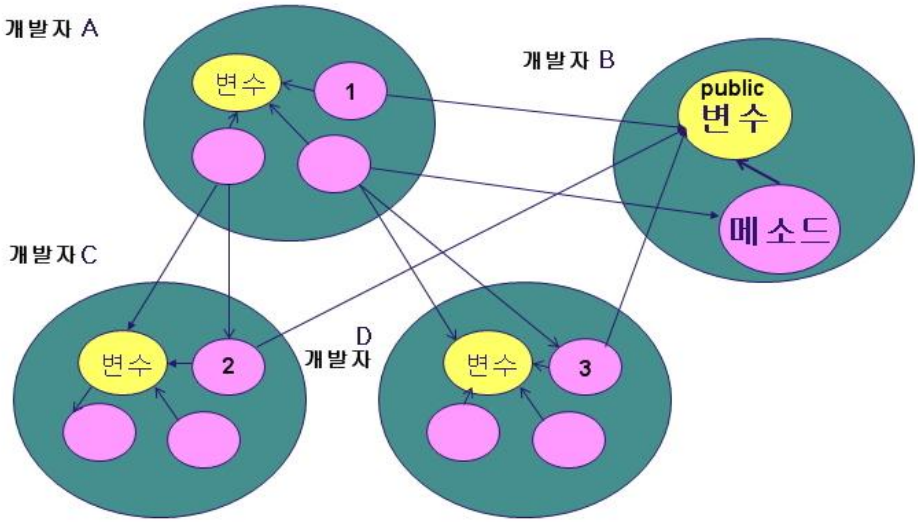
타인 내가

>> 접근 한정자 (Access Modifier, 제한자, 수정자)

1. 정의: 다른 클래스에서 현재 클래스의 필드(변수)와 메소드(함수)의 접근 가능 여부를 선언하는 기법

1) B 개발자의 변수에 A, C, D 개발자가 마음대로 접근하여 자신의 클래스에 로직을 구성할 수 있으므로 같은 처리 로직이 여러 클래스에 반복해서 발생될 수 있습니다. 이런 상황은 개발 속도가 떨어지고 개발이 종료 된후 소스 수정에 심각한 문제가 발생합니다. 기본적으로 변수는 모두 private 선언을 권장합니다.

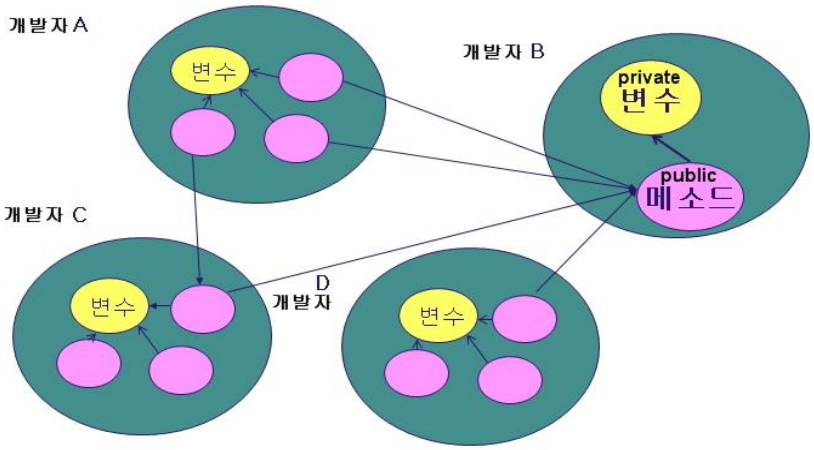


2) B 개발자의 변수가 private으로 선언이 되어 있어서 A, C, D 개발자는 접근이 불가능합니다.

이 때 로직이 필요한 경우 A, C, D 개발자는 B 개발자의 public 메소드에 기능 구현 여부를 확인 후 B 개발자가 구현한 로직을 공유하게 됩니다.

따라서 private 변수의 처리 로직은 전부 하나의 메소드에 구현이되어 모든 개발자가 활용하게되며 객체지향의 핵심 요소인 캡슐화(Encapsulation)를 지원하며 자바의 장점을 극대화 시키며 현재 국내에서 많이 사용되는 CBD개발의 핵심 이론으로 쓰입니다.

(CBD: Component Based Development)



2. 특징

1) 클래스간에 멤버 변수(필드)나 멤버 메소드(함수)에 접근하는 것을 제한할 수 있습니다.
2) 클래스간에 간섭을 막을 수 있음으로 콤포넌트의 독립성을 강화할 수 있습니다.
   . 필드 사용에 제한을 가할 수 있음.
3) 객체지향에서 캡슐화를 구현하는 핵심 기술입니다.
4) 접근 제한자의 상세화(상속, 패키지)

Modifier	Class	Package	Subclass	World
public	✔	✔	✔	✔
protected	✔	✔	✔	✘
*no modifier**	✔	✔	✘	✘
private	✔	✘	✘	✘

5) 일반적으로 필드(변수, 데이터)는 private을 선언,
   메소드(기능, 함수)는 public을 사용하면 OOP 구현에 크게 지장이 없음.

[예제]

>>>>> Data.java

```
package test;

public class Data {
  private String name = "";
  private int annual = 0;

  // setter : 멤버변수값을 지정하는 메소드
  public void setName(String name){
    this.name = name;
  }

  // getter : 멤버변수값을 가져오는 메소드
  public String getName(){
    return "* " + this.name + " * ";
  }

  public void setAnnual(int annual){
    this.annual = annual;
```

```
  }

  // 연봉 5%를 인상하세요.
  public int getAnnual(){
    return 
   }
}
```

>>>>> DataUse.java

```
package test;

public class DataUse {

  public static void main(String[] args) {
    // 객체 생성
    Data data = new Data();
    // data.name = "왕눈이";
    // The field Data.name is not visible
    data.setName("왕눈이");
    System.out.println(data.getName());

    data.setAnnual(20000000);
    // data.annual = 20000000;

    System.out.println(data.getAnnual());
  }
}
```

> **캡슐화** : 객체를 하나의 캡슐처럼 안전하게 만들어라 (정보 은닉)

- 멤버변수는 private으로, getter와 setter로 접근하시오 (java1에서는 여기까지)

  +

  Setter에 권한이 있는 검사, 검사 기록을 생성하여 이 모든 것을 더 넣어야 함

  => 그래야 캡슐화가 완성