# *Національний технічний університет України*

***«Київський політехнічний інститут»***

#### Факультет інформатики та обчислювальної техніки

### *Кафедра обчислювальної техніки*

## КУРСОВА РОБОТА

*з дисципліни "****Інженерія програмного забезпечення****"*

***Виконав:*** *Стасюк Максим Вікторович*

***Факультет***  *ІОТ*

***Група*** *ІО-92,* ***Залікова книжка №*** *9225*

***Перевірив: Болдак***

***Київ 2011***

**Зміст**

Вступ………………………………………………………………………………….3

1. Технічне завдання……………………………………………………………….4

1.1 Призначення розроблюваного проекту…………………………………….4

1.2 Вхідні данні…………………………………………………………………….4

* 1. Функціональність…………………………………………………………….4

2. Проектування інтерфейсу користувача………………………………………….6

2.1 Загальний огляд інтерфейсу…………………………………………………….6

* 1. Сценарії взаємодії…………………………………………………………….6

2.2.1 Початковий опис потоку подій роботи з "Diagram manager" ……………...6

2.2.2 Розробка орієнтирів …………………………………………………………..7

2.2.3 Розробка атрибутів……………………………………………………………7

2.2.4 Розробка інтенсивності використання……………………………………….9

2.2.5 Результуючий опис потоку подій…………………………………………….9

2.3 Діаграма граничних класів…………………………………………………….10

2.4 Зовнішній вигляд програми……………………………………………………10

2.5 Компоненти бібліотек SWING, що були використані в програмі…………..12

2.6 UML діаграма класів…………………………………………………………...13

3 Документація……………………………………………………………………..13

3.1 Документація для розробника…………………………………………………13

3.2 Документація для користувача……………………………………………….69

3.2.1. Системні вимоги……………………………………………………………69

3.2.2. Запуск програми……………………………………………………………69

3.2.2 Завантаження таблиці………………………………………………………69

3.2.3 Збереження таблиці…………………………………………………………69

3.2.4 Збереження графіка…………………………………………………………69

3.2.5 Закриття таблиці…………………………………………………………….69

3.2.6 Операції з даними…………………………………………………………...69

Висновок…………………………………………………………………………70

Литература………………………………………………………………………71

Додаток А…………………………………………………………………………72

**Вступ**

Розроблена програма призначена для роботи з csv таблицями, а також для зміни та візуалізації даних у вигляді кругової. Програма дозволяє завантажувати дані з csv таблиці, редагувати їх, а також зберегти змінену таблицю і побудовану діаграму. Програма допомагає людині візуально сприймати дані, для цього можливо змінювати колір даних.

Програмний продукт може бути застосований у будь-якій сфері де необхідно оперувати з табличними даними, наприклад у банківській сфері можливо зручно відковувати розподіл капіталу у певній сфері діяльності.

1. **Технічне завдання**

**1.1 Призначення розроблюваного проекту.**

В курсовій роботі необхідно створити програму для обробки csv таблиць та візуалізації табличних даних. Програмний продукт може бути застосований у всіх галузях де використовується табличне представлення даних, наприклад у банківській сфері.

**1.2 Вхідні данні.**

Вхідними даними для програми є файл csv формату, з таблицею. Формат CSV-файлу є наступним:

* Файл складається з необмеженої кількості рядків та стовпчиків.
* Данні розділяються будь-яким роздільником, який вказується при завантажені таблиці, за замовчуванням використовується крапка з комою.

Варіант завдання. 8Г – Кільцева діаграма, змінюється таблиця зі зміною графічної області.

**1.3 Функціональність.**

Програма візуалізації має містити наступну функціональність:

* Можливість завантаження/збереження даних у табличний формі в форматі CSV, DAT
* Синтаксичний розбір і верифікація формату CSV з відбудовою внутрішньої моделі даних, в разі помилок – формування виключення (Exception)
* Інтерфейс користувача (на основі компонентів бібліотеки SWING), який містить дві основні області – табличну і графічну, а також допоміжні компоненти – меню, кнопки, діалоги, тощо. Таблична область містить дані, що завантажені з CSV-файлу, а графічна – їй інтерпретацію у вигляді діаграми згідно варіанту
* Можливість редагування даних (модифікація елемента) в графічній області з синхронною зміною табличної області
* Можливість збереження результату графічної інтерпретації – діаграми у вигляді файлу формату JPEG, PNG.
* Можливість зміни кольору елементів діаграми

**1.4 Вимоги до реалізації.**

* Мова програмування Java з використанням бібліотек SWING та AWT
* Інтерфейс користувача має забезпечувати доступ до всієї функціональності програми;
* Проект має бути повністю задокументований за допомогою JavaDoc
* Можливість роботи програми з більше ніж з однією серією даних (в одному чи різних CSV-файлах)

**Перелік текстової документації.**

1. Титульний лист
2. Технічне завдання
3. Пояснювальна записка
4. **Проектування інтерфейсу користувача**

**2.1 Загальний огляд інтерфейсу**

При розробці програми використовувалась мова програмування Java, тому її можна використовувати на будь-якій ОС.

Головне вікно програми розділено на дві функціональні зони:

* зона відображення діаграми;
* зона відображення даних у вигляді таблиці.
  1. **Сценарії взаємодії**

**2.2.1 Початковий опис потоку подій роботи з "Diagrammer"**

1. Прецедент починається коли користувач запускає програму.

2. Користувач може виконати одну з таких дій:

1. Завантажити таблицю даних, при завантаженні таблиці користувач може:

- Вибрати шлях до файлу

- Вибрати роздільники

- Вказати чи є заголовки колонок і стовпців

2. Змінити колір відображення даних.

3. Змінити значення будь-якої клітинки таблиці, або діаграму з

4. Зберегти таблицю в файл, при цьому користувач може:

- Вибрати шлях до файлу

- Вибрати тип файлу

5. Зберегти діаграму у вигляді картинки, при цьому користувач може:

- Вибрати шлях до файлу

- Вибрати тип файлу

6. Вийти з програми.

7. Переглянути розробників програми.

8. Прецедент закінчується, коли користувач завершує роботу з додатком.

**2.2.2 Розробка орієнтирів**

1. Прецедент починається коли користувач запускає додаток "Diagrammer".

2. Користувач може виконати одну з таких дій:

1. Завантажити таблицю даних, за якою автоматично буде побудована діаграма, при завантаженні таблиці користувач може:

- вибрати шлях до файлу

- Вибрати роздільники (Користувачеві повинні бути показані роздільники за замовчуванням)

- Вказати чи є заголовки колонок і стовпців

2. Змінити колір відображених даних

3. Змінити будь-яку клітинку таблиці

4. Зберегти таблицю в файл, при цьому користувач може:

- Вибрати шлях до файлу

- Вибрати тип

6. Зберегти діаграму у вигляді картинки, при цьому користувач може:

- Вибрати шлях до файлу

- Вибрати тип

7. Вийти з програми.

8. Переглянути розробників програми.

**2.2.3 Розробка атрибутів**

1. Прецедент починається коли користувач запускає програму.

2. Користувач може виконати одну з таких дій:

1. Завантажити таблицю даних, за якою автоматично буде побудована діаграма, при завантаженні таблиці користувач може:

- вибрати шлях до файлу

- Вибрати роздільники (в середньому для однієї таблиці використовується близько 2 роздільників)

- Вказати чи є заголовки колонок і стовпців (у 80% випадків заголовок є)

2. Змінити колір відображення кожного рядка вибравши зі

3. Змінити будь-яку клітинку таблиці

4. Зберегти таблицю в файл, при цьому користувач може:

- Вибрати шлях до файлу

5. Зберегти діаграму у вигляді картинки, при цьому користувач може:

- Вибрати шлях до файлу

6. Вийти з програми.

7. Переглянути розробників програми.

8. Прецедент закінчується, коли користувач завершує роботу з додатком.

**2.2.4 Розробка інтенсивності використання**

1. Прецедент починається коли користувач запускає програму.

2. Користувач може виконати одну з таких дій:

1. Завантажити таблицю даних, за якою автоматично буде побудована діаграма, при завантаженні таблиці користувач може:

- вибрати шлях до файлу

- Вибрати роздільники (в середньому для однієї таблиці використовується близько 3 роздільників)

- Вказати бути чи заголовки колонок і стовпців (Виконується в 80% випадків)

Виконується в 95% випадків.

2. Змінити колір відображення кожного рядка вибравши зі списку (Виконується в 50% випадків)

3. Змінити будь-яку клітинку таблиці (Змінюється в 90% випадків).

4. Зберегти таблицю в файл, при цьому користувач може:

- Вибрати шлях до файлу

(Застосовується в 50% випадків)

5. Зберегти діаграму у вигляді картинки, при цьому користувач може:

- Вибрати шлях до файлу

(Застосовується в 50% випадків)

6. Вийти з програми.

7. Переглянути розробників програми. (Застосовується в 10% випадків)

8. Прецедент закінчується, коли користувач завершує роботу з додатком.

**2.2.5 Результуючий опис потоку подій**

1. Прецедент починається коли користувач запускає програму.

2. Користувач може виконати одну з таких дій:

1. Завантажити таблицю даних, за якою автоматично буде побудована діаграма, при завантаженні таблиці користувач може:

- Вибрати шлях до файлу

- Вибрати роздільники (Користувачеві повинні бути показані роздільники за замовчуванням)

- Вказати бути чи заголовки колонок і стовпців (Виконується в 80% випадків)

(Виконується в 95% випадків.)

2. Змінити колір відображення кожного рядка вибравши зі списку (Виконується в 50% випадків)

3. Змінити будь-яку клітинку таблиці і автоматично змінитися діаграма. (Змінюється в 90% випадків)

4. Зберегти таблицю в файл, при цьому користувач може:

- Вибрати шлях до файлу

- Вибрати зберігати чи ні заголовки

- Вибрати тип файлу csv або dat (Виконується в 70 \ 30% випадків)

(Застосовується в 50% випадків)

5. Зберегти діаграму у вигляді картинки, при цьому користувач може:

- Вибрати шлях до файлу

- Вибрати тип файлу

(Застосовується в 50% випадків) (Показувати користувачеві доступні типи файлів)

6. Вийти з програми. (Якщо дані не збережені, то попередити користувача)

7. Переглянути розробників програми. (Застосовується в 10% випадків)

8. Переглянути довідку до програми. (Застосовується в 20% випадків)

9. Прецедент закінчується, коли користувач завершує роботу з додатком.

**2.3 Діаграма граничних класів**

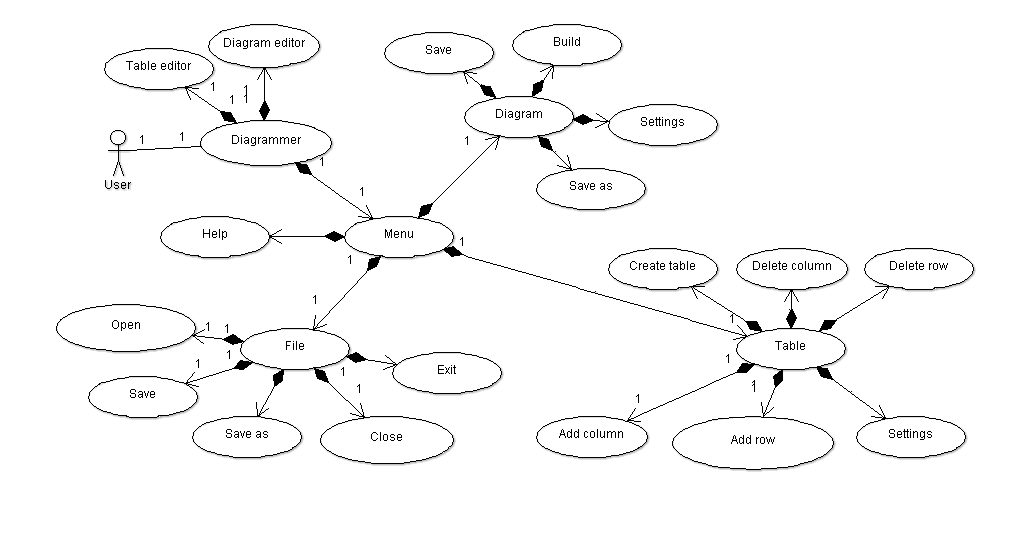
****

Рисунок 2.3 - Діаграма граничних класів

* 1. **Зовнішній вигляд програми**

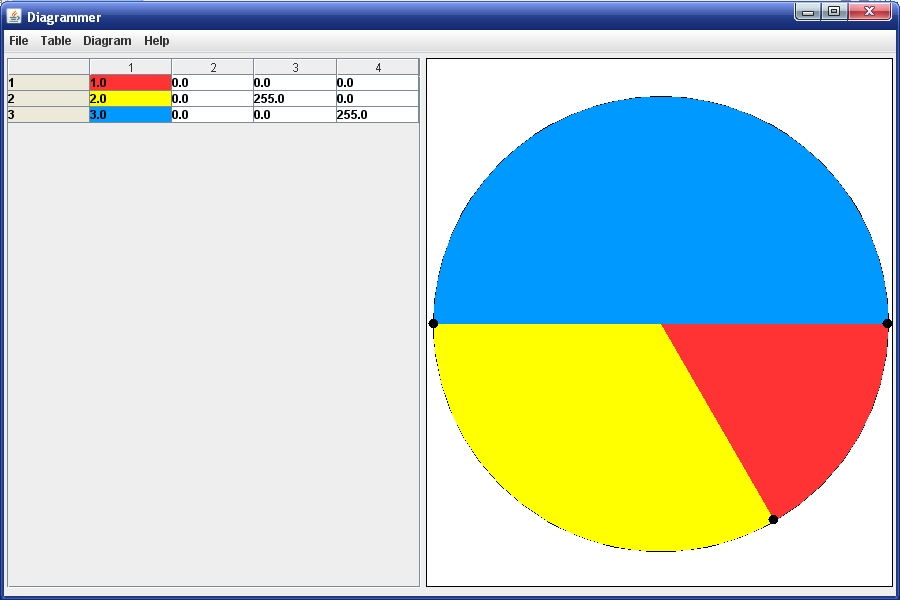


Рисунок 2.4.1 Головне вікно програми

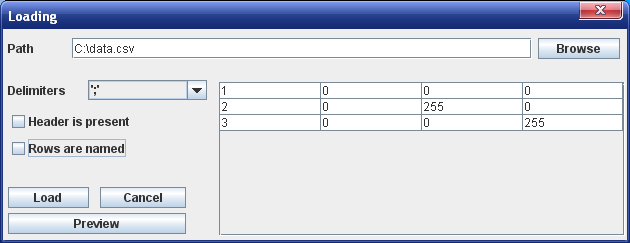


Рисунок 2.4.2 Вікно завантаження таблиці

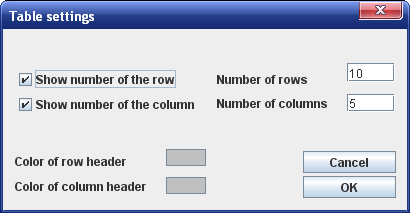


Рисунок 2.4.3 Вікно налаштувань таблиці

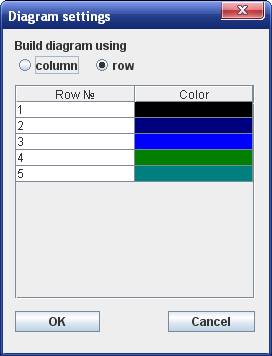


Рисунок 2.4.4 Вікно налаштувань діаграми

* 1. **Компоненти бібліотек SWING, що були використані в програмі**

|  |  |
| --- | --- |
| Головне вікно | javax.swing.JFrame |
| Головне меню | javax.swing.JMenuBar |
| Пункти меню | javax.swing.JMenu |
| Елементи меню | javax.swing.JMenuItem |
| Вибір кольору | javax.swing.JComboBox; |
| Кнопки | javax.swing.JButton |
| Таблиця | javax.swing.JTable |
| Панель для таблиці | javax.swing.JScrollPane |
| Панель для діаграми | javax.swing.JPanel |
| Діалоги | javax.swing.JDialog |
| Написи | javax.swing.JLabel |
| Текстові поля | javax.swing.JTextField |

* 1. **UML діаграма класів**

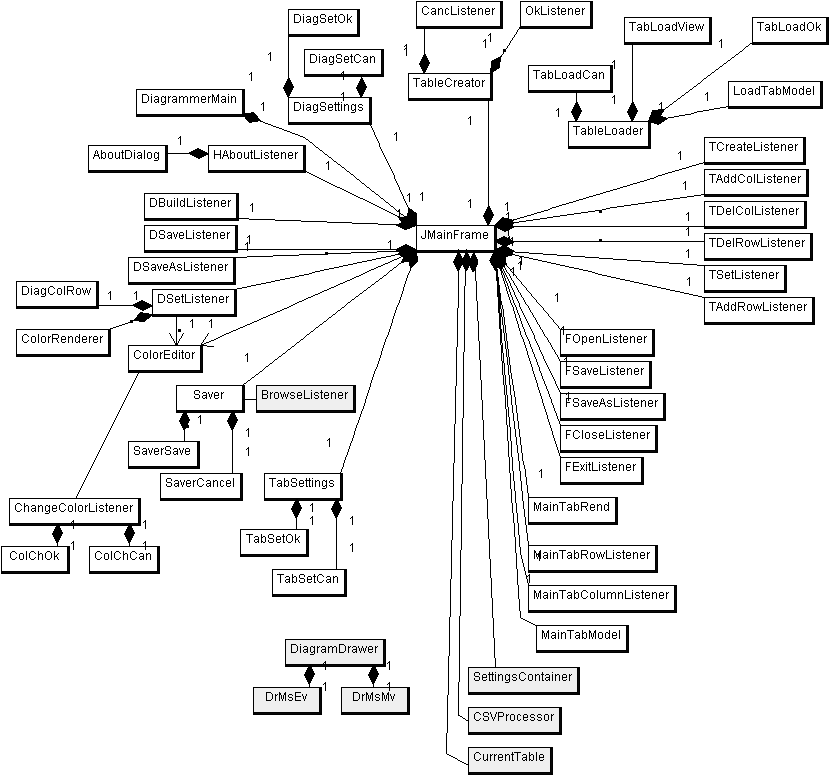


Рисунок 2.6.1- UML діаграма класів

1. **Документація**
   1. **Документація для розробника**

|  |  |
| --- | --- |
| **Packages** | |
| [**DiagCore**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\package-summary.html) |  |
| [**DiagrammerApp**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\package-summary.html) |  |

## Package DiagCore

|  |  |
| --- | --- |
| **Class Summary** | |
| [**BrowseListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\BrowseListener.html) |  |
| [**ChangeColorListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html) |  |
| [**ColChCan**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ColChCan.html) |  |
| [**ColChOk**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ColChOk.html) |  |
| [**CSVProcessor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html) |  |
| [**CurrentTable**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html) | class dedicated to store information about current table |
| [**DiagramDrawer**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html) | class dedicated to show diagram |
| [**DrMsEv**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DrMsEv.html) | Listener of a mouse |
| [**DrMsMv**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DrMsMv.html) | Listener of mouse |
| [**SettingsContainer**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html) |  |

## DiagCore Class BrowseListener

java.lang.Object

extended by **DiagCore.BrowseListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

public class **BrowseListener**

extends java.lang.Object

implements java.awt.event.ActionListener

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**BrowseListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\BrowseListener.html#BrowseListener(javax.swing.JTextField))(javax.swing.JTextField path) |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\BrowseListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0) |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### BrowseListener

public **BrowseListener**(javax.swing.JTextField path)

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagCore Class ChangeColorListener

java.lang.Object

extended by **DiagCore.ChangeColorListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

public class **ChangeColorListener**

extends java.lang.Object

implements java.awt.event.ActionListener

|  |  |
| --- | --- |
| **Field Summary** | |
| javax.swing.JButton | [**bt**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#bt) |
| boolean | [**change**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#change) |
| javax.swing.JColorChooser | [**col\_ch**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#col_ch) |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**ChangeColorListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#ChangeColorListener(javax.swing.JButton))(javax.swing.JButton but) |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0) |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### bt

public javax.swing.JButton **bt**

### change

public boolean **change**

### col\_ch

public javax.swing.JColorChooser **col\_ch**

|  |
| --- |
| **Constructor Detail** |

### ChangeColorListener

public **ChangeColorListener**(javax.swing.JButton but)

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagCore Class ChangeColorListener

java.lang.Object

extended by **DiagCore.ChangeColorListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

public class **ChangeColorListener**

extends java.lang.Object

implements java.awt.event.ActionListener

|  |  |
| --- | --- |
| **Field Summary** | |
| javax.swing.JButton | [**bt**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#bt) |
| boolean | [**change**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#change) |
| javax.swing.JColorChooser | [**col\_ch**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#col_ch) |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**ChangeColorListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#ChangeColorListener(javax.swing.JButton))(javax.swing.JButton but) |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\ChangeColorListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0) |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### bt

public javax.swing.JButton **bt**

### change

public boolean **change**

### col\_ch

public javax.swing.JColorChooser **col\_ch**

|  |
| --- |
| **Constructor Detail** |

### ChangeColorListener

public **ChangeColorListener**(javax.swing.JButton but)

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagCore Class CSVProcessor

java.lang.Object

extended by **DiagCore.CSVProcessor**

public class **CSVProcessor**

extends java.lang.Object

|  |  |
| --- | --- |
| **Field Summary** | |
| (package private)  java.util.ArrayList | [**dat\_lst**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#dat_lst)            array of rows |
| int | [**state**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#state) |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**CSVProcessor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#CSVProcessor())() |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**BuildTextData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#BuildTextData(java.lang.String[][]))(java.lang.String[][] dat) |
| java.lang.String | [**GetDelimeter**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#GetDelimeter())()            returns delimiter of the colomn |
| java.lang.String | [**GetTextData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#GetTextData(int))(int ind)            returns string with index ind. |
| int | [**GetTextDataSize**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#GetTextDataSize())()            returns number of strings in the list |
| boolean | [**LoadBinData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#LoadBinData(java.lang.String))(java.lang.String path)            deserialize data. path-contains path to file |
| boolean | [**LoadTextData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#LoadTextData(java.lang.String))(java.lang.String path)            loading data. path-contains path to file |
| java.lang.String[][] | [**ParseTextData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#ParseTextData())()            returns array of the strings |
| boolean | [**SaveBinData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#SaveBinData(java.lang.String))(java.lang.String path)            serialize data. path-contains path to file |
| boolean | [**SaveTextData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#SaveTextData(java.lang.String))(java.lang.String path)            saving data to file. path-contains path to file |
| void | [**SetDelimiter**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CSVProcessor.html#SetDelimiter(java.lang.String))(java.lang.String del)            sets delimiter of the colomn as del |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### dat\_lst

java.util.ArrayList **dat\_lst**

array of rows

### state

public int **state**

|  |
| --- |
| **Constructor Detail** |

### CSVProcessor

public **CSVProcessor**()

|  |
| --- |
| **Method Detail** |

### BuildTextData

public void **BuildTextData**(java.lang.String[][] dat)

