**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Н2ЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Заместитель руководителя департамента  Программной Инженерии  Факультета Компьютерных Наук  профессор  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.М. Гринкруг  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** | **RU.17701729.02.03-81 01-1** | | **ИССЛЕДОВАНИЕ И РЕАЛИЗАЦИЯ**  **СЕРИАЛАЙЗЕРА-ДЕСЕРИАЛАЙЗЕРА**  **АЦИКЛИЧЕСКОГО ГРАФА ИЗ**  **ИНСТАНСОВ JAVA BEANS COMPONENT**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.02.03-81 01-1-ЛУ** | | |
|  |  | |
| Исполнитель:  студент группы БПИ171  \_\_\_\_\_\_\_\_\_\_\_\_ /Данилов Алексей Андреевич /  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. | |
|  | | |
|  | |  |

**2019**

|  |  |  |  |
| --- | --- | --- | --- |
| **УТВЕРЖДЕНО**  **RU.17701729.02.03-81 01-1-ЛУ** | |  |  |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** | **RU.17701729.02.03-81 01-1** | | **ИССЛЕДОВАНИЕ И РЕАЛИЗАЦИЯ**  **СЕРИАЛАЙЗЕРА-ДЕСЕРИАЛАЙЗЕРА**  **АЦИКЛИЧЕСКОГО ГРАФА ИЗ**  **ИНСТАНСОВ JAVA BEANS COMPONENT**  **Пояснительная записка**  **RU.17701729.02.03-81 01-1**  **Листов 31** | | | | |
|  |  | | | |
|  | | | |
| **2019** | | | | |

СОДЕРЖАНИЕ

[**1.** **ВВЕДЕНИЕ** 4](#_Toc8685805)

[**1.1.** **Наименование программы** 4](#_Toc8685806)

[**1.2.** **Документы, на основании которых ведется разработка** 5](#_Toc8685807)

[**2.** **НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ** 5](#_Toc8685808)

[**2.1.** **Назначение программы** 5](#_Toc8685809)

[**2.1.1.** **Функциональное назначение** 6](#_Toc8685810)

[**2.1.2.** **Эксплуатационное назначение** 6](#_Toc8685811)

[**2.2.** **Краткая характеристика области применения** 6](#_Toc8685812)

[**3.** **ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ** 7](#_Toc8685813)

[**3.1.** **Постановка задачи на разработку программы** 7](#_Toc8685814)

[**3.2.** **Описание алгоритмов и функционирования программы** 7](#_Toc8685815)

[**3.2.1.** **Алгоритм шифрования информации с помощью решетки Кардано** 7](#_Toc8685816)

[**3.2.2.** **Алгоритм дешифрования информации с помощью решетки Кардано** 8](#_Toc8685817)

[**3.2.3.** **Алгоритм запуска сервера** 8](#_Toc8685818)

[**3.2.4.** **Алгоритм подключения клиента к серверу** 8](#_Toc8685819)

[**3.2.5.** **Алгоритм сохранения файла в формате .krd** 9](#_Toc8685820)

[**3.2.6.** **Алгоритм открытия файла в формате .krd** 10](#_Toc8685821)

[**3.3.** **Обоснование и описание выбора состава технических и программных средств** 11](#_Toc8685822)

[**3.3.1.** **Состав технических и программных средств** 11](#_Toc8685823)

[**3.3.2.** **Обоснование выбора технических и программных средств** 11](#_Toc8685824)

[**4.** **ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ** 12](#_Toc8685825)

[**4.1.** **Предполагаемая потребность** 12](#_Toc8685826)

[**4.2.** **Преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами** 12](#_Toc8685827)

[**5.** **ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ** 13](#_Toc8685828)

[**ПРИЛОЖЕНИЕ 1** 14](#_Toc8685829)

[**ПРИЛОЖЕНИЕ 3** 15](#_Toc8685830)

[**ПРИЛОЖЕНИЕ 4** 16](#_Toc8685831)

[**ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ** 30](#_Toc8685832)

**АННОТАЦИЯ**

В данном программном документе приведена пояснительная записка для исследования и разработки нового сериалайзера-десериалайзера ациклического графа из инстансов java beans component.

В разделе «Введение» указано наименование программы, краткое наименование программы и документы, на основании которых ведется разработка.

В разделе «Назначение и область применения» указано функциональное назначение программы, эксплуатационное назначение программы и краткая характеристика области применения программы.

В разделе «Технические характеристики» содержатся следующие подразделы: - постановка задачи на разработку программы; - описание алгоритма и функционирования программы с обоснованием выбора схемы алгоритма решения задачи и возможные взаимодействия программы с другими программами; - описание и обоснование выбора метода организации входных и выходных данных; - описание и обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указана предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

Настоящий документ разработан в соответствии с требованиями:

1) ГОСТ 19.101-77 Виды программ и программных документов [1];

2) ГОСТ 19.102-77 Стадии разработки [2];

3) ГОСТ 19.103-77 Обозначения программ и программных документов [3];

4) ГОСТ 19.104-78 Основные надписи [4];

5) ГОСТ 19.105-78 Общие требования к программным документам [5];

6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [6];

7) ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению [7].

Изменения к Пояснительной записке оформляются согласно ГОСТ 19.603-78 [8], ГОСТ 19.604-78 [9].

Перед прочтением данного документа рекомендуется ознакомиться с терминологией, приведенной в Приложении 1 настоящей пояснительной записки.

1. **ВВЕДЕНИЕ**
   1. **Наименование программы**

**Наименование программы:** «Исследование и реализация сериалайзера-десериалайзера ациклического графа из инстансов java beans component».

**Условное обозначение темы разработки:** «Research and Implementation of the Acyclic Graph Serializer-deserializer Using Java Beans Component Instances».

