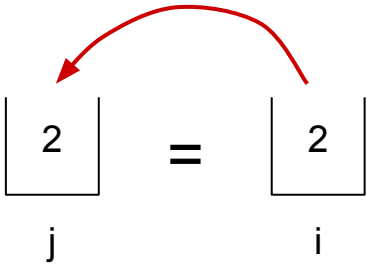
Referenser

Joachim von Hacht

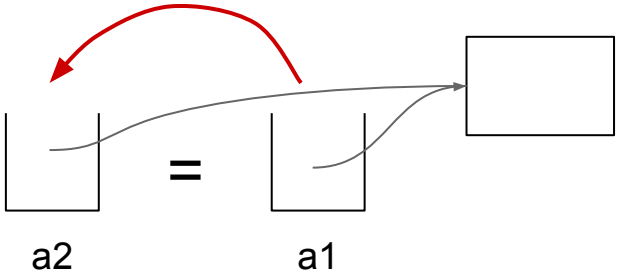
Vad kan hända med Referenser?

- Referenser kan bara peka på objekt
- En referensvariabel kan bara innehålla en referens
- Referensen kan vara till variabelns typ eller någon subtyp
- Vid tilldelning av referensvariabler kopieras referensen (pilen)
- En referensvariabel kan vara null, saknar referens (pekar inte på något objekt).

```
int i = 2;
int j;
j = i;
```




```
A a1 = new A();
A a2;
a2 = a1;
```



Referenssematik (1)

```
int[] a1 = {1, 2, 3};
int[] a2 = {4, 5, 6};      // Before
int[] a3 = doIt(a1, a2);   // Call
                            // After

int[] doIt(int[] x, int[] y) {
   x[0] = y[1];
   return x;
}
```

Visualisera semantiken genom att rita bild med variabler, referenser och objekt. Före och efter anrop!

Referenssemantik (2)

```
void program() {
   H h1 = new H(1);
   H h2 = new H(2);  // Before

   doIt(h1, h2);     // Call
   out.println(h1.i); // After
}

void doIt(H h1, H h2) {
   h1.i = 4;
   H tmp = h1;
   h1 = h2;
   h2 = tmp;
}

class H {
   int i;
   H(int i) { this.i = i; }
}
```

Visualisera semantiken genom att rita bild med variabler, referenser och objekt. Före och efter anrop!

Referenssemantik (3)

```
A[] as = new A[3];
A a1 = new A();
a1.i = 1;
a1.s = "aaa";

A a2 = new A();
a2.i = 2;
a2.s = "bbb";

as[0] = a1;
as[1] = a2;        // Before
as[2] = doIt(as); // Call
// After
```

Visualisera semantiken genom att rita bild med variabler, referenser och objekt. Före och efter anrop!

```
class A {
   int i;
   String s;
}
```

```
A doIt( A[] as ){
   as[0].s = as[1].s;
   return as[0];
}
```

Referenssemantik (4)

```
A a = new A();
a.arr = new int[]{1, 2, 3};
a.d = 1.5;
A b = new A();
b.arr = new int[]{4,5,6};
b.d = 2.5;  // Before

a = doIt( b); // Call
              // After
```

Visualisera semantiken genom att rita bild med variabler, referenser och objekt. Före och efter anrop!

```
class A {
   int[] arr;
   double d;
}
```

```
A doIt( A a ){
   A tmp = new A();
   tmp.arr = a.arr;
   tmp.d = a.d;
   return tmp;
}
```

Referenssemantik (5)

```
B b1 = new B("b");
A a1 = new A( b1 ); // Before
B b2 = doIt( a1 );  // Call
                    // After
```

Visualisera semantiken genom att rita bild med variabler, referenser och objekt. Före och efter anrop!

```
class A {
   int i;
   B b;
   A( B b ){
       this.b = b;
   }
}
```

```
B doIt( A a ){
   a.b.s = "c";
   return a.b;
}
```

```
class B {
   int i;
   String s;
   B ( String s ){
       this.s = s;
   }
}
```

Referenssemantik (6)

```
A a1 = new A(1, null);
A a2 = new A(2, a1);
A a3 = new A(3, a2);  // Before
a3.doIt(77);          // Call
                      // After
```

Visualisera semantiken genom att rita bild med variabler, referenser och objekt. Före och efter anrop!

```
class A {
    A a;
    int i = 1;
    A(int i, A a) {
        this.i = i;
        this.a = a;
    }
    void doIt(int i) {
        A t = a;
        while (t.a != null) {
            t = t.a;
        }
        t.a = new A(i, null);
    }
}
```

Göra sig av med Objekt

Är detta ett bra sätt att göra sig av med objekt?

```
void program() {
   int[] arr = {1, 2, 3};
   delete(arr);
   // arr gone??
}


// Method to delete an object
void delete(int[] arr) {
   arr = null;
}
```