

**İSKENDERUN TEKNİK ÜNİVERSİTESİ**

**MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**BİTİRME PROJESİ 2**

**SES FREKANS ANALİZİ**

**Danışman:** Dr. Öğr. Üyesi Gökhan Altan

**Öğrenci Ad:** Nuray Çelik (151123007)

## 

## Haziran 2020 / İskenderun-HATAY

1. [GİRİŞ](#_TOC_250028) 3
2. [KISALTMALAR VE TANIMLAR 3](#_TOC_250027)
3. [SES NEDİR? 4](#_TOC_250026)
4. [FREKANS NEDİR? 4](#_TOC_250025)
   1. [Açısal Frekans 5](#_TOC_250024)
   2. [Dalga Boyu 6](#_TOC_250023)
   3. [Müzikte Ses 8](#_TOC_250022)

5. SES SENTEZİ………………………………………………………....10

* 1. Fourier Dönüşümü Dijitalleşiyor..…………………………………………….11

1. WEB AUDIO API…………………………………………………….17
2. KULLANILAN TEKNOLOJİLER…………………………………...19
3. TEKNOLOJİLERİ TERCİH ETME SEBEBİ………………………...19
4. AVANTAJLARI/DEZAVANTAJLARI……………………………...20
5. PROJENİN YENİLİKÇİ YÖNÜ……………………………...............21

Projenin Web Görünümü Üzerinden Anlatımı………………………… …..22

10. PROJE KAYNAK KODLARI………………………………………23

11. PROJE GÖRSELLERİ………………………………………………47

12. Yararlanılan Kaynaklar……………………………………………....49

1. GİRİŞ

“Ses Frekans Analizi” Bitirme Projesi olarak çalışılmıştır. Projeye öncelikle konu araştırılması, kaynak taramaları ile başlanmıştır. Daha sonra web üzerinden gösterilmek amacıyla Html, CSS ve PHP ile wab sayfa tasarımı hazırlanmıştır ve uygulama tamamlamıştır. Bu proje raporunda öncelikle Ses ve frekans ile ilgili bilgilere yer verildi. Ardından proje kodları ve görselleri paylaşıldı.

1. KISALTMALAR VE TANIMLAR
   1. *frekans:* Birim zamanda oluşan ses dalga sayısıdır. Birim zaman genellikle 1 saniye olarak alınır. Birimi **Hertz (Hz)’**dir.
   2. *genlik:* Genlik, bir ses dalganın tepesinden çukuruna kadar olan düşey uzaklığın yarısıdır.
   3. *Analizer:* Frekans değerlerini görselleştirme
   4. *Fourier analizi:* Tabiattaki bütün [periyodik fonksiyonları](https://tr.wikipedia.org/wiki/Periyodik_fonksiyon) birbirine dik iki farklı periodik fonksiyonun artan frekanslardaki değerlerinin dik toplamı şeklinde gösterilebilir. [Fourier](https://tr.wikipedia.org/wiki/Fourier), bu toplamı [sinüs](https://tr.wikipedia.org/wiki/Sin%C3%BCs_fonksiyonu) ve [kosinüs fonksiyonlarını](https://tr.wikipedia.org/wiki/Kosin%C3%BCs_fonksiyonu) kullanarak göstermiştir.
   5. *FFT:* Hızlı Fourier dönüşümü(Fast Fourier Transform-FFT) titreşim analizinde kullanılan istatistik tabanlı, matematiksel işlemdir. Karışık sinyalleri ayrıştırır ve hangi frekansta ne şiddette bir titreşim olduğunu gösterir.FFT tekrarlanmayan sinyalleri dikkate almaz. Karmaşık sinyaller içinde periyodik olanları belirleyip harmonic bileşenlerine ayırır.
   6. *wavelet:* Sıfırdan başlayan, artan ve sonra sıfıra azalan bir [genliğe](https://en.wikipedia.org/wiki/Amplitude) sahip [dalga](https://en.wikipedia.org/wiki/Wave) benzeri bir [salınımdır](https://en.wikipedia.org/wiki/Oscillation).
   7. *wavelet transform:* Wavelet transform, tamamen farklı bir değer fonksiyonu ile Fourier dönüşümüne benzer.
   8. *Peryot:* Bir dalganın ne kadar sürede tamamlandığını belirtir. Birimi **saniye**‘dir.

# SES NEDİR?

Fizikte ses, gaz, sıvı veya katı gibi bir iletim ortamından akustik bir dalga olarak yayılan bir titreşimdir. İnsan fizyolojisi ve psikolojisinde ses, bu dalgaların alınması ve beyin tarafından algılanmasıdır. Sadece frekansları yaklaşık 20 Hz ile 20 kHz arasında olan akustik dalgalar, ses frekans aralığı, insanlarda işitsel bir algı ortaya çıkarır. Atmosferik basınçtaki havada, bunlar 17 metre (56 ft) ila 1.7 santimetre (0.67 inç) dalga boylarına sahip ses dalgalarını temsil eder. 20 kHz üzerindeki ses dalgaları ultrason olarak bilinir ve insanlar tarafından duyulamaz. 20 Hz altındaki ses dalgaları infrasound olarak bilinir. Farklı hayvan türlerinin değişen işitme aralıkları vardır.

Ses, hava, su ve katılar gibi bir araç boyunca uzunlamasına dalgalar ve ayrıca katılardaki enine dalga olarak yayılabilir. Ses dalgaları, bir stereo hoparlörün titreşen diyaframı gibi bir ses kaynağı tarafından üretilir. Ses kaynağı çevredeki ortamda titreşimler yaratır. Kaynak ortamı titreştirmeye devam ettikçe, titreşimler ses hızından kaynaktan uzaklaşarak ses dalgasını oluşturur. Kaynaktan sabit bir mesafede, ortamın basıncı, hızı ve yer değiştirmesi zaman içinde değişir. Anında, basınç, hız ve yer değiştirme uzayda değişir. Ortamdaki parçacıkların ses dalgasıyla hareket etmediğine dikkat edin. Bu bir katı için sezgisel olarak açıktır ve aynı sıvılar ve gazlar için de geçerlidir (yani, gaz veya sıvıdaki parçacıkların titreşimleri titreşimleri taşır, oysa parçacıkların zaman içindeki ortalama konumu değişmez). Yayılma sırasında, dalgalar ortam tarafından yansıtılabilir, kırılabilir veya zayıflatılabilir.

1. FREKANS NEDİR ?

Frekans, birim zaman başına yinelenen bir olayın gerçekleşme sayısıdır. Ayrıca mekansal frekans ve açısal frekansın kontrastını vurgulayan zamansal frekans olarak da adlandırılır. Frekans, saniyede yinelenen bir olayın bir oluşumuna eşit olan hertz (Hz) birimleri cinsinden ölçülür. Periyot, tekrar eden bir olayda bir çevrimin süresidir, bu nedenle periyot, frekansın karşılıklılığıdır. Örneğin: yeni doğmuş bir bebeğin kalbi dakikada 120 kez (2 hertz) sıklıkta atıyorsa, süresi, T, - atımlar arasındaki zaman aralığı - yarım saniyedir (60 saniyenin 120 atım ile bölünmesi). Frekans, mekanik titreşimler, ses sinyalleri (ses), radyo dalgaları ve ışık gibi salınım ve titreşim olaylarının oranını belirtmek için bilim ve mühendislikte kullanılan önemli bir parametredir.

Dönme, salınımlar veya dalgalar gibi döngüsel işlemler için frekans, birim zaman başına bir dizi döngü olarak tanımlanır.

Tekrarlanan bir olayın veya salınımın frekansı f ve periyodu T olmak üzere formül:

f = 1/T ‘dir.

Birimler

SI kaynaklı frekans birimi, Alman fizikçi Heinrich Hertz'den sonra adlandırılan hertz (Hz) 'dir. Bir hertz, bir olayın saniyede bir tekrarlandığı anlamına gelir.

Kolaylık olması açısından, okyanus yüzey dalgaları gibi daha uzun ve yavaş dalgaların frekanstan ziyade dalga periyodu ile açıklanma eğilimi vardır. Ses ve radyo gibi kısa ve hızlı dalgalar genellikle peryot yerine frekanslarıyla tanımlanır. Bu sık kullanılan dönüşümler aşağıda listelenmiştir:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Frequency** | 1 mHz (10−3 Hz) | 1 Hz (100 Hz) | 1 kHz (103 Hz) | 1 MHz (106 Hz) | 1 GHz (109 Hz) | 1 THz (1012 Hz) |
| **Period** | 1 ks (103 s) | 1 s (100 s) | 1 ms (10−3 s) | 1 µs (10−6 s) | 1 ns (10−9 s) | 1 ps (10−12 s) |

**Açısal Frekans**

Genellikle Yunanca ω (omega) harfi ile gösterilen açısal frekans, açısal yer değiştirmenin, θ, (dönüş sırasında) değişim hızı veya sinüzoidal dalga formunun (özellikle salınımlar ve dalgalarda) değişim hızı olarak tanımlanır. ) veya argümanın sinüs işlevine değişim hızı olarak:

y(t) = sin(**θ(t)) = sin(ωt)=sin(2πft)**

**d θ /dt = ω = 2πf**

**Açısal frekans genellikle saniyede radyan (rad / s) cinsinden ölçülür, ancak ayrık zamanlı sinyaller için örnekleme aralığı başına radyan olarak da ifade edilebilir, bu boyutsuz bir miktardır. Açısal frekans (radyan cinsinden) normal frekanstan (Hz cinsinden) 2π kat daha büyüktür.**

**Uzamsal frekans, zamansal frekansa benzer, ancak zaman ekseni bir veya daha fazla uzamsal yer değiştirme ekseni ile değiştirilir. Örneğin.:**

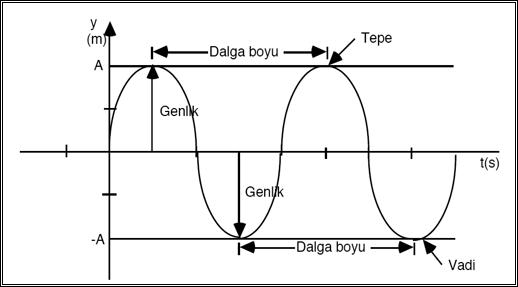
**y(t) = sin(θ (t, x)) = sin(ωt + kx)**

**dθ /dx = k**

**Dalga yayılımında**

**Dağıtıcı olmayan ortamdaki (yani, dalga hızının frekanstan bağımsız olduğu ortam) periyodik dalgalar için, frekansın dalga boyu, λ (lambda) ile ters bir ilişkisi vardır. Dağıtıcı ortamda bile, sinüzoidal bir dalganın f frekansı dalganın faz hızına v dalganın dalga boyu λ'ya eşittir:**

**f = v / λ**



**Dalga boyu**, bir dalga örüntüsünün tekrarlanan birimleri arasındaki mesafedir. Yaygın olarak Yunanca lamda (λ) harfi ile gösterilmektedir. Dalgaboyu frekans ile ters orantılıdır, dolayısıyla dalgaboyu uzadıkça frekans azalır

Elektromanyetik tayf en uzun dalga boyundan en kısa dalga boyuna şu grupları kapsar:

– Radyo dalgaları  
– Mikro dalgalar  
– Kızılötesi dalgar  
– Optik dalgalar  
– Ultraviole dalgalar  
– X ışını dalgaları  
– Gama dalgaları

**Örnek** Işık ve elektro-manyetik dalgalar için hızın saniyede 300.000.000 metre olduğu bilinmektedir. Buradan, dalga boyu **λ (metre) = 300.000.000 / f (Hz)** olarak hesaplanabilir.

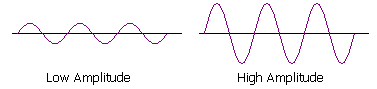
\* Frekansı 1.000 Hz olan bir dalganın dalga boyu 300.000 metredir.  
\* Frekansı 300 KHz (300.000 Hz) olan bir dalganın dalga boyu 1.000 metredir.  
\* Frekansı 10 MHz (10.000.000 Hz) olan bir dalganın dalga boyu 30 metredir.  
\* Frekansı 3 GHz (3.000.000.000 Hz) olan bir dalganın dalga boyu 0.1 metre, yani 10 santimetredir.

\* Dalga boyu 1.000 metre olan bir dalganın frekansı 300 KHz (300.000 Hz)’dir.  
\* Dalga boyu 1 metre olan bir dalganın frekansı 300 MHz (300.000.000 Hz)’dir.

**KHz**: kilohertz  
**MHz**: megahertz  
**GHz**: gigahertz

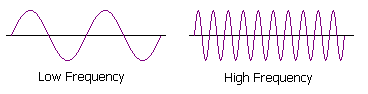
# Genlik ve Frekans

Düzenli bir titreşimin iki ana özelliği vardır, genlik ve frekans. Genlik ve frekans ses şeklini etkiler.

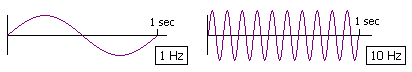


Genlik, titreşimin boyutudur ve bu, sesin ne kadar yüksek olduğunu belirler. Daha büyük titreşimlerin daha yüksek ses çıkardığını gördük.

Frekans, titreşimin hızıdır ve bu, sesin perdesini belirler.



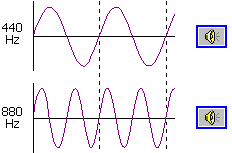
1 Hz frekans, saniyede bir dalga döngüsü anlamına gelir. 10 Hz'lik bir frekans saniyede on dalga döngüsü anlamına gelir, burada döngüler birbirine daha kısa ve yakındır.



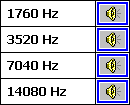
Bir notanın frekansını iki katına çıkardığınızda çok ilginç bir şey olur. Double frekansın perdesi daha yüksek, ancak bir şekilde orijinal nota ile aynıdır, ancak aradaki tüm frekansların perdeleri oldukça farklıdır.

Örnek olarak 440 Hz frekans aralığını kullanalım. Daha önce de belirtildiği gibi A notasıdır. 880 Hz frekans aralığı daha yüksektir, ancak aynı nota benzer.

Bu benzerliğin mantıklı bir nedeni var. Aşağıdaki ses dalgaları bize 880 Hz frekansının iki çevriminin 440 Hz frekansının tek bir çevriminin alanına tam olarak oturduğunu göstermektedir.



