Frane Glasinović, univ. bacc. ing. geod. et geoinf. ▸ *diplomski studij, Geodetski fakultet, Sveučilište u Zagrebu, Kačićeva 26, 10000 Zagreb, e-mail: fglasinovic@geof.hr*

PROGRAMIRANJE U AutoCAD-u (C#)

Sažetak

Programski paket AutoCAD ima veliki broj ugrađenih alata pa biste pomislili da u AutoCAD-u postoje funkcije za sve što vam padne na pamet. To je donekle i istina, ali ponekad ćete se naći u situaciji da ugrađeni alati jednostavni nisu ono što tražite. Nekada ćete poželjeti automatizirati ponavljajući proces ili pristupiti rješavanju problema na način koji više odgovara specifičnim okolnostima. U ovakvim slučajevima znanje iz programiranja biti će više nego dobrodošlo. U samo par linija koda riješiti ćete problem za koji ste mislili da ćete potrošiti sate. Ugrađene funkcije AutoCAD-a pomoći će da iskoristite maksimum na vrlo svojstven, specifičan i korisnički prilagođen način.

Ključne riječi
C++
programiranje
AutoCAD Civil 3D 2011

1. UVOD

AutoCAD je od samih početaka pisan na programskom jeziku C++ te se tako razvija i dan danas. Prije 2006. godine modifikacije AutoCAD-a bile su moguće preko automatiziranih funkcija na tri načina: programiranjem u ObjectARX-u (C++), u VisualLISP-u i u VBA-u (od verzije 2010, VBA više nije podržan). Od verzije 2007 Autodesk je dodao mogućnost proširenja u ObjectARX-u pomoću .NET jezika (detaljnije: URL-1). ObjectARX omogućava direktan pristup svim dijelovima otvorene arhitekture AutoCAD-a te pristup bazi objekata, grafičkom sustavu i svim ostalim fundamentalnim naredbama. VisualLISP je modifikacija Lilsp programskog jezika koja upravlja AutoCAD-ovim objektima. Razlika između VisualLISP-a i ObjectARX-a je što je VisualLISP interpretacijski programski jezik, tj. ne pretvara se u izvršni kod nego se izvršava liniju po liniju, što je na neki način i sporije izvršavanje. VisualLISP je više pogodan za razvijanje manjih i jednostavnijih aplikacija.

2. HELLO WORLD!

Krenimo redom. Programiranje modula ili aplikacija u .NET-u za AutoCAD razvija se u programu Microsoft® Visual Express (besplatna inačica Visual Studio paketa). Krajnji rezultat modula je .dll datoteka. Prilikom stvaranja novog projekta u Visual Express-u je potrebno odabrati tip projekta *Class Library*. Najvažnija stvar kod programiranja modula za AutoCAD je korištenje tzv. AutoCAD API-a. API (engl. Application Programming Interface) je sučelje pomoću kojega jedan program komunicira s drugim. Da bismo mogli pozvati AutoCAD objekte i funkcije potrebno se u projektu referencirati na dvije datoteke, *acmgd.dll* i *acdbmgd.dll*. *Acmgd.dll* je modul u kojem su opisane sve metode i klase koje se mogu koristiti prilikom programiranja aplikacija za proširenje AutoCAD-a. *Acdbmgd.dll* je modul sličan prethodnom, ali se odnosi na klase i funkcije za manipulaciju objektima koji se nalaze unutar baze AutoCAD crteža. Jako važna stvar prilikom postavljanja referenci na te dvije .dll datoteke je da se u *Properties* prozoru postavi opcija *Copy local* na *False*. U protivnome, AutoCAD neće moći pokrenuti našu aplikaciju. Poželjno je u opcijama projekta podesiti AutoCAD aplikaciju preko koje ćemo se vezati kada ulazimo u tzv. debug mode (*Project-> Ime_Projekta Properties-> Debug-> Start external program*). Debug mode nam omogućuje zaustavljanje programa u određenom trenutku prilikom njegovog izvršavanja. Kod otklanjanja pogrešaka u kodu, *Debug*

mode će nam biti od velike pomoći.

