|  |  |  |
| --- | --- | --- |
| Description: Description: Description: Description: http://www.tfzr.uns.ac.rs/Content/files/0/ZnakUniverziteta.bmp | **Univerzitet u Novom Sadu**  **Tehnički fakultet „Mihajlo Pupin“**  **Zrenjanin** | Description: Description: Description: Description: http://www.tfzr.uns.ac.rs/Content/files/0/Tehnicki_Fakultet_Mihajlo_pupin_Zrenjanin%281%29s.bmp |

**Predmet: Programski prevodioci**

**Tip Seminarskog rada: Treći (3)**

**Tema: Desktop aplikacija za evidenciju iznajmljivanja muzičke opreme**

|  |  |
| --- | --- |
| Profesor: doc.dr Ljubica Kazi  Asistent: Vuk Amižić | Student: Vuk Đaković  Broj indeksa: SI 3/20 |

**Zrenjanin, 2023.**

# SADRŽAJ

[1. UVOD 3](#_Toc145030991)

[2. KORISNIČKO UPUTSTVO 4](#_Toc145030992)

[2.1 Prikaz izgleda ekrana u NetBeans razvojnom okruženju na Java programskom jeziku 4](#_Toc145030993)

[2.2 Prikaz izgleda ekrana u Visual Studio razvojnom okruženju na C# programskom jeziku 9](#_Toc145030994)

[3. OPIS IMPLEMENTACIJE 14](#_Toc145030995)

[3.1 Delovi koda u Java programskom jeziku 14](#_Toc145030996)

[3.2 Delovi koda u C# programskom jeziku 19](#_Toc145030997)

[3.3 Objašnjenje ključnih delova koda 24](#_Toc145030998)

[4. GREŠKE PROGRAMSKOG KODA 25](#_Toc145030999)

[4.1 Greške u Java programskom jeziku 25](#_Toc145031000)

[4.2 Greške koda u C# programskom jeziku 31](#_Toc145031001)

# UVOD

Aplikacija za evidenciju iznajmljivanja muzičke opreme omogućava jednostavno upravljanje bazom podataka koja sadrži podatke o tipu, proizvodjaču, nazivu I ceni opreme, kao I datum iznajmljivanja I datum vraćanja opreme. Podaci se mogu unositi, menjati I brisati po potrebi, a mogu se i prikazivati u tabeli prema određenim filterima.

Program je kreiran u dva različita razvojna okruženja i programska jezika i u ovom radu će se prikazati postupak i tipovi grešaka koji se mogu pojaviti prilikom izrade.

Baza podataka za WPF C# aplikaciju u Visual Studio okruženju je kreirana u SQL Server Management Studio, dok je baza za Java aplikaciju u NetBeans razvojnom okruženju kreirana uz pomoć XAMPP Control Panel-a u okviru MySQL-a u phpMyAdmin-u.

# KORISNIČKO UPUTSTVO

## Prikaz izgleda ekrana u NetBeans razvojnom okruženju na Java programskom jeziku

Prilikom pokretanja programa, prikazuje se glavni prozor na kojem se nalaze text i combo box-ovi za unos podataka, kao i dugmad Dodaj, Izmeni, Obriši i Filtriraj. Ispod navedenih stavki se nalazi tabela za prikaz svih unetih podataka.

A screenshot of a computer

Description automatically generated

*Slika 1. Izgled glavnog prozora programa u NetBeans IDE*

Kada korisnik popuni prazna polja i pritisne dugme Dodaj, podaci ce se uneti u bazu podataka i prikazati u tabeli ispod.

A screenshot of a computer

Description automatically generated

*Slika 2. Izgled glavnog prozora pre dodavanja nove opreme u bazu podataka*

A screenshot of a computer

Description automatically generated

*Slika 3. Izgled prozora nakon dodavanja nove opreme*

Podaci u tabeli se mogu menjati tako što se pritiskom miša na željenu opremu ona izabere i zatim se novi podaci unesu u data polja i pritisne dugme Izmeni.

A screenshot of a computer

Description automatically generated

*Slika 4. Izgled glavnog prozora pre izmene podatka*

*A screenshot of a computer

Description automatically generated*

*Slika 5. Izgled glavnog prozora nakon izmene podatka*

Kada se podatak iz tabele izabere i pritisne se dugme Obriši, isti će biti izbrisan iz baze podataka i neće se prikazivati u tabeli.

A screenshot of a computer

Description automatically generated

*Slika 6. Izgled glavnog prozora pre brisanja podatka*

*A screenshot of a computer

Description automatically generated*

*Slika 7. Izgled glavnog prozora nakon brisanja podatka*

Primenom željenog filtera i pritiskom na dugme Filtriraj, podaci u tabeli će se prikazivati u skladu sa izabranim filterom.

