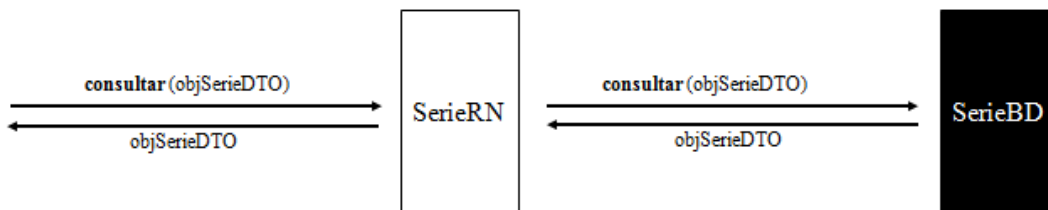
- Enviar um DTO preenchido com os atributos para cadastro (set). Se for uma chave-primária seqüencial ou nativa então o atributo correspondente deve ser configurado com null;
- Será retornado um DTO preenchido com a chave-primária. Por questão de uniformidade no tratamento dos métodos, mesmo que a chave-primária já seja conhecida antes do cadastramento ainda assim será retornado um DTO com o seu valor preenchido.

### Alterar (Controlado)



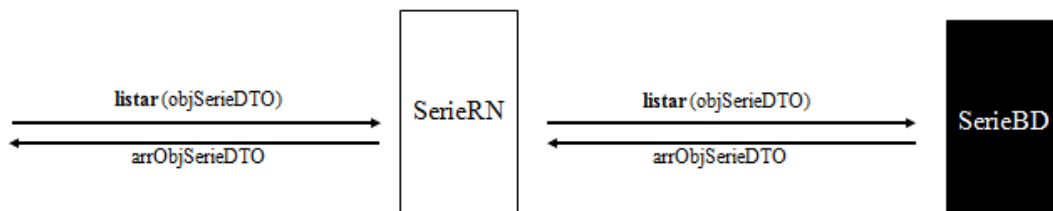
- Enviar um DTO preenchido (set) com a chave-primária e com os atributos para alteração;
- Nenhum DTO será retornado pois não haverá acréscimo de informações.

### Consultar (Conectado)



- Enviar um DTO com os atributos de retorno (ret) configurados e preenchido (set) com atributos que identifiquem uma determinada ocorrência (chave-primária ou chave-candidata);
- Se mais de uma ocorrência for selecionada será gerado um erro;
- É retornado um DTO com os valores dos atributos solicitados de acordo com os dados atualmente persistidos. Os campos retornados estarão disponíveis para uso no objeto (get).

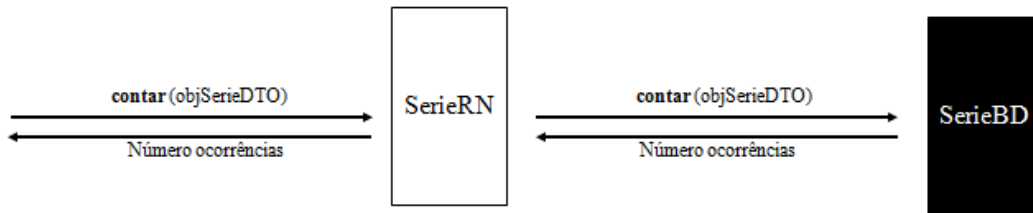
### Listar (Conectado)



- Enviar um DTO com os atributos de retorno (ret), pesquisa (set) e ordenação (setOrd) preenchidos;

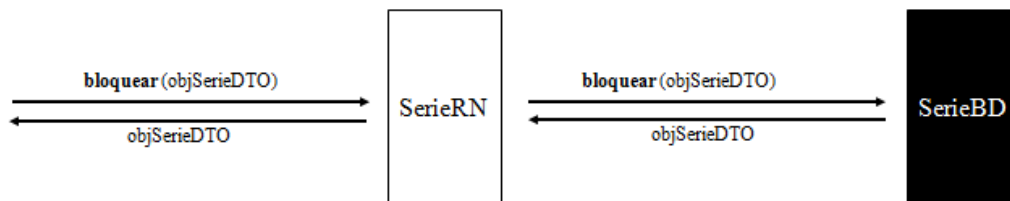
- Será retornada uma lista de DTOs com todos os registros que atenderam aos critérios de pesquisa e preenchido com os atributos solicitados disponíveis para uso (get);
- Este método não atenderá a todos os casos possíveis. Para outras pesquisas que se façam necessárias deve criado um método Conectado com estrutura semelhante, ou seja, recebendo um DTO e retornando uma lista.

