12. Developing reactive APIs

# 12.1 Working with Spring WebFlux

-Typical servlet web framworks (like Spring MVC) are blocking and multithreaded in nature, using a single thread per connection.

## 12.1.1 Introducing Spring WebFlux

## 12.1.2 Writing reactive controllers

# 12.2 Defining functional request handlers

# 12.3 Testing reactive controllers

## 12.3.1 Testing GET requests

## 12.3.2 Testing POST requests

## 12.3.3 Testing with a live server

# 12.4 Consuming REST APIs reactively

## 12.4.1 GETting resources

## 12.4.2 Sending resources

## 12.4.3 Deleting resources

## 12.4.4 Handling errors

## 12.4.5 Exchanging requests

# 12.5 Securing reactive web APIs

## 12.5.1 Configuring reactive web security

## 12.5.2 Configuring a reactive user details service

# Summary