KARADENİZ TEKNİK ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

YILDIZ SAVAŞLARI

MÜHENDİSLİK TASARIMI PROJESİ

MUSTAFA FURKAN KARAKULAK

2017-2018 GÜZ DÖNEMİ

KARADENİZ TEKNİK ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

YILDIZ SAVAŞLARI

MÜHENDİSLİK TASARIMI PROJESİ

MUSTAFA FURKAN KARAKULAK

2017-2018 GÜZ DÖNEMİ

IEEE Etik Kuralları
IEEE Code of Ethics

Mesleğime karşı şahsi sorumluluğumu kabul ederek, hizmet ettiğim toplumlara ve üyelerine en yüksek etik ve mesleki davranışta bulunmaya söz verdiğimi ve aşağıdaki etik kurallarını kabul ettiğimi ifade ederim:

1. Kamu güvenliği, sağlığı ve refahı ile uyumlu kararlar vermenin sorumluluğunu kabul etmek ve kamu veya çevreyi tehdit edebilecek faktörleri derhal açıklamak;
2. Mümkün olabilecek çıkar çatışması, ister gerçekten var olması isterse sadece algı olması, durumlarından kaçınmak. Çıkar çatışması olması durumunda, etkilenen taraflara durumu bildirmek;
3. Mevcut verilere dayalı tahminlerde ve fikir beyan etmelerde gerçekçi ve dürüst olmak;
4. Her türlü rüşveti reddetmek;
5. Mütenasip uygulamalarını ve muhtemel sonuçlarını gözeterek teknoloji anlayışını geliştirmek;
6. Teknik yeterliliklerimizi sürdürmek ve geliştirmek, yeterli eğitim veya tecrübe olması veya işin zorluk sınırları ifade edilmesi durumunda ancak başkaları için teknolojik sorumlulukları üstlenmek;
7. Teknik bir çalışma hakkında yansız bir eleştiri için uğraşmak, eleştiriyi kabul etmek ve eleştiriyi yapmak; hatları kabul etmek ve düzeltmek, diğer katkı sunanların emeklerini ifade etmek;
8. Bütün kişilere adilane davranmak; ırk, din, cinsiyet, yaş, milliyet, cinsi tercih, cinsiyet kimliği,veya cinsiyet ifadesi üzerinden ayırımcılık yapma durumuna girişmemek;
9. Yanlış veya kötü amaçlı eylemler sonucu kimsenin yaralanması, mülklerinin zarar görmesi, itibarlarının veya istihdamlarının zedelenmesi durumlarının oluşmasından kaçınmak;
10. Meslektaşlara ve yardımcı personele mesleki gelişimlerinde yardımcı olmak ve onları desteklemek.

IEEE Yönetim Kurulu tarafından Ağustos 1990'da onaylanmıştır.

ÖNSÖZ

“Yıldız Savaşları Oyunu” isimli bu çalışma, Karadeniz Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü’nde güz dönemi tasarım projesi olarak hazırlanmıştır.

Projede engin bilgileri ve tecrübesi ile tüm süreç boyunca desteğini esirgemeyen danışmanım Prof. Dr. Cemal Köse 'ye tüm kalbimle şükranlarımı sunmayı bir borç bilirim.

Mustafa Furkan Karakulak
Trabzon 2017

İÇİNDEKİLER

ÖZET

Yıldız Savaşları Oyunu, proje her kesimden insana hitap eden ve oynayabileceği şeklinde tasarlanmıştır.

Yıldız Savaşları Oyunu, kabaca tarif etmek gerekirse, projenin amacı android de çoklu oynanabilen ve fiziksel koşulların geçerli olduğu belli bir hikayesi olan yapay zekanın da dahil olduğu bir oyun yapmaktır. Oyun uzay da geçecektir ve uzayın koşulları oyun içeresinde geçerli olacaktır.

Genel olarak Yıldız savaşları oyunu, kullanıcıya birden fazla seçenek sunmaktadır. Hikâye modu, online oyun ve yapay zekâ ile birlikte oynama modudur.

Bu çalışmada çoklu oynanabilen 'Yıldız Savaşları Oyunu' gerçekleştirilmiştir. Proje Android Studio'da gerçekleştirilmiştir. Projenin amacı android de çoklu oynanabilen oyun yapmaktır. Oyun uzayda geçecek ve uzayın fiziksel koşulları oyun içerisinde gerçeklenmiştir. Oyun çoklu ve sisteme karşı oynanabilen ve oyunun tekli oynanışında kendisine özel hikayesi vardır. Kullanıcı yapay zekâ sayesinde sistemle birlikte oyunu oynayabilecektir. Ağ programlama kısmı ile kullanıcı ağ ortamındaki diğer kullanıcıları görebilecek ve bağlantı sağlayıp birlikte oynayabilmektedir.

1. GENEL BİLGİLER

1.1. Giriş

Bu bölüm, projenin tasarım süreci ve okumakta olduğumuz bu tezin yazımıyla alakalı genel bir açıklama ve tanım yapmak maksadıyla hazırlanmıştır.

Her yaştan insanın vazgeçilmez tutkusu olan oyun, teknolojinin gelişmesiyle beraber birçok yeniliğe ev sahipliği yapmış, ilk halinden tamamen değişik bir yüzle karşımıza çıkmıştır. Oyun programlama, talebin oluşturduğu beklenti yönündeki değişimlerle, kısa sürede büyük bir atağa geçerek sektörü eline almıştır. Bu proje her gün elimize aldığımız telefonlarımız içerisinde yerini alan bu teknolojinin, ürünü olan 'Yıldız Savaşları Oyunu" gerçeklenmiştir.

Öncelikle bilmelisiniz ki oyun yapımı bütünüyle bir sanattır. Elektronik ortamda çalışan pek çok kişinin oluşturduğu devasa bir kompozisyondur. Bu kompozisyonu oluşturan her ast bireye ise sanatçı / mühendis denir. Çalışanlar oyun yapımında uzman oldukları dallara ve bu dallar içindeki konulara göre gruplara ayrılırlar. Projede çalışan her kişi kendi dalı içinde üzerine özellikle yoğunlaştığı bir konuda çalışır.

Yıldız Savaşları Oyunu amacından bahsedilmesi gerekirse, kullanıcının zamanını zevkle geçirebileceği şekilde tasarlanmıştır.

Problemi iyi bir şekilde analiz etmek ve tanımlamak gerekiyor. Projenin başarılı bir sonuç elde edilmesi için hangi çıktıların elde edilmesine karar vermek gerekmektedir. İstenilen proje android tabanlı çoklu oynanabilen bir uzay savaş oyunudur. Belirtilen hedefler doğrultusunda ilk önce android ortamında oyun nasıl yazılır hangi araçlar kullanılır, bunların doğru şekilde karar verilmesi gerekiyor. Konu ile alakalı oyunların oynanıp bilgi edinilip, tecrübe amaçlı uygulamalar yapılması gerekmektedir. Oyunumuza ilk girildiği zaman karşımızda ki ara yüzde seçenekler vardır. Bu seçenekler, ayarlar, tekli oyuncu modu (hikâye modu), Çoklu oyuncu modu ve Yapay zekâ modu. Kullanıcının oyuna giriş yaptığı zaman, kendi profilini oluşturup bunları veri tabanında saklanması gerekiyor. Kullanıcı ile etkileşimli bir ara yüz vardır. Kullanıcıların kolay bir şekilde erişim sağlayabileceği bir ara yüz tasarlanmıştır. Kullanıcıdan alınan kişisel bilgilerin güvenli bir şekilde saklanması garanti altına alınmalıdır.

Kullanıcı hikâye modunu seçtiği zaman sistem kendisine otomatik olarak bir savaş gemisi verecek ve geminin kendine özel dayanıklılığı, ateş gücü ve motorlarının kuvveti

olacaktır. Bunlar daha sonra kullanıcının oynadıkça geliştirebileceği şekilde tasarlanmıştır. Hikâye modunda kullanıcın puanı olup düşman gemisi ve baseleri yok ettikçe bu puanı artacak ve bu puanlarını gemisini geliştirmede kullanılmıştır.

Oyun uzayda geçtiği için gezegenlerin kendilerine ait atmosfer şartları vardır. Şartlar oyunda içerine dahil edilmiş ve kullanıcıya gerçeğe yakın bir oyun keyfi imkânı sunulmaktadır. Hikâye modunda kullanıcı uzayda belli bir yerden başlayıp ve önüne gelen düşman gemilerini ve yaklaşmakta olduğu gezegenin koşullarına göre meteor ve göktaşlarını olup kullanıcı engelleri başarılı bir şekilde geçtikten sonra gezegene giriyor ve düşmanın baselerini yerleşim alanlarını imha ediyor.

Düşmanın da kullanıcıya karşı savunma sistemleri vardır. Kullanıcı bölümü bitirdikten sonra kullanıcıya puanı söylenecektir ve bölüm sonunda kullanıcı gemisini geliştirebilecektir. Bu doğrultuda oyuncunun kendisini geliştirebilir ve oynadıkça seviye atlayabilecek bir imkân sunulmuştur.

Kullanıcı Online mod da kendi profili ile bağlantı odası oluşturacak ve ağ üzerinden bağlanan diğer kullanıcılar bu odaya girebilecekler. Bağlantı kurulduktan sonra kullanıcılar çoklu şekilde hikâye modunu oynayabilecektir.

Yapay zekâ ile oynama kısmında, yine online mod da olduğu gibi çoklu oynayabilecektir. Diğer savaş makinesi sistem tarafından alfa – beta algoritmasına göre yönetilmektedir.

1.2 ANDROİD STUDİO KURULUMU

İlk olarak Java SE Downloads sayfasına ulaşıyoruz. Bu sayfaya ulaştığımızda altında "Java Platform (JDK) bu bağlantıya tıkladıktan sonra karşımıza gelen işletim sisteminize uygun JDK'yı indiriyoruz ve sayfaya geliyoruz.

Bu sayfaya da geldiğimizde Java SE Development Kit 8u11 bölümünün altında "Accept License Agreement" Radio butonunu işaretleyerek gerekli kural ve lisans koşullarını kabul ettiğinizi belirtiyoruz. Ardından size uygun olan işletim sistemini seçiyorsunuz. Listeye bakacak olursak Windows 7 ya da Windows 8 diye seçeneklerin olmaması fark etmediğini gösteriyor. Bizimkisi Windows x64 ve o yüzden onun karşısında ki jdk-8u11-windows-x64.exe'yi indiriyoruz. İndirme tamamlanıyor ve indirdiğimiz JDK kurulum dosyasına tıklıyoruz.

Son olarak Close diyerek JDK kurulumunu tamamlıyoruz. Ardından yine Google'da arama yapıp bu sefer Android SDK yazıyoruz. Karşımıza çıkan ilk bağlantıya yani Android SDK | Android Developers sayfasına giriyoruz.

Sayfada Karşımıza çıkan Download Eclipse ADT width the Android SDK for Windows butonuna tıklıyoruz.

Yönlendirildiğimiz sayfada yine gerekli koşulları ve yine bilgisayarımızın 32 bit mi yoksa 64 bit mi olduğuna göre seçimimizi yapıyoruz. Bunları yaptıktan sonra indirme butonu aktif oluyor ve tıklayarak indirmeye başlıyoruz.

İndirdiğimiz zip dosyasını açtıktan sonra içindeki klasörü istediğiniz yere çıkarabilirsiniz.

1.3 GENYMOTİON KURULUMU VE AYARLANMASI

İlk olarak Genymotion'un resmî sitesi olan www.genymotion.com 'a giriyoruz. Ardından burada karşımıza çıkan "Get Genymotion" yazılı butona tıklıyoruz. Karşımıza bize üyeyseniz giriş yapın, değilseniz üyelik formundan üye olun diye iki tane bağlantı veriyor.

Yönlendirildiğimiz sayfanın sağ kısmında üyelik formu, sol tarafından giriş formu mevcut. Üyelik formunda "User profile (optional)" altında Enterprise size (Kurumsal boyut) yani şirket amaçlı indiriyorsanız şirkette çalışan kişi sayısını belirtiyorsunuz, Usage type (Kullanım tipi) yani Genymotion'u ne amaçla kullanacağınızı belirtebiliyorsunuz. Üye olduktan sonra size bir mail yolluyor ve aktivasyon yapıyorsunuz. Ardından Genymotion'u indirebileceğiniz sayfaya geliyorsunuz.

Burada JDK'da olduğu gibi her işletim sistemine ait indirme seçenekleri mevcut. Windows için 32 bit ve 64 bit için tek bir indirme dosyası var. O yüzden ona tıklıyoruz ve indiriyoruz. Ardından indirdiğimiz exe dosyasına çift tıklayarak kuruluma başlıyoruz. Bir aşamada uyarı vererek "Oracle VM VirtualBox" kurayım mı diye soruyor ve "Oracle VM VirtualBox" kuruyoruz.

Kurulum tamamlandıktan sonra finish diyoruz ve hem Genymotion hem de "Oracle VM VirtualBox Manager" aynı anda açılıyor. Genymotion ilk açıldığında yeni bir emülatör kurulsun mu diye soruyor. Ardından karşımıza boş bir pencere açılıyor ve sağ alt kısımda Connect butonu görüyoruz. Ona tıkladığımızda Genymotion'un sitesine üye olurken belirttiğimiz kullanıcı adı ve şifreyi giriyoruz.

Kullanıcı adı ve şifreyi girdikten sonra karşımıza piyasada bulunan bir sürü tablet ve cep telefonu modeli listesi çıkıyor**.** Üst kısımda "Android Versiyon" yazan kısmın karşısında

son çıkan Android versiyonları mevcut. Örneğin 4.4.2 versiyonunu seçerseniz o versiyon ile çıkan telefon ve tabletleri görüyoruz. Hemen onun sağ tarafında “Device Model” var ve orada da telefon ve tablet modellerini seçerek o telefon ve tablete ait modeller çıkıyor. Kısacası filtreleme yapıyor.

Burada bize hem seçtiğimiz telefon ya da tabletin özelliklerini belirtiyor, ona istediğimiz ismi verebiliyoruz. İstediğimiz bir isim verdikten sonra tekrar Next dedikten sonra seçtiğimiz telefon ya da tablete ait gerekli indirmeleri yapıyor.

