

6.6 애노테이션을 이용한 객체 관리

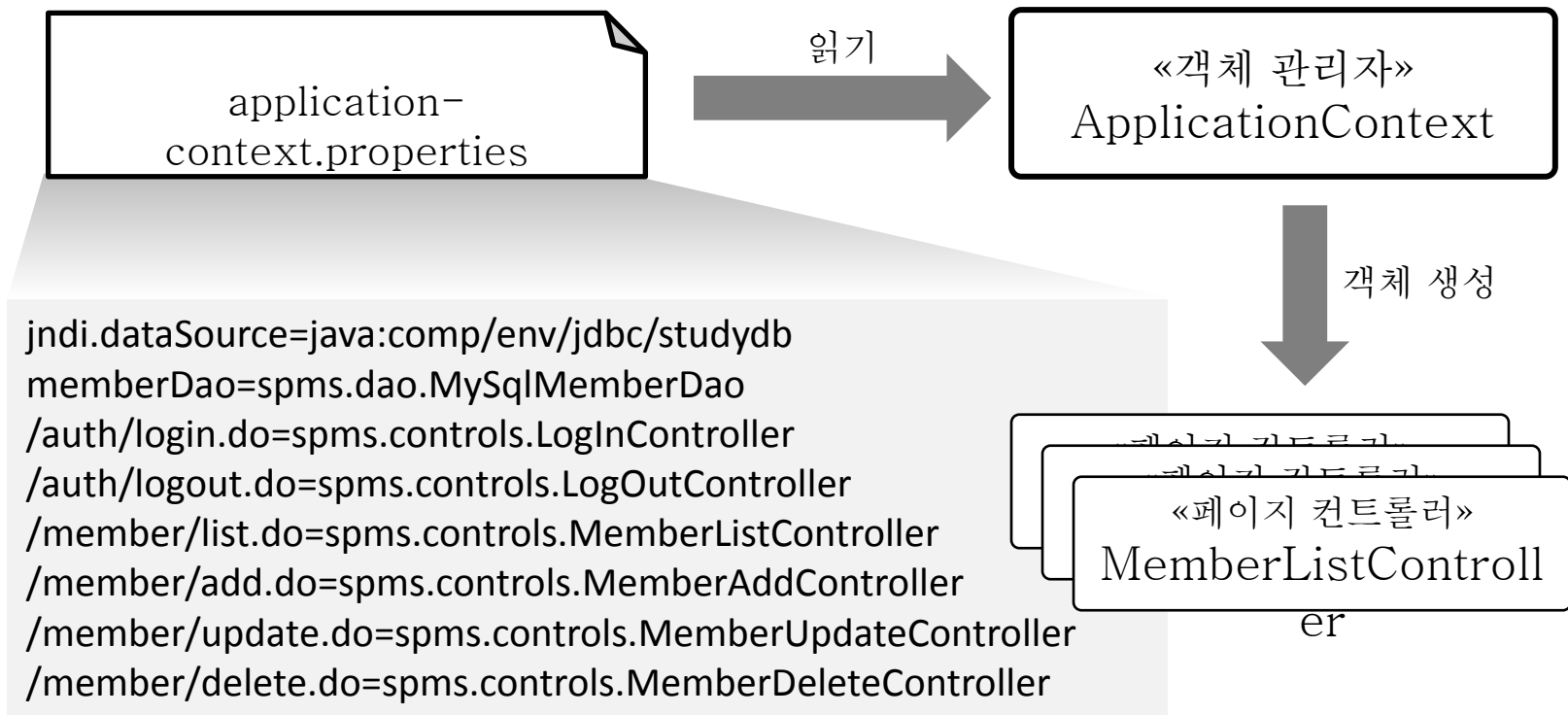
6.6 애노테이션을 이용한 객체 관리

“학습할 내용”

