

# Full Stack Yazılım Geliştirme Uzmanlığı (JAVA)

Yurt içi ve Yurt Dışı  
Staj İmkânlı Programlar

İstanbul – Londra – Amsterdam – Estonya



[europecodingschool.com/](https://europecodingschool.com/)





EUROPE  
CODING SCHOOL

Genel

Programlar

Staj İmkânı

İletişim



## Nedir?

"Full Stack Yazılım Geliştirme Uzmanlığı (Java) eğitimi ile yazılım dünyasına tam hakimiyet kazanın! Java tabanlı backend geliştirme, modern frontend teknolojileri ve tam yığın projelerle uçtan uca yazılım geliştirme becerileri edinin. Uygulamalı projeler ve güncel endüstri trendleriyle kariyerinize güçlü bir adım atın. Yazılım geliştirme yolculuğunuzda uzmanlaşma zamanı!"



# Program İeriđi

- Giriş ve Kurulum
  - Java Nedir?
  - Java'nın Tarihesi ve Sürümleri
  - JDK ve IDE kurulumu (IntelliJ, Eclipse tanıtımı)
  - Temel komut satırı araçları (javac, java vb.)
- Temel Java Sözdizimi
  - Deđişkenler ve Veri Tipleri
  - Operatörler (Aritmetik, Karşılaştırma, Mantıksal vb.)
- Kontrol Yapıları
  - if-else, switch-case
  - for, while, do-while döngüleri
  - Metotlar (Yöntemler)
  - Parametre kullanımı, dönüş deđerleri, overloading
- Nesne Yönelimli Programlama (OOP) Temelleri
  - Sınıflar ve Nesneler (Constructor dahil)
  - Kapsülleme (Encapsulation)
- Kalıtım (Inheritance)
  - Super ve Subclass kavramları
  - Override ve Overload farkı
- Polimorfizm (Polymorphism)
  - Dinamik ve Statik Polimorfizm örnekleri
- Soyutlama (Abstraction)
  - Abstract sınıflar ve arabirimler (interface)
- Paketler ve Erişim Belirleyiciler
  - Paketlerin Kullanımı
  - Erişim Belirleyiciler (public, private, protected)
- İstisnalar (Exceptions)
  - Exception Kavramı (Checked ve Unchecked ayrımı)
  - Try-Catch-Finally
- Kalıtım (Inheritance)
  - Super ve Subclass kavramları
  - Override ve Overload farkı
- Throw, Throws
  - Özel Exception sınıfları yazma
- Koleksiyonlar (Collections) ve Generics
  - Collection Framework Genel Bakış (List, Set, Map)
  - Temel implementasyonlar (ArrayList, LinkedList vb.)

# Program İçeriği

- Generics (Tür Belirleyici)
  - Tip güvenliği
  - Iterator ve ListIterator
- Java 8+ Yenilikleri
  - Lambda ifadeleri
  - Fonksiyonel Arabirimler
- Stream API
  - filter, map, reduce gibi fonksiyonlar
  - Method ve Constructor Reference
- Optional Sınıfı
  - Yeni Tarih ve Zaman API (java.time)
- Giriş/Çıkış (I/O) İşlemleri
  - Java I/O Temelleri (Stream, Reader/Writer)
  - File I/O
- Çoklu İş Parçacığı (Multithreading)
  - Thread Oluşturma ve Yönetimi
  - Senkronizasyon
- Eşzamanlılık (Concurrency)
  - Yardımcı sınıflar (CountDownLatch, CyclicBarrier vb.)
  - Concurrency API
- JDBC ve Veritabanı Bağlantısı
  - JDBC Mimarisi
  - CRUD İşlemleri
  - PreparedStatement ve CallableStatement



# Program İçeriği

- PROJE UYGULAMA
  - PROJE ADI: ATM UYGULAMA
- Proje Başlangıcı ve Temel Yapının Kurulması
  - ATM Projesi Tanıtımı
  - Projenin genel tanıtımı ve hedefleri
  - Fonksiyonel gereksinimlerin belirlenmesi (Para çekme, para yatırma, bakiye görüntüleme, kullanıcı giriş kontrolü)
  - Proje Yapısının Oluşturulması
  - Sınıf ve paket yapısının planlanması
  - Nesne yönelimli programlama (OOP) kullanılarak sınıfların tanımlanması (Kullanıcı, Hesap, ATM)
  - Constructor, getter/setter ile sınıfların tamamlanması
- Kullanıcı Girişi ve Hesap İşlemleri
  - Kontrol Yapıları ve Döngüler Uygulaması
  - Kullanıcı giriş ekranı (if-else, switch-case kullanımı)
  - Kullanıcı doğrulama ve hata yönetimi (Exception handling)
  - Temel Hesap İşlemleri
  - Bakiye sorgulama, para çekme, para yatırma metodlarının yazılması



