HackerのSoftware Engineering

ITPro Challenge! 2007/09/07

鵜飼文敏

ハッカーの流儀

- 普通の人をはるかに上回る生産性
- 高品質なソフトウェアを作りだす

ハッカーはどういうことを実践しているのか?

Hacker

定義

1. プログラム可能なシステムの細かい部分を探ったり、その機能を拡張する方法を探求したりすることに喜びを感じる人
2. 熱中して(さらには取りつかれたように)プログラミングする人、またはプログラミングを単に理論化するのではなく、プログラミングを楽しむ人
3. Hack valueを評価できる人
4. 手早くプログラミングするのが得意な人
5. ある特定のプログラミングのエキスパート、または頻繁にそれを使って仕事をする人(ex. A UNIX hacker)

(ハッカーズ大辞典より)

Hacker ethic

1. 情報の共有が実際にとても役に立つ善であると考え、フリーソフトウェアを買いたり情報や計算資源へのアクセスを実現することによって自分の技術を分け与えるのがhackerの倫理的義務であるという信念

(ハッカーズ大辞典より)

ソフトウェアの作り方

1. 要求仕様
   - 作るものを決める
2. 設計
   - どのように作るか考える
3. 実装
   - プログラムにする
4. テスト
   - 正しく作れたか確かめる
5. デバッグ
   - 正しく動くように直す
6. チューニング
   - 性能を改善する

ハッカーが作る場合

- 個人もしくは少人数のグループで
- 自ら作るものを決める
  - 利用できるもの・新規に作るべきもの
- 設計しつつ実装
- こまめにマイルストーンを設定
  - 小さい単位で動くもの・後で使うものから
  - テストしやすさ・デバッグしやすさ

要求仕様

- 目標を設定する
  - 何を作るべきか?
  - どのようなものを作れば便利になるか? 役に立つか?
  - 既存のシステムはどうなっているか?
    - 何は作らなくていいか? 利用できるか?
    - 何ができて、何ができないか? 限界はどこか?
    - それぞれどのように関係しているか?

Design document

- プロジェクトの説明
  - 何を作るか
    - どのように使われるか
  - なぜ作るのか、その理由
  - どのように作るか、その方針
  - 用語説明
- グループで開発する場合には必要
- たくさん書かないといけないような場合は問題が複雑すぎる。

設計

- 実現可能性、理解のしやすいシステムに
  - 何を利用するか
  - どのように利用されるか
    - どのようにテストするか
  - API、外部インターフェイス
  - アルゴリズムとデータ構造
  - デザインパターン

実装

- プログラミング
  - さまざまなノウハウ
    - 言語のイディオム
    - アルゴリズムとデータ構造、デザインパターン
  - 読みやすいコード
    - Code reading, code review
  - 集中
    - 頭の中にプログラムを入れる

Code Reading

- コードを読みとく力
  - 言語特有の書き方
  - アルゴリズムとデータ構造の実装の仕方
  - デザインパターン
  - わかりやすいネーミング

Code Review

- 人にコードを見てもらう/人のコードを見る
- 違う目で見ると気付くことがたくさんある
  - 細かく切り刻んで、起こりうる問題を見いだす
  - 別実装の提案
  - 明らかでない点を明らかにする
- レビューは新しいコードを書くのと同じくらい重要
- 開発チームを育てる

テスト

- ユニットテスト
  - テストしやすいコードを書く
  - 使いやすいAPI
  - ユニットが信頼できることを保証
- 危険なパターンをどれだけ想像できるか?
- コードを書いた時の思い込みをどれだけ捨てられるか?

デバッグ

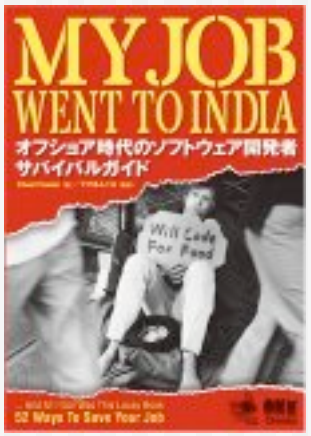
- どこがおかしいのか?
  - プログラムがどう実行されているのか
  - 範囲をせばめるテクニック
    - 防御的コーディング
    - ユニットテスト
    - デバッガ

チューニング

- どこがボトルネックか
  - アルゴリズムと計算コスト
  - 計測
    - プロファイラを使いこなす
  - 高速化
    - アルゴリズムの改善
    - 最適化テクニック

ハッカーに近づくには?

- ツールはフリーソフトウェア/オープンソースソフトウェアでそろう
- 必要なのは知識とその応用力
- プログラミングを楽しむ



ハッカーに必要な知識

- コンピュータサイエンス
  - Computer Science Unplugged
    - http://www.google.com/educators/activities/unpluggedTeachersDec2006.pdf
  - 要素とその組合せ方
    - アルゴリズムとデータ構造、デザインパターン
    - プログラミング言語、正規表現
    - コンパイラ、ライブラリ、OS、ハードウェア

知識の取得の仕方

- いい書籍を読む
- たくさんのコードを読む
- いろいろ試す
  - 改善してみる
  - 限界を知る、駄目なパターンを知る
- ハッカーにコードを見てもらう
  - コードを公開
  - コードレビュー

ハッカーと仕事する時の注意点

- おもしろい問題設定が重要
  - できてあたり前はおもしろくない
  - 実現不可能なことにとりくむのも無駄
- 情報の共有
- 競争と協力
- やる気・集中の持続
  - 割り込み・ミーティングを嫌う理由

宣伝

http://lc.linux.or.jp/

Any Questions?

Happy Hacking!