SAREhub Panel Documentation

Wydanie 1.0.0

Digitree Group S.A.

22 sty 2020

Spis treści

ROZDZIAŁ 1

System

Definicja systemu jest podstawową częścią integracji i stanowi wejście dla reszty konfiguracji w pliku *system.json*:

Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator systemu	TAK	Maksymalnie 32 znaki **A-Z, a-z, 0-9, -, _**
name	Nazwa systemu wyświetlana w panelu integracji	TAK	Maksymalnie 255 znaków
logo	Url do loga wyświetlanego w panelu integracji	NIE	Maksymalnie 255 znaków
description	Opis systemu	NIE	
version	Wersja konfiguracji integracji	TAK	Wersja w formacie Major.Minor.Patch lub Major.Minor
authorization	Obiekt zawierający konfigurację autoryzacji integracji	TAK	Opis szczegółowy poniżej
vars	Obiekt zawierający globalne wartości do użycia w konfiguracji	NIE	Mogą zostać nadpisane indywidualnie dla klienta w systemie.
modules	Tablica zawierająca definicje modułów	TAK	Lista definicji *modułów*

Przykład:

```
{
  "id": "id_systemu",
  "name": "Nazwa systemu",
  "logo": "test.pl/logo.png",
  "description": "Opis systemu",
  "version": "1.0.0",
  "authorization" : {"...": "..."},
  "vars": {
    "var1": "test val",
    "var2": 1
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
  },
  "modules": []
}
```

1.1 Autoryzacja systemu - authorization

Konfiguracja określa kilka możliwości integracji i autoryzacji systemu zewnętrznego.

Parametr	Opis	Wymagane	Uwagi
type	Określenie typu	TAK	Typy opisane poniżej
config	Dodatkowa konfiguracja dla typu autoryzacji	NIE	Zależnie od typu
restrictions	Ustawienia ograniczeń dla integracji	NIE	Opisane poniżej

```
{
  "type": "authorization_type",
  "config": {},
  "restrictions": {}
}
```

1.1.1 Typy autoryzacji - type

- Brak - none

Przy typie *none* użytkownik nie może sam wykonać integracji systemu w jego koncie i musi być uruchomiona ręcznie przez administratora lub automatycznie przy włączeniu integracji. Pole *config* jest ignorowane.

- OAUTH 2 - oauth

Pozwala na autoryzację systemu SAREhub z integrowanym systemem którą może wykonać użytkownik systemu. Wykorzystuje protokół oauth2. Konfiguracja typu **MUSI** zawierać następujące parametry:

Parametr	Opis	Wymagane	Uwagi
clientId	Id określone w konfiguracji serwera OAUTH	TAK	Musi byc zgodne z serwerem OAUTH
clientSecret	Secret określony w konfiguracji serwera OAUTH	TAK	Musi byc zgodne z serwerem OAUTH
urlAuthorize	Url pod którym uruchamiana jest autoryzacja(panel logowania)	TAK	
urlAccessToken	Url pod którym SAREhub może pobrać nowe tokeny	TAK	
urlResourceOwnerDetails	Url pod którym SAREhub może pobrać dane konta z którym jest wykonywana integracja	TAK	
provider	Określenie silnika integracji oauth	NIE	Domyślnie *sarehub_generic* lub nieokreślone, chyba, że wskazano inaczej

Przykład:

```
{
  "clientId": "idKlientaOauth",
  "clientSecret": "secretKlientaOauth",
  "urlAuthorize": "https://system.pl/oauth/authorize",
  "urlAccessToken": "https://system.pl/oauth/token",
  "urlResourceOwnerDetails": "https://system.pl/oauth/account"
}
```

1.1.2 Restrykcje dla integracji - restrictions

Pozwala na wskazanie ograniczeń dla danej integracji

Parametr	Opis	Wymagane	Uwagi
count	Ograniczenie liczby integracji jakie może wykonać użytkownik z integrowanym systemem	Nie	

Przykład:

```
{
  "count": 1
}
```

ROZDZIAŁ 2

Moduły

Każdy system **MUSI** posiadać definicję dla co najmniej jednego modułu.

Parametr	Opis	Wymagane	Uwagi
name	Unikalny nazwa modułu	TAK	Maksymalnie 32 znaki w formacie A-Z, a-z, 0-9, -, _
label	Nazwa modułu wyświetlana w panelu	TAK	Maksymalnie 255 znaków
interface_config	Obiekt określający zmienne do wykorzystania przy budowaniu formularza	NIE	Opis szczegółowy poniżej
event	Obiekt zawierający konfigurację obsługi zdarzeń	NIE	Opis szczegółowy poniżej
functionality	Obiekt zawierający konfigurację funkcjonalności	TAK	Lista definicji *funkcjonalności*
save	Obiekt zawierający konfigurację zapisu modułu	TAK	Opis szczegółowy poniżej
translation	Obiekt zawierający zbiór tłumaczeń używanych w module	TAK	Opis szczegółowy poniżej

Przykład:

```
{
  "name": "nazwa_modułu",
  "label": "Wyświetlana nazwa modułu",
  "interface_config": {},
  "event" : {},
  "functionality": {},
  "save": {},
  "translation": {}
}
```

2.1 Definicja zmiennych - interface_config

Służy głównie do zdefiniowania zmiennych które mają zostać wstawione w definicję formularza funkcjonalności i umożliwia pobranie zasobu z integrowanego systemu.

Parametr	Opis	Wymagane	Uwagi
apiGlobals	Lista zmiennych to zastąpienia w definicjach zmiennych **apiVariable**	NIE	Tablica gdzie wartością jest nazwa zmiennej w formacie **A-Z, _**, a wartoscią pola jest wartość stałej
apiVariable	Lista zmiennych to zastąpienia w definicjach formularzy funkcjonalności	NIE	Tablica gdzie wartością jest nazwa zmiennej w formacie **A-Z, _**, a wartoscią definicja zapytania do zewnętrznego API
prependApiVariable	Pozwala na wstawienie wartości do zmiennych z **apiVariable** przed tymi pobranymi z API	NIE	Tablica gdzie wartością jest nazwa zmiennej w formacie **A-Z, _**
appendApiVariable	Pozwala na wstawienie wartości do zmiennych z **apiVariable** za tymi pobranymi z API	NIE	Tablica gdzie wartością jest nazwa zmiennej w formacie **A-Z, _**

2.1.1 Definicja zapytania - apiVariable

Parametry definicji:

Parametr	Opis	Wymagane	Uwagi
url	Url do zasobu	TAK	
method	Metoda HTTP	TAK	GET, POST itp.
converter	Metoda parsująca odpowiedz z API	NIE	Wymaga ustalenia z zespołem SAREhub
options	Obiekt opcji zapytania	TAK	Opis szczegółowy poniżej

Opcje zapytania:

Parametr	Opis	Wymagane	Uwagi
headers	Tablica nagłówków zapytania	NIE	
form_params	Tablica parametrów wysłana w żądaniu POST	NIE	Tablica gdzie kluczem jest nazwa parametru, a wartością wartość parametru

Przykład:

```
{
  "...": "...",
  "VAR_MODULE": {
    "url": "http://test.pl/api/test",
    "method": "POST",
    "converter": "convertVarModule",
    "options": {
      "headers": [
        "application/x-www-form-urlencoded"
      ],
      "form_params": {
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
            "param1": "param2"
        }
    }
  },
  "...": "..."
}
```

2.2 Definicja zdarzeń - event

Pole posiada tylko jeden parametr - *listeners*, który pozwala na określenie listy wywołań funkcji w momencie wywołania przez system SAREhub określonych zdarzeń.

