T.C.

İSTANBUL AREL ÜNİVERSİTESİ

MESLEK YÜKSEK OKULU

WEB TABANLI

EĞİTİM SİSTEMİ

Mücahit Mustafa Perver

Ferhat Çark

BİLGİSAYAR TEKNOLOJİLERİ BÖLÜMÜ

BİLGİSAYAR PROGRAMCILIĞI PROGRAMI

MESLEKİ PROJE RAPORU

DANIŞMAN

Öğr. Gör. Timur İNAN

İSTANBUL 2017

T.C.

İSTANBUL AREL ÜNİVERSİTESİ

MESLEK YÜKSEK OKULU

WEB TABANLI

EĞİTİM SİSTEMİ

Mücahit Mustafa Perver

Ferhat Çark

BİLGİSAYAR TEKNOLOJİLERİ BÖLÜMÜ

BİLGİSAYAR PROGRAMCILIĞI PROGRAMI

MESLEKİ PROJE RAPORU

DANIŞMAN

Öğr. Gör. Timur İNAN

İSTANBUL 2017

ÖNSÖZ

“Java ile Oyun Programlama” adlı bu çalışma, İstanbul Arel Üniversitesi, Bilgisayar Programcılığı Programı, “Mesleki Proje Raporu” olarak hazırlanmıştır.

Çalışmaların yürütülmesi ve değerlendirilmesi sırasında yardımlarını esirgemeyen hocamız Öğr. Gör. Timur İNAN’a teşekkür ederiz.

Böyle bir mesleki projeyi bizlerin yapmasını sağlayan ve yazılım alanında kendimizi geliştirmemizi sağlayan bütün hocalarımıza, bize maddi olarak destek veren ailelerimize ve projeyi geliştirmemize destek veren bütün arkadaşlarımıza teşekkür ederiz.

MAYIS 2017, İstanbul

İÇİNDEKİLER

ŞEKİLLER

TABLOLAR

BÖLÜM 1

GİRİŞ VE AMAÇ

AMAÇ

Bilgisayardaki ve mobildeki video oyunları, teklonojinin eğlence alanındaki en yaygın kullanımlarından birisi olmaktadır. Video oyunlarının amaçları, insanların boş zamanlarında eğlendirerek veya bir şeyi beklerken ki süreyi sıkılmadan geçirmelerini sağlamaktır.

GİRİŞ

İlk olarak 1962 yılında Cambridge-Massachusetts'teki Higham Entstitüsünde Steve Russell, iki kişilik oyun seçeneği ve torpido fırlatma özellikleri içeren bir uzay gemisi oyunu geliştirdi. 1971 yılında, ilk jetonlu "Computer Space" isimli bir oyun makinesi oluşturuldu. 1971-1981 arasında Nutting Associates bu oyunun haklarını 500$'a yakın bir ücret ile alarak 1500 makine daha üretti fakat bu oyun beklenen ilgiyi görmedi. Nolan Bushnell bu işten kazandığı para ile kendi oyun geliştirme olan "Atari" şirketini kurdu.

1981'de Atari en hızlı büyüyen şirket haline gledi. 1988'de oyunlar yavaş yavaş renklenmeye başladı. 1989'da modem üzerinden oynanabilen ilk oyun da bu yıl piyasalara sürüldü. Ayrıca ilk CD-ROM oyunu "The Manhole" adlı oyun Activision tarafından geliştirildi. 1989 yılı ilklerin yılı olmuştur. 256-VGA grafik modunun kullanan ilk oyun, ses kartları, modem üzerinden oynanabilen ilk oyunda bu yıl piyasalara sürülmüştü. Bilgisayar oyunları sektörü bugünlerde ulaştığı noktayı 1989 yılında yapılan sıçramaya borçlu diyebiliriz.

Ev bilgisayarı olarak üretilip piyasaya sunulan ve kısa süre içinde geniş bir kitleye ulaşan, bilgisayar kavramının ev kullanıcıları arasında yaygınlaşmasında büyük katkısı olan Commodore 64 ve takipçisi Amiga500'dür. Üzeri gri, siyah tuşları olan, harf-rakam ve garip sembollere sahip, televizyona bağlandığında senin kontrolün alında olan, renkli bir ekrana sahip Amiga500 oyun kavramını tamamen altüst eden bir makine olmuştu. Çoğunlukla oyun amaçlı kullanılmıştır ve bir den çok programın aynı anda çalıştırılabilmesini sağlayan bir yapıya sahiptir. Ayrıca ilk ticari oyun olan Pong Atari firması tarafından tanıtılmıştır. 80'li yıllarda yavaş yavaş oyunlar türleşmeye başlamış ve metin tabanlı macera(adventure) ve rol yapma(RPG- Role Playing Game) oyunları yazılmıştır.

BÖLÜM 2

GENEL BİLGİLER

2.1. JAVA

Java, Sun Microsystems mühendislerinden James Gosling tarafından geliştirilmeye başlanmış gerçek nesneye yönelik, platformdan bağımsız, yüksek performanslı, çok işlevli, yüksek seviye, adım adım işetilen bir programlama dilidir. Java ilk çıktığında daha çok küçük cihazlarda kullanılmak için tasarlanmış ortak bir platform dili olarak düşünülmüştür ancak platform bağımsızlığı özelliği C ve C++ dilleriden çok daha üstün ve güvenli bir yazılım geliştirme ve işletme ortamı sunması nedeniyle hemen hemen her yerde kullanılmaya başlanmış. Şu anda özellikle kurumsal alanda ve mobil cihazlarda son derece popüler olan Java özellikle J2SE 1.4 ve gelecek 1.5 sürümü ile masaüstünde de gücünü arttırmayı hedefleniyor.

Java'nin ilk sürümü olan Java 1.0 (1995) Java Platform 1 olarak adlandırıldı ve tasarlama amaçına uygun olarak küçük boyutlu ve kısıtlı özelliklere sahipti. Daha sonra platformun gücü gözlendi ve tasarımında büyük değişiklikler ve eklemeler yapıldı. Bu büyük değişikliklerden dolayı geliştirilen yeni platforma Java Paltform 2 adı verildi ama versiyon numarası 2 yapılmadı, 1.2 olarak devam etti. 2004 sonbaharında çıkan Java 1.5, geçen 1.2, 1.3 ve 1.4 sürümlerinin ardından en çok gelişme ve değişikliği barındıran sürüm oldu.

Sürüm Tarihçesi

Java Dili, JDK 1.0’dan bu yana çeşitli değişikliklere uğramıştır.

1.0 (1996) — İlk sürüm.
1.1 (1997) — Önemli eklentiler. Örneğin iç sınıflar.
1.2 (4 Kasım, 1998) — Kod adı:Playground. API'de önemli değişiklikler, reflection özelliği ve JIT Derleyicisi nin eklenmesi. ancak dilin kendisinde önemli değişiklikler olmadı.
1.3 (8 Mayıs, 2000) — Kod Adı: Kestrel. HotSpot JVM sunuldu.
1.4 (13 Şubat, 2002) — Kod adı: Merlin. assert kelimesinin dile eklenişi, nio sınıfları ve pek çok yeni API değişikliği.
5.0 (29 Eylül, 2004) — Kod Adı: Tiger. (Önce 1.5 olarak duyruldu) Pek çok yeni dil özelliği eklendi.

Java Nasıl Çalışır?

Programcı java kodunu yazar.
Bu kod bir java derleyicisi ile derlenir. Sonuçta bytekod adı verilen bir tür makine kodu ortaya çıkar. Platform bağımsızlığını sağlayan şey bytecode'dir çünkü bir kere bytecode oluştuktan sonra yazılım tüm işletim sistemlerinde çalışabilir.
Bu byte kod Java virtüal Machine (Java Sanal Makinesı) tarafından adım adım işletilir.
Aşağıda java ve C++ kodunun geçirdiği aşamalar gösterilmiştir.

Yazılım

Java nesneye yönelik bir dil olduğundan tüm yazılım sınıflar ve nesneler üzerinden yürütülür. Sınıflar uygulamadaki nesnelerin tanımlandığı kod parçalarıdır. Java'da her bir sınıf bir dosya içerisinde yer alır. Dosyaların uzantıları .java şeklindedir. Dosya adı ise içinde tanımlanan sınıf ile aynıdır. Örneğin, BenimSınıf.java gibi.

Çalıştırma ve Java sanal makinesi

Sanal makine donanımdan bağımsız yazılım geliştirme ihtiyacına cevap verme amacıyla geliştirilen bir teknolojidir. Java'nın temel felsefesi olan "bir kere yaz, her yerde çalıştır" sanal makine sayesinde varolmuştur. Sanal makineyi bir yönden bir tür hayalı bir mikroişlemci gibi düşünebiliriz. Gerçek tüm mikroişlemciler (Intel Pentium, AMD Athlon, Sun Sparc vs) belirli bir grup komutu işlemek üzere tasarlanmıştır.

Bu komutlara işlemcinin komut kümesi adı verilir. Örneğin x86 komut kümesi gibi. Tüm yazılımlar çalışabilmek için önce bu komut kümesine dönüştürülür, daha sonra işlemci bu komutları sıra ile gerçek işlemci komutlarına dönüştürüp işletir. Java Sanal makinesi de Bytekod komut kümesini tıpkı işlemci gibi adım adım işletir. Java'nin interpreted bir dil olarak adlandırılmasının nedeni budur. Bytekod ilkel işlemlerin yanında (ilkel işlemler, mikroişlemci seviyesi komutlardır, aritmetik işlemler, bit işlemleri, bellek ve yığın işlemleri vs.) sanal makinenin üzerinde çalıştığı işletim sistemine yönelik işlemler de barındırır. Bu sayede Java Virtual Machine yazıcı, seri port, grafik, dosya servisi, ağ bağlantısı gibi yazılım ve donanım servislerine erişim yapabilir.Java'nın doğrudan bytecode çalıştırması performansının düşük olabileceği izlenimini verebilir. Ancak, JVM tasarımı geçen 10 yılda çok değişmiş ve geliştirilmiştir. Şu anda java'nın performansı çoğu alanda C++ dilinin performansına yakın bir seviyededir ve işlemci hızı ve bellek miktarının her geçen yıl katlanarak artması ile performans konusu çoğu uygulamada artık ikinci planda kalmıştır.

