

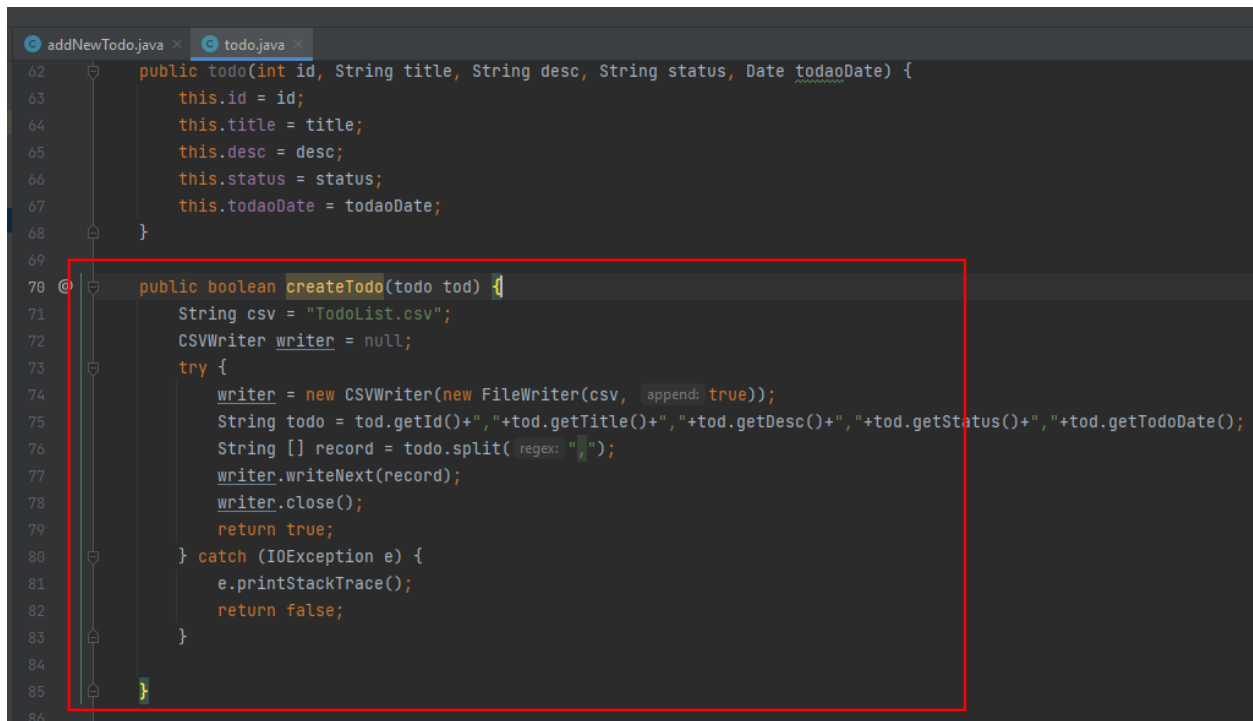
Function add new Todo into Todo List

```
addNewTodo.java
24 public addNewTodo() {
25     addButton.addActionListener(new ActionListener() {
26         @Override
27         public void actionPerformed(ActionEvent e) {
28
29         }
30     });
31     addButton.addActionListener(new ActionListener() {
32         @Override
33         public void actionPerformed(ActionEvent e) {
34             int Id = (int) (new Date().getTime()/1000);
35             String todoTitle = txtTodoTitle.getText();
36             String todoDesc = txtTodoDesc.getText();
37             String todoDate = txtDate.getText();
38             String todoStatus = cbTodoStatus.getSelectedItem().toString();
39             todo tod = new todo();
40             tod.setId(Id);
41             tod.setTitle(todoTitle);
42             tod.setDesc(todoDesc);
43             tod.setStatus(todoStatus);
44             Date date1= null;
45             try {
46                 date1 = new SimpleDateFormat( pattern: "YYYY-MM-DD").parse(todoDate);
47                 tod.setTodoDate(date1);
48                 tod.createTodo(tod);
49                 JOptionPane.showMessageDialog( parentComponent: null, message: "add successfully")
50             } catch (ParseException ex) {
51                 ex.printStackTrace();
52                 meesg.setText(ex.toString());
53                 JOptionPane.showMessageDialog( parentComponent: null,ex);
54             }
55         }
56     });
57 }
58 }
59 }
```

Get value from Form

Show message after  
insert data

Function write data into CSV file.



