Wikiprint Book

Title: Automatyzacja procesów workflow

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

Version: 133

Date: 12/06/25 05:54:47

Table of Contents

Automatyzacja procesów workflow 3
- *Podstawowe informacje* 3
- *API komend workflow* 3
- *Dla zaawansowanych* 6
- *Wykorzystanie własności, danych wejściowych i przypisań* 7
- *Trochę teorii* 7

Automatyzacja procesów workflow

Podstawowe informacje

Procedury workflow oparte są o notację BPMN i uwzględniają wszystkie najważniejsze elementy tej notacji. Składają się na nią:

- Etapy (Czynności - bloczki)
- Przejścia (strzałki)
- Decyzje (diament powodujący wyświetlenie decyzji dla użytkownika)
- Warunki (diament dokonujący ewaluacji warunków SQL)
- Złączenia (JOIN - w przypadku wymagania spełnienia poprzednich etapów)

Konfiguracja procedur pozwala tworzyć mapy procesów odnoszące się zarówno do dokumentów jak i spraw. Przykłady wykorzystania dostępne są tutaj: Wykorzystanie procedur

API komend workflow

W akcjach etapów można używać komend które będą wykonane w czasie aktywacji danego etapu. Komendy wybiera się z listy wyboru określając dodatkowe parametry np.

```
target = "20",dscrpt="Wezwanie, uwaga!"
status = "4",controlQuery="SELECT status = 3 FROM processes WHERE prc_id=$prc_id"
```

Nie zaleca się na obecnym etapie stosowania znaków specjalnych w wartościach parametrów, między nawiasami powinna się znaleść wartośc liczbowa ("5") lub tekstowa bez znaków specjalnych ("Wezwanie do wykonania etapy")

Przekaż dokument

Komenda służy do automatycznego przekazywania dokumentu na wybrane stanowiska.

Parametry:

- to = "1" - parametr wskazujący do kogo ma zostać przekazany oryginał , jeśli parametru nie będzie, lub będzie pusty oryginał zostaje.
- dw = "2,3,4,5" - do wiadomości
- udw = "6,7,8" - ukryte do wiadomości

Wszystkie wartości w parametrach to orunid z widoku orgtree_view.

Sprawdź czy pole jest wypełnione

Komenda służy do sprawdzania czy dane pole formularza jest wypełnione. Przyjmuje 2 parametry i oba są wymagane.

Parametry:

- field = "featid|8" lub "symbol" - pole które ma sprawdzić
- alert = "Wypełnij pole symbol" - wiadomość w przypadku pustej wartości w polu

Sprawdź prawdziwość warunku SQL

Komenda służy do sprawdzania warunku SQL.

Parametry:

- query = "SELECT cena IS NOT NULL FROM table WHERE prc_id = {PKEYVALUE}" - zapytanie SQL
- alert = "Wypełnij pole cena" - wiadomość w przypadku niespełnienia sql
- success = "Pole cena wypełnione" - wiadomość w przypadku spełnienia sql

Utwórz zdarzenie dla pracownika

Komenda służy do utworzenia zdarzenia określonego typu (domyślnie zadanie) dla pracownika.

Parametry:

- dscrpt = "Aktualizacja dokumentacji wdrożenia"
- emp_id = "{LOGGED_USER}" (lub usr_id z tabeli users)
- trmtyp = "TODO" - typ zdarzenia, może być do wyboru MEETING, PHONECALL, ALARM
- start_ = "featid|81"; - na kiedy ma dodać zdarzenie, wartość jest pobierana z cechy dlatego definicja np featid|81
- end___ = "featid|81"; - data zakończenia
- interval = "+ 4 days" - interval za jaki utworzyć się zdarzenie, można tworzyć wstecz poprzez -
- ptstid = "2" - id etapu z tabeli stages_def lub (ptstid="next") dla następnego etapu (ten parametr nie jest wymagany)

Parametry dscrpt i emp_id są wymagane.

Sprawdź czy istnieje dokument

Komenda służy do sprawdzania czy w danej sprawie występuje konkretny dokument.

