Zope-termékek

Reuven folytatja a Zope felfedezését. E hónapban megtudhatjuk, hogy a Zope-termékek miképpen teszik lehetővé a szolgáltatások újrahasznosítását.

Múlt hónapban egy pillantást vetettünk a nyílt forráskódú Zope alkalmazáskiszolgálóra. Pontosabban láthattuk, hogyan használhatjuk a Zope DTML (Dynamic Template Markup Language) tagjait egyszerű dinamikus oldalak előállítására, illetve hogyan kezelhetjük weblapunkat mindössze egyetlen böngésző segítségével.
Csakhogy aki már dolgozott a DTML-lel, tudja, hogy hamar elveszti a varázsát, amint valami összetettebbet próbálunk meg vele alkotni. A DTML akkor a leghatékonyabb, ha csak mértékkel használjuk, vagy ahol használata magától értetődik; ha olyan DTML-oldalakat készítünk, ahol fél tucat egymásba ágyazott feltételes (`<dtml-if>`) tagot kell alkalmaznunk, oldalunk hamar olvashatatlanná és nehezen karbantarthatóvá válik, nem beszélve arról, hogy nemigen lesz modulárisnak nevezhető.
A DTML másik nagy gondja, hogy központi elhelyezés helyett különálló dokumentumokban tárolódik. Így ha több helyen is újra akarunk egy szolgáltatást hasznosítani, a DTML-eljárásokat és dokumentumokat le kell másolnunk. Ez azt jelenti, hogyha új szolgáltatást akarunk bevinni avagy egy régit megváltoztatni, akkor az összes másolaton végig kell mennünk és el kell végeznünk a szükséges módosításokat.
A nehézségre a Zope-termék jelenti a megoldást. Minden Zope-termék tulajdonképpen egy objektumosztály (vagy osztálykészlet), amelyet a weblapunkon akárhány példányban létrehozhatunk.
Ebben a hónapban a Zope rugalmasságának magvát alkotó Zope-termékkel fogunk foglalkozni. A telepítést követően néhány meglévő terméket próbálunk ki, majd Pythonban a saját termékünket is elkészítjük.

Mire képes a termék?

A Zope-termék tulajdonképpen egy kódból, grafikából és DTML-ből álló csomag, amely egy adott újrahasznosítható szolgáltatást tesz elérhetővé. Ha például egy olyan egyszerű lapot szeretnénk készíteni, amely kiírja a pillanatnyi időt, a következő DHTML-dokumentumot kell létrehoznunk:

```
<p>A pontos idı: <dtml-var name="ZopeTime"
➥fmt="fCommon">.</p>
```

De mi a helyzet, ha lapunkat ki szeretnénk bővíteni, és az időjárás-előrejelzést is meg akarjuk jeleníteni, amit egy másik kiszolgálóról töltünk le HTTP-n keresztül? A DTML nem jó megoldás; még ha segítségével el is tudnánk készíteni a saját szolgáltatásunkat, az eredményt biztos, hogy nehéz lenne kezelni, és legalább ilyen ronda ügy lenne megírni is. Minthogy azonban a Zope-termékek Pythonban készültek, tetszés szerinti Python-modult használhatnak, és kimenetüket HTML-ben vagy bármilyen együttműködő formátumban jelentethetik meg.
Mivel minden termék önálló egésznek tekinthető, egységenként feltelepíthetjük vagy eltávolíthatjuk őket, akkor is, ha számos osztályt használnak.
Természetesen ez nem azt jelenti, hogy minden termék önmagában működik; hiszen az egyik termék nyugodtan használhatja a másik által nyújtott szolgáltatásokat.
A termékek és a DTML-dokumentumok mellett a Zope további két lehetőséget nyújt a dinamikus tartalom megjelenítésére: a Python-parancsfájlok (amelyeket szintén egy termék valósít meg) lehetővé teszik, hogy kisméretű Python-programokat írjunk és használjunk a Zope-ban. A másik lehetőséget választva a termékeket a ZClasses nevű rendszeren keresztül is létrehozhatjuk, szerkeszthetjük és használhatjuk. A ZClasses lehetővé teszi, hogy az új termékeket (és a hozzájuk tartozó osztályokat) mindössze a böngésző és a DTML segítségével rakjuk össze.
E négy lehetőség igen nagy rugalmasságot biztosít, de a kezdő Perl-programozóknak néha nehéz eldöntetni, melyiket is válasszák. Beehive ***The Book of Zope*** című művében azt ajánlja, hogy a projekthez eleinte ZClassest használjunk, majd mikor már mindenki megegyezett a tervben, a kódot a teljes értékű termékek szintjére tegyük át. Minél összetettebb feladatokat oldunk meg, annál valószínűbb, hogy a DTML és Python-parancsfájlok helyett a Zope-termékeket választjuk.

