

Programming Heuristics

- Identify the aspects of your application that vary and separate them from what stays the same
 - Take what varies and encapsulate it
- Program to an interface, not an implementation
 - Specify behavior by name, not by working code
- Favor Composition over Inheritance
 - Use "has-a" rather than "is-a"
- Classes and code should be open for extension, but closed to modification
 - The Open-Closed Principle

Implications for nanoGoogle?

- **What might change in going from release 0.9 to 1.0 to 2.0 in nanoGoogle**
 - **Should we worry about future changes?**
 - **Should we make things work now?**
 - **Can we do both?**
- **Strategy pattern**
 - **Algorithm varies independently from clients that use it,**
 - **What are the algorithms in nanoGoogle?**

Design patterns

“... describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice”

Christopher Alexander, quoted in GOF

- **Name**
 - good name is a handle for the pattern, builds vocabulary
- **Problem**
 - when applicable, context, criteria to be met, design goals
- **Solution**
 - design, collaborations, responsibilities, and relationships
- **Forces and Consequences**
 - trade-offs, problems, results from applying pattern: help in evaluating applicability

Towards being a hacker

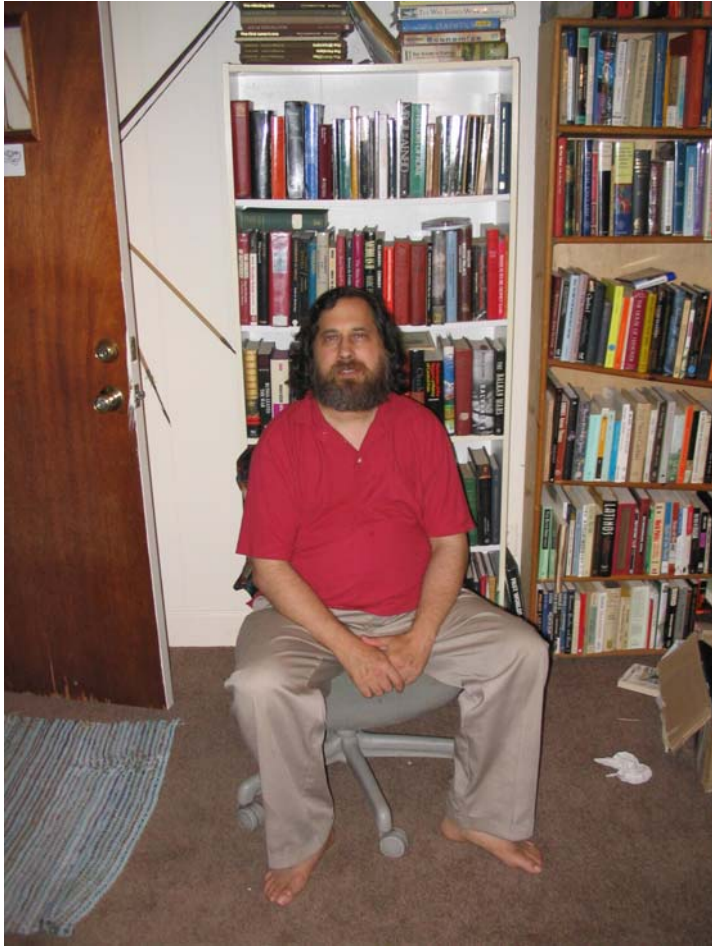
- See the hacker-faq (compsci 108 web page)
 - Hackers solve problems and build things, and they believe in freedom and voluntary mutual help. To be accepted as a hacker, you have to behave as though you have this kind of attitude yourself. And to behave as though you have the attitude, you have to really believe the attitude.
- The world is full of fascinating problems
 - no one should have to solve the same problem twice
 - boredom and drudgery are evil
 - freedom is good
 - attitude is no substitute for competence

You may not work to get reputation, but the reputation is a real payment with consequences if you do the job well.

Aside: ethics of software

- **What is intellectual property, why is it important?**
 - what about FSF, GPL, copy-left, open source,
 - what about money and monopolies
- **What does it mean to act ethically and responsibly?**
 - What about copying? stealing? borrowing?
 - No harm, no foul? Is this a legitimate philosophy?
 - Can software developers make a difference in the world?

Richard Stallman



- **Free Software movement**
 - Free as in speech
 - Not Free as in beer
- **Wrote emacs, gcc, gdb,...**
 - GNU's not Unix
- **Grace Murray Hopper award, Macarthur award, EFF Pioneer award,**