Złożenie pracy online:
2019-03-14 11:55:31
Kod pracy:
4150/37094/CloudA

Adrian Śmierciak
(nr albumu: 22659)

Praca inżynierska

Deweloperski dodatek do przeglądarki umożliwiający zaawansowaną edycję stron internetowych.

Developer extension to the browser allowing advanced website editions.

Wydział: Wydział Nauk Społecznych i Informatyki

Kierunek: Informatyka

Specjalność: grafika 3d i multimedia

Promotor: dr Henryk Telega

Składam serdeczne podziękowania dla Promotora za cierpliwość, cenne wskazówki i wyrozumiałość.

Streszczenie

Celem niniejszej pracy inżynierskiej było stworzenie aplikacji, która pozwala na szybką, a zarazem zaawansowaną edycję stron internetowych. W tym celu podjęto się zaprojektowania i zaimplementowania rozszerzenia do przeglądarki Google Chrome z edytorem wyglądu portali. W trakcie tworzenia programu użyto technologii webowych takich jak HTML, CSS i JavaScript. Rozszerzenie umożliwia prostą ingerencję w strony internetowe, a także posiada bardziej zaawansowane funkcje dedykowane programistom. Są to następująco: formatowanie tekstu, ustawianie odstępów i marginesów, edycja tła i rozmiaru, zmiana obrazków, a także dostosowywanie pozycji i wyświetlania elementów. Korzystanie z programu przedstawia użytkownikowi popularne i często używane przez deweloperów właściwości języka CSS oraz demonstruje całokształt procesu edycji stron internetowych.

Słowa kluczowe

dodatek, rozszerzenie, edycja, strona, CSS, Chrome

Abstract

The purpose of this study was to create an application that allows the user to edit websites in a quick but at the same time advanced way. To this end, design and implementation of the extension to the Google Chrome browser have been undertaken which resulted in creating editor responsible for editing the appearance of websites. During the creation of the application, web technologies such as HTML, CSS and JavaScript were used. The extension not only allows for simple interference in websites but also has some more advanced functions dedicated to programmers. They are as follows: text formatting, setting spacing and margins, editing the background and size, changing pictures, as well as adjusting and displaying website elements. Using the application presents to the user popular and often used by developers properties of the CSS language (syntax) and demonstrates the process of editing websites.

Keywords

addition, extension, edit, page, CSS, Chrome

Spis treści

Wstęp

Niewielkie grono osób korzystających z Internetu jest świadome, że zawartości stron oraz jej kompozycje, można samodzielnie edytować i przerabiać wykorzystując przeglądarkę.

W tej pracy autor podjął się ułatwienia procesu ingerencji w strony internetowe, poprzez zaprojektowanie i stworzenie rozszerzenia do przeglądarki z edytorem wyglądu. Aplikacja dedykowana jest na popularną przeglądarkę internetową Google Chrome, a jej budowa oparta jest o webowe technologie takie jak HTML, CSS i JavaScript.

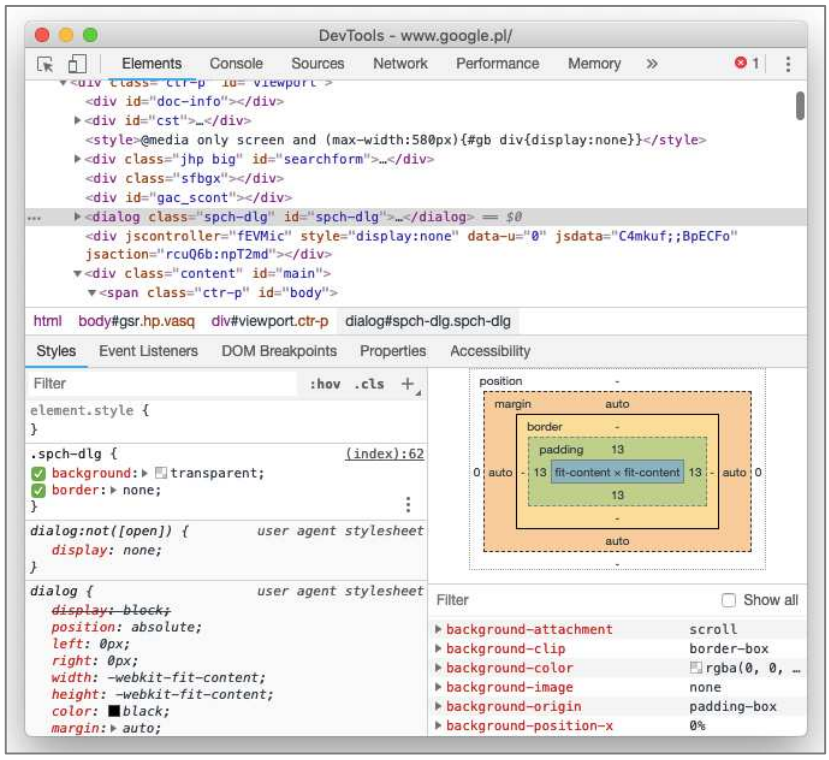
W rozdziale pierwszym przedstawiono aktualne sposoby edycji stron i problemy, które się z nimi wiążą. Drugi rozdział zawiera szczegółowy opis celu pracy i założenia procesu tworzenia aplikacji. W trzecim rozdziale przedstawiono najważniejsze informacje dotyczące implementacji programu oraz instalację rozszerzenia. W ostatnim rozdziale został opisany przebieg testów i osiągnięte wyniki.

W trakcie tworzenia programu wspomagano się głównie dokumentacją Chrome API, czyli Interfejsem Programowania Aplikacji od Google Chrome, a także różnego rodzaju literaturą webową zawierającą najnowsze standardy, takie jak CSS3, HTML5 czy ECMAScript6.

1. Edycja stron internetowych

Dla wielu użytkowników Internetu pojęcie edycji stron jest mało znane lub całkowicie obce. Aktualnie proces ten odbywa się za pomocą edytorów tekstowych, w których to programiści czy deweloperzy ingerują w kod, co przyczynia się do zmian wizualnych na stronach. Istnieje jeszcze jeden sposób na dokonywanie zmian w portalach. Nowsze przeglądarki oferują narzędzie deweloperskie (rysunek 1.1), pozwalające ingerować w strony internetowe, jednakże wymaga ono znajomości webowych technologii takich jak HTML czy CSS. Dodatkowo metoda ta, jak sama nazwa wskazuje jest przeznaczona dla deweloperów, przez co nie posiada przyjaznego interfejsu oraz może wydawać się skomplikowana.

