Angular Service

Service

공통 로직을 분리한 Class 파일이다.

일반적인 Class 파일에 @Injectable() 데코레이터만 기술하면 된다.

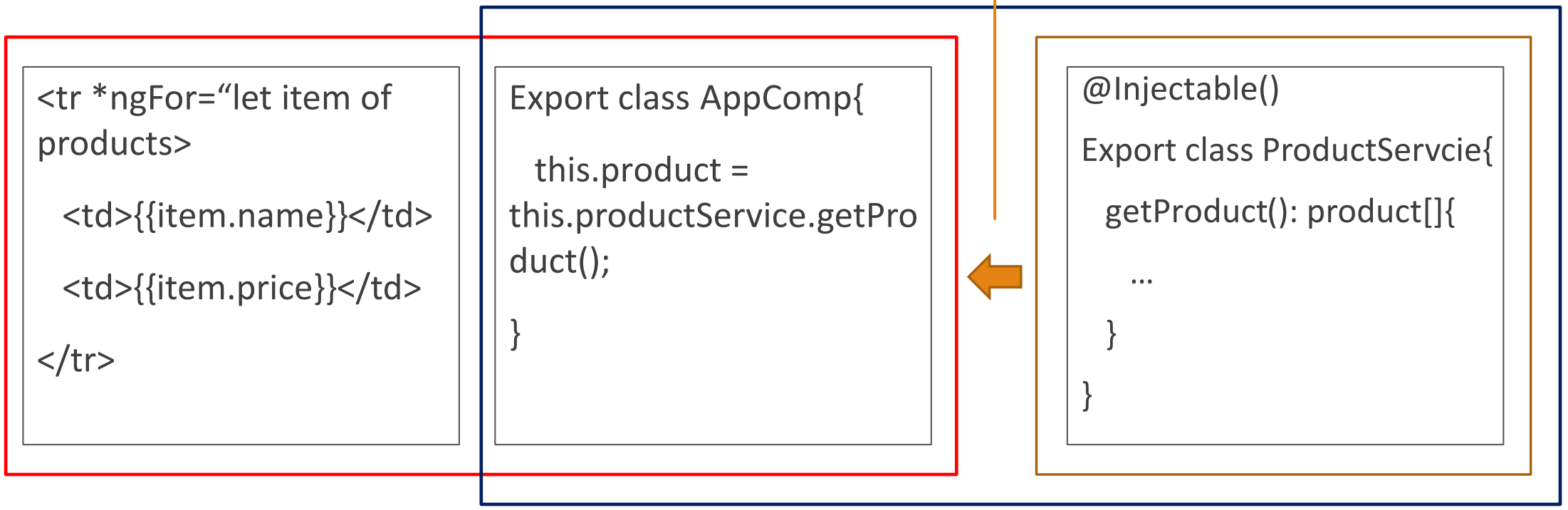
사실 @Injectable() 데코레이터는 특정 조건에서 생략 할 수 있다. 특정 조건은 서비스 클래스에서 “다른 서비스를 이용하지 않는 조건”이다.

단 서비스는 일반적인 서비스 클래스는 Ajax 통신을 위한 Http 등, 다른 서비스를 의존하는 경우가 대부분이다

현재 작성시에는 다른 서비스를 이용하지 않더라도 추후, 변경 등에 의해 다른 서비스를 사용하게 되면 생각하지 않았던 에러가 발생할 수 있으므로 @Injectable을 기술하는 것이 좋다,

Service

Component는 Template에 표시의 준비에 치중하고 비즈니스 로직은 Service에 일임하는 것이 원칙



Service

의존 관계를 선언하는 곳은 @NgModel과 Component의 providers 파라메터

Providers에 Service를 선언한 경우 Angular가 내부적으로 Provider Object를 생성한다.

```
@NgModule{
  providers: [ { provide: ProductService, useClass: ProductService, multi: flase } ]
}
```

