Being Simple

Daniel Jackson

For now, all rights reserved. Daniel Jackson, 2011.
a manager’s story

Fred as Frankenstein
› creator of the monster flow chart
› not even sure he understands it himself!

Jane, a disappointment
› gave her hard problems, but they turned out to be simple

story from:
two ways to construct software

I gave desperate warnings against the obscurity, the complexity, and over-ambition of the new design, but my warnings went unheeded. I conclude that there are two ways of constructing a software design: One way is to make it so simple there are obviously no deficiencies and the other way is to make it so complicated that there are no obvious deficiencies.

— Tony Hoare, Turing Award Lecture, 1980
how to be simple

› focus on problem, not solution
› question every assumption & ask stupid questions
› kill features: “the simplest thing that works” (STTW)
› explore crazy generalizations: everything’s a...
My view about the web is that it’s the great failure of computer systems research. Why did computer systems researchers not invent the web? And I can tell you the answer. It’s because it’s too simple...

If I had been there I would have mucked it up. I swear to God. The idea that you’re going to make a new TCP connection for every mouse click on a link? Madness! The idea that you’re going to have this crusty universal data type called HTML with all those stupid angle brackets? We never would have done that! But those were the things that allowed it to succeed.

—Butler Lampson, interviewed by Alan Kay in 2006 for the Computer Museum
simple enough?

› nothing left to take away
› design decisions follow inevitably
› the ‘duh’ effect
magsafe (apple, 2006)
sitekey (passmark, 2005)

to counter phishing
  › show customer previously selected image
examples of simplicity

simple interface
› undo vs. preventing mistakes
› HTTP get vs. FTP protocol
› B&W in Lightroom vs. Photoshop

simple implementation too
› email address as username
› in Unix, File = seq[Byte]
› trash can for file deletion
› document as paragraphs
simple search

use the good, shun the bad

It is rarely possible for standards committees to remove imperfections from a language... But you have the power to define your own subset.
—Douglas Crockford, in Javascript: The Good Parts

The Skater’s Principle of Language Use:
Stay in the middle, where the ice is thicker.
—Michael Jackson, in Software Requirements & Specifications

There is danger and misery at the edges.
—Douglas Crockford, in Javascript: The Good Parts
good & bad: example

example: Javascript equals
  › always use !==, not ==

> 0 == ''
  true
> 0 == '0'
  true
> '' == '0'
  false

> 0 === ''
  false
> 0 === '0'
  false
> '' === '0'
  false
a relentless pursuit of simplicity

Inside every large program there is a small program trying to get out.
— Tony Hoare

Simplicity does not precede complexity, but follows it.
— Alan Perlis