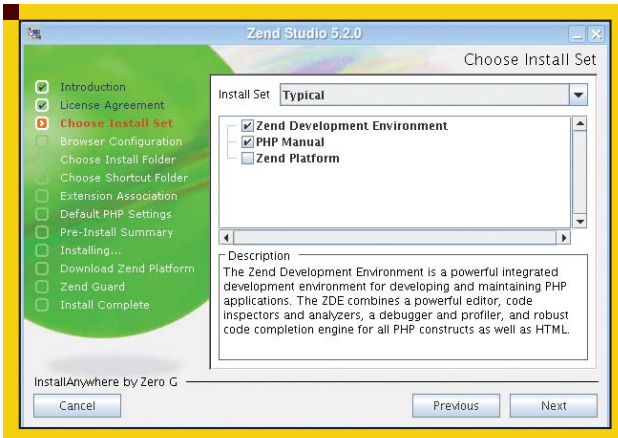
Zend Studio – PHP fejlesztés egységbe zárva

A PHP alkalmazások fejlesztése során számtalan eszközzel találkozhatunk: különféle szövegszerkesztőkkel, verziókövető rendszerekkel, dokumentációkészítővel és még valamilyen hibakereső eszközzel is. Még ennél is több szolgáltatást kapunk egybecsomagolva a Zend Studio Java alapú integrált fejlesztőkörnyezettől.

Az első lépések

A szoftvert a *Zend Technologies* készíti, a kipróbálásra szánt időkorlátos verziója letölthető a ➲ *www.zend.com* weboldalról, ugyanitt megvásárolható a *Professional* változat licence is 299$-ért. A program *Javaban* íródott, így a *Studio Windowson*, *Linuxon* és *Mac OS X*-en képes futni. A linuxos változat egy *tar.gz* archívum formájában érkezett hozzám, melyben egy végrehajtható fájl lapult. A telepítő eltér az átlagos linuxos telepítésektől, egy varázsló vezet végig a telepítés lépésein. Fontos, hogy a *Zend Platformot* is kiválasszuk a telepítendő összetevők közül, mert különben számtalan kiegészítő szolgáltatás nem fog működni. Ez a legnagyobb buktatója a telepítésnek, különben csak végig kell menni a lépéseken. Ahhoz, hogy a *Platformot* is használni tudjuk, kell legyen webszerver a rendszeren. Ha véletlenül hiányozna, ezt pótoljuk az *Apache* feltelepítésével! Miután követve a telepítő utasításait, életre keltettük a rendszert, nézzünk körbe, hogy mit is kaptunk!



1. ábra A telepítés menete

Szövegszerkesztés

A fejlesztési munka egy részében nem történik más, mint a programozó egy szövegszerkesztőbe beírja a program kódját. Egy jól megírt szerkesztő nagyban javítja a hatékonyságot. Vajon a *Zend Studioba* épített szerkesztővel mi a helyzet? Aki az *emacs* vagy a *vi* tradicionális, nagy tudású unixos szövegszerkesztőket mesteri szinten tudja kezelni, annak ez a szerkesztő nem lesz egyáltalán hatékony. Ellenben a kevésbé profikat az alábbiakkal kényezteti el. Kezeli a *PHP* és a *HTML* szintaxist. Ez alapján színezi a forráskódot, amivel sok elírás azonnal felfedezhető. Képes a kód kiegészítésére, azaz a már meglévő függvény és objektumneveket, tulajdonságokat automatikusan felajánl, a meglévő deklarációkat megmutatja, hozza a hozzá írt *DocBlock* dokumentációt is. Ez igaz a beépített és az általunk írt objektumokra és függvényekre is. Gépelés közben szintaktikai ellenőrzést végez, így azonnal jelzi a hibákat. Képes automatikusan tabulálni a beállított kódolási szabály szerint a kódot. Létre tud hozni *DocBlockokat* már meglévő függvény vagy objektum alapján, amit nekünk már csak ki kell egészíteni. Képes a megjegyzésbeli illetve a kódbeli egységeket (kapcsos zárójelekkel elhatárolt részek) kattintásra összecsukni illetve kinyitni utána, így mindig csak az a rész van a szemünk előtt, amivel épp ténylegesen dolgozunk. Képes a forráskódban navigálni, megkeresni objektumok vagy függvények deklarációját. Ezeket egyébként kigyűjti egy listába is. Mi az, amit ezért a tudásért fel kell áldoznunk? Egy már megszokott szerkesztő működését. Talán segítség azért, hogy a gyorsbillentyűk teljesen testre szabhatóak.

