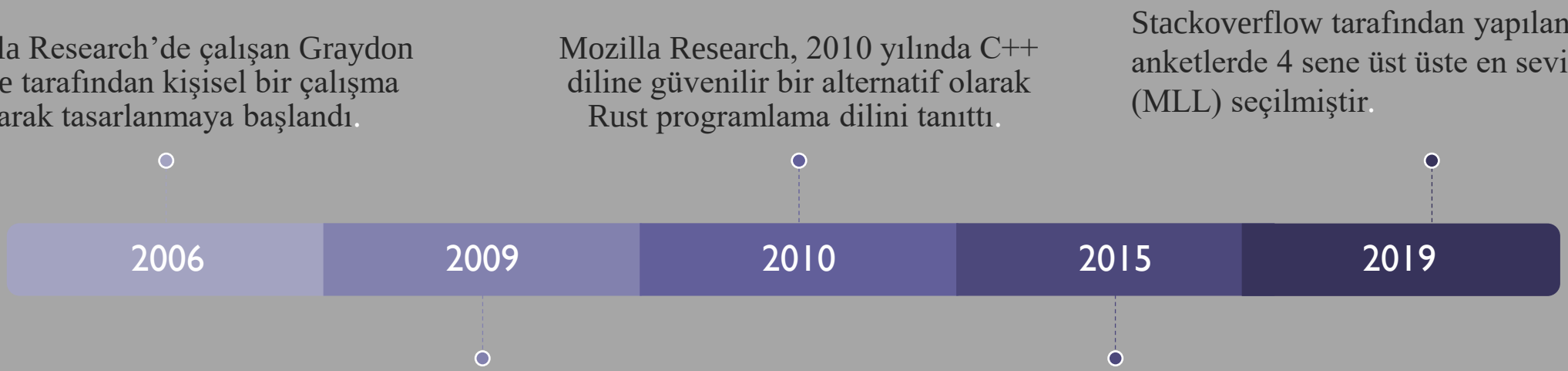
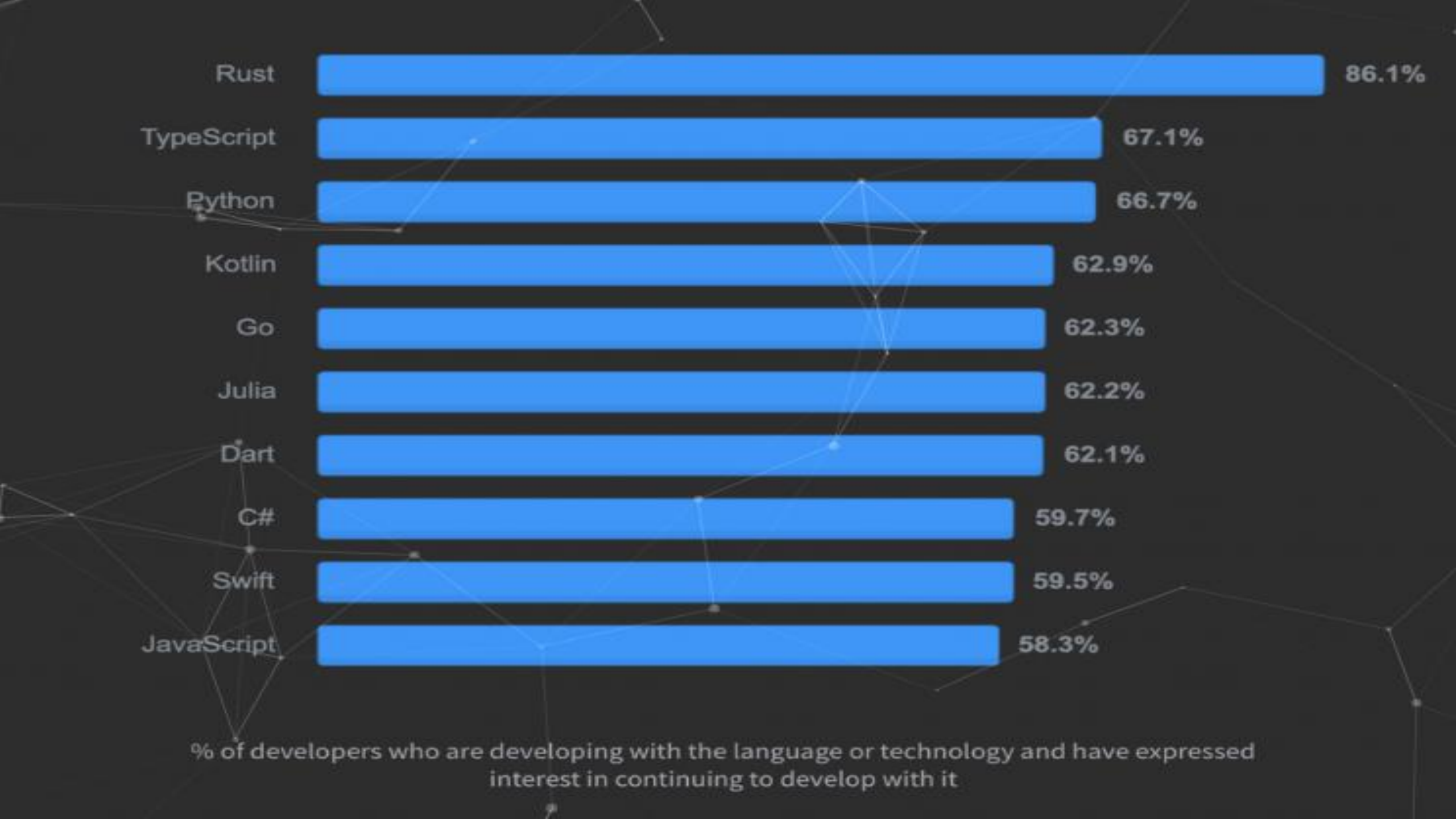
RUST PROGRAMMİNG LANGUAGE