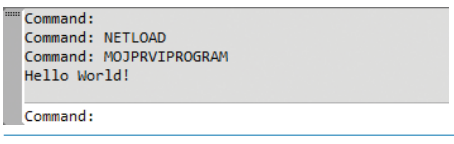
Najjednostavniji i najčešći primjer programiranja dan je u slijedećim linijama:

```
using Autodesk.AutoCAD.Runtime;
using Autodesk.AutoCAD.ApplicationServices;
using Autodesk.AutoCAD.EditorInput;

namespace MojPrviProgram
{
 public class Class1
 {
 [CommandMethod("mojprviprogram")]
 public void mojprviprogram()
 {
 Editor ed = Application.DocumentManager
                         .MdiActiveDocument.Editor;
 ed.WriteMessage("Hello World!");
 }
 }
}
```

Promotrimo kod liniju po liniju. Korisnici koji su već programirali u C# prepoznati će izraze koji počinju s riječi »using«, a označuju putanju do određene klase. Ti izrazi pomažu u čitljivosti i jednostavnosti samoga programa. Primjera radi, do klase Editor u tome bi slučaju morali napisati: »Autodesk.AutoCAD.EditorInput.Editor ed....«.

Using služi boljoj preglednosti i čitljivosti koda. Primijetit ćemo da prije metode *mojprviprogram* postoji tzv. atribut metode kojega prepoznajemo po uglatim zagradama prije imena metode. Atribut *CommandMethod* govori o slijedećoj metodi koja će postati dio AutoCAD-a i može ju se pozivati kao i bilo koju drugu AutoCAD naredbu. Argument klase *CommandMethod* je ime našeg programa. Metoda koja se nalazi ispod atributa može imati bilo koji naziv, u našem slučaju, zove se isto kao i ime našeg programa. Kako bismo ispisali »Hello World!« u komandnoj liniji je potrebno pozvati klasu koja ima mogućnosti ispisivanja teksta (klasa *Editor*). Klasa *Editor* je zadužena za sve funkcije koje se odnose na komandnu liniju. Nakon istanciranja klase *Editor*, na objektu »ed« pozivamo metodu *WriteMessage* koja ispisuje tekst. Tu završava programiranje, a počinje učitanje modula u AutoCAD. Prije toga ga je potrebno pretvoriti u izvršni kod (kompajlirati). U Visual Express-u to se radi tipkom F5 i AutoCAD automatski pokreće kompajliranje. Prije pokretanja aplikacije potrebno je modul učitati u AutoCAD. U komandnoj liniji pozivamo naredbu *netload* i odaberemo našu aplikaciju (nalazi se u Debug direktoriju našeg projekta). Na kraju pozivamo metodu *mojprviprogram* i u komandnoj liniji će se ispisati tekst »Hello World!« (slika 1).



SLIKA 1. Učitavanje i pozivanja modula u AutoCAD-u

3. X, Y, Z KOORDINATE - PRIMJER

Koliko puta se dogodilo da ste dobili koordinate točaka u nekom tekstualnom formatu, najčešće X, Y, Z i niste ih znali importirati. Ako imate AutoCAD Map, koji ima mogućnost unošenja točaka svih vrsta tekstualnih datoteka, onda nemate problema. Ako ipak nemate, osuđeni ste na korištenje ili ručnog upisivanja koordinata u AutoCAD (što može biti prilično naporan posao) ili savladavanje programiranja kojim ćete u par linija koda imati program koji će moći očitati sve vrste datoteka, bilo kojeg formata brojeva. Slijedeći primjer služi upravo prethodno navedenome.