Java ile ilgili bazı kavramlar

Hot Spot teknolojisi

Java sanal makinesi HotSpot adı verilen özel bir teknolojiyi içinde barındırır. HotSpot yani sıcak nokta , bir yazılımda sürekli olarak tekrarlanan ve üzerinden geçilen kod bölümlerine verilen bir isimdir. Java sanal makinesi bir kod çalışmaya başladıktan sonra sıklıkla kullanılan kod bloklarını gözler ve bir süre sonra bu bytecode bloklarının çalışılan sistemdeki gerçek işlemci komut karşılıklarını bir tür cep belleğe yazar ve zaman ilerledikçe artık byte kod üzerinden değil doğrudan sistemin öz komutlarını kullanarak yazılımın o bölümlerini işletmeye başlar. Bu şekilde ciddi performans avantajı sağlanmıştır.

JIT

Java ilk çıktığında bytecode işletme hızı çok iyi değildi. Yerine göre sistemin öz yazılımlarından 5-10 kat yavaş çalışıyordu. Bu nedenle bazı yazılım geliştirme şirketleri JIT yani "Just-in-time compile", "anında derleme" araçları üretmeye başladılar. Yapılan şey byte kodu sanal makinenin kurulu olduğu gerçek sistemin diline anında derleme yaparak dönüştürmesiydi. Bu sayede performansta ciddi artışlar sağlandı ama 2000 yılından sonra HotSpot teknolojisinin gelişmesi ile JIT'in işlevi VM'içinde yer almaya başlamış, işlemci hızı ve bellek miktarının dramatik biçimde artması ile dış JIT yazılımları popülerliğini kaybetmiştir. Bugün halen bir kaç ürün pazarda bulunsa da genellikle bu yöndeki ihtiyaç yok olmuş gibi gözükmektedir.

Java API

Java API, java yazılımlarında kullanılıan yazılım kütüphanelerine genel olarak verilen isimdir. Java API ile disk, grafik, ag, veri tabani, güvenlik gibi yüzlerce konuda kullanıcılara erişim imkanı sunulur. Java API J2SDK'nin bir parçasıdır.

Çöp toplayıcı (Garbage Collector)

Çöp toplayıcı java'nın en belirgin özelliklerinden birisidir. C++, C gibi dillerin en büyük handikaplarından birisi dinamik bellek yönetimidir. Yazılımda işaretçi (pointer) kullanarak dinamik olarak bellek ayırdıktan sonra o bellek ile işiniz bittiğinde mutlaka ayrılan belleği bellek yöneticiye özel metodlar yardımıyla (delete, destructor vs.) iade etmeniz gerekir yoksa bellek sızıntısı (Memory Leak) oluşur ve bu bir süre sonra yazılımın ve işletim sisteminin beklenenden farklı davranmasına yol açar.

Bugünün tüm büyük C ve C++ yazılımları az da olsa bellek sızıntısı içerir (işletim sistemleri dahil). Sızıntıların tespiti oldukça güçtür ve bulunması zor hatalara yol açar. Çöp toplayıcı sayesinde Java'da bir nesne oluşturulduktan sonra o nesne ile işiniz bittiğinde hiç bir şey yapmanız gerekmez. Sanal makine akıllı bir biçimde kullanılmayan bellek bölümlerini belirli aralıklarla ya da adaptif metodlarla sisteme otomatik olarak temizler ve sisteme iade eder. Bu işleme çöp toplama, ya da "garbage collection" adı verilir. Çöp toplama sistemlerinin yapısı oldukça karmaşıktır ve geçen yıllar içinde büyük gelişmeler kaydedilmiştir. Çöp toplayıcının varlığı java'da bellek sızıntısı olmayacağı anlamına gelmez ama bellek sızıntıları daha ender olarak ve farklı şekillerde karşınıza çıkar ve genellikle tedavi edilmesi daha kolaydır.

Jar

Jar, aslında bir tür sıkıştırma formatıdır. Jar ile derlenen java kodları ile oluşan yazılımın paketlenip taşınması kolay bir hale getirilir. Jar dosyaları temelde bytekod blokları içerir. Jar dosyaları genellikle kütüphane oluşturmada ya da uygun biçimde hazırlanırsa işletim sisteminden doğrudan çalıştırılabiliecek bir şekilde kullanılabilir (Executable jar, işletilebilir jar) jar dosyalarının içeriğini sıkışıtırma yazılımları ya da java yazılım geliştirme araçları ile inceleyebilirsiniz. Java 1.5 ile yeni bir tür jar oluşturma metodu da kullanıma girecek. Pack200 adı verilen hiper-compression algoritması ile jar dosyaları 8 kata varan oranlarda daha az yer kaplayacak. Bunun özellikle uzak uygulamaların kullanımını ciddi biçimde kolaylaştırması bekleniyor.

AWT ve Swing

AWT, ilk java ile birlikte geliştirilen temel grafik arayuz oluşturma kütüphanesine verilen isimdir ancak Java 2 platformu ile birlikte AWT'nın yetersiz görülmüş ve çok daha geniş ve gelişmiş özelliklere sahip Swing kütüphanesi sisteme eklenmiştir. Özellikle çok platform destekleyen yazılımlarda kullanıcı arayüzü geliştirme aracı olarak swing halen önemini korumaktadır. Swing önceleri işletim sisteminin kullandığı donanım grafik hızlandırma araçlarını kullanmadığından yavaşlığı ile eleştirilere hedef olmuştu. Özellikle Java 1.4 ile Swing, hem genel olarak sanal makinenin hızlanması ve kısmen donanım hızlandırmayı kullanması ile bu kötü şöhretinden sıyrılmaya başladı. Java 1.5 ile donanım özellikle OpenGL kullanımı ve yeni arayuz gösterim şekli ile java'nin masaüstü uygulama geliştirmede popülerleşmesi bekleniyor. AWT hala swingin bir alt katmanında, temel 2 boyutlu grafik işlemlerinde kullanılmaya devam ediyor.

SWT

SWT swing'e bir alternatif olarak IBM tarafından geliştirilen bir gösterim sistemidir. Swing'den en büyük farkı çalıştığı işletim sistminin grafik kütüphanesi ve komutlarını kullanmasıdır. Bu nedenle SWT uygulamaları Swing'e göre çoğu yerde daha hızlı ve işletim sistemindeki diğer uygulamaları andıran bir şekilde çalışmasını sağlar. Ancak yapı itibari ile SWT kullanımı Swing kadar efektif olamayabiliyor (özellikle olay mekanizması, tablo ve ağaç yapılarındaki yavaşlığı, ayrıca linux performansı ile SWT eleştirilmiştir). Swingin Java 1.5 ile performans açığını kapatacağı iddia edilse de SWT'nin de artık java camiasında kabul görmüş bir sistem olduğu aşıkardır. SWT'nin dezavantajı ise java'nin bir parçası olmamasıdır. yanı SWT uygulamaları SWT kütüphanesi ile birlikte dağıtılmaktadır. En bilinen SWT uygulaması ünlü java yazılım geliştirme aracı Eclipse'tir. Bununla birlikte son yıllarda Swing ile profesyonel derecede arayuza sahip masaüstü yazılımları da ortaya çıkmıştır. Sonuçta her şey yazılımcının aracı ne kadar efektif kullandığına bakıyor.

Applet

Applet, uzaktaki sistem üzerinden indirilip internet tarayıcı üzerinde çalıştırılabilien java uygulamalarına verilen isimdir. Java'nın son kullanıcılar tarafından tanınması applet sayesinde olmuştur dersek yanlış olmaz. Applet'ler sisteme zarar veremeyecek bir şekilde tasarlanmıştır ve bugün özellikle oyun sitelerinde halen yaygın olarak kullanılmaktadır. İçerisinde applet olan bir sayfayı açmaya çalıştığınızda tarayıcınız otomatik olarak java sanal makinesini çalıştırıp ekranın applet'e ayrılan bölümünde uygulamanın çalışmasını sağlar.

WebStart

Webstart teknolojisi Windows ve Linux sistemlerinin baş belası olan uygulama kurulum, güncelleme ve silme dertlerine deva olmak üzere tasarlanmış bir sistemdir. Özellikle Java 1.5 ile daha yaygın kullanılmaya başlayacağı tahmin edilen webstart teknolojisi kısaca yazılımların uzaktan yerel sisteme güvenli olarak kurulmasını ve korumalı bir alanda çalıştırılmasını sağlar. Appletlerin bir sonraki adımı olarak görülebilir. Bir webstart uygulamasını kurmak için internet üzerindeki özel bir bağlantıya tıklamak yeterli. Sistem otomatik olarak webstart sistemini çalıştırıp yazılımı java cep belleğine indirir.

İsterseniz masaüstüne kısayol koymasını da sağlar. Daha sonra sistem off-line ya da on-line olsa bile uygulama çalıştırılabilir, ve uzaktaki yazılım güncellendiğinde otomatik olarak -istenirse- yerel makinedeki yazılımın da güncelenmesi sağlanabilir. Kullanıcının özel olarak izin vermesi halinde uygulama yerel sisteme erişim hakkı kazanabilir.

Aksi taktirde webstart uygulamaları sisteme yazma işlemi gerçekleştiremezler (yanı virüs ve zararlı yazılım tehlikesini son derece aza indirger.)

Java Yazımı

Sınıf temelli nesneye yönelik bir dil olan Java, yazım olarak C++ ile benzerlikler arz eder. Java'nın yanında C#, Perl, JavaScript gibi diller de aynı dil ailesine aittir. "{}" şeklinde süslü parantezler içerisindeki bloklar, ++ arttırma ve—azaltma işleçleri bu dilin belirgin özelliklerindendir.