Parametry definicji wywołania:

Parametr	Opis	Wymagane	Uwagi
type	Nazwa funkcji do wywołania	TAK	Typy opisane poniżej
config	Dodatkowa konfiguracja wywołania	TAK	

2.2.1 Dostępne funkcje nasłuchujące

apiCallListener

Funkcja pozwala na wywołanie metody API.

Parametry konfiguracji:

Parametr	Opis	Wymagane	Uwagi
client	Identyfikator klienta Guzzle	TAK	
request	Konfiguracja żądania do API	TAK	

Parametry konfiguracji żądania

Parametr	Opis	Wymagane	Uwagi
method	Metoda HTTP	TAK	GET, POST itp.
url	Url do zasobu	TAK	
options	Opcję żądania	TAK	Opcję zgodne z Guzzle.

Dostępne zdarzenia

campaign.started

Zdarzenie wyłowywane gdy zostanie uruchomiona kampania.

Przykład:

```
{
  "event": {
      "listeners": [
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
        {"...": "..."},
        {
          "type": "apiCallListener",
          "config": {
            "client": "test_api",
            "request": {
              "method": "POST",
              "url": "resource/1",
              "options": {
                "name": "test"
              }
            }
            "event": [
              {
                "name": "campaign.started",
              }
            ]
          }
        },
        {"...": "..."},
      ]
    }
}
```

entityChangedListener

Funkcja pozwala na przeładowanie kampanii po wywołaniu zdarzenia.

Parametry konfiguracji:

Parametr	Opis	Wymagane	Uwagi
functionality	Lista funkcjonalności dla których ma być aktywne wywołanie	TAK	
event	Lista zdarzeń do nasłuchiwania	TAK	

Dostępne zdarzenia

entity.changed

Zdarzenie wyłowywane gdy zostanie zaktualizowana określona encja.

Konfiguracja zdarzenia:

Parametr	Opis	Wymagane	Uwagi
entity	Lista encji przy których ma zostac wywołana funkcja	TAK	Tablica gdzie kluczem jest nazwa encji, a wartoscią jest tablica identyfikatorów pól z definicji formularza funkcjonalności

Lista encji:

Encja	Opis
testGroup	Grupy testowe, należy wstawić null jako pierwszy element
notificationsTemplate	Szablon powiadomienia webpush
personalizationTemplate	Szablon personalizacji

Przykład:

```
{
  "event": {
      "listeners": [
        {"...": "..."},
        {
          "type": "entityChangedListener",
          "config": {
            "functionality": [
              "notifications"
            ],
            "event": [
              {
                "name": "entity.changed",
                "config": {
                  "entity": {
                    "testGroup": [
                      null
                    ],
                    "notificationsTemplate": [
                      "notification_template_id"
                    ],
                    "personalizationTemplate": [
                      "web_push_personalization"
                    ]
                  }
                }
              }
            ]
          }
        },
        {"...": "..."},
      ]
    }
}
```

integrationIdentityListener

Funkcja pozwala na modyfikację identyfikatorów SAREweb

Parametry konfiguracji:

Parametr	Opis	Wymagane	Uwagi
event	Lista zdarzeń do nasłuchiwania	TAK	

Dostępne zdarzenia

integration.completed

Zdarzenie wyłowywane po poprawnej integracji systemu.

Przykład:

```
{
  "event": {
      "listeners": [
        {"...": "..."},
        {
          "type": "integrationIdentityListener",
          "config": {
            "event": [
              {
                "name": "entity.changed",
                "config": {}
              }
            ]
          }
        },
        {"...": "..."},
      ]
    }
}
```

integration.deleted

Zdarzenie wyłowywane po poprawnym usunięciu integracji.

Przykład:

```
{
  "event": {
      "listeners": [
        {"...": "..."},
        {
          "type": "integrationIdentityListener",
          "config": {
            "event": [
              {
                "name": "entity.deleted",
                "config": {}
              }
            ]
          }
        },
        {"...": "..."},
      ]
    }
}
```

2.3 Zapis modułu - save

Służy do określenia w jaki sposób konfiguracja modułu ma być zapisana do wykożystania w flow kampanii. Zapis modułu wywoływany jest w momencie uruchomienia kampanii.

Parametr	Opis	Wymagane	Uwagi
type	Typ obsługi zapisu	TAK	Opisane poniżej
config	Konfiguracja zapisu	TAK	Zależnie od typu

2.3.1 Wywołanie API - api-call

Pozwala na zapis konfiguracji modulu w określonym API.

Parametry konfiguracji:

Parametr	Opis	Wymagane	Uwagi
client	Identyfikator klienta Guzzle	TAK	
request	Konfiguracja żądania do API	TAK	

Parametry konfiguracji żądania

Parametr	Opis	Wymagane	Uwagi
method	Metoda HTTP	TAK	GET, POST itp.
url	Url do zasobu	TAK	
options	Opcję żądania	TAK	Opcję zgodne z Guzzle.

Przykład:

```
{
  "save": {
    "type": "api-call",
    "config": {
      "client": "test_client",
      "request": {
        "method": "POST",
        "url": "resource/1",
        "options": {
          "json": {
            "name": "test"
          }
        }
      }
    }
  }
}
```

2.3.2 Zapis zależny od konfiguracji - multi

Pozwala na określenie konfiguracji zależnie od funkcjonalności:

Przykład:

```
{
  "save": {
    "type": "multi",
    "config": {
      "functionality": {
        "func1": {
          "type": "api-call",
          "config": {}
        },
        "func2": {
          "type": "mongo-api",
          "config": {}
        },
      }
    }
  }
}
```

2.3.3 Zapis do MongoDB - mongo-api

Pozwala na zapis konfiguracji modulu w bazie MongoDB. Typ używany głównie w wewnętrznych modułach systemu SAREhub.