### Contar (Conectado)



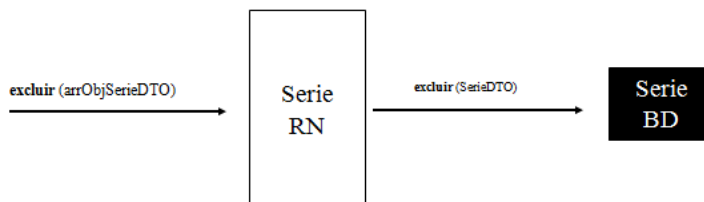
- Enviar um DTO com os atributos de pesquisa (set) preenchidos;
- Será retornado o número de ocorrências que atenderam aos critérios de pesquisa.

### Bloquear (Controlado)



- Enviar um DTO preenchido (set) com o atributo chave-primária;
- Será retornado um DTO com todos os atributos da ocorrência no banco;
- O registro ficará em lock até o fim da transação corrente.

### Excluir, Desativar e Reativar (Controlado)



- Enviar uma lista de DTOs com os atributos correspondentes a chave-primária preenchidos (set);
- A passagem de uma lista permite que o método seja utilizado para processamento em lote (nas interfaces é comum selecionar mais de um item para estas operações).
- O método da RN recebe uma lista enquanto que o método da BD recebe um DTO específico.

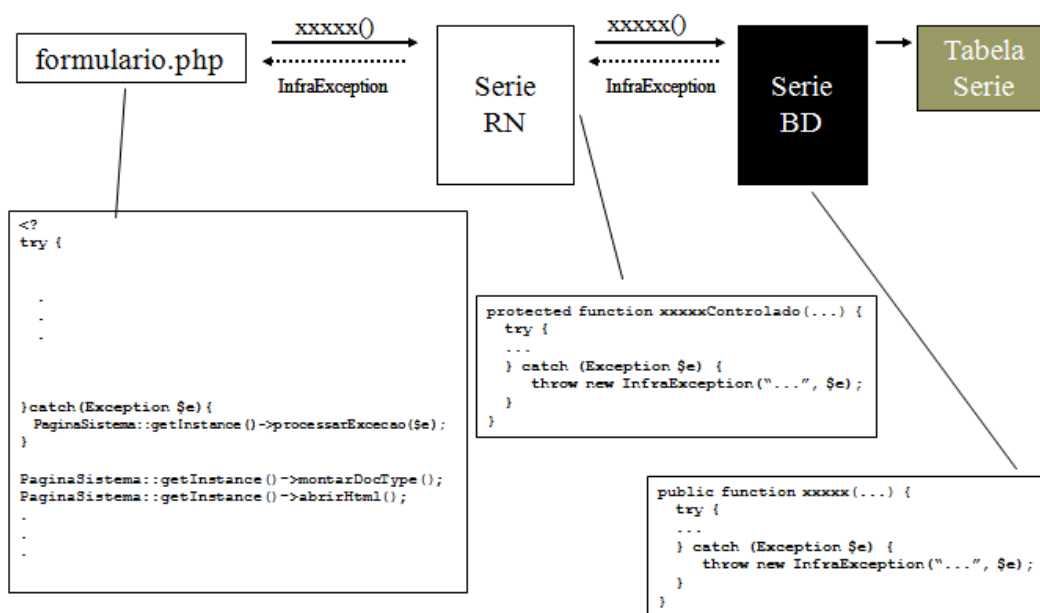
No SIP, por meio do menu "Recursos/Gerar Padrão PHP", é possível gerar automaticamente os recursos para utilização nos métodos padrão:

Observações:

- O método *contar* compartilha o mesmo recurso do método *listar*;
- O método *bloquear* compartilha o mesmo recurso do método *consultar*;
- O recurso *selecionar* é utilizado em telas de escolha de registros (lupas).

## InfraException

A classe *InfraException* fornece um mecanismo para transporte de validações e erros entre as camadas. Para que o tratamento ocorra de forma adequada todo código deve ser implementado com blocos *try...catch*. A *InfraException* lançada deverá retornar na hierarquia de métodos até o ponto da 1ª camada que iniciou o procedimento, que então deverá informar o erro ou validação para o usuário. Todas as mensagens geradas pelo PHP (exceto Warnings) serão encapsuladas e lançadas em uma *InfraException* automaticamente.





