

# Windows servisi

---

2014/15.13

# (Windows) servisi

## ❑ Windows servisi (engl. *Windows Services*)

- dugotrajni (*long-running*) izvršni programi
- nemaju korisničkog sučelja, ne zahtijevaju interakciju s korisnikom
- izvršavaju se neovisno o prijavi korisnika na računalo
- mogu biti automatski pokrenuti pri pokretanju operacijskog sustava
- OS Unix ima sličan tip programa pod nazivom demon (*daemon*)

## ❑ Životni vijek Windows servisa – nekoliko internih stanja

- instalacija servisa na računalo - servis se učitava u *Services Control Manager* (centralno mjesto za administraciju servisa)
- pokretanje – nakon što je servis instaliran može se pokrenuti.
  - iz *Management Console*, pozivom *Start* metode iz programskog koda, automatski prilikom pokretanja računala ...
- nakon pokretanja servis se nalazi u stanju *Running* dok ne bude zaustavljen (stanje *Stopped*), pauziran (*Paused*) ili dok računalo ne bude isključeno

# Upravljanje instaliranim servisima

- ❑ **Upravljanje servisom vrši se kroz *Services Management Console***
  - Control Panel \ Administrative Tools \ Services ili
  - My Computer – Manage \ Services and Applications \ Services
  
- ❑ **Pregled postojećih servisa na računalu i njihovo stanje**
  - pokretanje, zaustavljanje, pauziranje servisa
  - administriranje načina pokretanja servisa
    - automatski – servis se pokreće pri pokretanju OS
    - automatski (odgođeno) – nakon pokretanja zahtjevnijih servisa
    - ručno – pokreće korisnik iz konzole ili drugi program metodom *Start*
    - onemogućeno
  
- ❑ **Iako obično nemaju GUI, programer može dodati UI komponentu**
  - Omogućiti s “*Allow service to interact with desktop*” u *Logon* dijelu postavki

# Upravljanje servisima

The screenshot shows the Windows XP interface. In the background, the 'Control Panel > Administrative Tools' window is open, with 'Services' selected. The 'Services (Local)' console is also open, displaying a list of services. The 'ASP.NET State Service' is highlighted, and its properties are shown in the foreground dialog box.

**ASP.NET State Service Properties (Local Computer)**

General | Log On | Recovery | Dependencies

Service name: aspnet\_state  
Display name: ASP.NET State Service  
Description: Provides support for out-of-process session states for ASP.NET. If this service is stopped, out-of-process

Path to executable: C:\Windows\Microsoft.NET\Framework\v2.0.50727\aspnet\_state.exe

Startup type: Manual (selected)  
Automatic (Delayed Start)  
Automatic  
Manual  
Disabled  
Stopped

Service status: Stopped

Start, Stop, Pause, Resume buttons

You can specify the start parameters that apply when you start the service from here.

Start parameters:

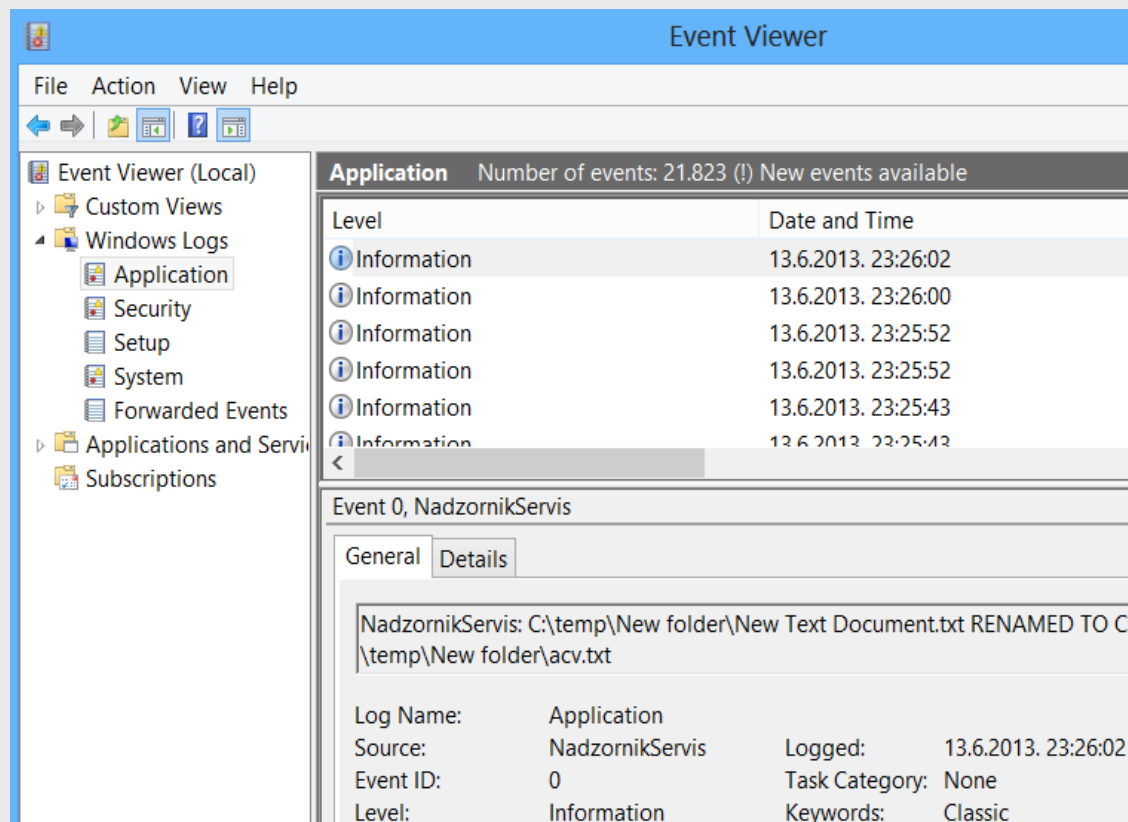
OK, Cancel, Apply buttons

Name	Status	Startup Type	Network Location
Application Experience	Started	Automatic	Local System
Application Information	Started	Automatic	Local System
Application Layer	Started	Automatic	Local System
Application Manager	Started	Automatic	Local System
ASP.NET State Service	Stopped	Manual	Network Location
Ati External Event	Started	Automatic	Local System
AVG7 Alert Manager	Started	Automatic	Local System
AVG7 Resident Shield	Started	Automatic	Local System
AVG7 Update Service	Started	Automatic	Local System
Background Intelligent Transfer Service	Started	Automatic (Delayed Start)	Local System
Base Filtering Engine	Started	Automatic	Local Service
Block Level Backup Engine	Started	Manual	Local System
Bluetooth Support Service	Started	Automatic	Local Service

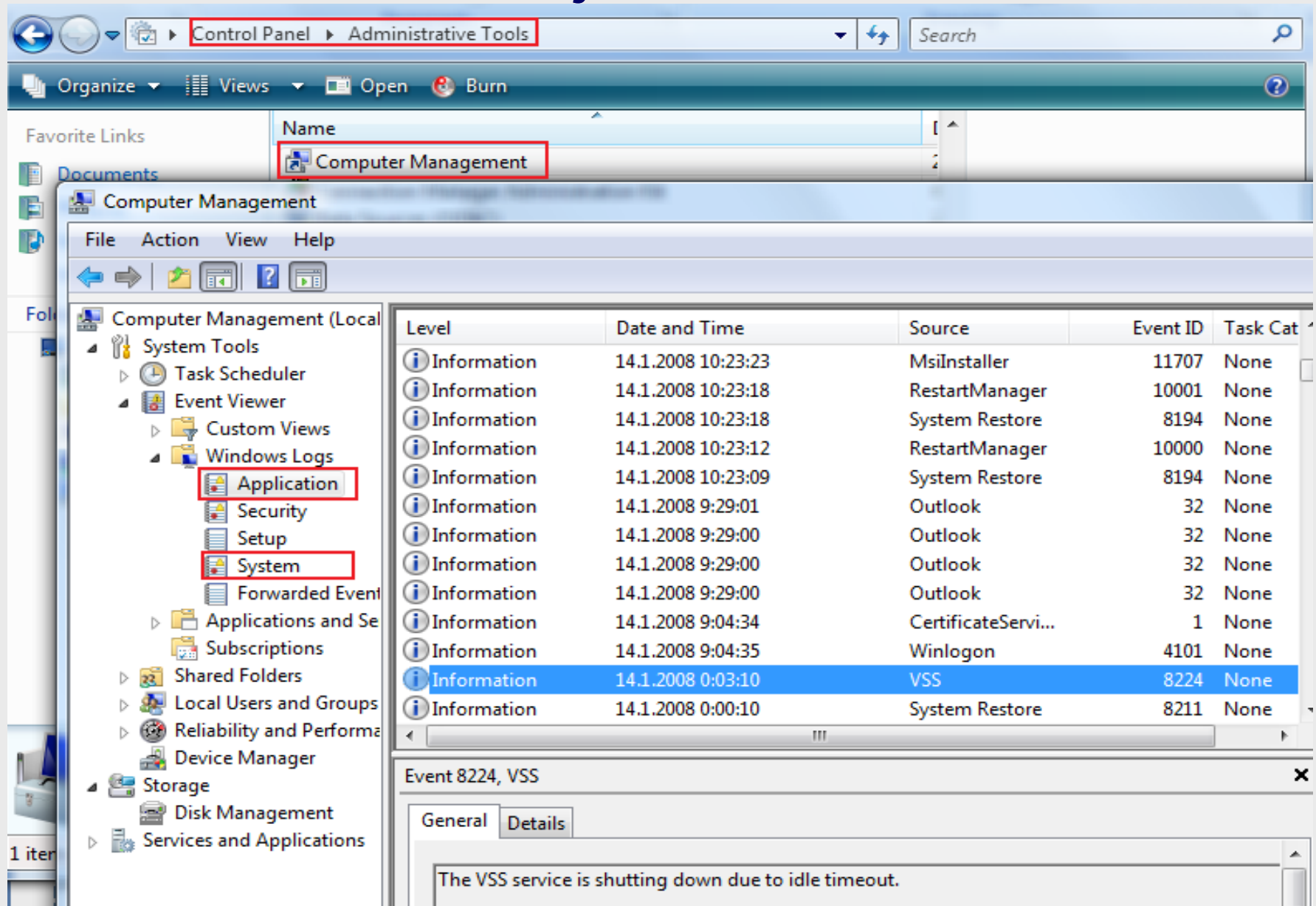
# Informacije o radu servisa

## ❑ Osnovne informacije o promjeni stanja servisa i njihovim akcijama zapisuju se u dnevnik događaja (*event log*)

- pregled dnevnika obavlja se u dijelu konzole *Event Viewer*
- Pokretanje: Control Panel \ Administrative Tools \ Event Viewer
- preglednik evidentira i događaje drugih aplikacija i OS



