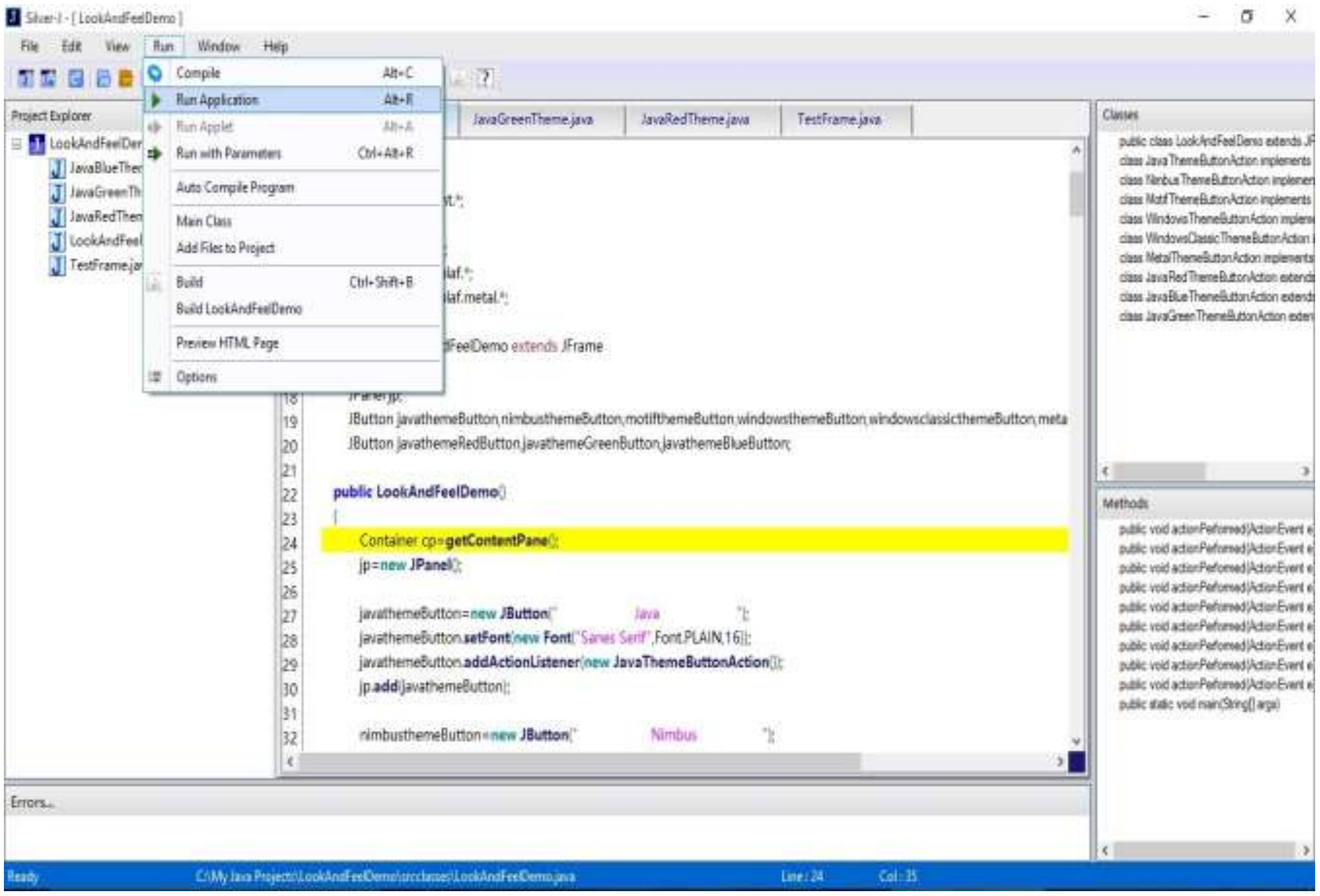
Java IDE in C#

(Silver-J)



```
                 t.*;

                 laf.*;
                 laf.metal.*;

                 FeelDemo extends JFrame

    JPanel jp;
    JButton javathemeButton,nimbusthemeButton,motifthemeButton,windowsthemeButton,windowsclassicthemeButton,meta
    JButton javathemeRedButton,javathemeGreenButton,javathemeBlueButton;

  public LookAndFeelDemo()
  {
     Container cp=getContentPane();
     jp=new JPanel();

     javathemeButton=new JButton("            Java            ");
     javathemeButton.setFont(new Font("Sanes Serif",Font.PLAIN,16));
     javathemeButton.addActionListener(new JavaThemeButtonAction());
     jp.add(javathemeButton);

     nimbusthemeButton=new JButton("            Nimbus            ");
```

```
public class LookAndFeelDemo extends JF
class JavaThemeButtonAction implements
class NimbusThemeButtonAction implemen
class MotifThemeButtonAction implements
class WindowsThemeButtonAction impleme
class WindowsClassicThemeButtonAction i
class MetalThemeButtonAction implements
class JavaRedThemeButtonAction extends
class JavaBlueThemeButtonAction extends
class JavaGreenThemeButtonAction exten
```

```
public void actionPerformed(ActionEvent e
public void actionPerformed(ActionEvent e
public void actionPerformed(ActionEvent e
public void actionPerformed(ActionEvent e
public void actionPerformed(ActionEvent e
public void actionPerformed(ActionEvent e
public void actionPerformed(ActionEvent e
public void actionPerformed(ActionEvent e
public void actionPerformed(ActionEvent e
public static void main(String[] args)
```

Requirements

Visual Studio 2013 Professional

.Net framework 4.5

ICSharpCode.TextEditor.dll library

-which I have used in this project,because it is free,but you can use any other library for syntax highlighting,I have also used some features of this library.
You can download from following sites :

https://www.nuget.org/packages/ICSharpCode.TextEditor/

You must satisfy the conditions of LGPL of this library.

Copyright

Introduction

I have created the named Silver-J project in Visual Studio and used TextEditor as ICSharpCode.TextEditor.dll. You can use any other text editor that you want.
In this document we will focus only on Creating Project file,Reading Project file,and starting processes for compiling java source code file.
I have defined all the functions in same project without creating other library(dll).
There may be some extra not needed,memory consuming code in it.
To understand this source code,read the Creating Advanced Notepad In C# article
You can use following articles to create or customize your IDE.

Creating Custom Windows Forms in C# using Panels

Creating Advanced Notepad in C#

Creating Code Completion in C#

Creating AutoComplete HTML Tags in C#

Creating DropDownControl in C#

I have used C# programming language to create this IDE but it is possible to create IDE in Java,the benefits of using java is that you can access all features of it,it also supports many syntax highlighting libraries.C# & Java are syntactically same.
Following article may help you to how to create Tabbed Notepad editor in java.

Creating Advanced Tabbed Notepad In Java

On Linux you can run C# application using Mono

But try to use this source code on Windows because it is developed in Visual Studio.

Files

For this project I have created many files where I can store the data. Following files are stored in the **files** folder at application location.

1. appletcode.slvjappletfile
2. config.slvjfile
3. defaultprojloc.slvjfile
4. files.slvjfile
5. jkeywords.slvjfile
6. jpackages.slvjfile
7. mainhtmlsource.slvjmainhtmlfile
8. themesfile.slvjthemefile
9. <project_name>.slvjproj

1) appletcode.slvjappletfile : this file contains HTML source code with <applet></applet> tag.

```
<!--
  Applet HTML source
-->

<html>

   <head></head>
   <title>projectname</title>

   <body>

      <h1>projectname</h1>
      <hr>

      <applet code="mainclassname" width="500" height="500">
      </applet>

   </body>

</html>
```

