**Unit és integrációs tesztek**

**Unit tesztelés**

a SF különösen fel van készítve tesztesetek implementálására

**a jUnit5, az AssertJ, a Hamcrest alapba a classpathra** kerül és ezen eszközök használata egyből biztosított

**amikor unit teszteket írunk, nem indítjuk be a Springes dolgokat,** az Application Contextet, mert ott már valami **integráció történne,** több bean indulna el stb.

uni teszt az, amikor csak egy java osztályt tesztelünk le

példa: EmployeesService sayHello() metódustá akarjuk tesztelni



**a Springet tudjuk non invasive módon használni,** így adott, hogy **a komponenseket tudjuk unit tesztelni** is

**Unit tesztelés függőséggel**

Hogy tesztelnénk le egy controllert?

ennek már van egy külső hivatkozása (XXXService)

**Springes beavatkozás nélküli teszteléshez az employeesServicet kicseréljük egy test doublere, egy mock objektumra + annotációk**

****

**Integrációs tesztek**

**Ebben az esetben már elindul a spring application contextje**; ő maga fogja létrehozni a beaneket, a köztük lévő kapcsolatok kialakítását=injektálást

az **adott teszt osztályt a @SpringBootTest annotációval** kell ellátni, ez tartalmazza **@Extendwith(SpringExtension.class**) annotációt -> azt mondja meg a jUnit5nek, hogy a teszt esetet SpringExtensionnel kell lefuttatni

**a jUnit 5ben az Extensionok valók arra, hogy a teszt esetek előtt és után valamiféle kódrészletek lefussanak;** jelen esetben beindítja a springet, az applicaiton contextet, ami majd le fogja példányosítani a beaneket

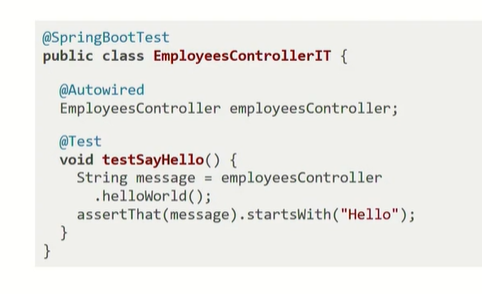
a Spring a különböző tesztesetek közötti cache-eli az application contextet, **vagyis nem indítja be mindannyiszor az AC-t** (és ezzel lassítva a tesztek futtatását), hanem **csak az első eset lefutása előtt, az utolsó után pedig leállítja, az összes teszt pedig ugyanabban az contextben fog lefutni**; **vigyázni kell,** hogy ne legyen ezért az AC-nek **állapotváltozása**; van arra annotáció, a teszt ha módosítja az AC állapotát, a futás során épüljön újra az AC, de ezt lehetőleg ne használjuk, mert nagyon lassít

a **beanek injektálhatóak a tesztbe** **@Autowired** annotációval

**Integrációs teszt példa**

**A tesztosztály neve ha IT végződést kap,** akkor a Mavennek nem a surefire pluginja, hanem a **failsave pluginja fogja elindítani**, tehát külön tudjuk választani, hogy a MAvennek melyik fázisában melyik tesztesetek fussanak le(unitoknál a unit tesztek, integrációsnál az integrációsak)

Az **@Autowired annotáció hatására injektálásra kerül az EmployeesController,** amit már a metódus hívása előtt az AC lepéldányosít a Serviccel együtt; itt nincs hivatkozás a servicere, mégis mindkettő példányosításra kerül a háttérben



**Unit és integrációs tesztek – gyakorlat**

**0.Alapba a Springes projekt tartalmazni fog egy ..Tests osztályt, ami egy olyan osztály, amin rajta van a @SpringBootTest annotáció s egy üres metódust tartalmaz; ezt is el lehet indítani, ez beindítja a SpringBootot és amúgy más nem történik; érdemes egy ilyen esetet bent tartani a projektben mert ezzel vizsgálható hogy a Spring megfelleően van-e felkonfigolva**

*Unit tesztek írása*

1.HelloService teszt

nincs függősége, nem kell hozzá semmi springes komponens

Alt Enter – Generate Test – és bejelölöm a metódust is!

externalok között a classpathon ott vannak az assertes eszközök

maga a teszt: aassertthat és sstartswith

2. HelloControllerTest

@Extendwith(Mock..) és @Mock, @InjectMocks

when()..thenReturn()

assertThat(message).isEqualTo..

*Integrációs tesztek írása*

*3.*HelloIT osztály a tesztosztályok között

@SpringBootTest anno rája, @autowired és HelloController atri

@Test sayHello, uaz mint előbb és assertThat()

->a valós osztályok együttműködését tesztelem

**Kérdések**

*Milyen teszt eszközök használata biztosított spinrgbootos project esetén?*

*Meddig unit teszt a unit teszt és már mior integrációs a teszt?*

*Hogy unit tesztelünk olyan osztályt, aminek van függősége?*

*Mi az a mock objektum? Hogy konfiguráljuk?*

*Mi fog történni @SpringBootTest használata esetén? Az ilyen teszt már minek minősül?*

*Mire jók jUnit5ben az extensionök?*

*Spring beindítása esetén a tesztesekre vonatkoztatva hogyna működik az AC? Mire kell figyelni?*

*Hogy injektálhatóak a beanek a tesztosztályba @SpringBoottest használata esetén?*

*Hogyan jelöljük az integráiós teszteket tartalmazó tesztoszátlyokat? Mi fut le ilyenkor Mavenben?*

*Külön tudjuk-e választani a unit és az integrációs tesztek futását?*