* 1. **Документы, на основании которых ведется разработка**

Программа выполнена в рамках темы курсовой работы «Программа для шифрования и дешифрования с помощью решетки Кардано» в соответствии с учебным планом подготовки бакалавров (НИУ ВШЭ, факультет компьютерных наук) по направлению «Программная инженерия».

Разработка ведется на основании приказа декана ФКН И.В. Аржанцева № 2.3-02/1504-02 от 15.04.19 "Об утверждении тем, руководителей курсовых работ студентов образовательной программы Программная инженерия факультета компьютерных наук".

1. **НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ**
   1. **Назначение программы**
      1. **Функциональное назначение**

Функциональным назначением разработанной библиотеки является сериализация десериализация объектов в формате XML

* + 1. **Эксплуатационное назначение**

Библиотека разработана, чтобы эксплуатироваться программистами, програмными инденерами, а так же студентами, желающими сериализовать объекты своей программы

* 1. **Краткая характеристика области применения**

Библиотека сериализует граф общего вида объекта в сериализованный файл формата XML и десериализует сериализованный файл в граф объекста.

1. **ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ**
   1. **Постановка задачи на разработку программы**

Разрабатываемая библиотека должна:

1. Сериализация ациклического графа;
2. Десериализация его сериализованного представления в файле;
   1. **Описание алгоритмов и функционирования программы**

В коде программы можно заметить некрасивый код: для каждого примитива проверяется его тип через if, написать по-другому невозможно из-за недостаточности функционала Java.

* + 1. **Алгоритм сериализации**

Алгоритм сериализации устроен следующим образом:

На вход подается объект который следует сериализовать, FileOutputStream и тип сериализации.

Метод make вызывает метод nonPrimitiveToString который пишет в FileOutputStream объект в виде XML.

Метод nonPrimitiveToString проверяет элемент на соответствие типа и существует несколько вариантов:

* В объекте присутствует примитив
* В объекте присутствует объект другого типа
* В объекте присутствует объект типа Iterable
* В объекте присутствует массив
* В объекте присутствует объект который мы ранее сериализовали

В первом случае вызывается соответствующий метод primitiveToString, который на основе типа сериализации записывает примитив в необходимом синтаксисе

objectList.add(o);

idList.add(objectType.getSimpleName() + "" + idList.size());

\_serializingSyntacse.nonPrimitiveBegin(objectType.getName(), name, output,

objectType.getSimpleName() + "" + (idList.size() - 1), tabs, -1,o==\_object);

BeanInfo beanInfo = Introspector.getBeanInfo(o.getClass(),Object.class);

PropertyDescriptor[] propertyDescriptors = beanInfo.getPropertyDescriptors();

for (PropertyDescriptor pd : propertyDescriptors) {

if (pd.getReadMethod() != null) {

if (pd.getPropertyType().isPrimitive() || pd.getPropertyType().isAssignableFrom(String.class))

primitiveToString( output, pd.getName(), pd.getReadMethod().invoke(o), tabs + 1, -1);

else if (pd.getReadMethod().invoke(o) != null)

nonPrimitiveToString(output, pd.getName(), pd.getReadMethod().invoke(o), tabs + 1, -1);

}

}

\_serializingSyntacse.nonPrimitiveEnd(objectType.getName(), name, output, tabs, index,o==\_object);

Во втором случае происходит рекурсивный вызов метода nonPrimitiveToString

В третьем случае для каждого элемента, вызывается рекурсивный nonPrimitiveToString

if (o instanceof Iterable) {

objectList.add(o);

idList.add(objectType.getSimpleName() + "" + idList.size());

\_serializingSyntacse.itarableBegin(objectType.getName(), name, output, objectType.getSimpleName() + "" + (idList.size() - 1), tabs, index,o==\_object);

for (Object value : ((Iterable) o)) {

nonPrimitiveToString(output, "", value, tabs + 1, index);

}

\_serializingSyntacse.iterableEnd(objectType.getName(), name, output, tabs, index,o==\_object);

В четвертом случае алгоритм распознает какого типа является массив, чтобы вывести каждый элемент с соответствующим типом

if (objectType == byte[].class) {

\_serializingSyntacse.arrayBegin(byte.class.getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((byte[]) o).length,o==\_object);

for (int i = 0; i < ((byte[]) o).length; i++)

primitiveToString(output, "", ((byte[]) o)[i], tabs + 1, i);

} else if (objectType == short[].class) {

\_serializingSyntacse.arrayBegin(short.class.getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((short[]) o).length,o==\_object);

for (int i = 0; i < ((short[]) o).length; i++)

primitiveToString(output, "", ((short[]) o)[i], tabs + 1, i);

} else if (objectType == int[].class) {

\_serializingSyntacse.arrayBegin(int.class.getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((int[]) o).length,o==\_object);

for (int i = 0; i < ((int[]) o).length; i++)

primitiveToString(output, "", ((int[]) o)[i], tabs + 1, i);

} else if (objectType == long[].class) {

\_serializingSyntacse.arrayBegin(long.class.getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((long[]) o).length,o==\_object);

for (int i = 0; i < ((long[]) o).length; i++)

primitiveToString(output, "", ((long[]) o)[i], tabs + 1, i);

} else if (objectType == char[].class) {

\_serializingSyntacse.arrayBegin(char.class.getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((char[]) o).length,o==\_object);

for (int i = 0; i < ((char[]) o).length; i++)

primitiveToString(output, "", ((char[]) o)[i], tabs + 1, i);

} else if (objectType == float[].class) {

\_serializingSyntacse.arrayBegin(float.class.getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((float[]) o).length,o==\_object);

for (int i = 0; i < ((float[]) o).length; i++)

primitiveToString(output, "", ((float[]) o)[i], tabs + 1, i);