# Program İçeriği

- **Veri Kaydetme ve Koleksiyon Kullanımı**
  - Kullanıcı Hesaplarının Yönetimi
  - Kullanıcı bilgilerini ve hesaplarını ArrayList veya HashMap ile yönetmek
  - Generics kullanımı ile tip güvenli veri yapıları
  - Dosya İşlemleri ile Veri Kalıcılığı
  - Kullanıcı bilgilerini dosyaya kaydetme ve okuma (File I/O)
  - Kullanıcı hesap hareketlerini log dosyasında tutma
- **İleri Düzey İşlemler ve Multithreading**
  - Çoklu İş Parçacığı Kullanımı
  - ATM'de eş zamanlı işlemler (örneğin, farklı hesaplar için aynı anda işlem yapılması)
  - Thread kullanımı ve senkronizasyon
  - Exception Handling ve Hata Yönetimi
  - Kullanıcı hataları ve özel exception sınıfları yazma
  - Hataların loglanması
- **Veritabanı Entegrasyonu**
  - DBC ile Veritabanı Bağlantısı
  - Kullanıcı bilgilerini veritabanına kaydetme ve okuma
  - Hesap hareketlerinin SQL sorguları ile işlenmesi (CRUD işlemleri)
  - PreparedStatement Kullanımı
- **Proje Tamamlanması ve Sunum**
  - Kapsamlı ATM Uygulaması
  - Tüm fonksiyonların entegre edilmesi
  - Son testler ve hata düzeltmeleri
  - Proje Sunumu
  - Proje özelliklerinin açıklanması
  - Kullanıcı senaryoları üzerinden proje tanıtımı



# Öğrencilerimizin Başarıları



Zeynep Yağmur  
Sönmez

Bölüm: Full Stack Developer Intern

Kurum: Pixellette

Ülke: London – UK 

Öğrencimiz Zeynep Yağmur Sönmez,  
Pixellette 'de Full Stack Developer Intern  
olarak global teknoloji kariyerine başladı.



Fatma  
Akyıldız

Bölüm: Data Science Intern

Kurum: Cyber Gear

Ülke: Dubai 

Öğrencimiz Fatma Akyıldız, Cyber Gear 'da  
Data Science Intern olarak global teknoloji  
kariyerine başladı.



Batuhan  
Yüksel

Bölüm: Data Analyst Intern

Kurum: House of Nova

Ülke: ABD 

Öğrencimiz Batuhan Yüksel,  
House of Nova 'da Data Analyst olarak  
global teknoloji kariyerine başladı.

# Öğrencilerimizin Başarıları



Damla  
Karacabağ

Bölüm: Business Analyst Intern  
Kurum: Metahub  
Ülke: London – UK

Öğrencimiz, Damla Karacabağ,  
Metahub 'da, **Business Analyst Intern**  
olarak staja başladı



Alp Eren  
Arı

Bölüm: Cyber Security Intern  
Kurum: Evenness  
Ülke: Hollanda

Öğrencimiz Alp Eren Arı,  
Evenness 'de **Cyber Security Intern** olarak  
global teknoloji kariyerine başladı.



Zeynep  
Demir

Bölüm: Frontend Developer Intern  
Kurum: Evenness  
Ülke: ABD

Öğrencimiz Zeynep Demir,  
Evenness 'de **Frontend Developer Intern** olarak  
global teknoloji kariyerine başladı.



# Öğrencilerimizin Başarıları



İrem Zehra  
Yalçın



Bölüm: Data Science Intern

Kurum: Your Global Village

Ülke: ABD 

Öğrencimiz İrem Zehra Yalçın,  
Your Global Village 'de **Data Science Intern**  
olarak global teknoloji kariyerine başladı.



Cemil  
Çağlar



Bölüm: Data Science Intern

Kurum: YLIIRN

Ülke: ABD 

Öğrencimiz Cemil Çağlar,  
YLIIRN 'de **Data Science Intern**  
olarak global teknoloji kariyerine başladı.



Furkan  
İpek



Bölüm: Frontend Developer Intern

Kurum: Evenness

Ülke: ABD 

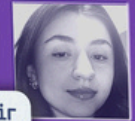
Öğrencimiz Furkan İpek,  
Evenness 'da **Frontend Developer Intern**  
olarak global teknoloji kariyerine başladı.



