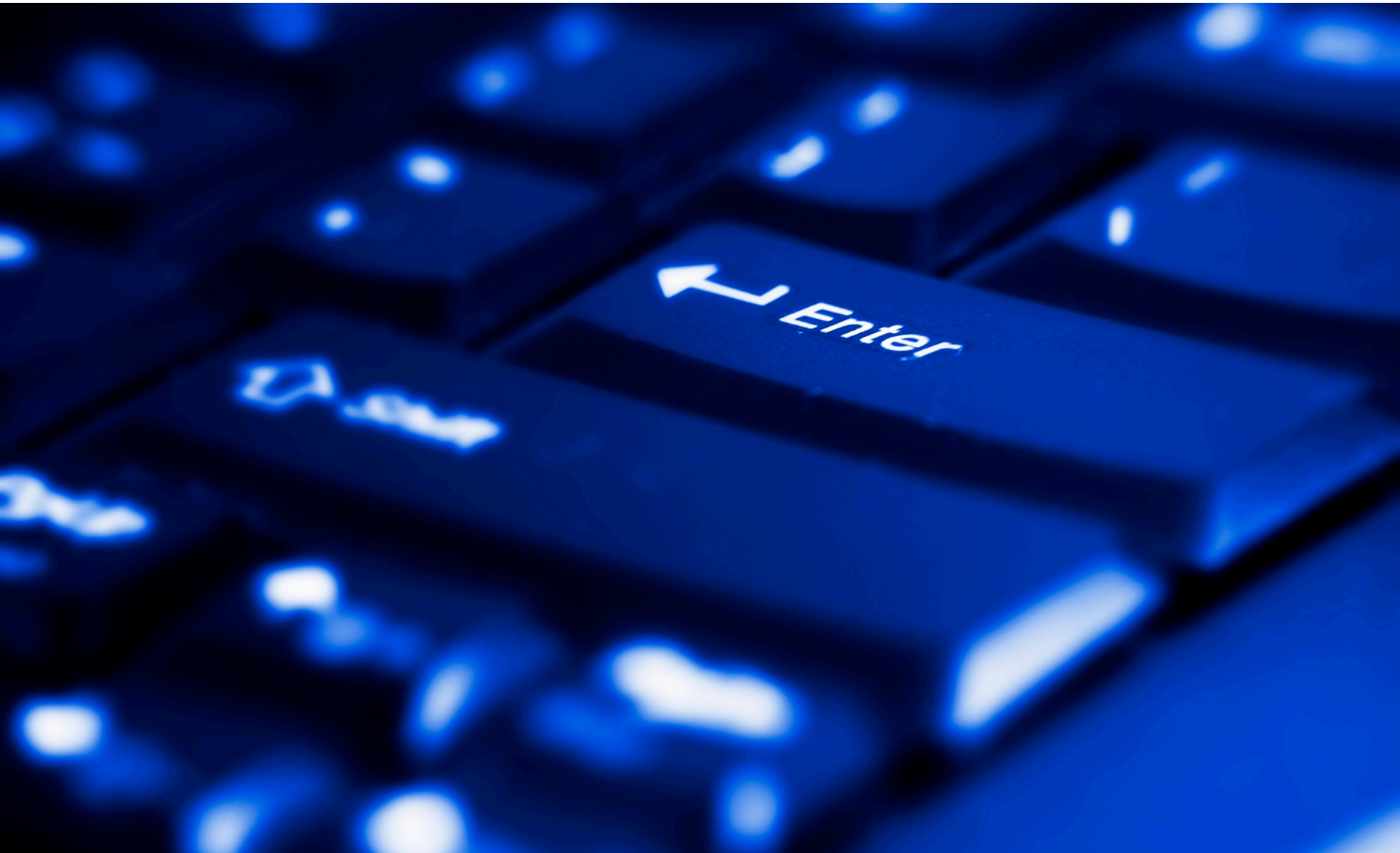
bns it

Spring Boot Fundamentals

Modern Application Architectures



Service

Modern Application Architectures

Spring MVC

- Part of Spring that's used to develop web applications
- Model View Controller
- Commonly used to develop
  - HTML web applications
  - REST endpoints
  - Serves „single page applications"

Spring Boot

- Convention over configuration for Spring
- Managed dependencies
- Production-ready features

Spring Boot = Spring + Jetty – XML Config

Spring Boot and Microservices

- Declares dependencies
- Configures Spring
- Configures Logging
- Loads properties
- Setups monitoring
- Adds security
- Adds metrics
- Talks to DB

Get started

Download, unzip and configure (PATH)

https://docs.spring.io/spring-boot/docs/current/reference/html/getting-started-installing-spring-boot.html

Eclipse STS

http://start.spring.io

Get started

- Configure project on http://start.spring.io
- You can also try **spring init** (spring help init)
- and then import project to IDE

- Other (suggested) option is to use Eclipse STS
- Run application
- Write example controller

Build with Maven

pom.xml (fragments)

```
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>1.4.1.RELEASE</version>
    <relativePath />
</parent>

<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
</dependencies>

<plugin>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-maven-plugin</artifactId>
</plugin>
```

Excercise: Generate two projects with different dependecies

Maven

Run the Spring Boot App from Maven:

\# **mvn spring-boot:run**

Creating „fat” JAR:

\# **mvn clean package**

(look inside the JAR)

Execute your JAR:

\# **java –jar target/your-jar.jar**

Example @RestController

```
@RestController
public class ExampleController {

    @RequestMapping("/")
    public String hello() {
        return "Hello World!";
    }

}
```

```
@RequestMapping(value="/{id}", method=RequestMethod.GET)
public ResponseEntity<Person> getPerson(@PathVariable Integer id){
    logger.info("PersonRestResponseEntityController  - getPerson - id: {}", id);
    Person person = personMapRepository.findPerson(id);
    return new ResponseEntity<>(person, HttpStatus.FOUND);
}
```

```
@RequestMapping(value="/{id}", method=RequestMethod.GET)
@ResponseStatus(HttpStatus.FOUND)
public Person getPerson(@PathVariable Integer id){
    logger.info("PersonRestController  - getPerson - id: {}", id);
    Person person = personMapRepository.findPerson(id);
    return person;
}
```

Client

Modern Application Architectures

```
@RestController
class GreetingsGateway {

    private RestTemplate restTemplate;

    @Autowired
    public GreetingsGateway(RestTemplate restTemplate) {
        super();
        this.restTemplate = restTemplate;
    }

    @RequestMapping("/hi/{name}")
    String greet(@PathVariable String name) {
        ResponseEntity<Greeting> responseEntity
            = this.restTemplate.exchange("http://greetings/greetings/{name}",
                HttpMethod.GET, /*body*/null, Greeting.class, name);
        return responseEntity.getBody().getGreeting();
    }
}
class Greeting {
    private String greeting;

    public String getGreeting() {
        return greeting;
    }
}
```