Microspeak: The bug farm

devblogs.microsoft.com/oldnewthing/20110920-00

September 20, 2011

Raymond Chen

In its most general sense, the term *bug farm* refers to something that is a rich source of bugs.

It is typically applied to code which is nearly unmaintainable. Code can arrive in this state through a variety of means.

- Poor initial design.
- An initial design that has been pushed far beyond its original specification (resulting in features built on top of other features in weird ways).
- Overwhelming compatibility constraints such that the tiniest perturbation is highly likely to cause some application somewhere to stop working.
- Responsibility for the code residing in people whom we shall euphemistically describe as "failing to meet your personal standards of code quality."

The term is most often used as a cautionary term, calling attention to areas where there is high risk that code you're about to write is going to result in a bug farm.

> Aren't we setting ourselves up for a bug farm?

> This could easily lead to a bug farm from different lifetimes for this various state objects.

The term is quite popular at Microsoft (**pre-emptive snarky comment**: because Microsoft software is all one giant bug farm). Here are some citations just from `blogs.msdn.com`:

> Layout runs under disable processing. The reason we did that is because, well, reentrant layout is a bug farm.

> A lot of testers suddenly realized that case sensitivity is a veritable bug farm on a project that thinks it is ready to go, but has not yet tried it.

> That type of implicit vs. explicit inference also turned out to be a bug farm.

> Did you forget to handle an entire set of test cases? Is the features implementation overly complex and going to be a bug farm?

Raymond Chen

Follow