Bu frekansı iki katına çıkarmaya devam edersek, ortaya çıkan tüm adımların birbirine benzediğini görüyoruz, ancak her biri bir öncekinden daha yüksek. Aslında, hepsi orijinali gibi A notası, ancak hepsi birbirinden ayrı bir **oktav**.



# Müzikte Ses

Ses dalgaları boyuna dalgalardır. Uzunlamasına bir dalgada, ortamın yer değiştirmesi dalganın yayıldığı yöne paraleldir. Havadaki ses dalgaları için, hava molekülleri ileri geri salınıyor ve enerjilerini hareketleriyle aynı yönde yayıyorlar.

Ses dalgaları genlik ve frekansları açısından analiz edilebilir . Bir sesin ses yüksekliği dalganın genliğine karşılık gelir ve desibel cinsinden ölçülür .  Çoğu müzikal ses, kısmi tonlar veya sadece kısmi olarak adlandırılan birçok frekansın üst üste binmesinden oluşur . Belirli bir ses için en düşük frekansa temel frekans denir .  Bu aynı zamanda doğal rezonans frekansıdır.

Temel frekansın tam katları olan kısmi tonlara harmonik denir . Temel frekansın iki katı frekansa sahip bir tona, temelden bir oktav daha yüksek olan ikinci harmonik denir . Örneğin, eşit huylu kromatik skaladaki orta C'nin 262 Hz'lik bir eğimi vardır ve aynı not bir oktavın üstünde 524 Hz'lik bir frekansa sahiptir. Birden fazla frekansın birleşimi, bir sese karakteristik kalitesini veren şeydir .

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Note** | **Letter name** | **Frequency (Hz)** | **Frequency ratio** | **Interval** |
| do | C | 264 | |  | | --- | | 9 | | 8 | | Whole |
| re | D | 297 | |  | | --- | | 10 | | 9 | | Whole |
| mi | E | 330 | |  | | --- | | 16 | | 15 | | Half |
| fa | F | 352 | |  | | --- | | 9 | | 8 | | Whole |
| sol | G | 396 | |  | | --- | | 10 | | 9 | | Whole |
| la | A | 440 | |  | | --- | | 9 | | 8 | | Whole |
| ti | B | 495 | |  | | --- | | 16 | | 15 | | Half |
| do | C*'* | 528 |  |  |
| **Table 1: Diatonic C Major scale** | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Note** | **Frequency (Hz)** | **Frequency ratio** | **Interval** |
| C | 262 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| C♯ or D♭ | 277 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| D | 294 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| D♯ or E♭ | 311 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| E | 330 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| F | 349 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| F♯ or G♭ | 370 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| G | 392 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| G♯ or A♭ | 415 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| A | 440 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| A♯ or B♭ | 466 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| B | 494 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | | Half |
| C*'* | 524 | |  |  |  | | --- | --- | --- | | 12 | http://www.webassign.net/wastatic/watex/img/sqrt1a.gif | 2 | |  |
| **Table 2: Equal-tempered chromatic scale** | | | |

1. SES SENTEZİ

Ses analizi karmaşık bir ses dalgasındaki frekans bileşenlerinin belirlenmesidir. Joseph Fourier, 1800'lerde frekans analizinin temelini oluşturan matematiği geliştirdi. Herhangi bir periyodik sinüzoidal fonksiyonun, karmaşıklığına bakılmaksızın, frekans bileşenlerinin bir toplamı olarak formüle edilebileceğini kanıtladı. Daha sonra, bir integral kullanılarak Fourier serisini oluşturan bireysel sinüs ve kosinüs dalgalarının genliğinin belirlenmesinin mümkün olduğu keşfedildi. Bu **Fourier Dönüşümü** olarak bilinir.

Frekans bileşenleri temel bir frekanstan ve bu temel ile ilgili harmonik frekanslardan oluşur. Fourier'in teoremi, bir ses ne kadar karmaşık olursa olsun, sesi bileşen frekanslarına bölmenin mümkün olduğunu söylüyor. Yani, o sesin farklı frekanslarını ve her frekans bileşeninin ne kadarının olduğunu belirlemek.

Fourier analizi, sesin temel frekansı ile başlar - sesin en uzun tekrarlanan paterninin frekansı. Daha sonra Fourier analizi ile elde edilebilen kalan tüm frekans bileşenleri - yani harmonik frekanslar - temel frekansın tam sayı katlarıdır. “Tam sayı çoklu” ile temel frekansınf0f0 , sonra her harmonik frekansı fnfn negatif olmayan bir tam sayı için eşittir ( n + 1 )f0(n+1)f0.

Fourier dönüşümü, bir sesin frekans bileşenlerini belirlemek için dijital filtrelerde ve frekans analiz yazılımında kullanılan matematiksel bir işlemdir.



*Şekil 1: Jean-Baptiste Joseph Fourier (1768-1830)*

Fourier bulgularını 1822'de Analitik Isı Teorisi kapsamında yayınladı .

**Fourier Dönüşümü Dijitalleşiyor**

Günümüzde bilgisayar tabanlı uygulamalarda iki tür Fourier Dönüşümü kullanılmaktadır: Ayrık Fourier Dönüşümü (DFT) ve Hızlı Fourier Dönüşümü (FFT).

1. Ayrık Fourier Dönüşümü (DFT) ayrık bir zaman bloğu üzerinde bir Fourier Dönüşümü gerçekleştirir. Bir DFT, rastgele sayıda veri noktasından oluşan herhangi bir zaman sinyalinde gerçekleştirilebilir.

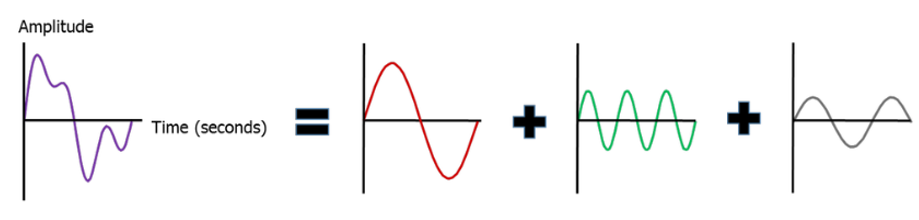
2. Hızlı Fourier Dönüşümü (FFT), bir Sayısal Fourier Dönüşümünü hesaplama açısından verimli bir şekilde gerçekleştiren bir algoritmadır. Analiz edilen zaman bloğunda (örneğin 512, 1024, 2048 ve 4096) iki numune örneği gerektirir.

**Fourier Dönüşümünün Temelleri**

Fourier, herhangi bir sinyalin, farklı genlik ve fazdaki bir dizi sinüs dalgası olarak temsil edilebileceğini gösterdi. Örneğin, mordaki aşağıdaki sinyal ( Şekil 2 ) sadece üç sinüs dalgasının toplamıdır.

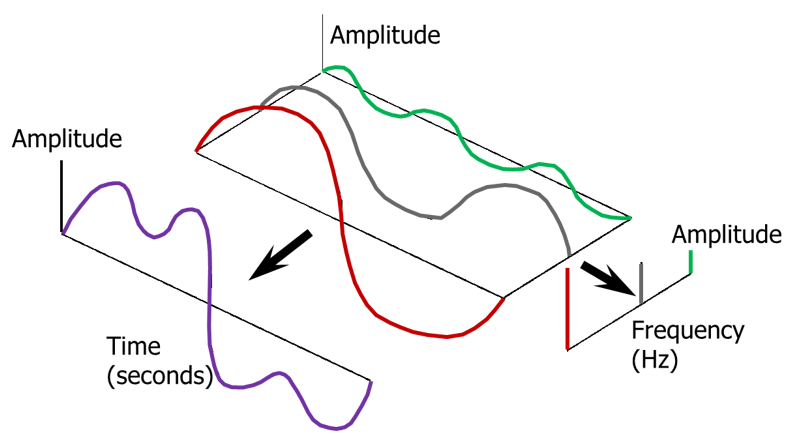
**Fourier Dönüşümünün Temelleri**

Fourier, herhangi bir sinyalin, farklı genlik ve fazdaki bir dizi sinüs dalgası olarak temsil edilebileceğini gösterdi. Örneğin, mordaki aşağıdaki sinyal ( Şekil 2 ) sadece üç sinüs dalgasının toplamıdır.



*Şekil 2: Karmaşık bir sinyal (mor, sol) bireysel sinüs dalgalarının (kırmızı, yeşil, siyah) toplamıdır.*

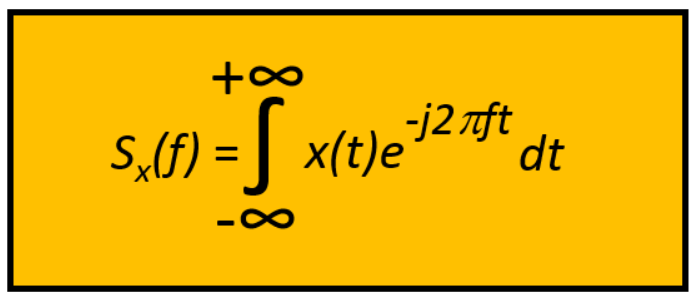
Fourier Dönüşümü, orijinal zaman sinyalinin sinüzoidlere ayrılmasını mümkün kılar. Her sinüzoidin ilişkili bir genliği, fazı ve frekansı vardır. Bu, aşağıdaki Şekil 3'te gösterilmektedir.



*Şekil 3: Fourier Dönüşümü, frekans alanından (sağ taraf) karmaşık bir zaman sinyalinin (sol, sağ) görüntülenmesini sağlar.*

**Fourier Dönüşüm Denklemi**

Fourier Dönüşümü için denklem Denklem 1'de verilmiştir :

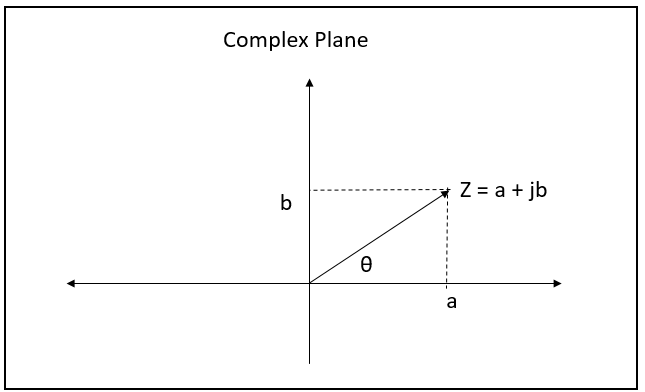
  
 *Denklem 1: Fourier Dönüşümü*

* S\_x ( f), frekans alanında Fourier Dönüşümünün çıktısıdır
* x (t), giriş zamanı etki alanı işlevidir
* 2 π f saniye başına radyan cinsinden frekanstır

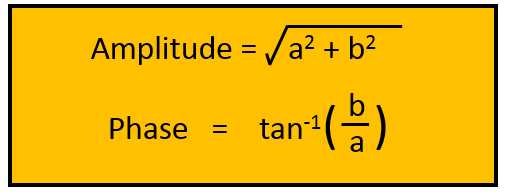
**Karmaşık sayılar**

Bir Fourier Dönüşümünün çıktısı, her biri elde edilen frekans spektrumundaki bir frekansa, genliğe ve faza karşılık gelen bir dizi karmaşık sayıdır. Karmaşık sayılar **a** + **jb** formundadır (genellikle Z olarak adlandırılır). Harfli **bir** süre, karmaşık sayının gerçek kısmına değinmektedir **b** hayali kısmıdır.

Karmaşık sayılar, Şekil 4'te gösterildiği gibi orijinal zaman sinyalindeki frekans bileşenlerinin genliği ve fazı hakkında bilgi içerir . Karmaşık sayının genliği hipotenüstür. Faz θ açısıdır .

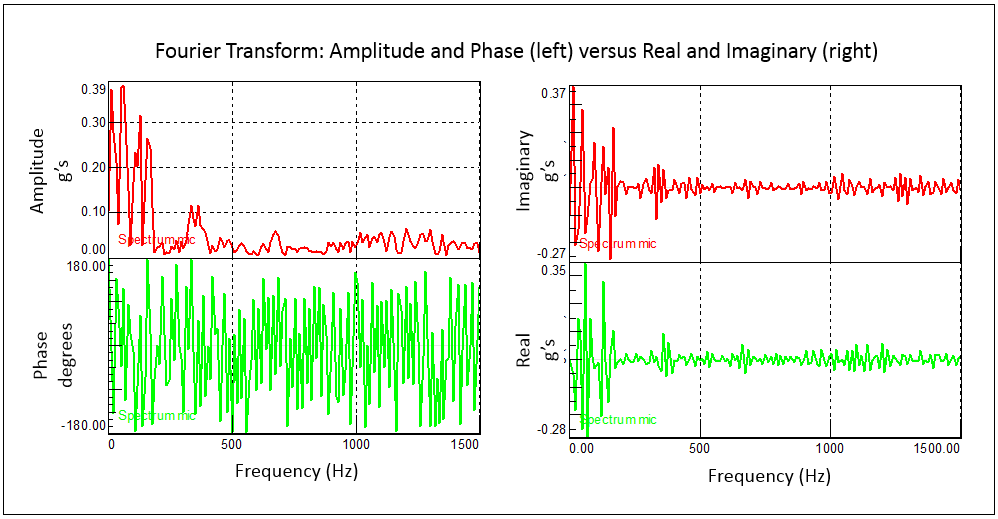
  
*Şekil 4: Karmaşık Düzlem*

Denklem 2 ve 3 , gerçek ve hayali karmaşık değerler ile genlik ve faz arasındaki ilişkiyi gösterir .



