Everything I Know about Programming, I Learned from Dilbert

Freely adapted from the article:

Principles of OO Design (Part 1)
by Alan Knight
Smalltalk Chronicles
Vol. 2, No. 1, March 2000

Never do any work that you can get someone else to do for you.

What not to do:

"Excuse me Smithers. I need to know the total bills that have been paid so far this quarter. No, don't trouble yourself. If you'll just lend me the key to your filing cabinet I'll go through the records myself. I'm not that familiar with your filing system, but how complicated can it be? I'll try not to make too much of a mess."

or:

```
compoundTest.getVector().addElement(
                               new FooTest(..) );
```

Never do any work that you can get someone else to do for you.

Instead:

"SMITHERS! I need the total bills that have been paid since the beginning of the quarter. No, I’m not interested in the petty details of your filing system. I want that total, and I’ll expect it on my desk within the next half millisecond."

Smithers actually understands his filing system, so he can probably do the work faster than we can, and he’s much less likely to mess everything up. *In seeking to do his job for him, we’re just making things worse.* They’ll get a lot worse when he switches over to that new filing system next week.

and:

```
compoundTest.addTest( new FooTest(..) );
```

Avoid responsibility.

If you must accept a responsibility, keep it as vague as possible.

For any responsibility you accept, try to pass the real work off to somebody else.

Avoid responsibility.

Instead of:

Maintain a collection of the what's-its to be frobbled.

Phrase it as:

Know which what's-its need to be frobbled.

The former is much too specific. The client doesn't care if you maintain a collection. It wants you to be able to report which what's-its require frobblification. That may be implemented by maintaining a collection — but maybe not!

Postpone decisions.

The great virtue of software is ***flexibility***.

Some decisions commit to a particular implementation.

So don’t make them!

Decisions are a **source of power**.

Let clients tell us how to solve problems.

Dilbert-based Practical Programming

Try not to care.

One of the great leaps in OO is to be able to answer the question "How does this work?" with "I don’t care".

Just do it!

An excellent slogan for projects that are suffering from analysis paralysis, the inability to do anything but generate reports and diagrams for what they’re eventually going to do.

Avoid commitment.

This is another way of expressing the principle of postponing decisions, but one which might strike a chord with you younger or unmarried programmers.

Dilbert-based Practical Programming

It’s not a good example if it doesn’t work.

At the end of the day, all that matters is code.

Steal everything you can from your parents.

A principle for those trying to make effective use of inheritance or moving into their first apartment.

Cover your #$%.

As in a bureaucracy, the most important thing is to make sure that it isn’t *your* fault. Make sure your code won’t have a problem even if things are going badly wrong elsewhere.