1. Ogólny Opis interfejsu, założenia projektowe, schemat działania (myślę że wystarczy to co Błazej mówił na prezentacji + grafiki z prezentacji ładnie Opisane)
2. Budowa aplikacji, moduły wymagane, moduły opcjonalne +6./
3. Opis modułów
4. Konfiguracja (cxse)
5. Jak podmienić moduł?
6. Skróty klawiszowe

**Jak developer ma korzystać z interfejsu, oczekiwania (jak modyfikować I rozwijać kod)**

Oczekiwania użytkownika zwykłego

Owl.cs.manchester.ac.uk/tutorials/ protegeowltutorial

Dokumentacja oryx

Czym jest graf, owl, definicje. Przypadek,

Mniejszy obrazek, zmiana rozwój I wprowadzanie coraz bardziej zaawansowane rzeczy, zakładaja. Jak budować usługę złożoną, co jest prostsze niż tworzenie wymagania.

Ćwiczenie jak zbudować usługę złożoną. Proste funkcje interfejsu. Jak dodać wierzchołek.

Usługa -> wymaganie -> editor (z formularzami) edycja wierzchołków manual-> usługa emulująca na koniec.

Typy proste usługi matematyczne.

1. Wyjaśnienie pojęć
2. Przykłady budowania usług za pomocą pojęć

Przykłady wywołania, dołączeina modułu

Ontologia (uzupełnianie wymagani)

Podgraf

Inne pojęcia wprowadzane stopniowo.

# Moduły

Placeholder na spis treści

Konfiguracja

Opis modułowej budowy interfejsu.

Silnik który ładuje wtyczki.

**przykłady**

## Template

### Opis

aa

### Model danych

aa

### Sposób działania modułu

aa

### Zdarzenia

**Zdarzenia obsługiwane przez moduł**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
|  |  |

**Zdarzenia wysyłane przez moduł ”**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
|  |  |

### Ustawienia

aa

|  |  |
| --- | --- |
| **Ustawienie** | **Opis** |
|  |  |

## Event manager

### Opis

Event manager jest jednym z najważniejszych modułów aplikacji. Ponieważ JavaScript jest językiem sterowanym zdarzeniami do stworzenia aplikacji SSDL\_SIG wykorzystano paradygmat porgramowania sterowanego zdarzeniami. Moduł daje możliwość definiowania wewnętrznych zdarzeń, które mogą być wywoływane w odpowiedzi na reakcję użytkownika, zdarzenie przeglądarki lub w innym dowolnym momencie podczas działania program. Zadaniem event managera jest obsługa wewnętrznych zdarzeń w aplikacji. Działanie większości modułów opiera się na jego działaniu. Jego zastosowanie umożliwia dodawanie nowych modułów bez ingerencji w już istniejące.

### Model danych

Obsługa zdarzeń realizowana jest przez funkcje **invokeEvent()** oraz **attachListener().**

Jako parameter funkcja invokeEvent przujmuje nazwę zdarzenia so wywołania oraz opcjonalne parametry które mają zostać przekazane do funkcji słuchacza.

Funkcja attachListener przyjmuje jako parametry nazwę zdarzenia, funkcję która zostanie wykonana w momencie wystąpienia zdarzenia, nazwę modułu w którym słuchacz został zadeklarowany oraz opis.

Moduł dostarcza również funkcję **detachListener()** która jako parameter przyjmuje id listenera zwróconego przez funkcję attachListener()

### Sposób działania modułu

W momencie wywołania funkcji invokeEvent moduł uruchamia wszystkie funkcje nasłuchujące na dane zdarzenie. Przykładowo podczas startu aplikacji, po załadowaniu silnika, wywoływane jest zdarzenie *initView.* Łańcuch funkcji wywoływanych w reakcji na kego wystąpienie może wyglądać następująco:



Można łatwo zauważyć, że funkcje wywoływane w odpowiedzi na wystąpienie zdarzenia mogą wywoływać kolejne. Pozwala to na tworzenie bardziej skomplikowanych łańcuchów zdarzeń co skutkuje precyzyjną reakcją na wystąpienie konktnego zdarzenia.

## Arrow

### Opis

Zadaniem modułu “arrow” jest rysowanie strzałek łączących wierzchołki w grafie. Strzałki rysowane są poprzez kliknięcie myszką od wyjścia wierzchołka do wejścia wierzchołka w grafie. Ponadto moduł reaguje na zaznaczenie strzałki poprzez zmianę jej koloru

### Model danych

Strzałka dołączana jest do wejść i wyjść wierzchołków, które łączy, a więc nie posiada własnej reprezentacji danych.

### Sposób działania modułu

Moduł obsługuje dwa tryby działania: ControlFlow oraz DataFlow.

**ControlFlow:**

Moduł wyszukuje najkrótsze możliwe połączenie pomiędzy konkretnymi wierzchołkami, pomiędzy którymi narysowana ma zostać strzałka.

**DataFlow:**

Rysuje strzałkę pomiędzy wyjściami i wejściami konkretnych wierzchołków

## Atomic

### Opis

### Model danych

aa

### Sposób działania modułu

aa

### Ustawienia

aa

|  |  |
| --- | --- |
| **Ustawienie** | **Opis** |
|  |  |

## History

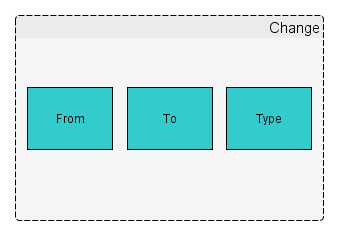
### Opis

Zadaniem modułu historii jest umożliwienie zapamiętania zmian dokonanych przez użytkownika, w celu ewentualnych ich cofnięcia bądź powtórzenia. Po zmianie moduł zapisuje zmienione dane w tablicy. Cofnięcie zmiany lub powtórzenie czynności możliwe jest za pomocą skrótów klawiszowych bądź też poprzez wybór odpowiedniej opcji z menu głównego.

Moduł jest powiązany z innymi modułami za pomocą zdarzeń, które są obsługiwane przez moduł historii.

### Model danych

Dane są przechowywane w tablicy i mają następującą postać:



From: Obiekt, w którym przechowywane są dane historyczne, to znaczy takie które są zachowane przed dokonaniem zmiany przez użytkownika.

To: Obiekt, w którym przechowywane są dane aktualne, to znaczy takie, które są w danym momencie wyświetlane, po zmianach dokonanych przez użytkownika.

