D言語のはなし(仮)

2023.2.25 A:Macアルコールセッション

想定聴衆

- そもそもよくわからない
  - プログラミング興味なかったらごめんなさいmm
- D言語聞いたこともない人も多そう

  → Googleにきいてみよう

• • •


D言語
d言語 rust
d言語 アンサイクロペディア
d言語 オワコン
d言語 とは
d言語 rust 比較
d言語 特徴
d言語 c++
d言語 演算子
d言語 流行らない

まあ気を取り直してね

目的

- D言語で人生を踏み外す人を増やしたい
  - まずは知ってもらうこと
  - よさについて理解してもらう

D言語とは

- 静的型つき言語 ← ここは人権レベル
- ネイティブコード ← 実行速度も優秀!
- GCつき ← メモリ管理をサボれる!
- 構文論に注力 ← C・C++へのアンチテーゼ
- 強力なコンパイル時メタプログラミング機能
- 安全性 ← GC以外にも言語機能として
- 標準ライブラリも豊富

かいつまんで

- そこそこ楽に書けて
- それなりに速くて
- わりと安全になってて
- メタプロで遊べて

  という言語

Hello, World!

- 見た目Javaっぽい

```
import std.stdio;
void main() {
    writeln("Hello, World");
}
```

- 実質Javaといえる

構文論

- Q: 以下のCプログラムの実行結果は？

```
int x = 42;
if (0  < x < 20) printf("small!\n");
if (20 < x < 40) printf("medium!\n");
if (40 < x < 80) printf("large!\n");
```

答え

- 全部出力される

```
// C言語はこう書き換える:
//   if (((0 < x) != 0) < 20)
if (0  < x < 20) printf("small!\n");
```

- 構文と意味論の対応が非直感的！

強力なメタプログラミング

- テンプレートによる抽象化
  - IFTI, alias parameters, constraints
- コンパイル時リフレクション
  - コンパイラが持ってる情報へのアクセス
  - traits

テンプレート

- C++のあれ
  - 基本的には同じもの
- 記法が簡単に
  - これでテンプレート関数が宣言できる

    T square(T)(T i) { return i * I }
  - これをIFTI(Implicit Function Template Instantation)と呼ぶ
    - 覚える必要はまったくない

Template alias parameters

- 任意のシンボルをパラメータにとれる
  - これがめちゃくちゃ強力
- 例: ラムダをテンプレートパラメータに

```
auto doFunc(alias f)() { return f(2, 2); }
doFunc!((a, b) => a + b);  // 4
```

Traits: compile-time reflection

- コンパイル時リフレクションを実現するための機能
- テンプレート制約との組み合わせで使う

```
auto hoge(T)(T x) if (__traits(isCopyable, T))
{ … }
```

メモリ安全性

以下の操作を言語機能でサポート

- バッファオーバーフロー対策(配列境界チェック)
- 不正なキャスト操作
- 不正なスタックフレーム操作(ref, scope)
- 競合状態(shared型)

  詳しくは
  https://qiita.com/kubo39/items/01f5ca9efae124f7441a
  で

でもRust使えばよくないですか(笑)

うるせえ！ 

まあ実際そう…(小声)

でも、ちょっとまってほしい

勝ち馬に乗るのはそんなに楽しいか？

漢なら逆張りをしていけ！

コミュニティも歓迎！

- Twitter: #D言語 or #dlang で！
- Slack: dlang-jp
- GitHub: github.com/dlang-jp