Parametry:

- dctpid = "9" - id typu dokumentu ze słownika (Ustawienia -> Panel sterowania -> Dokumenty -> Typy dokumentów -> kolumna id)
- state_ = "2" - identyfikator rodzaju dokumentu 1 - wychodzący, 2 - przychodzący 3 - wewnętrzny

Zmiana statusu

Komenda zmienia status zlecenia (sprawy/dokumentu) na podany w parametrze.

Parametry:

- tpstid = "9" - id statusu ze słownika statusów dla odpowiedniej klasy (Ustawienia -> Panel sterowania -> Ogólne -> Statusy -> kolumna id)

Uprawnij grupę pracowników do sprawy

Komenda dodaje wskazaną grupę jako uprawnioną w sprawie z zadeklarowaną maską. Jeśli grupa została dodana już wcześniej komenda jest pomijana.

Parametry:

- grpnam = "Pracownicy" - nazwa grupy która zostania przydzielona do sprawy (Pracownicy -> Grupy)
- attrib = "rwnd" - maska uprawnień

Znaczenie poszczególnych flag maski uprawnień:

- r - odczyt
- w - zapis zadań i dokumentów
- d - oglądanie wszystkich dokumentów
- n - powiadamianie o nowych dokumentach, zadaniach i komentarzach
- m - zarządzanie (karta ogólne i Uprawnienia)

Utwórz dokument

Komenda ta tworzy dokument określonego typu. Komenda przyjmuje następujące parametry:

Parametry:

- dctpid = "5" - id typu dokumenty (Ustawienia -> Panel sterowania -> Typy dokumentów -> kolumna ID
- dctptp = "Note" - nazwa klasy typu dokumentu (dctptp z tabeli types_of_documents) parametr zamienny z dctpid
- dscrpt = "Wezwanie do wykonania etapy procedury" - treść dokumentu
- do = "1" - id stanowiska na jakie zostanie przekazy dokument (orunid z tabeli organization_units)
- dw = "2,3,4" - (dw = do wiadomości) jeden lub więcej id (rozdzielone znakiem interpunkcyjnym, zwanym przecinkiem) jednostki do której przekazać kopie dokumentu (orunid z tabeli organization_units)

Utwórz przypomnienie

Komenda tworzy zdarzenie typu przypomnienie o danej procedurze/etapie dla wskazanego pracownika.

Parametry:

- usr_id = "2" - dla kogo zostanie utworzone przypomnienie (Pracownicy -> Konta pracowników -> kolumna id)
- start_ = "SQL::SELECT now()" - na kiedy ma ustawić przypomnienie
- dscrpt = "Przypomnienie o zatwierdzeniu etapu" - opis przypomnienia;

Przekaż dokument do przełożonego

Komenda służy do automatycznego przekazywania dokumentu do przełożonego pracownika wskazanego w parametrze emp_id.

Parametry:

- emp_id = "{LOGGED_USER}" - id użytkownika którego przełożony otrzyma dokument domyślnie id zalogowanego (usr_id z tabeli users)

Zarejestruj dokument

Komenda służy do automatycznej rejestracji dokumentu według ustalonej procedury.

Parametry:

- reg_id = "2" - id rejestru z tabeli registers w którym zostanie zarejestrowany dokument - używany wówczas gdy z góry znamy właściwy dziennik. Opcjonalnie można użyć innych parametrów, wówczas system obliczy właściwy dziennik (np. wtedy kiedy dokumenty mogą być rejestrowane w różnych dziennikach - np. jednostek rozliczeniowych)

Opcjonalnie można użyć innych parametrów, wówczas system obliczy właściwy dziennik (np. wtedy kiedy dokumenty mogą być rejestrowane w różnych dziennikach - np. jednostek rozliczeniowych):

- orunid = "{acorid}" - id jednostki organizacyjnej po której zostanie wyszukany dziennik (tabela registers kolumna orunid), w przypadku jeśli chcemy aby wartość orunid była pobrana bezpośrednio z formularza np pole jednostka rozliczeniowa na formularzu faktury parametr ten powinien wyglądać w następujący sposób orunid="{acorid}"
- regtyp = "RegOfVatNotes" - typ rejestru w danym orunid (kolumna regtyp z tabeli registers)
- type__ = "1" - typ rejestru (1 = wychodzący, 2 = przychodzący, 3 = wewnętrzny)

Utwórz załącznik z szablonu

Komenda automatycznie tworzy załącznik do dokumentu na podstawie podanego id szablonu (tabela templates kolumna tpl_id).

