스프링부트를 대하는 자세

권용근
줌인터넷
kingbbode@gmail.com
https://www.facebook.com/iwannaknowcoding
https://kingbbode.github.io

스프링부트를 대하는 자세

쓸 때 쓰더라도
알고 쓰자!

신입 개발자의 Pilot Project

```
dependencies {
    compile('org.springframework.boot:spring-boot-starter-web')
    compile('org.springframework.boot:spring-boot-starter-data-jpa')
    providedRuntime('org.springframework.boot:spring-boot-starter-tomcat')

    //JACKSON
    compile('com.fasterxml.jackson.core:jackson-annotations:2.8.0')
    compile('com.fasterxml.jackson.core:jackson-core:2.8.6')
    compile('com.fasterxml.jackson.core:jackson-databind:2.8.6')

    //FREEMARKER
    compile('org.springframework:spring-context-support:4.3.7.RELEASE')
    compile('org.freemarker:freemarker:2.3.23')

    runtime('mysql:mysql-connector-java')

    //TEST
    testCompile('org.springframework.boot:spring-boot-starter-test')
    testCompile('org.hamcrest:hamscrest-core:1.3')
    testCompile('org.hamcrest:hamscrest-library:1.3')
}
```

음..? 위에는 부트인데..

부트인듯, 부트아닌, 부트같은 Dependencies~

`Transaction`이 끝난 상태에서 `Team`을 조회하니까 `LazyInitializationException`이 발생할꺼야!

하는 순간 페이지가 정상적으로 노출되는 당황스러운 순간이,,!

신입개발자만이 아니였다. 나도..

```
# JPA (JpaBaseConfiguration, Hib
spring.data.jpa.repositories.ena
spring.jpa.database= # Target da
spring.jpa.database-platform= #
spring.jpa.generate-ddl=false #
                              l-auto= #
spring.jpa.hibernate.naming.impl
spring.jpa.hibernate.naming.phys
spring.jpa.hibernate.naming.stra
spring.jpa.hibernate.use-new-id-
spring.jpa.open-in-view=true # I
```

- 스프링 부트란 무엇인가?
- 왜 알아야하지?
- 어떻게 알 수 있을까?
- 직접 보는 방법
  - Stater Dependency
  - Auto Configuration
  - Application Properties

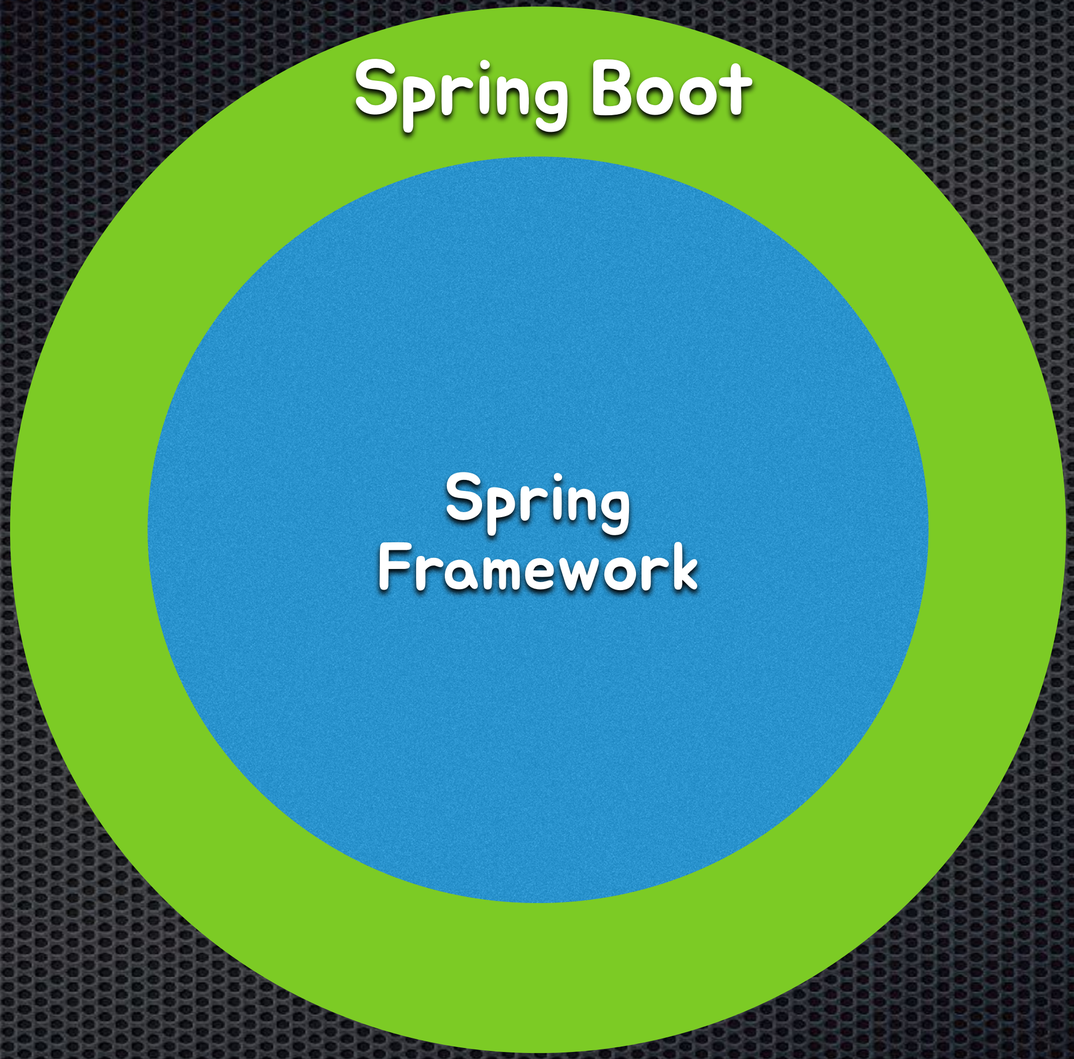
스프링 부트란 무엇인가!

'Just Run'

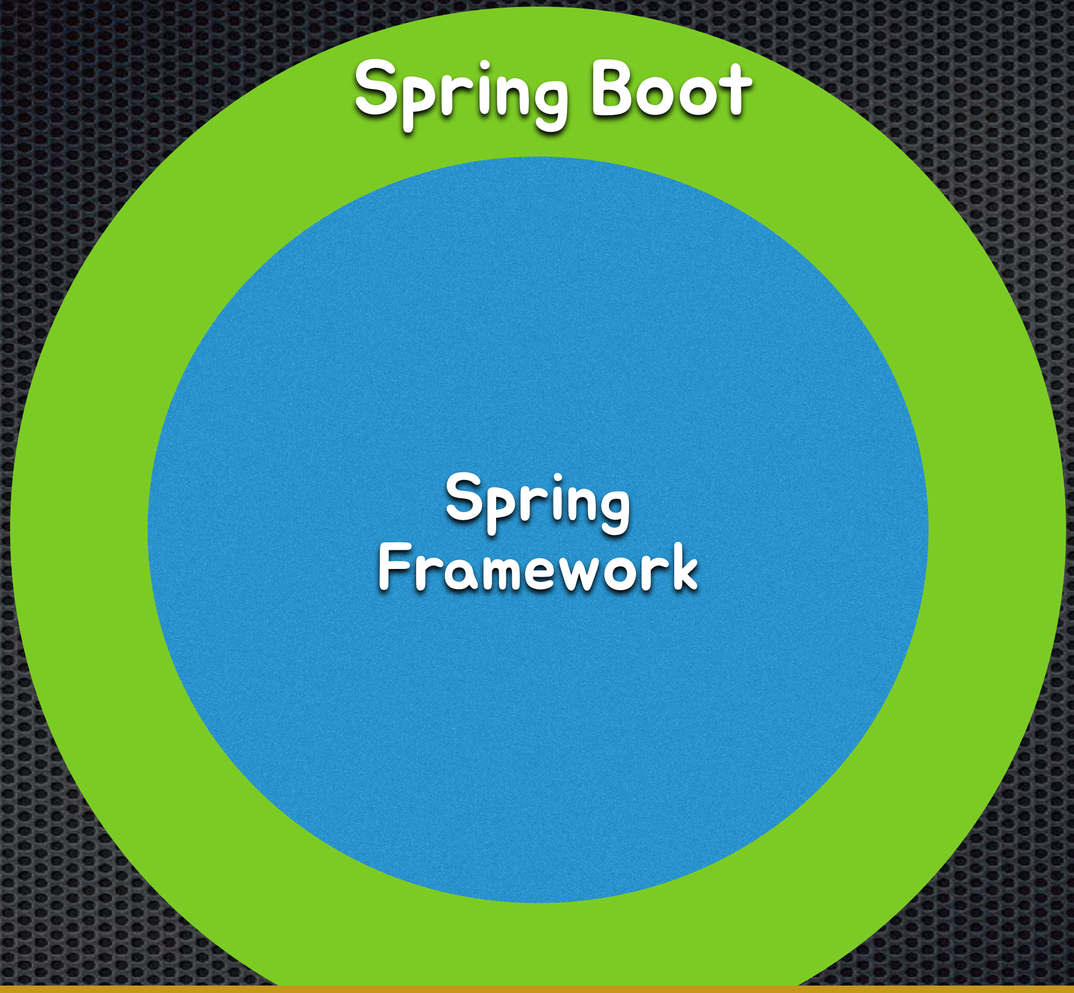
Most Spring Boot applications need very little Spring configuration.