Aşağıdaki komut satırları ekrana “Merhaba Dünya!” yazdıracaktır.

```
// MerhabaDünya.java
public class MerhabaDünya{
   public static void main(String[] args) {
      System.out.println("Merhaba Dünya!");
   }
}
```

- "MerhabaDünya.java" ile "public class MerhabaDünya" bu kısımda iki isim aynı olmalıdır aksi takdirde uygulama çalıştırılamaz.
- class: Sınıf tanımlayabilmek için class ön eki şarttır.
- public: Sınıfın dışarıdan erişebilir olduğu anlamına gelmektedir.
- static: Sınıf tarafından paylaşılabildiği anlamına gelmektedir.
- void: Bir değer döndürmediği anlamına gelmektedir.
- Public ve Static'e erişim belirleyicisi (access specifier) de denir.
- Void'e dönüş tipi (return type) de denir.
- String args[]: Parantezin içinde yöntemin aldığı parametreleri belirleriz. "String" sınıf adı, "args" da parametre adıdır. "[]" ise args'ın bir dizi (array) olduğunu belirtiyor.
- "System.out.println();" ile yazımızı yazdırıyoruz ve yeni satıra geçmesini sağlıyoruz.
- Yazımızı "System.out.print("Merhaba Dunya!");" ile de yazdırabilirdik fakat imleç yeni satıra geçmezdi.
- **Java Türkçe karakterleri (C# gibi) "değişken adlarında, sınıf adlarında" da kullanmamıza imkân tanır.**

Java'da yazdığımız yazılımları derlememiz için öncelikle sınıf adı ile aynı adı taşıyan dosya ismine sahip olmamız gerekmektedir. Yukarıdaki örnek yazılımı sınıf ismi olan "MerhabaDünya" ifadesini kullanıp uzantısı ile beraber "MerhabaDünya.java" ismi ile kaydedebiliriz. J2SDK veya benzer bir Java geliştirme ortamı kurulu sistemimizde yazılan uygulama aşağıdaki şekilde derlenebilir.

```
javac -encoding UTF-8 MerhabaDünya.java
```

javac, yazılan programı derleyerek ".class" uzantılı bir dosya üretir. ".class" sınıf dosyaları JVM'de çalışabilecek bytecode'lar içeren sınıf dosyalarıdır. Örnek uygulamayı çalıştırmak için:

```
java MerhabaDünya
```

yazabiliriz. Java komutu öncelikle sınıf yolunda (Bkz. Classpath) "MerhabaDünya" sınıfını arayacaktır. Bulduğu takdirde "MerhabaDünya" sınıfında "main" metodunu arayacaktır. Eğer metod bulunur ise bu metod icra edilecektir.

Daha Kapsamlı bir örnek

```
package TekCift;

import javax.swing.JOptionPane;

public class TekCift {

        /**
         * Tam sayı (int ingilizce de integer'ın kısaltmasıdır)
        private int kullanicininGirdigi;

        /**
         * Bu yapıcı fonksiyondur. TekCift objesi yaratılırken çağırılır.
         */
        public TekCift() {
        }
```

```
	/**
	 * Aşağıdaki ana fonksiyondur. Java yorumlayıcısı programı çalıştırmak için
	 * ilk bu fonksiyonu çağırır.
	 *
	 * @param argumanlar
	 *            Komut satırı argümanları (kullanılmıyor)
	 */
	public static void main(final String[] argumanlar) {
		TekCift obje = new TekCift();
		obje.pencereGoster();
	}

	/**
	 * Aşağıdaki fonksiyon "Sayı giriniz" diyaloğunu gösterir
	 */
	public void pencereGoster() {
		try {
			kullanicininGirdigi =
Integer.parseInt(JOptionPane.showInputDialog("Lütfen bir sayı giriniz."));
			hesapla();
		} catch (final NumberFormatException e) {
			System.err.println("HATA: Geçersiz bir değer girdiniz. Lütfen
sayısal bir değer giriniz.");
		}
	}

	/**
	 * Aşağıdaki fonksiyon kullanıcının girdiği değere göre çift ya da tek
	 * penceresi gösterir.
	 */
	private void hesapla() {
		if ((kullanicininGirdigi % 2) == 0) {
			JOptionPane.showMessageDialog(null, "Çift");
```

```
                } else {
                        JOptionPane.showMessageDialog(null, "Tek");
                }
        }
}
```

- package anahtar kelimesi dosyanın başında paket belirtmek için kullanılır.
- kullanıcı arayüzü gösterebilmek için javax.swing. JOptionPane kütüphanesine ihtiyacımız var. Dosyanın başındaki import cümlesi tam bu işe yarıyor.
- TekCift sınıfı kullanıcının girdiği bir tam sayıyı hafızasında tutmak için kullanıcının girdiği değişkenini tanımlıyor. Diğer sınıfların bu değişkeni görmemesi için private(özel) anahtar kelimesi kullanılmaktadır.

Java Veri Tipleri

Aşağıdaki tabloda Java programlama dilinde kullanılan veri tipleri ve değişkenler tanımlanmıştır.

Tam Sayılar	Reel Sayılar	Karakterler	Mantıksal Değerler
byte	float	char	boolean
short	double		
integer			
long			

Tablo 1.1 - Değişkenler

Detaylı olarak veri tipleri ve değişkenleri anlatmadan önce değişkenin nasıl tanımlanır bir göz atalım.

veriTipi değişkenAdi = değişkenDeğeri;

Tam Sayılar

Byte: Byte en küçük tam sayı tipidir. Büyüklüğü 8 bit kadardır ve *-128 ile +127* arası değer almaktadır. Kod geliştirirken **byte** anahtar kelimesi ile tanımlama yapılır.

byte byteDeger = 45;

Yukarıda byte veri tipiyle değeri 45 olan bir “byteDeger” değişkeni tanımlanmıştır.

Short: Büyüklüğü 16 bit olan short veri tipi *-32768 ile +32767* arasnda bir değer alabilir. Kod geliştirirken **short** anahtar kelimesi ile tanımlama yapılır.

short shortDeger = 64;

Yukarıdaki kod satırında short veri tipiyle değeri 64 olan "shorDeger" değişken isminde bir tanımlama yapılmıştır.

Integer: En çok kullanılan veri tipidir. 32 bitlik veri tipi, *-2.147.483.648 ile 2.147.483.647* arasında bir değer alabilir. Kod geliştirirken **int** anahtar kelimesi ile tanımlama yapılır.

int integerDeger = 75;

Yukarıdaki kod satırında integer veri tipiyle 75 olan, "integerDeger" değişken adıyla bir tanımlama yapılmıştır.

Long: En büyük tam sayı değeridir. 64 bitlik büyüklüğe sahiptir ve -9,223,372,036,854,775,808 ile 9,223,372,036,854,775,807 arasında bir değer alabilir. Kod geliştirilirken **long** anahtar kelimesi kullanılır. Bu veri tipi genelde bilimsel hesaplamalarda veya TC Kimlik numarası belirlemelerde kullanılabilir.

Reel Sayılar

Float: Bu veri tipi 32 bitlik büyüklüğe sahiptir ve 1.4×10^-45 ile 3.4×10^38 aralığında bir değer tanımlanabilir. Float veri tanımlanırken değişken verisinin sonunda "f" veya "F" koyularak veri tipinin float olduğu belirtilir. Aksi halde bunu double olarak algılayacağı için hata verecektir. Kod geliştirirken **float** anahtar kelimesi ile tanımlama yapılır.

Double: Bu veri tipi 64 bitlik büyüklüğe sahiptir ve 4.9×10^-324 ile 1.8×10^308 arasında bir değer tanımlanabilir. Kod geliştirirken **double** anahtar kelimesi ile tanımlama yapılır.

Karakterler

Char: Java da karakterler char veri tipi içinde saklanır. C/C++ gibi yazılım dillerinde char veri tipi 8 bitlik büyüklüğe sahipken, Java'da 16 bitlik büyüklüğe sahiptir. Bunun sebebi Java'nın Unicode karakter setini kullanıyor olmasıdır. Ayrıca Java ASCII kod yapısını da desteklemektedir. Kod geliştirirken **char** anahtar kelimesi ile tanımlama yapılır.

Mantıksal Değerler

Boolean: Java mantıksal değerlerini saklamak için boolean veri tipi kullanılmaktadır. Boolean veri tipi true ve false olmak üzere iki farklı değer alabilmektedir. boolean veri tipi genellikle koşul belirtirken veya bir döngüde kullanılabilir. Kod geliştirirken **boolean** anahtar kelimesi ile tanımlama yapılır.

Aşağıda verilen kod satırlarında bahsedilen tüm veri tiplerinin tanımlamaları yapılmıştır.

```
public class MainClass {
  public static void main(String[] args) {
    byte byteDeger = 4;
    short shortDeger = 7;
    int integerDeger = 234;
    long longDeger = 12332;

    float floatDeger = 34.5f;
    double doubleDeger = 345.2
    char charDeger1 = 65;
    char charDeger2 = 'A';
    boolean dogruDeger = true;
    boolean yanlisDeger = false;
```

Tanımlanan bütün veri tipleri aşağıdaki kodlar sayesinde ekrana yazdırılmıştır.

```
    System.out.println("Byte Değer: " + byteDeger);
    System.out.println("Short Değer: " + shortDeger);
    System.out.println("Integer Değer: " + integerDeger);
    System.out.println("Long Değer: " + longDeger);

    System.out.println("Float Deger: " + floatDeger);
    System.out.println("Double Deger: " + doubleDeger);

    System.out.println("Char Deger 1: " + charDeger1);
    System.out.println("Char Deger 2: " + charDeger2);

    System.out.println("Boolean Dogru: " + dogruDeger);
```

```
System.out.println("Boolean Yanlis: " + yanlisDeger);     } }
```

Java Nedir, Neler Yapabilirim?

Java TM platformu bilgisayar ağının varlığı da göz önüne alınarak uygulamaların/programların farklı işletim sistemleri üzerinde çalıştırılabilmesi düşüncesiyle geliştirilmiş yeni teknolojidir. Java teknolojisi kullanılarak aynı uygulama farklı ortamlarda çalıştırılabilir. Örneğin kişisel bilgisayarlarda, Macintosh bilgisayarlarda, üstelik cep telefonlarında…

Java TM platformu hem programlama dili, hem de bir ortam olarak düşünülmektedir. Programlama dili olarak, açık kodlu, nesneye yönelik (object-oriented), güvenli, sağlam, İnternet için elverişli bir teknolojidir denilebilir. Ortam olarak da orta katman (middleware) teknolojiler bulmak mümkündür.