Exemplo de tratamento de validações na camada de regras de negócio:

- Lançando uma validação em um alert:

```
$objInfraException = new InfraException();
if (...1...){
    $objInfraException->lancarValidacao('v1');
}
```

- Acumulando diversas validações e exibindo todas em um único alert:

```
$objInfraException = new InfraException();

if (...1...){
    $objInfraException->adicionarValidacao('v1');
}

if (...2...){
    $objInfraException->adicionarValidacao('v2');
}

if (...3...){
    $objInfraException->adicionarValidacao('v3');
}

$objInfraException->lancarValidacoes();
```

- ❑ `__construct($strDescricao=null, $e=null)`

Construtor.

<code>strDescricao</code>	Opcional. Texto indicando o motivo da exceção.
<code>e</code>	Opcional. Exceção original capturada no bloco <i>catch</i> .

- ❑ `adicionarValidacao($strDescricao)`

Armazena uma validação de Regra de Negócio.

<code>strDescricao</code>	Mensagem para exibição ao usuário.
---------------------------	------------------------------------

- ❑ `contemValidacoes( )`

Verifica se o objeto contém validações armazenadas retornando true/false.

- ❑ `lancarValidacao($strDescricao)`

Lança imediatamente uma exceção contendo a validação passada como parâmetro.

- ❑ `lancarValidacoes( )`

Verifica se objeto contém validações armazenadas caso afirmativo então lança o objeto.

## Estrutura de Diretórios

Abaixo a estrutura de diretórios recomendada:

Diretório	Conteúdo
dto	Classes de transporte <EntidadeDTO.php>
int	Classes de interface <EntidadeINT.php>
rn	Classes da regras de negócio <EntidadeRN.php>
bd	Classes de banco <EntidadeBD.php>
ws	Classes de web services <ServicoWS.php>
css	Código CSS
imagens	Imagens da aplicação
js	Código javascript

## 8. Padrão de Modelagem de Dados

### Regras Gerais

Regras de nomenclatura aplicáveis a todos os elementos (tabelas, colunas, índices, etc) do modelo de dados:

- usar somente letras, números e o caractere sublinhado;
- limitar o tamanho ao máximo de 30 caracteres;
- utilizar apenas letras minúsculas;
- palavras internas devem ser separadas pelo caractere sublinhado;
- suprimir as preposições;
- utilizar apenas palavras no singular.

### Tabelas

- Não utilizar verbos para designar nomes de tabelas, priorizar o uso de substantivos e adicionar o prefixo "md\_<instituição/módulo>":

```
md_abc_pedido  
md_abc_item  
md_abc_nota_fiscal
```

- Para tabelas que implementam relacionamentos (n x n) utilizar o prefixo "md\_<instituição/módulo>\_rel" seguido dos nomes das tabelas envolvidas (sem o prefixo "md\_<instituição/módulo>") e separados por sublinhado:

```
md_abc_rel_pedido_item
```

Se o relacionamento expressar um conceito forte do sistema então o nome deste conceito pode ser utilizado:

```
md_abc_administrador_sistema (e não md_abc_rel_usuario_sistema)
```

- Tabelas que representam conjuntos de valores relacionados a uma determinada entidade, devem conter um prefixo indicativo do tipo do conjunto seguido do nome da entidade:

md\_abc\_tipo\_pedido

## Colunas

- Para chaves primárias sequenciais utilizar o prefixo id seguido do nome da tabela:

id\_md\_abc\_pedido

- Para chaves primárias que fazem uso de chaves estrangeiras, manter a nomenclatura da chave estrangeira igual a da chave primária de origem:

id\_md\_abc\_tabela\_a

id\_md\_abc\_tabela\_b

- Prefixos recomendados:

sin\_ campo sinalizador que aceita apenas os valores S ou N

sta\_ status multi-valorado. Ex.: P=Processo, G=Doc. Gerado, E=Doc. Externo

dta\_ data

dth\_ data/hora

din\_ dinheiro

- Para exclusão lógica utilizar o nome de campo:

sin\_ativo

## Tipos de Dados

Quanto ao tipo de dado usado na representação, aconselha-se, para maior portabilidade, a escolha de um dos tipos principais definidos pelo padrão SQL-99 (ou SQL3):