# Informacije o radu servisa



The screenshot shows the Windows XP Administrative Tools window. The 'Control Panel' > 'Administrative Tools' path is highlighted in the address bar. In the 'Computer Management' console tree, the 'System' log under 'Windows Logs' is selected. The main pane displays a list of events. Event 8224, titled 'VSS', is highlighted. A details window for this event is open at the bottom, showing the message: 'The VSS service is shutting down due to idle timeout.'

Level	Date and Time	Source	Event ID	Task Cat
Information	14.1.2008 10:23:23	MsiInstaller	11707	None
Information	14.1.2008 10:23:18	RestartManager	10001	None
Information	14.1.2008 10:23:18	System Restore	8194	None
Information	14.1.2008 10:23:12	RestartManager	10000	None
Information	14.1.2008 10:23:09	System Restore	8194	None
Information	14.1.2008 9:29:01	Outlook	32	None
Information	14.1.2008 9:29:00	Outlook	32	None
Information	14.1.2008 9:29:00	Outlook	32	None
Information	14.1.2008 9:29:00	Outlook	32	None
Information	14.1.2008 9:04:34	CertificateServi...	1	None
Information	14.1.2008 9:04:35	Winlogon	4101	None
Information	14.1.2008 0:03:10	VSS	8224	None
Information	14.1.2008 0:00:10	System Restore	8211	None

Event 8224, VSS

General Details

The VSS service is shutting down due to idle timeout.

# Izrada Windows servisa

## ❑ Predložak projekta *Windows Service*

- automatski u projekt dodaje odgovarajući razred
- nasljeđuje razred *System.ServiceProcess.ServiceBase*
- ponašanje servisa definira se preopterećenjem postupaka *OnStart*, *OnStop*, *OnPause*, *OnContinue*,...

## ❑ Primjer: **WinServis \ FirmaWindowsService**

- Servis koji periodički dohvaća popis najboljih klijenata te ga sprema na disk u PDF obliku
- *ServiceBase* razred nalazi se u *System.ServiceProcess* knjižnici pa je referenca na knjižnicu također automatski dodana

```
using System.ServiceProcess;  
  
namespace FirmaWindowsService  
{  
    public partial class FirmaStatistika: ServiceBase  
    {  
        ...  
    }  
}
```

# Glavni program Windows servisa

## ❏ Primjer: WinServis \ FirmaWindowsService – Program.cs

- `Main` postupak mora izdati naredbu `Run` s popisom svih servisa koje projekt sadrži
- Pozivom metode `Run` servis se učitava / registrira u *Services Management Console*
- Ukoliko je projekt nastao korištenjem predloška odgovarajući kod u `Main` postupak je automatski dodan.

```
static void Main()  
{  
    ServiceBase[] ServicesToRun;  
    ServicesToRun = new ServiceBase[] {  
        new FirmaStatistika ()  
    };  
    ServiceBase.Run(ServicesToRun);  
}
```



# Događaji Windows servisa

## ❑ Događaji osnovnog razreda

### `System.ServiceProcess.ServiceBase`

- `OnStart` – izvršava se prilikom pokretanja servisa
- `OnPause` – prilikom pauziranja servisa
- `OnStop` – prilikom zaustavljanja servisa
- `OnContinue` – pri povratku iz stanja zaustavljanja u stanje izvršavanja
- `OnShutdown` – prilikom isključivanja sustava
- `OnCustomCommand` – prilikom zadavanja korisničke naredbe (prima *int* )
- `OnPowerEvent` – prilikom power management događaja (npr. baterija slaba, suspend)

## ❑ Funkcionalnost servisa započinje kodom u `OnStart`

- (u našem primjeru kod za pokretanje nadgledanja direktorija)

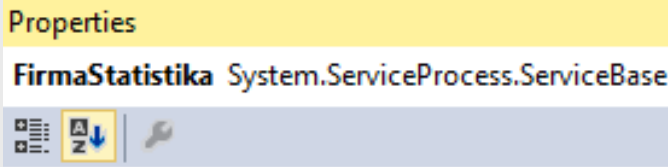
# Svojstva Windows servisa

## ❑ Svojstva `System.ServiceProcess.ServiceBase`

- `AutoLog` – automatsko zapisivanje u event log događaja (npr. *Install* ili *Start*)
- `CanHandlePowerEvent` – da li servis obrađuje *Power management* događaje (npr. baterija slaba, *suspend*)
- `CanStop` – mogućnost poziva zaustavljanja servisa (metoda `OnStop`)
- `CanPauseAndContinue` – mogućnost poziva `OnPause` i `OnContinue`
- `CanShutdown` – mogućnost izvršavanja `OnShutdown` pri isključivanju

## ❑ Primjer: `WinServis \ FirmaWindowsService – FirmaStatistika.cs`

- View Designer \ *Properties*
- postaviti `CanPauseAndContinue`, `CanShutdown` na *True*



The screenshot shows the Visual Studio Properties window for a project named 'FirmaStatistika'. The 'Properties' tab is selected, and the 'System.ServiceProcess.ServiceBase' properties are displayed. The 'CanShutdown' property is highlighted in blue.


(Name)	FirmaStatistika
AutoLog	True
CanHandlePowerEvent	False
CanHandleSessionChangeEvent	False
CanPauseAndContinue	True
CanShutdown	True
CanStop	True
ExitCode	0
Language	(Default)
Localizable	False
ServiceName	FirmaStatistika

# Konfiguracijska datoteka

## ❑ Primjer: WinServis \ FirmaWindowsService – app.config

- direktorij u koji će se izvješće spremiti i period kreiranja izvještaja (*Path*):
- postavke za spajanje na bazu podataka

```
<appSettings>
  <add key="Interval" value="60000"/>
  <add key="OutputDir" value="C:\Temp\FirmaStatistika\"/>
</appSettings>
<connectionStrings>
  <add name="Firma" connectionString...
```

- Konfiguracijska datoteka nakon kompilacije projekta ima naziv oblika NazivProjekta.exe.config (npr. FirmaWindowsService.exe.config)
- U slučaju promjene konfiguracijske datoteke potrebno osvježiti dio ili cijelu konfiguracijsku datoteku
  - Primjer:  WinServis \ FirmaWindowsService – FirmaStatistika.cs

```
private void RefreshConfigFile() {
    ConfigurationManager.RefreshSection("appSettings");
}
```

# Događaji servisa - OnStart

## ❑ Primjer: WinServis \ FirmaWindowsService – FirmaStatistika.cs

- **OnStart** ispisiye informaciju u sistemski dnevnik (log) i inicijalizira *Timer* koji će se aktivirati svakih n milisekundi
- *Timer* definiran s 4 parametra:
  - Postupak koji će se izvršiti nakon isteka zadanog vremena (definiran delegatom *TimerCallback*)
  - Objekt koji se prenosi postupku definiranom pomoću *TimerCallback*
  - Inicijalno vrijeme čekanja do prvog poziva
  - Interval poziva

```
protected override void OnStart(string[] args) {  
    IspisiInfo();  
    TimerCallback callback = new TimerCallback(DoJob);  
    int interval =  
        int.Parse(ConfigurationManager.AppSettings["Interval"]);  
    timer = new System.Threading.Timer(callback, null, 0, interval);  
}  
  
private void IspisiInfo() {  
    EventLog.WriteEntry(...
```

# Realizacija događaja *Timera*

## ❑ Primjer: WinServis \ FirmaWindowsService – FirmaStatistika.cs

- Događaj se poziva svakih n milisekundi (korištenjem *Timera*)
- U slučaju pogreške poruka se sprema u sistemski dnevnik (*Event Log*)

```
private void DoJob(object o) {  
    try {  
        string outputDir = ConfigurationManager.AppSettings["OutputDir"];  
        ...  
        string filename = string.Format("NajboljiKupci_{0}.pdf",  
            DateTime.Now.ToString("yyyy_MM_dd_HH_mm_ss"));  
        filename = Path.Combine(outputDir, filename);  
        byte[] bytes = StvoriIzvjesce();  
        File.WriteAllBytes(filename, bytes);  
        lastRun = DateTime.Now;  
    }  
    catch (Exception exc){  
        EventLog.WriteEntry(exc.Message + exc.StackTrace,  
            EventLogEntryType.Error);  
        ...  
    }  
}
```