İndirmenin tamamlanmasını bekliyoruz ve kurulum bittikten sonra Finish diyerek eklediğimiz emülatör listemize eklemiş oluyoruz.

2. YAPILAN ÇALIŞMALAR

2.1 ANDROİD NEDİR

Android; Google ve Open Handset Alliance tarafından, mobil cihazlar için geliştirilmekte olan, Linux tabanlı özgür ve ücretsiz bir işletim sistemidir. Sistem açık kaynak kodlu olsa da kodlarının ufak ama çok önemli bir kısmı Google tarafından kapalı tutulmaktadır. Google tarafından ücretsiz olmasının sebebi, sistemin daha hızlı ve çabuk gelişmesi, birçok popüler marka tarafından kullanılması ve bu sayede reklamlarını daha fazla kişiye ulaşmasını sağlamaktır.

Google, Android sistemi üzerinde çalışan Google Play marketteki oyun ve uygulamalar üzerinde aldığı reklamları yayınlayarak para kazanmaktadır. Android 'in desteklenen uygulama uzantısı ". apk" dır.

Android, aygıtların fonksiyonelliğini genişleten uygulamalar yazan geniş bir geliştirici grubuna sahiptir. Android için halihazırda 1 milyondan fazla uygulama bulunmaktadır. Google Play Store ise, Android işletim sistemi uygulamalarının çeşitli sitelerden indirilebilmesinin yanı sıra, Google tarafından işletilen kurumsal uygulama mağazasıdır. Geliştiriciler, ilk olarak aygıtı, Google'ın Java kütüphanesi aracılığıyla kontrol ederek Java dilinde yazmışlardır.

Open Handset Alliance, 5 Kasım 2007'de Android'i kurduğunu duyurmuştur ve ardından 34 adet donanım, yazılım ve telekom şirketi, mobil cihazlar için telif hakkı olmayan bir işletim sisteminin teknolojinin gelişimi için yararlı olduğu konusunda hemfikir olmuşlardır.

Android, Linux çekirdeği üzerine inşa edilmiş bir mobil işletim sistemidir. Bu sistem ara katman yazılımı, kütüphaneler ve API C diliyle yazılmıştır. Uygulama yazılımları ise, Apache harmony üzerine kurulu Java-uyumlu kütüphaneleri içine alan uygulama iskeleti üzerinden çalışmaktadır. Android, derlenmiş Java kodunu çalıştırmak için dinamik çevirmeli Android Runtime (ART) kullanır ve cihazların fonksiyonelliğini artıran uygulamaların geliştirilmesi için çalışan geniş bir programcı-geliştirici çevresine sahiptir. Google aynı zamanda işletim sistemindeki hataları bulan kullanıcıları para ödülü ile ödüllendirmektedir.

2.2 ANDROİD STUDİO İLE EKRAN TASARIM YAPMAK

2.2.1 EKRAN TASARIMI NASIL YAPILIR

2.2.1.1 ARAŞTIRMA

İlk önce uygulamanın fikrini iyice araştırmalıdır. Araştırma en basitinden daha önce yapılmış işleri tekrarlamamak için önemlidir. Bulunan örnek tasarımlarından kendi tasarımınıza katacağınız tarz, kompozisyon, grid gibi temel tasarım fikirleri geliştirilebilir. Tabi ki örnekleri incelemek var olan tasarımları kopyalamak için yapılmaz.

Araştırma süreci kopyalamak için değil o tasarımlardaki çalışmayan unsurları iyi analiz edip geliştirmek için önemlidir. Renk kombinasyonlarını, tipografi seçimlerini, görsel dili detaylı bir şekilde incelemek tasarlanan ara yüze değer katar. Başarılı olduğu kadar başarısız örnekler de incelenmeli ki onlardaki tasarım hataları tekrarlanmasın. Tabi burada en önemli konu bu dediğimiz tüm aşamalara gelmeden önce uygulamanın içeriğinin hazır olması.

Tasarımcı içeriği yaratmaz sadece ona görsel ve işitsel şekil/form verir. Bu yüzden içeriğin tasarım başlamadan tamamlanması gerekir ve tasarımcı o içeriği analiz ederek tasarımını ona göre hazırlar. Yarım kalmış veya tasarım sürecinde değişen içerik, tasarım aşamalarının değişmesine neden olur. Bundan dolayı tasarım araştırma süreci mobil uygulamanın içeriğine göre yapılır.

2.2.1.2 AKIŞ ŞEMASI

Akış Şeması, mobil uygulamanın içindeki ekranlar arasındaki geçişleri, hangi ekrandan hangi ekrana geçileceğini ve uygulama içindeki diğer navigasyonel fikirlerin tasarım süreci başlamadan önce görsel olarak düzenlenmesidir.

Aslında kısaca bir içerik planlama şemasıdır ve bunun tasarım süreci boyunca belgelenmiş olması önemlidir. Böylece, hangi ekranları tasarlayacağınızı sürecin başından itibaren belirlemiş olursunuz. Akış şeması, uygulamanın içinde ara yüzler arasındaki geçişin, çalışma akışının ve sürecinin her bir aşamasını (ara yüzünün) oklar ile birbirlerine bağlanarak görsel anlatımına denir. Mesela, bir sosyal medya uygulaması yapılıyorsa; “Giriş Yap” ekranından başka hangi ara yüzlere geçiş olacağı (mesela, “Şifreni mi

unuttun" veya "Kayıt Ol" ekranları). Bu akış şeması aslında "kullanıcı uygulama içinde hangi ara yüzden hangisine geçecek ve uygulama toplamda ne kadar ara yüz içerecek? Sorularına cevap vermek için tasarım aşamasının başında yapılmalıdır.

Akış şemasının ara yüz tasarımı aşaması başlamadan iyi analiz edilip sonlandırılması hem tasarımcı hem de yazılımcının işini kolaylaştırır.

2.2.1.3 WİREFRAMİNG

Türkçe 'ye tel kafes diye çevrilmiş "Wireframe" aslında ara yüz tasarımının şematik planıdır. Yani, tasarımın görsel bir iskeletidir. Genellikle elle çizilen ama günümüzde ara yüz tasarımının gelişmesiyle dijital ortamda da hazırlanabilir.

Wireframe, ara yüzdeki görsel elemanların yerleşimini genel hatlarıyla göstermek için yapılır. Ara yüzdeki kompozisyon ve elemanlar arasındaki hizalama hiyerarşi, kontrast, devamlılık, yakınlık, arka plan ve şekiller arasındaki ilişkiyi düzenlemek için ilk önce wireframe tasarlamakta yarar vardır.

Butonların nerede duracağı, imgelerin pozisyonları ve büyüklükleri ile metin aralarındaki ilişkiler bu süreçte çözülür. Farklı görsel senaryoların ne şekilde gösterileceği, hangi özelliklerin nerede duracağı, bilgi ve fonksiyon arasındaki öncelik ilişkisi wireframe tasarlanırken göz önüne alınmalıdır. Bunun için ara yüzde tutarlı bir grid oluşturmak en mantıklısı olacaktır. Grid oluşturmak hem tüm ara yüzler arasındaki tasarım tutarlılığını sağlar hem de tasarım elemanlarını yerleştirmeye yardımcı olur.

Uygulamanın ilk ara yüzünün tasarımını yaparken uygulamanın birden çok ara yüzü olacağı unutulmamalıdır. Mesela, "Kayıt Ol" ara yüzü ile "Arkadaşlar" ara yüzü birbirinden çok farklı olabilir. Farklı ara yüzlerle başa çıkmak için tasarımda kullanacağınız görsellerin her ara yüzdeki yerleşimi düşünmek gerekir. Tasarımda buna "tutarlılık" denir.

2.2.2 ANDORİD İLE TASARIM

Bir Android uygulamasında grafik kullanıcı ara yüzü View ve ViewGroup nesnelerinin hiyerarşik yapısıyla oluşturulur. View nesneleri genellikle butonlar, text fieldlar gibi kullanıcı arabirim nesneleridir. ViewGroup nesneleri ise görünmeyen view tanımlamaları içerir. Görünmeyen view tanımlamalarından kasıt XML'deki child yapısıdır aslında. Grid, vertical list gibi child view gibi nesneleri içerir.

Android, View ve ViewGroup'ların alt sınıflarına karşılık gelen bir XML sözcük yapısı içerir Böylece arabirimde kullanacağınız View'ları bir hiyerarşi kullanarak tanımlanabilir. Layoutlar (düzenler) ViewGroup alt sınıflarıdır.

2.2.2.1 ANDROİD LAYOUT MANAGER

Ekranda görünmesini istediğimiz görüntülerin, kontrollerin yani View ve ViewGrouptan inherit edilmiş her şeyin toplandığı sınıftır. ViewGrouptan inherit edilmiş bir sınıf yazarak kendi özel işimizde kullanabileceğimiz yani custom layout yapabiliriz.

2.2.2.1.1 LİNEAR LAYOUT NEDİR

Linearlayout en temel layout olmakla birlikte xml'i açtığımızda default olarak gelmektedir. Linear Layout kullanımı Android'de, tüm nesneleri tek bir yönde kullanmamızı sağlar. Linear layout sayesinde nesneleri android: orientation özelliğini kullanarak, tamamen yatay veya dikey olarak konumlandırabiliriz.

```
android: orientation="vertical"      //dikey konum
android: orientation="horizontal"    //yatay konum
```

2.2.2.1.2 RELATİVE LAYOUT NEDİR

Linear Layout da yatay ve dikey olarak konumlandırma varken, bu layout türümüzde böyle hazır bir şablon bulunmamaktadır. Relative Layout içine eklediğimiz android bileşenlerini birbirlerinin aşağısına, yukarısına, sağına ve soluna şeklinde konumlandırma yapabılır. Ayrıca, birden fazla iç içe layoutları kullandığında da ilgili layout un id sini belirterek, belirli methodlara göre hizalama yapılabilir.

2.3. SQLİTE NEDİR

Kullanımı çok basit, uygulama içerisinde az yer kaplayan ve mobil cihazlarda rahatlıkta kullanılabilen açık kaynak kodlu ve işletim sisteminden bağımsız bir veri tabanı kütüphanesi ve motorudur. SQLite, onlarca programlama dili ile kullanılabilir. Az yer kaplaması ve istenilen hızda veri işlemleri yapabilmesi tercih sebepleri arasındadır. SQLite veri tabanında tarafından desteklenen text, numeric, integer, real ve none veri tipleri mevcuttur.

2.3.1. SQLİTE MİMARİSİ

Packagelar; android database package veritabanları ile çalışmak için tüm genel sınıfları içerir. android.database.sqlite SQLite a ait spesifik sınıfları içermektedir.

SQLiteOpenHelper; Android uygulamasında create ve update veri tabanı methodları için genellikle SQLiteOpenHelper sınıfı kullanılmaktadır. Bu subclassta super() methodu kullanarak veri tabanı adı ve veri tabanı versiyonu belirtilir.

```
super(context, DATABASE_NAME, null, DATABASE_VERSION);
```

Bu sınıfta onCreate() ve onUpgrade() methodlarını override etmek gerekmektedir. onCreate() methodunda veri tabanı yoksa oluşturulması sağlanır. onUpgrade() methodunda ise veri tabanı versiyonunda güncelleme varsa, veri tabanı şemasının güncellenmesi sağlanır.

SQLiteOpenHelper sınıfı getReadableDatabase() ve getWriteableDatabase() methodları ile SQLiteDatabase objelerinde okuma ya da yazma işlemini yapmayı sağlar.

Veritabanı tablolarında tablonun primary key tanımlaması için _id standart şekilde kullanılmaktadır. Çoğu Android fonksiyonları bu standarttı içermektedir.

SQLiteDatabase; SQLiteDatabase SQLite veri tabanı ile çalışmak için temel sınıftır. Veritabanı işlemlerinde insert, update ve delete methodlarını sağlar. Buna ek olarak, SQLiteDatabase execSQL() methodunu da sağlar. Bu method SQL statement'ı direk çalıştırılmasına izin verir.

ContentValues nesnesi key ve value tanımlamayı sağlar. Key tablodaki sütunun tanımlanmasını, value de tablodaki o sütundaki içeriği temsil eder. ContentValues ile insert ve update methodları kullanılabilir.

Queryler rawQuery() ve query() methodları ya da SQLiteQueryBuilder sınıfı ile oluşturulabilir. rawQuery() methodu direk SQL select statement'ı girdi olarak kabul eder. query() methodu SQL sorgusu için yapısal bir arayüz sağlar.

Cursor; Cursor nesnesi bir sorgunun sonucunu temsil eder. Bütün veriyi hafıza yüklenmesine gerek olmadığı için etkili bir şekilde uygulamanın çalışmasını sağlar. Sorgu sonucundaki elementlerin sayısını öğrenmek için getCount() methodu kullanılır. Veriler arasında geçiş içinde moveToFirst() ve moveToNext() methodları kullanılır. isAfterLast() methodu ise sorgunun sonuna ulaşıp ulaşılmadığını kontrol etmek için kullanılır.

ListView, ListActivity ve SimpleCourseAdapter; ListViewler veri sonuçlarını gösterebilmeyi sağlar. ListActivityler ise ListViewlerin kullanımını kolaylaştıran aktivitelerdir. ListViewler ve veritabanları ile çalışabilmek için SimpleCursorAdapter ı kullanabiliriz.

2.4. SQLİTE İLE VERİ TABANI OLUŞTURMAK

SQLiteOpenHelper'dan türetilen bir sınıf oluşturmak:

```
public class TestDBHelper extends SQLiteOpenHelper {}
```

Bu bizlere 2 methodu kullanma zorunluluğu getirecektir.

onCreate (SQLiteDatabase db): Veri tabanı ilk oluşturulduğunda çağrılır. Tabloları, viewleri, trigger yapıları bu method içinde oluşturulur.

onUpgrade (SQLiteDatabase db, int eskiVersiyon, int yeniVersiyon): Veri tabanı üzerinde herhangi bir değişiklik olduğunda çağrılır.