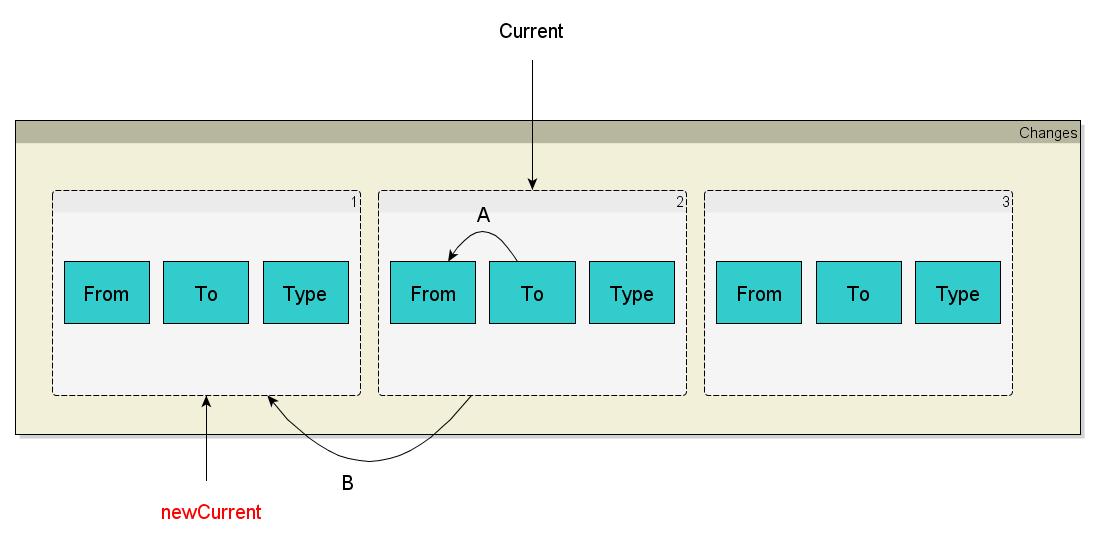
Type: Typ zmiany zapisany w postaci tekstowej pozwalający na rozróżnienie ostatniej dokonanej zmiany.

Tablica zmian zawierać może w sobie wiele powyższych obiektów. Ich liczba ustalana jest w osobnym pliku ustawień. Do wskazania na ostatnią dokonaną zmianę służy wskaźnik zdefiniowany w module.

### Sposób działania modułu

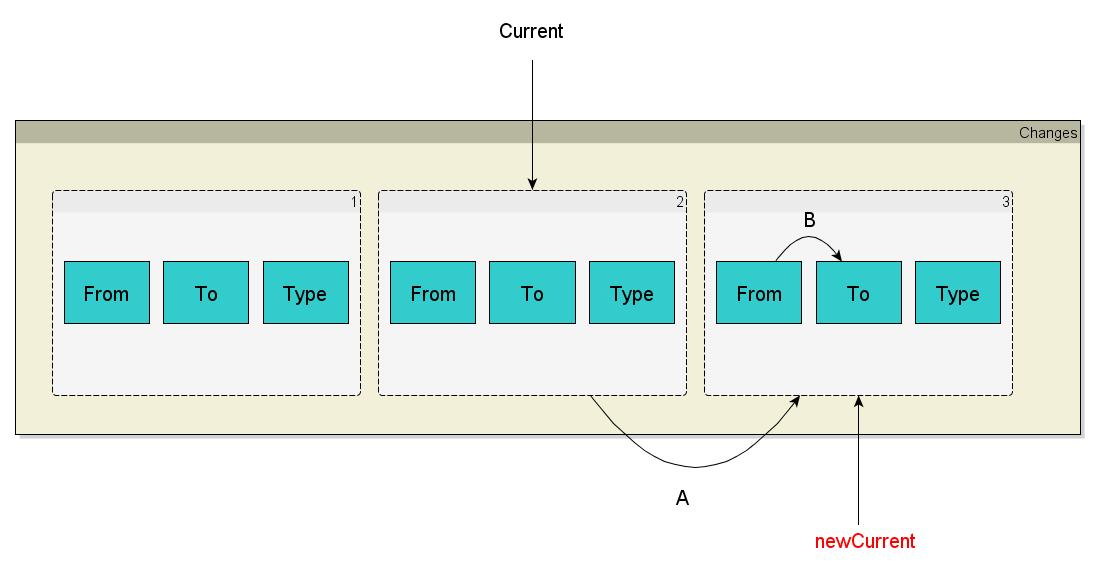
**Cofnięcie zmiany:**

Wskaźnik “Current” ustawiony jest na konkretnej pozycji. Przy cofnięciu zmiany zmieniają się dane, które są aktualnie wyświetlane. Dane, które należy podmienić determinowane są przez typ “Type”. Zmianę aktualnych danych reprezentuje na rysuku strzałka A. Następnie wskaźnik przesuwany jest na poprzednią pozycję w tablicy: strzałka B.



**Powtórzenie zmiany:**

Wskaźnik “Current” ustawiony jest na konkretnej pozycji. Przy powtórzeniu zmiany najpierw zmienia się położenie wskaźnika, co reprezentowane jest na rysunku przez strzałkę A, a następnie podmieniane są wyświetlane dane. Dane, które należy podmienić determinowane są przez typ “Type”. Zmianę aktualnych danych reprezentuje na rysuku strzałka B.

**Cofnięcie wszystkich zmian:**

Polega ono na przejściu z aktualnej zmiany wskazywanej przez wskaźnik do początku tablicy, co odbywa się na takiej samej zasadzie jak dla jednej zmiany, z tym że podmiana danych wykonywana jest w pętli aż osiągnięty zostanie początek tablicy.

**Powtórzenie wszystkich zmian:**  
  
Polega ono na przejściu z aktualnej zmiany wskazywanej przez wskaźnik na koniec tablicy zmian, co odbywa się na takiej samej zasadzie jak dla pojedynczej zmiany, z tym że podmiana danych dokonywana jest w pętli, aż osiągnięty zostanie koniec tablicy.

### Zdarzenia

**Zdarzenia obsługiwane przez moduł “history”:**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| historyChanged | Dodaje obiekt zmiany do tablicy zmian |
| shortcutBack | Cofa zmianę |
| shortcutForward | Powtarza zmianę |
| shortcutBackAll | Cofa wszystkie zmiany |
| shortcutForwardAll | Powtarza wszystkie zmiany |
| historyDeleteBlankNode | Usuwa wierzchołek jednocześnie przekazując go do modułu “blankNode” |

**Zdarzenia wysyłane przez moduł “history”**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| refreshView | Odświeża aktualnie wyświetlany widok |
| replaceSsdl | Podmienia wyświetlany graf. |

### Ustawienia

Ustawienia przechowywane są w oddzielnym folderze ustawień. Plik ustawień modułu *historySettings*:

|  |  |
| --- | --- |
| **Ustawienie** | **Opis** |
| historySize | Maksymalny rozmiar , jaki osiągnąć może tablica zawierająca zmiany w module historii. |

## contextmenu

### Opis

Moduł odpowiedzialny jest za wizualizację menu kontekstowego po kliknięciu przez użytkownika prawym przyciskiem myszy na konkretnym obiekcie, do którego podłączone jest menu kontekstowe. Istnieje także możliwość podłączenia menu kontekstowego do lewego przycisku myszy wykorzystując specjalny parametr

### Model danych

Menu kontekstowe zbudowane jest na zasadzie drzewa. Posiada ono własny model danych. Istnieje globalna tablica menu, do którego dodawany jest obiekt menu kontekstowego, co pozwala na zamknięcie wszystkich obiektów menu poprzez kliknięcie lewym przyciskiem myszy na dowolnym miejscu poza obiektem menu.