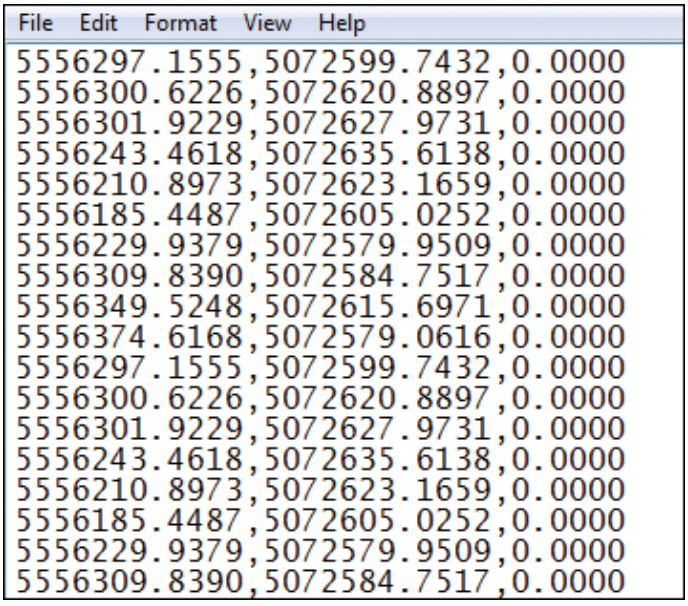
```
using Autodesk.AutoCAD.Runtime;
using Autodesk.AutoCAD.EditorInput;
using Autodesk.AutoCAD.DatabaseServices;
using Autodesk.AutoCAD.ApplicationServices;
using Autodesk.AutoCAD.Geometry;
using System.IO;

namespace Proba_Map2012
{public class Class1 {
 [CommandMethod("mojdrugiprogram")]
 public void mojdrugiprogram()
 {
 Database db = Application.DocumentManager.
MdiActiveDocument.Database;
 StreamReader datoteka = new StreamReader("koordinate.
txt");
 using (Transaction trans = db.TransactionManager.
StartTransaction())
 {
 string linija = "";
 while ((linija = datoteka.ReadLine()) != null)
 {
 string[] koordinate = linija.Split(',');
 double koord_X = double.Parse(koordinate[0]);
 double koord_Y = double.Parse(koordinate[1]);
 double koord_Z = double.Parse(koordinate[2]);

 Point3d acadKoordinata = new Point3d(koord_X, koord_Y,
koord_Z);
 DBPoint acadTočka = new DBPoint(acadKoordinata);

 BlockTable bt = trans.GetObject(db.BlockTableId,
 OpenMode.ForRead) as BlockTable;
 BlockTableRecord btr = trans.GetObject(
 bt[BlockTableRecord.ModelSpace], OpenMode.ForWrite)
 as BlockTableRecord;

 btr.AppendEntity(acadTočka);
 trans.AddNewlyCreatedDBObject(acadTočka, true);
 }
 trans.Commit();
}}}}
```



SLIKA 2. Lista koordinata u datoteci »koordinate.txt«

Ovaj je program nešto složeniji nego prethodni, ali se sastoji od nekih poznatih dijelova koda. Atribut *CommandMethod* definira ime našeg novog programa »mojdrugiprogram«. Između vitičastih zagrada metode *mojdrugiprogram* nalazi se cijeli kod koji čita datoteku, u ovom slučaju je to datoteka *koordinate.txt* (slika 2), razdjeljuje redak na X, Y i Z koordinatu te iscrtava novu točku u AutoCAD-u na tim koordinatama. To je bilo ukratko, krenimo sada korak po korak.

```
Database db = Application.DocumentManager.
MdiActiveDocument.Database;
```

Objekti kao što su blokovi, točke, linije, kružnice, stilovi linija i slojevi spremaju se u bazu crteža AutoCAD-a (slika 3). Za stvaranje novih AutoCAD točaka i dodavanje u crtež (spremanje u bazu) potrebno je prvo pristupiti bazi crteža AutoCAD-a. U prethodnom isječku koda smo upravo to napravili. Trenutnu bazu crteža proslijedili smo u objekt *db*. Svi elementi koji se nalaze u jednom AutoCAD crtežu, sada se nalaze u toj jednoj varijabli, točnije objektu.

```
StreamReader datoteka = new StreamReader("koordinate.txt");
```

Gornjom linijom koda otvaramo datoteku *koordinate.txt* kako bismo iz nje iščitali koordinate.

```
using (Transaction trans = db.TransactionManager.
StartTransaction())
```

Prethodna linija koda prilično je važna što se tiče pristupanja objektima iz baze crteža. Sve operacije pristupanja, modificiranja i brisanja objektima iz baze moraju se događati unutar transakcija. Želimo li u crtež dodati ili izbrisati novi sloj ili pohraniti promjene koje smo napravili ili ga izbrisati, te operacije s objektima u bazi radimo preko transakcija. Upravo nam transakcije osiguravaju konzistentnost podataka u bazi.

```
string linija = "";
 while ((linija = datoteka.ReadLine()) != null)
 {
 string[] koordinate = linija.Split(',');
 double koord_X = double.Parse(koordinate[0]);
 double koord_Y = double.Parse(koordinate[1]);
 double koord_Z = double.Parse(koordinate[2]);
 }
```