} else if (objectType == double[].class) {

\_serializingSyntacse.arrayBegin(double.class.getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((double[]) o).length,o==\_object);

for (int i = 0; i < ((double[]) o).length; i++)

primitiveToString(output, "", ((double[]) o)[i], tabs + 1, i);

} else if (objectType == boolean[].class) {

\_serializingSyntacse.arrayBegin(boolean.class.getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((boolean[]) o).length,o==\_object);

for (int i = 0; i < ((boolean[]) o).length; i++)

primitiveToString(output, "", ((boolean[]) o)[i], tabs + 1, i);

} else if (objectType == String[].class) {

\_serializingSyntacse.arrayBegin(String.class.getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((String[]) o).length,o==\_object);

for (int i = 0; i < ((String[]) o).length; i++)

primitiveToString(output, "", ((String[]) o)[i], tabs + 1, i);

} else {

\_serializingSyntacse.arrayBegin(((Object[])o)[0].getClass().getName(), name, output, objectType.getSimpleName() + "" + idList.size(), tabs, index, ((Object[]) o).length,o==\_object);

for (int i = 0; i < ((Object[]) o).length; i++)

nonPrimitiveToString(output, "", ((Object[]) o)[i], tabs + 1, i);

}

В пятом случае вызвается соответствующий метод для записи элементов которые сериализовались раннее в этом графе.

for (int i = 0; i < objectList.size(); i++) {

if (objectList.get(i) == o) {

\_serializingSyntacse.reference(o.getClass().getSimpleName(), name, idList.get(i), output, tabs, index);

return;

}

}

Таким образом, проходя по графу по алгоритму DFS можно записать объект в виде XML

* + 1. **Алгоритм десериализации**

Алгоритм десериализации является обратным алгоритму сериализации: вместо прохождения по объекту, алгоритм проходит по XML документуи существуют те же возможные варианты:

* Встречен примитив
* Встречен Объект
* Встречен массив
* Встречен Iterable тип
* Встречен объект который уже десериализовали

В первом случае парсится строка в соответствии с типом элемента

Object o = dfs.pop();

if (o.getClass() == Integer.class) {

dfs.push(Integer.parseInt(str));

check = false;

} else if (o.getClass() == Long.class) {

dfs.push(Long.parseLong(str));

check = false;

} else if (o.getClass() == Byte.class) {

dfs.push(Byte.parseByte(str));

check = false;

} else if (o.getClass() == Short.class) {

dfs.push(Short.parseShort(str));

check = false;

} else if (o.getClass() == Character.class) {

dfs.push(str.charAt(0));

check = false;

} else if (o.getClass() == Double.class) {

dfs.push(Double.parseDouble(str));

check = false;

} else if (o.getClass() == Float.class) {

dfs.push(Float.parseFloat(str));

check = false;

} else if (o.getClass() == Boolean.class) {

dfs.push(Boolean.parseBoolean(str));

check = false;

} else if (o.getClass() == String.class) {

dfs.push(str);

check = false;

} else {

dfs.push(o);

check = false;

}

Во втором случае создается инстанс этого элемента который добавляется в стек DFS.

String id = attributes.getValue("id");

try {

Class cl = Class.forName(attributes.getValue("class"));

Object o = cl.newInstance();

objectMap.put(id, o);

dfs.push(o);

}

catch (ClassNotFoundException | IllegalAccessException | InstantiationException e) {

e.printStackTrace();

}

В третьем случае создается инстанс объекта массива изменяется только метод пристваивания

try {

String id = attributes.getValue("id");

int length = Integer.parseInt(attributes.getValue("length"));

String className = attributes.getValue("class");

Class cl;

if ("byte".equals(className)) {

cl = byte.class;

} else if ("int".equals(className)) {

cl = int.class;

} else if ("boolean".equals(className)) {

cl = boolean.class;

} else if ("short".equals(className)) {

cl = short.class;

} else if ("double".equals(className)) {

cl = double.class;

} else if ("float".equals(className)) {

cl = float.class;

} else if ("char".equals(className)) {

cl = char.class;

} else if ("long".equals(className)) {

cl = long.class;

} else if ("string".equals(className)) {

cl = String.class;

} else {

cl = Class.forName(className);

}

Object o = Array.newInstance(cl, length);

dfs.push(o);

objectMap.put(id, o);

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

В четвертом случае так же создается инстанс объекта и изменяется тип добавления элементов коллекции в объект.

В пятом случае родителю присваивается уже ранее десериализованный объект

* + 1. **Алгоритм добавления собсвенного синтаксиса для сериализации**

Программа позволяет пользователям реализовывать свои типы сериализации объекта, создав класс, имплементировать интерфейс SerializerSyntacse и переопределев методы запускать сериалайзер с собственным синтаксисом.

Такое улучшение позволяет реализовывать своисинтакисы сериализации, переоределив необходимые методы

* 1. **Обоснование и описание выбора состава технических и программных средств**
     1. **Состав технических и программных средств**

Так как проектом является библиотека, то технические средства должны обеспечить работу JVM

* + 1. **Обоснование выбора технических и программных средств**

Проект был написан на 100% java со всеми классами которые есть в JDK таким образом ее можно использовать в любом проекте написанном на Java

* 1. **Результаты проведенного исследования**

1. **ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ**

В рамках данной работы расчет экономической эффективности не предусмотрен.

* 1. **Предполагаемая потребность**

Конечными пользователями библиотеки могут быть программисты, программные инженеры, а также студенты, желающие сериализовать объекты своей программы.

* 1. **Преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами**

Данное приложение:

1. распространяется бесплатно;
2. легкое в использовании;
3. возможно легкое добавление собственных синтаксисов;
4. не требует вложения денежных средств во время использования;
5. имеет неограниченный срок службы;
6. имеет JavaDOC.