Emine Erdem - 19360859008

Gelişme Süreci




Rust
86.1%
TypeScript
67.1%
Python
66.7%
Kotlin
62.9%
Go
62.3%
Julia
62.2%
Dart
62.1%
C#
59.7%
Swift
59.5%
JavaScript
58.3%
% of developers who are developing with the language or technology and have expressed interest in continuing to develop with it

GELİŞTİRİCİLER

Graydon Hoare

Software Developer

Brendan Eich

JavaScript Developer

Dave Herman

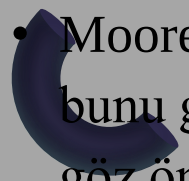
Servo Motor Developer

RUST NEDİR ?

- Rust; güvenli, eş zamanlı ve hızlı bir *sistem programlama dilidir. (Safe - Concurrent - Fast)*
- Birçok dilden ilham almıştır:
- **Sistem Programlama Dilleri** : (C , C++)

  Sözdizimi ve performans açısından.
- **Fonksiyonel Programlama Dilleri** : (Haskell , Erlang , Meta Language)

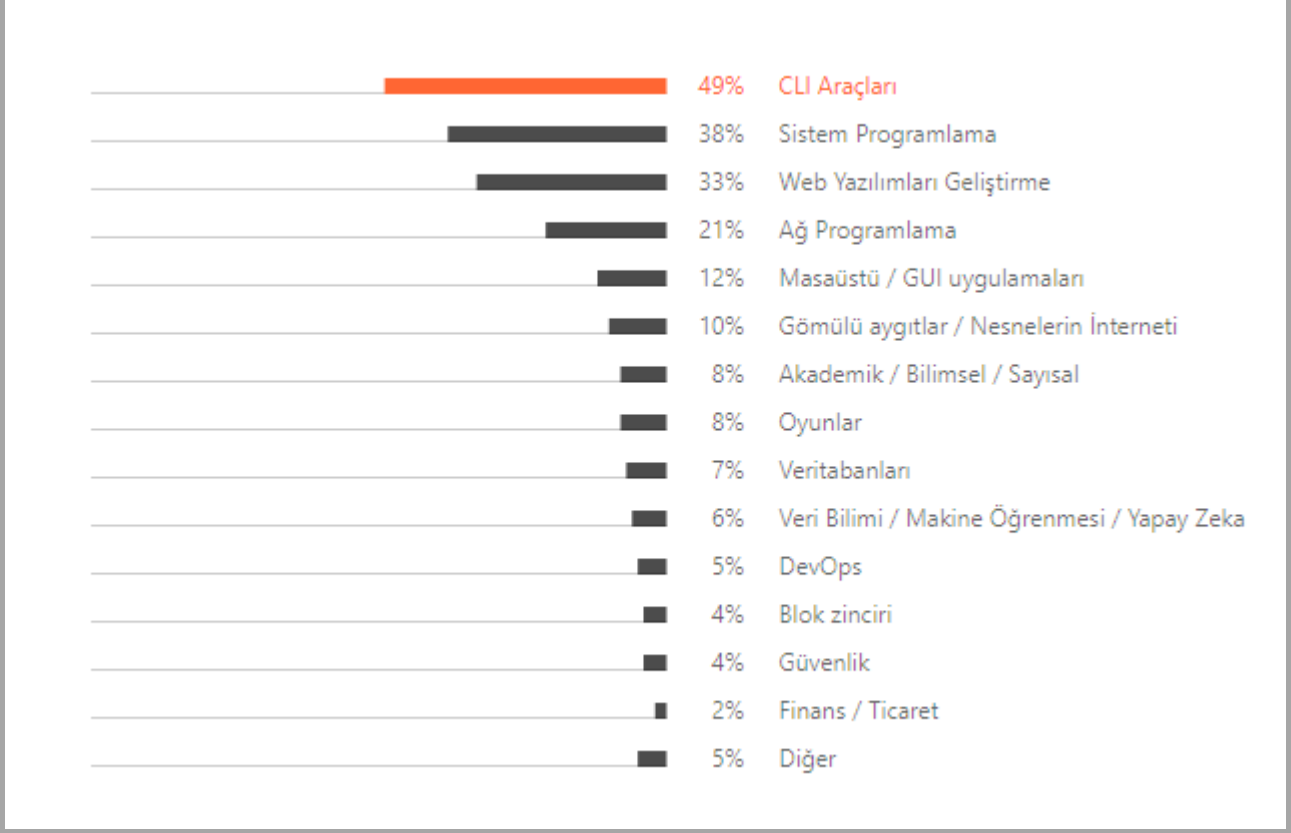
  Yalnızca sözdizimi açısından.
- **Rust neden modern bir dildir?**
- Moore yasasının geçerliliğini kaybetmesi üzerine concurrent/parelel programlama önem kazandı. Rust dili bunu göz önünde bulundurarak tasarlandı. Rust ekibi eski dillerden öğrendiği birçok tasarımsal sıkıntıları göz önünde bulundurarak tasarlandı ve geliştirilmeye devam ediliyor.



RUST NE İÇİN KULLANILIR ?

Rust aşağıdakiler için uygundur:

- Platformlar arası komut satırı desteği
- Sistem programlama
- Güçlü web uygulamaları oluşturma
- Gömülü sistem programlama
- Dağıtılmış çevrimiçi hizmetler oluşturma



Özellikleri

