

The background features a complex network graph with nodes represented by circles of varying sizes and binary node labels (e.g., 000, 100, 010, 110, 001, 111) in white and yellow. A person's hands are visible on the right side, using a laptop with a dark blue screen displaying a grid pattern.

# GraalVM & Native Image

Muhammet Ali KAYA - Furkan Şahin KULAKSIZ

# GraalVM Nedir?

GraalVM, Java uygulamalarınızı derleme zamanında bağımsız çalışabilir binaries(iki dosya) dosyalara derler. Bu binaries daha küçütür, 100x'e kadar daha hızlı başlar, en üst düzey performans sunar ve Java Sanal Makinesi'nde (JVM) çalışan uygulamalardan daha az bellek ve CPU kullanır.

GraalVM, uygulamanızın saldırısı yüzeyini azaltır. Kullanılmayan sınıfları, metodları ve alanları uygulama ikilisinden çıkarır. Yansıma(reflection) ve diğer dinamik Java dil özelliklerini yalnızca derleme zamanında sınırlar. Çalışma zamanında bilinmeyen herhangi bir kod yüklemez.

Spring Boot, Micronaut, Helidon ve Quarkus gibi popüler mikroservis çerçeveleri ile Oracle Cloud Infrastructure, Amazon Web Services, Google Cloud Platform ve Microsoft Azure gibi bulut platformları GraalVM'i destekler.

Profil tabanlı optimizasyon ve G1 (Garbage-First) çöp toplayıcı ile, Java Sanal Makinesi'nde (JVM) çalışan uygulamalara kıyasla daha düşük gecikme süresi ve aynı seviyede veya daha iyi bir performans ve verimlilik elde edebilirsiniz.



# Graal Compiler

Graal derleyicisi, bayt kodunu makine koduna dönüştüren, Java ile yazılmış dinamik bir tam zamanında (JIT) derleyicidir. Ancak klasik yöntemlerden ayırsız;

- Derleyici kod analizi ve optimizasyonuna yönelik yeni yaklaşımlarla JVM üzerinde çalışan programlarımız için optimize edilmiş bir performans sağlar.
- Belirli nesnelerin maliyetli tahsislerini ortadan kaldırabilemeye sağlayan «Kısmi kaçış analizi optimizasyonu» içerir.
- Erken Derleme sunar(AOT- Ahead-of-time), Java ve JVM tabanlı kodu yerel bir platformda yürütülebilir dosyaya çeviren gelişmiş bir ileri düzey (AOT) derleme teknolojisi olan Native Image ile birleştirir. Bu yerel yürütülebilir dosyalar neredeyse anında başlar, daha küçütür ve aynı Java uygulamasının daha az kaynağını tüketir; bu da onları bulut dağıtımları ve mikro hizmetler için ideal kılar.

GraalVM için 2 farklı compiler modeli bulunur;

- libgraal(AOT): Graal derleyicisi önceden yerel bir paylaşılan kitaplıkta derlenir. Bu çalışma modunda paylaşılan kitaplık HotSpot VM tarafından yüklenir. Derleyici, belleği HotSpot yığınından ayrı olarak kullanır ve işinmaya ihtiyaç duymadığından başlangıçtan itibaren hızlı çalışır. Bu, varsayılan ve önerilen çalışma modudur.