애노테이션을 이용하여
객체 준비를 자동화하기

6.6 애노테이션을 이용한 객체 관리

기존 방식 ➔ 프로퍼티를 이용한 방식

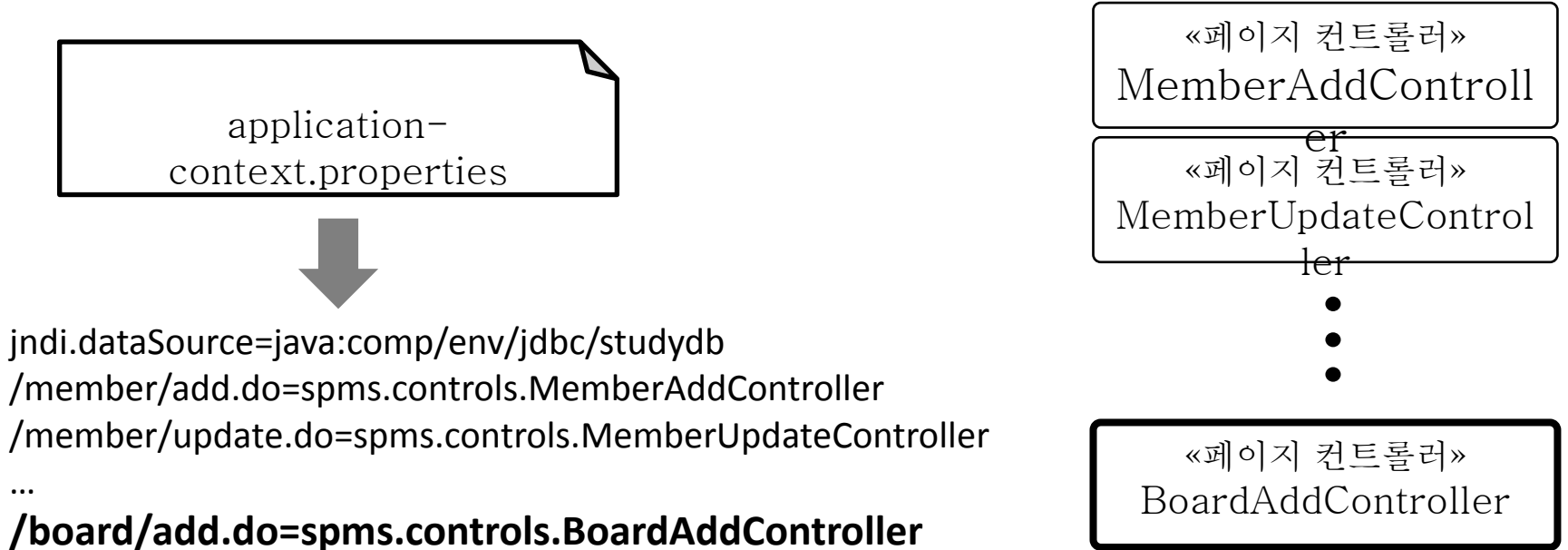


6.6 애노테이션을 이용한 객체 관리

기존 방식의 문제점

6.6 애노테이션을 이용한 객체 관리

DAO 또는 페이지 컨트롤러를 추가할 때
마다 프로퍼티 파일을 변경해야 함.



application-
context.properties

A diagram illustrating the configuration of controllers. On the left, a box labeled 'application-context.properties' has a large downward arrow pointing to a list of properties. On the right, a vertical stack of boxes represents the controllers. The first box is '«페이지 컨트롤러» MemberAddControll' (with 'er' on the next line), the second is '«페이지 컨트롤러» MemberUpdateControl' (with 'ler' on the next line), followed by three dots, and the last is '«페이지 컨트롤러» BoardAddController'. The boxes are connected by vertical lines.

jndi.dataSource=java:comp/env/jdbc/studydb
/member/add.do=spms.controls.MemberAddController
/member/update.do=spms.controls.MemberUpdateController
...
/board/add.do=spms.controls.BoardAddController

