D625CDBE8E3D41C2A3814D4C89B9155D3C4EF33E22474FB6AF784AFF2EEF01C07B6A12F963F347D29B55F8DE1B022AF2

**СОДЕРЖАНИЕ**

[1.ВВЕДЕНИЕ 3](#_Toc356776635)

[2. НАЗНАЧЕНИЕ РАЗРАБОТКИ 4](#_Toc356776636)

[2.1 Функциональное назначение 4](#_Toc356776637)

[2.2 Эксплуатационное назначение 4](#_Toc356776638)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 5](#_Toc356776639)

[3.1 Постановка задачи на разработку программы 5](#_Toc356776640)

[3.2 Описание алгоритма 6](#_Toc356776641)

[3.3 Описание и обоснование выбора входных и выходных данных 18](#_Toc356776642)

[3.4 Описание и обоснование выбора состава технических средств 18](#_Toc356776643)

[3.5 Описание и обоснование выбора состава технических средств 18](#_Toc356776644)

[4.ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 19](#_Toc356776645)

[5.СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 20](#_Toc356776646)

[ПРИЛОЖЕНИЕ: ОПИСАНИЕ КЛАССОВ, ПОЛЕЙ И МЕТОДОВ 21](#_Toc356776647)

# ВВЕДЕНИЕ

Наименование программы: «Программа генерации судоку алгоритмом Dancing Links» - программа, которая будет использоваться в качестве приложения для игры в судоку, позволяющего генерировать новые паззлы судоку и предоставлять их решение.

Краткое наименование программы: «Программа генерации судоку».

# 2. НАЗНАЧЕНИЕ РАЗРАБОТКИ

2.1 Функциональное назначение

Программа генерирует паззлы судоку, предоставляет пользователю интерфейс для их решения и автоматический решатель.

2.2 Эксплуатационное назначение

Программа будет использоваться для игры пользователя в судоку.

# 3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

* 1. Постановка задачи на разработку программы

1. Задание на курсовую работу. Приказ № \_\_\_\_\_\_\_\_ «Об утверждении тем и руководителей курсовых работ студентов образовательной программы «Программная инженерия» факультета компьютерных наук» от ХХ.ХХ.ХХХХ.
2. Техническое задание «Программа генерации судоку алгоритмом «Dancing Links».
3. План проекта разработки.

Цель разработки: создать продукт, позволяющий генерировать паззлы судоку при помощи алгоритма “Dancing links” и предоставляющий пользователю интерфейс для решения паззлов судоку.

* 1. Описание алгоритма

Алгоритм генерации паззла судоку состоит из трех основных частей:

1. Генерация случайного начального поля;
2. Решение получившегося паззла алгоритмом «Dancing Links»;
3. Удаление из решения по одной случайно выбранной цифре до тех пор, пока паззл имеет единственное решение.

Основной частью генерации является решение паззла-начального условия алгоритмом «Dancing links». Рассмотрим этот алгоритм подробнее:

Алгоритм «Dancing links» представляет собой модификацию алгоритма полного перебора и позволяет решать проблему точного покрытия. Для начала приведем пример решения такой проблемы, а затем покажем, как решение паззла судоку сводится к ней.

### 3.2.1 Проблема точного покрытия

Постановка задачи. Проблему точного покрытия можно сформулировать следующим образом: пусть имеется матрица из «0» и «1» (Рис. 3.2.1):

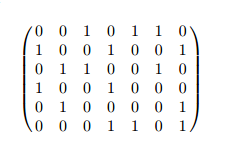


Рисунок 3.2.1

Требуется выбрать такое подмножество строк этой матрицы, что в каждом столбце этого подмножества строк остается ровно одна «1». Для данной матрицы это будут строки 1,4,5 (нумерация с единицы).

Структура данных. Теперь необходимо представить эту матрицу в виде связных списков. Добавим первой строкой заголовок столбца, который будет хранить в себе количество единиц в столбце. Также добавим еще один заголовок столбца, соответствующий «нулевой колонке». Все заголовки столбцов образуют двусвязный список, каждый ряд образует двусвязный список из ячеек с «1», каждый столбец также формирует двусвязный список из ячеек с «1» и заголовком столбца. Получается структура, представленная на Рис. 3.2.2:

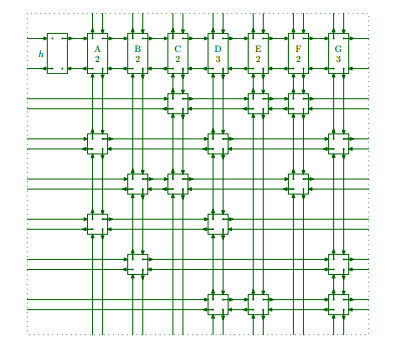
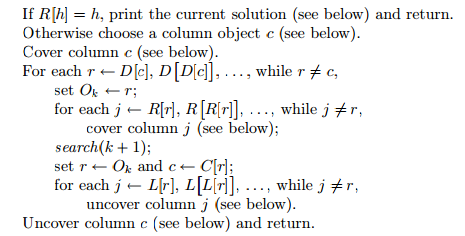
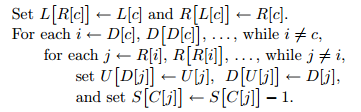


Рисунок 3.2.2

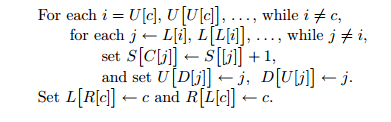
Алгоритм «Dancing links». Псевдокод метода поиска решения search(k) , запускаемого с k = 0, можно представить в следующем виде:



Псевдокод метода cover:



Псевдокод метода uncover:



Метод cover позволяет «рассмотреть» столбец: столбец изымается из двусвязного списка заголовков столбцов. Также из двусвязных списков всех столбцов изымаются строки, в которых в «рассматриваемом» столбце стоит «1». То есть, для нашего примера, «рассмотрение» столбца А приведет к конфигурации, приведенной на Рис. 3.2.3

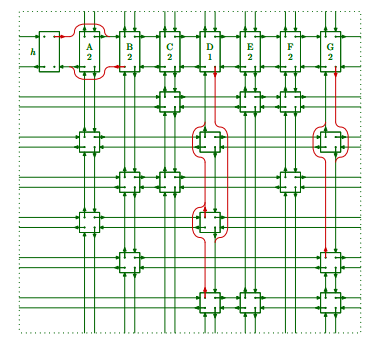


Рисунок 3.2.3

Операция uncover производит обратную операцию. Таким образом, рекурсивный метод search производит следующие действия:

1. Проверяем правого соседа начального заголовка h. Если он равен самому себе, это означает, что мы рассмотрели все столбцы, так как на каждом этапе мы выбираем одну строку, в которой стоит «1» в рассматриваемом столбце и удаляет все строки, которые могли бы вызывать конфликт (несколько «1» в одном столбце). То есть мы уже нашли такой набор строк, что в каждом его столбце ровно одна «1» - печатаем решение и выходим на предыдущий уровень вызова, где продолжается поиск других решений. Если же правый сосед h отличен от h, переходим к пункту 2.
2. Выбираем следующий столбец. Это может быть произвольный столбец. Для лучшего разбиения Д. Кнут предлагает выбирать столбец, в котором наименьшее количество «1».
3. «Рассматриваем» выбранный столбец. На данном этапе это сводится к тому, что мы определяем набор строк, в которых есть «1» в текущем столбце. При этом для дальнейшего рассмотрения мы убираем текущий столбец из заголовков (т.к. мы уже определили строки, одна из которых обязательно будет являться частью решения, т.к. содержит «1» в нужном столбце). Также мы удаляем эти строки из связанных списков столбцов, т.к. повторно ту же строку мы выбрать не можем.
4. В цикле выбираем по одной выбранной на предыдущем этапе строке. Это означает, что мы добавляем одну из строк к решению.
5. Далее нам необходимо «уладить конфликты» для выбранной строки. Это происходит потому, что выбирая строку, в которой стоит «1» для текущего столбца, мы выбираем эту же строку и для всех тех столбцов, в которых стоит «1» для данной строки. Для этого мы производим операцию cover для всех тех столбцов, в которых в текущей строке стоит «1».
6. На данном этапе у нас уже выбрана строка решения, и удалены все строки, выбор которых исключен из-за ее выбора. Поэтому мы рекурсивно вызываем метод search, для выбора следующего столбца из сокращенной матрицы.
7. При возвращении управления в текущей метод мы производим операцию uncover для всех столбцов, с которыми была произведена операция cover перед рекурсивным вызовом. Uncover производится в обратном порядке, чтобы не повредить структуру связного списка. Таким образом, мы возвращаем все условия к тому состоянию, когда мы определили набор строк для «рассматриваемого» столбца.
8. Переходим на следующую итерацию цикла, то есть выбираем другую строку, в которой стоит «1» для текущего столбца. Заметим, что в случае, когда заголовок столбца указывает сам на себя, ни одной итерации цикла не будет выполнено. Это говорит о том, что это тупиковая ветка решения: так как заголовок h не указывает сам на себя, у нас остался столбец, для которого в текущем наборе нет ни одной «1», но заголовок этого столбца указывает сам на себя: то есть не осталось строк, в которых есть «1» для данного столбца.
9. После завершения цикла выполняем uncover для текущего столбца и передаем управление в вызывающий метод.