*Denklem 2 ve 3: Kompleks sayı (a + jb) ile genlik ve faz arasındaki ilişki.*

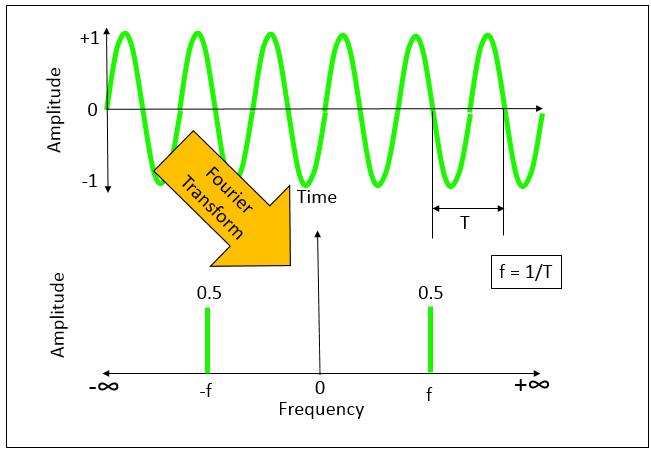
Fourier Dönüşümünün sonuçları genlik ve faz ( sol grafik, Şekil 5 ) VEYA gerçek ve hayali ( sağ grafik, Şekil 5 ) olarak gösterilebilir.

  
*Şekil 5: Bir zaman sinyalinin Fourier Dönüşümü, Genlik ve Faz (sol grafik) veya Gerçek ve Hayali (sağ grafik) olarak görüntülenebilen karmaşık bir spektrum ile sonuçlanır.*

Faz korunduğundan, orijinal zaman geçmişini geri yüklemek için spektral veriler üzerinde ters bir Fourier Dönüşümü gerçekleştirilebilir.

Çift Taraflı Spektrum

Fourier Dönüşümü, Şekil 6'da gösterildiği gibi çift taraflı bir spektrum üretir .

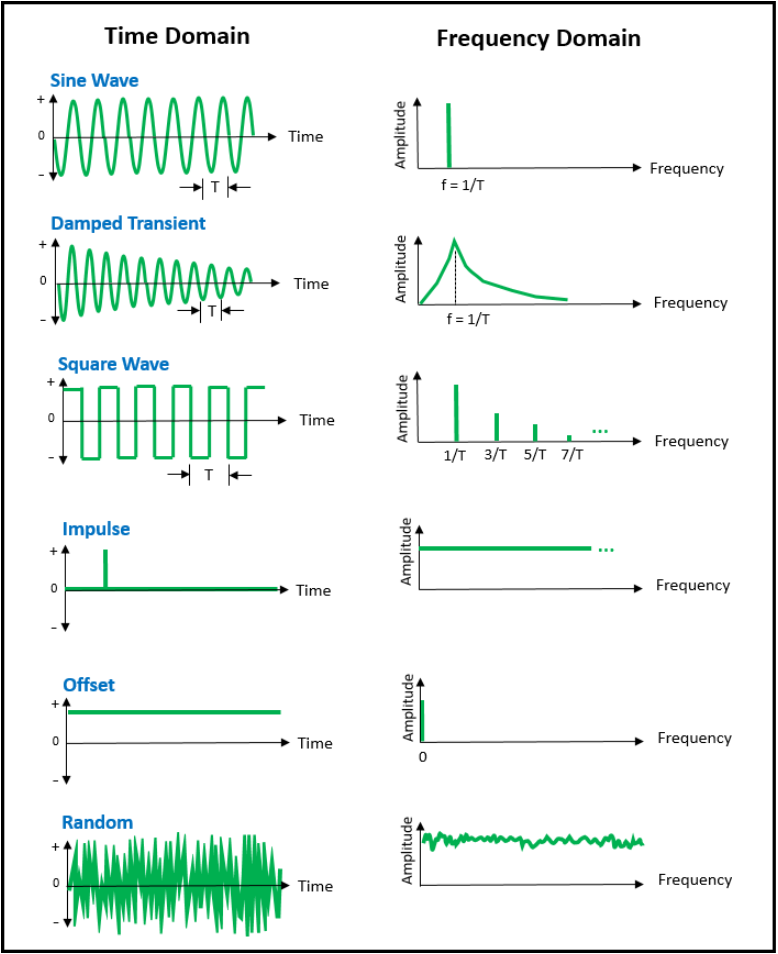
  
*Şekil 6: Hem FFT hem de DFT algoritması çift taraflı bir spektrum üretir.*

İki taraflı bir spektrum negatif ve pozitif frekanslardan oluşur. Fourier Dönüşümünün sınırları zaman içinde negatiften pozitif sonsuza gider, sonuçta elde edilen spektrumun frekans aralığı da değişir. Çift taraflı spektrumun genliği, Şekil 6'da gösterildiği gibi zaman alanındaki tepe genliğinin yarısıdır .

Geleneksel olarak, dijital veri toplama sistemleri, bir FFT veya DFT gerçekleştirirken, negatif frekans aralığını göstermez. Çift taraflı spektrumun pozitif kısmının genliği de iki ile çarpılır (genlikte değişmeden kalan sıfır Hertz hariç).

**Örnek Sinyaller ve Fourier Dönüşümleri**

Aşağıda, zaman alanı sinyallerinin ve Fourier Dönüşümlerinin bazı örnekleri bulunmaktadır

  
*Şekil 7: Zaman alanı sinyalleri (solda) ve karşılık gelen Fourier Dönüşümleri (solda).*

**Sinüs dalgası**

Sinüs dalgası, Fourier Dönüşümünün en temel bileşenidir. Sinüs dalgasının Fourier Dönüşümü, tek bir frekansta karşılık gelen faza (resimsiz) sahip tek bir genlik değeri üretir.

**Kare dalgası**

Frekans alanındaki kare dalga, genlik olarak sabit bir miktarda azalan tek numaralı harmoniklere sahiptir.

**Fourier Dönüşümünün Uygulamaları**

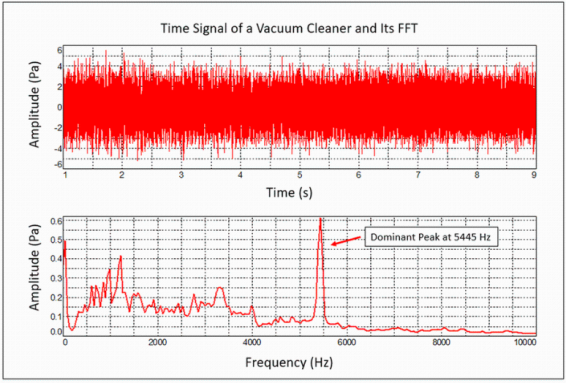
Fourier Dönüşümü, titreşim sorunlarının giderilmesinden görüntü işlemeye kadar birçok farklı kullanıma sahiptir.

Fourier Dönüşümü, bir sinyaldeki daha yüksek frekans bileşenlerini tanımlamak için kullanılabilir. Bu bileşenler istenmeyen gürültü veya titreşimin nedenini saptayabilir.

Bazı örnekler aşağıdadır:

**Elektrikli süpürge**

Sorunlu bir elektrikli süpürgeden zaman serisi ve ses basıncı verilerinin Fourier Dönüşümü'ne bakın ( Şekil 8 ). Elektrikli süpürge çalışırken keskin, hoş olmayan bir sese sahiptir. Zaman serileri bu elektrikli süpürgeyle ilgili belirli bir sorun göstermiyor.

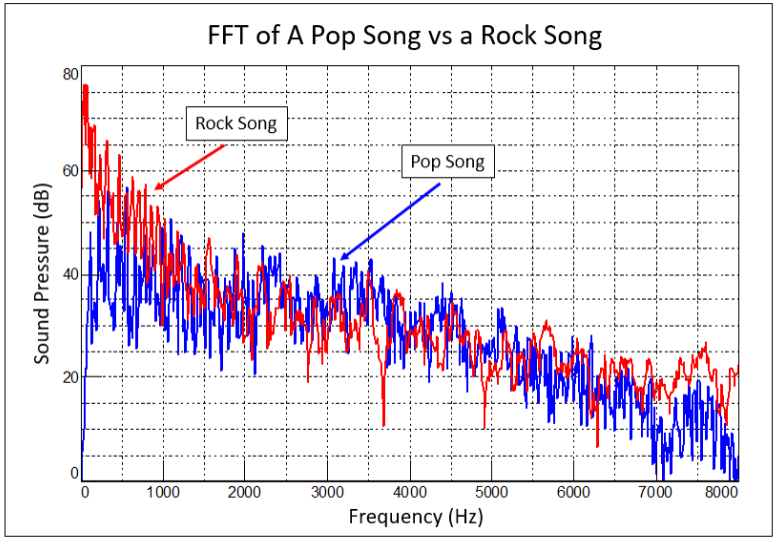
  
*Şekil 8: Elektrikli süpürge sesinin zaman alanı sinyali (üst grafik) bir sorunu belirtmese de, Fourier Dönüşümü (alt grafik) 5445 Hz'de büyük bir tepe gösterir.*

Zaman sinyalinin FFT'si 5445 Hz'de büyük bir tepe gösterir ve bu da hoş olmayan sese neden olur. Bu, fan kanadı geçişinden kaynaklanır. Mühendis, fan kanatları arasındaki mesafeyi ayarlayarak gürültülü fan sorununu çözebilir. Eşit aralıklı fan kanatları tek bir saf ton üretir.

**Müzik Analizi**

Sesi analiz ederken, bir frekans spektrumunun genliği genellikle desibel ( [logaritmik bir miktar](https://community.sw.siemens.com/s/article/basics-what-is-a-decibel-db-anyway-why-is-it-used) ) olarak görüntülenir. Desibel, insan ses algısının doğrusal genlikten daha iyi bir temsili olarak kabul edilir.

Bir rock şarkısının bir pop şarkısına karşı Fourier Dönüşümü'nü düşünün. Bir rock şarkısının bir pop şarkısından daha fazla bas (düşük frekans içeriği) olduğu gösterilebilir ( Şekil 9 ).

  
*Şekil 9: Bir Rock Şarkısı (kırmızı), Pop Şarkısı'ndan (mavi) daha düşük frekanslara sahiptir.*

1. **WEB AUDIO API**

HTML5 <audio>öğesinden önce, Web'in sessizliğini bozmak için Flash veya başka bir eklenti gerekiyordu. Web'deki ses artık bir eklenti gerektirmezken, ses etiketi sofistike oyunlar ve etkileşimli uygulamalar uygulamak için önemli sınırlamalar getirir.

Web Audio API, web uygulamalarındaki sesi işlemek ve sentezlemek için üst düzey bir JavaScript API'sıdır. Bu API'nın amacı, modern oyun ses motorlarında bulunan yetenekleri ve modern masaüstü ses üretim uygulamalarında bulunan bazı karıştırma, işleme ve filtreleme görevlerini eklemektir. Aşağıda, bu güçlü API'yi kullanmaya yönelik yumuşak bir giriş yer almaktadır.

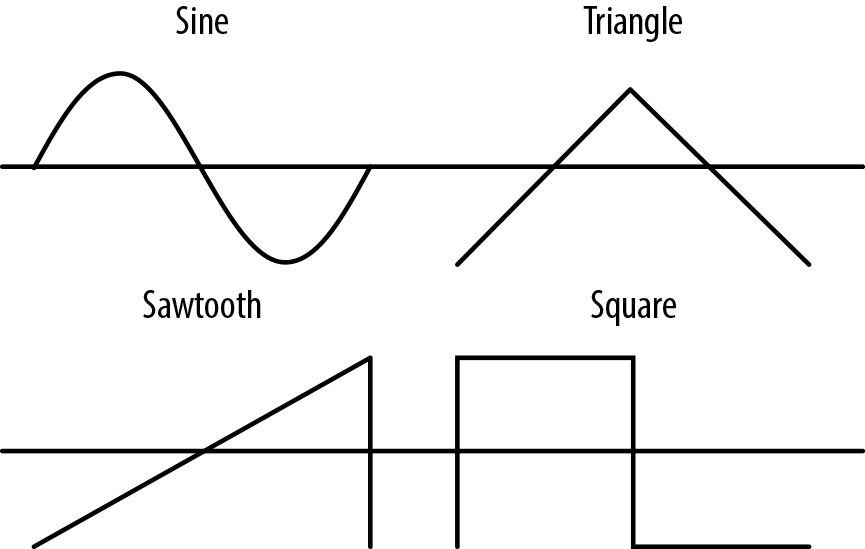
## AudioContext'i kullanmaya başlama

Bir [AudioContext](http://www.w3.org/TR/webaudio/#AudioContext-section) tüm sesleri yönetme ve oynama içindir. Web Audio API'sını kullanarak ses üretmek için bir veya daha fazla ses kaynağı oluşturun ve bunları AudioContext örnek tarafından sağlanan ses hedefine bağlayın. Bu bağlantının doğrudan olması gerekmez ve ses sinyali için işlem modülü olarak işlev gören çok sayıda ara [Ses Düğümünden geçebilir](http://www.w3.org/TR/webaudio/#AudioNode-section). Bu [yönlendirme](http://www.w3.org/TR/webaudio/#ModularRouting-section), Web Audio [spesifikasyonunda](http://www.w3.org/TR/webaudio) daha ayrıntılı olarak açıklanmaktadır .

Tek bir örneği AudioContext birden çok ses girişini ve karmaşık ses grafiklerini destekleyebilir, bu nedenle oluşturduğumuz her ses uygulaması için bunlardan sadece birine ihtiyacımız olacaktır. AudioNodes oluşturma ve ses dosyası verilerini çözme gibi ilginç Web Audio API işlevlerinin birçoğu yöntemdir AudioContext.

## Osilatör Tabanlı Doğrudan Ses Sentezi

Web Audio API'sındaki dijital ses, içinde yüzen bir dizi olarak temsil edilir. Kendi seslerimizi kaynağa ihtiyaç duymadan oluşturmak isteyebiliriz. Bunu, düzenli aralıklarla matematiksel bir işlevi değerlendiren ve bir diziye değerler atayan JavaScript kullanarak programlı olarak ses arabellekleri oluşturarak yapabiliriz. Bu yaklaşımı kullanarak, JavaScript ile sinüs dalgasımızın genliğini ve frekansını manuel olarak değiştirebilir, hatta rasgele sesler oluşturmak için birden fazla sinüs dalgasını bir araya getirebiliriz. Ancak Web Ses API osilatör -OscillatorNode- bizi karmaşık işlemlerden kurtarır. Web Ses API OscillatorNode düğümü yapılandırılabilir ve algılanabilen frekansa sahiptir Ayrıca üretilecek dalga türleri de vardır. Bu dalga tipleri sinüs, üçgen, testere dişi ve kare dalgalardan oluşur.