### GetDelimeter

public java.lang.String **GetDelimeter**()

returns delimiter of the colomn

### GetTextData

public java.lang.String **GetTextData**(int ind)

returns string with index ind. If there is no such string will be returned null

### GetTextDataSize

public int **GetTextDataSize**()

returns number of strings in the list

### LoadBinData

public boolean **LoadBinData**(java.lang.String path)

deserialize data. path-contains path to file

### LoadTextData

public boolean **LoadTextData**(java.lang.String path)

loading data. path-contains path to file

### ParseTextData

public java.lang.String[][] **ParseTextData**()

returns array of the strings

### SaveBinData

public boolean **SaveBinData**(java.lang.String path)

serialize data. path-contains path to file

### SaveTextData

public boolean **SaveTextData**(java.lang.String path)

saving data to file. path-contains path to file

### SetDelimiter

public void **SetDelimiter**(java.lang.String del)

sets delimiter of the colomn as del

## DiagCore Class CurrentTable

java.lang.Object

extended by **DiagCore.CurrentTable**

public class **CurrentTable**

extends java.lang.Object

class dedicated to store information about current table

|  |  |
| --- | --- |
| **Field Summary** | |
| java.util.ArrayList | [**col\_head**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#col_head) |
| int | [**cols**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#cols) |
| java.util.ArrayList | [**data**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#data) |
| java.util.ArrayList | [**diag\_col**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#diag_col) |
| int | [**diag\_ind**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#diag_ind) |
| java.lang.String | [**file**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#file) |
| java.lang.String | [**gfile**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#gfile) |
| java.util.ArrayList | [**row\_head**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#row_head) |
| int | [**rows**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#rows) |
| boolean | [**text\_format**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#text_format) |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**addColumn**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#addColumn())()            deletes column |
| void | [**addDiagColor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#addDiagColor(java.awt.Color))(java.awt.Color col)            adds color col which is used in diagram |
| void | [**addRow**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#addRow())()            adds row |
| static void | [**createTable**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#createTable(boolean, boolean))(boolean col\_hd, boolean row\_hd)            creates table |
| static void | [**createTable**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#createTable(int, int, boolean, boolean))(int r, int c, boolean col\_hd, boolean row\_hd)            creates table |
| static void | [**deleteTable**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#deleteTable())()            deletes table |
| double | [**getData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#getData(int, int))(int r, int c)            getter for data |
| java.util.ArrayList | [**getDiagVector**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#getDiagVector())() |
| java.lang.String | [**getHeader**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#getHeader(int, boolean))(int i, boolean row)            getter for header |
| java.lang.String[][] | [**getInTextFormat**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#getInTextFormat())() |
| static [CurrentTable](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html) | [**getTable**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#getTable())() |
| boolean | [**removeColumn**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#removeColumn(int))(int c)            removes column c |
| boolean | [**removeDiagColor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#removeDiagColor(int))(int ind)            removes color ind which is used in diagram |
| boolean | [**removeRow**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#removeRow(int))(int r)            removes row |
| boolean | [**setData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#setData(int, int, double))(int r, int c, double val)            setter for data |
| boolean | [**setDiagVector**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#setDiagVector(java.util.ArrayList))(java.util.ArrayList dat)            sets data gotten from diagram |
| boolean | [**setHeader**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html#setHeader(int, boolean, java.lang.String))(int i, boolean row, java.lang.String val)            setter for header |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### col\_head

public java.util.ArrayList **col\_head**

### cols

public int **cols**

### data

public java.util.ArrayList **data**

### diag\_col

public java.util.ArrayList **diag\_col**

### diag\_ind

public int **diag\_ind**

### file

public java.lang.String **file**

### gfile

public java.lang.String **gfile**

### row\_head

public java.util.ArrayList **row\_head**

### rows

public int **rows**

### text\_format

public boolean **text\_format**

|  |
| --- |
| **Method Detail** |

### addColumn

public void **addColumn**()

deletes column

### addDiagColor

public void **addDiagColor**(java.awt.Color col)

adds color col which is used in diagram

### addRow

public void **addRow**()

adds row

### createTable

public static void **createTable**(boolean col\_hd,

boolean row\_hd)

creates table

**Parameters:**

col\_hd - are headers present?

row\_hd - are rows present?

### createTable

public static void **createTable**(int r,

int c,

boolean col\_hd,

boolean row\_hd)

creates table

**Parameters:**

r - number of rows

c - number of columns

col\_hd - defines if headers are present

row\_hd - defines if rows are named

### deleteTable

public static void **deleteTable**()

deletes table

### getData

public double **getData**(int r,

int c)

getter for data

### getDiagVector

public java.util.ArrayList **getDiagVector**()

**Returns:**

vector of data for building diagram

### getHeader

public java.lang.String **getHeader**(int i,

boolean row)

getter for header

### getInTextFormat

public java.lang.String[][] **getInTextFormat**()

**Returns:**

table as String [][]

### getTable

public static [CurrentTable](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\CurrentTable.html) **getTable**()

**Returns:**

current table

### removeColumn

public boolean **removeColumn**(int c)

removes column c

### removeDiagColor

public boolean **removeDiagColor**(int ind)

removes color ind which is used in diagram

### removeRow

public boolean **removeRow**(int r)

removes row

### setData

public boolean **setData**(int r,

int c,

double val)

setter for data

### setDiagVector

public boolean **setDiagVector**(java.util.ArrayList dat)

sets data gotten from diagram

### setHeader

public boolean **setHeader**(int i,

boolean row,

java.lang.String val)

setter for header

## DiagCore Class DiagramDrawer

java.lang.Object

extended by java.awt.Component

extended by java.awt.Container

extended by javax.swing.JComponent

extended by javax.swing.JPanel

extended by **DiagCore.DiagramDrawer**

**All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible

public class **DiagramDrawer**

extends javax.swing.JPanel

class dedicated to show diagram

**See Also:**

[Serialized Form](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\serialized-form.html#DiagCore.DiagramDrawer)

|  |
| --- |
| **Nested Class Summary** |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JPanel** |
| javax.swing.JPanel.AccessibleJPanel |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JComponent** |
| javax.swing.JComponent.AccessibleJComponent |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Container** |
| java.awt.Container.AccessibleAWTContainer |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Component** |
| java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy |

|  |  |
| --- | --- |
| **Field Summary** | |
| int[][] | [**ang**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#ang)            contains angles |
| boolean | [**builded**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#builded)            shows if diagram is shown |
| int | [**CHANGE\_ELLIPSE**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#CHANGE_ELLIPSE)            size of ellipse than is used to change angles |
| java.util.ArrayList | [**color**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#color)            colors which are used in diagram |
| int | [**d\_cx**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#d_cx) |
| int | [**d\_cy**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#d_cy) |
| int | [**d\_r**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#d_r) |
| java.util.ArrayList | [**data**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#data)            list of double values |
| boolean | [**have\_data**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#have_data)            shows if all data is defined |
| [JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) | [**mf**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#mf)            handle on main frame |
| int | [**sz\_sel**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#sz_sel) |
| java.awt.Rectangle[] | [**szrc**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#szrc)            array of ellipses than are used to change angles |

|  |
| --- |
| **Fields inherited from class javax.swing.JComponent** |
| accessibleContext, listenerList, TOOL\_TIP\_TEXT\_KEY, ui, UNDEFINED\_CONDITION, WHEN\_ANCESTOR\_OF\_FOCUSED\_COMPONENT, WHEN\_FOCUSED, WHEN\_IN\_FOCUSED\_WINDOW |

|  |
| --- |
| **Fields inherited from class java.awt.Component** |
| BOTTOM\_ALIGNMENT, CENTER\_ALIGNMENT, LEFT\_ALIGNMENT, RIGHT\_ALIGNMENT, TOP\_ALIGNMENT |

|  |
| --- |
| **Fields inherited from interface java.awt.image.ImageObserver** |
| ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**DiagramDrawer**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#DiagramDrawer())()            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**Clear**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#Clear())()            clears diagram and data |
| void | [**countData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#countData())()            counts data according to angles on diagram |
| boolean | [**draw**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#draw(java.awt.Graphics))(java.awt.Graphics g)            draws diagram |
| java.util.ArrayList | [**GetColor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#GetColor())()            getter for color |
| java.util.ArrayList | [**GetData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#GetData())()            getter for data |
| java.awt.Dimension | [**getSize**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#getSize())() |
| void | [**paint**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#paint(java.awt.Graphics))(java.awt.Graphics g) |
| int | [**radTodeg**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#radTodeg(double))(double rad)            converts angles in rad to deg |
| void | [**repaint**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#repaint(java.awt.Graphics))(java.awt.Graphics g) |
| void | [**setBounds**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#setBounds(int, int, int, int))(int x, int y, int width, int height)            sets bounds of diagram |
| boolean | [**SetColor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#SetColor(java.util.ArrayList))(java.util.ArrayList col)            setter for color |
| boolean | [**SetData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html#SetData(java.util.ArrayList))(java.util.ArrayList dat)            setter for data |

|  |
| --- |
| **Methods inherited from class javax.swing.JPanel** |
| getAccessibleContext, getUI, getUIClassID, paramString, setUI, updateUI |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### ang

public int[][] **ang**

contains angles

### builded

public boolean **builded**

shows if diagram is shown

### CHANGE\_ELLIPSE

public int **CHANGE\_ELLIPSE**

size of ellipse than is used to change angles

### color

public java.util.ArrayList **color**

colors which are used in diagram

### d\_cx

public int **d\_cx**

### d\_cy

public int **d\_cy**

### d\_r

public int **d\_r**

### data

public java.util.ArrayList **data**

list of double values

### have\_data

public boolean **have\_data**

shows if all data is defined

### mf

public [JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) **mf**

handle on main frame

### sz\_sel

public int **sz\_sel**

### szrc

public java.awt.Rectangle[] **szrc**

array of ellipses than are used to change angles

|  |
| --- |
| **Constructor Detail** |

### DiagramDrawer

public **DiagramDrawer**()

constructor

|  |
| --- |
| **Method Detail** |

### Clear

public void **Clear**()

clears diagram and data

### countData

public void **countData**()

counts data according to angles on diagram

### draw

public boolean **draw**(java.awt.Graphics g)

draws diagram

**Parameters:**

g - Graphics for drawing

**Returns:**

true if success

### GetColor

public java.util.ArrayList **GetColor**()

getter for color

### GetData

public java.util.ArrayList **GetData**()

getter for data

### getSize

public java.awt.Dimension **getSize**()

**Overrides:**

getSize in class java.awt.Component

### paint

public void **paint**(java.awt.Graphics g)

**Overrides:**

paint in class javax.swing.JComponent

### radTodeg

public int **radTodeg**(double rad)

converts angles in rad to deg

**Parameters:**

rad -

**Returns:**

deg value

### repaint

public void **repaint**(java.awt.Graphics g)

### setBounds

public void **setBounds**(int x,

int y,

int width,

int height)

sets bounds of diagram

**Overrides:**

setBounds in class java.awt.Component

### SetColor

public boolean **SetColor**(java.util.ArrayList col)

setter for color

### SetData

public boolean **SetData**(java.util.ArrayList dat)

setter for data

## DiagCore Class SettingsContainer

java.lang.Object

extended by **DiagCore.SettingsContainer**

**All Implemented Interfaces:**

java.io.Serializable

public class **SettingsContainer**

extends java.lang.Object

implements java.io.Serializable

**See Also:**

[Serialized Form](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\serialized-form.html#DiagCore.SettingsContainer)

|  |  |
| --- | --- |
| **Field Summary** | |
| java.awt.Color | [**col\_col**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#col_col) |
| boolean | [**col\_for\_diag**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#col_for_diag) |
| static java.lang.String[] | [**del\_val**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#del_val) |
| int | [**delim**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#delim) |
| static int | [**DELIM\_NUMBER**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#DELIM_NUMBER) |
| java.lang.String | [**lst\_file**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#lst_file) |
| java.awt.Color | [**row\_col**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#row_col) |
| boolean | [**sh\_col\_num**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#sh_col_num) |
| boolean | [**sh\_row\_num**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#sh_row_num) |

|  |  |
| --- | --- |
| **Method Summary** | |
| static void | [**createSettings**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#createSettings())() |
| static java.awt.Color | [**getDefaultDiagramColor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#getDefaultDiagramColor(java.util.ArrayList))(java.util.ArrayList col) |
| static java.util.ArrayList | [**getDefaultDiagramColors**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#getDefaultDiagramColors(int))(int col\_num) |
| static [SettingsContainer](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html) | [**getSettings**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#getSettings())() |
| static boolean | [**loadSettings**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#loadSettings(java.io.File))(java.io.File in) |
| static boolean | [**saveSettings**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html#saveSettings(java.io.File))(java.io.File out) |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### col\_col

public java.awt.Color **col\_col**

### col\_for\_diag

public boolean **col\_for\_diag**

### del\_val

public static final java.lang.String[] **del\_val**

### delim

public int **delim**

### DELIM\_NUMBER

public static final int **DELIM\_NUMBER**

**See Also:**

[Constant Field Values](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\constant-values.html#DiagCore.SettingsContainer.DELIM_NUMBER)

### lst\_file

public java.lang.String **lst\_file**

### row\_col

public java.awt.Color **row\_col**

### sh\_col\_num

public boolean **sh\_col\_num**

### sh\_row\_num

public boolean **sh\_row\_num**

|  |
| --- |
| **Method Detail** |

### createSettings

public static void **createSettings**()

### getDefaultDiagramColor

public static java.awt.Color **getDefaultDiagramColor**(java.util.ArrayList col)

### getDefaultDiagramColors

public static java.util.ArrayList **getDefaultDiagramColors**(int col\_num)

### getSettings

public static [SettingsContainer](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\SettingsContainer.html) **getSettings**()

### loadSettings

public static boolean **loadSettings**(java.io.File in)

### saveSettings

public static boolean **saveSettings**(java.io.File out)

## Package DiagrammerApp

|  |  |
| --- | --- |
| **Class Summary** | |
| [**AboutDialog**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\AboutDialog.html) | shows information about application |
| [**CancListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\CancListener.html) | listener for button Cancel |
| [**ColorEditor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorEditor.html) | default editor of cells |
| [**ColorRenderer**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorRenderer.html) | default renderer of a color |
| [**DBuildListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DBuildListener.html) | Listener of menu item Diagram->Build |
| [**DiagColRow**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagColRow.html) | listener of radio buttons |
| [**DiagrammerMain**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagrammerMain.html) |  |
| [**DiagSetCan**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSetCan.html) | listener of button Cancel |
| [**DiagSetOk**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSetOk.html) | lister of button OK |
| [**DiagSettings**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html) | class dedicated to show and change settings of diagram |
| [**DSaveAsListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DSaveAsListener.html) | Listener of menu item Diagram->SaveAs |
| [**DSaveListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DSaveListener.html) | Listener of menu item Diagram->Save |
| [**DSetListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DSetListener.html) | Listener of menu item Diagram->Settings |
| [**FCloseListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FCloseListener.html) | Listener of menu item File->Close |
| [**FExitListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html) | Listener of menu item File->Exit |
| [**FOpenListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FOpenListener.html) | Listener of menu item File->Open |
| [**FSaveAsListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FSaveAsListener.html) | Listener of menu item File->SaveAs |
| [**FSaveListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FSaveListener.html) | Listener of menu item File->Save |
| [**HAboutListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\HAboutListener.html) | Listener of menu item Help->About |
| [**JMainFrame**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) | class dedicated to show main window and interact with user |
| [**LoadTabModel**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\LoadTabModel.html) | model of table for preview |
| [**MainTabColumnListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabColumnListener.html) | Monitors changing of column selection |
| [**MainTabModel**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html) |  |
| [**MainTabRend**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabRend.html) |  |
| [**MainTabRowListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabRowListener.html) | Monitors changing of rows selection |
| [**OkListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\OkListener.html) | listener for button OK |
| [**Saver**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html) | dialog that provides possibility to save table or diagram |
| [**SaverCancel**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SaverCancel.html) | Listener of button Save |
| [**SaverSave**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SaverSave.html) | Listener of button Save |
| [**SetTabModel**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html) | Model of table used to show settings |
| [**TableCreator**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableCreator.html) | dialog than helps to create table |
| [**TableLoader**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html) | class that loads table |
| [**TabLoadCan**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabLoadCan.html) | listener for button cancel |
| [**TabLoadOk**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabLoadOk.html) | Listener for button Load |
| [**TabLoadView**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabLoadView.html) | Listener for button View |
| [**TabSetCan**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSetCan.html) | listener for button Cancel |
| [**TabSetOk**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSetOk.html) | listener for button ok |
| [**TabSettings**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSettings.html) | dialog that is used to change settings of a table |
| [**TAddColListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TAddColListener.html) | Listener of menu item Table->Add column |
| [**TAddRowListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TAddRowListener.html) | Listener of menu item Table->Add row |
| [**TCreateListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TCreateListener.html) | Listener of menu item Table->Create |
| [**TDelColListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TDelColListener.html) | Listener of menu item Table->Delete column |
| [**TDelRowListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TDelRowListener.html) | Listener of menu item Table->delete row |
| [**TSetListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TSetListener.html) | Listener of menu item Table->Settings |

## DiagrammerApp Class AboutDialog

java.lang.Object

extended by java.awt.Component

extended by java.awt.Container

extended by java.awt.Window

extended by java.awt.Dialog

extended by javax.swing.JDialog

extended by **DiagrammerApp.AboutDialog**

**All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

public class **AboutDialog**

extends javax.swing.JDialog

shows information about application

**See Also:**

[Serialized Form](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\serialized-form.html#DiagrammerApp.AboutDialog)

|  |
| --- |
| **Nested Class Summary** |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JDialog** |
| javax.swing.JDialog.AccessibleJDialog |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Dialog** |
| java.awt.Dialog.AccessibleAWTDialog, java.awt.Dialog.ModalExclusionType, java.awt.Dialog.ModalityType |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Window** |
| java.awt.Window.AccessibleAWTWindow |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Container** |
| java.awt.Container.AccessibleAWTContainer |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Component** |
| java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy |

|  |
| --- |
| **Field Summary** |

|  |
| --- |
| **Fields inherited from class javax.swing.JDialog** |
| accessibleContext, rootPane, rootPaneCheckingEnabled |

|  |
| --- |
| **Fields inherited from class java.awt.Dialog** |
| DEFAULT\_MODALITY\_TYPE |

|  |
| --- |
| **Fields inherited from class java.awt.Component** |
| BOTTOM\_ALIGNMENT, CENTER\_ALIGNMENT, LEFT\_ALIGNMENT, RIGHT\_ALIGNMENT, TOP\_ALIGNMENT |

|  |
| --- |
| **Fields inherited from interface javax.swing.WindowConstants** |
| DISPOSE\_ON\_CLOSE, DO\_NOTHING\_ON\_CLOSE, EXIT\_ON\_CLOSE, HIDE\_ON\_CLOSE |

|  |
| --- |
| **Fields inherited from interface java.awt.image.ImageObserver** |
| ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**AboutDialog**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\AboutDialog.html#AboutDialog())()            constructor |  |

|  |
| --- |
| **Constructor Detail** |

### AboutDialog

public **AboutDialog**()

constructor

## DiagrammerApp Class ColorEditor

java.lang.Object

extended by javax.swing.AbstractCellEditor

extended by **DiagrammerApp.ColorEditor**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.io.Serializable, java.util.EventListener, javax.swing.CellEditor, javax.swing.table.TableCellEditor

class **ColorEditor**

extends javax.swing.AbstractCellEditor

implements javax.swing.table.TableCellEditor, java.awt.event.ActionListener

default editor of cells

|  |
| --- |
| **Field Summary** |

|  |
| --- |
| **Fields inherited from class javax.swing.AbstractCellEditor** |
| changeEvent, listenerList |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**ColorEditor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorEditor.html#ColorEditor(DiagrammerApp.DiagSettings))([DiagSettings](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorEditor.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent e)            is called when user tries to edit cell |
| java.lang.Object | [**getCellEditorValue**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorEditor.html#getCellEditorValue())()            returns new value |
| java.awt.Component | [**getTableCellEditorComponent**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorEditor.html#getTableCellEditorComponent(javax.swing.JTable, java.lang.Object, boolean, int, int))(javax.swing.JTable table, java.lang.Object value, boolean isSelected, int row, int column)            returns cell editor |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Methods inherited from interface javax.swing.CellEditor** |
| addCellEditorListener, cancelCellEditing, isCellEditable, removeCellEditorListener, shouldSelectCell, stopCellEditing |

|  |
| --- |
| **Constructor Detail** |

### ColorEditor

public **ColorEditor**([DiagSettings](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html) fr)

constructor

**Parameters:**

fr - handle on dialog

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent e)

is called when user tries to edit cell

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

### getCellEditorValue

public java.lang.Object **getCellEditorValue**()

returns new value

**Specified by:**

getCellEditorValue in interface javax.swing.CellEditor

### getTableCellEditorComponent

public java.awt.Component **getTableCellEditorComponent**(javax.swing.JTable table,

java.lang.Object value,

boolean isSelected,

int row,

int column)

returns cell editor

**Specified by:**

getTableCellEditorComponent in interface javax.swing.table.TableCellEditor

## DiagrammerApp Class ColorEditor

java.lang.Object

extended by javax.swing.AbstractCellEditor

extended by **DiagrammerApp.ColorEditor**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.io.Serializable, java.util.EventListener, javax.swing.CellEditor, javax.swing.table.TableCellEditor

class **ColorEditor**

extends javax.swing.AbstractCellEditor

implements javax.swing.table.TableCellEditor, java.awt.event.ActionListener

default editor of cells

|  |
| --- |
| **Field Summary** |

|  |
| --- |
| **Fields inherited from class javax.swing.AbstractCellEditor** |
| changeEvent, listenerList |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**ColorEditor**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorEditor.html#ColorEditor(DiagrammerApp.DiagSettings))([DiagSettings](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorEditor.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent e)            is called when user tries to edit cell |
| java.lang.Object | [**getCellEditorValue**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorEditor.html#getCellEditorValue())()            returns new value |
| java.awt.Component | [**getTableCellEditorComponent**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\ColorEditor.html#getTableCellEditorComponent(javax.swing.JTable, java.lang.Object, boolean, int, int))(javax.swing.JTable table, java.lang.Object value, boolean isSelected, int row, int column)            returns cell editor |

|  |
| --- |
| **Methods inherited from class javax.swing.AbstractCellEditor** |
| addCellEditorListener, cancelCellEditing, fireEditingCanceled, fireEditingStopped, getCellEditorListeners, isCellEditable, removeCellEditorListener, shouldSelectCell, stopCellEditing |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Methods inherited from interface javax.swing.CellEditor** |
| addCellEditorListener, cancelCellEditing, isCellEditable, removeCellEditorListener, shouldSelectCell, stopCellEditing |

|  |
| --- |
| **Constructor Detail** |

### ColorEditor

public **ColorEditor**([DiagSettings](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html) fr)

constructor

**Parameters:**

fr - handle on dialog

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent e)

is called when user tries to edit cell

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

### getCellEditorValue

public java.lang.Object **getCellEditorValue**()

returns new value

**Specified by:**

getCellEditorValue in interface javax.swing.CellEditor

### getTableCellEditorComponent

public java.awt.Component **getTableCellEditorComponent**(javax.swing.JTable table,

java.lang.Object value,

boolean isSelected,

int row,

int column)

returns cell editor

**Specified by:**

getTableCellEditorComponent in interface javax.swing.table.TableCellEditor

## DiagrammerApp Class DiagrammerMain

java.lang.Object

extended by **DiagrammerApp.DiagrammerMain**

public class **DiagrammerMain**

extends java.lang.Object

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**DiagrammerMain**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagrammerMain.html#DiagrammerMain())() |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| static void | [**main**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagrammerMain.html#main(java.lang.String[]))(java.lang.String[] args) |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### DiagrammerMain

public **DiagrammerMain**()

|  |
| --- |
| **Method Detail** |

### main

public static void **main**(java.lang.String[] args)

