



M Ű E G Y E T E M 1 7 8 2

Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék
Hibatűrő Rendszerek Kutatócsoport

SZOLGÁLTATÁSBIZTONSÁGRA TERVEZÉS LABORATÓRIUM
2011/2012 ŐSZI FÉLÉV

Automatikus tesztfuttatás

Mérési feladatok
(v1.0)

Készítette: Oláh János
(janos.olah@mit.bme.hu)

November 21, 2011

A MÉRÉS SORÁN ELVÉGZENDŐ FELADATOK

1. feladat: Egyszerű Selenium szkript rögzítése

Miután elindította a virtuális gépet (*meres/LaborImage*), győződjön meg arról, hogy fut a dotCMS tartalomkezelő. (A FireFox böngésző könyvjelzői között megtalálható az *url*.) Lépjen be az admin felületen (*admin@dotcms.com/admin*), röviden vizsgálja át az adminisztrációs lehetőségeket. A felső sorban egy külön lapon található az IRC kliens, melyet a mérésen tesztelni fogunk.

Miután végiggondolta, hogy hogyan tudna ehhez olyan teszteseteket készíteni, melyekkel le lehet fedni a teljes funkcionalitást, indítsa el a Selenium IDE-t az Eszközök menüből.

Az IDE segítségével hozzon létre néhány (1-3) tesztesetet, és próbálja őket lefuttatni. Próbáljon ki több változatot is, ismerkedjen meg az IDE-vel és a konfigurációs lehetőségeivel.

Az elkészült teszteseteket és a tapasztalatait dokumentálja a jegyzőkönyvbe!

2. feladat: Szkript exportálása és futtatása JUnit tesztként

A Selenium IDE ugyan nagyszerű eszköz, de nem éppen a legcélszerűbb komolyabb tesztelési projektek esetén. Ezért szeretnénk a teszteseteket könnyebben menedzselhetővé illetve végrehajthatóvá tenni.

Vizsgálja meg az exportálási lehetőségeket, majd exportálja az előző feladatban elkészült teszteseteket JUnit Java fájlként. Mivel mi a következőkben a Selenium WebDriver-t fogjuk használni, válassza ezt a lehetőséget.

Készítsen egy új Eclipse projektet, és hozzon létre benne egy üres Java projektet. Külső függőségként adja hozzá a projekthez a JUnit és a Selenium jar fájlokat (*/home/meres/Dep* könyvtár elemeit). Ehhez a projekthez adja hozzá az exportált teszteseteket.

Ezeket még szükséges kissé kiegészíteni, mivel a tesztelendő erőforrás nem érhető el az Eclipse workspace-ben. Épp ezért adjunk egy *main* függvényt a JUnit fájlhoz, így már lefuttathatóvá válnak.

Futtassa le a teszteseteket néhány alkalommal, próbálja módosítani a forrásfájlokat a WebDriver API segítségével. Például vezesse be változók használatát, próbáljon ki egyéb elérési utakat is (objektum név, XPath kifejezések, stb.).

Az elkészült tesztesetek érdekesnek ítélt részeit illetve tapasztalatait, valamint a tesztfut-tatás eredményét dokumentálja a jegyzőkönyvbe!

3. feladat: Maven projekt létrehozása

Bár az elkészült teszteseteket így már sokkal könnyebb kezelni, tovább egyszerűsíthetünk a saját dolgunkon, ha bevezetjük az egységes projektmenedzsmen-tet, melyet a Maven biztosít. Így például nem lesz gondunk a függőségekre, hiszen biztos, hogy minden projekt ugyanazokat használja, vagy nem lesz gond az sem, ha eltérő eszközöket használnak a fejlesztői csapaton belül, stb.

Hozzunk létre tehát egy új Maven projektet az Eclipsen belül. A varázsló első két oldalán fogadjuk el az alapértelmezett értékeket (de azért megvizsgálhatjuk, milyen sokféle előredefiniált projektfelépítés közül választhatnánk), a harmadik oldalon pedig adjunk nevet a projektünknek. Ezzel máris létrejött a projekt.

Vizsgáljuk meg az új Maven projekt felépítését, és nézzük meg a *pom.xml*-t is. Látható, hogy bekerült függőségként a JUnit, de például a Selenium nem. Ráadásul a JUnit-ból is van

már frissebb verzió, így az XML fájl függőségek részét írjuk felül az alábbi kódrésszel:

```
<dependencies>
  <dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>2.12.0</version>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.10</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

Ezután már nincs más hátra, mint a tesztek közé a megfelelő helyre átmásolni a korábban már JUnitként lefuttatott teszteseteket. (Itt figyeljünk arra, hogy a JUnit által ajánlott nevezési konvenciót a Maven ki is kényszeríti, vagyis csak akkor fog lefutni egy teszteset, ha **“Test” szóra végződik az osztály neve!**)

A Maven futtatásához pedig válasszuk a “Run as/Maven install” parancsot. Ez több célt is magába foglal, így a tesztelés is le fog futni.

Az elkészült teszteseteket illetve a Maven projekt készítése során szerzett tapasztalatait, valamint a futtatás eredményét dokumentálja a jegyzőkönyvbe!

4. feladat: Folytonos integrációt biztosító szerver konfigurálása

A teljes automatizálástól így már csak egy lépés választ el minket, a folytonos integrációt támogató szerver konfigurációja. A böngészőben menjen a Jenkins oldalára, és ismerkedjen meg a képernyőjével. Nézze meg a már létező Jenkins jobot, mely az IRC plugin telepítését végzi el a dotCMS-re.

Futtassa le a jobot a Jenkins felületéről, és nézze meg mi történik. A job úgy van konfigurálva, hogy SVN változás esetén elindul egy build. Módosítson egy sort az “ircplugin” Eclipse projektben, és hajtson végre egy commit műveletet. Nézze a Jenkins felületén, hogy mi történik! (Például módosítsa a *static_velocity/portlet/irc.vm* fájlt. Ez a fájl valósítja meg tulajdonképpen az IRC plugin.) Próbálja ki, hogy mi történik ha leállítja a dotCMS szerverét (*sudo sh /home/meres/dotcms/bin/shutdown.sh*), és úgy futtat egy buildet! (Ezután ne felejtse el elindítani!)

A feladat ez után, hogy létrehozzon egy új Jenkins jobot, mely a tesztek lefuttatását fogja végrehajtani. Ehhez először ossza meg a korábban létrehozott Maven projektet SVN-en keresztül (*/home/meres/SVN/my_selenium_project*), melyet a Jenkinsnek a *file:///home/meres/SVN/trunk/my_selenium_project* hivatkozással lehet majd átadni.

Hozza létre az új jobot. Konfigurálja úgy, hogy a “dotCMSPluginDeploy” job lefutása után azonnal fusson le ez is! Állítsa be az SVN figyelést is! Próbálja ki, mit történik, ha elrontja az egyik tesztesetet, és azt próbálja futtatni a Jenkins!

Az feladat során szerzett tapasztalatait, valamint a Jenkins job lefutásának eredményét dokumentálja a jegyzőkönyvbe!