Gerek Java programlama dili, gerekse bu dile bağlı alt teknolojiler, VBTM veya Borland DelphiTM gibi sadece belirli bir firma tarafından geliştirilmiş ürünler değillerdir. Java ve bu dile bağlı alt teknolojiler, Sun MicroSystems tarafından tanımlanmış belirtimlerden (specifications) oluşmaktadır. Bu belirtimlere sadık kalan her yazılım firması Java Sanal Makinası, kısaca JVM (Java Virtual Machine), veya Java programlama diline bağlı alt teknolojiler yazabilir (örneğin Application Server – Uygulama Sunucusu). Eğer bu belirtimlere sadık kalınmayıp standart dışı bir JVM veya Java programlama diline bağlı alt teknolojiler yazılmaya kalkışılırsa hukuki bir suç işlenmiş olur.

Peki belirtim (specifications) ne demektir? Sun MicroSystems, JVM veya Java programlama diline bağlı alt teknolojiler yazmak için belirli kurallar koymuştur; bu kurallar topluluğuna “belirtimler” denir. Örneğin biraz sonra ele alınacak olan çöp toplama sistemi (garbage collector)…

Çöp toplama sistemi daha önceden oluşturulmuş ancak şu an için kullanılmayan ve bellekte boşu boşuna yer işgal eden nesneleri belirleyerek otomatik olarak siler. Böylece Java programcısı “acaba oluşturduğum nesneyi bellekten silmiş miydim?” sorusunu sormaktan kurtulurlar, ki bu soru C++ programlama dillinde uygulama yazan kişilerin kendilerine sıkça sorması gereken bir sorudur. Şimdi bir yazılım firması hayal edelim, adının ABC yazılım firması olduğunu varsayalım. Bu firma, eğer bir JVM yazmak istiyorsa, bu çöp toplama sistemini, oluşturdukları JVM’in içerisine yerleştirmeleri gereklidir. Çünkü Sun MicroSystems‘ın belirtimlerinde, çöp toplama sistemi koşuldur! Eğer ABC firması üşenip de çöp toplama sistemini, oluşturdukları JVM’in içerisine yerleştirmezse hukuki bir suç işlemiş olur.

Şu anda en yaygın kullanılan JVM’ler, IBM ve Sun Microsystems‘ın üretilmiş olan JVM’lerdir; ayrıca, HP, Apple gibi bir çok firmanın üretmiş oldukları JVM’ler de bulunmaktadır.

Java ile Neler Yapılabilir?

Java diliyle projeler diğer programlama dillerine göre daha kolay, sağlıklı ve esnek şekilde yapılması mümkün olur. Kısaca göz atılırsa Java diliyle,

- GUI (7Grafiksel Kullanıcı Arayüzü) uygulamaları, Applet‘ler
- Veri tabanına erişimle ilgili uygulamalar
- Servlet, Jsp (Web tabanlı uygulamalar).
- Dağınık bileşenler (Distributed components) (örneğin EJB, RMI, CORBA).
- Cep telefonları, Smart kartlar için uygulamalar ve daha bir sürü benzeri uygulamalar yazmak mümkündür.

Java Eclipse

Eclipse, açık kaynak kodlu ve özgür bir tümleşik geliştirme ortamıdır (IDE). Ana odak noktası Java ve Java ile ilişkili teknolojiler olsa da, esnek yapısı sayesinde C ve Python gibi farklı diller için de kullanılmaktadır.

2001 yılında IBM tarafından başlatılan proje, Java'nın ana grafik sistemi olan Swing yerine bulunduğu platformda bulunan özellikleri doğrudan kullanan SWT'yi kullanarak Java dünyasında tartışmalara yol açmıştır. Hızlı arayüzü, şık görünümü ve çok kuvvetli özellikleriyle kısa zamanda Java geliştiricileri arasında en popüler geliştirme ortamı oldu. 2005 yılında Eclipse Projesi'nin yönetimi Eclipse Vakfı’na bırakılmıştır.

Android geliştirme ortamının önemli bir bileşeni olan Eclipse'in içinde yazılan programları denemek için öykünücü kurulabilmektedir. Eclipse ortamı, sunulan eklentilerle işlevleri geliştirilerek birçok alanda kullanılabilmektedir. Proje bu yazılım geliştirme platformu üzerinde geliştirilmiştir.

Diğer yazılım geliştirme platformları

- Anjuta
- Kdevelop
- Netbeans
- JCreator
- Intellij Idea
- Microsoft Visual Studio

Java Eclipse arayüzü ve hakkında bilgiler

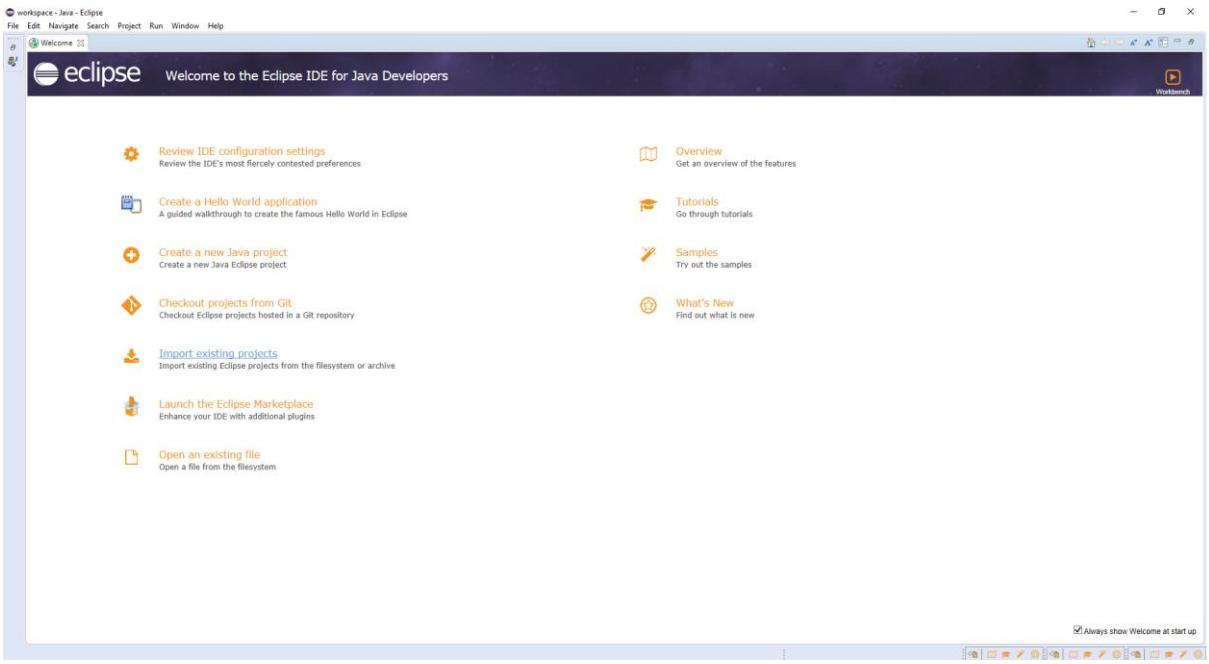
Daha önceki başlık altında da söylenildiği üzere proje bu platform üzerinde yazılmış ve geliştirilmiştir. Bu başlık altında ise Java Eclipse arayüzü hakkında anlatımlar yapılmıştır.

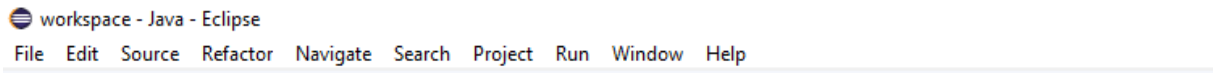
Java Eclipse bir bilgisayara kurulması gerektiği zaman öncelikle kullanılacak bilgisayarda JDK(Java Development Kit)'nın kurulu olması gerekmektedir.

JDK indirme linkine *buradan* ulaşabilirsiniz. Girilen sitede karşınıza 2 adet seçenek sunulmaktadır. Java Platform(JDK) 8u131 olan kısma tıklanmalıdır. Java SE Development Kit 8u131 olan kısma gelip "Accept License Agreement" butonu seçili iken aşağıda işletim sistemine uygun olan sürüm indirilir ve ilk aşama tamamlanmış olur.

Java Eclipse en son sürüm olarak Eclipse Neon kullanmaktadır. Eclipse indirme linkine *buradan* ulaşabilirsiniz. İndirme işlemleri tamamlandıktan sonra Java Eclipse sorunsuz bir şekilde çalışacaktır.

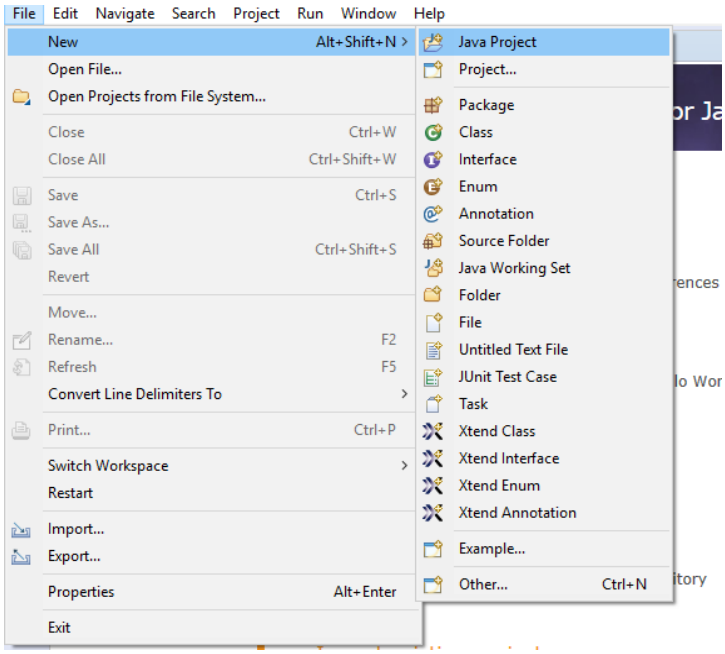
İlk açıldığında kullanıcı böyle bir ekran ile karşılaşacaktır. Kullanıcı sağda bulunan "Workbench" yazan kısma tıklar ve yeni bir sayfa açılır.