Hibakeresés

Nem az a kérdés, hogy fogunk-e hibákat véteni a kódolás során, hanem az, hogy milyen gyorsan fogjuk azokat kijavítani! A program teljes körű és kényelmesen használható hibakeresőt (*debugger*) tartalmaz.
A hibakeresőben pont azok a szolgáltatások érhetőek el, amit bárhol máshol kapnánk. Be lehet állítani töréspontokat, változók figyelését. A futtatást lehet szabályozni: soronként léptethetjük, elindíthatjuk, leállíthatjuk és adott helyre ugorhatunk. A futtatás során megkapjuk a *PHP* által adott figyelmeztetéseket is egy keretben. Ezen túl van *Analyze* menüpont,

ami elég sok programozói tévedést képes kiszűrni. Cserében persze néha olyankor is hiányosságot talál, ahol nincs: nem képes megérteni azért a kód célját, ez a nehéz feladat még mindig a fejlesztő dolga.

Teljesítményhangolás

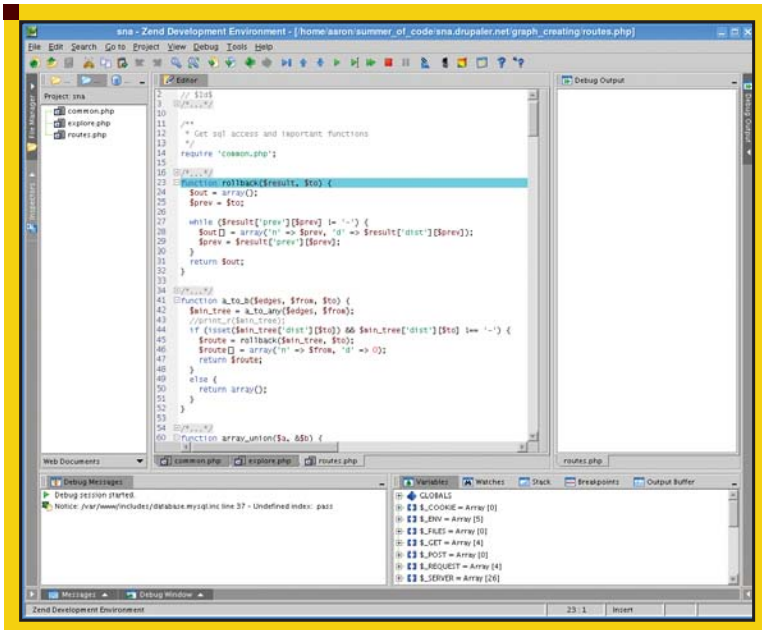
Nem kis feladat egy már meglévő alkalmazást nagyobb sebességre gyorsítani. Nincs rá titkos recept sajnos. Viszont a gyenge pontok megtalálása teljesen automatizálható, így a lényegre tudunk koncentrálni, ami pedig a rosszul teljesítő kódrészlet átírása, nem pedig a mérés maga. Ehhez a funkcióhoz telepíteni kell a *Platformot* is, mert ilyenkor a szkript a webszerveren kell fusson.

A *Tools/Profile URL* menüpontot választva csak egy teendőnk van: megadni, hogy milyen elérési úton található a kérdéses program. Ha csak a helyi gépen fejlesztünk, ehhez a lépéshez elérhetővé kell tenni a webszerveren keresztül is a kódot. Miután a szkript lefutott, kapunk egy új ablakot, amiben mindent megtudunk a program futtatásáról és könnyedén kiszűrhetjük a leglassabb részeket.

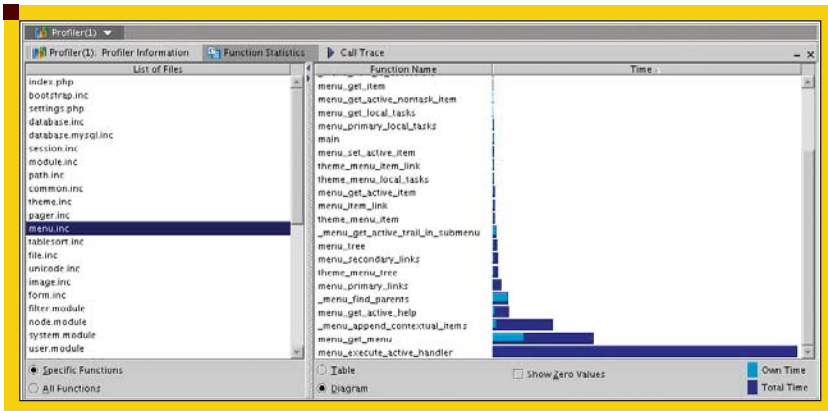
Amennyiben a programunk több fájlból áll, a futtatás után látni fogjuk egy csinos diagramon, hogy melyik fájlban mennyit időzik a végrehajtás. Ezen kívül minden fájlról részletes függvénylistát kapunk, hogy melyik függvény meddig futott és hányszor került meghívásra. Meg van különböztetve az az idő, amit ténylegesen a függvény fogyasztott el, meg az, amit csak a függvény által meghívott újabb függvények igényeltek. Kilistázhatjuk a függvényhívási hierarchiát is. Az oszlopok csökkenő illetve növekvő rendezésével gyorsan kibukik egy rosszul megírt függvény. Jól látszik az is, ha egy függvényt indokolatlanul sokszor hív meg egy hiba folytán a program.

