

# Technologie multimedialne

## Lista 3

Wejdź na stronę <https://pixabay.com/music/> i pobierz plik „vlog-beat-background-349853.mp3”.



Vlog Beat Background  
Tunetank

Celem tej listy jest praca ze ścieżkami dźwiękowymi pliku audio. Stosowane są efekty dźwiękowe (zmiana szybkości odtwarzania, zmiana głośności) oraz filtry. Realizacja odbywa się w Web Audio API. W ramach dyskusji nad oceną z listy należy umieć dodawać, usuwać oraz modyfikować węzły w grafie przepływu Web Audio API.

Lista na ocenę.

opracował: dr inż. Jakub Długosz

Wiodącym narzędziem tej listy jest [Web Audio API](#). Umożliwia ono:

1. generowanie dźwięków
2. wybór źródeł dźwięku/ów
3. mieszanie źródeł dźwięku
4. przetwarzanie dźwięku, w tym zastosowanie różnych filtrów
5. tworzenie wizualizacji.

Wszystko powyższe może odbywać się w czasie rzeczywistym.

**Filtr służy do wzmacniania, bądź osłabiania, amplitudy (głośności) dla wybranych zakresów częstotliwości w widmie.**

Przykłady filtrów:

1.  $WY[n] = WE[n-1]$  (filtr opóźniający)
2.  $WY[n] = 1/2 WE[n] + 1/2 WY[n-1]$  (filtr uśredniający),



Jedna z książek opublikowanych przez [Związek Inżynierów Audio](#) (ang. Audio Engineering Society). Pełna ich lista znajduje się na [tej stronie](#).

Odnosniki do materiałów prof. Reissa:

1. <https://www.youtube.com/watch?v=idhb45lc2xo>
2. <https://github.com/joshreiss/Working-with-the-Web-Audio-API>.

gdzie  $n$  reprezentuje numer próbki, WE - wejście (źródło), a WY - wyjście (np. głośniki lub słuchawki).

Przykładem inżynierii dźwięku wykorzystującej generowanie impulsu, filtr dolnoprzepustowy (inaczej niskoprzepustowy – jego działanie polega na tłumieniu wyższych częstotliwości) z modulacją amplitudową oraz pętlę opóźnienia jest [algorytm Karplusa–Stronga](#). Algorytm ten, wykorzystując cyfrowe przetwarzanie sygnałów, pozwala wygenerować realistyczne brzmienia instrumentów strunowych. Ciekawa implementacja tego algorytmu w języku Python wraz z opisem (praca Wojciecha Węgrzyna, WMil, UJ) znajduje się [na tej stronie](#).

**W ramach zadań na tej liście odtwarzanym plikiem jest ten podany w czerwonej ramce, jest to taki sam plik, jaki był podany na poprzedniej liście. Ten plik ma być odtwarzany w pętli (cyklicznie).**

Ta lista szczególnie nacisk kładzie na filtry z efektami audio, które udostępnia Web Audio API.

Są nimi (zgodnie z nazwą węzła w Web Audio API):

1. [BiquadFilterNode](#)
2. [ConvolverNode](#)
3. [DelayNode](#)
4. [DynamicsCompressorNode](#)
5. [GainNode](#)
6. [WaveSharperNode](#)
7. [PeriodicWave](#)
8. [IIRFilterNode](#).

## Z1

Używając Web Audio API zastosuj przy osadzonym na stronie WWW odtwarzaczu audio suwaki sterujące następującymi efektami dźwiękowymi:

1. przyspieszenie/spowolnienie odtwarzania w czasie rzeczywistym w zakresie 50-500%.
2. zwiększenie/zmniejszenie wysokości dźwięku w zakresie 50-150%.

Przykład implementacji suwaka znajduje się [tutaj](#).

## Z2

Umieść przy osadzonym na stronie WWW odtwarzaczu audio podwójny suwak lub dwa pojedyncze suwaki, które umożliwią określenie wartości  $a$  i  $b$  tworzących przedział częstotliwości  $[a, b]$  zawarty (będący podprzedziałem) w przedziale  $[10\text{Hz}, 20\text{kHz}]$ . Częstotliwości z zakresu  $[a, b]$  mają być usuwane z widma tzn. nie następuje ich odtwarzanie. Wartościami początkowymi dla  $a$  i  $b$  niech będzie  $10\text{Hz}$ .

Użyj węzła/-ów [BiquadFilterNode](#) starając się dobrać odpowiedni typ węzła oraz ew. jego parametry (`frequency`, `Q`, `gain`). Oto typy dla węzła [BiquadFilterNode](#):

1. lowpass (filtr dolnoprzepustowy)
2. highpass (filtr górnoprzepustowy)
3. bandpass (filtr pasmowy)
4. lowshelf (filtr dolnopółkowy)

5. highshelf (filtr górnopółkowy)
6. peaking (filtr szczytowy)
7. notch (filtr wycinający, odwrotność filtra pasmowego)
8. allpass (filtr wszechprzepustowy).

Dodatkowe odnośniki:

1. <https://www.youtube.com/watch?v=idhb45lc2xo>
2. [https://developer.mozilla.org/en-US/docs/Web/API/Web\\_Audio\\_API/Visualizations\\_with\\_Web\\_Audio\\_API](https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API/Visualizations_with_Web_Audio_API)
3. [https://developer.mozilla.org/en-US/docs/Web/API/Web\\_Audio\\_API/Best\\_practices#:~:text=Loading%20sounds%2Ffiles](https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API/Best_practices#:~:text=Loading%20sounds%2Ffiles)
4. <https://musiclab.chromeexperiments.com/>
5. [https://www.youtube.com/watch?v=WGN\\_BDYjncY](https://www.youtube.com/watch?v=WGN_BDYjncY)