Şekil 1.1 – Java Arayüzü Kategoriler

Bu kısımda programın başlıca kategorileri bulunmaktadır. Bunlar yazılım geliştirme esnasında kullanılan araçlardır. File, Edit, Source, Refactor, Navigate, Search, Run, Window, Help olmak üzere 9 adet kategori bulunmaktadır.



Şekil 1.2 – File Sekmesi

File sekmesi başlığı altında dosya işlemleri yapılmaktadır.

New: New sekmesinde yeni bir projenin açılması veya varolan bir projenin üzerine yeni bir sınıf, paket, arayüz kısımlarının tanılması sağlanmaktadır.

Open File… : Bu sekme bilgisayarınızda herhangi bir yerinde varolan bir projeyi küçük bir browser yardımıyla açmaktadır.

Close: Bulunduğunuz bölümü kapatır.

Close All: Projedeki bütün bölümleri kapatır.

Save: Projeyi kaydeder.

Save As…: Farklı kaydet seçeneği ile projeyi belirlenen yere kaydetme işlemi yapar.

Save All: Projede yapılan bütün değişiklikleri kaydeder.

Revert: Yapılan işlemleri geri alır.

Move: Varsayılan projeyi başka bir klasöre taşır.

Rename: Belirlenen projenin ismi değiştirilir.

Refresh: Projeyi yeniler.

Print: Projeyi yazdırmak için kullanılır.

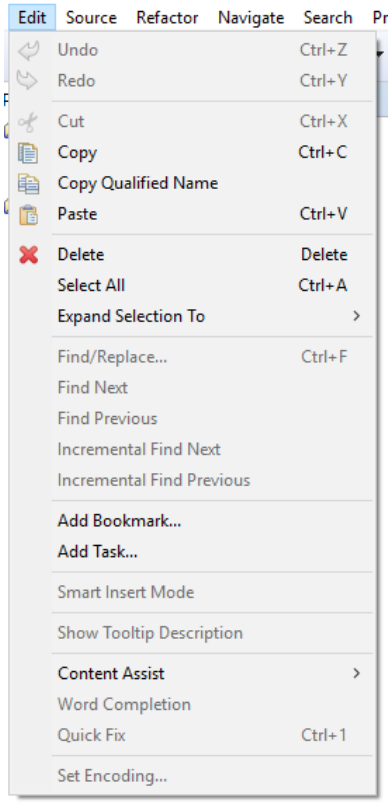
Switch Workspace: Proje Workspace'lerini değiştirmek için kullanılır.

Restart: Eclipse programını yeniden başlatır.

Import ve Export: Bir projeyi ve sınıfı uygulamadan export veya uygulmaya import etmek için kullanılır.

Properties: Eclipse özelliklerini ayarlamak için kullanılır.

Exit: Eclipse programını kapatmak için kullanılır.



Şekil 1.3 – Edit sekmesi

Bu sekme altında yapılan işlemleri tekrar düzenlemek amacıyla kullanılır. Bazılarını aşağıda açıklanmıştır.

Undo: Son yapılan işlemi geri alır.

Redo: Son yapılan işlemi ileri alır.

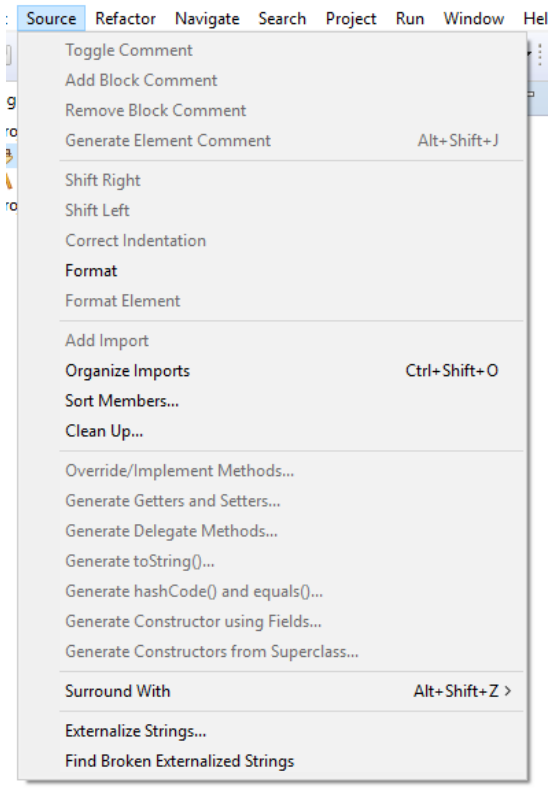
Cut: Seçili bölgeyi keser.

Copy: Seçili bölgeyi kopyalar.

Paste: Kopyalanan bölgeleri yapıştırır.

Delete: Seçili bölgeyi siler.

Select All: Bütün kod satırlarını seçer.



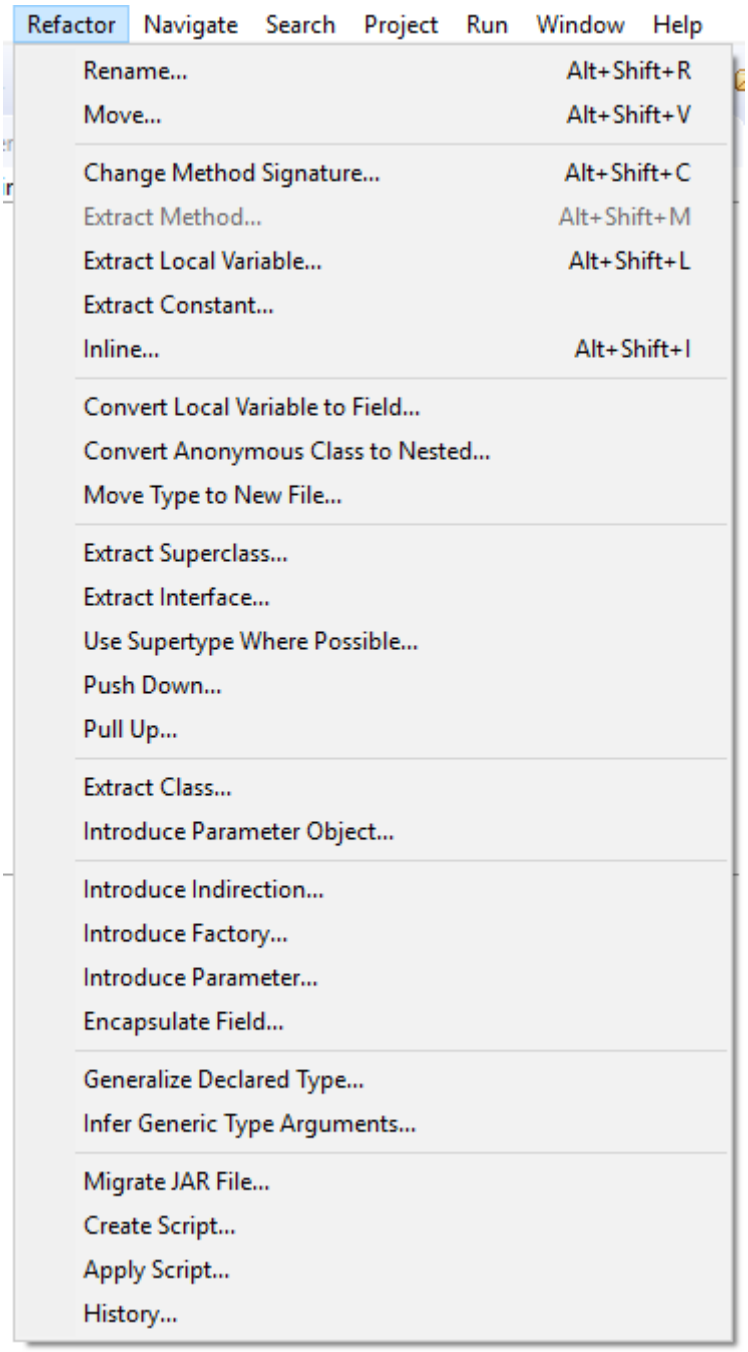
Şekil 1.4 – Source sekmesi

Source sekmesinde seçili olan satırı açıklama satırına çevirme, açıklama satırlarını veya Override/Implement metotlar ekleme, sınıflar için getter ve setter oluşturma gibi işlemler yapılır.

Toggle Comment: Üzerinde bulunan satırı açıklama satırı yapar veya açıklama satırı olan bir satırı normal kod satırına dönüştürür.

