

RUBY NEDİR?

“Ruby tıpkı insan vücudu gibi görünüşte basit, ancak iç yapısında oldukça karmaşıktır.” - Matz

Bu bölümde Ruby'nin genel özelliklerinden bahsedeceğiz. Yukarıdaki söz öbeği sizi korkutmasın. Ruby dengenin dilidir. Ruby programlama dilinin yaratıcısı Yukihiro Matsumoto (ya da bilgisayar dünyasında bilinen adıyla Matz) Ruby'yi yaratırken basit bir dil olması için değil, doğal bir dil olması için çalışmıştır; tıpkı yaşam gibi... Bu yüzden Ruby öğrenirken hiç zorluk çekmeyeceksiniz, çünkü Ruby günlük hayatınızdaki alışkanlıklarınızın bir programlama dilinde hayat bulmasından başka bir şey değildir. Ruby'yi öğrenmeye başladıktan kısa süre sonra, çoğu şeyi nasıl yapıldığını bilmeden tahmin eder hale geleceksiniz. Programlama camiasında *En Az Sürpriz Kuralı* olarak bilinen bu kural, Ruby ile beraber Matz'ın *En Az Sürpriz Kuralı* olarak anılmaya başlamıştır.

Ruby öğrenmeye başladıktan kısa süre sonra yapabildiklerinizi görünce şaşıracaksınız. Daha önce herhangi bir programlama diline aşina olanlar, aynı işi yapan bir programı Ruby ile ne kadar kısa sürede ve daha az kodla yaptığınızı gördüğünüzde Ruby'den vazgeçemeyeceksiniz.

Ruby programlama dilinin özelliklerine geçmeden önce, biraz da Ruby'nin nasıl ortaya çıktığından bahsedelim. Halihazırda piyasada yüzlerce programlama dili varken, niçin yeni bir programlama diline ihtiyaç var?

Bunu anlamak için biraz da dilin yaratıcısı Matz'ın geçmişine bakmamız gerekiyor. Matz'ın üniversite yıllarındaki en büyük hayali kendi programlama dilini tasarlamaktı. Henüz ortaokul çağlarında, babasının kendisine hediye ettiği bir hesap makinesi sayesinde programlama dilleriyle uğraşmaya başlamıştı ve kısa sürede BASIC, Eiffel, LISP, Perl, Python gibi programlama dillerinde ustalaşmıştı. Ancak bu programlama dillerinden hiçbiri onun beklentilerini tam olarak karşılamıyordu. Kendisini nesneye yönelik programlama mantığının bir hayranı olan nitelendiren Matz, nesneye yönelik programlamanın avantajlarını betikleme alanında göstermek istiyordu. Matz'ın hayalindeki programlama dili Python'un gücünü, ancak Perl'ün de nesneye yönelik yapısını örnek almalıydı. Bu yüzden uzun yıllar boyunca hayalini kurduğu, kendi programlama dilini yaratma fikrini 23 Şubat 1993 yılında hayata geçirdi ve kısa bir süre sonra ilk programını çalıştırmayı başardı. 1995 yılında piyasaya sürülen ilk sürümü Japon haber listelerinde ve kullanıcı gruplarında büyük yankı uyandırdı bu tarihten itibaren binlerce özgür yazılım taraftarı tarafından geliştirilen bir dil olmaya devam etti. Böylece Matz, henüz yirmi sekiz yaşındayken en büyük hayalini gerçekleştirmiş oldu: Ruby.

Peki Ruby adı nereden gelmektedir?

Ruby'nin tasarım amaçları arasında Perl'den daha güçlü bir dil yaratmanın da olduğundan bahsetmiştik. Perl, haziran ayını simgeleyen bir burç taşıdır. Ruby ise hazirandan sonra gelen

temmuz ayını simgelemektedir. Bu zekice kurgulanmış metaforu da açıkladıktan sonra, şimdi bu dilin iştah kabartan özelliklerini bakalım.

