Joel Test

CSE 3311 / 5324

Christoph Csallner
University of Texas at Arlington (UTA)

These slides contain material by:

→ The Joel Test: 12 Steps to Better Code
http://www.joelonsoftware.com/articles/fog0000000043.html

The Joel Test

- Joel Spolsky
  - Program manager on Microsoft's Excel team 1991-1994
  - Published the Joel Test in 2000
    - On "Joel on Software" blog (many great SE articles)
    - https://www.joelonsoftware.com/2000/08/09/the-joel-test-12-steps-to-better-code/
    - Has held up quite well over 20+ years now
  - Launched Stack Overflow in 2008
  - Launched Trello in 2011, sold it for USD 425M in 2017

The Joel Test: 12 steps to better code

- “I've come up with my own, highly irresponsible, sloppy test to rate the quality of a software team. The great part about it is that it takes about 3 minutes.”

- Practical entry to software quality
  - “you *really shouldn't* use it to make sure that your nuclear power plant software is safe”

- See software project management class CSE 4322

Use Joel Test to assess your project

- 12 yes/no questions
  - “Give your team 1 point for each "yes" answer.”
  - We will slightly adapt the test to our setup
  - Self-score your team’s process
  - Optional: Show off your score in team presentations

- Aim for a high score
  - “A score of 12 is perfect, 11 is tolerable, but 10 or lower and you've got serious problems. The truth is that most software organizations are running with a score of 2 or 3, and they need serious help, because companies like Microsoft run at 12 full-time.”

Questions 1,2,3

- Do you use source control?
  - – Git, Subversion, etc.
  - – Built-in redundancy
    - Makes it hard to lose important code, in case a disk dies
- Can you make a build in one step?
  - – From source code from repository
  - – Full build from scratch, all versions
  - – Full automation → Reproducible, minimize human errors
- Do you make daily builds?
  - – Notice accidental breaking changes early
  - – Do not block other developers from working

Question 4

- Do you have a bug database?
  - Have to keep track of bugs formally
  - Keeping them in your head will not work
  - Even for single-developer team
  - Minimum data for each bug:
    - complete steps to reproduce the bug
    - expected behavior
    - observed (buggy) behavior
    - who it's assigned to
    - whether it has been fixed or not
  - Simple five-column table may be enough (Google Docs)
  - Or: **Use Github’s built-in issue tracking system**

Questions 5,6

- Do you fix bugs before writing new code?
  - – The longer you delay fixing, the more expensive it will be
  - – Time for fixing a bug is harder to estimate than time to implement new code
  - →**The more bugs left to be fixed, the more uncertain the schedule**
    - See “Death March” book vs. “zero-defect”
- Do you have an up-to-date schedule?
  - – Other business tasks depend on code being ready
  - – Have to communicate, update schedule
  - – Forces focus on high-priority features

Question 7

- Do you have a spec?
  - Do not jump directly to code
  - Many features need discussion / refinement / feedback
    - **Faster for you to change docs vs change code**
  - Rule: “No code without spec“
  - E.g.: Each use case scenario = Separate issue in Github
    - Integrated commenting / assigning / managing like other issues
    - Makes it easy to refer from code commits to issue it implements
    - Makes it easier to read / understand commits

Question 8

- Do programmers have quiet working conditions?
  - Best work done in full concentration (“in the zone”)
  - Programmers, writers, scientists, basketball players, ..
  - Takes 15 minutes to get into the zone
  - Interruptions, noise, etc. quickly kick you out of the zone
  - →Find a **very** quiet work environment, try not to interrupt fellow software engineers
  - →Many developers at Microsoft have their own offices
    - →Maybe not an accident that Microsoft is one of the few software companies that has had sustained success over decades now

Questions 9,10,11

- Do you use the best tools money can buy?
  - Prevent programmers from waiting (being interrupted)
- Do you have testers?
  - Testers are typically cheaper than programmers
  - Microsoft/Google employ a lot of testers
  - Programmers still have to unit-test their code
- Do new candidates write code during their interview?
  - Would you hire a basketball player without knowing how he plays?

Question 12

- Do you do **hallway usability testing**?
  - People you run into in the hallway / gatherings
  - Make 5 people use your product, observe their intuitive reactions, observe how they (fail to) use your app’s features, listen to their feedback
  - This may give you a good preview how general audience will interact with your app
  - Also works with paper sketches, screenshots, or mocked-up system thrown together in prototyping tool

This course: Joel-10

- Do you use source control?
- Can you make a build in one step?
- Do you make daily builds?
  - Do you create a full build on each day you work on the project?
- Do you have a bug database?
- Do you fix bugs before writing new code?

