PCDB SuDoku Solver

Android Version

User Guide

This Document describes the structure and operation of the Android app PCDB SuDoku Solver. This app runs on Android tablet devices and enables you to:

- Set up and save an instance of a Killer SuDoku or SuDoku game
- Load an instance of an already-saved game
- Find a solution to the game, using hints and tips provided by the app, or
- Solve the game yourself, using tips and information provided by the app
- Save the solution for future use

0 DOCUMENT CONTROL

0.1 DOCUMENT INDEX

0.2 DOCUMENT HISTORY

Date	Version	Reason
21/08/2008	0.A	First Issue
17/11/2011	3.8	Reissue
23/03/2012	4.10	Reissue

0.3 ACKNOWLEDGEMENTS

There are many sites on the internet that explain the rules of SuDoku and how to solve SuDoku games. These rules are summarised in Section 3 below.

There are rather less sites devoted to Killer SuDoku. Amongst the sources used by the author are the following.

0.3.1 Nate Dorward

A number of Blogs by Canadian music reviewer and blogger **Nate Dorward** in 2006 were devoted to devising some highly complex and difficult Killer games, and explaining how to solve them. Many of Nate's followers said that these games took them six hours or more to solve. Nate documented a number of techniques that could be used to solve even the most difficult of them, and all of those techniques are implemented in this app.

Unfortunately Nate's blogs on Killer SuDoku are no longer available on the web.

0.3.2 John-Cristophe Godart

John-Cristophe Godart was a correspondent of Nate Dorward, and still blogging on Killer SuDoku in 2009. He has defined a set of nottion for Killer SuDoku, some (but not all) of which is followed in this Document.

0.3.3 Killersudokuonline.com

This website publishes daily and weekly games of varying degrees of difficulty. The author regularly tries the weekly "Mind bending" ones with this app and unfortunately has to admit, with what grace he can muster, that there are some that it can't yet solve. Merde!

The **killersudokuonline.com** website also contains explanatory material on some of the earlier games, which describe some techniques that have been implemented in the app.

0.3.4 Sudopedia.org

This is a free compendium of information about SuDoku. There is some, but very little, mention of Killer SuDoku.

0.3.5 The-Times.co.uk

The Times, among other newspapers, publishes daily Killer SuDoku games, and these were the instigation for the development of this app. To date, it has solved every Times game that has been tried...

The Times games are also available via its website, but this is behind a paywall.

1 INTRODUCTION

This Document describes the structure and operation of the Android app PCDB SuDoku Solver. This app runs on Android tablet devices and enables you to:

- Set up and save on your device an instance of a Killer SuDoku or SuDoku game
- Load an instance of an already-saved game
- Find a solution to the game, using hints and tips provided by the app, or
- Solve the game yourself, using tips and information provided by the app
- Save the solution for future use

1.1 BACKGROUND

SuDoku is a game that uses a 9 x 9 grid, split into nine 3 x 3 "boxes"[1]. The aim of the game is to place the digits 1 to 9 into the grid so that each digit occurs once and once only in each Row, Column and Box.

In **SuDoku**, some digits are already included in the grid, and the objective is to use these to fill in the remaining Cells.

In **Killer SuDoku**[2], no digits are included in the game definition. Instead, the grid is partitioned into a number of "groups" of up to nine touching Cells[3], represented by a dotted line enclosing the Cells or by use of different colours for each Group. No digit can be repeated within such a Group[4], and hence the size of a Group is limited to nine Cells. The sum of all the digits in the Group is given, usually printed in the top left-most Cell in the Group. The objective is to use these Groups, and their sums, to determine the digits in each Cell, subject to the standard SuDoku rules about each value being unique in each Row, Col or Block.

1.2 APP OBJECTIVES

The app enables you to solve most Killer SuDoku or SuDoku games, and provides support in terms of recording the options and candidates left at any point, and assistance in solving some part of the game so that you can make progress.