# Öğrencilerimizin Başarıları

Öğrencilerimiz  
**Dünya'nın**  
Her Yerinde!

Elif Özdemir



The Kaizen Future  
of Work Institute, Amerika

Osmangazi Üniversitesi



Öğrencilerimiz  
**Dünya'nın**  
Her Yerinde!

Kamuran Irmak Kılıç



MV AI World, Kanada

Osmangazi Üniversitesi



Öğrencilerimiz  
**Dünya'nın**  
Her Yerinde!

Betül Yazıcı



German University  
of Digital Science, Almanya

Bartın Üniversitesi



# +500 öğrenci stajını tamamladı.

Europe Coding School programlarını tamamlayarak  
sende stajını yurt içinde veya yurt dışında yapabilirsin.

Öğrencilerimiz  
**Dünya'nın**  
Her Yerinde!





# Eğitim Sonu İngilizce Sertifika İmkani

Eğitimi tamamlayan öğrencilerimize İngilizce sertifika verilmektedir.

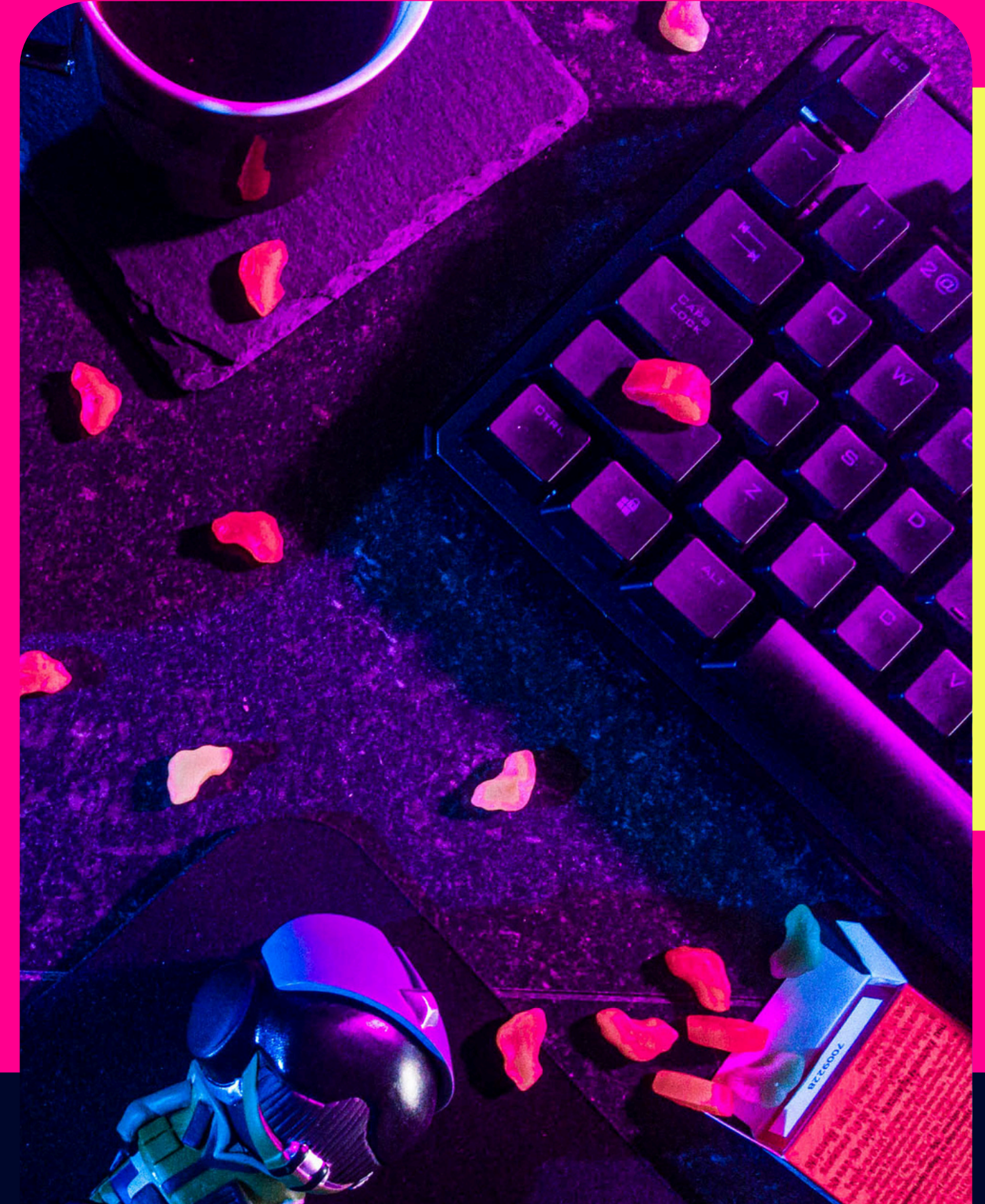
Ayrıca eğitim sonu yapılan sınavda 100 puan üstünden 85 ve üstü alan öğrencilerimize "Üstün Başarı Belgesi" verilmektedir.