# Izvešće o najboljim kupcima

## ❑ Izveštaj za tip podatka koristi razred koji sadrži osnovne podatke o najboljim partnerima

- Primjer : WinServis \ FirmaWindowsService – PrometPartnera.cs
- Za izvešće u projektu označeno da se kopira u direktorij koji sadrži kompiliranu verziju servisa (*Copy To OutputDirectory = Copy if newer*)

ReportNajboljiKupci.rdlc [Design]

### Najboljih 10 partnera

Id	Naziv	OIB	Promet
[IdPartnera]	[Naziv]	[OIB]	[Promet]
Ukupni promet najboljih partnera			[Sum(Promet)]

Report Data

- Built-in Fields
- Parameters
- Images
- Data Sources
- Datasets
  - DataSet1
    - IdPartnera
    - Naziv
    - OIB
    - Promet

#### Properties

##### ReportNajboljiKupci.rdlc File Properties

Build Action	Embedded Resource
Copy to Output Directory	Copy if newer
Custom Tool	
Custom Tool Namespace	
File Name	ReportNajboljiKupci.rdlc

# Stvaranje izvješća

## ❑ Primjer: WinServis \ FirmaWindowsService – FirmaStatistika.cs

- Dohvat liste najboljih kupaca (kroz ADO.Net u postupku NajboljiKupci) i pridruživanje konkretnom *DataSetu* izvještaja.
- Smještanje izvještaja unutar instance *ReportViewera*
  - Aktivni direktorij za servis će biti Windows\System32 pa je potrebno saznati gdje je izvršna datoteka servisa korištenjem `System.AppDomain.CurrentDomain.BaseDirectory`

```
private byte[] StvoriIzvjesce() {  
    var najbojiKupci = NajboljiKupci();  
    ReportViewer reportViewer = new ReportViewer();  
    string serviceDir = System.AppDomain.CurrentDomain.BaseDirectory;  
    reportViewer.LocalReport.ReportPath =  
        Path.Combine(serviceDir, "ReportNajboljiKupci.rdlc");  
    reportViewer.LocalReport.DataSources.Add(  
        new ReportDataSource("DataSet1", najbojiKupci));  
    reportViewer.LocalReport.Refresh();  
    byte[] bytes = ExportToPDF(reportViewer);  
    return bytes;  
}
```

# Spremanje izvješća kao PDF

- ❑ **Primjer:**  **WinServis \ FirmaWindowsService – FirmaStatistika.cs**
  - Postupak *Render* na izvještaju unutar instance *ReportViewer*a

```
private static byte[] ExportToPDF(ReportViewer reportViewer) {  
    Warning[] warnings;  
    string[] streamIds;  
    string mimeType = string.Empty;  
    string encoding = string.Empty;  
    string extension = string.Empty;  
    byte[] bytes = reportViewer.LocalReport.Render("PDF", null,  
        out mimeType, out encoding,  
        out extension, out streamIds, out warnings);  
    return bytes;  
}
```



# Događaji servisa - OnStop

**Primjer:**  **WinServis \ FirmaWindowsService – FirmaStatistika.cs**

- **OnStop** kod koji će izvršiti prilikom zaustavljanja servisa
  - U konkretnom primjeru uklanja *Timer*

```
protected override void OnStop()  
{  
    if (timer != null){  
        timer.Dispose();  
        timer = null;  
    }  
}
```

# Događaji servisa – OnCustomCommand

## ❑ Omogućuje ugradnju korisnički definiranih postupaka

- Prima *int* parametar koji određuje koju radnju servis treba obaviti
- Vrijednosti parametra od 0 do 127 su sistemski zauzete pa je za korisnički definirane komande dozvoljeno koristiti vrijednosti 128 do 256

## ❑ Primjer: WinServis \ FirmaWindowsService – FirmaStatistika.cs

- npr. komanda 128 neka osvježava podatke o intervalu stvaranja izvještaja i direktoriju u koji se sprema izvješće (promjena nastala korištenjem primjera *FirmaServiceManager* – objašnjeno naknadno)

```
protected override void OnCustomCommand(int command)
{
    base.OnCustomCommand(command);
    if (command == 128)
    {
        RefreshConfigFile ();
    }
}
```

# Ugradnja i izvođenje servisa

## ❑ Problem:

- Servis se u razvojnoj okolini ne može pokretati ili *debugirati* na način uobičajen za interaktivne aplikacije, pritiskom na F5 ili F11 ( *Visual Studio* )

## ❑ Prevedena izvršna datoteka servisa mora biti instalirana

- Treba izraditi prikladnu instalacijsku komponentu
- Instalacijska komponenta instalira i registrira servis na računalu te stvara ulaznu točku servisa za *Windows Service Control Manager*

## ❑ Tek nakon što se servis instalira i pokrene, može se koristiti *VS debugger* pri čemu se potrebno spojiti na proces servisa:

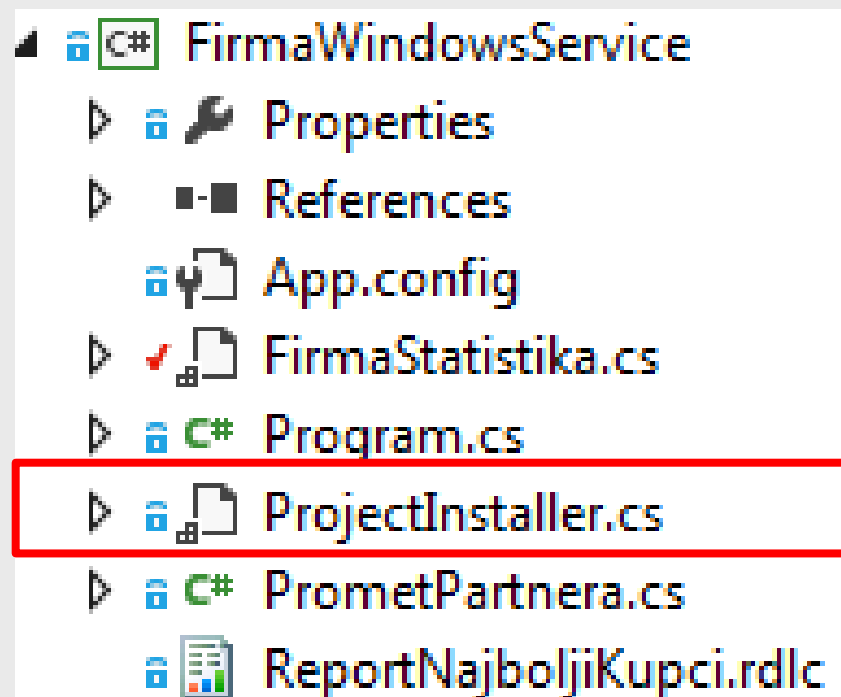
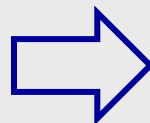
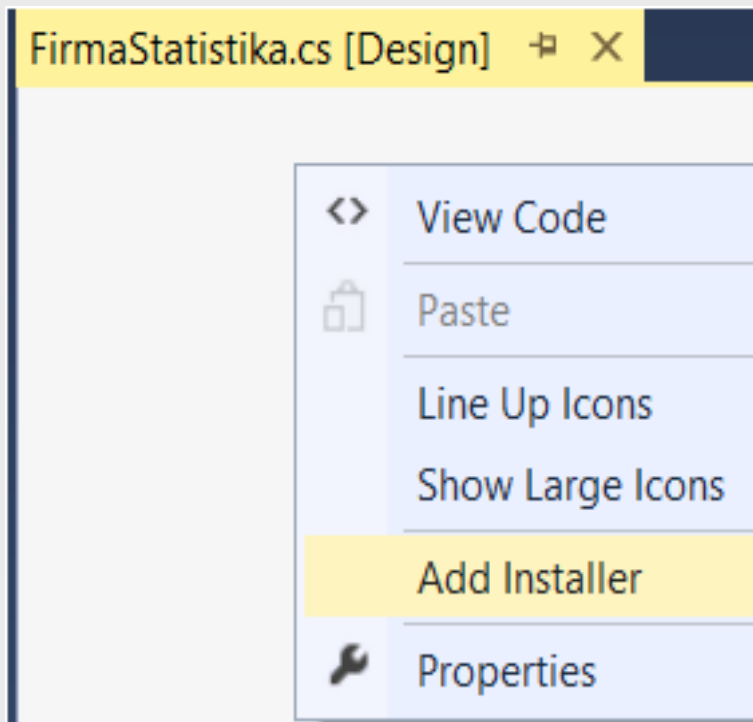
- *Tools / Attach to Process*

# Izrada instalacijske komponente za windows servis

## ❑ Izrada instalacijske komponente

- *Add Installer* funkcija iz skočnog izbornika datoteke FirmaStatistika.cs

## ❑ Primjer: WinServis \ FirmaWindowsService – FirmaStatistika.cs [Design]



# Instalacija servisa – ProjectInstaller.cs (1)

- ❑ **Add Installer** funkcija dodaje u projekt datoteku `ProjectInstaller.cs` koja sadrži dvije komponente: `serviceInstaller` i `serviceProcesInstaller`
  - Definiraju postavke servisa (vidi sljedeći slajd)

The screenshot shows the Visual Studio IDE with the `ProjectInstaller.cs` file open in Design view. The design view contains two components: `serviceInstaller1` and `serviceProcessInstaller1`. The Properties window is open, showing the properties for `serviceProcessInstaller1`. The properties are as follows:

(Name)	serviceProcessInstaller1
Account	LocalSystem
GenerateMember	True
HelpText	
Modifiers	Private
Parent	ProjectInstaller

The Properties window also shows the properties for `serviceInstaller1`:

(Name)	serviceInstaller1
DelayedAutoStart	False
Description	
DisplayName	
GenerateMember	True
HelpText	
Modifiers	Private
Parent	ProjectInstaller
ServiceName	FirmaStatistika
ServicesDependedOn	String[] Array
StartType	Manual

# Instalacija servisa – ProjectInstaller.cs (2)

- ❑ Objekti tipa `serviceInstaller` i `serviceProcesInstaller` pozvani su tijekom instalacije servisa
- ❑ Svojstva razreda `serviceInstaller`
  - `Description` – opis servisa
  - `DisplayName` – ime servisa koje će se prikazati korisniku (*friendly name*)
  - `ServiceName` – ime servisa po kojem OS identificirati servis. Obavezno mora biti jednako imenu razreda koji je naslijedio `ServiceBase`
- ❑ Svojstva razreda `serviceProcesInstaller`
  - `Account` – tip korisničkog računa pod kojim se servis izvršava

# Instalacija servisa


- ❑ Nakon dodavanja *installer* datoteke za servis, servis je potrebno instalirati na računalo na jedan od sljedećih načina
  - Korištenjem naredbe *installutil.exe*
    - [http://msdn.microsoft.com/en-us/library/50614e95\(v=vs.110\).aspx](http://msdn.microsoft.com/en-us/library/50614e95(v=vs.110).aspx)
    - Potrebno pokrenuti s administratorskim ovlastima
  - Ili izradom instalacijskog projekta koji kopira izvršne datoteke u instalacijski direktorij te instalira servis na računalo
    - VS 2010 i ranije: Projekt tipa *Install* (RPPP13-dodatak.pdf)
    - VS 2012 i kasnije: Install Shield Limited Edition (potrebno se registrirati i dodatno instalirati)
    - Neki drugi program/alat za izradu instalacija

# Aplikacija za upravljanje Windows servisom

## □ Primjer:

- Za FirmaWindowsService (iz prošlog primjera) izraditi upravljačku aplikaciju koja upravlja radom servisa (*Start, Stop, Pause, Continue*) te omogućava promjenu odredišne lokacije i intervala stvaranja PDF-a.
- Aplikacija ne treba imati formu već samo ikonu u Windows status traci za upravljanje servisom putem kontekstnog izbornika.
- Prilikom pokretanja aplikacija će trebati administratorske dozvole za rad sa servisom

## □ Projekt tipa *Windows Application*

- Primjer:  WinServis \ FirmaServiceManager
- Na formu aplikacije postaviti kontrole:
  - `NotifyIcon`, `ContextMenuStrip` i `FolderBrowserDialog`



# Razred NotifyIcon

❑ Komponenta koja kreira ikonu programa u Windows status traci

❑ Svojstva

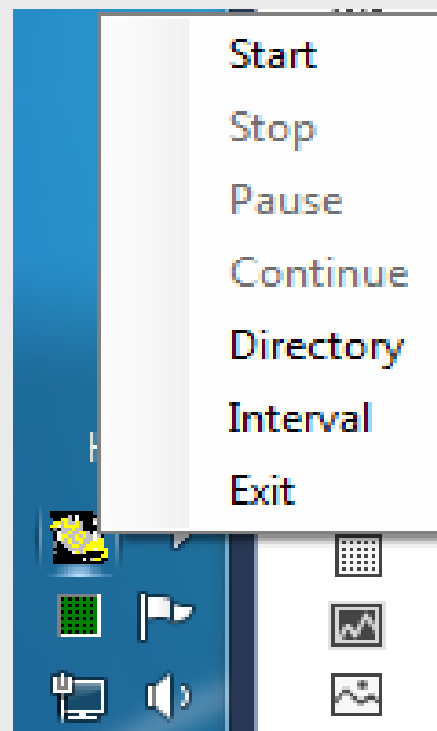
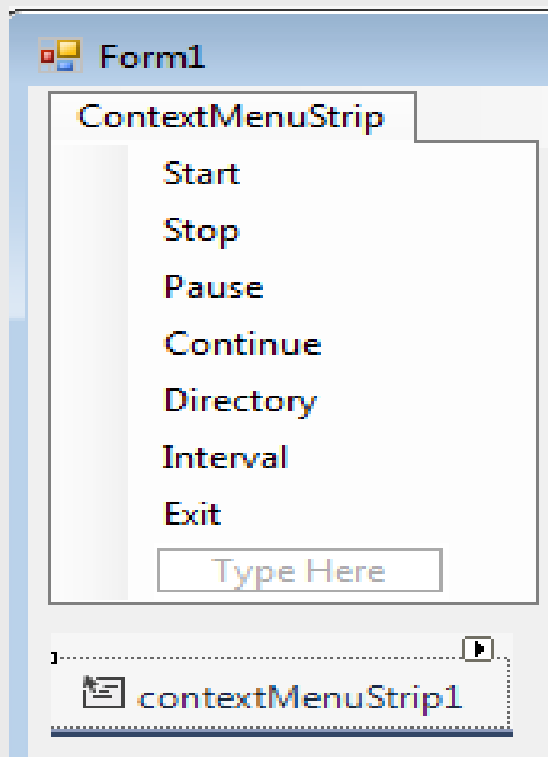
- `ContextMenu` – izbornik koji se prikazuje na desni klik ikone
- `Icon` – ikona komponente
- `Text` – tekst objašnjenja nad ikonom
- `Visible` – oznaka da će ikona biti prikazana
- `BalloonTipIcon` – ikona balončića vezanog uz ikonu
- `BalloonTipText` – tekst unutar balončića
- `BalloonTipTitle` – naslov tekst unutar balončića

❑ Događaji:

- `Click`, `DoubleClick`, `MouseMove`

# Povezivanje NotifyIcon i ContextMenuStrip

## ❑ Primjer: WinServis \ FirmaServiceManager – Form1.cs



## ❑ Primjer: WinServis \ FirmaServiceManager – Form1.cs[Design]

- Povezivanje NotifyIcon kontrole i ContextMenuStrip-a – Svojstvo ContextMenuStrip u NotifyIcon objektu treba postaviti na ContextMenuStrip objekt koji smo dodali na formu

# Razred `ServiceController`

## ❑ `ServiceController` razred omogućava:

- interakciju sa Windows servisima na lokalnom računalu i računalima na koje postoji pristup
  - dohvat dostupnih servisa
  - zadavanje naredbi za pokretanje, zaustavljanje, pauziranje servisa
  - zadavanje korisničkih (*custom*) naredbi

## ❑ Metode

- `GetServices` – dohvaća servise na računalu (`static` postupak)
- `Refresh` – osvježava vrijednosti svih svojstava
- `Start` – pokreće servis
- `Stop` – zaustavlja
- `Pause` – pauzira
- `Continue` – nastavlja

# Primjer ServiceController

## ❏ Primjer: 📁 WinServis \ FirmaServiceManager – Form1.cs

### ■ Upotreba ServiceController objekta

```
private ServiceController firmaService = null;

private bool CheckServiceInstallation()
{
    firmaService = ServiceController.GetService()
                    .Where(s => s.DisplayName
                               == "FirmaStatistika")
                    .FirstOrDefault();
    return firmaService != null;
}
```

# Primjer ServiceController

## ❑ Primjer: WinServis \ FirmaServiceManager – Form1.cs

- Pokretanje procesa (prilikom klika na “*Start*” u kontekstnom izborniku):

```
private void startToolStripMenuItem_Click(
    object s, EventArgs e){
    try
    {
        firmaService.Start();
    }
    catch (Exception ex) {}
    finally
    {
        UpdateServiceStatus();
    }
}
```

# Razred ManagementClass

## ❑ ManagementClass koristi servis WMI (Windows Management Instrumentation):

- Dohvat informacija o OS-u, uređajima, aplikacijama i servisima

## ❑ Primjer: WinServis \ FirmaServiceManager – Form1.cs

- Dohvat lokacije servisa kako bi se mijenjala njegova *.config* datoteka

```
private string GetConfigFilePath() {  
    string fullpath = null;  
    ManagementClass mc = new ManagementClass("Win32_Service");  
    foreach (ManagementObject mo in mc.GetInstances()) {  
        if (mo.GetPropertyValue("Name").ToString() ==  
            "FirmaStatistika"){  
            fullpath = mo.GetPropertyValue("PathName").  
                ToString().Trim('"');  
            break;  
        }  
    }  
    return fullpath + ".config";  
}
```

# Zapisivanje konfiguracije servisa

## ❏ Primjer: WinServis \ FirmaServiceManager – Form1.cs

```
private void directoryToolStripMenuItem_Click... {  
    ExeConfigurationFileMap fileMap = new ExeConfigurationFileMap();  
    fileMap.ExeConfigFilename = GetConfigFilePath();  
    Configuration config = ConfigurationManager.OpenMappedExeConfiguration  
        (fileMap, ConfigurationUserLevel.None);  
    folderBrowserDialog1.SelectedPath =  
        config.AppSettings.Settings["Path"].Value.ToString();  
    DialogResult result = folderBrowserDialog1.ShowDialog();  
    if (result != DialogResult.Cancel) {  
        config.AppSettings.Settings["Path"].Value =  
            folderBrowserDialog1.SelectedPath.ToString();  
        config.Save();  
        //obavijesti servis (dogovorena je poruka 128)  
        if (firmaService.Status == ServiceControllerStatus.Running)  
            firmaService.ExecuteCommand(128);  
    }  
}
```



