|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Changes** | **Reviser** |  |
| 2010.12.07 | Initial version | Jurgen |  |
|  |  |  |  |

# Purpose 🡺 Convention

## Dependencies

Dependency management in XCode is based on how Maven handles dependencies. When a package needs to verify, update or checkout its dependencies then the first step is to create the dependency tree. XCode follows some rules to solve dependency conflicts, these rules are listed below:

* Nearest
  + "Nearest definition" means that the version used will be the closest one to your project in the tree of dependencies, e.g. if dependencies for A, B, and C are defined as A -> B -> C -> D 2.0 and A -> E -> D 1.0, then D 1.0 will be used when building A because the path from A to D through E is shorter. You could explicitly add a dependency to D 2.0 in A to force the use of D 2.0
* Same Depth 🡪 Merge: If two dependency versions are at the same depth in the dependency tree, the version range descriptions will be merged using the following rules:
  + A higher version and wider version range is preferred
  + Type [x,) will win from (,y]
  + Type [x,) will win from [y,) if x<=y otherwise [y,) will win
  + Type [x,) will win from [y,z] if x<=y otherwise it will result in [y,)
  + Type [x,) will win from (,y][z,) if x>=z otherwise it will be result in [z,)
  + Type [x,y] will merge with [w,z] resulting in a possible [x,z] or [w,y]
  + Type [x,y] will merge with (,z][w,) resulting in a possible [x,z] or [w,y]

## Log Messages

When running XCode you might see messages related to the mediation of dependency types and versions.