Wikiprint Book

Title: Developer eDokumenty

Subject: eDokumenty - elektroniczny system obiegu dokumentów, workflow i CRM - DeployerGuide/Developer

Version: 118

Date: 01/20/26 19:20:48

Table of Contents

Developer eDokumenty 3
- *Przykładowe wywołanie obiektu Bean dokumentu* 3
- *Wywołanie formularza do potwierdzenie wykonania czynności* 3
- *Wywołanie komendy ze skryptu* 3
- *Jak odświeżyć formatkę po wykonaniu komendy* 3
- *Wywołanie "dymka"* 4
- *Wykorzystanie liczników* 4
- *Jak załatwić etap workflow* 4
- *Komponent ModernSelect* 4
- *Klasa ProjectsDataSet* 7
- *Klasa TextInput* 8
- *Dodanie pola status na pozycji zapotrzebowania* 10
- *Pole daty (CalendarInput)* 11
- *Pole tekstowe typu "Password" na Okienku Custom Widget* 11
- *Lista jednokrotnego wyboru w okienku Custom Widget (z ustawieniem domyślnej wartości)* 11
- *Lookup klientów z labelem* 11
- *Wykorzystanie identyfikatora zalogowanego użytkownika w Custom Widget* 12
- *Zmiana statusu dla wpisu w rejestrze* 12
- *Dodanie wpisu do rejestru* 12
- *Usuwanie wpisu z rejestru* 13
- *Pobranie tekstowej wartości cechy* 13
- *Obsługa Bean'ów poprzez fasadę MapService* 13
- *Otwieranie dialogów za pomocą metody openDialogByCls* 13

Developer eDokumenty

Dostęp do modelu obiektowego systemu eDokumenty:

Przykładowe wywołanie obiektu Bean dokumentu

Przykład pokazuje sposób zmiany statusu (tpstid) oraz oznaczenie dokumentu jako załatwiony:

```
include_once(MOD_PATH.'ADocuments/beans/Document.inc');

$doc_id = 123;
$document = Document::getInstance($doc_id);
if ($document->isReferenced()) {
   $document->set('tpstid', 2);
   $document->set('is_fix', TRUE);
   $document->save();
}
```

Wywołanie formularza do potwierdzenie wykonania czynności

Chodzi o formularz typu confir, który można wykorzystać w mechanizmie CustomWidget:

```
require_once(LIB_PATH.'widgets/ConfirmBox.inc');

// identyfikator sprawdzenia
$hwnd = Application::getShortName(__CLASS__.__LINE__);

// sprawdzenie czy przyszło potwierdzenie i jeśli jest i jest na Nie CONFIRM_NO
if (($confirmation = ConfirmBox::getConfirmation($hwnd)) AND ($confirmation == ConfirmBox::CONFIRM_NO)) {
// tutaj możesz coś zrobić jak users kliknął Nie
return FALSE;
}

// to samo się wywołuje i sprawdza
if (ConfirmBox::confirm($hwnd, Translator::translate('Treść pytania'), NULL, ConfirmBox::CONFIRM_YES|ConfirmBox::CONFIRM_N
// tutaj możesz coś zrobić jak user dał Tak CONFIRM_YES
}
```

```
include_once(MOD_PATH.'ADocuments/beans/Document.inc');

$doc_id = 123;
$document = Document::getInstance($doc_id);
if ($document->isReferenced()) {
   $document->set('tpstid', 2);
   $document->set('is_fix', TRUE);
   $document->save();
}
```

Wywołanie komendy ze skryptu

```
require_once('./commands/AddCommentCommand.inc');
$command = new AddCommentCommand();
$command->execute($bean, array('dscrpt' => 'test'.mktime(), 'notify' => 0));
```

Jak odświeżyć formatkę po wykonaniu komendy

Kod który należy dodać do komendy, by po jej wykonaniu odświeżyła się formatka.

```
JScript::add('App.DOCUMENTdlg'.$doc_id.'.refresh();');
```

Wywołanie "dymka"

Jak wywołać "dymek" informacyjny:

```
JScript::add('NewFrame.showInfo(\'Moja wiadomość\')');
// jeśli ma się pojawić nad konkretnym elementem
JScript::add('BalloonHint.showAbove($(\'id elementu\'), \'Powiadomienie\', \'Treść powiadomienia\',220,null,BHS_LCLOSE,nul
// jeśli nie zadziała metoda add należy użyć
JScript::registerOnLoad('...');
```

Wykorzystanie liczników

Liczniki można wykorzystywać albo do samego generowania numeru, albo do generowania całego złożonego symbolu.