It aims to solve games using strict logic. A well-formed game has only one solution. Less well formed games may have two or more solutions that satisfy the initial criteria, and the app won't be able to solve these, though it will enable you to make a choice between viable alternatives.

---

1 *Sometimes known as a "nonet".*

2 *Killer SuDoku is the name used by The Times newspaper in the games that it has been publishing since 2006. In Japan this SuDoku variant is known as* Samunamupure *or* Sum-SuDoku.

3 *The term"Cage" is commonly used for what PCDB SuDoku Solver calls a "Group". Ideally we would use the common term, but the app uses similar technology to group together unadjacent Cells as part of its solving techniques, and the term "cage" is inappropriate for a collection of non-adjacent Cells.*

4 *Early Times games did not insist on this rule, and at least one game can only be solved with a duplicated digit within a Group. The Times later clarified the rule, and virtually all Killer SuDoku games providers adhere to the "no repeated digits" rule.*

1.3 INSTALLATION

PCDB SuDoku Solver is available from Google Play (previously the Android Market), for a small fee, and is installed on your device when you pay the fee and download the app. Having purchased the app, you can re-download it to that device as often as you like, for example to pick up new a version.

1.4 CONSTRAINTS

PCDB SuDoku Solver is designed to make a good deal of information available to you while playing a game, and it isn't possible to display all of this in a readable manner on a small screen. Thus it is designed to run principally on tablet-type devices, and will not run on a device that has a screen resolution smaller than 1024 x 600. This limits it to devices with a 7" screen or larger. It operates in landscape mode, and while it will invert itself if the device is turned upside down, it will not switch to portrait mode.

1.5 MODE OF OPERATION AND INPUT DEVICES

The app is designed for use on a tablet-type device with no input devices other than a touch screen. A Android "virtual keyboard" is displayed in some circumstances, and these are described with examples where applicable. The main part of the screen is occupied by a 9x9 grid, and you can select any Cell on this by touching it. The other main input need is the ability to specify numbers in the range 0 to 9, an for this a Number Pad is permanently displayed as part of the UI of the app.

1.6 LIMITATIONS

The app uses a range of Android facilities, including some that were introduced only in Android Version 3.2 (the baseline used by the Samsung Galaxy Tab 10.1, which is the device for which it was developed). Some of the facilities in that baseline are not yet fully mature, and don't yet offer all the facilities that they should.

1.7 ANDROID PERMISSIONS

The app uses a number of Android features which are recorded as "permissions" and which you will have to acknowledge and accept when downloading it.

1.7.1 Use of Android External Filestore

It creates Game Definition Files within your device's external filestore as you input the definition of a game. Thes are visible outside the app, and can be displayed, examined and manipulated by the **My Files** app.

1.7.2 Use of the Internet

Help facilities are included within the app, and are invoked if you touch the **Help** button on the Action Bar. Although these don't in themselves use the Internet, there is a link to the PCDB Dev support website that does.

1.8 OTHER FEATURES

1.8.1 Use of USB

Although it doesn't use the USB host or accessory directly, using USB to connect to a PC enables you to download predefined games and upload games or log files to your PC. These files are compatible with those created by the Windows 7 version of the app, so you can use a USB connection to a PC to copy games to or from your device.

1.9 COPYRIGHT

PCDB SuDoku Solver is intended to help you solve Killer SuDoku and SuDoku games printed in various newspapers and websites. These are normally subject to copyright restrictions which you should make yourself aware of and abide by.

1.10 TERMS AND ABBREVIATIONS

A number of technical terms related to the games of SuDoku and Killer SuDoku are used in this document. These terms are defined here. When used in the body of the text, these terms are Capitalised.