## DiagrammerApp Class DiagSettings

java.lang.Object

extended by java.awt.Component

extended by java.awt.Container

extended by java.awt.Window

extended by java.awt.Dialog

extended by javax.swing.JDialog

extended by **DiagrammerApp.DiagSettings**

**All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

public class **DiagSettings**

extends javax.swing.JDialog

class dedicated to show and change settings of diagram

**See Also:**

[Serialized Form](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\serialized-form.html#DiagrammerApp.DiagSettings)

|  |
| --- |
| **Nested Class Summary** |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JDialog** |
| javax.swing.JDialog.AccessibleJDialog |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Dialog** |
| java.awt.Dialog.AccessibleAWTDialog, java.awt.Dialog.ModalExclusionType, java.awt.Dialog.ModalityType |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Window** |
| java.awt.Window.AccessibleAWTWindow |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Container** |
| java.awt.Container.AccessibleAWTContainer |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Component** |
| java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy |

|  |  |
| --- | --- |
| **Field Summary** | |
| boolean | [**change**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html#change)            shows if settings were changed |
| javax.swing.JRadioButton | [**diag\_col**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html#diag_col) |
| javax.swing.JRadioButton | [**diag\_row**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html#diag_row) |
| javax.swing.JTable | [**SetTab**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html#SetTab) |

|  |
| --- |
| **Fields inherited from class javax.swing.JDialog** |
| accessibleContext, rootPane, rootPaneCheckingEnabled |

|  |
| --- |
| **Fields inherited from class java.awt.Dialog** |
| DEFAULT\_MODALITY\_TYPE |

|  |
| --- |
| **Fields inherited from class java.awt.Component** |
| BOTTOM\_ALIGNMENT, CENTER\_ALIGNMENT, LEFT\_ALIGNMENT, RIGHT\_ALIGNMENT, TOP\_ALIGNMENT |

|  |
| --- |
| **Fields inherited from interface javax.swing.WindowConstants** |
| DISPOSE\_ON\_CLOSE, DO\_NOTHING\_ON\_CLOSE, EXIT\_ON\_CLOSE, HIDE\_ON\_CLOSE |

|  |
| --- |
| **Fields inherited from interface java.awt.image.ImageObserver** |
| ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**DiagSettings**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DiagSettings.html#DiagSettings())()            constructor |  |

|  |
| --- |
| **Method Summary** |

|  |
| --- |
| **Methods inherited from class java.awt.Dialog** |
| addNotify, getModalityType, getTitle, hide, isModal, isResizable, isUndecorated, setModal, setModalityType, setResizable, setTitle, setUndecorated, setVisible, show, toBack |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### change

public boolean **change**

shows if settings were changed

### diag\_col

public javax.swing.JRadioButton **diag\_col**

### diag\_row

public javax.swing.JRadioButton **diag\_row**

### SetTab

public javax.swing.JTable **SetTab**

|  |
| --- |
| **Constructor Detail** |

### DiagSettings

public **DiagSettings**()

constructor

## DiagrammerApp Class DSaveAsListener

java.lang.Object

extended by **DiagrammerApp.DSaveAsListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **DSaveAsListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Diagram->SaveAs

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**DSaveAsListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DSaveAsListener.html#DSaveAsListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DSaveAsListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### DSaveAsListener

**DSaveAsListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class DSaveListener

java.lang.Object

extended by **DiagrammerApp.DSaveListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **DSaveListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Diagram->Save

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**DSaveListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DSaveListener.html#DSaveListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DSaveListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### DSaveListener

**DSaveListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class DSetListener

java.lang.Object

extended by **DiagrammerApp.DSetListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **DSetListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Diagram->Settings

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**DSetListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DSetListener.html#DSetListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\DSetListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### DSetListener

**DSetListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class FCloseListener

java.lang.Object

extended by **DiagrammerApp.FCloseListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **FCloseListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item File->Close

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**FCloseListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FCloseListener.html#FCloseListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FCloseListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### FCloseListener

**FCloseListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class FExitListener

java.lang.Object

extended by **DiagrammerApp.FExitListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.awt.event.WindowListener, java.util.EventListener

class **FExitListener**

extends java.lang.Object

implements java.awt.event.ActionListener, java.awt.event.WindowListener

Listener of menu item File->Exit

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**FExitListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html#FExitListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |
| void | [**windowActivated**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html#windowActivated(java.awt.event.WindowEvent))(java.awt.event.WindowEvent arg0) |
| void | [**windowClosed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html#windowClosed(java.awt.event.WindowEvent))(java.awt.event.WindowEvent arg0) |
| void | [**windowClosing**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html#windowClosing(java.awt.event.WindowEvent))(java.awt.event.WindowEvent arg0) |
| void | [**windowDeactivated**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html#windowDeactivated(java.awt.event.WindowEvent))(java.awt.event.WindowEvent arg0) |
| void | [**windowDeiconified**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html#windowDeiconified(java.awt.event.WindowEvent))(java.awt.event.WindowEvent arg0) |
| void | [**windowIconified**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html#windowIconified(java.awt.event.WindowEvent))(java.awt.event.WindowEvent arg0) |
| void | [**windowOpened**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FExitListener.html#windowOpened(java.awt.event.WindowEvent))(java.awt.event.WindowEvent arg0) |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### FExitListener

**FExitListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

### windowActivated

public void **windowActivated**(java.awt.event.WindowEvent arg0)

**Specified by:**

windowActivated in interface java.awt.event.WindowListener

### windowClosed

public void **windowClosed**(java.awt.event.WindowEvent arg0)

**Specified by:**

windowClosed in interface java.awt.event.WindowListener

### windowClosing

public void **windowClosing**(java.awt.event.WindowEvent arg0)

**Specified by:**

windowClosing in interface java.awt.event.WindowListener

### windowDeactivated

public void **windowDeactivated**(java.awt.event.WindowEvent arg0)

**Specified by:**

windowDeactivated in interface java.awt.event.WindowListener

### windowDeiconified

public void **windowDeiconified**(java.awt.event.WindowEvent arg0)

**Specified by:**

windowDeiconified in interface java.awt.event.WindowListener

### windowIconified

public void **windowIconified**(java.awt.event.WindowEvent arg0)

**Specified by:**

windowIconified in interface java.awt.event.WindowListener

### windowOpened

public void **windowOpened**(java.awt.event.WindowEvent arg0)

**Specified by:**

windowOpened in interface java.awt.event.WindowListener

## DiagrammerApp Class FOpenListener

java.lang.Object

extended by **DiagrammerApp.FOpenListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **FOpenListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item File->Open

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**FOpenListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FOpenListener.html#FOpenListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FOpenListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### FOpenListener

**FOpenListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class FSaveAsListener

java.lang.Object

extended by **DiagrammerApp.FSaveAsListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **FSaveAsListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item File->SaveAs

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**FSaveAsListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FSaveAsListener.html#FSaveAsListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FSaveAsListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### FSaveAsListener

**FSaveAsListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class FSaveListener

java.lang.Object

extended by **DiagrammerApp.FSaveListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **FSaveListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item File->Save

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**FSaveListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FSaveListener.html#FSaveListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\FSaveListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### FSaveListener

**FSaveListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class HAboutListener

java.lang.Object

extended by **DiagrammerApp.HAboutListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **HAboutListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Help->About

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**HAboutListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\HAboutListener.html#HAboutListener())() |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\HAboutListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### HAboutListener

**HAboutListener**()

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class JMainFrame

java.lang.Object

extended by java.awt.Component

extended by java.awt.Container

extended by java.awt.Window

extended by java.awt.Frame

extended by javax.swing.JFrame

extended by **DiagrammerApp.JMainFrame**

**All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

public class **JMainFrame**

extends javax.swing.JFrame

class dedicated to show main window and interact with user

**See Also:**

[Serialized Form](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\serialized-form.html#DiagrammerApp.JMainFrame)

|  |
| --- |
| **Nested Class Summary** |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JFrame** |
| javax.swing.JFrame.AccessibleJFrame |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Frame** |
| java.awt.Frame.AccessibleAWTFrame |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Window** |
| java.awt.Window.AccessibleAWTWindow |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Container** |
| java.awt.Container.AccessibleAWTContainer |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Component** |
| java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy |

|  |  |
| --- | --- |
| **Field Summary** | |
| javax.swing.JPanel | [**DaigPane**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html#DaigPane) |
| [DiagramDrawer](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html) | [**diag\_drw**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html#diag_drw) |
| boolean | [**diag\_save**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html#diag_save) |
| javax.swing.JTable | [**Tab**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html#Tab) |
| boolean | [**tab\_save**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html#tab_save) |

|  |
| --- |
| **Fields inherited from class javax.swing.JFrame** |
| accessibleContext, EXIT\_ON\_CLOSE, rootPane, rootPaneCheckingEnabled |

|  |
| --- |
| **Fields inherited from class java.awt.Frame** |
| CROSSHAIR\_CURSOR, DEFAULT\_CURSOR, E\_RESIZE\_CURSOR, HAND\_CURSOR, ICONIFIED, MAXIMIZED\_BOTH, MAXIMIZED\_HORIZ, MAXIMIZED\_VERT, MOVE\_CURSOR, N\_RESIZE\_CURSOR, NE\_RESIZE\_CURSOR, NORMAL, NW\_RESIZE\_CURSOR, S\_RESIZE\_CURSOR, SE\_RESIZE\_CURSOR, SW\_RESIZE\_CURSOR, TEXT\_CURSOR, W\_RESIZE\_CURSOR, WAIT\_CURSOR |

|  |
| --- |
| **Fields inherited from class java.awt.Component** |
| BOTTOM\_ALIGNMENT, CENTER\_ALIGNMENT, LEFT\_ALIGNMENT, RIGHT\_ALIGNMENT, TOP\_ALIGNMENT |

|  |
| --- |
| **Fields inherited from interface javax.swing.WindowConstants** |
| DISPOSE\_ON\_CLOSE, DO\_NOTHING\_ON\_CLOSE, HIDE\_ON\_CLOSE |

|  |
| --- |
| **Fields inherited from interface java.awt.image.ImageObserver** |
| ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**JMainFrame**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html#JMainFrame())()            constructor of the class |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**updateTable**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html#updateTable())() |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait |

|  |
| --- |
| **Methods inherited from interface java.awt.MenuContainer** |
| getFont, postEvent |

|  |
| --- |
| **Field Detail** |

### DaigPane

public javax.swing.JPanel **DaigPane**

### diag\_drw

public [DiagramDrawer](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagCore\DiagramDrawer.html) **diag\_drw**

### diag\_save

public boolean **diag\_save**

### Tab

public javax.swing.JTable **Tab**

### tab\_save

public boolean **tab\_save**

|  |
| --- |
| **Constructor Detail** |

### JMainFrame

public **JMainFrame**()

constructor of the class

|  |
| --- |
| **Method Detail** |

### updateTable

public void **updateTable**()

## DiagrammerApp Class LoadTabModel

java.lang.Object

extended by javax.swing.table.AbstractTableModel

extended by **DiagrammerApp.LoadTabModel**

**All Implemented Interfaces:**

java.io.Serializable, javax.swing.table.TableModel

class **LoadTabModel**

extends javax.swing.table.AbstractTableModel

model of table for preview

|  |
| --- |
| **Field Summary** |

|  |
| --- |
| **Fields inherited from class javax.swing.table.AbstractTableModel** |
| listenerList |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**LoadTabModel**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\LoadTabModel.html#LoadTabModel())()            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| int | [**getColumnCount**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\LoadTabModel.html#getColumnCount())()            getter |
| java.lang.String | [**getColumnName**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\LoadTabModel.html#getColumnName(int))(int col)            getter |
| int | [**getRowCount**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\LoadTabModel.html#getRowCount())()            getter |
| java.lang.Object | [**getValueAt**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\LoadTabModel.html#getValueAt(int, int))(int row, int col)            getter |
| boolean | [**isCellEditable**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\LoadTabModel.html#isCellEditable(int, int))(int row, int col)            defines if cell is editable |
| void | [**setData**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\LoadTabModel.html#setData(java.util.ArrayList))(java.util.ArrayList str\_dat)            setter |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### LoadTabModel

**LoadTabModel**()

constructor

|  |
| --- |
| **Method Detail** |

### getColumnCount

public int **getColumnCount**()

getter

### getColumnName

public java.lang.String **getColumnName**(int col)

getter

**Specified by:**

getColumnName in interface javax.swing.table.TableModel

**Overrides:**

getColumnName in class javax.swing.table.AbstractTableModel

### getRowCount

public int **getRowCount**()

getter

### getValueAt

public java.lang.Object **getValueAt**(int row,

int col)

getter

### isCellEditable

public boolean **isCellEditable**(int row,

int col)

defines if cell is editable

**Specified by:**

isCellEditable in interface javax.swing.table.TableModel

**Overrides:**

isCellEditable in class javax.swing.table.AbstractTableModel

### setData

public void **setData**(java.util.ArrayList str\_dat)

setter

## DiagrammerApp Class MainTabColumnListener

java.lang.Object

extended by **DiagrammerApp.MainTabColumnListener**

**All Implemented Interfaces:**

java.util.EventListener, javax.swing.event.ListSelectionListener

class **MainTabColumnListener**

extends java.lang.Object

implements javax.swing.event.ListSelectionListener

Monitors changing of column selection

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**MainTabColumnListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabColumnListener.html#MainTabColumnListener(javax.swing.JTable))(javax.swing.JTable tab)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**valueChanged**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabColumnListener.html#valueChanged(javax.swing.event.ListSelectionEvent))(javax.swing.event.ListSelectionEvent e)            is called when column selection is changed |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### MainTabColumnListener

public **MainTabColumnListener**(javax.swing.JTable tab)

constructor

**Parameters:**

tab - table that is rendered

|  |
| --- |
| **Method Detail** |

### valueChanged

public void **valueChanged**(javax.swing.event.ListSelectionEvent e)

is called when column selection is changed

**Specified by:**

valueChanged in interface javax.swing.event.ListSelectionListener

## DiagrammerApp Class MainTabModel

java.lang.Object

extended by javax.swing.table.AbstractTableModel

extended by **DiagrammerApp.MainTabModel**

**All Implemented Interfaces:**

java.io.Serializable, javax.swing.table.TableModel

class **MainTabModel**

extends javax.swing.table.AbstractTableModel

|  |  |
| --- | --- |
| **Field Summary** | |
| int | [**selected**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#selected)            index of selected row or column |

|  |
| --- |
| **Fields inherited from class javax.swing.table.AbstractTableModel** |
| listenerList |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**MainTabModel**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#MainTabModel(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| int | [**getColumnCount**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#getColumnCount())()            returns number of columns |
| java.lang.String | [**getColumnName**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#getColumnName(int))(int col)            returns names of columns |
| int | [**getRowCount**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#getRowCount())()            returns number of rows |
| java.lang.Object | [**getValueAt**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#getValueAt(int, int))(int row, int col)            returns value that is saved in cell |
| boolean | [**isCellEditable**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#isCellEditable(int, int))(int row, int col)            returns true if cell can be edited |
| void | [**reinit**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#reinit())()            updates view of the table |
| void | [**setSelected**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#setSelected(int))(int sel)            sets selection of the table |
| void | [**setValueAt**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabModel.html#setValueAt(java.lang.Object, int, int))(java.lang.Object value, int row, int col)            writes value to the cell |

|  |
| --- |
| **Methods inherited from class javax.swing.table.AbstractTableModel** |
| addTableModelListener, findColumn, fireTableCellUpdated, fireTableChanged, fireTableDataChanged, fireTableRowsDeleted, fireTableRowsInserted, fireTableRowsUpdated, fireTableStructureChanged, getColumnClass, getListeners, getTableModelListeners, removeTableModelListener |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### selected

public int **selected**

index of selected row or column

|  |
| --- |
| **Constructor Detail** |

### MainTabModel

**MainTabModel**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### getColumnCount

public int **getColumnCount**()

returns number of columns

### getColumnName

public java.lang.String **getColumnName**(int col)

returns names of columns

**Specified by:**

getColumnName in interface javax.swing.table.TableModel

**Overrides:**

getColumnName in class javax.swing.table.AbstractTableModel

### getRowCount

public int **getRowCount**()

returns number of rows

### getValueAt

public java.lang.Object **getValueAt**(int row,

int col)

returns value that is saved in cell

**Parameters:**

row - row which contains cell

col - column which contains cell

### isCellEditable

public boolean **isCellEditable**(int row,

int col)

returns true if cell can be edited

**Specified by:**

isCellEditable in interface javax.swing.table.TableModel

**Overrides:**

isCellEditable in class javax.swing.table.AbstractTableModel

**Parameters:**

row - row which contains cell

col - column which contains cell

### reinit

public void **reinit**()

updates view of the table

### setSelected

public void **setSelected**(int sel)

sets selection of the table

**Parameters:**

sel - index of selected row/column

### setValueAt

public void **setValueAt**(java.lang.Object value,

int row,

int col)

writes value to the cell

**Specified by:**

setValueAt in interface javax.swing.table.TableModel

**Overrides:**

setValueAt in class javax.swing.table.AbstractTableModel

**Parameters:**

row - row which contains cell

col - column which contains cell

## DiagrammerApp Class MainTabRend

java.lang.Object

extended by java.awt.Component

extended by java.awt.Container

extended by javax.swing.JComponent

extended by javax.swing.JLabel

extended by **DiagrammerApp.MainTabRend**

**All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.SwingConstants, javax.swing.table.TableCellRenderer

class **MainTabRend**

extends javax.swing.JLabel

implements javax.swing.table.TableCellRenderer

|  |
| --- |
| **Nested Class Summary** |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JLabel** |
| javax.swing.JLabel.AccessibleJLabel |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JComponent** |
| javax.swing.JComponent.AccessibleJComponent |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Container** |
| java.awt.Container.AccessibleAWTContainer |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Component** |
| java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy |

|  |
| --- |
| **Field Summary** |

|  |
| --- |
| **Fields inherited from class javax.swing.JLabel** |
| labelFor |

|  |
| --- |
| **Fields inherited from class javax.swing.JComponent** |
| accessibleContext, listenerList, TOOL\_TIP\_TEXT\_KEY, ui, UNDEFINED\_CONDITION, WHEN\_ANCESTOR\_OF\_FOCUSED\_COMPONENT, WHEN\_FOCUSED, WHEN\_IN\_FOCUSED\_WINDOW |

|  |
| --- |
| **Fields inherited from class java.awt.Component** |
| BOTTOM\_ALIGNMENT, CENTER\_ALIGNMENT, LEFT\_ALIGNMENT, RIGHT\_ALIGNMENT, TOP\_ALIGNMENT |

|  |
| --- |
| **Fields inherited from interface javax.swing.SwingConstants** |
| BOTTOM, CENTER, EAST, HORIZONTAL, LEADING, LEFT, NEXT, NORTH, NORTH\_EAST, NORTH\_WEST, PREVIOUS, RIGHT, SOUTH, SOUTH\_EAST, SOUTH\_WEST, TOP, TRAILING, VERTICAL, WEST |

|  |
| --- |
| **Fields inherited from interface java.awt.image.ImageObserver** |
| ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**MainTabRend**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabRend.html#MainTabRend(javax.swing.JTable))(javax.swing.JTable tab)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| java.awt.Component | [**getTableCellRendererComponent**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabRend.html#getTableCellRendererComponent(javax.swing.JTable, java.lang.Object, boolean, boolean, int, int))(javax.swing.JTable tab, java.lang.Object val, boolean isSelected, boolean hasFocus, int row, int col) |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### MainTabRend

public **MainTabRend**(javax.swing.JTable tab)

constructor

**Parameters:**

tab - table that is rendered

|  |
| --- |
| **Method Detail** |

### getTableCellRendererComponent

public java.awt.Component **getTableCellRendererComponent**(javax.swing.JTable tab,

java.lang.Object val,

boolean isSelected,

boolean hasFocus,

int row,

int col)

**Specified by:**

getTableCellRendererComponent in interface javax.swing.table.TableCellRenderer

**Parameters:**

tab - table that is rendering

val - value that should be shown

isSelected -

hasFocus -

row - row that contains cell

col - column that contains cell

## DiagrammerApp Class MainTabRowListener

java.lang.Object

extended by **DiagrammerApp.MainTabRowListener**

**All Implemented Interfaces:**

java.util.EventListener, javax.swing.event.ListSelectionListener

class **MainTabRowListener**

extends java.lang.Object

implements javax.swing.event.ListSelectionListener

Monitors changing of rows selection

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**MainTabRowListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabRowListener.html#MainTabRowListener(javax.swing.JTable))(javax.swing.JTable tab)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**valueChanged**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\MainTabRowListener.html#valueChanged(javax.swing.event.ListSelectionEvent))(javax.swing.event.ListSelectionEvent e)            is called when row selection is changed |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### MainTabRowListener

public **MainTabRowListener**(javax.swing.JTable tab)

constructor

**Parameters:**

tab - table that is rendered

|  |
| --- |
| **Method Detail** |

### valueChanged

public void **valueChanged**(javax.swing.event.ListSelectionEvent e)

is called when row selection is changed

**Specified by:**

valueChanged in interface javax.swing.event.ListSelectionListener

## DiagrammerApp Class Saver

java.lang.Object

extended by java.awt.Component

extended by java.awt.Container

extended by java.awt.Window

extended by java.awt.Dialog

extended by javax.swing.JDialog

extended by **DiagrammerApp.Saver**

**All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

public class **Saver**

extends javax.swing.JDialog

dialog that provides possibility to save table or diagram

**See Also:**

[Serialized Form](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\serialized-form.html#DiagrammerApp.Saver)

|  |
| --- |
| **Nested Class Summary** |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JDialog** |
| javax.swing.JDialog.AccessibleJDialog |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Dialog** |
| java.awt.Dialog.AccessibleAWTDialog, java.awt.Dialog.ModalExclusionType, java.awt.Dialog.ModalityType |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Window** |
| java.awt.Window.AccessibleAWTWindow |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Container** |
| java.awt.Container.AccessibleAWTContainer |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Component** |
| java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy |

|  |  |
| --- | --- |
| **Field Summary** | |
| [JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) | [**mf**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html#mf) |
| javax.swing.JTextField | [**Path**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html#Path) |
| boolean | [**saved**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html#saved) |
| boolean | [**sv\_tab**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html#sv_tab) |
| javax.swing.JRadioButton | [**tp1\_ch**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html#tp1_ch) |
| javax.swing.JRadioButton | [**tp2\_ch**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html#tp2_ch) |

|  |
| --- |
| **Fields inherited from class javax.swing.JDialog** |
| accessibleContext, rootPane, rootPaneCheckingEnabled |

|  |
| --- |
| **Fields inherited from class java.awt.Dialog** |
| DEFAULT\_MODALITY\_TYPE |

|  |
| --- |
| **Fields inherited from class java.awt.Component** |
| BOTTOM\_ALIGNMENT, CENTER\_ALIGNMENT, LEFT\_ALIGNMENT, RIGHT\_ALIGNMENT, TOP\_ALIGNMENT |

|  |
| --- |
| **Fields inherited from interface javax.swing.WindowConstants** |
| DISPOSE\_ON\_CLOSE, DO\_NOTHING\_ON\_CLOSE, EXIT\_ON\_CLOSE, HIDE\_ON\_CLOSE |

|  |
| --- |
| **Fields inherited from interface java.awt.image.ImageObserver** |
| ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**Saver**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html#Saver(DiagrammerApp.JMainFrame, boolean))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) mf, boolean save\_table)            constructor |  |

|  |
| --- |
| **Method Summary** |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### mf

public [JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) **mf**

### Path

public javax.swing.JTextField **Path**

### saved

public boolean **saved**

### sv\_tab

public boolean **sv\_tab**

### tp1\_ch

public javax.swing.JRadioButton **tp1\_ch**

### tp2\_ch

public javax.swing.JRadioButton **tp2\_ch**

|  |
| --- |
| **Constructor Detail** |

### Saver

public **Saver**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) mf,

boolean save\_table)

constructor

**Parameters:**

save\_table - equals true while saving table, otherwise it is false

## DiagrammerApp Class SaverSave

java.lang.Object

extended by **DiagrammerApp.SaverSave**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **SaverSave**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of button Save