Każdy obiekt menu kontekstowego zawiera 3 dane:

1. Obiekt wywołujący menu
2. Korzeń struktury drzewiastej przechowującej opcje.
3. Zbiór podmenu, które sa aktualnie otwarte.

### Sposób działania modułu

Funkcja contextMenu buduje obiekt menu, która jako parametry przyjmuje nasłuchiwany obiekt i opcjonalnie możliwość dodanie akcji po kliknięciu, a także możliwość odblokowania propagacji zdarzenia kliknięcia.

### Zdarzenia

**Zdarzenia obsługiwane przez moduł “contextMenu”:**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| onEscape | Zamyka wszystkie menu |

**Zdarzenia wysyłane przez moduł “contextMenu”**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| Zdarzenie zdefiniowane przy inicjalizacji menu | Zdarzenie zdefiniowane przy inicjalizacji menu |

## Deployer

### Opis

Moduł rozmieszczający graf na kanwie.

### Model danych

Moduł posiada własny parser, ktróry przygotowuje tablicę węzłów, która posiada referencje rodzic- dziecko pomiędzy węzłami.

Na potrzeby działania modułu konieczne jest również zdefiniowanie zmiennych:

1. level- określająca poziom położenia wierzchołka w hierarchii
2. column- numer kolumny w siatce grafu liczony od lewej do prawej. Obrazuje kolejność węzłów na danym poziomie hierarchii.

W późniejszym etapie działania modułu tworzona jest macierz grafu, która zawiera obiekty reprezentujące węzły.

### Sposób działania modułu

Stosując macierz grafu moduł tworzy optymalną strukturę hierarchiczną, a an jej podstawie wylicza współrzędne połorzenia konkretnych wierzchołków grafu.

Moduł może działać w dwóch trybach:

1. Korzystający z metody “bruteforce” (graf poniżej 18 węzłów)
2. Korzystający z algorytmu genetycznego

### Zdarzenia

**Zdarzenia obsługiwane przez moduł**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| deploySsdl | Rozmiescza graf, wpisuje współrzędne wierzchołków i dopasowuje rozmiar kanwy. |
| redeploySsdl | Ponownie rozmieszcza graf. Ponadto wysyła zdarzenie historyChanged, które umozliwia nawigację po historii zmian. |

**Zdarzenia wysyłane przez moduł ”**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| ssdlDeployed | Informuje o zakończeniu działania rozmieszczania grafu. |

### Ustawienia

Ustawienia zawarte są w oddzielnym pliku

|  |  |
| --- | --- |
| **Ustawienie** | **Opis** |
| nodeW | Szerokość wierzchołka |
| nodeH | Wysokość wierzchołka |

## externalInterface

### Opis

Dodaje zestaw funkcji komunikacji zewnętrznych do modelu danych zawartego w silniku.

### Sposób działania modułu

Moduł służy za pośrednika przy wywoływaniu funkcjonalności innych modułów takich jak:

1. Parser
2. Validator
3. AtomicServices
4. Deparser

### Zdarzenia

**Zdarzenia wysyłane przez moduł ”**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| responseFormatError | Wysyłane w momencie błędnego format odpowiedzi |
| unknownAjaxError | Wysyłane w moemencie błędu w komunikacji porpzez Ajax |

## logger

### Opis

Moduł wypisujący informacje zwrotne do użytkownika. Można je zobaczyć poprzez kliknięcie na przycisk otwierający konsolę.

### Model danych

Moduł posiada własny model dancyh, który przechowuje zbiór wiadomości. Wiadomość ma następujący format:

1. Type- typ wiadomości
2. Title- opcjonalny nagłówek wiadomości
3. Message- treść wiadomości

Ponadto przechowywane są liczniki dla poszczególnych typów wiadomości oraz ustawienia ich wyświetlania.

### Sposób działania modułu

Moduł podłączony jest do widoku za pomocą funkcji, które wywołują dodanie konkretnego typu wiadomośći do danych. Moduł posiada także zdefiniowane funkcje, które otwierają i zamykają konsolę.

### Ustawienia

aa

|  |  |
| --- | --- |
| **Ustawienie** | **Opis** |
|  |  |

## shortcut

### Opis

Moduł obsługuje skróty klawiszowe, za pomocą których mogą być wywoływane odpowiednie funkcjonalności interfejsu.

### Model danych

Moduł wykorzystuje zewnętrzną bibliotekę do potrzeb działania interfejsu.

### Sposób działania modułu

Za pomocą odpowiedniej funkcji kontrolera skrót klawiszowy jest dodawany do silnika. Można go również usunąć za pomocą silnika. Wciśnięcie skrótu klawiszowego inicjuje zadane zdarzenie, które zdefiniowane jest przy dodawaniu skrótu klawiszowego do silnika.

### Zdarzenia

Moduł nie wywołuje żadnych zdarzenń ani nie nasłuchuje na żadne zdarzenia

### zdefiniowane skróty klawiszowe

|  |  |  |
| --- | --- | --- |
| **Skrót** | **Moduł definiujący** | **Wysyłane zdarzenie** |
| Escape | SpecialEvents | shortcutEscape |
| Ctrl+S | SpecialEvents | shortcutSave |
| Ctrl+A | SpecialEvents | shortcutSelectAll |
| Ctrl+C | SpecialEvents | shortcutCopy |
| Ctrl+X | SpecialEvents | shortcutCut |
| Ctrl+V | SpecialEvents | shortcutPaste |
| Ctrl+Z | SpecialEvents | shortcutBack |
| Ctrl+Y | SpecialEvents | shortcutForward |
| Delete | SpecialEvents | shortcutDelete |
| Shift+N | SpecialEvents | shortcutNew |
| Shift+Q | SpecialEvents | switchConsole |
| Shift+C | SpecialEvents | switchToCF |
| Shift+D | SpecialEvents | switchToDF |
| Shift+R | SpecialEvents | redeploySsdl |
| Shift+Z | History | shortcutBackAll |
| Shift+Y | History | shortcutForwardAll |
| Shift+I | Resizer | enlargeGraph |
| Shift+O | Resizer | reduceGraph |
| Shift+U | Resizer | restoreGraph |

## specialEvents

### Opis

Dopisuje zdarzenia odpowiedzialne za inicjację przepływu najważniejszych zdarzeń, które muszą wykonać się jako pierwsze przy uruchomieniu interfejsu.

### Model danych

Moduł nie posiada modelu danych

### Sposób działania modułu

Przypisuje do zdarzeń JavaScript zdarzenia silnika. Ponadto definiuje większość skrótów klawiszowych.

### Zdarzenia

**Zdarzenia wysyłane przez moduł ”**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| onEscape | Wywołuje zamknięcie wszystkich elementów wizualnych, które zostały wcześniej otworzone. Zdarzenie przypisane jest do JavaScriptowej funkcji “onclick” |

## subgraphNavigator

### Opis

Zapewnia możliwość przechodzenia po grafach oraz podgrafach, które mogą zawierać poszczególne wierzchołki.