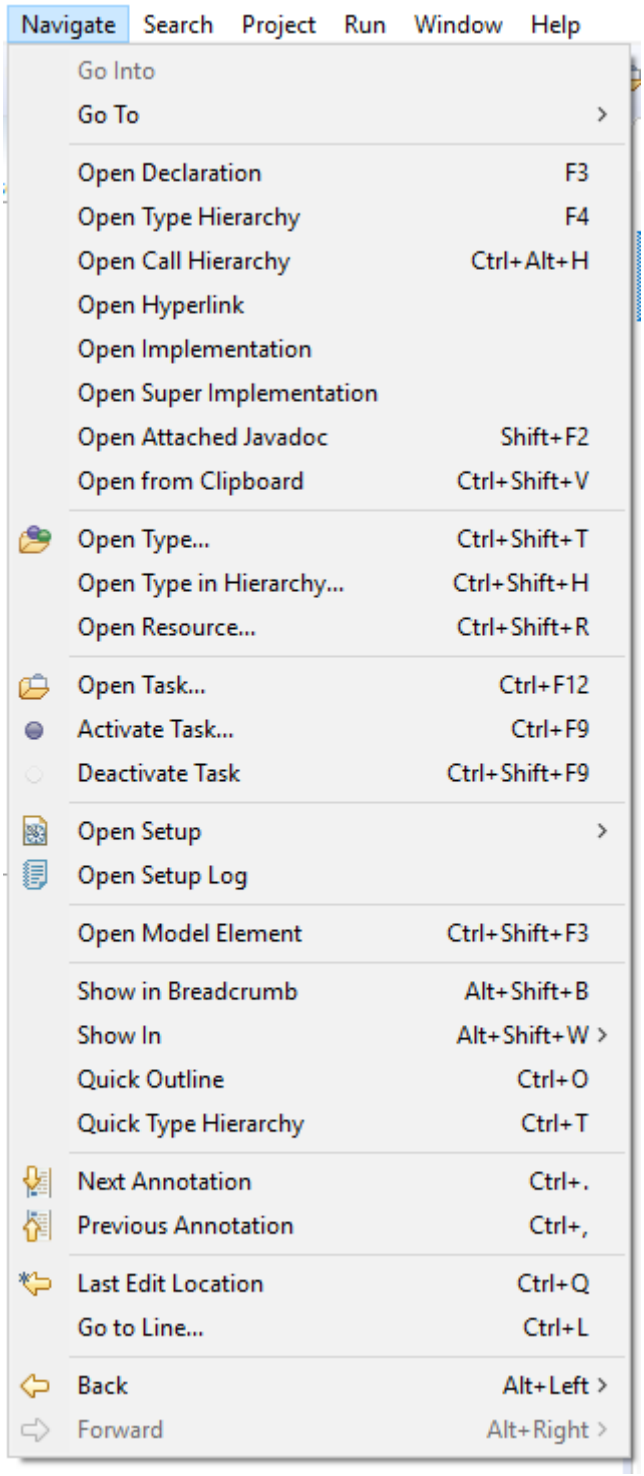
Add Block Comment: Seçili olan kod satırlarını açıklama bloğuna dönüştürür.

Remove Comment: Seçili olan açıklama bloklarını kod satırlarına dönüştürür.



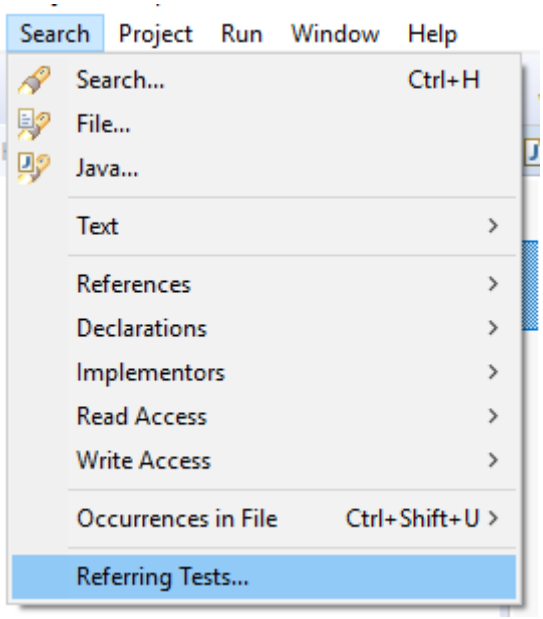
Şekil 1.5 – Refactor Sekmesi

Bu sekmede ise bazı dosya işlemleri yapılabiliyor. Örneğin; projenin yeniden isimlendirilebilmesi veya projenin başka bir klasöre taşınması gibi. Aynı zamanda sınıfların üzerine başka sınıf ve arayüzlerin tanıtılması gibi birçok farklı işlemde bu sekme altında yer alıyor.



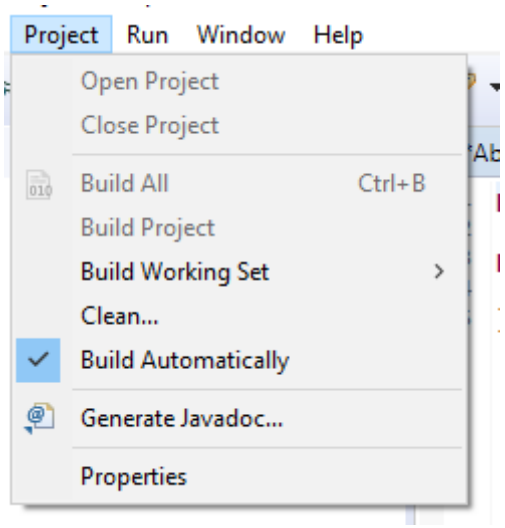
Şekil 1.6 – Navigate sekmesi

Navigate sekmesi, program içinde kısaca seyahat etmenizi sağlıyor da denebilir. Projenin herhangi bir kısmına kolayca gitmek veya projede bulunan bir linki açmak gibi birçok farklı işlem burada gerçekleştirilebiliyor.



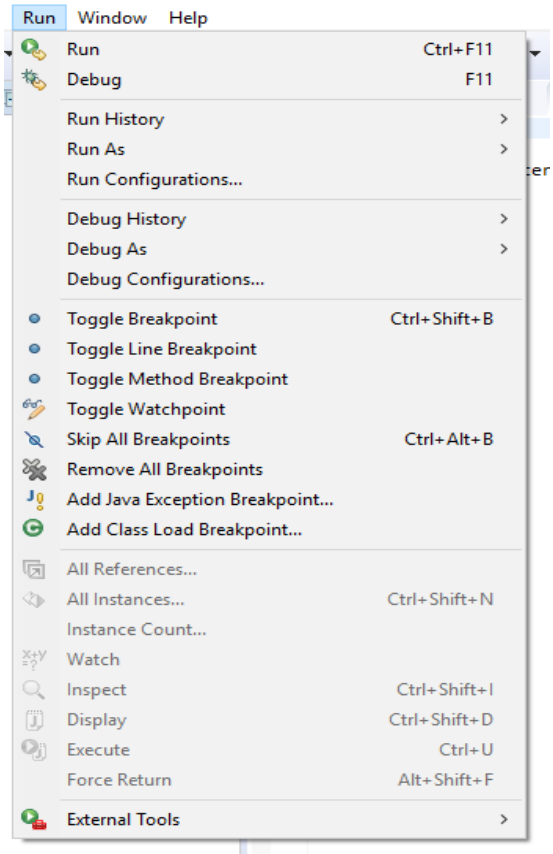
Şekil 1.7 – Search sekmesi

Bu sekme içerisinde uygulamadaki herhangi bir yazıyı, dosyayı veya java dosyasını(sınıf,paket vb.) bulmaya yardımcı araçlar bulunmaktadır.



Şekil 1.8 – Project sekmesi

Bu sekmede projeyi açma veya kapatma işlemleri yapılıyor. Projeyi Build ederek yazılan kodların çalışabilirliğinin kontrol edilmesi de mümkündür. Ayrıca proje özelliklerinede alttaki **Properties** seçeneğinden erişilebilmektedir.



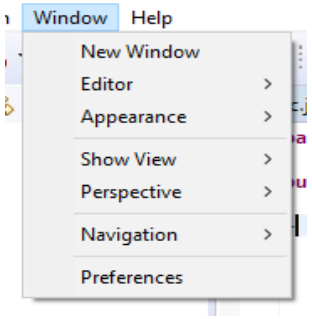
Şekil 1.9 – Run sekmesi

Bu sekmede projeyi çalıştırmak için gerekli her araç bulunuyor. Herhangi bir uygulamanın geliştirilme sürecindeyken deneme amaçlı çalıştırılması gerektiği durumlarda bu sekmeden **Run** seçeneği seçilerek uygulama başlatılabilir.

Run: Üzerinde bulunan uygulamayı çalıştırır.

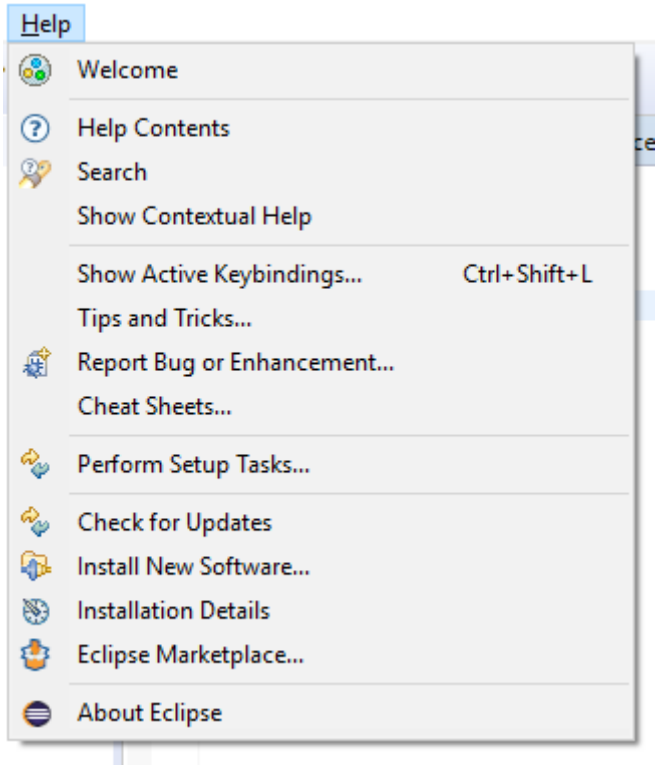
Run History: Önceden yapılmış bir uygulamayı çalıştırır.

Run as: Birden fazla proje bulunması durumunda kullanıcı çalıştırmak istediği projeyi seçerek çalıştırma işlemi başlanır.



Şekil 1.10 – Window sekmesi

Window sekmesinden yeni bir pencere açabilir, bulunduğumuz kısmı klonlayıp aynısından bir adet daha oluşturulabilir veya uygulama yazarken gerekli View ekranlarının (Package Explorer, Outline, Problems vb) gelmesi sağlanabilmektedir.