Sabitleri tanımlamak: static final String _ID=" _ID"; Sabitleri helper sınıf içinde tanımladığımız gibi farklı bir sınıf içerisinde de tanımlayabiliriz. Yeter ki erişilebilir ve kullanımı anlaşılabilir olsun

onCreate() ve Tablo Oluşturmak: Tablo oluşturma işlemlerimizi onCreate() methodu içerisinde yapıyoruz. Aşağıda bir kişi tablosu oluşturmak için gereken SQL cümleciğini ve bir tablonun nasıl oluşturulduğunu bulabilirsiniz.

```
db.execSQL("CREATE TABLE"+ KISITABLE+"("+_ID+" INTEGER PRIMARY
KEY AUTOINCREMENT,"+ ad+"       TEXT, "+soyad+" TEXT, "+yas+" INTEGER);");
```

onUpgrade(): Veri tabanında herhangi bir modifikasyon olduğunda onUpgrade() methodu çalıştırılır. Daha sonrasında çağrılan onCreate() methodu değişikleri hayata geçirmeye yarar. (Drop Table, Alter Table, Drop arg. Alter arg)

Veritabanı İşlemleri için Gereksinimler:

close() : Veri tabanını kapatır.

getReadableDatabase(): Okunabilir bir veri tabanı sunar. Select işlemleri için uygundur.

GetWritableDatabase(): Yazılabilir bir veri tabanı sunar update, delete ve insert işlemleri için uygundur.

Queryleri Çalıştırmak: Sqlite hem bilinen sql cümleciklerini (rawQuery()) desteklemektedir hem de kendi içinde özelleşmiş sorgulama methodları bulunmaktadır.

db. RawQuery(String sql, String[] selectionArgs)

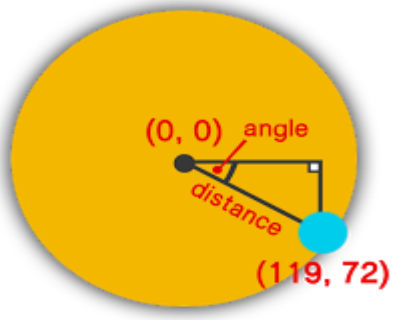
2.5. JOYSTİCK NASIL YAPILIR

Joystick, uygulamada iki ana parça ile birlikte yaygın şekilde kullanılır. Arka plan resmi, joystick 'in alanıdır. Ve kol görüntüsü (kullanıcının ekran ile temas halinde gözükür). Kullanıcı joystick 'in alanına dokunduğunda. Parmağınız gerçek joysticki taklit etmek için sürüklendiğinde hareket edecektir.

Şekil 1.1 Joystick Ana parçaları

onTouchListener için Kullanıcıların çalıştırdıklarında kendi hesaplarını oluşturmalarına izin verir. Koordinat düzleminde (0, 0) position_x, position_y referans dokusunun koordinatları (0, 0) basit trigonometri hesaplamasıyla kullanılan mesafeyi hesaplar.

Daha sonra köşe açısını (0, 0) noktasıyla hesaplanır, alana dokunulduğunda durum ACTION_DOWN'a karşılık gelecektir, tekrar kontrol edecektir. Dokunma aralığı gerekli periyotta kontrol edilir.



Şekil 1.2 Ekrana Dokunma Durumu

Blok sahibi, mesafe kumanda çubuğunun aralığını aşarsa kol göstergesi olarak kabul edilmez. Kumanda çubuğunun kapsamı, dokunma noktasında gösterilen tuval ile bir kol çekmek ise, kolun başlatılıp başlatılmadığını belirlemek için touch_state değişkeni kullanılır.

Şimdi, ACTION_MOVE'in durumuna bakalım; kullanıcı, ekrana dokunduğunda ve parmağınızı ekranda sürüklediğinde, touch_state değişkenini önce kolu gösterip göstermeyeceğini kontrol edecektir. O zaman iki koşulla kontrol edilecektir.

mesafe <= (params.width / 2) - OFFSET

Bu, parmağınızı joystick 'in alanına sürüklediğiniz bir durumdur, kolu bu dokunmayla ekrana çizdirilir.

Şekil 1.3 Kolun Ekrana Çizdirilmesi

mesafe> (params.width / 2) - OFFSET

Parmağın dokunma alanının dışında kaldığı bir durumdur. Kol sınırdaymış gibi gözükür.

Şekil 1.4 Ekran Dışı Dokunma

mLayout.removeView (draw): ACTION_UP parmağınızı ekrandan serbest bıraktığınızda kol görüntüsünü kaldıracaktır. Daha sonra joystick 'in bir sonraki dokunuşunu beklemek için touch_state değişkenini false olarak ayarlanır.

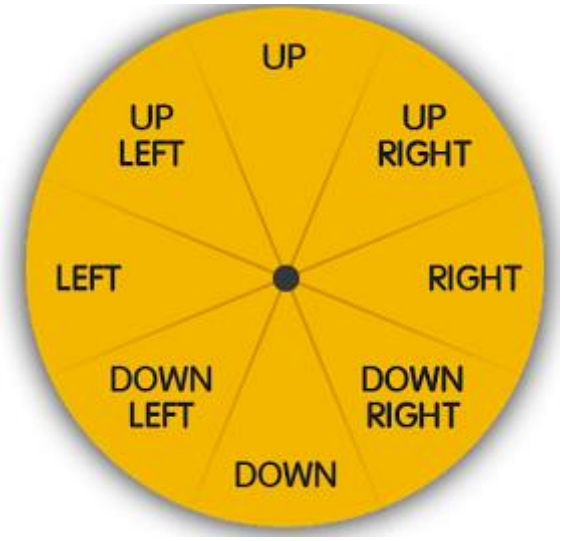
GetPosition: İşlevi, dokunmatik konum değerini okur. Bu, iki Tamsayı Dizisi döndüren bir işlevdir. Birincisi X-koordinatı, ikincisi Y-koordinatıdır.

GetX: İşlevi X eksenindeki konumu okur ve getY işlevi Y eksenindeki konumu okur. Bu iki işlev, okuyucunun yalnızca bir değeri okuması içindir.

GetAngle: İşlevi, tanjant açısını derece olarak (0, 0) okur.

GetDistance: İşlevi, kontak noktasının joystick alanının veya koordinatların merkezinden (0, 0) ne kadar uzakta olduğunu hesaplar.

Joystick'in hangi hareketin neye karşılık geldiği ana joystick resmi üzerinde konuma göre belirlenir.



Şekil 1.5 Joystick'in Bölümleri

2.6. DİNAMİK NESNE YARATMAK

Oyun oynarken nesnelerin dinamik olarak yaratılmasına ihtiyaç duyulur. Android Studio'da ekran da ekrana bir şey eklemek istediğimiz zaman bunu xml ile yaparız. Eğer kod ile dinamik olarak bir şeyler yaratmak istersek, ne yaratmak isteniyorsa ilk önce yeni bir tane o nesneden yaratılır. Ardından nesne ye hangi işlemler uygulanmak isteniyorsa uygulanır.

RelativeLayout.LayoutParams params = new RelativeLayout.LayoutParams(}

Bu kodun içerisine nesneye hangi işlemleri uygulamak istiyorsak uygularız ve daha sonra bu nesneyi görüntülemek istediğimiz ekranımıza ekleriz.

2.7. PUAN BİLGİSİ

Oyun içerisinde kullanıcının bölüm sonunda aldığı puanı merak eder ne kadar aşama ilerlediğini bilmek ister bunu sağlamak kesişim testinin sonucuna bağlıdır.

Puanın görüntülenmek istediği ekranda textView eklenir ve konumu ekranın sol üst köşesine sabitlenir. Başlangıç değeri olarak 0(sıfır) setlenir. Kesişim testinden gelen sonuç TRUE ise puan önceden belirlenmiş içerik kadar arttırılır ve ekranda ki puan bilgisi güncellenir. Eğer kesişim testinden gelen sonuç FALSE olursa puan bilgisinde herhangi bir güncelleme yapılmaz.

2.8. ARKA PLAN OLUŞTURULMASI

Oyun oynarken kullanıcının yerinde duruyormuş gibi değil de sanki uzay içinde hareket ediyormuş izlenimi yaratmak istenir. Bunu sağlayabilmek için android studi'da ScrollingImageView methodu vardır.

Drawable klasörünün içine arka plan da eklenmek istenen resim yüklenir. ScrollingImageView methodunu tanımladığımız yerde background olarak, drawable klasörüne eklediğimiz resim verilir. Daha sonra arka planı başlatılır.

Bu sayede oyun içerisine kullanıcıya uzayda hareket ediyormuş gibi bir izlenim bırakılır.

2.9. UZAY

Uzay sistemi, milyarlarca galaksiyi, yıldızı, gezegeni, gravitik dalgaları, kozmik ışıkları, gaz ve toz bulutlarını uyduları, göktaşlarını kısacası her şeyi içerisinde bulunduran sonsuz boşluktur. Diğer adı da evren (kozmos) olarak geçiyordur. Dünya, Samanyolu Galaksisinde yer alır. Samanyolu Galaksisinde Güneş de dahil olmak üzere yaklaşık 400 milyar yıldız olduğu tahmin edilmektedir.

Şekil 2.1. Samanyolu Galaksisi

Bizim galaksimize en yakın dış galaksi Andromeda Galaksisi ‘dir ve evrende Samanyolu Galaksisi ‘ne benzer milyarlarca galaksi sistemi bulunmaktadır.

Şekil 2.2 Andromeda Galaksisi

Uzay, Dünya'nın atmosferi dışında evrenin geri kalan kısmına verilen isimdir. Uzay'ın sınırları asla kesin değildir ve Uzay hep büyür. Atmosfer ile uzay arasında kesin bir sınır bulunmamaktadır, fakat Dünya'nın atmosferi yukarı doğru çıkıldıkça incelmektedir. Uzayda milyonlarca gökada bulunmaktadır. Bu gökadalar içinde milyonlarca güneş sistemleri, gezegenler ve gök taşları bulunmaktadır.

Uzay çok eski dönemlerden beri insanların büyük ilgisini çekmiş, sonu olup olmadığı; varsa, sınırlarının nereye kadar uzandığı bilginleri ve felsefecileri yakından ilgilendirmiştir. Uzayda yer alan gökcisimlerinin incelenmesi, bunların hareketlerinin diğer gökcisimlerinin davranışlarına yaygınlaştırılması, uzay hakkında çok az da olsa kimi fikirlerin ortaya atılmasını sağladı.

Çağlar geçtikçe insanların daha güçlü teleskoplarla uzayı incelemesi uzay hakkındaki bilgileri artırdı. Uçan cisimlerin ortaya çıkmasıyla Dünya'yı çevreleyen yakın uzay hakkındaki bilgiler, daha da artmaya başladı. Nihayet, güçlü füzeler, yapma uydular, Ay'a insanlı ya da insansız araçlar gönderilmesi, Güneş Sistemi içinde yolculuk yapacak yapma uyduların geliştirilmesi, çok güçlü radyo teleskoplarla uzayın derinliklerinin araştırılması, 20. yüzyılın ikinci yarısında insanlığın uzay hakkındaki bilgilerini önemli ölçüde genişletti. Bu arada teorik fizik ve astronomi konusunda devrim yapacak görüşler ortaya atan Einstein gibi bilginlerin uzay konusunda ortaya attıkları pek çok kuram, gözlemcilerin uzay üzerine verdikleri bulguların mantıklı bir şekilde açıklanmasını sağladı.

Uzay konusundaki ilk sağlam bilgiler, 19. yüzyıl sonu ile 20. yüzyıl başında, özellikle kuzey ülkelerinde kurulan gözlemevleri sayesinde alındı. ABD'nin Kaliforniya eyaletinde bulunan Palomar Gözlemevi, Dünya'da mevcut gözlemevlerinin en büyüğüdür. Buradaki aynalı teleskopun çapı 5 m., yüksekliği 40 m.dir. Bu gözlemevlerinde uzaydaki gökcisimlerinin kütlesi, hacmi, ışığının şiddeti vb. incelenmektedir. Uygulamalı fiziğin geliştirdiği tayf (spektrum) analizi, uzaydan gelen ışıklardan, cisimlerin hangi elementlerden oluştuğunu göstermektedir. 1932'de K. G. Jansky adındaki bir mühendisin rastlantı sonucu bulduğu uzaydan gelen radyo yayınları, daha sonraki yıllarda radyo teleskopların doğmasına ve uzayın derinliklerinin dinlenmesine, bu radyo yayınlarının kaynaklarının ve nedenlerinin bulunmasına yol açtı.

II. Dünya Savaşı sırasında Almanların geliştirdiği V-1 ve V-2 füzeleri daha sonraki yıllarda uzayın keşfi için yapılacak çalışmalarda büyük bir adım oldu. 1947-1956 yılları arasında özellikle ABD, uzay çalışmalarına büyük hız verdi. Yapılan uzay uçuşu denemelerinin hiçbiri bir uzay aracını yörüngeye oturtmayı başaramadı. Bu arada SSCB,

1957 yılında üç kademeli Vostok füzeleri ile "Sputnik" adındaki ilk yapma uyduyu Dünya çevresinde yörüngeye oturtarak uzay yarışında öne geçti.

Uydulardan elde edilen uzay üzerine bilgiler, canlıların, özellikle insanların uzayda yaşayabilmeleri için hangi koşulların yerine getirilmesi gerektiğini ortaya koydu. Böylece uzay tıbbı doğdu ve gelişti. Uzayda ilk insan ise 12 Nisan 1961 tarihinde SSCB'nin uzaya gönderdiği Yuri Gagarin oldu. Bu arada, insanların uzay boşluğuna yerleşmelerini sağlamak, uzayı uzaydan izlemek, Dünya üzerinde haberleşme kolaylıkları sağlamak için binlerce uydu yörüngeye yerleştirildi ya da uzayın boşluğuna fırlatıldı. Nihayet 1969 Temmuz'unda Ay'ın ABD'li astronotlar tarafından fethedilmesi, uzay çalışmalarında en önemi adımlardan biri oldu. Günümüzde uzay yarışı büyük bir hızla sürmektedir. Araştırmacılar galaksimizde bulunan 37 bin 964 gezegenin en az bizim kadar akıllı yaşam formlarına ev sahipliği yapabileceğini hesapladı.

Astrofizik uzmanı Duncan Forgan tarafından geliştirilen bir bilgisayar yazılımı bilinen 330 gezegenden topladığı veriler ışığında yaşam için elverişli olabilecek gezegenleri hesapladı.