A number of websites, including www.Sudopedia.org, define a nomenclature for Killer SuDoku. The terms used in this document differ in some ways from that. In particular, PCDB SuDoku Solver uses the term "Group" where Sudopedia uses "Cage". The reason for this is that a Cage is a term that applies to the visible partitioning of the Grid into sets of one or more Cells. PCDB SuDoku Solver also creates Groups for the sets of Cells identified by searching for Innies or Outies, or splitting other groups. These are not necessarily contiguous, and "Group" seems a better term that "Cage" for these

Added Group	A Group which is defined during the solving phase of a game, for example one that defines a set of Innies or Outies. An Added Group that extends across more than one House can have repeated instances of the same digit, so long as the instances are all in separate Houses.
Board	The 9x9 grid on which the game is played. This is split into nine 3x3 Boxes.
Box	The 9x9 board is subdivided into nine 3x3 Boxes, each comprising nine Cells. Boxes are numbered 1 to 9 thus: 123 456 789
Candidates	A string of up to nine digits, in the range 1 to 9, that represents the remaining possibilities for the solution in a given Cell. Candidates are conventionally represented in curly brackets thus: {356}. At the start of a Killer SuDoku game, the Candidates for every Cell are set to {123456789}. At the start of a SuDoku game, these Candidates are reduced by the presence of a preset digit in certain Cells in the same Row, Col or Box. The iterative analysis of the board, either by the user or by the program, reduces the set of Candidates further until, when the game is solved, there is only one Candidate left for each Cell.
Cell	One of the 81 spaces on the 9x9 Board. Cells are identified by their Row and Column, in tht order, separated by a comma and enclosed in brackets. Thus (3,4) represents the Cell at Row 3 Column 4.
Child	If a Group is split into two Subgroups, each is a Child of the containing (Father) Group. The Subgroups are Siblings of each other. The sums of the two Children must add up to those of the Father, so if they have sum ranges, rather than fixed sums, any reduction in the range of one Child can be used to make a corresponding reduction in the range of the other.

Col	A column of nine Cells, numbered starting from 1 at the left to 9 at the right
Complementary Groups	A solving technique. If two Groups are both restricted to the same two Houses, and have any Mandatories in common, those Mandatories can be removed from the rest of the Cells in those Houses.
Complex Elimination	A solving technique. If a Group can be treated like a doublet, in that it must contain either one of two Candidates, then that Group can be treated as a single Cell in a search for a similar Cell within the same House, which may enable you to eliminate other Candidates from that Cell.
Conjugate Chain	A solving technique. Find instances where two Candidates are in only two Cells in a given House. These have a "conjugate relationship"; one or other of the Cells must contain the value. The technique tries to follow a chain of such conjugate links, "colouring" the Cells alternately with two colours. One colour marks the "true" Cells, which are assumed to hold the value. The other marks the "false" Cells that don't. If a case is found where two Cells in such a chain in the same House have the same colour, then that must be the "false" colour and the value represented by that colour can be eliminated from the Cells marked with that colour.
Conjugate Group	A solving technique. Look for any House which is solely occupied by two Fixed Groups, where one of these Groups has Cells outside the House. If any single Cell in one Group is outside the House, then the Candidates in that Cell must be in the other (Conjugate) Group *within* the House. If only one Cell in the Conjugate Group has any of these Candidates in it, then remove any other Candidates in the Cell
Constrained Group	A Group which is either a Fixed Group, or which lies entirely within a single House. No digit can occur more than once in a Constrained Group
Doublet	A Doublet occurs where a House or Group contains two Cells each of which has the same two values as their Candidates, and those values occur nowhere else in the House or Group. It is a Naked Doublet if there are no other Candidates in those two Cells, and a Hidden Doublet if there are other Candidates.
Father	When *SplitGroups* splits a Group into two Subgroups, the two resulting Groups are defined as Siblings of each other and share the same Father. The sums of the two Siblings must add up to that of the Father, so if the Siblings have sum ranges rather than a fixed sum, any reduction to the range of one Child can be used to make a corresponding reduction to the range of the other.
Fixed Cell	In a SuDoku game, a value that is given for a Cell as part of the definition of the game.
Fixed Group	A Fixed Group is one that is shown on the grid that defines the game. Every Cell must belong to one and only one Fixed Group. No digit can occur more than once within a Fixed Group
Forcing Chain	
Game Definition File	A text (.txt) file that contains a definition of a SuDoku or Killer SuDoku game. When you use the app to input details of a game, these details are stored in a Game Definition File which you can load over and over again.
Group	A set of Cells for which the sum of all the values is known, either explicitly or as a Range. The Fixed Groups are those that define the shape of the Game, and no digit can occur more than once in a Fixed Group. Each Cell in the board belongs to one Fixed Group. Added Groups overlay these Fixed Groups. There is no requirement that all the digits in an Added Group are different, but the number of times a digit can occur is defined by the number of Houses that the Group occupies Note that Sudopedia and other sources use the term "Cage" instead of "Group", but as mentioned above "Group" is a better term in cases where an Added Group contains Cells that are not necessarily contiguous.
Group Bounds	These define respectively the range of Rows, Cols and Boxes that contain the Group. If the Group is confined to a single Row, Col or Box then it is said to be Constrained, and every value in the Cells in the Group must be unique. The Fixed Groups which form the game definition must have unique values, by definition. An added Group, however, can cross a House boundary, and it is possible for the Group to contain as many duplicate values as the number of Houses it occupies.
Group Pattern	This refers to a process whereby all the possible combinations of Cell values are calculated using the Cell's Candidates and the Group's Options. Any Candidate that does not