Jak załatwić etap workflow

Metody API można wywołać również na lokalnym systemie:
http://support.edokumenty.eu/trac/wiki/DeployerGuide/Others/eDokumentyApi/CompleteStage

```
require_once('./classes/eDokumentyApi/EDokApiClient.inc')
$api = new EDokApi();
$api->completeStage(1,2);
```

Komponent ModernSelect

```
/** Konstruktor
*  Przygotowuje nazwę dla tagu HTML, źródło requestów,
*  możliwość wielokrotnego wyboru i ustawia tryb pamiętania zawartości.
*  Przywraca ostatnio wybrane pozycje ze schowka.
*  Gdy wybrano tryb pamiętania zawartości przywraca zawartość ze schowka.
*  @param $name nazwa dla tagu HTML
*  @param $source źródło pętli zwrotnej, domyślnie </code>$_POST</code>
*  @param $multiple flaga wielokrotnego wyboru, domyślnie FALSE
*  @param $rememberList tryb pamiętania zawartości, domyślnie FALSE
*/
public function __construct($name, $source = NULL, $multiple = FALSE, $rememberList = NULL);

/** Dodaje parę klucz-etykietę do listy
*  Klucz znajdzie się w tagu <code>option</code>, etykieta
*  będzie wyświetlana jako element listy.
*  @param $k klucz
*  @param $v etykieta
*/
public function addItem($k, $v, $grpDsc = NULL);

/** Dodaje pary klucz-etykieta z tablicy
*  @param $a tablica asocjacyjna
*  @return boolean sukces/porażka
*/
public function addArray($a);

/** Dodaje pary klucz-etykieta z tablicy tablic asocjacyjnych
*  Tablice tablic asocjacyjnych mają postać<code>
*  array(
*          array(klucz => etykieta),
*          array(klucz => etykieta) [...]
```

```
*  );</code>
*  @param &$a referencja do tablicy
*  @param $keyName nazwa pola z kluczem
*  @param $valName nazwa pola z etykietą
*  @return boolean sukces/porażka
*  @since 0.12.0
*/
public function addAssocArray($a, $keyName, $valName, $grpDsc = NULL);

/** Usuwa z listy element o podanym kluczu
*  @param $k klucz
*/
public function removeItem($k);

/** Usuwa z listy wszystkie elementy i natychmiast uaktualnia schowek*/
public function removeAllItems();

/** Reset */
public function reset();

/** Clear */
public function clear();

/** Zaznacza element listy jako wybrany
*  Gdy nie jest dozwolony wybór wielokrotny czyści tablicę wybranych elementów.
*  @param $k klucz wybranego elementu
*/
public function selectItem($k);

/** Zaznacza wszystkie elementy listy jako wybrane */
public function selectAllItems();

/** Zaznacza element listy jako niewybrany
*  @param $k klucz
*/
public function deselectItem($k);

/** Zaznacza wszystkie elementy listy jako niewybrane */
public function deselectAllItems();

/** Przełącza element listy wybrany - niewybrany
*  Elementy wybrane stają sie niewybrane i vice versa.
*  @param $k klucz
*/
public function toggleItem($k);

/** Zaznacza element listy jako wybrana zgodnie z requestem */
public function selectItemOnEvent();

/** Zwraca wartość, która przyszła w requeście
*  @return klucz tagu SELECT
*/
public function getCurrentItem();

/** Set value*/
public function setValue($value);

/** Get value */
public function getValue();

/** Get text*/
```

```
public function getText();

/** Zwraca listę wybranych kluczy */
public function getSelectedKeys();

/** Zwraca referencję do tablicy klucz-etykieta z wybranymi elementami */
public function &getSelectedItems();

/** Zwraca referencję do tablicy klucz-etykieta ze wszystkimi elementami */
public function &getAllItems();

/** Ustawia źródło requestów
*  @param $source nazwa źródła
*/
public function setSource(&$source);

/** Ustawia parametry formatujące HTML tagu SELECT
*  @param $key nazwa parametru
*  @param $value wartość parametru
*/
public function setHtmlFormatting($k, $v);

/** Ustawia parametry CLASS i STYLE tagu HTML
*  @param $class klasa CSS, domyślnie NULL
*  @param $style styl CSS inline, domyślnie NULL
*/
public function setCSSFormatting($class = NULL, $style = NULL);

/** Zaraz po wyborze wysyła formularz javascriptem */
public function autoSubmit();

/** Zaraz po wyborze wysyła formularz gdy wartosc nie pusta */
public function autoSubmitNotNUll();

/** Sprawdza poprawność requestu
*  Sprawdza czy źródło requestów istnieje, znajduje się w nim dana
*  i czy dana jest w liście combo boksa.
*  @return boolean
*/
public function isRequestValid();

/** Ustawia rozmiar dla multi-selecta
*  @param $size wiadomo
*/
public function setSize($size);

/** Set read only */
public function setReadOnly($ro);

/** Enable */
public function enable();

/** Disable */
public function disable();

/** Zwraca flagę read-only całego formularza
*  @return boolean dostępny do zapisu
*/
public function isEnabled();

/** Serializuje widget do HTML i zapamiętuje sesję */
```

```
public function toHtml();
```