Ruby Programlama Dilinin Temel Özellikleri

Ruby dili öğrenmesi oldukça kolay ve hızlı bir dildir. Özellikle programlama ile yeni tanışacak kullanıcılar ve aynı işi daha kısa sürede ve yaptığı programlamadan zevk alarak bitirmek isteyen pratik programcılar için tasarlanmıştır. Aynı zamanda zamanında programlama dünyasında fırtınalar estiren bir çok dilin en iyi özelliklerini barındırmaktadır. Aşağıda Ruby'nin temel özelliklerini göstermeye çalışacağız. Hem aşına olduğunuz kavramlar, hem de ufkunuzu genişletecek özellikler olduğundan eminiz.

Ruby Yüzde Yüz Saf Nesneye Yönelik bir dildir

Son yılların bir çok popüler dili arkasına nesneye yönelimli programlama paradigmasını almıştır. Peki nesneye yönelik programlama mantığı tam olarak nedir? Öncelikle bu kavrama kısaca değinelim.

Programlama dillerinin icadından bugüne kadar hep daha iyi ve daha az hatalı programlar üretilmeye çalışılmıştır. Günümüze kadar gelen bu süreç içerisinde çok çeşitli metodolojiler üretilmiş ve uygulanmıştır. Pin board'ların tellerle bağlanmasından tutun kart okuyucularına, makine dilinden makro işlemciler kullanarak programlama yapılmasına ve sonunda geleneksel programlama paradigmalarından yapısal programlamaya kadar pek çok yol kat edilmiştir.

Geleneksel programlama paradigması temel olarak üzerinde çalışılacak bir takım veriler, ve bunların üzerinde işlem yapacak fonksiyonlardan oluşuyordu. Ancak burada işi yapan da, kontrol eden de, yöneten de yalnızca fonksiyonlar oluyordu. Ve her seferinde hataları yakalamak zorlaşıyordu, çünkü proje geliştikçe fonksiyonlar daha karmaşık ve anlaşılması zor kod yığınlarına dönüşüyordu. Bunun yerine elimizdeki hareketsiz ve biz gidip bir şeyler yapmadan işe yaramayan bu verilere bazı görevler veremez miydik ? İşte nesneye yönelik programlama sayesinde verilere yani artık nesnelere belli görevler vererek kod kalitemizi arttırıyor ve hata yapma riskimizi mümkün olan en alt seviyeye indiriyoruz. Nesneye yönelimli mantık günlük hayatta da sıkça karşılaştığımız bir paradigmadır. Etrafımızda gördüğümüz klavyeden tuşlara, kaleminden silgiye her şey birer nesnedir. Ve bu nesnelerin iki temel karakteristiği bulunmaktadır: durumlar ve davranışlar. Klasik bir örnek olarak arabayı ele alalım. Bir arabanın durumları neler olabilir? Markası, rengi, kaç vites olduğu, o anki hızı... Peki davranışları? Hızlanma, yavaşlama, durma.. bu günlük hayatımızdaki her nesneye uygulayabileceğimiz bir yöntemdir. İşte programlamada kullandığımız nesneler de günlük hayatta karşılaştığımız nesnelerin bir modellemesidir. Bu nesnelerin durumları değişkenler ile tutulur, değerleri ise metotlar yardımıyla değiştirilir.

Bir çok programlama dili saf nesneye yönelik bir dil olduklarını ifade etseler de, bir çoğu nesneye yönelik programlama mantığına aykırı olan ayrıntıları göz ardı etmektedir. Örneğin nesneye yönelik programlama mantığını etkin bir şekilde savunan Java dili bile sayıları nesne

olarak kabul etmez ve metotların yanında fonksiyon kullanımına da izin verir. Ancak Ruby'de her şey birer nesnedir ve kod parçasına kendisine ait özellikler ve olaylar verilebilir.

Ruby'de yönlendirdiğiniz ve bunların döndürdüğü sonuçlar da birer nesnedir. Metotlar bile nesneye mesaj olarak uyandırılırlar. Aşağıdaki örneğe bakalım:

Java Kodu:

```
sayi= Math.abs(sayi)
```

Ruby Kodu:

```
sayi=sayi.abs
```

Burada bir sayının mutlak değerini alma işleminin iki farklı dilde nasıl yapıldığını görüyoruz. Java'da mutlak değer alma işlemi **abs()** isimli bir fonksiyon ile yapılırken, Ruby'de bir metot yardımıyla nesneye **abs** mesajı geçilerek sayının mutlak değerini almak istediğimizi belirtiyoruz.