`Starter Dependency`

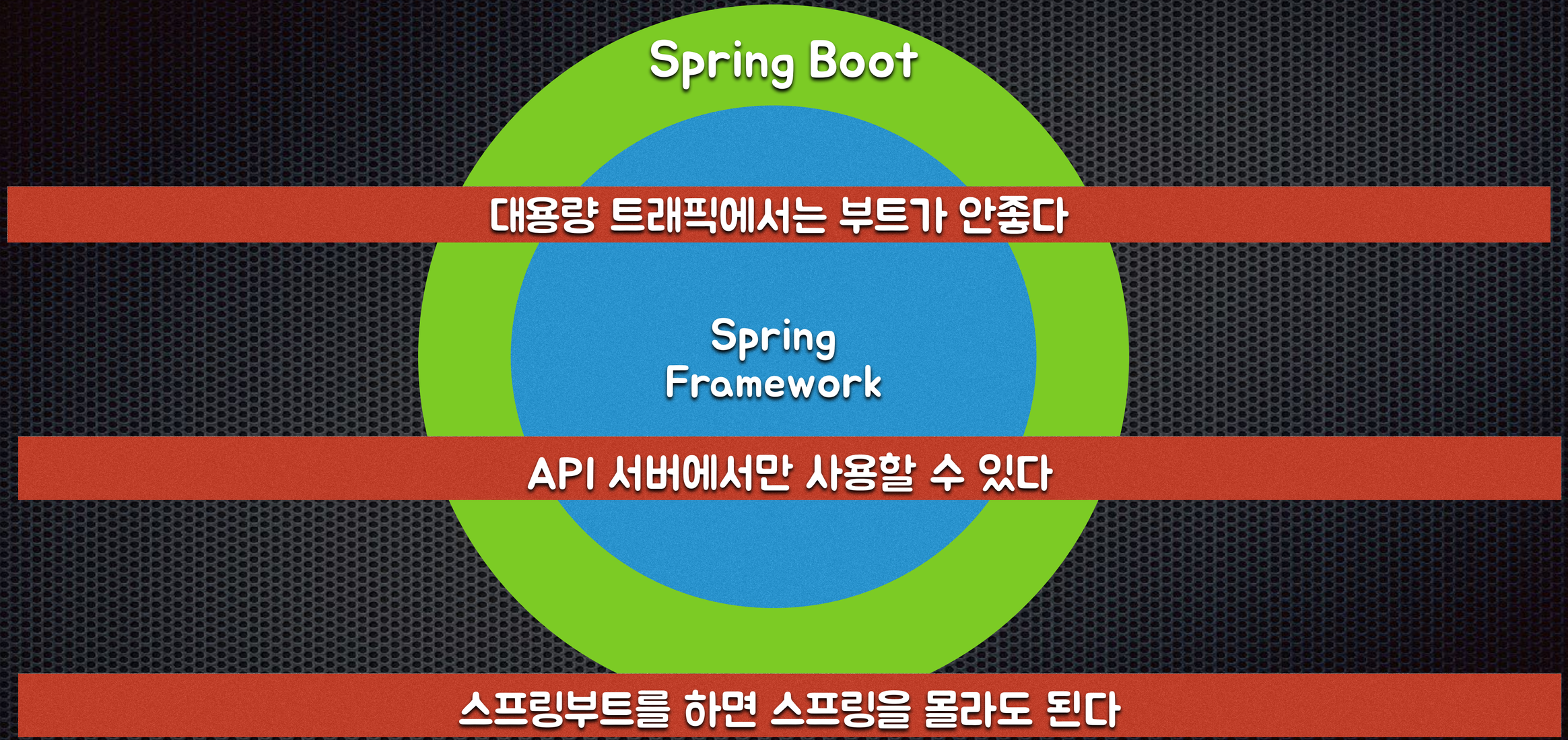
`Auto Configuration`

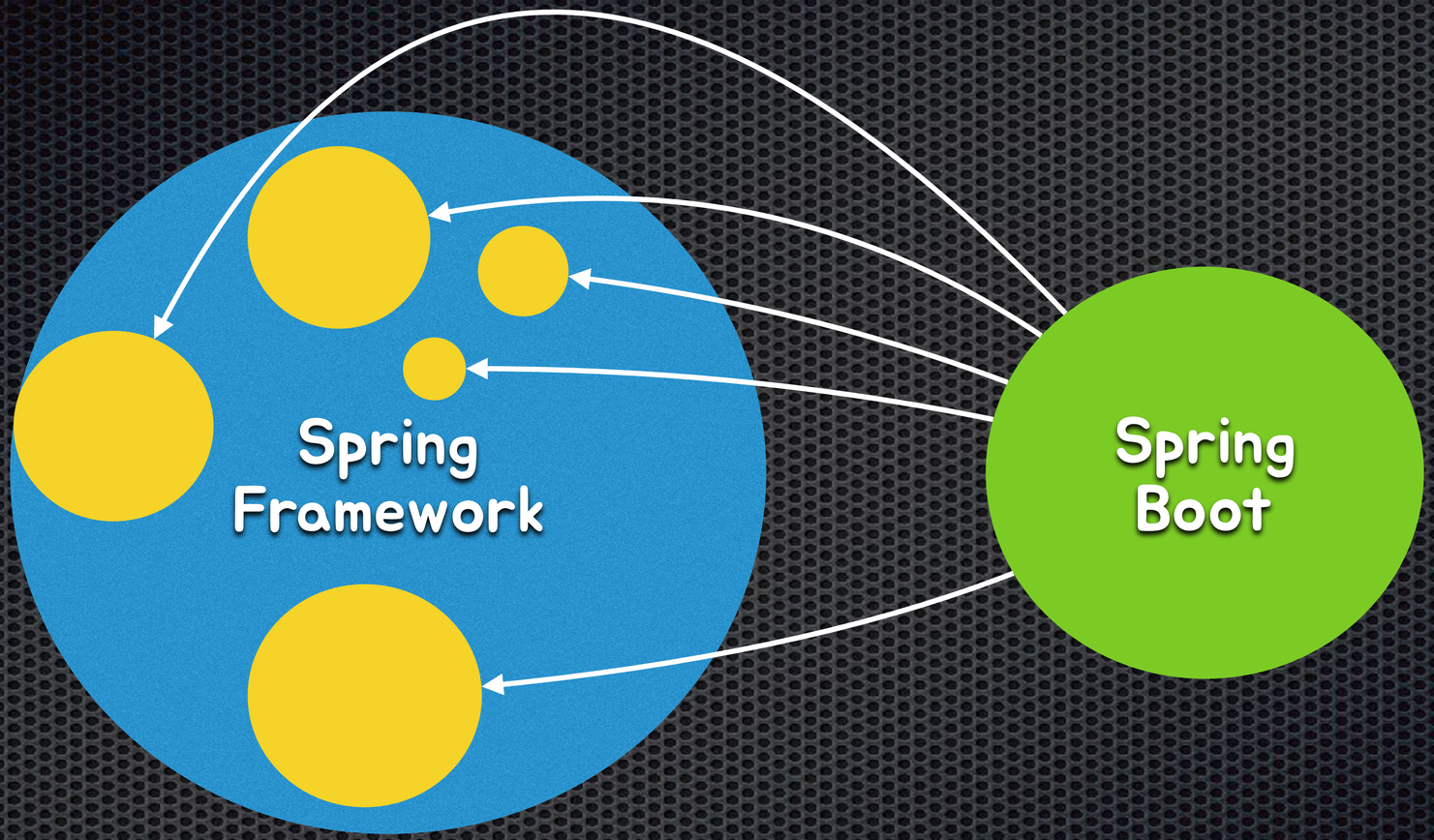

Spring Boot
Spring
Framework

Spring Boot에 대한 오해

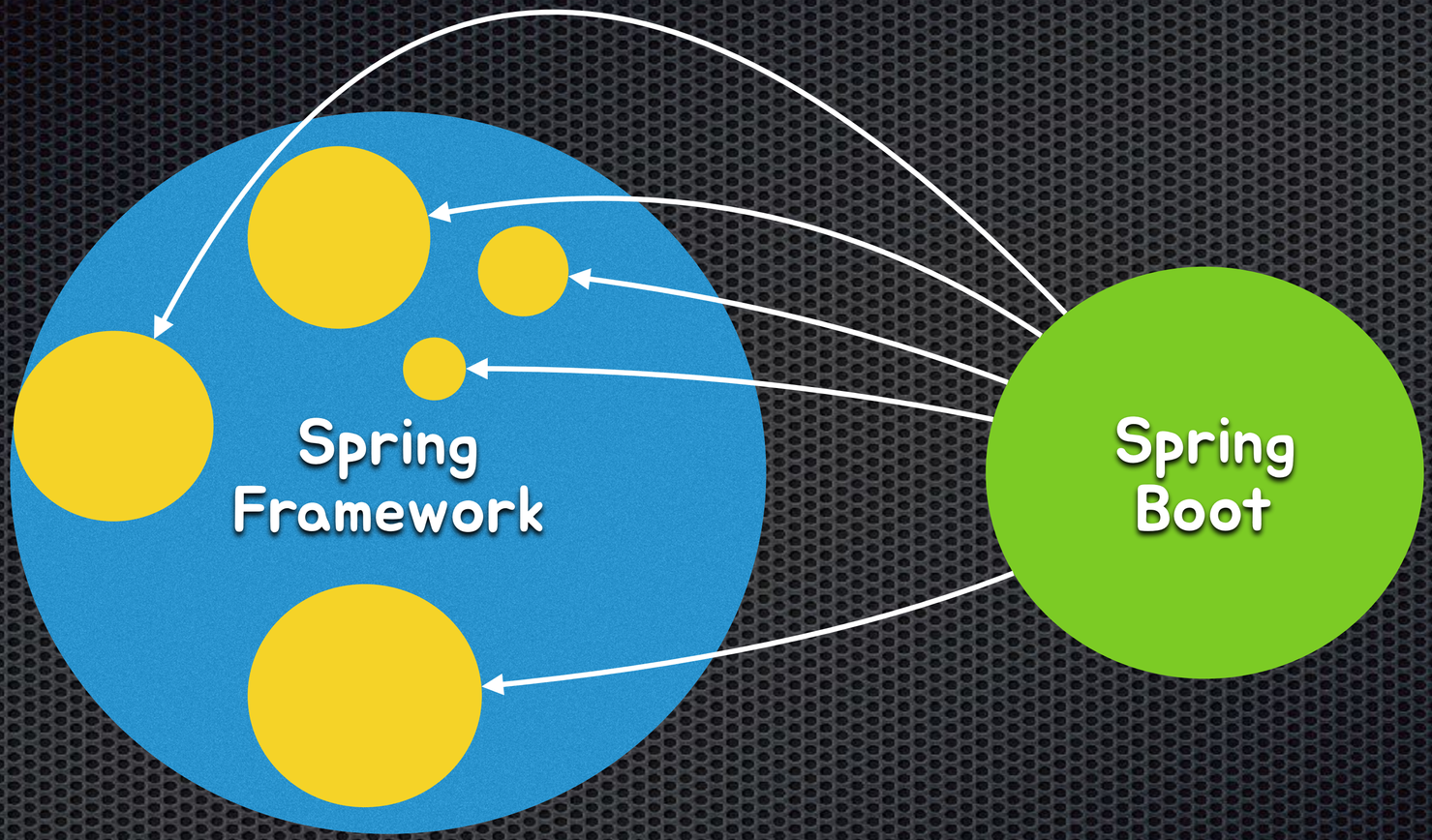


흔히들 오해하는 스프링 부트

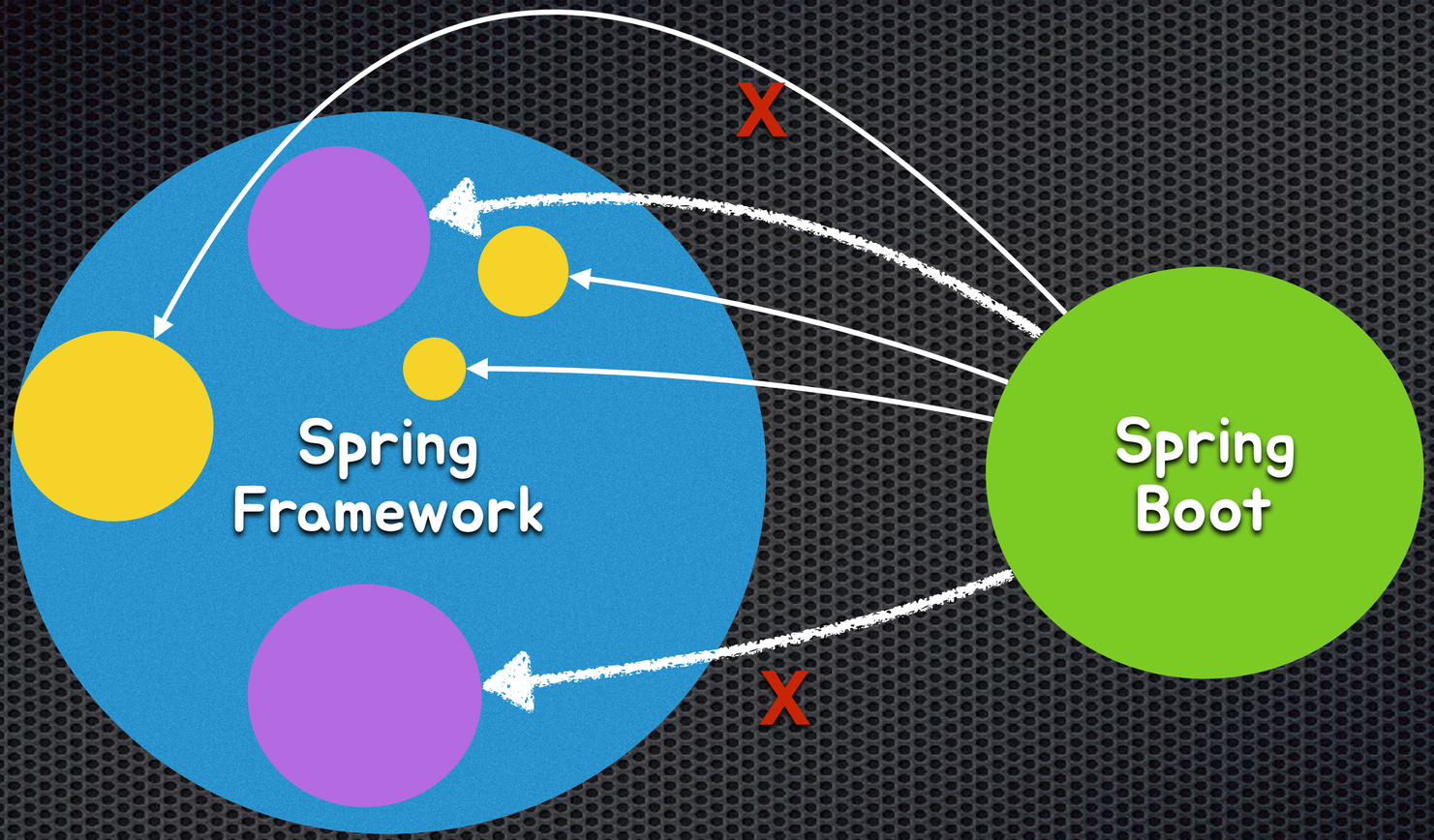

Spring Boot
대용량 트래픽에서는 부트가 안좋다
