The Joel Test:
12 Steps to Better Code

The Joel Test

In general, a score of ≤ 10 means you're in trouble.

1. Do you use source control?
2. Can you make a build in one step?
3. Do you make daily builds?
4. Do you have a bug database?
5. Do you fix bugs before writing new code?
6. Do you have an up-to-date schedule?
7. Do you have a spec?
8. Do you have quiet working conditions?
9. Do you use the best tools money can buy?
10. Do you have testers as part of the team?
11. Do you have interview candidates write code?
12. Do you do hallway usability testing?

Do you use source control?

- What are the benefits?
  - Allows multiple developers
  - Keep project in consistent state
  - Track changes and enable roll-back
  - Manage multiple versions
  - Save data in case of a disaster
  - Authoritative source for "daily build"

The ZFR should indicate the state of your repository.

Do you have a one-step build?

- A single script that
  - o [does a full checkout from scratch]
  - o rebuilds every line of code
  - o makes the binary executable files in all versions, languages and #ifdef combinations
  - o [creates the installation package]
  - o [creates the final media - CDROM, web site, …]

- All steps are *automated* and exercised regularly

- So, why is this valuable?

Do you do a daily build and test?

- Build the entire product every day and run a good test suite against the new version
  - build from checked in sources
  - automatic and frequent

- Goal: find out early that you've got problems and fix them before disaster strikes

- Benefits
  - Minimizes integration risk
  - Reduces risk of low quality
  - Supports easier defect diagnosis
  - Improves morale - developers, managers, customers

  The ZFR must include your build script/sequence and the output of a run.

Do you use a bug database?

- You can't keep the bug list in your head
  - Especially with multiple developers and multiple customers

  Moreover, looking at the history of bugs can be insightful!

- To characterize a bug consider:
  - how to reproduce it
  - expected behavior, actual behavior
  - responsible party, status, priority

- Best to use what is integrated with your code hosting
- Other examples: Trac, Bugzilla, text file

For the beta release assigment, we'll be asking to see a log of your bugs.

Do you fix bugs before writing new code?

Why not fix them later?

- Familiar with the code now
- Harder to find (and fix) later
- Later code may depend on this code (try building on quicksand…)
- Bugs may reveal fundamental problems
- Leaving all bugs to the end will make it harder to understand and keep the schedule

Do you have an up-to-date schedule?

- Keeps expectations realistic
  - For the team, customers, stakeholders
- Allows for more accuracy
  - Use experience to improve estimates
- Helps prevent feature creep
  - Don't take on anything without checking the schedule first

For the SDS, we asked for a schedule. For later releases, we ask you to highlight any changes, and keep all documents up to date.

Do you have a spec?

- Easier to fix problems at the design stage
- You know what you are trying to build
  - So do your teammates and customer
- More likely that you build the right thing
  - Pieces fit together
  - Customer is satisfied
- Conceptual integrity for your project
- Undocumented code has no commercial value
  - Joel's example: Netscape Navigator
  - Other examples: Vanu, Viaweb

Do you do hallway usability testing?

Grab someone in the hallway and make them use your code

- Key idea: get feedback *fast*
- A little feedback now >> lots of feedback later
- You will get most of the valuable feedback from the first few users

Joel's Disclaimer

- These are not the only factors that determine success or failure
  - A great team will not help if you are building a product no one wants
  - An incredibly talented team might produce an incredible product without these guidelines

- But all things being equal, these factors indicate a disciplined team that can consistently deliver

Other advice

Peopleware
Productive Projects and Teams
Second Edition Featuring Eight All-New Chapters
Tom DeMarco & Timothy Lister

THE NO ASSHOLE RULE
Building a Civilized Workplace and Surviving One That Isn't
#@?!#
delete
ROBERT I. SUTTON, PHD