A termékek kezelése

A Zope-termékek csaknem minden szempontból jól kezelhetők a ***Zope intézőben*** (management screen), amit Zope-kiszolgálónkon a ***/manage*** URL alatt érhetünk el. A termék intézőjének elővarázslásához a bal oldali keretben kattintsunk a ***control panel*** hivatkozásra, majd az így feljövő ***Zope vezérlőpult*** fő keretében válasszuk a ***Product Management*** (termékkezelés) hivatkozást. Itt a Zope-termékek listáját találjuk, valamint a képernyő tetején egy ***Add product*** (termék felvitele) gombot. A Weben keresztül is szerkeszthető termékeket (ilyen például a ZClasses, amelyet az előbb ismertettünk röviden) nyitott doboz jelöli, a beépített Zope-termékeket pedig zárt. A zárt doboz egyszerűen annyit jelent, hogy a terméket magát a Weben keresztül nem módosíthatjuk. Természetesen a legtöbb termék megengedi, hogy webes felületen keresztül egy vagy több tulajdonságát testreszabjuk, de a termék mindig változatlan marad, hacsak a forráskódot meg nem változtatjuk.
Minden termék valójában a Zope telepítési könyvtára alól nyíló ***lib/python/Products***-ban található alkönyvtár. A Sessions termék a ***lib/python/Products/Sessions***, a Transience termék pedig a ***lib/python/Products/Transience*** könyvtárban helyezkedik el (rendszeremen a ***/usr/local/zope/*** könyvtárba helyeztem a Zope-ot, így nálam a Sessions a ***/usr/local/zope/lib/python/Products/Sessions/*** könyvtárban helyezkedik el). A termékek könyvtára Python-kódot, szövegfájlokat és könyvtárakat tartalmaz:

- ***__init__.py***: ez az, amit a Zope modulbetöltéskor megkeres és végrehajt. Az ***__init__.py*** alapérték beállító eljárása (initialize method) többek között meghívja a

`context.registerClass` osztályt, amely (mint neve is sugallja) tudatja a Zope-pal, hogy termékünk egyáltalán létezik. Ismerteti, milyen szöveget kell majd a */manage* képernyőhozzáadás menüjében megjeleníteni (ezt a `meta_type` kapcsolóval oldja meg), és hogyan készíthetünk a termékből egy új példányt, ha az ***Add*** gombot megnyomják (ez a `constructors` kapcsolóval adható meg).

- ***README.txt***: miként neve is mutatja, ez a termékhez tartozó README (OLVASSEL) állomány. A ***Vezérlőpulton*** a termék nevére kattintva többek közt egy README táblát is felhoz. Itt a fájlrendszerben való keresgélés nélkül olvasgathatjuk a ***README.txt*** tartalmát. Ha a termék könyvtára nem tartalmaz ***README.txt*** nevű fájlt, a README tábla sem jelenik meg a képernyő tetején.
- ***version.txt***: ez a fájl tartalmazza a termék kötőjelekkel (-) elválasztott pillanatnyi változatszámát. A Foo termék 1.2.3-as változata például a következő tartalmú ***version.txt*** fájllal rendelkezik: Foo-1-2-3. A vezérlőpult ezt az adatot fogja megjeleníteni.
- ***Help*** fájlok: a termék egy help (súgó) könyvtárat tartalmazhat, ahol azok a fájlok találhatók, amelyek tartalmát a ***help*** hivatkozásra kattintva a Zope megjelenítheti. A Súgófájlok gyakran strukturált szöveget tartalmaznak, amely a Perl POD dokumentumrendszeréhez hasonló szellemiségű, egyszerűségre törekvő formázási rendszer. Strukturált szöveg egyszerű szövegszerkesztővel is könnyen elkészíthető, és ugyanilyen könnyen olvasható a `less`-hez hasonló szabványos Linux-eszközökkel.

A Zope a pillanatnyi termékek listáját csak indításkor nézi végig. Ez azt jelenti, hogyha új terméket telepítettünk, a vezérlőpultról a Zope-kiszolgálót újra kell indítanunk.