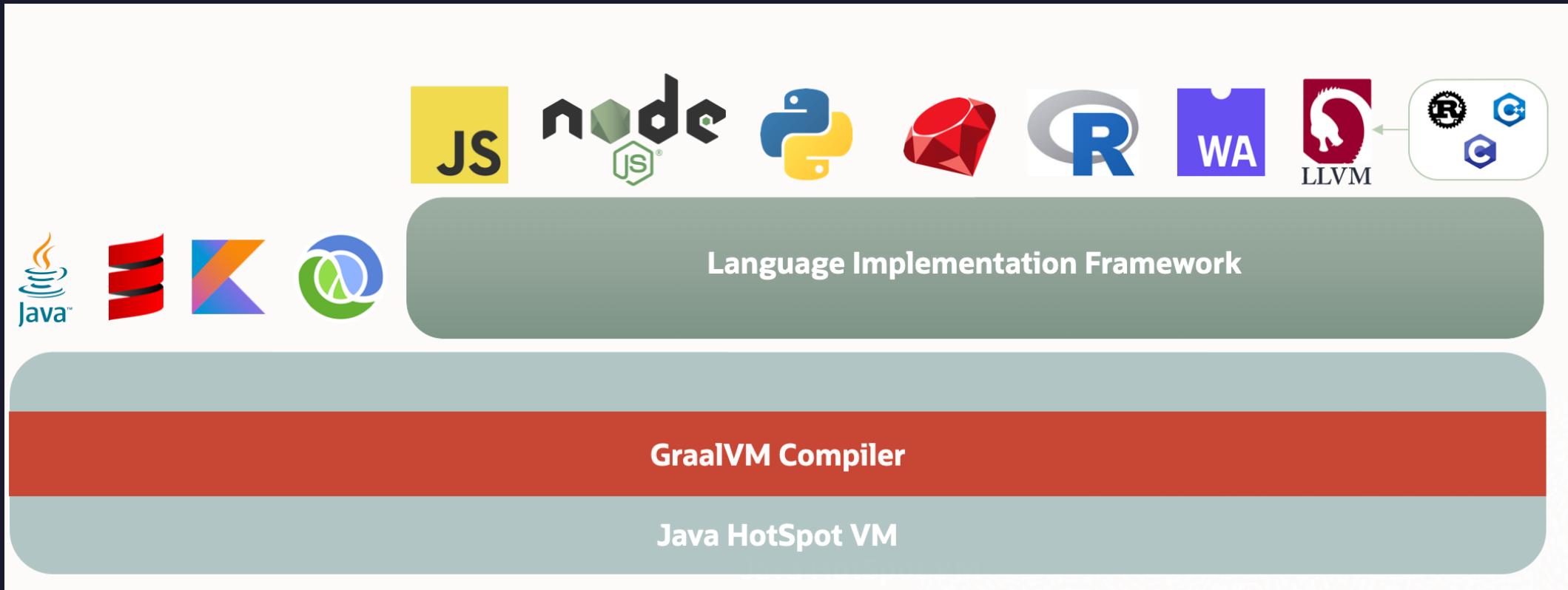
- jargraal(JIT) : Graal derleyicisi, Java uygulamasının geri kalıyla aynı işinme aşamasından geçer. Yani, sıcak yöntemleri derlenmeden önce ilk önce yorumlanır. Bu mod komut satırı seçeneğiyle seçilir -XX:-UseJVMCINativeLibrary. Bu, derleyicinin hızlı bir şekilde kod üretmeden önce derlenmesi gerekiğinden, en yüksek performansa ulaşma süresini geciktirecektir.