«페이지 컨트롤러»
MemberAddControll

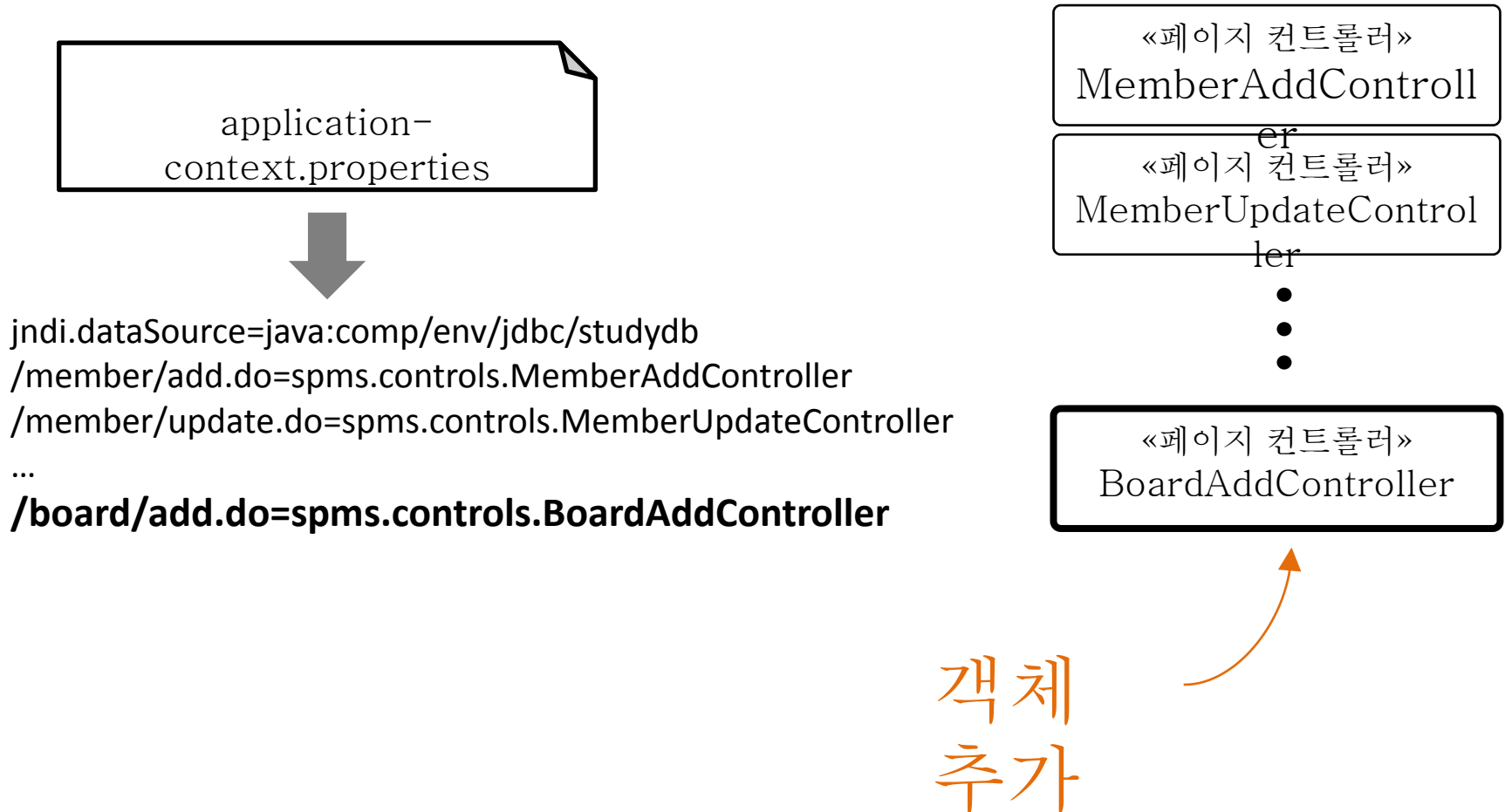
er
«페이지 컨트롤러»
MemberUpdateControl
ler

•
•
•

«페이지 컨트롤러»
BoardAddController

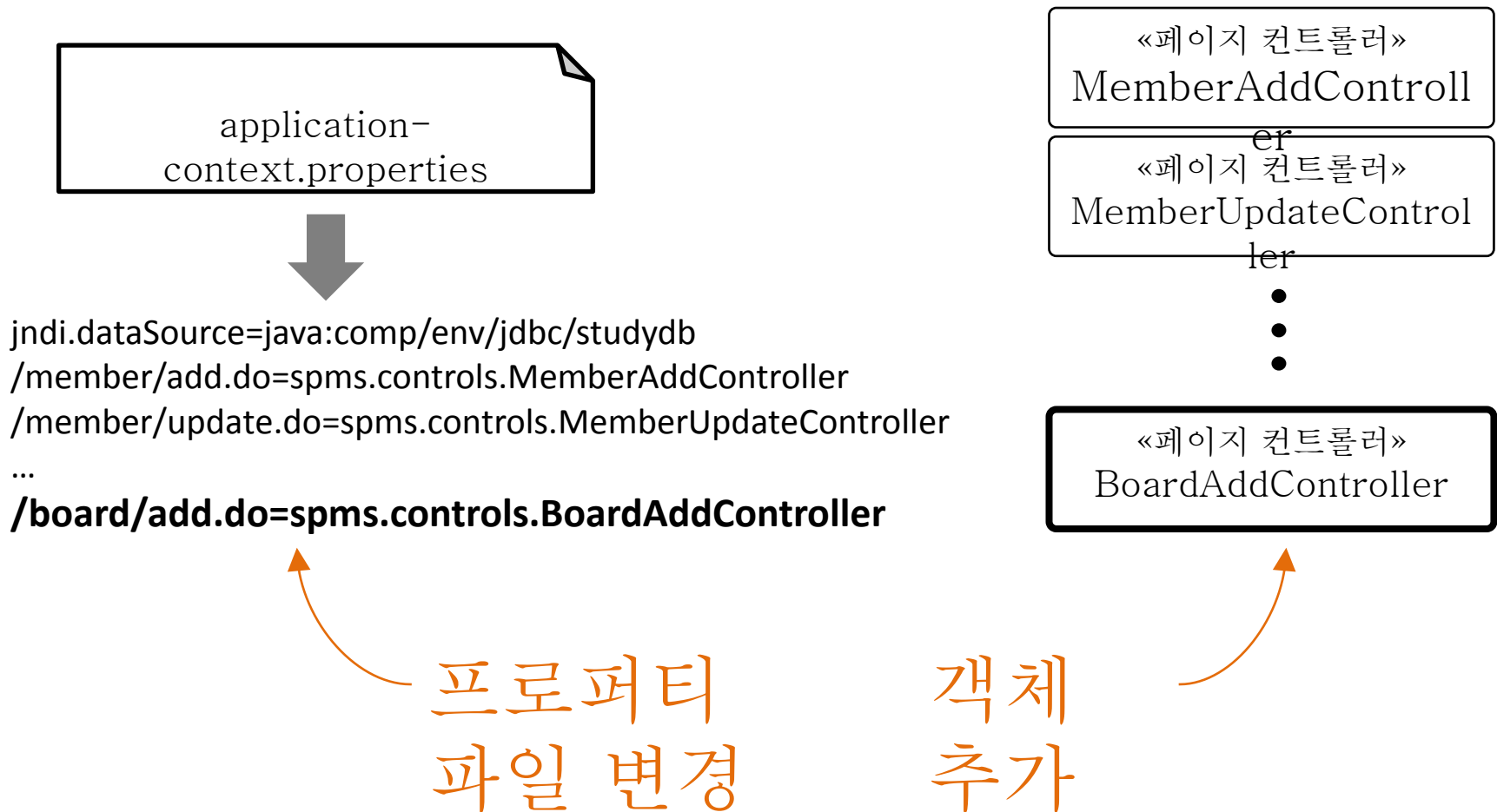
6.6 애노테이션을 이용한 객체 관리

DAO 또는 페이지 컨트롤러를 추가할 때
마다 프로퍼티 파일을 변경해야 함.



6.6 애노테이션을 이용한 객체 관리

DAO 또는 페이지 컨트롤러를 추가할 때
마다 프로퍼티 파일을 변경해야 함.




6.6 애노테이션을 이용한 객체 관리

해결 방안

6.6 애노테이션을 이용한 객체 관리

애노테이션을 이용하여
클래스 파일에 객체 정보를 담는다.



application-
context.properties

```
jndi.dataSource=java:comp/env/jdbc/studydb  
memberDao=spms.dao.MySqlMemberDao  
/member/add.do=spms.controls.MemberAddController
```

...

```
boardDao=spms.dao.MySqlBoardDao  
/board/add.do=spms.controls.BoardAddController
```