Spring
Framework
API 서버에서만 사용할 수 있다
스프링부트를 하면 스프링을 몰라도 된다



Spring Boot는 Framework가 아닌
Spring을 지원해주는 도구



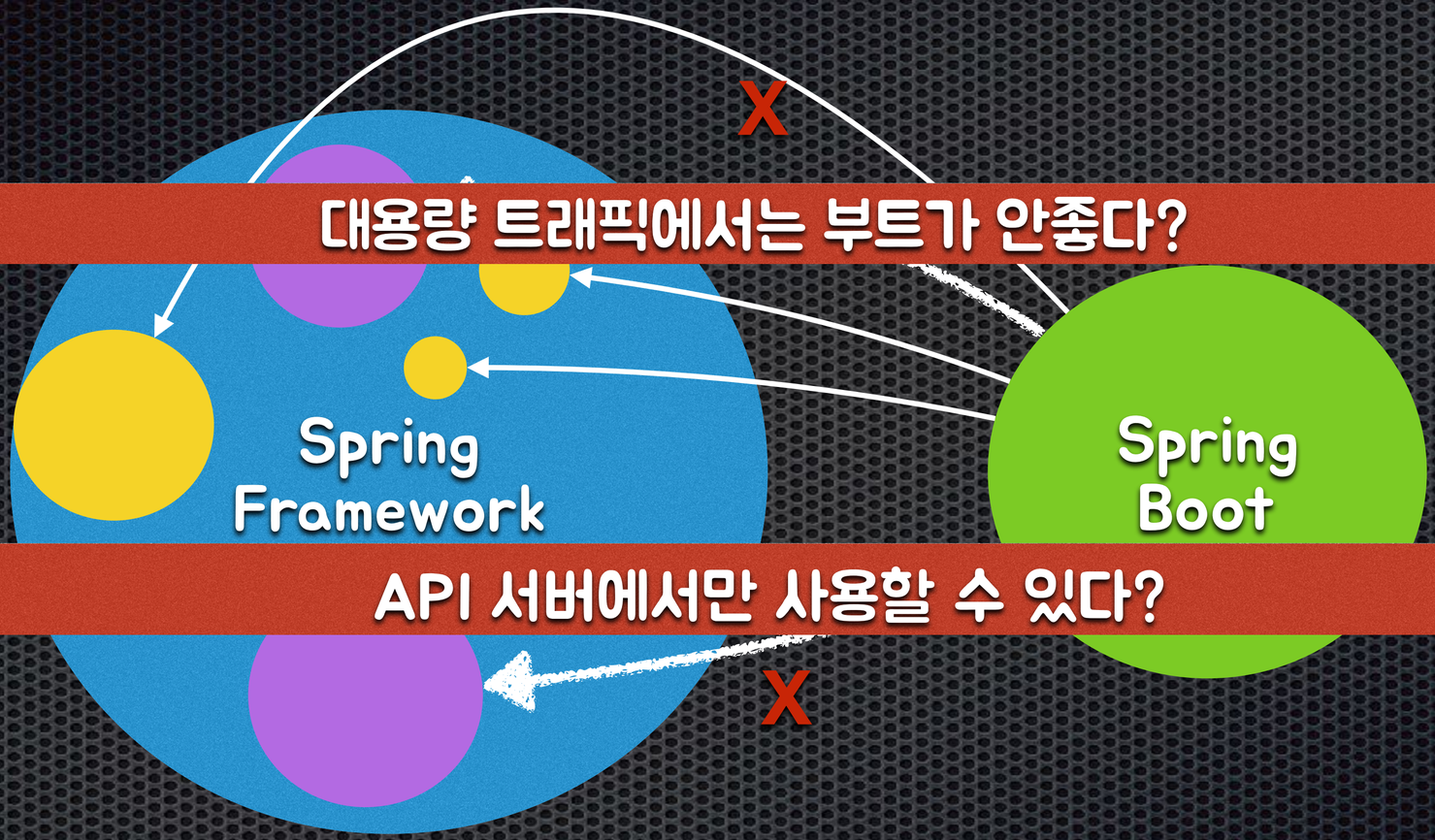
변경요소가 크지 않는 많은 설정(boilerplate)들을 기본적으로 지원



필요하다면 언제든 걷어내면 된다

X

대용량 트래픽에서는 부트가 안좋다?



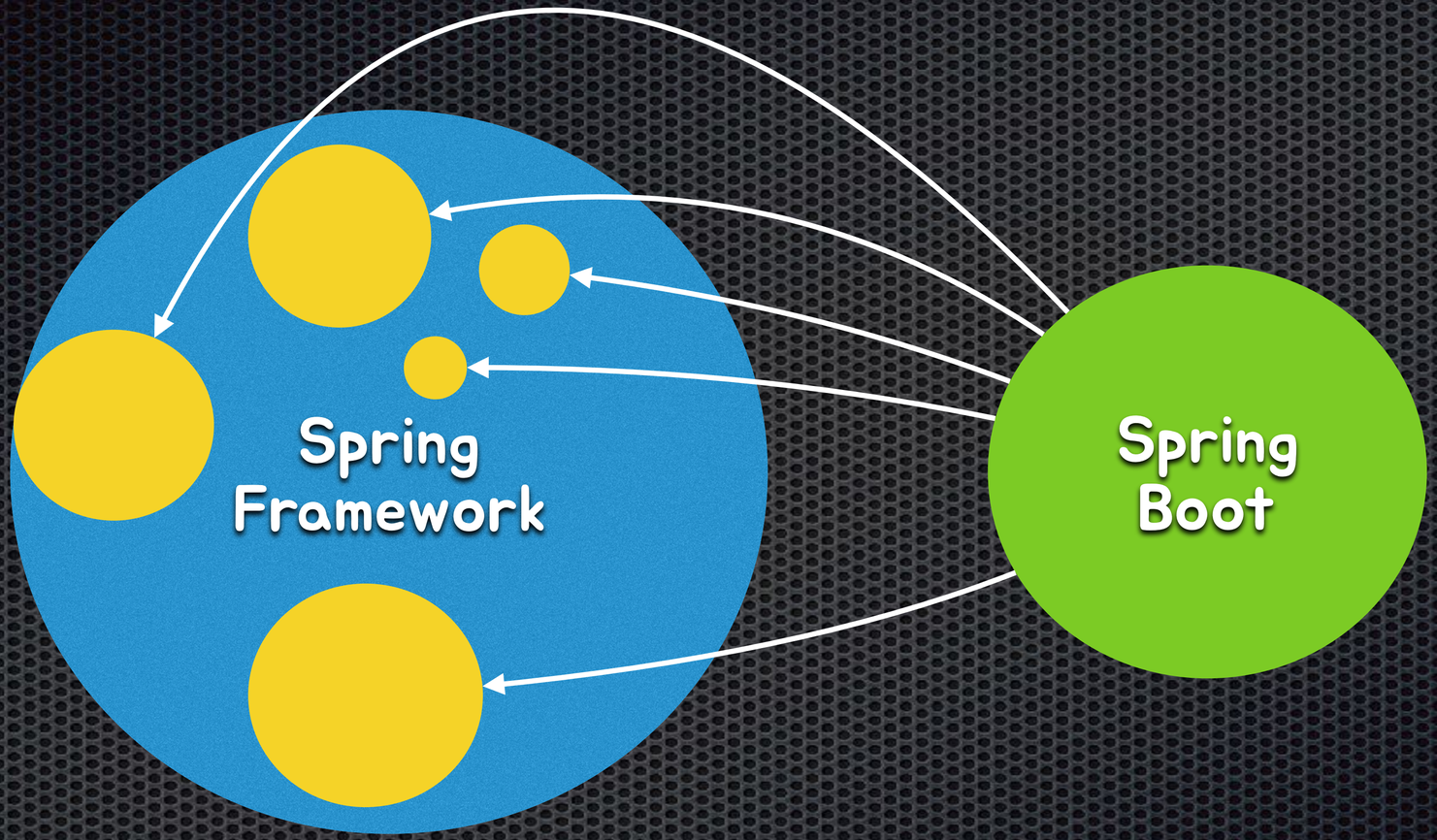
API 서버에서만 사용할 수 있다?