- NULL ifadesi yerine Rust'ta 'Option' türü yer almaktadır. Bu tür içerisinde Some veya None olan test edilebilir iki adet veriyi barındırmaktadır.
- Garbage Collector yok onun yerine 'Borrowing & Ownership' kavramları bulunur.
- RFC
- Değişmezlik (Immutable)
- FFI ile WebAssembly desteği bulunmaktadır.
- Doküman içerisinde çalışabilen testler -> ///doc
- Decanstring özelliği ile dışarıdan bir değişken aldığımız zaman, alabileceği tipin ne olduğunu algılayıp ona göre doldurur.
- Memory-Safe bir dildir.
-

Rust'ı Diğer Dillerden Ayıran 3 Temel Özellik

BORROWING (ÖDÜNÇ ALMA)

- **Bazı durumlarda değişkenin sahipliğini devretmeden değişkenin verisine erişmek isteyebiliriz. İşte bunun gibi durumlarda, Rust'ın borrowing mekanizmasını devreye giriyor. Bir nesneyi değer olarak göndermek yerine(T), onu referans olarak(&T) gönderebiliyoruz. Referanslar, en basit haliyle, veri yerine diğer bir değişkenin bellek adresini tutan yapılardır.**

OWNERSHIP (SAHİPLİK)

- **Bir kaynağın sadece bir sahibi olabilir. Tabi bazı değişkenler hiçbir kaynağa sahip olmayabilir. (Örneğin referanslar) Bir başka değişkene atama yaptığımızda (Örn: let x = y;) veya fonksiyona parametre olarak değişkeni verdiğimizde(Örn: fonk(x)) o kaynağın sahibi de aktarılır. Buna Rust terminolojisinde move (taşıma) adı veriliyor. Bir kaynağı taşıdığımızda, artık o kaynağın bir önceki sahibi kullanılamaz olur. Bu da silinen bir kaynağı hala işaret eden bir pointer'ın olmasının önüne geçiyor.**