Ruby'de herşeyin bir nesne olduğunu söylemiştik. Şimdi bu örneğimize bakalım:

```
5.times { puts "Ruby!" }
```

Yukarıdaki kodda daha sonra inceleyeceğimiz **times** metodu, kendisini çağıran nesnenin değeri kadar önüne aldığı kod bloğunu tekrarlar. Yazdığımız bu kod ekrana alt alta 5 kere Ruby kelimesini basar. Buradaki kullanımda 5 rakamı bir nesne olarak kullanılmış ve bir sayıya bile olay yüklenmiştir.

Ruby Açık Kaynak Kodlu Özgür Bir Dildir

Ruby dili, tıpkı GNU/Linux işletim sistemi gibi açık kaynak kodludur ve özgür bir dildir. Tamamen C programlama dili ile yazılmıştır ve ücretsiz olarak dağıtılmakta ve binlerce geliştirici tarafından geliştirilmektedir. Özgür yazılım lisansına uymak kaydıyla Ruby'nin nasıl çalıştığını öğrenmek için kaynak kodunu açıp inceleyebilir, ihtiyacınıza göre üzerinde istediğiniz değişikliği yapabilirsiniz.

Hatta bu dil ile yazdığınız uygulamaları hiçbir telif hakkı ödemek zorunda olmadan satabilir, istediğiniz herkese dağıtabilir, hatta kaynak kodlarını satıp para bile kazanabilirsiniz.

Ruby Yorumlanan/Derlenen Bir Dildir

Ruby genelde yorumlanan bir betik dili olarak anılsa da, uzun süredir üzerinde çalışılan YARV adı verilen sanal makinenin 2007 yılının başından itibaren kaynak kod depolarına eklenmesi sayesinde byte-code olarak derlenen dil ünvanı almayı da hak etmiştir. Aslında 2007 yılından önce de Ruby'nin kullandığı sanal makineler mevcuttu ancak hiçbirisi sanal makine tekniklerine

uygun olarak çalışmıyordu. Bu durum da Ruby'nin daha yavaş çalışmasına sebep oluyordu. Şimdiye kadar soyut söz dizimi ağacı tekniğinin yaptığı işi artık YARV sanal makinası yapacaktır. Ancak YARV halen geliştirme sürecinde olduğu için biz daha çok Ruby'nin yorumlanan bir dil olması üzerinde duracağız.

Paragrafın başında Ruby'nin yorumlanan bir betik dili olduğundan söz ettik. Peki betik dili ne anlama geliyor? Betik dillerinin çalışma prensibi, düz dosyaya yazılı kod ya da komutların dilin yorumlayıcısı tarafından anlaşılaraq çalıştırılması esasına dayanır. Bu diller edit-run-edit adı verilen teknik ile çalıştığı için hızlı geliştirme olanağı sağlamaktadır. Betik dilleri diğer derlenen diller gibi çalıştırılmadan önce ikilik çalıştırılabilir dosyalara dönüştürülmezler. Bu durum betik dillerinin yakın zamana kadar gerçek programlama dilleri kategorisinde anılmamasına neden olmuştur. Ancak artık betik dilleri sıradüzensel interaktif komutlar bütününden, tam anlamıyla donatılmış programlama dillerine dönüşmüştür. Betikleme dilleri artık derlenen bir programlama dilinin yaptığı her şeyi yapabilmektedir: Ruby, Python, Perl bu dillerden bir kaçına örnek teşkil eder. Betikleme dilleri bugün kişisel yönetici araçları, web uygulama framework'leri, devasa verilerin analizleri gibi çok çeşitli işlere hizmet etmektedir.

Hızlı Prototip Geliştirme

Yukarıda bahsettiğimiz edit-run-edit tekniği sayesinde hızlı prototip geliştirme imkanına sahip olursunuz. Özellikle yazılım şirketlerinde uygulanan bu yöntem, büyük bir projeye başlamadan önce yazılımın nasıl olduğu hakkında az çok fikir sahibi olmak için uygulanır.

Ruby hem betikleme dili olmanın getirdiği avantajlar, hem de yapısında barındırdığı basit söz dizimi ve anlaşılabilirliği sayesinde proje hakkında en kısa zamanda bir taslak hazırlamanıza yardımcı olacaktır.

