How It Works

This topic describes how Osmorc works and what its limitations are.

OSGi and the IntelliJ IDEA module system

OSGi consists of several layers. For an IDE to support development of OSGi applications, handling the module layer is the most important part because it has the biggest impact on how applications are developed.

The module layer defines a module system that allows for the specification of dependencies between the modules - called bundles - on a pretty fine grained level. It's possible to define which packages of a bundle are visible to other bundles and which other bundles or packages are needed by a bundle.

So if an IDE doesn't support OSGi's module layer, the application will not be compilable in that IDE until the developer recreates the bundle dependencies in the module system of the used IDE. While this is always possible to some extent, it is very inflexible and error prone.

Only the basic dependencies between bundles can be recreated with IntelliJ IDEA module system since it only allows to define which module uses which other modules. It is not possible to define which packages a module exports and it's also not possible to define which packages are needed by a module. Only dependencies on whole modules and not parts of modules can be expressed.

The twofold approach

Osmorc uses a twofold approach to the described problem which will work in most cases but will fail in some special cases.

Osmorc uses Eclipse Equinox to resolve bundle dependencies. For each bundle dependency a module dependency is created no matter which part of bundle B bundle A depends on, module A will have a module dependency on module B. So this first step doesn't care about which packages are exported or imported. So bundle A can see and use all packages and classes in bundle B. The fine grained rules are taken care of by the Invalid Import inspection. It checks whether the packages and classes from bundle B imported in a class of bundle A are actually exported by bundle B and imported by bundle A. If this check fails, the import or usage is marked as an error.

Limitations

The twofold approach implemented in Osmorc has several limitations. Inspection errors don't prevent compilation. So if the Invalid Import inspection marks usages and imports as errors, those errors won't prevent the compilation of the application. So it's possible to change a manifest file in a manner that makes the imports of a class invalid for Osmorc. If the class is not opened and checked by the developer and the Invalid Import inspection is also not via the **Analyze | Inspect Code** action, the developer will stay unaware of bugs introduced by his change. Only at runtime will the problem become apparent. So it's always a good idea to run the Invalid Import inspection on all classes of a project frequently.

See Also

Reference:

- OSGi Facet Page
- OSGi

Web Resources:

- Developer Community