Adatbázis-kezelés

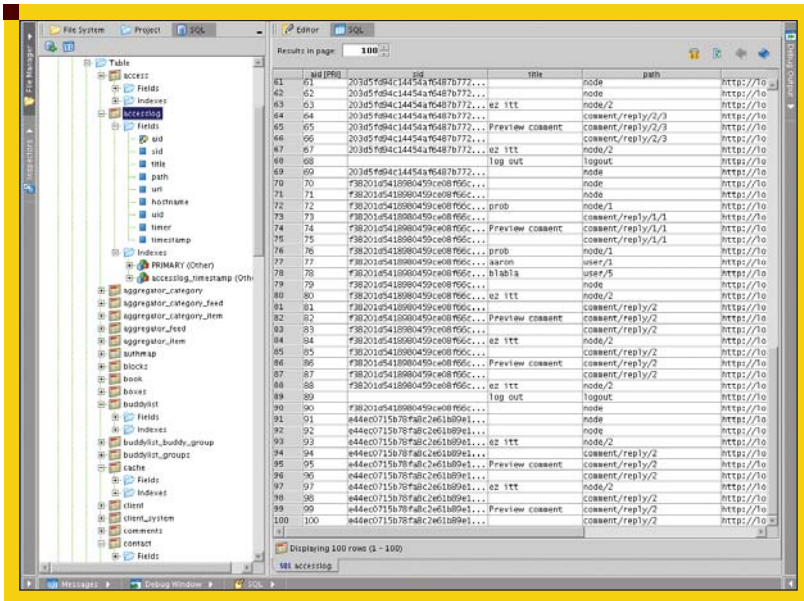
Kevés olyan *PHP* nyelven megírt alkalmazás van, mely ne használná valamelyik *SQL* adatbázismotort. Igaz ugyan, hogy általában ezekhez jár kezelőfelület, biztos ami biztos, a *Studioba* bele van építve egy *SQL* ügyfél, ami számtalan kiszolgálóval (a nevesebb példák: *MySQL*, *MSSQL*, *Oracle*, *PostgreSQL*) képes együttműködni. A kezelőfelületbe jól beépül



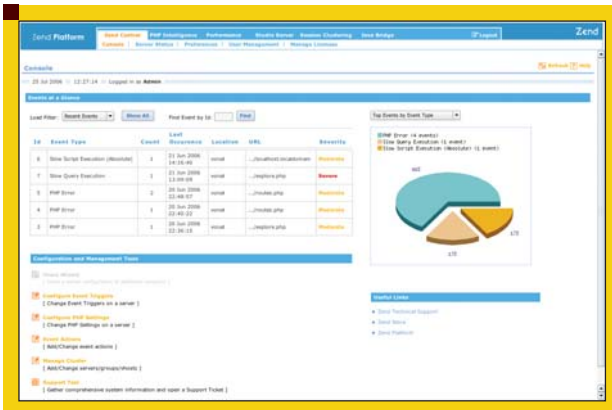
2. ábra Hibakeresés akcióban



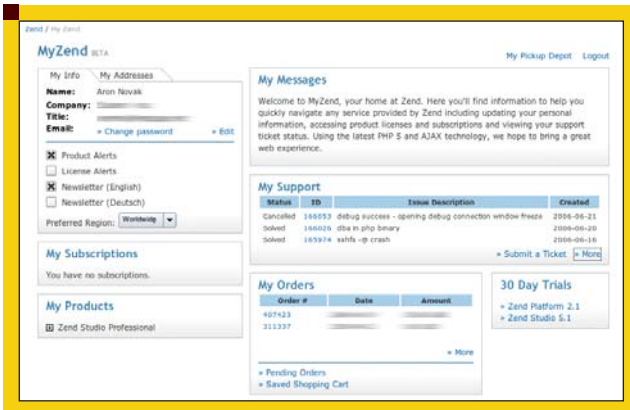
3. ábra Teljesítménymérés



4. ábra Adatbáziskezelés a Studion belül



5. ábra A Zend Platform webes kezelőfelülete



6. ábra A terméktámogatás központja

a kliens, egy mozdulattal válthatunk a programkód és az adatbázis között. A szokásos funkciókat kapjuk: a táblákat listázhatjuk, beszúrhatunk rekordokat, áttervezhetjük a struktúrát. Természetesen a saját kézzel írott *SQL* lekérdezéseinket is futtathatjuk, azok végeredményét éppúgy láthatjuk. Az sem hátrány, hogy ha több adatbázis-kiszolgálót használunk, akkor ezentúl egy egységes felületen keresztül érhetjük el azokat. Persze azt ne feledjük, hogy az egyes termékekre hangolt *SQL* ügyfelek többet nyújtanak általában, mint egy olyan, amelyik több fajtával is együttműködik.

