

## Table of Contents

|  |    |
|--|----|
| <i>Developer eDokumenty</i>  | 2  |
| <i>Podstawowy kod/nagłówki dla skryptów typu standalone/front controller</i>                 | 2  |
| <i>Przykładowe wywołanie obiektu Bean dokumentu</i>  | 2  |
| <i>Wywołanie formularza do potwierdzenie wykonania czynności</i>                             | 2  |
| <i>Wywołanie komendy ze skryptu</i>  | 3  |
| <i>Jak odświeżyć formatkę po wykonaniu komendy</i>   | 3  |
| <i>Wywołanie "dymka"</i>   | 3  |
| <i>Wykorzystanie liczników</i>   | 3  |
| <i>Jak załatwić etap workflow</i>  | 3  |
| <i>Komponent DBSelect</i>  | 3  |
| <i>Komponent ModernSelect</i>  | 4  |
| <i>Klasa ProjectsDataSet</i>   | 4  |
| <i>Klasa TextInput</i>   | 6  |
| <i>Dodanie pola status na pozycji zapotrzebowania</i>  | 8  |
| <i>Pole daty (CalendarInput)</i>   | 8  |
| <i>Pole tekstowe typu "Password" na Okienku Custom Widget</i>                                | 8  |
| <i>Lista jednokrotnego wyboru w okienku Custom Widget (z ustawieniem domyślnej wartości)</i> | 8  |
| <i>Lookup klientów z labelem</i>   | 8  |
| <i>Wykorzystanie identyfikatora zalogowanego użytkownika w Custom Widget</i>                 | 9  |
| <i>Zmiana statusu dla wpisu w rejestrze</i>  | 9  |
| <i>Dodanie wpisu do rejestru</i>   | 9  |
| <i>Usuwanie wpisu z rejestru</i>   | 10 |
| <i>Pobranie tekstowej wartości cechy</i>   | 10 |
| <i>Obsługa Bean'ów poprzez fasadę MapService</i>   | 10 |
| <i>Otwieranie dialogów za pomocą metody openDialogByCls</i>                                  | 10 |
| <i>Jak wywołać kółeczko "pracy systemu" (kręcące się kółeczko)</i>                           | 12 |
| <i>Jak wywołać mechanizm wgrywania pliku na serwer</i>                                       | 12 |

## Developer eDokumenty

### Podstawowy kod/nagłówki dla skryptów typu standalone/front controller

```
// poniższy kod zakłada że skrypt znajduje się w apps/edokumenty/scripts/
// aktualnym foderem musi być zawsze apps/edokumenty/
chdir(dirname(__FILE__).'../');
require_once('../../framework/lib/util/Translator/Translator.inc');

include('./config.inc');

error_reporting(ERROR_REPORTING);

define('SCRIPT_PATH', '');
define('ASYNC_SERVICE', false);

include('../../framework/globalVar.inc');
include_once(LIB_PATH.'system/SysContext.inc');

include('./localVar.inc');
include(LIB_PATH.'util/Debug.inc');
include(LIB_PATH.'system/Request.inc');
include_once(LIB_PATH.'system/UserRights.inc');
include(LIB_PATH.'system/CookieSysReg.inc');
include(LIB_PATH.'system/Application.inc');
require_once(LIB_PATH.'system/VarPathService.inc');

...TUTAJ WPISZ WŁAŚCIWY SKRYPT...
```

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($wnd, Translator::translate('Treść pytania'), NULL, ConfirmBox::CONFIRM_YES|ConfirmBox::CONFIRM_NO) == ConfirmBox::CONFIRM_YES) {
// 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,null);');
// 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 EDokApiClient();
$api->completeStage(1,2);

```

### Komponent DBSelect

```

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

```

```

$this->prcref = new DBSelect($this->name.'prcref');
$this->prcref->top = '10px';
$this->prcref->left = '100px';
$this->prcref->width = '300px';
$this->prcref->defaultItemCaption = '-- '.Translator::translate('brak').' --';
$this->prcref->query = 'SELECT prtpid,prtppnm FROM procedures_def WHERE is_del IS NOT TRUE ORDER BY prtppnm ASC';
$this->prcref->update();

```

## Komponent ModernSelect

Przykład tworzenia obiektu

```

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

$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);
$select->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 miały 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, $choiceItemText = NULL)

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

/**

```

```

* getKeyColumn
* Zwraca nazwe kolumny ktora jest aktualnie wykorzystywana 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 $orderBy    lista kolumn po ktorych nedzie sortowal np usrnem, firnam
* @return array
*/
public function getCustomData($columns, $keyColumn, $textColumn, $statement, $orderBy, $groupBy = FALSE);

/**
* getSQLQuery
* Zwraca wyłączenie 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 validator 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
 * przywrócone 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');

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

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

Jeżeli lista ma być zasilana SQLem to używamy komponentu [DBSelect](#). Dla statycznych danych można użyć [ModernSelect](#)

### 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 dodatkowe 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'];

        $scan_change = (((($stat !== 'FINAL') && ($stat !== 'ACCEPTED')) || UserRights::checkSysAcc('bswfms.extras.pr
        if (!$scan_change) {
            if (($stat === 'ACCEPTED') && $bean->get('stcuid')) {
                if (!($scan_change = UserRights::checkUsrAcc($bean->get('stcuid')))) {
                    throw new UserRightsException(NULL, NULL, sprintf(Translator::translate('Zmianę sta
                }
            }
        }
        if (!$scan_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.feaid)
$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');

