**Bevezetés a Spring Boot használatába**

**Spring Boot – 1.**

**(Spring Framework sok gépeléssel járó alkalmazás konfigurálási műveleteinek automatikus elvégzésére ->beanek testresazabása, összekapcsolása, 3rd partyk integrálása); Autoconfiguration = conveinton over configuration; classpathon lévő 3rd partyk automaitkus felkonfigja intelligens(leggyakrabban használt) alapértékekkel; self contained alkalmazások-> tartalmazza a futáshoz szükséges konténert is; JVMet tartalmazó sajátgépen jar állomány futtatása; klasszikus Springes konfig, ha el akarunk térni az intelligens alapértékektől)**

A Spring Framework régóta létező keretrendszer, a **Spring Boot** sokkal **fiatalabb;** a **Frameworkre épül** rá és annak **a használatát könnyíti** meg

azért hozták létre, mert a Frameworkben egy alkalmazás **konfigurálása**, a **beaneknek testreszabása** és **összekapcsolása**, és a 3rd party **libraryk** **integrálása** sok munkával, gépeléssel konfigurációval járt -> a boot ezeket **automatikusan elvégzi**

kitalálták az **AutoConfigurationt** - > a Spring Boot megvizsgálja, **milyen** 3rd party **libraryk** vannak a **classpathon**, **és automaitkusan felkonfigurálja** ezeket ún. **intelligens alapértékekkel** (**convention over configuraiton**; ahogy általában a leggyakrabban szokták, úgy integrálja)

**Self contained alkalmazásokat** hoz létre; be lehet úgy állítani (ez a default mód is), hogy maga **az alkalmazás tartalmazza a futáshoz szükséges konténert** is; pl. ha Tomcatet használunk(ez a default), **nem kell nekünk külön telepíteni egy Tomcatet** és arra feltelepíteni a projektünk war állományát, hanem valójában **egy jar állomány fog létrejönni, amibe be van** **csomagolva** a Tomcat is; **ezt a jar állományt el lehet indítani** úgy, hogy az adott **gépen kizárólag egy java virtuális gép van telepítve**

Ha az i**ntelligens alapértékektől el szeretnénk térni,** megtehetjük; ilyenkor a **klasszikus Springes konfigurációt** kell használnunk

**Spring Boot – 2.**

**(nagyvállalati üzemeltethetőség -> monitorozás és beavatkozás Actuatorokkal; start.spring.io-> project template zipként; starter projektek=moduláris, előre beállított verzió kompatibilitás; klasszikus Maven struktúra; applicaiton.resources állomány, statics, templates, Applicaiton.Java indításra és Tests.java)**

további előnyei a **nagyvállalati üzemeltethetőség** -> hogy egy **üzemeltető infókat tudjon kinyerni** az alkalmazásból és akár **be is tudjon avatkozni** (=tudja monitorozni), erre a Bootban bevezetett eszközök az ún. **Actuatorok**; ezek **végpontok**, akár http végpontok, amelyeket **REST webszolgáltatásként meg lehet hívni**, de a Boot **támogatja** az **IMX**-et is, ami egy **szabvány** tipikusan a **Java alkalmazások üzemeltetésére**

része a Boot-nak továbbá, hogy létrehoztak egy **start.spring.io weboldalt** -> itt le **lehet generálni egy indító alkalmazás sablont** (verziók, 3rd party libraryk, adatbáziskezelő, webszolgáltatás kezelési eszközök kiválasztása -> legenerál egy **zip állományt**, ami egy **project template**, ki kell csomagolni és megnyitni a fejlesztő eszközben és annyi; maven esetén pom.xml is előre le lesz generálva)

**tartalmaz Starter projekteket;** a Boot önmagában **moduláris**, nekünk csak **azt kell felvenni függőségként** a pomba, amire nekünk **szükségünk van** -> nem nekünk kell kitalálni, hogy mely verziók **kompatibilisek**, **melyik verziókat kell használni**, ezek **előre beállítottak és auto-teszteltek** -> ezért **különösen alkalmasak kis microservice alkalmazások fejlesztésére**, mert egy **különálló, önmagában futtatható állományt** kapunk, ami mindent tartalmaz a működéshez és ráadásul ez **könnyen telepíthető** és **üzemeltethető** is

ha a start.spring.io által legenerált zip állományt megnézzük -**> klasszikus maven struktúra;** resourcesben **kiemelt fontosságú állomány** az **applicaiton.properties**, a bootal kapcsolatosan **minden itt van konfigurálva; statics könyvtár**:webes alkalmazások esetén ide kerülnek a nem módosuló **statikus html oldalak,** **css** és **javascript** fileok; **templates könyvtár**: ha **server oldali html generálást** akarunk, az adott **template** ehhez ide kerül; két előre legenerált osztály: **Apllicaiton.java** -> ezzel lehet **elindítani az alkalmazást** **Tests.java** -> egy **üres tesztesetet** tartalmaz, ami a **Spring jó konfigurálását és elindulását vizsgálja**

