

Title: Developer eDokumenty

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

Version: 118

Date: 06/07/25 03:07:58

Table of Contents

<i>Developer eDokumenty</i>	3
<i>Jak odświeżyć formatkę po wykonaniu komendy</i>	3
<i>Wykorzystanie liczników</i>	3
<i>Komponent ModernSelect</i>	3

Developer eDokumenty

Dostęp do modelu obiektowego systemu eDokumenty:

Przykładowe wywołanie obiektu Bean dokumentu (zmiana statusu):

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

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

Jak odświeżyć formatkę po wykonaniu komendy

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

Wykorzystanie liczników

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

Komponent ModernSelect

Metody

```
/** 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-etykieta 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 się 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);');

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

$select->selectItem(2);

/**
 * setRightsChecker
 *
 * @param IFieldsRightsChecker $rightsChecker
 * @param string $alias
 * @final
 * @return void
 */
final public function setRightsChecker(IFieldsRightsChecker $rightsChecker, $alias = FALSE);

/**
 * setFieldValidator
 *
 * @param IFieldValidator $fieldValidator
 * @param string $alias
 * @return void
 */
final public function setFieldValidator(IFieldValidator $fieldValidator, $alias = FALSE);

/**
 * getRightsChecker
 *
 * @final
 * @return void
 */
final public function getRightsChecker();

/**
 * getFieldValidator
 *
 * @return void
 */
final public function getFieldValidator();

/**
 * setAlias
 *
 * @param string $alias
 * @return void
 */
final public function setAlias($alias);

/**
 * getAlias
 *
 * @final
 * @return void
 */
final public function getAlias();

```

```
/**
 * isPermitted
 *
 * @param string $alias
 * @return void
 */
final public function isPermitted($alias = FALSE);

/**
 * isRequired
 *
 * @param string $alias
 * @return void
 */
final public function isRequired($alias = FALSE);
```