

Excelsior Knowledge Base

PRB: Class, method, or field not found during compilation

Article ID: 000011

Last Revised On: 29-Sep-2005

The information in this article applies to:

- Excelsior JET since version 2.5

SYMPTOMS

The compiler issues an error message `Class C not found or Undeclared identifier "C.foo"`, although all necessary classes and jars seem to be present in the project.

CAUSE

When a Java program executes on a (fully) dynamic virtual machine, such as Sun HotSpot, references between classes are resolved on demand at run time. This *late binding* feature, in particular, allows inconsistent programs to run successfully on such VMs, as described in the following example:

Suppose a certain class `Foo` is referenced by some other classes, e.g. its fields and/or methods are referenced. But `Foo` is never actually used when the application runs, i.e. it is never instantiated, none of its static methods are called, and none of its static fields are accessed. A dynamic virtual machine would not load the class `Foo` at all and therefore the program would execute flawlessly even if the file `Foo.class` is not present on the system.

Similarly, if the file `Foo.class` is present, but does not contain a particular method or field that is referenced by another class, execution on a dynamic virtual machine would fail only if and when the code referencing that field or method receives control.

On the contrary, Excelsior JET by default attempts to resolve all references between classes at compile time. This requires all imported classes to be available during compilation, and does not permit references to absent fields and methods.

RESOLUTION

If an error message `Class C not found or Undeclared identifier "C.foo"` is displayed, check your source and class files for consistency. Most probably, some part of your application was not recompiled, or some redundant class file was not removed.

Unfortunately, not all vendors check their class libraries for consistency, so third party jars (and zips) often contain references to absent classes, fields, or methods. We call such libraries *fuzzy*. For instance, Oracle's JDBC driver, `classes12.zip`, is fuzzy.

If your application is consistent, but requires a third-party fuzzy jar, do the following.

If a class from a third party jar references a class that is not present in that jar or elsewhere in the class path, set **CLASSABSENCE** option in the project file to "HANDLE":

```
-classabsence=handle
```

Note that if the class absent during compilation becomes available at run time, it will be loaded using the JIT compiler and the reference to that class will be successfully resolved.

Note for the users of Excelsior JET 3.7 and below: this option is available in Professional Edition only. If you are using the Standard Edition of Excelsior JET 3.7, set the **CLASS-ABSENCE** option to “IGNORE”:

```
-classabsence=ignore
```

With such setting, the compiler will issue warnings and insert code raising the respective exception at locations where the absent classes or members are used, so as to match a dynamic JVM behavior.

Another problem arises if the respective class is found during compilation, but it does not contain a field or method that is referenced by another class. In such case, enable the option **IGNOREMEMBERABSENCE** in the project file:

```
+ignorememberabsence
```

The options cited in this article can be set using the JET Control Panel, on the page **Options**.

REFERENCES

1. Excelsior JET User’s Guide (<http://www.excelsior-usa.com/jetdoc.html>), Chapter “The JET Control Panel”, section “Setting options”.
2. Excelsior JET User’s Guide (<http://www.excelsior-usa.com/jetdoc.html>), Chapter “Mixed Compilation Model”, section “Reflective Shield”.