MODIFIKATORI VIDLJIVOSTI

2020/21

PRIRODNO-MATEMATIČKI FAKULTET, UNIVERZITET U KRAGUJEVCU

UČAURIVANJE ENKAPSULACIJA

- Učaurivanje predstavlja mehanizam sakrivanja informacija.
- Zašto?

  Class creators vs. client programmer

  zaštita podataka i metoda

- Učaurivanje se izvodi upotrebom modifikatora vidljivosti/pristupa
- Modifikatorima pristupa (vidljivosti) se određuje opseg dostupnosti elemenata koda (polja, metoda, tipova). Svaki element ima definisanu vidljivost, implicitno ili eksplicitno definisanu.
- Modifikatori pristupa se mogu primeniti na:
  - Tipove
  - Elemente tipova – polja i metodi

  ne mogu na varijble definisane unutar metoda.

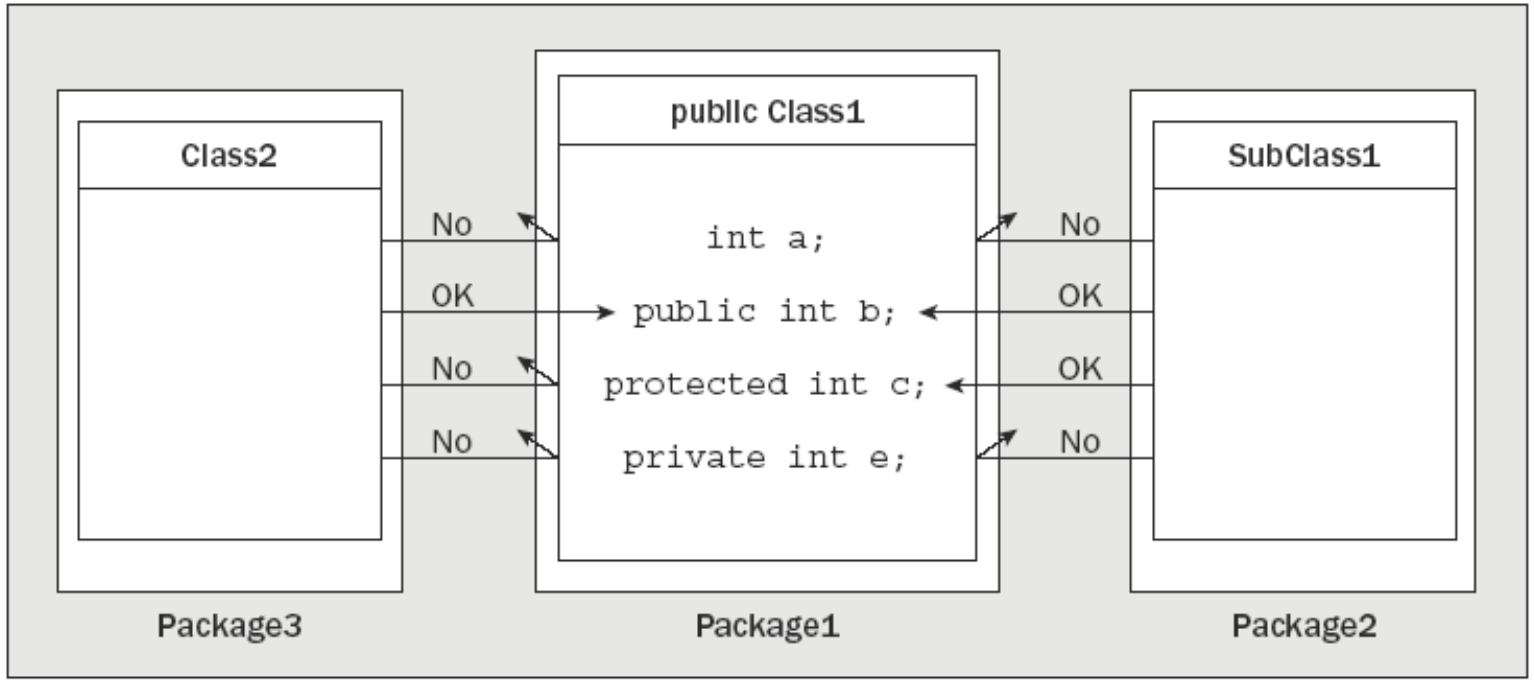
NASLEĐIVANJE - REVIDIRANO

- Objekat podklase uvek sadrži kompletan objekat superklase klase, ali to ne znači da su svi članovi superklase dostupni metodama koje su specifične samo za podbklasu!
- **nasleđivanje**: uključivanje članova bazne klase u izvedenu klasu na način da su dostupni (accessible) u izvedenoj klasi
- nasleđeni član bazne klase je onaj koji je dostupan u izvedenoj klasi