LIFE TIME (KULLANIM SÜRESİ)

- **Lifetime Rust hafıza güvenliği modelinin önemli bir parçasıdır. Her referans bir Lifetime tanımı bulundurmalıdır. Ham işaretçiler güvensiz sayıldıklarından Rust dili ham adresler ile Lifetime kullanımını şart koşmaz.**

- **Genel kullanım küçük a harfi ile başlaması ve devam etmesi seklindedir. Argüman olarak alınan değer sonuç olarak döndüğünden yasam sureleri ayindir ve dolaysı ile Lifetime on ekleri aynı harftir.**

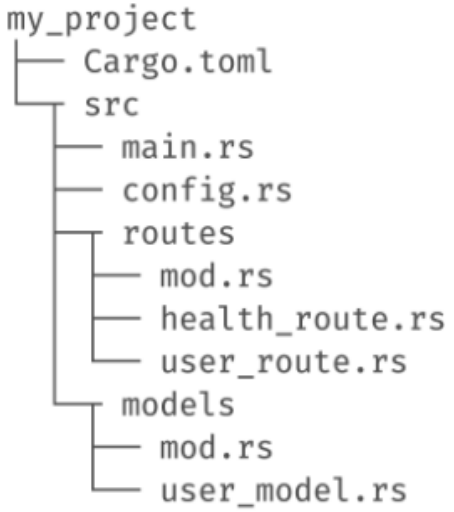
30 Mayıs 2022

Cargo - Crate Deposu:

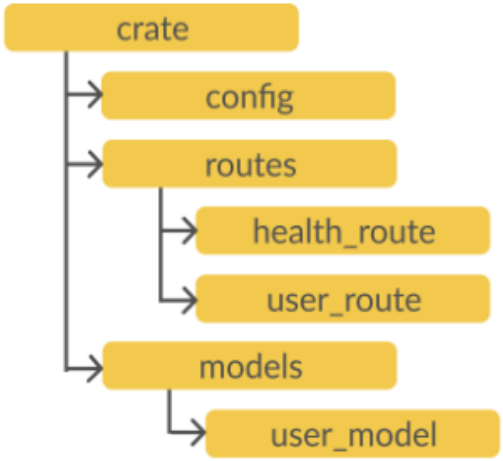
- Rust ekosistemini diğer sistem programlama dillerinden ayıran en önemli farklardan biri de ‘Cargo’ isimli paket yöneticisidir. Cargo ile derlenebilen Rust paketlerine ‘Crate’ adı verilir. Bir Crate en az bir ‘rs’ kod dosyası ve Toml formatında Cargo.toml isimli meta bilgi dosyasından oluşur. Kod dosyasının adı kütüphane türündeki Cargo projeleri için ‘lib.rs’ uygulama türündekiler için ise ‘main.rs’ olarak (Cargo kısa adları ile Lib ve Bin) adlandırılmaktadırlar.
- https://crates.io/

- Crate'ler duruma göre kütüphane veya executable dosya olusturmaya yarıyor. *cargo new project* dediğimiz zaman 'Project' adında yeni bir crate oluşturmuş oluyoruz.
- İki çeşit crate bulunur; *main.rs* ve *lib.rs*
- *mod* -> modül
- *pub* -> public

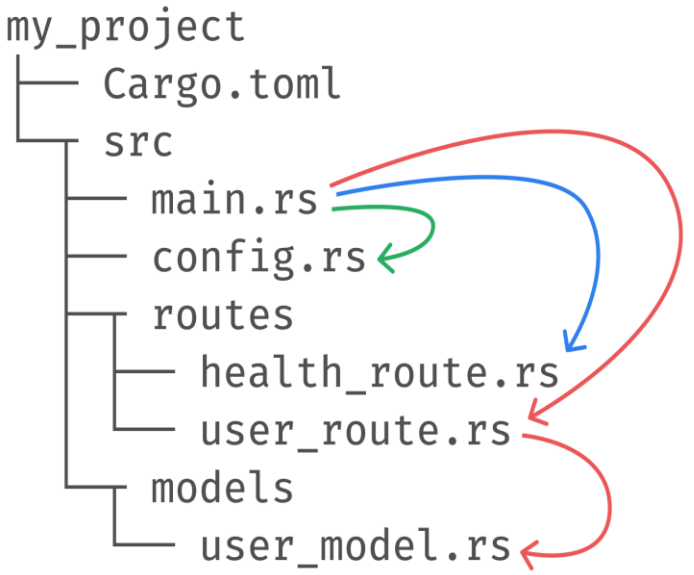
```
.
├──product.rs
├──user/
│   ├──mod.rs
│   ├──profile.rs
│   │       pub struct Profile{}
│   └──employee.rs
└──main.rs // crate root
    mod user;
    mod product;

mod profile;  // profile modulunu user modulune sub-modul olarak dahil ed
iyoruz.
mod employee; // employee modulunu user modulune sub-modul olarak dahil e
diyoruz.

pub use profile; // profile modulune hem iceriden hem disaridan erisime a
ciyoruz.
use employee; // Sadece user altindaki moduller employee modulune erisebi
lir.
```