*Osilatörün üretebileceği temel ses dalgası şekilleri türleri*

**function** play(semitone) {

*// Create some sweet sweet nodes.*

**var** oscillator = context.createOscillator();

oscillator.connect(context.destination);

*// Play a sine type curve at A4 frequency (440hz).*

oscillator.frequency.value = 440;

oscillator.detune.value = semitone \* 100;

*// Note: this constant will be replaced with "sine".*

oscillator.type = oscillator.SINE;

oscillator.start(0);

}

**Web Audio API Frekans Analizi**

Web Audio API ile ses analizi yapmanın ana yolu AnalyserNodes kullanmaktır. Bu düğümler sesi hiçbir şekilde değiştirmez ve içeriğinizin herhangi bir yerine yerleştirilebilir. Bu düğüm grafiğinizde olduğunda, ses dalgasını incelemeniz için zaman alanı ve frekans alanı üzerinde iki ana yol sağlar. Elde ettiğiniz sonuçlar, belirli bir tampon boyutu (2048- 1024-512 gibi) FFT analizine dayanmaktır.

Düğümün çıktısını özelleştirmek için şunlar kullanılır

**fftSize**

Bu, analizi gerçekleştirmek için kullanılan arabellek boyutunu tanımlar. İki kişilik bir güç olmalı. Daha yüksek değerler, bazı performans kaybı pahasına, sinyalin daha ayrıntılı analizine neden olacaktır.

**frequencyBinCount**

Bu, otomatik olarak fftSize/ 2 olarak ayarlanan salt okunur bir özelliktir .

**smoothingTimeConstant**

Bu, sıfır ile bir arasında bir değerdir. Bir değeri büyük bir hareketli ortalama penceresine ve düzgün sonuçlara neden olur. Sıfır değeri hareketli ortalama olmaması ve hızlı dalgalanan sonuçlar anlamına gelir.

1. **KULLANILAN TEKNOLOJİLER**

Geliştiriciler için web teknolojileri kullanıldı. Bunlar HTML5, CSS, JavaScript tercih edilen üç temel dillerdir. Web API lerinden Mozilla Web Audio AudioContext API, Web Canvas API, Web Audio JavaSCript API ile projede yeraldı.

Veritabanı bağlantısı için PHP, MySQL tercih edildi.

Editör olarak da Sublime Text ile çalışıldı.

1. **TEKNOLOJİLERİ TERCİH ETME SEBEBİ**

Web sitelerinin belirli bir şekilde görünmesini ve çalışmasını sağlamak için web geliştiricileri farklı diller kullanır. World Wide Web'i oluşturan üç temel dil HTML, CSS ve JavaScript'dir.

BT dünyasında, internet, geliştirme veya tüketici kullanımı için önemli bir platformdur. Bir web sitesi geliştirirken, genellikle üç ana dil devreye girer. Bu diller JavaScript, CSS ve HTML'dir. HTML, çoğu web sayfasının bel kemiğidir. Temel olarak, başlıklardan paragraflara, gövdeye, bağlantılara ve hatta resimlere kadar belirli bir web sitesinin nasıl görüneceğinin yapısını oluşturmak için kullanılır.

JavaScript tartışmasız en açık programlama dilidir: ECMA-262, spesifikasyonu bir ISO standardıdır. Bu şartnameyi, bağımsız birçok kuruluşun uygulaması yakından takip etmektedir. Bu uygulamalardan bazıları açık kaynaklıdır. Ayrıca, dilin gelişimi, tüm büyük tarayıcı satıcıları da dahil olmak üzere birçok şirketi içeren bir komite olan TC39 tarafından ele alınmaktadır. Bu şirketlerin çoğu normal olarak rakiplerdir, ancak dilin yararına birlikte çalışırlar.

PHP hemen hemen tüm işletim sistemlerine kurulabilmekte ve hemen her HTTP sunucusu ile çalışabilmektedir. Peki PHP nedir? PHP ( PHP için özyinelemeli kısaltma : Hypertext Preprocessor ), web geliştirme için özellikle uygun olan ve HTML'ye yerleştirilebilen, yaygın olarak kullanılan bir açık kaynak kodlu genel amaçlı betik dildir.

HTML çıktısının (C veya Perl'deki gibi) bir sürü komut yerine, PHP sayfalarında "şey" yapan gömülü kod içeren HTML bulunmasıdır. PHP kodu, "PHP moduna" girip çıkabilmeyi sağlayan özel başlangıç ​​ve bitiş işleme talimatları < ? php ? > içine eklenmiştir. PHP'yi istemci tarafı JavaScript gibi bir şeyden ayıran şey, kodun sunucuda yürütülmesidir ve böylece istemciye gönderilen HTML üretilir. İstemci, komut dosyasını çalıştırmanın sonuçlarını alır, ancak temel kodun ne olduğunu bilemez. Web sunucunuzu, tüm HTML dosyalarınızı PHP ile işleyecek şekilde yapılandırabilirsiniz. PHP'yi kullanırken en iyi şey yeni bir kullanıcı için son derece basit ancak profesyonel bir programcı için birçok gelişmiş özelliklerinin olmasıdır. PHP'nin özelliklerinin uzun listesinin okunmasından korkmayın. Kısa sürede içeri atlayabilir ve birkaç saat içinde basit betik yazmaya başlayabilirsiniz.

PHP, MySQL ile iyi bütünleşir ve MySQL kullanmanıza yardımcı olacak kullanışlı işlevlerle dolu bir kütüphane içerir. PHP'de yazılmış birçok [veritabanı](https://en.wikipedia.org/wiki/Database) yöneticisi bile vardır .

MySQL, PHP'nin çalıştığı sunucunun bir parçası değil, farklı bir sunucudur. MySQL birçok veritabanı sunucularının biridir, bu açık kaynak kodludur.

1. **AVANTAJLARI DEZAVANTAJLARI**

**Avantaj:**

JavaScript çok esnek bir dildir, oldukça zarif bir çekirdeğe sahiptir ve nesne yönelimli programlama ve fonksiyonel programlama karışımını kullanmanızı sağlar. Web geliştiricileri için tarayıcı üzerinde consol uygulaması ile hata düzeneleme imkanı sunmaktadır. JavaScript'i tamamlayan ve dili daha kullanışlı hale getiren HTML5'ten daha fazla teknoloji de bulunmaktadır.

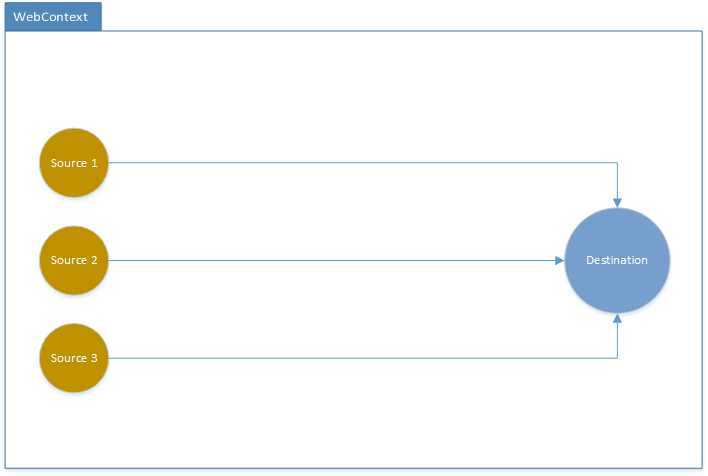
**Dez avantaj:** Web Audio API lerinden AudioContext tüm tarayıcılar tarafından desteklenmez.

1. **PROJENİN YENİLİKÇİ YÖNÜ**

WebAudio API, web uygulamalarındaki sesi işlemek ve sentezlemek için üst düzey bir JavaScript API'sıdır. Asıl işleme, Assembly, C, C ++ gibi temel uygulama yapılacaktır.

API, tek veya birden çok giriş kaynağını(source) bir hedefe (destination) yönlendiren bir grafikten oluşur. API bize, birleştirilebilen ve filtreleri uygulamak veya öncekilerin çıktılarını değiştirmek için kullanılan farklı AudioNodes'lar sunacaktır. Grafik, kaynakları, düğümleri yeniden oluşturmamıza ve sonucu hedefe yönlendirmemize yardımcı olacak AudioContext adlı bir arayüzle temsil edilir.

Düğümlerin çoğu, birkaç istisna dışında bir sonrakine bağlanabilir. Bu düğümler hedef gibi çıkmaz sokaklardır.



AudioContext arayüzü, her biri bir AudioNode tarafından temsil edilen, birbirine bağlanmış ses modüllerinden yapılmış bir ses işleme grafiğini temsil eder. Bir ses bağlamı hem içerdiği düğümlerin oluşturulmasını hem de ses işlemenin veya kod çözmenin yürütülmesini kontrol eder. Her şey bir bağlam (context) içinde gerçekleştiğinden, başka bir şey yapmadan önce bir AudioContext oluşturmanız gerekir. Her seferinde yeni bir tane başlatmak yerine bir AudioContext oluşturmanız ve yeniden kullanmanız önerilir ve aynı anda birkaç farklı ses kaynağı ve ardışık düzen için tek bir AudioContext kullanmak uygundur.

var AudioContext = window.AudioContext || window.webkitAudioContext;

var audioCtx = new AudioContext();

var oscillatorNode = audioCtx.createOscillator();

var gainNode = audioCtx.createGain();

var finish = audioCtx.destination;

// etc.

## <audio> etiketinden ses kullanma

Yüklemek istediğimiz sesi getirmek için <audio> etiketini kullanabiliriz. WebContext'deki createMediaElementSource yardımcı işlevini kullanarak sesi yüklemek için bir MediaElementAudioSourceNode oluşturabiliriz.

**Projenin web görünümü üzerinden anlatımı**

Sayfanın ilk çalışmasını sesi işlemek ve sentezlemek için üst düzey bir JavaScript API kullanıldı. Ses ve frekans sliderlarına bağlı osilatör ile oluşturulan tek tonlu ses incelendi. Bu ses 20-20000 hertz aralığında gözlemlenebiliyor ve duyulabiliyor. Ayrıca kulağın duyma sınırları da farkediliyor. Bu işlem sine, square, sawtooth ve triangle dalga formları ile sağlanıyor. Dalga şekillerini oluşturmak için JavaScript API Analyser düğümünden ve HTML’den faydalanılıyor.

Bir alt işlemde Web Auido Api ile mp3, wav gibi ses dosyalarının JavaScript API ‘nin Analyser düğümünden FFT (Fast Fourier Transform) kodları ile frekans değerleri 256 index boyutunda anlık alınıp sayfada gösteriliyor. Dalga formlarının grafiksel yapısı ise analiz düğümü ve html ile sağlanıyor. Ayrıca frekansların büyüklüğünü gösteren grafik bar Web Canvas Api sağlanıyor. Ses dosyalarının uygulamaya dahil edilmesi için PHP ile hazırlanan ekle, sayfa ve sil sayfalarına ait butonlarla gerçekleştiriliyor. Seç butonuyla uygulamaya eklediğimiz müzik, ses dosyaları seçilip play butonuyla çalıştırılıyor. Yine HTML <audio> etiketine erişimi sağlayan Web Auido Api AudioContext düğümü ile ses dosyalarının çalma hızını ayarlayabiliyoruz. Hız ile frekansın doğru orantılı olduğu bilinmektedir**.  V = f x λ**formulü hız ve frekans hakkında bir fikir sunar. Ayrıca bu bölüm

Bir alt işlemde ise Web Auido Api osilatör düğümü ile oluşturulan 2 adet osilator sesi ile siren sesi oluşturulmuş. Bu ses, saniyeler üzerinden frekansı değiştirilerek incelenmiştir. Bu bölüm güzel bir müzik çalar olarak da kullanılabilir.

Son olarak Web Canvas Api ile oluşturulan sinus dalga formu ve kare dalga formuna dönüştüren işlemler ve bu işlemler üzerinde freakans ve genlik grafiksel olarak gösterilmeye çalışılmıştır.

Projede yenilik olarak bakıldığında uygulamayı hazırlamış oldum. Web Auido Api ile ilgili az çalışma bulunuyor. Bu açıdan çeşitlilik sağlamış oldum.