Samanyolu galaksisinde bulunan gezegenlerin sıcaklık, su ve mineral zenginliği gibi değişkenler gözetilerek incelenmesi sonucunda üç senaryo göz edilerek hesaplamalar yapıldı. Birinci senaryoda hem yaşamın hem de evrimin zor olacağı bir algoritma yaratan yazılım, buna göre 361 gezegenin yaşama olanak tanıyabileceğini hesapladı. İkinci senaryoda yaşamın başlaması zor olsa da evrime imkan tanıyabilecek bir algoritma kullanan yazılım bu sayıyı 31 bin 513'e çıkardı. Üçüncü senaryoya göre yaşamın gezegenler arası göktaşlarıyla taşınabileceği varsayımına dayanan bir algoritma kullanan yazılım bu sayının 38 bine yaklaştığını gösterdi.

Hesaplamanın tek hücreli yaşam formlarından daha ziyade analog organları olan ve bilinç sahibi yaşam formlarını yaratmak için gerekli olan koşulları araştırdığını belirten bilim insanları, en zor koşullarda bile 361 gezegenin zaman içinde akıllı yaşam formlarına ev sahipliği yapabileceğini, akıllı uzaylılarla insan ırkının iletişim kurmasının 300 ve 400 yıl gibi bir süre alabileceğini belirtiyorlar.

2.9.1 EVRENİN İÇERİGİ VE FİZİKİ YAPISI

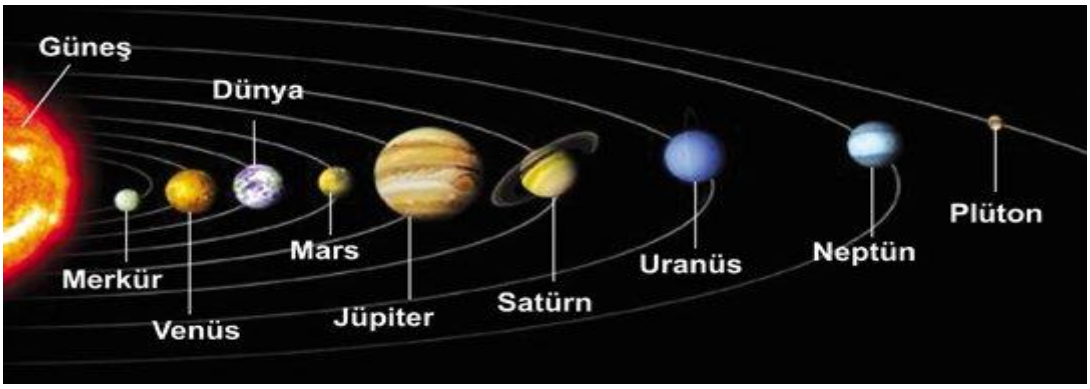
1-Galaksiler
2-Elektromanyetik radyasyon
3-Nötral ve iyonize hidrojen
4-Toz parçacıkları
5-Galaksilerden gelen ışıklar
6-Süpernova ve Galaktik patlamalardan oluşan kozmik ışınlar
7-Kütlesi olmayan nötronlar
8-Gravitik dalgalar.

Sadece bizim galaksimizde 400 milyar yıldız (Güneş) bulunduğu tahmin edilmektedir. Bizim galaksimiz gibi içinde yıldızları ve gezegenleri barındıran milyarlarca galaksi vardır. Galaksiler, gazlar, yıldızlar, tozlar ve gezegenler içeren en büyük madde topluluğudur. Bir galaksinin en sonunda alacağı biçim küre biçimidir daha sonra muhtemelen Karadelik haline gelir. Ayrıca yıldızlar arasında çok büyük miktarlarda gaz ve toz bulutları ve belki de bilinmeyen milyarlarca gezegen ile onların uyduları bulunmaktadır. İnsanoğlunun daha ilk çağlardan beri süregelen merakı, düşünen ve araştırmacı yapısı hemen her konuda olduğu gibi uzayıda araştırma ve inceleme yapmasına neden olmaktadır. Günümüzde NASA (National Aeronautics and Space Administration, Ulusal Havacılık ve Uzay Dairesi olarak tercüme edilebilir), ESA (the European Space Agency, Avrupa Uzay Ajansı) gibi kuruluşların yanı sıra Rusya, Japonya, Kanada, Çin gibi ülkelerde uzay araştırmalarında öncülük yapmaktadır.

Uzay araştırmalarının başlıca nedenlerini şu şekilde sıralayabiliriz:

1. Güneş sistemimizin araştırılıp incelenmesi, gezegenlerin yapısı
2. Dünya dışında yaşam olasılığının araştırılması
3. Galaksiler, yıldızlar, karadelikler ve diğer uzay yapıtaşlarının incelenmesi

2.9.2 GÜNEŞ SİSTEMİ VE DİĞER GEZEGENLER



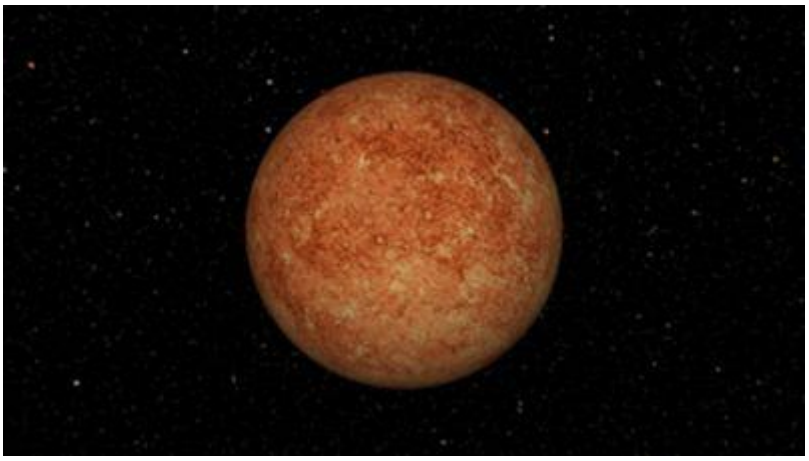
Şekil 2.3 Güneş sistemi

Güneş sistemi yaşama, 4,6 milyar yıl önce, içinde kayaç ve buz parçacıkları bulunan büyük bir gaz bulutu kütlesi olarak başlamıştır. Bulut kendi çekim gücü nedeniyle sıkıştığında güneş oluşmuş, tanecikler de bir araya gelerek gezegenleri ortaya çıkarmıştır.

Güneşin iç bölümünde nükleer füzyonla hidrojen helyuma dönüşür ve bu dönüşüm sonucu açığa çıkan enerji, önce ışık yuvarına gider oradan da uzaya gider.

MERKÜR

Güneşe en yakın gezegen Merkür'dür. Ortalama 57,9 milyon km. olan Merkür-Güneş uzaklığı astronomideki diğer uzaklıklara kıyasla gerçekten çok küçüktür.



Şekil 2.4 Merkür

Güneşe çok yakın olduğundan, gündüz vakti Merkür'deki sıcaklık 423 C ye kadar çıkar. Ama güneş battığı zaman sıcaklığın –183 C ye kadar indiği olur. Güneşe bu kadar yakın olmasına karşın bazı uzmanlar Merkür'de hala kraterlerin güneş görmeyen yerlerinde buz bulunabileceğini düşünüyorlar.

Bir teoriye göre Merkür, bundan milyonlarca yıl önce 2 kez hemen hemen kendisi kadar büyük gök cisimleriyle çarpıştı. İlk çarpışma sonucunda Merkür neredeyse tümüyle sıvılaştı, ağır metaller dibe batarak büyük çekirdeği oluşturdu. İkinci çarpışma sonucunda da kabuğun büyük bir kısmı parçalanarak ince bir kabuk kaldı.

VENÜS

Güneşe en yakın ikinci gezegendir. Güneşe uzaklığı 108 milyon km.dir. Dünyaya en yakın konuma geldiğinde güneş ve aydan sonra en parlak cisimdir. Işığı bazen gölgeler oluşturabilir.

Şekil 2.5 Venüs

Venüs'ün atmosferi çok yoğundur. Öylesine yoğundur ki; dünyadaki en güçlü teleskopla bile yeryüzü şekillerinin görülmesi imkansızdır. Atmosferinin basıncı yüzünden ezileceğinden, gökyüzünden yağan sülfürik asitten yanacağından, atmosferi nefes almaya uygun olmadığından büyük bir olasılıkla hiçbir insan Venüs'ün yüzeyine ayak basamayacaktır.

Venüs çok yavaş döner. Kendi çevresinde dönmesi 243 gün sürerken, güneş çevresinde dönmesi 224 gün sürer. Bu nedenle bir Venüs günü bir Venüs yılından daha büyüktür.

DÜNYA

Dünya, güneş sisteminde yaşam olan tek gezegendir. Güneşe uzaklığı ortalama 149,6 milyon km.dir. Dünya, demir ve nikel bir çekirdeği saran kayaç tabakasından oluşur. Derinlere indikçe sıcaklık artar.

Şekil 2.6 Dünya

Yaklaşık 4,6 milyar yıl önce, bir gaz ve toz bulutu yoğunlaşarak güneşi oluşturmuştur. Bulutun içindeki başka maddeler birleşerek dünya ve diğer gezegenleri oluşturmuştur. Dünyada demir ve nikel eriyerek çekirdeği oluşturmuştur. 4 milyar yıl önce dünyanın kabuğu oluşup yanardağlardan çıkan su buharı yoğunlaşarak denizleri meydana getirmiştir.

MARS

Dünyanın yarısı büyüklüğünde olan Mars birçok yönden dünyaya benzer. Mars gününden sadece bir saat uzundur. Marsta da dünyadaki gibi mevsimler vardır. Ama güneşe uzaklığı 227,4 milyon km. olduğu için ortalama sıcaklığı –28C dir. Ayrıca bir Mars yılı 687 dünya günü sürer.

Şekil 2.7 Mars

Marstaki nehir yatakları Mars'ın ikliminin bir zamanlar daha sıcak, atmosfer basıncının da suyun yüzeye çıkmasını sağlayacak kadar yüksek olduğunu gösteriyor. Belki de bilinmeyen bir olay Mars'ın atmosferinin uzaya kaçmasına ve demirce zengin olan toprağının pas rengi almasına neden olmuştur.

Uzay yolculuklarının ateşli taraftarları 2030 yılı civarında insanoğlunun Mars'a ayak basacağını umuyorlar.

Daha sonra Mars'ta üsler kurulacak, bu üsler büyüyüp gelişecek ve en sonunda uzayın daha uzak bölgelerine yapılacak yolculuklar için fırlatma rampası olarak kullanılacaktır.

JÜPİTER

Güneş sistemindeki en büyük gezegen Jüpiter'dir. 16 uydudan oluşan ailesi ile minik bir güneş sistemine benzer. Çok küçük olan katı çekirdeği dışında minyatür bir güneş gibi hemen hemen tümüyle gazdan oluştuğu için Jüpiter diğer gezegenlerden farklı gözükür.

Şekil 2.8 Jüpiter

3 Aralık 1973 tarihinde, Jüpiter'e ulaşan Pioneer-10, dünyaya Jüpiter'in bulutlarına ait ilginç fotoğraflar gönderdi. 1979 yılında Voyager araçları Jüpiter'in dünyadan görülemeyecek kadar ince 3 tane halkası olduğunu buldular.

Jüpiter'deki kırmızı leke ilk kez İngiliz astronom Robert Hooke tarafından 1664 yılında gözlenmiştir. Aşağıdan yukarıya doğru hızla yükselen maddenin yarattığı 8 km. yüksekliğinde, 40.000 km. uzunluğunda ve 14.000 km. genişliğinde olan bir fırtınadır. Saatte 500 km. hızla esen bu fırtına önüne çıkan küçük fırtınaları yutarak büyür.

SATÜRN

Güneş sistemindeki ikinci gezegen olan Satürn, güneşe uzaklık sıralamasında 6. dır. Jüpiter gibi Satürn'de neredeyse tümüyle gazdan oluşur. Kendi çapının beş katı çapa sahip olan çok güzel görünüşlü halkaları oldu için Satürn'e "Halkalı Gezegen" de denir.

Şekil 2.9 Satürn

Satürn'ün yoğunluğu o kadar azdır ki büyük bir göle konsa batmayacak kadar hafiftir. Satürn'ün halkaları aletleri oldukça ilkel olan eski astronomların aklını karıştırmıştı. Galileo 1610 yılında ilk kez teleskopla Satürn'e baktığında, sanki üçlü bir gezegen sistemiymiş gibi, her iki yanında birer uydu gördüğünü sanarak şaşırmıştı. İki yıl sonraysa uydular görünmez olmuştu.

Satürn'ün en büyük uydusu Titan'dır. Merkür'den daha büyük olan bu uydunun yoğun ve kalın bir atmosferi vardır. Bir uydudan çok küçük bir gezegene benzer. 21.yy.ın başlarında Amerikan Cassini uzay sondasından ayrılacak olan Avrupa yapımı bir sondanın, uydunun atmosferine sokulması planlanıyor.

URANÜS

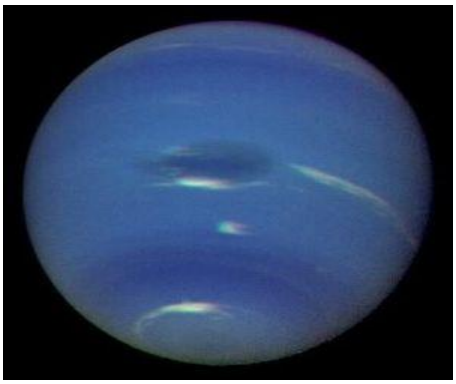
Uranüs, 1781 yılında İngiliz astronom William Herschel tarafından bulundu. Daha önce iki kez gözetlenmiş ama yeni bir gezegen olduğu anlaşılamamıştı. Uranüs'ün güneşten ortalama uzaklığı 2 milyar 869 milyon km.dir. Uranüs, güneş çevresindeki bir dönüşünü 84 yıldan biraz daha uzun bir zamanda tamamlar.

Şekil 2.10 Uranüs

Uranüs güneş çevresindeki yörüngesinde yan yatmış olarak döner, tıpkı yuvarlanan bir varil gibi. Bu nedenle de zaman zaman her iki kutbu da bize doğru döner. Bu garip dönüşe, milyarlarca yıl önce dev bir gök taşının gezegene çarpması neden olmuş olabilir.