2) config.slvjfile : This file contains the data about everything of our application,like JDK path,Browser path,To show ToolStrip,StatusStrip or not,SplitContainer values. e.g:

```xml
<?xml version="1.0" encoding="utf-8"?>
<!--
            Silver-J
Free light weight IDE for Java
 Copyright(c) 2016. All rights reserved
-->
<SilverJConfiguration>
<!--JDK Path-->
<JDKPath>C:\Program Files\Java\jdk1.8.0_25\bin</JDKPath>
<!-- web browser -->
<WebBrowser>C:\Program Files (x86)\Google\Chrome\Application\chrome.exe</WebBrowser>
 <!--text editor font & font size-->
<Font>Microsoft YaHei UI</Font>
<FontSize>10</FontSize>
<!--views-->
<!--Appearance-->
 <Appearance>Default</Appearance>
<!--text editor views-->
<ShowLineNumbers>true</ShowLineNumbers>
<ShowLineHighlighter>true</ShowLineHighlighter>
<ShowInvalidLines>false</ShowInvalidLines>
<ShowEndOfLineMarker>false</ShowEndOfLineMarker>
<ShowVisibleSpaces>false</ShowVisibleSpaces>
<BracesMatching>true</BracesMatching>
<AutoCompleteBraces>true</AutoCompleteBraces>
<!--split containers-->
<SplitContainer1>1132</SplitContainer1>
<SplitContainer2>559</SplitContainer2>
<SplitContainer3>280</SplitContainer3>
<SplitContainer4>315</SplitContainer4>
 <!--tabs,documents etc views-->
<TabsAlignment>Top</TabsAlignment>
<ShowStatusStrip>true</ShowStatusStrip>
 <ShowToolStrip>true</ShowToolStrip>
<ShowProjectExplorer>true</ShowProjectExplorer>
<ShowClassesView>true</ShowClassesView>
<ShowMethodsView>true</ShowMethodsView>
<ShowErrorList>true</ShowErrorList>
```

```xml
<ShowErrorDialog>false</ShowErrorDialog>
<ShowStartPageOnStartUp>true</ShowStartPageOnStartUp>
<!--Auto Compile Java-->
<AutoCompileJava>false</AutoCompileJava>
<!--Auto Completion Mode-->
<AutoCompletion>true</AutoCompletion>
</SilverJConfiguration>
```

These all values are read when application starts(MainForm_Load) and sets the value to components,and also overwrite new values to config.slvjfile when application closes(Form_Closing) or menu item selected such as View->Status Strip or ToolStrip.

3) defaultprojloc.slvjfile : This file contains the current created or opened project name,project folder path,projectt folder file(.slvjproj) and project type.
   e.g:

```xml
<?xml version="1.0" encoding="utf-8"?>
<SilverJ>
 <DefaultProjectLocation>C:\My Java Projects</DefaultProjectLocation>
<CurrentProjectName>LookAndFeelDemo</CurrentProjectName>
 <CurrentProjectFileName>C:\My Java
Projects\LookAndFeelDemo\LookAndFeelDemo.slvjproj</CurrentProjectFileName>
 <CurrentProjectType>ApplicationType</CurrentProjectType>
</SilverJ>
```

4) files.slvjfile : This file contain the file names read from <VisualFile></VisualFile> tag from opened project file name.

5) jkeywords.slvjfile : This file contains the java keywords which be used for code completion(see MyTabPage.cs file)

6) jpackages.slvjfile : This file contains the java packages which be used to insert packages option Edit->Insert->Packages

7) mainhtmlsource.slvjmainhtmlfile : Contains the HTML source code when New->HTML file is added or created.

8) themesfile.slvjthemefile : This file contain the java code when Create Java Class with Java Themes

```
/*******************************************
            projectname
             themename
*******************************************/
import java.awt.*;
import javax.swing.*;

modifier class classname extends JFrame
{
 modifier classname()
{
//set title & location to frame
setTitle("classname");
setLocation(200,100);

//add panel and button to container
 JPanel jp=new JPanel();
 JButton b=new JButton();
 b.setText("Look And Feel");
 jp.add(b);

 //add container to frame
Container cp=getContentPane();
cp.add(jp,BorderLayout.CENTER);
//define your code here
}

//main method
public static void main(String[] args)
{

 //adding theme to JFrame using UIManager class
 // and static method setLookAndFeel
    try
{
UIManager.setLookAndFeel("themetype");
 }
catch (Exception e)
{
e.printStackTrace();
}

JFrame jf=new classname();
```

```
        jf.setSize(500,500);
        jf.setVisible(true);
        }
}
```

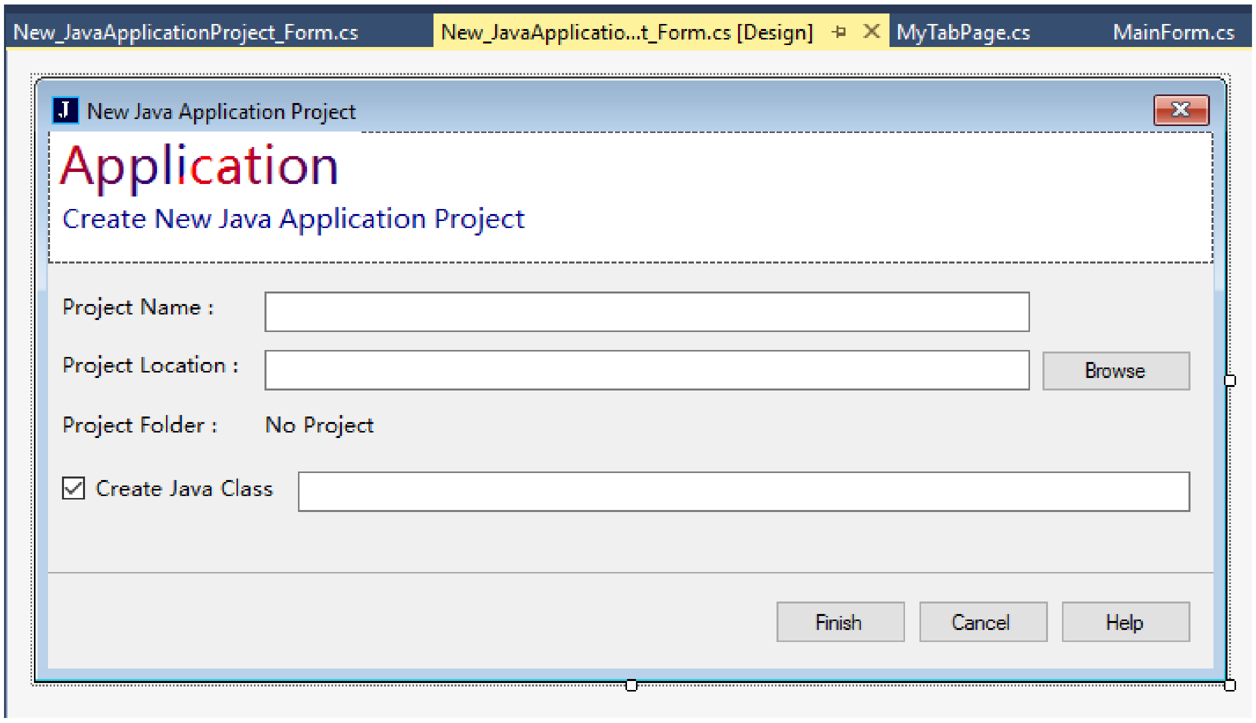
9) <project_name>.slvjproj : This file is stored in the users project folder. This file contain ProjectName, ProjectLocationFolder, ProjectType, MainClassFile, JavaClassFile, VisualFile, OtherFile.
e.g of **LookAndFeelDemo.slvjproj**

```
<?xml version="1.0" encoding="utf-8"?>
<SilverJProject>
  <!--Silver-J (1.0) Java Application Project-->
  <ProjectName>LookAndFeelDemo</ProjectName>
  <ProjectLocationFolder>C:\My Java Projects\LookAndFeelDemo</ProjectLocationFolder>
  <ProjectLocationFile>C:\My Java
Projects\LookAndFeelDemo\LookAndFeelDemo.slvjproj</ProjectLocationFile>
  <ProjectType>ApplicationType</ProjectType>
  <MainClassFile>C:\My Java
Projects\LookAndFeelDemo\srcclasses\LookAndFeelDemo.java</MainClassFile>
  <JavaClassFile>C:\My Java Projects\LookAndFeelDemo\srcclasses\JavaBlueTheme.java</JavaClassFile>
  <JavaClassFile>C:\My Java Projects\LookAndFeelDemo\srcclasses\JavaGreenTheme.java</JavaClassFile>
  <JavaClassFile>C:\My Java Projects\LookAndFeelDemo\srcclasses\JavaRedTheme.java</JavaClassFile>
  <JavaClassFile>C:\My Java Projects\LookAndFeelDemo\srcclasses\LookAndFeelDemo.java</JavaClassFile>
  <JavaClassFile>C:\My Java Projects\LookAndFeelDemo\srcclasses\TestFrame.java</JavaClassFile>
  <VisualFile>C:\My Java Projects\LookAndFeelDemo\srcclasses\LookAndFeelDemo.java</VisualFile>
  <VisualFile>C:\My Java Projects\LookAndFeelDemo\srcclasses\JavaGreenTheme.java</VisualFile>
  <VisualFile>C:\My Java Projects\LookAndFeelDemo\srcclasses\JavaRedTheme.java</VisualFile>
  <VisualFile>C:\My Java Projects\LookAndFeelDemo\srcclasses\TestFrame.java</VisualFile>
</SilverJProject>
```

VisualFile is the JavaClassFile which is only used to open this file in tab or not when project is opened.