X

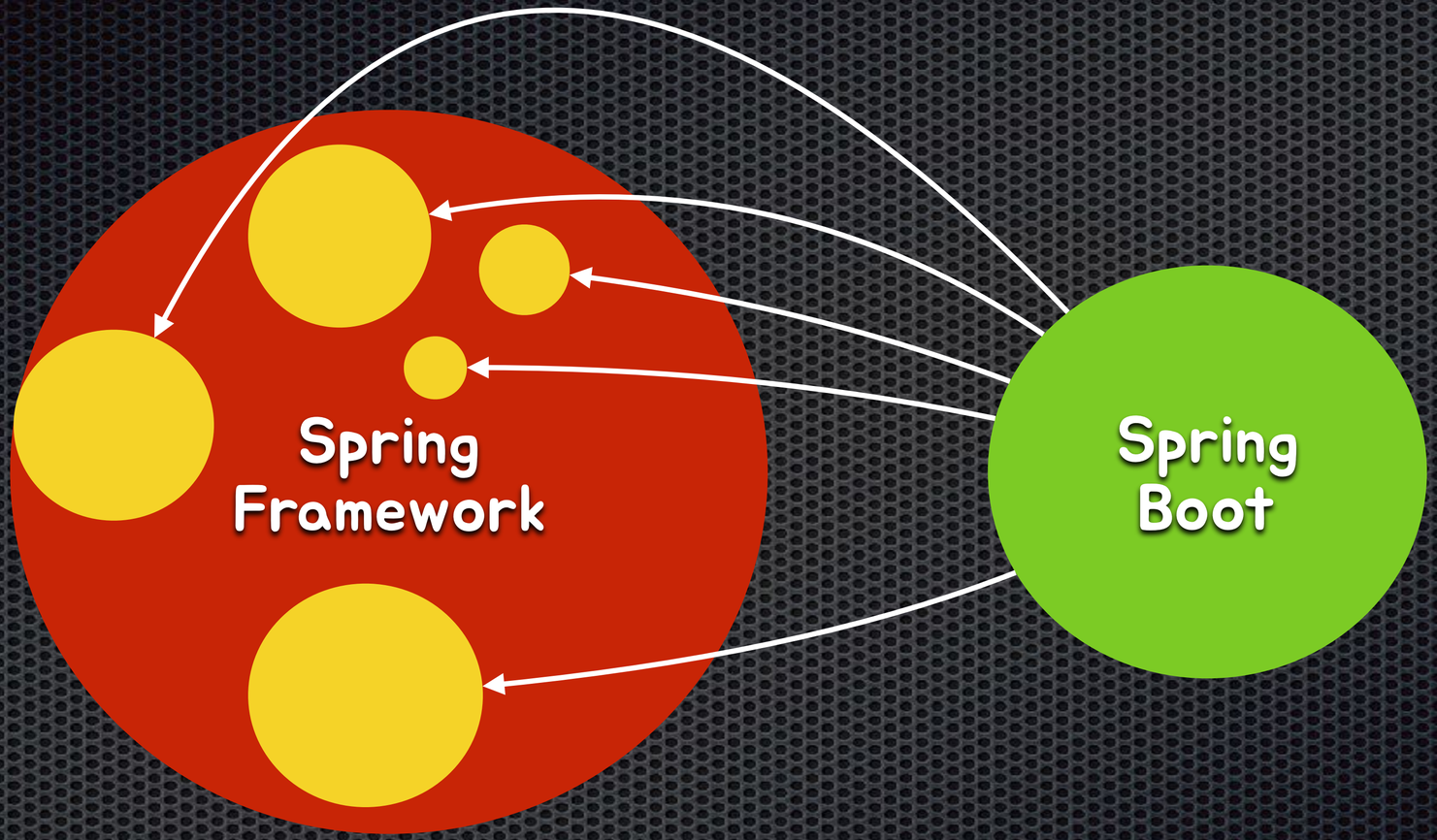
스프링부트를 하면 스프링을 몰라도 된다?


X
대용량 트래픽에서는 부드기 안좋다?
스프링 부트는 지원해주는 도구일 뿐!
API 서버에서만 사용할 수 있다?
X
스프링부트를 하면 스프링을 몰라도 된다?

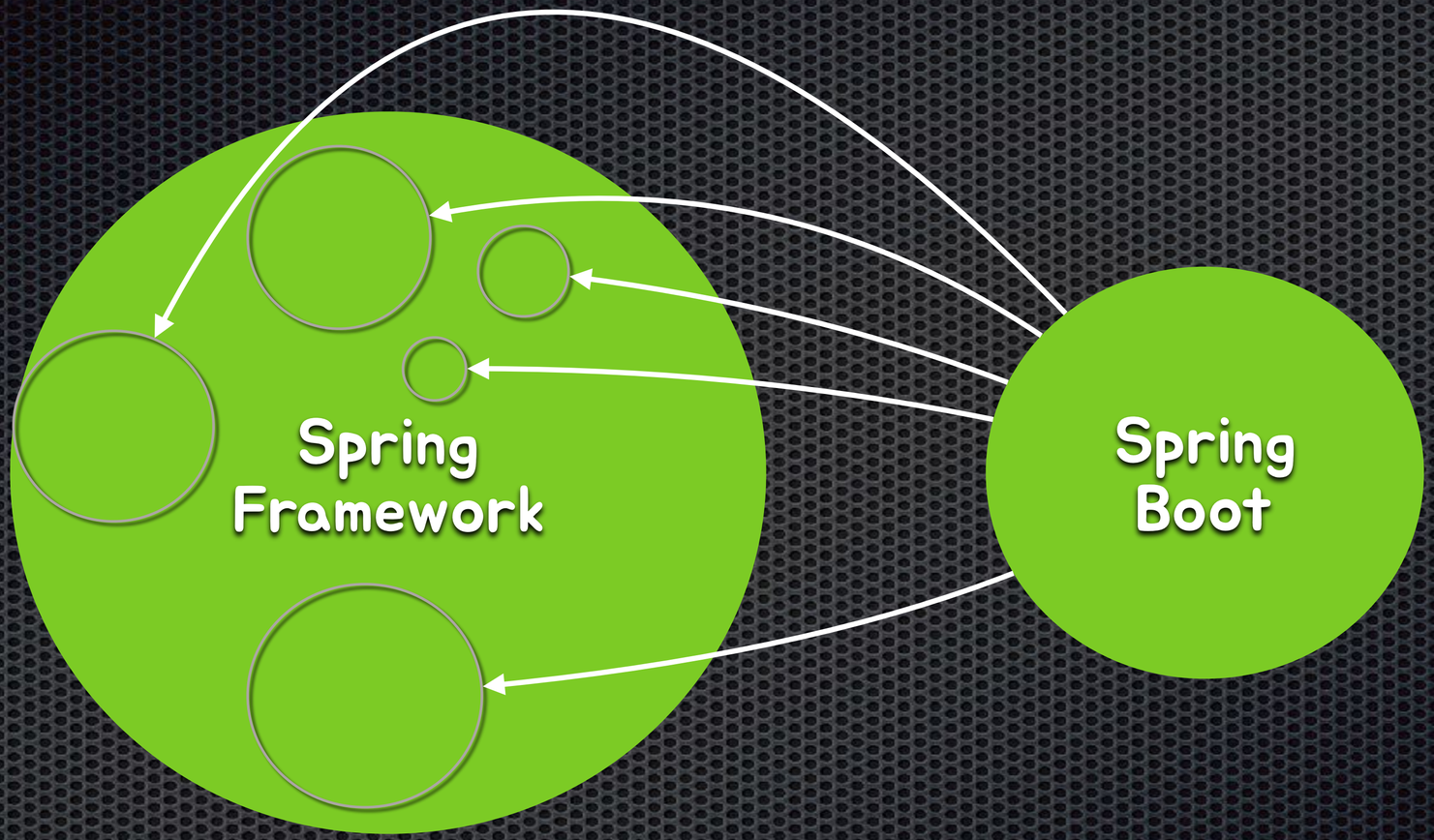
왜 부트를 이해해야 할까?



Spring Boot는 Spring Framework의 구성과 설정을 대신해주고 있다.

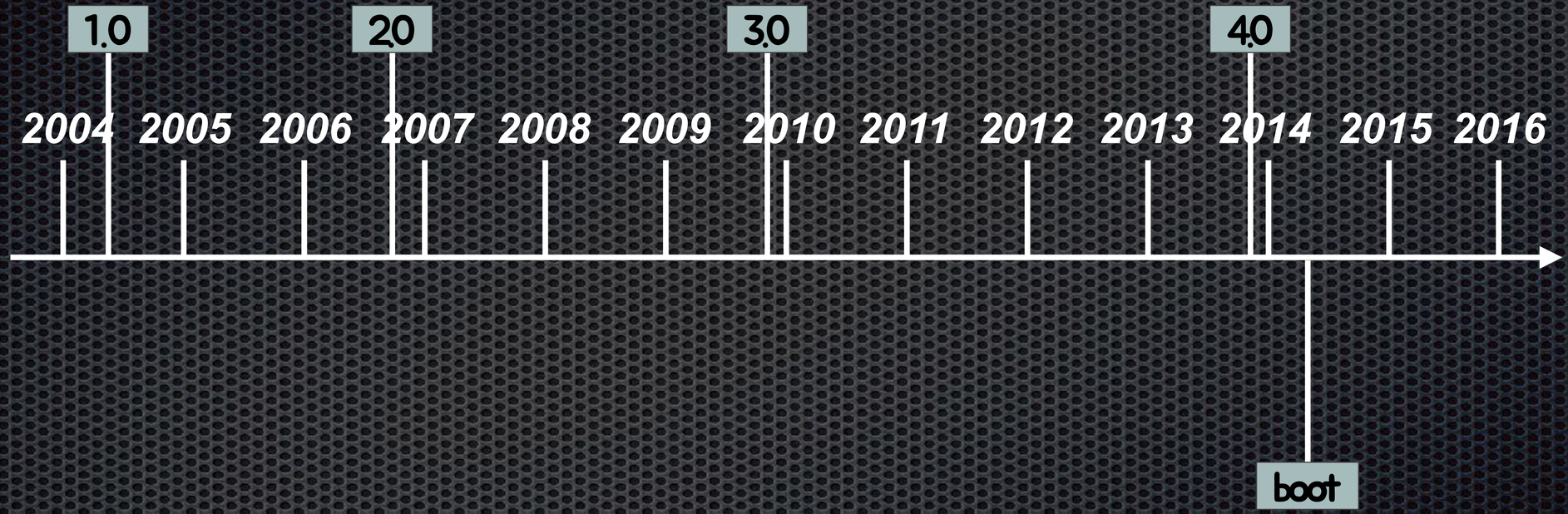


Spring Boot를 모를 때 문제를 해결하려고 하는 Boundery



Spring Boot를 이해한다면 우리가 문제를 해결할 수 있는
Boundery는 넓어질 것!

스프링 역사



10여년 전에 등장한 스프링

웹에는 10년간 축적되온 Legacy 방법들이 (아주 많이) 존재

프리마커 어떻게 하지? 구글링!

freemarker + spring configuration and simplest example - Stack Overflow

stackoverflow.com/.../freemarker-spring-configuration-and-simpl... ▾ 이 페이지 번역하기

2015. 3. 10. - Despite there are a lot of discussion around **freemarker** + **spring** ... This example is using java-based config and is still using a web project, but ...

날짜도 비교적 얼마 안됐네!

8

pom.xml

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context-support</artifactId>
</dependency>
<dependency>
    <groupId>org.freemarker</groupId>
    <artifactId>freemarker</artifactId>
</dependency>
```

applicationContext.xml

```
<bean id="freeMarkerConfigurationFactory" class="org.springframework.ui.freemarker.F
    <property name="templateLoaderPath" value="classpath:/META-INF/freemarker"/>
    <property name="preferFileSystemAccess" value="false"/>
</bean>
<bean id="freeMarkerConfiguration" class="freemarker.template.Configuration" factory
```

끝! 동작도 잘 되네?