	contribute to one of the possible patterns can be eliminated.
Hidden Doublet	A Hidden Doublet occurs where two Candidate values occur in only two Cells within a House or Group. Other Candidates in these Cells can be eliminated
Hidden Quad	A Hidden Quad occurs where four Candidate values occur in only four Cells within a House or Group. Other Candidates in these Cells can be eliminated. Not all the Cells need to contain all four values
Hidden Quin	A Hidden Quin occurs where five Candidate values occur in only five Cells within a House or Group. Other Candidates in these Cells can be eliminated. Not all the Cells need to contain all five values
Hidden Triplet	A Hidden Triplet occurs where three Candidate values occur in only three Cells within a House or Group. Other Candidates in these Cells can be eliminated. Not all the Cells need to contain all three values
House	A Row, Column or Box of nine Cells
House Set	Two or more adjacent Houses
Innie	An Innie is a Cell within a House or House Set that belongs to a Group which is partly, but not entirely, within that House or House Set. Because the sum of the Cells in the Houses is known, and so is the sum of the Cells that are entirely within those Houses, then the range of values in the Innies can be calculated. If there is only one Innie, its value can be determined there and then.
Level	A degree of difficulty as stated by the Source of a Game. Different sources use different sets of levels, with different meanings. For example, the Times uses the levels Easy, Midl, Difficult, Fiendish and Super Fiendish for its SuDoku games, and Gentle, Moderate, Tricky, Tough and Deadly for its Killer SuDoku ones.
Mandatory	Of a Group, one of more values that occur in each of the Group's remaining Options. For example, if the Options or a 12(4) Group are [1236,1245], then the values {12} are its Mandatories
Naked Doublet	A Naked Doublet occurs where two Cells in a House or Group contain the same two Candidates and no others. Each of these Candidates must occur in one or the other of these two Cells, and can be removed from other Cells in the House or Group
Naked Quad	A Naked Quad occurs where four Cells in a House or Group contain the same four Candidates and no others. Each of these Candidates must occur in one or another of these four Cells, and can be removed from other Cells in the House or Group
Naked Quin	A Naked Quin occurs where five Cells in a House or Group contain the same five Candidates and no others. Each of these Candidates must occur in one or another of these five Cells, and can be removed from other Cells in the House or Group
Naked Triplet	A Naked Triplet occurs where three Cells in a House or Group contain the same three Candidates and no others. Each of these Candidates must occur in one or another of these three Cells, and can be removed from other Cells in the House or Group
Options	This is a set of comma-separated strings that represent all the different ways that a Group can be populated. Options are conventionally represented by a set of comma-separated digit strings in square brackets, so that for example a 11(2) Group has Options of [29,38,47,56]. Each digit string is the same length as the number of Cells in the Group.
Outie	An Outie is a Cell that is outside a House or House Set but belongs to a Group which is partly within that House or House Set. The sum of all the Outies can be calculated and can be used to help determine the range of values in the Outies. If there is only one Outie, then its value can be determined straight away.
Pair	A Pair is a set of two values, one or the other of which occurs in every Option of a Group. If a Group has one or more Pairs, and lives completely within a single House, the Pair can be treated as a two-value Cell in a search for a Doublet comprising the Group and one other Cell in the same House
Peer	Of a Group, another instance of a Group which comprises the same Cells, but has a different Father. All Peers have some properties in common, such as their Sum range and Options, and if a change in these properties is made to one Group it should be propagated to all its Peers
Quad	A Quad occurs where a House or Group contains four Cells each of which has the same four values as Candidates, and those values occur nowhere else in the House or Group. It is a Naked Quad if there are no other Candidates in those four Cells, and a Hidden Quad if