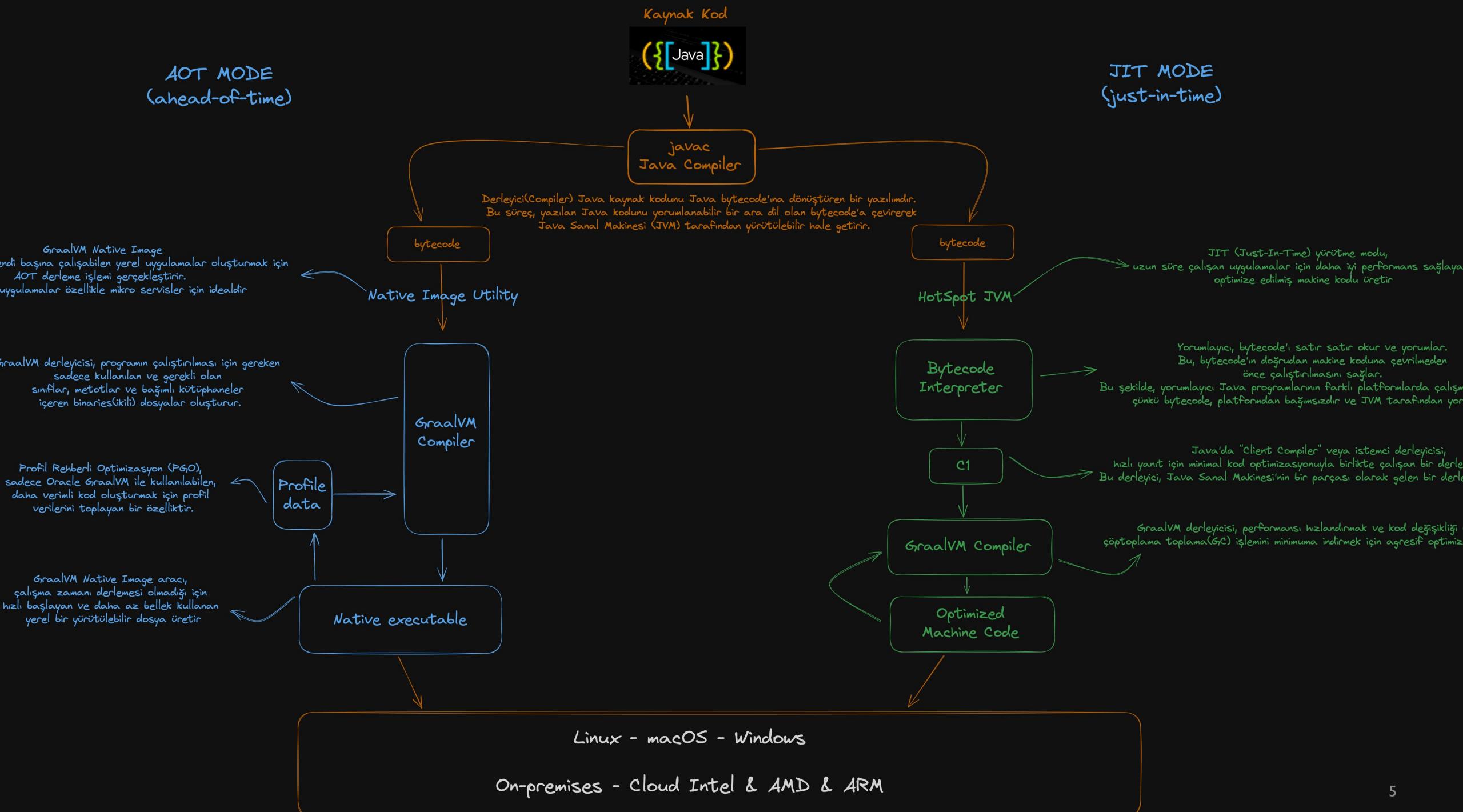


# GraalVM Architecture

GraalVM, HotSpot Java Sanal Makinesi'ne Java ile yazılmış gelişmiş bir tam zamanında (JIT) optimizasyon derleyicisi ekler.

GraalVM'nin dil uygulama çerçevesi (Truffle), Java ve JVM tabanlı dilleri çalıştırmanın yanı sıra, JVM'de JavaScript, Ruby, Python ve diğer birçok popüler dilin çalıştırılmasını mümkün kılar. GraalVM Truffle ile Java ve desteklenen diğer diller birbirleriyle doğrudan birlikte çalışabilir ve verileri aynı bellek alanında ileri geri aktarabilir.





# Native-Image Nedir?

GraalVM Native Image, Oracle tarafından geliştirilen bir teknolojidir ve Java Virtual Machine (JVM) üzerinde çalışan Java uygulamalarını native (yerel) bir biçimde derlemeye olanak tanır. Bu, Java uygulamalarının geleneksel JVM tabanlı çalışma ortamı yerine doğrudan işletim sistemi üzerinde çalışabilen hızlı ve optimize edilmiş bir native uygulamaya dönüştürülmesini sağlar.

