The inside-out rule

The inside-out rule is used to determine the declaration to which a variable reference or method call corresponds. We illustrate with an example.

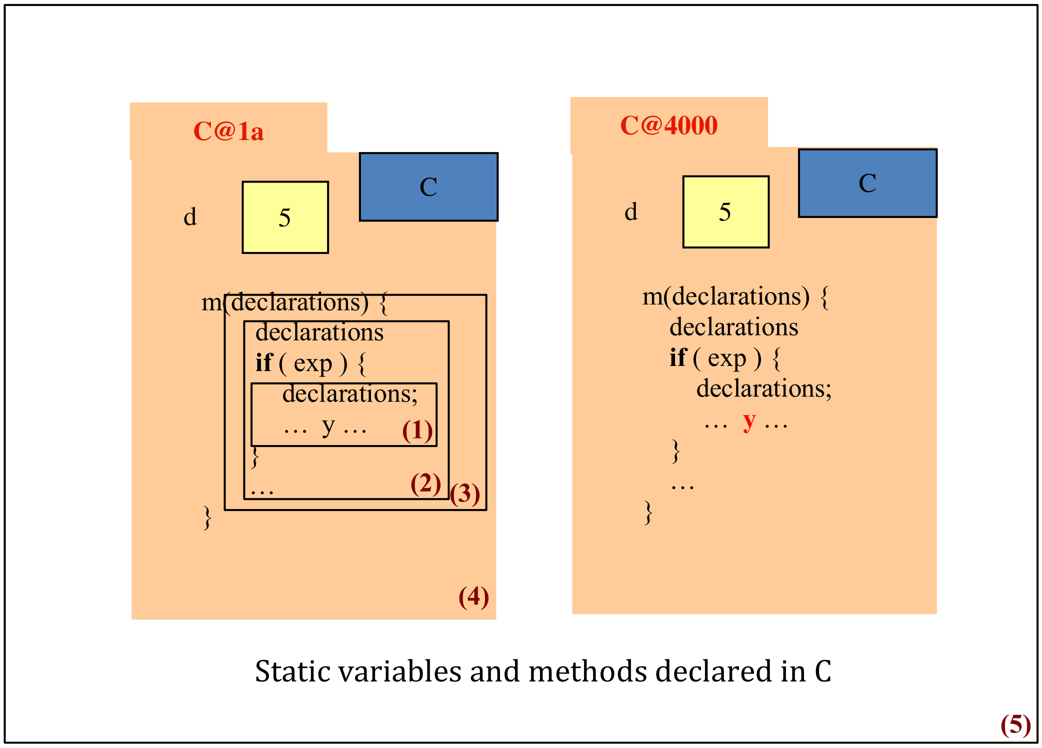
The box below contains objects of class C and all static variables and static methods declared in class C. Look at the red y in method m in the rightmost object of class C. To determine the declaration to which it corresponds, start looking in the closest scope, then in its surrounding scope, etc., until the appropriate declaration is found.

In this case, in the leftmost object shown below, the three scopes within the method have been outlined as rectangles. Thus, look:

first in the declarations in the block (marked (1));
then in the declarations in the surrounding scope (marked (2));
then in the declarations in the surrounding scope, the parameter declarations (marked (3));
then in the declarations in the surrounding scope, the fields and methods in the object (marked (4)) —if
looking for a method, use the bottom-up rule;
and finally in the static variables and declarations (marked (5)).

If no appropriate declaration is found, the variable reference or method call is syntactically incorrect and the program will not be compiled.

Container for C’s objects and static components


C@1a
C
d
5
m(declarations) {
declarations
if (exp) {
declarations;
… y …
}
…
}
(1)
(2)
(3)
(4)
C@4000
C
d
5
m(declarations) {
declarations
if (exp) {
declarations;
… y …
}
…
}
Static variables and methods declared in C
(5)