### Model danych

Moduł nie posiada własnego modelu danych

### Sposób działania modułu

Modul przy każdym odświeżeniu widoku buduje nową ścieżkę grafu, który jest aktualnie wyświetlany w interfejsie graficznym. Pozwala na przechodzenie pomiędzy dowolnymi elementami drzewa jego podgrafów lub grafów macierzystych. Korzysta przy tym z funkcjonalności modułu contextMenu do wyświetlania drzewa grafu i podgrafów

### Zdarzenia

**Zdarzenia obsługiwane przez moduł**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| initView | Inicjalizuje część graficzną modułu |
| refreshView | Pokazuje aktualną ścieżkę grafów wyświetlanym drzewie |

**Zdarzenia wysyłane przez moduł ”**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| onEscape | Zamyka wyświetlane elementy graficzne |
| historyChanged | Dodaje zmianę podmiany grafu do historii |
| replaceSsdl | Przeładowuje ssdl oraz inicjalizuje przeładowanie widoku |

## ViewResizer

### Opis

Inicjuje zmienne ustawień odpowiedzialne za konfigurację przeglądarki użytkownika wymagane do pracy z interfejsem oraz rozmieszczenia obiektów w interfejsie.

### Model danych

Moduł nie posiada modelu danych

### Sposób działania modułu

Definiuje zestaw funkcji odpowiedzialnych za przekształcenia okna przeglądarki dostosowując interfejs do napotkanych warunków (np rozdzielczość).

Dołącza się głównie do zdarzeń języka JavaScript.

### Zdarzenia

**Zdarzenia obsługiwane przez moduł**

|  |  |
| --- | --- |
| **Zdarzenie** | **Opis działania** |
| initView | Dopasowuje interfejs do rozmiaru okna. Definiuje podstawowe zmienne |
| refreshView | Wykonuje operacje na kanwie: dostosowuje ją do wyświetlanego grafu. |

# Zdarzenia

### Konwencja

Do Opisu przepływu danych między zdarzeniami przyjęto następującą konwencję:



Module1: Elipsa symbolizująca moduł wywołujący zdarzenie

InvokeEvent: Strzałka wychodząca z elipsy, wchodząca do prostokąta symbolizuje wywołanie zdarzenia przez moduł

Event: Prostokąt symbolizujący zdarzenie. Etykieta jest nazwą zdarzenia

ListenEvent Strzałka wychodząca z prostokąta, wchodząca do elipsy symbolizuje reakcję modułu na zdarzenie, na które moduł ten nasłuchuje.

Module2: Elipsa symbolizująca moduł obsługujący zdarzenie, na które nasłuchuje.

## addBlankNode

### Opis

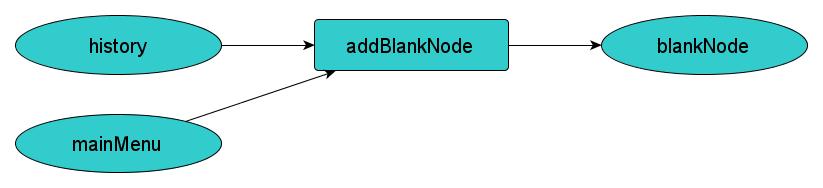
Zdarzenie wywoływane jest przez użytkownika za pośrednictwem menu głównego bądź za pośrednictwem modułu historii. Zdarzenie obsługiwane rzez moduł “blankNode” dodaje do grafu pusty obiekt i wywołuje moduł “nodeVisualiser”, który rysuje dany wierzchołek.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| History | Wywoływacz |
| mainMenu | Wywoływacz |
| blankNode | Słuchacz |

### graf przepływu



## atomicParseError

### Opis

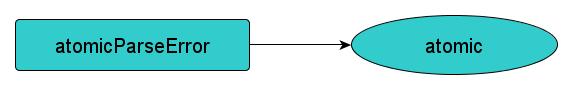
Zdarzenie wywoływane w momencie, gdy połączenie Ajax pobierające dane o usługach atomowych zwróci błąd

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| Atomic | Słuchacz |

### graf przepływu



## atomicparserperformed

### Opis

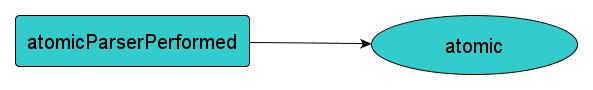
tODO

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| Atomic | Słuchacz |

### graf przepływu



## deploySSDL

### Opis

Zdarzenie wywoływane przez moduł “ssdlParser”, informujące o sparsowaniu danego grafu do postaci danych wykorzystywanych w interfejsie, które obsługiwane jest przez moduł “deployer” znajdujący położenie wierzchołków grafu

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| ssdlParser | Wywoływacz |
| Deployer | Słuchacz |

### graf przepływu



## engineLoaded

### Opis

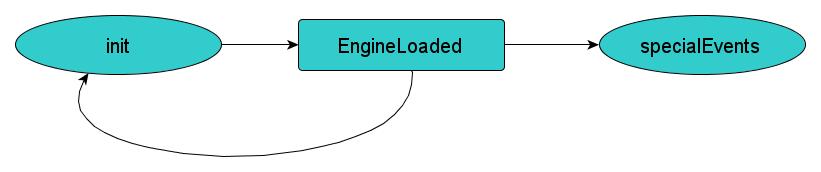
EngineLoaded jest to zdarzenie, które wywoływane jest przy inicjalizacji interfejsu. Ładuje SSDL.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| Init | Wywoływacz, Słuchacz |
| specialEvents | Słuchacz |

### graf przepływu



## enlargeGraph

### Opis

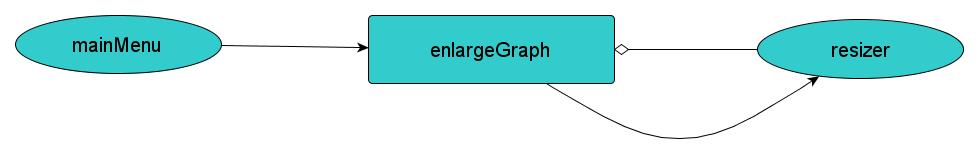
Zdarzenie w momencie, gdy konieczne żądane jest powiększenie grafu. Obsługiwane jest przez moduł “resizer”, który zmienia ustawienia rysowania wierzchołków grafu, a następnie komunikuje się z modułem “nodeVisualiser”, aby graf został przerysowany. Ponadto moduł “resizer” definiuje skrót klawiszowy, za pomocą którego graf może być powiększony.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| mainMenu | Wywoływacz |
| Resize | Słuchacz |

### graf przepływu



## formSubmitted

### Opis

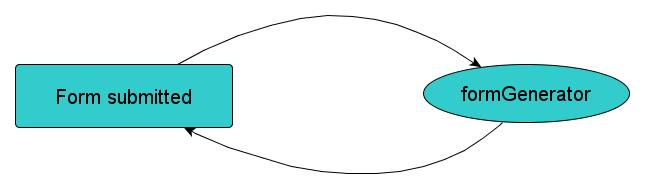
Zdarzenie wywoływane jest w momencie kliknięcia przycisku zapisz znajdującego się pod formularzem. Przekazuje dane z formularza zmodyfikowanego przez użytkownika do wierzchołka i zamyka formularz. Następuje również wyczyszczenie danych w celu uniknięcia ich nadpisania przy następnym wykorzystaniu formularza.