1. **ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ**
2. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
11. ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды. – М.: Изд-во стандартов, 1997.
12. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
13. Oracle // Java Platform Standard Edition 8 Documentation - https://docs.oracle.com/javase/8/docs/

# **ПРИЛОЖЕНИЕ 1**

**ТЕРМИНОЛОГИЯ**

|  |
| --- |
| Таблица 1 |

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| **Сериализация** | Процесс преобразования структур данных или состояния объекта в формат, который может быть сохранен (например, в файле или буфере памяти) |
| **Десериализация** | Обратный процесс сериализации |
| **XML** | Расширяемый язык разметки. Рекомендован Консорциумом Всемирной паутины (W3C). Спецификация XML описывает XML-документы и частично описывает поведение XML-процессоров (программ, читающих XML-документы и обеспечивающих доступ к их содержимому). |
| **JSON** | Текстовый формат обмена данными, основанный на JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми. |

# **ПРИЛОЖЕНИЕ 2**

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ**

Описание и функциональное назначение классов.

Таблица 2

|  |  |
| --- | --- |
| Класс | Назначение |
| Language | Класс, представляющий библиотеку русского и английского языка. Содержит свойства возвращающие строки. |
| convert | Класс, представляющий метод для перевода размера текста из pt в pixels. |
| ClientObject | Класс подключенного клиента, содержащий необходимые для сервера поля. |
| ServerObject | Класс сервера имеет в себе множество необходимых для корректной работы поля и методы. |
| AboutProgramm | Класс, представляющий окно программы с информацией об этой программе. Наследуется от Window. |
| Client | Класс, представляющий окно-клиент программы. Наследуется от Window. |
| EnteringField | Класс, представляющий окно программы с выбором размера решетки. Наследуется от Window. |
| GrillEditor | Класс, представляющий окно программы для создания решетки. Наследуется от Window. |
| Help | Класс, представляющий окно программы, отображающее техническую помощь. Наследуется от Window. |
| InformationCoder | Класс, представляющий окно программы для шифрования информации. Наследуется от Window. |
| InformationDecoder | Класс, представляющий окно программы для дешифрования информации. Наследуется от Window. |
| MainMenu | Класс, представляющий главное окно программы. Наследуется от Window. |
| MainWindow | Класс, представляющий окно программы для выбора языка. Наследуется от Window. |
| Server | Класс, представляющий окно программы для запуска сервера. Наследуется от Window. |
| ServerAndClientSetttings | Класс, представляющий окно программы для ввода ip и порта. Наследуется от Window. |
| ServerOrClient | Класс, представляющий окно программы для выбоа между клиента и сервера. Наследуется от Window. |