Parametry konfiguracji:

Parametr	Opis	Wymagane	Uwagi
module	Nazwa modułu	TAK	
rule	Reguła w formacie JSON	TAK	Reguły sa łączone dla modułu

Przykład:

```
{
  "save": {
    "type": "mongo-api",
    "config": {
      "module": "test_module",
      "rule": {
        "event": "test",
        "condition": "test"
      }
    }
  }
}
```

2.4 Tłumaczenia - translation

Definicja tłumaczeń do użycia w konfiguracji formularzy.

Przykład:

```
{
  "translation": {
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
    "pl": {
      "name": "nazwa",
    }
  }
}
```

Tłumaczenia w formularzach można użyć poprzez dyrektywę %TRANSLATE|klucz_w_translacji%.

Przykład:

```
{
  "label": "%TRANSLATE|name"
}
```

ROZDZIAŁ 3

Funkcjonalność

Każdy moduł MOŻE zawierać jedną lub kilka funkcjonalności. Funkcjonalność stanowi najbardziej podstawą cześć modułu systemu i składa sie głównie z „bloczka” w schemacie kampanii i formularza.

Parametr	Opis	Wymagane	Uwagi
label	Nazwa funkcjonalności wyświetlana w systemie	TAK	
config	Konfiguracja funkcjonalności	TAK	
node	Konfiguracja bloczka wyświetlanego w schemacie	TAK	
form	Konfiguracja formularza bloczka	TAK	

3.1 Definicja bloczka - node

Parametr	Opis	Wymagane	Uwagi
icon	Ikona wyświetlana na bloczku	NIE	
tab	Kategoria bloczka	TAK	source - trigger, analitics - filtry, executable - akcje
endpoints	Lista wejść i wyjść bloczka	TAK	

3.1.1 Lista wejść i wyjść - endpoints

Definicja wejść i wyjść bloczka funkcjonalności.

Parametr	Opis	Wymagane	Uwagi
source	Lista wyjść z bloczka	TAK	
target	Lista wejść do bloczka	TAK	

Konfiguracja wejścia/wyjścia składa się z następujących parametrów:

Parametr	Opis	Wymagane	Uwagi
typ	Typ wejścia/wyjścia	TAK	
params	Obiekt dodatkowych opcji WE/WY	TAK	

Typy wejść

- universal

Podstawowy typ wejścia używany w wszystkich bloczkach

Typy wyjść

- universal

Podstawowy typ wejścia, używany gdy potrzeba tylko jednego wyjścia z bloczka.

- plus

Typ wyjścia dla spełnienia warunku. Używany głównie w filtrach.

- minus

Typ wyjścia dla niespełnienia warunku. Używany głównie w filtrach.

Dodatkowe opcje

Każde WE/WY bloczka może posiadać następujące dodatkowe opcję

Parametr	Opis	Wymagane	Uwagi
maxConnections	Maksymalna liczba połączeń	NIE	Domyślnie 1. Aby wyłączyć limit należy wstawić -1

Przykład:

```
{
"endpoints": {
  "source": [
    {
      "type": "universal",
      "params": {
        "maxConnections": -1
      }
    }
  ],
  "target": [
    {
      "type": "universal",
      "params": {}
    }
  ]
}
```

}

3.2 Definicja formularza - form

Definicja formularza konfiguracji funkcjonalności.

Parametr	Opis	Wymagane	Uwagi
fields	Lista pól formularza	TAK	
validation	Konfiguracja walidacji ustawień bloczka	TAK	

3.2.1 Lista pól - fields

Pole tekstowe - text, url, email

Podstawowe pole tekstowe. Ustawiając typ url lub email, nastąpi wymuszenie poprawności wpisanej wartość dla odpowiedniego typu.

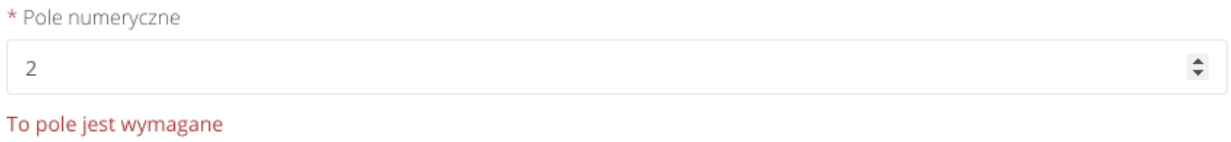
Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
label	Etykieta pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)
placeholder	Teks pomocniczy	TAK	Możliwe tłumaczenia (%TRANSLATE%)
required	Wymagalność pola	NIE	true/false
maxlength	Maksymalna liczba znaków	NIE	

Przykład:

```
{
  "id": "param1",
  "type": "text",
  "label": "%TRANSLATE|pole_tekstowe%",
  "placeholder": "%TRANSLATE|podpowiedz_pola%",
  "required": true,
  "maxlength": 30
}
```

Pole numeryczne - number

Pole edycyjne przyjmujące tylko liczby.



Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
label	Etykieta pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)
placeholder	Teks pomocniczy	TAK	Możliwe tłumaczenia (%TRANSLATE%)
required	Wymagalność pola	NIE	true/false
readonly	Pole tylko do odczytu	NIE	true/false
min	Minimalna wartość, jaką może przyjąć pole	NIE	
max	Maksymalna wartość, jaką może przyjąć pole	NIE	
step	Określa, o jaki zakres może zmienić się wartość pola	NIE	

Przykład:

```
{
  "id": "param1",
  "type": "number",
  "label": "%TRANSLATE|pole_tekstowe%",
  "placeholder": "%TRANSLATE|podpowiedz_pola%",
  "required": true,
  "min": 1,
  "max": 20,
  "step": 2,
}
```

Obszar tekstowy - textarea

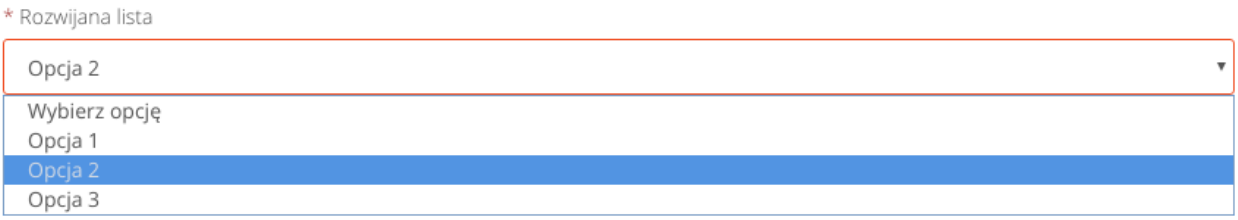
Wieloliniowe pole tekstowe.

Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
label	Etykieta pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)
placeholder	Teks pomocniczy	TAK	Możliwe tłumaczenia (%TRANSLATE%)
required	Wymagalność pola	NIE	true/false
rows	Maksymalna liczba wierszy	NIE	

Przykład:

```
{
  "id": "param1",
  "type": "textarea",
  "label": "%TRANSLATE|obszar_tekstowy%",
  "placeholder": "%TRANSLATE|podpowiedz_pola%",
  "required": true,
  "rows": 5
}
```

Rozwijana lista - select



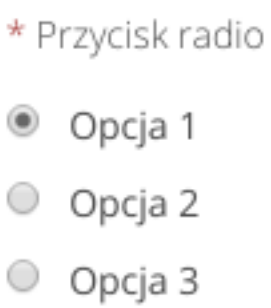
Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
label	Etykieta pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)
required	Wymagalność pola	NIE	true/false
reload	Określenie czy po zmianie wartości formularz ma się przeładować	NIE	true/false
values	Opcje do wyboru w polu	TAK	
default	Domyślnie wybrana opcja	NIE	Identyfikator opcji

Przykład:

```
{
  "id": "param1",
  "type": "select",
  "label": "%TRANSLATE|rozwijana_lista%",
  "required": true,
  "default": "option2",
  "values": [
      {
        "id": "option1",
        "label": "%TRANSLATE|opcja1%"
      },
      {
        "id": "option2",
        "label": "%TRANSLATE|opcja2%"
      },
      {
        "id": "option3",
        "label": "%TRANSLATE|opcja3%"
      }
  ]
}
```

Przycisk radio - radio

Grupa przycisków radio.



Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
label	Etykieta pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)
required	Wymagalność pola	NIE	true/false
values	Opcje do wyboru w polu	TAK	
default	Domyślnie wybrana opcja	NIE	Identyfikator opcji

Przykład:

```
{
  "id": "param1",
  "type": "radio",
  "label": "%TRANSLATE|przycisk_radio%",
  "required": true,
  "values": [
```

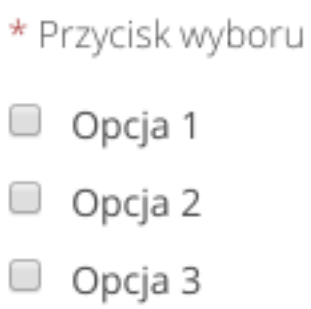
(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
      {
        "id": "option1",
        "label": "%TRANSLATE|opcja1%"
      },
      {
        "id": "option2",
        "label": "%TRANSLATE|opcja2%"
      },
      {
        "id": "option3",
        "label": "%TRANSLATE|opcja3%"
      }
  ]
}
```

Przycisk wyboru - checkbox

Grupa przycisków wyboru.



Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
label	Etykieta pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)
required	Wymagalność pola	NIE	true/false
reload	Określenie czy po zmianie wartości formularz ma się przeładować	NIE	true/false
values	Opcje do wyboru w polu	TAK	

Przykład:

```
{
  "id": "param1",
  "type": "checkbox",
  "label": "%TRANSLATE|przycisk_wyboru%",
  "required": true,
  "values": [
      {
```

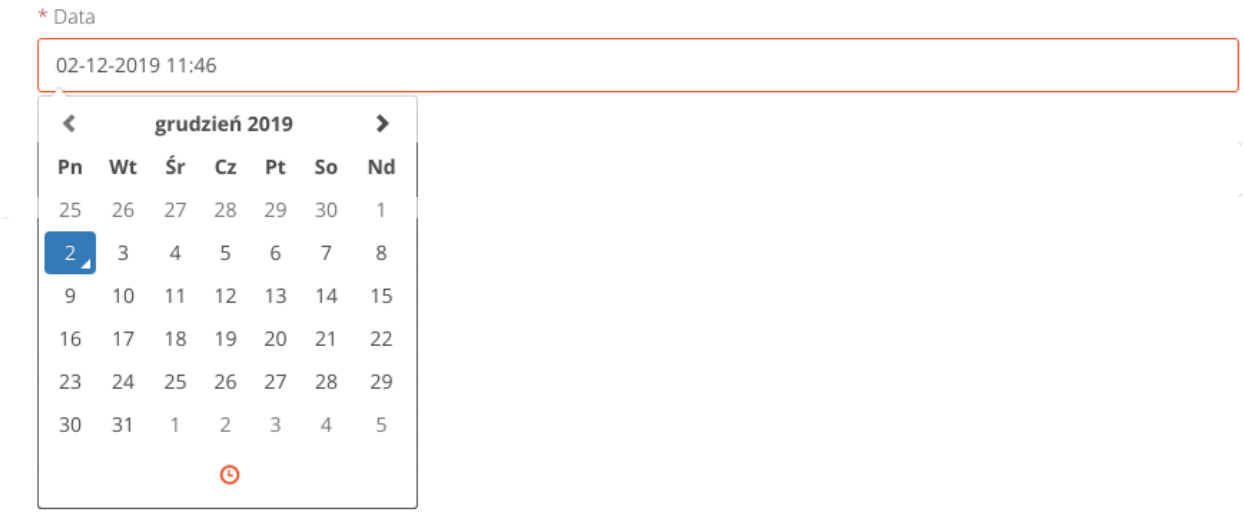
(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
        "id": "option1",
        "label": "%TRANSLATE|opcja1%"
      },
      {
        "id": "option2",
        "label": "%TRANSLATE|opcja2%"
      },
      {
        "id": "option3",
        "label": "%TRANSLATE|opcja3%"
      }
  ]
}
```

Wybór daty - date

Pole wyboru daty i godziny.



Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
label	Etykieta pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)
placeholder	Tekst pomocniczy	TAK	Możliwe tłumaczenia (%TRANSLATE%)
required	Wymagalność pola	NIE	true/false
format	Format daty/godziny do wpisania	TAK	• d(dzień bez zero), dd(dzień z zero) np.: 5, 05 • D(krótki dzień tygodnia), DD(pełen dzień tygodnia) np.: Pon, Poniedziałek • m(miesiąc bez zero), mm(miesiąc z zero) np.: 7, 07 • M(Krótka nazwa miesiąca), MM(Pełna nazwa miesiąca) np.: Mar, Marzec • yy(2 cyfrowy rok), yyyy(4 cyfrowy rok) np.: 12, 2012
min_view	Minimalny widok do wybory	NIE	• 0, days, month - Widok wyboru dnia miesiąca • 1, months, year - Widok wyboru miesiąca • 2, years, decade - Widok wyboru roku(widok dekady) • 3, decades, century - Widok wyboru dekady(widok stulecia)
max_view	Maksymalny widok do wyboru	NIE	j.w.
min_date	Minimalna możliwa data do wyboru	NIE	„now” dla obecnej daty i godziny
output_format	Format wynikowy	NIE	Domyślnie X, Opcję zgodne z Moment.

Przykład:

```
{
  "id": "param1",
  "type": "date",
  "label": "%TRANSLATE|data%",
  "placeholder": "%TRANSLATE|podaj_date_i_godzine%",
  "label": "%TRANSLATE|data%",
  "format": "DD-MM-YYYY HH:mm",
  "min_date": "now",
  "max_view": "year",
  "required": true
}
```

Wybór dnia tygodnia - week-picker

Pole wyboru dnia tygodnia.

Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
label	Etykieta pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)
required	Wymagalność pola	NIE	true/false

Przykład:

```
{
  "id": "param1",
  "type": "week-picker",
  "label": "%TRANSLATE|dzien_tygodnia%",
  "required": true
}
```

Grupy pól - array

Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
label	Etykieta pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)
max	Maksymalna liczba grup	TAK	• rule - Maksymalna liczba grup • info - Informacja która się wyświetla po osiągnięciu maksimum. Możliwe tłumaczenia (%TRANSLATE%)
min	Minimalna liczba grup	TAK	
fields	Lista pól w grupie	TAK	
addButton	Opcje przycisku dodawania grupy	TAK	• show - Czy przycisk ma być wyświetlony • disabled - Czy przycisk ma być wyłączony • label - Etykieta przycisku. Możliwe tłumaczenia (%TRANSLATE%)

Właściwość etykiety pola(**label**) dla pól w grupie przyjmuję postać tablicy obiektów o następującej postaci:

Parametr	Opis	Wymagane	Uwagi
index	Indeks grupy	TAK	Numerowane od 0. Należy wstawić false dla pozostałych indeksów
text	Tresć etykiety pola	TAK	Możliwe tłumaczenia (%TRANSLATE%)

Przykład:

```
{
  "id": "param1",
  "label": "%TRANSLATE|grupa%",
  "type": "array",
  "min": 1,
  "max": {
    "rule": 5,
    "info": "%TRANSLATE|osiagnales_juz_limit_5_adresow_url%"
  },
  "init": true,
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
  "addButton": {
    "show": true,
    "disabled": false,
    "label": "%TRANSLATE|dodaj_strone%"
  },
  "fields": [
    {
      "id": "param2",
      "type": "text",
      "label": [
        {
          "index": 0,
          "text": "%TRANSLATE|pierwsze_pole%"
        },
        {
          "index": false,
          "text": "%TRANSLATE|pole_tekstowe%"
        }
      ],
      "placeholder": "%TRANSLATE|podpowiedz_pola%",
      "required": true,
      "maxlength": 30
    }
  ]
}
```

Tekst statyczny - paragraph

Parametr	Opis	Wymagane	Uwagi
id	Unikalny identyfikator pola	TAK	
content	Treść	TAK	Możliwe tłumaczenia (%TRANSLATE%)
style	Styl elementu	NIE	Styl jako obiekt

Przykład:

```
{
  "id": "param1",
  "type": "paragraph",
  "content": "%TRANSLATE|tresc%",
  "style": {
    "margin-top": "24px"
  }
}
```

3.2.2 Pola zależne

Prosta zależność

Jeśli jest potrzeba ukazania pola zależnie od wypełnienia innego pola, należy użyć właściwości **show_if**, która jest tablicą identyfikatorów od którego zależy pole.

Przykład:

```
{
  "id": "param2",
  "type": "select",
  "label": "%TRANSLATE|rozwijana_lista%",
  "required": true,
  "show_if": [
    "param1"
  ],
  "default": "option2",
  "values": [
      {
        "id": "option1",
        "label": "%TRANSLATE|opcja1%"
      },
      {
        "id": "option2",
        "label": "%TRANSLATE|opcja2%"
      },
      {
        "id": "option3",
        "label": "%TRANSLATE|opcja3%"
      }
  ]
}
```

Dla pól *select*, *checkbox* oraz *radio* możliwe jest zdefiniowanie listy pól zależnych od wartosci pola rodzica.

Zależności bez przeładowania - dependent

Jeśli nie jest wymagane przeładowanie całego formularza można użyć pola **dependent** (np. gdy pola zależne nie posiadają zmiennych)

Przyklad:

```
{
  "id": "param1",
  "type": "select",
  "label": "%TRANSLATE|rozwijana_lista%",
  "required": true,
  "default": "option2",
  "values": [
      {
        "id": "option1",
        "label": "%TRANSLATE|opcja1%"
        "dependent": [
          {
            "id": "param2",
            "type": "number",
            "label": "%TRANSLATE|pole_tekstowe%",
            "placeholder": "%TRANSLATE|podpowiedz_pola%",
            "required": true,
            "min": 1,
            "max": 20,
            "step": 2,
          }
        ]
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
      },
      {
        "id": "option2",
        "label": "%TRANSLATE|opcja2%"
      },
      {
        "id": "option3",
        "label": "%TRANSLATE|opcja3%"
      }
  ]
}
```

Zależności z przeładowaniem - sub

Jeśli pola zależne posiadają zmienne należy użyć **sub**. Dodając parametr *reload* do rodzica następuje wymuszenie przeładowania całego formularza.

Przyklad:

```
{
  "id": "param1",
  "type": "select",
  "label": "%TRANSLATE|rozwijana_lista%",
  "required": true,
  "reload": true,
  "default": "option2",
  "values": [
      {
        "id": "option1",
        "label": "%TRANSLATE|opcja1%"
        "sub": [
          {
            "id": "param2",
            "type": "number",
            "label": "%TRANSLATE|pole_tekstowe%",
            "placeholder": "%TRANSLATE|podpowiedz_pola%",
            "required": true,
            "min": 1,
            "max": 20,
            "step": 2,
          }
        ]
      },
      {
        "id": "option2",
        "label": "%TRANSLATE|opcja2%"
      },
      {
        "id": "option3",
        "label": "%TRANSLATE|opcja3%"
      }
  ]
}
```

3.2.3 Walidacja pól - fields

Konfiguracja bloczka zapisywana jest w obiekcie **settings** w którym kluczem jest identyfikator pola a wartościa jest wartość uzupełniona danego pola.

```
{
  "settings": {
    "param1": "wartosc pola",
    "param2": 2
  },
}
```

Przed zapisaniem schematu kampanii, każdy bloczek jest sprawdzany pod kątem poprawności uzupełnienia jego konfiguracji.

Aby zdefiniować konfigurację walidacji należy przygotować model walidacji zgodny z JSON Schema draft-07. W modelu walidacji należy uwzględnić cały obiekt settings.

Przykład:

```
{
  "settings": {
    "param1": "wartosc pola",
    "param2": 5
  }
}
```

```
{
  "validation": {
    "model": {
      "type": "object",
      "required": [
        "settings"
      ],
      "properties": {
        "settings": {
          "type": "object",
          "required": [
            "param1",
            "param2"
          ],
          "properties": {
            "param1": {
              "type": "string"
            },
            "param2": {
              "type": "number"
            }
          }
        }
      }
    }
  },
}
```

ROZDZIAŁ 4

Wstępne parsowanie pliku

Część konfiguracji przed właściwym użyciem przez silnik jest wstępnie parsowana. Ten proces może być kontrolowany przy użyciu dyrektyw preprocesora.

