

型システム入門メモ

maton

第 19 章 事例：Featherweight Java

19.1 導入

Featherweight Java がサポートする機能は以下。

- 相互再帰的なクラス定義
- オブジェクト生成
- フィールド参照
- メソッド呼び出し
- メソッドオーバーライド
- `this` を介したメソッド再帰
- 部分型付け
- キャスト

19.2 概観

副作用を捨てると嬉しい副作用がある。

19.3 名前的型システムと構造的型システム

Java は部分型を定義する際に明示的に上位型を `extends` したり `implements` するので、名前的。

19.4 定義

演習 19.4.1. [★] S-Top 規則が不要なのは、`Object` がすべてのクラスの上位型であることがクラス宣言による部分型付け規則と、クラス宣言が持つ健全性条件から明らかとなるからである。部分型付け規則により、あるクラスが他のクラスを継承していれば、それらが部分型関係を持つことがわかる。そして、クラス表が健全性条件を満たすとき、すべてのクラスの上位型関係を辿っていくといずれ `Object` 型関係にたどり着く。これは `Object` が S-Top 規則における Top 型と同等の役割を果たしていることに他ならない。

演習 19.4.2. [★★] 18 章で紹介されたレコードと部分型付けを持つラムダ計算と同等の言語機能があるので、FJ の構文をすべて 18 章のラムダ計算の構文糖衣とする。これによって議論は更に簡潔になる。Java 風の計算体系に更に機能を加えたければ新たな構文糖衣を導入すればよく、何らかの証明が必要な場合はラムダ計算の世界で行えば既存の議論を再利用できるためである。

演習 19.4.3. [推奨, ★★★ ⇨] (破壊的代入). 大雑把には、フィールドが全て参照であるのみなし、コンストラクタで初期化、メソッドボディ内で破壊的代入をサポートする。

演習 19.4.4. [★★★ ⇨] (try-catch). 未着手。

演習 19.4.5. [★★ ⇨] (包摂規則). 包摂規則は、15 章 (p.142) で導入されている T-Sub のような規則で、ざっくり言えば、部分型関係あるなら暗黙的にアップキャストしてよい、という規則である。そのため、T-UCast を無くすることが可能となる。しかし、構文的には明示的にも暗黙的にもアップキャストできたほうが嬉しいだろう。また、15 章でも議論したように、ダウンキャストに対しては明示的な指定と失敗時のフォローが必要であり、包摂規則だけでは T-DCast や T-SCast を取り除くことはできないだろう。

演習 19.4.6. [★★★] (インターフェイス). 未着手。

(1) Java 流のインターフェイスで FJ を拡張する。 I はインターフェイスを表すものとする。 S はシグネチャ宣言を表すものとする。これはメソッド宣言とは異なり、メソッドボディを持たない形式である。また、構文中で引数の列として登場するクラス \overline{C} の中にはインターフェイスが含まれてもよいものとする。

- クラス宣言に次を加える : $\text{interface } I \text{ extends } \overline{I} \{ \overline{S} \}$
- クラス宣言に次を加える : $\text{class } C \text{ implements } \overline{I} \{ \overline{C} \overline{f} ; K \overline{M} \}$
- シグネチャ宣言に次を加える : $C \text{ m } (\overline{C} \overline{x})$
- 項に次を加える : $(I) t$

本当は部分型付けや型付け、評価にも手を入れるべきだが、略。

(2) インターフェイスが存在するとき部分型関係の結びの下で必ずしも閉じているとは限らないことを示す。結びとは 2 つの型の最小の上位型である。未着手。

(3) 条件式に対する Java の型付け規則はどのようなものかを示し、それが妥当であることを示

す。未着手。

演習 19.4.7. [***] (super). 未着手。

19.5 性質

演習 19.5.1. [***] (保存定理). 未着手。

演習 19.5.5. [*** \leftrightarrow] (FJ 実装). 未着手。

演習 19.5.6. [**** \leftrightarrow] (多相型実装). 未着手。