

**초격차 패키지 Online.**

# 아는 사람만 아는 Boot 기능

## **PART1 | Spring Native [Experimental]**

더 빠른 Boot

## **PART2 | Testcontainers**

Docker의 활용

## **PART3 | Config Server**

외부 설정을 전담하는 독립적인 서버를 만들어보자

## **PART4 | Config Client**

외부 설정 서버에 접근하기

# 아는 사람만 아는 Boot 기능

## 2 Testcontainers

## 외부 장치와 연결된 기능의 통합 테스트

## 2. Testcontainers

### (예) MySQL에 접근하는 부분의 테스트

- H2 이용: 편하지만 MySQL 과 동일한 환경은 아님 (예상 못한 문제 발생)
- Mocking: 테스트 가능해 지지만, 이것도 MySQL 로 인해 발생하는 문제를 감지 못함
- 로컬에 프로젝트가 필요한 것과 동일한 세팅으로 MySQL 설치
  - 번거로움, 리소스를 잡아먹음
  - 새로 셋업하는 개발 컴퓨터에서는 실패하는 테스트 (MySQL 존재를 몰랐기 때문에)
- MySQL 을 docker에 올림
  - 테스트를 실행할 때마다 docker를 올리고, 내리는 작업을 따로 해줘야 함

## Testcontainers

## 2. Testcontainers

### 도커를 활용할 수 있는 또 한가지 방법

- DB 등의 외부 장치를 코드로 표현하고, 자동으로 docker 이미지로 만들고 등록/해제
- 외부 장치와 연관이 있는 부분의 통합 테스트를 용이하게
- 자바 라이브러리, JUnit 호환
- <https://www.testcontainers.org/>



# Testcontainers: 실습

## 2. Testcontainers

## 정리

## 2. Testcontainers

- redis 를 컨테이너로 만들고 소스코드로 표현이 가능해졌다 (단 docker는 설치해야 ^^)
- 테스트용 컨테이너가 자동으로 올라가고 내려가서 편리 - 그만큼 테스트 속도는 느려짐
- 컨테이너로 등록하여 동적으로 변하는 설정 - @DynamicPropertySource
- Logger 를 주입하면 컨테이너 내부에서 일어나는 일을 관찰 가능
- 잘 안되는 경우도 있음 (ex: VaultContainer 이용한 Vault 연동)
- 문서와 참고자료가 부족한 편
  - 대부분의 공식 예제 코드가 JUnit4로 작성됨
  - 새롭게 소개된 메소드의 사용법 등 자료가 부족
  - deprecated 된 기능의 이유와 대안이 문서화 되어있지 않음

## Reference

## 2. Testcontainers

- <https://www.testcontainers.org/>
- <https://hub.docker.com/search>
- <https://docs.spring.io/spring-boot/docs/current/reference/html/howto.html#howto.testing.testcontainers>