Tipo	Parâmetro	Significado
integer	-	Números inteiros com sinal, o número de bits utilizado na representação é dependente da implementação (geralmente 32 bits).
smallint	-	Números inteiros pequenos com sinal, o número de bits utilizado na representação é dependente da implementação (geralmente 16 bits).
numeric	[(precisão [,decimais]) ]	Números decimais com precisão fixa. Ao criar uma coluna do tipo numeric é necessário especificar o comprimento total do número e o número de casas decimais. Este tipo é recomendado para representação de moedas. Muitos bancos de dados possuem um tipo money (não padronizado) que, na maioria das vezes, é um campo numeric com precisão e decimais específicos.
decimal	[(precisão [,decimais]) ]	Semelhante ao numeric, entretanto deve possuir uma precisão maior permitindo mais casas decimais.

float	[( <i>precisão</i> )]	Números com precisão única em ponto flutuante. A faixa de valores e a precisão da representação dependem da implementação.
double	-	Números com precisão dupla em ponto flutuante. A faixa de valores e a precisão da representação dependem da implementação, mas é sempre igual ou melhor do que o tipo float.
blob	[( <i>tamanho</i> )]	Usado para armazenagem de qualquer dado em formato binário. O significado do parâmetro <i>tamanho</i> depende da implementação do banco, podendo ser medido em Kb, Mb ou até Gb.
char	[( <i>tamanho</i> )]	Usado na representação de seqüências de caracteres de tamanho fixo. Tenha cuidado com o parâmetro <i>tamanho</i> – para maior portabilidade, evite comprimentos superiores a 1000 caracteres.
varchar	( <i>tamanho</i> )	Usado na representação de seqüências de caracteres de tamanho variável.
clob	[( <i>tamanho</i> )]	Semelhante ao tipo blob só que aplicado a caracteres.
date	-	Um valor de data no formato YYYY-MM-DD. A faixa de valores para o ano pode variar de 1 a 9999).
time	[( <i>precisão</i> )]	Um valor de hora no formato hh:mm:ss.nnn. O parâmetro <i>precisão</i> indica as frações de segundo representadas, e é dependente da implementação (geralmente variando entre 0 e 6).
timestamp	[( <i>precisão</i> )]	Data e hora no formato YYYY-MM-DD hh:mm:ss.nnn. O parâmetro <i>precisão</i> indica as frações de segundo representadas, e é dependente da implementação (geralmente variando entre 0 e 6).
boolean	-	Valor lógico booleano (verdadeiro/falso).

### Chave Primária

Utilizar o prefixo pk seguido do nome da entidade:

pk\_md\_abc\_pedido

### Chave Alternativa

Utilizar o prefixo ak seguido do nome da entidade e do nome do(s) campo(s) que a compõem:

ak\_md\_abc\_pedido\_codigo

### Chave Estrangeira

Utilizar o prefixo fk seguido do prefixo "md\_<instituição/módulo>" e dos nomes da entidade que possui a chave estrangeira e da entidade à qual a chave faz referência (sem o prefixo "md\_<instituição/módulo>"):

fk\_md\_abc\_item\_pedido

## Índices

Para índices utilizar o prefixo i(01-99) seguido do nome da entidade.

```
i01_md_abc_pedido  
i02_md_abc_pedido  
...  
i99_md_abc_pedido
```

## Seqüências

Utilizar o prefixo seq seguido do nome do objeto ao qual a seqüência atende:

```
seq_md_abc_pedido
```

São utilizadas quando os DTOs possuem o tipo da chave primária "nativa" e, dependendo do banco de dados utilizado, podem ser tabelas ou sequences:

- MySQL

```
create table seq_md_abc_pedido (id int not null primary key AUTO_INCREMENT, campo  
char(1) null)
```

- SQL Server

```
create table seq_md_abc_pedido (id int identity(1,1), campo char(1) null)
```

- Oracle

```
CREATE SEQUENCE seq_md_abc_pedido START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE
```

## 9. Padrão de Codificação PHP

### Nomes de Arquivos

Devem possuir apenas letras minúsculas, números e o caractere sublinhado acrescidos da extensão “.php”.

Classes ou interfaces devem utilizar arquivos individuais, neste caso, o nome do arquivo deverá obrigatoriamente ser o mesmo da classe ou interface acrescido do sufixo “.php”.