Create Java Project

Add new Windows Form to your project. Here I used **New_JavaApplicationProject_Form** (New_JavaApplicationProject_Form.cs file) and design it with **Project Name textbox**, **Project Location textbox**, **Create Java Class checkbox** with **Browse**, **Finish** & **Cancel** buttons.



New_JavaApplicationProject_Form.cs
New_JavaApplicatio...t_Form.cs [Design]
MyTabPage.cs
MainForm.cs
New Java Application Project
Application
Create New Java Application Project
Project Name :
Project Location :
Browse
Project Folder :
No Project
Create Java Class
Finish
Cancel
Help

In this file New_JavaApplicationProject_Form.cs we will just read a file \\files\\defaultprojloc.slvjfile .
We will also take input from above form as Project name,project location etc.
I have used following variable to store it,

```
String projloc = ProjectLocationTextBox.Text;
String projfolder = projectfolderlabel.Text;
String projectname = ProjectNameTextBox.Text;
String projectfile;
```

Now we need to create a folder(directory) same as project name in the read project location folder(projfolder).Once directory is created we will create classes, src, srcclasses named directories in it.
Once project directory created, we need to create project file with file extension as .slvjproj (file name is same as created project name).In that file we will save our project data as defined in above **File** section.we will save data in XML format.
Following is the function that performs actions.

```
    public void CreateJavaProject()
     {
        String projloc = ProjectLocationTextBox.Text;
        String projfolder = projectfolderlabel.Text;
        String projectname = ProjectNameTextBox.Text;
        String projectfile;

        if (projloc != "" && projectname != "")
        {
          projectfile = projectname + ".slvjproj";
          if (checkBox1.Checked == false)
          {
            if (Directory.Exists(projloc + "\\" + projfolder))
            {
              MessageBox.Show("Entered project name folder is already exists in current
location","Error...........");
            }
            else
            {
              //create project directory & project file
              Directory.CreateDirectory(projloc + "\\" + projfolder);

              //create classes directory
              Directory.CreateDirectory(projloc + "\\" + projfolder + "\\classes");

              //creating & writing slvjproj file
              using (XmlWriter xmlwriter = XmlWriter.Create(projloc + "\\" + projfolder + "\\" +
projectfile))
              {
                xmlwriter.WriteStartDocument();
                xmlwriter.WriteStartElement("SilverJProject");
                xmlwriter.WriteString("\n");
                xmlwriter.WriteComment("Silver-J (1.0) Java Application Project");
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("ProjectName", projectname);
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("ProjectLocationFolder", projloc + "\\" + projfolder);
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("ProjectLocationFile", projloc + "\\" + projfolder + "\\" +
projectfile);
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("ProjectType", "ApplicationType");
                xmlwriter.WriteEndElement();
                xmlwriter.WriteEndDocument();
                xmlwriter.Close();
            }

            String defaultprojfilepath = Application.StartupPath + "\\files\\defaultprojloc.slvjfile";

            projectfilename = projloc + "\\" + projfolder + "\\" + projectfile;

            XmlDocument doc = new XmlDocument();
            doc.Load(defaultprojfilepath);
            doc.SelectSingleNode("SilverJ/DefaultProjectLocation").InnerText =
ProjectLocationTextBox.Text;
            doc.SelectSingleNode("SilverJ/CurrentProjectName").InnerText =
ProjectNameTextBox.Text;
            doc.SelectSingleNode("SilverJ/CurrentProjectFileName").InnerText = projectfilename;
            doc.SelectSingleNode("SilverJ/CurrentProjectType").InnerText = "ApplicationType";
            doc.Save(defaultprojfilepath);

            a = 1;

            this.Close();
            isfinished = true;
        }
    }


    else if (checkBox1.Checked == true)
    {
        if (JavaClassTextBox.Text != "")
        {
            String classname = JavaClassTextBox.Text;
            if (classname.Contains(".java"))
            {
            }
            else
            {
                classname = classname + ".java";
            }

            String javafilename = classname;
            if (Directory.Exists(projloc + "\\" + projfolder))
            {
                MessageBox.Show("Entered project name folder is already exists in current location",
"Error............");
            }
            else
            {
                //create project directory & project file
                Directory.CreateDirectory(projloc + "\\" + projfolder);
                //create srcclasses directory
                Directory.CreateDirectory(projloc + "\\" + projfolder + "\\srcclasses");
                //create classes directory
                Directory.CreateDirectory(projloc + "\\" + projfolder + "\\classes");

                String fname = projloc + "\\" + projfolder + "\\srcclasses\\" + javafilename;
                createdfilename = fname;
                String filename = fname.Substring(fname.LastIndexOf("\\") + 1);

                //creating & writing slvjproj file
                using (XmlWriter xmlwriter = XmlWriter.Create(projloc + "\\" + projfolder + "\\" +
projectfile))
                {
                    xmlwriter.WriteStartDocument();
                    xmlwriter.WriteStartElement("SilverJProject");
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteComment("Silver-J (1.0) Java Application Project");
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteElementString("ProjectName", projectname);
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteElementString("ProjectLocationFolder", projloc + "\\" + projfolder);
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteElementString("ProjectLocationFile", projloc + "\\" + projfolder + "\\" +
projectfile);
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteElementString("ProjectType", "ApplicationType");
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteElementString("MainClassFile", fname);
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteElementString("JavaClassFile", fname);
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteElementString("VisualFile", fname);
                    xmlwriter.WriteEndElement();
                    xmlwriter.WriteEndDocument();
                    xmlwriter.Close();
                }

                isSaved = true;
```

```
                    String defaultprojfilepath = Application.StartupPath + "\\files\\defaultprojloc.slvjfile";


                    projectfilename = projloc + "\\" + projfolder + "\\" + projectfile;

                    XmlDocument doc = new XmlDocument();
                    doc.Load(defaultprojfilepath);
                    doc.SelectSingleNode("SilverJ/DefaultProjectLocation").InnerText =
ProjectLocationTextBox.Text;
                    doc.SelectSingleNode("SilverJ/CurrentProjectName").InnerText =
ProjectNameTextBox.Text;
                    doc.SelectSingleNode("SilverJ/CurrentProjectFileName").InnerText = projectfilename;
                    doc.SelectSingleNode("SilverJ/CurrentProjectType").InnerText = "ApplicationType";
                    doc.Save(defaultprojfilepath);

                    a = 2;

                    this.Close();
                    isfinished = true;
                }
            }
        }
    }
}
```

First check if project location and project name are not empty strings.
Then check whether **Create Java Class** check box is checked or not,if it is not checked then just create project without .java source file.
First create our java project file(projectfile = projectname + ".slvjproj";).
Then we create this file and start to write that file by using **XmlWriter** class.
We will write Project Name, Project Location Folder, Project type tags with its read values .

```
                    //creating & writing slvjproj file
                    using (XmlWriter xmlwriter = XmlWriter.Create(projloc + "\\" + projfolder + "\\" +
projectfile))
                    {
                        xmlwriter.WriteStartDocument();
                        xmlwriter.WriteStartElement("SilverJProject");
                        xmlwriter.WriteString("\n");
```

```
                xmlwriter.WriteComment("Silver-J (1.0) Java Application Project");
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("ProjectName", projectname);
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("ProjectLocationFolder", projloc + "\\" + projfolder);
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("ProjectLocationFile", projloc + "\\" + projfolder + "\\" +
projectfile);
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("ProjectType", "ApplicationType");
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("MainClassFile", fname);
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("JavaClassFile", fname);
                xmlwriter.WriteString("\n");
                xmlwriter.WriteElementString("VisualFile", fname);
                xmlwriter.WriteEndElement();
                xmlwriter.WriteEndDocument();
                xmlwriter.Close();
            }
```