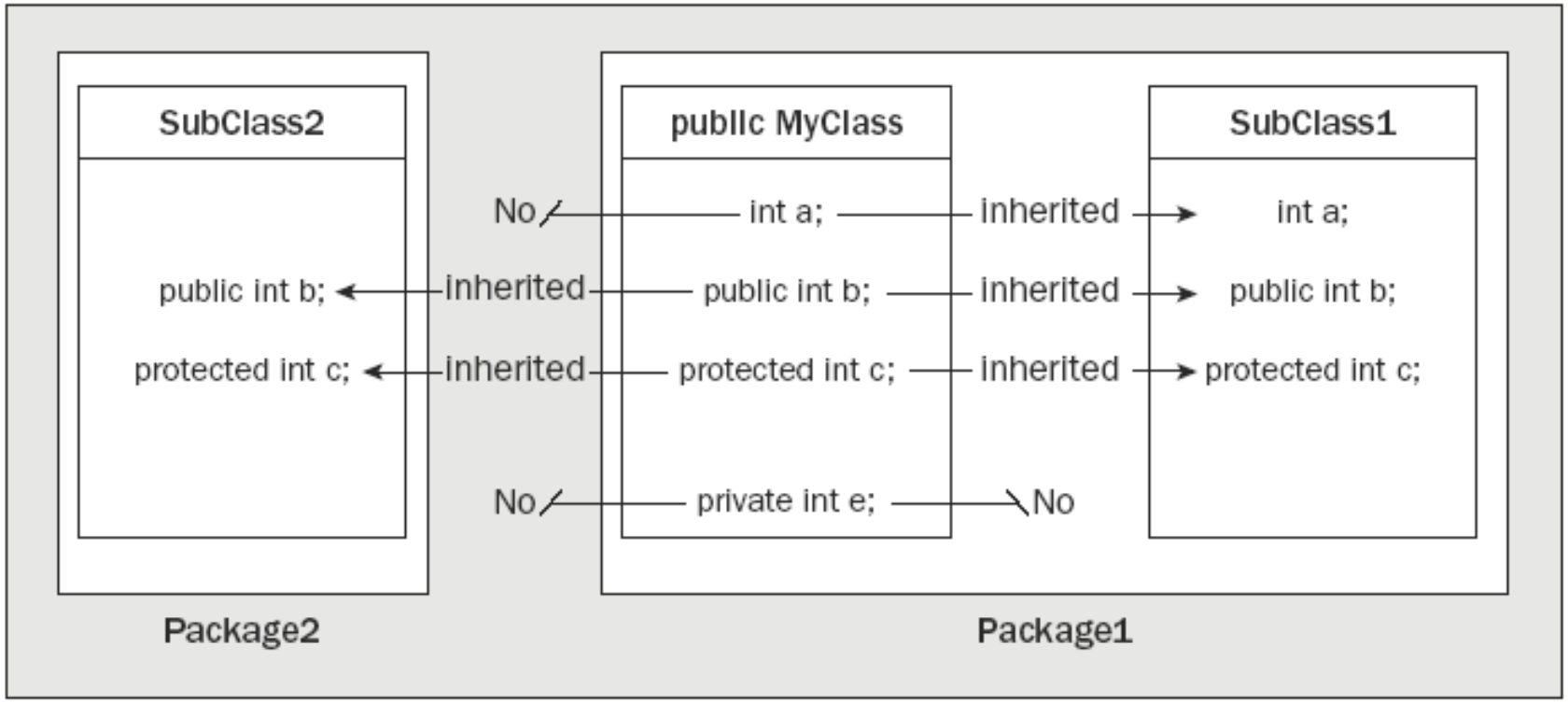
VRSTE MODIFIKATORA VIDLJIVOSTI

Tip \ vidljivost	Dostupno (vidljivo)			
	klasi	podklasi	paketu	bilo kome
Private	X			
protected	X	X*	X	
public	X	X	X	X
Nothig(default) Package Friendly	X		X	

VIDLJIVOST ČLANOVA



DOSTUPNOST NASLEĐENIH ČLANOVA



PRIVATE

```
class Alpha {
    private int iamprivate;
    private void privateMethod() {
        System.out.println("privateMethod");
    }
}
class Beta {
    void accessMethod() {
        Alpha a = new Alpha();
        a.iamprivate = 10;      // illegal
        a.privateMethod();      // illegal
    }
}
Beta.java:9: Variable iamprivate in class Alpha not accessible from class Beta.
        a.iamprivate = 10;     // illegal
          ^
1 error
Beta.java:12: No method matching privateMethod() found in class Alpha.
        a.privateMethod();         // illegal
1 error
```

PRIVATE

```
class Alpha {
    private int iamprivate;
    boolean isEqualTo(Alpha anotherAlpha) {
      if (this.iamprivate==anotherAlpha.iamprivate) //legal
            return true;
        else
            return false;
    }
}
```

PROTECTED

```
package Greek;
public class Alpha {
    protected int iamprotected;
    protected void protectedMethod() {
        System.out.println("protectedMethod");
    }
}
```

```
package Greek;
public class Gamma {
   void accessMethod(Alpha a) {
     a.iamprotected = 10;    // legal
     a.protectedMethod();    // legal
   }
}
```

```
package Latin;
import Greek.*;
public class Delta extends Alpha {
    void accessMethod(Alpha a, Gamma g) {
        iamprotected = 10;    // legal
        a.iamprotected = 10;  // illegal
        protectedMethod();    // legal