# **ПРИЛОЖЕНИЕ 4**

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ, ПОЛЕЙ,**

**МЕТОДОВ И СВОЙСТВ**

Таблица 3.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание классов, полей, методов и свойств класса Language.cs | | | | |
| Свойства | | | | |
| Имя | Модификатор  доступа | Тип | Доступ | Назначение |
| Rus | public | bool | get | Информация о языке, является ли он русским. |
| MainMenu | public | string | get | Возвращает строку на русском или английском языке. |
| GrillEditor | public | string | get | Возвращает строку на русском или английском языке. |
| InformationCoder | public | string | get | Возвращает строку на русском или английском языке. |
| InformationDecoder | public | string | get | Возвращает строку на русском или английском языке. |
| Chat | public | string | get | Возвращает строку на русском или английском языке. |
| AboutProgramm | public | string | get | Возвращает строку на русском или английском языке. |
| Help | public | string | get | Возвращает строку на русском или английском языке. |
| SizeOfGrill | public | string | get | Возвращает строку на русском или английском языке. |
| Continue | public | string | get | Возвращает строку на русском или английском языке. |
| ErrorMessage | public | string | get | Возвращает строку на русском или английском языке. |
| Error | public | string | get | Возвращает строку на русском или английском языке. |
| Cancle | public | string | get | Возвращает строку на русском или английском языке. |
| Hide | public | string | get | Возвращает строку на русском или английском языке. |
| Show | public | string | get | Возвращает строку на русском или английском языке. |
| Save | public | string | get | Возвращает строку на русском или английском языке. |
| Open | public | string | get | Возвращает строку на русском или английском языке. |
| KardanoGrild | public | string | get | Возвращает строку на русском или английском языке. |
| Warning | public | string | get | Возвращает строку на русском или английском языке. |
| WarningMessage | public | string | get | Возвращает строку на русском или английском языке. |
| ShowGrill | public | string | get | Возвращает строку на русском или английском языке. |
| HideGrill | public | string | get | Возвращает строку на русском или английском языке. |
| Get | public | string | get | Возвращает строку на русском или английском языке. |
| Enter | public | string | get | Возвращает строку на русском или английском языке. |
| Chose | public | string | get | Возвращает строку на русском или английском языке. |
| serverclient | public | string | get | Возвращает строку на русском или английском языке. |
| NetWarning | public | string | get | Возвращает строку на русском или английском языке. |
| Server | public | string | get | Возвращает строку на русском или английском языке. |
| Client | public | string | get | Возвращает строку на русском или английском языке. |
| LogIn | public | string | get | Возвращает строку на русском или английском языке. |
| Send | public | string | get | Возвращает строку на русском или английском языке. |
| Connected | public | string | get | Возвращает строку на русском или английском языке. |
| Start | public | string | get | Возвращает строку на русском или английском языке. |
| ServerEnter | public | string | get | Возвращает строку на русском или английском языке. |
| Server\_leave | public | string | get | Возвращает строку на русском или английском языке. |
| Server\_start | public | string | get | Возвращает строку на русском или английском языке. |
| aboutprogramm | public | string | get | Возвращает строку на русском или английском языке. |
| Delete | public | string | get | Возвращает строку на русском или английском языке. |
| GrillEditorHelp | public | string | get | Возвращает строку на русском или английском языке. |
| InformationCoderHelp | public | string | get | Возвращает строку на русском или английском языке. |
| InformationDecoderHelp | public | string | get | Возвращает строку на русском или английском языке. |
| ChatHelp | public | string | get | Возвращает строку на русском или английском языке. |
| KardanoGridInfo | public | string | get | Возвращает строку на русском или английском языке. |
| CardanoGrill | public | string | get | Возвращает строку на русском или английском языке. |
| Методы | | | | |
| Имя | Модификатор  доступа | Тип | Аргументы | Назначение |
| Language | public | конструктор | bool Rus | Конструктор класса Language. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание классов, полей методов и свойств класса convert.cs  Таблица 3.2 | | | | |
| Методы | | | | |
| Имя | Модификатор  доступа | Тип | Аргументы | Назначение |
| doit | public | int | int px | Возвращает конвертированное число из pt в px. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание классов, полей методов и свойств класса ServerObject находящегося в Server.xml.cs  Таблица 3.3 | | | | |
| Поля | | | | |
| Имя | Модификатор  доступа | Тип | Назначение | |
| tcpListener | private | TcpListener | Сервер для прослушивания. | |
| clients | public | List<Client  Object> | Лист клиентов. | |
| Методы | | | | |
| Имя | Модификатор  доступа | Тип | Аргументы | Назначение |
| AddConnection | public | void | ClientObject  clientObject | Добавление клиента. |
| RemoveConnection | public | void | string id | Удаление клиента. |
| Listen | public | void | - | Прослушивание входящих подключений. |
| BroadcastMessage | public | void | string message, string id | Трансляция сообщений подключенным клиентам. |
| Disconnect | public | void | - | Отключение всех клиентов и закрытие сервера. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание классов, полей, методов и свойств класса ClientObject находящегося в Server.xml.cs  Таблица 3.4 | | | | |
| Поля | | | | |
| Имя | Модификатор  доступа | Тип | Назначение | |
| userName | private | string | Никнейм клиента. | |
| client | private | TcpClient | Клиент. | |
| server | private | ServerObject | объект сервера. | |
| Методы | | | | |
| Имя | Модификатор  доступа | Тип | Аргументы | Назначение |
| ClientObject | public | конструктор | TcpClient tcpClient, ServerObject serverObject | Конструктор класса ClientObject. |
| Process | public | void | - | Основной процесс клиента. |
| GetMessage | private | string | - | Метод для получения сообщений от сервера. |
| Close | public | void | - | Закрытие подключения |
| Свойства | | | | |
| Имя | Модификатор  доступа | Тип | Доступ | Назначение |
| Id | public | string | get | Возвращает уникальный Id клиента. |