Rys. 1.1. Przykład narzędzia deweloperskiego Google Chrome



Źródło: Opracowanie własne

Obecnie istnieje niewiele sposobów na edycję stron internetowych, a do ich prawidłowego wykorzystania potrzebna jest co najmniej dobra znajomość odpowiednich języków programowania. Brakuje narzędzi, które mogłyby uprościć zaawansowaną edycję stron nie wymagając przy tym innych programów, a także pozwoliłyby na ingerencję w wygląd portali osobom nie znającym technologii webowych.

2. Cel i założenia projektu

Celem pracy jest rozwiązanie problemu jakim jest brak narzędzia pozwalającego na szybką i łatwą, ale też zaawansowaną edycję stron. Do zrealizowania tego zadania autor podjął się zaprojektowania i zaimplementowania dodatku do przeglądarki z edytorem wyglądu stron internetowych.

Aplikacja zostanie stworzona w formie rozszerzenia, którego domyślną platformą będzie popularna przeglądarka Google Chrome. Główne możliwości edytora będą następujące: formatowanie tekstu, ustawianie odstępów i marginesów, edycja tła i rozmiaru, zmiana obrazków, a także dostosowywanie pozycji i wyświetlania elementów. Odrębną, bardziej zaawansowaną funkcją, będzie możliwość generowania wybranych zmian w postaci kodu języka CSS. Rozszerzenie zostanie dodatkowo wzbogacone o panel z ustawieniami, w którym użytkownik będzie miał możliwość dopasowania edytora do indywidualnych potrzeb czy upodobań. W panelu znajdować się będą następujące opcje: wybór języka (polski lub angielski), zmiana rozmiaru liter (dla osób niedowidzących lub mających problem z czytaniem tekstu pisanego małym rozmiarem czcionki) oraz dobór motywu.

3. Charakterystyka i implementacja

Rozdział ten poświęcony jest charakterystyce i implementacji projektu. Złożony jest z czterech podrozdziałów. W pierwszym opisane są używane technologie oraz narzędzia. W kolejnym zostały zaprezentowane funkcje użytkowe programu wraz ze sposobem ich zastosowania. Następny podrozdział przedstawia techniczne aspekty implementacji. Natomiast ostatni zawiera szczegółowy opis instalacji dodatku w przeglądarce.

3.1 Użyte technologie i narzędzia

W aplikacji opisywanej w niniejszej pracy wykorzystano następujące technologie i narzędzia: Chrome (Extension) API, język programowania JavaScript, biblioteka jQuery, biblioteka jQuery UI, wtyczka Spectrum, hipertekstowy język znaczników HTML, język CSS, a także program graficzny Adobe Illustrator CC.

Chrome API lub inaczej nazywany Extension API to Interfejs Programowania Aplikacji, zbudowany do tworzenia rozszerzeń przeglądarki Google Chrome. Rozszerzenia to programy, które pozwalają na dostosowanie zachowania przeglądarki i jej niektórych funkcji, w zależności od indywidualnych potrzeb lub preferencji użytkownika. Programiści i zwykli użytkownicy Internetu na co dzień korzystają z rozszerzeń. Są to zazwyczaj programy odpowiadające za blokowanie reklam, drobne narzędzia pozwalające przechowywać notatki, czy chociażby aplikacje czytające zaznaczony tekst. W prezentowanej pracy Chrome API jest wykorzystany do implementacji rozszerzenia z opisanym w rozdziale drugim edytorem stron internetowych.

JavaScript jest skryptowym językiem programowania opracowanym w 1995 roku przez firmę Netscape, a konkretniej przez amerykańskiego programistę Brendana Eicha. Najczęstsze zastosowania języka można znaleźć na stronach internetowych, aczkolwiek JavaScript rozwija się w takim tempie, że w ostatnich latach znajduje swoje wykorzystanie także w innych dziedzinach, takich jak na przykład programowanie robotów. W niniejszym projekcie język jest wykorzystany do obsługi wszystkich funkcji użytkowych aplikacji, a także do wysyłania zapisanych ustawień użytkownika do pamięci przeglądarki.

jQuery jest biblioteką stworzoną w celu ułatwienia korzystania z języka JavaScript. Jej główną funkcjonalnością jest manipulacja modelem DOM, do czego również została użyta w projekcie. Natomiast jQuery UI jest kolekcją dodatków, motywów oraz efektów graficznych zaprojektowanych za pomocą HTML, CSS i biblioteki jQuery.

Spectrum jest wtyczką napisaną w języku JavaScript, przy pomocy biblioteki jQuery. Jej główną i jedyną funkcją użytą w edytorze jest narzędzie do wybierania i zapisywania kolorów.

HTML to hipertekstowy język znaczników. Jest on używany do kreowania oraz tworzenia zawartości i struktury dokumentów hipertekstowych, takich jak strony internetowe. Język ten umożliwił stworzenie edytora wyglądu wraz ze stroną ustawień aplikacji.

CSS, czyli Kaskadowe Arkusze Stylów to język opracowany w 1996 roku przez organizację W3C. Odpowiada on za wyświetlanie oraz formę prezentacji stron internetowych. W opisywanej aplikacji język CSS użyty jest do nadania odpowiedniej oprawy graficznej wszystkim funkcjonalnym elementom.

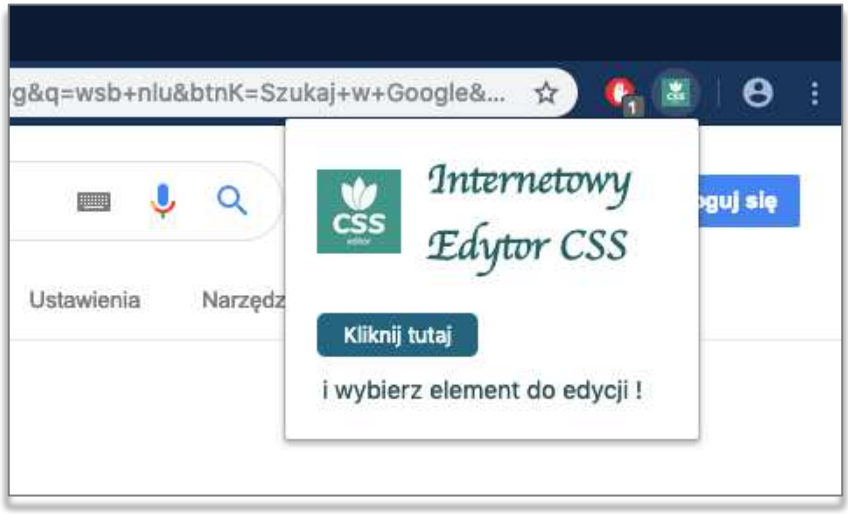
Adobe Illustrator CC jest graficznym programem stosowanym do edycji oraz tworzenia grafiki wektorowej. W prezentowanej pracy został wykorzystany do stworzenia logo rozszerzenia oraz flag dwóch państw, których języki są użyte w aplikacji.