**pom.xml**

**(parent project , webes alkalmazás esetén default konténer=Tomcat; függőségek verziószámaival; Jackson(json és java beans megfeleltetés);Hibernate Validator, boot-starter-test, Mockito, AssertJ, jSONAssert)**

a file egy **másik projektnek lesz a leszármazottja** -> egy **parent project** lesz definiálva, ahol fel vannak véve a **függőségek verziószámai** (ezért nem nekünk kell azt definiálni) = org.springframework.boot::spring-boot-starter-parent

**webes alkalmazás** kiválasztása **esetén** felveszi függőségként a **default konténert a Tomcatet,** és felveszi a **Jacksont**, ami a **Java objektumok** és a **json dokumentumok** közötti **megfeleltetésért** feleős, és felveszi a **Hibernate Validatort** is, amivel a bean validaiton API-t tudjuk használni, a **beérkező adatok validálására,** ellenőrzésére tudjuk ezt használni

**Teszthez szükség van a boot starter-test moduljára**; ez sok függőséget felvesz; ma már a jUnit5öt használja; mockolásra a Mockitot; assertelésre a Hamcrestet és az Assertjt(ezt használjuk!); XML összrhasonlításra XMLUnit, de elterjettebb formátum REST esetén a jSON, ehhez az összehasonlításra JSONassert, és json kifejezéske írására a JsonPath

**Bevezetés a Spring Boot használatába – gyakorlat**

Spring Bootal HelloWorld alalmazás összeállítása

start.spring.io

beállítások:

1.Maven Project és Java

2. 2.3.3.as verzió kiválasztása (jelenleg ez a legfrissebb stabil verzió)

3. Project Metadata

group id: microservices.training

artifact: empoyees

name ua.

package: employees

Jar X, Java 14 X

4.Add dependencies -> WEB -> Spring WEB (spring mvc)

5.Generate -> zip állomány, vázzal

6.sajátgépes kitömörítés

7.pom .xmlen keresztül projekt megnyitása IDEAban

8. pom.xml megnéz; parent project és stb check; projecten belül External libraries lenyitása -> rengeteg sor; EmployeesApllicaiton osztály mainel, üres applicaitonproperties állomány, üres teszt eset

9. HelloController osztály létrehozása

@**Controller** annotáció rája

metódus írása, ami a bejövő http kérésekre válaszol:

public String sayHello(){

return „Helo Spring Boot” + LocalDateTme.now();

}

**@GetMapping(”/”)** annotáció rája -> get kérésre fog a metódus válaszolni, ha / címre érkezik be (főoldal)

**@ResponseBody** annotáció rája -> amivel visszatérünk, az a responseban jelenjen meg -> **Kiváltható** azzal, ha eredetileg a Controller annotáció helyett **@RestController** annotációt rakok rája (A Spring ilyenkor úgy kezeli, mintha minden metóduson rajta lenne a ResponseBodya nno)

10.main futtatása

a háttérben megindul a build, a compile, majd elindításra kerül az alkalmazás; elindult a beépített Tomcat container, települ rá az alkalmazás és hallgatózik a 8080as porton; ha átkapcsolok böngészőre és meghívom a localhost:8080-as címet, látom az üzit, frissítésre frissül a dátum (dinamikuan kerül visszaadásra amikor beérkezik a http request)

->

nem kellett a frameworkot konfigurálni, pom automatikusan létrejött, EmployeesApplicaitont is létrehozta, nekünk csak a Springes beanünket kellett implementálni, minden mást levett a vállunkról

**Kérdések**

*Mi a Spring Framework és a Spring Boot kapcsolata? Miért hozták létre?*

*Mi az az AutoConfiguration és az intlelinges alapértékek?*

*Mi az a sel contained alkalmazás? Hogy jön létre ilyenkor a project és hogy indítjuk el?*

*Mit kell használni akkor, ha el akarunk térni az intellinges alapértékektől?*

*Hogy szolgálja a nagyvállalai üzemeltethetőségek az ún. Actuatorokkal?*

*mire jó a sstart.spring.io weboldal és miket tartalmaz alapba?*

*Hogy néz ki egy Springbootos project struktúrája?*

*Minek a leszármazottja lesza projektünk és hogy lesz ez definiálna?*

*Miket vesz fel a project webes alkalmazás esetén még függőségként?*

*Milyen startert vesz fel a testekhez és ez miket ránt be magával?*