File System Tree



Module System Tree



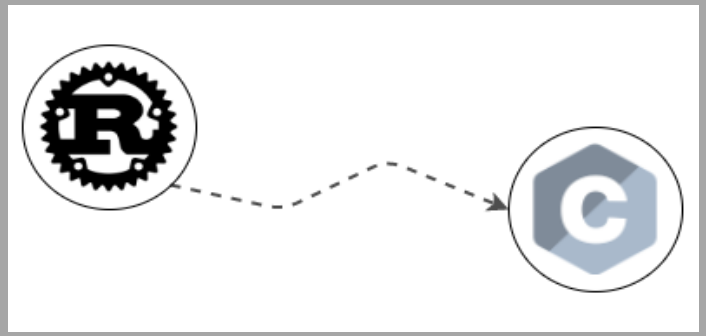
main.rs ⇒
config.rs

main.rs ⇒
routes/health_route.rs

main.rs ⇒
routes/user_route.rs ⇒
models/user_model.rs

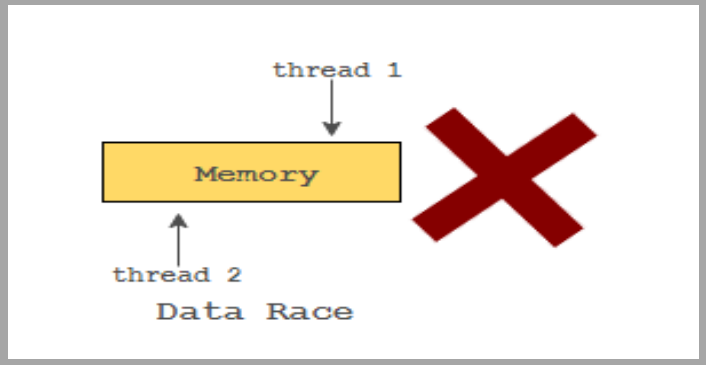
Verimli C Bağlama

Rust dili, C dili ile birlikte çalışabilir. C API'leri ile iletişim kurmak için bir yabancı fonksiyon arayüzü sağlar. Sahiplik kuralları nedeniyle bellek güvenliğini garanti edebilir.



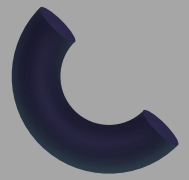
Veri Yarışları (Data Race/Race Condutation)#