Uranüs'ün halkaları 1977 yılında, astronomlar gezegenin arkasından bir yıldızı gözledikleri sırada bulundu. Yıldızın ışığı beklenenden 5 dk. önce sönükleşince yıldızın ışığını engelleyenin bir uydu olabileceği düşünüldü. Aynı şey gezegenin öbür yanında da tekrarlanınca bunun bir halka sistemi sonucu olduğu anlaşıldı.

NEPTÜN



Şekil 2.11 Neptün

PLÜTON

Şekil 2.12 Plüton

2.9.3 KUYRUKLU YILDIZLAR, METEORLAR VE ASTEROİTLER

KUYRUKLU YILDIZLAR

Kuyruklu yıldızlar, Güneş sisteminin oluşum döneminden arda kalmış kayaç ve buz kütleleridir. Gök bilimciler, bu buzlu kayaçların, Hollanda'lı gökbilimci Jan Oort'un adıyla anılan ve Güneş Sisteminin en dışındaki Oort bulutu bölgesinde yer almaktadır.

METORLAR

Gökte kısa bir an için görülen ışık çizgilerinin nedeni meteorlardır. Kuyruklu yıldızlardan kalan kayaç ya da toz parçacıklarının saniyede 70 km. yi bulan hızlarla atmosfere girip yanmaları sonucu oluşurlar. Kuyruklu yıldızlar, yörüngelerinde dönerken kopan parçacıkların atmosfere girip yanmasıyla gökte "meteor yağmuru" denilen görüntü yaratılır.

ASTEROİTLER

Asteroitler, güneş çevresindeki yörüngelerde dönen ve gezegenlerden daha küçük olan gökcisimleridir. Günümüze kadar keşfedilenlerin sayısı 4000'i geçmektedir. Boyları küçük taş parçaları ile yüzlerce km. çaplı kütleler arasında değişir. Asteroitlerin çoğu Mars ile Jüpiter arasında uzanan Asteroit kuşaklarında yer alır. Ancak "Truvalılar" adı verilen, iki grup halinde Jüpiter'in yörüngesini izlerler. Öbürleri güneşin çevresinde dönerler.

2.10 GEZEGENLERİN FİZİKSEL KOŞULLARI

2.10.1 MARSIN FİZİKSEL KOŞULLARI

Mars'ın yarıçapı Dünya'nınkinin yaklaşık yarısı kadardır. Yoğunluğu Dünya'nınkinden daha az olup, hacmi Dünya'nın hacminin %15'i, kütlesi ise Dünya'nınkinin %11'i kadardır. Mars'ın Merkür'den daha büyük ve daha ağır olmasına karşılık, Merkür ondan daha yoğundur. Bu yüzden Merkür'ün yüzeyindeki yerçekimi Mars'ınkinden daha fazladır. Mars, boyutu, kütlesi ve yüzeyindeki yerçekimi bakımından Dünya ile Ay arasında yer alır. Mars yüzeyinin kızıl-turuncu görünümü hematit ya da pas adıyla tanınan demir oksitten (Fe_2O_3) kaynaklanır.

Marsın yer çekimi kuvveti 3.711 m/s^2 (0.376 g) dır. Ekvatoral dönme hızı 868,22 km/h (241,17 m/s) dır.

Fiziksel özellikler	
Ortalama yarıçap	3,389.5 ± 0.2 km[a] [3]
Ekvatoral yarıçap	3,396.2 ± 0.1 km[a] [3]
	0.533 Earths
Kutupsal yarıçap	3,376.2 ± 0.1 km[a] [3]
	0.531 Earths
Basıklık	0.00589 ± 0.00015
Yüzey alanı	144,798,500 km²[4]
	0.284 Earths
Hacim	1.6318 × 10¹¹ km³[5]
	0.151 Dünya
Kütle	6.4171 × 10²³ kg[6]
	0.107 Dünya
Ortalama yoğunluk	3.9335 ± 0.0004 g/cm³[5]
Yüzey kütle çekimi	3.711 m/s²[5]
	0.376 *g*
Atalet momenti faktörü	0.3662 ± 0.0017[7]
Kurtulma hızı	5.027 km/s
Yıldız dönme dönme periyodu	1.025957 d
	Şablon:RA[5]
Ekvatoral dönme hızı	868.22 km/h (241.17 m/s)
Eksen eğikliği	25.19° to its orbital plane[8]
Kuzey kutbu sağ açıklık	Şablon:RA
	317.68143°
Kuzey kutbu dik	52.88650°

Şekil 3.1 Marsın Fiziksel Özellikleri

2.10.2 MERKÜR FİZİKSEL KOŞULLARI

Merkür Güneş Sistemi'ndeki en küçük ve Güneş'e en yakın gezegen. Yaklaşık 88 Dünya gününe eşit yörünge süresi ile Güneş Sistemi'ndeki diğer gezegenlerden daha hızlıdır. Dünya'dan bakıldığında, kendi yörüngesi etrafında 116 günde hareket ettiği görünür. Bilinen hiç doğal uydusu yoktur. Adını tanrıların habercisi Roma tanrısı Merkür'den alır.

Neredeyse ısıyı koruyacak bir atmosferi olmamasından dolayı, Merkür'ün yüzey sıcaklığı Güneş Sistemi'ndeki diğer tüm gezegenlerden daha fazla günlük olarak değişir.

Bazı ekvatoral bölgelerde gece 100 K (−173 °C; −280 °F)'den gündüz 700 K (427 °C; 800 °F)'e kadar değişir. Kutuplar sürekli olarak 180 K (−93 °C; −136 °F)'in altındadır.

Merkür'ün ekseni Güneş Sistemi'ndeki en küçük (yaklaşık derecenin 1⁄30'i) eksen eğikliğidir. Ancak Merkür'ün dış merkezliği Güneş Sistemi'ndeki bilinen tüm gezegenlerinkinden büyüktür.[a] Günötede Güneş'e günberide olduğundan 1.5 kat daha uzaktır. Merkür'ün yüzeyi aşırı derecede kraterlidir ve görünüm olarak Ay'a benzer. Bu milyarlarca yıldır jeolojik olarak etkin olmadığını gösterir.

Merkür, Güneş ile 3:2 rezonansta gelgitsel ya da çekimsel kilitlidir ve Güneş Sistemi'nde eşsiz bir yörüngede döner. Duran yıldızlara göre, Güneş etrafındaki her iki devrine karşılık kendi ekseninde tam olarak üç defa döndüğü görülür. Güneş'ten görüldüğü gibi yörünge hareketi ile dönen bir gözlemci çerçevesi içerisinde, sadece her iki Merkür yılında bir dönüyormuş gibi görünür.

Marsın yer çekimi kuvveti 3.701 m/s² (0.377 g) dir. Ekvatoral dönme hızı 10.892 km/h dir.

Fiziksel özellikleri			
Ekvatoral çap	4879.4 km (0.383 Dünya çapı)		
Yüzey alanı	7.5×10^{7} km² (0.147 Dünya yüzeyi)		
Hacim	6.083×10^{10} km³ (0.056 Dünya hacmi)		
Kütle	3.302×10^{23} kg (0.055 Dünya kütlesi)		
Ana özkütle	5.427 g/cm³		
Ekvatoral yerçekimi	3.701 m/s² (0.377 g)		
Kaçış hızı	4.435 km/s		
Dönme periyodu	58.6462 gün (58 gün 15.5088 h)		
Dönme hızı	10.892 km/h (ekvatorda)		
Eksen eğikliği	~0.01°		
Kuzey kutbunun bahar açısı	281.01° (18 h 44 d 2 s) [1]		
Dik açıklık	61.45°		
Albedo	0.10-0.12		
Yüzey sıcaklığı	min	ana	max
	90 K	440 K	700 K
Ort. yüzey sıcaklığı: Gündüz	623 K		
Ort. yüzey sıcaklığı: Gece	103 K		

Şekil 3.2 Merkür Fiziksel Özellikleri

2.10.3 VENÜS FİZİKSEL ÖZELLİKLERİ

Venüs, Güneş Sisteminde, Güneş'e uzaklık bakımından ikinci sıradaki, sıcaklık bakımından da birinci sıradaki gezegen.

Güneşe uzaklık bakımından ikinci sırada olmasına rağmen en sıcak gezegen olmasının nedeni de atmosferinin gelen güneş ışınlarının dışarı çıkmasına izin vermemesidir. Hatta bazı kişiler eskiden Dünya gibi üzerinde canlıların yaşadığı yeşil bir gezegen olduğunu da söylerlerdi. Ayrıca Zühre, Çolpan veya Çoban Yıldızı olarak da bilinir. Bu gezegen adını Eski Roma tanrıçası Venüs (Eski Yunan Mitolojisinde Afrodit)'ten almıştır. Kendi ekseni etrafında, Güneş Sistemindeki diğer tüm gezegenlerin aksi istikamette döner. Güneş etrafındaki dönüşünü 224.7 Dünya gününde tamamlar.

Büyüklüğü açısından Dünya ile benzerlik gösterdiğinden Dünya ile kardeş gezegen veya dünyanın ikizi olarak da bilinmektedir. Gökyüzünde Güneş'e yakın konumda bulunduğundan ve yörüngesi Dünya'nınkine göre Güneş'e daha yakın olduğundan yeryüzünden sadece Güneş doğmadan önce veya battıktan sonra görülebilir. Bu yüzden Venüs Akşam Yıldızı, Sabah Yıldızı veya Tan Yıldızı olarak da isimlendirilir. Çoban Yıldızı da denmektedir. Görülebildiği zamanlar, gökyüzündeki en parlak cisim olarak dikkat çeker.

Venüs yer çekimi kuvveti 8,87 m/s² (0,91g) dir. Ekvatoral dönme hızı 868,22 km/h (241,17 m/s) dir.

Fiziksel Özellikler	
Ekvator Bilinen yarıçapı	6,051.8 ± 1.0 km km (0.949 9 x Yer)
Basıklık	0
Hacim	9.28×10^{11} km³ 0.857 Yer
Kütle	$4.868\ 5\times10^{24}$ kg 0.815 Yer
Yoğunluk	5,243 g/cm³ (0,95 x Yer)
Eksen eğikliği	177,36° (ters dönüş)
Dönme süresi	−243.018 5 gün (ters yönde)
Yerçekimi	8,87 m/s² (0,91 x Yer)
Kurtulma hızı	10,36 km/saniye (0,93 x Yer)
Albedo	0.67 (geometric) 0.90 (Bond)
Yüzey sıcaklığı ortalama	735 K (464 °C)
Atmosfer Özellikleri	
Yüzey Basıncı	93 bar (9.3 Pascal)
Atmosfer	~%96.5 Karbon Dioksit ~%3.5 Nitrojen

Şekil 3.3 Venüs Fiziksel Özellikleri

2.11 FİZİKSEL KOŞULLARIN PROGRAMA UYGULANMASI

Ekran içerisinde hareket eden nesnelere oyun içerisinde gerçekçi bir deneyim katmak için nesnelere yer çekimi özelliği veririz.

Bunu sağlamak için yeni bir fonksiyon oluşturulur ve 3 parametre alır. İlk parametre yer çekiminin kuvveti, ikinci parametre oyun içerisinde hangi ekrana uygulanacak ise o verilir. Üçüncü parametre ise yer çekimi verilmek istenen nesne verilir.

Fonksiyonun genel biçimi aşağıdaki şekildedir.

```
void applyDisplay (int yer çekimi, Rect ekran, Rect nesne)
```

2.12. ANDROİD'DE SOCKET PROGRAMLAMA

Kullanıcılar birlikte hikâye modu oynarken birbirlerinin konumuna ve neye ateş edip yok ettiği bilgisine ihtiyaç duyarlar. Birbirleri arasında veri alışverişini sağlamak için android studio'da socket kütüphaneleri vardır.

Socket programlama temel yaklaşım client-server mantığıdır. Burada client diğer clientlara bir şey göndermek istediği zaman bunu server üzerinde yapar. Hiçbir client diğer clientlar'ın ip adresini bilmediği için aradaki bilgi alışverişi server sayesinde olur.

Soketler, bir tür süreçler arası haberleşme (interprocessing) yöntemidir. Soket, soyut bir tanımla haberleşme uç noktalarıdır. Pratik olarak soketler dosyalara benzer. Soketten okumak ile dosyadan okumak arasında hiçbir fark yoktur. Aklınıza gelebilecek hemen her internet programı socket program olarak çalışır.

Android uygulamada Soket programlama yapmak için Socket.IO kütüphanesini kullanılır.

Client tarafından servera gönderilecek bilgi konum bilgisi ve ateş ettiği yer. Timer ile otomatik olarak bu bilgiler servera gönderilir. Otomatik olarak gönderilmesi gerektiği için Timer kulanılır ve belirlenen her periyotta clientlar, servera durum bilgilerini gönderir.

2.12.1 SOKET PROGRAMLAMA NEDİR

Soket, bilgisayar ağı programlamanın en temel teknolojilerinden biridir. Soketler, ağ yazılım uygulamalarının, ağ donanımında ve işletim sistemlerinde yerleşik standart mekanizmaları kullanarak iletişim kurmasını sağlar.

Soket, tam olarak iki yazılım arasındaki tek bir bağlantıyı (noktadan-noktaya bağlantı olarak adlandırılır) temsil eder. İkiden fazla yazılım, çoklu socket kullanarak istemci / sunucu veya dağıtılmış sistemler ile iletişim kurabilir. Örneğin, birçok Web tarayıcısı, sunucu üzerinde yapılan socket aracılığıyla tek bir Web sunucusu ile aynı anda iletişim kurabilir.