| Id | private | string | set | Задает уникальный Id клиента. |
| Stream | public | Network  Stream | get | Возвращает поток данных клиента. |
| Stream | private | Network  Stream | set | Задает поток данных клиента. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Описание классов, полей, методов и свойств класса AboutProgramm.xaml.cs  Таблица 3.5 | | | | | | | |
| Поля | | | | | | | | |
| Имя | | Модификатор  доступа | | Тип | | Назначение | | |
| language | | private | | Language | | Язык окна | | |
| Методы | | | | | | | |
| Имя | Модификатор  доступа | | Тип | | Аргументы | | Назначение |
| About  Programm | public | | конструктор | | Language l | | Конструктор окна AboutProgramm. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание классов, полей, методов и свойств класса Client.xaml.cs  Таблица 3.6 | | | | |
| Поля | | | | | |
| Имя | Модификатор  доступа | Тип | Назначение | | |
| CodedGrid | private | string[] | Массив, из файла .krd. | | |
| H | private | uint | Высота решетки. | | |
| W | private | uint | Ширина решетки. | | |
| \_context | private | SynchronizationContext | Обеспечивает базовую функциональность для распространения контекста синхронизации в различных моделях синхронизации. | | |
| userName | private | string | Никнейм клиента. | | |
| SERVERADRESS | private | IPEndPoint | IP и порт сервера. | | |
| client | private | TcpClient | Клиент. | | |
| stream | private | NetworkStream | Поток данных. | | |
| lng | private | Language | Язык окна. | | |
| BBoxes | private | Button[,] | Массив представляющий собой решетку Кардано. | | |
| TBoxes | private | TextBox[,] | Массив ячеек для записи текста. | | |
| Методы | | | | | |
| Имя | Модификатор  доступа | Тип | Аргументы | Назначение | |
| Client | public | конструктор | uint h,  uint w,  Language language,  IPEndPoint iPEnd | Конструктор окна Client. | |
| GetTextButtonClick | private | void | object sender, RoutedEventArgs e | Метод обрабатывающий событие при нажатии кнопки, который вводит текст в ячейки. | |
| ReciveMessage | private | void | - | Метод для принятия сообщений от клиента. | |
| Connect | private | void | object sender, RoutedEventArgs e | Метод для подключения к серверу. | |
| SendMessage\_Click | private | void | object sender, RoutedEventArgs e | Метод обрабатывающий событие на нажатие на кнопку, который отправляет сообщение на сервер. | |
| Message\_  KeyDown | private | void | object sender, KeyEventArgs e | Метод обрабатывающий событие на нажатие Enter, который отправляет сообщение на сервер | |
| Window\_  SizeChanged | private | void | object sender, SizeChangedEventArgs e | Метод обрабатывающий событие при изменении размера окна. | |
| Close\_  MouseEnter | private | void | object sender, MouseEventArgs e | Метод обрабатывающий событие при наведении на крестик. | |
| Close\_  MouseLeave | private | void | object sender, MouseEventArgs e | Метод обрабатывающий событие при выходе из области крестика. | |
| Close\_  MouseDown | private | void | object sender, MouseButtonEventArgs e | Метод обрабатывающий событие при нажатии на крестик. | |
| Hide\_  MouseEnter | private | void | object sender, MouseEventArgs e | Метод обрабатывающий событие при наведении на Hide pannel. | |
| Hide\_  MouseLeave | private | void | object sender, MouseEventArgs e | Метод обрабатывающий событие при выходе из области Hide pannel. | |
| Hide\_  MouseDown | private | void | object sender, MouseButtonEventArgs e | Метод обрабатывающий событие при нажатии на Hide pannel. | |
| Open | public | void | object sender, EventArgs e | Метод открывающий решетку Кардано. | |
| SH | public | void | object sender, EventArgs e | Метод скрывающий и показывающий решетку. | |
| EnterEv | private | void | object sender, KeyEventArgs e | Метод обрабатывающий событие при нажатии на Enter, который вводит текст в ячейки. | |
| DELETEALL\_  Click | private | void | object sender, RoutedEventArgs e | Метод обрабатывающий событие при нажатии на кнопку, который очищает все ячейки от текста. | |
| Getcode | private | void | object sender, EventArgs e | Метод обрабатывающий событие при нажатии на кнопку, который выводит зашифрованный текст. | |
| ku | private | void | object sender, TextChangedEventArgs e | Метод упрощающий способ печати в ячейки. | |
| del | private | void | object sender, KeyEventArgs e | Метод упрощающий способ печати в ячейки. | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание классов, полей, методов и свойств класса EnteringField.xaml.cs  Таблица 3.7 | | | | |
| Поля | | | | | |
| Имя | Модификатор  доступа | Тип | Назначение | | |
| L | private | Language | Язык окна. | | |
| TypeOf  Window | private | char | Тип окна для открытия. | | |
| Методы | | | | | |
| Имя | Модификатор  доступа | Тип | Аргументы | Назначение | |
| EnteringField | public | конструктор | Language language,  char symbol | Конструктор окна EnteringField. | |
| Continue  Button\_Click | private | void | object sender, RoutedEventArgs e | Метод обрабатывающий событие при нажатии кнопки, который вызывает метод для создания окна. | |
| CreateNew  Window | private | void | - | Метод для открытия нового окна. | |
| Wigth\_  KeyDown | private | void | object sender, KeyEventArgs e | Метод обрабатывающий событие, при нажатии на Enter, который создает новое окно. | |
| Wigth\_Copy\_  KeyDown | private | void | object sender, KeyEventArgs e | Метод обрабатывающий событие, при нажатии на Enter, который создает новое окно. | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Описание классов, полей, методов и свойств класса GrillEditor.xaml.cs  Таблица 3.8 | | | | |
| Поля | | | | | |
| Имя | Модификатор  доступа | Тип | Назначение | | |
| H | private | uint | Высота решетки. | | |
| W | private | uint | Ширина решетки. | | |
| l | private | Language | Язык окна. | | |
| BBoxes | private | Button[,] | Массив представляющий собой решетку Кардано. | | |
| TBoxes | private | TextBox[,] | Массив ячеек для записи текста. | | |
| BaseHeight | private | double | Базовая высота окна. | | |
| BaseWidth | private | double | Базовая ширина окна. | | |
| cancle | private | Button | Кнопка отмены. | | |
| HideShow | private | Button | Кнопка показать/скрыть все ячейки. | | |
| buffer | private | Button | Вспомогательная кнопка для отмены. | | |
| save | private | Button | Кнопка сохранения решетки. | | |
| Методы | | | | | |
| Имя | Модификатор  доступа | Тип | Аргументы | Назначение | |
| GrillEditor | public | конструктор | uint h,  uint w, Language language | Конструктор окна GrillEditor. | |
