

This PDF is generated from: <https://www.ferraxegalia.es/Sun-15-Oct-2023-28083.html>

Title: Super superconductor capacitor

Generated on: 2026-01-21 05:39:10

Copyright (C) 2026 GALICIA CONTAINERS. All rights reserved.

For the latest updates and more information, visit our website: <https://www.ferraxegalia.es>

-----

I'm currently learning about class inheritance in my Java course and I don't understand when to use the `super()` call? Edit: I found this example of code where `super.variable` is used: `class A { ...`

The automatic insertion of `super ()` by the compiler allows this. Enforcing `super` to appear first, enforces that constructor bodies are executed in the correct order which would ...

"super" object has no attribute "`__sklearn_tags__`". This occurs when I invoke the `fit` method on the `RandomizedSearchCV` object. I suspect it could be related to compatibility ...

I wrote the following code. When I try to run it as at the end of the file I get this stacktrace: `AttributeError: "super" object has no attribute do_something` class `Parent`: `def ...`

What is the difference between `List<T>` and `List<T> extends T`? I used to use `List<T>` extends `T`, but it does not allow me to add elements to it `list.add (e)`, whereas the `Li...`

As for chaining `super::super`, as I mentioned in the question, I have still to find an interesting use to that. For now, I only see it as a hack, but it was worth mentioning, if only for the differences ...

`super()` lets you avoid referring to the base class explicitly, which can be nice. But the main advantage comes with multiple inheritance, where all sorts of fun stuff can happen.

`super()` is a special use of the `super` keyword where you call a parameterless parent constructor. In general, the `super` keyword can be used to call overridden methods, ...

In fact, multiple inheritance is the only case where `super()` is of any use. I would not recommend using it with classes using linear inheritance, where it's just useless overhead.

The one without super hard-codes its parent"s method - thus is has restricted the behavior of its method, and subclasses cannot inject functionality in the call chain. The one ...

Web: <https://www.ferraxegalia.es>