Bűvészkedés a kiszolgálón

Ha a *Platform* telepítését is kértük, akkor a webkiszolgálónk otthont ad a *Zend Platform* kezelőfelületének is, ahol már a *Studio* fejlesztőkörnyezeten túlmutató dolgokat találunk. Magához a *Studiohoz* jár egy fejlesztői *Platform* licenc is, ami lehetővé teszi többek között, hogy a *PHP* programjaink szerveren történő hibakeresésé-hez és teljesítményhangolásához használjuk. De mint láthatjuk, a *Platform* ennél sokkal többre képes. Az *5. ábrán* a nyitóoldalát látjuk a webes kezelőfelületnek, ahol az eseménynapló összegzése található. A *Platformban* van *PHP* gyorsítótár is, aminek segítségével a dinamikus tartalmat az *Apache* gyorsabban képes kiszolgálni. Erről jelentést is kapunk a weboldalon: pontosan látható, hogy mekkora haszonra teszünk szert, hogy a gyorsítótár üzemel. Számszerűen látjuk, hogy mennyivel gyorsabban futnak le a szkriptek. Ha valódi oldalt üzemeltetünk, akkor hasznos, hogy különféle eseményekre figyelést tudunk beállítani. Az előre beállított eseményfigyelésekre példa: lassú szkriptvégrehajtás és futásidejű programhibák. A *Platform* a *Studio* felhasználóinak azért is fontos, mert ezen a felületen lehet beállítani, hogy működjön a távoli hibakeresés. A *Studio Server* menüpont alatt kezelhetjük azon számítógépek listáját, melyek ezt a szervert felhasználhatják hibakeresésre.

Terméktámogatás

Mivel egy kereskedelmi termékről van szó, mi sem természetesebb, hogy jár hozzá terméktámogatás. Ennek minősége épp olyan szempont kell legyen, mint a szoftver szolgáltatásai, hiszen melyik szoftvernél ne futnánk előbb-utóbb olyan szituációba, amit magunk nem tudunk megoldani? Egy zárt forráskódú terméknél még fontosabb a támogatás, hiszen itt a megfelelő hozzáértés birtokában sem tudjuk orvosolni a problémáinkat, ellentétben a nyílt forráskódú eszközökkel. Akadt hiba a program használata közben, nem is kicsi: az Apache webszerver folyamatosan *Segmentation Fault* hibákat dobott bizonyos *PHP* kódok végrehajtásakor. A hibajelentés menete ki van dolgozva: fel kell adni egy hibajegyet, aztán minden ezzel kapcsolatos ügymenet ahhoz tartozik. A *6. ábrán* lehet látni a *Zend* megoldását arra a problémára, amire a nyílt forrás a *BugZillát* készítette. A hibajegy feladása után egy munkanappal ténylegesen érkezik reakció a beadott problémára, kérdésre. Az előbb említett Apache hiba részleteit ugyan több héten keresztül kérdezgették a hibajegyen keresztül, de végül ezt a nagyobb hiányosságot is orvosolták egy új binárissal. A reakcióidő az ígért értéknek megfelelt, hiszen minden új problémára egy napon belül volt válasz. A program árában egy év támogatás van. Egyébként gesztusértékű, hogy az időkorlátos próbaverzióval is adhatunk fel hibajegyet.

Hátulütők

Ezeknek a megoldásoknak a jó része elérhető nyílt forráskódú verzióban is: *PHP* gyorsítótár, hibakereső, teljesítménytesztelő mind létezik szabad licenc alatt is. Amiről nincs tudomásom: hasonló tudással felvértezett grafikus integrált fejlesztőkörnyezet, ahol a kódkiegészítéstől elkezdve egészen a távoli hibakeresésig minden egy fedél alatt megtalálható. Ez a megoldás pontosan ellentétes a unixos filozófiával, miszerint minden program lásson el egy részfeladatot, de azt nagyon jól és a sok kis eszköz együttes használatával bármi megoldható. Kérdés, hogy megéri-e az árát a *Studio*? Próbáljuk ki! Ha kevésbé hatékonyan tudunk benne fejleszteni, mint az eddigi szabad szoftveres megoldásainkkal, akkor csak mentünk egy kört a próbaverzióval.

Novák Áron
(aaron@szentimre.hu)

BME-VIK-es hallgató, műkedvelő rendszergazda. Jelenleg leginkább a NetBeans-szel és mindenféle hordozható eszközzel foglalkozik, legalábbis mindazokkal, amelyeket meg lehet szólaltatni Linux alatt.