| Window\_  SizeChanged | private | void | object sender, SizeChangedEventArgs e | Метод обрабатывающий событие при изменении размера окна. | |
| HideShowall | public | void | object sender, EventArgs e | Метод скрывающий и показывающий все ячейки. | |
| ButtonClickEvent | private | void | object sender, EventArgs e | Метод обрабатывающий событие при нажатии на ячейку, который скрывает нажатую ячейку. | |
| Cancle | private | void | object sender,  EventArgs e | Метод обрабатывающий событие при нажатии на кнопку, отменяет последнее действие. | |
| Save | private | void | object sender,  EventArgs e | Метод обрабатывающий событие при нажатии на кнопку, который сохраняет сделанную решетку. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Описание классов, полей, методов и свойств класса Help.xaml.cs  Таблица 3.9 | | | | | | | |
| Поля | | | | | | | | |
| Имя | | Модификатор  доступа | | Тип | | Назначение | | |
| lng | | private | | Language | | Язык окна | | |
| Методы | | | | | | | |
| Имя | Модификатор  доступа | | Тип | | Аргументы | | Назначение |
| Help | public | | конструктор | | Language l | | Конструктор окна Help. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Описание классов, полей, методов и свойств класса InformationCoder.xaml.cs  Таблица 3.10 | | | | | | | | |
| Поля | | | | | | | | |
| Имя | Модификатор  доступа | | Тип | | Назначение | | | |
| H | | private | uint | | Высота решетки. | | | | |
| W | | private | uint | | Ширина решетки. | | | | |
| language | | private | Language | | Язык окна. | | | | |
| Bboxes | | private | Button[,] | | Массив представляющий собой решетку Кардано. | | | | |
| Tboxes | | private | TextBox[,] | | Массив ячеек для записи текста. | | | | |
| BaseHeight | | private | double | | Базовая высота окна. | | | | |
| BaseWidth | | private | double | | Базовая ширина окна. | | | | |
| ShowHide | private | | Button | | Кнопка показать/скрыть решетку. | | | |
| GText | private | | TextBox | | Поле для вывода зашифрованного сообщения | | | |
| CodedGrid | private | | string[] | | Решетка. | | | |
| Методы | | | | | | | | |
| Имя | Модификатор  доступа | | Тип | | Аргументы | | Назначение | |
| Information  Coder | public | | конструктор | | uint h,  uint w, Language l | | Конструктор окна InformationCoder. | |
| GetTextButtonClick | private | | void | | object sender, EventArgs e | | Метод обрабатывающий событие при нажатии на кнопку, и пользователь получает зашифрованное сообщение. | |
| Window\_  SizeChanged | private | | void | | object sender, SizeChangedEventArgs e | | Метод обрабатывающий событие при изменении размера окна. | |
| Open | public | | void | | object sender, EventArgs e | | Метод открывающий решетку Кардано. | |
| SH | public | | void | | object sender, EventArgs e | | Метод скрывающий и показывающий решетку. | |
| kd | | private | | void | | object sender, TextChangedEventArgs e | | Метод упрощающий способ печати в ячейки. | |
| del | | private | | void | | object sender, KeyEventArgs e | | Метод упрощающий способ печати в ячейки. | |
| Описание классов, полей, методов и свойств класса InformationDecoder.xaml.cs  Таблица 3.11 | | | | | | | | |
| Поля | | | | | | | | |
| Имя | Модификатор  доступа | | Тип | | Назначение | | | |
| H | | private | uint | | Высота решетки. | | | | |
| W | | private | uint | | Ширина решетки. | | | | |
| language | | private | Language | | Язык окна. | | | | |
| Bboxes | | private | Button[,] | | Массив представляющий собой решетку Кардано. | | | | |
| Tboxes | | private | TextBox[,] | | Массив ячеек для записи текста. | | | | |
| BaseHeight | | private | double | | Базовая высота окна. | | | | |
| BaseWidth | | private | double | | Базовая ширина окна. | | | | |
| ShowHide | private | | Button | | Кнопка показать/скрыть решетку. | | | |
| GText | private | | TextBox | | Поле для ввода зашифрованного сообщения | | | |
| CodedGrid | private | | string[] | | Решетка. | | | |
| Методы | | | | | | | | |
| Имя | Модификатор  доступа | | Тип | | Аргументы | | Назначение | |
| InformationDecoder | public | | конструктор | | uint h,  uint w, Language l | | Конструктор окна InformationDecoder. | |
| GetTextButtonClick | private | | void | | object sender, EventArgs e | | Пользователь вводит зашифрованное сообщение. | |
| Window\_  SizeChanged | private | | void | | object sender, SizeChangedEventArgs e | | Метод обрабатывающий событие при изменении размера окна. | |
| Open | public | | void | | object sender, EventArgs e | | Метод открывающий решетку Кардано. | |
| SH | public | | void | | object sender, EventArgs e | | Метод скрывающий и показывающий решетку. | |
| ENTERTEXT | private | | void | | - | | Метод заполняющий ячейки текстом. | |
| GetTextButtonClick | private | | void | | object sender, EventArgs e | | Метод обрабатывающий событие при нажатии на кнопку, и пользователь получает зашифрованное сообщение. | |
| EnterEv | private | | void | | object sender, KeyEventArgs e | | Метод обрабатывающий событие при нажатии на Enter в поле для ввода текста в ячейки. | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Описание классов, полей, методов и свойств класса MainMenu.xamal.cs  Таблица 3.12 | | | | | | |
| Поля | | | | | | |
| Имя | Модификатор  доступа | Тип | | Назначение | | |
| classlng | private | Language | | Язык окна. | | |
| Методы | | | | | | |
| Имя | Модификатор  доступа | Тип | | Аргументы | | Назначение |
| MainMenu | public | конструктор | | Language language | | Конструктор окна MainMenu. |
| GrillEditor  Button\_Click | private | void | | object sender, RoutedEvent  Args e | | Метод обрабатывающий событие при нажатии на кнопку, который открывает редактор решетки. |
| Information  CoderButton\_Click | private | void | | object sender, RoutedEvent  Args e | | Метод обрабатывающий событие при нажатии на кнопку, который открывает шифратор информации. |
| InformationDecoderButton\_Click | private | void | | object sender, RoutedEvent  Args e | | Метод обрабатывающий событие при нажатии на кнопку, который открывает шифратор информации. |
| ChatButton\_  Click | private | void | | object sender, RoutedEvent  Args e | | Метод обрабатывающий событие при нажатии на кнопку, который открывает чат. |
| Close\_  MouseEnter | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при наведении на крестик. |
| Close\_  MouseLeave | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при выходе из области крестика. |
| Close\_  MouseDown | private | void | | object sender, MouseButtonEventArgs e | | Метод обрабатывающий событие при нажатии на крестик. |