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**SaverSave**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SaverSave.html#SaverSave(DiagrammerApp.Saver))([Saver](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SaverSave.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when button is pushed |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### SaverSave

**SaverSave**([Saver](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\Saver.html) fr)

constructor

**Parameters:**

fr - dialog

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when button is pushed

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class SetTabModel

java.lang.Object

extended by javax.swing.table.AbstractTableModel

extended by javax.swing.table.DefaultTableModel

extended by **DiagrammerApp.SetTabModel**

**All Implemented Interfaces:**

java.io.Serializable, javax.swing.table.TableModel

class **SetTabModel**

extends javax.swing.table.DefaultTableModel

Model of table used to show settings

|  |  |
| --- | --- |
| **Field Summary** | |
| java.util.ArrayList | [**row\_col**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#row_col) |

|  |
| --- |
| **Fields inherited from class javax.swing.table.DefaultTableModel** |
| columnIdentifiers, dataVector |

|  |
| --- |
| **Fields inherited from class javax.swing.table.AbstractTableModel** |
| listenerList |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**SetTabModel**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#SetTabModel())()            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| java.lang.Class | [**getColumnClass**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#getColumnClass(int))(int c)            returns class of column |
| int | [**getColumnCount**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#getColumnCount())()            returns number of columns |
| java.lang.String | [**getColumnName**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#getColumnName(int))(int col)            returns column name |
| int | [**getRowCount**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#getRowCount())()            returns number of rows |
| java.lang.Object | [**getValueAt**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#getValueAt(int, int))(int row, int col)            returns value in cell |
| boolean | [**isCellEditable**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#isCellEditable(int, int))(int row, int col)            returns if cell is editable |
| void | [**setRowCount**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#setRowCount(int))(int rows)            setter for row count |
| void | [**setValueAt**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\SetTabModel.html#setValueAt(java.lang.Object, int, int))(java.lang.Object value, int row, int col)            setts value in cell |

|  |
| --- |
| **Methods inherited from class javax.swing.table.DefaultTableModel** |
| addColumn, addColumn, addColumn, addRow, addRow, convertToVector, convertToVector, getDataVector, insertRow, insertRow, moveRow, newDataAvailable, newRowsAdded, removeRow, rowsRemoved, setColumnCount, setColumnIdentifiers, setColumnIdentifiers, setDataVector, setDataVector, setNumRows |

|  |
| --- |
| **Methods inherited from class javax.swing.table.AbstractTableModel** |
| addTableModelListener, findColumn, fireTableCellUpdated, fireTableChanged, fireTableDataChanged, fireTableRowsDeleted, fireTableRowsInserted, fireTableRowsUpdated, fireTableStructureChanged, getListeners, getTableModelListeners, removeTableModelListener |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### row\_col

public java.util.ArrayList **row\_col**

|  |
| --- |
| **Constructor Detail** |

### SetTabModel

public **SetTabModel**()

constructor

|  |
| --- |
| **Method Detail** |

### getColumnClass

public java.lang.Class **getColumnClass**(int c)

returns class of column

**Specified by:**

getColumnClass in interface javax.swing.table.TableModel

**Overrides:**

getColumnClass in class javax.swing.table.AbstractTableModel

### getColumnCount

public int **getColumnCount**()

returns number of columns

**Specified by:**

getColumnCount in interface javax.swing.table.TableModel

**Overrides:**

getColumnCount in class javax.swing.table.DefaultTableModel

### getColumnName

public java.lang.String **getColumnName**(int col)

returns column name

**Specified by:**

getColumnName in interface javax.swing.table.TableModel

**Overrides:**

getColumnName in class javax.swing.table.DefaultTableModel

### getRowCount

public int **getRowCount**()

returns number of rows

**Specified by:**

getRowCount in interface javax.swing.table.TableModel

**Overrides:**

getRowCount in class javax.swing.table.DefaultTableModel

### getValueAt

public java.lang.Object **getValueAt**(int row,

int col)

returns value in cell

**Specified by:**

getValueAt in interface javax.swing.table.TableModel

**Overrides:**

getValueAt in class javax.swing.table.DefaultTableModel

### isCellEditable

public boolean **isCellEditable**(int row,

int col)

returns if cell is editable

**Specified by:**

isCellEditable in interface javax.swing.table.TableModel

**Overrides:**

isCellEditable in class javax.swing.table.DefaultTableModel

### setRowCount

public void **setRowCount**(int rows)

setter for row count

**Overrides:**

setRowCount in class javax.swing.table.DefaultTableModel

### setValueAt

public void **setValueAt**(java.lang.Object value,

int row,

int col)

setts value in cell

**Specified by:**

setValueAt in interface javax.swing.table.TableModel

**Overrides:**

setValueAt in class javax.swing.table.DefaultTableModel

## DiagrammerApp Class TableCreator

java.lang.Object

extended by java.awt.Component

extended by java.awt.Container

extended by java.awt.Window

extended by java.awt.Dialog

extended by javax.swing.JDialog

extended by **DiagrammerApp.TableCreator**

**All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

public class **TableCreator**

extends javax.swing.JDialog

dialog than helps to create table

**See Also:**

[Serialized Form](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\serialized-form.html#DiagrammerApp.TableCreator)

|  |
| --- |
| **Nested Class Summary** |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JDialog** |
| javax.swing.JDialog.AccessibleJDialog |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Dialog** |
| java.awt.Dialog.AccessibleAWTDialog, java.awt.Dialog.ModalExclusionType, java.awt.Dialog.ModalityType |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Window** |
| java.awt.Window.AccessibleAWTWindow |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Container** |
| java.awt.Container.AccessibleAWTContainer |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Component** |
| java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy |

|  |  |
| --- | --- |
| **Field Summary** | |
| boolean | [**changed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableCreator.html#changed) |
| javax.swing.JCheckBox | [**col\_head**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableCreator.html#col_head) |
| javax.swing.JTextField | [**col\_num**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableCreator.html#col_num) |
| javax.swing.JCheckBox | [**row\_head**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableCreator.html#row_head) |
| javax.swing.JTextField | [**row\_num**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableCreator.html#row_num) |

|  |
| --- |
| **Fields inherited from class javax.swing.JDialog** |
| accessibleContext, rootPane, rootPaneCheckingEnabled |

|  |
| --- |
| **Fields inherited from class java.awt.Dialog** |
| DEFAULT\_MODALITY\_TYPE |

|  |
| --- |
| **Fields inherited from class java.awt.Component** |
| BOTTOM\_ALIGNMENT, CENTER\_ALIGNMENT, LEFT\_ALIGNMENT, RIGHT\_ALIGNMENT, TOP\_ALIGNMENT |

|  |
| --- |
| **Fields inherited from interface javax.swing.WindowConstants** |
| DISPOSE\_ON\_CLOSE, DO\_NOTHING\_ON\_CLOSE, EXIT\_ON\_CLOSE, HIDE\_ON\_CLOSE |

|  |
| --- |
| **Fields inherited from interface java.awt.image.ImageObserver** |
| ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TableCreator**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableCreator.html#TableCreator())()            constructor |  |

|  |
| --- |
| **Method Summary** |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### changed

public boolean **changed**

### col\_head

public javax.swing.JCheckBox **col\_head**

### col\_num

public javax.swing.JTextField **col\_num**

### row\_head

public javax.swing.JCheckBox **row\_head**

### row\_num

public javax.swing.JTextField **row\_num**

|  |
| --- |
| **Constructor Detail** |

### TableCreator

public **TableCreator**()

constructor

## DiagrammerApp Class TableLoader

java.lang.Object

extended by java.awt.Component

extended by java.awt.Container

extended by java.awt.Window

extended by java.awt.Dialog

extended by javax.swing.JDialog

extended by **DiagrammerApp.TableLoader**

**All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

public class **TableLoader**

extends javax.swing.JDialog

class that loads table

**See Also:**

[Serialized Form](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\serialized-form.html#DiagrammerApp.TableLoader)

|  |
| --- |
| **Nested Class Summary** |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JDialog** |
| javax.swing.JDialog.AccessibleJDialog |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Dialog** |
| java.awt.Dialog.AccessibleAWTDialog, java.awt.Dialog.ModalExclusionType, java.awt.Dialog.ModalityType |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Window** |
| java.awt.Window.AccessibleAWTWindow |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Container** |
| java.awt.Container.AccessibleAWTContainer |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Component** |
| java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy |

|  |  |
| --- | --- |
| **Field Summary** | |
| javax.swing.JComboBox | [**Delim**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html#Delim) |
| javax.swing.JCheckBox | [**head**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html#head) |
| javax.swing.JTextField | [**PathEdit**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html#PathEdit) |
| javax.swing.JTable | [**ResTab**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html#ResTab) |
| javax.swing.JCheckBox | [**row\_name**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html#row_name) |

|  |
| --- |
| **Fields inherited from class javax.swing.JDialog** |
| accessibleContext, rootPane, rootPaneCheckingEnabled |

|  |
| --- |
| **Fields inherited from class java.awt.Dialog** |
| DEFAULT\_MODALITY\_TYPE |

|  |
| --- |
| **Fields inherited from class java.awt.Component** |
| BOTTOM\_ALIGNMENT, CENTER\_ALIGNMENT, LEFT\_ALIGNMENT, RIGHT\_ALIGNMENT, TOP\_ALIGNMENT |

|  |
| --- |
| **Fields inherited from interface javax.swing.WindowConstants** |
| DISPOSE\_ON\_CLOSE, DO\_NOTHING\_ON\_CLOSE, EXIT\_ON\_CLOSE, HIDE\_ON\_CLOSE |

|  |
| --- |
| **Fields inherited from interface java.awt.image.ImageObserver** |
| ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TableLoader**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html#TableLoader())()            constructor |  |

|  |
| --- |
| **Method Summary** |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### Delim

public javax.swing.JComboBox **Delim**

### head

public javax.swing.JCheckBox **head**

### PathEdit

public javax.swing.JTextField **PathEdit**

### ResTab

public javax.swing.JTable **ResTab**

### row\_name

public javax.swing.JCheckBox **row\_name**

|  |
| --- |
| **Constructor Detail** |

### TableLoader

public **TableLoader**()

constructor

## DiagrammerApp Class TabLoadOk

java.lang.Object

extended by **DiagrammerApp.TabLoadOk**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **TabLoadOk**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener for button Load

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TabLoadOk**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabLoadOk.html#TabLoadOk(DiagrammerApp.TableLoader))([TableLoader](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabLoadOk.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when the button is pushed |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### TabLoadOk

public **TabLoadOk**([TableLoader](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html) fr)

constructor

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when the button is pushed

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class TabLoadView

java.lang.Object

extended by **DiagrammerApp.TabLoadView**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **TabLoadView**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener for button View

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TabLoadView**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabLoadView.html#TabLoadView(DiagrammerApp.TableLoader))([TableLoader](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabLoadView.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when button is pressed |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### TabLoadView

public **TabLoadView**([TableLoader](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TableLoader.html) fr)

constructor

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when button is pressed

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class TabSettings

java.lang.Object

extended by java.awt.Component

extended by java.awt.Container

extended by java.awt.Window

extended by java.awt.Dialog

extended by javax.swing.JDialog

extended by **DiagrammerApp.TabSettings**

**All Implemented Interfaces:**

java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.WindowConstants

public class **TabSettings**

extends javax.swing.JDialog

dialog that is used to change settings of a table

**See Also:**

[Serialized Form](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\serialized-form.html#DiagrammerApp.TabSettings)

|  |
| --- |
| **Nested Class Summary** |

|  |
| --- |
| **Nested classes/interfaces inherited from class javax.swing.JDialog** |
| javax.swing.JDialog.AccessibleJDialog |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Dialog** |
| java.awt.Dialog.AccessibleAWTDialog, java.awt.Dialog.ModalExclusionType, java.awt.Dialog.ModalityType |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Window** |
| java.awt.Window.AccessibleAWTWindow |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Container** |
| java.awt.Container.AccessibleAWTContainer |

|  |
| --- |
| **Nested classes/interfaces inherited from class java.awt.Component** |
| java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy |

|  |  |
| --- | --- |
| **Field Summary** | |
| boolean | [**change**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSettings.html#change) |
| javax.swing.JButton | [**col\_col**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSettings.html#col_col) |
| javax.swing.JCheckBox | [**col\_num**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSettings.html#col_num) |
| javax.swing.JTextField | [**cols**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSettings.html#cols) |
| javax.swing.JButton | [**row\_col**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSettings.html#row_col) |
| javax.swing.JCheckBox | [**row\_num**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSettings.html#row_num) |
| javax.swing.JTextField | [**rows**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSettings.html#rows) |

|  |
| --- |
| **Fields inherited from class javax.swing.JDialog** |
| accessibleContext, rootPane, rootPaneCheckingEnabled |

|  |
| --- |
| **Fields inherited from class java.awt.Dialog** |
| DEFAULT\_MODALITY\_TYPE |

|  |
| --- |
| **Fields inherited from class java.awt.Component** |
| BOTTOM\_ALIGNMENT, CENTER\_ALIGNMENT, LEFT\_ALIGNMENT, RIGHT\_ALIGNMENT, TOP\_ALIGNMENT |

|  |
| --- |
| **Fields inherited from interface javax.swing.WindowConstants** |
| DISPOSE\_ON\_CLOSE, DO\_NOTHING\_ON\_CLOSE, EXIT\_ON\_CLOSE, HIDE\_ON\_CLOSE |

|  |
| --- |
| **Fields inherited from interface java.awt.image.ImageObserver** |
| ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TabSettings**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TabSettings.html#TabSettings())()            constructor |  |

|  |
| --- |
| **Method Summary** |

|  |
| --- |
| **Methods inherited from class javax.swing.JDialog** |
| addImpl, createRootPane, dialogInit, getAccessibleContext, getContentPane, getDefaultCloseOperation, getGlassPane, getGraphics, getJMenuBar, getLayeredPane, getRootPane, getTransferHandler, isDefaultLookAndFeelDecorated, isRootPaneCheckingEnabled, paramString, processWindowEvent, remove, repaint, setContentPane, setDefaultCloseOperation, setDefaultLookAndFeelDecorated, setGlassPane, setJMenuBar, setLayeredPane, setLayout, setRootPane, setRootPaneCheckingEnabled, setTransferHandler, update |

|  |
| --- |
| **Methods inherited from class java.awt.Dialog** |
| addNotify, getModalityType, getTitle, hide, isModal, isResizable, isUndecorated, setModal, setModalityType, setResizable, setTitle, setUndecorated, setVisible, show, toBack |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait |

|  |
| --- |
| **Field Detail** |

### change

public boolean **change**

### col\_col

public javax.swing.JButton **col\_col**

### col\_num

public javax.swing.JCheckBox **col\_num**

### cols

public javax.swing.JTextField **cols**

### row\_col

public javax.swing.JButton **row\_col**

### row\_num

public javax.swing.JCheckBox **row\_num**

### rows

public javax.swing.JTextField **rows**

|  |
| --- |
| **Constructor Detail** |

### TabSettings

public **TabSettings**()

constructor

## DiagrammerApp Class TAddColListener

java.lang.Object

extended by **DiagrammerApp.TAddColListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **TAddColListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Table->Add column

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TAddColListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TAddColListener.html#TAddColListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TAddColListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### TAddColListener

**TAddColListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class TAddColListener

java.lang.Object

extended by **DiagrammerApp.TAddColListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **TAddColListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Table->Add column

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TAddColListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TAddColListener.html#TAddColListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TAddColListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### TAddColListener

**TAddColListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class TAddColListener

java.lang.Object

extended by **DiagrammerApp.TAddColListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **TAddColListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Table->Add column

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TAddColListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TAddColListener.html#TAddColListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TAddColListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### TAddColListener

**TAddColListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class TDelColListener

java.lang.Object

extended by **DiagrammerApp.TDelColListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **TDelColListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Table->Delete column

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TDelColListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TDelColListener.html#TDelColListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TDelColListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### TDelColListener

**TDelColListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class TDelColListener

java.lang.Object

extended by **DiagrammerApp.TDelColListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **TDelColListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Table->Delete column

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TDelColListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TDelColListener.html#TDelColListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TDelColListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### TDelColListener

**TDelColListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

## DiagrammerApp Class TSetListener

java.lang.Object

extended by **DiagrammerApp.TSetListener**

**All Implemented Interfaces:**

java.awt.event.ActionListener, java.util.EventListener

class **TSetListener**

extends java.lang.Object

implements java.awt.event.ActionListener

Listener of menu item Table->Settings

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [**TSetListener**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TSetListener.html#TSetListener(DiagrammerApp.JMainFrame))([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)            constructor |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [**actionPerformed**](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\TSetListener.html#actionPerformed(java.awt.event.ActionEvent))(java.awt.event.ActionEvent arg0)            is called when user choose menu item |

|  |
| --- |
| **Methods inherited from class java.lang.Object** |
| clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait |

|  |
| --- |
| **Constructor Detail** |

### TSetListener

**TSetListener**([JMainFrame](file:///C:\Documents%20and%20Settings\MaxCm\Рабочий%20стол\programming\work\Diagrammer\doc\DiagrammerApp\JMainFrame.html) fr)

constructor

**Parameters:**

fr - handle on main frame

|  |
| --- |
| **Method Detail** |

### actionPerformed

public void **actionPerformed**(java.awt.event.ActionEvent arg0)

is called when user choose menu item

**Specified by:**

actionPerformed in interface java.awt.event.ActionListener

* 1. **Документація для користувача**

**3.2.1. Системні вимоги**

- Java Runtime Environment Standart Edition 5 или выше.

**3.2.2. Запуск програми**

- Через командну строку: "java -jar <шлях до jar файлу>".

- Подвійний клік на jar файлі.

**3.2.2 Завантаження таблиці**

- File->Open

- Встановити прапорці за необхідністю, додати роздільники.

- Натиснути на кнопку Load

**3.2.3 Збереження таблиці**

- File>Save

- вибрати шлях до файлу та через крапку додати розширення

**3.2.4 Збереження графіка**

- Diagram->Save або

- вибрати шлях до файлу та через крапку додати розширення

**3.2.5 Закриття таблиці**

-File->Close

**3.2.6 Операції з даними**

- Для зміни кольору Diagram->Settings

- Для зміни даних необхідно натиснути на потрібну ячейку і встановити потрібне значення, в разі невідповідності формату введеного значення, виведеться повідомлення.

**Висновок**

В результаті виконання роботи, було отримано програмний продукт, який відповідає усім технічним вимогам встановленими в технічному завданні. Розроблена програма призначена для роботи з таблицями, а також для зміни та візуалізації даних у вигляді кругової діаграми. Програма дозволяє завантажувати дані з csv файлу, редагувати їх, а також зберегти змінену таблицю і побудовану діаграму. Для зручності є можливість змінювати колір відображених даних. Програмний продукт може бути застосований у будь-якій сфері де необхідно оперувати з табличними даними.

**Литература**

1. Г. Шилдт. Полный справочник по Java SE6: 7-е издание. Москва: «Вильямс», 2007. - 1035 с.( ISBN 0-07-226385-7/ ISBN 978-5-8459-1168-1).

2. Г.Шилдт. Swing руководство для начинающих: .Москва: «Вильямс», 2007. - 704 с. ( ISBN 0-07-226314-8/ISBN 978-5-8459-1162-9).

3. Б. Эккель. Философия Java. Библиотека программиста: 2-е издание. «Питер», 2009. - 637 c. (ISBN 5-88782-105-1).

4. Berthold Daum. Professional Eclipse 3 for Java™ Developers. «Wrox», 2004. - 575 c. (ISBN 0-470-02005-9).

5. Kathy Sierra/Bert Bates. SCJP Sun®Certifed Programmer for Java™ 6 Study Guide. «McGraw-Hill», 2008. - 890 c. (ISBN 0-07-159107-9).

6. И.Н. Блинов. Java .ПРОМЫШЛЕННОЕ ПРОГРАММИРОВАНИЕ. Минск, «УниверсалПресс», 2007. -704 c. (ISBN 978-985-6699-63-7).

7. Гамма Э., Хелм Р., Джонсон Р.Приемы объектно-ориентированного проектирования. «Питер», 2008. -336 c. (ISBN 5-469-01136-4 ).

8. Д. Блох. Java. Эффективное программирование. Москва: «Лори», 2002. – 224 с.

9. И.Н. Блинов, В.С. Романчик. Java 2. Практическое руководство. Мн.: «УниверсалПресс», 2005. – 400 с.

10. Д. Блох. Java. Эффективное программирование. М.: «Лори», 2002. – 224 с.

