Dokumentacja techniczna dla programistów Wieloformatowych Obiektów Multimedialnych i

Interaktywnych (WOMI)

na platformie epodreczniki.pl

Spis treści

[Wprowadzenie 3](#_Toc526410806)

[Założenia: 3](#_Toc526410807)

[Struktura zasobu: 3](#_Toc526410808)

[Manifest 3](#_Toc526410809)

[2. metadata.json - plik zawierający metadane opisujące WOMI 4](#_Toc526410810)

[3. Plik wejścia 4](#_Toc526410811)

[Interfejs programistyczny (API) dla WOMI 4](#_Toc526410812)

[Dostępne biblioteki programistyczne 6](#_Toc526410813)

[Wspólne pliki 7](#_Toc526410814)

[Użycie Reader API w WOMI typu "Baw się i ucz" / "Pomyśl i działaj" 7](#_Toc526410815)

[Ustawienia WOMI 7](#_Toc526410816)

[DODATEK 8](#_Toc526410817)

[Lista silników WOMI wspieranych wstecznie: 8](#_Toc526410818)

# Wprowadzenie

Głównym celem dokumentacji technicznej dla programistów Wieloformatowych Obiektów Multimedialnych i Interaktywnych (WOMI) jest opis interfejsów programistycznych dla oprogramowania dostarczanego i uruchamianego przez zewnętrznych programistów na platformie epodreczniki.pl.

Komponenty typu WOMI są to elementy e-podręczników składające się z plików pozwalających na zaawansowaną prezentacje tekstową i graficzną w przeglądarce internetowej. Obecnie na platformie wspierane są tylko technologie zgodne z ustalonym i potwierdzonym przez W3C otwartym standardem HTML5, w szczególności sam język HTML5, CSS2, CSS3 i JavaScript.

WOMI nie pozwalają osadzać elementów oprogramowania napisanych w innych niż wyżej wymienionych technologiach, w szczególności: Adobe Flash, MS Silverligh, które wymagają dodatkowych komponentów instalowanych w systemie.

# Założenia:

Struktura ma zapewniać prostotę implementacji, spójność pomiędzy platformami, niezależność oraz izolację od kodów portalu.

Zdefiniowany skrypt zostanie wywołany w izolowanym kontekście (iframe), a komunikacja pomiędzy ramkami realizowana jest przez mechanikę PostMessage.

# Struktura zasobu:

## Manifest

Plik manifest json, definiujący wykorzystany silnik, wersję silnika, plik wejścia oraz dodatkowe opcje. Zalecane jest używanie ostatniej wersja silnika epodreczniki.

{

 "engine": "epodreczniki",

 "mainFile": "womi.js",

 "version": "1.0",

 "options": {},

 "womiIds": [

 "id\_womi\_1", "id\_womi\_2"

 ]

}

Dostawcy mają możliwość utworzenia uniwersalnego zasobu.
W takim przypadku wywoływany jest pliku wejścia z katalogu dostawcy zamiast tego w zasobie.

{
 "engine": "epodreczniki",
 "provider": "dostawca",
 "providerEngine": "womi.js",
 "version": "1.0",
 "options": {}
}

## 2. metadata.json - plik zawierający metadane opisujące WOMI

## 3. Plik wejścia

**Przykład 1. minimalny kod zasobu interaktywnego**

Epodreczniki.v1(function (root, api, options) {

});

**Przykład 2. modyfikacja DOM**

Epodreczniki.v1(function (root, api, options) {

 // Załaduj bibliotekę jquery

 api.loadLibrary(['jquery'], function ($) {

 // Dodaj nagłówek

 $(root).append('<h1>Example</h1>');

 });

});

**Przykład 3. wykorzystanie API**

Epodreczniki.v1(function (root, api, options) {

 // Pobierz wartość zmiennej "zmienna1"

 api.getVar('zmienna1')

 .then(function (value) {

 // Powiększ wartość zmiennej i zapisz

 api.setVar('zmienna1', value + 1);

 });

});

# Interfejs programistyczny (API) dla WOMI

W ogólności interfejs programistyczny (API) ma na celu dostarczyć funkcjonalność pozwalającą na pełną integrację WOMI z platformą epodreczniki.pl. API zostało zaprojektowane ze szczególnym uwzględnieniem dla kontekstu w jakim zostało uruchomione WOMI.

