The Craftsman: 29
Dosage Tracking VI
Move the Date

Robert C. Martin
6 August 2004

...Continued from last month.

---

21 Feb 2002, 1130

Between April and July of 1939 the political and astronomical news continued to worsen. War seemed inevitable, and the Earth remained at the center of Clyde's most probable path. The Stockholm Contingent changed from a small gathering of scientists in the bar, to a small and secret network of scientists around the world. Though the group had no official leader, it was Lise Meitner who provided vision and direction. Based in Stockholm, she remained able to communicate with most of her colleagues around the world. Slowly and carefully she began to recruit them.

By September, when the fits and starts of hostilities finally erupted into full-scale war in Europe, the Stockholm Contingent had members in neutral and hostile nations alike. This group did not know exactly what it was going to do, but they were convinced that the world should be paying more attention to Clyde than to fighting. They were also convinced that the newly discovered Uranium fission reactions were the key defense against Clyde, and too terrible to be used as a weapon of war. They also knew that the warring states would consider their views to be treasonous.

Jerry and Jasmine walked ahead of us on our way back to the lab. The whole way they talked and laughed, and acted all buddy-buddy. Avery and I looked disgustedly at each other, but kept quiet.

Back in the lab, we sat down and started working on the test fixture again.

"So, what are we going to do with that date?" asked Jerry.

"We need to save it so that the application can access it when it needs to know today's date." I replied.

Avery still didn't like this, and made his opinion known by rolling his eyes; but he stayed mute.

"Right." Said Jerry. "Now let me show you how we do that."

He typed the following code.

```
public class UtilitiesTest extends TestCase {
  public void testDateGetsTodayWhenNoTestDateIsSpecified() throws Exception {
    Date now = new Date();
    Utilities.testDate = null;
    assertEquals(now, Utilities.getDate());
  }
}
```

Jerry looked over at Avery and asked: “Can you make that pass Avery?”

Avery sighed disdainfully as he took the keyboard. He said: “I suppose you’ll want me to use a *public* variable.” And then he typed the following without waiting for an answer:

```
public class Utilities {
  public static Date testDate = null;

  public static Date getDate() {
    return new Date();
  }
}
```

He clicked the test button and got a green bar.

“Great!” Said Jerry.

But Avery looked impatiently at Jerry and said:

“Yeah, but watch this.”

Avery started clicking the test button repeatedly. He got three or four green bars, and then the test failed with the message:

```
junit.framework.AssertionFailedError:
  expected:<Tue Feb 21 11:33:16 2000 (subjective)>
  but was: <Tue Feb 21 11:33:16 2000 (subjective)>
```

Before Jerry could react, Avery said: “You wrote the test wrong. Sometimes the two dates won’t be exactly the same. The difference is probably as small as a millisecond, but it’s enough to fail the test. I’ll fix that by writing the test a bit more intelligently.” And he changed the test as follows:

```
  public void testDateGetsTodayWhenNoTestDateIsSpecified() throws Exception {
    GregorianCalendar now = new GregorianCalendar();
    GregorianCalendar systemDate = new GregorianCalendar();
    Utilities.testDate = null;
    now.setGregorianChange(new Date());
    systemDate.setGregorianChange(Utilities.getDate());
    long difference = now.getTimeInMillis() - systemDate.getTimeInMillis();
    assertTrue(Math.abs(difference) <= 1);
  }
```

And then Avery turned towards Jerry and hit the test key repeatedly. He never once looked at the screen, but the test passed every time.

Jerry looked back at Avery coldly and said. “I concede the point, Avery. Thank you. However, that function is a bit cluttered now.” And Jerry took the keyboard back and made the following changes:

```
  public void testDateGetsTodayWhenNoTestDateIsSpecified() throws Exception {
    Utilities.testDate = null;
    assertTrue(DatesAreVeryClose(new Date(), Utilities.getDate()));
  }

  private boolean DatesAreVeryClose(Date date1, Date date2) {
    GregorianCalendar c1 = new GregorianCalendar();
    GregorianCalendar c2 = new GregorianCalendar();
    c1.setGregorianChange(date1);
    c2.setGregorianChange(date2);
    long differenceInMS = c1.getTimeInMillis() - c2.getTimeInMillis();
    return Math.abs(differenceInMS) <= 1;
  }
```

"OK, I think that expresses our intent a bit better." Jerry said as he watched the test repeatedly pass.

Avery just sniffed.