	there are other Candidates
Quin	A Quin occurs where a House or Group contains five Cells each of which has the same five values as Candidates, and those values occur nowhere else in the House or Group. It is a Naked Quin if there are no other Candidates in those five Cells, and a Hidden Quin if there are other Candidates
Row	A row of nine Cells, numbered from 1 at the top to 9 at the bottom.
Sibling	When a Group is split into two Subgroups, the two resulting Groups are defined as Siblings of each other, and the containing Group as their Father. Although the two Subgroups may have sum ranges rather than fixed sums, these ranges are complementary in that the eventual sums must add up to those of their Father. Thus, as the range of one Sibling is reduced, a corresponding reduction can be made to that of the other Sibling.
Signed Group	A Signed Group is created where there is an Innie/Outie combination such that the sum of the values in one set of Cells (the Positive Cells), minus the value in another Cell or Cells (the Negative Cells), has a known value.
Singlet	A Singlet occurs where a digit occurs in only one of the set of Cell Candidates in a given Row, Col or Box. By definition, that Singlet must be the solution for that Cell, and can be removed from the Candidates of all other Cells in the same Row, Col or Box.
Solved	A game is said to be Solved when every Cell has either been filled in by the user, or there is only a single Candidate left for it.
Split Group	A large Group can be split into two smaller Subgroups, each of which has a variable sum but with the relationship that the sums of the two Subgroups add up to the sum of the larger (Father) Group.
Subgroup	*Split Groups* looks at each Group and tries to split it into two Subgroups, for example along House boundaries. Each is a Sibling of the other. The sums of the two Subgroups must add up to that of the father Group. If it is only possible to assign a sum range to the two Subgroups, then any reduction in the range of one Subgroup can lead to a corresponding reduction in the range of its sibling.
Subtraction Combo	A process whereby the discrete Candidates in the Cells in a Group are summed and the Group's sum subtracted from the result. If a unique set of Candidates that adds up to the difference can be found, then those Candidates can be eliminated from the Cells in the Group
Sum Range	In a Fixed Group, the sum that the Cells' values must add to is fixed. Where a Group is split, it may not be so clear what the sum of a Subgroup may be, but it will be clear what range it will lie in. At the simplest, this Sum Range is the range from the smallest sum that could be made with the number of Cells in the Subgroup, to the largest. A 4-cell Group, for example, has a range of 10..30. However, that will only be the case where each Cell in the Subgroup has a full set of Candidates {123456789}. As the Candidates in the Cells are reduced by various techniques, the Sum Range of the Group can be reduced accordingly.
Swordfish	A solving technique
Triplet	A Triplet occurs where a House or Group contains three Cells each of which has the same three values as Candidates, and those values occur nowhere else in the House or Group. It is a Naked Triplet if there are no other Candidates in those three Cells, and a Hidden Triplet if there are other Candidates.
Unsigned Group	A Group in which all the Cells contribute to the sum, or range, of that Group.
X-Wing	A solving technique
XY-Wing	A solving technique

2 USING THE APP

2.1 OVERALL STRUCTURE

When you load **PCDB SuDoku Solver**, the form shown below is displayed. This form handles the major functionality of the app.