# Program Sonunda Eldedeceğin Ünvanlar

- 01 Full Stack Developer
- 02 Backend Developer
- 03 Frontend Developer
- 04 Senior Software Engineer





# Ortalama Sektör Maaşları

- 1 – 3 yıl: **+50.000 TL / Ay**
- 3 – 5 yıl: **+80.000 TL / Ay**
- 5 – 10 yıl: **+120.000 TL / Ay**
- Hali hazırda kariyer platformlarında 3000 üzerinden iş ilanı bulunmaktadır.







EUROPE  
CODING SCHOOL

Genel

Programlar

Staj İmkânı

İletişim

60.000 TL

# Full Stack Yazılım Geliştirme Uzmanlığı (Java)

**Program Süresi:** 80 Saat – 13 Hafta

**Program Günleri:** Cumartesi (3 saat) – Pazar (3 saat)

**Eğitim Başlangıç Tarihi:** 22 Mart 2024

**Program İçeriği:** Full Stack Uygulamalı Development

**Eğitim Tipi:** Canlı ve videolarına sonradan erişilebilen.

**Sertifika İmkânı:** İngilizce Sertifika

**Sertifika İmkânı:** Staj İmkânı Yurt İçi ve Yurt Dışı

```
BufferUtil;
Texture;

OutputStream;
Option;
InputStream;
Buffer;
Map;
OpenGL;
Exception;
Matrix;

dia.opengl.GL.*;

try {
    Integer factoryIDCounter = 0;
    ID = 0;
    factoryID;
    mesh;
    Buffer vertexCache;
    Buffer normalCache;
    Map<Integer, ModelObject> objects = new HashMap<Integer, ModelObject>();
    renderSpec = DefRenderSpec.INSTANCE;
    = 0;

    y(Mesh mesh, RenderSpec renderer) {
        = 0;
        = renderer;
        = BufferUtil.newFloatBuffer(mesh.getVertexBuf().capacity());
        = BufferUtil.newFloatBuffer(mesh.getNormalBuf().capacity());
        factoryIDCounter) {
            inter++;

            factoryIDCounter;

            factory newInstance(File meshFile, RenderSpec renderer)
            saveException {

            .getName().endsWith(".mesh")) : "Parameter 'meshFile' is invalid";
            ;
            n ois = null;
            em.nanoTime();

            ectInputStream(new FileInputStream(meshFile));
            o.engine.model.Mesh) ois.readObject();

            ion ex) {
                dSaveException(ex.getMessage(), ex.getCause());
            }
            ex) {
                e();
            }
        }
    }

    package glDemo.engine.model;
    import glDemo.spatial.TreeGraph;
    import glDemo.spatial.TreeModel;
    import java.io.IOException;
    import java.io.ObjectInputStream;
    import java.io.ObjectOutputStream;
    import java.io.Serializable;
    import java.nio.FloatBuffer;
    import java.util.Arrays;
    import glDemo.linear.Matrix;
    import glDemo.linear.Quaternion;
    import glDemo.linear.Vector;

    public class ModelNode implements TreeNode, Transformable {
        public static final long serialVersionUID = 142857L;
        private static final int STEP = 1;
        private transient TreeGraph graph = null;
        private String name;
        private int parentIndex;
        private int[] childIndices = null;
        private transient Matrix local = Matrix.IDENTITY;