Soket tabanlı yazılım genellikle ağdaki iki ayrı bilgisayarda çalışır, ancak soketler aynı bilgisayarda yerel olarak (süreçler arası) iletişim kurmak için de kullanılabilir. Soketler çift yönlüdür, yani bağlantı taraflarından birinde hem veri gönderip hem de alma yeteneği vardır. Bazen iletişimi başlatan bir uygulama "istemci" ve diğer uygulama "sunucu" olarak

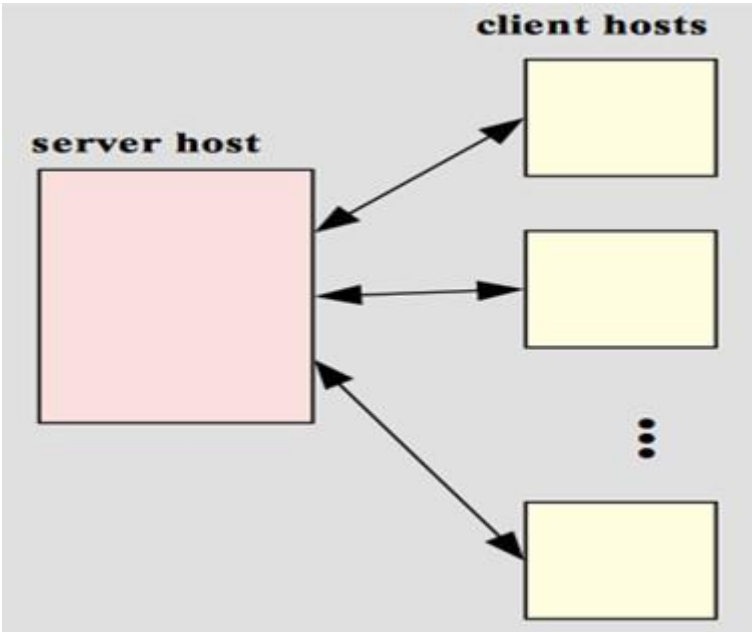
adlandırılır, ancak bu terminoloji, akran ağlarına karışıklığa neden olmakta ve genellikle önlenmelidir.

2.12.2 İSTEMCİ SUNUCU

Sunucu terimi ağa bağlı diğer istemci bilgisayarlara bilgi ve hizmet sağlayan bir yazılım uygulamasını çalıştıran konak bilgisayarı ifade eder.

İstemci birisinin sunucuda bulunan bilgilere erişmek için kullandığı bilgisayar uygulamasına verilen addır. Web tarayıcı güzel bir istemci örneğidir.

İstemci sunucu sistemlerin en önemli özelliği istemcinin sunucuya bir istek göndermesi ve sunucunun istemciye uygun bilgiyi yanıt vermesidir. Web tarayıcısı ve web sunucusu bileşimi büyük olasılıkla en yaygın kullanılan istemci sunucu sistemi örneğidir.



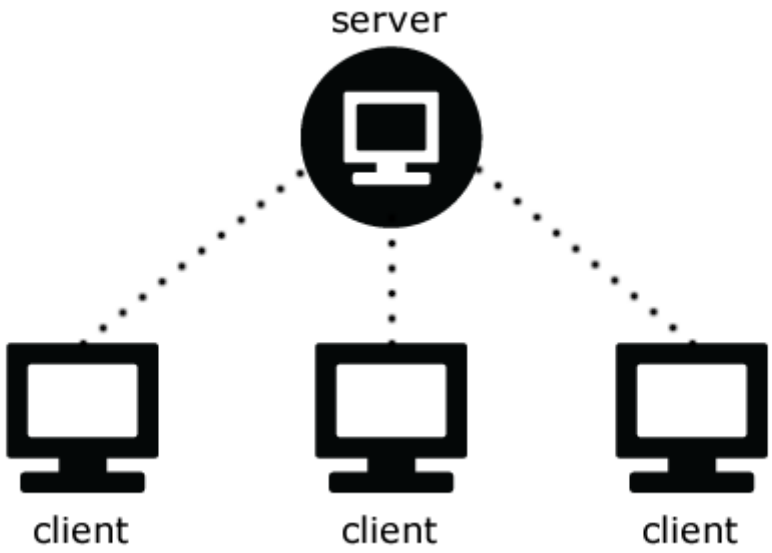
Şekil 4.1 İstemci – Sunucu

2.12.3 İSTEMCİ

Bir istemci, istemci / sunucu ilişkisindeki istekte bulunan program veya kullanıcıdır. İstemci genellikle bir ağ üzerinden erişilebilen başka bir sistem veya bilgisayarda bulunur. Bu terim ilk önce kendi programlarını çalıştıramayan cihazlar için kullanıldı ve bir ağ üzerinden olabilecek uzak bilgisayarlara bağlanıldı. Buna göre hizmet veya kaynağı sunan

bir sunucu bulunmakta ve istemciler bu sunucuya bağlanarak bu hizmetten faydalanmaktadır.

- Direkt olarak bir kullanıcı tarafından davet edilir ve yalnızca tek bir oturum için çalışır.
- Bir kullanıcının bilgisayarında veya cihazında yerel olarak çalışır.
- Bir sunucu ile iletişimi aktif olarak başlatır.
- Çok güçlü donanıma ihtiyaç duymaz.



Şekil 4.2 İstemci Modeli

2.12.4 SUNUCU

Teknik olarak ele alındığında sunucu; verilerin, bilgisayar ağları üzerinden kullanıcıların erişimine açık olarak barındırıldıkları bilgisayar sistemlerine verilen isim. Daha basit bir ifadeyle sunucu, kullanıcının verilere online olarak ulaşabilmesini sağlayan, bu verileri onların kullanımına sunan sistemlerdir.

Sunucu denilince genellikle akla ilk olarak bir donanım bileşeni gelir ama bir sunucu sadece fiziksel bir cihazdan oluşmaz. Bu cihazlar, bir sunucu olarak çalışabilmeleri için özel yazılımlara ihtiyaç duyarlar. Bu yazılımlar da sunucu cihazlarının içerdikleri verileri, kullanım amacına en uygun şekilde sunabilmelerini sağlar. Örneğin bir oyun sunucusu ile

bir e-posta sunucusunda bulunan temel yazılımlar benzerlik gösterse de, kullanım amacını doğru şekilde yerine getirmesini sağlayan farklı ve özel yazılımlara sahiptirler ve bu amaca göre optimize edilmişlerdir.

Bir sunucunun en önemli özelliği veri akışını güvenli, kesintisiz ve istikrarlı bir performans ile sağlayabilmesidir. Bu yüzden her sunucu donanımsal olarak gelişmiş özelliklere sahiptir ve çoğu zaman ek donanımlar ile güçlendirilir. Örneğin olası bir enerji sorununa karşı kesintisiz güç kaynağı yapısı, durmadan çalışan makinelerin aşırı ısınmasına karşı özel soğutma yapıları, verilerin yüksek hızda ve hatasız bir şekilde istemcilere aktarılabilmesi veya gerektiğinde yedeklenebilmesi için özel kablolar, büyük sunucu sistemlerinin önemli ek birimleri arasında sayılabilir.

Şekil 4.3 Sunucu Modeli

2.12.5 2.12.5 İSTEMCİ SUNUCU MİMARİSİ

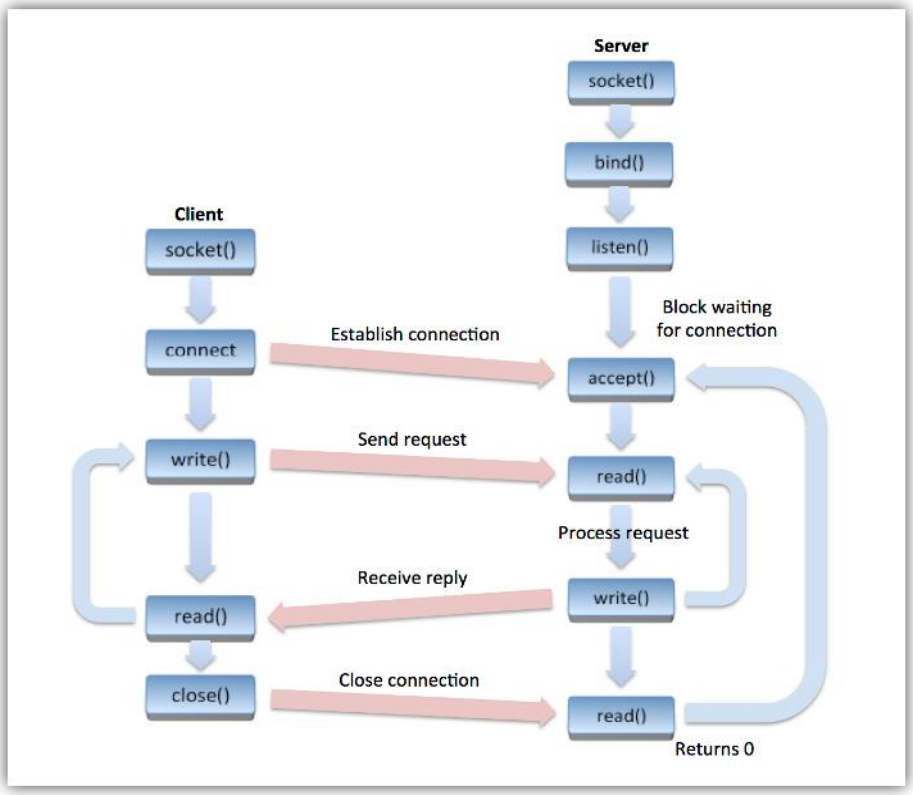
İstemci-sunucu mimarisi, bilgisayar ağı ile birbirine bağlı iki yazılımın biri birinden hizmet almasını amaçlar. Bu programlardan sunucu yazılım hizmet verirken, istemci yazılım ise hizmet alan taraftır. Sunucunun verdiği hizmet, zamanı söylemek kadar basit olabileceği gibi dosya aktarımı ya da e-posta göndermek gibi daha karmaşık servisler de olabilir. İstemcinin bu hizmetleri alabilmesi için önce sunucu ile bağlantı kurması ve daha sonra bağlantıyı kullanarak servise erişmesi gerekir. Soket, bilgisayar ağı üzerinden bu bağlantının kurulması ve servise ulaşılması için gerekli olan yazılım alt yapısını sunar. Soket, işletim sistemlerince desteklenen, farklı programlama dillerinden uzaktan ulaşabileceğimiz bir yazılım erişim noktasıdır. Soket programlama modeli istemci-sunucu mimarisi ile uyumludur.

Sunucu tarafta Soket haberleşmenin adımları:

Servisi duyur
Bağlantı için bekle
Bağlanan istemci için servis yap
İstemci ile bağlantıyı kopar
İkinci adıma geri dön

İstemci tarafta Soket haberleşmenin adımları:

Servisin sunucusunu bul
Sunucu ile bağlantı kur
Servis için istek gönder
Sunucuya ile olan bağlantıyı kopar



Şekil 4.4 İstemci Sunucu Mimarisi

2.12.6 TCP/IP NEDİR

TCP/IP (Transmission Control Protocol/Internet Protocol), bilgisayarlar ile veri iletme/alma birimleri arasında organizasyonu sağlayan, böylece bir yerden diğerine veri iletişimini olanaklı kılan pek çok veri iletişim protokolüne verilen genel addır. (Yani, TCP/IP protokolleri bilgisayarlar arası veri iletişiminin kurallarını koyar). Bu protokollere örnek olarak, dosya alma/gönderme protokolü FTP (File Transfer Protocol), Elektronik posta iletişim protokolü SMTP (Simple Mail Transfer Protocol), TELNET protokolü (Internet üzerindeki başka bir bilgisayarda etkileşimli çalışma için geliştirilen *login* protokolü) verilebilir. Adını sıkça duyduğumuz WWW ortamında birbirine link objelerin iletilmesini sağlayan protokol ise Hyper Text Transfer Protocol (HTTP) olarak adlandırılmaktadır.

TCP/IP protokolü aynı zamanda, diğer iletişim ağlarında da kullanılabilir. Özellikle pek çok farklı tipte bilgisayarı veya iş istasyonlarını birbirine bağlayan yerel ağlarda (LAN) kullanımı yaygındır.

2.12.7 UDP NEDİR

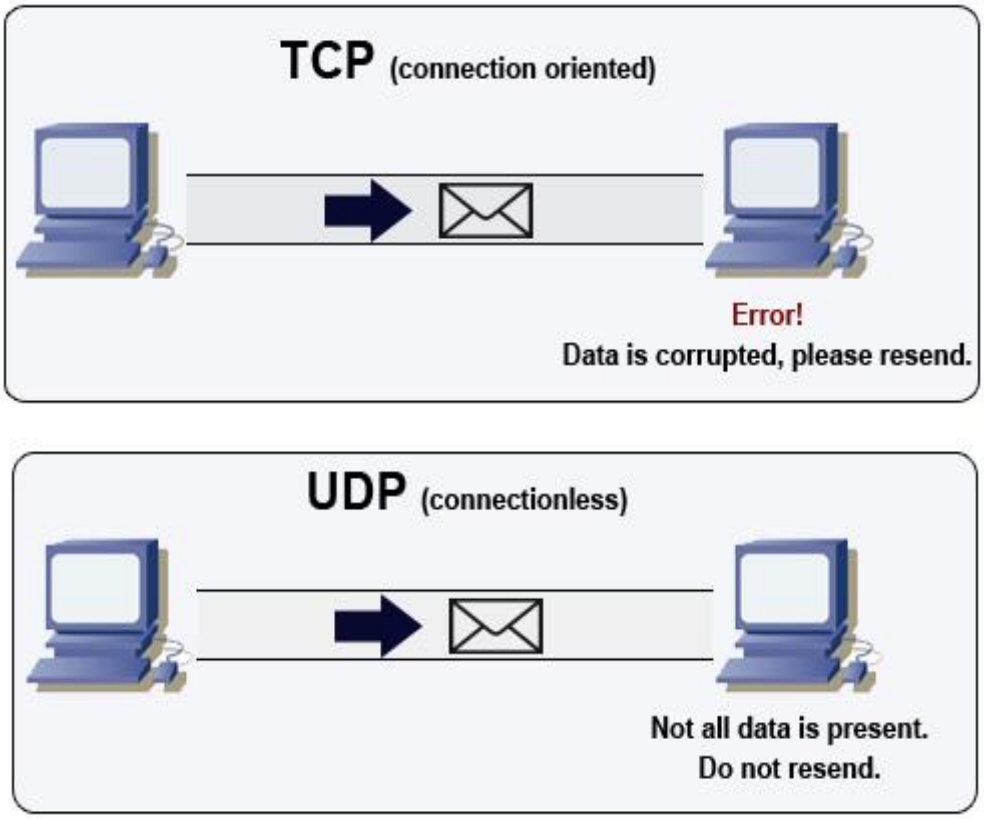
DP (User Datagram Protocol – Kullanıcı Veribloğu İletişim Kuralları), TCP/IP protokol takımının iki aktarım katmanı protokolünden birisidir. Verileri bağlantı kurmadan yollar.