```
@Bean
public ViewResolver viewResolver() {
    FreeMarkerViewResolver resolver = new FreeMarkerViewResolver();
    resolver.setPrefix("");
    resolver.setSuffix(".ftl");
    return resolver;
}

@Bean
public FreeMarkerConfigurer freemarkerConfig() {
    FreeMarkerConfigurer freeMarkerConfigurer = new FreeMarkerConfigurer();
    freeMarkerConfigurer.setTemplateLoaderPath("classpath:/templates/");
    return freeMarkerConfigurer;
}
```

8

pom.xml

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context-support</artifactId>
</dependency>
<dependency>
    <groupId>org.freemarker</groupId>
    <artifactId>freemarker</artifactId>
</dependency>
```

applicationContext.xml

```
<bean id="freeMarkerCon       tionFactory" class="       springframework.ui.freemarker.F
    <property name="templ       aderPath" value="       path:/META-INF/freemarker"/>
    <property name="preferl       stemAccess" v       "false"/>
</bean>
<bean id="freeMarkerConfigura       class="f       rker.template.Configuration" factory
```

```
@Bean
public ViewResolver viewResolve       {
    FreeMarkerViewResolver res       er = new       eMarkerViewResolver();
    resolver.setPrefix("");
    resolver.setSuffix(".f       ;
    return resolver;
}

@Bean
public FreeMarkerConfigurer freemarkerConfig() {
    FreeMarkerConfigurer freeMarkerConfigurer = new FreeMarkerConfigurer();
    freeMarkerConfigurer.setTemplateLoaderPath("classpath:/templates/");
    return freeMarkerConfigurer;
}
```

```
compile('org.springframework.boot:spring-boot-starter-freemarker')
```

우리가 쓰는 것은 스프링부트

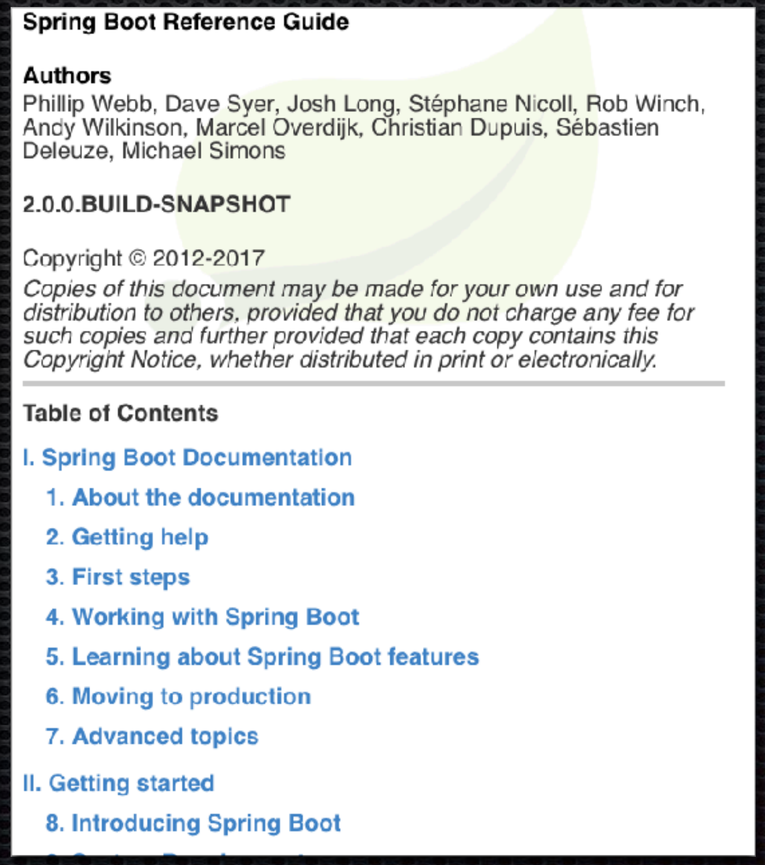
스프링 부트를 대하는 자세

어떻게 알 수 있을까?

Reference Guide

Spring Boot Reference Guide

- https://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/
- https://github.com/ihoneymon/translate-spring-boot-reference/blob/master/README.md



Spring Boot Reference Guide

Authors

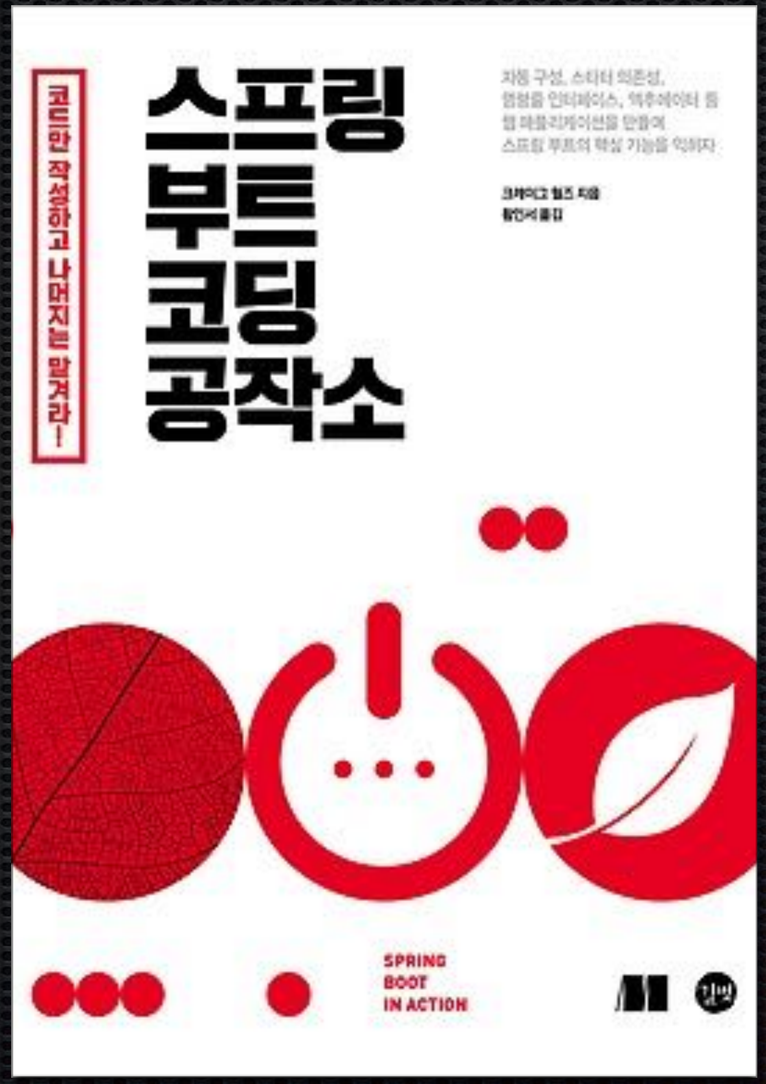
Phillip Webb, Dave Syer, Josh Long, Stéphane Nicoll, Rob Winch, Andy Wilkinson, Marcel Overdijk, Christian Dupuis, Sébastien Deleuze, Michael Simons

2.0.0.BUILD-SNAPSHOT

Table of Contents

추천

스프링
부트
코딩
공작소

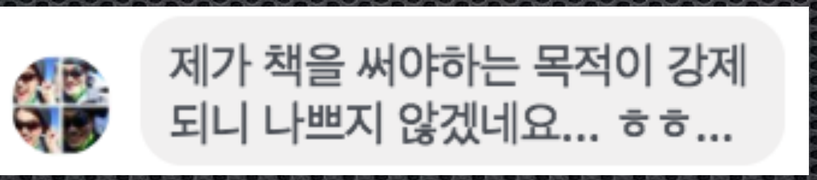


추천

스프링부트
프로그래밍
입문

추천

Boot!
Spring Boot

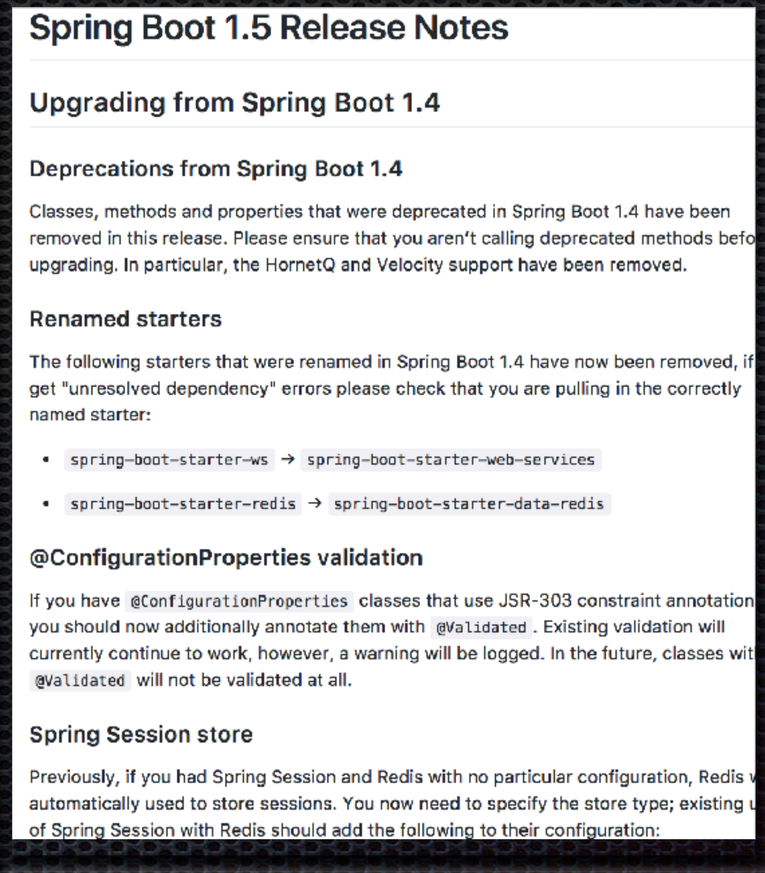


COMING
SOON!

Release Note

Spring Boot Release Notes

- https://github.com/spring-projects/spring-boot/wiki



Spring Boot 1.5 Release Notes

Upgrading from Spring Boot 1.4

Deprecations from Spring Boot 1.4

Classes, methods and properties that were deprecated in Spring Boot 1.4 have been removed in this release. Please ensure that you aren't calling deprecated methods befo upgrading. In particular, the HornetQ and Velocity support have been removed.

Renamed starters

The following starters that were renamed in Spring Boot 1.4 have now been removed, if get "unresolved dependency" errors please check that you are pulling in the correctly named starter:

- `spring-boot-starter-ws` → `spring-boot-starter-web-services`
- `spring-boot-starter-redis` → `spring-boot-starter-data-redis`

@ConfigurationProperties validation

If you have `@ConfigurationProperties` classes that use JSR-303 constraint annotation you should now additionally annotate them with `@Validated`. Existing validation will currently continue to work, however, a warning will be logged. In the future, classes wit `@Validated` will not be validated at all.

Spring Session store

Previously, if you had Spring Session and Redis with no particular configuration, Redis automatically used to store sessions. You now need to specify the store type; existing of Spring Session with Redis should add the following to their configuration:

직접 보자!

`Starter Dependency`

필요한 Dependency를 찾아야 해!

```
compile('org.springframework:spring-web:4.3.6.RELEASE')
compile('org.springframework.data:spring-data-jpa:1.11.0.RELEASE')
compile('org.hibernate:hibernate-entitymanager:5.0.11.Final')
```

JPA를 사용하기 위해 필요한 Dependency를 찾아야 한다

Starter Dependency

```
ext {
    springBootVersion = '1.5.2.RELEASE'
}
```

```
compile('org.springframework.boot:spring-boot-starter-web')
compile('org.springframework.boot:spring-boot-starter-data-jpa')
```

Starter Dependency를 사용한다면?