3.2 Możliwości aplikacji i ich zastosowanie

URUCHOMIENIE APLIKACJI

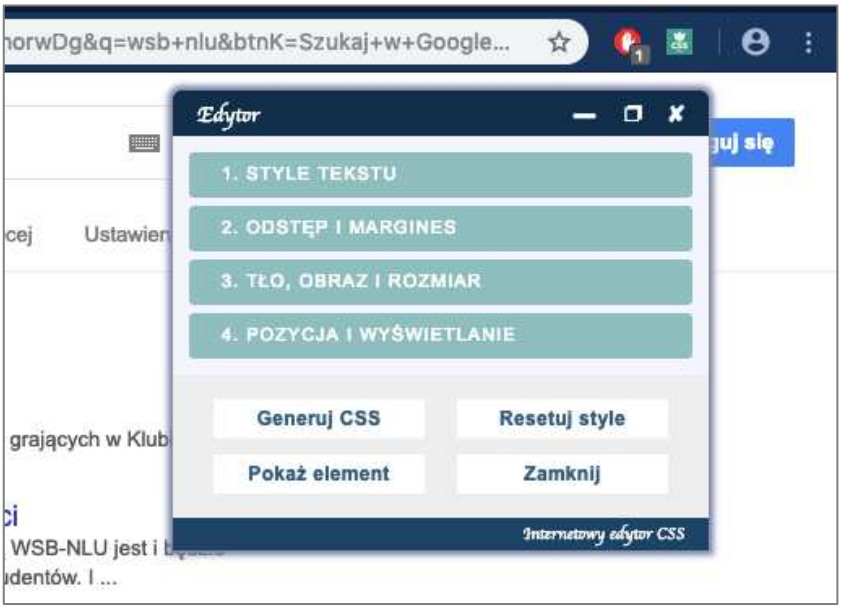
Istnieją dwa sposoby uruchomienia rozszerzenia. Pierwszy z nich to kliknięcie na ikonę z logo aplikacji w prawym górnym rogu przeglądarki. Zostanie wówczas wyświetlone małe okno (rysunek 3.1) z logiem i tytułem aplikacji oraz z przyciskiem *Kliknij tutaj*. Po kliknięciu przycisku, należy wybrać element ze strony do edycji. Może to być każdy element, począwszy od tekstu i obrazka, kończąc na przyciskach i formularzach. Po kliknięciu wybranego obiektu na stronie pojawi się okno edycji (rysunek 3.2).

Rys. 3.1. Początkowe okno rozszerzenia



Źródło: Opracowanie własne

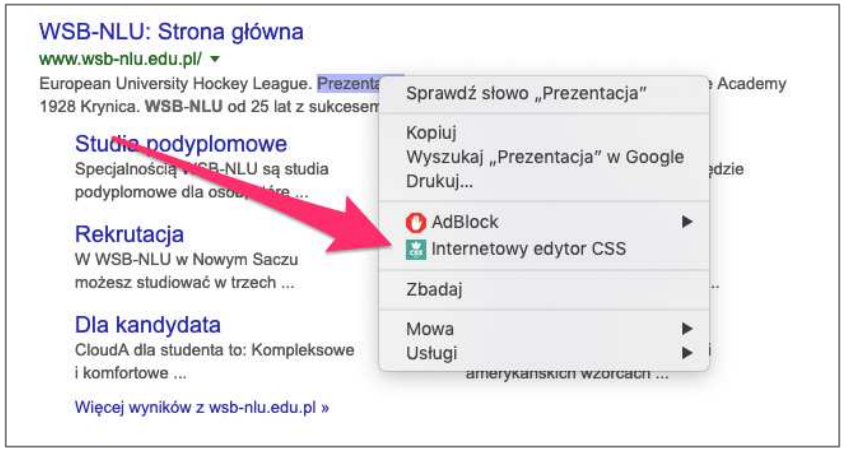
Rys. 3.2. Okno edycji



Źródło: Opracowanie własne

Drugi sposób uruchomienia rozszerzenia to kliknięcie prawym przyciskiem myszy na element, który ma być edytowany. Otworzy się wtedy menu kontekstowe z możliwością wyboru omawianego rozszerzenia (rysunek 3.3). Kliknięcie tej opcji spowoduje pojawienie się okna edycji na stronie.

Rys. 3.3. Menu kontekstowe z rozszerzeniem



Źródło: Opracowanie własne

GŁÓWNE FUNKCJE

Rozszerzenie posiada wiele funkcji użytkowych, przez co możliwości te zostały podzielone w oknie edycji na cztery grupy. Pierwsza z nich to *Style tekstu*, druga *Odstęp i margines*, trzecia została nazwana *Tło, obraz i rozmiar*, natomiast ostatnia *Pozycja i wyświetlanie*.

W pierwszej grupie znajdują się opcje pozwalające formatować tekst (rysunek 3.4). Są to następująco: zmiana koloru tekstu wraz z całą paletą kolorów oraz możliwością stworzenia własnych odcieni (rysunek 3.5), zmiana rozmiaru tekstu, dodanie stylu (pogrubienie, kursywa, podkreślenie), ustawienie wyrównania (do lewej, wycentrowany, do prawej), a także zmiana zawartości tekstu.