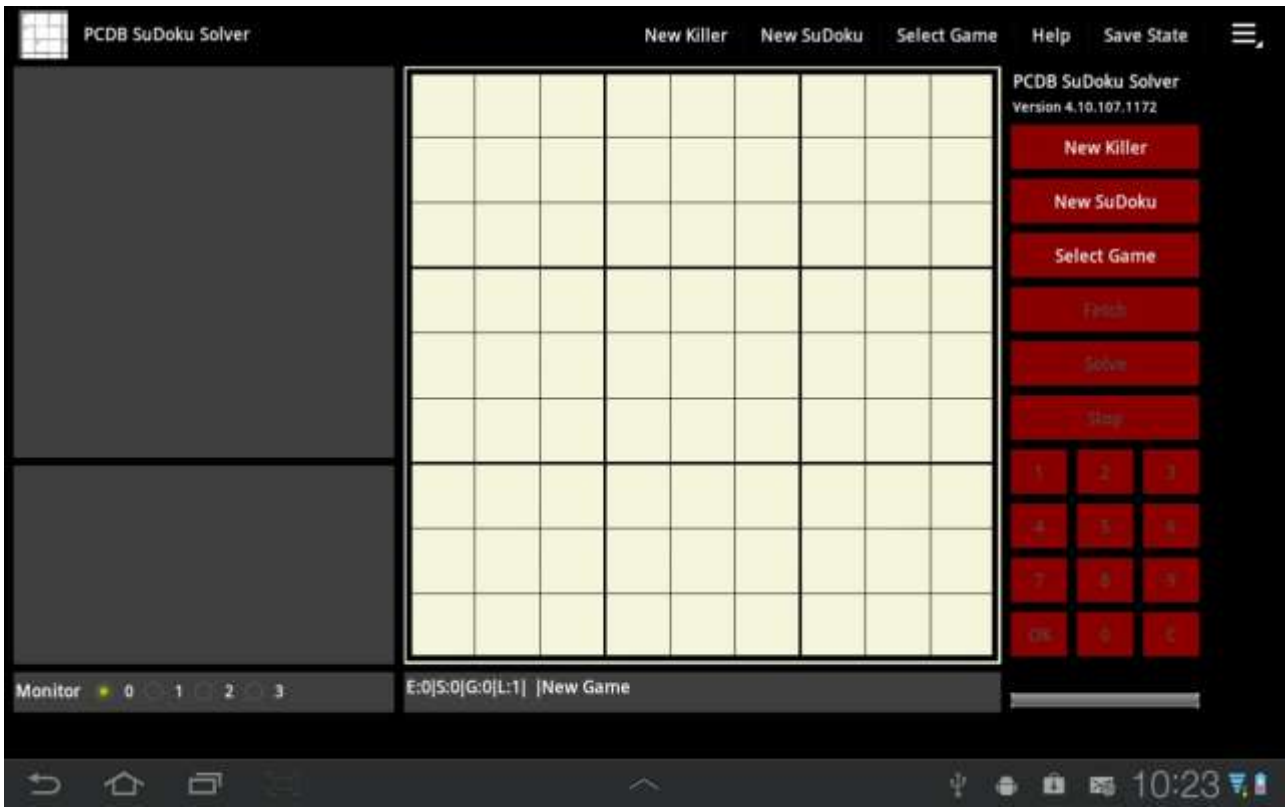

Figure 1

The areas of the screen are used as follows.

2.1.1 Status Bar

The Android Status Bar is located at the top of the screen. It shows the app logo, and some of the most important menu options.

