Clean Spring

September 2016 KCSUG

Frank Moley

"It is not enough for code to work."

"It is not enough for code to work."

~ Robert "Uncle Bob" Martin

This Presentation

- I have a few points I want to specifically discuss
- I would really like to get this to be more of a group discussion, please don't be shy
- My opinions are just that, my opinions. Everyone is entitled to their own. Keep this respectful!

- I would really like to get this to be more of a group discussion, please don't be shy
- My opinions are just that, my opinions. Everyone is entitled to their own. Keep this respectful!

My Topics

- OOP
- Configuration
- Exception Handling
- Separation of Concerns/Code Structure/Abstractions
- Unit Testing

OOP in Spring

* my biggest issue with code written leveraging Spring

Java and OOP

- Before we talk about Spring, let us come to agreement on Java being an OOP language
- What is an object, POJO, DTO, bean
- Constructors, setters, autowired and OOP

- Before we talk about Spring, let us come to agreement on Java being an OOP language
- What is an object, POJO, DTO, bean
- Constructors, setters, autowired and OOP
- When do we use each?

Configuration

I see what works

Use what works

- XML is dead, give up, let it go
- Quit mixing and matching, just stick with one
- Leverage AutoConfig and Component Scan
- Don’t configure what you get for free…

Exception Handling

Exception Handling

Don’t continue the pain

- There is a reason Spring transforms exceptions to RTE
- Quit the destruction of good contracts by throwing exceptions from lower levels
- If you aren’t going to handle the exception, convert it to an RTE for your system… “The buck stops here” mentality

- There is a reason Spring transforms exceptions to RTE
- Quit the destruction of good contracts by throwing exceptions from lower levels
- If you aren't going to handle the exception, convert it to an RTE for your system… "The buck stops here" mentality

Code Structure

Don't do too much

Don’t do too much

- Yes microservices and micro applications are the thing today, but that doesn’t mean you should do everything in your controller
- Stop doing thirty things in a single method
- Yes Spring reduces code, don’t reverse the gains of using Spring by munging the levels of your code together

Don’t do too little

- Don’t abstract to the point of pass throughs
- Don’t wrap Spring Abstractions

Don't do too little

- Don't abstract to the point of pass throughs
- Don't wrap Spring Abstractions
- Don't prematurely layer your code

Unit Testing

Unit Testing

Test

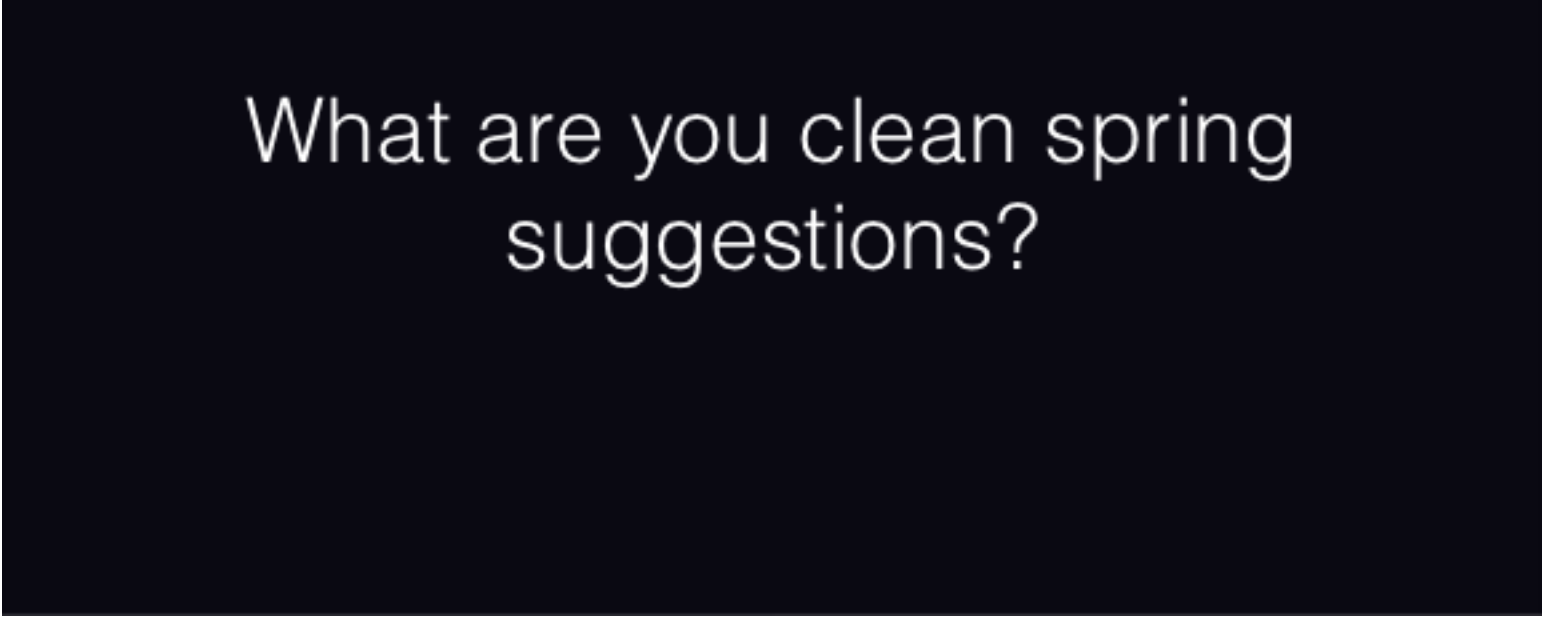
- TDD works
- Well tested code is usually cleaner code
- Leverage mocking frameworks and Spring abstractions (if you haven't used MockMVC, you should be)
- Unit tests != integration tests != system tests
- If your test is external to your CI process and it isn't automated, it isn't getting run

Testing continued

Testing continued

- If it cannot be tested, it isn't written cleanly
- If you need to jump through hoops to test it, it isn't written cleanly
- Since you are using TDD, there is always time to test

What are you clean spring suggestions?


What are you clean spring
suggestions?