认识面向基础架构语言Rust

推荐大家使用Rust来编写健壮的应用程序

苏林

知数堂 zhishutang.com

Datafuse

自我介绍

- 前折800互联网研发团队负责人, 10余年一线研发经验
- 目前是多点Dmall技术Leader
- 具有多年的软件开发经验, 精通Ruby、Java、Rust等开发语言
- 同时也参与过Rust中文社区日报维护工作.

分享内容

- 诞生背景: 想要解决什么问题
- 设计思想: 如何解决这些问题
- 发展前景: 未来Rust将大有可为
- Hello world: 了解Rust如何执行

诞生背景: 想要解决什么问题

诞生背景: 想要解决什么问题

- 认识Rust作者 Graydon Hoare

诞生背景: 想要解决什么问题

- 认识Rust作者 Graydon Hoare
- 存在两个难题
  - 1、很难编写内存安全的代码
  - 2、很难编写线程安全的代码

诞生背景: 想要解决什么问题

- 认识Rust作者 Graydon Hoare
- 存在两个难题
  - 1、很难编写内存安全的代码
  - 2、很难编写线程安全的代码
  - Ada语言

诞生背景: 想要解决什么问题

- 认识Rust作者 Graydon Hoare
- 存在两个难题
- 1、很难编写内存安全的代码
- 2、很难编写线程安全的代码
- Ada语言
- C/C++ 得以普及的原因
- 性能可以和 C/C++ 媲美，还能保证安全性，同时可以提供高效的开发效率，代码还得容易维护

诞生背景: 想要解决什么问题

可以说，Rust 诞生之初就是奔着 C/C++ 去的，Hoare 在一次采访中就曾说过：

> *纵观周围，大部分堆栈级的系统代码都是用 C 或者 C++ 编写的，而那正是我们的目标所在。同时，我们的目标人群正是那些纠结的 C/C++ 程序员，实际上就是我们自己。如果你也和我们一样，不断重复地迫使自己因为 C++ 的高效和部署特性而选择它来进行系统级的开发，却又希望可以编写一些更加安全而省心的程序的话，希望我们可以给你一些帮助。*

正是因为 Hoare 以这种观点作为基石，才使得今天的 `Rust成为了一门同时追求安全、并发和性能的现代系统级编程语言` 。

设计思想: 如何解决这些问题

设计思想: 如何解决这些问题

- 为了达成目标，Rust 语言遵循了四条设计哲学：
- 1、安全
- 2、并发
- 3、高效
- 4、零成本抽象

设计思想: 如何解决这些问题

- 1、安全

来看安全。Rust 是静态的，拥有丰富的类型系统和所有权语义模型，保证了内存安全性和线程安全性。 举个例子，C 语言中很容易出现整数溢出，如果被黑客利用，就会出现安全问题，而 Rust 中的每个值都只能被一个所有者拥有，所以 C 语言常遇到的这类问题，对 Rust 来说都不是问题。

同时，借助类型系统的强大，Rust 编译器可以在编译期就对类型进行检查，及时发现内存不安全的问题，使开发者能在编译阶段就将诸多类型错误扼杀于萌芽之中。当然，凡事都有两面性，之前，我一位从 C++ 转向 Rust 的程序员朋友就吐槽过，“C++ 是调试的时候想撞墙，而 Rust 是编译的时候想撞墙。”

设计思想: 如何解决这些问题

- 2、并发

来看并发。并发和并行是技术圈内永不会过时的话题，而 Rust 在设计层面就提供了一整套机制来保证并发的安全性。举个例子，数据并发在多线程程序中是一个常见的危险因素，而 Rust 通过所有权模型，非常清晰地定义了一个安全的边界，保证你的代码不会出问题。

设计思想: 如何解决这些问题

- 3、高效

来看高效。Rust 没有运行时机制，也没有垃圾回收机制，所以它非常快并且内存效率极高。同时，Rust 可以为关键性能服务提供支持，还可以轻松地与其他语言集成。

设计思想: 如何解决这些问题

- 4、零成本抽象

除了安全、并发、高效，Rust 还追求高效开发和性能。

编程语言如果想做到高效开发，就必须拥有一定的抽象表达能力。关于抽象表达能力，最具代表性的语言就是Ruby。Ruby 代码和 Rust 代码的对比示意如代码清单 1-1 所示。