### Powiązane moduły

**Zdarzenia obsługiwane przez moduł**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| formGenerator | Wywoływacz, Słuchacz |

### graf przepływu



## historyChanged

### Opis

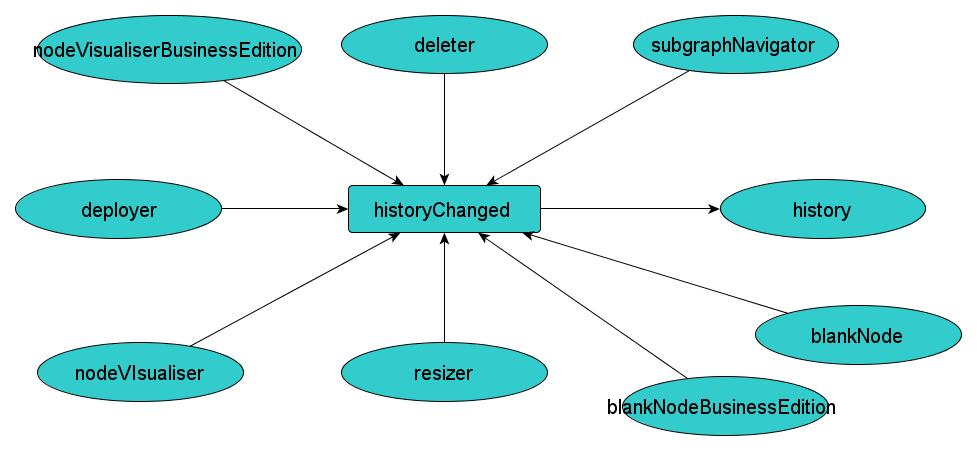
Zdarzenie wywoływane przez szereg różnych modułów, które zapewnia komunikację z modułem historii. Moduł historii obsługując zdarzenie dodaje zmianę do historii zmian, przez co umożliwia cofnięcie bądź powtórzenie zmian.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| Deployer | Wywoływacz |
| nodeVisualiserBusinessEdition | Wywoływacz |
| Deleter | Wywoływacz |
| subgraphNavigator | Wywoływacz |
| blankNode | Wywoływacz |
| resizer | Wywoływacz |
| nodeVisualiser | Wywoływacz |
| History | Słuchacz |

### graf przepływu



## historyDeleteBlanknode

### Opis

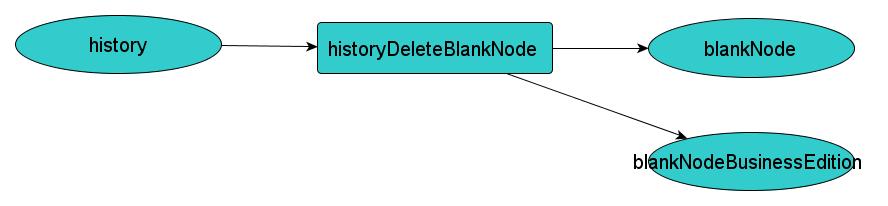
Zdarzenie wywoływane przez moduł historii w celu umożliwienia przechowania danych przez moduł “blankNode“, który tworzy nowy obiekt wierzchołka na potrzeby ewentualnego przekazania go później do modułu historii przy zmianach wprowadzanych przez użytkownika.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| History | Wywoływacz |
| blankNode | Słuchacz |
| blankNodeBusinessEdition | Słuchacz |

### graf przepływu



## initView

### Opis

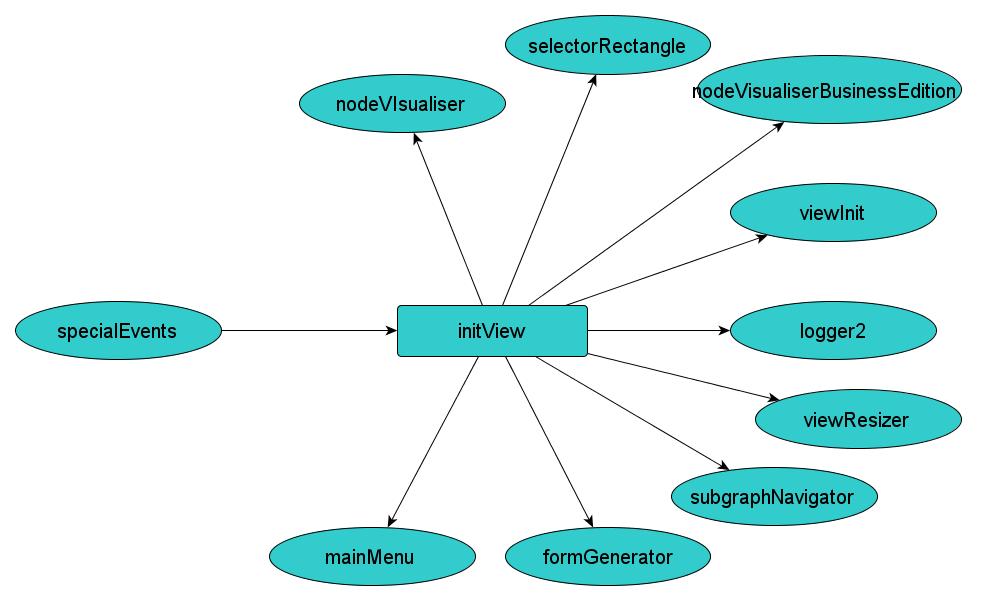
Zdarzenie specjalne wywoływane na początku działania interfejsu. Informuje o załadowaniu widoku.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| specialEvents | Wywoływacz |
| nodeVisualiser | Słuchacz |
| selectorRectangle | Słuchacz |
| nodeVisualiserBusinessEdition | Słuchacz |
| viewInit | Słuchacz |
| Logger2 | Słuchacz |
| viewResizer | Słuchacz |
| subgraphNavigator | Słuchacz |
| formGenerator | Słuchacz |
| mainMenu | Słuchacz |

### graf przepływu



## onEscape

### Opis

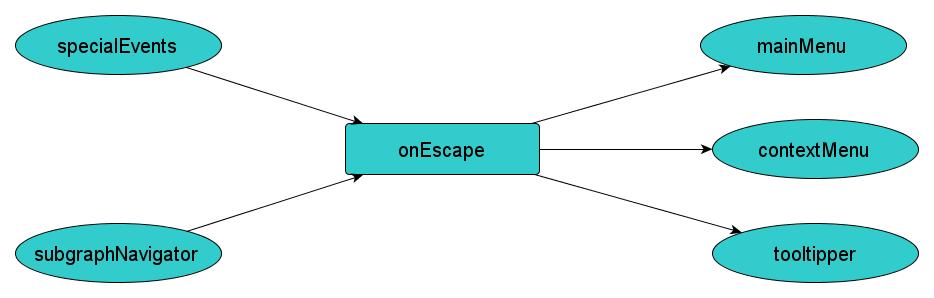
Zdarzenie wywoływane w momencie kliknięcia myszką na na pustym polu znajdującym się na kanwie pozwalające na informację o potrzebie ukrycia elementów widoku pokazywanych przez dane moduły