1. **PROJE KAYNAK KODLARI**
2. **index.php**

<?php

include\_once("sql/Database.php");

?>

<!DOCTYPE html>

<html lang="tr">

<head>

<!-- Required meta tags -->

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<!--[if le IE 9]> -->

<script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>

<!-- Bootstrap CSS -->

<link rel="stylesheet" type="text/css" href="bootstrap-4.1/css/bootstrap.min.css">

<link rel="stylesheet" type="text/css" href="bootstrap-4.1/css/bootstrap.css">

<link rel="stylesheet" type="text/css" href="bootstrap-4.1/css/sesFrekans.css">

<script type="text/javascript" src="bootstrap-4.1/js/bootstrap.min.js"></script>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">

<title>Ses Frekans Analizi</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">

</head>

<body >

<header style="width: 100%; height: 200px; margin-bottom: 100px; background-image: url('image/iste.jpg');">

<div class="container-fluid" >

<div class="headerBilgi"><pre>

İskenderun Teknik Üniversitesi

Mühendislik ve Doğa Bilimleri Fakültesi

Bilgisayar Mühendisliği Bölümü

</pre></div>

</div>

</header>

<div class="clearfix"></div>

<div class="container sameHeight" style="margin-bottom: 100px; ">

<!-- Content here -->

<div class="col cellHeight" style="float: left; width: 75% !important; margin-bottom: 30px;">

<form id="indexForm">

<fieldset>

<legend class="kontrolLegend">Ses Frekans Analizi</legend>

<p style="text-align: center;padding: 10px;">Javascript, kendi seslerinizi ve müziklerinizi üretmak için harika imkanlar sunuyor. Bu şekilde oluşturulan sesin frekansa ve ses ayarı ile genliğe bağlı değişimini incleyebilirsiniz.</p>

<div class="form-check">

<label for="customRange3" style=" font-weight: 600;font-size: 17px;">Volume</label>

<input type="range" class="slider" name="vol" min="0" max="200" step="2" value ="30" onchange="updateSliderVol(this.value)" id="myVolume" oninput="myPlayer.setVolume(this.value/100);">

<p>Value: <span style="color:black;" id="demo1">30</span></p>

</div>

<script>

function updateSliderVol(slideAmount) {

var sliderVol = document.getElementById("demo1");

sliderVol.innerHTML = slideAmount;

}

</script>

<div class="form-check">

<label for="customRange3" style=" font-weight: 600;font-size: 17px;">Frekans</label>

<input type="range" class="slider" name="freq" min="20" max="20000" value ="220" step="2" onchange="updateSliderFreq(this.value)" id="myRange" oninput="myPlayer.setFrequency(this.value);">

<p>Value: <span id="demo">220</span></p>

</div>

<script>

function updateSliderFreq(slideAmount) {

var sliderFreq = document.getElementById("demo");

sliderFreq.innerHTML = slideAmount;

}

</script>

<fieldset class="form-group radioField">

<div class="col">

<div class="" align="center" >

<div class="custom-control custom-radio custom-control-inline">

<label style="text-align: left; padding-right:50px; font-weight: 600;font-size: 17px;">Wave Type</label>

<input type="radio" id="customRadioInline1" name="wave" value="sine" checked onclick="myPlayer.setWaveType(this.value);" class="custom-control-input">

<label class="custom-control-label" for="customRadioInline1">sine</label>

</div>

<div class="custom-control custom-radio custom-control-inline">

<input type="radio" id="customRadioInline2" name="wave" value="square" onclick="myPlayer.setWaveType(this.value);" class="custom-control-input">

<label class="custom-control-label" for="customRadioInline2">square</label>

</div>

<div class="custom-control custom-radio custom-control-inline">

<input type="radio" id="customRadioInline3" name="wave" value="sawtooth" onclick="myPlayer.setWaveType(this.value);" class="custom-control-input">

<label class="custom-control-label" for="customRadioInline3">sawtooth</label>

</div>

<div class="custom-control custom-radio custom-control-inline">

<input type="radio" id="customRadioInline4" name="wave" value="triangle" onclick="myPlayer.setWaveType(this.value);" class="custom-control-input">

<label class="custom-control-label" for="customRadioInline4">triangle</label>

</div>

</div>

</div>

</fieldset>

<div id="analyser"></div>

<script type="text/javascript" src="bootstrap-4.1/js/soundplayersource.js"></script>

<script>

const checkRadio = (field) => {

if ((typeof field.length == "undefined") && (field.type == "radio")) {

if (field.checked)

return field.value;

} else {

for (let i = 0; i < field.length; i++) {

if (field[i].checked)

return field[i].value;

}

}

return false;

};

var AudioContext = window.AudioContext || window.webkitAudioContext;

var audioFirst = new (window.AudioContext || window.webkitAudioContext)();

var analyser = audioFirst.createAnalyser();

let analyserDisplay = document.getElementById("analyser");

let myPlayer;

if (analyser.getFloatTimeDomainData) {

analyser.fftSize = 1024;

analyser.connect(audioFirst.destination);

myPlayer = new SoundPlayer(audioFirst, analyser);

for (let i = 0; i < analyser.fftSize; i++) {

analyserDisplay.appendChild(document.createElement("DIV"));

}

let dataArray = new Float32Array(analyser.fftSize);

setInterval(function() {

analyser.getFloatTimeDomainData(dataArray);

for (let i = 0; i < dataArray.length; i++) {

analyserDisplay.children[i].style.height = (50 + (20 \* dataArray[i])) + "px";

}

}, 500);

} else {

// getFloatTimeDomainData not supported

myPlayer = new SoundPlayer(audioFirst);

analyserDisplay.style.display = "none";

}

</script>

<div class="butonFrekansListe">

<ul style="list-style-type: none;text-align: center; padding-left: 0;">

<li><button type="submit" name="play" class="btn btn-primary" onclick="myPlayer.play(form.freq.value, form.vol.value/100, checkRadio(form.wave));

this.style.display = 'none';

form.stop.style.display = 'inline';

" value="Play &#9658;"><a href="#onay">Play</a></button>

<button type="submit" name="stop" style="display: none;" class="btn btn-primary" onclick="

myPlayer.stop();

form.play.style = 'inline';

this.style.display = 'none';

" value="Stop &#10073;&#10073;"><a href="#onay">Stop &#10073;</a></button></li>

</ul>

<p style="color:black!important;">İnsan kulağı 20 - 20000 Hertz arasındaki sesleri duyabilir.</p>

</div>

</fieldset>

</form>

</div>

<div class="col cellHeight1" style="width: 25%; float: left; text-align: center">

<form>

<fieldset>

<legend class="mysqlLegend">Ses / Video</legend>

<div class="butonListe">

<ul style="list-style-type: none; padding-left: 0;">

<li><button type="submit" class="btn btn-primary"><a href="ekle.php">Ekle</a></button></li>

<li><button type="submit" class="btn btn-primary"><a href="edit.php">Güncelle</a></button></li>

<li><button type="submit" class="btn btn-primary"><a href="clear.php">Sil</a></button></li>

</ul>

</div>

</fieldset>

</form>

</div>

<div class="clearfix"></div>

<!-------------------------------------------------------------->

<div>

<div class="container sameHeight" style="margin-bottom: 100px; ">

<!-- Content here -->

<div class="col cellHeight cellHeight3" style="float: left; width: 75% !important;">

<form id="indexForm" method="GET" class="barForm" action="#gidilenNokta" enctype="multipart/form-data">

<fieldset>

<legend class="kontrolLegend">Müzik - Fast Fourier Transform</legend>

<h6 style="text-align: center">Müzik seçmek için seç butonuna tıklamalısınız.</h6>

<p style="padding-top: 10px;text-align: center">Ses şiddeti, ses dalgalarının genliğini etkiler. Yüksek volume dalganın genliğini büyütür. </p>

<p style="padding-bottom: 10px;text-align: center">Hız frekansla doğru orantılıdır. <b>V = f x λ </b>formulü hız ve frekans hakkında bir fikir sunar. </p>

<div class="form-check">

<label for="customRange3" style=" font-weight: 600;font-size: 17px;" id="gidilenNokta">Volume</label>

<input type="range" class="slider" name="form2vol" min="0" max="100" step="1" value ="30" onchange="" id="form2myVolume" oninput="">

<p>Value: <span style="color:black;" id="form2demo1">0</span></p>

</div>

<script>

form2myVolume.onchange = function(e) {

var elm = e.target;

var source = elm.value;

audio.volume = parseFloat(source)/100;

var sliderVol = document.getElementById("form2demo1");

sliderVol.innerHTML = parseFloat(source);

};

</script>

<div class="form-check">

<label for="customRange3" style=" font-weight: 600;font-size: 17px;">Hız</label>

<input type="range" class="slider" name="hizRange" min="0.1" max="6" value ="1" step="0.1" onchange="" id="hizRange" oninput="myFunctionHiz(this.value)">

<p>Value: <span id="demoForm2">0</span></p>

</div>

<script>

document.addEventListener("oninput", myFunctionHiz);

function myFunctionHiz(e) {

audioFreq = event.target.value;

var sliderHiz = document.getElementById("demoForm2");

sliderHiz.innerHTML = parseFloat(audioFreq);

}

</script>

<div id="analyserform"></div>

<p>FFT boyutumuz 512 olduğu için yarısı kadar 256 indexe frekans değeri veriliyor. FFT boyutu 64 de alabilİrdik, bu sefer 32 indis ile değerleri alacaktık.

<p style="font-weight: 600;">Frekans Value: <br><span style="font-weight: 400;color:black; font-size: 14px;" id="getfreqvalue"><br>0</span></p>

<div class="clearfix"></div>

<div>

<div id="mp3\_player">

<div id="audio\_box"></div>

<canvas id="analyser\_render"></canvas>

</div>

<div class="clearfix"></div>

<div class="butonFrekansListe" id="butonFrekansListe">

<ul style="list-style-type: none;text-align: center;margin-left: 25%">

<?php

include\_once("sql/Database.php");

include "classes/Pages.php";

$page = new Pages();

if (isset($\_GET['sec']) ) {

?>

<form method="GET" action="#gidilenNokta">

<div class="form-group seciliForm">

<label class="control-label col-md-3 col-sm-2 col-xs-12" style="margin-right: 83%" for="first-name">Ses Listesi<span class="required">\*</span>

</label>

<div class="col-md-9 col-sm-9 col-xs-12">

<select id="select" name="sesSelect" required="required" class="form-control col-md-7 col-xs-12">

<option>Ses Kaydı Seçiniz</option>

<?php

$getName = $page->getAllTable("tbl\_sesfrekans");

if ($getName) {

while ($result = $getName->fetch\_assoc()) {

echo '<option value="'.$result['ad'].'">'. $result['sesad'].'</option>';

} }

?>

</select>

<a href="#sec" style="padding-right: 7px;">

<input type="submit"style="border-color:#55595a; background-color: #fff !important; opacity:.7; font-weight: 400 !important; border-width: 1px; border-radius: 5px!important;height: 30px;" value="Tamam" name="onay\_"></a>

</div>

</div>

</form>

<?php

?><li class="duzen"><button type="submit" id="sec" name="sec" class="btn btn-primary"><a href="#ok">Seç</a></button></li>

<?php

}

else

{

?>

<li class="duzen"><button type="submit" name="sec" id="sec" class="btn btn-primary"><a href="#ok">Seç</a></button></li>

<?php

}

?>

<?php

$ad="";

if($\_SERVER['REQUEST\_METHOD'] == 'GET' && isset($\_GET['onay\_'])){

$ad = $\_GET['sesSelect'];

?>

<script type="text/javascript">

$(document).ready(function(){

$(".seciliForm").hide();

return false;

});

</script>

<?php

}

?>

<li class="duzen"><button type="submit" name="play1" onclick="" class="playButton1 btn btn-primary" id="playClick" class="">Play</button>

</li>

<li class="duzen"><button type="submit" name="" id="stopAudio" onclick="" class="stopButton1 btn btn-primary">Stop &#10073;</button></li>

</ul>

</div>

</fieldset>

</form>

<script type="text/javascript" src="bootstrap-4.1/js/soundplayersource.js"></script>

<script>

var canvas2, ctx2, analyser2, fbc\_array, bars, bar\_x, bar\_width, bar\_height;

var source;

var filter;

analyser2=null;

source =null;

\_analyser=null;

var audioFreq = 1.0;

// window.webkitAudioContext = window.AudioContext || window.webkitAudioContext;

\_context = new (window.AudioContext || window.webkitAudioContext)();

var audio = new Audio();

audio.src= '<?php echo $ad;?>';

//audio.src='ses-video/6f66a65cb2.mp3'

audio.crossOrigin = 'anonymous';

audio.loop = true;

audio.controls = false;

audio.volume = 50/100;

audio.playbackRate = audioFreq;

document.body.appendChild(audio);

source = \_context.createMediaElementSource(audio);

var \_analyser = \_context.createAnalyser();

source.connect(\_analyser);

source.connect(\_context.destination);

let analyserDisplay\_ = document.getElementById("analyserform");

if (\_analyser.getFloatTimeDomainData) {

\_analyser.fftSize = 128;

\_analyser.connect(\_context.destination);

source.connect(\_context.destination);

//Alltaki iki satır grafikle ilgili değil frekans değerlerini yazdırmak için//

var bufferLength = \_analyser.frequencyBinCount;

var frequencyData = new Uint8Array(bufferLength);

////////////////////////////////////////

for (let i = 0; i < \_analyser.fftSize; i++) {

analyserDisplay\_.appendChild(document.createElement("DIV"));

}

let \_dataArray = new Float32Array(\_analyser.fftSize);

setInterval(function() {

\_analyser.getFloatTimeDomainData(\_dataArray);

for (let i = 0; i < \_dataArray.length; i++) {

analyserDisplay\_.children[i].style.height = (50 + (30 \* \_dataArray[i])) + "px";

}

\_analyser.getByteFrequencyData(frequencyData);

console.log("Frekans Değerleri : "+ frequencyData);

var sliderVol = document.getElementById("getfreqvalue");

sliderVol.innerHTML = frequencyData;

}, 500);

} else {

analyserDisplay\_.style.display = "none";

}

// Initialize the MP3 player after the page loads all of its HTML into the window

window.addEventListener("load", initMp3Player, false);

function initMp3Player(){

document.getElementById('audio\_box').appendChild(audio);

analyser2 = \_context.createAnalyser(); // AnalyserNode method

canvas2 = document.getElementById('analyser\_render');

ctx2 = canvas2.getContext('2d');

// Re-route audio playback into the processing graph of the AudioContext

//source2 = \_context.createMediaElementSource(audio);

source.connect(analyser2);

analyser2.fftSize = 1024;

analyser2.connect(\_context.destination);

source.connect(\_context.destination);

frameLooper();

}

// frameLooper() animates any style of graphics you wish to the audio frequency

// Looping at the default frame rate that the browser provides(approx. 60 FPS)

function frameLooper(){

window.webkitRequestAnimationFrame(frameLooper);

fbc\_array = new Uint8Array(analyser2.frequencyBinCount);

analyser2.getByteFrequencyData(fbc\_array);

ctx2.clearRect(0, 0, canvas2.width, canvas2.height); // Clear the canvas

ctx2.fillStyle = "rgb(167, 81, 92)"// Color of the bars

bars = 100;

for (var i = 0; i < bars; i++) {

bar\_x = i \* 3;

bar\_width = 2;

bar\_height = -(fbc\_array[i] / 2);

// fillRect( x, y, width, height ) // Explanation of the parameters below

ctx2.fillRect(bar\_x, canvas2.height, bar\_width, bar\_height);

// console.log('Frekans : ' + fbc\_array);

}

}

audio.addEventListener('timeupdate', function(){

if(!isNaN(audio.currentTime)) {

audio.playbackRate = audioFreq;

}

});

playClick.onclick = function(e) {

e.preventDefault();

var elm = e.target;

if(audio) {

audio.pause();

}

audio.load(); //call this to just preload the audio without playing

audio.play(); //call this to play the song right away

//

};

stopAudio.onclick = function(e) {

e.preventDefault();

var elm = e.target;

audio.load(); //call this to just preload the audio without playing

audio.pause(); //call this to play the song right away

audio.currentTime=0;

};

</script>

</div>

</div>

<br>

<!-------------------------------------------------------------->

<div class="col " style="float: left; width: 70.5% !important; background-color: #b2b7b83d; margin:2.5%; margin-right:10%; text-align: center; padding: 10%;">

<h3>Siren Sesi Frekans İnceleme</h3> <br>

<h5>0-3 saniye arası</h5>

<p>

Siren sesini 2 ayrı osilatör ile oluşturduk. İlk osilatörün başlangıç frekansını 1200 hertz, ikinci osilatörün freakansını ise 1/0.380 hertz olarak düzenledik. Dalga formu sine default olarak belirlendi, alternatif dalga formları ise square, sawtooth ve triangle seçilebilir.

