Masters of Un-done Software

Ward Cunningham
AboutUs.org
February 2008

Wield the Computer's Power

- Masters know what to do next by **reading** the un-done program
- What the un-done program does is **meaningful** even if not desirable
- The whole team **cooperates** to master modern programs
- Masterful teams use **tests** to guide their collaboration

Example One: WyCash

- Financial application had to get **calculations** right
- Interactive **reports** felt like spreadsheets with objects behind each row
- Fit-style tests **slipped** into WyCash as new kind of row
- Lesson: Abstracted **conversions** valuable, but uncommon
- Lesson: Analyst & developer can **pair-program**
- http://c2.com/doc/oopsla91.html
- http://c2.com/doc/oopsla92.html

result.htm - Microsoft Word

File Edit View Insert Format Tools Table Window Documents To Go Help

77% Recount

Basic Employee Compensation

For each week, hourly employees are paid a standard wage per hour for the first 40 hours worked, 1.5 times their wage for each hour after the first 40 hours, and 2 times their wage for each hour worked on Sundays and holidays.

Here are some typical examples of this:

Payroll.Fixtures.WeeklyCompensation			
StandardHours	HolidayHours	Wage	Pay()
40	0	20	$800
45	0	20	$950
48	8	20	$1360 *expected* $1040 *actual*

Page 1 Sec 1 1/1 At 1" Ln 1 Col 1 REC TRK EXT OVR E

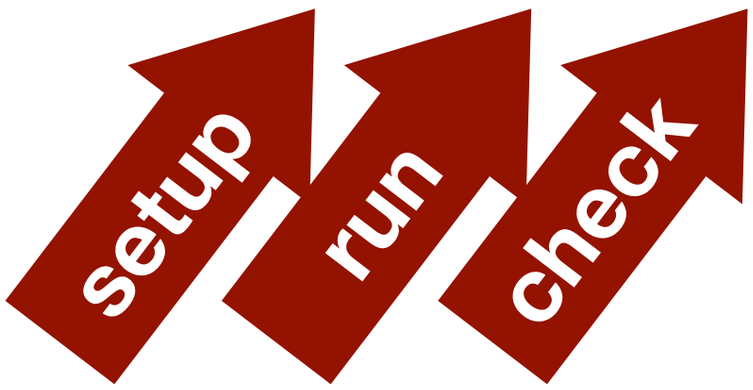
Example Two: IoGame

- Multi-player game in a powerful but **unfamiliar** language
- New assert semantics can test objects while **compiling**
- Test-Accelerated development calls for tests only when **uncertain**
- Lesson: Way too much of test code is about being **separate**
- Lesson: Tests **explain** what is expected better than types
- http://c2.com/~ward/io/IoGame
- http://c2.com/~ward/io/IoGamesUnitTests.pdf

Method Setup

We modify the built-in abstraction for time intervals to format using a less precise but more user friendly notation. We test this method by setting different interval magnitudes into the clones of the receiver (the prototype, Duration) and then trying the conversions. The first argument of the assert is everything up to the comma, in this case, two statements separated by a semicolon.

```
Duration do (
        asString := method (
                s := (asNumber floor)
                if ((m := ((s / 60) floor)) < 2, return s asString .. " seconds")
                if ((h := ((m / 60) floor)) < 2, return m asString .. " minutes")
                if ((d := ((h / 24) floor)) < 2, return h asString .. " hours")
                if ((w := ((d /  7) floor)) < 2, return d asString .. " days")
                if ((m := ((d / 30) floor)) < 2, return w asString .. " weeks")
                if ((y := ((d /365) floor)) < 2, return m asString .. " months")
                y asString .. " years"
        )
        assert (setSeconds (5); asString, "5 seconds")
        assert (setSeconds (90); asString, "90 seconds")
        assert (setSeconds (120); asString, "2 minutes")
        assert (setDays (14); asString, "2 weeks")
        assert (setDays (20); asString, "2 weeks")
)
```



Example Three: Eclipse Portal

- Automation of **long-running** collaborations
- Test scripts capture **screen fragments** to tell collaboration story
- Post processing organizes fragments as a **swim-lane** diagram
- Lesson: Plenty of web technology to **visualize** complex situations
- Lesson: Deploy tests as **documentation** to keep both up-to-date
- http://portal.eclipse.org
- http://c2.com/pnsqc2007/paper.pdf

committer election

	developer1	candidate	foo-dev	pmcmember	developer2	technology-pmc
0 start 1 2 3	4 5 6 7 8 nominate 9 10 11 12 13 14 15 16 save 19 20 21 22 23 (result) 24 variation 25 variation 26 variation 27	17 nominated as committer	18 vote started	28 29 30 (result) 31		
32 1 day later	55 56 57 58 59 (result) 60	49 voting summary 50 vote successful		33 34 35 votes 36 37 revert 38 39 40 vote 41 42 43 vote 44 (result) 45 46 variation 47 variation 48 variation	51 pmc approval needed	

Committer Election for technology.foo

Candidate: Karl Candidate [edit]
Your vote: +1 [votes]
Vote summary: 1/0/0 with 1 pending (7 days remaining)

from committer_vote/committer_vote

Lessons

- Test against the interface that best exposes your fears
- Couple test results to decision making
- Accept that tests are part of your product and invest accordingly