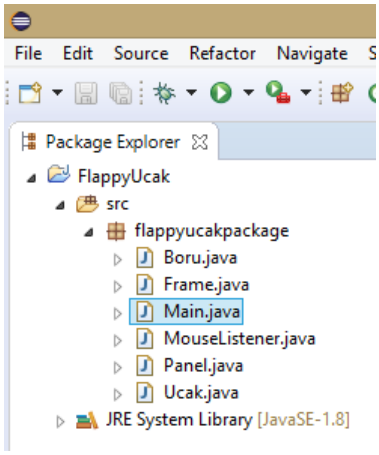
Şekil 1.11 – Help sekmesi

Bu sekmede Eclipse programı hakkında bilgiler ve birkaç yardımcı olacak araç kullanılabilir. Örneğin, **Welcome** butonuna tıklandığında direkt yeni bir proje açıklacaktır. Ayrıca Eclipse Marketplace isimli uygulama pazarından uygulama için gerek olan ekstra araçlarda bu kısımda bulunmaktadır.

BÖLÜM 3

ÇALIŞMALAR

İlk önce proje oluşturuldu daha sonra paket oluşturuldu , paketin içine sınıflar oluşturuldu.



Şekil 2.1 – Sınıfların Oluşturulması

Frame sınıfı JFrame olarak kalıtıldı sonra Frame'e ait bilgiler girildi.

```
public Frame()
{
    this.setVisible(true);//Görünürlüğü aktif yapıyor
    this.setLocation(500, 100);//Çerçevemizin başlangıç konumunu belirledik
    this.setSize(582,796);//Çerçeve boyutunu belirledik
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);//Kapatma operatörü atadık
```

Panel sınıfı JPanel olarak kalıtıldı. Panel sınıfına paintComponent metodu oluşturuldu. MouseListener sınıfı MouseListenere implement edildi ,boolean veri tipinde clicked değişkeni tanımlandı değeri false olarak atandı. MousePressed olayı gerçekleştiğinde clicked değeri true olarak atandı sonra Frame sınıfının bilgilerine MouseListener sınıfı eklendi ve Trame metodu çağrıldı.

Trame metodunun içine oyun döngüsü oluşturuldu ve oyun döngüsünün içine panel.repaint() kodu ile panel sınıfına yeniden çizildi.

```
public void Trame() {

    while(true)//oyun döngüsü
    {


        panel.repaint();//paneli yeniden çizmeye yarıyor
        try {
            Thread.sleep(13);
        } catch (InterruptedException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

Şekil 2.2 – Paneli yeniden çizme

Charge metodu oluşturuldu, oyun klasörü içindeki score.txt'den en iyi skoru değerini alıp eniyi değişkenine atıyor

```
public void Charge()//En iyi skoru score txteden okuyor
{

    try {
        File f=new File("score.txt");
        BufferedReader reader=new BufferedReader(new FileReader(f));
        String line=reader.readLine();
        eniyi=Integer.parseInt(line);
        reader.close();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

Şekil 2.3 – En iyi skorun yazılması

Kaydet metodu eniyi değeri score text dosyasına atması için oluşturuldu.

```
public void Kaydet()//En iyi skoru score txtsine kayıt ediyor
{

    try {
        File f=new File("score.txt");
        BufferedWriter writer=new BufferedWriter(new FileWriter(f));
        writer.write(Integer.toString(eniyi));

        writer.flush();
        writer.close();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

Şekil 2.4 – Kaydetme işlemleri

Panel sınıfına Image dizisi oluşturuldu, chargeTexture metodu oluşturuldu ve resimler bu metodun içinden image dizisine aktarıldı.

```
public void chargeTexture()//Img dizisine resimler aktarıldı
{
    Img[0]=new ImageIcon("assets/0.png").getImage();
    Img[1]=new ImageIcon("assets/1.png").getImage();
    Img[2]=new ImageIcon("assets/2.png").getImage();
    Img[3]=new ImageIcon("assets/3.png").getImage();
    Img[4]=new ImageIcon("assets/4.png").getImage();
    Img[5]=new ImageIcon("assets/5.png").getImage();
    Img[6]=new ImageIcon("assets/6.png").getImage();
    Img[7]=new ImageIcon("assets/7.png").getImage();
    Img[8]=new ImageIcon("assets/8.png").getImage();
    Img[9]=new ImageIcon("assets/9.png").getImage();

    Img[10]=new ImageIcon("assets/arkaplan.png").getImage();

    Img[11]=new ImageIcon("assets/ucak1.png").getImage();
    Img[12]=new ImageIcon("assets/ucak2.png").getImage();
    Img[13]=new ImageIcon("assets/ucak3.png").getImage();

    Img[14]=new ImageIcon("assets/tahta.png").getImage();
    Img[15]=new ImageIcon("assets/zemin.png").getImage();
    Img[16]=new ImageIcon("assets/oyunbitti.png").getImage();
    Img[17]=new ImageIcon("assets/boruB.png").getImage();
    Img[18]=new ImageIcon("assets/boruH.png").getImage();
    Img[19]=new ImageIcon("assets/yeni.png").getImage();

    Img[20]=new ImageIcon("assets/hazir.png").getImage();
    Img[21]=new ImageIcon("assets/tikla.png").getImage();
```

Şekil 2.5 – Resimlerin diziye aktarılması

Çerçeveye resimleri çizeceğimiz paintComponent metoduna gerekli resimler çizildi.

```
public void paintComponent(Graphics g)//ekrana çizim işlemleri bu methodda yapılıyor
{
    Frame.MousePos[0]=MouseInfo.getPointerInfo().getLocation().x+getLocationOnScreen().x;//mousenin görüntü üzerindeki x
    Frame.MousePos[0]=MouseInfo.getPointerInfo().getLocation().y+getLocationOnScreen().y;//mousenin görüntü üzerindeki y

Graphics2D g2d=(Graphics2D)g;//g değişkeni 2d grafik olarak alınıp g2d değişkenine aktarılıyor

g.drawImage(Img[10], 0, 0, null);// arka plan resmini çiziyoruz
g.drawImage(Img[15], -Frame.groundCycle, 640, null);//zemin resmimizi çiziyoruz x değerini zemin hareket etsin diye
//groundCycle değişkenini alıyoruz - olarak
}
```

Şekil 2.6 – Çerçeveye gerekli resimler çizildi

Ucak sınıfının değişkenleri tanımlandı.

```
public class Ucak {

    public static double vy=0;
    public static double y=400;//uçağın y değeri


    public static boolean basladi=false;

    public static int cycle_speed=10;//ucak resimlerinin animasyon hızı
    public static int cycle_pos=0;
    public static int tex=0;//uçak resminin değişmesi için tanımlanan değişken
```

Şekil 2.7 – Ucak sınıfı değişkenleri tanımlandı

Ucak sınıfına Guncelle metodu yazıldı.

```
public void Guncelle(){

    cycle_speed=(int)(14/9*vy+4/3);//ucak animasyon hızı belirliyoruz

    if(vy<3)//vy değeri 3den küçükse animasyon hızı 6 değerini alıyor
    {
        cycle_speed=6;
    }
    if(vy<12)//vy değeri 12den küçükse animasyon hızı 20 değerini alıyor
    {
        cycle_speed=20;
    }

    if(vy<8)//vy değeri 8den cycle_pos 1 artıyor
    {
        cycle_pos++;
        if(cycle_pos>cycle_speed)//cycle_pos büyükse cycle_speedden cycle_pos 0 değerini alıyor tex 1 artıyor
            {
            cycle_pos=0;
            tex++;
            if(tex>2)tex=0;//tex değeri 2 den büyükse tex 0 değerini alıyor
            }
    }
        else //vy değeri 8den küçük değilse
        {
            tex=0;
        }
```

Şekil 2.8 – Guncelle metodunun yazılması

Uçak çizildi ve oyun döngüsüne Ucak.Guncelle() metodu çağrıldı.

```
if(basladi)//ucak basladi değeri true ise
{
    vy+=0.5;//vy değeri 0.5 arttı
    y+=vy;//y değeri vy değeri kadar arttı
    if(y > 630 || y < 10)//Uçağın y değeri 630 dan büyükse veya 10dan küçükse
    {
        if(Frame.skor > Frame.eniyi)//skor değeri eniyi skordan büyükse en iyi skor score.txtine kaydedilecek
        {
            Frame.yeni_eniyi=true;//yeni en iyi değeri true oldu
            Frame.eniyi=Frame.skor;//eniyi değere skor atandı
            Frame.Kaydet();//ve score.txt ye kaydedildi
        }
        Frame.menu=true;//menu aktif hale gelicek
    }
    if(y>630)
    {
        y=630;//y değeri 630dan büyükse y'ye 630 değeri atandı
    }
}
else
{
    y=Math.sin(Frame.frame_real*Math.PI/15)/3;//başlangıçta uçağımız yükselip inmesi kodu yazıldı
}
```

Şekil 2.9 – Guncelle metodunun çağrılması

```
g2d.translate(160, Ucak.y);//bu kod ile çeviriyoruz
g.drawImage(Img[11+Ucak.tex], -34, -24, null);//sonra uçağımızı çiziyoruz
g2d.translate(-160, -Ucak.y);
```

Şekil 2.10 – Ucak ekrana çizildi.

Frame sınıfının Theme metoduna mouse tıklandığında uçağın zıplaması sağlandı ve tıklandıktan sonra tıklanma değeri yeniden false olarak atandı.

```
if(ml.clicked)//mouse tıklandığın çalışacaklar
{
    if(!Ucak.basladi)Ucak.basladi=true;//Ucak sınıfının basladi değişkeni true değilse true olarak değiştirildi
    Ucak.vy=-8;//Ucak yukarıya doğru zıplıyor ve ucağın y değeri düşüyor

}
```

Şekil 2.11 – Uçağın zıplaması

Sonra Boru sınıfına ait değişkenler tanımlandı.