Rys. 3.4. Grupa *Style tekstu*

Źródło: Opracowanie własne

Rys. 3.5. Paleta edycji koloru

Źródło: Opracowanie własne

Opcje znajdujące się w drugiej grupie odpowiadają za ustawianie odstępów i marginesów (rysunek 3.6). Są one podzielone na dwie osobne rubryki, a każda z nich posiada pole, w którym należy wprowadzić liczbę, o jaką będzie zmieniana wartość odstępu czy marginesu. W obu przypadkach mamy cztery możliwości kierunków do wyboru, góra, dół, prawo i lewo.

Rys. 3.6. Grupa *Odstęp i margines*

Źródło: Opracowanie własne

Kolejna grupa odpowiada za edycję tła, obrazów i rozmiaru elementów (rysunek 3.7). Znajduje się w niej taka jak w grupie pierwszej paleta kolorów, lecz tym razem odpowiadająca za tło elementu. Dalej umieszczone są dwa pola, odpowiadające za wysokość oraz szerokość wybranego obiektu. Ostatnia opcja to podmiana obrazka. Po kliknięciu przycisku *Wybierz plik...* otwiera się okno, w którym użytkownik może wybrać zdjęcie bądź obraz z dysku. Następnie klikając drugi przycisk *Zmień obraz*, program przekonwertuje wybrany plik do typu obsługiwanego przez przeglądarkę i podmieni zdjęcie.

Rys. 3.7. Grupa *Tło, obraz i rozmiar*

Źródło: Opracowanie własne

W ostatniej grupie znajdują się dwa pola z możliwością wyboru pozycji i wyświetlania elementu (rysunek 3.8). Po kliknięciu w każde z pól wyświetli się menu, z którego użytkownik może wybrać odpowiednią opcję.

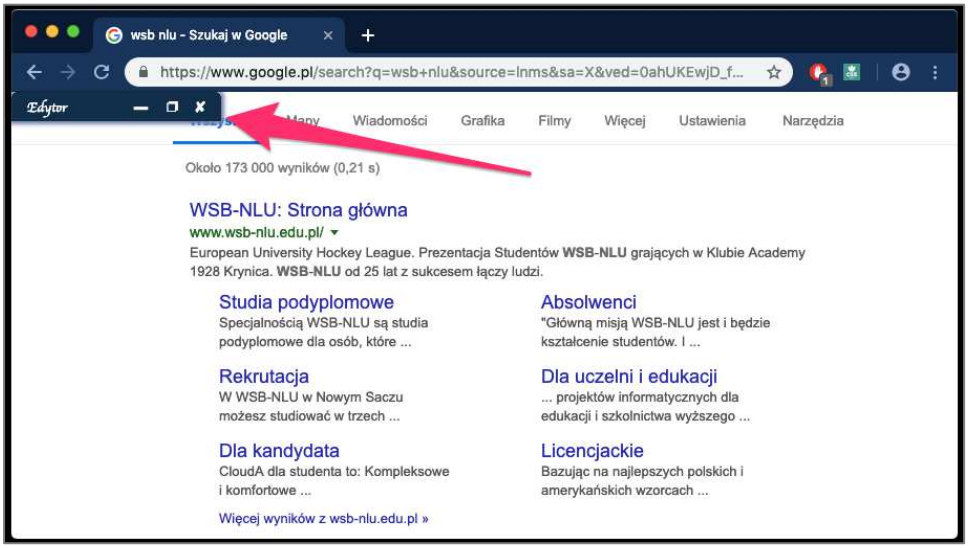
Rys. 3.8. Grupa *Pozycja i wyświetlanie*

Źródło: Opracowanie własne

FUNKCJE DODATKOWE

Na górze edytora z prawej strony znajdują się trzy ikony. Pierwsza z lewej odpowiada za minimalizowanie okna edycji i ukrywanie go w lewym górnym rogu strony, aby nie przeszkadzało w przeglądaniu portalu (rysunek 3.9). Kolejna ikona przywraca edytor do jego pierwotnego rozmiaru i miejsca. Ostatnia ikona zamyka edytor.

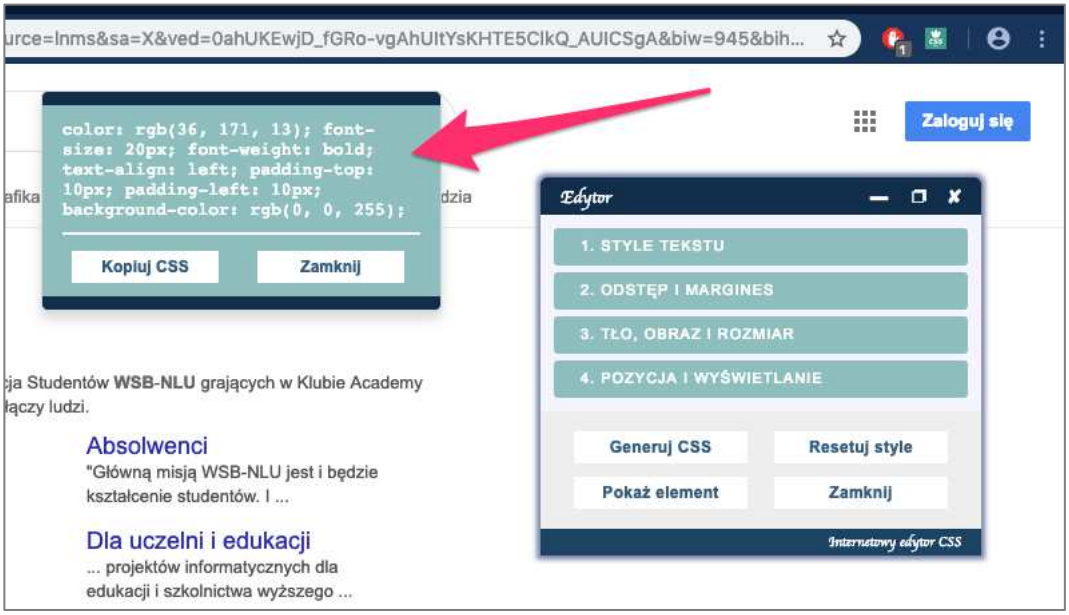
Rys. 3.9. Zminimalizowany edytor



Źródło: Opracowanie własne

Cztery przyciski w dolnej części edytora nie odpowiadają za edycję wybranego elementu, lecz mają inne zastosowanie. Pierwszy przycisk *Generuj CSS*, otwiera nowe okno z wygenerowanym kodem CSS odpowiadający za edycję wybranego elementu (rysunek 3.10). Ze wspomnianego okna można bezpośrednio skopiować kod za pomocą przycisku *Kopiuj CSS.* Następny przycisk *Resetuj style* przywraca edytowany element do pierwotnego stanu, nie zamykając przy tym okna edycji. Przycisk *Pokaż element* przewija stronę do wybranego elementu i podświetla go. Ostatni przycisk *Zamknij* odpowiada za zamknięcie edytora.