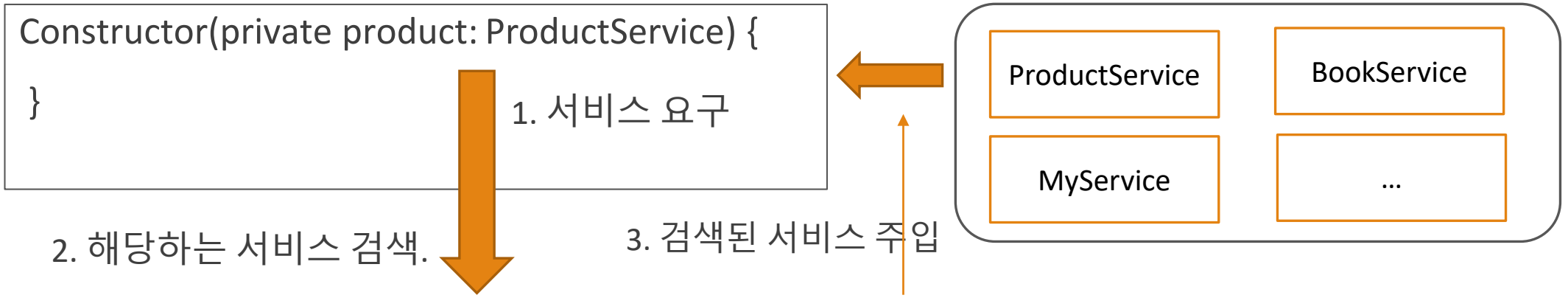
프로퍼티	설명
Provide	Service을 주입할 때 사용하는 DI Token
useXXX	Service(인스턴스)의 생성 방법
multi	같은 DI Token에 대해 복수의 Provider를 추가 할 것인가를 결정

Service

DI Token: Service를 구별하기 위한 Key가 되는 값이다. Angular는 Service가 요청되는 경우, 미리 준비한 Token List를 검색하여, 대응하는 Provider 정보에 기반하여 서비스를 작성한다.

Token은 일반적으로 Service의 타입과 동일하다.

Service

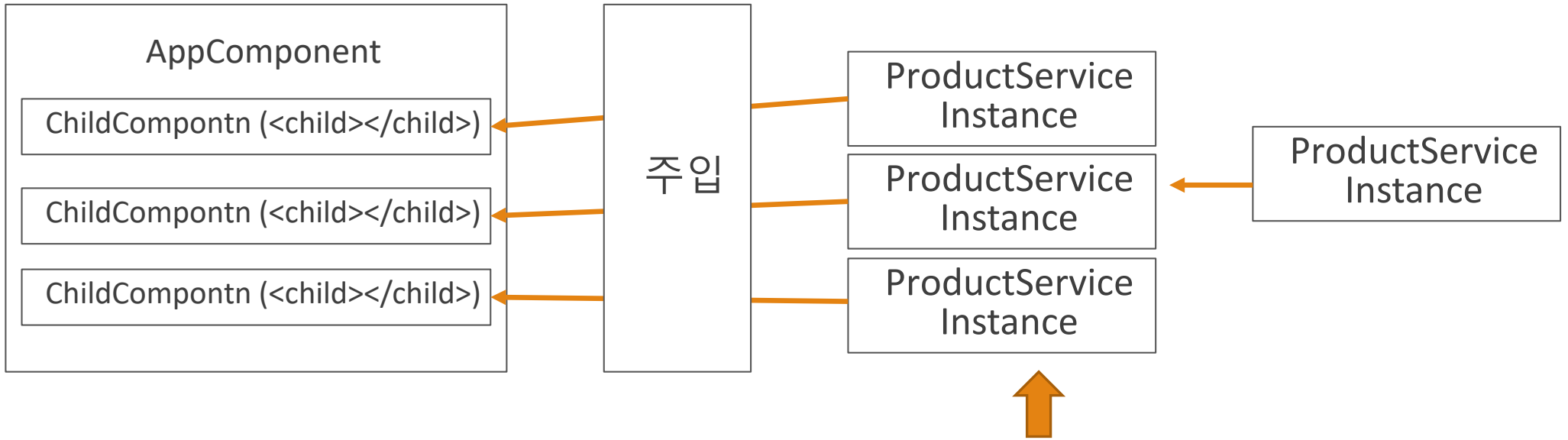


Token	생성방법	복수 지원
BookService	useValue(new Book())	true
ProductServcie	useClass(ProductService)	False
...	...	...
MyService	useFacory(...)	false

지정된 형(provide)을 Token으로 하여 요구할 때 마다 인스턴스를 생성하여 전달(useClass).