Przykład tworzenia obiektu

```
$select = new ModernSelect('myselect');
$select->setCSSFormatting(NULL, 'width:200px; left:10px; top:10px; position:absolute;');
$select->setHTMLFormatting('onchange', 'App.mySelectOnChange(this.value);');

// Napełnianie ręczne
$select->addItem(1, Translator::translate('One'));
$select->addItem(2, Translator::translate('Two'));
$select->addItem(3, Translator::translate('Three'));

// Napełnianie z bazy
$db = PgManager::getInstance();
$rows = $db->select('my_table', 'id____, text__', 'NOT is_del', false, PGSQL_ASSOC);
if (is_array($rows) && !empty($rows)) {
   $select->addAssocArray($rows, 'id____', 'text__');
}

$select->selectItem(2);
$selest->selectItemOnEvent();
```

Klasa ProjectsDataSet

```
/**
* __construct
* Wymagany konstruktor aby lista domyslnych
* parametrow nie byla za dluga dla samej funkcji
*
* Uwaga parametr $setAllLabel i $choiceItem musza miec przemienne wartosci w
* innym przypadku gdy oba beda mialy TRUE priorytet ma $choiceItem
*
* @param boolean $setAllLabel czy ma dodac label -- wszyscy -- z wszystkimi id
* @param boolean $choiceItem  czy ma byc label -- wybierz -- z NULL jako id
* @param boolean $unique      czy gdzy sie powtorza to ma wywalac
* @return void
*/
public function __construct($setAllLabel = TRUE, $choiceItem = FALSE, $unique = FALSE, $allLabelText = NULL, $choiceItemTe

/**
* getDefaultData
*
* @access protected
* @return void
*/
protected function getDefaultData();

/**
* getKeyColumn
* Zwraca nazwe kolumny ktora jest aktualnie wykorzystywania przy
* pobieraniu danych i odpowiada kluczowi.
*
* Kolumna ta nie jest jednoznaczna z ta ktora jest w tablicy wynikowej
* i odpowiada za wartosc klucza (jest nia zawsze id____)
*
* @return string
*/
public function getKeyColumn();
```

```
/**
* getLabelColumn
* Zwraca nazwe kolumny ktora jest odpowiedzialna za wyswietlany
* opis. Kolumny moga byc inne dla kazdej metody zdefiniowanej
* w odpowiedniej klasie.
*
* Kolumna ta nie jest jednoznaczna z ta ktora jest w tablicy wynikowej
* i odpowiada za wartosc labela (jest nia zawsze text__)
*
* @return string
*/
public function getLabelColumn();

/**
* getAllData
* Pobiera dane wedlug standardowej konfiguracji klasy
* Zobacz metode getDefaultData tam sa dane
*
* @access public
* @return array
*/
public function getAllData();

/**
* getCustomData
* Metoda pobiera dane wedlug danej konfiguracji
* Jednak tabela pozostaje taka jaka zdefiniujemy w metodzie getDefaultData
* Zalecane jest aby jednak dopisywac odpowiednia metode do odpowiedniej klasy
* aby nie powielac kodu
*
* @param string $columns    kolumny jakie zostana uzyte z zapytaniu
* @param string $keyColumn  nazwa kolumny ktora bedzie kluczem w selecie
* @param string $textColumn nazwa kolumny ktora bedzie labelem w selecie
* @param string $statement  warunek po jakim ma selectowac jesli wszystko to 1=1
* @param string $orderBy    lista kolumn po ktorych nedzie sortowal np usrnam, firnam
* @return array
*/
public function getCustomData($columns, $keyColumn, $textColumn, $statement, $orderBy, $groupBy = FALSE);

/**
* getSQLQuery
* Zwraca wyłącznie jak wygląda zapytanie o listę danych.