A screenshot of a computer

Description automatically generated

*Slika 8. Izgled tabele nakon primenjenog filtera*

## Prikaz izgleda ekrana u Visual Studio razvojnom okruženju na C# programskom jeziku

Prilikom pokretanja programa, prikazuje se glavni prozor na kojem se nalaze text i combo box-ovi i dva date picker-a za unos podataka, kao i dugmad Dodaj, Izmeni, Obriši i Filtriraj. Ispod navedenih stavki se nalazi tabela za prikaz svih unetih podataka.

A screenshot of a computer screen

Description automatically generated

*Slika 9. Izgled glavnog prozora programa u Visual Studio IDE*

Kada korisnik popuni prazna polja i pritisne dugme Dodaj, podaci ce se uneti u bazu podataka i prikazati u tabeli ispod.

A screenshot of a computer

Description automatically generated

*Slika 10. Izgled glavnog prozora pre dodavanja nove opreme u bazu podataka*

A screenshot of a computer

Description automatically generated

*Slika 11. Izgled prozora nakon dodavanja nove opreme*

Podaci u tabeli se mogu menjati tako što se pritiskom miša na željenu opremu ona izabere i zatim se novi podaci unesu u data polja i pritisne dugme Izmeni.

A screenshot of a computer

Description automatically generated

*Slika 12. Izgled glavnog prozora pre izmene podatka*

A screenshot of a computer

Description automatically generated

*Slika 13. Izgled glavnog prozora nakon izmene podatka*

Kada se podatak iz tabele izabere i pritisne se dugme Obriši, isti će biti izbrisan iz baze podataka i neće se prikazivati u tabeli.

A screenshot of a computer

Description automatically generated

*Slika 14. Izgled glavnog prozora pre brisanja podatka*

A screenshot of a computer

Description automatically generated

*Slika 15. Izgled glavnog prozora pre brisanja podatka*

Primenom željenog filtera i pritiskom na dugme Filtriraj, podaci u tabeli će se prikazivati u skladu sa izabranim filterom.

*A screenshot of a computer

Description automatically generated*

*Slika 16. Izgled tabele nakon primenjenog filtera*

# OPIS IMPLEMENTACIJE

## Delovi koda u Java programskom jeziku

|  |
| --- |
| private void prikazTabele() {  try {  Class.forName("com.mysql.jdbc.Driver");  Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/bazaiznajmljivanjemuzickeopreme","root","");  PreparedStatement ps = con.prepareStatement("SELECT \* FROM Oprema");  ResultSet rs = ps.executeQuery();  java.sql.ResultSetMetaData stData = rs.getMetaData();  int q = stData.getColumnCount();  DefaultTableModel RecordTable = (DefaultTableModel) Tabela.getModel();  RecordTable.setRowCount(0);  while (rs.next()) {  Vector<Object> columnData = new Vector<>();  int i;  for (i = 1; i <= q; i++) {  columnData.add(rs.getInt("ID\_opreme"));  columnData.add(rs.getString("Tip"));  columnData.add(rs.getString("Proizvodjac"));  columnData.add(rs.getString("Naziv"));  columnData.add(rs.getInt("Cena"));  columnData.add(rs.getString("Datum\_iznajmljivanja"));  columnData.add(rs.getString("Datum\_vracanja"));  }  RecordTable.addRow(columnData);  }  } catch (Exception ex) {  Logger.getLogger(muzickaoprema.class.getName()).log(Level.SEVERE, null, ex);  }  } |

*Listing 1. Kod za prikaz elemenata u tabeli*

|  |
| --- |
| private void btnDodajActionPerformed(java.awt.event.ActionEvent evt) {  try {  String tip=cbxTip.getSelectedItem().toString();  String proizvodjac=txtProizvodjac.getText();  String naziv=txtNaziv.getText();  String cenaStr=txtCena.getText();  int cena=Integer.parseInt(cenaStr);  String datum\_iznajmljivanja=txtDatum\_iznajmljivanja.getText();  String datum\_vracanja=txtDatum\_vracanja.getText();  Class.forName("com.mysql.jdbc.Driver");  Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/bazaiznajmljivanjemuzickeopreme","root","");  PreparedStatement ps = con.prepareStatement("insert into Oprema(Tip, Proizvodjac, Naziv, Cena, Datum\_iznajmljivanja, Datum\_vracanja)values(?,?,?,?,?,?)");  ps.setString(1,tip);  ps.setString(2,proizvodjac);  ps.setString(3,naziv);  ps.setInt(4,cena);  ps.setString(5, datum\_iznajmljivanja);  ps.setString(6, datum\_vracanja);  ps.executeUpdate();  JOptionPane.showMessageDialog(this, "Oprema dodata!");  prikazTabele();  // TODO add your handling code here:  } catch (Exception ex) {  Logger.getLogger(muzickaoprema.class.getName()).log(Level.SEVERE, null, ex);  }  } |