Rys. 3.10. Okno z wygenerowanym kodem CSS

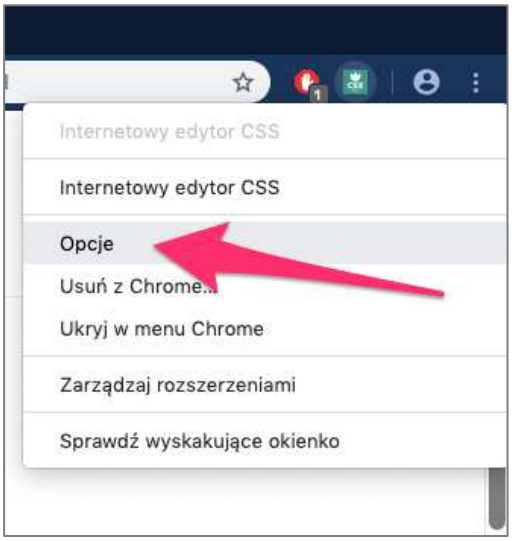


Źródło: Opracowanie własne

USTAWIENIA ROZSZERZENIA

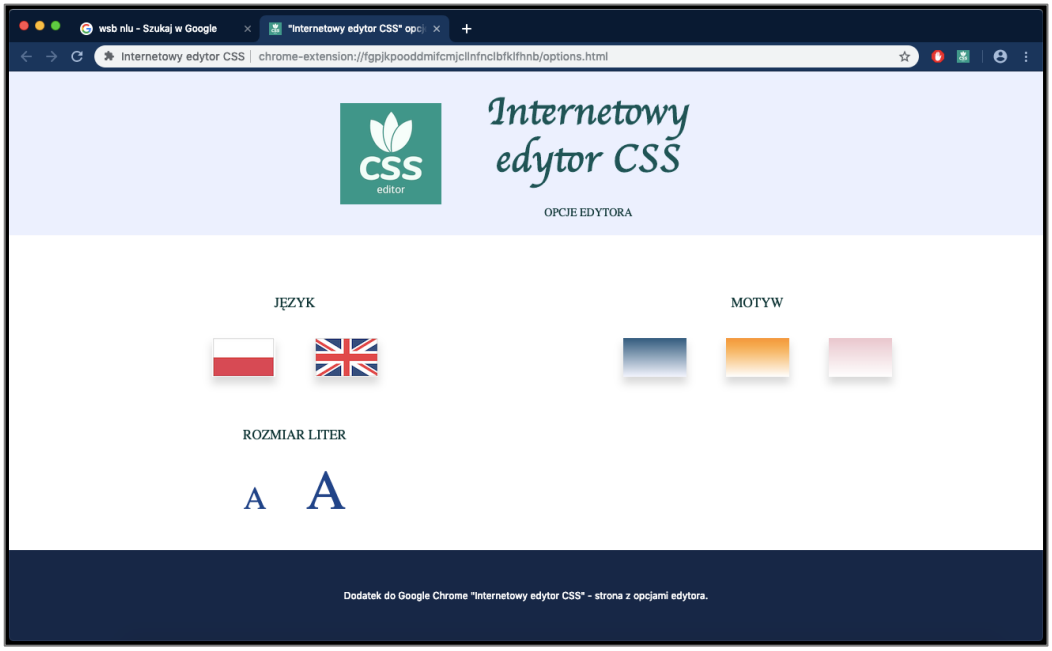
Kliknięcie prawym przyciskiem myszy w ikonę rozszerzenia daje możliwość przejścia do strony z ustawieniami aplikacji. W tym celu należy wybrać wariant *Opcje* w menu kontekstowym (rysunek 3.11). W przeglądarce zostanie otworzona nowa karta, a w niej załaduje się strona z ustawieniami edytora (rysunek 3.12).

Rys. 3.11. Menu kontekstowe rozszerzenia



Źródło: Opracowanie własne

Rys. 3.12. Strona z opcjami edytora



Źródło: Opracowanie własne

Strona z opcjami pozwala użytkownikowi na kilka zmian w edytorze. Pierwszą z nich jest zmiana języka, do wyboru są dwa warianty: język polski i język angielski. Kolejną możliwością zmiany, jaką posiada użytkownik, jest kolor motywu. Do wyboru są trzy opcje, o różnej kolorystyce. Ostatnią możliwą zmianą jest wielkość liter edytora. Opcja ta jest szczególnie przydatna dla osób niedowidzących lub mających trudności z czytaniem wyrazów pisanych małą czcionką. Poniżej przedstawiono porównanie wariantów ustawień edytora (rysunek 3.13, rysunek 3.14, rysunek 3.15).

Rys. 3.13. Język edytora

Źródło: Opracowanie własne

Rys. 3.14. Motyw edytora

Źródło: Opracowanie własne

Rys. 3.15. Wielkość liter w edytorze

Źródło: Opracowanie własne

3.3 Techniczne aspekty

Ten podrozdział ma na celu przedstawienie czytelnikowi podstawowych informacji na temat funkcjonowania programu.

SCHEMAT DZIAŁANIA APLIKACJI

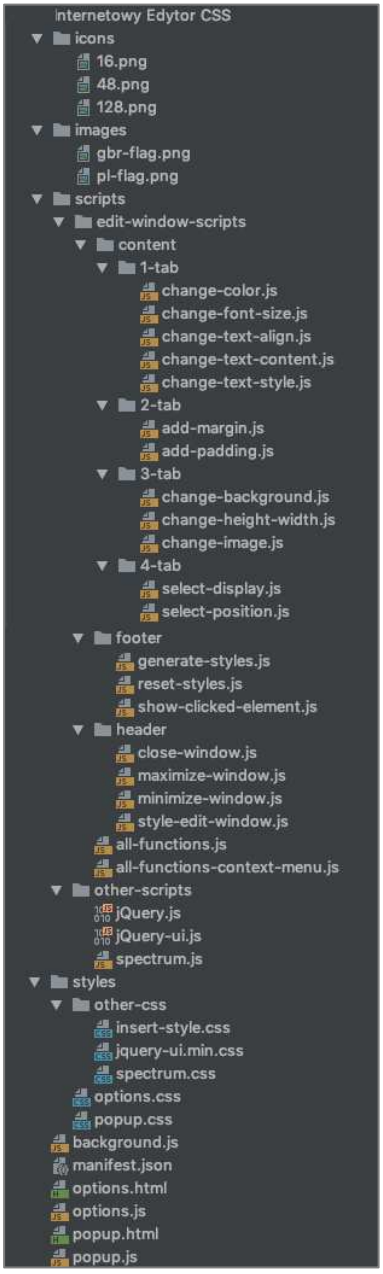
Działanie opisywanej aplikacji opiera się na przedstawionych poniżej czynnościach. Kolejność nieznacznie różni się w zależności od funkcji wykonywanej przez program.