**Додаток А**

Код програми

package DiagrammerApp;

import java.io.File;

import java.io.IOException;

import javax.swing.JOptionPane;

import DiagCore.CurrentTable;

import DiagCore.SettingsContainer;

import Interface.\*;

public class DiagrammerMain

{

public static void main(String [] args)

{

File setf=new File("settings.dat");

if(!SettingsContainer.loadSettings(setf))

{

SettingsContainer.createSettings();

JOptionPane.showConfirmDialog(null, new String("Setting are default, because error while loading settings occured."),

"Warning", JOptionPane.OK\_OPTION, JOptionPane.INFORMATION\_MESSAGE);

}

JMainFrame mf=new JMainFrame();

mf.setVisible(true);

}

}

package DiagrammerApp;

import DiagCore.CSVProcessor;

import DiagCore.CurrentTable;

import DiagCore.DiagramDrawer;

import DiagCore.SettingsContainer;

import com.cloudgarden.layout.AnchorLayout;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Component;

import java.awt.FlowLayout;

import java.awt.Graphics2D;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.WindowEvent;

import java.awt.event.WindowListener;

import java.awt.image.BufferedImage;

import java.io.File;

import javax.swing.BorderFactory;

import javax.swing.GroupLayout;

import javax.swing.JButton;

import javax.swing.JComponent;

import javax.swing.JDialog;

import javax.swing.JLabel;

import javax.swing.JMenu;

import javax.swing.JMenuBar;

import javax.swing.JMenuItem;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.KeyStroke;

import javax.swing.LayoutStyle;

import javax.swing.ListSelectionModel;

import javax.swing.WindowConstants;

import javax.swing.border.LineBorder;

import javax.swing.event.ListSelectionEvent;

import javax.swing.event.ListSelectionListener;

import javax.swing.table.AbstractTableModel;

import javax.swing.table.DefaultTableModel;

import javax.swing.table.TableCellRenderer;

import javax.swing.table.TableModel;

import javax.swing.SwingUtilities;

/\*

check MainTableModel. Diagram should be changed after editing table+bugs while editing

change parsing of text data

\*/

/\*\*

\* class dedicated to show main window and interact with user

\*/

public class JMainFrame extends javax.swing.JFrame

{

private JMenuBar Menu;

private JMenuItem FOpen;

private JMenuItem TCreate;

private JMenuItem TDelCol;

private JMenuItem TDelRow;

public JTable Tab;

public JPanel DaigPane;

private JScrollPane TabScr;

private JMenuItem HAbout;

private JMenuItem DBuild;

private JMenuItem DSett;

private JMenuItem DSaveAs;

private JMenuItem DSave;

private JMenuItem FSaveAs;

private JMenuItem TSett;

private JMenuItem TAddCol;

private JMenuItem TAddRow;

private JMenuItem FExit;

private JMenuItem FClose;

private JMenuItem FSave;

private JMenu Help;

private JMenu Diagram;

private JMenu Table;

private JMenu File;

public boolean tab\_save, diag\_save;

public DiagramDrawer diag\_drw;

/\*\*

\*constructor of the class

\*/

public JMainFrame()

{

super();

initGUI();

}

/\*\*

\* create GUI

\*/

private void initGUI()

{

try

{

FlowLayout thisLayout = new FlowLayout();

getContentPane().setLayout(thisLayout);

this.setDefaultCloseOperation(WindowConstants.DO\_NOTHING\_ON\_CLOSE);

this.setPreferredSize(new java.awt.Dimension(900, 600));

this.setName("MainFrame");

this.setTitle("Diagrammer");

FExitListener FExitList=new FExitListener(this);

this.addWindowListener(FExitList);

{

TabScr = new JScrollPane();

getContentPane().add(TabScr);

TabScr.setPreferredSize(new java.awt.Dimension(414, 530));

{

TableModel TabModel = new MainTabModel(this);

Tab = new JTable();

TabScr.setViewportView(Tab);

Tab.setModel(TabModel);

Tab.setDefaultRenderer(Object.class, new MainTabRend(Tab));

Tab.setSelectionMode(ListSelectionModel.SINGLE\_SELECTION);

Tab.setCellSelectionEnabled(false);

if(SettingsContainer.getSettings().col\_for\_diag)

{

Tab.setRowSelectionAllowed(false);

Tab.setColumnSelectionAllowed(true);

}

else

{

Tab.setRowSelectionAllowed(true);

Tab.setColumnSelectionAllowed(false);

}

Tab.getSelectionModel().addListSelectionListener(new MainTabRowListener(Tab));

Tab.getColumnModel().getSelectionModel().addListSelectionListener(new MainTabColumnListener(Tab));

}

}

{

DaigPane = new JPanel();

BorderLayout DaigPaneLayout = new BorderLayout();

getContentPane().add(DaigPane);

DaigPane.setPreferredSize(new java.awt.Dimension(467, 529));

DaigPane.setBorder(new LineBorder(new java.awt.Color(0,0,0), 1, false));

DaigPane.setLayout(DaigPaneLayout);

diag\_drw=new DiagramDrawer();

DaigPane.add(diag\_drw);

diag\_drw.mf=this;

diag\_drw.setBounds(0, 0, DaigPane.getBounds().width, DaigPane.getBounds().height);

}

{

Menu = new JMenuBar();

setJMenuBar(Menu);

{

File = new JMenu();

Menu.add(File);

File.setText("File");

{

FOpen = new JMenuItem();

File.add(FOpen);

FOpen.setText("Open");

FOpen.addActionListener(new FOpenListener(this));

}

{

FSaveAs = new JMenuItem();

File.add(FSaveAs);

FSaveAs.setText("Save as");

FSaveAs.addActionListener(new FSaveAsListener(this));

}

{

FSave = new JMenuItem();

File.add(FSave);

FSave.setText("Save");

FSave.addActionListener(new FSaveListener(this));

}

{

FClose = new JMenuItem();

File.add(FClose);

FClose.setText("Close");

FClose.addActionListener(new FCloseListener(this));

}

{

FExit = new JMenuItem();

File.add(FExit);

FExit.setText("Exit");

FExit.addActionListener(FExitList);

}

}

{

Table = new JMenu();

Menu.add(Table);

Table.setText("Table");

{

TAddRow = new JMenuItem();

Table.add(TAddRow);

TAddRow.setText("Add row");

TAddRow.addActionListener(new TAddRowListener(this));

}

{

TAddCol = new JMenuItem();

Table.add(TAddCol);

TAddCol.setText("Add column");

TAddCol.addActionListener(new TAddColListener(this));

}

{

TDelRow = new JMenuItem();

Table.add(TDelRow);

TDelRow.setText("Delete row");

TDelRow.addActionListener(new TDelRowListener(this));

}

{

TDelCol = new JMenuItem();

Table.add(TDelCol);

TDelCol.setText("Delete column");

TDelCol.addActionListener(new TDelColListener(this));

}

{

TCreate = new JMenuItem();

Table.add(TCreate);

TCreate.setText("Create table");

TCreate.addActionListener(new TCreateListener(this));

TCreate.addActionListener(new TCreateListener(this));

}

{

TSett = new JMenuItem();

Table.add(TSett);

TSett.setText("Settings");

TSett.addActionListener(new TSetListener(this));

}

}

{

Diagram = new JMenu();

Menu.add(Diagram);

Diagram.setText("Diagram");

{

DBuild = new JMenuItem();

Diagram.add(DBuild);

DBuild.setText("Build");

DBuild.addActionListener(new DBuildListener(this));

}

{

DSave = new JMenuItem();

Diagram.add(DSave);

DSave.setText("Save");

DSave.addActionListener(new DSaveListener(this));

}

{

DSaveAs = new JMenuItem();

Diagram.add(DSaveAs);

DSaveAs.setText("Save as");

DSaveAs.addActionListener(new DSaveAsListener(this));

}

{

DSett = new JMenuItem();

Diagram.add(DSett);

DSett.setText("Settings");

DSett.addActionListener(new DSetListener(this));

}

}

{

Help = new JMenu();

Menu.add(Help);

Help.setText("Help");

{

HAbout = new JMenuItem();

Help.add(HAbout);

HAbout.setText("About");

HAbout.addActionListener(new HAboutListener());

}

}

}

pack();

this.setSize(900, 600);

} catch (Exception e) {

//add your error handling code here

e.printStackTrace();

}

}

public void updateTable(){((MainTabModel)Tab.getModel()).reinit();}

}

//-------------------------------------------------------

//Listeners for FILE

/\*\*

\* Listener of menu item File->Open

\*/

class FOpenListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

FOpenListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

TableLoader tl=new TableLoader();

tl.setVisible(true);

if(CurrentTable.getTable()!=null)

((MainTabModel)mf.Tab.getModel()).reinit();

}

}

/\*\*

\* Listener of menu item File->Save

\*/

class FSaveListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

FSaveListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable().file==null) return;

String [][] str\_dat=CurrentTable.getTable().getInTextFormat();

CSVProcessor csv=new CSVProcessor();

csv.SetDelimiter(SettingsContainer.getSettings().del\_val[SettingsContainer.getSettings().delim]);

csv.BuildTextData(str\_dat);

if(CurrentTable.getTable().text\_format)

mf.tab\_save=csv.SaveTextData(CurrentTable.getTable().file);

else

mf.tab\_save=csv.SaveBinData(CurrentTable.getTable().file);

}

}

/\*\*

\* Listener of menu item File->SaveAs

\*/

class FSaveAsListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

FSaveAsListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

Saver sv=new Saver(mf, true);

sv.setVisible(true);

mf.tab\_save=sv.saved;

}

}

/\*\*

\* Listener of menu item File->Close

\*/

class FCloseListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

FCloseListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

CurrentTable.getTable().deleteTable();

mf.updateTable();

mf.diag\_drw.Clear();

}

}

/\*\*

\* Listener of menu item File->Exit

\*/

class FExitListener implements ActionListener, WindowListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

FExitListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()!=null)

{

if(!mf.tab\_save)

{

if(JOptionPane.showConfirmDialog(null, new String("Table is not saved. Continue?"),

"Error", JOptionPane.OK\_CANCEL\_OPTION, JOptionPane.INFORMATION\_MESSAGE)

!=JOptionPane.OK\_OPTION) return;

}

if(!mf.diag\_save)

{

if(JOptionPane.showConfirmDialog(null, new String("Diagram is not saved. Continue?"),

"Error", JOptionPane.OK\_CANCEL\_OPTION, JOptionPane.INFORMATION\_MESSAGE)

!=JOptionPane.OK\_OPTION) return;

}

}

SettingsContainer.getSettings().saveSettings(new File("settings.dat"));

System.exit(0);

}

public void windowActivated(WindowEvent arg0){}

public void windowClosed(WindowEvent arg0){}

public void windowClosing(WindowEvent arg0){this.actionPerformed(null);}

public void windowDeactivated(WindowEvent arg0){}

public void windowDeiconified(WindowEvent arg0){}

public void windowIconified(WindowEvent arg0){}

public void windowOpened(WindowEvent arg0){}

}

//-------------------------------------------------------

//Listeners for Table

/\*\*

\* Listener of menu item Table->Add row

\*/

class TAddRowListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

TAddRowListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

CurrentTable ct=CurrentTable.getTable();

SettingsContainer sc=SettingsContainer.getSettings();

ct.addRow(); mf.updateTable();

if(sc.col\_for\_diag) ct.addDiagColor(sc.getDefaultDiagramColor(ct.diag\_col));

if(mf.diag\_drw.builded)

{

mf.diag\_drw.SetColor(ct.diag\_col);

mf.diag\_drw.SetData(ct.getDiagVector());

}

mf.tab\_save=false;

}

}

/\*\*

\* Listener of menu item Table->Add column

\*/

class TAddColListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

TAddColListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

CurrentTable ct=CurrentTable.getTable();

SettingsContainer sc=SettingsContainer.getSettings();

ct.addColumn(); mf.updateTable();

if(!sc.col\_for\_diag) ct.addDiagColor(sc.getDefaultDiagramColor(ct.diag\_col));

if(mf.diag\_drw.builded)

{

mf.diag\_drw.SetColor(ct.diag\_col);

mf.diag\_drw.SetData(ct.getDiagVector());

}

mf.tab\_save=false;

}

}

/\*\*

\* Listener of menu item Table->delete row

\*/

class TDelRowListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

TDelRowListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

if(CurrentTable.getTable().rows==0) return;

MainTabModel md=(MainTabModel)mf.Tab.getModel();

if(md.selected==-1) return;

CurrentTable.getTable().removeRow(md.selected);

if(SettingsContainer.getSettings().col\_for\_diag)

CurrentTable.getTable().removeDiagColor(md.selected);

else

{

if(CurrentTable.getTable().diag\_ind==md.selected)

CurrentTable.getTable().diag\_ind=-1;

}

md.setSelected(-1);

md.reinit();

if(mf.diag\_drw.builded)

{

mf.diag\_drw.SetColor(CurrentTable.getTable().diag\_col);

mf.diag\_drw.SetData(CurrentTable.getTable().getDiagVector());

}

mf.tab\_save=false;

}

}

/\*\*

\* Listener of menu item Table->Delete column

\*/

class TDelColListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

TDelColListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

if(CurrentTable.getTable().cols==0) return;

MainTabModel md=(MainTabModel)mf.Tab.getModel();

if(md.selected==-1) return;

CurrentTable.getTable().removeColumn(md.selected);

if(!SettingsContainer.getSettings().col\_for\_diag)

CurrentTable.getTable().removeDiagColor(md.selected);

else

{

if(CurrentTable.getTable().diag\_ind==md.selected)

CurrentTable.getTable().diag\_ind=-1;

}

md.setSelected(-1);

md.reinit();

if(mf.diag\_drw.builded)

{

mf.diag\_drw.SetColor(CurrentTable.getTable().diag\_col);

mf.diag\_drw.SetData(CurrentTable.getTable().getDiagVector());

}

mf.tab\_save=false;

}

}

/\*\*

\* Listener of menu item Table->Create

\*/

class TCreateListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

TCreateListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()!=null && (!mf.tab\_save || !mf.tab\_save))

{

if(JOptionPane.showConfirmDialog(null, new String("Data is not saved. Continue?"),

"Error", JOptionPane.OK\_CANCEL\_OPTION, JOptionPane.INFORMATION\_MESSAGE)

!=JOptionPane.OK\_OPTION) return;

}

TableCreator tc=new TableCreator();

tc.setVisible(true);

if(tc.changed)

{

mf.updateTable(); mf.tab\_save=false;

if(mf.diag\_drw.have\_data) mf.diag\_drw.Clear();

}

}

}

/\*\*

\* Listener of menu item Table->Settings

\*/

class TSetListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

TSetListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

TabSettings ts=new TabSettings(); ts.setVisible(true);

if(ts.change)

{

((MainTabModel)mf.Tab.getModel()).reinit();

if(mf.diag\_drw.have\_data) mf.diag\_drw.Clear();

mf.tab\_save=false;

}

}

}

//-----------------------------------------------------------

//Listeners for Diagram

/\*\*

\* Listener of menu item Diagram->Build

\*/

class DBuildListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

DBuildListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

if(((MainTabModel)mf.Tab.getModel()).selected==-1) return;

CurrentTable.getTable().diag\_ind=((MainTabModel)mf.Tab.getModel()).selected;

mf.diag\_drw.SetData(CurrentTable.getTable().getDiagVector());

mf.diag\_drw.SetColor(CurrentTable.getTable().diag\_col);

mf.diag\_drw.repaint();

((MainTabModel)mf.Tab.getModel()).reinit();

mf.tab\_save=false; mf.diag\_save=false;

}

}

/\*\*

\* Listener of menu item Diagram->Save

\*/

class DSaveListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

DSaveListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

if(CurrentTable.getTable().gfile==null) return;

if(!mf.diag\_drw.have\_data) return;

mf.diag\_save=true;

File out=new File(CurrentTable.getTable().gfile);

if(!out.isFile() || !out.exists())

{

try{out.createNewFile();}

catch(Exception e2)

{

JOptionPane.showConfirmDialog(null, new String("Can not create file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

mf.diag\_save=false; return;

}

}

BufferedImage bi=new BufferedImage(mf.diag\_drw.getBounds().width, mf.diag\_drw.getBounds().height, BufferedImage.TYPE\_INT\_RGB);

Graphics2D g2=bi.createGraphics();

g2.setClip(0, 0, mf.diag\_drw.getBounds().width, mf.diag\_drw.getBounds().height);

mf.diag\_drw.draw(g2);

String tp=new String();

int i;

for(i=CurrentTable.getTable().gfile.length()-1; i>=0; i++)

if(CurrentTable.getTable().gfile.charAt(i)=='.') break;

for(; i<CurrentTable.getTable().gfile.length(); i++)

tp+=CurrentTable.getTable().gfile.charAt(i);

try

{

javax.imageio.ImageIO.write(bi, tp, out);

}

catch(Exception e)

{

JOptionPane.showConfirmDialog(null, new String("Error while writing to file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

mf.diag\_save=false;

}

}

}

/\*\*

\* Listener of menu item Diagram->SaveAs

\*/

class DSaveAsListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

DSaveAsListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

if(CurrentTable.getTable()==null) return;

if(!mf.diag\_drw.have\_data) return;

Saver sv=new Saver(mf, false); sv.setVisible(true);

mf.diag\_save=sv.saved;

}

}

/\*\*

\* Listener of menu item Diagram->Settings

\*/

class DSetListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

DSetListener(JMainFrame fr){mf=fr;}

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

boolean pcfd=SettingsContainer.getSettings().col\_for\_diag;

DiagSettings ds=new DiagSettings(); ds.setVisible(true);

if(ds.change)

{

mf.tab\_save=false; mf.diag\_save=false;

if(SettingsContainer.getSettings().col\_for\_diag)

{

mf.Tab.setRowSelectionAllowed(false);

mf.Tab.setColumnSelectionAllowed(true);

}

else

{

mf.Tab.setRowSelectionAllowed(true);

mf.Tab.setColumnSelectionAllowed(false);

}

MainTabModel md=(MainTabModel)mf.Tab.getModel();

if(pcfd!=SettingsContainer.getSettings().col\_for\_diag)

{

md.setSelected(-1);

CurrentTable.getTable().diag\_ind=-1;

if(mf.diag\_drw.have\_data) mf.diag\_drw.Clear();

}

else

{

if(mf.diag\_drw.have\_data)

{

mf.diag\_drw.SetColor(CurrentTable.getTable().diag\_col);

mf.diag\_drw.repaint();

}

}

md.reinit();

}

}

}

//----------------------------------------------------

//Listeners for HELP

/\*\*

\* Listener of menu item Help->About

\*/

class HAboutListener implements ActionListener

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* is called when user choose menu item

\*/

public void actionPerformed(ActionEvent arg0)

{

AboutDialog ad=new AboutDialog();

ad.setVisible(true);

}

}

//------------------------------------------------------

//For JTable

class MainTabModel extends AbstractTableModel

{

/\*\*

\* handle on main frame

\*/

private JMainFrame mf;

/\*\*

\* number of rows and columns

\*/

private int rows, cols;

/\*\*

\* index of selected row or column

\*/

public int selected;

/\*\*

\* constructor

\* @param fr handle on main frame

\*/

MainTabModel(JMainFrame fr)

{

mf=fr;

selected=-1;

if(CurrentTable.getTable()==null){rows=0; cols=0;}

else

{

rows=CurrentTable.getTable().rows;

if(CurrentTable.getTable().col\_head!=null) rows++;

cols=CurrentTable.getTable().cols;

if(CurrentTable.getTable().row\_head!=null) cols++;

if(SettingsContainer.getSettings().sh\_row\_num) cols++;

}

}

/\*\*

\* returns number of columns

\*/

public int getColumnCount(){return cols;}

/\*\*

\* returns number of rows

\*/

public int getRowCount(){return rows;}

/\*\*

\* returns names of columns

\*/

public String getColumnName(int col)

{

if(SettingsContainer.getSettings().sh\_col\_num)

{

if(SettingsContainer.getSettings().sh\_row\_num)

{

if(col==0) return new String("");

col--;

}

return new String(Integer.toString(col+1));

}

return new String("");

}

/\*\*

\* returns value that is saved in cell

\* @param row row which contains cell

\* @param col column which contains cell

\*/

public Object getValueAt(int row, int col)

{

if(CurrentTable.getTable()==null) return null;

if(CurrentTable.getTable().col\_head!=null)

{

if(row==0)

{

if(CurrentTable.getTable().row\_head!=null) col--;

if(SettingsContainer.getSettings().sh\_row\_num) col--;

return CurrentTable.getTable().getHeader(col, false);

}

row--;

}

if(SettingsContainer.getSettings().sh\_row\_num)

{

if(col==0) return new String(Integer.toString(row+1));

col--;

}

if(CurrentTable.getTable().row\_head!=null)

{

if(col==0) return CurrentTable.getTable().getHeader(row, true);

col--;

}

if(col>=0 && row>=0)

return CurrentTable.getTable().getData(row, col);

return new String("");

}

/\*\*

\* returns true if cell can be edited

\* @param row row which contains cell

\* @param col column which contains cell

\*/

public boolean isCellEditable(int row, int col)

{

if(CurrentTable.getTable()==null) return false;

if(SettingsContainer.getSettings().sh\_row\_num && col==0) return false;

return true;

}

/\*\*

\* writes value to the cell

\* @param row row which contains cell

\* @param col column which contains cell

\*/

public void setValueAt(Object value, int row, int col)

{

int r=row, c=col;

if(CurrentTable.getTable()==null) return;

if(CurrentTable.getTable().col\_head!=null)

{

if(row==0)

{

if(CurrentTable.getTable().row\_head!=null) col--;

if(SettingsContainer.getSettings().sh\_row\_num) col--;

if(col>=0) CurrentTable.getTable().setHeader(col, false, (String)value);

return;

}

row--;

}

if(SettingsContainer.getSettings().sh\_row\_num)

{

if(col==0) return;

col--;

}

if(CurrentTable.getTable().row\_head!=null)

{

if(col==0)

{

CurrentTable.getTable().setHeader(row, true, (String)value);

return;

}

col--;

}

try

{

CurrentTable.getTable().setData(row, col, Double.parseDouble((String)value));

}

catch(Exception e)

{

JOptionPane.showConfirmDialog(null, new String("Type double is expected."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

}

fireTableCellUpdated(r, c);

}

/\*\*

\* updates view of the table

\*/

public void reinit()

{

if(CurrentTable.getTable()==null){rows=0; cols=0;}

else

{

rows=CurrentTable.getTable().rows;

if(CurrentTable.getTable().col\_head!=null) rows++;

cols=CurrentTable.getTable().cols;

if(CurrentTable.getTable().row\_head!=null) cols++;

if(SettingsContainer.getSettings().sh\_row\_num) cols++;

}

this.fireTableStructureChanged();

this.fireTableDataChanged();

}

/\*\*

\* sets selection of the table

\* @param sel index of selected row/column

\*/

public void setSelected(int sel)

{

selected=sel;

fireTableDataChanged();

}

}

class MainTabRend extends JLabel implements TableCellRenderer

{

/\*\*

\* handle on table which is rendered

\*/

private JTable tab;

/\*\*

\* constructor

\* @param tab table that is rendered

\*/

public MainTabRend(JTable tab)

{

setOpaque(false);

this.tab=tab;

}

/\*\*

\* @param tab table that is rendering

\* @param val value that should be shown

\* @param isSelected

\* @param hasFocus

\* @param row row that contains cell

\* @param col column that contains cell

\*/

public Component getTableCellRendererComponent(JTable tab, Object val,

boolean isSelected, boolean hasFocus,

int row, int col)

{

if(CurrentTable.getTable()==null) return null;

this.setForeground(new Color(0, 0, 0));

this.setOpaque(false); this.setBackground(new Color(255, 255, 255));

if(CurrentTable.getTable().col\_head!=null)

{

if(row==0)

{

setText((String)val);

if(SettingsContainer.getSettings().col\_col!=null)

{

setOpaque(true);

setBackground(SettingsContainer.getSettings().col\_col);

}

return this;