Starter Dependency

spring-boot-starter-*

└ Meta-INF

└ Pom.xml

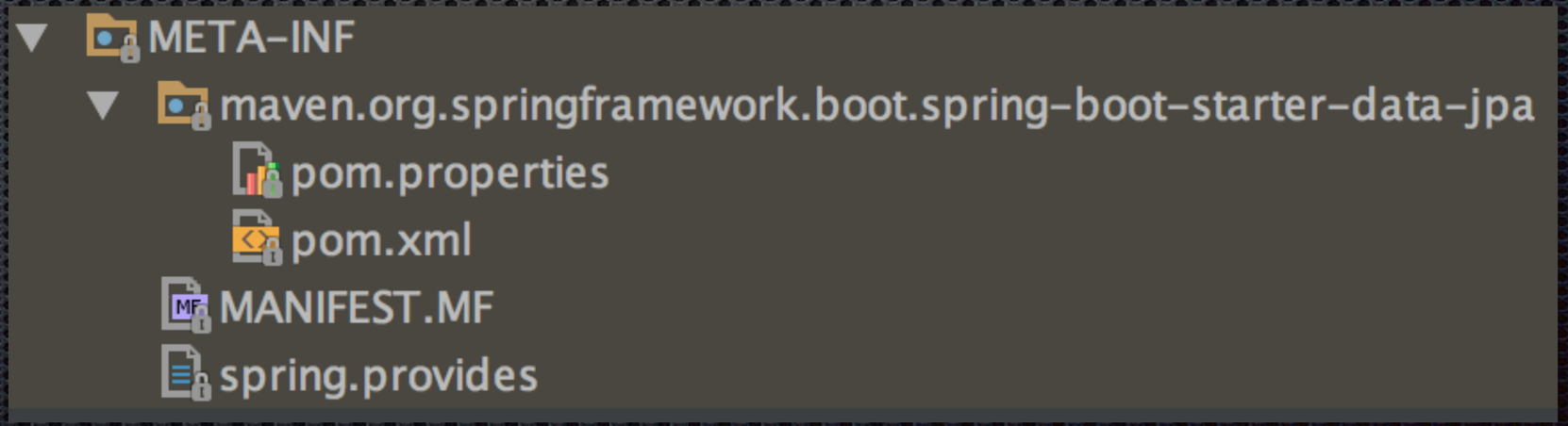
Starter Dependency

```
buildscript {
    ext {
        springBootVersion = '1.5.2.RELEASE'
    }
```

```
compile('org.springframework.boot:spring-boot-starter-data-jpa')
```

↓

org.springframework.boot:spring-boot-starter-data-jpa:1.5.2.RELEASE



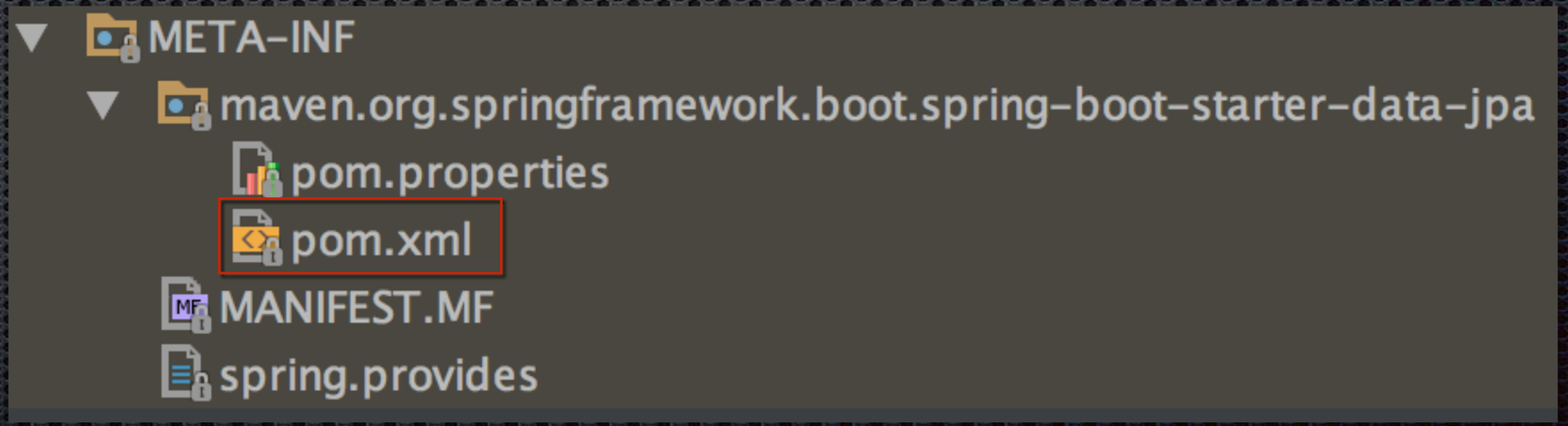
Starter Dependency

```
buildscript {
    ext {
        springBootVersion = '1.5.2.RELEASE'
    }
```

```
compile('org.springframework.boot:spring-boot-starter-data-jpa')
```

↓

org.springframework.boot:spring-boot-starter-data-jpa:1.5.2.RELEASE



Starter Dependency

org.springframework.boot:spring-boot-starter-data-jpa:1.5.2.RELEASE

pom.xml

```
<dependencies>
    <artifactId>spring-boot-starter</artifactId>
    <artifactId>spring-boot-starter-aop</artifactId>
    <artifactId>spring-boot-starter-jdbc</artifactId>
    <artifactId>hibernate-core</artifactId>
    <artifactId>hibernate-entitymanager</artifactId>
    <artifactId>javax.transaction-api</artifactId>
    <artifactId>spring-data-jpa</artifactId>
</dependencies>
```

Starter Dependency는 필요한 Dependency를 정의해놓았다!

Starter Dependency

spring-boot-starter-*

└ Meta-INF

└ Pom.xml

Starter Dependency는 필요한 Dependency를 Pom.xml에 정의해놓았다!

Version 간 호환성도 찾아야 해!

```
compile('org.springframework:spring-web:4.3.7.RELEASE')
compile('org.springframework.data:spring-data-jpa:1.11.1.RELEASE')
compile('org.hiberante:hibernate-entitymanager:jar:5.0.12.Final')
```

Version Matching에 대한 보장이 있는가? 이것도 찾아야 한다

Version Matching

```
buildscript {
    ext {
        springBootVersion = '1.5.2.RELEASE'
    }
    repositories {
        mavenCentral()
        maven { url "https://repo.spring.io/snapshot" }
        maven { url "https://repo.spring.io/milestone" }
        maven { url "https://repo.spring.io/release" }
    }
    dependencies {
        classpath("org.springframework.boot:spring-boot-gradle-plugin:${springBootVersion}")
    }
}
```

```
compile('org.springframework.boot:spring-boot-starter-web')
compile('org.springframework.boot:spring-boot-starter-data-jpa')
```

Stater Dependency를 사용한다면?

Version Matching

dependency management

- spring-boot-dependencies
  - spring-boot-parent
    - spring-boot-starters
      - spring-boot-stater-*

Version Matching

org.springframework.boot:spring-boot-starter-data-jpa:1.5.2.RELEASE


pom.xml

```
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starters</artifactId>
    <version>1.5.2.RELEASE</version>
</parent>
```

Parent로 사용하는 spring-boot-staters

Version Matching

spring-boot / spring-boot-starters / pom.xml

```
<parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-parent</artifactId>
        <version>2.0.0.BUILD-SNAPSHOT</version>
        <relativePath>../spring-boot-parent</relativePath>
</parent>
```

spring-boot-staters의 parent는 spring-boot-parent

https://github.com/spring-projects/spring-boot/blob/master/spring-boot-starters/pom.xml

Version Matching

spring-boot / spring-boot-parent / pom.xml

```
<dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>org.apache.maven</groupId>
                <artifactId>maven-aether-provider</artifactId>
                <version>3.2.1</version>
            </dependency>
        </dependencies>
</dependencyManagement>
```

spring-boot-parent는 개발환경에 대한
Dependency들을 Management하고 있다

https://github.com/spring-projects/spring-boot/blob/master/spring-boot-parent/pom.xml

Version Matching

spring-boot / spring-boot-parent / pom.xml

```
<parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-dependencies</artifactId>
        <version>2.0.0.BUILD-SNAPSHOT</version>
        <relativePath>../spring-boot-dependencies</relativePath>
</parent>
```

spring-boot-parent의 parent는 spring-boot-dependecies

https://github.com/spring-projects/spring-boot/blob/master/spring-boot-parent/pom.xml

Version Matching

spring-boot / spring-boot-dependencies / pom.xml

```
<hazelcast-hibernate4.version>3.7.1</hazelcast-hibernate4.version>
<hazelcast-hibernate5.version>1.1.3</hazelcast-hibernate5.version>
<hibernate.version>5.0.12.Final</hibernate.version>
<hibernate-validator.version>5.3.4.Final</hibernate-validator.version>
```

```
</dependencyManagement>
```

spring-boot-dependencies는 spring boot 관련된
Depenency들을 Management하고 있다

https://github.com/spring-projects/spring-boot/blob/v1.5.2.RELEASE/spring-boot-dependencies/pom.xml

Version Matching

dependency management

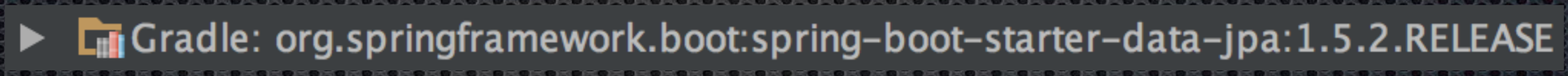
- spring-boot-dependencies
  - spring-boot-parent
    - spring-boot-starters
      - spring-boot-stater-*

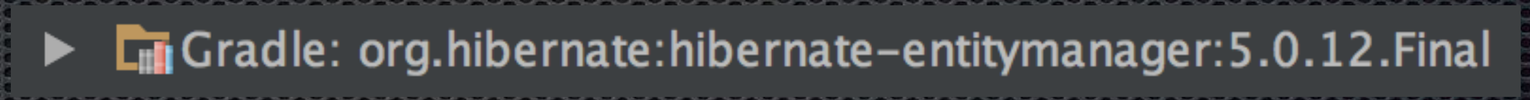
spring-boot-dependencies는 spring boot 관련된
Depenency들을 Management하고 있다

Starter Dependency

```
buildscript {
    ext {
        springBootVersion = '1.5.2.RELEASE'
    }
```

```
compile('org.springframework.boot:spring-boot-starter-data-jpa')
```





Stater Dependency만으로 필요했던 Dependency가
Version까지 딱!