Ruby Türemiş Bir Dildir

Ruby dilinin yaratıcısı Matz'ın bir programlama dilleri dahisi olması yüzünden, Ruby dilinde bir çok programlama dilinden miras alınmış özellikler olduğundan bahsetmiştik. Bu sayede ne bozulmamış olanı düzeltme gereği kalmamıştır, ne de tekerlek yeniden keşfedilmiştir. Zamanında popüler olan ancak gereksinimlere ayak uyduramadığı için gözden düşen pek çok programlama dilinin en can alıcı özellikleri Ruby'de toplanmıştır. Bu sayede C, Python, Perl gibi en çok kullanılan programlama dillerine yakın bir söz dizimi de oluşturulmuştur. Ruby'nin özelliklerini miras aldığı dillerden bazılarını Smalltalk, CLU, Lisp, C++, Perl, Python, Eiffel olarak sıralayabiliriz.

Ruby Esnek Bir Dildir

Ruby kullanıcıları kısıtlamayan bir dildir. Çok sınırlı bir çekirdek haricinde Ruby'de istediğiniz bölüme müdahale edebilir ve yeniden tanımlayabilirsiniz. Örneği toplama işlemini + operatörü

ile değil, topla adında bir metot ile yapmak isteyelim. Aşağıdaki basit kod ile bu işi yapabiliriz:

```
class Numeric
  def topla(x)
    self.+(x)
  end
end

y = 2.topla 3
```

Yukarıdaki örnekte gördüğümüz Numeric sınıfı, Fixnum ya da Bignum gibi sayı sınıflarının temel aldığı üst sınıftır. Bu örnekte Numeric sınıfına topla isminde yeni bir metot ekledik. Sınıf ve metot tanımlamalarını sonraki bölümlerde göreceğiz. Bu yüzden şimdilik **topla** metodunun ne yaptığını sözsel olarak ifade edelim: topla metodu argüman olarak aldığı değişkeni, kendisini çağıran nesne ile toplar. Burada 2 rakamı topla metodunu çağıran nesne, topla metodunun aldığı argüman ise 3 rakamı olmaktadır. Böylece varolan Numeric sınıfını kendi istediğimiz doğrultusunda farklılaştırdık ve yeni bir metot ekledik.

Ruby Gücünü Bloklardan Alır

Ruby'nin esnek bir dil olarak anılmasının sebeplerinden biri de bloklardır. Blok kavramı Ruby'ye özgür bir kavram değildir, pek çok programlama dili blokları kullanarak aynı işi daha kısa kod parçacıklarıyla halletme şansına sahip olurlar. Ruby'nin blokları fonksiyonel programlama dillerinden esinlenilerek yaratılmıştır. Matz blok kavramını Lisp'ten aldığını ifade etmiştir. Blokların işlevi basitçe, kullanıcıya öğeleri yinelemek için kendine uygun bir yol çizmesine izin vermesidir. Blokları isimsiz fonksiyonlar olarak görebiliriz. Lisp ya da Python'daki lambda kavramına karşılık gelen blokların asıl amacı dile işlevsellik katmasıdır. Aşağıdaki örneği inceleyelim:

```
dizi = ["ruby", -5.7, 8]

dizi.each { |e| puts e }
```

Şimdilik yukarıdaki kodu anlamanıza gerek yok. Zaten bir kaç bölüm sonra ayrıntılı bir şekilde işleyeceğiz. Yine de yabancılık çekmemeniz için kısaca bahsederseniz, yukarıda **dizi** isimli değişkene bir dizi atıyoruz. Ruby'de dizilere dizge, ondalık sayı ya da tamsayı gibi farklı türden elemanlar koyabildiğimize dikkat edin.

İkinci satır için şimdilik yaptığımız işin mantığını öğrenmeniz yeterli. Yorumlayıcıya kısaca şunu diyoruz: “Dizi isimli değişkenin barındırdığı her eleman için (**each**) bu elemanı al ve **e** isimli değişkene koy. Daha sonra **puts** ile bu elemanı ekrana bas”.