GraalVM Native Image'in temel avantajlarından biri, uygulamaların başlatılma sürelerini dramatik bir şekilde azaltmasıdır. Geleneksel JVM tabanlı uygulamalar genellikle başlatılma süreleri konusunda eleştirilere maruz kalırken, GraalVM Native Image ile bu süreler önemli ölçüde iyileştirilebilir. Bu, özellikle mikroservis mimarileri ve serverless uygulamalar gibi hızlı başlatma gereksinimleri olan ortamlarda önemlidir.

GraalVM Native Image aynı zamanda daha düşük bellek tüketimi sağlar, bu da özellikle hafif konteynerler ve edge cihazları gibi kaynak sınırlı ortamlarda avantaj sağlar. Ayrıca, native olarak derlenmiş uygulamalar genellikle daha iyi performansa sahiptir, çünkü işletim sistemi düzeyinde optimize edilmiş ve JVM'in ek yüklerinden kurtulmuş olurlar.

Bu teknolojinin uygulama alanları geniş bir yelpazede bulunabilir, özellikle Java uygulamalarının daha hızlı başlatılmasına, daha düşük kaynak tüketimine ve daha iyi performansa ihtiyaç duyulan durumlarda GraalVM Native Image kullanılmaktadır.

**GraalVM™** 

**Native Image**

# Native-Image Bazı Terimler

## LLVM (Low-Level Virtual Machine):

1. LLVM, düşük seviyeli bir sanal makine ve derleme teknolojisi olup, genellikle C ve C++ gibi dillerin derlenmesi için kullanılır.
2. LLVM, modüler bir yapıya sahiptir ve geniş bir diller yelpazesi için kullanılabilir. Ayrıca, yüksek performanslı ve optimize edilmiş makine kodu üretebilme yeteneğine sahiptir.
3. Java uygulamalarını derlerken, GraalVM tarafından kullanılabilecek bir backend olarak da kullanılabilir.

GraalVM'nin "native-image" aracı, Java uygulamalarını LLVM tabanlı bir derleme süreci kullanarak optimize edip, doğrudan makine koduna çevirme amacı taşır. Bu, genellikle performans ve bellek tüketimi avantajları sağlayabilir. Ancak, bu tür derleme süreçleri genellikle uygulama bağımlılıklarını ve çeşitli Java özelliklerini destekleme konusunda bazı kısıtlamalara neden olabilir.



## Native Image

# Native-Image Bazı Terimler

JNI, Java Native Interface'ın kısaltmasıdır. JNI, Java dilinde yazılmış bir uygulamanın, Java sanal makinesi dışındaki (native) diğer programlar veya kütüphanelerle iletişim kurabilmesini sağlayan bir programlama arayüzüdür. Bu sayede Java uygulamaları, özellikle performans gerektiren veya özel sistem kaynaklarına erişim gerektiren durumlar için, örneğin C, C++, Assembly gibi dillerde yazılmış kütüphaneleri kullanabilir.

İşte JNI'nin temel özellikleri:

## 1. \*\*Java ve Native Dil İletişimi:\*\*

- JNI, Java dilinde yazılmış uygulamaların, native (makine dilinde) yazılmış diğer dillerdeki kodlarla etkileşim kurabilmesini sağlar.
- Bu, Java uygulamalarının platforma özgü veya performans gerektiren işlevleri çağrılabilmesine imkan tanır.

## 2. \*\*Native Method Interface (NMI):\*\*

- JNI, Java sınıflarına özel native metotları (native methods) tanımlamak için kullanılır. Bu native metotlar, Java kodu içinde tanımlanmaz; ancak bir başka dilde yazılmış bir kütüphane içinde bulunabilir ve Java tarafından çağrılabılır.

## 3. \*\*Java Platform Bağımsızlığı:\*\*

- JNI, Java'nın platform bağımsızlığı özelliğini korur. Java uygulaması, native kodla etkileşimde bulunsa da, JNI bu etkileşimi genellikle Java uygulamasının platform bağımsızlığını koruyacak şekilde yönetir.

