**스프링 빈과 의존관계**

* 컴포넌트 스캔과 자동 의존관계 설정
* 자바 코드로 직접 스프링 빈 등록하기

: **스프링 컨테이너란?**  
스프링 컨테이너는 스프링 빈의 생명 주기글 관리하고, 생성된 스프링 빈들에게 추가적인 기능을 제공하는 역할을 한다. 즉 개발자는 New 연산자, 인터페이스 호출, 등 방식으로 객체를 생성하고 소멸하지만, 스프링 컨테이너를 사용하면 해당 역할을 대신해준다.

: **스프링 빈이란?**  
스프링 빈은 스프링 컨테이너에 의해 관리되는 자바 객체(POJO)를 의미한다.

**컴포넌트 스캔과 자동 의존관계 설정**  
회원 컨트롤러가 회원서비스와 회원 리포지토리를 사용할 수 있게 의존관계를 준비하는 과정

회원 컨트롤러에 의존관계 추가

package hello.hellospring.controller;

import hello.hellospring.service.MemberService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

@Controller

public class MemberController {

private final MemberService memberService;

@Autowired

public MemberController(MemberService memberService) {

this.memberService = memberService;

}

}

* 생성자에 @Autowired가 있으면 스프링이 연관된 객체를 스프링 컨테이너에서 찾아 넣어준다.  
  이렇게 **의존관계를 외부에서 넣어주는것을 DI, 의존성 주입이라고 한다.**  
  ex) 회원 컨트롤러에서 회원서비스를 사용하는데, 회원서비스에서 추가해주는 것이 아닌 외부인 회원컨트롤러에서 @Autowired를 이용해서 의존관계를 주입해준다.
* 이전 테스트에서 개발자가 직접 주입했고, 여기서는 @Autowired에 의해 스프링이 주입해준다

**error 발생**

Consider defining a bean of type 'hello.hellospring.service.MemberService' in

your configuration.

Solution: memberService가 스프링 빈으로 초기등록이 되어 있지 않기 때문에 등록해주면 된다.

* 초기 등록이란 @Controller @Service 와 같이 해당 로직의 어노테이션을 말한다.

**스프링 빈을 등록하는 2가지 방법**

* 컴포넌트 스캔과 자동 의존관계 설정
* 자바 코드로 직접 스프링 빈 등록하기

**컴포넌트 스캔 원리**

* @Component 에노테이션이 있으면 스프링 빈으로 자동 등록된다.
* @Controller 가 있으면 스프링 빈으로 자동 등록되는데, 그 이유도 컴포넌트 스캔 때문이다.
* @Component를 포함하는 다음 에노테이션도 스프링 빈으로 자동 등록된다.
  + @Controller
  + @Service
  + @Repository

**회원 서비스 스프링 빈 등록**

@Service

public class MemberService {

private final MemberRepository memberRepository;

@Autowired

public MemberService(MemberRepository memberRepository) {

this.memberRepository = memberRepository;

}

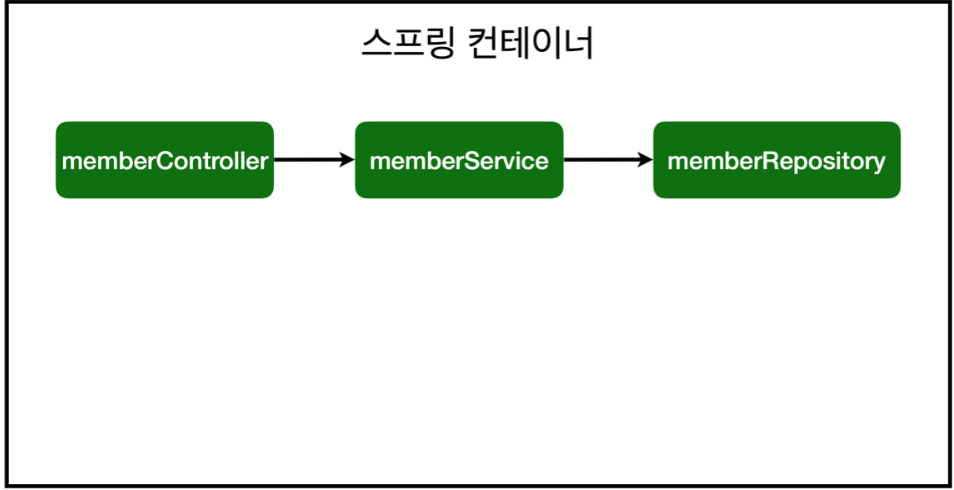
}

**Tip:**  
생성자에 @Autowired를 사용하면 객체 생성 시점에 스프링 컨테이너에서 해당 스프링 빈을 찾아서 주입한다. 생성자가 1개만 있으면 생략할 수 있다.  
즉, @Controller @Repository @Service 를 연결해준다 -> **DI**  
(컴포넌트 스캔 방식)  
**회원 리포지토리 스프링 빈 등록**

@Repository

public class MemoryMemberRepository implements MemberRepository {

}



* memberService와 memberRepository가 스프링 컨테이너에 스프링 빈으로 등록되었다.

**tip:**  
스프링은 스프링 컨테이너에 스프링 빈을 등록할 때, 기본적으로 싱글톤으로 등록한다. (유일하게 하나만 등록해서 공유) 따라서 같은 스프링 빈이면 모두 같은 인스턴스다. 설정으로 싱글톤이 아니게 설정할 수 있지만, 특별한 경우를 제외하면 대부분 싱글톤을 사용한다.

자바코드로 직접 스프링 빈 등록하기

* 회원 서비스와 회원 리포지토리의 @Service, @Repository, @Autowired 에노테이션을 제거하고 진행한다.

package hello.hellospring;

import hello.hellospring.repository.MemberRepository;

import hello.hellospring.repository.MemoryMemberRepository;

import hello.hellospring.service.MemberService;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class SpringConfig {

@Bean

public MemberService memberService() {

return new MemberService(memberRepository());

}

@Bean

public MemberRepository memberRepository() {

return new MemoryMemberRepository();

}

}

여기서는 향후 메모지 리포지토리를 다른 리포지토리로 변경할 예정이므로, 컴포넌트 스캔 방식 대신에 자바 코드로 스프링 빈을 설정한다.

**Tip:**  
1) XML로 설정하는 방식도 있지만 최근에는 잘 사용하지 않는다.  
2) DI에는 필드 주입, setter 주입, 생성자 주입, 이렇게 3가지 방법이 있다. 의존관계가 실행중에 동적으로 변하는 경우는 거의 없으므로 생성자 주입을 권장한다.  
3) 실무에서는 주로 정형화된 컨트롤러, 서비스,리포지토리 같은 코드는 컴포넌트 스캔을 사용한다.  
4) **주의,** @Autowired를 통한 DI는 helloController, memberService 등과 같이 스프링이 관리하는 객체에서만 동작한다. 스프링 빈으로 등록하지 않고 내가 직접 생성한 객체에서는 동작하지 않는다.