Parametry:

- tpl_id = "23" - id szablonu z tabeli templates, parametr jest wymagany

Wysyłanie powiadomienia

Komenda pozwala na wysłanie powiadomienia na zadany sposób.

Parametry:

- dscrpt = "Zebranie zarządu" - treść powiadomienia
- grp_id = "2" - id grupy, której pracownicy otrzymają powiadomienie (kolumna grp_id z tabeli groups)
- grpnam = "Zarząd" - zamienny parametr do grp_id, nazwa grupy (kolumna grpnam z tabeli groups)
- usr_id = "2" - id użytkownika (users.usr_id) do którego zostanie wysłane powiadomienie
- type__ = "Communicator" - rodzaj powiadomienia, obecnie wspierane są Communicator - wewnętrzny komunikator, Document - notatka służbowa, Mail - powiadomienie zostanie wysłane na adres mailowy zapisany w kartotece pracownika

Aby komenda zadziałała musi być podany jeden z 3 parametrów: grp_id, grpnam lub usr_id. Mogą być podane wszystkie parametry jednak wtedy priorytet ma grp_id przed grpnam.

Wyślij wiadomość email

Komenda pozwala na wysłanie wiadomości email do określonych odbiorców. Dodatkowo istnieje możliwość automatycznego załączenia załączników dokumentu do emaila.

Parametry:

- from__ = "SQL::SELECT email FROM users WHERE usr_id = {LOGGEG_USER}" - pole od kogo domyślnie jest wstawiane eDokumenty <wartość pola from__ z tabeli smtp_configuration>
- to____ = "prezes@…" - pole do w przypadku parametru liczbowego zostanie wstawiony adres email kontaktu o identyfikatorze podanym w parametrze np to____="{contid}"
- cc____ = "kierownik@…" - pole kopia w przypadku parametru liczbowego zostanie wstawiony adres email kontaktu o identyfikatorze podanym w parametrze np cc____ = "featid::98"
- bcc___ = "archiwumfirma.eu" - pole kopia ukryta w przypadku parametru liczbowego zostanie wstawiony adres email kontaktu o identyfikatorze podanym w parametrze np bcc___="{contid}"
- subjct = "Rekrutacja zakończona" - temat wiadomości zwykły tekst bez znaczników html. Pole jest wymagane.
- body__ = "Zakończono proces rekrutacji. W załączniku dostępne są wyniki" - treść wiadomości zwykły tekst bez znaczników html. Pole jest wymagane.
- tpl_id = "1" - identyfikator szablonu (templates.tpl_id). Jeśli szablon jest typu html parametr body zostanie pominięty a treścią maila będzie wygenerowany dokument z szablonu. W przypadku szablonu typu RTF zostanie on dodany jako załącznik - parametr body nie zostanie pominięty.
- attach = "1" - flaga oznaczająca czy do wysyłanej wiadomości mają zostać dołączone wszystkie załączniki jakie są w dokumencie. Działa tylko w przypadku procedury przypisanej do dokumentu.

Spośród parametrów to____, cc____ oraz bcc___ wystarczy aby tylko jeden był podany aby wiadomość została wysłana.

Informacja dodatkowa

Wszystkie parametry maja możliwość pobierania wartości w następujący sposób:

- z cechy (nie dotyczy cech z list wyboru definiowanych przez użytkownika)
  - emp_id = "featid|81"
  - emp_id = "featid::81"
- bezpośrednio z otwartego formularza
  - emp_id = "{rspuid}" gdzie wartość {rspuid} zostanie zamieniona na wartość w kolumnie (polu) odpowiedniej tabeli - dla formularza dokumentu documents (plus dodatkowe tabele) dla sprawy processes.