## 4. \*\*JNI Kütüphaneleri:\*\*

- JNI, genellikle C veya C++ gibi dillerde yazılmış kütüphaneleri Java uygulamalarına entegre etmek için kullanılır.
- JNI kütüphaneleri, Java uygulamasının native kodla iletişim kurmasını sağlar ve genellikle Java Native Interface API'larını kullanarak bu iletişimini yönetir.

Özetle, JNI, Java uygulamalarının platforma özgü veya özel native kodları kullanabilmesine imkan tanıyan bir arayüsdür. Bu, özellikle sistem kaynaklarına doğrudan erişim veya yüksek performans gerektiren uygulamalarda kullanılabilir. Ancak, JNI kullanılırken dikkatli olunmalı ve güvenlik önlemleri alınmalıdır, çünkü hatalı kullanım sistem kararlılığını ve güvenliğini tehditeye atabilir.



# Polyglot Programming

GraalVM, kullanıcıların Truffle dil uygulama çerçevesi (bundan böyle “Truffle”) aracılığıyla değerleri bir dilden diğerine sorunsuz bir şekilde aktaran çok dilli uygulamalar yazmasına olanak tanır.

Truffle Language Implementation Framework, polyglot programlamanın temelini oluşturan bir çerçevedir. Bu çerçeve, farklı programlama dillerinin bir arada çalıştırılmasını ve entegrasyonunu destekler.

Truffle, özel dillerin JVM (Java Sanal Makinesi) veya GraalVM üzerinde çalışacak şekilde optimize edilmesine olanak sağlar. Bu çerçeve, herhangi bir dil için çalışma zamanı ortamını oluşturmak için soyutlamalar sağlar. Bu soyutlamalar, dilin özelliklerini temsil eden yapıları içerir.

Polyglot programlamada, Truffle, farklı dillerin bu ortak çalışma zamanı üzerinde bir arada çalışmasını sağlar. Örneğin, Java, JavaScript, Ruby gibi farklı dillerin bir arada kullanılabilirnesine imkan tanır. Bu dillerin birbiriyle etkileşim içinde olmasını, veri paylaşımını veya bir dilin diğerini çağırmasını sağlar.

Truffle'ın sağladığı bu esneklik sayesinde, farklı dillerin bir arada kullanıldığı uygulamalar geliştirilebilir. Örneğin, bir uygulamanın çekirdek yapısını Java ile oluşturabilirken, özel bir işlevi Python veya JavaScript ile yazabilir ve bu diller arasında iletişim kurabilirsiniz. Kısacası, Truffle Language Implementation Framework, polyglot programlamaya imkan tanıyan bir çerçeve olarak farklı dillerin bir arada çalışmasını sağlar. Bu sayede, farklı dillerin sunduğu avantajları kullanarak projelerin ihtiyaçlarına daha uygun ve esnek çözümler geliştirmek mümkün olur



# Neler Değiştirdi

NAME	CPU(cores)	MEMORY(bytes)
beemeadministrator	1m	330Mi
beemeauthdeployment	1m	232Mi
beemechallengedeployment	3m	568Mi
beemechallengedeployment	107m	641Mi
beemechallengedeployment	82m	576Mi
beemechallengedeployment	4m	320Mi
beemechallengedeployment	2m	315Mi
beemechallengedeployment	95m	8368Mi
beememediadeployment	51m	8284Mi
beememediadeployment	65m	3684Mi
beememediadeployment	31m	8318Mi
beememediadeployment	30m	4094Mi
beememediadeployment	105m	4968Mi
beememediadeployment	133m	7499Mi
beememediadeployment	104m	9939Mi
beemeuserdeployment	53m	1193Mi
beemeuserdeployment	54m	888Mi
beemeuserdeployment	75m	689Mi
beemeuserdeployment	56m	739Mi
beemeuserdeployment	138m	852Mi
beemeuserdeployment	72m	570Mi
beemeuserdeployment	42m	769Mi
beemeuserdeployment	117m	452Mi
beemevideoeditdeployment	1m	355Mi
deploymentzipkin-6c	4m	3612Mi
grafana-6978bdfb86-	1m	69Mi
managementservicede	1m	53Mi
mongodbdeployment-7	772m	12994Mi
prometheus-alertmanager-0	1m	22Mi
prometheus-kube-state-metrics-5fb6fbbf78-8sg7l	1m	17Mi
prometheus-prometheus-node-exporter-87vhv	1m	11Mi
prometheus-prometheus-node-exporter-8lqbs	1m	12Mi
prometheus-prometheus-node-exporter-jjttb	1m	9Mi
prometheus-prometheus-pushgateway-7d55869d46-rwl5x	1m	18Mi
prometheus-server-6bd74cf9bc-rmsgv	4m	499Mi