«DAO»
MemberDao

«페이지 컨트롤러»
MemberAddController

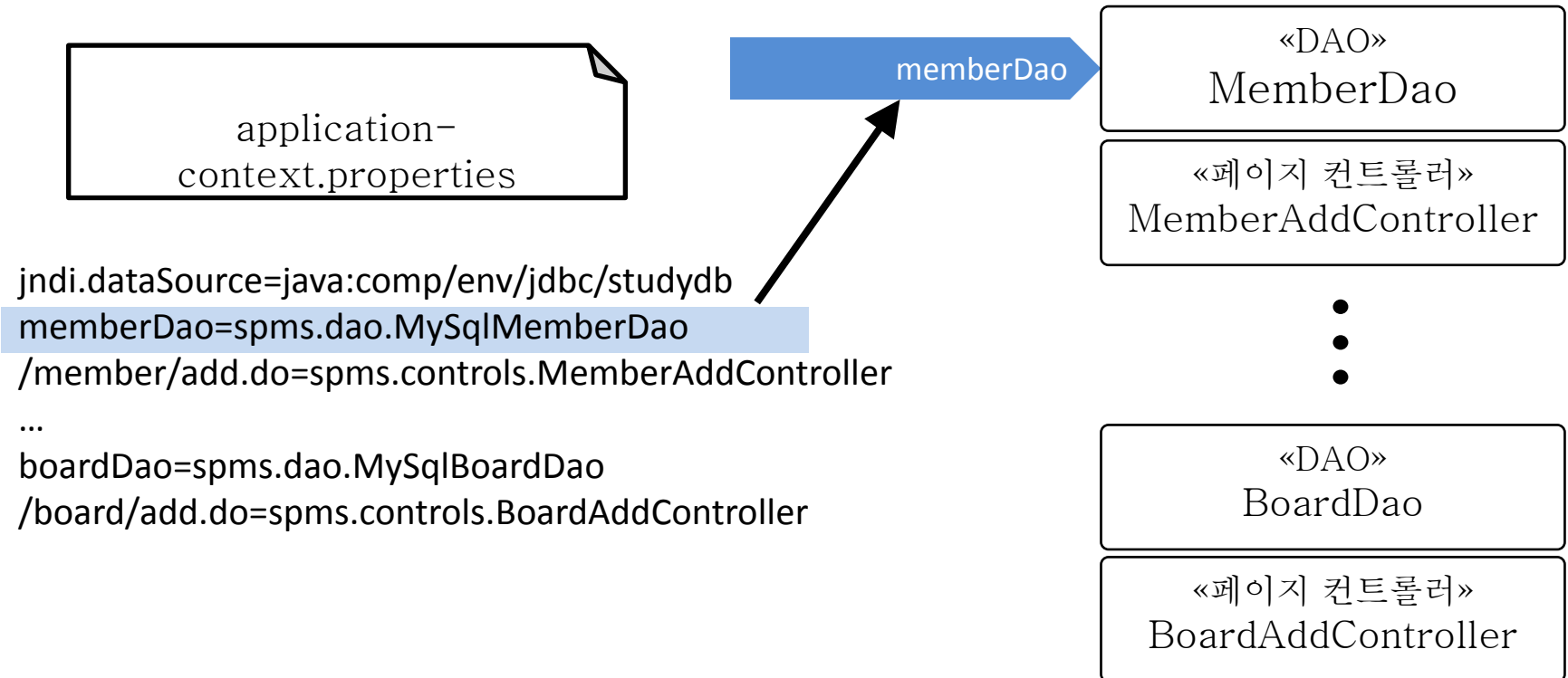
•
•
•

«DAO»
BoardDao

«페이지 컨트롤러»
BoardAddController

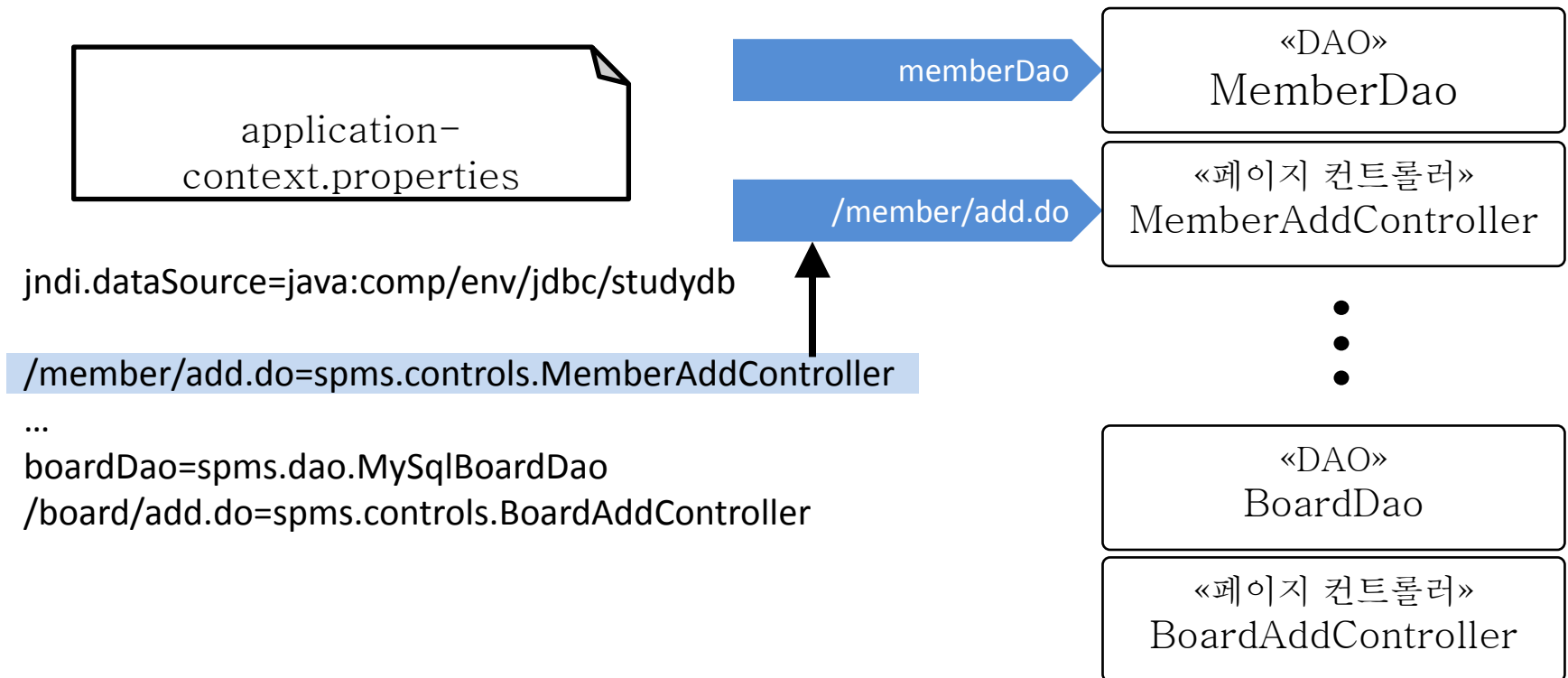
6.6 애노테이션을 이용한 객체 관리

DAO 객체 이름을 클래스 파일에 보관

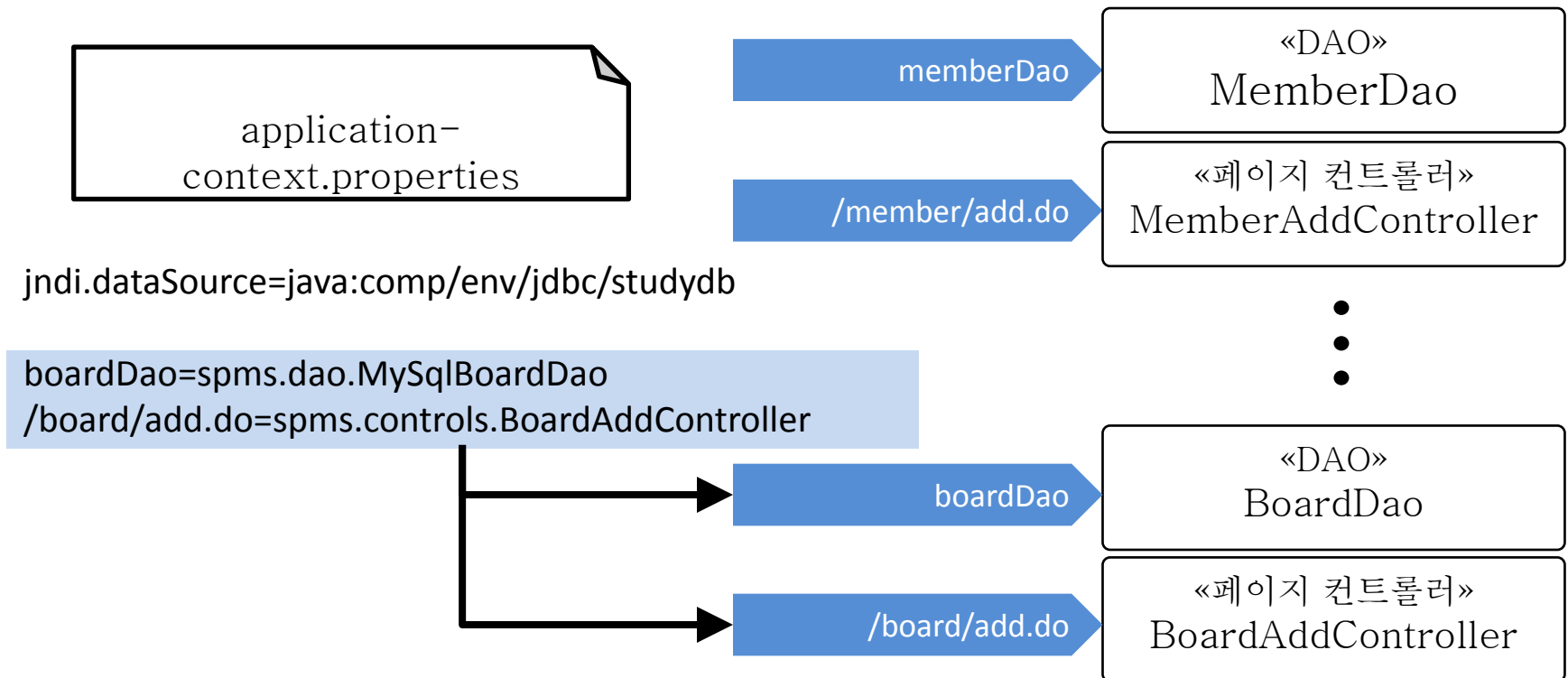


6.6 애노테이션을 이용한 객체 관리

페이지 컨트롤러의 서블릿 경로 정보를 클래스 파일에 보관



6.6 DAO 객체나 페이지 컨트롤러를 이용한 객체 관리 DAO 객체나 페이지 컨트롤러를 추가할 때도 단지 애노테이션만 클래스에 선언해 주면 된다.



6.6 애노테이션을 이용한 객체 관리

실습 시나리오

6.6 애노테이션을 이용한 객체 관리

1단계.

객체 이름을 보관할 애노테이션 정의

«애노테이션»
Component

```
@Retention(RetentionPolicy.RUNTIME)
public @interface Component {
    String value() default "";
}
```

6.6 애노테이션을 이용한 객체 관리

2단계.

DAO 및 페이지 컨트롤러에 애노테이션 적용

«애노테이션»
Component

«DAO»
MemberDao

```
@Component("memberDao")  
public class MySqlMemberDao  
    implements MemberDao {  
  
    ...  
}
```

6.6 애노테이션을 이용한 객체 관리

2단계.