Ruby Dinamik Tanımlanan Bir Dildir

Ruby'de değişken bildirimleri gereksizdir, çünkü değişkenlerin türü yoktur. Ruby bu işlevi duck-

typing programlama paradigması ile yapar. Bu sayede daha önce herhangi bir yerde tanımlamadığınız bir değişkene direkt olarak bir değer vererek kullanabilirsiniz. Verdiğiniz bu değer bir dizge ise değişkeninize dizge muamelesi, eğer sayı değeri verdiyseniz sayı muamelesi yapılır. Ancak buna rağmen değişkenler arası tip dönüştürmesinin sıkı kontrolü yapılır. Bu özellik programlama camiasında "strongly typed" olarak anılır ve olası hataların gözden kaçmaması için eklenen bir özelliktir. Bu yüzden değişkenler arası tip geçişlerinde daha sonra bahsedeceğimiz **to_s**, **to_a** gibi geçiş metotları kullanılır.

Değişkenlerin yaşam alanları ise basit kurallar ile belirlenir. Örneğin küçük harfle başlayan bir değişken yerel, @ işareti ile başlayan bir değişken örnek (instance), \$ işareti ile başlayan bir değişken ise global değişken muamelesi görür.

Ruby Basit ve Tutarlı Söz Dizimine Sahiptir

Ruby'nin söz dizimi basit ve tutarlıdır. Daha önce bahsettiğimiz gibi pek çok dilden miras aldığı özellikler sayesinde kolay anlaşılır ve alışılır bir söz dizimine sahiptir. Programlama ile uğraşanlar sık sık üzerinde uğraşacakları kodu yazmaya başlamadan önce bir kağıda ya da taslağa izleyecekleri adımları yazarlar. Böylece kod yazmaya başlayınca izleyeceğimiz adımlara bakarak kolaylık sağlarsınız. Yazılan bu kodlar pseudo kod olarak anılırlar. Peki, yazdığınız bu pseudo kodlar bilgisayarınızda direkt olarak çalışsaydı ne hissederdiniz? İşte Ruby, size bu zevki yaşatacak kadar kolay ve eğlenceli bir söz dizimine sahiptir.

Aşağıda ekrana Ruby Programlama Dili yazan, biri Java'da diğeri Ruby'de yazılmış iki kod görüyorsunuz:

Java Kodu:

```
System.out.println("Ruby Programlama Dili");
```

Ruby Kodu:

```
print "Ruby Programlama Dili"
```

Yukarıdaki kodu biraz daha geliştirelim ve daha sonra göreceğimiz for döngüsü yardımıyla ekrana 0'dan 9'a kadar olan rakamları basalım.

Java Kodu:

```
for( i=0; i<10; i++)  
    System.out.println( i );
```

Ruby Kodu:

```
for i in 0..10  
    print i  
end
```

Ruby Platform Bağımsızdır

Ruby her ne kadar *nix türevi işletim sistemleri üzerinde geliştirilse de, DOS, Windows 95/98/Me/NT/2000/XP, MacOS, BeOS ve OS/2 gibi pek çok işletim sistemi üzerinde de çalışabilmektedir. Bu yüzden yüksek taşınabilir bir dil olarak anılır.

Ruby Unix ve türevleri üzerindeki tüm sistem çağrılarını erişebildiği gibi, Win32 API'si sayesinde hemen hemen Windows üzerindeki tüm sistem çağrılarını da erişebilmektedir.

Ruby Popüler Bir Dildir

Ruby programlama dili, camiaya sunulduğu 1995 yılından itibaren dünyanın dört bir yanından programcıların dikkatini çekse de 2006 yılı Ruby'nin geniş kitleler tarafından kabul gördüğü bir yıl olmuştur. Dünyanın önde gelen şehirlerinde aktif kullanıcı grupları oluşmuştur ve Ruby konferansları ve etkinlikleri düzenlenmektedir.

Ruby'nin genel tartışma listesi Ruby-Talk günde 200'den fazla e-posta trafiğine sahiptir. Aşağıdaki grafikte Ruby tartışma listesinin yıllara göre istatistiği görülmüyor.

resim: buyume-istatistigi.png

Ruby, programlama dillerinin istatistiklerini inceleyen TIOBE şirketinin araştırmalarına göre şu an dünyanın en popüler 11.ci dili konumundadır. Araştırmacılar 6 ay içinde Ruby'nin ilk 10'a gireceği konusunda fikir birliği içindedirler.