```
int x=0;//Boru için x değişkeni
int yukseklik=0;//Boru için yükseklik değişkeni
boolean gecti=false;//Boru için geçildi değişkeni
```

Şekil 2.12 – Boru değişkenlerinin tanımlanması

Boru sınıfına metodlar yazıldı

```
public Boru(int _x,int y)//
{
    x=_x;
    yukseklik=y;
}

public int getX()//x değerini almak için yazıldı
{
    return x;
}
public int getYukseklik()//y değerini almak için yazıldı
{
    return yukseklik;
}

public void X()//x değeri -2 düşücek ve uçağa yaklaşıcak
//x değeri 110 olduğunda gecti değişkenin değeri true olacak
{
    x-=2;
    if(x==110) gecti=true;
}
public boolean getGecti()//Uçak boruyu gecti ise gecti değeri false olacak ve true değeri dönecek gecmedi ise false
//değeri dönecek
{
    if(gecti)
    {
        gecti=false;
        return true;
    }
    return false;
}
```

Şekil 2.13 – Boru sınıfına metodların yazılması

```
public boolean Carpisma()
{
    //Çarpışma kontrol ediliyor
    if(x < 180 && x+102 > 140 && (Ucak.y+20 > yukseklik+70 || Ucak.y-20 < yukseklik-70)){

        return true;
    }
    return false;
}
```

Şekil 2.14 – Çarpışmaya ait metodun yazılması

Sonra Frame sınıfında borular dizisi oluşturuldu. Oyun döngüsü içine boru dizisinde ki borulara x ve y değerleri verildi , döngü oluşturuldu, boruIs değeri true ise borular[i]'nin X metodu çağrıldı borular[i]'nin getGecti metodu true ise skor 1 artırıldı ve borular[i]'nin x değeri -110'dan küçükse boruIs[i] değeri false olarak değiştirildi.

```
if(frame==200)//frame değeri 200 olursa
{
    frame=0;//frame değerini 0a esitle
    //Boru
    int i=0;
    while(boruIs[i])i++;
    borular[i]=new Boru(580, (int)(Math.random()*390+125));
    //boru x değeri 580 veriliyor ve random yükseklik değeri atanıyor.

    boruIs[i]=true;//boru boolean değeri true oluyor
}
for(int i=0; i<10; i++)
{
    if(boruIs[i])//boru true ise
    {
        borular[i].X();//borunun X methodu çağrılıyor
        if(borular[i].getGecti())skor++;//boru gecti değeri true ise skor 1 artıyor
        if(borular[i].getX() < 110)boruIs[i]=false;//boruis değeri false olarak değiştiriliyor
    }
}
```

Şekil 2.15 – Oyun içinde boruların görevleri

Sonra Panel sınıfına borular çizildi.

```
for(int i=0; i<10; i++)
{
    if(Frame.boruIs[i])//boru true ise ekrana cizilecek
    {
        g.drawImage(Img[17], Frame.borular[i].getX(), Frame.borular[i].getYukseklik()+70, null);
        g.drawImage(Img[18], Frame.borular[i].getX(), Frame.borular[i].getYukseklik()-562, null);
    }
}
```

Şekil 2.16 – Boruların çizilmesi

Ucak sınıfının Guncelle metodunun içine boru ile çarpışma kontrol edildi.

```
if(basladi)//ucak basladi değeri true ise
{
    vy+=0.5;//vy değeri 0.5 arttı
    y+=vy;//y değeri vy değeri kadar arttı

    boolean carpisma=false;
    for(int i=0; i<10; i++)
    {
        if(Frame.boruIs[i])//boru true ise kontrol edicek
        {
            //Eğer carpısma true ise
            //ucak sınıfında ki carpisma değişkenin değeri true olarak değiştirildi.

            if(Frame.borular[i].Carpisma())carpisma=true;

        }
    }
```

Şekil 2.17 – Boru ile çarpışmanın kontrol edilmesi

Panel sınıfında ekrana skor çizildi.

```
String skor=Integer.toString(Frame.skor+8);
if(!Frame.menu)
{
    for(int i=0;i<skor.length();i++)
    {

        g.drawImage(Img[Integer.parseInt(String.valueOf(skor.charAt(i)))], 280, 54, null);

    }
}
```

Şekil 2.18 – Skorun çizilmesi

Sonra hazir resmi ve tıklama resmi ekrana çizildi.

```
if(!Ucak.basladi)//Ucak basladi değeri false ise
{
    g.drawImage(Img[20], 114, 150, null);
    g.drawImage(Img[21], 250, 250, null);
}
```

Şekil 2.19 – Başlangıç ekranına resimlerin çizilmesi

Oyun bittiğinde ise game over yazısı ve tahta çizildi eniyi_skor değeri true ise new yazısıda tahtaya çizildi.

```
else//menu değeri true ise
{
    g.drawImage(Img[16], 119, 150, null);//game over yazısı çizildi
    g.drawImage(Img[14], 141, 250, null);//skor tahtası çizildi
    if(Frame.yeni_eniyi)g.drawImage(Img[19], 273, 362, null);//new yazısı çizildi
}
```

Şekil 2.20 – Bitiş ekranına resimlerin çizilmesi

Restart resmini çizdik Frame sınıfındaki MousePos dizisini burada kullanıcaz oyunumuz yeniden başlasın diye, restart resmine tıklandığında oyun en başına dönmektedir.

```
else//Restart resmine tıklayınca oyun yeniden başlıyacak
{
    if(ml.clicked)//mouse tıklandıysa
    {
        //Panel sınıfından aldığımız mouse pozisyonlarını kullanıyoruz restart resminin
        //x ve y koordinatlarındaki yerini bildiğimiz için değerleri yapıyoruz
        if(MousePos[0] > 213 && MousePos[0] < 369 && MousePos[1] > 500 && MousePos[1]< 556)
        {
            Ucak.basladi=false;
            Ucak.y=400;
            skor=0;
            for(int i=0; i<10; i++)
            {
                if(boruIs[i])boruIs[i]= false;
            }
            frame=0;
            yeni_eniyi=false;
            menu=false;


        }
    }
}
```

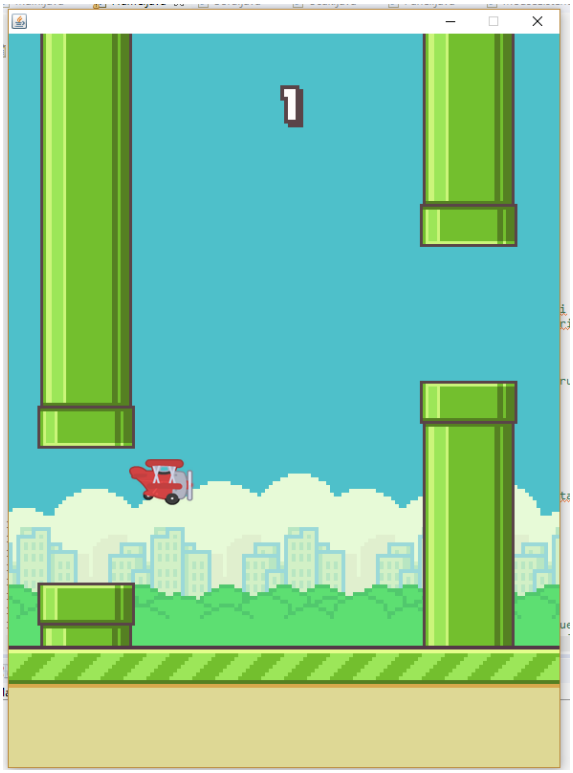
Şekil 2.21 – Yeniden başlatma metodunun yazılması

Yapılan işlemler bu kadardır.

BÖLÜM 4

SONUÇLAR

Yapılan projede elde edilen sonuç ekranı aşağıdaki resimde verilmiştir.



Şekil 3.1 – Oyun ekranı

Yapılan çalışmaların faydaları, Java görselinde ilerleme kaydettik. Jframe ve Jpanel sınıfları hakkında ilerleme kaydettik.

Uygulamanın yapım esnasında, resime tıklama anında bir çok sorun ile karşılaşıldı, boruların birbirine olan mesafesiyle ilgili bazı sorunlar ile karşılaşıldı.

Tavsiye olarak bu uygulamaya benzer bir uygulama geliştirmek istenildiği durumlarda, öncelik olarak Jframe ve Jpanel sınıfları hakkında araştırma yapılmalıdır. Pencereye resim çizmek ile ilgili araştırmalar yapılmalıdır.

KAYNAKÇA

Java Frame Nedir ?, Nasıl yapılır ? : http://www.ugurkizmaz.com/YazilimMakale-1633-Java-Frame-Nedir--Nasil-Yapilir-.aspx, 27 Mart 2012

Java Programlama: Frame ve Panel ekleme işlemleri: http://www.serefakyuz.com/2011/06/java-programlama-gui-bolum1-frame-ve-panel-ekleme-islemleri.html, 8 Haziran 2011

Java Görsel Programlama hakkında bilgiler: http://members.comu.edu.tr/msahin/courses/gorsel_prog_files/ders07.pdf

Java DrawImage nasıl yazıldığı ve ne olduğu hakkında bilgiler: https://docs.oracle.com/javase/tutorial/2d/images/drawimage.html

Java 2D grafik ve DrawImage hakkında bilgiler: http://www.java2s.com/Tutorial/Java/0261__2D-Graphics/DrawImage.htm

Java MouseListener nedir ve nasıl yazıldığı hakkında bilgiler: https://docs.oracle.com/javase/tutorial/uiswing/events/mouselistener.html

ÖZGEÇMİŞ

Adı Soyadı: Ferhat Çark

Doğum Yeri ve Tarihi: Antalya – 25/10/1997

E-Posta: ferhatcark@outlook.com

Staj ve İş Deneyimleri: Küçükçekmece Belediyesi/Bilgi İşlem Bölümü – Lise Stajı

Gerçekleştirdiği Projeler: Henüz gerçekleştirilmiş bir projesi bulunmamaktadır.

Adı Soyadı: Mücahit Mustafa Perver

Doğum Yeri ve Tarihi: Bakırköy – 26/03/1997

E-Posta: mustiperver@gmail.com

Staj ve İş Deneyimleri: WebHome Şirketi – Lise Stajı

Gerçekleştirdiği Projeler: Henüz gerçekleştirilmiş bir projesi bulunmamaktadır.