A termékek telepítése

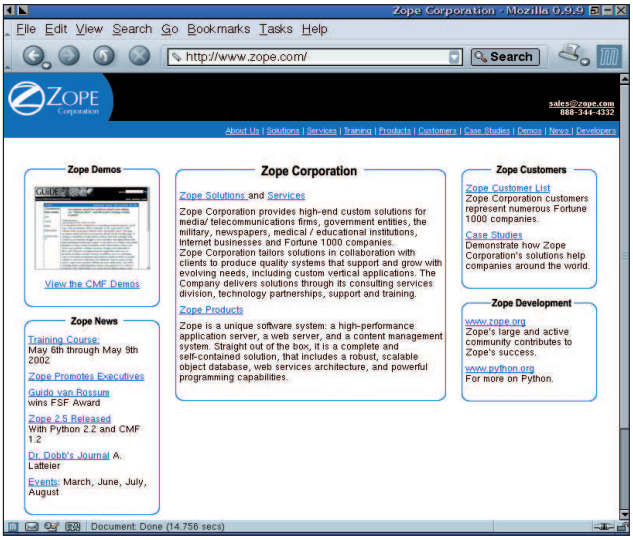
Most hogy láttuk, mit tartalmazhat egy átlagos termék, ideje feltelepítenünk egyet! A Zope honlapjáról töltsük le, a ***lib/python/Products*** könyvtárba csomagoljuk ki, majd a Zope-ot indítsuk újra. Ha minden jól ment, frissen telepített termékünk meg kell jelenjen a ***Vezérlőpult*** képernyőjén. Ráadásul a termék új példányait a webszerkezet tetszőleges helyén máris létrehozhatjuk. Példaképp a Zope Squishdot termékével készítsünk egy Slashdot-másolatot. Első lépés a Squishdot-másolat letöltése a ➲ http://www.zope.org/Products címről. A Squishdot-csomagot több társával egyetemben a Feedback csoport alatt találjuk, és valószínűleg valahol az elsők közt kell keresnünk a termékek listájában. Kattintsunk a hivatkozásra, ami a Squishdot letölthető változatához visz bennünket – e sorok írásakor a legújabb változat az 1.3.0-s. Figyeljük meg, hogy egy viszonylag összetett termék is milyen kis méretű; a Squishdot általam letöltött változata ugyanis alig volt több, mint 256 KB.
A Squishdotot telepítéséhez ki kell csomagolnunk a ***lib/python/Products*** könyvtárba. Feltételezve, hogy az újonnan letöltött fájlokat a */downloads* könyvtárba tesszük, a Squishdotot a következőképpen csomagolhatjuk ki:

```
# Itt Æll tsuk be sajÆt Zope k nyvtÆrunkat
export ZOPE=/usr/local/zope

# vÆltsunk a termØkk nyvtÆrba
cd $ZOPE/lib/python/Products

# Squishdot kicsomagolÆsa az aktuÆlis
# k nyvtÆrba
tar -zxvf /downloads/Squishdot-1-3-0.tar.gz
```

A régebbi Zope-termékeket nem a ***lib/python/Products***-ban állva, hanem a Zope saját könyvtárból kell kicsomagolni. Úgy tűnik, sajnos nemigen van olyan módszer, amivel kicsomagolás nélkül egyértelműen el lehetne dönteni, miként csomagolták össze a terméket, így nem marad más hátra, mint a kicsomagolás:

```
tar -ztvf /downloads/ProductName.tar.gz
```

Ha a fájlnevek ***lib/python/Products*** útvonallal kezdődnek, akkor kicsomagolás előtt a ***$ZOPE/lib/python/Products*** helyett a ***$ZOPE*** könyvtárba kell váltanunk.
A Squishdot telepítéséhez egyetlen dolgot kell csak megtennünk: csomagoljuk ki a terméket. Mivel azonban a Zope csak induláskor nézi meg a termékeket, a kiszolgálót még az előtt újra kell indítanunk, mielőtt a rendszeren Squishdot-példányokat készíthetnénk. Erre a legjobb módszer, ha a ***Vezérlőpulton*** a ***Restart*** (újraindítás) gombra kattintunk. Ne essünk pánikba, ha