DAO 및 페이지 컨트롤러에 애노테이션 적용

«애노테이션»
Component

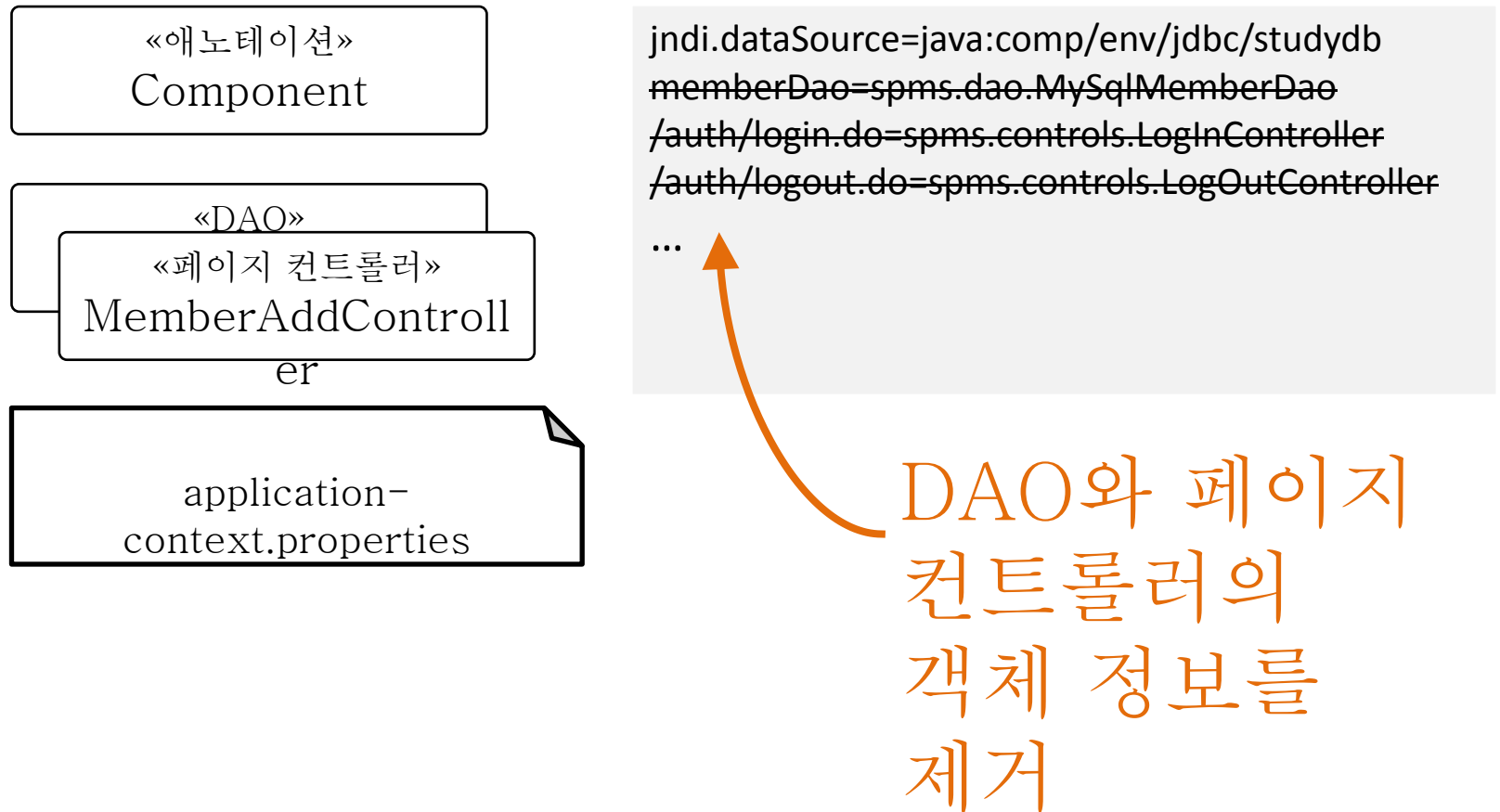
«DAO»
MemberDao

«페이지 컨트롤러»
MemberAddController

```
@Component("/member/add.do")  
public class MemberAddController  
    implements Controller, DataBinding {  
    ...  
}
```


6.6 애노테이션을 이용한 객체 관리

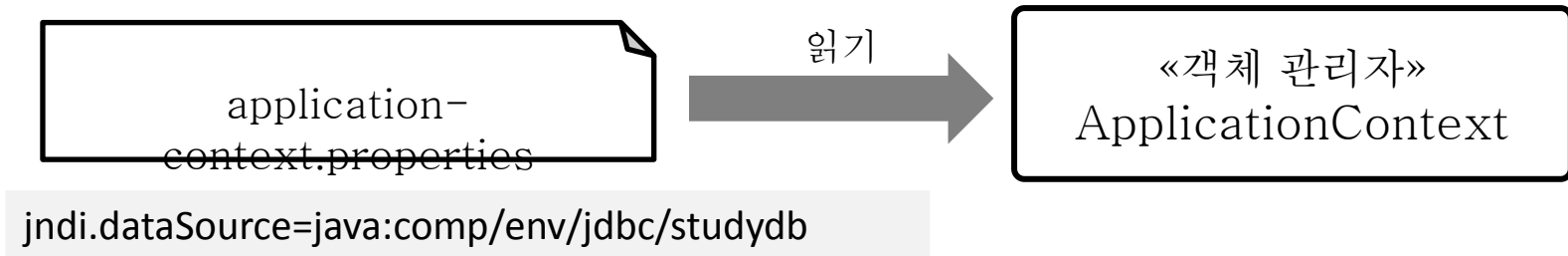
3단계. 프로퍼티 파일 변경



6.6 애노테이션을 이용한 객체 관리

4단계.

