Βήματα Δημιουργίας Project και Οδηγίες Walkthrough

1. Δημιουργία του Spring Boot Project

Μπορείς να χρησιμοποιήσεις το Spring Initializr (https://start.spring.io/)

- Ανοίγεις το Spring Initializr.
- Ρυθμίζεις τις παραμέτρους:

Project: Maven Project

Language: Java

Spring Boot Version: Επιλέγεις την τρέχουσα σταθερή έκδοση (π.χ., 3.x.x).

Group: gr.uom

Artifact: init

Name: init

Packaging: Jar

Java Version: Επιλέγεις την έκδοση της Java που χρησιμοποιείς (π.χ., 17).

- Προσθέτεις τις εξαρτήσεις:

Spring Web για να δημιουργήσεις REST APIs.

- Πατάς **Generate** για να κατεβάσεις το project σε μορφή .zip.
- Αποσυμπιέζεις το .zip και το ανοίγεις στο IDE.

2. Αρχείο pom.xml

Dependencies:

Μέσα στο <dependencies>, προσθέτεις κάθε εξάρτηση με το <dependency> tag.

Οι εξαρτήσεις βοηθούν στη λήψη των βιβλιοθηκών που χρειάζεσαι για το project, όπως το spring-boot-starter-web για REST εφαρμογές.

Java Version:

Η Java version ορίζεται στο <properties>, και συγκεκριμένα στο <java.version>.

Είναι σημαντικό να ορίσεις την έκδοση της Java που θα χρησιμοποιήσεις (π.χ., 17) για να είναι συμβατή με το IDE και το build σύστημα. Εάν δεν παίζει με την 17, δοκίμασε με την **1.8**. Μπορείς να δεις ποια έκδοση έχει το σύστημα με την εντολή στο τερματικό **java --version**.

Spring Boot Version:

Η έκδοση του Spring Boot που χρησιμοποιεί το project μπαίνει και αυτή στο <properties> με την ετικέτα <spring-boot.version>.

Ορίζοντας την έκδοση του Spring Boot, το project διασφαλίζει τη χρήση μιας σταθερής και κατάλληλης βιβλιοθήκης για το development. Ασφαλής έκδοση είναι η **2.7.5**.

Build Plugins:

Το spring-boot-maven-plugin είναι απαραίτητο για να γίνει σωστά το build και να παραχθεί ένα αυτοδύναμο JAR, το οποίο θα μπορείς να τρέξεις ανεξάρτητα.

Η έκδοση του plugin μπορεί να ταιριάζει με την έκδοση του Spring Boot που έχεις ορίσει στα properties για συμβατότητα.

Γιατί Είναι Σημαντικές Αυτές οι Ρυθμίσεις;

- Συμβατότητα και Αξιοπιστία: Με το να ορίζεις τη συγκεκριμένη έκδοση Java και Spring Boot, το project διασφαλίζει ότι όλες οι βιβλιοθήκες είναι συμβατές και το περιβάλλον εργασίας σταθερό.
- Διαχείριση και Αναβάθμιση Εξαρτήσεων: Χάρη στο pom.xml, οι εξαρτήσεις μπορούν να αναβαθμιστούν εύκολα. Αντικαθιστώντας μια έκδοση, το Maven φροντίζει για την αυτόματη λήψη της νέας βιβλιοθήκης.
- Απλοποίηση του Build: Το spring-boot-maven-plugin κάνει εύκολο το build και τη διανομή του project, παράγοντας ένα ενιαίο JAR αρχείο που περιλαμβάνει όλες τις εξαρτήσεις.

Με αυτό τον τρόπο, το pom.xml βοηθά στη δομή και τη διαχείριση της εφαρμογής, κάνοντας το development και την ανάπτυξη πιο οργανωμένα και αξιόπιστα.

3. Δημιουργία των Classes και Controllers

Στη συνέχεια, δημιουργούμε τα αρχεία κώδικα.