        private transient Quaternion orientation = Quaternion.IDENTITY;
        private transient Vector translation = Vector.ZERO;
        private boolean leaf = false;
        private int startIndex;
        private int endIndex;
        private int size;
        private transient float[] inVec = new float[STEP];
        private transient float[] outVec = new float[STEP];
        private transient String toStringCache = null;
        private ModelNode() {}

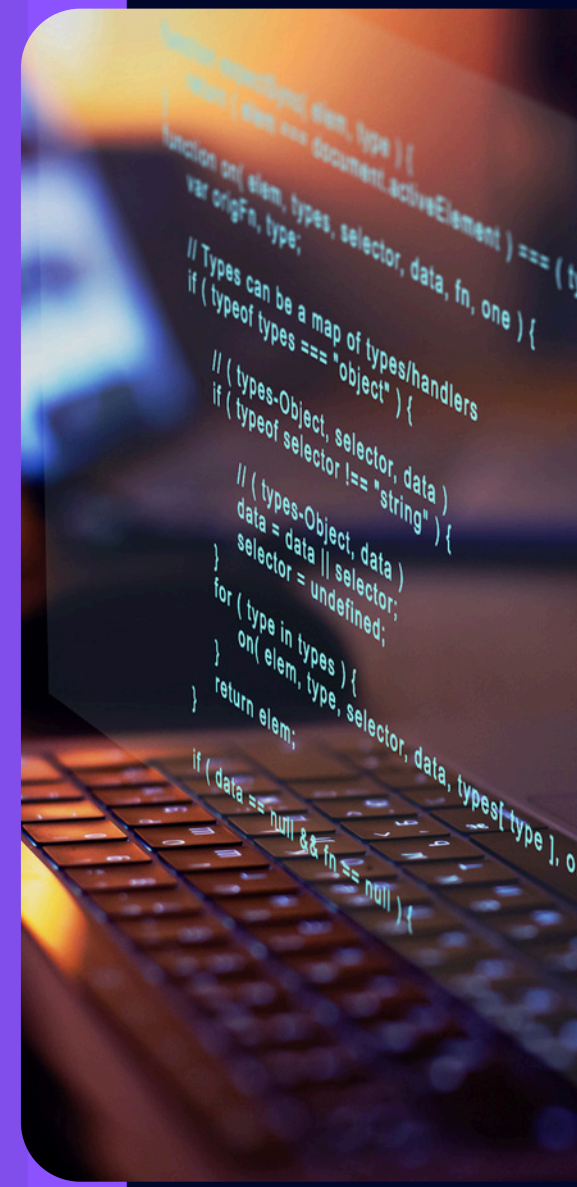
        public ModelNode(String name, int parentIndex, int[] childIndices) {
            this.name = name;
            this.parentIndex = parentIndex;
            if (childIndices == null || childIndices.length != 0) {
                this.leaf = true;
            } else {
                this.childIndices = childIndices;
            }
            this.startIndex = (startVert * STEP);
            if (startIndex < 0) {
                startIndex = 0;
            }
            this.size = noVerts * STEP;
            this.endIndex = this.startIndex + this.size;

            public void fillVertexBuffer(int envMask, Matrix local, Quaternion orientation, Vector translation) {
                assert base != null : "Matrix must not be null";
                assert in.capacity() == out.capacity() : "Buffers must have same capacity";
                Matrix tm = base.mul(this.local);
                in.position(startIndex);
                out.position(startIndex);

                while (in.position() < this.endIndex) {
                    in.get(inVec, 0, STEP);
                    out.set(outVec, 0, STEP);
                    outVec[0] = tm.m00 * inVec[0] + tm.m01 * inVec[1] + tm.m02 * inVec[2] + tm.m03;
                    outVec[1] = tm.m10 * inVec[0] + tm.m11 * inVec[1] + tm.m12 * inVec[2] + tm.m13;
                    outVec[2] = tm.m20 * inVec[0] + tm.m21 * inVec[1] + tm.m22 * inVec[2] + tm.m23;
                    outVec[3] = tm.m30 * inVec[0] + tm.m31 * inVec[1] + tm.m32 * inVec[2] + tm.m33;
                    out.position(outVec.length);
                }
            }
        }
    }

    function on( elem, types, selector, data, fn, one ) {
    var origFn, type,
    // Types can be a map of types/handlers
    if ( typeof types === "object" ) {
        // ( types-Object, selector, data )
        if ( typeof selector !== "string" ) {
            // ( types-Object, data )
            data = data || selector;
            selector = undefined;
        }
        for ( type in types ) {
            on( elem, type, selector, data, types[type], one );
        }
    }
    return elem;
}

if ( data == null && fn == null ) {
```





# İletişim ve Kayıt İçin

- Eğitim Danışmanlarımızı sizi araması için: [Başvuru Formu](#)
- Whatsapp Destek Hattı: [ECS Canlı Destek](#)
- Instagram: [İncele](#)

 Website

<https://europecodingschool.com/>

