

So following on from the topic of a narrowing cast, we're now going to look at the opposite, a widening

cast.

Now, a widening cast, as you might guess, is really the opposite of a narrowing cast.

There can be situations where you want to change from the narrow view of an object, the superclass

view, if you like, to a wider view, giving you access to more elements in the object.

Let's call this the subclause view.

This means a variable that is a subclass can be assigned a Supercluster object at runtime.

Now, this might not sound as though it will be of much use, but what you will find is that it is mainly

used when a narrowcast was previously used and we want to restore the object back to its true identity.

This means after a widening cast, all the methods and attributes of the subclass will once again be

available for us to use in our programs.

Now, the way we implement a widening cast differs a little bit from a narrowing cast in that with a

narrow cast, we have the option of using the move statement or the equals operator to assign a subclass

reference to the variable or in widening casts.

We can't do this if we try to.

We're going to end up with syntax errors.

This is because the variable holding the reference to the superclass object can contain a reference

to any object of any class at runtime, and the syntax checker cannot determine if the variable is actually

pointing to the superclass or one of its subclasses.

So what we have to do in our code is tell ABAB not to carry out any syntax check, and we do this by

using the special casting operator, as you can see here, the question mark and equal to sign.

And this instructs ABBA to skip the syntax check until immediately before the assignment is executed

at runtime.

Now, as you can imagine, this opens up your code to the possibility of runtime errors, so you must

make sure you have code in place to catch any real time errors and to help with this Sappi of implemented

a system wide error message code move, cast error.

This one down here.