Figure 2

Touching the "menu" icon on the right hand side gives access to some more options:

Figure 3

- *Solve* – This, if enabled, tells the app to attempt to solve the game, and is equilavent to the "Solve" button on the main screen.
- *Restore Game* – This will reload any game whose state you have saved by using the Save State entry on the menu, or by stopping a game you haven't completely defined or solved.
- *Left-Handed*. The main app layout is designed for right-handed people, who use their right hand to select options on the screen. If you are left-handed, you can select the "Left-Handed" option. You will have to reload the app, but when you do it will display reversed, like this.

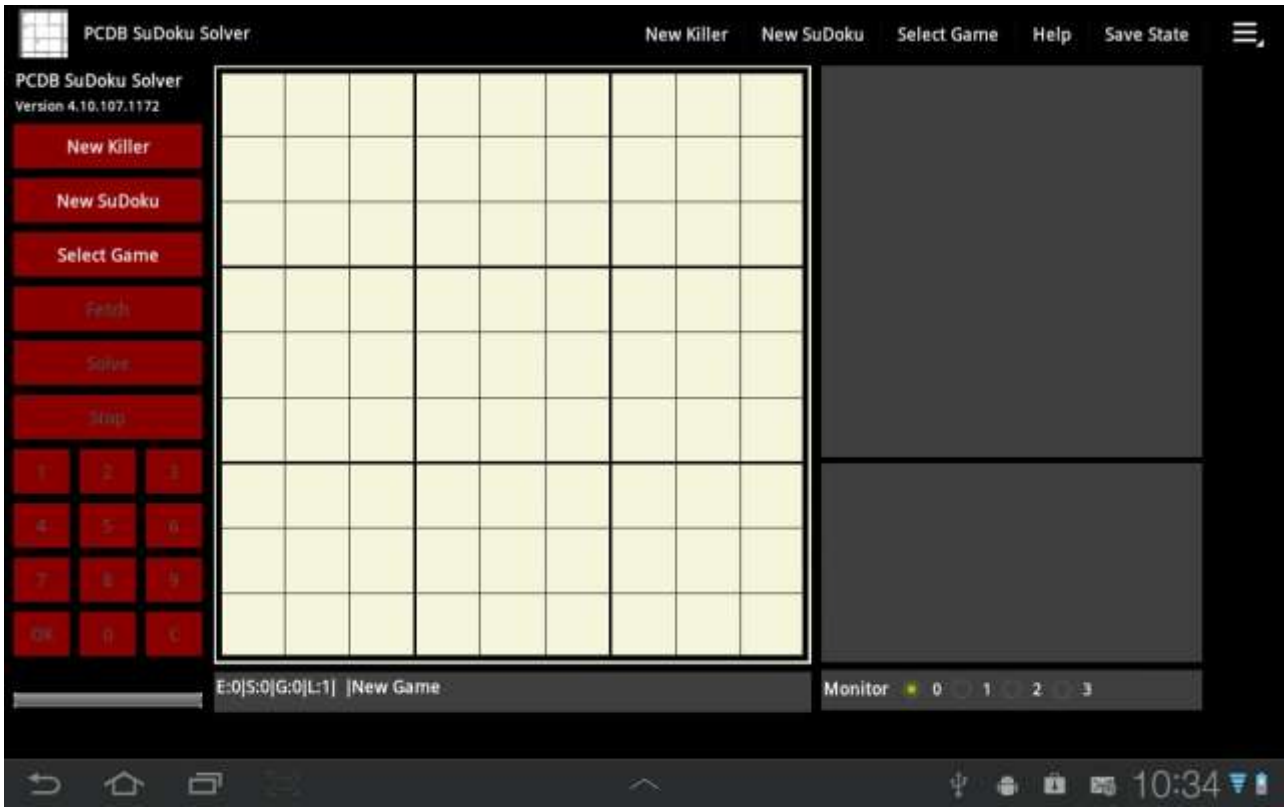

Figure 4

It will load like this until you click on the same menu item (now called "Right-Handed").