1. Δημιουργία της Κλάσης Person

Η κλάση Person περιέχει δεδομένα σχετικά με ένα άτομο και μία μέθοδο getData() για επιστροφή των δεδομένων ως κείμενο.

```
public class Person {
    private String name;    // Όνομα ατόμου
    private int age;        // Ηλικία ατόμου
    private String location; // Τοποθεσία ατόμου

    // Κατασκευαστής για αρχικοποίηση των παραμέτρων του Person
    public Person(String name, int age, String location) {
        this.name = name;
        this.age = age;
        this.location = location;
    }
```

```
    // Μέθοδος που επιστρέφει τα δεδομένα του ατόμου σε μορφή κειμένου
    public String getData() {
        return "Hello " + name + " " + age + " from " + location;
    }
}
```

2. Δημιουργία του Controller (Main Controller)

Το αρχείο Controller.java περιέχει διάφορα endpoints για να καλείς μέσω HTTP αιτημάτων.

```
import java.util.ArrayList;
import java.util.List;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@RestController // Δηλώνουμε ότι αυτή η κλάση είναι REST Controller
@RequestMapping("/hello") // Όλα τα endpoints θα ξεκινούν με "/hello"
public class Controller {
     // Εδώ βάζουμε τα endpoint που θα χρησιμοποιήσουμε από τους clients
    // Βασικό endpoint που επιστρέφει ένα μήνυμα
    @GetMapping
    public String hello() {
        return "Hello World";
    }

    // Endpoint που λαμβάνει το όνομα του χρήστη από το path
   // (path="hello/{name}" ή απλά "") και το εμφανίζει
```

```
    @GetMapping(path="/{name}")
    public String helloName(@PathVariable(value="name") String name) {
        // Στο PathVariable δηλώνω στο value την τιμή που θέλουμε να
πάρει.
        // Και το όρισμα λέω ότι έρχεται από το path
        return "Hello " + name;
    }
    // URL για το endpoint: http://localhost:8081/hello/Stathis

    // Endpoint που λαμβάνει το όνομα και την ηλικία του χρήστη
    @GetMapping(path="/helloandage/{name}")
    public String helloNameAndAge(
        @PathVariable(value="name") String name,
        @RequestParam(value="age") int age) {
        // Στο PathVariable δηλώνω το value ποιο είναι αυτό που θέλουμε να
πάρει.
        // Και το όρισμα λέω ότι έρχεται από το path
        return "Hello " + name + " " + age;
    }
    // URL για το endpoint: http://localhost:8081/helloandage/Stathis?age=40

    // Endpoint που λαμβάνει το όνομα, ηλικία και τοποθεσία του χρήστη
    @GetMapping(path="/helloandlocation/{name}")
    public String helloNameAndLocation(
        @PathVariable(value="name") String name,
        @RequestParam(value="age") int age,
        @RequestParam(value="location") String location) {
```

```
//                // Στο PathVariable δηλώνω το value ποιο είναι αυτό που θέλουμε να πάρει.
//                return "Hello " + name + " " + age + " from " + location;


        // Αντί για το παραπάνω, δημιουργούμε αντικείμενο της κλάσης Person και επιστρέφουμε τα δεδομένα του
        Person person = new Person(name, age, location);
        // Επιστρέφω την εκτύπωση των δεδομένων του Person.
        return person.getData();
    }
    // URL για το endpoint: http://localhost:8081/helloandlocation/Stathis?age=40&amp;location=Thessaloniki


    // Endpoint που επιστρέφει λίστα με μηνύματα σε μορφή JSON
    @GetMapping(path="/list")
    public List<String> getList() {
        List<String> list = new ArrayList<>();
        list.add("Hola");
        list.add("Me llamo Estathis");
        list.add("Tengo 40 años");
        return list;
    }
}
```

3. Δημιουργία του GoodNight Controller

Ο GoodNight Controller περιέχει ένα απλό endpoint που επιστρέφει το μήνυμα "Good Night".

```
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
```

```java
@RestController // Δηλώνουμε ότι αυτή η κλάση είναι REST Controller
@RequestMapping("/goodnight") // Τα endpoints θα ξεκινούν με "/goodnight"
public class GoodNight {

    @GetMapping() // Endpoint που επιστρέφει το μήνυμα "Good Night"
    public String goodnight() {
        return "Good Night";
    }
}
// URL για το endpoint: http://localhost:8081/goodnight
```

4. Αρχική Κλάση (Entry Point)

Η κλάση InitApplication είναι το κύριο σημείο εκκίνησης της εφαρμογής Spring Boot. Στην συγκεκριμένη άσκηση, δεν κάνει κάτι.

```java
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication // Δηλώνει ότι αυτή η κλάση είναι το σημείο εκκίνησης του Spring Boot
public class InitApplication {

    public static void main(String[] args) {
        SpringApplication.run(InitApplication.class, args); // Εκκίνηση εφαρμογής
    }
}
```

Εκκίνηση της Εφαρμογής

- Άνοιξε την κύρια κλάση InitApplication και εκτέλεσέ την.
- Η εφαρμογή θα ξεκινήσει τοπικά στο http://localhost:8081 (ή στη θύρα που ορίζεται).