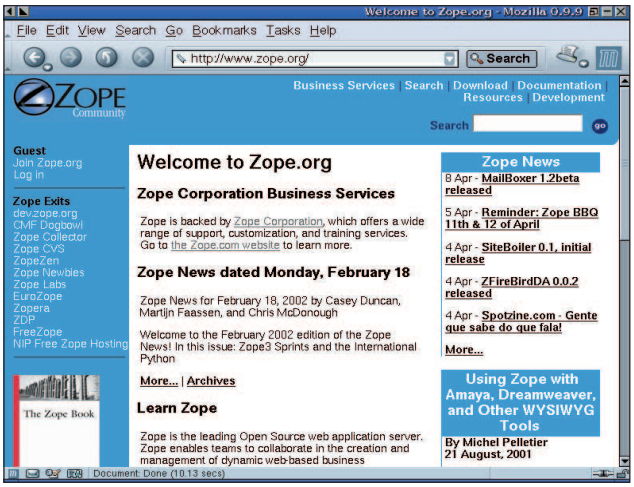


a ***Restartra*** kattintás után a böngésző arra figyelmeztet, hogy a kiszolgáló nem fut, vagy ha valami fura kinézetű Python kivételnyomelemzés (exception backtrace) jelenik meg. Egyszerűen várjunk néhány másodpercig, majd a bal oldali keretben megint kattintsunk a ***Vezérlőpult*** hivatkozásra, és már működnie is kell. Termékünk sikeres felvételét ellenőrizendő a ***Vezérlőpulton*** visszatérhetünk a ***Termékkezelő*** (Product Management) lapra. Amennyiben a frissen feltelepített termék (jelen esetben a Squishdot) nem jelent meg a listában, ellenőrizzük, hogy jól sikerült-e kicsomagolnunk, illetve a jogosultságok lehetővé teszik-e, hogy a Zope-felhasználó elérje a termék állományait.

A termék használata

Ettől kezdve lehetőségünk nyílik rá, hogy a Zope-kiszolgáló saját (/) könyvtárába mozgatása által új Squishdot-honlapot hozzunk létre. A listából ki kell választanunk a Squishdot-lapot, majd kattintsunk az ***Add*** (hozzáadás) gombra. Ez meghívja `context.registerClass` constructors kapcsolójában megnevezett eljárásokat, amelyet a Squishdot ***__init__.py*** programjának beállítófüggvénye indít el.
Máris elkezdtük Squishdot-lapunk létrehozását. Csakhogy a figyelmeztető levelek elküldéséhez a Squishdot a Zope MailHost objektumát is használja (amely az SMTP-kiszolgálónak felel meg). Ha még nem készítettük el és nem határoztuk meg a `MailHost`-ot, a Squishdot beállítóképernyője emlékeztet minket erre.

Amikor a Squishdot a `MailHost` után kutat, a keresést a pillanatnyi könyvtárban kezdi. Ha nem talál `MailHost`-objektumot, a könyvtárfán felfelé folytatja a keresést , amíg el nem éri a /-t, vagy nem talál egy `MailHost`-objektumot. Bár mindez egyszerű dolognak látszik, nagyon jól megvilágítja a Zope alapfilozófiáját jellemző szerzeményezés elgondolást. Egyben azt is jelenti, hogy a különböző Squishdot-honlapok a leveleket különféle SMTP-kiszolgálókon keresztül küldhetik,



