

**Wikiprint Book**

**Title: Przewodnik wdrożeniowca > Konfiguracja Custom Widgets ...**

**Subject: Archiwum - eDokumenty - elektroniczny system obiegu dokumentów, workflow i CRM -  
DeployerGuide/Others/CustomWidgets**

**Version: 47**

**Date: 06/21/26 18:54:18**

## Table of Contents

<i>Przewodnik wdrożeniowca &gt; Konfiguracja Custom Widgets / Dodatku</i>	3
<i>Wprowadzenie</i>	3
<i>Tworzenie własnego przycisku</i>	3
<i>    Tworzenie skryptu</i>	3
<i>    Przygotowanie grafiki/ikony</i>	3
<i>    Menadżer Custom Widget</i>	3
<i>Wstawienie do pliku xml - CustomModules.xml</i>	4
<i>Uprawnienia</i>	4

## [Przewodnik wdrożeniowca](#) > Konfiguracja Custom Widgets / Dodatku

### Wprowadzenie

W systemie eDokumenty istnieje możliwość definiowania własnych przycisków na pasku tzw. pluginów lub custom widgets - tzw. toolbarze.

### Tworzenie własnego przycisku

Wszelkie informacje odnośnie dodatków są przechowywane w tabeli `custom_widgets`, która zawiera 9 kolumn. Zanim jednak wprowadzimy dane do tabeli należy przygotować skrypt oraz ikonę dla przycisku, który ten skrypt będzie uruchamiał.

### Tworzenie skryptu

Skrypt ma postać pliku z rozszerzeniem `.inc` (np. `Test.inc`). Jest to klasa zapisana przy pomocy języka PHP. Listing przykładowej klasy został umieszczony poniżej:

```
<?php

final class Test {

    public static function init($args) {

        // wyświetla monit eDokumentowy
        JScript::alert(Translator::translate('Witaj to ja Twój plugin'));

        // wyświetla monit JavaScriptowy alert
        JScript::add('alert(666)');

        //Jeśli ma zostać odświeżona lista to na końcu naszego skryptu dodajemy
        if (isset($args['js_after'])) {
            JScript::add($args['js_after']);
        }

    }

}

print_r($args);

Test::init($args);

?>
```

Jak widać powyżej w skrypcie można wykorzystywać także język JavaScript. Jednakże należy mieć na uwadze, że błędny lub szkodliwy skrypt może wpłynąć negatywnie na stabilność systemu !!!

Utworzony skrypt umieszczamy w katalogu `public_html/apps/edokumenty/scripts`

### Przygotowanie grafiki/ikony

Ikona musi mieć rozmiar 24x24px format np. png i być umieszczona w katalogu: `public_html/framework/img/toolbarIcons/24x24/`.

### Menadżer Custom Widget

Przycisku definiowane są w Panel sterowania > Manager Custom Widget. Gdzie pola:

`parametry = {"script":"Test.inc","image":"24x24\ikona.png"}` - parametry dodatku w formacie JSON, w tym:

- `script` to nazwa pliku z katalogu `public_html/apps/edokumenty/scripts`
- `image` - oznacza ikonę z katalogu `public_html/framework/img/toolbarIcons/24x24/ikona.png`. Nazwa ikony musi być poprzedzona

`24x24\nazwa.rozszerzenie`, w tym przykładzie `24x24\ikona.png`.

- `params` - może określać dodatkowe parametry przekazywane do skryptu, np. `"params": "doc_id:{DOC_ID}"` przekaże id dokumentu
- `onClick` - akcja do wywołania po kliknięciu w widget np. `App.openDialogByCls('PROCESS', 14376)` czyli `{ "onClick": "App.openDialogByCls('PROCESS', 14376)"`

}

- `Typ = button|JSMenulItem` - typ: `button` = przycisk, `JSMenulItem` = opcja na menu dotyczy menu dokumentu i sprawy
- `Tekst` - Nazwa na pasku narzędziowym (może być pusta).
- `Pomoc` - Opis, który pokaże się w tooltipie (dymku podpowiedzi)
- `Użycie - c_path` - przyjmuje wartości:
  - `contacts/toolbar` dla modułu Klienci
  - `employees/toolbar` dla modułu Pracownicy
  - `adocuments/toolbar` dla modułu Dokumenty.
  - `aisodocs/toolbar` dla modułu ISO (funkcjonalność dostępna od wersji 5.0.87)
  - `servicemod/toolbar` dla modułu Serwis.
  - `aregisters/toolbar` dla modułu dzienniki.
  - `custom/id_widgetu` dla Custom Widgetu
  - `task_trmtyp(TODO|PHONECALL|MEETING)/toolbar` dla kartoteki zdarzenia o podanym typie (`trmtyp`).
  - `vatnote_costs_list_form/toolbar` - lista pozycji na zakładce koszty w fakturze
  - `document_products_form_{dctpid}/toolbar` - lista pozycji (produkty) na danym dokumencie
  - `process_products_form/toolbar` - lista pozycji (produkty) na sprawie
  - `document_{dctpid}/menu` - menu na dokumencie pod przyciskiem Dokument
  - `process/menu` - menu na sprawie pod przyciskiem Dodatkowe zadania
  - `processes/toolbar` - menu w module Sprawy
  - `product/toolbar` - pasek zadań na kartotece produktu
  - `custom/5` (gdzie 1 jest równe `cswgid` z tabeli `custom_widgets`, czyli trzeba zobaczyć jaka jest największa wartość i wiemy że jak dodamy to nasz `custom_widget` będzie miał id np. 5) - taka konstrukcja pozwala używać później deklaracji w XML w `CustomModule`.
  - `document_travelcosts_{dctpid}/toolbar` - koszty delegacji
  - `cregisterview/toolbar::{{cregids}}` - pasek zadań dla rejestru konstrukcja ścieżki obejmuje 2 definicje: `cregisterview/toolbar` - widget pokaże się na każdym rejestrze lub `cregisterview/toolbar::{{cregids}}` - widget pokaże się na wybranych rejestrach. Zmienna `{cregids}` może być w formie 1 lub 1,2,3 gdzie identyfikator numeryczne to id z `cregisters.register`

Istotą działania modułu dodatkowego jest przekazywanie danych z zaznaczonych elementów pod kluczami

- `contid` dla modułu Klienci
- `doc_id` dla modułu Dokumenty

## Wstawienie do pliku xml - CustomModules.xml

```
<buttons>
  <button>
    <custom_widget>
      3
    </custom_widget>
  </button>
</buttons>
```

## Uprawnienia

Domyślnie każdy użytkownik ma dostęp do utworzonego w ten sposób przycisku. Ograniczenie dostępu jest realizowane poprzez tabelę `access`, której opis jest zamieszczony w artykule: <http://support.edokumenty.eu/trac/wiki/DeployerGuide/Others/SettingRightsForFields>

Dane do ustawiania uprawnień dla przycisku:

- `clsnam` = `CUSTOM_WIDGET`
- `keyval` = `custom_widgets.cswgid`