# Spring boot? Usiamo anche Vaadin

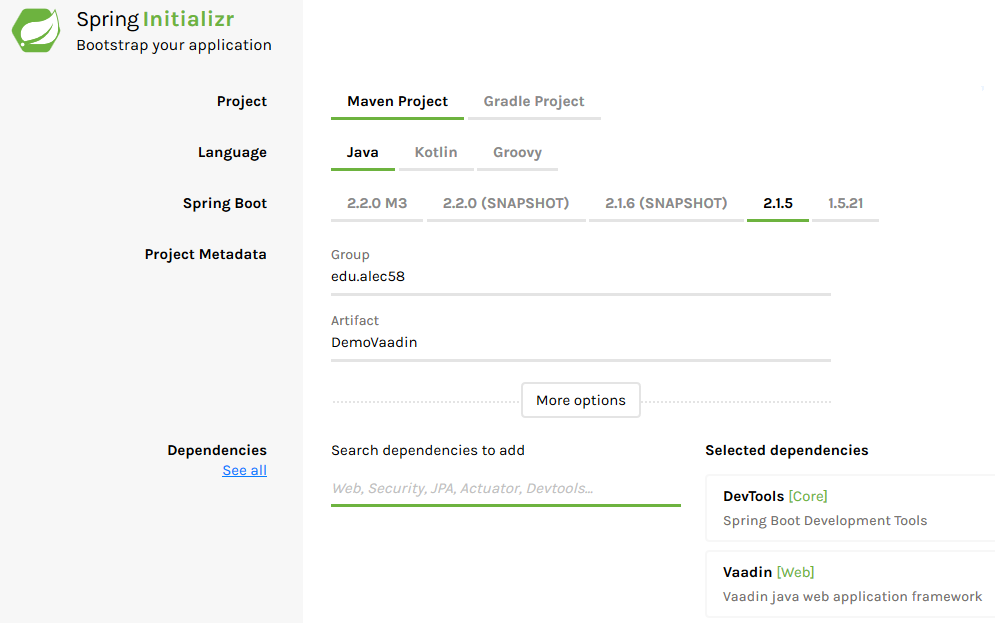
Nel post precedente ([Vaadin con Spring boot? Si può fare](https://alec58.altervista.org/2019/06/vaadin-con-spring-boot-si-puo-fare/)) abbiamo visto com'è semplice fare un progetto java che utilizzi Vaadin, grazie a Spring boot.

In quel progetto manca però la parte di test.

Se generiamo il progetto da Spring boot abbiamo anche questo, ma attualmente il modello non è perfettamente a punto, e dunque bisogna fare alcuni piccoli interventi.

## Generiamo il progetto: Spring Initializr

Come per tutti i progetti Spring boot, partiamo da Spring Initializr, all'indirizzo <https://start.spring.io/>, dove impostiamo i seguenti dati:



In particolare, sono fondamentali le dipendenze:

* Vaadin (ovviamente)
* DevTools (per lavorare più comodamente)

Generiamo il progetto e scarichiamo lo zip, che scompattiamo (ad es. con 7zip: menu contestuale / 7-Zip / Estrai qui)

## Sistemiamo i problemi

Come anticipato, il modello generato ha attualmente alcuni piccoli problemi.

Dal nostro IDE (ad es. NetBeans) apriamo il progetto scompattato in precedenza (File / Open project).

Se ci sono problemi, facciamo per prima cosa un Clean & build (Run / Clean and build project): questo ci fa vedere il nodo con i file sorgenti, scarica le librerie richieste ed elimina gli errori di compilazione.

## La prima esecuzione

Mandiamo ora in esecuzione il progetto: nelle proprietà non è impostata la classe di partenza (la Main class), ma comunque viene suggerita l'unica classe che ha i requisiti necessari: accettiamo la proposta, selezionando "Remember permanently" in modo da sistemare una volta per tutte le proprietà.

Apriamo un brovser e ci portiqamo su <http://localhost:8080/>

NB: se la porta 8080 è occupata, la modifichiamo intervenendo sul file application.properties (che si trova nel nodo Other sources / src/main/resources / <default package>), terminiamo l'esecuzione corrente e rieseguiamo.

Et voilà! … ma non funziona niente ☹

## Ultimi aggiustamenti

Per prima cosa, la classe principale deve estendere SpringBootServletInitializer:

@SpringBootApplication

public class DemoVaadinApplication extends SpringBootServletInitializer {

E poi dobbiamo ovviamente inserire la classe che realizza la pagina di interfaccia con l'utente, ad es.:

import com.vaadin.flow.component.button.Button;

import com.vaadin.flow.component.notification.Notification;

import com.vaadin.flow.component.orderedlayout.VerticalLayout;

import com.vaadin.flow.router.Route;

@Route

public class MainView extends VerticalLayout {

public MainView() {

Button button = new Button("Saluta",

e -> Notification.show("Ciao a tutti"));

add(button);

}

}

(click destro sul nodo Source packages / edu.alec58.demoVaadin / New / Java class -> MainView)

NB: la route di default (ovvero quella mappata su localhost:8080) si DEVE chiamare MainView.