}

row--;

}

if(SettingsContainer.getSettings().sh\_row\_num)

{

if(col==0)

{

setText((String)val); setBackground(new Color(236, 233, 216));

setOpaque(true);

return this;

}

col--;

}

if(CurrentTable.getTable().row\_head!=null)

{

if(col==0)

{

this.setText((String)val);

if(SettingsContainer.getSettings().row\_col!=null)

{

setBackground(SettingsContainer.getSettings().row\_col);

setOpaque(true);

}

return this;

}

col--;

}

String dv=Double.toString((Double)val);

String rv=new String();

int i, j;

for(i=dv.length()-1; i>=0; i--)

if(dv.charAt(i)=='.') break;

i+=3;

for(j=0; j<i && j<dv.length(); j++)

rv+=dv.charAt(j);

this.setText(rv);

MainTabModel md=(MainTabModel)tab.getModel();

if(SettingsContainer.getSettings().col\_for\_diag==true)

{

if(CurrentTable.getTable().diag\_ind==col)

{

this.setBackground((Color)CurrentTable.getTable().diag\_col.get(row));

setOpaque(true);

}

else

{

if(md.selected==col)

{

setForeground(new Color(0, 255, 0));

setOpaque(true);

}

}

}

if(SettingsContainer.getSettings().col\_for\_diag==false)

{

if(CurrentTable.getTable().diag\_ind==row)

{

this.setBackground((Color)CurrentTable.getTable().diag\_col.get(col));

setOpaque(true);

}

else

{

if(md.selected==row)

{

setForeground(new Color(0, 255, 0));

setOpaque(true);

}

}

}

return this;

}

}

/\*\*

\* Monitors changing of rows selection

\*/

class MainTabRowListener implements ListSelectionListener

{

/\*\*

\* handle on table which is rendered

\*/

private JTable tab;

/\*\*

\* constructor

\* @param tab table that is rendered

\*/

public MainTabRowListener(JTable tab){this.tab=tab;}

/\*\*

\* is called when row selection is changed

\*/

public void valueChanged(ListSelectionEvent e)

{

if(CurrentTable.getTable()==null) return;

if(!tab.getRowSelectionAllowed()) return;

int ind=tab.getSelectedRow();

if(CurrentTable.getTable().col\_head!=null) ind--;

if(ind>=0)

{

MainTabModel md=(MainTabModel)tab.getModel();

md.setSelected(ind);

}

}

}

/\*\*

\* Monitors changing of column selection

\*/

class MainTabColumnListener implements ListSelectionListener

{

/\*\*

\* handle on table which is rendered

\*/

private JTable tab;

/\*\*

\* constructor

\* @param tab table that is rendered

\*/

public MainTabColumnListener(JTable tab){this.tab=tab;}

/\*\*

\* is called when column selection is changed

\*/

public void valueChanged(ListSelectionEvent e)

{

if(CurrentTable.getTable()==null) return;

if(!tab.getColumnSelectionAllowed()) return;

int ind=tab.getSelectedColumn();

if(SettingsContainer.getSettings().sh\_row\_num) ind--;

if(CurrentTable.getTable().row\_head!=null) ind--;

if(ind>=0)

{

MainTabModel md=(MainTabModel)tab.getModel();

md.setSelected(ind);

}

}

}

package DiagrammerApp;

import javax.swing.JButton;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.WindowConstants;

import javax.swing.SwingUtilities;

/\*\*

\* shows information about application

\*/

public class AboutDialog extends javax.swing.JDialog

{

private JLabel jLabel1;

private JLabel jLabel2;

private JLabel jLabel3;

private JLabel jLabel4;

private JPanel ImgPane;

private JButton Like;

/\*\*

\* constructor

\*/

public AboutDialog()

{

super();

initGUI();

this.setModal(true);

}

/\*\*

\* creates GUI

\*/

private void initGUI() {

try {

setDefaultCloseOperation(WindowConstants.DISPOSE\_ON\_CLOSE);

getContentPane().setLayout(null);

this.setResizable(false);

this.setTitle("About");

{

jLabel1 = new JLabel();

getContentPane().add(jLabel1);

jLabel1.setText("Diagrammer Application v1.0");

jLabel1.setBounds(12, 6, 214, 15);

}

{

jLabel2 = new JLabel();

getContentPane().add(jLabel2);

jLabel2.setText("Cursach edition");

jLabel2.setBounds(12, 27, 267, 14);

}

{

jLabel3 = new JLabel();

getContentPane().add(jLabel3);

jLabel3.setText("No copy rights");

jLabel3.setBounds(12, 47, 190, 14);

}

{

jLabel4 = new JLabel();

getContentPane().add(jLabel4);

jLabel4.setText("For palivo enabled");

jLabel4.setBounds(12, 67, 237, 14);

}

{

Like = new JButton();

getContentPane().add(Like);

Like.setText("I like it");

Like.setBounds(12, 93, 108, 21);

}

{

ImgPane = new JPanel();

getContentPane().add(ImgPane);

ImgPane.setBounds(180, 9, 190, 132);

}

pack();

this.setSize(387, 184);

} catch (Exception e) {

//add your error handling code here

e.printStackTrace();

}

}

}

package DiagrammerApp;

import java.awt.Color;

import java.awt.Component;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

import javax.swing.AbstractCellEditor;

import javax.swing.ButtonGroup;

import javax.swing.JButton;

import javax.swing.JColorChooser;

import javax.swing.JDialog;

import javax.swing.JLabel;

import javax.swing.JRadioButton;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.WindowConstants;

import javax.swing.border.Border;

import javax.swing.event.TableModelEvent;

import javax.swing.event.TableModelListener;

import javax.swing.table.DefaultTableCellRenderer;

import javax.swing.table.DefaultTableModel;

import javax.swing.table.TableCellEditor;

import javax.swing.table.TableCellRenderer;

import javax.swing.table.TableModel;

import javax.swing.SwingUtilities;

import DiagCore.CurrentTable;

import DiagCore.SettingsContainer;

/\*\*

\* class dedicated to show and change settings of diagram

\*/

public class DiagSettings extends javax.swing.JDialog

{

private JScrollPane ScrSetTab;

private JButton Ok;

private ButtonGroup buttonGroup;

private JLabel jLabel1;

public JRadioButton diag\_row;

public JRadioButton diag\_col;

private JButton Cancel;

public JTable SetTab;

/\*\*

\* shows if settings were changed

\*/

public boolean change;

/\*\*

\* constructor

\*/

public DiagSettings()

{

super();

initGUI();

this.setModal(true);

}

/\*\*

\* creates GUI

\*/

private void initGUI()

{

try {

this.setDefaultCloseOperation(WindowConstants.DO\_NOTHING\_ON\_CLOSE);

getContentPane().setLayout(null);

this.setTitle("Diagram settings");

this.setResizable(false);

{

ScrSetTab = new JScrollPane();

getContentPane().add(ScrSetTab);

ScrSetTab.setBounds(12, 56, 240, 214);

{

int i, r;

String [] head={"Row №", "Color"};

Object [][] data;

if(SettingsContainer.getSettings().col\_for\_diag)

r=CurrentTable.getTable().rows;

else

r=CurrentTable.getTable().cols;

data=new Object[r][];

for(i=0; i<r; i++)

{

data[i]=new Object[2];

data[i][0]=new String(Integer.toString(r+1));

data[i][1]=CurrentTable.getTable().diag\_col.get(i);

}

SetTab = new JTable(data, head);

ScrSetTab.setViewportView(SetTab);

SetTabModel mod=new SetTabModel();

SetTab.setModel(mod);

SetTab.setDefaultRenderer(Color.class, new ColorRenderer());

SetTab.setDefaultEditor(Color.class, new ColorEditor(this));

SetTab.setAutoResizeMode(JTable.AUTO\_RESIZE\_ALL\_COLUMNS);

}

}

{

Ok = new JButton();

getContentPane().add(Ok);

Ok.setText("OK");

Ok.setBounds(12, 282, 85, 21);

Ok.addActionListener(new DiagSetOk(this));

}

{

Cancel = new JButton();

getContentPane().add(Cancel);

Cancel.setText("Cancel");

Cancel.setBounds(165, 282, 87, 21);

Cancel.addActionListener(new DiagSetCan(this));

} {

diag\_col = new JRadioButton();

getContentPane().add(diag\_col);

diag\_col.setText("column");

diag\_col.setBounds(12, 27, 72, 18);

diag\_col.setActionCommand("column");

diag\_col.addActionListener(new DiagColRow(this));

getButtonGroup().add(diag\_col);

if(SettingsContainer.getSettings().col\_for\_diag)

diag\_col.setSelected(true);

}

{

diag\_row = new JRadioButton();

getContentPane().add(diag\_row);

diag\_row.setText("row");

diag\_row.setBounds(89, 25, 67, 22);

diag\_row.setActionCommand("row");

diag\_row.addActionListener(new DiagColRow(this));

getButtonGroup().add(diag\_row);

if(!SettingsContainer.getSettings().col\_for\_diag)

diag\_row.setSelected(true);

}

{

jLabel1 = new JLabel();

getContentPane().add(jLabel1);

jLabel1.setText("Build diagram using");

jLabel1.setBounds(12, 9, 134, 14);

}

pack();

this.setSize(272, 356);

} catch (Exception e) {

//add your error handling code here

e.printStackTrace();

}

}

/\*\*

\* returns buttonGroup

\*/

private ButtonGroup getButtonGroup()

{

if(buttonGroup == null) {

buttonGroup = new ButtonGroup();

}

return buttonGroup;

}

}

/\*\*

\* Model of table used to show settings

\*/

class SetTabModel extends DefaultTableModel

{

public ArrayList row\_col;

private int rows;

/\*\*

\* constructor

\*/

public SetTabModel()

{

if(SettingsContainer.getSettings().col\_for\_diag)

rows=CurrentTable.getTable().rows;

else

rows=CurrentTable.getTable().cols;

row\_col=CurrentTable.getTable().diag\_col;

}

/\*\*

\* setter for row count

\*/

public void setRowCount(int rows)

{

if(rows==this.rows) return;

this.rows=rows;

this.fireTableStructureChanged();

}

/\*\*

\* returns number of columns

\*/

public int getColumnCount(){return 2;}

/\*\*

\* returns number of rows

\*/

public int getRowCount()

{

return rows;

}

/\*\*

\* returns column name

\*/

public String getColumnName(int col)

{

if(col==0) return new String("Row №");

if(col==1) return new String("Color");

return null;

}

/\*\*

\* returns value in cell

\*/

public Object getValueAt(int row, int col)

{

if(col==0)

return new String(Integer.toString(row+1));

if(col==1)

return row\_col.get(row);

return null;

}

/\*\*

\* returns class of column

\*/

public Class getColumnClass(int c)

{

if(c==0) return String.class;

return Color.class;

}

/\*\*

\* returns if cell is editable

\*/

public boolean isCellEditable(int row, int col)

{

if (col==1) return true;

return false;

}

/\*\*

\* setts value in cell

\*/

public void setValueAt(Object value, int row, int col)

{

if(col==1) row\_col.set(row, (Color)value);

fireTableCellUpdated(row, col);

}

}

/\*\*

\* default renderer of a color

\*/

class ColorRenderer extends JLabel implements TableCellRenderer

{

/\*\*

\* constructors

\*/

public ColorRenderer(){setOpaque(true);}

/\*\*

\* is called when cell must be shown

\*/

public Component getTableCellRendererComponent(JTable table, Object color,

boolean isSelected, boolean hasFocus,

int row, int column)

{

setBackground((Color)color);

return this;

}

}

/\*\*

\* default editor of cells

\*/

class ColorEditor extends AbstractCellEditor implements TableCellEditor, ActionListener

{

private Color cur\_col;

private JButton but;

private JColorChooser col\_ch;

private JDialog dlg;

private DiagSettings ds;

/\*\*

\* constructor

\* @param fr handle on dialog

\*/

public ColorEditor(DiagSettings fr)

{

ds=fr;

but = new JButton();

but.addActionListener(this);

but.setActionCommand("edit");

but.setBorderPainted(false);

col\_ch = new JColorChooser();

dlg = JColorChooser.createDialog(but, "Pick a Color", true, col\_ch,

this, null);

}

/\*\*

\* is called when user tries to edit cell

\*/

public void actionPerformed(ActionEvent e)

{

if(e.getActionCommand().equals("edit"))

{

but.setBackground(cur\_col);

col\_ch.setColor(cur\_col);

dlg.setVisible(true);

fireEditingStopped();

}

else

{

cur\_col = col\_ch.getColor();

}

}

/\*\*

\* returns new value

\*/

public Object getCellEditorValue(){return cur\_col;}

/\*\*

\* returns cell editor

\*/

public Component getTableCellEditorComponent(JTable table, Object value,

boolean isSelected, int row, int column)

{

cur\_col = (Color)value;

return but;

}

}

/\*\*

\* lister of button OK

\*/

class DiagSetOk implements ActionListener

{

/\*\*

\* handle on dialog

\*/

private DiagSettings ds;

/\*\*

\* constructor

\*/

DiagSetOk(DiagSettings fr){ds=fr;}

/\*\*

\* is called when button is pressed

\*/

public void actionPerformed(ActionEvent arg0)

{

ds.change=true;

SettingsContainer.getSettings().col\_for\_diag=ds.diag\_col.isSelected();

CurrentTable.getTable().diag\_col=new ArrayList(((SetTabModel)ds.SetTab.getModel()).row\_col);

ds.dispose();

}

}

/\*\*

\* listener of button Cancel

\*/

class DiagSetCan implements ActionListener

{

/\*\*

\* handle on dialog

\*/

private DiagSettings ds;

/\*\*

\* constructor

\*/

DiagSetCan(DiagSettings fr){ds=fr;}

/\*\*

\* is called when button is pressed

\*/

public void actionPerformed(ActionEvent arg0)

{

ds.change=false;

ds.dispose();

}

}

/\*\*

\* listener of radio buttons

\*/

class DiagColRow implements ActionListener

{

/\*\*

\* handle on dialog

\*/

private DiagSettings ds;

/\*\*

\* constructor

\*/

public DiagColRow(DiagSettings fr){ds=fr;}

/\*\*

\* is called when selection is changed

\*/

public void actionPerformed(ActionEvent e)

{

if(e.getActionCommand().equals("row"))

{

SetTabModel m=(SetTabModel)ds.SetTab.getModel();

m.row\_col=SettingsContainer.getSettings().getDefaultDiagramColors(CurrentTable.getTable().cols);

m.setRowCount(CurrentTable.getTable().cols);

}

if(e.getActionCommand().equals("column"))

{

SetTabModel m=(SetTabModel)ds.SetTab.getModel();

m.row\_col=SettingsContainer.getSettings().getDefaultDiagramColors(CurrentTable.getTable().rows);

m.setRowCount(CurrentTable.getTable().rows);

}

}

}

package DiagrammerApp;

import java.awt.Graphics2D;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import javax.swing.ButtonGroup;

import javax.swing.JButton;

import javax.swing.JCheckBox;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JRadioButton;

import javax.swing.JTextField;

import javax.swing.WindowConstants;

import javax.swing.SwingUtilities;

import DiagCore.\*;

/\*\*

\* dialog that provides possibility to save table or diagram

\*/

public class Saver extends javax.swing.JDialog

{

public JMainFrame mf;

public JTextField Path;

private JButton Browse;

private JLabel jLabel1;

private JLabel jLabel2;

private JButton Cancel;

private JButton Save;

private ButtonGroup buttonGroup;

public JRadioButton tp2\_ch;

public JRadioButton tp1\_ch;

public boolean sv\_tab;

public boolean saved;

/\*\*

\* constructor

\* @param save\_table equals true while saving table, otherwise it is false

\*/

public Saver(JMainFrame mf, boolean save\_table)

{

super();

sv\_tab=save\_table;

this.mf=mf;

initGUI();

this.setModal(true);

}

/\*\*

\* creates GUI

\*/

private void initGUI()

{

try {

this.setDefaultCloseOperation(WindowConstants.DO\_NOTHING\_ON\_CLOSE);

getContentPane().setLayout(null);

this.setResizable(false);

this.setTitle("Save");

{

Path = new JTextField();

getContentPane().add(Path);

Path.setBounds(80, 9, 332, 21);

}

{

Browse = new JButton();

getContentPane().add(Browse);

Browse.setText("Browse");

Browse.setBounds(430, 9, 104, 21);

Browse.addActionListener(new BrowseListener(Path));

}

{

jLabel1 = new JLabel();

getContentPane().add(jLabel1);

jLabel1.setText("Path");

jLabel1.setBounds(12, 12, 62, 14);

}

{

jLabel2 = new JLabel();

getContentPane().add(jLabel2);

jLabel2.setText("Type");

jLabel2.setBounds(12, 47, 50, 14);

}

{

tp1\_ch = new JRadioButton();

if(sv\_tab) tp1\_ch.setText("csv");

else tp1\_ch.setText("png");

tp1\_ch.setBounds(80, 45, 129, 18);

tp1\_ch.setSelected(true);

getContentPane().add(tp1\_ch);

getButtonGroup().add(tp1\_ch);

}

{

tp2\_ch = new JRadioButton();

if(sv\_tab) tp2\_ch.setText("dat");

else tp2\_ch.setText("jpg");

tp2\_ch.setBounds(229, 45, 109, 18);

getContentPane().add(tp2\_ch);

getButtonGroup().add(tp2\_ch);

}

{

Save = new JButton();

getContentPane().add(Save);

Save.setText("Save");

Save.setBounds(354, 40, 80, 21);

Save.addActionListener(new SaverSave(this));

}

{

Cancel = new JButton();

getContentPane().add(Cancel);

Cancel.setText("Cancel");

Cancel.setBounds(450, 40, 84, 21);

Cancel.addActionListener(new SaverCancel(this));

}

pack();

this.setSize(554, 106);

} catch (Exception e) {

//add your error handling code here

e.printStackTrace();

}

}

/\*\*

\* @return buttonGroup

\*/

private ButtonGroup getButtonGroup()

{

if(buttonGroup == null)

{

buttonGroup = new ButtonGroup();

}

return buttonGroup;

}

}

/\*\*

\* Listener of button Save

\*/

class SaverSave implements ActionListener

{

/\*\*

\* handle on dialog

\*/

private Saver sv;

/\*\*

\* constructor

\* @param fr dialog

\*/

SaverSave(Saver fr){sv=fr;}

/\*\*

\* is called when button is pushed

\*/

public void actionPerformed(ActionEvent arg0)

{

if(sv.Path.getText().equals(""))

{

JOptionPane.showConfirmDialog(null, new String("Path to file must be defined."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

File out;

if(sv.sv\_tab)

{

if(sv.tp1\_ch.isSelected()) out=new File(sv.Path.getText()+".csv");

else out=new File(sv.Path.getText()+".dat");

if(!out.exists())

{

try{out.createNewFile();}

catch(IOException e1)

{

JOptionPane.showConfirmDialog(null, new String("Can not create file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

}

CurrentTable.getTable().file=new String(sv.Path.getText());

String [][] str\_dat=CurrentTable.getTable().getInTextFormat();

CSVProcessor csv=new CSVProcessor();

csv.SetDelimiter(SettingsContainer.getSettings().del\_val[SettingsContainer.getSettings().delim]);

csv.BuildTextData(str\_dat);

if(sv.tp1\_ch.isSelected())

{

csv.SaveTextData(sv.Path.getText());

CurrentTable.getTable().text\_format=true;

}

else

{

csv.SaveBinData(sv.Path.getText());

CurrentTable.getTable().text\_format=false;

}

}

else

{

if(sv.tp1\_ch.isSelected()) CurrentTable.getTable().gfile=new String(sv.Path.getText()+".png");

else CurrentTable.getTable().gfile=new String(sv.Path.getText()+".jpeg");

out=new File(CurrentTable.getTable().gfile);

if(!out.isFile() || !out.exists())

{

try{out.createNewFile();}

catch(Exception e2)

{

JOptionPane.showConfirmDialog(null, new String("Can not create file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

}

BufferedImage bi=new BufferedImage(sv.mf.diag\_drw.getSize().width, sv.mf.diag\_drw.getSize().height, BufferedImage.TYPE\_INT\_RGB);

Graphics2D g2=bi.createGraphics();

g2.setClip(0, 0, sv.mf.diag\_drw.getBounds().width, sv.mf.diag\_drw.getBounds().height);

sv.mf.diag\_drw.draw(g2);

try

{

if(sv.tp1\_ch.isSelected()) javax.imageio.ImageIO.write(bi, "png", out);

else javax.imageio.ImageIO.write(bi, "jpeg", out);

}

catch(Exception e)

{

JOptionPane.showConfirmDialog(null, new String("Error while writing to file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

sv.mf.diag\_save=false;

}

}

sv.saved=true;

sv.dispose();

}

}

/\*\*

\* Listener of button Save

\*/

class SaverCancel implements ActionListener

{

/\*\*

\* handle on dialog

\*/

private Saver sv;

/\*\*

\* constructor

\* @param fr dialog

\*/

SaverCancel(Saver fr){sv=fr;}

/\*\*

\* is called when button is pushed

\*/

public void actionPerformed(ActionEvent arg0)

{

sv.saved=false;

sv.dispose();

}

}

package DiagrammerApp;

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JCheckBox;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JTextField;

import javax.swing.SwingUtilities;

import DiagCore.CurrentTable;

/\*\*

\* dialog than helps to create table

\*/

public class TableCreator extends javax.swing.JDialog

{

private JButton Ok;

private JLabel jLabel2;

private JLabel jLabel1;

public JTextField row\_num;

public JTextField col\_num;

public JCheckBox row\_head;

public JCheckBox col\_head;

private JButton Cancel;

public boolean changed;

/\*\*

\* constructor

\*/

public TableCreator()

{

super();

initGUI();

}

/\*\*

\* creates GUI

\*/

private void initGUI()

{

try

{

{

getContentPane().setLayout(null);

this.setTitle("Create table");

this.setModal(true);

this.setDefaultCloseOperation(DO\_NOTHING\_ON\_CLOSE);

}

{

Ok = new JButton();

getContentPane().add(Ok, "Center");

Ok.setText("OK");

Ok.setBounds(12, 92, 114, 22);

Ok.addActionListener(new OkListener(this));

}

{

Cancel = new JButton();

getContentPane().add(Cancel);

Cancel.setText("Cancel");

Cancel.setBounds(166, 92, 114, 21);

Cancel.addActionListener(new CancListener(this));

}

{

col\_head = new JCheckBox();

getContentPane().add(col\_head);

col\_head.setText("Header");

col\_head.setBounds(12, 15, 137, 18);

}

{

row\_head = new JCheckBox();

getContentPane().add(row\_head);

row\_head.setText("Labeled rows");

row\_head.setBounds(12, 40, 185, 18);

}

{

col\_num = new JTextField();

getContentPane().add(col\_num);

col\_num.setBounds(289, 14, 55, 20);

}

{

row\_num = new JTextField();

getContentPane().add(row\_num);

row\_num.setBounds(289, 39, 55, 20);

}

{

jLabel1 = new JLabel();

getContentPane().add(jLabel1);

jLabel1.setText("Number of columns");

jLabel1.setBounds(362, 14, 213, 20);