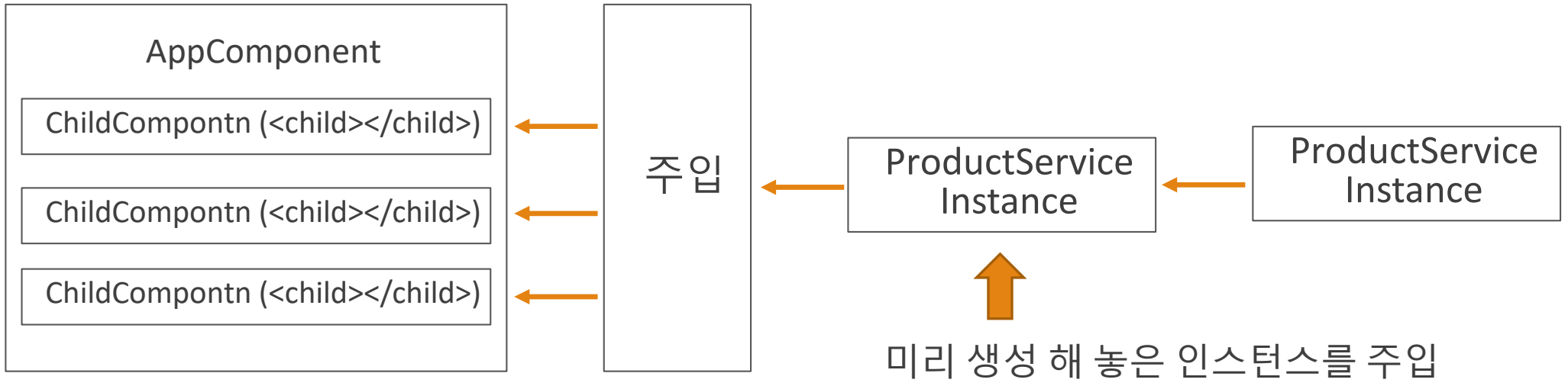
useClass

{ provide: ProductService, useClass: ProductService }



useValue

{ provide: ProductService, useValue: ProductService }



useExisting

사용 목적

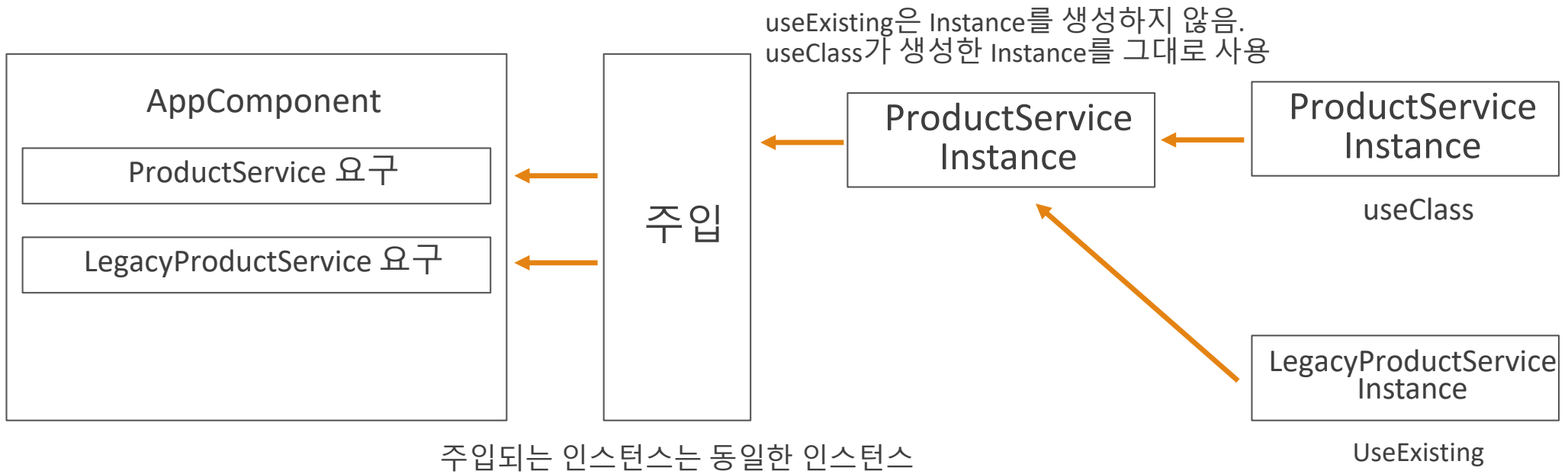
- AppComponent가 기존의 LegacyProductService에 의존하고 있음
- 현재는 LegacyProductService 서비스와 호환성이 있는 ProdcutService를 제공하고 있지만 여러가지 이유로 인하여 Component를 변경할 수 없음.
  (= LegacyProductService에서 ProductService로 변경 할 수 없음)

```
providers: [
    { provide: ProductServcie: useClass: ProductClass},
    { provide: LegacyProductService, useExisting: ProductService}
]
```

useExisting

```
{ provide: ProductService, useClass: ProductService }

{ provide: LegacyProductService, useExisting: ProductService }
```



provide

클래스가 아닌 텍스트, Object, Array 등은 Type 생성이 안되므로 독립적으로 Service로 주입 받아 사용할 수 없다. (AngularJS의 constant or value)

이 경우 InjectionToken(변수명 or 값)을 이용한다.