Joel-10 (cont’d)

- Do you have an up-to-date schedule?
- Do you have a spec?
- Do programmers have quiet working conditions?
- Do you use the best tools money can buy?
  - Here: Are you using the best free tools available?
- Do you do hallway usability testing?
  - Ask your friends, roommates, fellow students

IN-CLASS EXERCISE: WHAT IS YOUR JOEL-10 SCORE?

What is your Joel-10 score?

I. Do you use source control?

II. Can you make a build in one step?

III. Do you make daily builds?

IV. Do you have a bug database?

V. Do you fix bugs before writing new code?

VI. Do you have an up-to-date schedule?

VII. Do you have a spec?

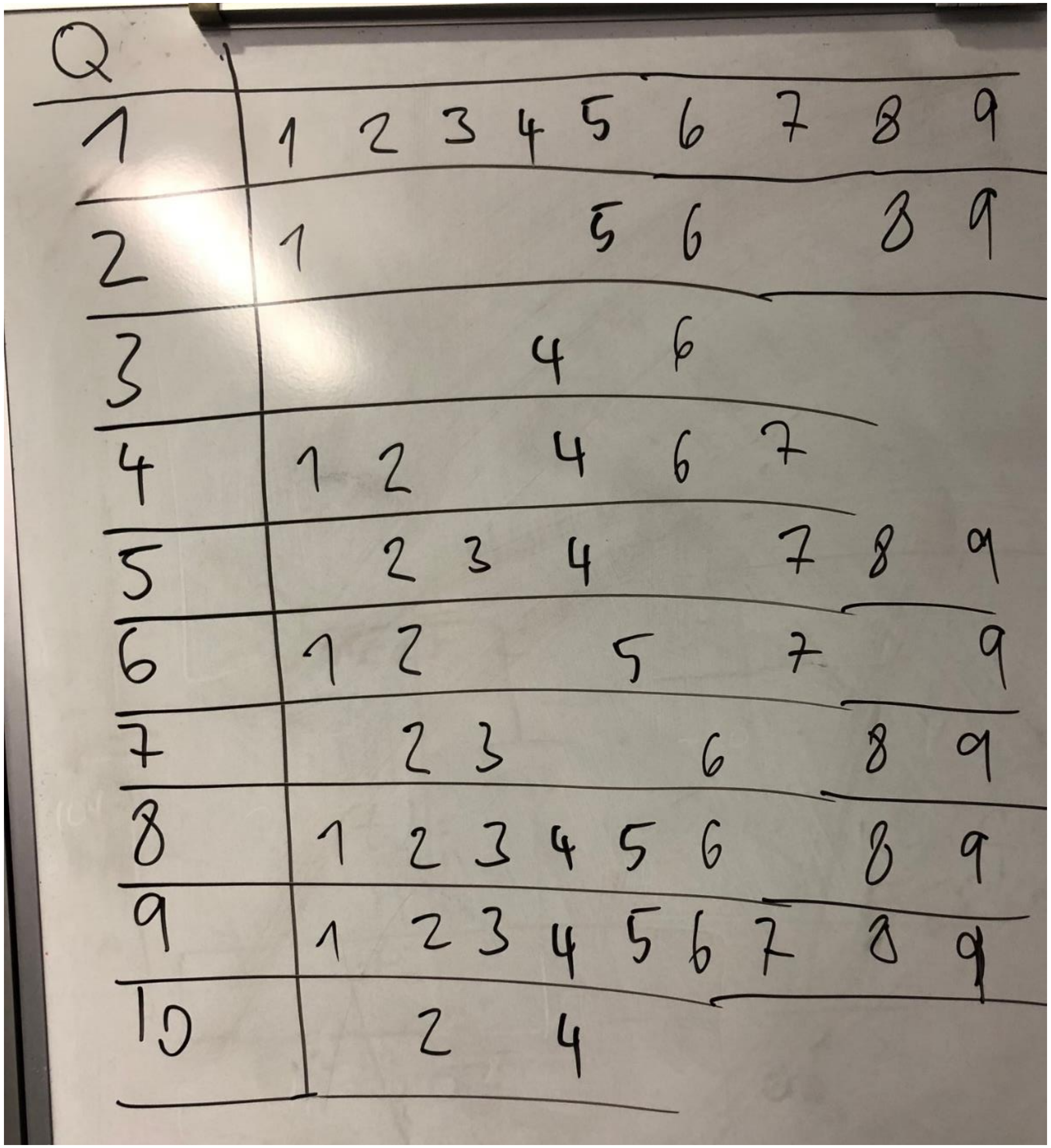
VIII. Do programmers have quiet working conditions?