$classnam = '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

$bean = MapService::getBean($classnam, $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:

```
// Otwarcie dialogu dokumentu o identyfikatorze doc_id = 991
App.openDialogByCls('DOCUMENT',991);

// Otwarcie dialogu nowej wiadomości email z zainicjowanymi polami konta nadawcy, adresata i tematu
App.openDialogByCls('EMAIL',0,({acntid:1,'to___':'support@edokumenty.eu','subjct':'Hello World!'}).toJSONString());
```

Główne klasy dialogów

| Dialog                       | Wartość clsnam | Wartość keyval         | Dodatkowe parametry  |
|------------------------------|----------------|------------------------|--|
| Kontrahent w trybie podglądu | CONTACT        | contacts.contid        |  |
| Kontrahent w trybie edycji   | CONTACT_EDIT   | contacts.contid        |  |
| Osoba kontaktowa             | CONTACTPERSON  | contact_persons.copeid | contid: identyfikator kontrahenta  |
| Sprawa                       | PROCESS        | processes.prc_id       | clsnam: identyfikator typu powiązanego obiektu (DOSS - teczka, PROCESS - sprawa nadrzędna)<br>keyval: identyfikator obiektu powiązanego  |
| Dokument                     | DOCUMENT       | documents.doc_id       | dctpid: identyfikator typu dokumentu   |
| Wiadomość email              | EMAIL          | emails.eml_id          | contid: identyfikator kontrahenta filtrujący lookup wyszukiwania adresata<br>prc_id: identyfikator sprawy z którą zostanie powiązany zarchiwizowany email<br>to___: adresaci w polu Do<br>cc___: adresaci w polu Kopia<br>bcc___: adresaci w polu Ukryta kopia<br>subjct: temat wiadomości<br>body__: treść wiadomości<br>attchm: tablica tablic załączników (0 => ścieżka, 1 => nazwa, 2 => rozmiar)<br>fileid: identyfikator pliku załącznika<br>file_docid: identyfikator dokumentu, z którego pobrane zostaną wszystkie załączniki |
| Spotkanie                    | MEETING        | events.evntid          |  |
| Termin                       | EVENT          | events.evntid          |  |
| Zadanie                      | TODO           | events.evntid          |  |
| Rozmowa telefoniczna         | PHONECALL      | events.evntid          |  |
| Adnotacja                    | EVENTNOTE      | events.evntid          |  |
| Alarm                        | ALARM          | events.evntid          |  |
| Wydarzenie korporacyjne      | CORPEVENT      | events.evntid          |  |

|                    |                    |                                   |   |
|--------------------|--------------------|-----------------------------------|---|
| Podgląd pliku      | ATTACHMENT_PREVIEW | files.fileid                      |   |
| Profil użytkownika | USERPROFILE        | users.usr_id                      |   |
| Książka adresowa   | ADDRESSBOOK        |                                   |   |
| Urządzenie         | DEVICE             | devices.devcid                    |   |
| Kampania           | CAMPAIGN           | campaigns.campid                  |   |
| Produkt            | PRODUCT            | depository.depoid                 |   |
| Projekt            | PROJECT            | projects.projid                   |   |
| Karta RCP          | RCP                | rcp_cards.rcp_id                  |   |
| Podgląd raportu    | REPORT_VIEW        | reports.reports.rep_id            | parametry dla konkretnego raportu<br>np.:<br>FILTER_STRING:'grp_id = 1 AND<br>NOT is_del'<br>DATE_FROM:'2015-01-01' |
| Pozycja rejestru   | CREGISTER_ENTRY    | cregisters.register_entry.id_____ | parametry przykładowa dla nowego<br>wpisu w rejestrze:<br>{cregid:1,cre_id:4}                                       |

### Jak wywołać kółeczko "pracy systemu" (kręcące się kółeczko)

Jeśli z jakis przyczyn dla naszego skryptu to kółeczko się nie pojawi domyślnie trzeba wywołać kod JavaScript: Kręc się:

```
JScript::add('showLoadingDialog();');
```

Koniec kręcenia

```
JScript::add('hideLoadingDialog();');
```

Jeśli wywołanie JScript::add trzeba sprawdzić z JScript::registerOnLoad.

### Jak wywołać mechanizm wgrzywania pliku na serwer

```
// widget do wgrzywania pliku w klasie np MyClassCustomWidget
require_once(LIB_PATH.'widgets/SimpleUploader.inc');
$fileUpload = new SimpleUploader($this->name.'fileUpload');
$fileUpload->left = '102px';
$fileUpload->bottom = '10px';
$fileUpload->width = '140px';
$fileUpload->height = '20px';
$fileUpload->addEventListener(SimpleUploader::ON_UPLOAD_EVENT, $this->path, 'MyClassCustomWidgetUploadDispatcher', 'onFileUpload');
$fileUpload->style = 'display:inline;';
$this->add($fileUpload);

// klasa MyClassCustomWidget musi zawierać metodę setFileData zgodnie z MyClassCustomWidgetUploadDispatcher::onFileUpload

public function onFileUpload($data) {

    $filePath = $data[0];
    $fileName = $data[1];

    // tutaj możemy wykonywać operacje związane z plikiem

}

// druga klasa z tym samym pliku
final class MyClassCustomWidgetUploadDispatcher {
```

```
public function __construct($name) {  
  
    $this->name = $name;  
  
}  
  
public function onFileUpload($fileUploader) {  
  
    if ($uf = $fileUploader->getUploadedFile()) {  
        $o = new MyClassCustomWidgetUploadDispatcher($this->name, FALSE, FALSE, FALSE);  
        $o->setFileData($uf);  
    }  
  
}  
}
```