`Auto Configuration`

Auto Configuration

```java
@Bean
public DataSource getDefaultDataSource() {
    DataSource dataSource = new DataSource();
    dataSource.setUrl(env.getProperty("jdbc.url"));
    dataSource.setUsername(env.getProperty("jdbc.user"));
    dataSource.setPassword(env.getProperty("jdbc.password"));

    dataSource.setDriverClassName(env.getProperty("jdbc.driverClassName"));
```

```
example.jdbc.url=jdbc:mysql://localhost:3306/kingbbode
example.jdbc.user=root
example.jdbc.password=kingbbode
example.jdbc.driverClassName=com.mysql.jdbc.Driver
```

Auto Configuration

```java
@Bean
public DataSource getDefaultDataSource() {
    DataSource dataSource = new DataSource();
    dataSource.setUrl(env.getProperty("jdbc.url"));
    dataSource.setUsername(env.getProperty("jdbc.user"));
    dataSource.setPassword(env.getProperty("jdbc.password"));

    dataSource.setDriverClassName(env.getProperty("jdbc.driverClassName"));
```

```
example.jdbc.url=jdbc:mysql://localhost:3306/kingbbode
example.jdbc.user=root
example.jdbc.password=kingbbode
example.jdbc.driverClassName=com.mysql.jdbc.Driver
```

Auto Configuration

```
spring.datasource.url=jdbc:h2:~/wedding;DB_CLOSE_DELAY=-1;DB_CLOSE_ON_EXIT=FALSE
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=
```

application.properties에 간단한 정보만 입력

Auto Configuration

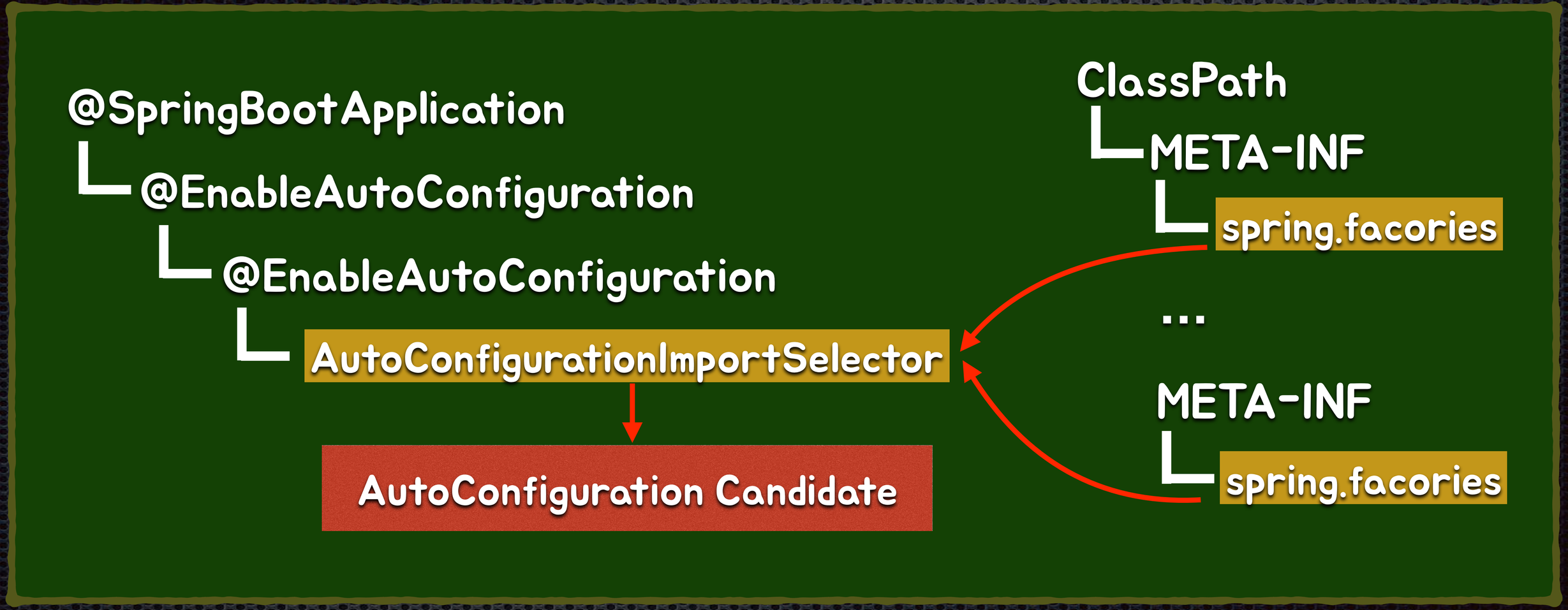
```java
@Bean
public ViewResolver viewResolver() {
    FreeMarkerViewResolver resolver = new FreeMarkerViewResolver();
    resolver.setCache(true);
    resolver.setPrefix("");
    resolver.setSuffix(".ftl");
    resolver.setContentType("text/html; charset=UTF-8");
    return resolver;
}

@Bean
public FreeMarkerConfigurer freemarkerConfig() {
    FreeMarkerConfigurer freeMarkerConfigurer = new FreeMarkerConfigurer();
    freeMarkerConfigurer.setTemplateLoaderPath("classpath:/templates/");
    return freeMarkerConfigurer;
}
```

Auto Configuration

Freemarker를 사용할 경우 아무 것도 필요하지 않다.

어떻게?

Auto Configuration



@EnableAutoConfiguration

```java
@SpringBootApplication
public class SpringBootAttitudeApplication {
```

↓

```java
@EnableAutoConfiguration
@ComponentScan(excludeFilters = {
        @Filter(type = FilterType.CUSTOM, classes = TypeExcludeFilter.class),
        @Filter(type = FilterType.CUSTOM, classes = AutoConfigurationExcludeFilter.class) })
public @interface SpringBootApplication {
```

↓

```java
@Import(EnableAutoConfigurationImportSelector.class)
public @interface EnableAutoConfiguration {
```

핵심은 EnalbeAutoConfigurationImportSelector!

spring-boot-autoconfigure

org.springframework.boot:spring-boot-starter

pom.xml

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-autoconfigure</artifactId>
</dependency>
<dependency>
```

EnableAutoConfiguration을 포함한 Class를 가지고 있는
spring-boot-autoconfigure는
spring-boot-starter가 Denpendency로 가지고 있다

AutoConfigurationImportSelector

```
protected List<String> getCandidateConfigurations(AnnotationMetadata metadata,
        AnnotationAttributes attributes) {
    List<String> configurations = SpringFactoriesLoader.loadFactoryNames(
            getSpringFactoriesLoaderFactoryClass(), getBeanClassLoader());
    Assert.notEmpty(configurations,
            "No auto configuration classes found in META-INF/spring.factories. If you "
                    + "are using a custom packaging, make sure that file is correct.");
    return configurations;
}
```

```
protected Class<?> getSpringFactoriesLoaderFactoryClass() {
    return EnableAutoConfiguration.class;
}
```

ClassPath의 모든 라이브러리의
`META-INF/spring.factories`에서
EnableAutoConfiguration 설정 리스트를
읽어온다

spring.factories

```
# Auto Configure
org.springframework.boot.autoconfigure.EnableAutoConfiguration=\
org.springframework.boot.autoconfigure.admin.SpringApplicationAdminJmxAutoConfiguration,\
org.springframework.boot.autoconfigure.aop.AopAutoConfiguration,\
org.springframework.boot.autoconfigure.amqp.RabbitAutoConfiguration,\
org.springframework.boot.autoconfigure.batch.BatchAutoConfiguration,\
org.springframework.boot.autoconfigure.cache.CacheAutoConfiguration,\
org.springframework.boot.autoconfigure.cassandra.CassandraAutoConfiguration,\
org.springframework.boot.autoconfigure.cloud.CloudAutoConfiguration,\
org.springframework.boot.autoconfigure.context.ConfigurationPropertiesAutoConfiguration,\
org.springframework.boot.autoconfigure.context.MessageSourceAutoConfiguration,\
org.springframework.boot.autoconfigure.context.PropertyPlaceholderAutoConfiguration,\
org.springframework.boot.autoconfigure.couchbase.CouchbaseAutoConfiguration,\
```

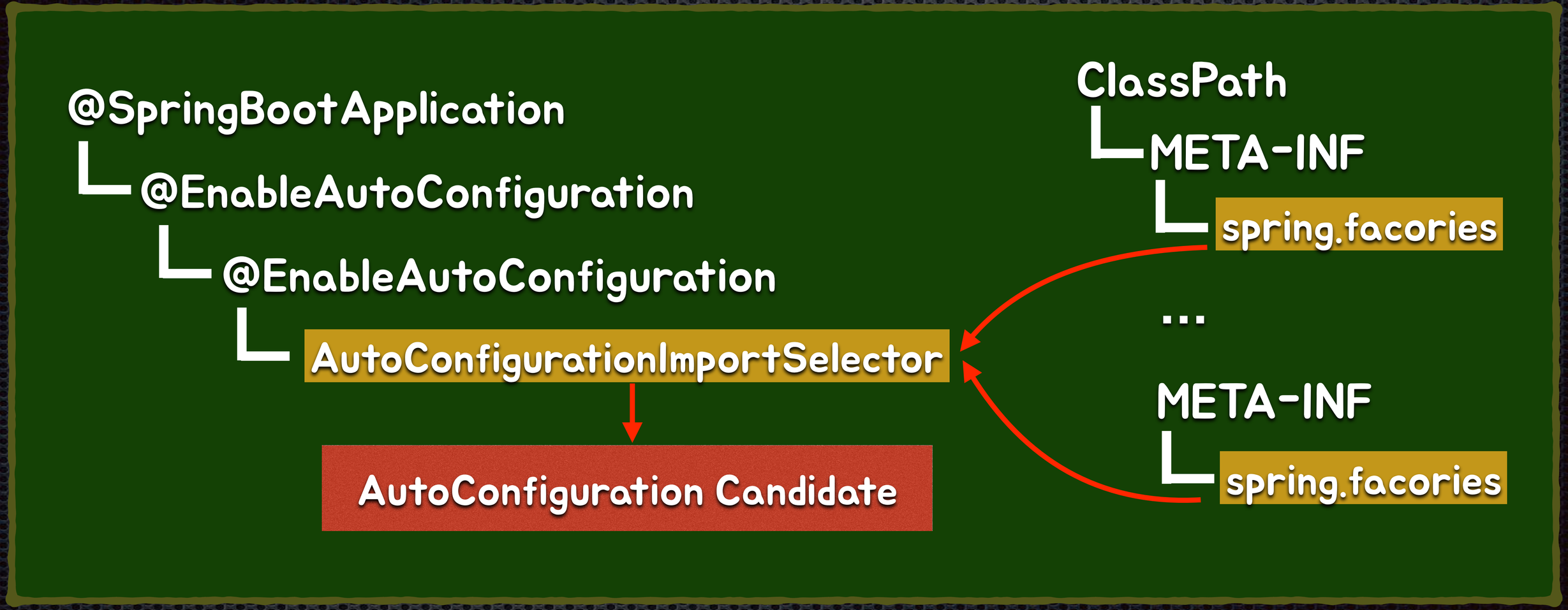
.

.

.

spring.factories는 설정 리스트를 정의

Auto Configuration



조건에 의한
Auto Configuration

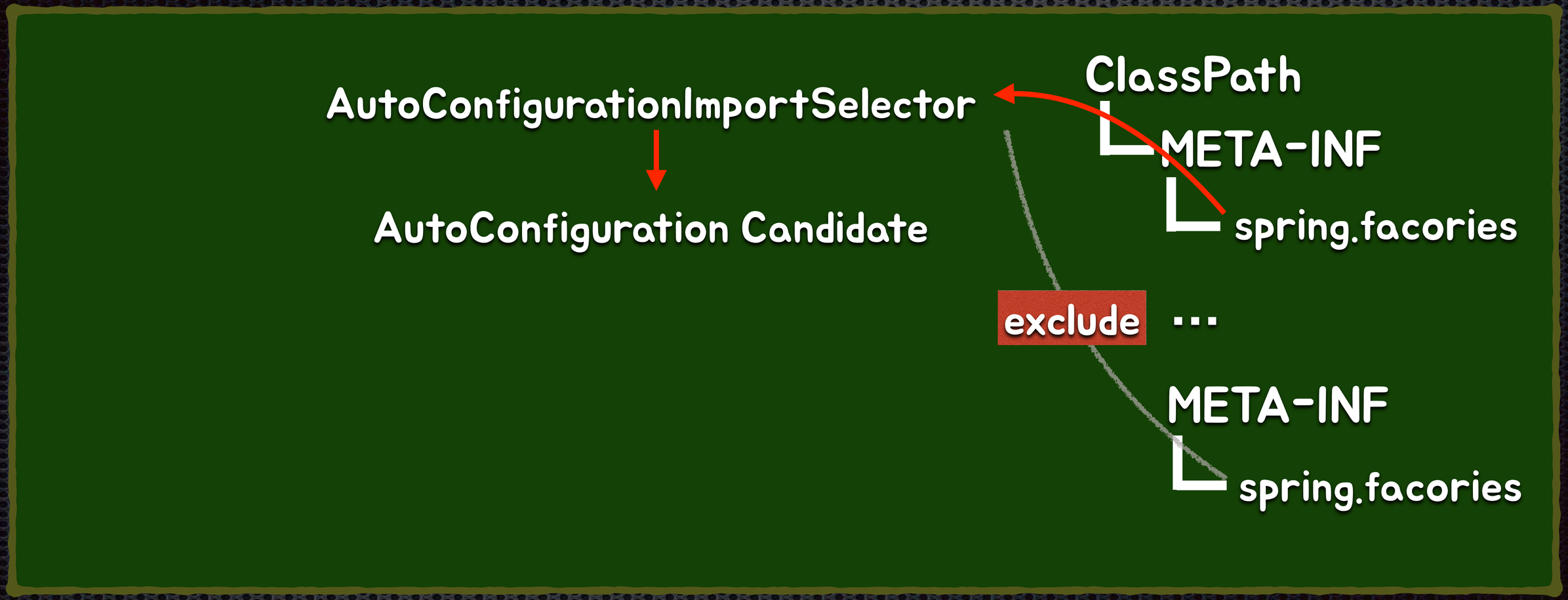
Exclude