API definiuje następujące metody:

* pobranie aktualnego kontekstu (URL do zasobu, czy użytkownik zalogowany, w jakiej lekcji został uruchomiony zasób interaktywny)

context(): Object;
* załadowanie plików CSS

loadCSS(path: string): Promise;
* pobieranie wartości zapisanej zmiennej

getVar(name: string): Promise;
* zapisanie zmiennej

setVar(name: string, value: Object): Promise;
* pobranie adresu URL dla strumienia audio

getAudioUrl(id: string): Promise;
* pobranie adresu URL dla strumienia video

getVideoUrl(id: string): Promise;
* pobranie aktualnego kontekstu (URL do zasobu, czy użytkownik

upload(id?: string, name: string, fileData: string): Promise;
* pobranie aktualnego kontekstu (URL do zasobu, czy użytkownik

getUpload(id?: string): Promise;
* rejestracja funkcji która ma zostać wywołana przy otrzymaniu wiadomości

onMessage(callback: Function);
* rejestracja funkcji która ma zostać wywołana przy ponownym połączeniu do serwera

onConnect(callback: Function);
* rejestracja funkcji która ma zostać wywołana przy utracie połączenia z serwerem

onDisconnect(callback: Function);

# Dostępne biblioteki programistyczne

Do wykorzystania w "imporcie" w module:

* *'jquery'* - biblioteka *jquery*
* *'jqueryui'* - dodatek do biblioteki *jquery* - mechanizmy interfejsu użytkownika
* *'declare'* - biblioteka *declarejs*
* *'underscore'* - biblioteka *underscore.js*
* *'backbone'* - biblioteka *backbone.js*

## Wspólne pliki

WOMI może składać się z innych modułów, które są importowane w głównym module. Jednak dodatkowe moduły często mogą być ponownie użyte w nowych WOMI, dlatego zalecane by zaimplementować je w formie "biblioteki". W pierwszej kolejności zaleca się implementacje wspólnych modułów w jednym WOMI, a następnie bezpośredni kontakt z administratorem platformy w celu sprawdzenia, zatwierdzenia i wgrania wytworzonych bibliotek na serwer statyczny.

## Użycie Reader API w WOMI typu "Baw się i ucz" / "Pomyśl i działaj"

W WOMI tego typu można używać analogicznie API dla WOMI w ramce, czyli ReaderApi i EmbedApi.

# Ustawienia WOMI

WOMI powinno zostać opisane dwoma plikami:

* *manifest.json* - plik zawierający definicję silnika dla WOMI
* *metadata.json* - plik zawierający metadane opisujące WOMI

Pole engine służy do podania silnika przetwarzania dla danego WOMI. Pole przyjmuje wartość „epodręczniki” lub inny silnik z dodatku „Lista silników WOMI wspieranych wstecznie”.

W przypadku zagnieżdżania WOMI w WOMI listę takich obiektów należy zdefiniować poprzez *womiIds.*

Każde WOMI musi mieć podaną proporcję**.** *Height ratio* w tym wypadku to stosunek wysokości do szerokości dla danego WOMI, pozwala to skalować WOMI z zachowaniem jego proporcji.

# DODATEK

## Lista silników WOMI wspieranych wstecznie:

* *edge\_animation*: dla animacji ze środowiska *Adobe Edge*
* *createjs\_animation*: animacje *CreateJS*
* *ge\_animation*: animacje Grupy Edukacyjnej
* *custom\_womi*: dla WOMI jako moduł *requirejs*
* *framed\_html* - dla WOMI, które ma być osadzone w ramce jako HTML
* *custom\_logic\_exercise\_womi*: podobne do *custom\_womi*, pozwala tworzyć WOMI, które nie mają rozmiaru, mogą też ładować same z siebie inne WOMI i tworzyć fragmenty treści
* *ace\_editor*: silnik dla edytora *Ace*
* *svg\_editor*: silnik dla edytora *SVG Edit*
* *geogebra*: WOMI typu *geogebra*
* *swiffy*: WOMI typu *swiffy*