}

{

jLabel2 = new JLabel();

getContentPane().add(jLabel2);

jLabel2.setText("Number of rows");

jLabel2.setBounds(362, 39, 213, 20);

}

this.setSize(590, 165);

} catch (Exception e) {

e.printStackTrace();

}

}

}

/\*\*

\* listener for button OK

\*/

class OkListener implements ActionListener

{

/\*\*

\* handle to dialog

\*/

private TableCreator tc;

/\*\*

\* constructor

\*/

public OkListener(TableCreator mf){tc=mf;}

/\*\*

\* is called when the button is pressed

\*/

public void actionPerformed(ActionEvent arg0)

{

int c, r;

try

{

c=Integer.parseInt(tc.col\_num.getText());

r=Integer.parseInt(tc.row\_num.getText());

}

catch(Exception e)

{

JOptionPane.showConfirmDialog(null, new String("Number of rows or columns must be integer."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

CurrentTable.createTable(r, c, tc.col\_head.isSelected(), tc.row\_head.isSelected());

tc.changed=true; tc.dispose();

}

}

/\*\*

\* listener for button Cancel

\*/

class CancListener implements ActionListener

{

/\*\*

\* handle to dialog

\*/

private TableCreator tc;

/\*\*

\* constructor

\*/

public CancListener(TableCreator mf){tc=mf;}

/\*\*

\* is called when the button is pressed

\*/

public void actionPerformed(ActionEvent arg0)

{

tc.changed=false;

tc.dispose();

}

}

package DiagrammerApp;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;

import java.util.ArrayList;

import javax.swing.ComboBoxModel;

import javax.swing.DefaultComboBoxModel;

import javax.swing.JButton;

import javax.swing.JCheckBox;

import javax.swing.JComboBox;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JScrollPane;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.WindowConstants;

import javax.swing.table.AbstractTableModel;

import javax.swing.table.DefaultTableModel;

import javax.swing.table.TableModel;

import javax.swing.SwingUtilities;

import DiagCore.\*;

/\*\*

\* class that loads table

\*/

public class TableLoader extends javax.swing.JDialog

{

private JScrollPane ScrResTab;

public JTable ResTab;

public JTextField PathEdit;

private JLabel jLabel2;

private JButton Cancel;

private JButton Load;

public JCheckBox row\_name;

public JCheckBox head;

public JComboBox Delim;

private JButton Browse;

private JLabel jLabel1;

private JButton ViewTab;

/\*\*

\* constructor

\*/

public TableLoader()

{

super();

initGUI();

this.setModal(true);

}

/\*\*

\* creates GUI

\*/

private void initGUI()

{

try {

this.setDefaultCloseOperation(WindowConstants.DO\_NOTHING\_ON\_CLOSE);

getContentPane().setLayout(null);

this.setTitle("Loading");

this.setResizable(false);

{

ScrResTab = new JScrollPane();

getContentPane().add(ScrResTab);

ScrResTab.setBounds(216, 53, 406, 154);

{

ResTab = new JTable();

ScrResTab.setViewportView(ResTab);

ResTab.setModel(new LoadTabModel());

}

}

{

jLabel1 = new JLabel();

getContentPane().add(jLabel1);

jLabel1.setText("Path");

jLabel1.setBounds(5, 12, 58, 14);

}

{

PathEdit = new JTextField();

getContentPane().add(PathEdit);

PathEdit.setBounds(69, 9, 460, 21);

if(SettingsContainer.getSettings().lst\_file!=null)

PathEdit.setText(SettingsContainer.getSettings().lst\_file);

}

{

Browse = new JButton();

getContentPane().add(Browse);

Browse.setText("Browse");

Browse.setBounds(535, 9, 82, 21);

Browse.addActionListener(new BrowseListener(PathEdit));

}

{

ComboBoxModel DelimModel = new DefaultComboBoxModel();

Delim = new JComboBox();

getContentPane().add(Delim);

Delim.setModel(DelimModel);

Delim.setBounds(85, 51, 119, 20);

for(int i=0; i<SettingsContainer.getSettings().DELIM\_NUMBER; i++)

Delim.addItem(new String("'"+SettingsContainer.getSettings().del\_val[i]+"'"));

Delim.setSelectedIndex(SettingsContainer.getSettings().delim);

}

{

jLabel2 = new JLabel();

getContentPane().add(jLabel2);

jLabel2.setText("Delimiters");

jLabel2.setBounds(5, 54, 74, 14);

}

{

head = new JCheckBox();

getContentPane().add(head);

head.setText("Header is present");

head.setBounds(5, 83, 146, 18);

}

{

row\_name = new JCheckBox();

getContentPane().add(row\_name);

row\_name.setText("Rows are named");

row\_name.setBounds(5, 110, 146, 18);

}

{

Load = new JButton();

getContentPane().add(Load);

Load.setText("Load");

Load.setBounds(5, 158, 81, 21);

Load.addActionListener(new TabLoadOk(this));

}

{

Cancel = new JButton();

getContentPane().add(Cancel);

Cancel.setText("Cancel");

Cancel.setBounds(97, 158, 86, 21);

Cancel.addActionListener(new TabLoadCan(this));

}

{

ViewTab = new JButton();

getContentPane().add(ViewTab);

ViewTab.setText("Preview");

ViewTab.setBounds(5, 184, 178, 21);

ViewTab.addActionListener(new TabLoadView(this));

}

pack();

this.setSize(630, 243);

} catch (Exception e) {

//add your error handling code here

e.printStackTrace();

}

}

}

/\*\*

\* Listener for button Load

\*/

class TabLoadOk implements ActionListener

{

/\*\*

\* handle on dialog

\*/

private TableLoader tl;

/\*\*

\* constructor

\*/

public TabLoadOk(TableLoader fr){tl=fr;}

/\*\*

\* is called when the button is pushed

\*/

public void actionPerformed(ActionEvent arg0)

{

int r, c, i, j;

int rv, cv;

double v;

boolean fe;

String [][] str\_dat;

CSVProcessor csv=new CSVProcessor();

SettingsContainer.getSettings().delim=tl.Delim.getSelectedIndex();

if(tl.PathEdit.getText().equals(new String("")))

{

JOptionPane.showConfirmDialog(null, new String("Path is not defined."),

"Error", JOptionPane.OK\_CANCEL\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

File in=new File(tl.PathEdit.getText());

String nm=in.getName();

String ext=new String("");

for(i=nm.length()-1; i>=0; i--)

{

if(nm.charAt(i)=='.') break;

}

if(i<0)

{

JOptionPane.showConfirmDialog(null, new String("Unknown format of the file."),

"Error", JOptionPane.OK\_CANCEL\_OPTION, JOptionPane.INFORMATION\_MESSAGE);

return;

}

i++;

for(; i<nm.length(); i++) ext+=nm.charAt(i);

if(ext.equals("csv"))

{

if(!csv.LoadTextData(tl.PathEdit.getText())) return;

}

else

{

if(ext.equals("dat"))

{

if(!csv.LoadBinData(tl.PathEdit.getText())) return;

}

else

{

JOptionPane.showConfirmDialog(null, new String("Unknown format of the file."),

"Error", JOptionPane.OK\_CANCEL\_OPTION, JOptionPane.INFORMATION\_MESSAGE);

return;

}

}

csv.SetDelimiter(SettingsContainer.getSettings().del\_val[SettingsContainer.getSettings().delim]);

if(null==(str\_dat=csv.ParseTextData()) || csv.state==-1)

{

JOptionPane.showConfirmDialog(null, new String("Table is not loaded."),

"Error", JOptionPane.OK\_CANCEL\_OPTION, JOptionPane.INFORMATION\_MESSAGE);

return;

}

r=str\_dat.length; if(tl.head.isSelected()) r--;

c=str\_dat[0].length; if(tl.row\_name.isSelected()) c--;

CurrentTable.createTable(r, c, tl.head.isSelected(), tl.row\_name.isSelected());

r=0; c=0;

if(tl.head.isSelected())

{

r=1;

if(tl.row\_name.isSelected()) i=1;

else i=0;

for(j=0; i<str\_dat[0].length; i++, j++)

{

CurrentTable.getTable().setHeader(j, false, str\_dat[0][i]);

}

}

if(tl.row\_name.isSelected())

{

c=1;

if(tl.head.isSelected()) i=1;

else i=0;

for(j=0; i<str\_dat.length; i++, j++)

{

CurrentTable.getTable().setHeader(j, true, str\_dat[i][0]);

}

}

fe=false;

for(i=r, rv=0; i<str\_dat.length; i++, rv++)

{

for(j=c, cv=0; j<str\_dat[i].length; j++, cv++)

{

try

{

v=Double.parseDouble(str\_dat[i][j]);

CurrentTable.getTable().setData(rv, cv, v);

}

catch(NumberFormatException e1)

{

CurrentTable.getTable().setData(rv, cv, 0); fe=true;

}

}

}

if(fe)

{

JOptionPane.showConfirmDialog(null, new String("Errors in number format occured."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

}

SettingsContainer.getSettings().lst\_file=new String(tl.PathEdit.getText());

tl.dispose();

}

}

/\*\*

\* Listener for button View

\*/

class TabLoadView implements ActionListener

{

/\*\*

\* handle to dialog

\*/

private TableLoader tl;

/\*\*

\* constructor

\*/

public TabLoadView(TableLoader fr){tl=fr;}

/\*\*

\* is called when button is pressed

\*/

public void actionPerformed(ActionEvent arg0)

{

int i, j, sr=0, sc;

double v;

boolean fe;

String [][] str\_dat;

CSVProcessor csv=new CSVProcessor();

if(tl.PathEdit.getText().equals(new String("")))

{

JOptionPane.showConfirmDialog(null, new String("Can not load table."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

File in=new File(tl.PathEdit.getText());

String nm=in.getName();

String ext=new String("");

for(i=nm.length()-1; i>=0; i--)

{

if(nm.charAt(i)=='.' || nm.charAt(i)=='\\') break;

}

if(i<0)

{

JOptionPane.showConfirmDialog(null, new String("Can not load table."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

if(nm.charAt(i)=='\\') return;

i++;

for(; i<nm.length(); i++) ext+=nm.charAt(i);

if(ext.equals("csv"))

{

if(!csv.LoadTextData(tl.PathEdit.getText())) return;

}

else

{

if(ext.equals("dat"))

{

if(!csv.LoadBinData(tl.PathEdit.getText())) return;

}

else

{

JOptionPane.showConfirmDialog(null, new String("Can not load table."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

}

csv.SetDelimiter(SettingsContainer.getSettings().del\_val[tl.Delim.getSelectedIndex()]);

if(null==(str\_dat=csv.ParseTextData()))

{

JOptionPane.showConfirmDialog(null, new String("Can not load table."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

fe=false;

ArrayList res=new ArrayList();

for(i=0; i<str\_dat.length; i++)

{

if(sr<5 || str\_dat[i].length!=str\_dat[0].length) res.add(str\_dat[i]);

else

{

for(j=0; j<str\_dat[i].length; j++)

{

if(!((i==0 && tl.head.isSelected()) || (j==0 && tl.row\_name.isSelected())))

{

try{v=Double.parseDouble(str\_dat[i][j]);}

catch(NumberFormatException e1){fe=true; break;}

}

}

if(j!=str\_dat[i].length) res.add(str\_dat[i]);

}

}

if(fe)

{

JOptionPane.showConfirmDialog(null, new String("Errors in number format occured."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

}

((LoadTabModel)tl.ResTab.getModel()).setData(res);

}

}

/\*\*

\* listener for button cancel

\*/

class TabLoadCan implements ActionListener

{

private TableLoader tl;

TabLoadCan(TableLoader fr){tl=fr;}

public void actionPerformed(ActionEvent arg0)

{

tl.dispose();

}

}

/\*\*

\* model of table for preview

\*/

class LoadTabModel extends AbstractTableModel

{

/\*\*

\* handle on main frame

\*/

private TableLoader mf;

private int rows, cols;

/\*\*

\* data of a table

\*/

private ArrayList data;

/\*\*

\* constructor

\*/

LoadTabModel(){rows=0; cols=0; data=new ArrayList();}

/\*\*

\* getter

\*/

public int getColumnCount(){return cols;}

/\*\*

\* getter

\*/

public int getRowCount(){return rows;}

/\*\*

\* getter

\*/

public String getColumnName(int col){return new String("");}

/\*\*

\* getter

\*/

public Object getValueAt(int row, int col)

{

if(data.size()<=row) return null;

String [] rw; rw=(String [])data.get(row);

if(rw.length<=col) return new String("");

return new String(rw[col]);

}

/\*\*

\* defines if cell is editable

\*/

public boolean isCellEditable(int row, int col){return false;}

/\*\*

\* setter

\*/

public void setData(ArrayList str\_dat)

{

data=str\_dat;

rows=data.size();

cols=0;

for(int i=0; i<data.size(); i++)

if(((String [])data.get(i)).length>cols) cols=((String [])data.get(i)).length;

this.fireTableStructureChanged();

}

}

package DiagrammerApp;

import java.awt.event.\*;

import javax.swing.BorderFactory;

import javax.swing.JButton;

import javax.swing.JCheckBox;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JTextField;

import javax.swing.WindowConstants;

import javax.swing.border.BevelBorder;

import javax.swing.border.LineBorder;

import javax.swing.SwingUtilities;

import DiagCore.ChangeColorListener;

import DiagCore.CurrentTable;

import DiagCore.SettingsContainer;

/\*\*

\* dialog that is used to change settings of a table

\*/

public class TabSettings extends javax.swing.JDialog

{

public JCheckBox row\_num;

public JCheckBox col\_num;

public JTextField cols;

private JLabel jLabel3;

private JLabel jLabel4;

private JButton Ok;

private JButton Cancel;

public JButton col\_col;

public JButton row\_col;

private JLabel jLabel2;

private JLabel jLabel1;

public JTextField rows;

public boolean change;

/\*\*

\* constructor

\*/

public TabSettings()

{

super();

initGUI();

this.setModal(true);

}

/\*\*

\* creates GUI

\*/

private void initGUI()

{

try {

this.setDefaultCloseOperation(WindowConstants.DO\_NOTHING\_ON\_CLOSE);

this.setTitle("Table settings");

getContentPane().setLayout(null);

this.setResizable(false);

{

row\_num = new JCheckBox();

getContentPane().add(row\_num);

row\_num.setText("Show number of the row");

row\_num.setBounds(12, 41, 173, 18);

row\_num.setSelected(SettingsContainer.getSettings().sh\_row\_num);

}

{

col\_num = new JCheckBox();

getContentPane().add(col\_num);

col\_num.setText("Show number of the column");

col\_num.setBounds(12, 66, 191, 18);

col\_num.setSelected(SettingsContainer.getSettings().sh\_col\_num);

}

{

rows = new JTextField();

getContentPane().add(rows);

rows.setBounds(344, 34, 48, 19);

rows.setText(Integer.toString(CurrentTable.getTable().rows));

}

{

cols = new JTextField();

getContentPane().add(cols);

cols.setBounds(344, 65, 48, 18);

cols.setText(Integer.toString(CurrentTable.getTable().cols));

}

{

jLabel1 = new JLabel();

getContentPane().add(jLabel1);

jLabel1.setText("Number of rows");

jLabel1.setBounds(214, 43, 112, 14);

}

{

jLabel2 = new JLabel();

getContentPane().add(jLabel2);

jLabel2.setText("Number of columns");

jLabel2.setBounds(214, 67, 118, 14);

}

{

jLabel3 = new JLabel();

getContentPane().add(jLabel3);

jLabel3.setText("Color of row header");

jLabel3.setBounds(12, 125, 139, 14);

}

{

jLabel4 = new JLabel();

getContentPane().add(jLabel4);

jLabel4.setText("Color of column header");

jLabel4.setBounds(12, 151, 139, 13);

}

{

row\_col = new JButton();

getContentPane().add(row\_col);

row\_col.setBounds(163, 120, 40, 17);

row\_col.setBackground(SettingsContainer.getSettings().row\_col);

row\_col.addActionListener(new ChangeColorListener(row\_col));

}

{

col\_col = new JButton();

getContentPane().add(col\_col);

col\_col.setBounds(163, 148, 40, 17);

col\_col.setBackground(SettingsContainer.getSettings().col\_col);

col\_col.addActionListener(new ChangeColorListener(col\_col));

}

{

Ok = new JButton();

getContentPane().add(Ok);

Ok.setText("OK");

Ok.setBounds(300, 147, 93, 22);

Ok.addActionListener(new TabSetOk(this));

}

{

Cancel = new JButton();

getContentPane().add(Cancel);

Cancel.setText("Cancel");

Cancel.setBounds(300, 122, 93, 22);

Cancel.addActionListener(new TabSetCan(this));

}

pack();

this.setSize(410, 213);

} catch (Exception e) {

//add your error handling code here

e.printStackTrace();

}

}

}

/\*\*

\* listener for button ok

\*/

class TabSetOk implements ActionListener

{

private TabSettings ts;

public TabSetOk(TabSettings fr){ts=fr;}

/\*\*

\* is called when button is pressed

\*/

public void actionPerformed(ActionEvent arg0)

{

int c, r;

int i;

try{r=Integer.parseInt(ts.rows.getText());}

catch(Exception e1)

{

JOptionPane.showConfirmDialog(null, new String("Number of rows is integer value."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

try{c=Integer.parseInt(ts.cols.getText());}

catch(Exception e2)

{

JOptionPane.showConfirmDialog(null, new String("Number of columns is integer value."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

if(r<=0 || c<=0)

{

JOptionPane.showConfirmDialog(null, new String("Number of rows or columns must be more then 0."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return;

}

ts.change=true;

SettingsContainer.getSettings().col\_col=ts.col\_col.getBackground();

SettingsContainer.getSettings().row\_col=ts.row\_col.getBackground();

SettingsContainer.getSettings().sh\_col\_num=ts.col\_num.isSelected();

SettingsContainer.getSettings().sh\_row\_num=ts.row\_num.isSelected();

if(r>CurrentTable.getTable().rows)

{

for(i=0; i<r-CurrentTable.getTable().rows; i++)

CurrentTable.getTable().addRow();

}

if(c>CurrentTable.getTable().cols)

{

for(i=0; i<c-CurrentTable.getTable().cols; i++)

CurrentTable.getTable().addColumn();

}

if(r<CurrentTable.getTable().rows)

{

for(i=CurrentTable.getTable().rows; i>r; i--)

CurrentTable.getTable().removeRow(i-1);

}

if(c<CurrentTable.getTable().cols)

{

for(i=CurrentTable.getTable().cols; i>c; i--)

CurrentTable.getTable().removeColumn(i-1);

}

//change table;

ts.dispose();

}

}

/\*\*

\* listener for button Cancel

\*/

class TabSetCan implements ActionListener

{

private TabSettings ts;

public TabSetCan(TabSettings fr){ts=fr;}

/\*\*

\* is called when button is poressed

\*/

public void actionPerformed(ActionEvent arg0)

{

ts.change=false;

ts.dispose();

}

}

package DiagCore;

import java.awt.Color;

import java.io.\*;

import java.util.ArrayList;

public class SettingsContainer implements Serializable

{

public Color row\_col, col\_col;

public int delim;

public boolean sh\_row\_num, sh\_col\_num;

public boolean col\_for\_diag;

public String lst\_file;

public static final int DELIM\_NUMBER=3;

public static final String [] del\_val={";", "; ", " "};

private SettingsContainer(){}

private static SettingsContainer set;

public static void createSettings()

{

set=new SettingsContainer();

set.delim=0;

set.row\_col=new Color(192, 192, 192);

set.col\_col=new Color(192, 192, 192);

set.col\_for\_diag=true;

}

public static SettingsContainer getSettings()

{

return set;

}

public static boolean loadSettings(File in)

{

if(!in.exists() || !in.isFile()) return false;

try

{

ObjectInputStream s=new ObjectInputStream(new FileInputStream(in));

set=(SettingsContainer)s.readObject();

s.close();

}

catch(Exception e1){return false;}

return true;

}

public static boolean saveSettings(File out)

{

if(out==null) return false;

if(!out.exists() || !out.isFile()) return false;

try

{

ObjectOutputStream d=new ObjectOutputStream(new FileOutputStream(out));

d.writeObject(set);

d.close();

}

catch(IOException e1){return false;}

return true;

}

public static ArrayList getDefaultDiagramColors(int col\_num)

{

if(col\_num<0) return null;

ArrayList res=new ArrayList();

int dc=255/((int)Math.exp(1.0/3.0\*Math.log(col\_num))+1);

int r=0, g, b, cn=0;

while(cn!=col\_num && r<=255)

{

g=0;

while(cn!=col\_num && g<=255)

{

b=0;

while(cn!=col\_num && b<=255)

{

res.add(new Color(r, g, b));

cn++; b+=dc;

}

g+=dc;

}

r+=dc;

}

return res;

}

public static Color getDefaultDiagramColor(ArrayList col)

{

Color res;

int i;

long c=0;

if(col==null)

return new Color((int)(255\*Math.random()), (int)(255\*Math.random()), (int)(255\*Math.random()));

while(true)

{

c++;

res=new Color((int)(255\*Math.random()), (int)(255\*Math.random()), (int)(255\*Math.random()));

for(i=0; i<col.size(); i++)

{

if(res.equals((Color)col.get(i))) break;

}

if(i==col.size()) return res;

if(c==16777217) return null;

}

}

}

package DiagCore;

import java.awt.Color;

import java.awt.Dimension;

import java.awt.Graphics;

import java.awt.Rectangle;

import java.awt.event.MouseEvent;

import java.awt.event.MouseListener;

import java.awt.event.MouseMotionListener;

import java.util.ArrayList;

import javax.swing.\*;

import DiagrammerApp.JMainFrame;

/\*\*

\* class dedicated to show diagram

\*/

public class DiagramDrawer extends JPanel

{

/\*\*

\* handle on main frame

\*/

public JMainFrame mf;

/\*\*

\* constructor

\*/

public DiagramDrawer()

{

super();

super.setBackground(new Color(255, 255, 255));

sz\_sel=-1;

this.addMouseListener(new DrMsEv(this));

this.addMouseMotionListener(new DrMsMv(this));

this.setBounds(0, 0, 300, 300);

mf=null; builded=false; have\_data=false;

}

/\*\*

\* list of double values

\*/

public ArrayList data;

/\*\*

\* contains angles

\*/

public int [][] ang;

/\*\*

\* colors which are used in diagram

\*/

public ArrayList color;

/\*\*

\* size of ellipse than is used to change angles

\*/

public int CHANGE\_ELLIPSE=10;

/\*\*

\* array of ellipses than are used to change angles

\*/

public Rectangle [] szrc;

/\*\*

\* shows if diagram is shown

\*/

public boolean builded;

/\*\*

\* shows if all data is defined

\*/

public boolean have\_data;

public int d\_cx, d\_cy, d\_r;

private Rectangle drect;

public int sz\_sel;//index of size\_change\_rect captured by mouse

/\*\*

\* clears diagram and data

\*/

public void Clear()

{

data=null; ang=null; color=null; szrc=null;

builded=false; have\_data=false;

repaint();