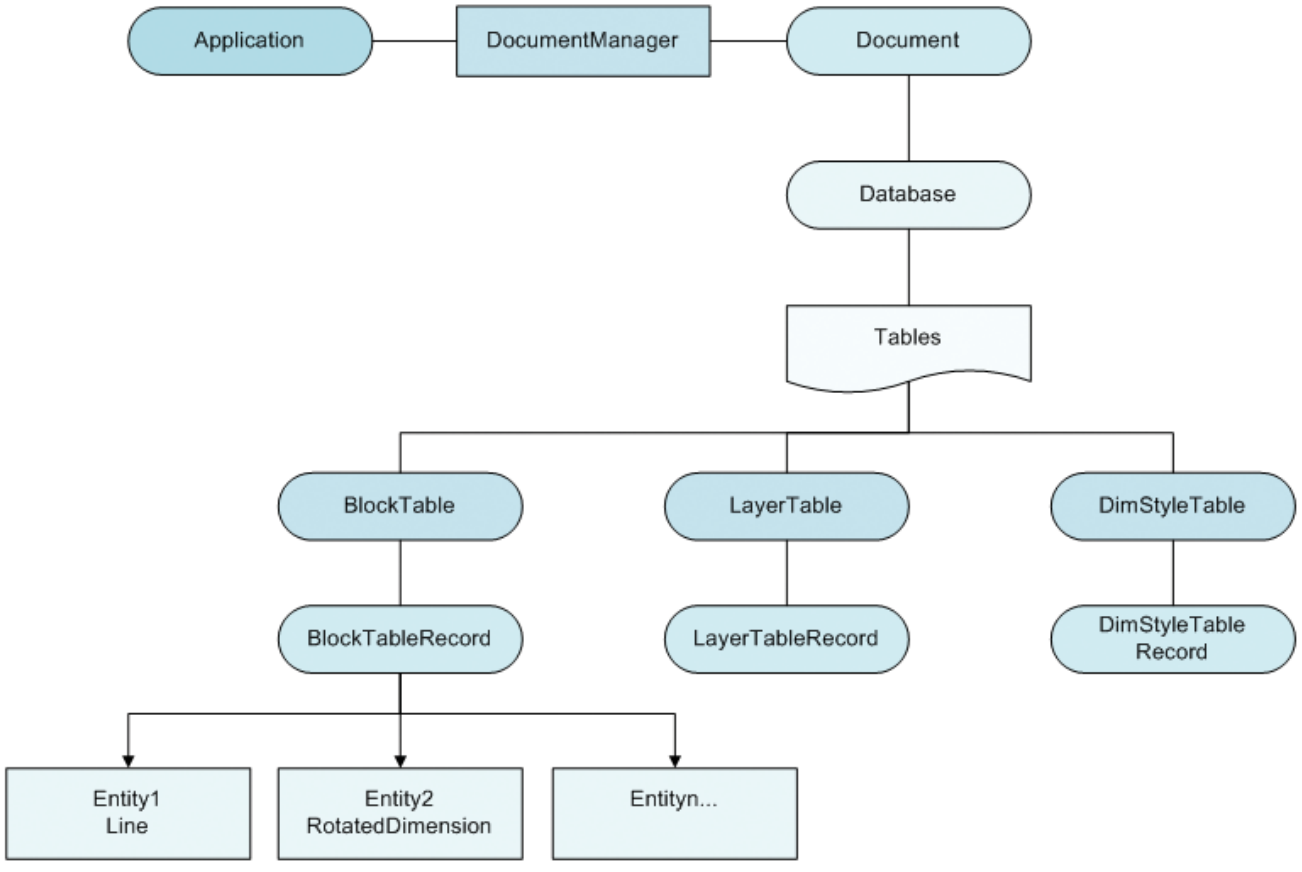
Prethodni se dio direktno ne tiče programiranja u AutoCAD-u. U ovom dijelu koda najvažnije je shvatiti pomoć petlje *while* koja očitava redove linija tekstualne datoteke. Dok god postoji tekst, *while* petlja raščlanjuje linije u listu, koja se kasnije iz znakova pretvara (metoda *Parse*) u decimalne brojeve (za X, Y i Z koordinate). Sada kada znamo koordinate, možemo ih iskoristiti za definiranje lokacije točke u AutoCAD-u. Prije stvaranja nove točke, treba napraviti slijedeće:

```
Point3d acadKoordinata = new Point3d(koord_X, koord_Y,
koord_Z);
```