</p><br>

<h5>3-12 saniye arası</h5>

<p>

Birinci osilatörün frekansını 3. saniyeden 7. saniyeye kadar düzenli olarak 100 hertz yine ikinci osilatörün freksını ise 10 hertz artırdık. Frekanslar 12. saniyeye kadar aynı kaldı.

</p><br>

<h5>12-20 saniye arası</h5>

<p>

Birinci osilatörün frekansını 12. saniyeden 16. saniyeye kadar düzenli olarak 500 hertz yine ikinci osilatörün freksını ise 20 hertz artırdık. Frekanslar 20. saniyeye kadar aynı kaldı.

</p><br>

<h5>20 saniye sonrası</h5>

<p>

Birinci osilatörün frekansını 20 saniye sonra 24. saniyeye kadar düzenli olarak 1000 hertz, yine ikinci osilatörün freksını ise 20 hertz azalttık. Frekanslar bu süreden sonra değişmedi.

</p>

<div>

<button class="playGenlikButton" onclick="play()">Play/Stop</button>

</div>

<br>

<div>

<div class="" align="left" >

<div class="custom-control custom-radio custom-control-inline">

<label style="text-align: left; padding-right:50px; font-weight: 600;font-size: 17px;">Wave Type</label>

<input type="radio" id="customRadioInline5" name="waveform" value="sine" checked onchange="changetype()" class="custom-control-input">

<label class="custom-control-label" for="customRadioInline5">sine</label>

</div>

<div class="custom-control custom-radio custom-control-inline">

<input type="radio" id="customRadioInline6" name="waveform" value="square" onchange="changetype()" class="custom-control-input">

<label class="custom-control-label" for="customRadioInline6">square</label>

</div>

<div class="custom-control custom-radio custom-control-inline">

<input type="radio" id="customRadioInline7" name="waveform" value="sawtooth" onchange="changetype()" class="custom-control-input">

<label class="custom-control-label" for="customRadioInline7">sawtooth</label>

</div>

<div class="custom-control custom-radio custom-control-inline">

<input type="radio" id="customRadioInline8" name="waveform" value="triangle" onchange="changetype()" class="custom-control-input">

<label class="custom-control-label" for="customRadioInline8">triangle</label>

</div>

</div>

</div>

<!-- <input type="range" class="slider" name="" min="0" max="10000" step="1" value ="" id="freqslider" onchange="changefreq()"> -->

</div>

<br>

<script>

var play,

changefreq,

oscillator,

changetype;

var oscProp = {

type:"sine",

frequency:1200,

playing:false

}

var context = new (window.AudioContext || window.webkitAudioContext)();

var now = context.currentTime;

var seconderfreq = 1/0.380;

window.onload = function(){

play = function(){

if(oscProp.playing){

oscillator.stop();

lfo.stop();

oscProp.playing = false;

}

else

{

oscillator = context.createOscillator();

oscillator.type = oscProp.type;

oscillator.frequency.setValueAtTime(oscProp.frequency, now);

lfo = context.createOscillator();

lfo.type = oscProp.type;

lfo.frequency.setValueAtTime(seconderfreq, now);

// 3 saniye içinde bir "kontrol noktası" belirleyin - bu rampanın başlangıç noktası olacaktır.

oscillator.frequency.setValueAtTime(oscProp.frequency, now+3);

lfo.frequency.setValueAtTime(seconderfreq, now+3);

// sonraki 4 saniye boyunca bir rampayı frekeans + 100Hz olarak ayarlayın.

oscillator.frequency.linearRampToValueAtTime(oscProp.frequency+100,now+7);

lfo.frequency.linearRampToValueAtTime(seconderfreq+1/0.1, now+7);

oscillator.frequency.setValueAtTime(oscProp.frequency, now+12);

lfo.frequency.setValueAtTime(seconderfreq, now+12);

oscillator.frequency.linearRampToValueAtTime(oscProp.frequency+500,now+16);

lfo.frequency.linearRampToValueAtTime(seconderfreq+1/0.05, now+16);

oscillator.frequency.setValueAtTime(oscProp.frequency, now+20);

lfo.frequency.setValueAtTime(seconderfreq, now+20);

oscillator.frequency.linearRampToValueAtTime(oscProp.frequency-1000, now+24);

lfo.frequency.linearRampToValueAtTime(seconderfreq-1/0.05 , now+24);

lfoGain = context.createGain();

lfoGain.gain.value = 450;

lfo.connect(lfoGain);

lfoGain.connect(oscillator.frequency);

oscillator.connect(context.destination);

oscillator.start();

lfo.start();

oscProp.playing = true;

}

}

changefreq = function(){

oscProp.frequency = document.getElementById("freqslider").value \* 50;

play();

play();

}

changetype = function(){

oscProp.type = document.querySelector("input[name = 'waveform']:checked").value;

play();

}

}

</script>

</div>

<!-------------------------------------------------------------->

<div class="clearfix"></div>

<div class="col anime1" style="width: 50%; float: left; text-align: center; margin-top: 3%;">

<br>

<br>

<br>

<h3 style="text-align: center;">Sinüs Dalga - Kare Dalga</h3>

<br>

<canvas id="canvasfft" width="520" height="305"></canvas>

<script>

//let canvas = document.querySelector('canvas')

var canvasfft = document.getElementById('canvasfft');

let ctx = canvasfft.getContext('2d')

let originX = 110

let originY = canvasfft.height/2

let a=1;

let t = 0;

let wave = []

function draw()

{

//console.log(wave.length)

//cleaning wave

if(wave.length > 500)

{

wave.pop()

}

ctx.fillStyle = 'rgb(49, 52, 53)'

let x = originX

let y = originY

let lastX = originX

let lastY = originY

ctx.fillRect(0, 0, canvasfft.width, canvasfft.height);

let r = 80

for(let i=0; i<a; i++)

{

let n = (i\*2)+1

let radius = r \* (2/(n\*Math.PI))

x +=radius\*Math.cos(n\*t)

y +=radius\*Math.sin(n\*t)

//drawing lines

ctx.beginPath()

ctx.strokeStyle = 'rgb(167, 81, 92)'

ctx.lineJoin = 'round'

ctx.lineCap = 'round'

ctx.fillStyle = 'rgb(167, 81, 92)'

ctx.moveTo(lastX, lastY)

ctx.lineTo(x,y)

ctx.stroke()

//drawing circles

ctx.beginPath()

ctx.strokeStyle = 'rgb(167, 81, 92)'

ctx.arc(lastX, lastY, radius, 0, 2\*Math.PI)

ctx.stroke()

//drawing dots at the ends of line

ctx.beginPath()

ctx.fillStyle = 'rgba(167, 81, 92, 0.3)'

//ctx.arc(x, y, radius, 7,0, 2\*Math.PI)

ctx.stroke()

lastX = x

lastY = y

}

wave.unshift(y)

//drawing line from circler to wave

ctx.beginPath()

ctx.strokeStyle = 'rgb(167, 81, 92)'

ctx.lineJoin = 'round'

ctx.lineCap = 'round'

ctx.fillStyle = 'rgb(167, 81, 92)'

ctx.moveTo(x, y)

ctx.lineTo(300, wave[0])

ctx.stroke();

//drawing wave

ctx.beginPath()

ctx.strokeStyle = 'rgb(167, 81, 92)'

ctx.lineJoin = 'round'

ctx.lineCap = 'round'

ctx.fillStyle = 'rgb(167, 81, 92)'

ctx.moveTo(200, wave[0])

for(let i=0; i<wave.length; i++)

{

ctx.lineTo(300+i, wave[i])

}

ctx.stroke()

t -= 0.05 // ters yönde hare

}

setInterval(()=>draw(), 21)

document.addEventListener("oninput", myFunctionsin);

function myFunctionsin(e) {

a = event.target.value;

var sliderType = document.getElementById("demoSinuscanva");

sliderType.innerHTML = parseFloat(a);

}

</script>

<br> <br>

<div class="form-check">

<label for="customRange3" style=" font-weight: 600;font-size: 17px;">Sin wave - Square wave</label>

<input type="range" class="slider" name="form2vol" min="1" max="20" step="1" value ="" onchange="" id="sliderfft" oninput="myFunctionsin(this.value)">

<p>Tepe Sayıs: <span style="color:black;" id="demoSinuscanva">0</span></p>

</div>

<div style="margin-top: 100px; padding:30px; padding-left: 0; margin-left: 0;">

<ul style="list-style-type: none;text-decoration: none;font-size: 10px; text-align: left;">

<h6>Teşekkürler</h6>

<li><a href=" https://www.the-art-of-web.com/javascript/creating-sounds/" class="thanks" >https://www.the-art-of-web.com/javascript/creating-sounds/</a></li>

<li><a href="https://gist.github.com/gkhays/e264009c0832c73d5345847e673a64ab?fbclid=IwAR3Y4vsHyWBiWPDLxVroxkRmBpPFauCJT0okNhJhjDfwcIBlbEo1jtQlmHM" class="thanks">https://gist.github.com/gkhays/e264009c0832c73d5345847e673a64ab?fbclid=IwAR3Y4vsHyWBiWPDLxVroxkRmBpPFauCJT0okNhJhjDfwcIBlbEo1jtQlmHM</a></li>

<li><a href="http://www.developphp.com/video/JavaScript/Analyser-Bars-Animation-HTML-Audio-API-Tutorial" class="thanks">http://www.developphp.com/video/JavaScript/Analyser-Bars-Animation-HTML-Audio-API-Tutorial</a></li>

<li><a href="https://stackoverflow.com/" class="thanks">https://stackoverflow.com/</a></li>

</ul>

</div>

</div>

<div class="col anime2" style="width: 50%; float: right; text-align: center;margin-top: 3%; vertical-align: middle; align-items: center;">

<br>

<br>

<br>

<h3 style="text-align: center;">Salınımlı Sinüs Dalgası</h3>

<br>

<div style="background-color: rgb(49, 52, 53); float: right;">

<canvas id="canvas1" width="500" height="300"></canvas>

<script type="text/javascript">

var genlik = 50;

var frequencyCn = 20;

function display(ctex,disp) {

var width = ctex.canvas.width;

var height = ctex.canvas.height;

var xMin = 0;

ctex.beginPath();

ctex.strokeStyle = "rgb(128,128,128)";

// X-Axis

ctex.moveTo(xMin, height/2);

ctex.lineTo(width, height/2);

// Y-Axis

ctex.moveTo(width/2, 0);

ctex.lineTo(width/2, height);

// Starting line

ctex.moveTo(0, 0);

ctex.lineTo(0, height);

ctex.stroke();

}

function displayCoor(ctex, y) {

var radius = 3;

ctex.beginPath();

// Hold x constant at 4 so the point only moves up and down.

ctex.arc(4, y, radius, 0, 2 \* Math.PI, false);

ctex.fillStyle = 'rgba(167, 81, 92, 0.3)';

ctex.fill();

ctex.lineWidth = 1;

ctex.stroke();

}

function sineWave(ctex, xOffset, yOffset) {

var width = ctex.canvas.width;

var height = ctex.canvas.height;

var scale = 20;

ctex.beginPath();

ctex.lineWidth = 2;

ctex.strokeStyle = 'rgb(167, 81, 92)';

var x = 4;

var y = 0;

//ctex.moveTo(x, y);

ctex.moveTo(x, 50);

while (x < width) {

y = height/2 + genlik \* Math.sin((x+xOffset)/frequencyCn);

ctex.lineTo(x, y);

x++;

// console.log("x="+x+" y="+y);

}

ctex.stroke();

ctex.save();

//console.log("Drawing point at y=" + y);

displayCoor(ctex, y);

ctex.stroke();

ctex.restore();

}

function drawing() {

var canvas1 = document.getElementById("canvas1");

var cntxt = canvas1.getContext("2d");

cntxt.fillRect(0, 0, canvas1.width, canvas1.height);

cntxt.clearRect(0, 0, 500, 500);

display(cntxt);

cntxt.save();

sineWave(cntxt, step, 50);

cntxt.restore();

step += 4;

window.requestAnimationFrame(drawing);

}

window.addEventListener("load", init, false);

function init() {

window.requestAnimationFrame(drawing);

}

var step = -4;

isPlayingen = false

var contextn = new (window.AudioContext || window.webkitAudioContext)();

var audion = new Audio();

audion.src='ses-video/99a1c1601c.mp3'

audion.crossOrigin = 'anonymous';

audion.loop = true;

audion.controls = false;

audion.volume = 20/100;

document.addEventListener("onclick",playgenlikClick);

function playgenlikClick()

{

if(audio) {

audion.pause();

}

isPlayingen = true;

audion.load();

audion.play();

};

document.addEventListener("onclick",stopgenlikClick);

function stopgenlikClick()

{

isPlayingen = false;

audion.pause();

audion.currentTime=0;

};

document.addEventListener("onclick",togglegenlikstop);

function togglegenlikstop() {

isPlayingen ? stopgenlikClick() : playgenlikClick();

//isPlaying = !isPlaying;

};

</script>

</div>

<div class="clearfix"></div>

<br>

<div class="form-check">

<label for="customRange3" style=" font-weight: 600;font-size: 17px;">Frekans</label>

<input type="range" class="slider" name="form2vol" min="0" max="100" step="1" value ="" onchange="" id="myfreqCanva" oninput="">

<p>Frekans Value: <span style="color:black;" id="demofreqCanva">0</span></p>

</div>

<div class="form-check">

<label for="customRange3" style=" font-weight: 600;font-size: 17px;">Genlik/Volume</label>

<input type="range" class="slider" name="form2vol" min="0" max="100" step="1" value ="" onchange="" id="mygenlikCanva" oninput="">