# Zadaci za vježbu

## ☐ Implementirati Windows servis koji

- implementira WCF ugovor koji pruža podatke o najboljim kupcima u obliku tekstualnih podataka i u obliku PDF izvještaja
  - WCF servis je izložen koristeći TCP protokol

## ☐ Više o windows servisima

- [http://msdn.microsoft.com/en-us/library/y817hyb6\(v=vs.110\).aspx](http://msdn.microsoft.com/en-us/library/y817hyb6(v=vs.110).aspx)



# Programska dokumentacija

---

# Dokumentacija programskog koda

## ❑ XML dokumentacijski komentari

- omogućavaju pisanje dokumentacije koja se odnosi na programski kod
- označavaju se trostrukom kosom crtom : ///
- dodaju se ispred programskog bloka na koji se odnose
- sadrže posebne XML oznake koje opisuju dokumentirani kod

## ❑ Primjer: WinServis \ FirmaWindowsService – FirmaStatistika.cs

```
/// <summary>  
/// Dohvaća najbolje kupce, pridružuje izvještaju i stvara PDF  
/// </summary>  
/// <returns>sadržaj PDF-a izvještaja</returns>  
private byte[] StvoriIzvjesce()
```

## ❑ /doc opcija prevoditelja prilikom prevođenja uključuje traženje XML komentara u kodu te kreiranje dokumentacijske XML datoteke

- Alternativno: kontekstni izbornik : Project properties \ Build \ XML documentation file
- prevođenje proizvede *NazivAsemblija.XML*
- dodavanjem reference na neki dll, uzima se i istoimena xml datoteka (ako postoji) – služi za prikaz dokumentacijskih komentara

# XML oznake

## ❑ XML oznake za komentiranje programskog koda

- `<summary>` - sažetak opisa razreda ili člana
- `<remarks>` - nadopunjuje opis o razredu ili članu
- `<param>` - opis parametra metode
- `<paramref>` - oznaka da se opis odnosi na parametar
- `<permission>` - prava pristupa
- `<returns>` - povratne vrijednosti metode
- `<value>` - opis vrijednosti svojstva
- `<exception>` - iznimke koje mogu biti bačene u opisanom kodu. Sadrži referencu na iznimku.
- `<typeparam>` - opis generičkog razreda ili metode
- `<typeparamref>` - tip parametra kod generičkih razreda ili metoda

# XML oznake (nastavak)

- `<c>` - tekst unutar oznake je kod
- `<code>` - tekst unutar oznake je višeretkovni kod
- `<example>` - primjer
- `<include>` - uključivanje vanjske datoteke s XML dokumentacijom
- `<list>` - lista, nabrojanje elemenata
- `<para>` - paragraf teksta
- `<see>` - omogućava stvaranje linka
- `<seealso>` - link u "See also" sekciji

## □ Za detaljniji opis oznaka pogledati:

- [http://msdn.microsoft.com/en-us/library/5ast78ax\(v=vs.110\).aspx](http://msdn.microsoft.com/en-us/library/5ast78ax(v=vs.110).aspx)

# Primjer XML komentara

❏ **Primjer:**  **WinServis \ FirmaWindowsService \ FirmaStatistika.cs**

```
/// <summary>
/// Service za periodičko stvarnje PDF izvještaja o najboljim kupcima
/// </summary>
public partial class FirmaStatistika : ServiceBase{
    ...
    /// <summary>
    /// Pretvara izvještaj u PDF.
    /// Izvještaj prethodno mora biti povezan s podacima
    /// </summary>
    /// <param name="reportViewer">kontrola koja sadrži izvještaj
    /// koji se želi izvesti u PDF</param>
    /// <returns>PDF sadržaj</returns>
    private static byte[] ExportToPDF(ReportViewer reportViewer)
    {
        ...
    }
}
```

# Generirana XML datoteka

## ❑ Primjer: WinServis \ FirmaWindowsService \ bin \ debug \ FirmaWindowsService.XML

```
<?xml version="1.0"?>
<doc>
  <assembly>
    <name>FirmaWindowsService</name>
  </assembly>
  <members>
    <member name="T:FirmaWindowsService.FirmaStatistika">
      <summary>
        Service za periodičko stvarnje PDF izvještaja o najboljim kupcima
      </summary>
    </member>
    ...
    <member
name="M:FirmaWindowsService.FirmaStatistika.ExportToPDF(Microsoft.Reporting.W
inForms.ReportViewer)">
      <summary>
        Pretvara izvještaj u PDF. Izvještaj prethodno mora biti povezan s
podacima
      </summary>
      <param name="reportViewer">kontrola koja sadrži izvještaj koji se
želi izvesti u PDF</param>
      <returns>PDF sadržaj</returns> ...
    </member>
  </members>
</doc>
```

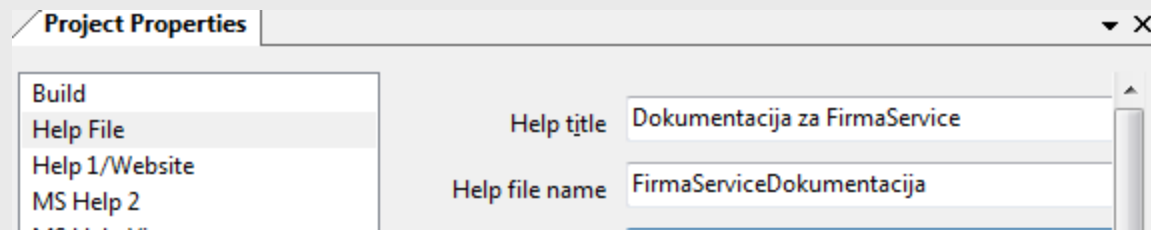
# Izrada *help* datoteke iz XML komentara

- ❑ Generirana XML datoteka nije čitljiva/primjenjiva za korisnika
- ❑ Sandcastle Help File Builder (<http://shfb.codeplex.com/>)
  - GUI za alat Sandcastle za generiranje dokumentacije iz XML komentara
  - interno korišten za izradu dokumentacije .NET frameworka
  - podržava više formata dokumentacije
  - mogućnost integracije s Visual Studiom
  - Instalacija s gornje poveznice
    - *Html Help 1* Compiler (ne treba instalirati podršku za *Help 2*)
    - Microsoft Sandcastle Tools
    - MAML Schema IntelliSense for Visual Studio
    - Sandcastle Help File Builder
    - Sandcastle Help File Builder Visual Studio Package
- ❑ Nakon instalacije *help* datoteke mogu se raditi
  - Koristeći kroz samostalni alat (vidi sljedeći slajd) ili
  - kao novi projekt unutar Visual Studia (File \ New Project \ Documentation \ Sandcastle Help File Builder Project)

# Generiranje dokumentacije pomagalom *Sandcastle*

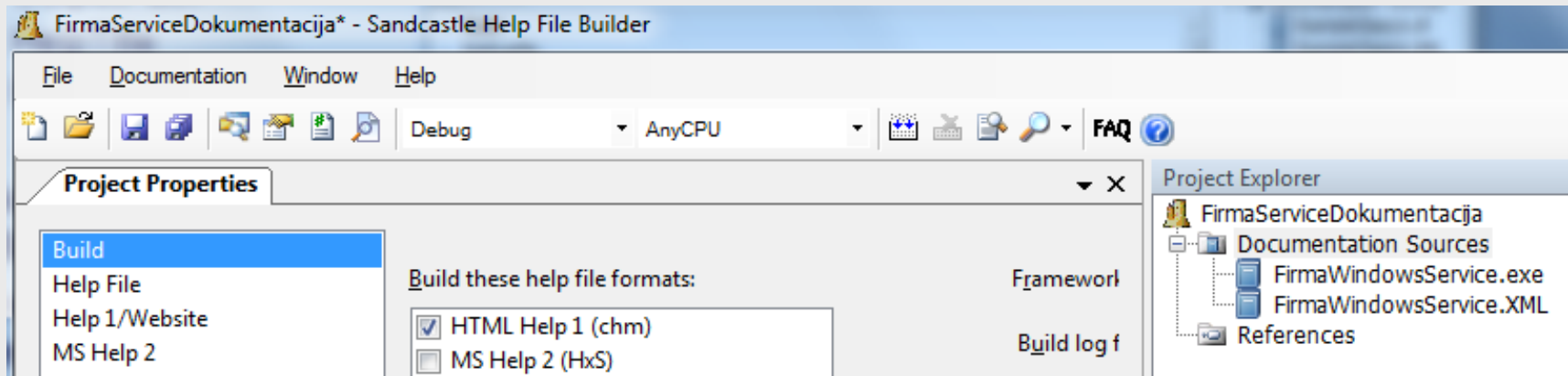
## ❑ File\ New Project, pa odabrati naziv i lokaciju po želji

- Pod *Documentation Sources* dodati datoke za koje treba izraditi dokumenfaciju (dll, exe, xml, sln, proj)
- Postaviti uobičajena svojstva
  - HelpFileFormat
  - OutputPath
  - HelpTitle
  - HtmlHelpName
  - ...



## ❑ Pokrenuti Documentation \ Build Project

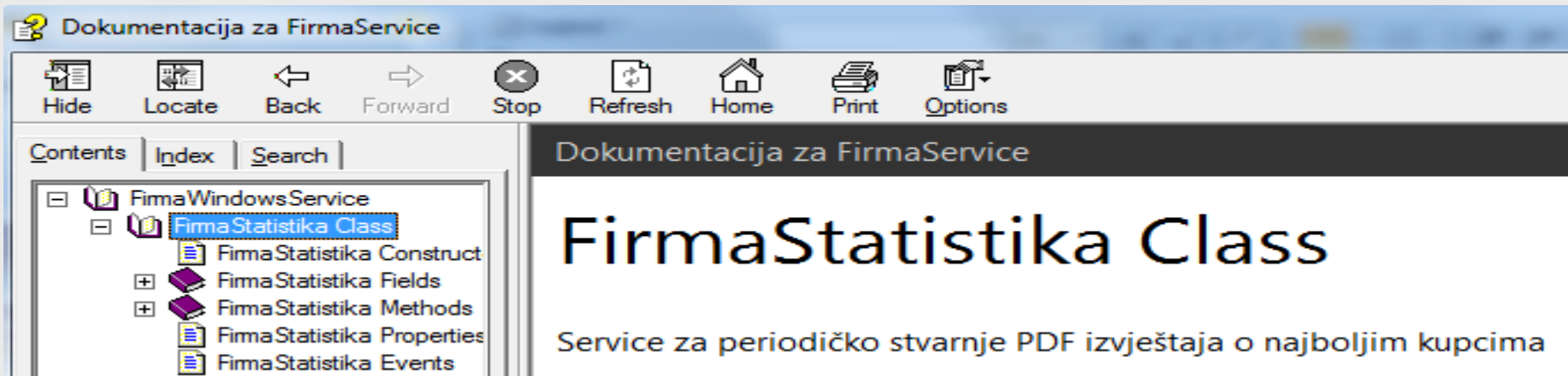
- U ovisno o odabranom formatu generiraju se različiti tipovi datoteka. Najznačajniji su:
  - chm (HtmlHelp1)
  - msch (MSHelpViever) (može se ugraditi u lokalni help)





# Primjer generirane dokumentacije



## ❑ Primjer CHM datoteke (HTMLHelp verzija 1)



## FirmaStatistika Methods

The [FirmaStatistika](#) type exposes the following members.

### ▲Methods

Name	Description
 <a href="#">DoJob</a>	Glavni postupak koji se izvršava istekom vremena na Timeru. Iz konfiguracijske datoteke čita naziv mape u koju će se podaci izvesti (ključ OutputDir). Stvoreni PDF ima sufiks s datumom izvještaja.
 <a href="#">ExportToPDF</a>	Pretrvara izvještaj u PDF. Izvještaj prethodno mora biti povezan s podacima

# Internacionalizacija aplikacija

---

# Internacionalizacija

## ❑ Internacionalizacija =

- globalizacija (mogućnost podrške različitih jezika i postavki)
- + lokalizacija (prijevod aplikacije na konkretni jezik)

## ❑ **Culture** - regionalne postavke

- format datuma, ispis novčanog iznosa...
- Svojstvo Page.Culture (System.Web.UI)

## ❑ **UICulture** – identifikator (govornog) jezika korisničkog sučelja

- određuje koji resursi su uključeni (loaded) na web stranicu
- Svojstvo GlobalizationSection.UICulture (System.Web.Configuration)
- Svojstvo Page.UICulture (System.Web.UI)

## ❑ **Primjeri oznaka jezičnih postavki**

- en-GB : engleski (Ujedinjeno kraljevstvo)
- en-US : engleski (SAD)
- hr-HR : hrvatski (Hrvatska)
- hr-BA : hrvatski (Bosna i Hercegovina)
- ...

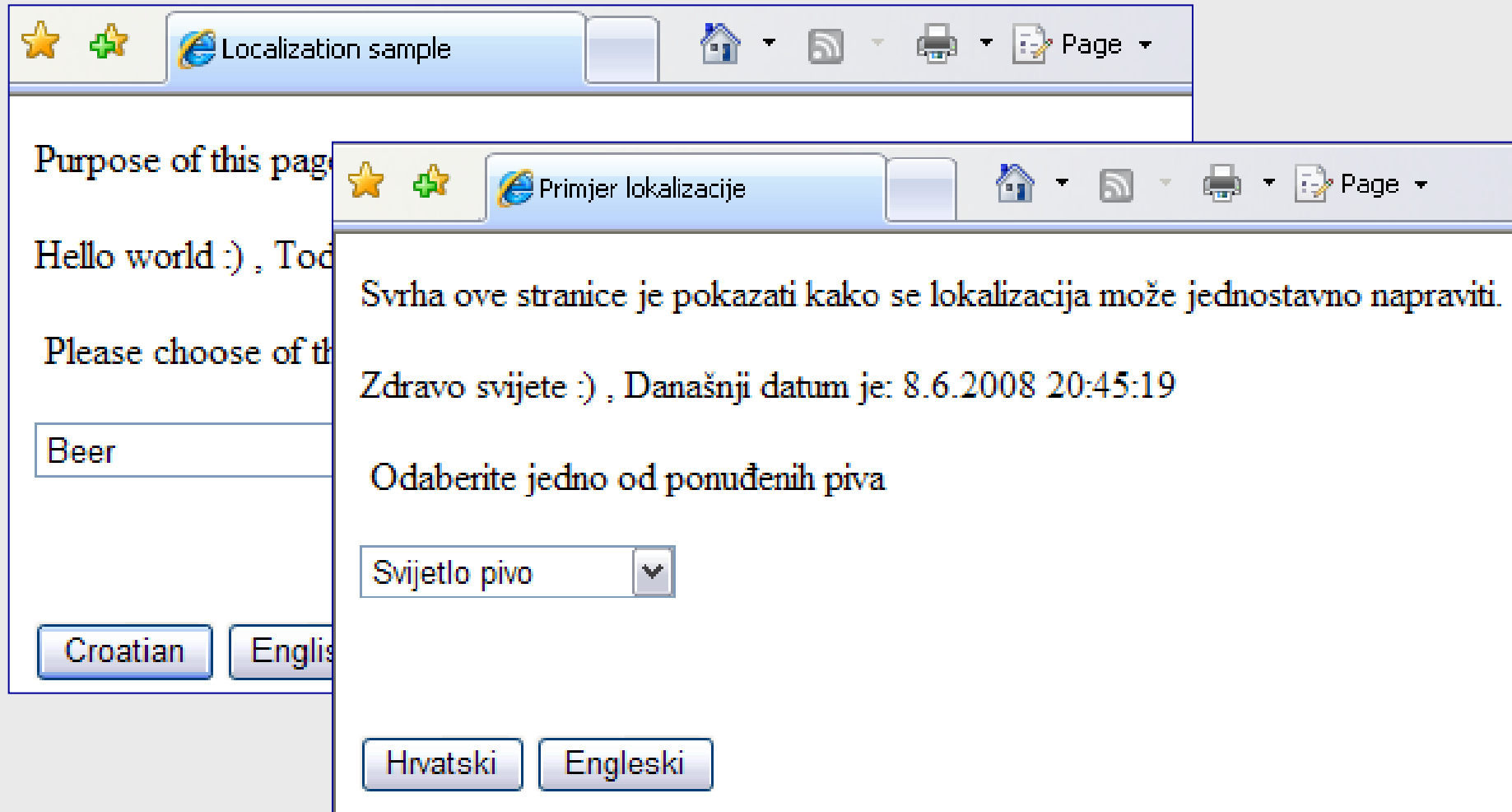
## ❑ **Cjelokupni popis:**

- <http://msdn.microsoft.com/en-us/library/dd318693.aspx>


# Internacionalizacija ASP.NET aplikacija

## ❑ Primjer : Lokalizacija \ LokalizacijaWeb – Default.aspx

```
<%@ Page Language="C#" ... Culture="auto" UICulture="auto"
```



# Internacionalizacija ASP.NET aplikacija

- ❑ **Tekstovi na različitim jezicima pohranjuju se u *Resource* datoteke**
  - Naziv datoteke oblika `naziv.jezicna_postavka.resx`
- ❑ **Globalni resursi**
  - mapa `App_GlobalResources` u korijenu web aplikacije
  - `Poruke.resx` ("Pozdrav svijete") - default
  - `Poruke.en-US.resx` ("Hello world") - prijevod
  - `Poruke.hr-HR.resx` ("Pozdrav svemiru") - opcionalno
- ❑ **Lokalni resursi za pojedinu *aspx* stranicu, ili kontrole (*ascx*)**
  - podmapa `App_LocalResources` u mapi stranice, npr.
  - `Default.aspx.resx`
  - `Default.aspx.en-US.resx`
  - `KorisnickaKontrola.ascx.de-DE.resx`
- ❑ **Nakon što se napravi *Publish Web Site*, moguće je mijenjati samo lokalne resurse.**
- ❑ **Primjer :  Lokalizacija \ LokalizacijaWeb**

# Povezivanje kontrola i datoteka s resursima

- ❑ Internacionalizirati/lokalizirati se mogu samo ASP.NET kontrole

- nije moguće lokalizirati html kontrole

- ❑ Kontrola koja se želi lokalizirati proširuje se atributom **meta:resourcekey**

```
<asp:Label ID="lblOdabir" runat="server"
    meta:resourcekey="lblOdabirResource1"/>
```

- ❑ Za lokalizaciju veće tekstovne cjeline

```
<asp:Localize ID="Localize1" runat="server"
    meta:resourcekey="Localize1Resource1">
```

- ❑ Unutar **resx** datoteke postavljaju se tekstovi za pojedini jezik

```
lblOdabirResource1.Text
```

```
Localize1Resource1.Text
```

- ❑ Pri učitavanju stranice ASP.NET automatski povezuje kontrole i tekstove iz odgovarajuće datoteke s resursima

# Kreiranje datoteke s resursima




## ❑ Automatsko generiranje datoteke s resursima

- *Design view pojedine ascx ili aspx stranice -> Tools -> Generate Local Resource.*
- Sve serverske kontrole se proširuju navedenim atributom `meta:resourcekey` uz prijenos postojećih tekstova u `resx` datoteku.
- Nakon generiranja `resx` datoteke za pojedinu kontrolu, promjene teksta kontrole u dizajnu nemaju efekta dok se ponovo ne generira `LocalResource`.
- Resursi kontrola kojih više nema na stranici ostaju u datoteci.
  - Ne smetaju, ali povećavaju veličinu datoteke i nečitljivost

## ❑ Datoteke za ostale jezike treba ručno napraviti !

# Primjer datoteke s resursima

❑ Primjer :  Lokalizacija \ LokalizacijaWeb – Default.aspx.resx

App_LocalResources/Default.aspx.resx			
abc Strings ▾  Add Resource ▾  Remove Resource    ▾			
	Name ▲	Value	Comment
	btnENResource1.Text	Engleski	
	btnENResource1.ToolTip		
	btnHRResource1.Text	Hrvatski	
	btnHRResource1.ToolTip		
	ddlOdabirResource1.ToolTip		
	lblOdabirResource1.Text	Odaberite jedno od ponuđenih piva	
	lblOdabirResource1.ToolTip		
	lblOdabranaStavkaResource1.Text		
	lblOdabranaStavkaResource1.ToolTip		
	ListItemResource1.Text	Svijetlo pivo	
	ListItemResource1.Value	Svijetlo pivo	



# Lokalizacija web stranice

❑ Lokalizirati se mogu i ostali elementi, npr. elementi padajuće liste

❑ Primjer :  Lokalizacija \ LokalizacijaWeb – Default.aspx

```
<asp:DropDownList ID="ddlOdabir" runat="server"
    meta:resourcekey="ddlOdabirResource1">
    <asp:ListItem meta:resourcekey="ListItemResource1">
        Svijetlo pivo</asp:ListItem>
    <asp:ListItem meta:resourcekey="ListItemResource2">
        Tamno pivo    </asp:ListItem>
    ...
</asp:DropDownList>
<asp:Button ID="btnOdabir" runat="server" Text="Button"
    OnClick="btnOdabir_Click"
    meta:resourcekey="btnOdabirResource1" />
```

# Aktiviranje lokalizacije

## ❑ Postavljanje jednog jezika za sve stranice (*web.config*)

```
<system.web>  
  <globalization culture="en-US" uiCulture="en-US"/>  
</system.web>
```

## ❑ Postavljanje jezika za pojedinačnu stranicu (*stranica.aspx*)

```
<%@ Page ... Culture="en-US" UICulture="en-US" ... %>
```

- primjer: promjena oznake jezika u deklaraciji stranice

## ❑ Dinamička promjena trenutnog jezika za neku stranicu:

- promjena *Culture* – treba prekriti *InitializeCulture* postupak stranice

```
protected override void InitializeCulture() {  
    string culture = Session["Culture"] as string;  
    if (culture != null) {  
        Page.Culture = culture;  
        Page.UICulture = culture;  
    }  
}
```

```
protected void btnEN_Click(object sender, EventArgs e) {  
    Session["Culture"] = "en-US";  
    Response.Redirect(Request.Url.OriginalString);  
}
```

# Dinamička lokalizacija programski

## ❑ Dohvat lokaliziranog teksta programskim kodom ako je

- lokalizirani tekst u lokalnim resursima navedene stranice

```
this.GetLocalResourceObject("Naziv parametra")
```

- lokalizirani tekst u lokalnim resursima neke druge stranice

```
HttpContext.GetLocalResourceObject  
("Virtualni put", "Naziv parametra")
```

- lokalizirani tekst u globalnim resursima

```
this.GetGlobalResourceObject  
("Prefiks resource datoteke", "Naziv parametra")
```

ili

```
Resources.PrefiksResourceDatoteke.NazivParametra
```

```
lblPozdrav.Text = GetGlobalResourceObject(  
    "Poruke", "PozdravnaPoruka").ToString();  
lblPozdrav.Text = Resources.Poruke.PozdravnaPoruka;  
  
lblDatum.Text = HttpContext.GetLocalResourceObject(  
    "~/Ostalo", "TekstZaDatum").ToString();
```

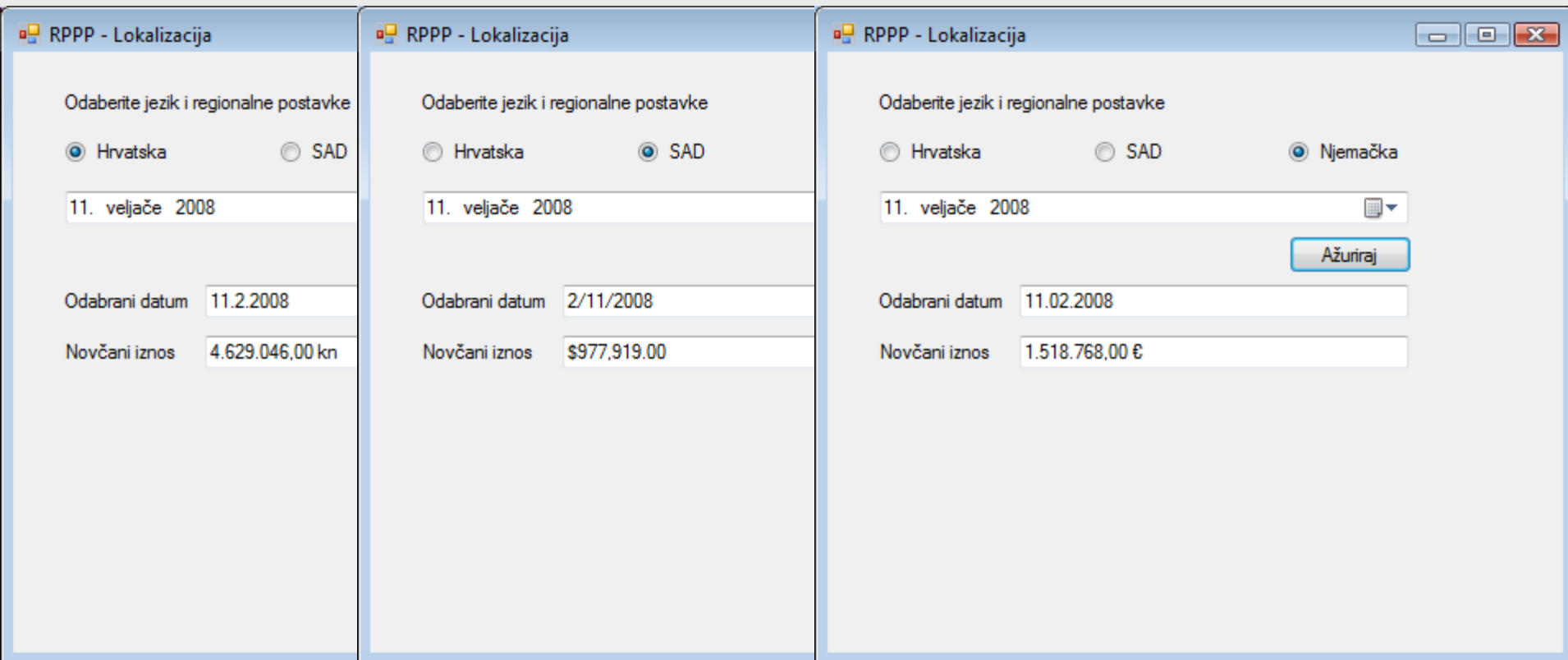
# Internacionalizacija MVC aplikacije

- ❑ Slično internacionalizaciji ASP.NET WebForms aplikacije
- ❑ Globalni resursi u mapi `App_GlobalResource`
  - Dostupni u pogledu s `@Resources.NazivResourceDatoteke.Kljuc`  
Npr. za datoteku *Globalni.resx* i ključ *Poruka* u pogledu se navodi `@Resources.Globalni.Poruka`
- ❑ Lokalni resursi u `App_LocalResources` u istoj mapi gdje je konkretni pogled
  - Dostupni u pogledu s `@HttpContext.GetLocalResourceObject(this.VirtualPath, "Kljuc")`  
npr. za *Index.cshtml* i resurse u *App\_LocalResources\Index.cshtml.resx* i ključ *Poruka*  
`@HttpContext.GetLocalResourceObject(this.VirtualPath, "Poruka")`
- ❑ Postavljanje jezika vrši se
  - za pojedini upravljač prekrivanjem postupka *ExecuteCore*
  - za sve upravljače u datoteci *Global.asax* u postupku *Application\_BeginRequest*

# Internacionalizacija Windows aplikacija

## ❑ Primjer : Lokalizacija \ LokalizacijaApp

- Primjer ispisuje datum i slučajno odabrani novčani iznos u formatu koji ovisi o trenutno odabranim regionalnim postavkama
- Ispis ovisi o vrijednosti `CurrentCulture` (za razliku od `CurrentUICulture` koji definira jezik sučelja)



The image displays three side-by-side screenshots of a Windows application window titled "RPPP - Lokalizacija". Each screenshot shows the same interface with different regional settings selected, demonstrating how the application's output (date and currency) changes based on the selected culture.

Selected Region	Selected Language	Selected Date Format	Selected Currency
Hrvatska	Hrvatska	11. veljače 2008	4.629.046,00 kn
SAD	SAD	11. veljače 2008	\$977,919.00
Njemačka	Njemačka	11.02.2008	1.518.768,00 €

Each screenshot includes a title bar, a header "Odaberite jezik i regionalne postavke", radio buttons for "Hrvatska", "SAD", and "Njemačka", a date input field, a date label "Odabrani datum", and a currency label "Novčani iznos". The third screenshot also features an "Ažuriraj" button.

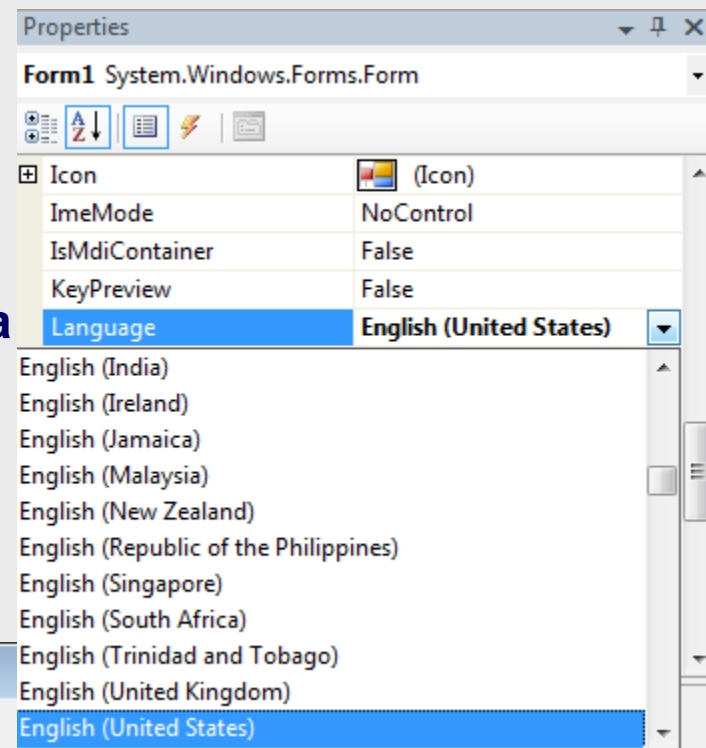
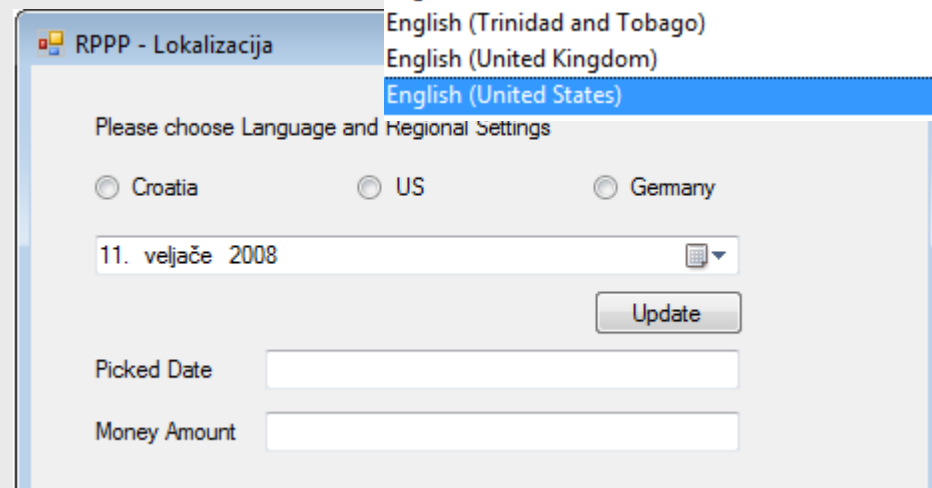
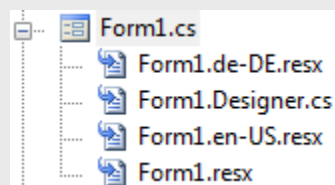
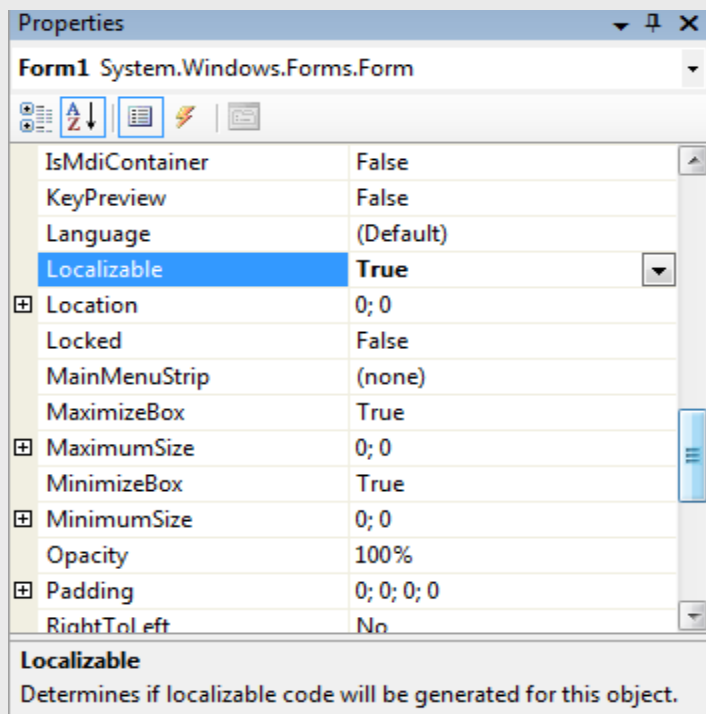
# Internacionalizacija Windows aplikacija

- ❑ Promjena regionalnih postavki za formate - promjenom svojstva
  - `Application.CurrentCulture` ili
  - `Thread.CurrentThread.CurrentCulture`
- ❑ Navednom svojstvu pridružuje se željeni `CultureInfo` iz prostora imena `System.Globalization`.

```
private void btnAzuriraj_Click(object sender, EventArgs e) {  
    if (rbHrvatska.Checked) {  
        Application.CurrentCulture = new CultureInfo("hr-HR");  
    }  
    else if (rbSAD.Checked) {  
        Thread.CurrentThread.CurrentCulture = new CultureInfo("en-US");  
    }  
    else if (rbNjemacka.Checked) {  
        Application.CurrentCulture = new CultureInfo("de-DE");  
    }  
    tbDatum.Text = dateTimePicker1.Value.ToShortDateString();  
    tbNovcaniIznos.Text = new Random().Next(1000, 5000000).ToString("c");  
}
```

# Internacionalizacija Windows kontrola

- ❑ Želi li se određena windows aplikacija, odnosno tekstovi na pojedinoj formi lokalizirati, potrebno je postaviti svojstvo `Localizable` na `True`
- ❑ Odabirom pojedinog jezika pod svojstvom *Language*, otvara se izgled forme s natpisima postavljenim za odabrani jezik (u početku isti kao i za pretpostavljeni jezik)
- ❑ Pri prvoj promjeni stvara se nova *resx* datoteka



# Internacionalizacija pojedinih kontrola

## ☐ Nije moguće lokalizirati sve kontrole:

- npr. `DateTimePicker` i `MonthCalendar` se ne mogu lokalizirati te se uvijek prikazuju u ovisnosti o jezičnim i regionalnim postavkama na računalu

## ☐ Aktiviranje lokalizacije kontrola na formi mora se obaviti u konstruktoru forme, prije inicijalizacije samih kontrola


```
public Form1() {  
    Thread.CurrentThread.CurrentUICulture =  
        new CultureInfo("hr-HR");  
    InitializeComponent();  
}
```

## ☐ Naknadna promjena nema (automatskog) utjecaja na jezik u formi!

## ☐ Ako se prije inicijalizacije ne postave jezične postavke, koriste se pretpostavljene (definirane u postavkama računala).



# Internacionalizacija Windows aplikacija

- ❑ Ako je potrebno prikazati različiti prijevod nezvezan uz pojedinu kontrolu na formi, moguće je koristiti zasebne datoteke s resursima
  - (Add New Item → Resources File)
  - Primjer :  Lokalizacija \ LokalizacijaApp \ Tekstovi \ \*.resx
  
- ❑ Za razliku od web-aplikacije, nakon izgradnje aplikacije, tekstove za različite jezike nije moguće mijenjati (nalaze se unutar dll datoteka)

# Internacionalizacija Windows aplikacija

## ❑ Pristup pojedinom tekstu unutar datoteke s resursima

- razredom `ResourceManager` iz prostora imena `System.Resources`.

## ❑ Uobičajeni konstruktor razreda `ResourceManager` je oblika:

```
ResourceManager(string baseName,  
                 System.Reflection.Assembly assembly)
```

- Za `baseName` se koristi string koji se formira iz naziva aplikacije i putanje do `resx` datoteke pri čemu se za odvajanje kazala koristi točka
- Za *assembly* se koristi vlastiti *assembly* navedene forme

## ❑ Dohvat pojedine vrijednosti se vrši pozivom postupka `GetString`

- Drugi parametar je jezična postavka za koju se obavlja dohvat.
- Ako se izostavi, koristi se trenutna vrijednost **sučelja** (`CurrentUICulture`)

```
ResourceManager rm =  
    new ResourceManager("LokalizacijaApp.MojiResursi",  
                        typeof(Form1).Assembly);  
//tbProbnaPoruka.Text = rm.GetString("Poruka");  
tbProbnaPoruka.Text = rm.GetString("Poruka", culture);
```