W podanym przypadku pole rspuid (osoba odpowiedzialna w sprawie) jest wartością z kolumny rspuid z tabeli processes.

- jako wynik zaptania SQL
  - emp_id = "SQL::SELECT usr_id FROM users WHERE usr_id = featid|81"
  - emp_id = "SQL::SELECT usr_id FROM users WHERE usr_id = featid::81"
  - emp_id = "SQL::SELECT usr_id FROM users WHERE usr_id = {rspuid}"
- predefiniowane parametry
  - {LOGGED_USER} - id zalogowanego użytkownika
  - {ENT_ID} - id jednostki na której pracuje użytkownik
  - {PKEYVALUE} - wartość klucza głównego (id dokumentu/sprawy) patrz klucz główny tabeli documents/processes.

Predefiniowane parametry można używać w następujący sposób

- emp_id = "SQL::SELECT rspuid FROM processes WHERE prc_id = {PKEYVALUE}"
- emp_id = "{LOGGED_USER}"

Sposób definicji parametrów można łączyć np.:

- emp_id = "SQL::SELECT usr_id FROM users WHERE usrnam = featid:81 AND adddat > '{adddat}'::timestamp AND usr_id != {LOGGED_USER} AND ent_id = {ENT_ID}"

lub można wykonywać działania (przykład teoretyczny nigdzie nie występuje taki parametr ;))

- netto_ = "SQL::SELECT {brutto}::int * featid::89::int"

Dodatkowo dla dokumentów różnych typów można podawać nazwy kolumn z dodatkowych tabel np dla faktur tabela vatnote itd.

Dla zaawansowanych

W workflow biorą udział następujące tabele:

- procedures_def - tabela procedur - przechowuje informacje o procedurze np. Zatwierdzenie faktury kosztowej
- stages_def - tabela etapów - przechowuje definicje poszczególnych etapów np. Akceptacja Prezesa
- stages - instancje etapów - przechowuje informacje o zapisanych etapach konkretnych procesów: spraw, dokumentów
- proc_actions - akcje powiązane z procedurami lub z etapami, wykonują się przed lub po zapisie np. beforeStageChange
- action_commands - komendy wykonywane przez system na akcjach - wybierane spośród zawartych w katalogu commands - można dodać parametry, które dodają się do standardowych dwóch Obiektu Akcji oraz obiektu encji powiązanej z wykonywaną akcją np. Dokument albo Sprawa

Wykorzystanie własności, danych wejściowych i przypisań

Własności służą do zdefiniowania dodatkowych atrybutów procedury - można je traktować jako zmienne procedury.

Dane wejściowe służą tym samym czym odczyt standardowego wejścia w konsoli czy programie (czyli pobraniu od użytkownika znaków)

Przypisania służą nadaniu wartości dla zmiennych procedury jak również nadaniu wartości atrybutom etapu którego dotyczą.

- Tworzymy etap w którym ustalimy Opiekuna (np. Określenie opiekuna)
- W zakładce Dane wejściowe dodajemy element typu array o nazwie "Opiekun". W efekcie tego w etapie pojawia się link monitujący "Edytuj dane"
- W etapie który chcemy zlecić Opiekunowi dodajemy Przypisanie: własność: {stages.orgarr} (czyli tablica wykonujących zadanie workflow) wyrażenie: {procedures.Opiekun}

Przypisanie też możemy użyć bez konieczności pobierania danych od użytkownika, możemy je pobrać z bazy danych. Dla tego przykładu gdybyśmy chcieli pobrać Opiekuna klienta którego dotyczy sprzedaż dodalibyśmy Przypisanie z własnością: {stages.orgarr} i wyrażeniem:

```
SELECT o.orunid FROM contacts c  JOIN processes USING(contid) JOIN orgtree_view o ON o.usr_id = c.macrtk WHERE prc_id = {
```

Trochę teorii

Tworzenie prostych procesów workflow nie wymaga dużego przygotowania, ale do tworzenia bardziej zaawansowanych modeli konieczna jest minimalna znajomość teoretycznych zasad rządzących przepływem procesów.

Podstawy teoretyczne