Pomoću klase *Point3d* izradit ćemo »točku« s koordinatama koje smo prethodno izvukli iz tekstualne datoteke. Pri tome, objekt *Point3d* nema nikakve veze s točkom kao geometrijskim objektom. *Point3d* predstavlja objekt koji pokazuje na točno određenu lokaciju u prostoru. Pomoću objekta *acadKoordinata* moći ćemo AutoCAD točku (točku kao geometrijski objekt) smjestiti u prostoru.

```
DBPoint acadTočka = new DBPoint(acadKoordinata);
```

AutoCAD točka je prethodnom naredbom postala geometrijski



SLIKA 3. Hijerarhijski prikaz AutoCAD objekata u API sučelju

objekt. Kod kreiranja objekta u konstruktor klase smo proslijedili objekt *acadKoordinata* i to kako bismo definirali lokaciju točke. Kako bi ta točka postala vidljiva nju treba ubaciti u bazu AutoCAD crteža:

```
BlockTable bt = trans.GetObject(db.BlockTableId,
 OpenMode.ForRead) as BlockTable;
BlockTableRecord btr = trans.GetObject(
 bt[BlockTableRecord.ModelSpace], OpenMode.ForWrite)
 as BlockTableRecord;
```

Najjednostavnije rečeno, baza crteža se sastoji od nekoliko vrsta tablica. Jedna od tih tablica je i *BlockTable* u koju se spremaju svi geometrijski objekti (točke, linije, polilinije, kružnice, lukovi i drugi geometrijski objekti). Preko objekta transakcije *GetObject* ćemo pristupiti bazi. U toj liniji koda smo preko transakcije otvorili tablicu *BlockTable*. Svaki geometrijski objekt (entitet) se u prethodnoj tablici predstavlja kao jedan zapis, točnije kao *BlockTableRecord*. Kako bismo pomoću njega točku pretvorili u *BlockTableRecord* zapis moramo pristupiti i tom objektu.

```
btr.AppendEntity(acadTočka);
```

S ovom linijom smo entitet (točku) pretvorili u zapis u tablici i ona je postala novi entitet u crtežu. Kako bismo u bazi potvrdili ove promjene opet se vraćamo na transakcije.

```
trans.AddNewlyCreatedDBObject(acadTočka, true);
```

O novom entitetu moramo obavijestiti i transakciju te za kraj ispisujemo:

```
trans.Commit();
```

Time sve prethodne promjene potvrđujemo. Zadnjom linijom koda sve promjene na bazi su trajno pohranjene i postaju vidljive u crtežu.