1. Interakcja użytkownika
2. Przesłanie danych do pamięci przeglądarki
3. Pobranie danych z pamięci przeglądarki
4. Wstrzyknięcie skryptu do aktualnej karty
5. Wywołanie zdarzenia
6. Obsługa zdarzenia
7. Zmiana formy prezentacji elementu

ARCHITEKTURA APLIKACJI

Rozszerzenie zostało zaimplementowane na przeglądarkę Google Chrome, wobec czego jest dostosowane do systemów operacyjnych ją w pełni obsługujących, czyli Windows oraz macOS. Aplikacja składa się dwóch plików HTML wraz z odpowiadającymi dwoma plikami CSS, a także z 27 plików JavaScript, przez co została podzielona na kilka katalogów wraz z podkatalogami (rysunek 3.16).

Rys. 3.16. Struktura aplikacji



Źródło: Opracowanie własne

Podstawą aplikacji jest plik *manifest.json* (rysunek 3.17), zawierający wszystkie wymagane przez Google API informacje i dane na temat budowanego rozszerzenia takie jak: nazwa, opis, wersja, ikony z logo czy pliki odpowiadające za akcję przeglądarki.

Rys. 3.17. Fragment pliku *manifest.json*

```
{
  "name": "Internetowy edytor CSS",
  "description": "Edycja DOM",
  "version": "1.0",
  "options_page": "options.html",
  "manifest_version": 2,

  "icons": {
    "128": "icons/128.png",
    "48": "icons/48.png",
    "16": "icons/16.png"
  },

  "permissions": [
    "activeTab",
    "storage",
    "contextMenus"
  ],

  "background": {
    "scripts": ["background.js"],
    "persistent": false
  },
```

Źródło: Opracowanie własne

Stworzenie wyglądu rozszerzenia wymagało podzielenia procesu na trzy części. Wygląd pierwszego okna wyświetlającego się po kliknięciu ikony aplikacji zdefiniowano w plikach *popup.html* i *popup.css*, natomiast jego funkcjonalność została zawarta w pliku *popup.js* (rysunek 3.18). To ten plik odpowiada za wstrzyknięcie całej reszty skryptów do aktualnie otwartej strony.

Rys. 3.18. Fragment pliku *popup.js*

```

//wszystkie pliki z folderu edit-window-scripts
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/header/style-edit-window.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/header/close-window.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/header/minimize-window.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/header/maximize-window.js'});

chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/footer/show-clicked-element.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/footer/generate-styles.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/footer/reset-styles.js'});

chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/1-tab/change-color.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/1-tab/change-font-size.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/1-tab/change-text-style.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/1-tab/change-text-align.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/1-tab/change-text-content.js'});

chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/2-tab/add-padding.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/2-tab/add-margin.js'});

chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/3-tab/change-background.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/3-tab/change-height-width.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/3-tab/change-image.js'});

chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/4-tab/select-position.js'});
chrome.tabs.executeScript(null, {file: 'scripts/edit-window-scripts/content/4-tab/select-display.js'});

```

Źródło: Opracowanie własne

Podobnie prezentuje się struktura plików strony z ustawieniami edytora. Pliki odpowiedzialne za jej budowę i działanie to następująco: *options.html*, *options.css* oraz *options.js*. W ostatnim z nich po interakcji użytkownika następuje przesyłanie danych do pamięci przeglądarki z informacjami o zmienionych ustawieniach (rysunek 3.19). Przy następnym uruchomieniu aplikacji, pobierze ona wcześniej zapisane dane z pamięci Google Chrome i na ich podstawie wyświetli edytor z wybranymi przez użytkownika ustawieniami.

Rys. 3.19. Fragment pliku *options.js*

```
4
5     //JEZYKI
6     $('.pl').on( e: 'click', t: () => {
7         chrome.storage.sync.set({lang: 'pl'});
8     });
9
10    $('.eng').on( e: 'click', t: () => {
11        chrome.storage.sync.set({lang: 'eng'});
12    });
13
14    //MOTYWY
15    $('.dark').on( e: 'click', t: () => {
16        chrome.storage.sync.set({theme: 'dark'});
17    });
18
19    $('.light').on( e: 'click', t: () => {
20        chrome.storage.sync.set({theme: 'light'});
21    });
22
23    $('.kopi').on( e: 'click', t: () => {
24        chrome.storage.sync.set({theme: 'kopi'});
25    });
26
27    //ROZMIAR CZCIONKI
28    $('.normal').on( e: 'click', t: () => {
29        chrome.storage.sync.set({size: 'normal'});
30    });
```

Źródło: Opracowanie własne

Ostatnim plikiem z pierwszego szczebla hierarchii w strukturze plików jest *background.js*. Do jego funkcji należy wyświetlanie okna edycji poprzez menu kontekstowe, a schemat jego działania jest identyczny jak w przypadku pliku *popup.js*.

Reszta plików z rozszerzeniem *js* to grupa odpowiedzialna za prezencję i działanie okna edycji. Wszystkie pliki z podkatalogów *content*, *footer* i *header* posiadają w sobie definicje funkcji użytych do wywołania zdarzeń odpowiedzialnych za edycję wybranego elementu (rysunek 3.20). Następnie funkcje te są wywoływane z odpowiednimi parametrami w pliku *all-functions.js* lub *all-functions-context-menu.js*, w zależności od tego, jak została otwarta aplikacja (rysunek 3.21).