<p>Genlik Value: <span style="color:black;" id="demogenCanva">0</span></p>

<p>Genliği, müziğin ses şiddeti ile değiştirebilirsiniz.</p>

<button class="playGenlikButton" style="" onclick="togglegenlikstop();">Play/Stop</button>

</div>

<script>

mygenlikCanva.onchange = function (e) {

audion.volume=e.target.value/100;

genlik = audion.volume\*100;

source = genlik;

var slidergen = document.getElementById("demogenCanva");

slidergen.innerHTML = parseInt(genlik);

console.log("Genlik" + genlik);

}

myfreqCanva.onchange = function(e) {

var elm = e.target;

var source = parseInt(elm.value);

var periot = parseFloat(source);//-100 min, 1 max, step 1

frequencyCn = parseInt(101-periot);

var sliderFreq = document.getElementById("demofreqCanva");

sliderFreq.innerHTML = periot;

console.log("Frekans" + periot);

/\*frequencyCn = parseInt(source);//-100 min, 1 max, step 1

var sliderFreq = document.getElementById("demofreqCanva");

sliderFreq.innerHTML = parseInt((200-(-1\*frequencyCn)));

console.log("Frekans" + parseInt((200-(-1\*frequencyCn))));

\*/

};

</script>

</div>

<!--------------------------------------------------------------->

</div>

</div>

<div class="clearfix"></div>

<footer style="width: 100%; height: 200px; margin-top: 200px; background-image: url('image/iste.png');">

<div class="container-fluid" >

<div class= "footerDiv">

Copyright © 2020 İSTE. Tüm Hakları Saklıdır & by Nuray Çelik

</div>

</div>

</footer>

<!-- Optional JavaScript -->

<!-- jQuery first, then Popper.js, then Bootstrap JS -->

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49" crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js" integrity="sha384-ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy" crossorigin="anonymous"></script>

<script type="text/javascript" src="bootstrap-4.1/js/soundplayersource.js"></script>

</body>

</html>

1. **sesfrekans.css**

#bar {

position: fixed;

top: 20%;

left: 40%;

width: 40%;

height: 40%;

-webkit-transition: 0.1s ease all;

}

a.thanks

{

text-decoration: none !important;

color:black !important;

}

a.thanks:hover

{

color:#a7515c !important;

}

.p {

background-color: blue;

height: 100%;

width: 10%;

float: left;

}

.slidecontainer {

width: 100%;

}

.slider {

-webkit-appearance: none;

-ms-appearance: none;

width: 100%;

height: 15px;

border-radius: 5px;

background: #d3d3d3;

outline: none;

opacity: 0.7;

-webkit-transition: .2s;

transition: opacity .2s;

}

.slider:hover {

opacity: 1;

}

.slider::-webkit-slider-thumb {

-webkit-appearance: none;

appearance: none;

width: 25px;

height: 25px;

border-radius: 50%;

background: #a7515c;

cursor: pointer;

}

.slider::-moz-range-thumb {

width: 25px;

height: 25px;

border-radius: 50%;

background: #cf1f46;

cursor: pointer;

}

#analyser>div {

flex: 1;

border-top: 2px solid #a7515c;

transition: .1s;

}

#analyserform>div {

flex: 1;

border-top: 2px solid #a7515c;

transition: .1s;

}

.radioField

{

margin-top: 2rem !important;

margin-bottom: 2rem !important;

}

div#mp3\_player{ width:100%; height:220px; background:#2c3032; padding:5px; margin:50px auto; }

div#mp3\_player > div > audio{ width:500px; background:#2c3032; float:left; }

div#mp3\_player > canvas{ width:100%; height:150px; background: #313435; float:left;margin-top: -5px;}

#analyser

{

display: flex;

align-items: flex-end;

margin: 1em 0;

height: 100px;

border: 1px solid #ccc;

}

#analyserform

{

display: flex;

align-items: flex-end;

margin: 1em 0;

height: 100px;

border: 1px solid #ccc;

}

:checked.custom-control-input ~ .custom-control-label::before

{

color: #a7515c !important;

background-color: #a7515c !important;

border-color: #a7515c !important;

border : none !important;

outline: none !important;

}

.duzen

{

text-align:center !important;

float: left;

padding-right: 10px;

}

.butonFrekansListe

{

margin: 0 auto;

margin-top: 20px;

vertical-align: middle;

width: 100%;

align-items: center;

text-align: center;;

}

.butonFrekansListe>ul>li

{

text-align: center;

}

.butonFrekansListe>ul>li>button

{

border-radius: 50% !important;

width: 80px !important;

height: 60px !important;

outline: none !important;

border:none !important;

background-color: #55595a !important;

border-color: #55595a !important;

font-weight: 600;

opacity: .9;

}

.butonFrekansListe>ul>li>button:hover,.butonFrekansListe>ul>li>button:active, .butonFrekansListe>ul>li>button:focus,.butonFrekansListe>ul>li>button:link

{

border-radius: 50% !important;

width: 80px !important;

height: 60px !important;

outline: none !important;

border:none !important;

background-color: #a7515c !important;

border-color: #a7515c !important;

font-weight: 600;

opacity: 1 !important;

box-shadow: none !important;

}

.butonFrekansListe>ul>li>button>a

{

color: white;

text-decoration: none;

}

.butonListe

{

margin: auto;

margin-top: 50px;

vertical-align: middle;

width: 100%;

align-items: center;

}

.butonListe>ul>li

{

margin-bottom: 10%;

}

.butonListe>ul>li>button

{

border-radius: 50% !important;

width: 80px !important;

height: 60px !important;

outline: none !important;

border:none !important;

background-color: #55595a !important;

border-color: #55595a !important;

font-weight: 600;

opacity: .9;

}

.butonListe>ul>li>button:hover,.butonListe>ul>li>button:active, .butonListe>ul>li>button:focus,.butonListe>ul>li>button:link

{

border-radius: 50% !important;

width: 80px !important;

height: 60px !important;

outline: none !important;

border:none !important;

background-color: #a7515c !important;

border-color: #a7515c !important;

font-weight: 600;

opacity: 1 !important;

box-shadow: none !important;

}

.butonListe>ul>li>button>a

{

color: white;

text-decoration: none;

}

.mysqlLegend

{

text-align: center !important;

font-weight: 700 !important;

opacity: .9 !important;

font-size: 1.9rem;

padding-top: 30px;

}

.kontrolLegend

{

text-align: center !important;

font-weight: 700 !important;

opacity: .9 !important;

font-size: 2rem;

margin-bottom: 30px;

}

form

{

background-color: #b2b7b83d;

vertical-align: middle !important;

height: 800px;

margin: 0 auto !important;

width: 100%;

padding: 5%;

}

form.barForm

{

height:1200px !important;

}

select:focus

{

border-color: rgba(85,89,90) !important;

background-color: #fff !important;

}

select:hover

{

border-color: rgba(85,89,90) !important;

}

select:active

{

border-color: rgba(85,89,90) !important;

background-color: #fff !important;

}

select

{

border-color: rgba(85,89,90) !important;

}

select option

{

border-color: rgba(85,89,90) !important;

outline-color: #d3d3d3 !important;

background-color: #fff !important;

}

select option:active

{

outline-color: #d3d3d3 !important;

background-color: #fff !important;

border-color: rgba(85,89,90) !important;

}

select option:focus

{

outline-color: #d3d3d3 !important;

background-color: #fff !important;

}

select option:hover

{

outline-color: #d3d3d3 !important;

background-color: #d3d3d3 !important;

border-color: rgba(85,89,90) !important;

}

#kaydet

{

border-radius: 50% ;

width: 85px !important;

height: 60px !important;

outline: none !important;

background-color: #a7515c;

border-color: #a7515c ;

font-weight: 600;

}

}

#kaydet:hover

{

opacity: 0.7 !important;

}

.playGenlikButton

{

border-radius: 50% !important;

width: 90px !important;

height: 60px !important;

outline: none !important;

border: none !important;

background-color: #a7515c !important;

border-color: #a7515c !important;

font-weight: 600;

color: #fff;

}

.playGenlikButton:hover

{

opacity: .9;

}

#ekleForm

{

height: 500px;

}

#editForm

{

height: auto;

}

.footerDiv

{

margin: 0 auto;

text-align: center;

vertical-align: middle;

color: white;

font-size: 17px;

width: 100%;

font-weight: 600;

height: 80px;

position: relative;

padding-top: 80px;

opacity: .9;

left: 0;

}

.headerBilgi

{

text-align: center;

top: 0;

margin: 0 auto;

left: 0;

padding: 3%;

font-size: 25px;

font-weight: 600;

color: white;

opacity: .9;

}

pre

{

color:#fff !important;

}

@media only screen and (max-width: 768px) {

.cellHeight

{

float: none !important;

width: 80% !important;

margin-bottom: 5%;

max-height:800px !important;

vertical-align: middle;

}

form

{

height: auto;

}

.cellHeight1

{

float: none !important;

width: 75% !important;

clear: both;

}

.butonFrekansListe>ul>li

{

/\*float: none !important;\*/

margin-bottom: 2%;

}

.headerBilgi

{

text-align: right;

font-size: 20px;

}

}

@media only screen and (max-width: 608px) {

.headerBilgi

{

font-size: 12px;

}

}

@media only screen and (max-width: 413px) {

.cellHeight1

{

padding-top: 25%;

}

.butonFrekansListe

{

padding-left: 10%;

}

}

@media only screen and (max-width: 342px) {

.butonFrekansListe {

padding-left: 1%;

}

}

@media only screen and (max-width: 425px) {

.cellHeight{

margin-top: 284px;

width: 100% !important;

}

.cellHeight1{

margin-top: 284px;

width: 100%!important;

}

.cellHeight3

{

margin-bottom: 829px !important;

}

}

@media only screen and (max-width: 1200px) {

.cellHeight3

{

width: 100%!important;

margin-top: 200px;

}

.anime1, .anime2

{

width: 100% !important;

margin-top: 100px;

text-align: center;

}

}

@media only screen and (max-width: 790px) {

.cellHeight3

{

margin-bottom: 700px !important;

}

.anime1

{

margin-bottom: 200px !important;

}

.sameHeight

{

margin-bottom: 559px !important;

}

.duzen {

float: none !important;

}

}

1. **soundplayersource.js**

function SoundPlayer(audioContext,filterNode){

this.audioCtx=audioContext;

this.gainNode=this.audioCtx.createGain();

if(filterNode){this.gainNode.connect(filterNode);

}else{this.gainNode.connect(this.audioCtx.destination);

}

this.oscillator=null;}

SoundPlayer.prototype.setFrequency=function(val,when){

if(when){

this.oscillator.frequency.setValueAtTime(val,this.audioCtx.currentTime+when);

}else{this.oscillator.frequency.setValueAtTime(val,this.audioCtx.currentTime);

}return this;};

SoundPlayer.prototype.setVolume=function(val,when){

if(when){this.gainNode.gain.exponentialRampToValueAtTime(val,this.audioCtx.currentTime+when);

}

else{

this.gainNode.gain.setValueAtTime(val,this.audioCtx.currentTime);

}

return this;};

SoundPlayer.prototype.setWaveType=function(waveType){

this.oscillator.type=waveType;

return this;};

SoundPlayer.prototype.play=function(freq,vol,wave,when){

this.oscillator=this.audioCtx.createOscillator();

this.oscillator.connect(this.gainNode);

this.setFrequency(freq);

if(wave){this.setWaveType(wave);}

this.setVolume(1/1000);

if(when){this.setVolume(1/1000,when-0.02);

this.oscillator.start(when-0.02);

this.setVolume(vol,when);

}else{this.oscillator.start();

this.setVolume(vol,0.02);

}

return this;};

SoundPlayer.prototype.stop=function(when){

if(when){this.gainNode.gain.setTargetAtTime(1/1000,this.audioCtx.currentTime+when-0.05,0.02);

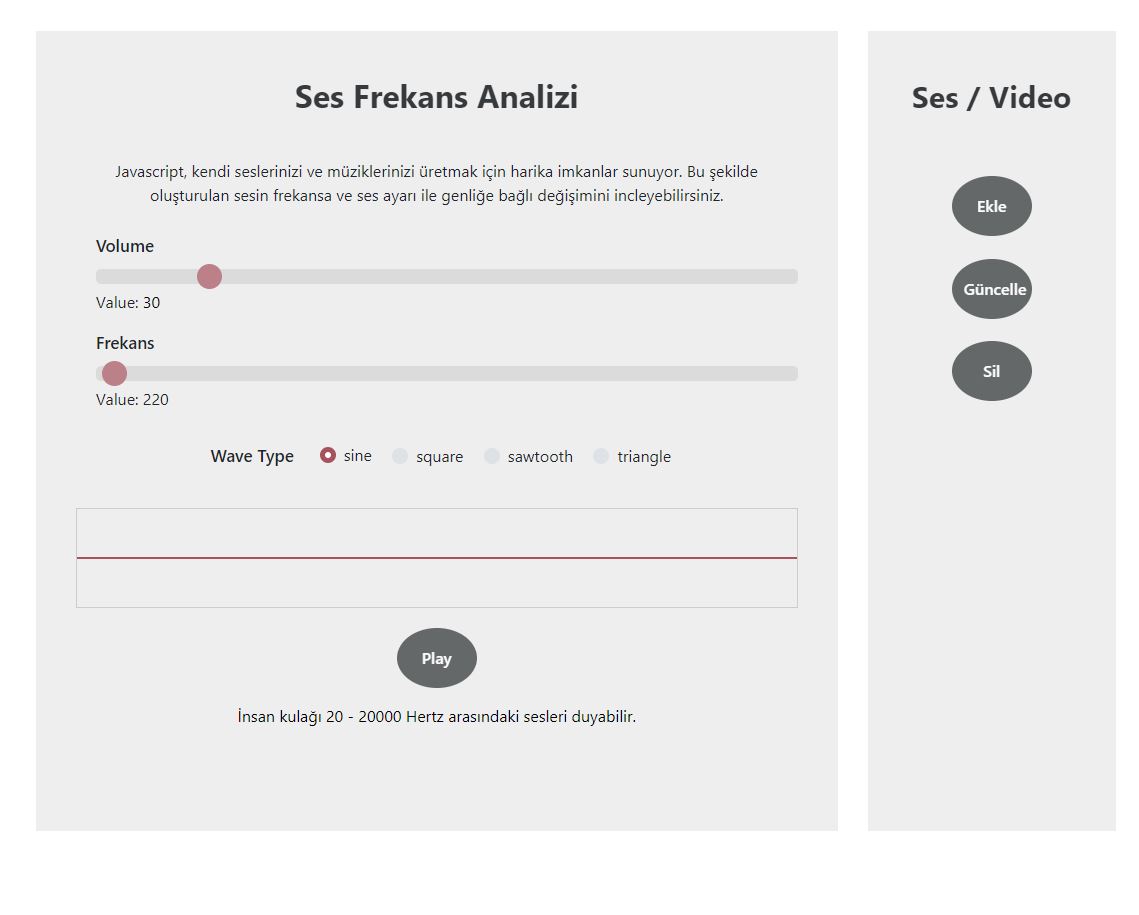
this.oscillator.stop(this.audioCtx.currentTime+when);}

