Heap pollution is a technical term. It refers to references which have a type that is not a supertype of the object they point to.

List<A> listOfAs = new ArrayList<>();

List<B> listOfBs = (List<B>)(Object)listOfAs; // points to a list of As

This can lead to "unexplainable" ClassCastExceptions.

// if the heap never gets polluted, this should never throw a CCE

B b = listOfBs.get(0);

@SafeVarargs does not prevent this at all. However, there are methods which provably will not pollute the heap, the compiler just can't prove it. Previously callers of such APIs would get annoying warnings that were completely pointless, but had to be suppressed at every call site. Now the API author can suppress it once at the declaration site.

However, if the method actually is *not* safe, users will no longer be warned.

[Type safety: Potential heap pollution via varargs parameter subtrees [duplicate]](http://stackoverflow.com/questions/31357422/type-safety-potential-heap-pollution-via-varargs-parameter-subtrees)

|  |  |  |
| --- | --- | --- |
| How can I check that my code actually safe?  It's safe if the visitors only rely on the fact that the elements of subtrees are Tree<E>, and do not rely on the fact that subtrees is Tree<E>[]. If that is the case, then you should annotate the visitmethod with @SafeVarargs.   |  |  | | --- | --- | | [share](http://stackoverflow.com/a/31361366/6707834)[edit](http://stackoverflow.com/posts/31361366/edit) | answered Jul 11 '15 at 20:12  [newacct](http://stackoverflow.com/users/86989/newacct)  **74.8k**16111159 | |
|  | |  |  |  |  | | --- | --- | --- | --- | | |  |  | | --- | --- | |  |  | | the visit method is just iterating over the list of varargs. Is that safe> – [St.Antario](http://stackoverflow.com/users/2786156/st-antario) [Jul 12 '15 at 11:48](http://stackoverflow.com/questions/31357422/type-safety-potential-heap-pollution-via-varargs-parameter-subtrees#comment50714409_31361366) | | |  |  | | --- | --- | |  |  | | @St.Antario: yes – [newacct](http://stackoverflow.com/users/86989/newacct) [Jul 12 '15 at 17:57](http://stackoverflow.com/questions/31357422/type-safety-potential-heap-pollution-via-varargs-parameter-subtrees#comment50720820_31361366) |   add a comment |
| up vote1down vote | This [Java Tutorial](https://docs.oracle.com/javase/tutorial/java/generics/nonReifiableVarargsType.html#vulnerabilities) about Generics and Non-Reifiable Types covers your question  Heap pollution occurs when a variable of a parameterized type refers to an object that is not of that parameterized type |