Initial state

- Can use start.spring.io, we will start from the repo

Rest controller (step1)

```
data class Quote(val person: String, val text: String)

@RestController
@RequestMapping("/api/quotes")
class QuoteController {
    val quotes = mutableSetOf(
            Quote("Frank Underwood", "Money is the Mc-mansion in Sarasota that starts falling apart after 10 years."),
            Quote("Frank Underwood", "Power is a lot like real estate."))

    @RequestMapping
    fun allQuotes() = quotes
}
```

Kotlin Jackson module (step2)

- Point out how annotation array params are treated
  - Can use @PostMapping, @GetMapping, etc instead
  - Will be fixed in 1.1

```
@RequestMapping(method = arrayOf(RequestMethod.POST))
fun addQuote(@RequestBody quote: Quote) = quotes.add(quote)
```

Kotlin Jackson module (step2)

- Try this without jackson module, blows up on constructor

```
http POST localhost:8080/api/quotes person=Jon
text=Hi
```

- Add jackson module

```
compile 'com.fasterxml.jackson.module:jackson-
module-kotlin:2.7.1-2'
```

- No need to register with Spring Boot 1.4.0+

Rest template extension functions (step3)

→ Show OMDB API website

```
data class Title(@JsonProperty("Title") val title: String, @JsonProperty("Year") val year: String)
data class SearchResults(@JsonProperty("Search") val titles: List<Title>)

@RestController
@RequestMapping("/api/titles")
class TitleController(val restTemplate: RestTemplate) {
    @RequestMapping
    fun searchTitles(@RequestParam q: String) =
        restTemplate.getForObject<SearchResults>("http://www.omdbapi.com/?s=$q")
            .titles
}

@Bean
open fun objectMapperBuilder() =
        Jackson2ObjectMapperBuilder()
                .featuresToDisable(DeserializationFeature.FAIL_ON_UNKNOWN_PROPERTIES)
```

HATEOAS extension functions (step4)

```
compile 'org.springframework.hateoas:spring-hateoas'
```

Add to controller:

```
...
  .toPagedResources()
  .linkEachStatic("firstWord", { t -> "http://www.omdbapi.com/?s=${t.title.split(' ').first()}" })
```

CGLib problem (step5)

```
compile 'org.aspectj:aspectjweaver:1.8.7'`
```

```
@EnableAspectJAutoProxy(proxyTargetClass = true)

@Repository
class QuoteRepository {
    private val quotes = mutableSetOf(
            Quote("Frank Underwood", "Money is the Mc-mansion in Sarasota that starts falling apart after 10 years."),
            Quote("Frank Underwood", "Power is a lot like real estate. It’s all about location, location, location."))

    fun allQuotes() = quotes
    fun addQuote(q: Quote) = quotes.add(q)
}
```

CGLib problem (step5)

Add:

```
@Aspect
@Component
class LoggingAspect {
    @Before("execution(* org.kug.spring.QuoteRepository.*(..))")
    fun quoteRepositoryInteraction() {
        println("interacted with repository")
    }
}
```

Make changes to controller:

```
@RestController
@RequestMapping("/api/quotes")
class QuoteController(val repo: QuoteRepository) {
    @RequestMapping
    fun allQuotes() = repo.allQuotes()

    @RequestMapping(method = arrayOf(RequestMethod.POST))
    fun addQuote(@RequestBody quote: Quote) = repo.addQuote(quote)
}
```

CGLib problem solution (step6)

Remove `proxyTargetClass = true`

```
interface QuoteRepository {
    fun allQuotes(): Set<Quote>
    fun addQuote(q: Quote): Boolean
}

@Repository
class QuoteRepositoryImpl: QuoteRepository {
    private val quotes = mutableSetOf(
            Quote("Frank Underwood", "Money is the Mc-mansion in Sarasota that starts falling apart after 10 years."),
            Quote("Frank Underwood", "Power is a lot like real estate. It’s all about location, location, location."))

    override fun allQuotes() = quotes
    override fun addQuote(q: Quote) = quotes.add(q)
}
```

Field injection

Don't do this unless you need to! Ruins smart casts, etc.

```
@Autowired
lateinit var repo: QuoteRepository
```