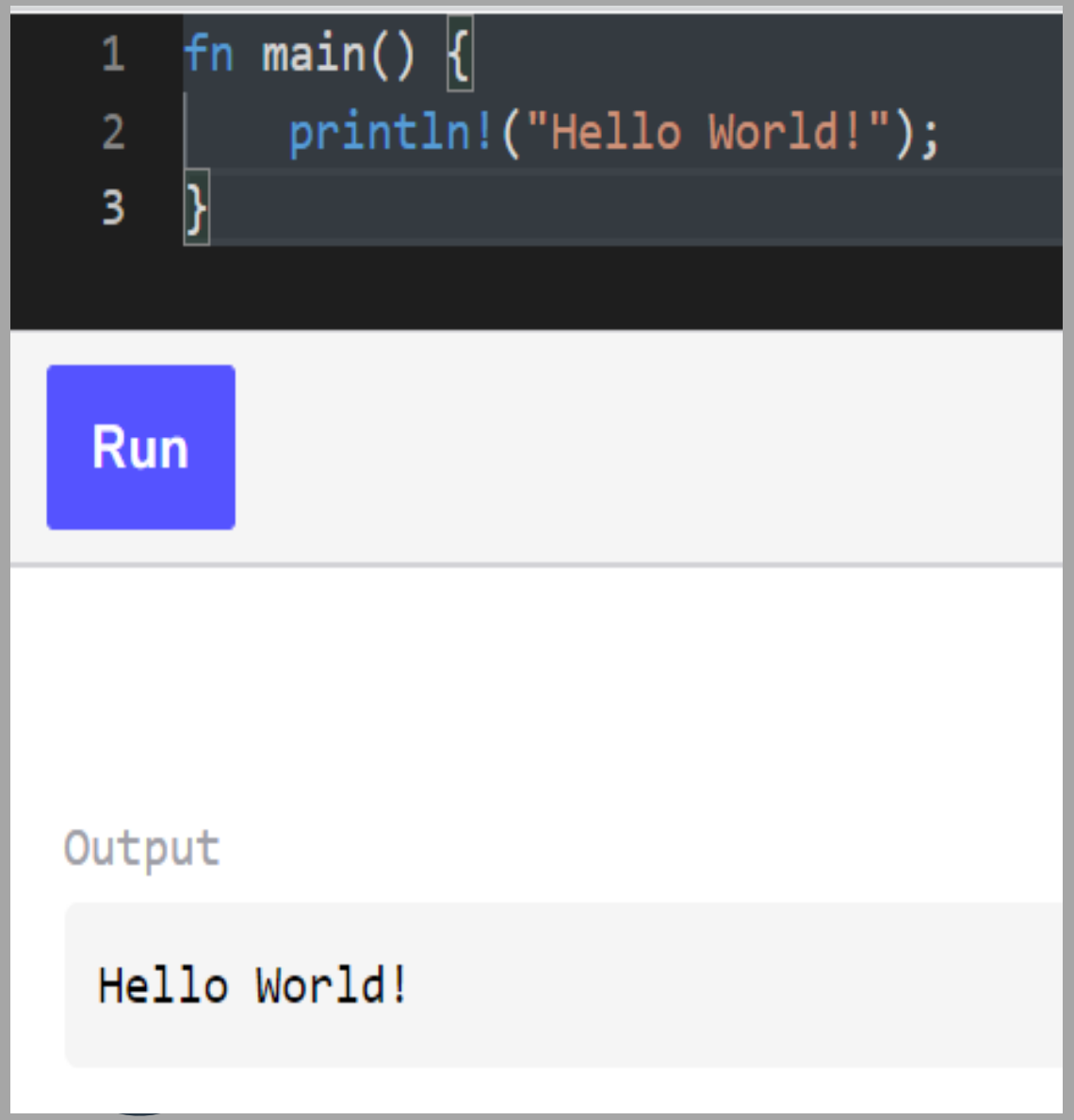
Veri yarışı, iki veya daha fazla iş parçacığının aynı bellek konumuna erişebildiği bir durumdur. Rust, veri yarışlarından kaçınmak için sahiplik(ownership) kavramını kullanır.
Race Condition'lar genellikle uygulamanın doğru çalışmasını bozdukları için bug olarak adlandırılırlar.



Primitif Tipler:

- Bool
- Char
- Numerik (i8, i16, i32, i64, u8, u16, u32, u64, isize, usize, f32, f64)
- Array
- Slice
- Tuple (T, U…)
- Fonksiyonlar





Baskı Tipleri:

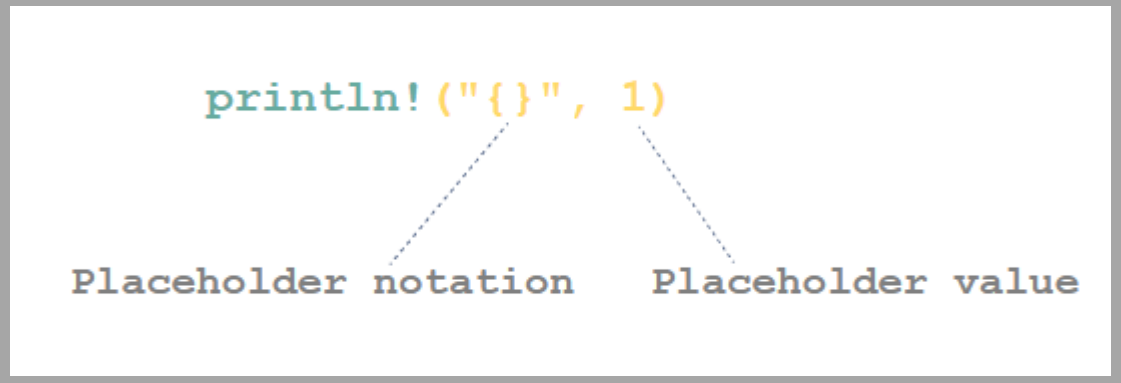
print!

println!

eprint!

eprintln!