IX. Do you use the best (free) tools money can buy?

X. Do you do hallway usability testing?

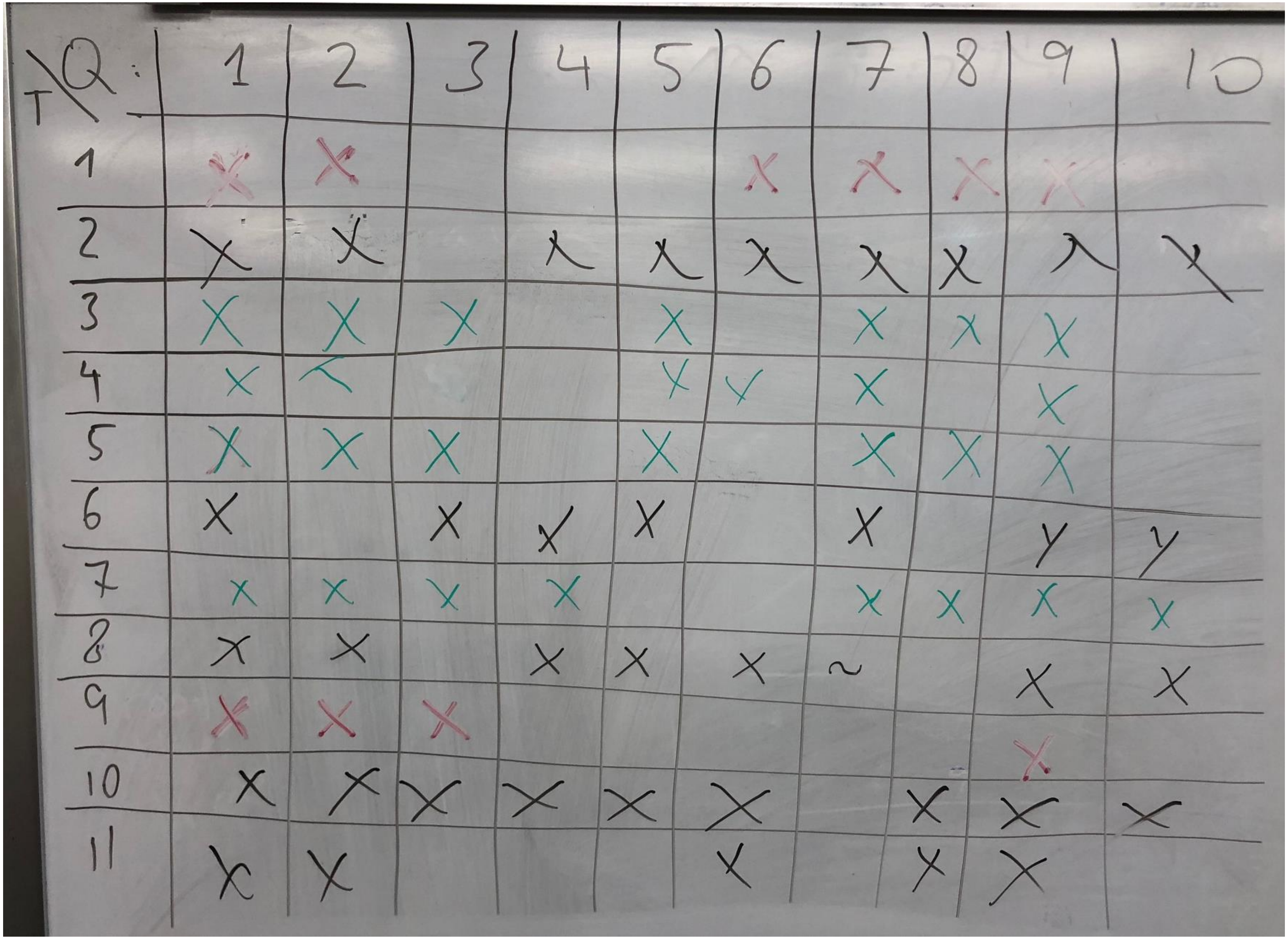
Results Fall 2017 (9/27/2017)

- All 9 teams use source control & use the best tools
- Only 2 teams do daily builds or hallway usability testing



Q									
1	1	2	3	4	5	6	7	8	9
2	1				5	6		8	9
3				4		6			
4	1	2		4		6	7		
5		2	3	4			7	8	9
6	1	2			5		7		9
7		2	3			6		8	9
8	1	2	3	4	5	6		8	9
9	1	2	3	4	5	6	7	8	9
10		2		4					

Results Fall 2018 (9/19/2018)



T \ Q:	1	2	3	4	5	6	7	8	9	10
1	X	X				X	X	X	X	
2	X	X		X	X	X	X	X	X	X
3	X	X	X		X		X	X	X	
4	X	<			X	✓	X		X	
5	X	X	X		X		X	X	X	
6	X		X	X	X		X		y	y
7	X	X	X	X			X	X	X	X
8	X	X		X	X	X	~		X	X
9	X	X	X						X	
10	X	X	X	X	X	X		X	X	X
11	X	X				X		X	X	