Gelişmiş bilgisayar ağlarında paket anahtarlı bilgisayar iletişiminde bir datagram modu oluşturabilmek için UDP protokolü yazılmıştır. Bu protokol minimum protokol mekanizmasıyla bir uygulama programından diğerine mesaj göndermek için bir prosedür içerir. Bu protokol ‘transaction’ yönlendirmelidir.

Paketin teslim garantisini isteyen uygulamalar TCP protokolünü kullanır.
Geniş alan ağlarında (WAN) ses ve görüntü aktarımı gibi gerçek zamanlı veri aktarımlarında UDP kullanılır.

UDP bağlantı kurulum işlemlerini, akış kontrolü ve tekrar iletim işlemlerini yapmayarak veri iletim süresini en aza indirir. UDP ve TCP aynı iletişim yolunu kullandıklarında UDP ile yapılan geçek zamanlı veri transferinin servis kalitesi TCP’nin oluşturduğu yüksek veri trafiği nedeniyle azalır.

UDP'yi kullanan protokollerden bazıları DNS, TFTP, ve SNMP protokolleridir. Uygulama programcıları birçok zaman UDP'yi TCP'ye tercih eder, zira UDP ağ üzerinde fazla bant genişliği kaplamaz. UDP güvenilir olmayan bir aktarım protokolüdür. Ağ üzerinden paketi gönderir ama gidip gitmediğini takip etmez ve paketin yerine ulaşıp ulaşmayacağına onay verme yetkisi yoktur. UDP üzerinden güvenilir şekilde veri göndermek isteyen bir uygulama bunu kendi yöntemleriyle yapmak zorundadır.



Şekil 4.5 TCP – UDP

2.13. YAPAY ZEKA NEDİR

Yapay zekâ, bilgisayarın veya bilgisayar kontrolündeki bir robotun çeşitli faaliyetleri zeki canlılara benzer şekilde yerine getirme kabiliyetidir. Yapay zekâ çalışmaları genellikle insanın düşünme yöntemlerini analiz ederek, bunların benzeri yapay yönergeleri geliştirmesine yöneliktir. Yani bilgisayarın, insanlar tarafından gerçekleştirilen görevleri yerine getirmesini sağlar. Başka bir deyişle, Yapay zekâ bilgisayarın insanlar gibi düşünmesini sağlar. Makinelerin karmaşık sorunları insana benzer şekilde çözmesine

yardımcı olur. Zekâ ve akıl gerektiren sorunlar artık bilgisayarları kullanarak etkili bir şekilde çözülebilir.

Yapay zekâ temelde bilgisayar bilimi ile ilişkili olmasına rağmen tarım, tıp, matematik ve biyoloji gibi farklı alanlarda sayısız uygulama için yararlıdır.

Yapay zekâ programları karmaşık verilerdeki kalıpları tanımak, tecrübelerinden faydalanmak ve insanlar tarafından alınan kararları almak için insan bilgisine dayanmaktadır.

Yapay zekâ sistemleri bir şeyler gözlemlemekte ve daha sonra önceden belirlenmiş parametreler temelinde onu tanımaya çalışmaktadır. Dolayısıyla, belirli bir duruma göre yapay zekâ sistemleri, sorunu çözmek için görev yapmakta ve buna tepki vermektedir.

Projenin yapay zekâ kısmında kullanıcı yapay zekaya karşı oynarken, karşıda oynadığı sistemin yapay zekâ olabilmesi için seçim yapması gerekmektedir. Bu yüzden yapay zekaya karşı oynayan kullanıcının yanında ek uzay gemileri olacak ve yapay zekâ olası durumdan en iyi durumu çıkarabilsin.

2.13.1 YAPAY ZEKANIN FAYDALARI

Hata şansı neredeyse sıfırdır. Uzay araştırmalarında alanı keşfetmek için akıllı robotlar kullanılabilir. Bunlar makine olduğu için gezegensel atmosferlere fiziksel durumlarını ve işlevlerini etkilemeyecek şekilde adapte olabilirler.

Dünyanın en dip noktalarına ulaşmak için akıllı robotlar programlanabilir. Yeraltı madenlerine ulaşmak için akıllı robotları kullanmak zaman ve para konusunda ciddi faydalar sağlayabilir. Bu makineler, insanların sahip olduğu sınırlamaların üstesinden gelmek için kullanılabilir.

Akıllı telefonlar yapay zekâ uygulamasına mükemmel bir örnektir. Kullanıcının yazdıklarının ne olacağını tahmin etmede (bunu mu demek istemiştiniz gibi) ve yazım hatalarını düzeltme gibi konularda, makineler çok faydalıdır. Kişisel asistanlık görevini gören Siri, kullanıcılara en iyi veya en kısa güzergahları veren GPS ve Harita uygulamaları, trafik ve zaman tahmini gibi yapay zekâ uygulamaları yapay zekanın en iyi örneklerindendir.

2.13.2 YAPAY ZEKANIN ZARARLARI

Bakım ve onarımları maliyetlidir. Programların değişen gereksinimlere göre güncellenmesi gerekmekte ve makinelerin daha akıllı hale getirilmesi gerekmektedir. Arıza durumunda tamir masrafları çok yüksek olabilir. Kaybedilen kod veya verilerin geri yüklenmesine ilişkin prosedürler zaman kaybına yol açabilir ve masraflı olabilir.

Yapay zekanın uygulanmasına ilişkin önemli bir husus etik ve ahlaki değerlerle ilgilidir. İnsanoğlunun kopyalarını yaratmak etik olarak doğru olur mudur?

Makineler muazzam miktarda veriyi depolayabilir, ancak depolama, erişim ve geri alma, insan beyninde olduğu kadar etkili değildir. Uzun süreler boyunca tekrarlayan görevleri yerine getirebilirler, ancak insanlar gibi tecrübe ederek daha iyi hale gelmezler.

İnsanları değiştiren makine fikri harika görünüyor. Bizi tüm acılardan kurtaracak gibi görünüyor. Yapay zekâ dünyasında, tamamen yürekten, aidiyet duygusuyla ve özveriyle çalışmak gibi insani özelliklere yer yok. Hastanelerde çalışan robotları düşünün. Onların insanların yapacakları ilgiyi ve endişeyi gösterebilir mi? Çevrimiçi asistanlar bir insanın yapacağı hizmeti verebilir mi?

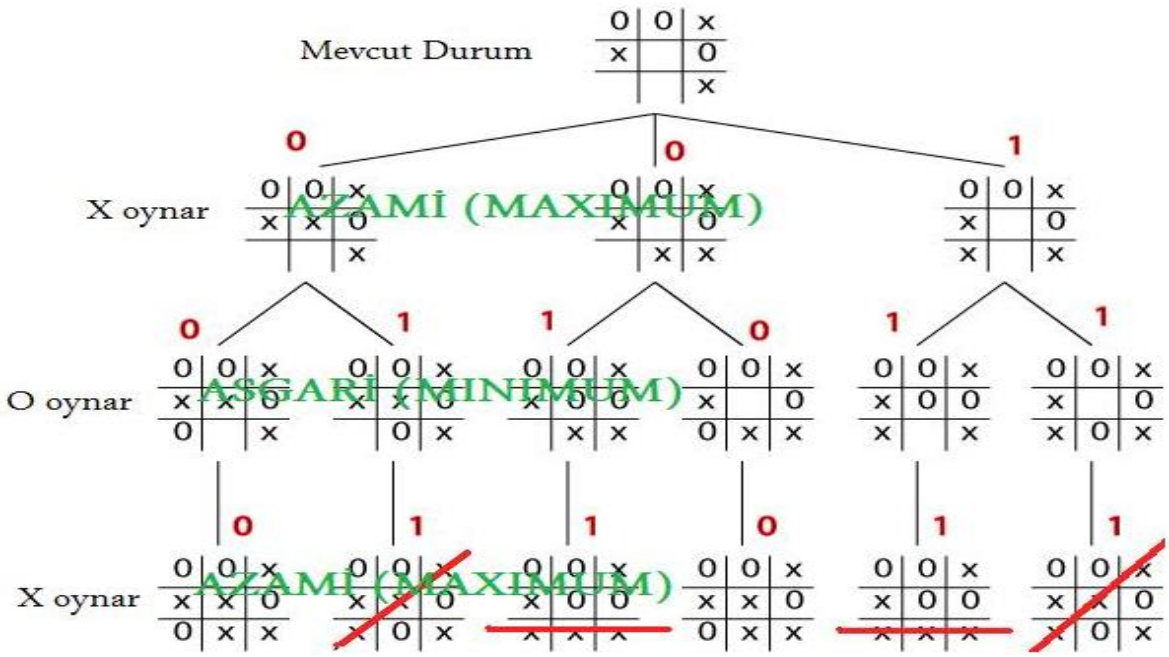
Yaratıcı alanlarda istihdam edilen akıllı makineleri düşünün. Sizce robotlar, insan zihnini yaratıcı düşünce veya özgünlük içinde mükemmelleştirebilir veya insan yaratıcılığıyla rekabet edebilir mi? İnsanlar duygusal yaratıklardır; düşünüp hissedebilirler. Duyguları düşüncelerini yönlendirir. Makinelerde ise durum böyle değil. İnsanın sahip olduğu sezgisel yetenekler, insanların önceki bilgilere dayanarak yargılayabilecekleri şekilde, sahip oldukları yetenekler makineler tarafından kopyalanamaz.

2.14 ALFA BETA BUDAMA

Bilgisayar bilimlerinde özellikle yapay zekâ ve karar mekanizmalarının uygulanmasında çok kullanılan bir ağaç dolaşma algoritmasıdır. İsmindeki budama da bu ağaç üzerindeki bazı dalları kesmesinden gelmektedir.

Şekildeki ağacın kökünde olan mevcut durumda yapılabilecek bütün ihtimaller gösterilmiştir. Bu ağaç bir tictactoe oyunu için tasarlandığından ve oyundaki hamle sayısı sınırlı olduğundan bilgisayarın hesaplaması açısından sorun oluşturmaz. Ancak hamle sayısının çok fazla olduğu satranç veya GO gibi oyunlar düşünülürse bütün hamlelerin

hesaplanması bilgisayarlar için (günümüz teknolojisinde hâlâ) imkansızdır. Bu durumda hesaplamayı kolaylaştırmak için bu ağaç üzerinde bir iyileştirilmeye gidilmesi gerekir.



Şekil 5.1 Alfa Beta Budama

İşte tam bu noktada alfa beta budaması devreye girer. Ağacın bazı dallarının işlenmesi ve hamlelerinin sonucunun hesaplanması gereksizdir. Alfa beta budaması bu dalları tespit ederek budar. Yani bu dalları ve altlarını hesaplamayarak performans artışı hedeflenir.

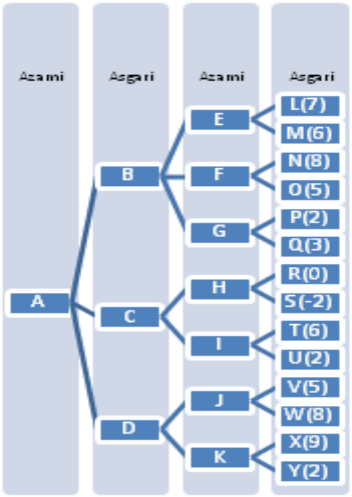
Klasik minimax ağacı işlenirken iki adımlı arama yapılır. Birinci adıma ağaçtaki düğümlere (nodes) sezgisel (heuristic) değerler atanır. İkinci adımda ise en derinden başlanarak bu değerler yukarı doğru işletilir. En sonunda mevcut konumda yapılacak hamleye karar verilir.

Bu anlamda alfa beta araması olarak isimlendirebileceğimiz yeni arama algoritmamız derin öncelikli aramadır (depth first search). Öncelikle alfa ve beta isminde iki değer belirlenir. Burada alfa ile anlatılan o ana kadar ulaşılan azami değerdir. Yani alfa değeri hiçbir zaman azalamaz hep artar, buna mukabil beta değeri ise anlık asgari değerdir. Beta değeri de hiçbir zaman artamaz hep azalır.

İşte tam bu noktada elimizde alfa ve beta değerlerini bulunduruyorsak artık budama yapabiliriz. Örneğin ağacın herhangi bir düğümündeyken elimizde alfa değeri olarak 16 bulunsun. Bu değerden büyük olan düğümlere bakılmasına gerek yoktur çünkü o ana kadar olan en büyük değer zaten elimizdedir, bu değerden büyük olan değerleri taşıyan alt ağaçlar sonucu etkilemeyecektir. Öyleyse elimizdeki alfa değerinden büyük olan dalları budayabiliriz. Bunun ismi alfa budamasıdır.

Benzer şekilde beta değeri elimizdeki en küçük anlık değeri tutacaktır. Bu değerden küçük olan dalların dolaşılması da anlamsızdır ve budanabilir. Bunun ismi de beta budamasıdır. Yani budanan dal, beta'nın altına kalması veya alfa'nın üstünde olmasına göre beta veya alfa budaması ismini alır.

Örneğin şekildeki ağacı ele alalım:



Şekil 5.2 Alfa Beta Budaması Örnek

Yukarıdaki şekil üzerinde, A kök düğümünden başlanarak minimax dolaşmamızı gerçekleştirelim. Öncelikle alfa ve beta değerlerini atayalım. Örneğin alfa değeri olarak –¥ ve beta değeri olarak ¥ aldığımızı düşünelim. Bu durumda yukarıdaki ağacın dolaşılması sırasında alfa ve beta değerleri güncellenerek dolaşılacaktır.

İlk olarak kök A'dan B -> E -> L şeklinde en derine indiğimizi düşünelim (depth first search).Bulduğumuz değer 7 olacak ve dolayısıyla asgari değerin arandığı bu seviyede L ve

M düğümleri (nodes) için en küçük değer 6 olacaktır (6 < 7) dolayısıyla bu iki düğümün taranmasından sonra E düğümü için 6 bulunacaktır. (bir üst seviyeye asgari değer çıkarılıyor)

Benzer şekilde normal bir minimax ağaç dolaşması sırasında F için 5 ve G için 2 bulunacaktır. Ancak E düğümünün altındaki düğümler taranırken alfa ve beta değerleri güncellenir. Alfa değeri olarak 6 ve beta değeri olarak 7 bulunacaktır.