*Listing 2. Prikaz koda za dodavanje elemenata u bazu podataka*

|  |
| --- |
| private void btnIzmeniActionPerformed(java.awt.event.ActionEvent evt) {  DefaultTableModel RecordTable=(DefaultTableModel)Tabela.getModel();  int izabraniRed=Tabela.getSelectedRow();    try {  int ID\_opreme=Integer.parseInt(RecordTable.getValueAt(izabraniRed, 0).toString());  String tip=cbxTip.getSelectedItem().toString();  String proizvodjac=txtProizvodjac.getText();  String naziv=txtNaziv.getText();  String cenaStr=txtCena.getText();  int cena=Integer.parseInt(cenaStr);  String datum\_iznajmljivanja=txtDatum\_iznajmljivanja.getText();  String datum\_vracanja=txtDatum\_vracanja.getText();  Class.forName("com.mysql.jdbc.Driver");  Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/bazaiznajmljivanjemuzickeopreme","root","");  PreparedStatement ps = con.prepareStatement("update Oprema set Tip=?,Proizvodjac=?,Naziv=?,Cena=?,Datum\_iznajmljivanja=?,Datum\_vracanja=? where ID\_opreme=?");  ps.setInt(7, ID\_opreme);  ps.setString(1,tip);  ps.setString(2,proizvodjac);  ps.setString(3,naziv);  ps.setInt(4,cena);  ps.setString(5, datum\_iznajmljivanja);  ps.setString(6, datum\_vracanja);  ps.executeUpdate();  JOptionPane.showMessageDialog(this, "Promena izvršena!");  prikazTabele();  // TODO add your handling code here:  } catch (Exception ex) {  Logger.getLogger(muzickaoprema.class.getName()).log(Level.SEVERE, null, ex);  }  } |

*Listing 3. Prikaz koda za izmenu elemenata u bazi podataka*

|  |
| --- |
| private void btnObrisiActionPerformed(java.awt.event.ActionEvent evt) {  DefaultTableModel RecordTable=(DefaultTableModel)Tabela.getModel();  int izabraniRed=Tabela.getSelectedRow();    try {  int ID\_opreme=Integer.parseInt(RecordTable.getValueAt(izabraniRed, 0).toString());  Class.forName("com.mysql.jdbc.Driver");  Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/bazaiznajmljivanjemuzickeopreme","root","");  PreparedStatement ps=con.prepareStatement("delete from Oprema where ID\_opreme=?");  ps.setInt(1, ID\_opreme);  ps.executeUpdate();  JOptionPane.showMessageDialog(this, "Oprema obrisana!");  prikazTabele();  } catch (Exception ex) {  Logger.getLogger(muzickaoprema.class.getName()).log(Level.SEVERE, null, ex);  }  } |

*Listing 4. Prikaz koda za brisanje elemenata iz baze podataka*

|  |
| --- |
| private void TabelaMouseClicked(java.awt.event.MouseEvent evt) {  DefaultTableModel RecordTable=(DefaultTableModel)Tabela.getModel();  int izabraniRed=Tabela.getSelectedRow();    cbxTip.setSelectedItem(RecordTable.getValueAt(izabraniRed, 1) != null ?  RecordTable.getValueAt(izabraniRed, 1).toString() : "");  txtProizvodjac.setText(RecordTable.getValueAt(izabraniRed, 2) != null ?  RecordTable.getValueAt(izabraniRed, 2).toString() : "");  txtNaziv.setText(RecordTable.getValueAt(izabraniRed, 3) != null ?  RecordTable.getValueAt(izabraniRed, 3).toString() : "");  txtCena.setText(RecordTable.getValueAt(izabraniRed, 4) != null ?  RecordTable.getValueAt(izabraniRed, 4).toString() : "");  txtDatum\_iznajmljivanja.setText(RecordTable.getValueAt(izabraniRed, 5) != null ?  RecordTable.getValueAt(izabraniRed, 5).toString() : "");  txtDatum\_vracanja.setText(RecordTable.getValueAt(izabraniRed, 6) != null ?  RecordTable.getValueAt(izabraniRed, 6).toString() : "");  } |

*Listing 5. Prikaz koda za selekciju elemenata iz tabele pritiskom miša*

|  |
| --- |
| private void btnFiltrirajActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  DefaultTableModel model = (DefaultTableModel) Tabela.getModel();  TableRowSorter<DefaultTableModel> sorter = new TableRowSorter<>(model);  Tabela.setRowSorter(sorter);  String izabranaCena = cbxFilter\_cena.getSelectedItem().toString();  String izabraniTip = cbxFilter\_tip.getSelectedItem().toString();    if (izabranaCena.equals("Od najniže")){  sorter.setComparator(4, Comparator.naturalOrder());  Tabela.getRowSorter().toggleSortOrder(4);  } else if (izabranaCena.equals("Od najviše")) {  sorter.setComparator(4, Comparator.reverseOrder());  Tabela.getRowSorter().toggleSortOrder(4);  }    if (izabraniTip.equals("Abecedno uzlazno")){  sorter.setComparator(1, Comparator.naturalOrder());  Tabela.getRowSorter().toggleSortOrder(1);  } else if (izabraniTip.equals("Abecedno silazno")) {  sorter.setComparator(1, Comparator.reverseOrder());  Tabela.getRowSorter().toggleSortOrder(1);  }  } |

