

# ビューティフルコード

ネットワーク応用通信研究所 / 楽天 技術研究所

Yukihiro "Matz" Matsumoto

まつもと ゆきひろ

matz@ruby-lang.org

# コードの美とはなにか

---

- コードとはなにか
- ソフトウェアは美しいのか

# コードは工業製品ではない

---

- よくある誤解
- ソフトウェア工場
- 大量生産しない
- コピーは一瞬

# コードは設計である

---

- 職人芸
- 一品もの
- 善し悪しがある

# コードは実用品である

---

- 実用に供してナンボ
- 用の美

# コードは読み物である

---

- CODE READING
- 教材
- 知識の宝庫
- オープンソース

# コードは人を感動させる

---

- パワー
- 効率
- 美しさ

# パワーの美

---

- 「できない」を「できる」  
に
- アルゴリズムの力

# 計算量

---

- O 記法

- $O(1)$

- $O(n)$

- $O(n \cdot \log(n))$

- $O(n^2)$

# ソートアルゴリズム

---

- バブルソート
- マージソート
- クイックソート

# 不可能を可能に

---

## Rubyの文字列処理

- SJIS + EUC
- + UTF-8
- → M17N

# 効率の美

---

- 生産性
- 簡潔さ

# 簡潔さは力なり

---

- Succinctness is Power
- Paul Graham

# 簡潔さは力？

---

- Brooksの生産性不変の法則
- 本質に集中
- 実行可能擬似言語

# JavaによるHello World

---

```
class Sample {  
    public static void main(String[] argv){  
        System.out.println("Hello World");  
    }  
}
```

# RubyによるHello World

---

```
print "Hello World\n"
```

# 冗長の排除

---

- DRY
  - Don't Repeat Yourself

# DRY

---

```
class User<ActiveRecord::Base  
end
```

# 怠惰のための勤勉

---

- 手抜きは美しくない
- 苦勞を見せびらかすのは粹じゃない

# 水鳥のごとく

---

- 水上は優雅に
- 水面下では懸命に

# シンプルは美しい

---

- シンプルとはなにか
- しばしば誤解される

# 現実には複雑だ

---

- ソフトウェアも複雑だ
- 避けがたい現実
- 現実を直視する

# 人の心は単純じゃない

---



- 単純さが好き
- 複雑さも好き
- 簡単な問題がいい
- 難しい問題もいい

# 単純さの罫

---

- 単純さはゴールじゃない
- どこを単純にするか

# 人間にフォーカス

---

- 思いやりはうれしい
- やさしいものは美しい
- 人間のためのソフトウェア

# バランス

---

- 要は人間がどう感じるか
- 唯一の正解がない
- 非理系的

# 思考の流れ

---

〇〇を~して、~して、~する

```
puts STDIN.each.take(10)
```

- STDINを
- 各行取り出して、
- 最初の10行を切り出す

# 思考の流れ

---

```
main = do cs <- getContents  
          putStr (unlines (take n (lines cs)))
```

Haskell版

# 外面の美

---

- 人間がどう感じるか
- 時代・前提によって変化

# 人間は進化する

---

- 新たな知識
- 慣れ
- 最適化

# 人間を進化させる

---

- 現状に安住するより
- 前向きに進みたい
- 長い目で見た効率

# 内面の美

---

- 外には見えない
- 外面ほど価値観が変化しない

# 内面の美

---

- コードに内在
- プログラマには関心を持ってほしい

# コードはアートだ

---

- 人間のために
- 製品でなく作品

# プログラマ=アーティスト ト?

---

- アートを作る人
- 自覚があるか

# アーティスト

---

- 歯車ではない
- 作業員ではない

# アーティスト

---

- 創造的
- 自発的

# エンジニア+アーティスト

---

考慮すべきこと

- 納期

- 顧客

- チーム

# アーティストの条件

---

- 自覚
- 自発

# 美しいコード

---

理解に基づいたコード

- 人間（外面の美）
- 機械（外面の美）

# 「理解」が鍵

---

- どれだけ知っているか
- どれだけ理解できるか
- どれだけ考慮できるか

# まとめ

---

- コードは美しい
- コードはアート
- プログラマはアーティスト

# まとめ

---

## 誤解の蔓延

- ソフトウェア工場
- シンプルは善
- アーティストは「仕事」にならない

ご清聴ありがとうございました  
ございました