        a.protectedMethod();  // illegal
    }
}
```

PUBLIC

```
package Greek;
public class Alpha {
    public int iampublic;
    public void publicMethod() {
        System.out.println("publicMethod");
    }
}
package Roman;
import Greek.*;
class Beta {
    void accessMethod() {
        Alpha a = new Alpha();
        a.iampublic = 10;        // legal
        a.publicMethod();        // legal
    }
}
```

DEFAULT - PACKAGE

```
package Greek;
public class Alpha {
    public int iampublic;
    public void publicMethod() {
        System.out.println("publicMethod");
    }
}
package Roman;
import Greek.*;
class Beta {
    void accessMethod() {
        Alpha a = new Alpha();
        a.iampublic = 10;       // illegal
        a.publicMethod();       // illegal
    }
}
```

```
package Greek;
class Gama {
    void accessMethod() {
        Alpha a = new Alpha();
        a.iampackage = 10;      // legal
        a.packageMethod();      // legal
    }
}
```

ODABIR VIDLJIVOSTI ATRIBUTA BAZNE KLASE

- Metode koje sačinjavaju alat za komunikaciju sa spoljašnjim svetom/klasama se definišu kao `public`.
- Podaci članovi ne treba da budu `public` osim konstanti namenjenih za opštu upotrebu.
- Ako očekujete da će drugi ljudi koristiti vaše klase za izvođenje sopstvenih tada podatke članove definišite kao `private`, ali obezbedite `public` get-ere i set-ere.
- Koristite `protected` kada želite neometan pristup od strane klasa u istom paketu, a za klase iz drugih paketa dozvoljen samo ako su podklase.
- Izostavljanje modifikatora pristupa omogućava vidljivost člana klase u svim klasama paketa, dok je za klase van paketa to isto kao i upotreba private atributa.