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| specialEvents | Wywoływacz |
| subgraphNavigator | Wywoływacz |
| mainMenu | Słuchacz |
| contextMenu | Słuchacz |
| tooltipper | Słuchacz |

### graf przepływu



## openForm

### Opis

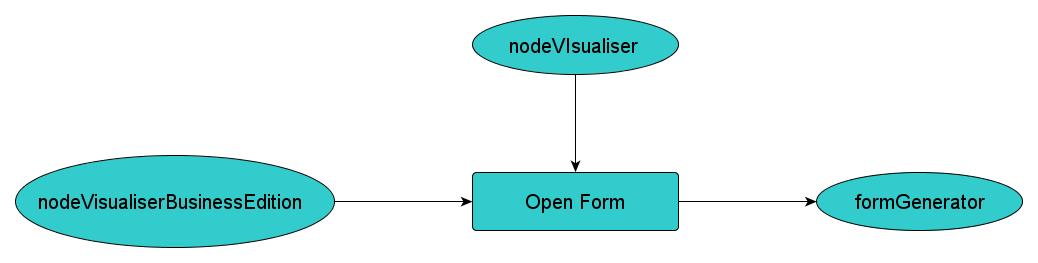
Zdarzenie reaguje na akcję otwarcia formularza. Akcja ta może być wywołana przez użytkownika za pośrednictwem modułów obsługujących wizualizację wierzchołków.

### Powiązane moduły

**Zdarzenia obsługiwane przez moduł**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| nodeVisualiserBusinessEdition | Wywoływacz |
| nodeVisualiser | Wywoływacz |
| formGenerator | Słuchacz |

### graf przepływu



## SSDLloaded

### Opis

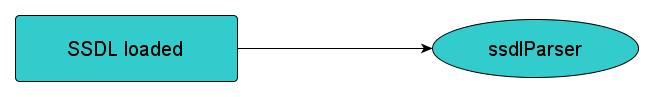
Zdarzenie wywoływane automatycznie przodczas zdarzenia EngineLoaded. Informuje, że ssdl został załadowany.

### Powiązane moduły

**Zdarzenia obsługiwane przez moduł**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| ssdlParser | Słuchacz |

### graf przepływu



## ssdlParsed

### Opis

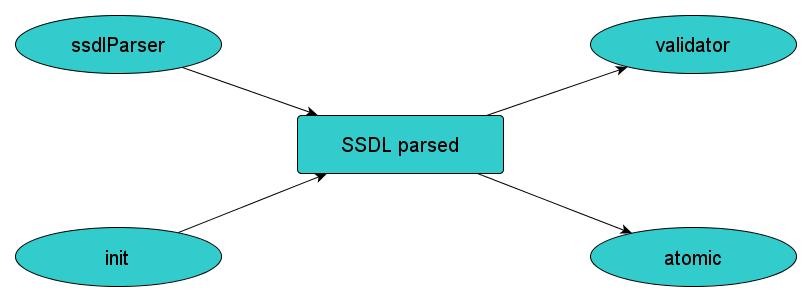
Zdarzenie informujące o fakcie przeparsowania SSDL

### Powiązane moduły

**Zdarzenia obsługiwane przez moduł**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| ssdlParser | Wywoływacz |
| Init | Wywoływacz |
| Validator | Słuchacz |
| Atomic | Słuchacz |

### graf przepływu



## parserGotData

### Opis

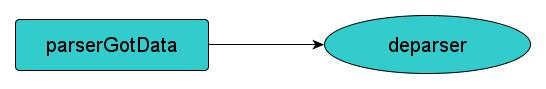
Zdarzenie wywoływane w momencie, gdy todo

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| Deparser | Słuchacz |

### graf przepływu



## parserPassed

### Opis

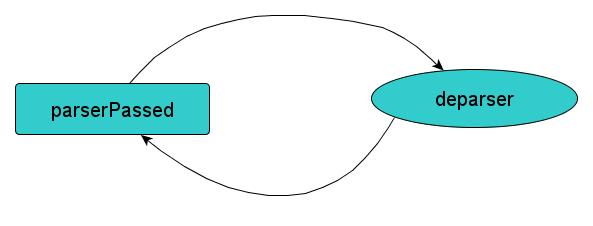
Zdarzenie wywoływane w momencie, gdy połączenie Ajax pobierające dane o usługach atomowych zakończy się powodzeniem

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| deparser | Wywoływacz, słuchacz |

### graf przepływu



## parsertranserror

### Opis

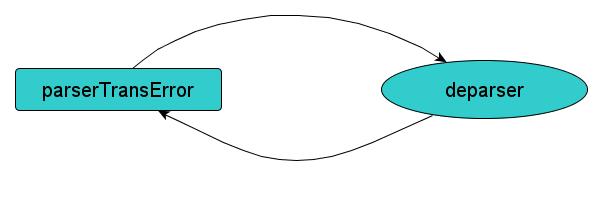
todo

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| Deparser | Wywoływacz, słuchacz |

### graf przepływu



## redeployssdl

### Opis

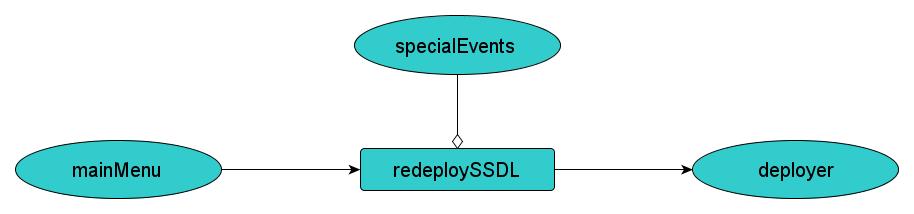
Zdarzenie wywołwane za pomocą menu głównego, bądź skrótu klawiszowego zdefiniowanego w module secialEvents, które obsługiwane jest przez moduł deployer, który rozmieszcza graf.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| mainMenu | Wywoływacz |
| deployer | Słuchacz |

### graf przepływu



## reducegraph

### Opis

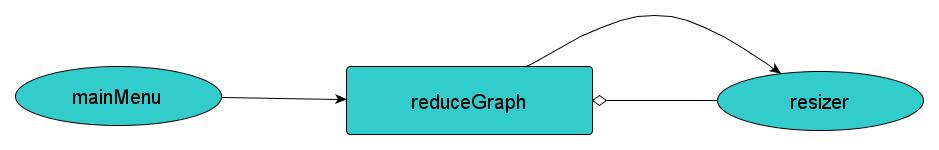
Zdarzenie w momencie, gdy konieczne żądane jest pomniejszenie grafu. Obsługiwane jest przez moduł “resizer”, który zmienia ustawienia rysowania wierzchołków grafu, a następnie komunikuje się z modułem “nodeVisualiser”, aby graf został przerysowany. Ponadto moduł “resizer” definiuje skrót klawiszowy, za pomocą którego graf może być pomniejszony.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| mainMenu | Wywoływacz |
| resizer | Słuchacz |