NAME	CPU(cores)	MEMORY(bytes)
beemeadministrator	1m	330Mi
beemeauthdeployment	1m	90Mi
beemechallengedeployment	1m	627Mi
beemechallengedeployment	1m	627Mi
beemechallengedeployment	1m	649Mi
beemechallengedeployment	1m	314Mi
beememediadeployment	2m	315Mi
beememediadeployment	1m	354Mi
beememediadeployment	40m	1391Mi
beememediadeployment	147m	1810Mi
beememediadeployment	3m	1304Mi
beememediadeployment	1m	908Mi
beememediadeployment	1m	302Mi
beememediadeployment	4m	1320Mi
beememediadeployment	61m	1238Mi
beemeuserdeployment	8m	998Mi
beemeuserdeployment	7m	827Mi
beemeuserdeployment	7m	865Mi
beemeuserdeployment	3m	636Mi
beemeuserdeployment	6m	1061Mi
beemeuserdeployment	4m	920Mi
beemeuserdeployment	5m	1169Mi
beemeuserdeployment	3m	968Mi
beemevideoeditdep	2m	366Mi
deploymentzipkin-6c	1m	3604Mi
grafana-6978bdfb8	1m	69Mi
managementservice	2m	53Mi
mongodbdeployment	419m	13031Mi
prometheus-alertmanager-0	1m	22Mi
prometheus-kube-state-metrics-5fb6fbbf78-8sg7l	1m	18Mi
prometheus-prometheus-node-exporter-87vhv	1m	11Mi
prometheus-prometheus-node-exporter-8lqbs	1m	11Mi
prometheus-prometheus-node-exporter-jjttb	1m	9Mi
prometheus-prometheus-pushgateway-7d55869d46-rwl5x	1m	17Mi
prometheus-server-6bd74cf9bc-rmsgv	22m	479Mi



# Sorunlar ve Çözümleri

## 1- Türkçe sorunu

İşletim sisteminin dilinin Türkçe olması, class kullanımlarında ve methodların çevriminde sorumlara yol açıyor. Her ne kadar düzeltildiği iletilese de uyguladığım bir çok GraalVM versiyonun native-image build işlemleri hata vermiş çalışmamıştır. İşletim sisteminin dilini İngilizce ye çevirdikten sonra hatalar son bulmuştur.

## 2- Yanlış ve fazla kurulum

Kurulumları genellikle Sdkman kullanarak oluşturuyoruz. Ancak lokalde kurulu olan java sürümleri, path ayarları, JAVA\_HOME gibi configlerden ötürü imaj oluştururken anlamsız hatalar alabilirsiniz. Bazı C++ kütüphanelerinin olmadığı bulunamadığı yönünde hatalar sıkılıkla yaşanabilir. Tüm bu sorunlardan kaçınmak için bilgisayarınızda kurulu olan sdkman ile kurduğunuz tüm aktif sürümleri iptal ederek güncel tek bir sürüm bırakırsanız sorunlarınız düzellecektir.



The background of the slide features a complex network of glowing nodes and connections. The nodes are represented by small circles in various colors, including white, blue, red, and yellow. These nodes are interconnected by thin lines of the same color, creating a web-like structure that suggests a global or local network. The background is a dark, solid color, which makes the glowing nodes stand out. The overall effect is one of connectivity and complexity.

Teşekkürler...