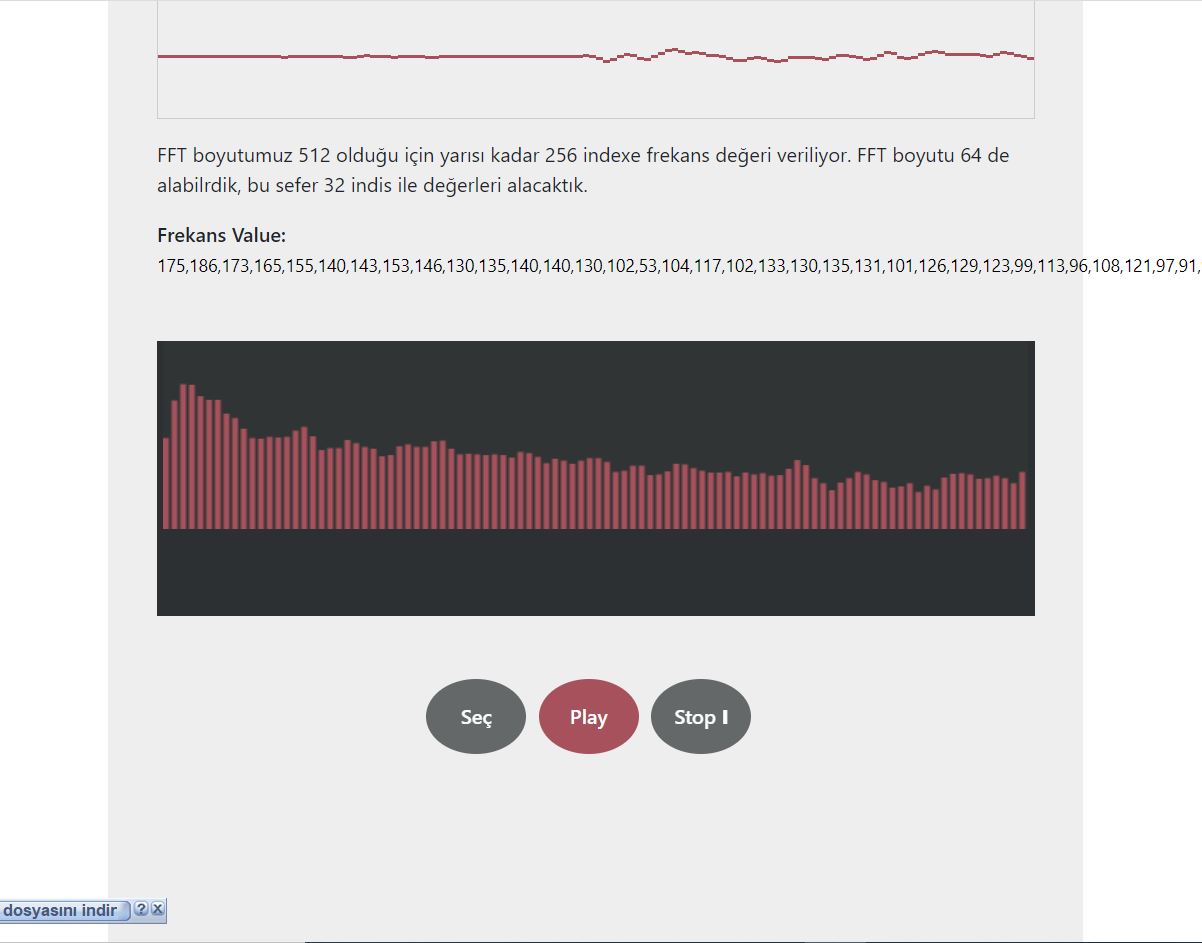
else{this.gainNode.gain.setTargetAtTime(1/1000,this.audioCtx.currentTime,0.02);

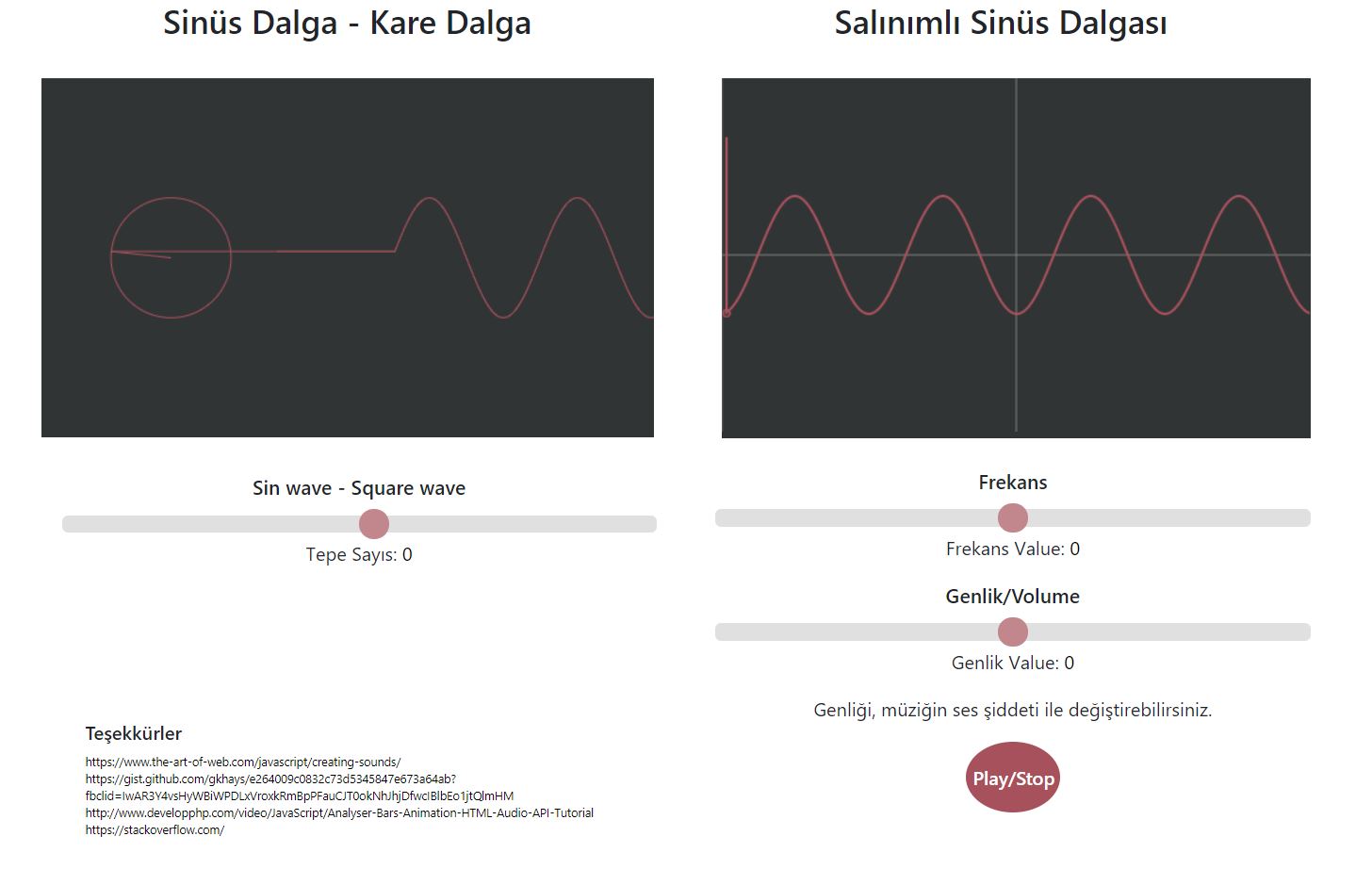
this.oscillator.stop(this.audioCtx.currentTime+0.05);}

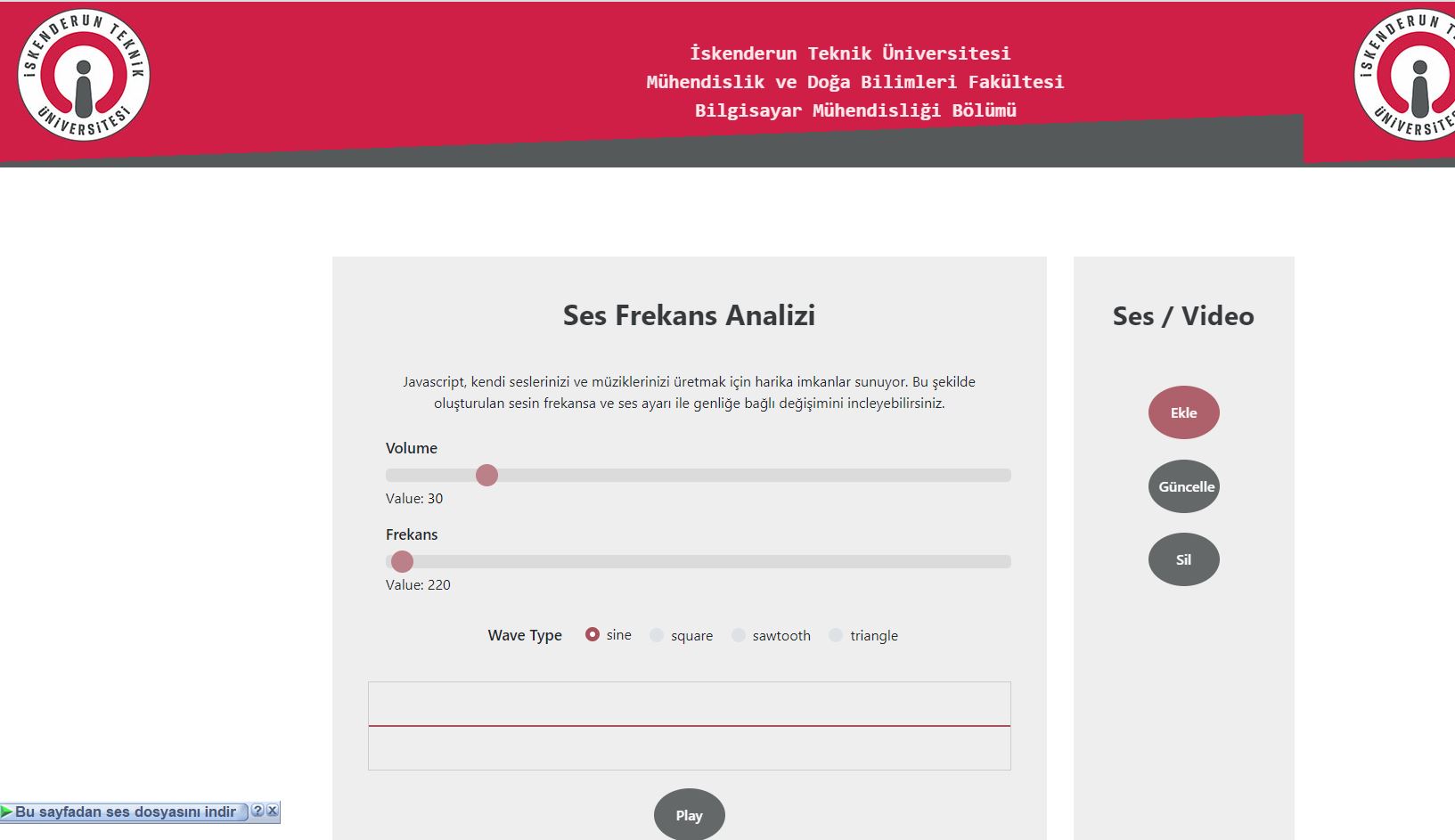
return this;};

1. PROJE GÖRSELLERİ









Yararlanılan Kaynaklar:

* <https://www.the-art-of-web.com/javascript/creating-sounds/>
* <https://gist.github.com/gkhays/e264009c0832c73d5345847e673a64ab?fbclid=IwAR3Y4vsHyWBiWPDLxVroxkRmBpPFauCJT0okNhJhjDfwcIBlbEo1jtQlmHM>
* <https://stackoverflow.com/>
* <http://www.developphp.com/video/JavaScript/Analyser-Bars-Animation-HTML-Audio-API-Tutorial>
* <https://learning.oreilly.com/library/view/javascript-creativity/9781430259459/9781430259442_Ch03.xhtml#Sec21>
* <https://tr.wikipedia.org/>
* <http://gwyddion.net/documentation/user-guide-en/wavelet-transform.html>
* <https://diyot.net/dalga-boyu/>
* <https://www.howmusicworks.org/103/Sound-and-Music/Amplitude-and-Frequency>
* <https://www.webassign.net/question_assets/unccolphysmechl1/lab_8/manual.html#:~:text=Sound%20waves%20can%20be%20analyzed,of%20the%20sound%20we%20hear.&text=The%20lowest%20frequency%20for%20a%20given%20sound%20is%20called%20the%20fundamental%20frequency.>
* <https://rion-sv.com/support/st_frequency_en.aspx>
* <http://digitalsoundandmusic.com/2-2-3-sound-analysis/>
* <https://community.sw.siemens.com/s/article/what-is-the-fourier-transform>
* <https://www.html5rocks.com/en/tutorials/webaudio/intro/>
* <https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API>
* <https://en.wikibooks.org/wiki/Introduction_to_Information_Technology/Web_Technologies>
* <http://speakingjs.com/es5/ch02.html>
* <https://k6.io/blog/webaudio_explained>