### graf przepływu



## selectnone

### Opis

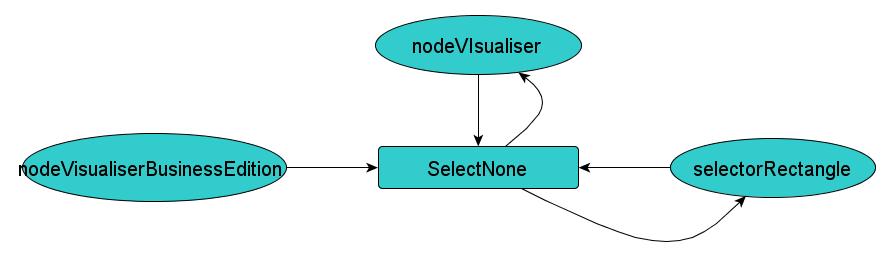
Zdarzenie odpowiadające za odznaczenie zaznaczonych elementów grafu

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| nodeVisualiser | Wywoływacz, słuchacz |
| nodeVisualiserBusinessEdition | Wywoływacz, słuchacz |
| selectorRectangle | Wywoływacz, słuchacz |

### graf przepływu



## shortcutBack

### Opis

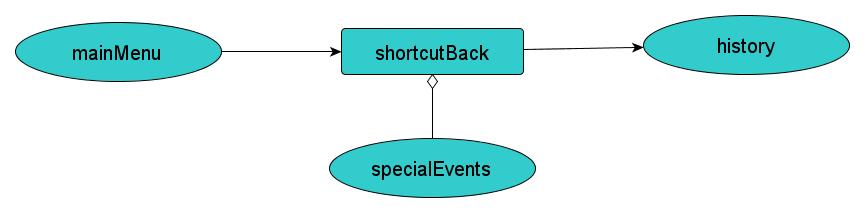
Zdarzenie wysyłane w momencie żądania przez użytkownika cofnięcia ostatniej dokonanej zmiany. Obsługiwane jest przez moduł historii, a skrót klawiszowy do wywołania tego zdarzenia zdefiniowany jest w module “specialEvents”

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| mainMenu | Wywoływacz |
| History | Słuchacz |

### graf przepływu



## shortcutbackall

### Opis

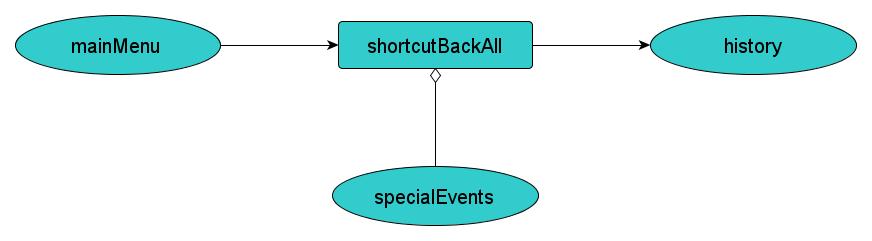
Zdarzenie wysyłane w momencie żądania przez użytkownika cofnięcia do początku historii zmian. Obsługiwane jest przez moduł historii, a skrót klawiszowy do wywołania tego zdarzenia zdefiniowany jest w module “specialEvents”

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| mainMenu | Wywoływacz |
| History | Słuchacz |

### Graf przepływu



## shortcutDelete

### Opis

Zdarzenie wywoływane w momencie żądania przez użytkownika usunięcia zaznaczonego wierzchołka poprzez wciśnięcie klawisza “delete”.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| deleter | Słuchacz |

### graf przepływu



## shortcutEscape

### Opis

Zdarzenie wywoływane w momencie wciśnięcia przez użytkownika klawisza “Escape” pozwalające na zamknięcie formularza powiązanego z modułem “formGenerator”

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| formGenerator | słuchacz |

### graf przepływu



## shortcutForward

### Opis

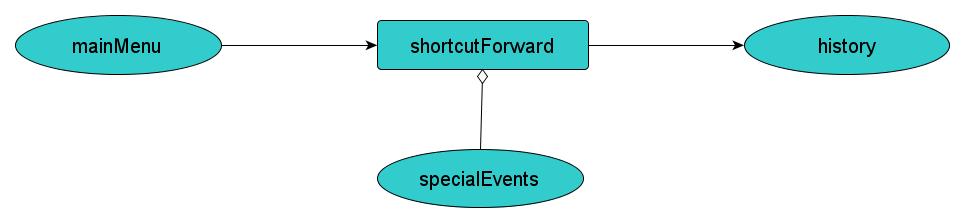
Zdarzenie wysyłane w momencie żądania przez użytkownika powtórzenia ostatniej dokonanej zmiany. Obsługiwane jest przez moduł historii, a skrót klawiszowy do wywołania tego zdarzenia zdefiniowany jest w module “specialEvents”

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| mainMenu | Wywoływacz |
| history | Słuchacz |

### graf przepływu



## shortcutForwardAll

### Opis

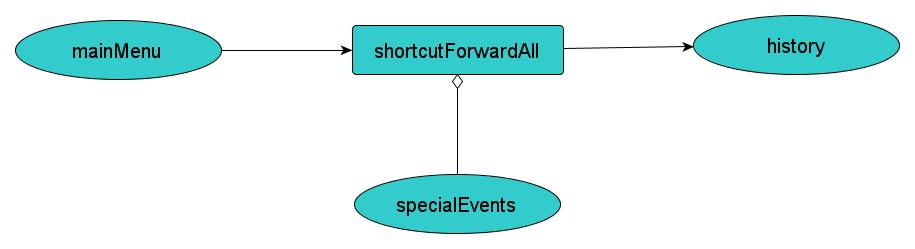
Zdarzenie wysyłane w momencie żądania przez użytkownika powtórzenia zmian do ostatniej dokonanej zmiany. Obsługiwane jest przez moduł historii, a skrót klawiszowy do wywołania tego zdarzenia zdefiniowany jest w module “specialEvents”

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| mainMenu | Wywoływacz |
| History | Słuchacz |

### graf przepływu



## ssdlDeployed

### Opis

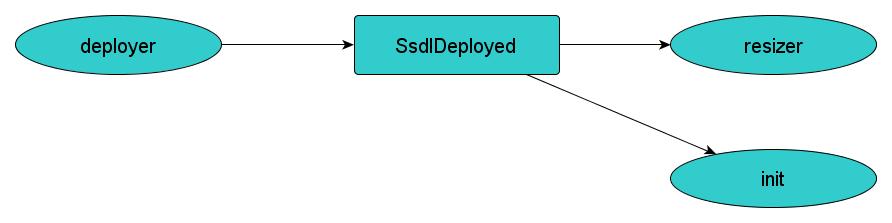
Zdarzenie wykorzystywane do inicializacji modułu resizer, inforumuje o rozmieszczeniu grafu na kanwie.