Şimdi ağacın yukarı doğru bütün düğümlerinde bu durumu düşünebiliriz. Yani E düğümü için en küçük 6 en büyük 7 olurken B düğümü için en büyük 7 ve en küçük 6 ve A düğümü için en küçük 6 en büyük 7 olacaktır.

Burada dikkat edilecek husus, yukarı doğru her seviyede alfa ve betanın yer değiştirmesidir. Çünkü her seviyede aranan değer değişmektedir, örneğin en alt seviyede asgari değer aranırken bir üstünde azami bir üstünde asgari … şeklinde gitmektedir.

E-F-G düğümleri arasından en büyük değeri alacağımızı biliyoruz. Ayrıca L-M-N-O-P-Q düğümlerinden de en küçüğünü alacağımızı biliyoruz. O halde ağaç dolaşılırken LMNOPQ düğümlerinin asgarisi, bir önceki daldakinden daha büyükse o düğümden geri kalanlar budanabilir.

Ağaçta kaldığımız noktadan devam ederek budama işleminin nasıl yapıldığına bakalım. F düğümünün altında 8 ve 5 değerlerine bakılacak, küçük olan 5 dönecek ve alfa ve beta değerlerinde bir değişiklik olmayacaktır. Çünkü 5 değeri bir üst seviyede önemsizdir. Bir üst seviyede azami değeri alacağımız için ve E düğümünde zaten 6 olduğu için bu seviyede 5 anlamsızdır.

Dolaşmaya G düğümü ile devam edelim. P düğümüne ulaştığımızda 2 değerini göreceğiz.

İşte tam bu noktada artık geri kalan düğümleri budayabiliriz. Çünkü 2 değeri elimizdeki alfa değerinden küçüktür. Diğer düğümlerin budanmasının (yani Q düğümüne hiç bakılmamasının) anlamını berber inceleyelim.

G düğümünün altındaki en küçük değer 2 (veya daha küçük de olabilir) ancak biz 2 bulunca geri kalanları buduyoruz çünkü her halükârda yukarı (yani G düğümü seviyesine) 2 veya daha küçük bir sayı çıkacaktır. Çıkan bu sayı ise G düğümü seviyesinde anlamsız olacaktır çünkü bu seviyede en büyük değer aranacaktır. En büyük değer zaten 6 olarak E düğümünde bulunmuştu. Dolayısıyla E düğümünde 6 olduğu bir durumda diğer düğümlerin altındaki en küçük düğümün önemi yoktur.

Dolaşmaya H düğümü ile devam edelim. Altındaki düğümlerden en küçüğünü alalım (0 ve -2'den küçük olanı) -2, H değeri olarak alınır.

Benzer şekilde I değeri T ve U düğümlerinden küçük olanı olarak alınır yani I değeri olarak da 2 bulunur. Ardından C Değeri olarak H ve I düğümlerinden büyük olan yani 2 alınır.

Bu seviyede yani BCD seviyesinde B değerini 6 ve C değerini 2 olarak bulduk şimdi sıra D düğümünde. D düğümünün altındaki J ve J'nin de altındaki V düğümüne derin öncelikli arama gereği bakıldığı anda D düğümü artık kökten budanabilir. Sebebi ise şayet V değeri olan 5 yukarıdaki J düğümüne atanırsa (ki bu seviyedeki en küçük değer olarak kabul edelim, daha küçük bir değer olursa daha da kesin olarak budama yaparız) ve J düğümünün değeri olarak da K düğümünden büyük olduğunu kabul edersek (K'nın daha büyük olması halinde budama yine daha kesin olur) K'nın üstündeki D düğümüne değer olarak 5 gelir ve 5 değeri, daha önce C düğümünün değeri olarak hesapladığımız 2'den büyük olduğu için zaten bir üst seviye olan A düğümüne çıkamaz. Dolayısıyla D düğümü ve altındakilerin hesaplanması gereksizdir.

Yukarıdaki örnekte Q ve W düğümleri budanırken alfa budaması (alpha pruning) ve K düğümü budanırken de beta budaması (beta prunning) kullanılmış olunur. Kısaca alfa> beta olduğu durumlarda budama yapılır.

2.15 NESNELERE KESİŞİM TESTİ UYGULAMAK

Yıldız savaşları oyununda kullanıcılar birbirleriyle oynarken ateş ettikleri zaman ateşlerinin karşı düşman ile temas haline girdiği zaman bir test yapılması gerekmektedir. Kesişim testi sağlanırsa eğer gerekli işlemlerin yapılması gerekmektedir.

Kesişim testini sağlamak için android de Rectangle sınıfı kullanılır kullanım şeklinde yeni bir Rectangel nesnesi oluşturulur ve oluşturulan Rectangel nesnesine kendi karakterimizin ve düşman karakterin eklenir. Kullanım şekli aşağıdaki gibidir.

```
Rect rectangle_nesnesine verilecek isimi = new Rect();
testi_yapmak_istedigimiz_karakter .getHitRect(Rectangle_nesnesinin ismi);
```

Yukardaki şekilde nesnemizi Rectangle nesnesine eklendi. Diger karakter içinde aynı işlemler yapılır ve kesişim testi kontol edilir.

```
if (Rect.intersects(benim_ateşim, Düsman_Karakter))
```

Yukardaki if bloğunun içinde gerekli işlemler yapılır. Yukarıdaki if bloğu karakterimiz ateş ettiği zaman düşman karakterle kesişip kesişmediği kontrolünü yapar.

3. SONUÇLAR

Bu projede android ortamında nasıl oyun yapılabileceği, socket programlama, dinamik olarak nesne nasıl yaratılır. Nesneye nasıl ateş ettirilir, nesnenin ekran içerisinde hareketi nasıl sağlanır. Yapay zekâ nedir ve hangi algoritmayı kullanmanın daha verimli olacağı görüldü.

Android Studio da oyun motoru kullanmadan nasıl oyun yapılır öğrenilmiş olundu.

Yaptığım yıldız savaşları oyunu projesi ile birlikte android ile yapılabilecek oyunları ve ne kadar ilerletilebileceğini görmüş oldum.

Programlama aracı olarak Android Studio kullanıldı. İnternetteki bol bilgi çeşitliliği ile öğrenmek ve uygulamak programlamayı daha da kolaylaştırdı.

Tasarım projesi aldığımız programlama ağırlıklı derslerimizi proje üzerinde pratiğe dökmek açısından oldukça çok faydalı olmuştur ve ilk android de oyun projem olarak güzel bir başlangıç şansı vermiştir.

4. ÖNERİLER

Projede ilk adım olarak istenen şeyler iyi bir şekilde analiz edilmesi gerekiyor. Oyun motorları oyun geliştirmeyi kolaylaştırmaktadır fakat bir programcı oyunun arka planında neler olduğunu görmesi gerekmektedir.

Projenin detayları iyi bir şekilde anlaşıldıktan sonra projenin modüllere bölünmesi çok önemlidir. Bu sayede programcı hem ne yapacağını iyi bir şekilde belirler hem de zaman sıkıntısı çekmemiş olur.

Android ortamında oyun geliştirirken görülmüş olduğu üzere, türkçe kaynak yok denebilecek kadar azdır. Bu yüzden her programcının orta düzeyde de olsa İngilizce bilgisinin olması gerekmektedir.

Android de uygulama geliştirmenin temel problemlerinden birisi donanım bağımsız sistemlerde aynı sonucu elde etmektir. Donanım bağımsız bir oyun yapabilmek için ekran içerisindeki nesne boyutlarına direk değer değil de ekran boyutunun belli bir oranı şekilde verilmesi gerekmektedir.

Projenin daha iyi bir hale getirilmesi ve kullanıcıların ilgisini çekmek için grafik kısmına ağırlık verilip gerekirse grafik kısmı için ayrı bir kişinin çalışması gerekmektedir.

5. KAYNAKLAR

1. http://sqlitebrowser.org/ SQLite 3 Aralık 2017
2. http://bit.ly/2Akf8C2 SQLite Nedir 3 Aralık 2017
3. http://bit.ly/2jG2SSF Kullanıcı Girişi Oluşturma 3 Aralık 2017
4. http://bit.ly/2AQzXq7 XML Nedir 3 Aralık 2017
5. http://bit.ly/2ABw3k5 SQLLite Open Helper 3 Aralık 2017
6. http://huseyinbodur.net/?p=215 Veri tabanı İşlemleri 3 Aralık 2017
7. http://bit.ly/1rlyvmH Login Ekranı Yapımı 3 Aralık 2017
8. http://bit.ly/2nm0YLX Harici Veri Tabanı 3 Aralık 2017
9. http://bit.ly/2jFcNru SQLLite ve Veri Tabanı 3 Aralık 2017
10. http://bit.ly/2zKvLE1 Intent Kavaramı 3 Aralık 2017
11. http://bit.ly/2ApXmeI Splash Screen 3 Aralık 2017
12. http://bit.ly/2pruqRE ScrollingImageView 3 Aralık 2017
13. http://bit.ly/2icuGxC JoyStick 3 Aralık 2017
14. http://bit.ly/2ArcMze JoyStick 3 Aralık 2017
15. http://bit.ly/2zVJrQv Nesne Hareket Ettirme 3 Aralık 2017
16. http://bit.ly/2AQFLQk Daire Çizme 3 Aralık 2017
17. http://bit.ly/2jFZCXx Dinamik TextView 3 Aralık 2017
18. http://bit.ly/2AR7111 Android Nedir 3 Aralık 2017
19. http://bit.ly/2Bu3diB Uygulama Geliştirmek için Gerekenler 3 Aralık 2017
20. http://bit.ly/2zHCxtY Arayüz Tasarımı 3 Aralık 2017
21. http://bit.ly/2AkTttz Android Layout Manager 3 Aralık 2017
22. http://bit.ly/2j9eTjO Relative Layout 3 Aralık 2017
23. http://bit.ly/2jFfmtC Dinamik GUI 3 Aralık 2017
24. https://www.wikizero.com/tr/Gezegen Gezegen Nedir 3 Aralık 2017
25. https://www.wikizero.com/tr/Mars Mars Koşulları 3 Aralık 2017
26. http://bit.ly/2j9HJRf Merkür Koşulları 3 Aralık 2017
27. https://www.wikizero.com/tr/Ven%C3%BCs Venüs Koşulları 3 Aralık 2017
28. http://bit.ly/2idgUec Yer Çekimini Ayarlama 3 Aralık 2017
29. http://bit.ly/2jGVuGB Gravity 3 Aralık 2017
30. http://bit.ly/2jDWcV0 Socket Programlama 3 Aralık 2017

31. http://bit.ly/2BG2oUQ Yapay Zekâ 3 Aralık 2017
32. http://bit.ly/2BuzDJR Alfa Beta Algoritması 3 Aralık 2017
33. http://bit.ly/2Bu0Mwg Yapay Zekâ Kazanımları 3 Aralık 2017
34. Musa Çavuş, Android ve Oyun Programlama – 5. Baskı, Şubat 2016 Ankara, Seçkin Yayıncılık
35. Şeref Akyüz, Android Oyun Programlama – 3. Baskı, Eylül 2016 İstanbul, Dikeyeksen Yayıncılık
36. Uğur Gelişken, Mobil Oyun Tasarımı ve Programlama – 1. Baskı, Nisan 2015 İstanbul, Umuttepe Yayıncılık

STANDARTLAR ve KISITLAR FORMU

Projenin hazırlanmasında uyulan standart ve kısıtlarla ilgili olarak, aşağıdaki soruları cevaplayınız.

1. Projenizin tasarım boyutu nedir? (Yeni bir proje midir? Var olan bir projenin tekrarı mıdır? Bir projenin parçası mıdır? Sizin tasarımınız proje toplamının yüzde olarak ne kadarını oluşturmaktadır?)

Yeni bir projedir. Tasarım projenin %60'nı oluşturmaktadır bence çünkü tasarım projesinde neyin nasıl yapılacağı ve nelerin kullanılacağı belirlendikten sonra, proje aşamasında bunları uygulamak daha kolaydır. Programcı programlama sırasında neyi nasıl yapacağına karar vermekle zaman kaybetmez.

2. Projenizde bir mühendislik problemini kendiniz formüle edip, çözdünüz mü? Açıklayınız.

Herhangi bir mühendislik problemini kendim formüle edip çözmedim var olan şeyleri değiştirerek kendi projeme entegre ettim.

3. Önceki derslerde edindiğiniz hangi bilgi ve becerileri kullandınız?

Programlamaya giriş, C++ nesneye yönelimli programlama, Algoritmalar, Yapay Zeka, Windows Programlama, Ağ Programlama, Bilgisayar Ağları, Yazılım Mühendisliği

4. Kullandığınız veya dikkate aldığınız mühendislik standartları nelerdir? (Proje konunuzla ilgili olarak kullandığınız ve kullanılması gereken standartları burada kod ve isimleri ile sıralayınız).

5. Kullandığınız veya dikkate aldığınız gerçekçi kısıtlar nelerdir? Lütfen boşlukları uygun yanıtlarla doldurunuz.

a) Ekonomi

Projenin modüllere ayrılıp zaman yönetimini iyi bir şekilde planlamak.

b) Çevre sorunları:

Ağda yaşanacak olan kopmalar yüzünden oyunun çoklu oynanamaması.

c) Sürdürülebilirlik:

Oyunun geliştirilebilir bir şekilde tasarlanmıştır. Projenin yeni sürümleri isteğe göre yapılabilir.

d) Üretilebilirlik:

Projenin algoritmik açıdan gerçeklenebilecek şart ve amaca uygun üretimi yapılabilir. Oyun sektörü hızlı bir şekilde davam etmekte ve büyümektedir.

e) Etik:

Projede toplum yararı dışında etik açıdan hiçbir sorun yoktur. Kullanılan kodlara veya bilgilere ait kaynakları eklemek.

f) Sağlık:

Projede herhangi bir sağlık sorununa yol açan bir durum yoktur

g) Güvenlik:

Projede güvenlik açısından tehlike oluşturabilecek bir zararı yoktur

h) Sosyal ve politik sorunlar:

Proje sosyal ve politik bir sorun teşkil etmemektedir.