* Przydatne w przypadku komponentu DBSelect.
*
* Data set konfiguruje zapytanie a DBSelect wyświetla ;)
*
* @access public
* @return void
*/
public function getSQLQuery();
```

Klasa TextInput

```
/** Konstruktor
*  Nadaje wartość domyślną, przywraca wartość ze schowka, przechwytuje wartość
*  formularza HTML. Na żądanie waliduje czy pole puste.
*  @param $name nazwa tagu INPUT
*  @param $defaultValue domyślna wartość pola
*  @param $mandatory flaga kontroli obowiązkowego wypełnienia
*/
```

```
function TextInput($name, $defaultValue = NULL, $mandatory = FALSE, $maxLength = NULL, $password = FALSE);

/** Set placeholder */
public function setPlaceholder($text) ;

/** Ustawia maksymalną długość treści pola */
function setMaxLength($maxLength);

/** Ustawia walidator z warunkiem niepustości */
function makeMandatory();

/** Pokazuje element HTML */
function show();

/** Ukrywa element HTML */
function hide();

/** Zwraca stan widoczności elementu HTML */
function isVisible();

/** Ustawia tryb tylko do odczytu elementu HTML
* przywrocone by tswienty potrzebne do disablowania formularzy
*  @param $ro flaga read-only
*/
function setReadOnly($ro);

/** Ustawia element HTML do zapisu */
function enable();

/** Ustawia element HTML tylko do odczytu */
function disable();

/** Zwraca stan ustawienia do zapisu
*  @return boolean dostępny do zapisu
*/
function isEnabled();

/** Zapamiętuje stan elementu */
function store();

/** Przywraca stan elementu */
function restore();

/** Ustawia wartość pola tagu
*  @param $v wartość pola
*/
function setValue($v);

/** Zwraca zawartość pola tagu */
function getValue();

/** Czyści wartość pola tagu i ustawia flagę wyczyszczenia */
function reset();

/** Clear */
function clear();

/** Ustawia klasę i styl CSS tagu HTML
*  @param $class string, klasa CSS
*  @param $style string, styl inline CSS
*/
```

```
function setCSSFormatting($class = NULL, $style = NULL);

/** Ustawia parametry formatujące tag HTML
*  Przykład: <code>setHtmlFormatting('width', '100%')</code>
*  @param $k klucz tagu
*  @param $v wartość klucza
*/
function setHtmlFormatting($k, $v);

/** Zwraca string formatujący tag HTML */
function getHtmlFormatting();

/** Ustawia fokus na elemencie, gdy user przesunie nad nim wskaźnik myszy
*  @since 0.21.5
*/
function setHoverFocus();

/** Ustawia element, który będzie kliknięty po naciśnięciu Enter
*  @since 0.21.5
*/
function setAutoSubmit($buttonName);

/** Ustawia warunek walidacji
*  @param $regExp wyrażenie regularne walidujące
*  @param $errorMsg komunikat błędu, domyślnie symbol wykrzyknika
*/
function setValidator($regExp, $errorMsg);

/** Waliduje
*  @return boolean, wynik walidacji
*/
function isRequestValid();

/** Zwraca komunikat walidatora */
function getErrorMessage();

/** Ustawia komunikat błędu
*  @param $errorMessage komunikat z błędem, domyślnie błąd z walidatora
*  @since 0.21.4
*/
function setErrorMessage($errorMessage = TRUE);

/** Usuwa komunikat błędu
*  @since 0.21.4
*/
function clearErrorMessage();

/** Get name */
function getName();

/** Serializuje do HTML */
function onEnterGoTo($domElement);

/** To HTML */
function toHtml();
```