```
@SpringBootApplication(exclude = {FreeMarkerAutoConfiguration.class})
```

```
public @interface SpringBootApplication {

    @AliasFor(annotation = EnableAutoConfiguration.class, attribute = "exclude")
    Class<?>[] exclude() default {};

    @AliasFor(annotation = EnableAutoConfiguration.class, attribute = "excludeName")
    String[] excludeName() default {};
```

Auto Configuration Exclude

조건에 의한 AutoConfiguration



Conditional

```java
@Retention(RetentionPolicy.RUNTIME)
@Target({ElementType.TYPE, ElementType.METHOD})
public @interface Conditional {

    /**
     * All {@link Condition}s that must {@linkplain Condition#matches match}
     * in order for the component to be registered.
     */
    Class<? extends Condition>[] value();

}
```

Spring Framework 4.0에서 등장한 @Conditional

Conditional

```java
public interface Condition {

    /**
     * Determine if the condition matches.
     * @param context the condition context
     * @param metadata metadata of the {@link org.springframework.core.type.AnnotationMetadata class}
     * or {@link org.springframework.core.type.MethodMetadata method} being checked.
     * @return {@code true} if the condition matches and the component can be registered
     * or {@code false} to veto registration.
     */
    boolean matches(ConditionContext context, AnnotatedTypeMetadata metadata);

}
```

matches가 true이면 Bean을 등록

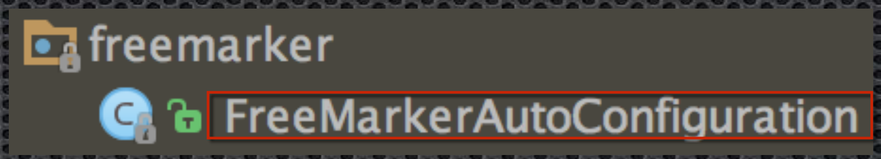
Conditional 확장

@ConditionalOnBean : 특정 Bean이 있을 때
@ConditionalOnMissingBean : 특정 Bean이 없을 때
@ConditionalOnClass : 특정 Class가 있을 때

Condition 인터페이스가 구현되어있는 확장 Annotation이 있다!

Conditional

org.springframework.boot:spring-boot-autoconfigure:1.5.2.RELEASE



spring-boot-autoconfigure
-> freemarker/FreemarkerAutoConfiguration

Conditional

```
@ConditionalOnClass({ freemarker.template.Configuration.class,
        FreeMarkerConfigurationFactory.class })
@AutoConfigureAfter(WebMvcAutoConfiguration.class)
@EnableConfigurationProperties(FreeMarkerProperties.class)
public class FreeMarkerAutoConfiguration {
```

```
@ConditionalOnNotWebApplication
public static class FreeMarkerNonWebConfiguration
```

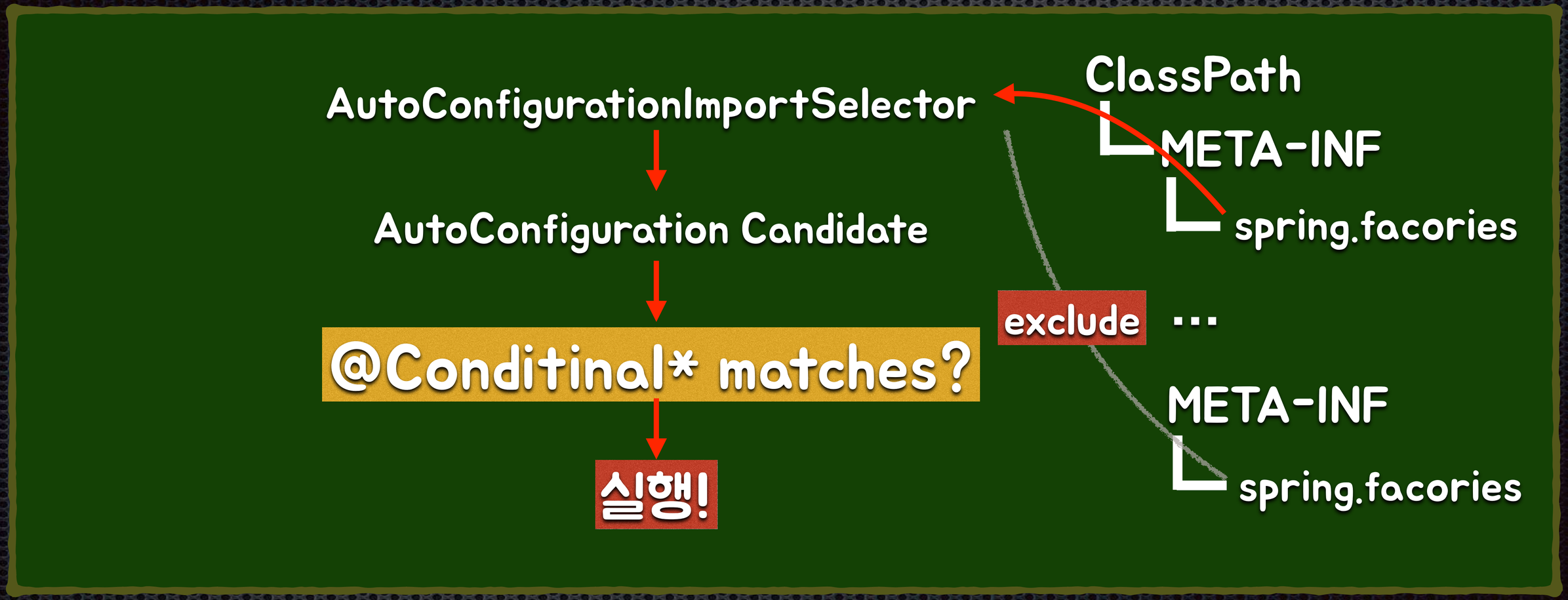
```
@ConditionalOnMissingBean
public FreeMarkerConfigurationFactoryBean
```

```
@ConditionalOnEnabledResourceChain
public ResourceUrlEncodingFilter resourceUrlEncodingFilter()
```

```
@ConditionalOnProperty(name = "spring.freemarker.enabled", matchIfMissing = true)
public FreeMarkerViewResolver freeMarkerViewResolver() {
```

AutoConfiguration은 @Conditional(확장)을 사용

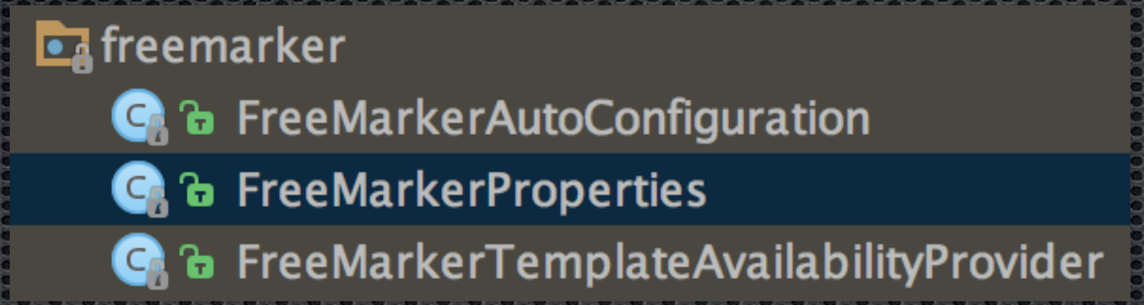
조건에 의한 AutoConfiguration



`Properties`

Properties

org.springframework.boot:spring-boot-autoconfigure:1.5.1.RELEASE




```
@ConfigurationProperties(prefix = "spring.freemarker")
public class FreeMarkerProperties extends AbstractTemplateVie
```

spring boot는 Java Config보다 더 간단히
Properties나 Yml로 설정을 제공

Properties

FreeMarkerProperties

```
public static final String DEFAULT_TEMPLATE_LOADER_PATH = "classpath:/templates/";

public static final String DEFAULT_PREFIX = "";

public static final String DEFAULT_SUFFIX = ".ftl";
```

기본 값의 선언을 볼 수 있다.

http://docs.spring.io/spring-boot/docs/current/reference/html/common-application-properties.html

같이 까보기
(소스코드)

https://github.com/kingbbode/spring-boot-attitude

Table 13.1. Spring Boot application starters

spring-boot-starter-freemarker	Starter for building MVC web applications using FreeMarker views

같이 까보기
(Reference Guide)

Spring Boot 1.3.0 Release Notes

Upgrading from Spring Boot 1.2

Deprecations from Spring Boot 1.2

Classes, methods and properties that were deprecated in Spring Boot 1.2 have been removed in this release. Please ensure that you aren't calling deprecated methods before upgrading.

Jackson

Spring Boot 1.2 registered any Jackson Module beans with every `ObjectMapper` in the application context. This made it impossible to take complete control of an `ObjectMapper` bean's modules. Spring Boot 1.3 will only register Jackson Module beans with `ObjectMappers`

같이 까보기
(Release Note)

Conclusion

- 스프링 부트는 스프링 프레임워크를 지원해주는 도구
  - Stater Dependency
  - Auto Configuration
  - Properties
- 무엇을 해주는지는 알고 쓰자!
  - 알면 알수록 개발 생산성이 높아질 것!


그 뜻은.. 한때는 뭔가를
두려웠는데 이제는 그렇지 않다는 뜻이야

Q&A