*Listing 6. Prikaz koda za filtriranje elemenata u tabeli*

## Delovi koda u C# programskom jeziku

|  |
| --- |
| private void prikazTabele()  {  SqlConnection konekcija = new SqlConnection();  konekcija.ConnectionString = ConfigurationManager.ConnectionStrings["BazaIznajmljivanjeMuzickeOpreme"].ConnectionString;  konekcija.Open();  SqlCommand komanda = new SqlCommand();  komanda.CommandText = "SELECT \* FROM [Oprema]";  komanda.Connection = konekcija;  SqlDataAdapter dataAdapter = new SqlDataAdapter(komanda);  DataTable tabela = new DataTable("Oprema");  dataAdapter.Fill(tabela);  Tabela.ItemsSource = tabela.DefaultView;  } |

*Listing 7. Kod za prikaz elemenata u tabeli*

|  |
| --- |
| private void Dodaj\_Klik(object sender, RoutedEventArgs e)  {  SqlConnection konekcija = new SqlConnection();  konekcija.ConnectionString = ConfigurationManager.ConnectionStrings["BazaIznajmljivanjeMuzickeOpreme"].ConnectionString;  konekcija.Open();  SqlCommand komanda = new SqlCommand();  komanda.CommandText = "INSERT INTO [Oprema](Tip, Proizvodjac, Naziv, Cena, Datum\_iznajmljivanja, Datum\_vracanja) VALUES(@Tip, @Proizvodjac, @Naziv, @Cena, @Datum\_iznajmljivanja, @Datum\_vracanja)";  komanda.Parameters.AddWithValue("@Tip", cbxTip.Text);  komanda.Parameters.AddWithValue("@Proizvodjac", txtProizvodjac.Text);  komanda.Parameters.AddWithValue("@Naziv", txtNaziv.Text);  komanda.Parameters.AddWithValue("@Cena", txtCena.Text);  komanda.Parameters.AddWithValue("@Datum\_iznajmljivanja", dtDatum\_iznajmljivanja.SelectedDate ?? (object)DBNull.Value);  komanda.Parameters.AddWithValue("@Datum\_vracanja", dtDatum\_vracanja.SelectedDate ?? (object)DBNull.Value);  komanda.Connection = konekcija;  int provera = komanda.ExecuteNonQuery();  if (provera == 1)  {  MessageBox.Show("Podaci su uspešno upisani!");  prikazTabele();  }  ponistiUnos();  } |

*Listing 8. Prikaz koda za dodavanje elemenata u bazu podataka*

|  |
| --- |
| private void Izmeni\_Klik(object sender, RoutedEventArgs e)  {  SqlConnection konekcija = new SqlConnection();  konekcija.ConnectionString = ConfigurationManager.ConnectionStrings["BazaIznajmljivanjeMuzickeOpreme"].ConnectionString;  konekcija.Open();  SqlCommand komanda = new SqlCommand();  komanda.CommandText = "UPDATE [Oprema] SET Tip = @Tip, Proizvodjac = @Proizvodjac, Naziv = @Naziv, Cena = @Cena, Datum\_iznajmljivanja = @Datum\_iznajmljivanja, Datum\_vracanja = @Datum\_vracanja WHERE ID\_opreme = @ID\_opreme";  komanda.Parameters.AddWithValue("@ID\_opreme", txtID\_opreme.Text);  komanda.Parameters.AddWithValue("@Tip", cbxTip.Text);  komanda.Parameters.AddWithValue("@Proizvodjac", txtProizvodjac.Text);  komanda.Parameters.AddWithValue("@Naziv", txtNaziv.Text);  komanda.Parameters.AddWithValue("@Cena", txtCena.Text);  komanda.Parameters.AddWithValue("@Datum\_iznajmljivanja", dtDatum\_iznajmljivanja.SelectedDate ?? (object)DBNull.Value);  komanda.Parameters.AddWithValue("@Datum\_vracanja", dtDatum\_vracanja.SelectedDate ?? (object)DBNull.Value);  komanda.Connection = konekcija;  int provera = komanda.ExecuteNonQuery();  if (provera == 1)  {  MessageBox.Show("Podaci su uspešno promenjeni!");  prikazTabele();  }  ponistiUnos();  } |

*Listing 9. Prikaz koda za izmenu elemenata u bazi podataka*

|  |
| --- |
| private void Obrisi\_Klik(object sender, RoutedEventArgs e)  {  SqlConnection konekcija = new SqlConnection();  konekcija.ConnectionString = ConfigurationManager.ConnectionStrings["BazaIznajmljivanjeMuzickeOpreme"].ConnectionString;  konekcija.Open();  SqlCommand komanda = new SqlCommand();  komanda.CommandText = "DELETE FROM [Oprema] WHERE ID\_opreme = @ID\_opreme";  komanda.Parameters.AddWithValue("@ID\_opreme", txtID\_opreme.Text);  komanda.Connection = konekcija;  int provera = komanda.ExecuteNonQuery();  if (provera == 1)  {  MessageBox.Show("Podaci su uspešno obrisani");  prikazTabele();  }  ponistiUnos();  } |

