Distribuirani procesi Vježbe 02

Vinko Petričević

Korištenje COM objekata

- Dodati u reference
- Eventualno koristiti namespace - using
- Koristimo ga kao svaki drugi objekt

Korištenje COM objekata – primjer 2.2

- U Wordu koristeći opciju Tools-Record New Macro snimiti Macro
- Editirati njegov kôd u VBA-u
- U VS-u kreirati novi Console Application C# projekt
- Dodati u reference Word
- Eventualno koristiti namespace – using
- Kreirati novi objekt tipa Word.Application
- Koristimo ga kao svaki drugi objekt
- Kopirati kod iz VBA u VS, preraditi mu sintaksu Basic -> C#
- U primjeru 2.2 se kreira novi dokument koji se samo spremi u datoteku “d:\1.doc”

Socketi

- Socket API-jem možemo ostvariti komunikaciju kao što se to radilo na kolegiju Mreže računala
- Using System.Net.Sockets;
- TcpListener – klasa koja treba na serveru
- TcpClient – klasa koja treba za komunikaciju
- NetworkStream – klasa za slanje/primanje podataka
- U primjeru 2.1 imamo jednostavan primjer slijenta i servera. Za svaki kreiramo novi C# Console Application

.net remoting

- Postoji više načina. Mi ćemo koristiti onaj najsličniji DCOMu
- Udaljeni objekti su oni koji su izvan korisnikove aplikacijske domene
- Možemo ih slati po vrijednosti ili po referenci
- Korisnik mu pristupa jednako kao i lokalnom objektu, a .net framework se brine da se sve dobro izvrši

.net remoting

- Ako se šalje po referenci, izvršava se u aplikacijskoj domeni gdje je kreiran (može biti i na udaljenom računalu), korisniku je to nebitno. Svi pozivi se *marshaliraju*
- Ako se šalje po vrijednosti, objekt (i svi *zavisni* objekti) se kopiraju (serijaliziraju) u aplikacijsku domenu pozivatelja i tamo izvršavaju

Klijent/server veza

- Prvo treba kreirati Channel za komunikaciju
- Dodati referencu za System.Runtime.Remoting

```
using System.Runtime.Remoting;
using System.Runtime.Remoting.Channels;
//using System.Runtime.Remoting.Channels.Http;
using System.Runtime.Remoting.Channels.Tcp;

…
//          HttpChannel channel = new HttpChannel(port);
            TcpChannel channel = new TcpChannel(port);

            ChannelServices.RegisterChannel(channel);
```

Postaviti objekt javnim

- Nakon toga postaviti objekt javnim

```
RemotingConfiguration.RegisterWellKnownServiceType(
            typeof(Tip),
            "Naziv",
//          WellKnownObjectMode.SingleCall);
            WellKnownObjectMode.Singleton);
```

- Sada klijent može kreirati objekt (treba prvo dodati referencu na servera)

```
Tip mine = (Tip)Activator.GetObject(
//          typeof(Tip), "http://localhost:port/Naziv");
            typeof(Tip), "tcp://localhost:port/Naziv");
```

Slanje objekata

- Slanje po vrijednosti (server mora imati pristup Tip-u. Najjednostavnije da se u folderu gdje je server nalazi Klijent.exe

```
[Serializable()] class Tip { … }
```

- Slanje po referenci

```
class Tip : MarshalByRefObject { … }
```

Slanje po vrijednosti – primjer 2.3

- Potrebno je kreirati 2 nova Console Application C# projekta
- U reference oba dodati System.Runtime.Remoting
- U prvi kopirati kod servera u prvi, te ga kompajlirati
- U drugi kopirati kod klijenta, a u reference dodati novokreirani server, te program kompajlirati
- Da bi radilo slanje klijentske klase po vrijednosti, potrebno je kopirati klijentski .exe u folder odakle ćemo izvršavati server, jer nakon što se klijentska klasa deserijalizira na serveru, .net framework mora imati i kôd njezinih funkcija
- Pokrenimo prvo server (prije pokretanja klijentski .exe moramo iskopirati), pa klijent

Zadatak

- Napravite jednostavan chat server
- Server pamti poruke
- Klijent šalje poruke
- Po završetku klijent ispiše sve poruke
- Neka server sačeka nekoliko sekundi prije primanja svake poruke