# Why are singletons considered to be a bad practice? [duplicate]

Asked 15 years, 9 months ago Modified 11 years, 3 months ago

Viewed 2k times

0

**This question already has answers here**:

Closed 15 years ago.

## Duplicate:

> What is so bad about Singletons?

I was reading this question, and was surprised to see that (s)he considered a singleton to be considered a "bad practice," and in fact thought that this was common knowledge.

I've used singletons quite a bit in any project that uses iBatis to load the queries from XML. It great improves speed in these instances. I'm not sure why you wouldn't use them in a case like this.

So... why are they bad?

c# asp.net

Share

Improve this question

Follow

edited Jun 20, 2020 at 9:12

Community Bot
1 ● 1

asked Mar 24, 2009 at 18:52

Matt Grande
12.2k ● 6 ● 63 ● 89

## 5 Answers

Sorted by: Highest score (default)

3

They are not necessarily bad, just misused and overused. People seem inexplicably attracted to the pattern and look for new and creative ways to shoehorn it into their application whether or not it really is applicable.

Share Improve this answer

Follow

answered Mar 24, 2009 at 18:56

Andrew Hare
351k ● 75 ● 645 ● 641

---

3

They're a tool, and like any tool there are times you should use them and times you should use something else. In this case, it's very often true that something else (Factory, static class) would be better in situations that at first glance may seem appropriate for a Singleton.

When `Design Patterns` came out it seemed like everyone jumped on the Singleton bandwagon — they were everywhere, even places they shouldn't be. What you see now is a (perhaps well-deserved) backlash. Not that you shouldn't use them at all, but it might be a good idea to take a step back and look at all the options available.

Share Improve this answer Follow

edited Mar 24, 2009 at 19:26

answered Mar 24, 2009 at 18:57

Joel Coehoorn
**415k** ● 114 ● 577 ● 813

---

2

Singletons are not bad practice at all. In fact they are extremely useful for many situations. But they do have two major areas ripe for abuse and/or failure:

- Unit testability
- Multi-threading

Both can be handled, but beginners often neglect to do so (usually through ignorance) and it ends up causing far more trouble than they know how to deal with.

Share Improve this answer Follow

answered Mar 24, 2009 at 18:57

Rex M
**144k** ● 34 ● 291 ● 315

---

0

They are usually considered bad in conjunction with unit testing.

If you have a singleton, that means one or more of your classes are using it somewhere in some methods. That's a dependency you cannot spoof when unit-testing your class because the class is directly using it without requesting it through either its constructor or through a property.

That is usually why people say they are "bad".

Also, in a multi-threaded app, lots of people implement them badly enough that there is a chance for the instanciation to be called more than once.

Share Improve this answer Follow

answered Mar 24, 2009 at 18:57

Denis Troller
7,491 ● 1 ● 25 ● 37

That's not the only reason they're bad for multi-threading. Done wrong they can become a bottleneck that forces serialized access to a resource that could otherwise be used in parallel. – Joel Coehoorn Mar 24, 2009 at 19:00

---

0

They arent necessarily bad, its that they tend to be overused, and used a lot when they aren't needed.

Share Improve this answer Follow

edited Sep 5, 2013 at 3:53

answered Mar 24, 2009 at 18:57

Neil N
25.3k ● 17 ● 86 ● 146