Yer Tutucular:



- Single Placeholder
- Multiple Placeholder

Yorum Türleri:

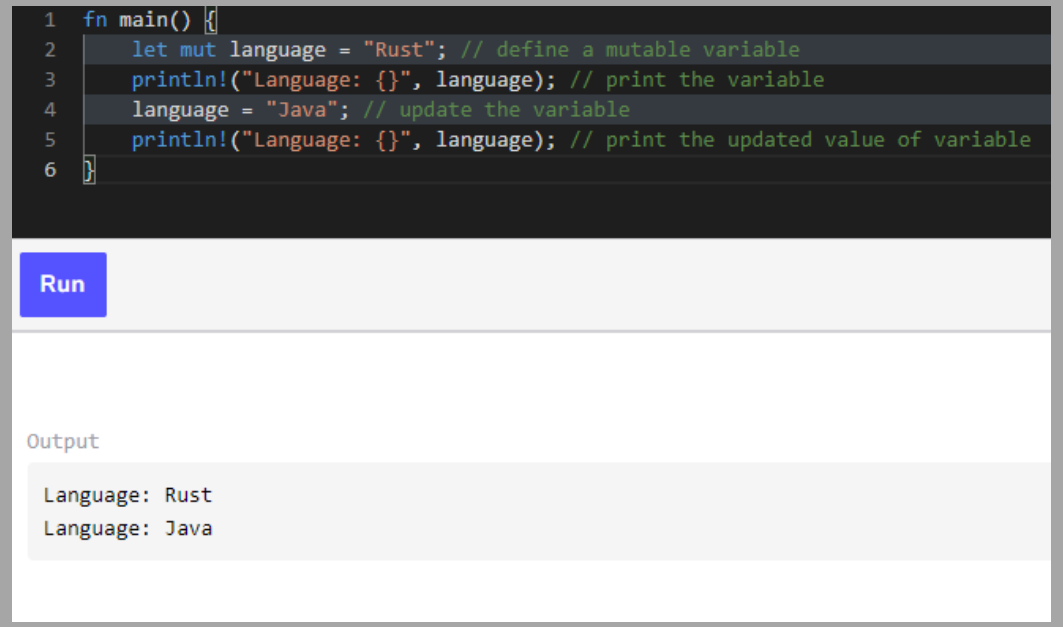
- // -> Satır yormları
- /* */ -> Blok yorumları
- /// -> Dış doküman yorumları
- //! -> Dahili doküman yorumları

Değişkenler:

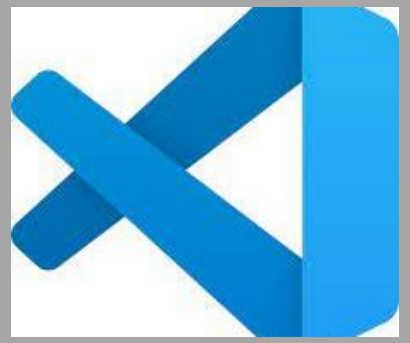
- *let* ile tanımlanır.
- Adlandırma kuralları;

  Tüm harfler küçük olmalıdır.

  Tüm kelimeler ‘_’ kullanılarak ayrılmalıdır.
- Tanımlanan bir değişken ancak *mut* (muttable) ile değiştirilebilir.



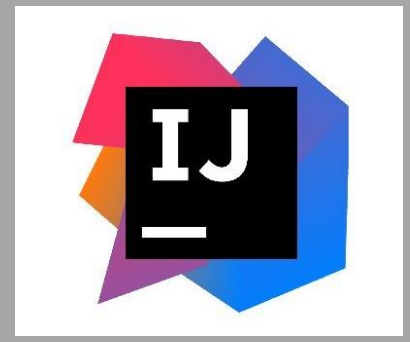
Kullanılan IDE'ler



VS Code

Vim

CLion



IntelliJ Idea

RUST FOUNDATION

Dinlediğiniz İçin Teşekkürler