4.1 Dyrektywy parsowania

4.1.1 Dyrektywy warunkowe - @anyOf i @oneOf

Dopuszczalne sa następujące formaty dyrektywy:

- Pełna

```
{
  "@anyOf|@oneOf": {
    "items": [
        {
          "if": "reguła",
          "content": "zawartość do wstawienia przy spełnieniu warunku"
        }
    ],
    "options": {
    }
  }
}
```

- Skrócona definicja

```
{
  "@anyOf|@oneOf": [
    {
      "if": "reguła",
      "content": "zawartość do wstawienia przy spełnieniu warunku"
    }
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
  ],
}
```

W skróconej wersji nie jest możliwe przekazanie dodatkowych opcji.

@anyOf

Dyrektywa pozwala na zdefiniowane listy konfiguracji które mają być wybrane jeśli zostaną spełnione określone warunki.

Dodatkowe opcje:

Parametr	Opis	Wymagane	Uwagi
merge	Opcja określa czy przy kilku spełnionych warunkach ich zawartość ma był połączona	NIE	Domyślnie false

@oneOf

Dyrektywa pozwala na wybranie wyłącznie jednego wariantu po spełnieniu warunku. Przy parsowaniu jeden warunek **MUSI** zostać spełniony, więc zalecane jest, aby ostatni warunek był określony przez wyrażenie true. Jeśli warunek zostanie spełniony parsowanie zostanie zatrzymane.

Poniższa tabela przedstawia najważniejsze różnice miedzy dyrektywami @oneOf i @anyOf:

@oneOf	@anyOf
Przynajmniej jeden warunek musi zostać spełniony	Żaden warunek nie musi być spełniony
Maksymalnie jeden warunek może zostać spełniony	Wszystkie warunki mogą być spełnione
Parsowanie zostanie zatrzymane przy| pierwszym spełnionym warunku	Parsowanie zostanie zatrzymane po sprawdzeniu wszystkich warunków

Jeśli reguła jest skomplikowana i składa sie z kilku warunków możliwy jest podział reguły na listę warunków:

```
{
  "@anyOf|@oneOf": [
    {
      "if": [
        {
          "rule": "reguła1"
        },
        {
          "rule": "reguła2",
          "type": "OR/AND"
        }
      ],
      "content": "zawartość do wstawienia przy spełnieniu warunku"
    }
  ],
}
```

4.1.2 Pętla - @repeat

Dyrektywa pozwala na cykliczne powielenie konfiguracji, przechodząc po kolekcji znajdującej się w kontekście silnika.

```
{
  "@repeat": {
    "var": "nazwa zmiennej pod którą będzie dostępny obecny element",
    "in": "kolekcja po której następuje iteracja",
    "content": "zawartość do powielenia"
  }
}
```

Przykład:

```
{
  "@repeat": {
    "var": "i",
    "in": "testArray",
    "content": "{{i.id}}"
  }
}
```

4.1.3 Wyrażenia - expression

Dyrektywa pozwala na wstawienie wyniku wyrażenia lub zmiennych we wskazane miejsce. Wyrażenie **MUSI** być wstawione w postaci **{{wyrażenie}}**. W jednym polu może znajdować się wiele bloków wyrażeń.

Przykład:

```
{
    "exp1" : "{{settings[0]}} {{settings[1]}}",
    "exp2" : "{{settings[0] * settings[1]}}"
}
```

Jeśli wynikiem wyrażenia jest liczba lub typ logiczny i jest to jedyne wyrażenie w danym polu to wynik jest automatycznie przekonwertowany.

Przykład:

```
{
    "exp1" : "{{1}}",
}
```

Wynikiem będzie:

```
{
    "exp1" : 1,
}
```

4.1.4 Formatowanie - @formatter

Dyrektywa pozwala na przeformatowanie bloku konfiguracji z jednego typu na drugi.

```
{
  "type": "typ konwertera",
  "config": {
  },
  "value": {}
}
```

string

Konwersja do łańcucha znaków. Używane głównie przy dyrektywie **@anyOf** do generowania tekstu zależnie od warunków.

Opcje - config:

Parametr	Opis	Wymagane	Uwagi
concat	Łącznik fragmentów łańcucha	NIE	

```
{
  "@formatter": {
    "type": "string",
    "config": {
      "concat": " and "
    },
    "value": {
      "@anyOf": [
        {
          "if": true,
          "content": "Hello"
        },
        {
          "if": true,
          "content": "World"
        }
      ]
    }
  }
}
```

Czego wynikiem będzie „Hello and World”.

4.1.5 Przetwarzanie szablonu TWIG - @twigRender

Dyrektywa pozwala na wstawienie wyniku parsowania zewnętrznego pliku Twig. Kontekstem parsowania pliku jest kontekstem preparsera.

```
{
  "@twigRender": {
    "file": "ścieżka do pliku relatywnie do system.json"
  },
  "options": {
  }
}
```

Opcje - options:

Parametr	Opis	Wymagane	Uwagi
wrap	Pozwala na wstawienie tekstu przed i po wyniku parsowania twiga	NIE	Aby wstawić treść wygenerowana z szablonu twiga należy użyć **{content}**

Przykład:

```
{
  "@twigRender": {
    "file": "twig/test.twig"
  },
  "options": {
    "wrap": "<!- {content} -->"
  }
}
```

4.2 Kontekst parsowania

Kontekst jest zbiorem zmiennych oraz funkcji które można użyć w powyższych dyrektywach. Dostępność niektórych zmiennych zależy od parsowanego fragmentu konfiguracji.

4.2.1 Funkcje pomocnicze - utils

Funkcje dostępne są w każdym parsowanym fragmencie

isEmpty

Sprawdzenie czy zmienna posiada wartość. Odpowiednik empty z języka PHP.

Zwracana wartość: Boolean

Parametry:

Parametr	Opis	Wymagane	Uwagi
data	Zmienna do sprawdzenia	TAK	

Przykład:

```
{
  "test": "{{utils.isEmpty(data)}}"
}
```

implode

Łączenie elementów w łańcuch znaków. Odpowiednik implode z języka PHP.

Zwracana wartość: String

Parametry:

Parametr	Opis	Wymagane	Uwagi
data	Tablica elementów do złączenia	TAK	
glue	Łącznik elementów	TAK	

Przykład:

```
{
  "test": "{{utils.implode(', ', data)}}"
}
```

round

Zaokrąglenie liczby zmiennoprzecinkowej. Odpowiednik round z języka PHP.