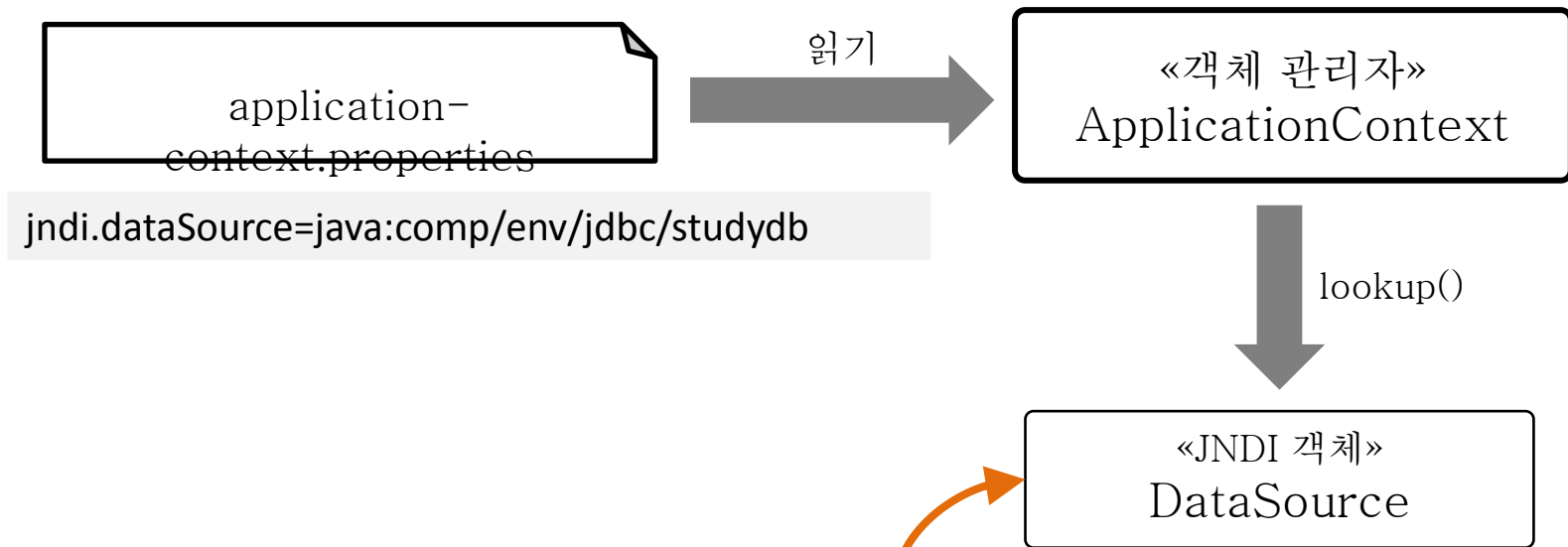
ApplicationContext 클래스 변경



6.6 애노테이션을 이용한 객체 관리

4단계.

ApplicationContext 클래스 변경



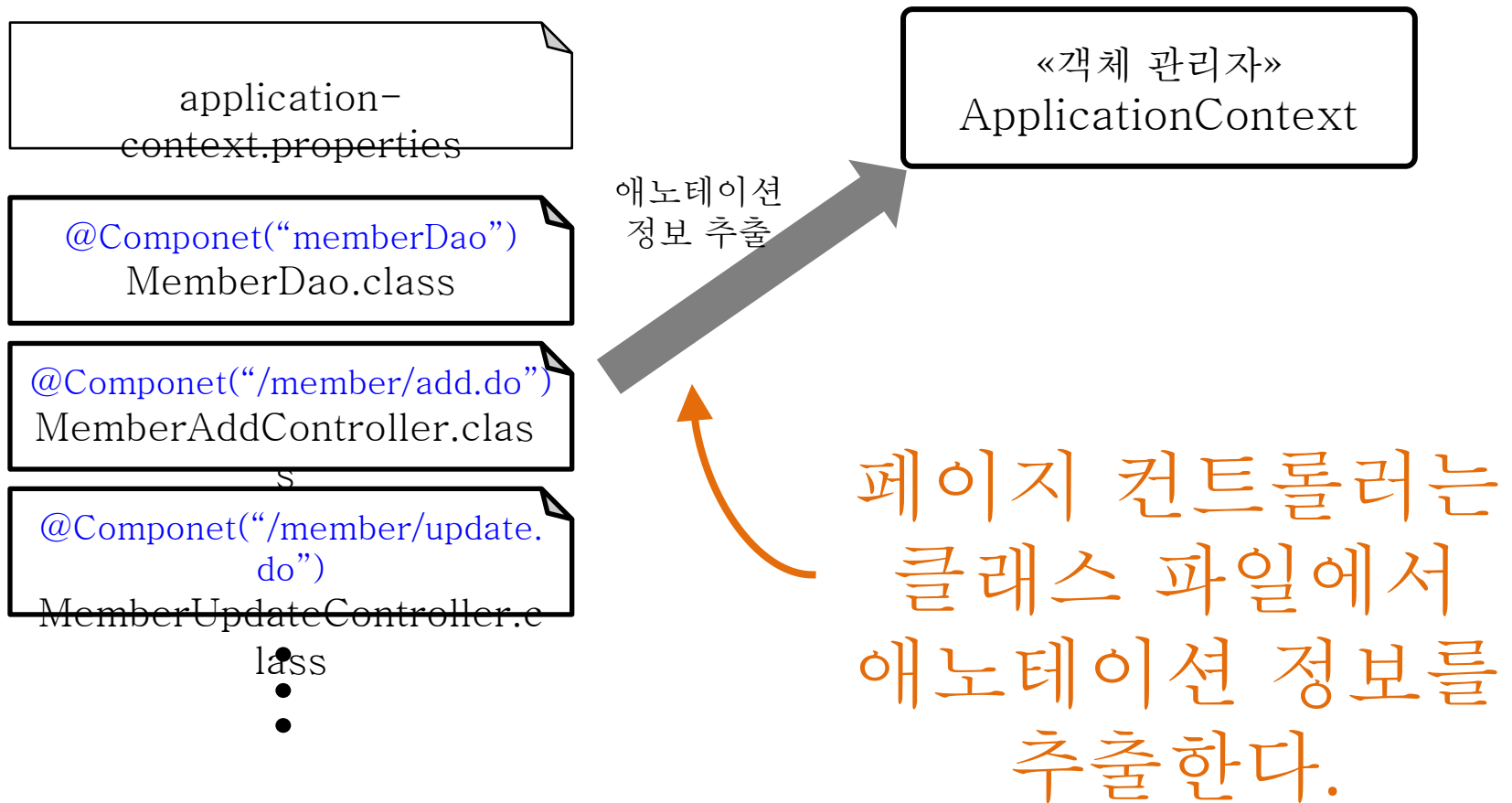
기존에
수행하던 작업
중

DataSource

6.6 애노테이션을 이용한 객체 관리

4단계.