Rys. 3.20. Fragment przykładowego pliku z podkatalogu *content*

```
14
15      //margin-top plus i minus
16      function marginTop(selector, plusMinus) {
17          selector.on( e: 'click', t: function () {
18              clickedElementVariable.css('margin-top', parseFloat(clickedElementVariable.css('margin-top')) + plusMinus
                  + 'px');
19          });
20      }
21
22      //margin-bottom plus i minus
23      function marginBottom(selector, plusMinus) {
24          selector.on( e: 'click', t: function () {
25              clickedElementVariable.css('margin-bottom', parseFloat(clickedElementVariable.css('margin-bottom')) +
                  plusMinus + 'px');
26          });
27      }
28
29      //margin-left plus i minus
30      function marginLeft(selector, plusMinus) {
31          selector.on( e: 'click', t: function () {
32              clickedElementVariable.css('margin-left', parseFloat(clickedElementVariable.css('margin-left')) +
                  plusMinus + 'px');
33          });
34      }
35
36      //margin-left plus i minus
37      function marginRight(selector, plusMinus) {
```

Źródło: Opracowanie własne

Rys. 3.21. Fragment pliku *all-functions.js*

```
105
106                 //================================================
107                 //  Footer scripts
108                 //================================================
109
110                 //show-clicked-element.js
111                 showClickedElement($clickedElement);
112
113                 //generate-styles-js
114                 generateStyles($editWindow, generateWindowEngOrPl, $clickedElement);
115
116                 //reset-style.js
117                 resetStyles($editWindow, $clickedElement);
118
119
120                 //================================================
121                 //  Content scripts
122                 //================================================
123
124                 //=============== 1 ================
125
126                 //change-color.js
127                 changeColor($clickedElement);
128
```

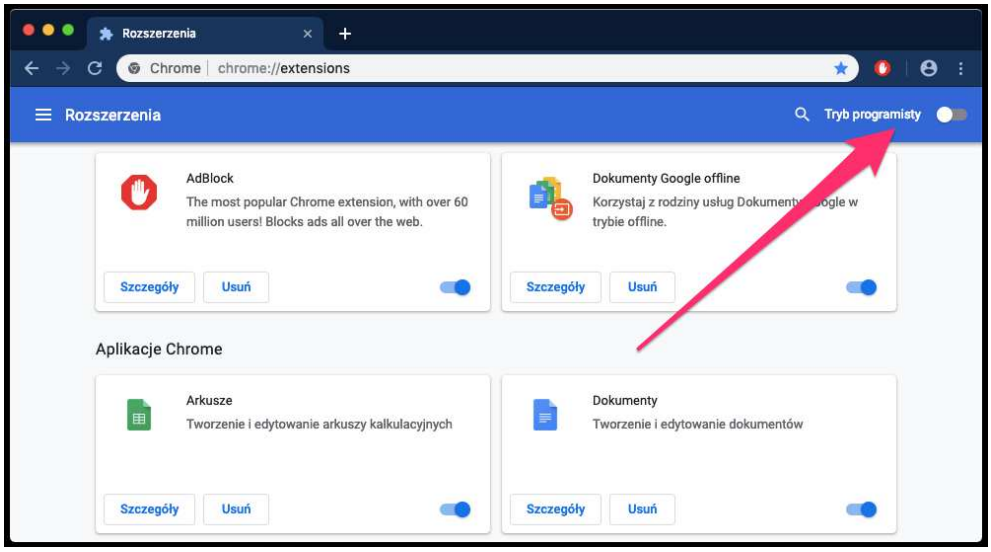
Źródło: Opracowanie własne

Pozostałe pliki to ikony i obrazki zastosowane w aplikacji, a także zewnętrzne biblioteki takie jak jQuery czy jQuery UI.

3.4 Instalacja rozszerzenia

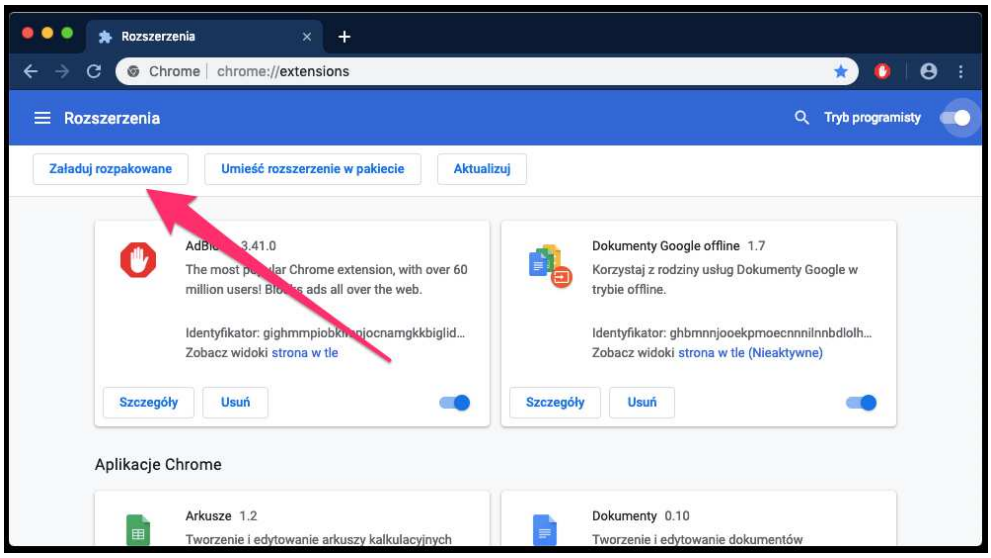
Do uruchomienia opisywanego w pracy rozszerzenia potrzebna jest przeglądarka internetowa Google Chrome. Można ją pobrać za pomocą każdej innej przeglądarki. Powinno się w tym celu wpisać w wyszukiwarkę frazę *Google Chrome*, a następnie przejść do pierwszej wyszukanej strony i dalej podążać za instrukcją instalacji. Gdy przeglądarka jest już zainstalowana, należy odszukać plik *PracaInzynierska.zip*, dołączony do pracy w formie załącznika na płytce, a następnie rozpakować go. W folderze znajduje się główny katalog rozszerzenia o nazwie *Internetowy Edytor CCS*. W dalszej kolejności należy otworzyć przeglądarkę Google Chrome i przejść pod adres *chrome://extensions/*. Aby móc dodać rozszerzenie, powinno się włączyć opcje *Tryb programisty* w prawym górnym rogu strony (rysunek 3.22). Po włączeniu opcji w lewym rogu pojawi się przycisk *Załaduj rozpakowane* (rysunek 3.23). Po kliknięciu przycisku zostanie wyświetlone okno z wyborem katalogu rozszerzenia. W tym miejscu należy wybrać wyżej wspomniany katalog *Internetowy Edytor CSS* i kliknąć *wybierz*. Jeśli rozszerzenie dodało się pomyślnie, w prawym górnym rogu przeglądarki powinna pojawić się ikona z logiem aplikacji (rysunek 3.24).