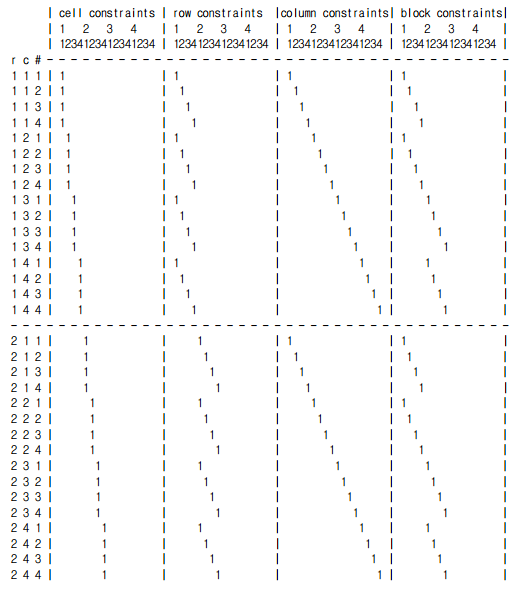
### 3.2.2 Игра судоку как проблема точного покрытия

Судоку – это паззл, представляющий собой паззл размером 9\*9, разбитый на 9 квадратов 3\*3. Цель игры – заполнить поле цифрами от 1 до 9 так, чтобы в каждом столбце, строке и квадрате каждая цифра встречалась только один раз.

Теперь рассмотрим проблему представления судоку в виде матрицы из «0» и «1». Для этого рассмотрим ограничения, которые накладываются на паззл судоку:

1. Только одна цифра располагается в клетке;
2. В строке не может быть двух одинаковых цифр;
3. В столбце не может быть двух одинаковых цифр;
4. В квадрате не может быть двух одинаковых цифр;

Это говорит о том, что судоку можно представить в виде таблицы из «0» и «1», где каждая строка будет представлять все возможные комбинации строки, столбца и значения цифры-кандидата, а столбцами будут все возможные ограничения. То есть, допустим, строка, представляющая кандидата 1 в столбце 1 и строке 1 и строка, представляющая кандидата 2 в той же строке и столбце, будут иметь «1» в одном и том же столбце матрицы, представляющим собой ограничение «в ячейке: строка 1, столбец 1 может находиться только одна цифра». Или, например, строки (1, 1, 1) и (1, 1, 2) будут иметь «1» в столбце ограничения «в строке 1 кандидат 1 встречается только один раз». Таким образом, матрица для судоку будет иметь =729 строк (все возможные комбинации строки, столбца и кандидата) и \*4 =324 столбцов (4 ограничения для каждой пары параметров, подлежащих ограничению). На рис. 3.2.4 представлена матрица ограничений для судоку 4\*4 с числами от 1 до 4 (реальная таблица слишком большая):



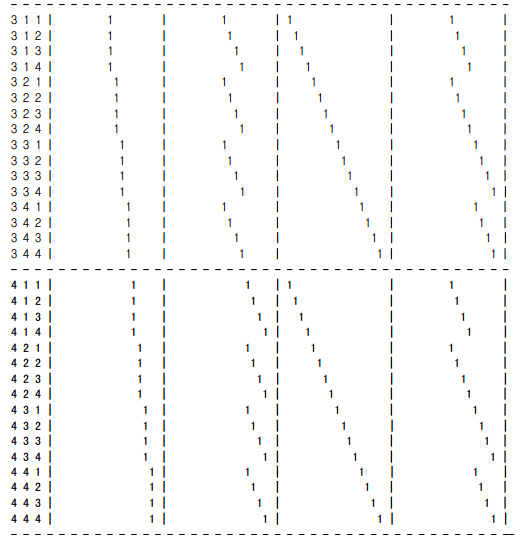


Рисунок 3.2.4

Для представления начальных условий необходимо удалить все строки, в которых встречается «1» хотя бы в одном из тех столбцов, в которых есть «1» в строке для предопределенного значения. То есть, если у нас в начальном условии в строке 1, столбце 1 стоит 1, то надо удалить все строки, где или столбец = 1, строка = 1, или столбец = 1, кандидат = 1, или строка = 1, кандидат = 1, или квадрат =1, кандидат = 1.

### 3.2.3 Эвристические алгоритмы

Также в программе реализованы эвристические алгоритмы решения судоку для определения сложности сгенерированного паззла судоку. Их описание[4]:

#### Одиночки

Метод заключается в отыскании в таблице одиночек, т.е. ячеек, в которых возможна только одна цифра и никакая другая. Записываем эту цифру в данную ячейку и исключаем ее из других клеток этой строки, столбца и блока. Например: в данной таблице имеются три «одиночки» (они выделены желтым цветом). (Рис 3.2.5)

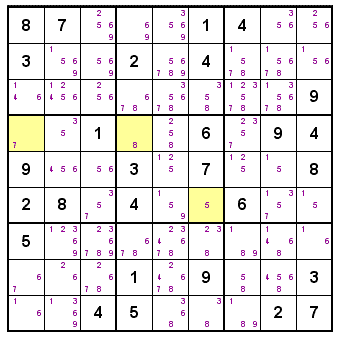


Рисунок 3.2.5

#### Скрытые одиночки

Если в ячейке стоит несколько кандидатов, но один из них не встречается больше ни в одной другой ячейке данной строки (столбца или блока), то такой кандидат называется «скрытой одиночкой». В следующем примере кандидат «4» в зеленом блоке найден только в центральной ячейке. Значит, в этой ячейке обязательно будет «4». Заносим «4» в данную ячейку и вычеркиваем из других ячеек 2-го столбца и 5-ой строки. Аналогично, в желтом столбце кандидат «2» встречается один раз, следовательно, в данную ячейку заносим «2» и исключаем «2» из ячеек 7-ой строки и соответствующего блока. (Рис. 3.2.6):

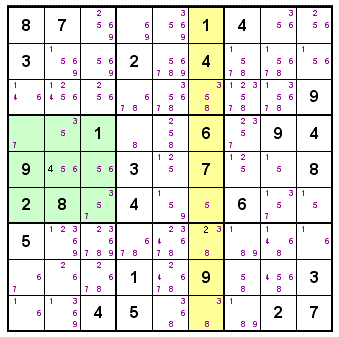


Рисунок 3.2.6

#### Запертый кандидат

Бывают случаи, когда кандидат в пределах блока находится только в одном строке (или в одном столбце). В силу того, что одна из этих ячеек обязательно будет содержать этого кандидата, из всех остальных ячеек данной строки (столбца) этого кандидата можно исключить.

В примере на Рис. 3.2.7, центральный блок содержит кандидата «2» только в центральном столбце (желтые ячейки). Значит, одна из этих двух ячеек точно должна быть «2», и никакие другие ячейки в том ряду вне этого блока не могут быть «2». Поэтому «2» может быть исключен как кандидат из других ячеек этого столбца (ячейки зеленого цвета).

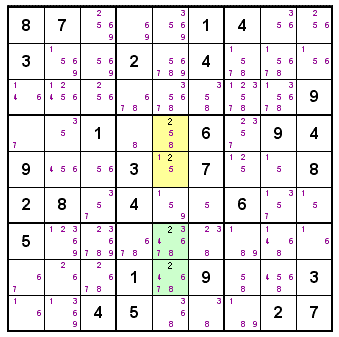


Рисунок 3.2.7

#### Открытые пары

Если две ячейки в группе (строке, столбце, блоке) содержат идентичную пару кандидатов и ничего более, то никакие другие ячейки этой группы не могут иметь значения этой пары. Эти 2 кандидата могут быть исключены из других ячеек в группе. На рис. 3.2.8, кандидаты «1» и «5» в колонках восемь и девять формируют Открытую Пару в пределах блока (желтые ячейки). Поэтому, так как одна из этих ячеек должна быть «1», а другая должны быть «5», кандидаты «1» и «5» исключаем из всех других ячеек этого блока (зеленые ячейки).