Zwracana wartość: Float

Parametry:

Parametr	Opis	Wymagane	Uwagi
val	Liczba do Zaokrąglenia	TAK	
precision	Precyzja(liczba cyfr po przecinku)	NIE	Domyślnie 0

Przykład:

```
{
  "test": "{{utils.round(price, 2)}}"
}
```

escape

Dekodowanie łańcucha znaków do postaci bezpiecznej do użycia w Twig

Zwracana wartość: String

Parametry:

Parametr	Opis	Wymagane	Uwagi
str	Tekst do zdekodowania	TAK	

createDateTime

Tworzenie obiektu DateTime.

Zwracana wartość: DateTime

Parametry:

Parametr	Opis	Wymagane	Uwagi
time	formuła dnia i godziny	NIE	Domyślnie „now”. Zgodnie z formatem
timezone	Formuła strefy czasowej lub przesunięcie(np. +0200)	NIE	Domyślnie null. Zgodnie z strefami

Przykład:

```
{
  "test": "{{utils.createDateTime('2019-12-06 12:00:00').getTimeStamp()}}"
}
```

includes

Sprawdzenie czy jakiś tekst występuje w innym. Odpowiednik strpos z języka PHP.

Zwracana wartość: Boolean

Parametry:

Parametr	Opis	Wymagane	Uwagi
str	Tekst w którym następuje wyszukiwanie	TAK	
substring	Tekst do wyszukania	TAK	

Przykład:

```
{
  "test": "{{utils.includes('Hello World', 'Hello')}}"
}
```

replace

Zamiana tekstu na inny. Odpowiednik str_replace z języka PHP.

Zwracana wartość: String

Parametry:

Parametr	Opis	Wymagane	Uwagi
search	Tekst lub tablica do wyszukania	TAK	
replace	Tekst lub tablica na które ma być dokonana zmiana	TAK	
subject	Tekst w którum następuje wyszukiwanie	TAK	

Przykład:

```
{
  "test": "{{utils.includes('World', 'Universe', 'Hello World')}}"
}
```

arrayFilter

Filtrowanie elementów po przekazanych parametrach

Zwracana wartość: Array

Parametry:

Parametr	Opis	Wymagane	Uwagi
input	Tablica wejściowa	TAK	
params	Obiekt parametrów filtra lub JSON tego obiektu	TAK	

Przykład:

```
{
  "test": "{{utils.arrayFilter(data, '{\"param\": \"val\"}')}}"
}
```

arrayColumn

Zwraca tablicę z wartościami z wskazanej kolumny. Odpowiednik array_column z PHP.

Zwracana wartość: Array

Parametry:

Parametr	Opis	Wymagane	Uwagi
input	Tablica wejściowa	TAK	
param	Klucz/kolumna która ma być zwrócona	TAK	

Przykład:

```
{
  "test": "{{utils.arrayColumn(data, 'param')}}"
}
```

4.2.2 Parametry konta - session

getHubId()

Pobranie identyfikatora konta

Przykład:

```
{
  "test": "{{session.getHubId()}}"
}
```

4.2.3 Treści - contentRepository

Pobranie treści z systemu.

```
{
  "test": "{{contentRepository.getContent('typTreści', identyfikator)}}"
}
```

Dostępne treści

- personalization - szablon personalizacji
- webPushTemplate - szablon powiadomienia push
- staticContent - treść statyczna

Przykład:

```
{
  "test": "{{contentRepository.getContent('webPushTemplate', 1).getTitle()}}"
}
```

4.2.4 mediaHelper

generatePublicUrl

Pozwala na wygenerowanie publicznego url do określonego zasobu

Parametry:

Parametr	Opis	Wymagane	Uwagi
hubId	Identyfikator konta w systemie	TAK	
media	Obiekt zasobu	TAK	
type	Typ zasobu	NIE	Domyślnie reference
absolute	Czy ścieżka ma być pełna	NIE	Domyślnie: true

Przykład:

```
{
  "test": "{{mediaHelper.generatePublicUrl(session.hubId, notification.icon}"
}
```

4.2.5 Ustawienia bloczka

Podczas zapisu bloczka (sekcja *save*) w kontekscie zapisane są wszystkie uzupełnione wartości z formularza. Parametry dostępne są w obiekcie **settings**.

```
{
  "settings": {
    "param1": "val1",
    "param2": 1
  }
}
```

```
{
  "test": "{{settings['param1']}}"
}
```

ROZDZIAŁ 5

Obsługa zdarzeń

System pozwala na określenie akcji jaka ma być wykonana przy określonych zdarzeniach systemu.

Definicja obsługi zdarzeń składa się listy zdarzeń, gdzie kluczem jest nazwa zdarzenia, a wartością jest obiekt składający się z dodatkowej konfiguracji - **config** oraz listy funkcji do wykonania podczas wywołania zdarzenia - **listeners**.

Przykład:

```
{
  "events": {
    "entity.changed": {
      "config": {},
      "listeners": ["..."]
    },
    "campaign.saved": {
      "config": {},
      "listeners": ["..."]
    }
  }
}
```

Defincja funkcji składa się z pola **type** oznaczającą nazwę funkcji oraz dodatkowej konfiguracji - **config**.

Przykład:

```
{
  "events": {
    "entity.changed": {
      "config": {},
      "listeners": [
        {
          "type": "findEntityInCampaign",
          "config" : {
            "...": "..."
          }
        }
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
        ]
    }
}
```

Możliwa jest również definicja składająca się tylko z nazwy, jeśli w określonym przypadku nie jest wymagana konfiguracja(np. gdy poprzednia funkcja ustawi odpowiednie parametry):

Przykład:

```
{
  "events": {
    "entity.changed": {
      "config": {},
      "listeners": [
        "reloadCampaigns"
      ]
    }
  }
}
```

Funkcje są uruchamiana zgodnie z kolejnością z jaką zostały zdefiniowane - z góry na dół. Funkcje mogą zapisywać parametry do wykorzystania w następnej funkcji. Konfiguracje funkcji są wstępnie

5.1 Dostępne zdarzenia

5.1.1 Zmiana encji systemowej - entity.changed

Zdarzenie wyłowywane w momencie zapisu określonej encji systemowej - szablonu oraz zgodę web-push, personalizacji, treść statyczna oraz grupy testowej.

Zdarzenie posiada następujące właściwości:

Parametr	Opis	Uwagi
type	typ zapisanej encji	personalizationTemplate, productFeed, testGroup, staticContent, notificationsTemplate, notificationsPromptTemplate
entity	obiekt zapisanej encji	

5.1.2 Pomyślne zintegrowanie systemu - integration.completed

Zdarzenie wyłowywane w momencie gdy użytkownik pomyślne zakończy proces integracji.

Zdarzenie posiada następujące właściwości:

Parametr	Opis	Uwagi
system	obiekt zintegrowanego systemu	
systemAccount	obiekt zintegrowanego konta w systemie	

5.1.3 Pomyślne usunięcie zintegrowanego systemu - integration.deleted

Zdarzenie wyłowywane w momencie gdy użytkownik pomyślne usunie integrację z konta

Zdarzenie posiada następujące właściwości:

Parametr	Opis	Uwagi
system	obiekt zintegrowanego systemu	
systemAccount	obiekt zintegrowanego konta w systemie	

5.1.4 Uruchamianie kampanii - campaign.starting

Zdarzenie wywoływane podczas uruchamiania kampanii.