Once we done this we need to change the contents of file defaultprojloc.slvjfile
In this file we will save current created/opened project name,location folder,project location file,project type.

```
            String defaultprojfilepath = Application.StartupPath + "\\files\\defaultprojloc.slvjfile";

            projectfilename = projloc + "\\" + projfolder + "\\" + projectfile;

            XmlDocument doc = new XmlDocument();
            doc.Load(defaultprojfilepath);
            doc.SelectSingleNode("SilverJ/DefaultProjectLocation").InnerText =
ProjectLocationTextBox.Text;
            doc.SelectSingleNode("SilverJ/CurrentProjectName").InnerText =
ProjectNameTextBox.Text;
            doc.SelectSingleNode("SilverJ/CurrentProjectFileName").InnerText = projectfilename;
            doc.SelectSingleNode("SilverJ/CurrentProjectType").InnerText = "ApplicationType";
            doc.Save(defaultprojfilepath);
```

If **Create Java Class** check box is checked then we will perform all action as above just adding following tags and info to it.where fname is entered java class file name

```
                xmlwriter.WriteElementString("MainClassFile", fname);
```

```
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteElementString("JavaClassFile", fname);
                    xmlwriter.WriteString("\n");
                    xmlwriter.WriteElementString("VisualFile", fname);
```

I am keeping track of created project, means if Project is created without creating a java class(means Create Java Class checkbox is unchecked) then **CheckProjectType()** function return 1 otherwise it returns 2(variable a is used for this). This if for to create file,adding tabs to tabcontrol, adding projectname & filename to treeview etc. in MainForm.cs file.
Consider you have already designed you main IDE.
Now event for clicking on File->New->New Java Application Project
First show the New java application dialog form.

ReadCurrentProjectFileName() function returns the string as project file name by reading file \files\defaultprojloc.slvjfile by reading tag CurrentProjectFileName.

If **CheckProjectType()** is 2 then we will create create a file by reading JavaClassFile tag text from current project,add tab to tabcontrol with control of texteditor,changing text of MainForm, adding projectname & file names to ProjectExplorerTreeView and then we will save that file.

```
  private void File_New_JavaApplicationProjectMenuItem_Click(object sender, EventArgs e)
   {
      New_JavaApplicationProject_Form njap = new New_JavaApplicationProject_Form(this,
ProjectExplorerTreeView, myTabControl);
      njap.ShowDialog();

      String projectname = "";
      String javaclassfilename = "";

        //create project with class
        if (njap.CheckProjectType() == 2)
        {
            if (ReadCurrentProjectFileName() != "" && File.Exists(ReadCurrentProjectFileName()))
            {
                //first reading project name & java class file when project type = 2
                String projectfilename = ReadCurrentProjectFileName();
                if (File.Exists(projectfilename))
                {
                    using (XmlReader xmlreader = XmlReader.Create(projectfilename))
                    {
                        while (xmlreader.Read())
                        {
                            if (xmlreader.IsStartElement())
                            {
                                switch (xmlreader.Name.ToString())
                                {
                                    case "ProjectName": projectname = xmlreader.ReadString();
                                        break;

                                    case "JavaClassFile": javaclassfilename = xmlreader.ReadString();
                                        break;
                                }
                            }
                        }
                    }

                    if (projectfilename != "" && javaclassfilename != "" && njap.getCreatedFileName() != "")
                    {
                        //adding nodes to tree view
                        //changing main form name to project name
                        //adding file name to filenametoolstriplabel
                        ProjectExplorerTreeView.Nodes.Clear();
                        myTabControl.TabPages.Clear();

                        String prjname = projectfilename.Substring(projectfilename.LastIndexOf("\\") + 1);
                        prjname = prjname.Remove(prjname.Length - 9);
                        this.Text = "Silver-J - [ " + prjname + " ]";

                        String jclassfilename = javaclassfilename.Substring(javaclassfilename.LastIndexOf("\\")
+ 1);
                        String jclassnamewithoutjava = jclassfilename.Remove(jclassfilename.Length - 5);

                        MyTabPage mytabpage = new MyTabPage(this);
                        mytabpage.Text = jclassfilename;
                        mytabpage.textEditor.Text = "/*******************************\n
"+prjname+"    \n*******************************/"
                            + "\npublic class " + jclassnamewithoutjava + "  {" + "\n
" + "\n}";
                        mytabpage.textEditor.ContextMenuStrip = textEditorContextMenuStrip;
                        myTabControl.TabPages.Add(mytabpage);
                        myTabControl.SelectedTab = mytabpage;

                        TreeNode projecttreenode = new TreeNode();
                        projecttreenode.Text = prjname;
                        projecttreenode.ImageIndex = 6;
                        projecttreenode.SelectedImageIndex = 6;
                        ProjectExplorerTreeView.Nodes.Add(projecttreenode);
                        ProjectExplorerTreeView.SelectedNode = projecttreenode;

                        TreeNode trnode = ProjectExplorerTreeView.Nodes[0];
                        TreeNode jclassnode = new TreeNode();
                        jclassnode.Text = jclassfilename;
                        jclassnode.ImageIndex = 2;
                        jclassnode.SelectedImageIndex = 2;
                        trnode.Nodes.Add(jclassnode);
                        ProjectExplorerTreeView.ExpandAll();

                        if (njap.getCreatedFileName() != "")
                        {
                            try
                            {
                                StreamWriter strw = new StreamWriter(njap.getCreatedFileName());
                                strw.Write(mytabpage.textEditor.Text);
                                strw.Close();
                                strw.Dispose();
                            }
                            catch
                            { }
                        }
                    }
                }
            }
        }

         //create project without class
        else if (njap.CheckProjectType() == 1)
        {
            if (ReadCurrentProjectFileName() != "" && File.Exists(ReadCurrentProjectFileName()))
            {
                String projectfilename2 = ReadCurrentProjectFileName();
                if (File.Exists(projectfilename2))
                {
                    using (XmlReader xmlreader = XmlReader.Create(projectfilename2))
                    {
                        while (xmlreader.Read())
                        {
                            if (xmlreader.IsStartElement())
```

```
                        {
                            switch (xmlreader.Name.ToString())
                            {
                                case "ProjectName": projectname = xmlreader.ReadString();
                                    break;
                            }
                        }
                    }
                }

                if (projectfilename2 != "")
                {
                    ProjectExplorerTreeView.Nodes.Clear();
                    myTabControl.TabPages.Clear();

                    String prjname = projectfilename2.Substring(projectfilename2.LastIndexOf("\\") + 1);
                    prjname = prjname.Remove(prjname.Length - 9);
                    this.Text = "Silver-J - [ " + prjname + " ]";


                    TreeNode projecttreenode = new TreeNode();
                    projecttreenode.Text = prjname;
                    projecttreenode.ImageIndex = 6;
                    projecttreenode.SelectedImageIndex = 6;
                    ProjectExplorerTreeView.Nodes.Add(projecttreenode);
                    ProjectExplorerTreeView.SelectedNode = projecttreenode;
                }
            }
        }
    }


    if (njap.IsFinished() == true)
    {
        FilenameToolStripLabel.Text = "Silver-J";
        AddFilesToProjectExplorerTreeView();
        WriteCurrentFileNames();
        CopyAllSourceFilesToSRCFolder();
        UpdateWindowsList_WindowMenu();
        SetVisibilityOfToolStripButtons();
        myTabControl_SelectedIndexChanged(sender, e);
    }
}
```

If njap.isFinished()==true it will call all functions that we need to change or modify the IDE such as adding files to project explorer tree view,set visibility of toolstrip buttons,changing text of FilenameToolStripLabel to current opened file name.

WriteCurrentFileNames() this function reads all files from current opened project file name & write those file name to \files\files.slvjfile for save operation.

CopyAllSourceFilesToSRCFolder() this function copies the all files from srcclasses to src folder without including .class file.

UpdateWindowsList_WindowMenu() this function checks tabs in tabcontrol and add those tab texts to Window menu by creating new menu items for selecting each tab by clicking on that menu item.

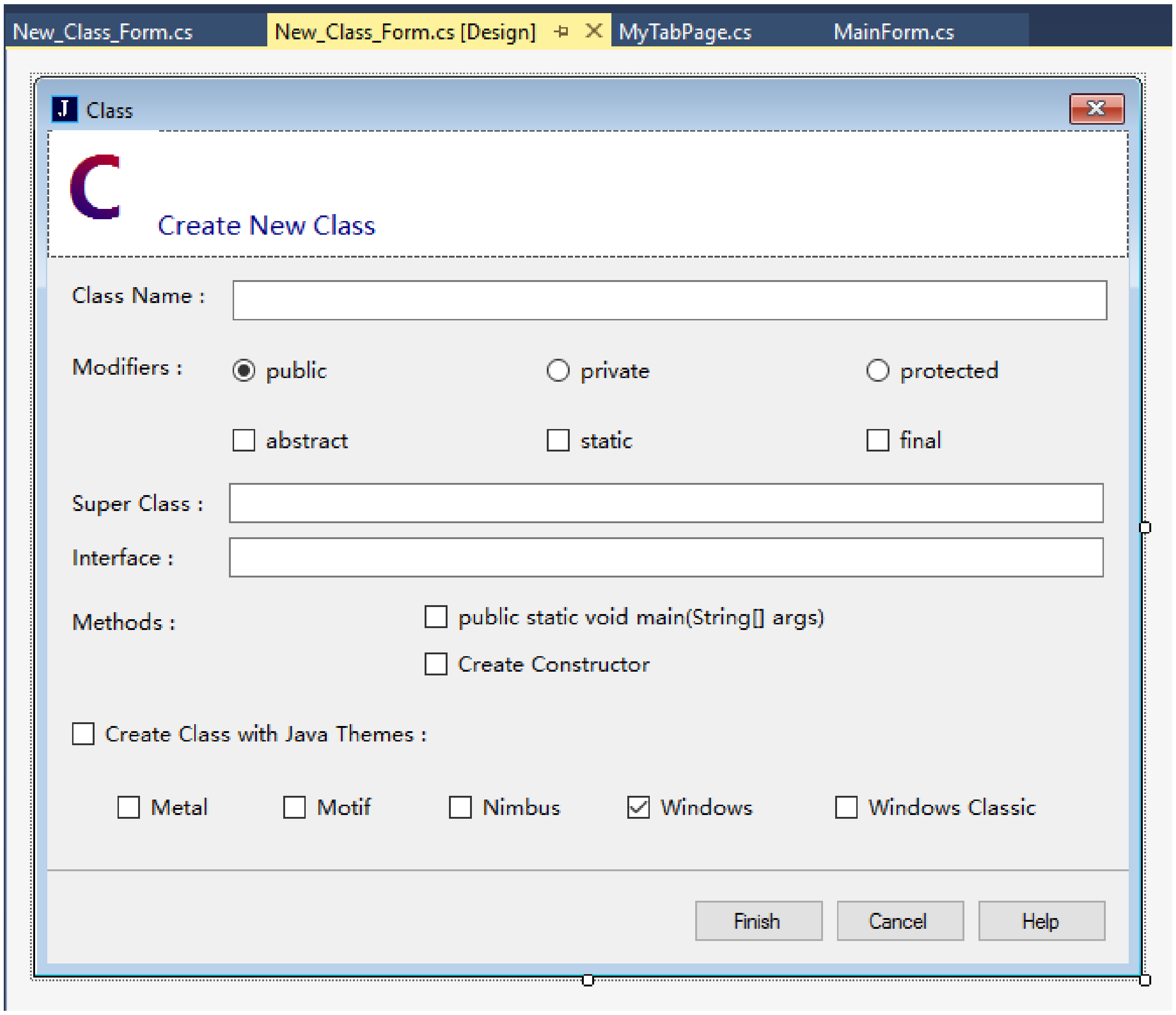
What we are doing here is that,
In **New_JavaApplicationProject_Form.cs** file,we are creating project file(<project_name>.slvjproj),writing project data to this file, creating directories (Project named directory in that classes,src,srcclasses directories).
In **MainForm.cs** file,we are reading created project file(<project_name>.slvjproj), Adding tabs to tabcontrol with text of created filename,creating that file & writing that file,also adding contents to ProjectExplorerTreeview.
Where tab is **MyTabPage** which is going to add to tabcontrol.(see MyTabPage.cs)
See following article for better understanding to add tabs

Creating Advanced Notepad in C#

Same actions to create New Java Applet Project, only one extra content to create means **index.html** file.

New Java Class

Now we need a class form that create a java class.To create class a java project must created or opened in IDE. following is the form that create or add java class to current opened project in the application.



Class Name :

Modifiers : [✓ public] private protected

abstract static final

Super Class :

Interface :

Methods : public static void main(String[] args) Create Constructor

Create Class with Java Themes :

Metal Motif Nimbus [✓ Windows] Windows Classic

In file **New_Class_Form.cs**, we are reading Class Name, also reading Modifiers,Super Class etc. Once Finish button is clicked, the tab is added to tabcontrol with same as class name by just adding .java to the end of it, and returning this filename with full path(**getCreatedFileName()** function).

In **MainForm.cs** file, In File_New_ClassMenuItem_Click() event, we will get this file name (getCreatedFileName()) , read contents from texteditor which is added to tabcontrol by **New_Class_Form.cs** file, creating this file and writing this contents to file using StreamWriter class.
Finally adding tag <JavaClassFile> & <VisualFile> with text as created filename.
Also adding <MainClassFile> tag, this tag text is read when we want to compile a file.

```
    private void File_New_ClassMenuItem_Click(object sender, EventArgs e)
    {
        MyTabPage tb = new MyTabPage(this);
        New_Class_Form ncf = new New_Class_Form(tb, myTabControl);
        ncf.ShowDialog();
        String filename = ncf.getCreatedFileName();
        int sel = myTabControl.SelectedIndex;

        if (sel != -1)
        {
            var mytexteditor = myTabControl.TabPages[sel].Controls[0];
            mytexteditor.ContextMenuStrip = textEditorContextMenuStrip;

            if (filename != "")
            {
                //check file name is already exists
                //if not then create that file
                if (File.Exists(filename))
                {
                    MessageBox.Show("The file name you entered is already exists in the folder or already
added to your project", "Error......");
                }
                else
                {
                    try
                    {
                        StreamWriter strw = new StreamWriter(File.Create(filename));
                        strw.Write(mytexteditor.Text);
                        strw.Close();
                        strw.Dispose();
                    }
                    catch
                    { }
                    //adding entry to current opened project file
                    //JavaClassFile & VisualFile
                    if (ReadCurrentProjectFileName() != "")
                    {
                        String projectfilename = ReadCurrentProjectFileName();
                        XmlDocument xmldoc = new XmlDocument();
                        xmldoc.Load(projectfilename);
                        XmlNode node = xmldoc.CreateNode(XmlNodeType.Element, "JavaClassFile", null);
                        node.InnerText = filename;
                        xmldoc.DocumentElement.AppendChild(node);
                        xmldoc.Save(projectfilename);

                        XmlDocument xmldoc2 = new XmlDocument();
                        xmldoc2.Load(projectfilename);
                        XmlNode node2 = xmldoc2.CreateNode(XmlNodeType.Element, "VisualFile", null);
                        node2.InnerText = filename;
                        xmldoc2.DocumentElement.AppendChild(node2);
                        xmldoc2.Save(projectfilename);
                    }
                }
            }
        }


        if (ncf.IsFinished() == true)
        {
            AddFilesToProjectExplorerTreeView();
            WriteCurrentFileNames();
            CopyAllSourceFilesToSRCFolder();
            UpdateWindowsList_WindowMenu();
            myTabControl_SelectedIndexChanged(sender, e);

            //check srcclasses directory exists or not
            if (Directory.Exists(getCurrentProjectLocationFolder() + "\\srcclasses"))
            {
                String mainclassfile = "";
                using (XmlReader reader = XmlReader.Create(ReadCurrentProjectFileName()))
                {
                    while (reader.Read())
                    {
                        if (reader.IsStartElement())
                        {
                            switch (reader.Name.ToString())
                            {
                                case "MainClassFile":
                                    mainclassfile = reader.ReadString();
                                    break;
                            }
                        }
                    }
                }
                if (mainclassfile == "")
                {
                    if (ReadCurrentProjectFileName() != "")
                    {
                        String projectfilename = ReadCurrentProjectFileName();
                        XmlDocument xmldoc = new XmlDocument();
                        xmldoc.Load(projectfilename);
                        XmlNode node = xmldoc.CreateNode(XmlNodeType.Element, "MainClassFile", null);
                        node.InnerText = filename;
                        xmldoc.DocumentElement.AppendChild(node);
                        xmldoc.Save(projectfilename);
                    }
                }
            }
        }
    }
```

New Package - Here we will create a folders in the project folder.when aaa.bbb.ccc package name is entered then we will replace . to \,(aaa\bbb\ccc) and will create these directories in current opened project location folder and also will create java class in aaa\bbb\ccc folder.

For other options like,
File->New->Interface, File->New->Enums
File->New->HTML File, File->New->CSS File
File->New->Text File, File->New->JavaScript File
File->New->SQL File, File->New->XML File
File->New->New File

Are same code as create New Java Class, just to create files with their own extensions.

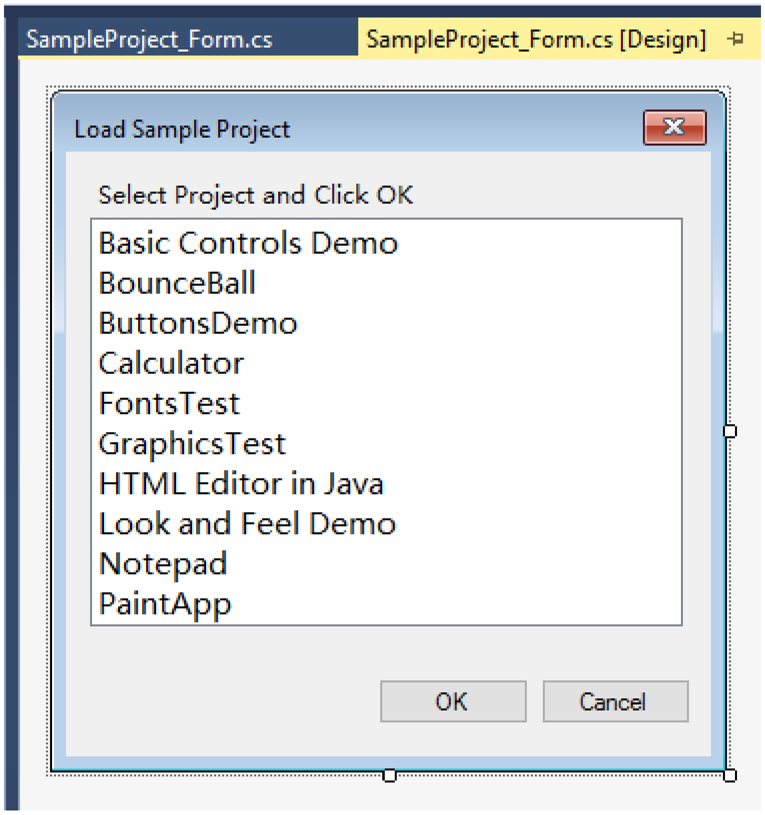
Open Project : (File->Open Project) This is nothing but the read file(.slvjproj) from OpenFileDialog, read contents from it, and perform actions about it such as adding application form text to project name text, adds tabs to tabcontrol by getting files by tags
<JavaClassFile>,<HTMLFile> etc. Once project is opened adding Project Type, ProjectName, Project Location Folder to \files\defaultprojloc.slvjfile.

OpenFiles : (File->Open Files) This is same as above(Open Project) only to read files, adding their names to ProjectExplorerTreeView, opening those files on tabs and adding all file names with their full path to \files\files.slvjfile.

Save : (File->Save) For saving a opened file on tab, I am using FilenameToolStripLabel to show/store current opened file name full path on tab. I am checking FilenameToolStripLabel contains ‘\’ or not, if it contains then read texteditor contents from current selected tab & write it to file from FilenameToolStripLabel.Text or here you can read files\files.slvjfile where contains all current opened project files with their full path.

Save All : (File->Save All) For this, iam reading a file \files\files.slvjfile, comparing each tab with each file name read from files.slvjfile, if it matches then setting that tab to selected, save that file by reading current set tab texteditor with filename

Load Sample Project



Select Project and Click OK

Basic Controls Demo
BounceBall
ButtonsDemo
Calculator
FontsTest
GraphicsTest
HTML Editor in Java
Look and Feel Demo
Notepad
PaintApp

I am providing some sample java projects here.
First design your sample project form for selecting what kind of project want to create.
Create sample project is nothing but to create a Project file(.slvjproj), Project directory, creating sub directories in it(src,classes,srcclasses), creating project files(.java) in those directories, adding this created project information to \files\defaultprojloc.slvjfile & just call OpenProject function with created project file name.
These project files are located at Samples folder.
First lets define variables to store what files need to be read from Samples folder.
Here I defined notepad_files(notepad source file), notepad_2(source file name we want to create in/as project)

```
String[] notepad_files = {
                    Application.StartupPath+"\\Samples\\Notepad\\Notepad.slvjfile"
                    };

String[] notepad_2 ={
                   "Notepad.java"
                 };


String defaultprojfilepath = Application.StartupPath + "\\files\\defaultprojloc.slvjfile";


public String ProjectLocaionFoder()
{
   String projectfolder = "";
   using (XmlReader reader = XmlReader.Create(defaultprojfilepath))
   {
      while (reader.Read())
      {
         if (reader.IsStartElement())
         {
            switch (reader.Name.ToString())
            {
               case "DefaultProjectLocation":
                  projectfolder = reader.ReadString();
                  break;
            }
         }
      }
   }

   return projectfolder;
}


      String selected_item = listBox1.SelectedItem.ToString();
      String projectfolder = ProjectLocaionFoder();
```

listBox1 is the control added to form (see above image).selected_item contain selected sample project name and projectfolder contains users project folder.

```
      if (selected_item == "Notepad")
      {
         project_name = "Notepad";
```

```
            String directory = projectfolder + "\\Notepad";
            String projectfile = "Notepad.slvjproj";

            if (Directory.Exists(directory))
            {
                MessageBox.Show("The selected project is already exists in current location",
"Error............");
            }

            else
            {

                Directory.CreateDirectory(directory);

                if (Directory.Exists(directory))
                {
                    project_folder = directory;

                    project_file = directory + "\\" + projectfile;
                }

                if (File.Exists(notepad_files[0]))
                {
                    String filename = notepad_files[0];
                    String fname = filename.Substring(filename.LastIndexOf("\\") + 1);
                    fname = fname.Remove(fname.Length - 9);
                    fname = fname + ".java";

                    if (Directory.Exists(directory))
                    {
                        String src = directory + "\\src";
                        String srcclasses = directory + "\\srcclasses";
                        String classes = directory + "\\classes";

                        Directory.CreateDirectory(src);
                        Directory.CreateDirectory(srcclasses);
                        Directory.CreateDirectory(classes);

                        String content = "";

                        content = File.ReadAllText(notepad_files[0]);

                        if (Directory.Exists(srcclasses))
                        {
                            StreamWriter strw = new StreamWriter(File.Create(srcclasses + "\\" + fname));
                            strw.Write(content);
                            strw.Close();
                            strw.Dispose();
                        }

                        //creating & writing slvjproj file
                        using (XmlWriter xmlwriter = XmlWriter.Create(directory + "\\" + projectfile))
                        {
                            xmlwriter.WriteStartDocument();
                            xmlwriter.WriteStartElement("SilverJProject");
                            xmlwriter.WriteString("\n");
                            xmlwriter.WriteComment("Silver-J (1.0) Java Application Project");
                            xmlwriter.WriteString("\n");
                            xmlwriter.WriteElementString("ProjectName", "Notepad");
                            xmlwriter.WriteString("\n");
                            xmlwriter.WriteElementString("ProjectLocationFolder", directory);
                            xmlwriter.WriteString("\n");
                            xmlwriter.WriteElementString("ProjectLocationFile", directory + "\\" + projectfile);
                            xmlwriter.WriteString("\n");
                            xmlwriter.WriteElementString("ProjectType", "ApplicationType");
                            xmlwriter.WriteString("\n");
                            xmlwriter.WriteElementString("MainClassFile", directory + "\\srcclasses" + "\\" +
notepad_2[0]);
                            xmlwriter.WriteString("\n");
                            xmlwriter.WriteElementString("JavaClassFile", directory + "\\srcclasses" + "\\" +
notepad_2[0]);
                            xmlwriter.WriteString("\n");
                            xmlwriter.WriteElementString("VisualFile", directory + "\\srcclasses" + "\\" +
notepad_2[0]);
                            xmlwriter.WriteEndElement();
                            xmlwriter.WriteEndDocument();
                            xmlwriter.Close();
                        }

                    }
                }

                is_project_created = true;
            }
}
```

Here we are just performing actions same as creating new Java application project with Create Java Class checkbox is checked.