*Listing 10. Prikaz koda za brisanje elemenata iz baze podataka*

|  |
| --- |
| private void Tabela\_SelectionChanged(object sender, SelectionChangedEventArgs e)  {  DataGrid dg = sender as DataGrid;  DataRowView dr = dg.SelectedItem as DataRowView;  if (dr != null)  {  txtID\_opreme.Text = dr["ID\_opreme"].ToString();  cbxTip.Text = dr["Tip"].ToString();  txtProizvodjac.Text = dr["Proizvodjac"].ToString();  txtNaziv.Text = dr["Naziv"].ToString();  txtCena.Text = dr["Cena"].ToString();  dtDatum\_iznajmljivanja.Text = dr["Datum\_iznajmljivanja"].ToString();  dtDatum\_vracanja.Text = dr["Datum\_vracanja"].ToString();  }  } |

*Listing 11. Prikaz koda za selekciju elemenata iz tabele pritiskom miša*

|  |
| --- |
| private void Filtriraj\_Klik(object sender, RoutedEventArgs e)  {  ICollectionView prikazi = CollectionViewSource.GetDefaultView(Tabela.ItemsSource);  prikazi.SortDescriptions.Clear();  string izabranaCena = (cbxFilter\_cena.SelectedItem as ComboBoxItem)?.Content.ToString();  string izabraniTip = (cbxFilter\_tip.SelectedItem as ComboBoxItem)?.Content.ToString();  if (izabranaCena == "Od najniže")  {  prikazi.SortDescriptions.Add(new SortDescription("Cena", ListSortDirection.Ascending));  }  else if (izabranaCena == "Od najviše")  {  prikazi.SortDescriptions.Add(new SortDescription("Cena", ListSortDirection.Descending));  }  if (izabraniTip == "Abecedno uzlazno")  {  prikazi.SortDescriptions.Add(new SortDescription("Tip", ListSortDirection.Ascending));  }  else if (izabraniTip == "Abecedno silazno")  {  prikazi.SortDescriptions.Add(new SortDescription("Tip", ListSortDirection.Descending));  }  } |

*Listing 12. Prikaz koda za filtriranje elemenata u tabeli*

## Objašnjenje ključnih delova koda

* Kod u listingu 1 i 7 povezuje tabelu sa bazom podataka i prikazuje sve elemente u tabeli.
* Kod u listingu 2 i 8 omogućava unos podataka koji su upisani u text i combo box-ovima i date picker-ima pritiskom na dugme Dodaj.
* Kod u listingu 3 i 9 vrši izmenu podataka pritiskom na dugme Izmeni.
* Kod u listingu 4 i 10 vrši brisanje podataka pritiskom na dugme Obriši.
* Kod u listingu 5 i 11 omogućava selekciju podataka u tabeli, kako bi se popunili text i combo box-ovi i omogućilo brže i lakše brisanje ili menjanje izabranih podataka.
* Kod u listingu 6 i 12 omogućava odabir filtera iz dva combo-boxa koji imaju opcije filtriranja po ceni od najniže do najviše i obrnuto i po tipu abecedom silazno i uzlazno. Pritiskom na dugme Filtriraj, filteri će se primeniti.

# GREŠKE PROGRAMSKOG KODA

## Greške u Java programskom jeziku

**Leksička greška:**



*Slika 17. Izgled koda sa leksičkom greškom*

U ovom kodu je napravljena leksička greška jer je pogrešno napisan tip podataka Stringb i ne može se prepoznati kao validna ključna reč Java jezika.

Formalna gramatika:

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<tip\_podataka> <razmak> <promenljiva> <znak\_jednakosti > <objekat> <znak\_dodele\_atributa> <funkcija> <naziv\_funkcije> <zaglavlje\_funkcije> <pocetak\_funkcije> <kraj\_funkcije> <znak\_kraja\_naredbe>** |
| **Terminali:** | **"String", " ", "proizvodjac", "=", "txtProizvodjac", ".", "getText", "(", ")", ";"** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<naredba> = <tip\_podataka>, <razmak>, <objekat>, <znak\_jednakosti>, <objekat>, <znak\_dodele\_atributa>, <sadrzaj\_objekta>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 2. **<funkcija> = <naziv\_funkcije>, <zaglavlje\_funkcije>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 3. **<zaglavlje\_funkcije> = <pocetak\_fukcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 4. **<tip\_podataka> = "String";** 5. **<promenljiva> = "proizvodjac";** 6. **<razmak> = " ";** 7. **<znak\_jednakosti> = "=";** 8. **<objekat> = "txtProizvodjac";** 9. **<znak\_dodele\_atributa> = ".";** 10. **<naziv\_funkcije> = "getText";** 11. **<pocetak\_funkcije> = "(";** 12. **<kraj\_funkcije> =")";** 13. **<znak\_kraja\_naredbe> = ";";** |

A white background with red text

Description automatically generated

*Slika 18. Izgled compiler-a*

Greška se može popraviti tako što se pravilno napiše tip podataka String.

**Logička greška:**

**A black and pink math equations

Description automatically generated**

*Slika 19. Izgled koda sa logičkom greškom*

Logička greška se dešava jer je zamenjeno mesto elementima.

Formalna gramatika:

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<tip\_podataka> <razmak> <promenljiva> <znak\_jednakosti > <objekat> <znak\_dodele\_atributa> <funkcija> <naziv\_funkcije> <zaglavlje\_funkcije> <pocetak\_funkcije> <kraj\_funkcije> <znak\_kraja\_naredbe>** |
| **Terminali:** | **"String", " ", "proizvodjac", "=", "txtProizvodjac", ".", "getText", "(", ")", ";"**  **"String", " ", "naziv", "=", "txtNaziv", ".", "getText", "(", ")", ";"** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<naredba> = <tip\_podataka>, <razmak>, <objekat>, <znak\_jednakosti>, <objekat>, <znak\_dodele\_atributa>, <sadrzaj\_objekta>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 2. **<funkcija> = <naziv\_funkcije>, <zaglavlje\_funkcije>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 3. **<zaglavlje\_funkcije> = <pocetak\_fukcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 4. **<tip\_podataka> = "String";** 5. **<promenljiva> = "proizvodjac", "naziv";** 6. **<razmak> = " ";** 7. **<znak\_jednakosti> = "=";** 8. **<objekat> = "txtProizvodjac", "txtNaziv";** 9. **<znak\_dodele\_atributa> = ".";** 10. **<naziv\_funkcije> = "getText";** 11. **<pocetak\_funkcije> = "(";** 12. **<kraj\_funkcije> =")";** 13. **<znak\_kraja\_naredbe> = ";";** |

Compiler ne prepoznaje logičku grešku.

Greška se popravlja tako se redosled elementima.

**Sintaksna greška:**



*Slika 20. Izgled koda sa sintaksnom greškom*

Sintaksna greška nastaje jer nije napisan simbol za kraj naredbe (;).

Formalna gramatika:

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<tip\_podataka> <razmak> <promenljiva> <znak\_jednakosti > <objekat> <znak\_dodele\_atributa> <funkcija> <naziv\_funkcije> <zaglavlje\_funkcije> <pocetak\_funkcije> <kraj\_funkcije> <znak\_kraja\_naredbe>** |
| **Terminali:** | **"String", " ", "naziv", "=", "txtNaziv", ".", "getText", "(", ")", ";"** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<naredba> = <tip\_podataka>, <razmak>, <objekat>, <znak\_jednakosti>, <objekat>, <znak\_dodele\_atributa>, <sadrzaj\_objekta>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 2. **<funkcija> = <naziv\_funkcije>, <zaglavlje\_funkcije>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 3. **<zaglavlje\_funkcije> = <pocetak\_fukcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 4. **<tip\_podataka> = "String";** 5. **<promenljiva> = "naziv";** 6. **<razmak> = " ";** 7. **<znak\_jednakosti> = "=";** 8. **<objekat> = "txtNaziv";** 9. **<znak\_dodele\_atributa> = ".";** 10. **<naziv\_funkcije> = "getText";** 11. **<pocetak\_funkcije> = "(";** 12. **<kraj\_funkcije> =")";** 13. **<znak\_kraja\_naredbe> = ";";** |

A screenshot of a computer screen

Description automatically generated

*Slika 21. Izgled compiler-a*

Greška se može popraviti tako što se napiše simbol za kraj naredbe (;).

**Semantička greška:**



*Slika 22. Izgled koda sa semantičkom greškom*

Semantička greška nastaje jer se prosleđuje pogrešan tip podataka.

Formalna gramatika:

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<tip\_podataka> <razmak> <promenljiva> <znak\_jednakosti > <objekat> <znak\_dodele\_atributa> <funkcija> <naziv\_funkcije> <zaglavlje\_funkcije> <pocetak\_funkcije> <kraj\_funkcije> <znak\_kraja\_naredbe>** |
| **Terminali:** | **"String", " ", "naziv", "=", "txtNaziv", ".", "getSelectedItem", "(", ")", ";"** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<naredba> = <tip\_podataka>, <razmak>, <objekat>, <znak\_jednakosti>, <objekat>, <znak\_dodele\_atributa>, <sadrzaj\_objekta>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 2. **<funkcija> = <naziv\_funkcije>, <zaglavlje\_funkcije>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 3. **<zaglavlje\_funkcije> = <pocetak\_fukcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 4. **<tip\_podataka> = "String";** 5. **<promenljiva> = "naziv";** 6. **<razmak> = " ";** 7. **<znak\_jednakosti> = "=";** 8. **<objekat> = "txtNaziv";** 9. **<znak\_dodele\_atributa> = ".";** 10. **<naziv\_funkcije> = "getSelectedItem";** 11. **<pocetak\_funkcije> = "(";** 12. **<kraj\_funkcije> =")";** 13. **<znak\_kraja\_naredbe> = ";";** |

A screenshot of a computer

Description automatically generated

*Slika 23. Izgled compiler-a*

Greška se popravlja tako što se prosleđuje tačan tip podatka.