### Classes e Interfaces

- Nomes de classe devem ser substantivos no singular tendo como prefixo a sigla da instituição:

```
MdAbcPedido, MdAbcItem
```

- Utilizar a primeira letra de cada palavra em maiúscula:

```
MdAbcNotaFiscal
```

- Aconselha-se a supressão de preposições:

```
MdAbcRequisicaoPagamento
```

### Instâncias de Classes

- Para nomear instâncias de classes utilize o prefixo *obj* seguido do nome da classe:

`objMdAbcPedido, objMdAbcRequisicaoPagamento`

- Se for necessário utilizar mais de uma instância da mesma classe utilize um sufixo descritivo separado pelo caractere sublinhado:  
`objMdAbcPedidoOriginal, objMdAbcPedidoReclassificado`

## Constantes

- Devem ser declaradas com todas as letras em maiúsculo:  
`MD_ABC_FATOR`
- Termos compostos devem ser separados por sublinhado:  
`MD_ABC_CONSUMO_MINIMO`

## Atributos e Variáveis de Métodos

Os atributos e variáveis devem ser nomeados utilizando um prefixo e um qualificador. O prefixo é definido de acordo com o tipo do atributo e o qualificador deve descrever o melhor possível o seu significado dentro da lógica do sistema. O qualificador deve conter a primeira letra em maiúscula e as demais em minúsculas. Termos compostos devem utilizar a primeira letra de cada termo em maiúscula e as demais em minúsculas (não deve ser utilizado o caractere sublinhado):

<b>Tipo do Atributo</b>	<b>Prefixo</b>	<b>Exemplo</b>
Número	num	numIdadeMinima
String	str	strNome
Data	dta	dtaNascimento
Data/Hora	dth	dthEntrega
Array	arr	arrObjProtocolo
Booleano	bol	bolEncontrouUnidade

## Métodos

Os nomes dos métodos devem ser verbos no infinitivo e conter apenas letras minúsculas. Termos compostos devem utilizar a primeira letra, do segundo termo em diante, em maiúscula (não deve ser utilizado o caractere sublinhado):

`cadastrar`  
`calcularJuros`  
`gerarEstatisticas`

## Elementos HTML

Os elementos HTML devem ser nomeados utilizando um prefixo e um qualificador. O prefixo é definido de acordo com a tabela abaixo e o qualificador deve descrever o propósito do componente contendo a primeira letra em maiúscula e as demais em minúsculas. Termos compostos devem utilizar a primeira letra de cada termo em maiúscula e as demais em minúsculas (não deve ser utilizado o caractere sublinhado):

<b>Elemento</b>	<b>Prefixo</b>	<b>Exemplo</b>
<code>[input type=] text</code>	txt	txtNome
<code>[input type=] password</code>	pwd	pwdSenha

[input type=] checkbox	chk	chkUnidadeProtocolo
[input type=] radio	rdo	rdoNivelAcesso
[input type=] submit	sbm	sbmSalvar
[input type=] file	fil	filAnexo
[input type=] hidden	hdn	hdnIdCidade
[input type=] image	img	imgLupa
[input type=] button	btn	btnFechar
Form	frm	frmAnotacaoCadastro
Div	div	divEndereco
Table	tbl	tblLocalizadores
iFrame	ifr	ifrmArvore
TextArea	txa	txaDescricao
Select	sel	selUnidades

## Indentação

Deve-se utilizar DOIS à QUATRO espaços para cada nível de indentação (não utilizar tabulação).

## 10. Gerador de Código

As operações básicas podem ser geradas no framework InfraPHP através de um gerador de código disponível no endereço:

<https://infra-php-desenv.trf4.jus.br/gerador>

A geração do código é realizada com base na interpretação dos comandos DDL utilizados para criação da base de dados. É muito importante seguir o padrão de modelagem de dados, principalmente com relação ao prefixos dos campos, pois o gerador utilizará estas informações para gerar máscaras para os campos na interface e validações específicas tanto em javascript como na camada de regras de negócio.

Os comandos SQL podem ser escritos em vários dialetos e o gerador aceita somente alguns deles, exemplo:



```
CREATE TABLE md_abc_aquisicao(
    id_md_abc_aquisicao integer NOT NULL ,
    id_md_abc_projeto int NOT NULL ,
    descricao varchar(50) NOT NULL ,
    din_custo numeric(12,2) NOT NULL
);
```

```
ALTER TABLE md_abc_aquisicao ADD CONSTRAINT pk_md_abc_aquisicao PRIMARY KEY (id_md_abc_aquisicao
ASC);
```

```
CREATE INDEX i01_md_abc_aquisicao ON md_abc_aquisicao (id_md_abc_projeto ASC);
```

```
CREATE TABLE md_abc_projeto(
    id_md_abc_projeto    int NOT NULL ,
    identificacao        varchar(50) NOT NULL ,
    descricao            varchar(max) NULL ,
    dta_cadastramento   datetime NOT NULL ,
    sin_ativo            char(1) NOT NULL
);
```

```
ALTER TABLE md_abc_projeto ADD CONSTRAINT pk_md_abc_projeto PRIMARY KEY (id_md_abc_projeto
ASC);
```

```
ALTER TABLE md_abc_aquisicao ADD CONSTRAINT fk_md_abc_projeto_aquisicao FOREIGN KEY
(id_md_abc_projeto) REFERENCES md_abc_projeto(id_md_abc_projeto);
```