Compile & Run

Compiling using IDE means run commands through processes. But here to compile a java source file we dont need any window to show, we just need output of compilation such as errors & warnings. It is easy to create process. We need boolean value whether processes is started or not for compilation.
Consider you are compiling your java programs using cmd, same we are doing here.

```
e.g:    cd C:\MyJava\Hello
        javac.exe HelloWorld.java
```

See following code;
Where EXE is application javac.exe with full path
WorkingDirectory is our source file directory(srcclasses)(above e.g C:\MyJava\Hello)
FileName is our main java source file which we want to compile(HelloWorld.java)
We need only one java source file to compile, you can provide more files.
The main source file is saved in the project file(.slvjproj) with tag <MainClassFile>. This file can be changed by going to Run->Main Class

```
public bool Compile(String EXE, String WorkingDirectory, String FileName)
{
   bool processStarted = false;

   if (File.Exists(EXE))
   {
      process.StartInfo.FileName = EXE;
      process.StartInfo.Arguments = FileName;
      process.StartInfo.WorkingDirectory = WorkingDirectory;
      process.StartInfo.CreateNoWindow = true;
      process.StartInfo.ErrorDialog = false;
      process.StartInfo.UseShellExecute = false;
      process.StartInfo.RedirectStandardOutput = true;
      process.StartInfo.RedirectStandardError = true;
      processStarted = process.Start();
   }
```

```
            else
            {
                MessageBox.Show("Unable to compile java file. Check your Java Path settings: Current Java
Path : ");
            }
            return processStarted;
        }



        public void CompileJava(String file, String jdkpath)
        {
            String mystr = file;
            if (mystr.Contains(".java"))
            {
                mystr = mystr.Remove(mystr.Length - 5);
            }
            if (this.Compile(jdkpath + "\\javac.exe", Path.GetDirectoryName(file), Path.GetFileName(file)))
            {
                ErrorReader = process.StandardError;
                string response = ErrorReader.ReadToEnd();

                if (response != "")
                {
                    ErrorTextBox.Text = response;
                    if (showErrorDialog == true)
                    {
                        MessageBox.Show("" + response, "Errors");
                    }
                }
                else if (response == "")
                {
                    ErrorTextBox.Text = "Program Compiled Successfully.................!";
                }
                else if (response.Contains("uses or overrides a deprecated API."))
                {
                    ErrorTextBox.Text = "Program Compiled Successfully.................!";
                }
                else
                {
                    ErrorTextBox.Text = "Program Compiled Successfully.................!";
                }
            }
            else if (File.Exists("" + mystr + ".class"))
            {
                ErrorTextBox.Text = "Program Compiled Successfully.................!";
            }
```

```
    //set caret position to error line number
    if(myTabControl.TabCount>0)
    {
       if(myTabControl.SelectedTab.Text.Contains(".java"))
       {
          String compilejavafilename = myTabControl.SelectedTab.Text;
          if (ErrorTextBox.Text.Contains(compilejavafilename))
          {
             int select_index = myTabControl.SelectedIndex;
             var texteditor = (TextEditorControl)myTabControl.TabPages[select_index].Controls[0];
             RichTextBox rtb = new RichTextBox();
             rtb.Text = texteditor.Text;

             for(int i=0;i<rtb.Lines.Length;i++)
             {
                if(ErrorTextBox.Lines[0].Contains(i.ToString()))
                {
                   texteditor.ActiveTextAreaControl.TextArea.Caret.Line = i-1;
                }
             }
          }
       }
    }
 }
```

showErrorDialog is to hold whether to show dialog after clicking on compile menu item. If error found then show it, and set caret to that error line number.

You know how to run java programs on cmd.
First you compile java source file, it will create .class file by going to the location folder. And then you run that class file using java.exe command.

```
e.g:    cd C:\MyJava\Hello
        java.exe HelloWorld.class
```

Instead of using only java.exe, I am using full path of java.exe file(jdk_path\bin\java.exe) as a string.

```
String projectlocationfolder = getCurrentProjectLocationFolder();
if (Directory.Exists(projectlocationfolder + "\\srcclasses"))
{
   if (filename.Contains(".java"))
```

```
        {
            String ffname = filename.Remove(filename.Length - 5);
            ffname = ffname + ".class";

            if (File.Exists(ffname))
            {
                ProcessStartInfo ProcessInfo;
                //Process process;
                String javapath = "\"" + jdkpath + "\\java.exe" + "\"";
                String getfilename = filename.Substring(filename.LastIndexOf("\\") + 1);
                String fname = "";
                if (getfilename.Contains(".java"))
                {
                    fname = getfilename.Remove(getfilename.Length - 5);
                }
                ProcessInfo = new ProcessStartInfo("cmd.exe", "/K" + "  cd/   &&  cd " + workingDirectory
+ "  &&  " + javapath + "  " + fname);
                ProcessInfo.CreateNoWindow = true;
                ProcessInfo.UseShellExecute = true;
                Process.Start(ProcessInfo);
            }
        }
    }
```

```
                ProcessInfo = new ProcessStartInfo("cmd.exe", "/K" + "  cd/
&&  cd " + workingDirectory + "  &&  " + javapath + "  " + fname);
```

This line run the program.first it starts the cmd.exe with command cd/.
Then goes to project directory(workingDirectory(srcclasses)) by using cd.
Then call java.exe filename full path with .class filename.
Same for **Run with Parameter** option just adding those parameters after fname.
Same for **Run Applet** option only just to use appletviewer.exe with HTML source file.

Build Executable Jar

To create executable jar file we need a manifest file(manifest.mf). that file contain following contents.

```
Manifest-Version: 1.0
Ant-Version: Apache Ant 1.9.4
Created-By: 1.8.0_25-b18 (Oracle Corporation)
Class-Path:
X-COMMENT: Main-Class will be added automatically by build
Main-Class:
Main-Class:
```

Where Main-Class: should contain mailclass filename(only filename without any file extension).
Created-By: is the JDK version.

To create jar file using cmd,
First go to working directory.

```
jar.exe cmf <manifest_file> <outputfile> <class_files>
```

e.g:

```
jar.exe cmf manifest.mf Hello.jar HelloWorld.class
```

In Run_BuildMenuItem_Click(), I have used a textbox to paste all .class files in it and use textbox.Text as class files. It will not add .java source files.
It also adds directories to jar file by adding directory names to textbox.
The process is same as running java programs, just changing in commands, inputs & arguments.
Once jar file is created, we will create a **build** folder in project location folder & copy created jar file to there.

```
private void Run_BuildMenuItem_Click(object sender, EventArgs e)
 {
    String projectfolderpath = getCurrentProjectLocationFolder();
    if (projectfolderpath != "")
    {
       if (Directory.Exists(projectfolderpath + "\\srcclasses"))
       {
          String mainclass = getMainClassFileName();
          String mnclass = mainclass.Substring(mainclass.LastIndexOf("\\") + 1);
          mnclass = mnclass.Remove(mnclass.Length - 5);
          //create mainclass manifest file
          String maintexttomanifest = "Manifest-Version: 1.0"
             + "\nAnt-Version: Apache Ant 1.9.4"
             + "\nCreated-By: 1.8.0_25-b18 (Oracle Corporation)"
             + "\nClass-Path: "
             + "\nX-COMMENT: Main-Class will be added automatically by build"
             + "\nMain-Class: " + mnclass
             + "\nMain-Class: " + mnclass;

          String manifestfile = projectfolderpath + "\\srcclasses\\mainclass.mf";
          StreamWriter strw = new StreamWriter(manifestfile);
          strw.Write(maintexttomanifest);
          strw.Close();
          strw.Dispose();

          if (File.Exists(manifestfile))
          {
             String srcclassesfolderpath = projectfolderpath + "\\srcclasses";
             List<String> packagenameslist = getPackageNamesList();
             TextBox textbox = new TextBox();
             string[] files = Directory.GetFiles(srcclassesfolderpath);

             foreach (string filePath in files)
             {
                if (Path.GetExtension(filePath) != ".java" && Path.GetExtension(filePath) != ".mf")
                {
                   textbox.Paste(" " + Path.GetFileName(filePath));
                }
             }

             //check package folder exists or not
             RichTextBox rtb = new RichTextBox();
             String[] dirs = Directory.GetDirectories(srcclassesfolderpath);
             foreach (String packagefolder in packagenameslist)
             {
                rtb.Text = rtb.Text.Insert(rtb.SelectionStart, "" + packagefolder + "  ");
```

```
            }

            //get folder by leaving package folders
            foreach (String dirsname in dirs)
            {
                String dirs2 = dirsname.Substring(dirsname.LastIndexOf("\\") + 1);
                if (rtb.Text.Contains(dirs2)) { }

                else
                {
                    textbox.Paste(" " + dirs2 + "* ");
                }
            }


            //create jar file
            if (File.Exists(manifestfile))
            {
                String jarfilepath = "\"" + getJDKPath() + "\\jar.exe" + "\"";
                String prjfolder = getCurrentProjectLocationFolder() + "\\srcclasses";
                String manifestfilename = manifestfile.Substring(manifestfile.LastIndexOf("\\") + 1);
                String classes = textbox.Text;
                String mainclassfile = mainclass.Substring(mainclass.LastIndexOf("\\") + 1);
                mainclassfile = mainclassfile.Remove(mainclassfile.Length - 5);
                String jarfilename = mainclassfile + ".jar";

                ProcessStartInfo ProcessInfo;
                Process Process;
                ProcessInfo = new ProcessStartInfo("cmd.exe", "/K " + "cd/  &&  cd  " + prjfolder + " &&
" + jarfilepath + "  cmf   " + manifestfilename + "  " + jarfilename + "   " + classes);
                ProcessInfo.CreateNoWindow = true;
                ProcessInfo.UseShellExecute = true;
                ProcessInfo.WindowStyle = ProcessWindowStyle.Hidden;
                Process = Process.Start(ProcessInfo);
            }

            //create build folder and moved created jar file to build folder
            String createdjarfile = mainclass.Remove(mainclass.Length - 5) + ".jar";
            if (File.Exists(createdjarfile))
            {
                String buildfolder = getCurrentProjectLocationFolder() + "\\build";

                if (Directory.Exists(buildfolder))
                {
                    String createdjarfile2 = createdjarfile.Replace("srcclasses", "build");
                    File.Delete(createdjarfile2);
                    File.Copy(createdjarfile, createdjarfile2);
```

```
                        try
                        {
                            File.Delete(createdjarfile);
                        }
                        catch { }

                        ShowAboutToolStripLabel.Text = "Build Completed";
                        Run_BuildMenuItem_Click(sender, e);
                    }
                    else
                    {
                        Directory.CreateDirectory(buildfolder);
                        String createdjarfile2 = createdjarfile.Replace("srcclasses", "build");
                        File.Copy(createdjarfile, createdjarfile2);

                        ShowAboutToolStripLabel.Text = "Build Completed";
                        Run_BuildMenuItem_Click(sender, e);
                    }
                }
            }
        }
    }
}
```

Appearance

IDE also has different themes. Here I am providing Default, System, Light,Dark & Night appearances.
Appearance means just changing colors of the components.
For MenuStrip & ToolStrip we need to write code for rendering their appearance by drawing. these renderer classes are defined in MyRenderer.cs file.
I have created MyMenuStripZ, MyToolStripZ classes inheriting MenuStrip & ToolStrip & adding renderers to it from fileMyRenderer.cs.
Here the code that set Night theme to IDE

```
if(type=="Night")
  {
     MyMenuStripZ.Renderer = new NightMenuRenderer();
     textEditorContextMenuStrip.Renderer = new NightMenuRenderer();
     ProjectExplorerTreeViewContextMenuStrip.Renderer = new NightMenuRenderer();
     myTabControlContextMenuStrip.Renderer = new NightMenuRenderer();
     ErrorsListContextMenuStrip.Renderer = new NightMenuRenderer();
     MyToolStripZ.Renderer = new NightToolStripRenderer();
     myTabControl.DrawMode = TabDrawMode.OwnerDrawFixed;

     myTabControl.Transparent1 = 255;
     myTabControl.Transparent2 = 255;
     toolstrippanel.Transparent1 = 255;
     toolstrippanel.Transparent2 = 255;
     projectexplorerpanel.Transparent1 = 255;
     projectexplorerpanel.Transparent2 = 255;
     classespanel.Transparent1 = 255;
     classespanel.Transparent2 = 255;
     methodspanel.Transparent1 = 255;
     methodspanel.Transparent2 = 255;
     errorslistpanel.Transparent1 = 255;
     errorslistpanel.Transparent2 = 255;

     MyMenuStripZ.BackColor = Color.FromArgb(255, 30, 30, 30);
     toolstrippanel.StartColor = Color.FromArgb(30, 30, 30);
     toolstrippanel.EndColor = Color.FromArgb(30, 30, 30);
     MyToolStripZ.BackColor = Color.FromArgb(250, 30, 30, 30);

     projectexplorerpanel.StartColor = Color.FromArgb(15, 15, 15);
     projectexplorerpanel.EndColor = Color.FromArgb(15, 15, 15);
```

```
    classespanel.StartColor = Color.FromArgb(15, 15, 15);
    classespanel.EndColor = Color.FromArgb(15, 15, 15);
    methodspanel.StartColor = Color.FromArgb(15, 15, 15);
    methodspanel.EndColor = Color.FromArgb(15, 15, 15);
    errorslistpanel.StartColor = Color.FromArgb(15, 15, 15);
    errorslistpanel.EndColor = Color.FromArgb(15, 15, 15);

    projectexplorerlabel.ForeColor = Color.White;
    classeslabel.ForeColor = Color.White;
    methodslabel.ForeColor = Color.White;
    errorslabel.ForeColor = Color.White;

    myTabControl.ActiveTabStartColor = Color.FromArgb(10, 10, 30);
    myTabControl.ActiveTabEndColor = Color.FromArgb(10, 10, 30);
    myTabControl.NonActiveTabStartColor = Color.FromArgb(60, 60, 60);
    myTabControl.NonActiveTabEndColor = Color.FromArgb(60, 60, 60);

    myTabControl.TextColor = Color.White;

    splitContainer1.BackColor = Color.FromArgb(255,10, 10, 10);

    ProjectExplorerTreeView.BackColor = Color.FromArgb(255, 25, 25, 25);
    ClassesTreeView.BackColor = Color.FromArgb(255, 25, 25, 25);
    MethodsTreeView.BackColor = Color.FromArgb(255, 25, 25, 25);
    ErrorTextBox.BackColor = Color.FromArgb(255, 25, 25, 25);

    ProjectExplorerTreeView.ForeColor = Color.White;
    ClassesTreeView.ForeColor = Color.White;
    MethodsTreeView.ForeColor = Color.White;
    ErrorTextBox.ForeColor = Color.FromArgb(255, 255, 220, 220);
}
```

File Association

To associate our project files(.slvjproj) files or to open project by clicking on project file, we need to identify whether provided argument contains project file or not(.slvjproj), if contain then call OpenProject function. Here I have declared OpenProject(string filename) function otherwise it will call OpenFiles_FromCMD(args) function. These are defined in Program.cs file.

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;
using System.IO;
using System.Threading;
namespace Silver_J
{
   static class Program
   {
      /// <summary>
      /// The main entry point for the application.
      /// </summary>
      [STAThread]
      static void Main(String[] args)
      {
         if (args != null && args.Length > 0)
         {
            if (args.Length == 1)
            {
               String file = args[0];
               //when filename extension is .slvjproj then call OpenProject function
               //if file is other type then call OpenFiles_FromCMD
               if (Path.GetExtension(file) == ".slvjproj")
               {
                  Application.Run(new Start_Form());
                  MainForm mf = new MainForm();
                  mf.OpenProject(file);
                  Application.EnableVisualStyles();
                  Application.Run(mf);
               }
               else
               {
                  Application.Run(new Start_Form());
```

```
                        MainForm mf = new MainForm();
                        mf.OpenFiles_FromCMD(args);
                        Application.EnableVisualStyles();
                        Application.Run(mf);
                    }
                }
                else
                {
                    Application.Run(new Start_Form());
                    MainForm mf = new MainForm();
                    mf.OpenFiles_FromCMD(args);
                    Application.EnableVisualStyles();
                    Application.Run(mf);

                }
            }
            else
            {
                Application.EnableVisualStyles();
                Application.SetCompatibleTextRenderingDefault(false);
                Application.Run(new Start_Form());
                Application.Run(new MainForm());
            }
        }
    }
}
```