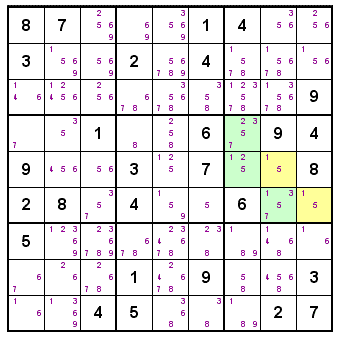


Рисунок 3.2.8

#### Скрытые пары

Если в двух ячейках в группе (строке, столбце, блоке) содержат кандидаты, среди которых идентичная пара, не встречающаяся ни в одной другой ячейке данного блока, то никакие другие ячейки этой группы не могут иметь значения этой пары. Следовательно, все другие кандидаты этих двух ячеек могут быть исключены. На рис. 3.2.9 кандидаты «7» и «5» в центральной колонке находятся только в ячейках желтого цвета, значит, всех остальных кандидатов из этих ячеек можно исключить.

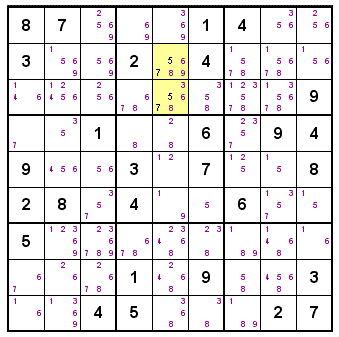


Рисунок 3.2.9

#### X-wing

Если значение имеет только два возможных местоположения в какой-то строке (столбце), то оно обязательно должно быть назначено в одну из этих ячеек. Если же существует еще одна строка (столбец), где этот же кандидат также может быть только в двух ячейках и столбцы (строки) этих ячеек совпадают, то ни одна другая ячейка этих столбцов (строк) не может содержать данную цифру. Рассмотрим пример (Рис. 3.2.10):

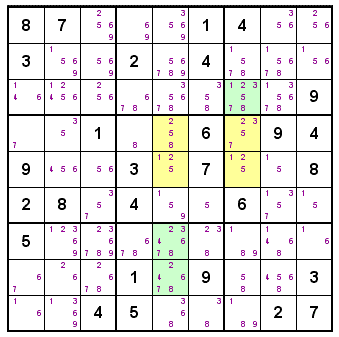


Рисунок 3.2.10

В 4-ой и 5-ой строках цифра «2» может быть только в двух ячейка желтого цвета, причем эти ячейки находятся в одинаковых столбцах. Следовательно, цифра «2» может быть записана только двумя способами: 1) если «2» записать в 5-ый столбец 4-ой строки, то из желтых ячеек «2» надо исключит и тогда в 5-ой строке положение «2» определяется однозначно 7-ым столбцом (Рис. 3.2.11)

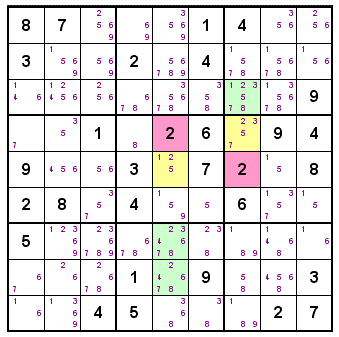


Рисунок 3.2.11

2) если «2» записать в 7-ой столбец 4-ой строки, то из желтых ячеек «2» надо исключит и тогда в 5-ой строке положение «2» определяется однозначно 5-ым столбцом.(Рис. 3.2.12)

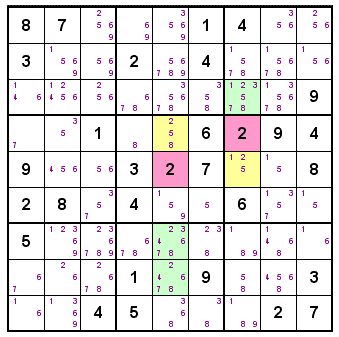


Рисунок 3.2.12

Следовательно, 5-ый и 7-ой столбец обязательно будут иметь цифру «2» либо в 4-ой строке, либо в 5-ой. Тогда из других ячеек данных столбцов цифру «2» можно исключить (зеленые клетки).

Вышеописанные эвристики соответствуют легкому уровню сложности судоку. Таким образом, если в процессе решения судоку использование только этих эвристик приводит к нахождению решения, уровень паззла судоку – легкий. Если же на каком-то этапе применение этих эвристик не позволяет продвинуться в решении паззла (определить значение ячейки или вычеркнуть возможного кандидата в ячейку из списка), то уровень паззла – трудный.

## Описание и обоснование выбора входных и выходных данных

В качестве формата входных и выходных данных разработан формат .sdk. Наличие собственного формата способствует снижению риска случайного повреждения данных.

## Описание и обоснование выбора состава технических средств

* Рекомендуется компьютер, оснащенный процессором Intel Pentium/Celeron, AMD K6/Athlon/Duron или совместимым с частотой 300 МГц или более (одно- или двухпроцессорная система). Минимальная частота процессора - 233 МГц;
* Рекомендуется 128 МБ ОЗУ или более. Минимально допустимый объем - 64 МБ (при наличии 64 МБ ОЗУ возможно снижение производительности и функциональности);
* Монитор и видеоадаптер Super VGA с разрешением 800 X 600 или более высоким;
* Клавиатура и мышь или совместимое указывающее устройство.

Данные требования к техническим средствам позволяют работать с приложением на большинстве современных ЭВМ.

## Описание и обоснование выбора состава технических средств

Программа разработана на языке Java. Это позволяет запускать программу на большинстве современных платформ при наличии установленной виртуальной машины Java JRE.

# 4.ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

Большинство сервисов для игры в судоку содержат базу судоку, а не генерируют новые паззлы. При этом, почти все программы, генерирующие судоку определяют сложность по количеству свободных полей, не используя эвристических алгоритмов. Данная программа реализует генерацию судоку и определение уровня сложности сгенерированного паззла судоку при помощи эвристических алгоритмов.

# 5. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

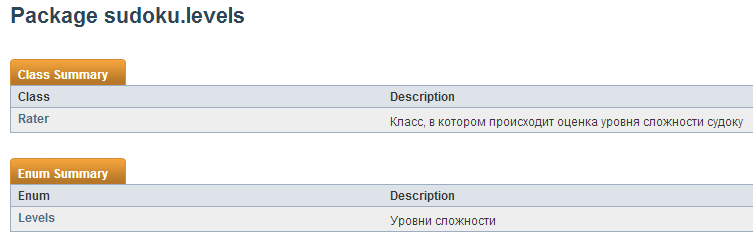
1. “Dancing links”, Donald E. Knuth, Millenial Perspectives in Computer Science, 2000, С. 187-214;
2. Robert Beezer, Mathematics and Computer Science University of Puget Sound [Электронный ресурс], Washington 1994 – Режим доступа: <http://buzzard.ups.edu/talks/beezer-2010-AIMS-sudoku.pdf> (Дата обращения 10.02.2013).
3. Colorado State University [Электронный ресурс], Fort Collins 2011 – Режим доступа <http://www.cs.colostate.edu/~cs420dl/slides/DLX.pdf> (Дата обращения 12.03.2013).
4. The University of Utah [Электронный ресурс], Salt Lake City, 2010 – Режим доступа <http://www.math.utah.edu/~yzhang/teaching/1030/Sudoku.pdf> (Дата обращения: 05.05.2013).
5. Cудоку-клуб [Электронный ресурс], 2007 – Режим доступа <http://www.sudoku-club.ru/howto.html> (Дата обращения 17.03.2013).