Rys. 3.22. Opcja *Tryb programisty*



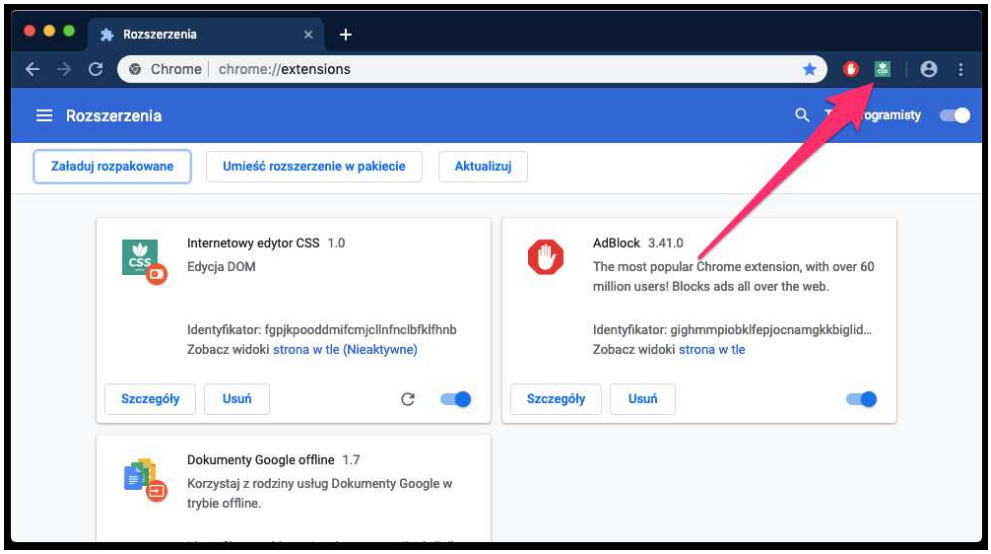
Źródło: Opracowanie własne

Rys. 3.23. Przycisk *Załaduj rozpakowane*



Źródło: Opracowanie własne

Rys. 3.24. Ikona z logiem aplikacji



Źródło: Opracowanie własne

4. Testy aplikacji

Niniejsza aplikacja testowana była na trzech różnych wersjach przeglądarki Google Chrome (70, 71, 72) oraz na dwóch systemach (Windows, macOS). Rozszerzenie było sprawdzane pod kątem szybkości oraz funkcjonalności na 20 popularnych portalach. Były to między innymi: *facebook.com*, *youtube.com*, *google.com*, *amazon.com*, *twitter.com* oraz *linkedin.com*. Program z założenia powinien działać na wszystkich stronach internetowych. Do testów zostały wybrane często odwiedzane portale i na każdym z nich wyniki były takie same. Rozszerzenie funkcjonowało szybko i sprawnie, najdłuższe opóźnienia wykonywania skryptów oscylowały w okolicach kilku milisekund. Nie wyłapano żadnych błędów.

Podsumowanie

Celem opisywanej pracy było rozwiązanie problemu z brakiem narzędzia pozwalającego na szybką, ale też na zaawansowaną edycję stron internetowych. Aby osiągnąć zamierzony cel, zaprojektowano i stworzono rozszerzenie do przeglądarki Google Chrome, dające wcześniej wymienione możliwości.

Aplikacja testowana była na kilkudziesięciu portalach, gdzie osiągnęła bardzo dobre wyniki i funkcjonowała bez zarzutów.

Wykonany projekt podyktowany był rozwojem i ewolucją stron internetowych w ostatnich kilku latach i ma za zadanie naświetlić oraz wspomóc ten proces. Wykonane rozszerzenie nie wyczerpuje tematu edycji stron, a sama aplikacja ma predyspozycje do rozwoju w wielu aspektach, takich jak edycja dźwięków i filmów na stronach czy dodawanie zaawansowanych animacji.

Bibliografia

Crockford D. (2008). *JavaScript - mocne strony.* Gliwice: Wydawnictwo HELION.

Duckett J. (2014). *HTML i CSS. Zaprojektuj i zbuduj witrynę WWW.* Gliwice: Wydawnictwo HELION.

Haverbeke M. (2011). *Zrozumieć JavaScript. Wprowadzenie do programowania.* Gliwice: Wydawnictwo HELION.

Kessin Z. (2011). *Html5. Programowanie Aplikacji.* Gliwice: Wydawnictwo HELION.

McFarland D. (2006). CSS. *Nieoficjalny podręcznik.* Gliwice: Wydawnictwo HELION.

Meyer E. (2000). *CSS: kaskadowe arkusze stylów: przewodnik encyklopedyczny.* Gliwice: Wydawnictwo HELION.

Simpson K. (2014). *Tajniki języka JavaScript. Wskaźnik this i prototypy obiektów.* Gliwice: Wydawnictwo HELION.

Simpson K. (2015). *Tajniki języka JavaScript. Typy i składnia.* Gliwice: Wydawnictwo HELION.

Simpson K. (2015). *Tajniki języka JavaScript. Asynchroniczność i wydajność.* Gliwice: Wydawnictwo HELION.

Stefanov S. (2012). *JavaScript. Wzorce.* Gliwice: Wydawnictwo HELION.

Dokumentacja Chrome API. Odczytano 20 lutego 2019, z *https://developer.chrome.com/apps/api_index.*

Dokumentacja HTML5. Odczytano 20 lutego 2019, z *https://dev.w3.org/html5/html-author/.*

Dokumentacja JavaScript. Odczytano 20 lutego 2019, z *https://developer.mozilla.org/pl/docs/Web/JavaScript.*

Dokumentacja jQuery. Odczytano 20 lutego 2019, z *https://api.jquery.com/.*

Spis rysunków