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| deployer | Wywoływacz |
| resizer | Słuchacz |
| init | Słuchacz |

### graf przepływu



## switchConsole

### Opis

Zdarzenie odpowiadające za pokazywanie widoku konsoli. Skrót klawiszowy zdefiniowany jest w module “specialEvents”

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| logger2 | Słuchacz |

### graf przepływu



## switchToCF

### Opis

Zdarzenie wywoływane w momencie żądania zmiany trybu wyświetlania na tryb przepływu kontroli. Skrót klawiszowy do wywołania zdarzenia zdefiniowany jest w module specialEvents

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| nodeVisualiser | słuchacz |

### graf przepływu



## switchToDF

### Opis

Zdarzenie wywoływane w momencie żądania zmiany trybu wyświetlania na tryb przepływu danych. Skrót klawiszowy do wywołania zdarzenia zdefiniowany jest w module specialEvents

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| nodeVisualiser | słuchacz |

### graf przepływu



## validationError

Zdarzenie wywoływane w momencie, gdy wystąpi błąd podczas walidacji

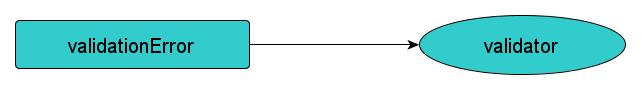
### Opis

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| validator | Słuchacz |

### graf przepływu



## validationFailed

### Opis

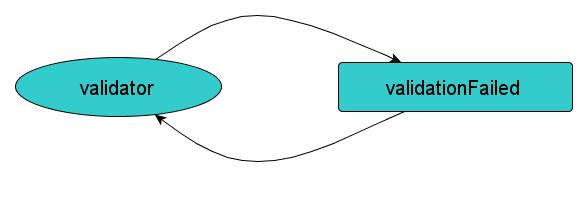
Zdarzenie wywoływane za pomocą Ajax poprzez moduł validator.Wyświetla wiadomości o błędach, które wystąpiły przy walidacji

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| validator | Wywoływacz, słuchacz |

### graf przepływu



## validationPassed

### Opis

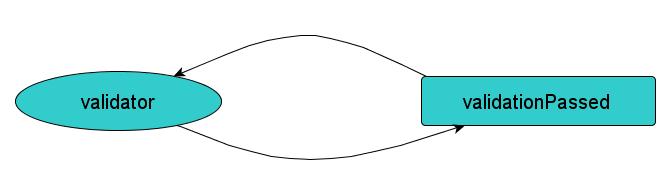
Zdarzenie wywoływane w momencie, gdy walidacja przebiegła pomyźlnie

### Powiązane moduły

**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| validator | Wywoływacz, słuchacz |

### graf przepływu



## validationPerformed

### Opis

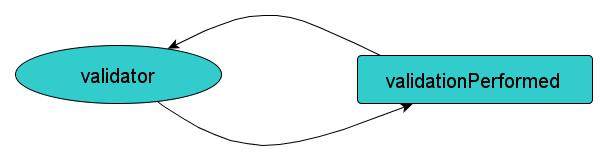
Zdarzenie wywoływane w momencie zakończenia walidacji. Obsługiwane jest przez moduł “validator” , który przetwarza wyniki walidacji.

### Powiązane moduły

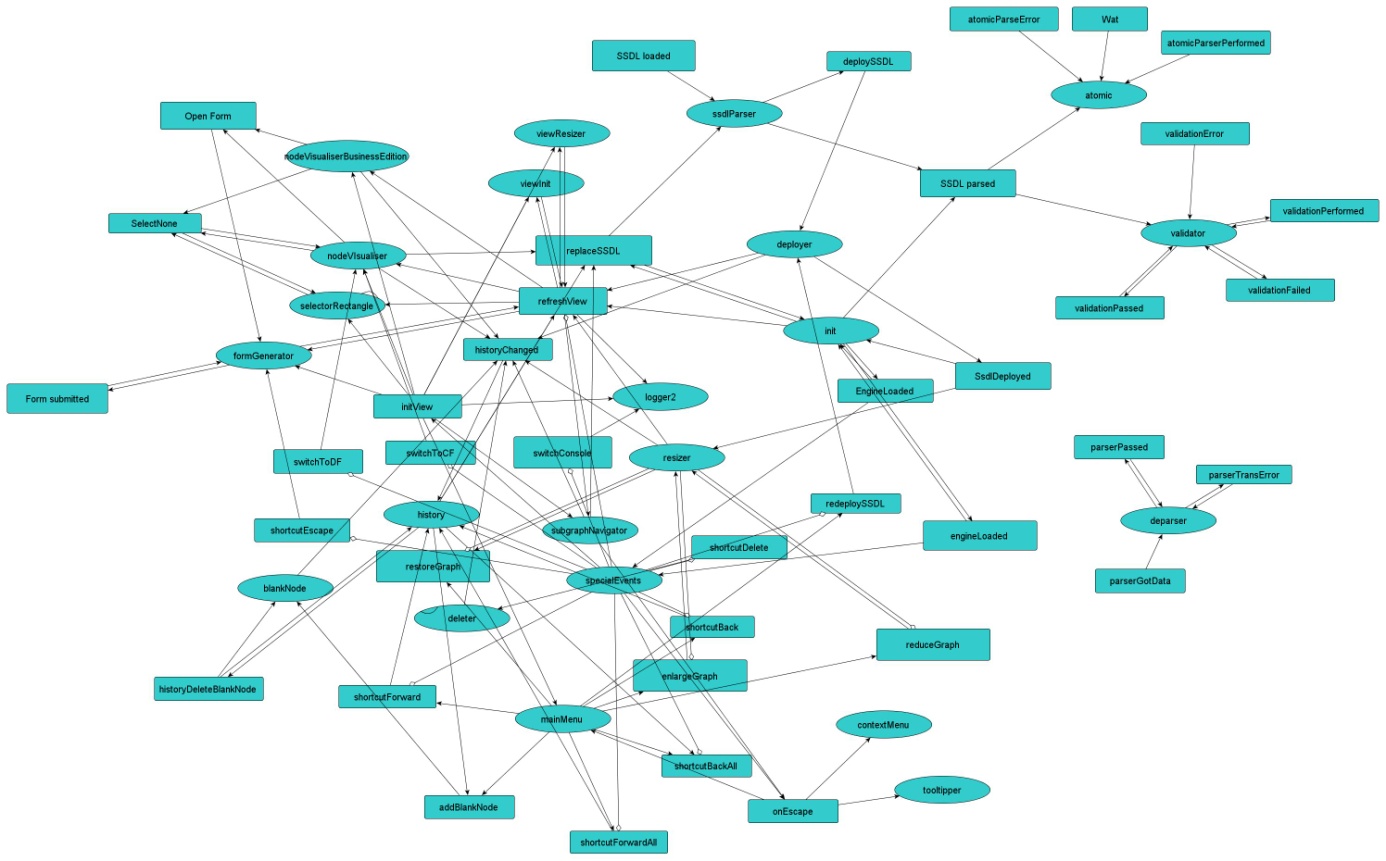
**Zdarzenia powiązane z modułem**

|  |  |
| --- | --- |
| **Moduł** | **Rola** |
| validator | Wywoływacz, słuchacz |

### graf przepływu



## Wszystkie zdarzenia

Poglądowa mapa zdarzeń wygląda następująco:  


Graf zdarzeń w wysokiej rozdzielczośći dostępny jest pod adresem:  
TODO