| Hide\_  MouseEnter | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при наведении на Hide pannel. |
| Hide\_  MouseLeave | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при выходе из области Hide pannel. |
| Hide\_  MouseDown | private | void | | object sender, MouseButtonEventArgs e | | Метод обрабатывающий событие при нажатии на Hide pannel. |
| AboutProgrammButton\_Click | private | void | | object sender, RoutedEvent  Args e | | Метод обрабатывающий событие при нажатии на кнопку, который открывает информацию о программе. |
| HelpButton\_  Click | private | void | | object sender, RoutedEvent  Args e | | Метод обрабатывающий событие при нажатии на кнопку, который открывает помощь. |
| GrillEditor  Button\_MouseEnter | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при наведении на кнопку редектора решетки. |
| Information  CoderButton\_MouseEnter | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при наведении на кнопку шифратора решетки. |
| InformationDecoderButton\_MouseEnter | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при наведении на кнопку дешифратора решетки. |
| ChatButton\_  MouseEnter | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при наведении на кнопку чата. |
| Описание классов, полей, методов и свойств класса MainWindow.xamal.cs  Таблица 3.13 | | | | | | |
| Методы | | | | | | |
| Имя | Модификатор  доступа | Тип | | Аргументы | | Назначение |
| MainWindow | public | конструктор | | - | | Конструктор окна MainWindow |
| RUS\_Mouse  Enter | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при наведении на картинку русского флага. |
| GB\_Mouse  Enter | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при наведении на картинку британского флага. |
| RUS\_Mouse  Leave | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при выходе из области картинки русского флага. |
| GB\_Mouse  Leave | private | void | | object sender, MouseEventArgs e | | Метод обрабатывающий событие при выходе из области картинки британского флага. |
| GB\_Mouse  Down | private | void | | object sender, MouseButtonEventArgs e | | Метод обрабатывающий событие при нажатии на картинку русского флага. |
| RUS\_Mouse  Down | private | void | | object sender, MouseButtonEventArgs e | | Метод обрабатывающий событие при нажатии на картинку британского флага. |
|  |  |  | |  | |  |
| Описание классов, полей, методов и свойств класса Server.xaml.cs  Таблица 3.14 | | | | | | |
| Классы | | | | | | |
| Имя | Назначение | | | | | |
| ClientObject | Класс клиента | | | | | |
| ServerObject | Класс сервера | | | | | |
| Поля | | | | | | |
| Имя | Модификатор  доступа | Тип | | Назначение | | |
| H | private | uint | | | Высота решетки. | | |
| W | private | uint | | | Ширина решетки. | | |
| \_context | private | SynchronizationContext | | | Обеспечивает базовую функциональность для распространения контекста синхронизации в различных моделях синхронизации. | | |
| language | private | Language | | Язык окна. | | |
| Chat | private | TextBox | | Поле для чата. | | |
| command | private | TextBox | | Поле для ввода сообщения от сервера. | | |
| Send | private | Button | | Кнопка для отправки сообщения. | | |
| IP | private | IPEndPoint | | Ip и порт сервера. | | |
| server | private | ServerObject | | Сервер. | | |
| listenThread | private | Thread | | Отдельный поток для прослушивания. | | |
| Методы | | | | | | |
| Имя | Модификатор  доступа | Тип | | Аргументы | | Назначение |
| Server | public | конструктор | | uint h, uint w, Language lng, IPEndPoint ipadress | | Конструктор окна Server |
| KeyDown | private | void | | object sender, KeyEventArgs e | | Метод обрабатывающий событие на нажатие Enter, который отправляет сообщение всем клиентам |
| Window\_Closing | private | void | | object sender, System.ComponentModel.CancelEventArgs e | | Метод обрабатывающий событие при закрытии окна. |
| sendcommand | private | void | | object sender, RoutedEventArgs e | | Метод обрабатывающий событие на нажатие кнопки, который отправляет сообщение. |
| Описание полей методов и свойств класса ServerAndClientSetttings.xaml.cs  Таблица 3.15 | | | | | | |
| Поля | | | | | | |
| Имя | Модификатор  доступа | Тип | Назначение | | | |
| Т | private | char | Переменная, обозначающая тип окна. | | | |
| Lang | private | Language\_  Library.  Language | Язык окна. | | | |
| H | private | uint | Высота решетки. | | | |
| W | private | uint | Ширина решетки. | | | |
| Методы | | | | | | |
| Имя | Модификатор  доступа | Тип | Аргументы | | | Назначение |
| ServerAnd  ClientSetttings | public | конструктор | uint h, uint w, Language\_  Library.Language l,  char Type\_of\_window | | | Конструктор окна ServerAnd  ClientSetttings. |
| IP\_KeyDown | private | void | object sender, KeyEventArgs e | | | Метод обрабатывающий событие на нажатие Enter, который запускает сервер/клиент |
| Port\_KeyDown | private | void | object sender, KeyEventArgs e | | | Метод обрабатывающий событие на нажатие Enter, который запускает сервер/клиент |
| Continue\_Click | private | void | object sender, RoutedEventArgs e | | | Метод обрабатывающий событие на нажатие кнопки, который запускает сервер/клиент |
| Описание полей методов и свойств класса ServerOrClient.xaml.cs  Таблица 3.16 | | | | | | |
| Поля | | | | | | |
| Имя | Модификатор  доступа | Тип | Назначение | | | |
| lng | private | Language | Язык окна | | | |
| h | private | uint | Высота решетки. | | | |
| w | private | uint | Ширина решетки. | | | |
| Методы | | | | | | |
| Имя | Модификатор  доступа | Тип | Аргументы | | | Назначение |
| Server\_MouseEnter | private | void | object sender, MouseEventArgs e | | | Метод обрабатывающий событие при наведении на логотип сервера. |
| Client\_MouseEnter | private | void | object sender, MouseEventArgs e | | | Метод обрабатывающий событие при наведении на логотип клиента. |
| Server\_MouseLeave | private | void | object sender, MouseEventArgs e | | | Метод обрабатывающий событие при выходе из области логотипа сервера. |
| Client\_MouseLeave | private | void | object sender, MouseEventArgs e | | | Метод обрабатывающий событие при выходе из области логотипа клиента. |
| Client\_MouseDown | private | void | object sender, MouseButtonEventArgs e | | | Метод обрабатывающий событие при нажатии на логотип клиента. |
| Server\_MouseDown | private | void | object sender, MouseButtonEventArgs e | | | Метод обрабатывающий событие при нажатии на логотип сервера. |

# **ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированх |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |