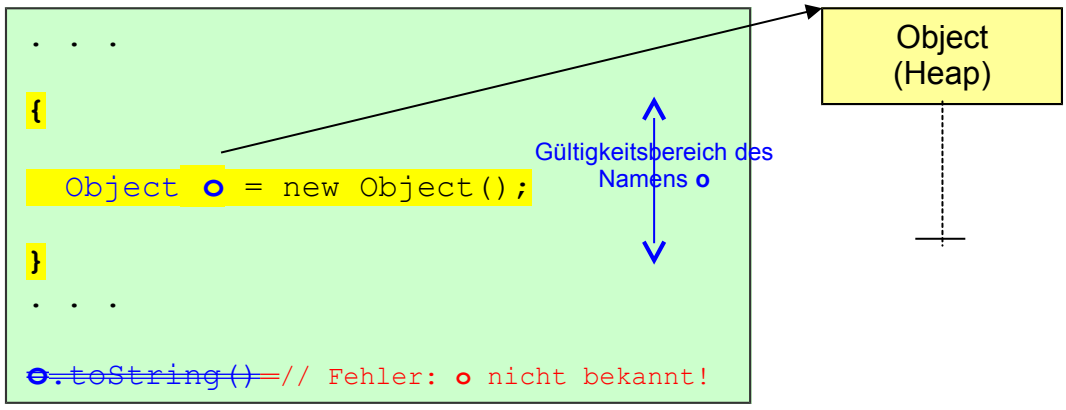
Gültigkeits- und Sichtbarkeitsbereich von Objekten

In einem Programm werden Objekte möglichst nahe an dem Ort (d.h. Codezeile) bzw. zu dem Zeitpunkt erstellt, wo und wann sie benötigt werden. Prinzipiell gilt, dass Objekte in einem ausführbaren Programmabschnitt erstellt werden müssen. Das bedeutet wiederum, dass sie an den jeweiligen „Gültigkeitsbereich“ gebunden sind (z.B. main). Die Verwendbarkeit eines Objekts entspricht diesem Gültigkeitsbereich - Objekte sind in einem Programm keineswegs immer und überall verwendbar, sondern nur dort, wo ihre Referenz (Objektname) gültig und sichtbar ist.

Um die Bedeutung des Gültigkeits- und Sichtbarkeitsbereichs von Objekten für die Programmierung richtig zu verstehen, muß der Unterschied zwischen der Referenz und dem eigentlichen Objekt beachtet werden. Die Referenz enthält die (virtuelle) Speicher-Adresse des eigentlichen Objekts. In einem Programm gibt es keine andere Möglichkeit, das Objekt anzusprechen, als durch Nutzung dieses Namens.

Eine Referenz verhält sich genauso wie jede lokale Variable. Für seine Nutzbarkeit ist das Lokalitätsprinzip maßgebend, das die Gültigkeit eines Bezeichners auf den Stack-Bereich des jeweiligen Programmabschnitts (den durch geschwungene Klammern definierten Block) begrenzt, in dem die Bezeichnung deklariert wurde. Außerhalb seines lokalen Bereichs ist ein Objektname nicht bekannt, nicht gültig und nicht verwendbar.



Formal bedeutsam ist das Lokalitätsprinzip aber nur für die Referenz. Die Objektdaten auf dem Heap sind dem Lokalitätsprinzip nicht unterworfen, sondern bleiben prinzipiell auch nach Verlassen des Gültigkeitsbereichs erhalten. Das bedeutet aber nicht, dass das Objekt immer und überall genutzt werden kann. Der Zugriff auf ein Objekt ist immer abhängig von der Existenz einer gültigen Referenzvariablen und diese hat immer eine irgendwie geartete lokale Gültigkeit.

Tatsächlich existiert ein Objekt nur solange, wie ein gültiger Name darauf verweist. Wenn es keine gültige Referenz mehr gibt, dann sorgt die automatische Garbage Collection (siehe nächstes Kapitel) dafür, dass die Objektdaten aus dem Speicher entfernt werden. Dies bedeutet praktisch, dass ein Objekt nur solange existiert, wie ein gültiger Name darauf weist. Wenn die Referenzvariable ihren Lokalitätsbereich verläßt oder sonstwie ungültig wird und wenn keine andere Referenzvariable mehr auf das Objekt weist, dann wird das Objekt durch die GC endgültig aus dem Speicher entfernt.

Der Gültigkeitsbereich der Referenzvariablen bestimmt die Lebensdauer der Objektdaten.

Natürlich ist es immer möglich, den beschränkten Gültigkeitsbereich eines existierenden Objekts durch Zuweisung an eine Referenzvariable mit erweitertem Gültigkeitsbereich zu vergrößern. Da es in Java jedoch keine Möglichkeit gibt, globale Variablen anzulegen, die sozusagen über den Klassen stehen, muß insbesondere bei der Herstellung von Klassenbeziehungen darauf geachtet werden, dass gültige Objektreferenzen verwendet werden (siehe unten Kapitel Klassenbeziehungen).