I was feeling a bit like I was in a war zone. I didn't know why Avery and Jerry were showing such animosity towards each other, but I was concerned that it was distracting them from the task at hand. So I grabbed the keyboard and said:

"OK, I think I know what the next test is." And I wrote:

```
public void testThatTestDateOverridesNormalDate() throws Exception {
  Date t0 = new GregorianCalendar(1959, 11, 5).getTime();
  Utilities.testDate = t0;
  assertEquals(t0, Utilities.getDate());
}
```

Jerry watched the test fail and said: "Right Alphonse. Now make it pass."

But Avery grabbed the keyboard and said: "I will." And he wrote the obvious code:

```
public static Date getDate() {
  return testDate != null ? testDate : new Date();
}
```

And as the test passed, he muttered: "And that's one hell of a lot of time wasted to get one simple and obvious line of code working."

"It wasn't *wasted*…." Jerry started. But then he stopped and shook his head. He took a deep breath and said: "OK, now we have to get the `DtrackContext` fixture to set the `testDate`. So he wrote the following test:

```
public class DTrackContextTest extends TestCase {
  public void testExecuteSetsTestDate() throws Exception {
    Date t0 = new GregorianCalendar(1959, 11, 5).getTime();
    DTrackContext c = new DTrackContext();
    c.todaysDate = t0;
    Utilities.testDate = null;
    c.execute();
    assertEquals(t0, Utilities.testDate);
  }
}
```

As I watched the test fail, I said: "OK Jerry, I see what you've done. You've created an instance of the fixture and jammed the `t0` date into `todaysDate` just like FitNesse[1] would. Then you expect t0 to somehow get set into the `Utilties.testDate` field. I suppose that it's the `execute` method of the fixture that does this?"

"Right. Remember that the FitNesse table we are working on looks like this:" and he pulled the table up on the screen."

DTrack Context
Today's date
2/21/2002

"As FitNesse processes this table it will jam the 2/21/2002 date into the `todaysDate` field of the `DTrackContext` fixture, and then it will call `execute()` on that fixture."

"OK, then I can make this test pass by doing this:" and I grabbed the keyboard and typed:

[1] www.fitnesse.org

```
public class DTrackContext extends ColumnFixture {
  public Date todaysDate;

  public void execute() throws Exception {
    Utilities.testDate = todaysDate;
  }
}
```

The tests passed, and I understood a bit more about how FitNesse worked. Then Jerry said: "You should run the acceptance test too." So I went to the `RegisterNormalSuit` page on the FitNesse server and clicked the test button. There was no change.

DTrack Context
Today's date
2/21/2002

"Good." Said Jerry. "We haven't broken anything."

"We haven't accomplished anything either." Said Avery.

Jerry glared at Avery, looked sadly over at me, then sighed and got out of his chair and walked over to Jean. The two of them started talking quietly. I looked at Avery and said: "Avery, what's wrong, why are you complaining so much?"

"He a dufus!" Said Avery. "He doesn't know *anything* about object oriented design, and all this testing nonsense is just make-work for morons. We've been working on this suit registration requirement for nearly four hours, and the only production code we've got to show for it is this:", and he pulled up the `Utilities` class on the screen.

```
public class Utilities {
  public static Date testDate = null;

  public static Date getDate() {
    return testDate != null ? testDate : new Date();
  }
}
```

"And even this class is nothing but a hack to allow testing. Four hours, three people, that's twelve man hours, and we've done jack cheese! I think this whole group is wacked out. I think Mr. C must be an idiot. How can he think he'll get this DTrack project done in two months when we squander hours and hours doing nothing but these stupid tests? If I were the captain I'd put Mr. C. out the airlock."

As Avery ranted, his voice grew louder and his eyes started to bug out. He didn't notice Jerry and Jean walking up behind him.

"That will be quite enough of that, young man!" Jean said sternly.

Avery snapped around, and his face went from anger to dismay. Jean was no longer the kindly grandmother we had come to know. She was something else entirely. It was clear from her tone of voice and the expression on her face that we were to remain silent and attentive, and that we were not to move.

"Speaking of the captain and his chief software craftsman that way is not acceptable behavior in this department. I won't tolerate it. Avery, yesterday you insulted Jason so badly he refused to work with you anymore. So I put you to work with Alphonse to see if any of his good manners might rub off on you. You two seemed to be getting along, and I dared hope that you had learned your lesson. But now you've been hostile to Jerry, and to the whole team, department, and even the *ship*. What shall we do with you?"

To be continued...

The source code for this article can be found at:
`www.objectmentor.com/resources/articles/CraftsmanCode/Craftsman_29_DosageTrackingSystem.zip`