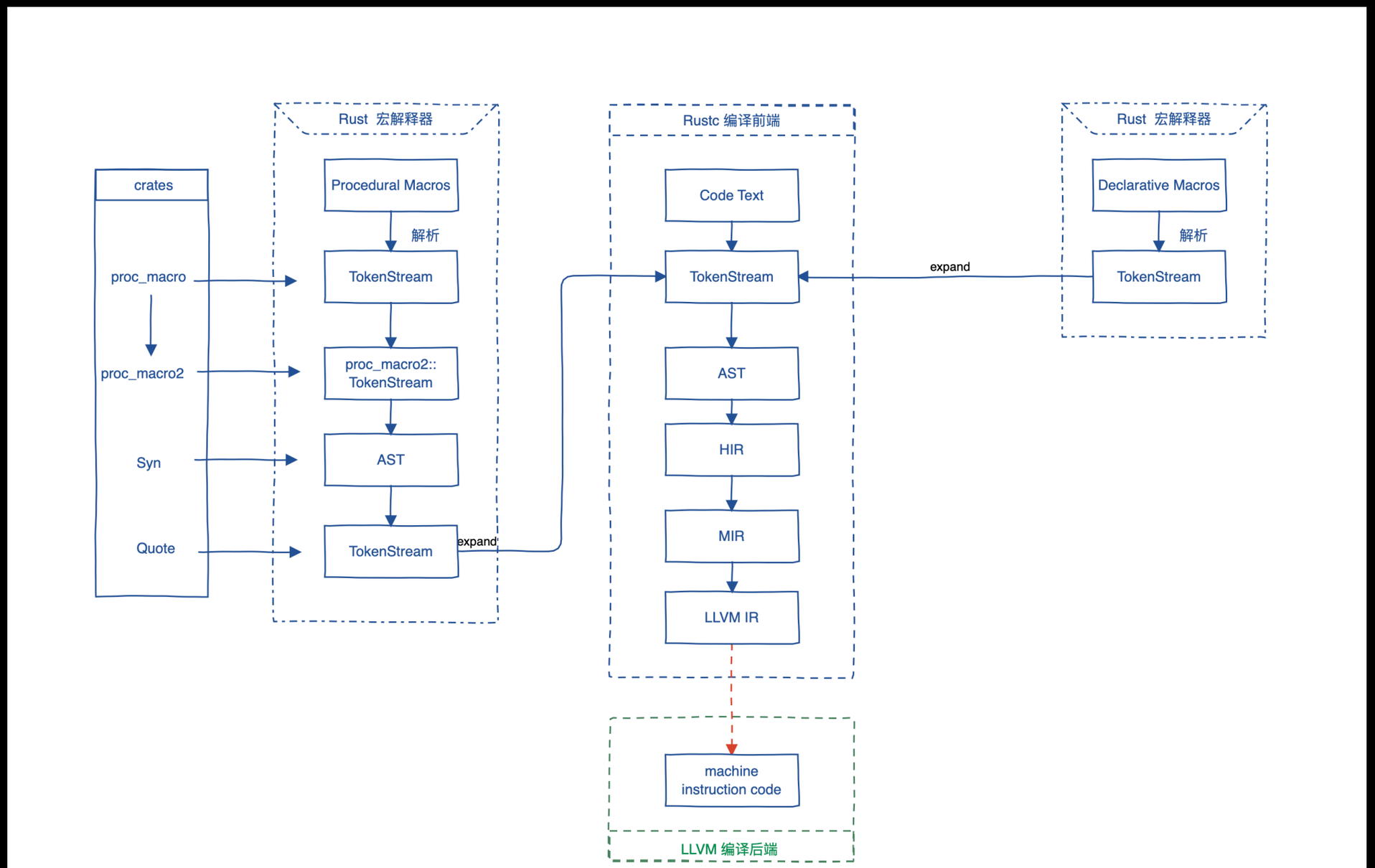
```
# Ruby代码
5.times{ puts "Hello Ruby"}
2.days.from_now;
```

```
# Rust代码
5.times(|| println!(“Hello Rust”));
2.days().from_now();
```

发展前景: 未来Rust将大有可为

- 微软宣布将探索使用 Rust 编程语言作为 C、C++ 和其他语言的替代方案
- AWS 在其开源博客上发文陆续使用 Rust 编写了多款产品, AWS 更是将 Rust 编译器团队负责人纳入麾下，进一步壮大了其内部的 Rust 团队。
- 国内PingCAP公司的TiKV已经快3.0了，使用Rust开发。
- 国内区块链公司秘猿，Nervos公链，也使用Rust开发。
- 知乎搜索引擎用Rust构建。
- 阿里蚂蚁金服时序数据库、淘宝广告推荐算法都已经使用了Rust。
- 腾讯云也在开始布局Rust
- Parity、以太坊区块链。
- Atlassian，在后端使用Rust。
- Dropbox，在前后端均使用了Rust。
- Facebook，使用Rust 重写了源码管理工具。
- Google，在Fuchsia 项目中部分使用了Rust。
- Microsoft，在Azure IoT 网络上部分使用了Rust。
- npm，在其核心服务上使用了Rust。
- RedHat，使用Rust 创建了新的存储系统。
- Reddit，使用Rust 处理评论。
- Twitter，在构建团队中使用Rust。

- Hello world: 了解Rust如何执行
- https://www.rust-lang.org/zh-CN/
-



总结

Rust 的产生看似偶然，其实是必然。未来的互联网注重安全和高性能是必然的趋势。GH 看到了这一点，Mozilla 也看到了这一点，所以两者才能一拍即合，创造出 Rust。

Rust 从 2006 年诞生之日开始，目标就很明确——追求安全、并发和高性能的现代系统级编程语言。

所以，你准备好学习 Rust 了吗?

QA环节

加群一起交流Rust & Datafuse

知数堂
zhishutang.com

Datafuse