The screenshot shows an IDE with two tabs: 'addNewTodo.java' and 'todo.java'. The 'todo.java' tab is active, displaying the following Java code:

```
62 public todo(int id, String title, String desc, String status, Date todaoDate) {
63     this.id = id;
64     this.title = title;
65     this.desc = desc;
66     this.status = status;
67     this.todaoDate = todaoDate;
68 }
69
70 public boolean createTodo(todo tod) {
71     String csv = "TodoList.csv";
72     CSVWriter writer = null;
73     try {
74         writer = new CSVWriter(new FileWriter(csv, append: true));
75         String todo = tod.getId()+","+tod.getTitle()+","+tod.getDesc()+","+tod.getStatus()+","+tod.getTodoDate();
76         String [] record = todo.split(regex: ",");
77         writer.writeNext(record);
78         writer.close();
79         return true;
80     } catch (IOException e) {
81         e.printStackTrace();
82         return false;
83     }
84 }
85 }
```

The code defines a constructor for the 'todo' class and a 'createTodo' method. The 'createTodo' method writes the data of a 'todo' object to a CSV file named 'TodoList.csv'. It uses a 'CSVWriter' and a 'FileWriter' to write the data. The data is formatted as a string with commas separating the fields, and then split into an array of strings before being written to the CSV file. The method returns 'true' if the write operation is successful and 'false' if it fails due to an 'IOException'.

## Function Read Data from CSV file

```
7 @ public List<todo> readFileTodoList(String fileName) throws IOException {
8     if (fileName.isEmpty()){
9         fileName = "TodoList.csv";
10    }
11    List<todo> result = new ArrayList<>();
12    BufferedReader br = new BufferedReader(new FileReader(new File(fileName)));
13    try {
14        // Read first line
15        String line = br.readLine();
16        // Make sure file has correct headers
17        if (line==null) throw new IllegalArgumentException("File is empty");
18
19        // Run through following lines
20        while ((line = br.readLine()) != null) {
21            // Break line into entries using comma
22            String[] items = line.split( regex: "," );
23            try {
24                // If there are too many entries, throw a dummy exception, if
25                // there are too few, the same exception will be thrown later
26                if (items.length>4) throw new ArrayIndexOutOfBoundsException();
27                // Convert data to person record
28                todo tod = new todo();
29                tod.setId(Integer.valueOf(removeFirstandLast(items[0])));
30                tod.setTitle (removeFirstandLast(items[1]));
31                tod.setDesc(removeFirstandLast(items[2]));
32                tod.setStatus(removeFirstandLast(items[3]));
33                String sDate1=removeFirstandLast(items[4]);
34                Date date1=new SimpleDateFormat( pattern: "YYYY-MM-DD").parse(sDate1);
35                tod.setTodoDate(date1);
36                result.add(tod);
37            } catch (ArrayIndexOutOfBoundsException | NumberFormatException | NullPointerException | ParseException e) {
38                // Caught errors indicate a problem with data format -> Print warning and continue
39                System.out.println("Invalid line: "+ line);
40            }
41        }
42    }
43 }
```

## Function open Form

```
addNewTodo.java x todo.java x loginForm.java x
16 private JTextField txtUserName;
17 private JPasswordField txtPassword;
18 private JButton btnLogin;
19 private JLabel message;
20
21 public loginForm() {
22     btnLogin.addActionListener(new ActionListener() {
23         @Override
24         public void actionPerformed(ActionEvent e) {
25
26             String username = txtUserName.getText();
27             String password = txtPassword.getText();
28             if(new user().isLoggingSuceess(username,password)==true){
29                 new listTodos().showListTodos();
30             }else {
31                 message.setText("Login False");
32             }
33         }
34     });
35 }
36
37
38 public void showLoginForm(){
39     JFrame frame = new JFrame( title: "Login Form");
40     frame.setPreferredSize( new Dimension( width: 640, height: 480 ) );
41     frame.setContentPane(new loginForm().loginPanel);
42     frame.pack();
43     frame.setVisible(true);
44 }
45 }
```