-package : 파일(클래스, 인터페이스, 데이터, 이미지 등)을 보관. 폴더와 동일. 용도에 따라 분리해서 사용가능.  
3단 패키지를 기본 구조로 잡음.  
ex) com.gd.smart ┎ .controller -> 조작, 관리. main  
 ┠ .service -> 기능(메소드). 추가생성클래스 service  
 ┖ .util -> 공통기능  
  
 국가도메인, 업체명, 도메인명

-import : 경로 정보  
ex) import com.gd.smart.controller;

-접근권한(주로 사용하는 3가지 권한)  
1) public(≒default) : 어디서나 접근 가능  
2) protected(≒friendly) : 상속관계 + 동일 패키지  
3) private : 내 클래스 안에서만 쓸 수 있음  
\*전역변수의 경우 권한이 필요함. 권한 자체는 더 넓어질 수 있다(오버라이딩)

-캡슐화 : 데이터의 직접적인 접근을 막아서 데이터를 보호하는 것  
1) 장점 : 자료변조 막을 수 있음. 규칙적임.  
2) 동의어 : bean, DTO(Data Type Object), DO(Data Object), VO(Value Object)  
3) 전역변수의 권한은 무조건 private. 변수의 값 할당과 취득을 위한 setter, getter 메소드 생성  
 private int no; // 다른클래스에서 직접적으로 no에 값을 넣을 수 없음.  
 public int getNo() { // setNo메소드를 이용하여 변수를 변경

return no; // 변수를 불러오기 위해 getNo 메소드를 이용

}  
 public void setNo(int no) {

this.no = no;

}

-List(인터페이스) : 목록의 형태로 데이터를 관리  
 클래스 ┖ArrayList  
 ┖LinkedList  
  
 List<클래스> 변수명 = new ArrayList<클래스>();  
 ex) List<String> list = new ArrayList<String>();

|  |  |
| --- | --- |
|  | list |
| 0 | “abc” |
| 1 | “def” |
| 2 |  |
| 3 |  |

\*문자열만 들어갈 수 있음

-제너릭 : 형태 지정 <클래스, …> -> 대상의 자료 개수만큼 지정

-ArrayList 관련 내용  
1) list.add(값) : 리스트에 값을 추가한다  
2) list.get(인덱스번호) : 리스트에서 인덱스번호의 값을 가져온다.  
3) remove(인덱스번호) : 리스트에서 인덱스번호를 지운다.  
4) indexOf(값) : 리스트에서 값이 있는 인덱스번호를 취득한다.  
5) contains(값) : 리스트에서 값이 있는지를 확인. boolean으로 결과를 보여줌  
6) set(인덱스번호, 값) : 리스트에서 인덱스번호의 값을 주어진 값으로 변경한다.  
7) clear() : 리스트를 비운다.  
8) isEmpty() : 리스트가 비어있는지 확인.  
9) size() : 리스트의 개수를 출력

-for의 다른 사용 방법  
for(타입 변수명 : 목록형데이터) -> 목록형 데이터의 개수만큼 반복한다. 반복시 변수에 데이터를 변수에 넣어준다.  
 for(int data : list) {  
 System.out.println(data);  
 }