Dodanie pola status na pozycji zapotrzebowania

```
require_once(LIB_PATH.'forms/DBSelect.inc');

$tpstid = new DBSelect('yourTpstidSelectName');
$tpstid->top = '10px';
$tpstid->left = '70px';
$tpstid->width = '200px';
$tpstid->query = 'SELECT tpstid, dscrpt FROM types_of_processes_states WHERE clsnam = \'FKDEMANDELEMENT\' ORDER BY dscrpt'
$tpstid->update();
```

Pole daty (CalendarInput)

```
require_once(LIB_PATH.'forms/CalendarInput.inc');

/* Wartość domyślna - bieżący czas */
$defaultValue = date('Y-m-d', time());

$adddat = new CalendarInput('yourInputName', $defaultValue);
$adddat->top = '10px';
$adddat->left = '10px';
$adddat->width = '90px';

/* Wersja tylko z datą */
$adddat->dateFormat = 'y-m-d';

/* Wersja z datą i godziną */
$adddat->dateFormat = 'y-m-d h:i';
```

Pole tekstowe typu "Password" na Okienku Custom Widget

```
require_once(LIB_PATH.'forms/PasswordInput.inc');

$passwd = new PasswordInput('yourInputName');
$passwd->setCSSFormatting(NULL, 'width:200px; top:10px;left:120px; position:absolute;');
```

Lista jednokrotnego wyboru w okienku Custom Widget (z ustawieniem domyślnej wartości)

```
require_once(LIB_PATH.'forms/ModernSelect.inc');

$list__ = new ModernSelect('yourInputName');
$list__->setCSSFormatting(NULL, 'width:200px; top:10px;left:120px; position:absolute;');
$list__->addItem({KEY}, {VALUE});
```

Lookup klientów z labelem

```
require_once(LIB_PATH.'forms/Label.inc');
require_once('./classes/LookupWidget/LookupWidget2.inc');
require_once('./classes/LookupWidget/Contact/ContactLookupManager.inc');

$this->lcontid = new Label($this->name.'lcontid');
$this->lcontid->top = '10px';
$this->lcontid->left = '370px';
$this->lcontid->width = '113px';
$this->lcontid->height = '20px';
$this->lcontid->text = CLIENT_NAME.':';

$this->contid = new LookupWidget2($this->name.'contid', new ContactSearchEngine(), FALSE, TRUE);
$this->contid->top = '10px';
```

```
$this->contid->left = '490px';
$this->contid->width = '280px';

// z opcją dodawania nowego plus dodtkowe komponenty
ContactLookupManager::manage($this->contid);
```

Wykorzystanie identyfikatora zalogowanego użytkownika w Custom Widget

Id (usr_id) zalogowanego pracownika kryje się w sesji pod kluczem

```
SysContext::$usr_info['usr_id']
```

