C# i WPF

U okviru vežbi iz predmeta *Inženjerstvo upotrebljivosti u infrastrukturnim sistemima*, biće korišćen programski jezik **C#** u okviru razvojnog okruženja **Visual Studio**. Ovo će biti realizovano kroz **WPF (*Windows Presentation Foundation*) Framework**.

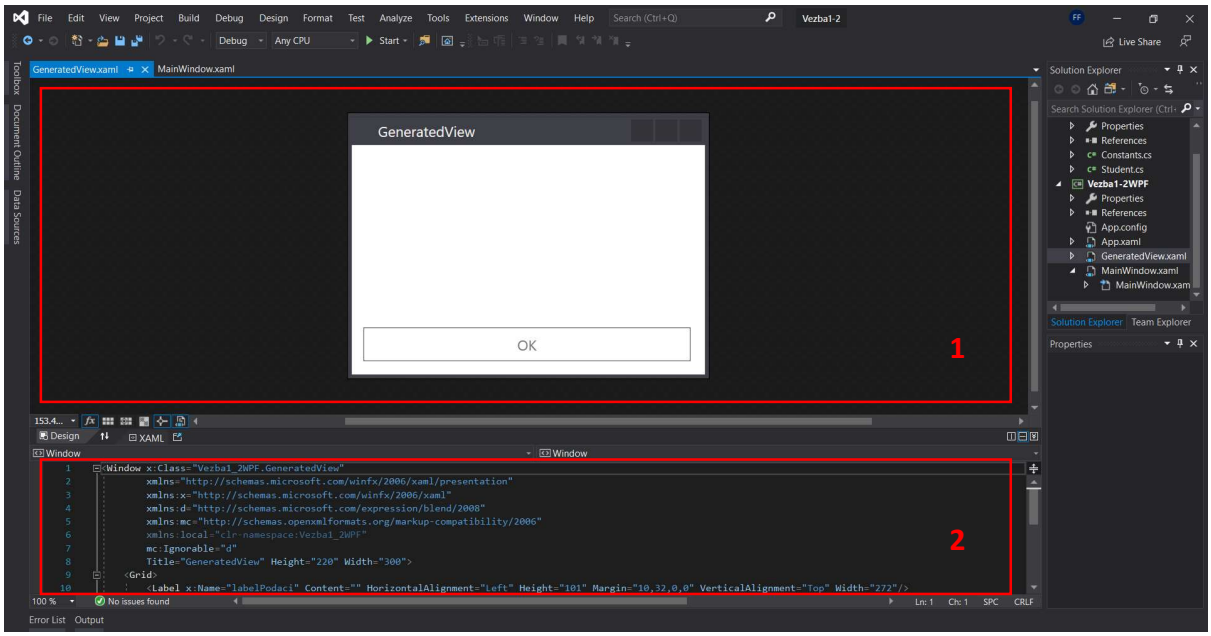
WPF je **UI (*User Interface*) Framework** u kojem se kreiraju *desktop klijentske aplikacije*. Sadrži široki skup karakteristika za razvoj aplikacija, počev od njenog modela i potom resursa, kontrola, grafika, rasporeda, Data Binding-a itd.

WPF je deo ***.NET***-a, tako da se funkcionalnost definiše kroz **C#** kod, dok se izgled (dizajn) aplikacije definiše **XAML (*eXtensible Application Markup Language*)** kodom. Činjenica da je XAML markap jezik znači da će se njegovi elementi deklarisati pomoću *tagova*. Tag predstavlja rezervisanu reč koja se definiše između znakova „<“ i „>“ kako je prikazano na **Listingu 1**.

```
<Button Click="ButtonClick">Show updates</Button>
```

Listing 1. Primer taga u WPF XAML kodu

Kada se pokrene WPF projekat u okviru **Visual Studija**, korisniku/programeru se najpre prikazuje ***Designer*** deo projekta (**Slika 1**). U okviru ovog dela razvojne aplikacije, korisnik/programer definište izgled aplikacije kroz pisanje XAML koda.



Deo označen brojem 1 predstavlja izgled same aplikacije na osnovu unesenog XAML koda, dok deo označen brojem 2 predstavlja prostor gde se unosi XAML kod.

U okviru WPF-a, osnovna gradivna jedinica aplikacije je **prozor (*Window*)**. Svaka instanca novog prozora potrebnog da se formira aplikacija nasleđuje baznu klasu *Window*, koja sa sobom nosi osnovnu implementaciju.

Kada se kreira novi projekat, već je kreiran „glavni prozor“ (*MainWindow*) koji će biti prikazan kada se pokrene sama aplikacija. Pošto na samom prozoru nema ničega, na njega se mogu dodati elementi interakcije (jednim imenom se zovu ***kontrole***).

Svaki prozor ima svoje zaglavlje u kojem se definiše njegovo ime, korišćeni namespace-ovi, naslov koji piše na naslovnoj liniji prozora i dimenzije samog prozora (**Listing 2**).

```
<Window x:Class="Vezba1_2WPF.GeneratedView"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:Vezba1_2WPF"
        mc:Ignorable="d"
        Title="GeneratedView" Height="220" Width="300">
    <Grid>

    </Grid>
</Window>
```

Listing 2. Početna definicija prozora

Prozor se deklariše Window tagom. Postoje tri tipa tagova:

1. Otvarajući, piše se tako da je rezervisana reč između znakova „<“ i „>“ i zahteva postojanje zatvarajućeg taga kojim se kompletira definicija objekta date klase.
2. Zatvarajući, piše se identično otvarajućem tagu, samo što posle znaka „<“ sledi „/“. Zahteva da postoji otvarajući tag pre njega.
3. Samozatvarajući, piše se identično otvarajućem, samo što se pre znaka „>“ piše „/“.

Navođenjem taga, definiše se jedan, konkretan objekat klase čija je rezervisana reč u njemu.

Kada se dizajnira izgled aplikacije u WPF-u, ona ima strukturu stabla. Svaki prozor ima svoje stablo, u čijem korenu je tag <*Window*>. Kada se kreira prozor, struktura stabla postoji jer je između otvarajućeg i zatvarajućeg taga koji definiše prozor smešten par <*Grid*> tagova. Na ovaj način, ovde instancirani objekat klase **Grid** je „dete“ objektu klase **Window** (**Listing 2**). Klasa Grid predstavlja neku vrstu „kontejnera“, što znači da se u njoj mogu ređati različite kontrole koje služe za interakciju.

Prilikom pokretanja aplikacije, da bi se prikazao izgled prozora poziva se funkcija *InitializeComponent()*, koja je obavezna u konstruktoru svakog prozora (njegovom xaml.cs fajlu). Njen zadatak je da prikaže sve što se nalazi unutar <*Window*> tagova.

Kao što je to slučaj u objektnom programiranju, sve klase, pa i ove koje se koriste za dizajniranje u WPF-u imaju *property*-je (u nastavku, *svojstva*). U XAML kodu dizajna, svojstva se navode unutar otvarajućeg taga, kao što je u **Listingu 2** za prozor definisan naslov (*Title*), visina (*Height*) i širina (*Width*).

Kada je pred korisnikom/programerom prazan dizajn prozora, on tu može Drag&Drop tehnkiom dizajnirati izgled ili navoditi unosom XAML koda. U Visual Studiju, izborom opcije **View -> Toolbox**, sa leve strane će se pojaviti opcije gde se mogu izabrati kontrole koje se mogu dodati na prozor (**All WPF Controls**).

Najvažnije kontrole za početak rada u WPF-u će biti: ***Label***, ***TextBox*** i ***Button***. Dodatne kontrole koje će biti objašnjene ovde će biti ***RadioButton*** i ***ComboBox***.

Navođenjem taga *<Label>* (i njegovog zatvarajućeg para, ili ako je samozatvarajući) definiše se konkretan objekat klase **Label**. Ova klasa modeluje kontrolu za prikaz teksta. Njeno najvažnije svojstvo je ***Content*** (tipa string, što znači da se sa njim može manipulisati kao sa svakim drugim stringom), kojim se definiše tekstualni sadržaj same labele. Ovo je prikazano u **Listingu 3**.

Kada se na dizajn doda kontrola za koju je moguće menjati sadržaj iz C# koda, daje joj se i ime (svojstvo ***Name***). Ovo ime odgovara imenu promenljive, odnosno po kom imenu joj se može pristupiti iz C# koda i na standardan način pristupa svojstvima, promeniti sadržaj nekog od njih.

```
<Label x:Name="labelIme" Content="Ime:" HorizontalAlignment="Left" Height="28"
Margin="10,78,0,0" VerticalAlignment="Top" Width="116"/>
```

Listing 3. Prikaz XAML koda za definiciju labele.

Kontrolama se položaj i dimenzije mogu zadavati direktno iz XAML koda ili jednostavnim prevlačenjem i smanjivanjem/povećavanjem preko ugaonih tački po površini prozora. U WPF-u se položaj svega određuje u odnosu na gornji levi ugao prozora, pa je tako definisano i svojstvo **Margin**, koje prikazuje pomeraj u pikselima u odnosu na gornju (10) i levu (78) ivicu (preostale dve nule su pomeraj od donje i desne, koji se ne mora zadavati).

Klasa **TextBox** modeluje polje za unos teksta. Pored imena, zarad izvlačenja sadržaja koji je upisan u C# kodu, glavno svojstvo za **TextBox** je ***Text*** *(isto tipa string)*, u kome je navedeno šta je uneseno u polju.

Klasa **Button** modeluje dugme na koje se može kliknuti. Dugme će biti glavni interakcioni element na početku rada u WPF-u. Prilikom definisanja dugmeta, potrebno mu je dati ime, a da bi se definisalo šta na njemu piše, definiše se svojstvo ***Content***. Pošto je ideja da se na dugme može kliknuti, što bi trebalo da pozove neku funkcionalnost. Pored definisanja svojstava, dugmetu, ali i svim ostalim kontrolama se mogu dodeliti funkcije koje reaguju na različite događaje (*event handler*).

Za dugme, za sada je najvažniji **Click** događaj. Da bi se on definisao, potrebno je u XAML delu koda uneti među ostalim svojstvima ključnu reč „Click“ i posle znaka „=“ pod navodnicima navesti ime funkcije koja će odreagovati na klik, ili dozvoliti Visual Studiju da je on sam, automatski generiše (**Listing 4**). Nakon toga, u xaml.cs C# kodu, pojaviće se metoda u kojoj se u C# programskom jeziku definiše funkcionalnost koja će se pozvati prilikom klika na dato dugme (**Listing 5**).

```
<Button x:Name="buttonGenerisi" Content="Generisi" HorizontalAlignment="Left"
Height="28" Margin="10,330,0,0" VerticalAlignment="Top" Width="157" Background="White"
Foreground="#FF707070" Click="buttonGenerisi_Click"/>
```

Listing 4. Prikaz XAML koda za definiciju dugmeta.

```
private void buttonGenerisi_Click(object sender, RoutedEventArgs e)
{

}
```

Listing 5. Prikaz C# koda zaglavlja metode koja reaguje na Click događaj.

U primeru za prvu nedelju vežbi, ova metoda (u MainWindow.xaml.cs) se menja tako da se pri kreiranju novog objekta klase Student, uzimaju vrednosti iz **TextBox**-eva (primer: *textBoxIme.Text*), koje se prosleđuju kao vrednosti za polja Ime i Prezime.

RadioButton klasa modeluje kontrolu koja omogućava da se od ponuđenih opcija izabere isključivo jedna, tako što korisnik označi opciju koju želi tako što označi dati RadioButton. Kada se posmatraju njenova svojstva (**Listing 6**), bitno je ime, kako bi bilo moguće proveriti stanje drugih svojstava iz C# koda. Svojstvo ***Content*** određuje šta piše pored polja na koje treba da se klikne da bi se RadioButton označio, ***IsChecked*** definiše da li je taj konkretan RadioButton označen ili ne.

Kada se definiše više RadioButton-a, oni nisu međusobno isključivi (kada se izabere jedan, ne poništavaju se ostali). Da bi se postigao taj efekat, potrebno je definisati svojstvo ***GroupName***, kojim se određuje koji RadioButton-i pripadaju kojoj grupi i time, oni iz iste grupe postaju međusobno isključivi.

```
<RadioButton x:Name="radioButtonM" IsChecked="True" Content="Muski" GroupName="Pol"
HorizontalAlignment="Left" Height="16" Margin="103,202,0,0" VerticalAlignment="Top"
Width="127"/>
```

Listing 6. Prikaz XAML koda za definiciju RadioButton-a.

ComboBox klasa modeluje kontrolu koja odgovara „drop down listi“ odnosno od opcija ponuđenih unutar liste nudi mogućnost izbora jedne. Ova klasa kao najvažnija svojstva ima izvor podataka za ponuđene opcije, svojstvo ***ItemsSource***, koje se može definisati i iz XAML koda (biće objašnjeno na nekom od narednih termina) i iz C# koda (ako je **ComboBox**-u u XAML-u zadato ime) i svojstvo ***SelectedItem***, koje daje indikaciju koja je opcija izabrana.

```
<ComboBox x:Name="comboBoxSmer" HorizontalAlignment="Left" Height="28"
Margin="103,256,0,0" VerticalAlignment="Top" Width="229"/>


comboBoxSmer.ItemsSource = new List<String>() {"Primenjeno softversko inzenjerstvo",
"Animacija u inzenjerstvu", "Racunarstvo i automatika", "Zastita zivotne sredine"};
```

Listing 7. Prikaz XAML koda za definiciju ComboBox-a i C# koda za definiciju opcija.

Da bi aplikacija mogla da sadrži i koristi više prozora, potrebno je najpre dodati ih u projekat. U Visual Studiju, da bi se dodao novi prozor u projekat, potrebno je kliknuti desnim tasterom miša na projekat

i izabrati opciju Add -> Window(WPF)... Pozoru se daje ime i to će biti ime klase tog prozora, tako da se pri njegovom pozivanju, instancira objekat klase imena koje je navedeno pri njegovom kreiranju.

U primeru za prvu nedelju vežbi je napravljen novi prozor imena GeneratedView. Kako bi se on prikazao mora se instancirati kako je prikazano u Listingu 8.

```
GeneratedView newWindow = new GeneratedView(novi);
newWindow.ShowDialog();
```

Listing 8. Instanciranje novog prozora i njegovo prikazivanje

Kada se doda novi prozor, on je identičan „glavnom“ kada se napravi novi projekat. Da bi se prikazao na ekranu, nakon svog instanciranja, potrebno je pozvati i metodu *Show()* ili *ShowDialog()*. Metoda *Show* će prozor prikazati tako da se korisnik klikom na glavni prozor može na njega vratiti, dok će metoda *ShowDialog* sprečiti korisnika da radi bilo šta u glavnom prozoru dok ne izađe iz novootvorenog prozora.

Kao i svaka druga klasa, prozor može imati više konstruktora, odnosno da postoji konstruktor sa parametrima. Kako svi objekti klase prozor nasleđuju baznu klasu Window, potrebno je da u svakoj varijanti svog konstruktora poziva metodu *InitializeComponent()*, jer se u suprotnom neće prikazati.

Da bi se izašlo iz prozora potrebno je pozvati metodu *Close()*. Primer ovih metoda se može naći u klasi ***GeneratedView.xaml.cs*** u okviru Visual Studio projekta za prvu nedelju vežbi.

Još jedan važan element interakcije su događaji GotFocus i LostFocus, koje poseduje svaka kontrola. Oni predstavljaju događaj kada je kontrola „u fokusu“, odnosno, kada korisnik nju koristi za interakciju. U slučaju **TextBox**-a, to će biti kada se na njega klikne kako bi se uneo neki tekst. Kontrola gubi fokus onog trenutka kada neka druga dobije fokus. Na ove događaje se reaguje isto kao i na klik, a u slučaju primera iz prve nedelje vežbi, to je iskorišćeno da se u slučaju ostavljanja **TextBox**-a praznim (njegovo svojstvo ***Text*** je prazan string) u njemu upiše *place-holder* (primer: „ovde unesite ime“).

Sve ove funkcionalnosti su iskorišćene da se kreira primer za prvu nedelju vežbi koji prikazuje formu za unos studenata, gde kada se unesu podaci određenog studenta, u novom prozoru se prikazuju ti isti podaci.