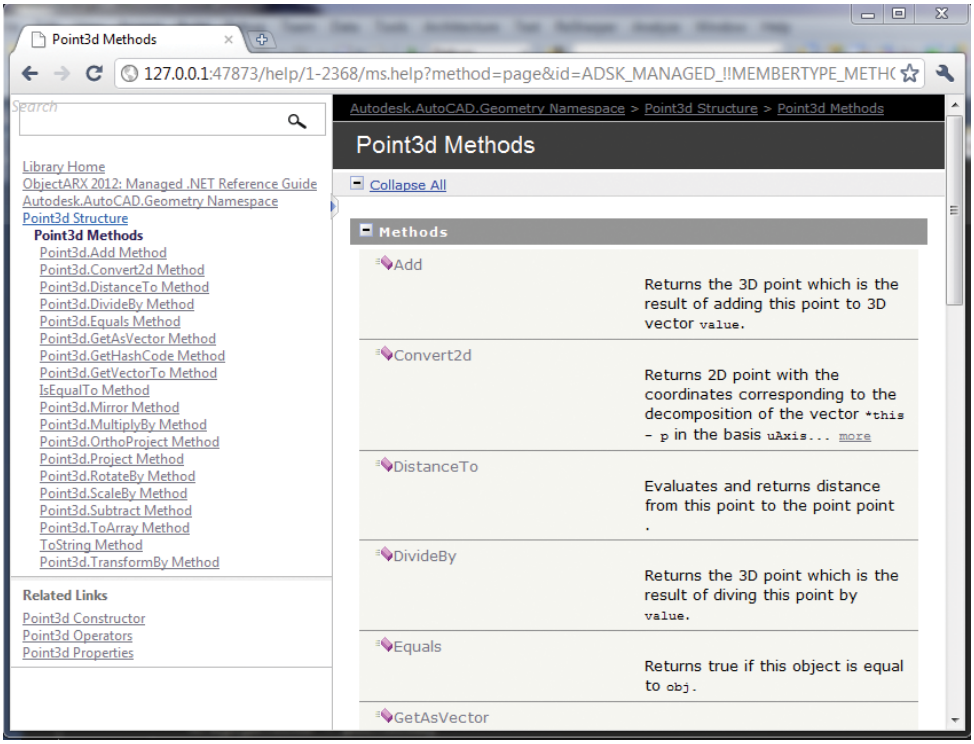
4. REFERENCE ZA POMOĆ

Svaki programer u jednom trenutku dođe do mrtve točke i nije siguran kako dalje. U glavi zna kako riješiti taj problem samo ne zna s kojim »alatom«. Ne zna sve mogućnosti koje mu pruža programski jezik ili sučelje koje koristi (API). Neke od mogućnosti koje mogu dobro doći i onima koji se tek upuštaju u programiranje u AutoCAD-u, a i onima iskusnijima su:

1. Od ugrađenih alata koje dolaze s Visual Express-om najviše će vam pomoći alat pod nazivom ObjectBrowser. ObjectBrowser nije ništa drugo nego preglednik svih klasa i metoda koje se nalaze u modulima (referencama) koje ste učitali u projekt. Na jednostavan se način daje pregled klasa koje se nalaze u pojedinim modulima te se prikazuju povratni i ulazni tipovi varijabli koje se traže za raspoložive metode. Također, ObjectBrowser od velike je pomoći kada niste sigurni postoji li eventualno neka klasa ili metoda koja baš radi ono što želite.
2. Ako ste tek krenuli programirati u AutoCAD-u, dobro mjesto za početak je uputstvo *AutoCAD .NET Developer's Guide* (URL-2). Ondje ćete pronaći upute od samoga početka programiranja i postavljanja projekta. Svako poglavlje nudi primjere koje ćete naći u VB.NET i C# programskom jeziku što će uvelike doprinjeti samom razumijevanju poglavlja.
3. Kako bi se dodatno olakšalo programiranje u AutoCAD-u, Autodesk se pobrinuo za još jednu vrstu pomoći, a to je integriranje objašnjenja svake pojedine klase preko sustava pomoći koja je već ugrađena u Visual Express. Ovakvu vrstu pomoći morati ćete prvo instalirati na računalo. Stranica na kojoj se nalazi instalacija je URL-3. Ukoliko za neku klasu nismo sigurni što predstavlja ili koje su mogućnosti neke metode, dovoljno je označiti tu klasu (postaviti kursor) i stisnuti tipku F1. Visual Express će vas tada automatski preusmjeriti na objašnjenje i na specifične klase/metode (slika 4).
4. Na kraju, ali i ne manje važno: svakom instalacijom AutoCAD-a dolazi hrpa uputa i novosti koje su dodane u pojedinoj verziji AutoCAD-a. Direktorij u kojem se sve to nalazi smješten je u instalacijskom direktoriju AutoCAD-a (najčešće je to Program Files->Autodesk->AutoCAD) pod nazivom Help.

5. ZA KRAJ

Malom školom nismo ni zagrebali površinu raznolikih mogućnosti koje krije sučelje programiranja u AutoCAD-u. Za kvalitetno ovladavanje klasama i metodama koje se nude na raspolaganju kroz vaše je prste potrebno proći mnogo linija koda. Osim toga, paralelno je izuzetno korisno prolaziti i analizirati primjere aplikacija razvijenih od strane drugih programera. Ako vam ovo izgleda kao nemoguć zadatak probajte si postaviti cilj i nemojte odustati. Kada jednom dođete do cilja, znanje koje ste usputno prikupili biti će veće od krajnjeg rezultata. Sretno kodiranje!



SLIKA 4. Preko tipke F1 lako možemo doći do liste svih metoda koje posjeduje objekt »Point3d«

LITERATURA

› URL-1: Wikipedia: .NET Framework, http://en.wikipedia.org/wiki/.NET_Framework, (10.04.2011.).
› URL-2: AutoCAD .NET Developer's Guide, http://docs.autodesk.com/ACD/2011/ENU/filesMDG/WS1a9193826455f5ff2566ffd511ff6f8c7ca-4875.htm, (14.4.2011.).
› URL-3: Autodesk - Developer Center, http://usa.autodesk.com/adsk/servlet/item?siteID=123112&id=785550, (14.4.2011.).