Zmiana statusu dla wpisu w rejestrze

```
require_once(MOD_PATH.'CRegisters/beans/CRegisterEntry.inc');

$id____ = id wpisu w rejestrze;
$tpstid = id statusu;

$bean = new CRegisterEntry($id____);

// tego ifa z zawartością można usunąć jeżeli bezwarunkowo chcemy zmienić status.. czyli bez sprawdzania uprawnień
if ($bean->get('tpstid')) {
	$res = $this->db->select('types_of_processes_states', 'status,tpstnm', 'tpstid='.$bean->get('tpstid'), FALSE, PGSQL
	if (is_array($res)) {
		$res = $res[0];
		$stat = $res['status'];
		$desc = $res['tpstnm'];

		$can_change = ((($stat !== 'FINAL') && ($stat !== 'ACCEPTED')) || UserRights::checkSysAcc('bswfms.extras.pr
		if (!$can_change) {
			if (($stat === 'ACCEPTED') && $bean->get('stcuid')) {
				if (!($can_change = UserRights::checkUsrAcc($bean->get('stcuid')))) {
					throw new UserRightsException(NULL, NULL, sprintf(Translator::translate('Zmianę sta
				}
			}
		}
		if (!$can_change) {
			throw new UserRightsException(NULL, NULL, sprintf(Translator::translate('Zmianę statusu %s może dok
		}
	}
}

$bean->set('tpstid', $tpstid);
$bean->save();
```

Dodanie wpisu do rejestru

```
require_once(MOD_PATH.'CRegisters/beans/CRegisterEntry.inc');

$id____ = FALSE; // jeśli edycja wpisu to podajemy id____
$cregid = 1; // podajemy id rejestru z tabli cregisters.register

$bean = new CRegisterEntry($id____, $cregid);

/* Każda kolumna wypełniania jest poprzez $bean->set. Nie musimy wypełnić wszystkich.
* $bean->set('tpstid', 1);
*/
$bean->save();
```

Usuwanie wpisu z rejestru

```
require_once(MOD_PATH.'CRegisters/services/CRegistersService.inc');

// wersja bezpieczna zalecana
$params = array(
'id____' => 1, // lub array(1, 2, 3) 1 wpis albo kolejne wpisy jako tablica
);

$srv = new CRegistersService();
$srv->deleteEntries($params);

// wersja mniej bezpieczna
$id____ = 1; // jeśli edycja wpisu to podajemy id____
$cregid = 1; // podajemy id rejestru z tabli cregisters.register

$bean = new CRegisterEntry($id____, $cregid);
$bean->delete();
```

Pobranie tekstowej wartości cechy

```
require_once('./classes/FeatureBox/FeaturesHelper.inc');
$featureID = 1; // id cechy (features.featid)
$tblnam = 'contacts'; // nazwa tabeli bazowej (kolumna features.tblnam) zazwyczaj jest to nazwa tabeli w bazie, której dot
$tbl_id = 1; // identyfikator obiektu w systemie np. dla klientów jest to contid (contacts.contid)
$value = FeaturesHelper::getTextValue($featureID, $tblnam, $tbl_id);
```

Obsługa Bean'ów poprzez fasadę MapService

```
require_once(LIB_PATH.'util/MapService.inc');

$clsnam = 'CONTACT'; // klasa obiektu aby zobaczyć wszystkie klasy można użyć metody $map = MapService::getMap();
$keyval = 1; // identyfikator danego obiektu jeśli chcemy utworzyć nowy wtedy $keyval = FALSE

$bean = MapService::getBean($clsnam, $keyval);

// ustawienie atrybutu
$bean->set('kolumna', 'wartość');

// zapisanie beana
$id = $bean->save();

// $id - w zależności od klasy oznacza doc_id, prc_id, contid
```

Otwieranie dialogów za pomocą metody openDialogByCls

Tworząc szablony HTML/Flexy, bądź implementując akcje JavaScript w łatwy sposób można posługiwać się dialogami większości obiektów używając metody openDialogByCls.