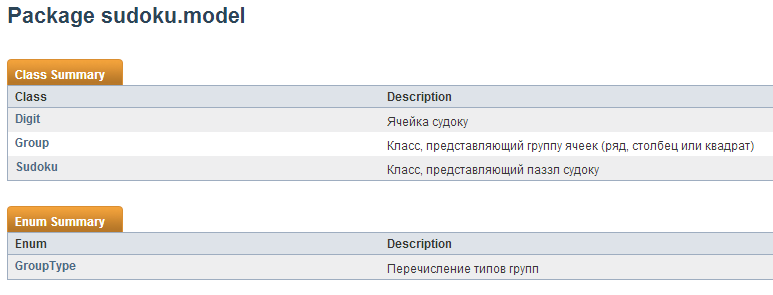
# ПРИЛОЖЕНИЕ

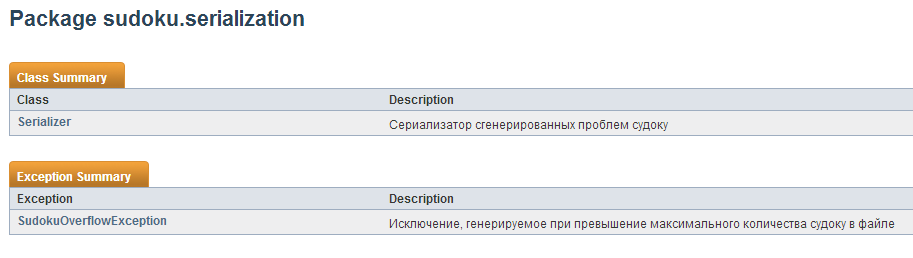
# ОПИСАНИЕ КЛАССОВ, ПОЛЕЙ И МЕТОДОВ

## Описание классов:





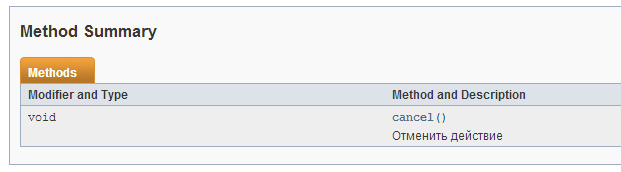




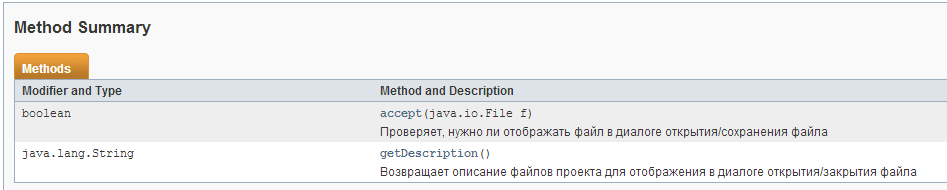


## Описание полей и методов:

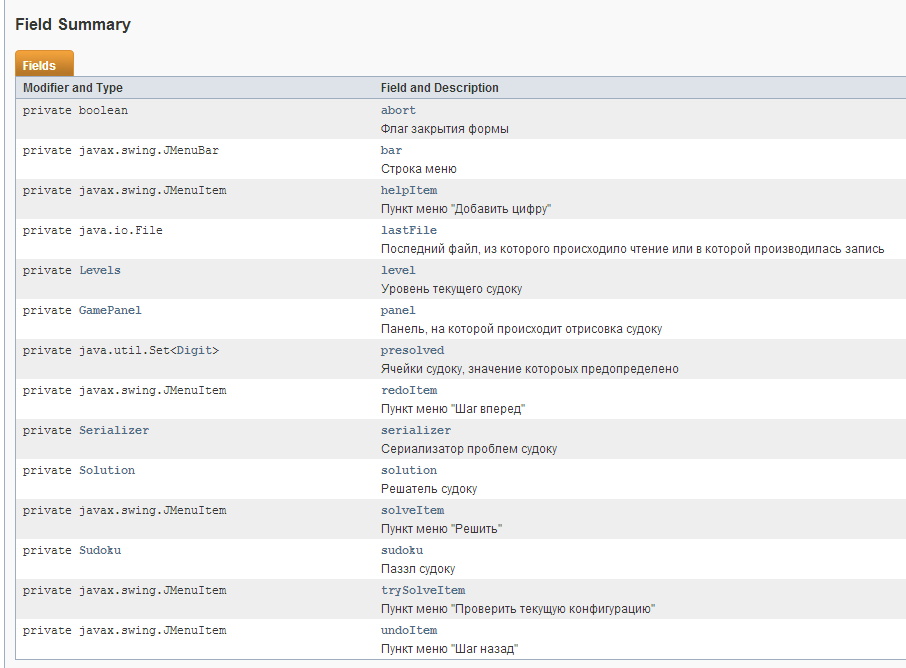
Интерфейс Cancelable:

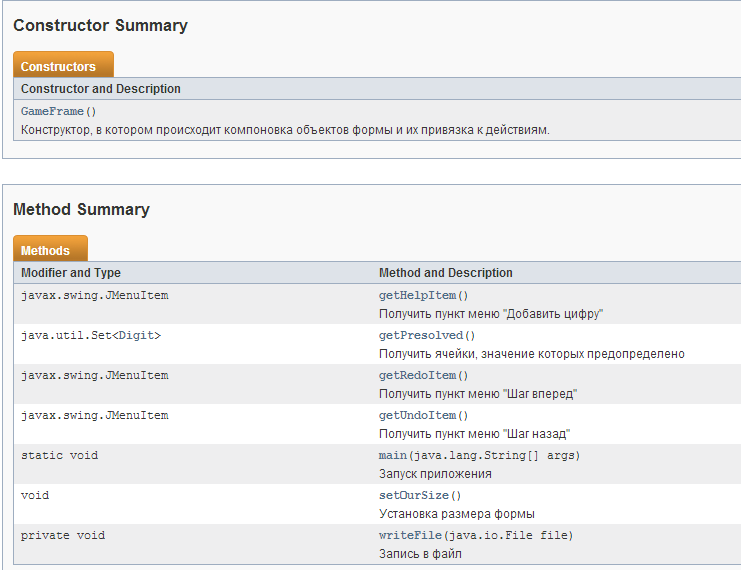


Класс GameFileFilter:

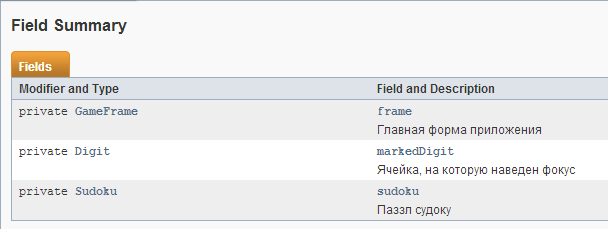


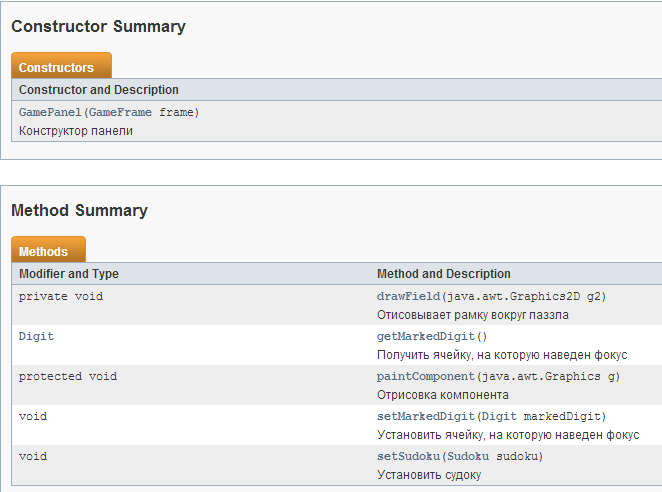
Класс GameFrame:



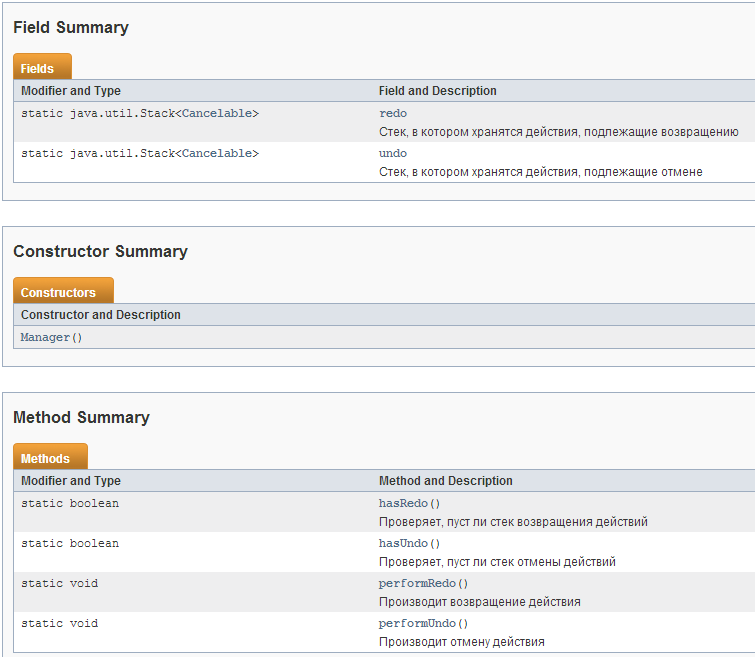


Класс GamePanel:

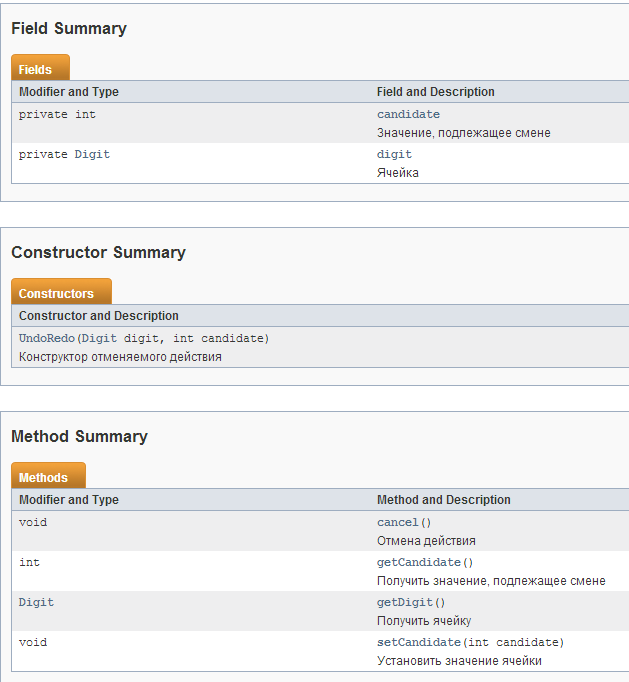




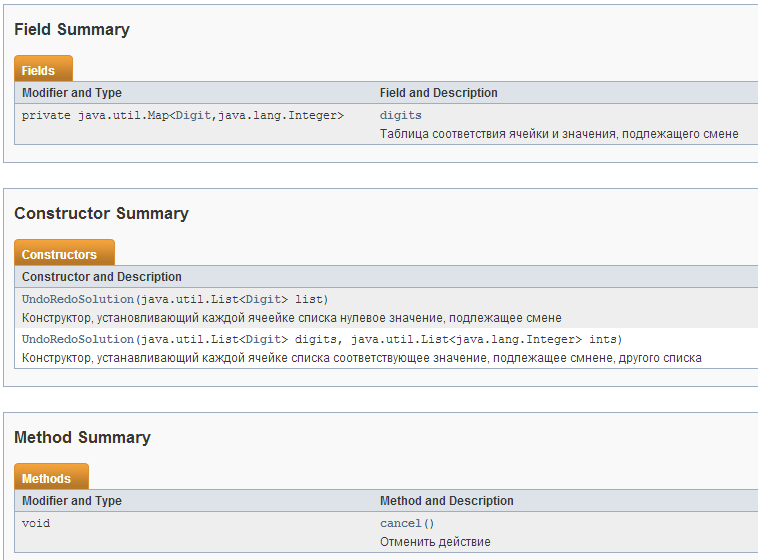
Класс Manager:



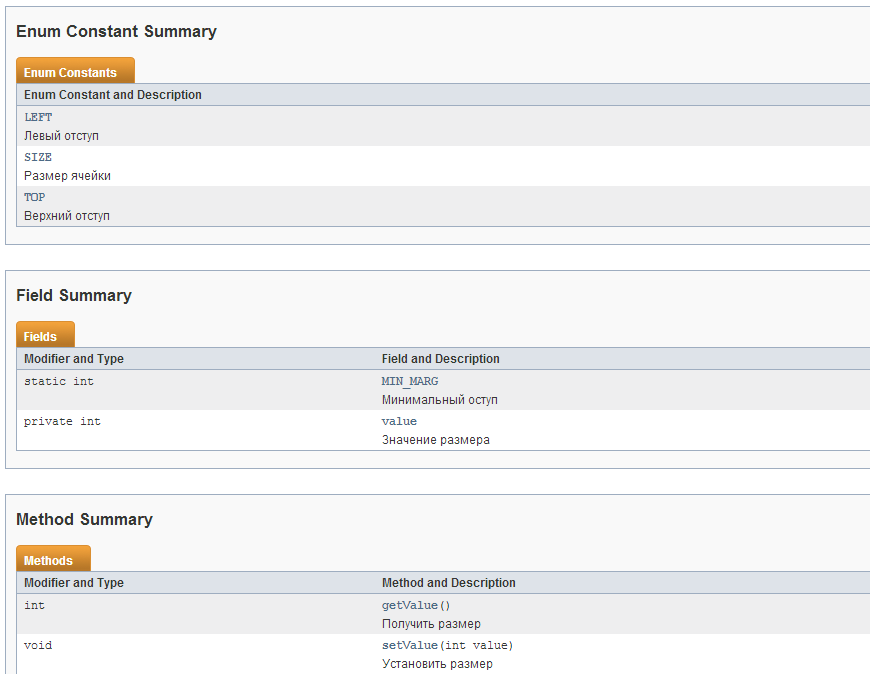
Класс UndoRedo:



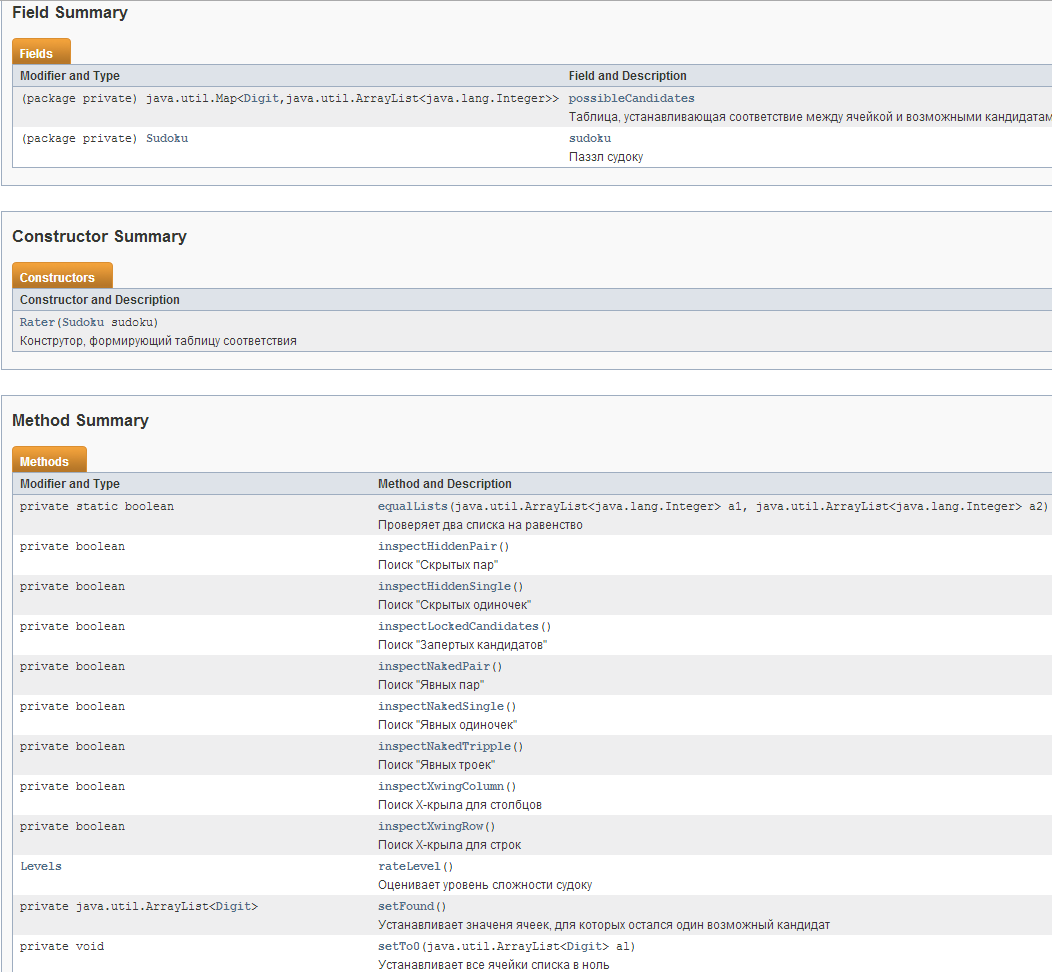
Класс UndoRedoSolution:



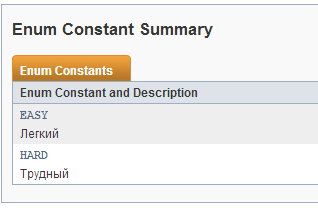
Перечисление Sizes:



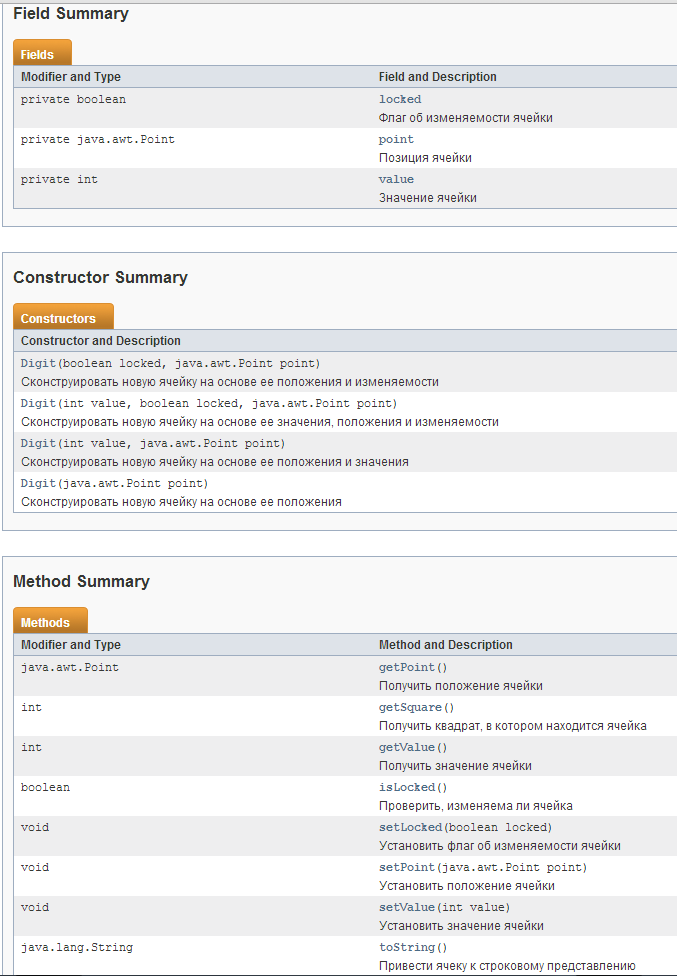
Класс Rater:



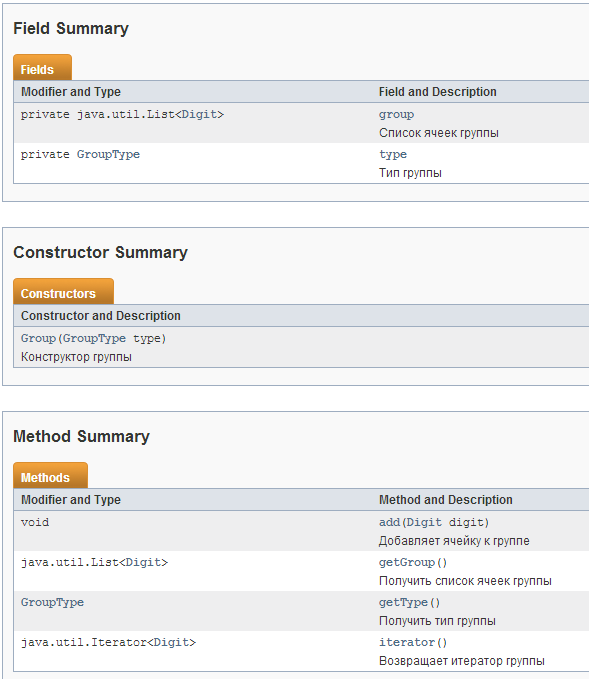
Перечисление Levels:



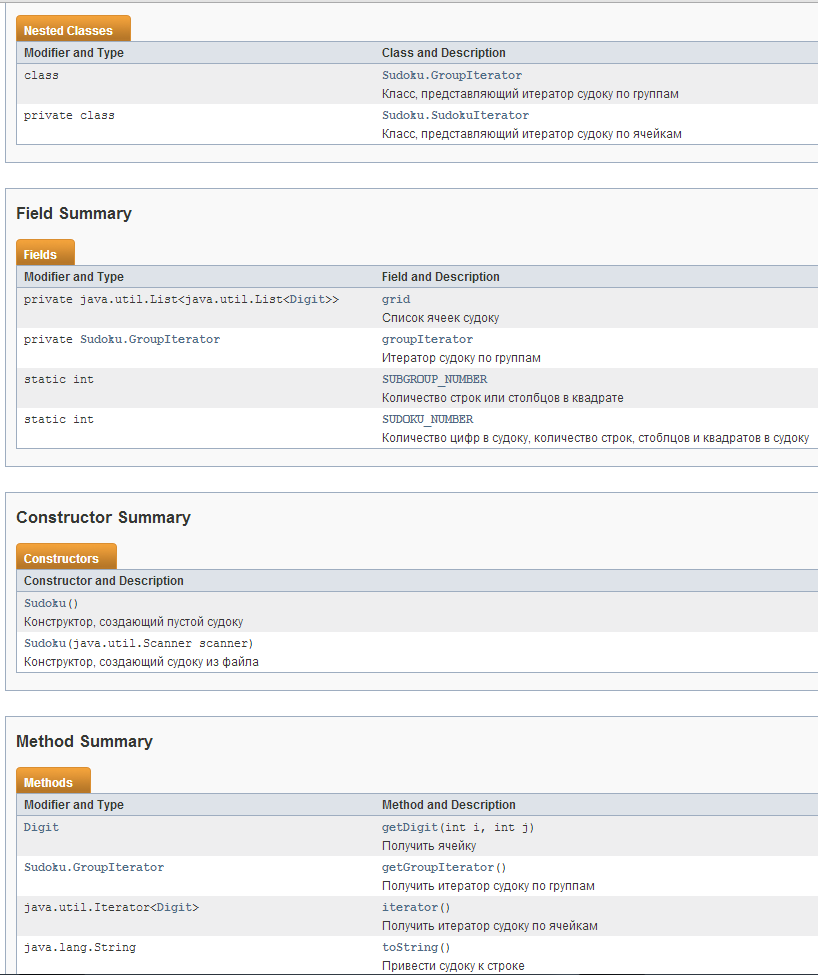
Класс Digit:



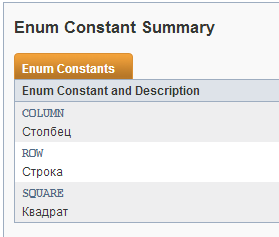
Класс Group:



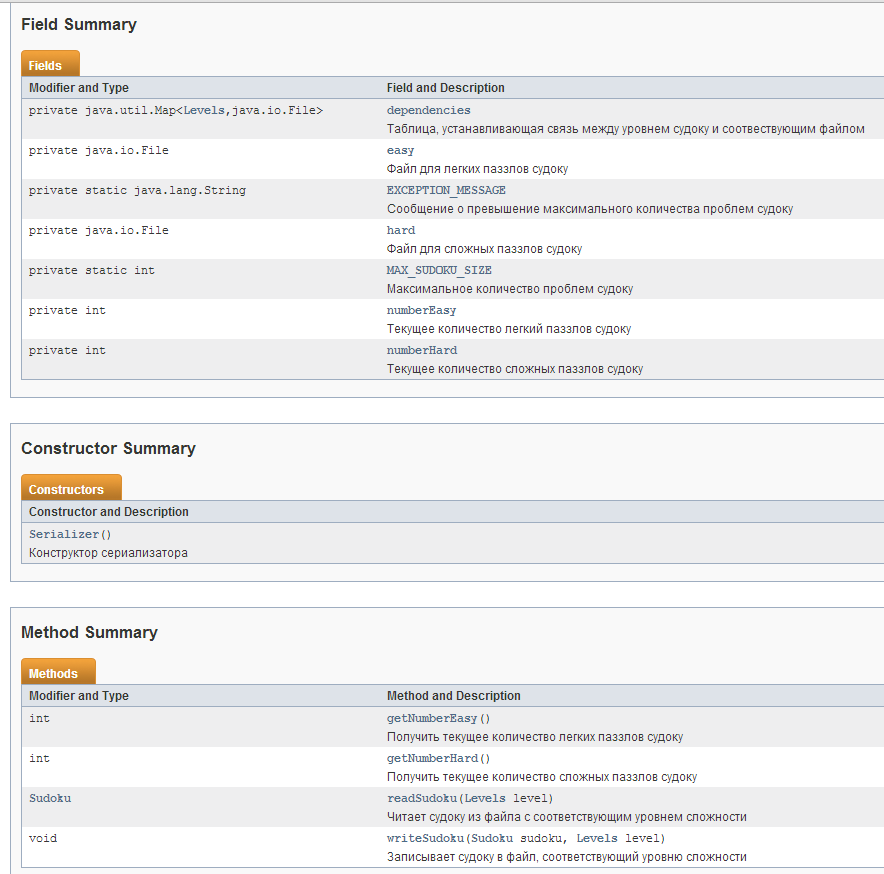
Класс Sudoku:



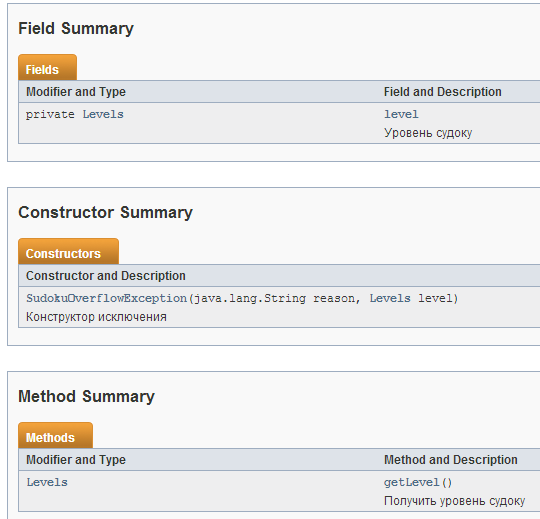
Перечисление GroupType:



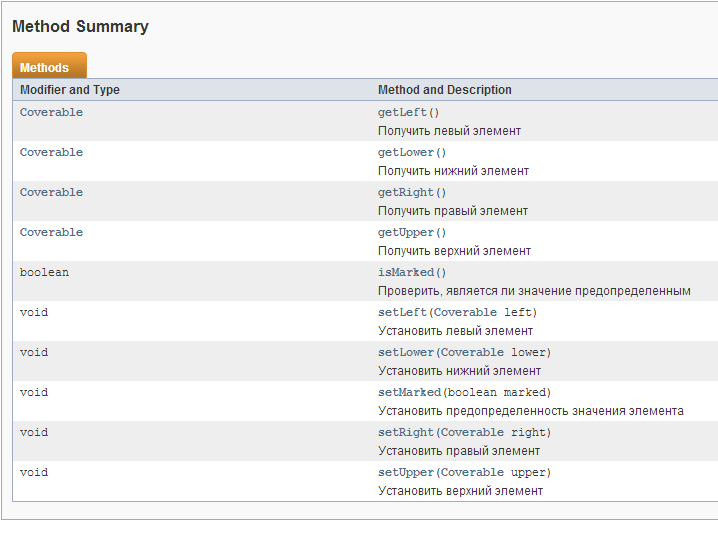
Класс Serializer:



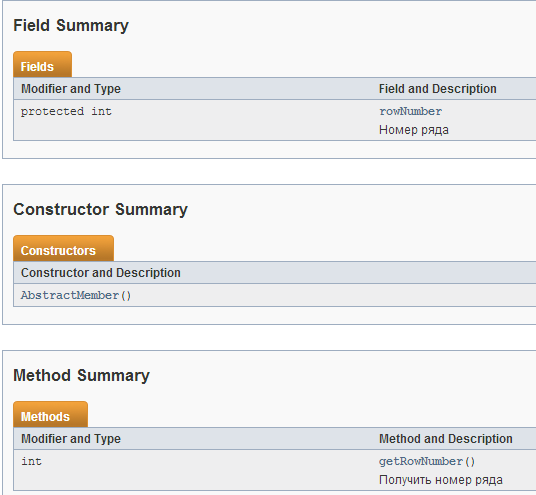
Класс SudokuOverflowException:



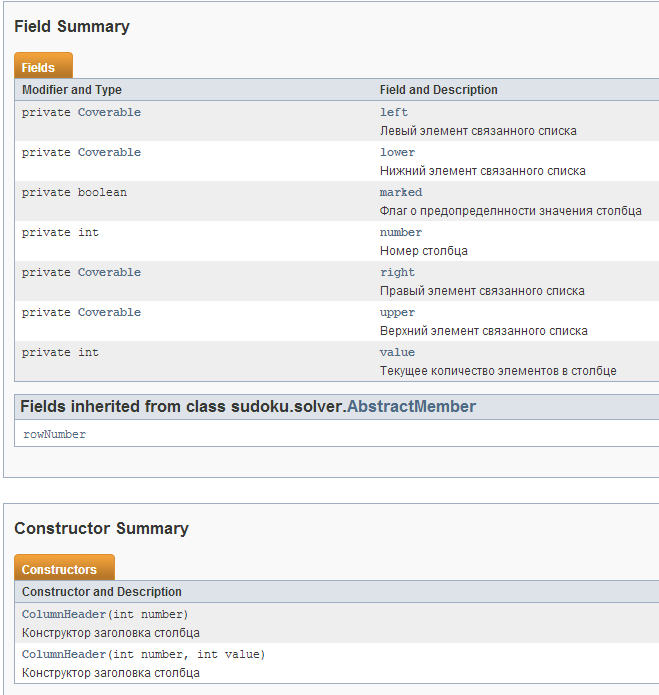
Интерфейс Coverable:

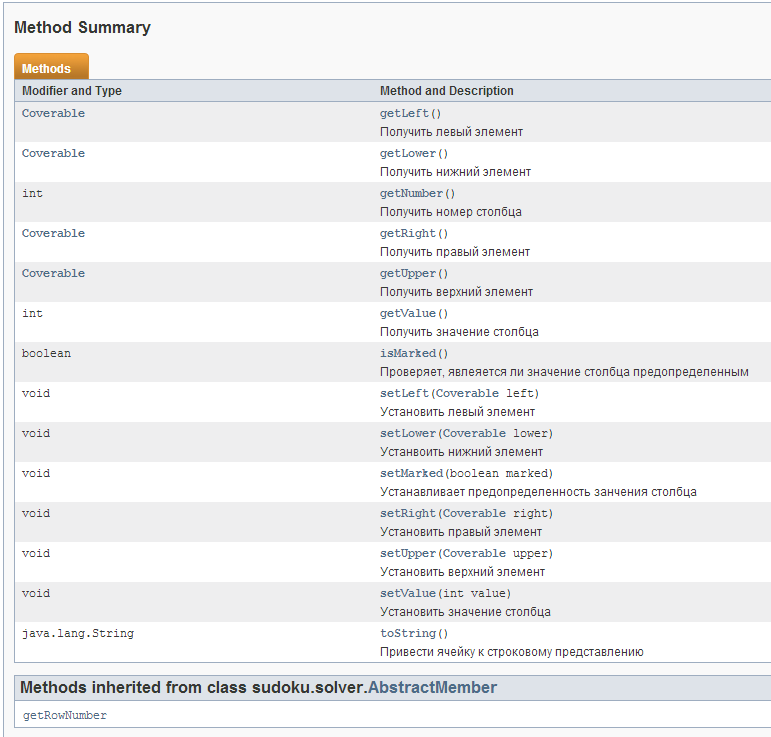


Класс AbstractMember:

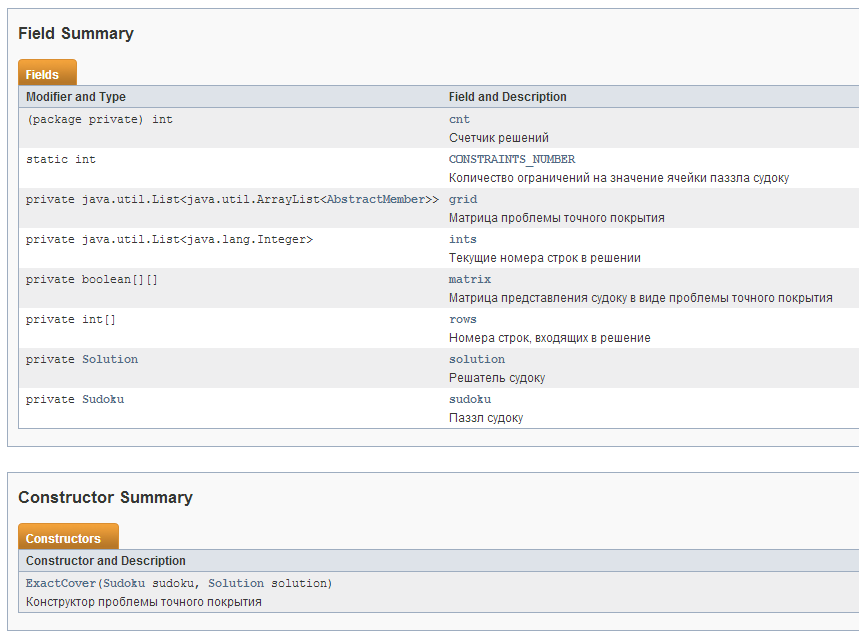


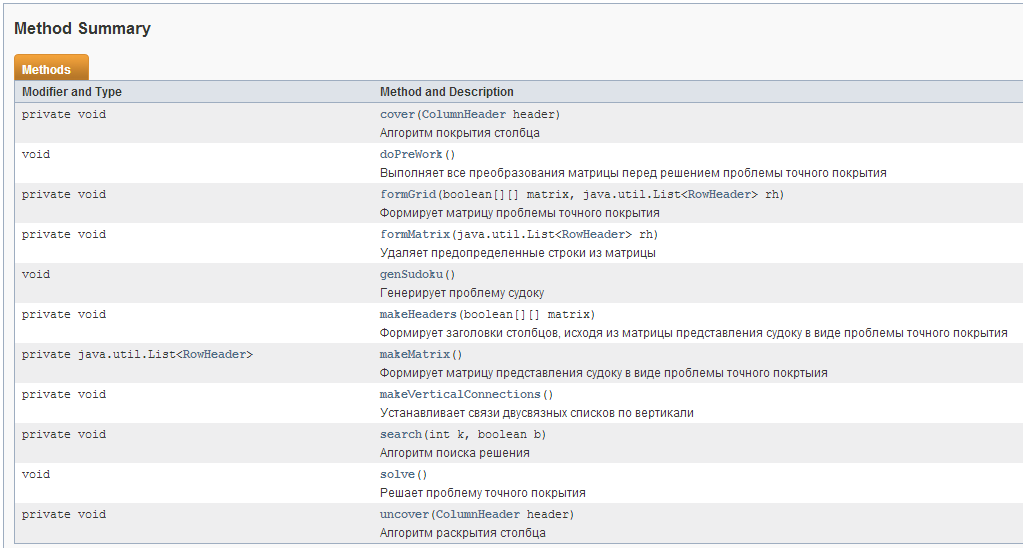
Класс ColumnHeader:



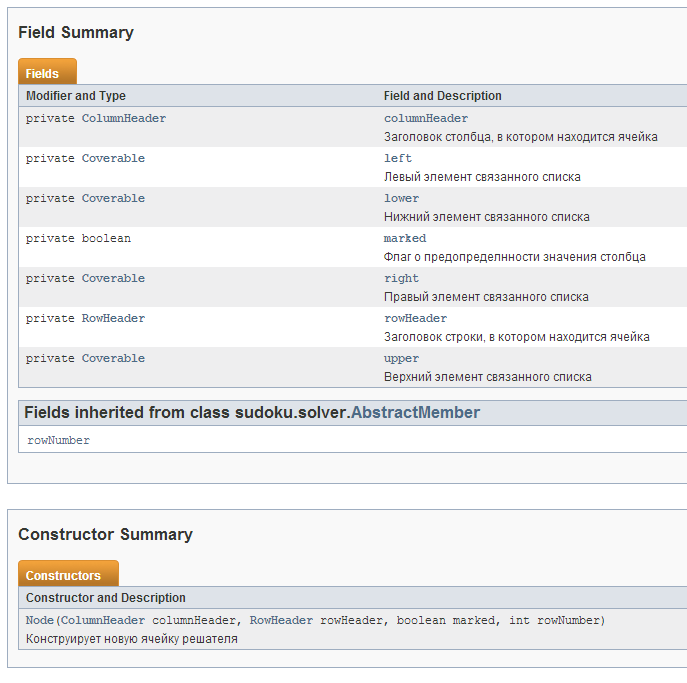


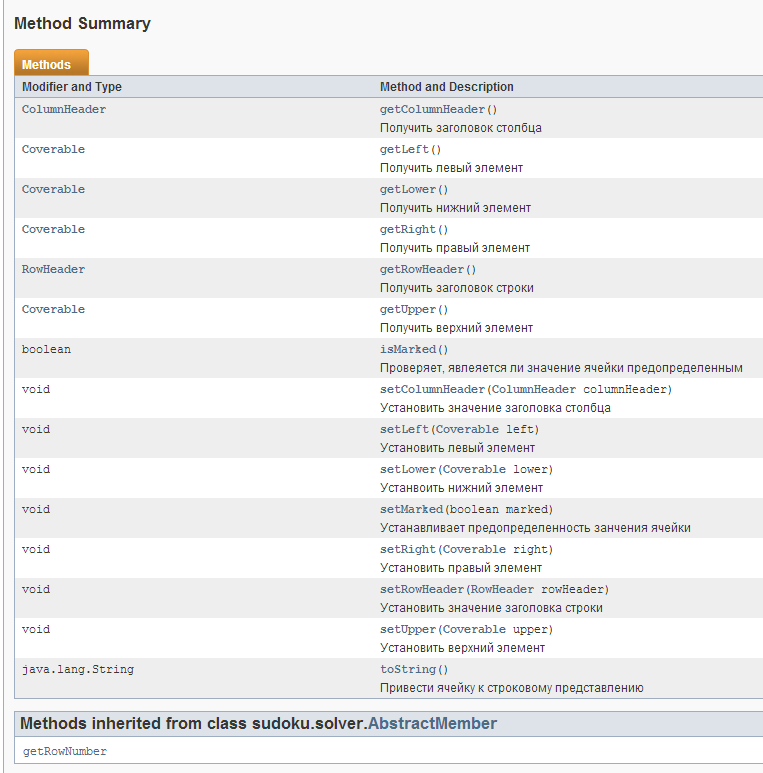
Класс ExactCover:



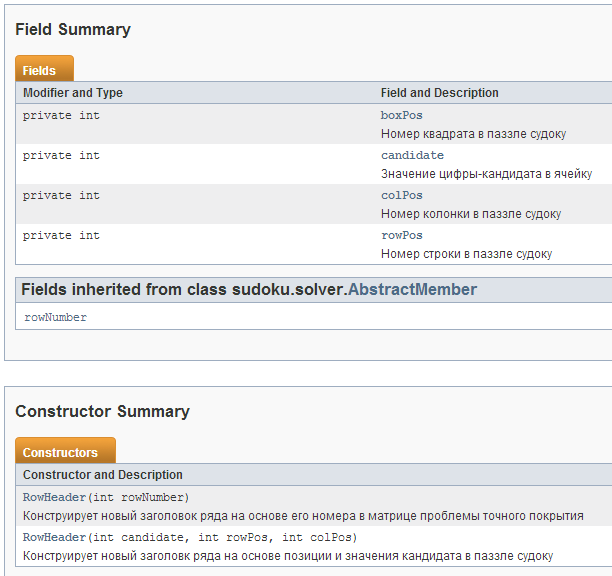


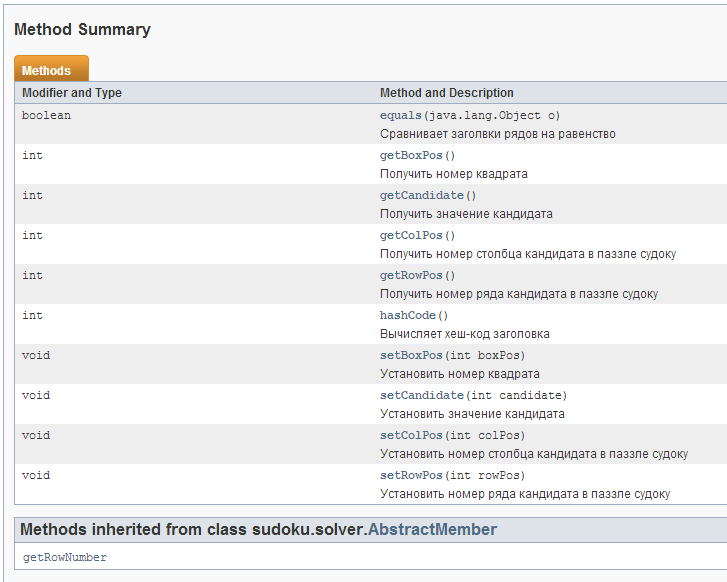
Класс Node:



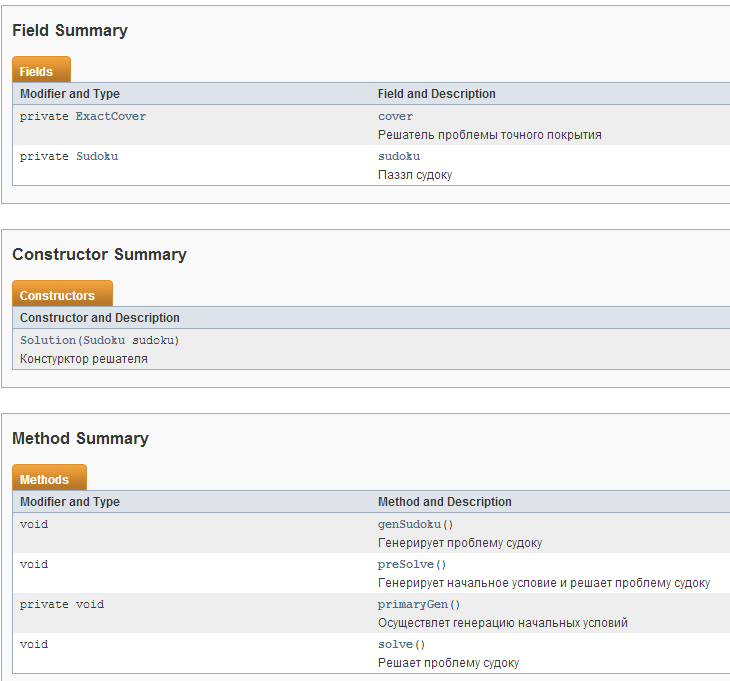


Класс RowHeader:

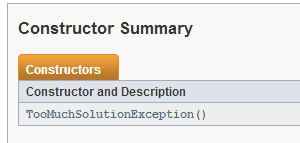




Класс Solution:



Класс TooMuchSolutionException:



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | | | | | | | | | | | | | | | | | |  |
|  | Лист регистрации изменений | | | | | | | | | | | | | | | | | | |  |
|  |  |  |  | | | | | | |  |  |  |  |  |  |  |  |  |  |  |
|  | Изм. |  | Номера листов | | | | | | |  | Всего   листов  (страниц)   в докум. |  | № докум. |  | сопроводительного   докум. и дата |  | Подп. |  | Дата |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | ненных |  | ненных |  | новых |  | анну-  лиро-  ванных |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

изме-

заме-