**Runtime greška:**



*Slika 24. Izgled koda sa runtime greškom*

Runtime greška nastaje kada se izostavi slovo "r" iz imena drajvera za konekciju sa bazom podataka.

Formalna gramatika:

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<funkcija>, <naziv\_funkcije>, <zaglavlje\_funkcije>, <pocetak\_funkcije>, <naziv\_drajvera>, <kraj\_funkcije>, <znak\_kraja\_naredbe>** |
| **Terminali:** | **"Class", "forName", "(", "com.mysql.jdbc.Driver", ")", ";"** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<Naredba> = <funkcija>, <naziv\_funkcije>, <zaglavlje\_funkcije> <pocetak\_funkcije>, <naziv\_drajvera>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 2. **<funkcija> = <naziv\_funkcije>, <zaglavlje\_funkcije>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 3. **<zaglavlje\_funkcije> = <pocetak\_fukcije>, <naziv\_drajvera>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 4. **<pocetak\_funkcije > = "(";** 5. **<naziv\_drajvera> = "com.mysql.jdbc.Driver";** 6. **<kraj\_funkcije> = ")";** 7. **<znak\_kraja\_naredbe> = “;”;** |

A screenshot of a computer

Description automatically generated

*Slika 24. Izgled compiler-a*

Greška se popravlja dodavanjem slova izostavljenog slova "r" u naziv dravjera.

## Greške koda u C# programskom jeziku

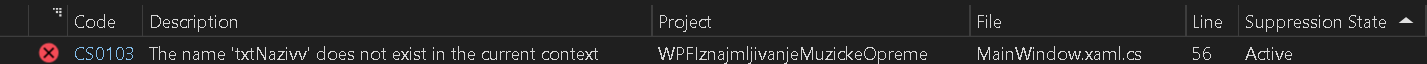
**Leksička greška:**



*Slika 25, Izgled leksičke greške u kodu*

Leksička greška nastaje kada se pogrešno napiše ime promenljive "Naziv" i ne može se prepoznati kao validna ključna reč jezika C#.

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<objekat> <znak\_dodele\_atributa> <tip\_podataka> <razmak> <znak\_jednakosti> <znak\_kraja\_naredbe>;** |
| **Terminali:** | **"txtNaziv", ".", "Text", " ", "=", ";"** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<naredba> = <objekat> <znak\_dodele\_atributa> <tip\_podataka> <razmak> <znak\_jednakosti> <znak\_kraja\_naredbe>;** 2. **<objekat> = "txtNaziv";** 3. **<znak\_dodele\_atributa> = ".":** 4. **<tip\_podataka> = "Text";** 5. **<razmak> = " ";** 6. **<znak\_jednakosti> = "=";** 7. **<znak\_kraja\_naredbe> = ";";** |



*Slika 26. Izgled compiler-a*

Greška se popravlja tako što se pravilno ispiše tip podataka.

**Logička greška:**

****

*Slika 27. Izgled logičke greške u kodu*

Logička greška nastaje kada se zameni mesto elementima.

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<objekat> <znak\_dodele\_atributa> <tip\_podataka> <razmak> <znak\_jednakosti> <varijabla> <znak\_dodele\_atributa> <funkcija> <naziv\_funkcije> <zaglavlje\_funkcije> <pocetak\_funkcije> <kraj\_funkcije> <znak\_kraja\_naredbe>** |
| **Treminali:** | **"txtProizvodjac", ".", "Text", " ", "=", "dr["Proizvodjac"]", "ToString", "(", ")", ";";**  **"txtNaziv", ".", "Text", " ", "=", "dr["Naziv"]", "ToString", "(", ")", ";";** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<naredba> = <tip\_podataka>, <razmak>, <objekat>, <znak\_jednakosti>, <objekat>, <znak\_dodele\_atributa>, <sadrzaj\_objekta>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 2. **<funkcija> = <naziv\_funkcije>, <zaglavlje\_funkcije>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 3. **<zaglavlje\_funkcije> = <pocetak\_fukcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 4. **<objekat> = "txtProizvodjac", "txtNaziv";** 5. **<znak\_dodele\_atributa> = ".";** 6. **<razmak> = " ";** 7. **<znak\_jednakosti> = "=";** 8. **<varijabla> = "dr["Proizvodjac"]", "dr["Naziv"]";** 9. **<naziv\_funkcije> = "ToString";** 10. **<pocetak\_funkcije> = "(";** 11. **<kraj\_funkcije> = ")";** 12. **<znak\_kraja\_naredbe> = ";";** |

Compiler ne prepoznaje logičku grešku.

**Sintaksna greška:**

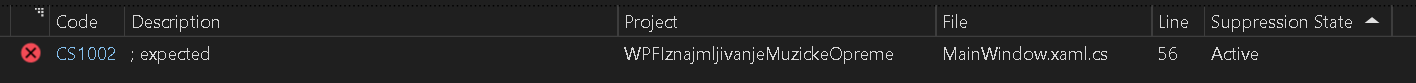


*Slika 28. Izgled sintaksne greške u kodu*

Sintaksna greška nastaje kada se izostavi znak kraja naredbe (;).

Formalna gramatika:

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<objekat> <znak\_dodele\_atributa> <tip\_podataka> <razmak> <znak\_jednakosti> <znak\_kraja\_naredbe>;** |
| **Terminali:** | **"txtNaziv", ".", "Text", " ", "=", ";"** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<naredba> = <objekat> <znak\_dodele\_atributa> <tip\_podataka> <razmak> <znak\_jednakosti> <znak\_kraja\_naredbe>;** 2. **<objekat> = "txtNaziv";** 3. **<znak\_dodele\_atributa> = ".":** 4. **<tip\_podataka> = "Text";** 5. **<razmak> = " ";** 6. **<znak\_jednakosti> = "=";** 7. **<znak\_kraja\_naredbe> = ";";** |



*Slika 29. Izgled compiler-a*

Greška će biti otklonjena kada se upiše znak za kraj naredbe (;).

**Semantička greška:**



*Slika 30. Izgled semantičke greške u kodu*

Semantička greška nastaje kada se prosleđuje pogrešan tip podataka.

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<objekat> <znak\_dodele\_atributa> <tip\_podataka> <razmak> <znak\_jednakosti> <varijabla> <znak\_dodele\_atributa> <funkcija> <naziv\_funkcije> <zaglavlje\_funkcije> <pocetak\_funkcije> <kraj\_funkcije> <znak\_kraja\_naredbe>** |
| **Terminali:** | **"txtNaziv", ".", "Text", " ", "=", "dr["Naziv"]", "ToString", "(", ")", ";";** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<naredba> = <tip\_podataka>, <razmak>, <objekat>, <znak\_jednakosti>, <objekat>, <znak\_dodele\_atributa>, <sadrzaj\_objekta>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 2. **<funkcija> = <naziv\_funkcije>, <zaglavlje\_funkcije>, <pocetak\_funkcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 3. **<zaglavlje\_funkcije> = <pocetak\_fukcije>, <kraj\_funkcije>, <znak\_kraja\_naredbe>;** 4. **<objekat> = "txtProizvodjac";** 5. **<znak\_dodele\_atributa> = ".";** 6. **<razmak> = " ";** 7. **<znak\_jednakosti> = "=";** 8. **<varijabla> = "dr["Proizvodjac"]";** 9. **<naziv\_funkcije> = "ToString";** 10. **<pocetak\_funkcije> = "(";** 11. **<kraj\_funkcije> = ")";** 12. **<znak\_kraja\_naredbe> = ";";** |



*Slika 31. Izgled compiler-a*

Greška će biti otklonjena kada se napiše validan tip podataka.

**Runtime greška:**

****

*Slika 32. Izgled runtime greške*

Greška nastaje kada se pogrešno napiše ime baze podataka u konekcionom stringu.

Formalna gramatika:

|  |  |
| --- | --- |
| **Varijable:** | **<naredba>**  **<objekat> <znak\_dodele\_atributa> <tip\_podataka> <razmak> <znak\_jednakosti> <konekcioni\_string> <znak\_kraja\_naredbe>** |
| **Terminali:** | **"konekcija", "ConnectionString", " ", "=", "ConfigurationManager.ConnectionStrings["BazaIznajmljivanjeMuzickeOpreme"].ConnectionString", ";";** |
| **Startni simbol:** | **<naredba>** |
| **Pravila:** | 1. **<naredba> = <objekat>, <znak\_dodele\_atributa>, <tip\_podataka>, <razmak>, <znak\_jednakosti>, <konekcioni\_string>, <znak\_kraja\_naredbe>;** 2. **<objekat> = "konekcija";** 3. **<znak\_dodele\_atributa> = ".";** 4. **<tip\_podataka> = "ConnectionString";** 5. **<znak\_jednakosti> = "=";** 6. **<konekcioni\_string> = "ConfigurationManager.ConnectionStrings["BazaIznajmljivanjaMuzickeOpreme"].ConnectionString";** 7. **<znak\_kraja\_naredbe> = ";";** |

A black background with white text

Description automatically generated

*Slika 33. Izgled compiler-a*

Greška se otklanja tako što se pravilno napiše ime baze podataka u konekcionom stringu.

# ZAKLJUČAK

I Java i C# programski jezik imaju svoje jedinstvene načine implementacije elemenata ali C# ima prednosti za bolju implementaciju i posebno je dobar izbor za kreiranje korisničkog interfejsa i omogućava lakšu i bržu izradu aplikacija sa bogatim grafičkim korisničkim interfejsom. Sa druge strane, Java je pogodnija za implementaciju elemenata na različitim operativnim sistemima i poznata je po svojoj platformskoj nezavisnosti.