Definicja:

```
/**
* @param clsnam   Text     Identyfikator klasy obiektu
* @param keyval   Integer  Identyfikator obiektu
* @param op       Text     Dodatkowe parametry w formacie JSON
*/
App.openDialogByCls(clsnam, keyval, op)
```

Przykładowe wywołania:

```
App.openDialogByCls('DOCUMENT',991);
App.openDialogByCls('EMAIL',0,({acntid:1,mode__:'new','to____':'support@edokumenty.eu','subjct':'Hello World!'}).toJSONStr
```

CONTACT	CONTACT
CONTACT_EDIT	CONTACT_EDIT
ADDRESS	ADDRESS
ADDRESSHISTORY	ADDRESSHISTORY
PROCESS	PROCESS
DOCUMENT	DOCUMENT
RCP	RCP
MEETING	MEETING
EVENT_RESOURCE_MANAGE	EVENT_RESOURCE_MANAGE
EVENT	EVENT
TODO	TODO
PHONECALL	PHONECALL
EVENTNOTE	EVENTNOTE
ALARM	ALARM
CORPEVENT	CORPEVENT
FILE	FILE
ATTACHMENT_PREVIEW	ATTACHMENT_PREVIEW
EMAIL	EMAIL
VNCOST	VNCOST
VFOLDER	VFOLDER
P_ACTIVITY	P_ACTIVITY
P_SUBPROCESS	P_SUBPROCESS
P_CONDITION	P_CONDITION
P_DECISION	P_DECISION
P_MERGE	P_MERGE
P_PROPERTY	P_PROPERTY
P_ASSIGNMENT	P_ASSIGNMENT
KNOWLEDGEBASE	KNOWLEDGEBASE
USER	USER
USERPROFILE	USERPROFILE
GROUP	GROUP
ORGANIZATIONUNIT	ORGANIZATIONUNIT
FKPROCESSELEMENT	FKPROCESSELEMENT
FKRCPELEMENT	FKRCPELEMENT
FKDEMANDELEMENT	FKDEMANDELEMENT
FKPZELEMENT	FKPZELEMENT
FKWZELEMENT	FKWZELEMENT

FKORDERELEMENT	FKORDERELEMENT
FKOFFERELEMENT	FKOFFERELEMENT
FKVATNOTEELEMENT	FKVATNOTEELEMENT
FKCUSTOMDOCELEMENT	FKCUSTOMDOCELEMENT
FKPMMELEMENT	FKPMMELEMENT
FKCONTRACTELEMENT	FKCONTRACTELEMENT
FKDOCPAYMENTPOS	FKDOCPAYMENTPOS
ADDRESSBOOK	ADDRESSBOOK
DEVICE	DEVICE
INVENTORYSHEETELEMENT	INVENTORYSHEETELEMENT
ARCHIVEELEMENT	ARCHIVEELEMENT
CAMPAIGN	CAMPAIGN
PRODUCT	PRODUCT
PRODUCTHISTORY	PRODUCTHISTORY
PROJECT	PROJECT
REPORT	REPORT
REPORT_VIEW	REPORT_VIEW
REPORT_JOB	REPORT_JOB
REPORT_QUERY	REPORT_QUERY
REPORTHISTORY	REPORTHISTORY
CONTACTPERSON	CONTACTPERSON
BSCONNECT	BSCONNECT
BSCONNECTDATASOURCE	BSCONNECTDATASOURCE
BSCONNECTOBJECT	BSCONNECTOBJECT
BSCONNECTACTION	BSCONNECTACTION
BSCONNECTHISTORY	BSCONNECTHISTORY
BSCONNECTLOGFILES	BSCONNECTLOGFILES
BSCONNECTLOGFILESDT	BSCONNECTLOGFILESDT
BSCONNECTSUMMARY	BSCONNECTSUMMARY
BSCONNECTCRONLIST	BSCONNECTCRONLIST
BSCONNECTCRON	BSCONNECTCRON
BSCONNECTCRONACTIONLIST	BSCONNECTCRONACTIONLIST
BOOKMARK	BOOKMARK
BOOKMARK_FOLDER	BOOKMARK_FOLDER
DELEGATION_TCOST	DELEGATION_TCOST
CREGISTER	CREGISTER
CREGISTER_ENTRY	CREGISTER_ENTRY
CREGISTER_FIELD	CREGISTER_FIELD
PRICELIST	PRICELIST
EMAILACCOUNT	EMAILACCOUNT