### 1) interpretar os comandos SQL:

**Tribunal Regional Federal da 4ª Região**

Gerador de Código

**SQL**

Interpretar Ajuda ERwin

Script SQL:

```
CREATE TABLE md_abc_aquisicao(
    id_md_abc_aquisicao integer NOT NULL ,
    id_md_abc_projeto int NOT NULL ,
    descricao varchar(50) NOT NULL ,
    din_custo numeric(12,2) NOT NULL
);

ALTER TABLE md_abc_aquisicao ADD CONSTRAINT pk_md_abc_aquisicao PRIMARY KEY (id_md_abc_aquisicao ASC);

CREATE INDEX i01_md_abc_aquisicao ON md_abc_aquisicao (id_md_abc_projeto ASC);

CREATE TABLE md_abc_projeto(
    id_md_abc_projeto int NOT NULL ,
    identificacao varchar(50) NOT NULL ,
    descricao varchar(max) NULL ,
    dta_cadastramento datetime NOT NULL ,
    sin_ativo char(1) NOT NULL
);

ALTER TABLE md_abc_projeto ADD CONSTRAINT pk_md_abc_projeto PRIMARY KEY (id_md_abc_projeto ASC);

ALTER TABLE md_abc_aquisicao ADD CONSTRAINT fk_md_abc_projeto_aquisicao FOREIGN KEY (id_md_abc_projeto) REFERENCES
```

### 2) escolher uma tabela para geração:

**Tribunal Regional Federal da 4ª Região**

Gerador de Código

**Tabelas**

Gerar Meta Dados Voltar

Linguagem:  
PHP ▼

<input checked="" type="radio"/>	Tabela
<input type="radio"/>	md_abc_aquisicao
<input type="radio"/>	md_abc_projeto

### 3) preencher o usuário, arquivo principal (que inclui a InfraPHP) e as classes de sessão, página e banco do sistema:



**Tribunal Regional Federal da 4ª Região**

Gerador de Código

---

**PHP**

Usuário:  Módulo Principal:

InfraSessao:  InfraPagina:  InfraIBanco:

4) informar o campo principal, os rótulos dos campos e teclas de atalho:

**Tribunal Regional Federal da 4ª Região**

Gerador de Código

---

**md\_abc\_projeto**

Artigo: Singular:   Plural:

☒ Gerar ordenação de colunas na tela de listagem

	Campo	Artigo	Rótulo	Tecla Atalho
<input type="radio"/>	id_md_abc_projeto			
<input checked="" type="radio"/>	identificacao	a	Identificação	I
<input type="radio"/>	descricao	a	Descrição	D
<input type="radio"/>	dta_cadastramento	a	Data de Início	a
<input type="radio"/>	sin_ativo			

Tabelas Relacionadas:

Campo principal:

- será retornado automaticamente na tela de lista junto com o ID;
- será gerado um método para montagem de combo na classe MdAbcProjetoINT buscando por este campo;
- no DTO das tabelas relacionadas ele será recuperado automaticamente, ou seja, em MdAbcAquisicaoDTO será adicionada uma FK para *md\_abc\_projeto* e um atributo relacionado "IdentificacaoMdAbcProjeto". Para que isso ocorra é necessário configurar o campo principal da tabela *md\_abc\_projeto* antes de gerar o código de *md\_abc\_aquisicao*. Se o código de *md\_abc\_projeto* foi gerado em outra ocasião basta selecionar a tabela e avançar até o passo final escolhendo apenas o campo principal (sem preencher todos os campos).

5) Ao final o código poderá ser visualizado ou baixado:

Tribunal Regional Federal da 4ª Região

Gerador de Código

PHP - Classes

Tabelas

Voltar

Classe	Ações
MdAbcProjetoDTO	 
MdAbcProjetoBD	 
MdAbcProjetoRN	 
MdAbcProjetoINT	 
md_abc_projeto_cadastro	 
md_abc_projeto_lista	 