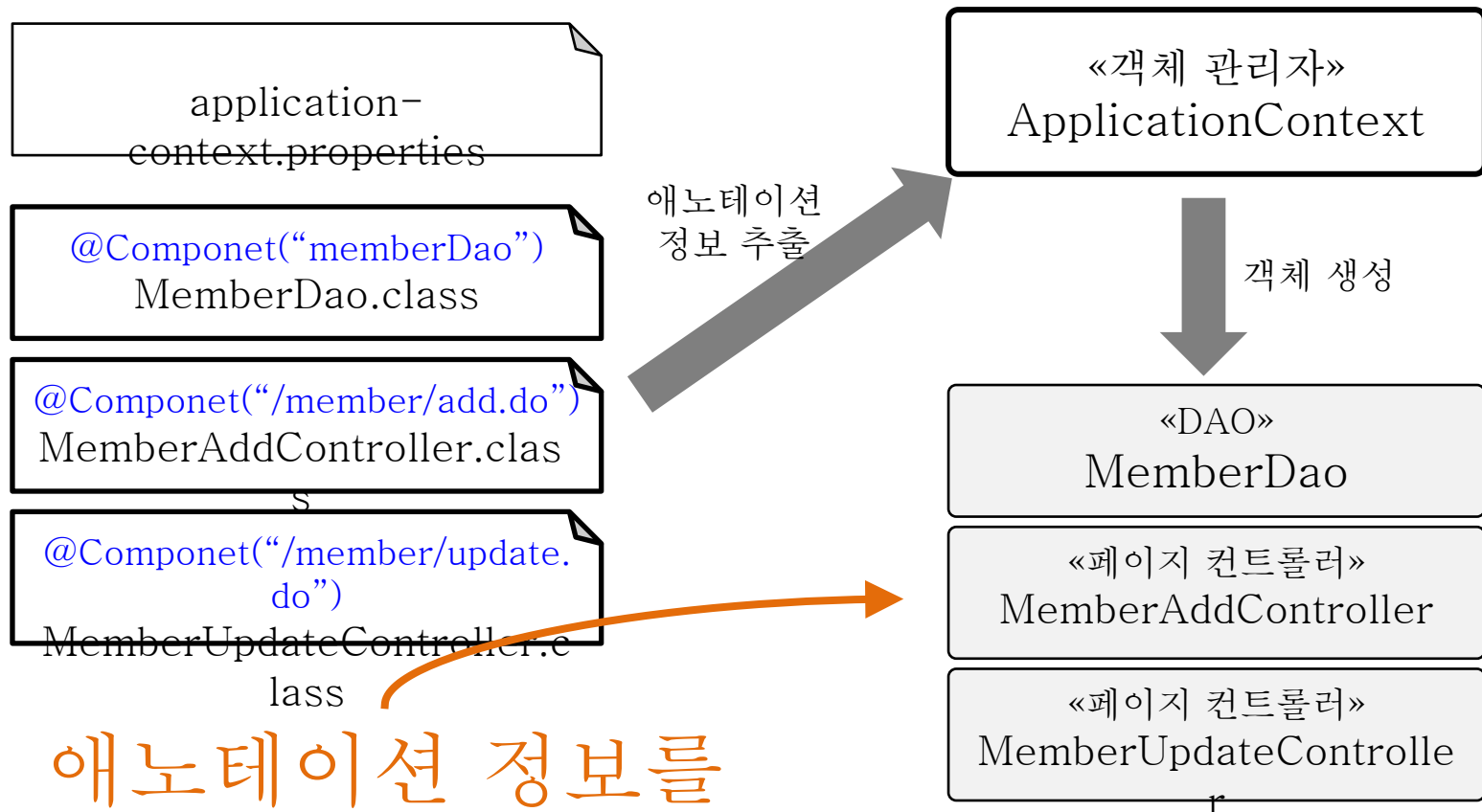
ApplicationContext 클래스 변경



6.6 애노테이션을 이용한 객체 관리

4단계.

ApplicationContext 클래스 변경



6.6 애노테이션을 이용한 객체 관리

참고!

6.6 애노테이션을 이용한 객체 관리

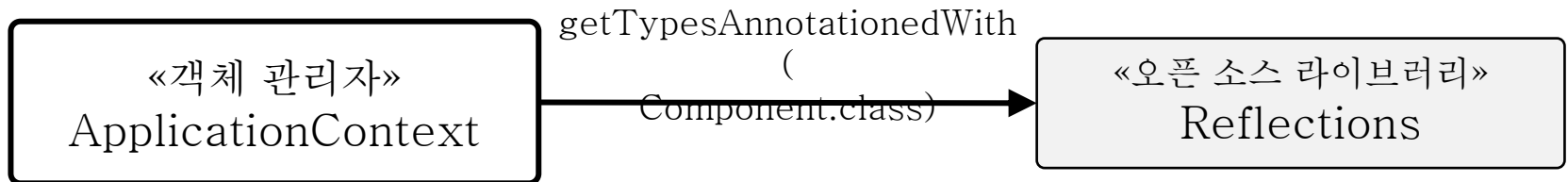
클래스 파일에서 애노테이션 정보를 추출할 때 Reflections 오픈 소스 라이브러리 사용

«객체 관리자»
ApplicationContext

«오픈 소스 라이브러리»
Reflections

6.6 애노테이션을 이용한 객체 관리

클래스 파일에서 애노테이션 정보를 추출할 때 Reflections 오픈 소스 라이브러리 사용



6.6 애노테이션을 이용한 객체 관리

클래스 파일에서 애노테이션 정보를 추출할 때 Reflections 오픈 소스 라이브러리 사용