egyszerűen úgy, hogy egynél több `MailHost`-objektumot készítünk. Azaz egy teljes körben alapértelmezett `MailHost`-ot adhatunk meg a gyökérben (/), amit aztán az alkönyvtárakba helyezett `MailHost`-objektumokkal szükség szerint felülbírálunk. A szerzeményezéses elgondolás átszövi a Zope-ot, ami abban nyilvánul meg, hogy helyileg szinte bármit megadhatunk és újradefiniálhatunk – a `MailHost`-okat, a felhasználókat, de akár a fejléceket és a stíluslapokat is.
Jelen esetben a `MailHost`-ot a terméklistából kiválasztva, majd az ***Add***-ra kattintva a / könyvtárban egy `MailHost`-példányt hozunk létre. Tekintve, hogy a `MailHost` objektum az SMTP-kiszolgálónak felel meg, ezen objektum beállítása meglehetősen magától értetődő, mindössze arra van szükség, hogy a Zope-kiszolgáló SMTP-kiszolgálóját megadjuk. Minthogy a legtöbb linuxos gép saját levelezőkiszolgálót futtat, a `localhost` valószínűleg megfelelő érték lesz.
A kötelező ID (azonosító) mező célja a `MailHost` egyedi azonosítása a működő könyvtárban, hiszen a Zope az URL-eken belül az objektumok egyértelmű azonosítására ID-ket használ. Miként a fájlnév egyedi azonosító a könyvtárban, a Zope objektum-ID is egyedi azonosító egy könyvtárban vagy egy másik objektumban. A választható ***Title*** (cím) mező inkább az emberek, semmint az alatta fekvő Zope-kiszolgáló kedvéért van itt; ha megadjuk, ez az objektumcím fog a Zope-kiszolgáló csatolófelületén megjelenni.
Miután `MailHost` objektumunkat létrehoztuk, visszatérünk a Zope / gyökérhez tartozó fő kezelőfelületéhez. Láthatjuk, hogy új `MailHost` objektumunk (amit egy kis borítékikon jelképez) az általunk megadott névvel együtt megjelent az objektumok listájában.
Készen állunk, hogy létrehozzuk Squishdot-helyünket. A listából kiválasztva, majd a jobb felső sarokban található ***Add*** gombot lenyomva vegyük fel az új Squishdot-hely objektumot, válasszunk egy ID-t (azaz URL-elérési utat), tetszés szerinti címet, levélgazdát (mailhost), majd Squishdot-lapunkhoz válasszunk ki még pár alapvető kapcsolót. Én például ID-ként az ***atf*** szó mellett döntöttem, az egyéb beállítási lehetőségeket pedig az alapértelmezett értéken hagytam.
Ha Squishdot-lapomat meg szeretném nézni, utasítom a böngészőt, hogy jelenítse meg a ➲ http://localhost:8080/atf/ címet. A Zope észleli a ***/atf*** kérelmet és látja, hogy egy Squishdot-objektumra hivatkozunk. Ezután a Zope megkéri az objektumot, hogy jelenítse meg magát. Nagy valószínűséggel egy bemutatkozó képernyőt fogunk látni, ami erősen hasonlít a Slashdotra, de a Zope hajtja meg.
Annyi Squishdot-lapot készítünk, amennyit csak akarunk, szem előtt tartva, hogy minden honlapnak saját egyedi ID-vel kell rendelkeznie. Így készíthetünk egy moderált lapot, egy moderálatlan lapot és egy másik, belső lapot a szervezet saját használatára – mindegyik saját URL-lel és védett saját felhasználóhalmazzal, valamint csoportokkal rendelkezik.
A Squishdot-honlap (site) módosításához a módosítandó objektum nevéhez egyszerűen csapjuk hozzá a ***/manage*** karaktereket, például: http://localhost:8080/atf/manage. E sor a Squishdot-lapunkhoz rendelt Zope-kezelőfelületet hozza be. A képernyő tetején található táblákat használva csaknem minden olyan kapcsolóját átállíthatjuk, amelynek köze van a Squishdothoz, a moderátorszabályoktól kezdve a honlap nevének színéig.

Összefoglalás

Ebben a hónapban a Zope-termékekkel ismerkedtünk meg: láttuk, hogyan tölthetünk le, telepíthetünk és állíthatunk be új termékeket a rendszerünkön. Bár a termékek természetüktől fogva az egyszerű DTML-lapoknál jóval összetettebbek, központosított kódjuk és nagyobb rugalmasságuk folytán komoly feladatokra sokkal alkalmasabbak, mint a DTML.
A következő hónapban azt vizsgáljuk meg, miképpen írhatunk saját Zope-termékeket Python- és DTML keverék használatával.

Linux Journal 2002. március, 95. szám

Reuven M. Lerner (reuven@lerner.co.il) kisebb webes és internetes módszerekkel foglalkozó tanácsadó cég tulajdonosa és vezetője. A cikk megjelenésének időpontjában valószínűleg már végleg elkészült Core Perl című könyvével, melyet idén jelentet meg a Prentice-Hall. Az ATF honlapon érhető el (➲ http://www.lerner.co.il/atf/).

Kapcsolódó címek

A nyílt forrású Zope-terméket a Zope Corporation (Digital Creations) készíti és végzi a karbantartását. Több adatot a Zope honlapján érhető el ➲ http://www.zope.com

A Zope-közösség fő fejlesztési oldala a ➲ http://www.zope.org. Erről az oldalról letölthetjük a program legújabb változatát, hozzászólhatunk a vitákhoz, letölthetjük a Zope-terméket, tanulhatunk a Zope-ról és részesei lehetünk a Zope-közösségnek.

A Squishdot termék a saját oldalán érhető el ➲ http://www.squisdot.org

Többet tudhatunk meg a Python nyelvről a ➲ http://www.python.org címen.