Gittikçe popülerleşen bir dili ilk öğrenenlerden olmak her zaman fark yaratan avantajlardan biridir.

Öğrenmesi Kolaydır

Aynı zamanda Ruby dili yeni bir programlama dili öğrenecekler için harika bir kılavuz görevi görmektedir. Yüzde yüz nesneye yönelik yapısı sayesinde en önemli programlama paradigmalarından nesneye yönelim mantığını anlamanın yanında, bundan sonra öğreneceğiniz herhangi bir dil için sağlam bir temel atmanıza yardımcı olur. Basit ve tutarlı söz dizimi sayesinde, yeni başlayacaklar için kolay ve hızlı bir altyapı sunar ve kısa sürede yapabildiklerinizi görünce kendinize olan güveniniz de artmasına ve daha istekli programlama yapmanıza yardımcı olur.

Halihazırda kullandığınız ve ustalaştığınız programlama dillerine sahipseniz bile, Ruby öğrenmek yarar sağlayacaktır. Özellikle gereksiz sözcüklerle dolu olan programlama dilleri ile yazacağınız kodu çok daha kısa sürede ve daha az satırda yazarak kritik noktalarda ciddi avantajlar sağlayabilirsiniz.

Python gibi bazı programlama dilleri bir şeyi yapmanın yalnızca bir tek yolunu sunar. Oysa Ruby bir şeyi yapmanın birden çok yolunu sunarak programcılara kendi yöntemlerini uygulama özgürlüğü verir. Bu özellik sayesinde farklı programcılar aynı işi yapmak için geliştirdikleri çözümleri birbiriyle paylaşma ve farklı stratejiler öğrenme fırsatı yakalarlar.

Ve Diğer Özellikler

Ruby, Java ve Python'da olduğu gibi kolay hata yakalama mekanizmasına sahiptir.

Ruby siz görmeseniz de her Ruby nesnesi için işaretle ve temizle (mark & sweep) çöp toplayıcısı sayesinde kullanılmayan nesneleri bellekten siler.

Ruby'de C eklentileri yazmak Python ya da Perl'de olduğundan daha kolaydır. Bu sayede kodunuzun hızlı çalışmasının kritik önem taşıdığı yerlere C kodu gömmeniz mümkün hale gelir.

Ruby işletim sistemi izin verdiği sürece dinamik olarak kütüphane yükleyebilir.

Ve daha pek çok iştah kabartan özellik Ruby dilinde mevcuttur.

Ruby Ne Gibi Ticari Uygulamalarda Kullanılıyor ?

Simülasyonlarda,

NASA Langney araştırma merkezi simülasyonlarını gerçekleştirmek için Ruby kullanmaktadır. Aynı zamanda Motorola bazı simülasyonlarının hem senaryolarını oluşturmak hem de verilerin sürecini gerçekleştirmek için Ruby kullanmaktadır.

Robotbilimde,

Siemens, bir servis robotunun kontrolünü sağlamak için Ruby'yi kullanmıştır.

Oyunlarda,

Japonya'da ticari bir oyun firması, Ruby ile geliştirdiği RPG oyununu Haziran 2004'te piyasaya sürmüştür.

Telefonculukta,

UCB, kablosuz telefonları ve trafiğin yükünü kontrol etmek için Ruby'yi kullanmaktadır.

3G kablosuz telefonculuk řirketi, ~150K'lık C++ koduna karşı, ~6K'lık Ruby kodunu kullanmıřtır. Bu sayede inanılmaz bir hız avantajı saęlamıř ve kara geçmiřtir.

Bilimde,

Yüksek yoğunluklu yıldız sistemlerinin modellemesi üzerinde çalıřan ACS řirketi de projelerinde Ruby kullanılmaktadır.

Bunlar dıřında İřletmelerde, Ağ Yönetiminde, Sistem Yönetiminde, Ağ Uygulamalarında ve daha pek çok yerde Ruby kullanılmaktadır. Ruby ile yazılmıř ticari uygulamaların kapsamlı bir listesi için *RubyGarden.org* adresini ziyaret etmenizi öneririm.

Ülkemizden örnek vermek gerekirse, bünyesinde onlarca site barındıran Pilli Network'ün [*pilli.com*] altyapısı Ruby ile yazılmıř RubyonRails isimli uygulama ile yazılmıřtır.