Zdarzenie posiada następujące właściwości:

Parametr	Opis	Uwagi
account	obiekt konta systemu	
campaign	obiekt kampanii	

5.1.5 Uruchamianie bloczka kampanii - campaign.node.starting

Podobne jak *campaign.starting* tylko wywołane dla każdego bloczka w kampanii

Parametr	Opis	Uwagi
block	obiekt uruchamianego bloczka	
campaign	obiekt kampanii	
session	obiekt konta systemu	

5.1.6 Uruchomiona kampania - campaign.started

Zdarzenie wywoływane przed zakończeniem procesu uruchamiana kampanii.

Zdarzenie posiada następujące właściwości:

Parametr	Opis	Uwagi
account	obiekt konta systemu	
campaign	obiekt kampanii	

5.1.7 Uruchamianie bloczka kampanii - campaign.node.started

Podobne jak *campaign.started* tylko wywołane dla każdego bloczka w kampanii

Parametr	Opis	Uwagi
block	obiekt uruchamianego bloczka	
campaign	obiekt kampanii	
session	obiekt konta systemu	

5.2 Dostępne funkcję obsługi zdarzeń

5.2.1 Wyszukiwanie encji w kampanii - findEntityInCampaign

Funkcja pozwala na wyszukanie kampanii w których użyta jest dana encja.

Konfiguracja:

Parametr	Opis	Wymagane	Uwagi
campaigns	Pozwala na ograniczenie kampanii do wyszukania	NIE	
entityId	Lista gdzie kluczem jest identyfikator pola ustawień bloczka, a wartością jest właściwość encji powiązana z polem	TAK	identyfikatory grup testowych - test_group_id

Parametry ograniczenia wyszukiwania kampanii - **campaigns**

Parametr	Opis	Wymagane	Uwagi
functionality	Funkcjunalność do wyszukania w kampanii	NIE	
status	Status kampanii	NIE	active - aktywne, inactive - wstrzymane

Przykład z użyciem zdarzenia entity.changed:

```
{
  "events": {
    "entity.changed": {
      "config": {},
      "listeners": [
        {
          "type": "findEntityInCampaign",
          "config": {
            "campaigns": {
              "functionality": [
                "email",
                "database",
                "mobile"
              ],
              "status": "active"
            },
            "entityId": {
              "@anyOf": {
                "items": [
                  {
                    "if": "event.getType() = 'testGroup'",
                    "content": {
                      "test_group_id": "{{event.getEntity().getId()}}"
                    }
                  },
                  {
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
                    "if": "event.getType() = 'personalizationTemplate'",
                    "content": {
                      "personalization_template_id": "{{event.getEntity().getId()}}",
                      "data_source_cart": "{{event.getEntity().getId()}}",
                      "data_source_recommendation": "{{event.getEntity().getId()}}"
                    }
                  }
                ],
              "options": {
                "merge": true
              }
            }
            }
          }
        }
      ]
    }
  }
}
```

Funkcja zapisuje wyszukanie identyfikatory w parametrze campaigns w właściwości id.

5.2.2 Ponowne uruchomienie kampanii - reloadCampaigns

Funkcja pozwala na przeładowanie kampanii w koncie.

Konfiguracja:

Parametr	Opis	Wymagane	Uwagi
campaigns	Pozwala na ograniczenie kampanii do wyszukania	TAK	

Parametry ograniczenia wyszukiwania kampanii - **campaigns**

Parametr	Opis	Wymagane	Uwagi
id	Id kampanii	NIE	
status	Status kampanii	NIE	active - aktywne, inactive - wstrzymane
type	Typ kampanii	NIE	

Przykład:

```
{
  "events": {
    "entity.changed": {
      "config": {},
      "listeners": [
        {
          "type": "reloadCampaigns",
          "config": {
            "campaigns": {
              "id": "{{params['campaigns']['id']}}"
            }
          }
        }
```

(ciąg dalszy na następnej stronie)

(kontynuacja poprzedniej strony)

```
        ]
      }
    }
}
```

Jeśli poprzednia funkcja zapisała parametr campaigns (np. findEntityInCampaign), funkcję **reloadCampaigns** można zapisać bez konfiguracji

Przykład:

```
{
  "events": {
  "entity.changed": {
    "config": {},
    "listeners": [
      {
        "type": "findEntityInCampaign",
        "config": {
          "campaigns": {
            "functionality": [
              "email",
              "database",
              "mobile"
            ],
            "status": "active"
          },
          "entityId": {
            "@anyOf": {
              "items": [
                {
                  "if": "event.getType() = 'testGroup'",
                  "content": {
                    "test_group_id": "{{event.getEntity().getId()}}"
                  }
                },
              ],
              "options": {
                "merge": true
              }
            }
          }
        }
      },
      "reloadCampaigns"
    ]
  }
}
```

5.2.3 Wywołanie API - api-call

Funkcja służy do wywołania zapytania do API

Konfiguracja:

Parametr	Opis	Wymagane	Uwagi
client	Klient API Guzzle	TAK	
request	Konfiguracja żądania	TAK	

Konfiguracja żądania:

Parametr	Opis	Wymagane	Uwagi
method	Metoda HTTP	TAK	GET, POST itp.
url	Url do zasobu	TAK	
options	Opcję żądania	TAK	Opcję zgodne z Guzzle.

Przykład:

```
{
  "events": {
    "campaign.starting": {
      "config": {},
      "listeners": [
        {
          "type": "api-call",
          "config": {
            "client": "test_api",
            "request": {
              "method": "PUT",
              "url": "hubs/{{event.getAccount().id}}/campaigns/{{event.getCampaign().
↪getId()}}/test",
              "options": {
                "json": {
                  "test": 1
                }
              }
            }
          }
        }
      ]
    }
  }
}
```

Wejściowym plikiem dla zdefiniowanej konfiguracji **MUSI** być plik *system.json*, który określa system z którym jest definiowana konfiguracja.

ROZDZIAŁ 6

Format i struktura plików

Konfigurację **MUSI** być zdefiniowana w pliku JSON.

6.1 Podział pliku

Mimo że całą konfigurację można zapisać w jednym pliku, możliwy jest podział konfiguracji na wiele plików. Odnośnik do zewnętrznego pliku należy zdefiniować pod właściwością *@ref*, gdzie wartością **MUSI** być ścieżka do dołączanego pliku relatywnie do pliku *system.json*.

Przykład:

```
{
  "modules": [
        {
          "@ref" : "modules/module.json"
        }
      ]
}
```