}

/\*\*

\* getter for data

\*/

public ArrayList GetData(){return data;}

/\*\*

\* setter for data

\*/

public boolean SetData(ArrayList dat)

{

if(dat==null) return false;

if(dat.size()==0) return false;

double v;

data=new ArrayList();

int i;

for(i=0; i<dat.size(); i++)

{

v=((Double)dat.get(i)).doubleValue(); if(v<0) v\*=-1;

data.add(new Double(v));

}

if(color!=null) have\_data=true;

countAngles();

repaint();

return true;

}

/\*\*

\* getter for color

\*/

public ArrayList GetColor(){return color;}

/\*\*

\* setter for color

\*/

public boolean SetColor(ArrayList col)

{

if(col==null) return false;

color=col;

if(data!=null) have\_data=true;

repaint();

return true;

}

/\*\*

\* sets bounds of diagram

\*/

public void setBounds(int x, int y, int width, int height)

{

super.setBounds(x, y, width, height);

d\_r=width; if(d\_r>height) d\_r=height;

d\_r-=CHANGE\_ELLIPSE;

drect=new Rectangle();

drect.width=d\_r; drect.height=d\_r;

drect.x=x+(width-d\_r)/2; drect.y=y+(height-d\_r)/2;

d\_r/=2; d\_cx=drect.x+d\_r; d\_cy=drect.y+d\_r;

countAngles(); repaint();

}

public Dimension getSize()

{

return new Dimension(super.getWidth(), super.getHeight());

}

public void paint(Graphics g)

{

builded=draw(g);

}

public void repaint(Graphics g)

{

builded=draw(g);

}

/\*\*

\* draws diagram

\* @param g Graphics for drawing

\* @return true if success

\*/

public boolean draw(Graphics g)

{

super.paint(g);

if(!have\_data)return false;

if(color.size()!=data.size()) return false;

int i;

if(data.size()>1) szrc=new Rectangle[data.size()];

else szrc=null;

for(i=0; i<data.size(); i++)

{

g.setColor(Color.BLACK);

g.drawArc(drect.x, drect.y, drect.width, drect.height, -ang[i][0], -ang[i][1]);

g.setColor((Color)color.get(i));

g.fillArc(drect.x, drect.y, drect.width, drect.height, -ang[i][0], -ang[i][1]);

g.setColor(Color.WHITE);

g.fillOval(drect.x+d\_r/2, drect.y+d\_r/2, d\_r, d\_r);

}

if(data.size()==1) return true;

g.setColor(new Color(0, 0, 0));

for(i=0; i<data.size(); i++)

{

szrc[i]=new Rectangle();

szrc[i].x=(int)(d\_cx+d\_r\*Math.cos(Math.PI/180.0\*ang[i][0])-CHANGE\_ELLIPSE/2);

szrc[i].y=(int)(d\_cy+d\_r\*Math.sin(Math.PI/180.0\*ang[i][0])-CHANGE\_ELLIPSE/2);

szrc[i].height=szrc[i].width=CHANGE\_ELLIPSE;

g.fillOval(szrc[i].x, szrc[i].y, szrc[i].width, szrc[i].height);

}

return true;

}

/\*\*

\* counts data according to angles on diagram

\*/

public void countData()

{

if(!have\_data) return;

if(ang==null) return;

if(data.size()==0) return;

int za=radTodeg(Math.asin(((double)CHANGE\_ELLIPSE)/(2\*d\_r)));

double v;

double tot=0;

int i;

for(i=0; i<data.size(); i++) tot+=((Double)data.get(i)).doubleValue();

if(tot==0) tot=100;

tot/=(360-za\*data.size());

for(i=0; i<data.size(); i++)

{

data.remove(i); data.add(i, new Double((ang[i][1]-za)\*tot));

}

if(mf!=null && CurrentTable.getTable()!=null)

{

CurrentTable.getTable().setDiagVector(data);

mf.updateTable();

}

}

/\*\*

\* counts default angles

\* @return true if success

\*/

private boolean countAngles()

{

if(data==null) return false;

if(data.size()==0) return false;

int za=radTodeg(Math.asin(((double)CHANGE\_ELLIPSE)/(2\*d\_r)));

int zn=0;

double tot;

int i, arc;

ang=new int[data.size()][];

tot=0;

for(i=0; i<data.size(); i++)

{

ang[i]=new int[2];

if(((Double)data.get(i)).doubleValue()==0) zn++;

tot+=((Double)data.get(i)).doubleValue();

}

if(zn==data.size())

{

tot=360/zn; arc=0;

for(i=0; i<data.size(); i++)

{

ang[i][0]=arc;

arc+=ang[i][1]=(int)tot;

}

}

else

{

tot=(360-data.size()\*za)/tot; arc=0;

for(i=0; i<data.size(); i++)

{

ang[i][0]=arc;

arc+=ang[i][1]=(int)(((Double)data.get(i)).doubleValue()\*tot)+za;

}

}

arc=360;

for(i=0; i<data.size(); i++) arc-=ang[i][1];

ang[data.size()-1][1]+=arc;

repaint();

return true;

}

/\*\*

\* converts angles in rad to deg

\* @param rad

\* @return deg value

\*/

public int radTodeg(double rad){return (int)(180.0/Math.PI\*rad);}

}

/\*\*

\* Listener of a mouse

\*/

class DrMsEv implements MouseListener

{

/\*\*

\* handle to diagram

\*/

private DiagramDrawer dd;

/\*\*

\* constructor

\* @param ddraw handle to diagram

\*/

public DrMsEv(DiagramDrawer ddraw){dd=ddraw;}

/\*\*

\* is called when left button of the mouse is pressed

\*/

public void mousePressed(MouseEvent e)

{

if(!dd.have\_data) return;

if(dd.data.size()!=dd.color.size()) return;

if(dd.szrc==null) return;

int x=e.getX(), y=e.getY();

int i;

if((e.getModifiers() & e.BUTTON1\_MASK)!=0)

{

for(i=0; i<dd.data.size(); i++)

if(dd.szrc[i].contains(x, y)){dd.sz\_sel=i; break;}

return;

}

}

/\*\*

\* is called when left button of the mouse is released

\*/

public void mouseReleased(MouseEvent e)

{

if(!dd.have\_data) return;

if(dd.data.size()!=dd.color.size()) return;

if(dd.szrc==null) return;

dd.sz\_sel=-1;

return;

}

public void mouseClicked(MouseEvent e){}

public void mouseEntered(MouseEvent e){}

public void mouseExited(MouseEvent e){}

}

/\*\*

\* Listener of mouse

\*/

class DrMsMv implements MouseMotionListener

{

/\*\*

\* handle to diagram

\*/

private DiagramDrawer dd;

/\*\*

\* constructor

\* @param ddraw handle to diagram

\*/

public DrMsMv(DiagramDrawer ddraw){dd=ddraw;}

public void mouseMoved(MouseEvent e){}

/\*\*

\* called when mouse is dragged

\*/

public void mouseDragged(MouseEvent e)

{

if(!dd.have\_data) return;

if(dd.data.size()!=dd.color.size()) return;

if(dd.szrc==null) return;

double a;

int za=dd.radTodeg(Math.asin(((double)dd.CHANGE\_ELLIPSE)/(2\*dd.d\_r)));

int i, j, da;

if(dd.sz\_sel!=-1 && (e.getModifiers() & e.BUTTON1\_MASK)!=0)

{

if(dd.d\_cx-e.getX()==0)

{

if(e.getY()>dd.d\_cy) a=Math.PI;

else a=3\*Math.PI/2.0;

}

else

{

a=Math.atan(((double)(e.getY()-dd.d\_cy))/((double)(e.getX()-dd.d\_cx)));

if(e.getX()-dd.d\_cx<0) a=Math.PI+a;

else if(e.getY()-dd.d\_cy<0) a=2\*Math.PI+a;

}

a=180.0/Math.PI\*a;

i=dd.sz\_sel; i--; if(i<0) i+=dd.data.size();

da=dd.ang[dd.sz\_sel][0];

if(da>270 && a<90) da=-(360-da+(int)a);

else

if(da<90 && a>270) da=(360-(int)a+da);

else da=da-(int)a;

if(dd.ang[i][1]-da>=za && dd.ang[dd.sz\_sel][1]+da>=za && da!=0)

{

dd.ang[dd.sz\_sel][0]=(int)a; dd.ang[dd.sz\_sel][1]+=da; dd.ang[i][1]-=da;

a=0;

for(j=0; j<dd.data.size(); j++) a+=((Double)dd.data.get(j)).doubleValue();

if(a==0)

{

dd.ang[i][1]=dd.ang[dd.sz\_sel][1]=(360-za\*(dd.data.size()-2))/2;

dd.ang[i][0]=dd.ang[dd.sz\_sel][0]-dd.ang[i][1];

if(dd.ang[i][0]<0) dd.ang[i][0]+=dd.radTodeg(2\*Math.PI);

da=dd.ang[dd.sz\_sel][0]+dd.ang[dd.sz\_sel][1];

if(da>=dd.radTodeg(2\*Math.PI)) da-=dd.radTodeg(2\*Math.PI);

for(j=0; j<dd.data.size(); j++)

{

if(j!=dd.sz\_sel && j!=i)

{

dd.ang[j][0]=da; da+=dd.ang[j][1]=dd.CHANGE\_ELLIPSE;

if(da>=dd.radTodeg(2\*Math.PI)) da-=dd.radTodeg(2\*Math.PI);

}

}

}

dd.countData(); dd.repaint();

}

}

}

}

package DiagCore;

import java.awt.Color;

import java.util.ArrayList;

/\*\*

\* class dedicated to store information about current table

\*/

public class CurrentTable

{

public String file;

public String gfile;

public boolean text\_format;

public int rows, cols, diag\_ind;

public ArrayList diag\_col;

public ArrayList data;

public ArrayList col\_head;

public ArrayList row\_head;

private static CurrentTable cur\_tab;

/\*\*

\* constructor

\*/

private CurrentTable(){}

/\*\*

\* creates table

\* @param r number of rows

\* @param c number of columns

\* @param col\_hd defines if headers are present

\* @param row\_hd defines if rows are named

\*/

public static void createTable(int r, int c, boolean col\_hd, boolean row\_hd)

{

cur\_tab=new CurrentTable();

cur\_tab.file=null;

ArrayList n;

cur\_tab.data=new ArrayList(); cur\_tab.rows=r; cur\_tab.cols=c;

cur\_tab.diag\_ind=-1;

int i, j;

if(col\_hd)

{

cur\_tab.col\_head=new ArrayList();

for(i=0; i<cur\_tab.cols; i++) cur\_tab.col\_head.add(new String(""));

}

else cur\_tab.col\_head=null;

if(row\_hd)

{

cur\_tab.row\_head=new ArrayList();

for(i=0; i<cur\_tab.rows; i++) cur\_tab.row\_head.add(new String(""));

}

else cur\_tab.row\_head=null;

for(i=0; i<cur\_tab.rows; i++)

{

n=new ArrayList();

for(j=0; j<cur\_tab.cols; j++) n.add(new Double(0));

cur\_tab.data.add(n);

}

if(SettingsContainer.getSettings().col\_for\_diag)

cur\_tab.diag\_col=SettingsContainer.getDefaultDiagramColors(CurrentTable.getTable().rows);

else

cur\_tab.diag\_col=SettingsContainer.getDefaultDiagramColors(CurrentTable.getTable().cols);

}

/\*\*

\* creates table

\* @param col\_hd are headers present?

\* @param row\_hd are rows present?

\*/

public static void createTable(boolean col\_hd, boolean row\_hd)

{

createTable(1, 1, col\_hd, row\_hd);

}

/\*\*

\* deletes table

\*/

public static void deleteTable(){cur\_tab=null;}

/\*\*

\* @return current table

\*/

public static CurrentTable getTable(){return cur\_tab;}

/\*\*

\* adds row

\*/

public void addRow()

{

ArrayList n=new ArrayList();

int i;

if(row\_head!=null) row\_head.add(new String("new"));

for(i=0; i<cols; i++) n.add(new Double(0));

data.add(n); rows++;

}

/\*\*

\* deletes column

\*/

public void addColumn()

{

ArrayList n;

int i;

if(col\_head!=null) col\_head.add(new String("new"));

for(i=0; i<rows; i++)

{

n=(ArrayList)data.get(i);

n.add(new Double(0));

}

cols++;

}

/\*\*

\* removes row

\*/

public boolean removeRow(int r)

{

if(r<0 || r>=rows) return false;

data.remove(r);

if(row\_head!=null) row\_head.remove(r);

rows--; return true;

}

/\*\*

\* removes column c

\*/

public boolean removeColumn(int c)

{

if(c<0 || c>=cols) return false;

ArrayList n;

int i;

for(i=0; i<rows; i++)

{

n=(ArrayList)data.get(i);

n.remove(c);

}

if(col\_head!=null) col\_head.remove(c);

cols--; return true;

}

/\*\*

\* removes color ind which is used in diagram

\*/

public boolean removeDiagColor(int ind)

{

if(ind<0 || ind>=diag\_col.size()) return false;

diag\_col.remove(ind); return true;

}

/\*\*

\* adds color col which is used in diagram

\*/

public void addDiagColor(Color col){diag\_col.add(col);}

/\*\*

\* @return vector of data for building diagram

\*/

public ArrayList getDiagVector()

{

ArrayList res, n;

int i;

if(diag\_ind<0 || (SettingsContainer.getSettings().col\_for\_diag && diag\_ind>=cols)

|| (!(SettingsContainer.getSettings().col\_for\_diag) && diag\_ind>=rows))

return null;

if(SettingsContainer.getSettings().col\_for\_diag)

{

res=new ArrayList();

for(i=0; i<rows; i++)

{

n=(ArrayList)data.get(i);

res.add(n.get(diag\_ind));

}

}

else

res=(ArrayList)data.get(diag\_ind);

return res;

}

/\*\*

\* sets data gotten from diagram

\*/

public boolean setDiagVector(ArrayList dat)

{

if(diag\_ind<0 || (SettingsContainer.getSettings().col\_for\_diag && rows!=dat.size())

|| (!SettingsContainer.getSettings().col\_for\_diag && cols!=dat.size()))

return false;

int i;

ArrayList n;

if(SettingsContainer.getSettings().col\_for\_diag)

{

for(i=0; i<rows; i++)

{

n=(ArrayList)data.get(i);

n.remove(diag\_ind); n.add(diag\_ind, dat.get(i));

}

}

else

{

data.remove(diag\_ind); data.add(diag\_ind, dat);

}

return true;

}

/\*\*

\* getter for data

\*/

public double getData(int r, int c)

{

ArrayList n;

if(r<0 || c<0 || r>=rows || c>=cols || data==null) return Double.NaN;

n=(ArrayList)data.get(r);

return (Double)n.get(c);

}

/\*\*

\* setter for data

\*/

public boolean setData(int r, int c, double val)

{

ArrayList n;

if(r<0 || c<0 || r>=rows || c>=cols || data==null) return false;

n=(ArrayList)data.get(r);

n.remove(c); n.add(c, new Double(val));

return true;

}

/\*\*

\* getter for header

\*/

public String getHeader(int i, boolean row)

{

if(i<0 || (!row && i>=cols) || (row && i>=rows)) return null;

if(row) return new String((String)row\_head.get(i));

else return new String((String)col\_head.get(i));

}

/\*\*

\* setter for header

\*/

public boolean setHeader(int i, boolean row, String val)

{

if(i<0 || (!row && i>=cols) || (row && i>=rows)) return false;

if(row) row\_head.add(i, val);

else col\_head.add(i, val);

return true;

}

/\*\*

\* @return table as String [][]

\*/

public String [][] getInTextFormat()

{

int c, r;

int i, j, k;

c=cols; if(row\_head!=null) c++;

r=rows; if(col\_head!=null) r++;

String [][] str\_dat=new String[r][];

if(col\_head!=null)

{

str\_dat[0]=new String[c];

if(row\_head!=null){str\_dat[0][0]=new String(""); j=1;}

else j=0;

for(i=0; i<cols; i++, j++)

{

str\_dat[0][j]=new String(getHeader(i, false));

}

i=1;

}

else i=0;

for(r=0; r<rows; r++, i++)

{

str\_dat[i]=new String[c];

if(row\_head!=null)

{

str\_dat[i][0]=new String(getHeader(r, true));

j=1;

}

else j=0;

for(k=0; k<cols; k++, j++)

str\_dat[i][j]=new String(Double.toString(getData(r, k)));

}

return str\_dat;

}

}

package DiagCore;

import java.util.\*;

import java.io.\*;

import javax.swing.JOptionPane;

public class CSVProcessor

{

public CSVProcessor()

{

dat\_lst=null; col\_del=" ";

}

/\*\*delimiter of colomn\*/

private String col\_del;

/\*\*sets delimiter of the colomn as del\*/

public void SetDelimiter(String del)

{

if(del!=null) col\_del=new String(del);

}

/\*\*returns delimiter of the colomn\*/

public String GetDelimeter(){return new String(col\_del);}

/\*\*array of rows\*/

ArrayList dat\_lst;

/\*\*loading data. path-contains path to file\*/

public boolean LoadTextData(String path)

{

File in; in=new File(path);

if(!in.exists())

{

JOptionPane.showConfirmDialog(null, new String("File with table does not exist."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

if(!in.canRead())

{

JOptionPane.showConfirmDialog(null, new String("File with table can not be read."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

BufferedReader s; String ln;

try{s=new BufferedReader(new FileReader(path));}

catch(Exception e1)

{

JOptionPane.showConfirmDialog(null, new String("Error while reading file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

while(true)

{

try{ln=s.readLine();}

catch(Exception e2)

{

try{s.close();}catch(Exception e4){}

JOptionPane.showConfirmDialog(null, new String("Error while reading file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

if(ln==null) break;

if(dat\_lst==null){dat\_lst=new ArrayList();}

dat\_lst.add(ln);

}

try{s.close();}catch(Exception e3){}

return true;

}

/\*\*saving data to file. path-contains path to file\*/

public boolean SaveTextData(String path)

{

if(dat\_lst.isEmpty())

{

JOptionPane.showConfirmDialog(null, new String("No data for saving."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

File out; out=new File(path);

if(!out.exists())

{

try{out.createNewFile();}catch(Exception e1)

{

JOptionPane.showConfirmDialog(null, new String("Error while creating file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

}

String ln;

PrintWriter d;

try{d=new PrintWriter(new OutputStreamWriter(new FileOutputStream(path)));}

catch(Exception e2)

{

JOptionPane.showConfirmDialog(null, new String("Error while writing to file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

for(int i=0; i<dat\_lst.size(); i++)

{

ln=(String)dat\_lst.get(i); d.write(ln); d.println();

}

d.close();

return true;

}

/\*\*serialize data. path-contains path to file\*/

public boolean SaveBinData(String path)

{

File out=new File(path);

if(!out.exists())

{

try{out.createNewFile();}catch(Exception e1)

{

JOptionPane.showConfirmDialog(null, new String("Error while creating file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

}

try

{

ObjectOutputStream d=new ObjectOutputStream(new FileOutputStream(path));

d.writeObject(dat\_lst);

d.close();

}

catch(Exception e1)

{

JOptionPane.showConfirmDialog(null, new String("Error while writing to file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

return true;

}

/\*\*deserialize data. path-contains path to file\*/

public boolean LoadBinData(String path)

{

File in=new File(path);

if(!(in.exists() && in.isFile()))

{

JOptionPane.showConfirmDialog(null, new String("File does not exist."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

try

{

ObjectInputStream s=new ObjectInputStream(new FileInputStream(path));

dat\_lst=(ArrayList)s.readObject();

s.close();

}

catch(Exception e1)

{

JOptionPane.showConfirmDialog(null, new String("Error while reading from file."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

return false;

}

return true;

}

/\*\*returns string with index ind. If there is no such string will be returned null\*/

public String GetTextData(int ind)

{

if(ind<0 || ind>=dat\_lst.size()) return null;

return (String)dat\_lst.get(ind);

}

/\*\*returns number of strings in the list\*/

public int GetTextDataSize(){return dat\_lst.size();}

/\*\*returns array of the strings\*/

public String [][] ParseTextData()

{

state=0;

if(dat\_lst==null) return null;

if(dat\_lst.size()==0) return null;

StringTokenizer st;

ArrayList res, row;

String t;

boolean er\_oc=false;

String [][] ress; ress=null;

int cn=-1, i, ccn, j;

res=new ArrayList();

for(i=0; i<dat\_lst.size(); i++)

{

st=new StringTokenizer((String)dat\_lst.get(i), col\_del);

row=new ArrayList(); ccn=0;

while(st.hasMoreTokens())

{

t=st.nextToken();

row.add(t); ccn++;

}

if(cn==-1) cn=ccn;

else

if(cn!=ccn)

{

er\_oc=true;

}

res.add(row);

}

if(er\_oc)

{

state=-1;

JOptionPane.showConfirmDialog(null, new String("Different numbers of columns in rows."),

"Error", JOptionPane.OK\_OPTION, JOptionPane.ERROR\_MESSAGE);

}

if(res.size()>0)

{

ress=new String[res.size()][];

for(i=0; i<res.size(); i++)

{

row=(ArrayList)res.get(i); ress[i]=new String[row.size()];

for(j=0; j<row.size(); j++) ress[i][j]=(String)row.get(j);

}

}

state=cn;

return ress;

}

public void BuildTextData(String [][] dat)

{

int i, j;

String row;

if(dat==null) return;

dat\_lst=new ArrayList();

for(i=0; i<dat.length; i++)

{

row=new String("");

for(j=0; j<dat[i].length; j++)

{

row+=dat[i][j];

if(j+1!=dat[i].length) row+=col\_del;

}

dat\_lst.add(row);

}

}

public int state;

}

package DiagCore;

import java.awt.event.\*;

import javax.swing.\*;

public class ChangeColorListener implements ActionListener

{

public JButton bt;

public boolean change;

public JColorChooser col\_ch;

public ChangeColorListener(JButton but){bt=but; change=false;}

public void actionPerformed(ActionEvent arg0)

{

col\_ch= new JColorChooser();

ActionListener c\_ok, c\_can;

c\_ok=new ColChOk(this); c\_can=new ColChCan();

JDialog col\_dlg=JColorChooser.createDialog(bt, "Pick a Color", true,

col\_ch, c\_ok, c\_can);

col\_ch.setColor(bt.getBackground());

col\_dlg.setVisible(true);

}

}

class ColChOk implements ActionListener

{

private ChangeColorListener cl;

public ColChOk(ChangeColorListener cl){this.cl=cl;}

public void actionPerformed(ActionEvent arg0)

{

cl.bt.setBackground(cl.col\_ch.getColor());

cl.change=true;

}

}

class ColChCan implements ActionListener

{

public void actionPerformed(ActionEvent arg0)

{

}

}

package DiagCore;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;

import javax.swing.JFileChooser;

import javax.swing.JTextField;

public class BrowseListener implements ActionListener

{

private JTextField path;

public BrowseListener(JTextField path)

{

this.path=path;

}

public void actionPerformed(ActionEvent arg0)

{

JFileChooser fc=new JFileChooser("Select file");

fc.showOpenDialog(null);

File f=fc.getSelectedFile();

if(f==null) path.setText("");

else path.setText(f.getPath());

}

}