Για να αλλάξεις την πόρτα που τρέχει η εφαρμογή Spring Boot, μπορείς να το κάνεις εύκολα μέσω του αρχείου **application.properties** που βρίσκεται συνήθως στον φάκελο **src/main/resources**.

Με χρήση του application.properties

Πρόσθεσε την παρακάτω γραμμή στο αρχείο application.properties:

server.port=8081

Σε αυτό το παράδειγμα, η πόρτα αλλάζει από την προεπιλεγμένη (8080) στην πόρτα 8081. Μπορείς να ορίσεις οποιαδήποτε διαθέσιμη πόρτα επιθυμείς.

Έλεγχος των Endpoints

Αφού ξεκινήσει η εφαρμογή, μπορείς να δοκιμάσεις τα endpoints μέσω του browser ή ενός εργαλείου όπως το Postman:

http://localhost:8081/hello

http://localhost:8081/hello/Stathis

http://localhost:8081/helloandage/Stathis?age=40

http://localhost:8081/helloandlocation/Stathis?age=40&location=Thessaloniki

http://localhost:8081/hello/list

http://localhost:8081/goodnight

Λίστα με Annotations και Επεξήγηση

@SpringBootApplication

- Τοποθεσία: InitApplication κλάση.
- Λειτουργία: Σηματοδοτεί την κεντρική κλάση μιας Spring Boot εφαρμογής. Το συγκεκριμένο annotation ενεργοποιεί τρία βασικά χαρακτηριστικά:
    - @Configuration: Επιτρέπει τη δήλωση beans στην κλάση.
    - @EnableAutoConfiguration: Επιτρέπει στη Spring Boot να διαμορφώσει αυτόματα τις ρυθμίσεις της εφαρμογής.
    - @ComponentScan: Σαρώνει τον κώδικα για κλάσεις με τα annotations @Controller, @Service, @Repository ώστε να τις συμπεριλάβει στο Spring Context.

@RestController

- Τοποθεσία: Controller και GoodNight κλάσεις.
- Λειτουργία: Συνδυασμός των annotations @Controller και @ResponseBody. Δηλώνει ότι η κλάση αυτή είναι ένα RESTful controller, και οι μέθοδοι της επιστρέφουν JSON ή άλλο αντικείμενο ως HTTP response αντί για view.

@RequestMapping

- Τοποθεσία: Controller και GoodNight κλάσεις.
- Λειτουργία: Χρησιμοποιείται για τον ορισμό του βασικού URL στο οποίο θα απαντά η κλάση. Ορίζοντας π.χ. @RequestMapping("/hello") στην Controller κλάση, όλες οι διαδρομές (endpoints) μέσα στην κλάση θα ξεκινούν από το /hello.

@GetMapping

- Τοποθεσία: Στις μεθόδους των controllers.
- Λειτουργία: Δηλώνει ότι μια μέθοδος είναι HTTP GET request endpoint. Χρησιμοποιείται για να ανακτήσει δεδομένα από τον server, όπως στη μέθοδο hello() που επιστρέφει "Hello World" και τις άλλες helloName, helloNameAndAge.

@PathVariable

- Τοποθεσία: Στις παραμέτρους των μεθόδων όπως helloName(@PathVariable(value="name") String name).
- Λειτουργία: Δίνει τη δυνατότητα να λαμβάνει η μέθοδος δυναμικά τιμές από το URL (π.χ., /hello/Stathis), τοποθετώντας το path variable από το URL στο ορισμένο όρισμα της μεθόδου.

@RequestParam

- Τοποθεσία: Στις παραμέτρους των μεθόδων όπως helloNameAndAge(@RequestParam(value="age") int age).
- Λειτουργία: Λαμβάνει παραμέτρους από το URL, οι οποίοι ορίζονται μετά το ? στο URL (π.χ., /helloandage/Stathis?age=40). Χρησιμοποιείται για την πρόσβαση σε query parameters που δεν είναι μέρος της διαδρομής (path).

Αυτά είναι τα κύρια annotations που έχεις χρησιμοποιήσει στον κώδικά σου! Τα annotations κάνουν τον κώδικα πιο καθαρό και εύκολο στη διαχείριση, αυτοματοποιώντας λειτουργίες και μειώνοντας την ανάγκη για πολλές δηλώσεις.