**הורשה המשך ...**

**משתנה מחלקה יכול להצביע בפועל על מופע של מחלקת צאצא שלו.**

**פולימורפיזם:**

* **ניתן במחלקת בן לדרוס מתודה של מחלקת האב.**
* **כאשר קוד שנכתב לאב רץ בפועל על מופע של הבן, הקוד "מתאים" את עצמו לבן ע"י כך שכאשר נקרא מתודה של האב הנדרס בבן, נקרא המתודה של הבן (הקריאה הולכת לפי המופע).**

**פולימורפיזם פירושו "רב צורתיות". המשמעות שלו כאן ש"קוד מתבצע בהתאם להקשר".**

**פולימורפיזם הוא גם רצון לאפשר כתיבת תוכנית המבטאת רעיון "מה מבוקש" מבלי להיכנס לפרטים.**

**בבן יופיעו:**

* **הבנאים של הבן,**
* **מתודות ושדות שקיימות בבן אבל לא באב.**
* **מתודות של האב שהבן דורס.**
* **ירושה מתבצעת מה-bytecode (הגרסה המקומפלת) של האב**
* **אין בקוד מקור בכדי לרשת.**

**בשפת Java קיימת הגדרת מחלקה מיוחדת שנקראית object.**

**היא נחשבת סבא-רבא של כל מחלקה.**

**משתנה מסוג object יכול להצביע על מופע של כל מחלקה, מעין פוינטר void.**

**יכול לשמש מימוש מבני נתונים האוגרים נתונים מכל סוג.**

**דוגמא לפולימורפיזם:**

**Class A**

**{**

 **public void a1()**

**{**

**}**

 **public void a2()**

**{**

 **a1();**

**}**

**} // A**

**class B extends A**

**{**

 **@override**

 **void a1()**

 **{**

 **}**

 **} // B**

**main()**

**{**

 **A a = new A();**

 **a.a1(); // 1**

 **a.a2(); // 1**

 **B b = new B();**

 **a = b; // upcasting**

**a = (A)b;**

 **// upcasting**

**a.a1(); //2**

**a.a2(); // 2**

 **b = (B)a ; // downcasting**

**// b = a; // will not compile**

**a = new A();**

**// b = (A) a; // will compile, //generates run time error**

**}**

**אמצעי ה-downcasting נועד להביא בחשבון את האפשרות שמשתנה של מחלקת אב מצביע בפועל על מופע של מחלקת בן.**

**במקרה שהאב באמת מצביע על מופע של הבן, ניתן להציב את הכתובת למצביע של הבן. הבעיה היא שזה יכול להיות לא נכון לכן זה נאסר בזמן קומפילציה אלא אם יש casting אז זה נבדק בזמן ריצה.**

**בכדי לא לעוף, אפשר להתנות:**

**If a instanceof B**

 **b = (B)a;**

**B b = new B();**

**b.a2();**

**יקרא ל-a2 של A שיקרא ל-a1 של B (ולא של A).**

**אין צורך לדרוס את a2 ב-B רק משום שהוא קורא למתודה קיימת ב- Aשנדרסת ב-B.**

**לעומת זאת,**

**A a = new A();**

**a.a2();**

**יקרא כמובן ל-a2 של A שיקרא ל-a1 של A.**