The rest of this document shows screen shots from the Right-Handed version.

There are two other memu items available, not shown in the screen shot above.

- *Reset Preferences* – this resets all lists of Copyrights, Sources, and memories of these used by *Save Game* and *Select Game*

- *Clear Logs* – this deletes any logs files generated by solving games with a Monitor Level of 2 or above.

2.1.2 App Identification

The app identification shows the name of the game, and its current version.

Figure 5

The version you see may be later than those shown in the screen shots in this document.

2.1.3 Game Grid

The game grid is a 9 x 9 grid which comprises the playing surface for the game. The grid is split into nine Boxes, each comprising 3 x 3 Cells.

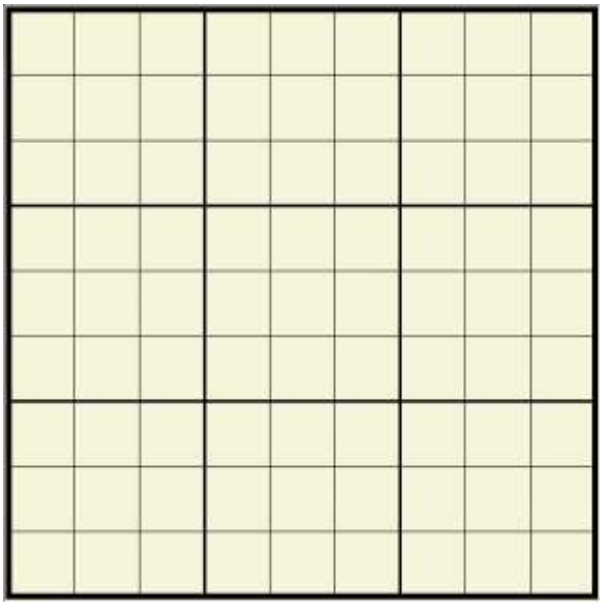

Figure 6

2.1.4 Action Line

The Action Line at the bottom of the screen is used to indicate the result of number button presses during the definition or play of a game.

Figure 7

2.1.5 Group Definitions Area

The Group Definitions Area is located to the right of the Game Grid. It is only for use in Killer SuDoku games, and lists the Groups defined as part of the game instance, and those added later by the analysis of the game. Here is an example.

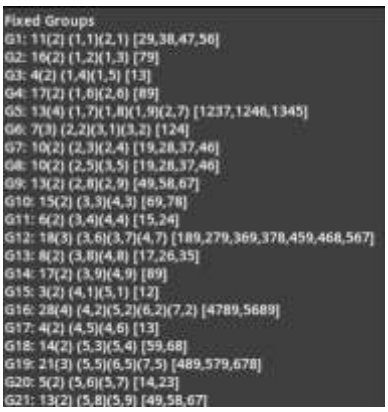

Figure 8

You can scroll down this list by flicking it upwards.

2.1.6 Control Buttons

A number of buttons are shown on the uppermost left part of the form. Those that can be used at any one time are enabled; the others are disabled and their captions are shown in faint. Some of these mirror the main menu options shown in the Status Bar (see above). But the content and meaning of each of the Control Buttons can vary according to the current game type and state. The initial set of values is as shown here:

Figure 9

The options that you can use at any given time are shown enabled, i.e. displayed in white text. Disabled controls have their text shown in black.

2.1.7 Monitor Controls

At the lowermost left part of the form is a control that enables you to monitor the progress of the analysis part of the game.

Figure 10

The app can run at a number of monitor levels. The current level is indicated by a marker next to the relevant entry; in this case **0.** The monitor level defines the amount of monitoring carried out. Values are:

Level	***Information Monitored***
0	None

1	When **Solve** is pressed, the app stops after any change to Candidates or Options with a pop-up screen explaining the change. You can then choose how often the app stops again – after every change, after each different process completes, after each Cell is solved, or at the end of the game.
2	A log file is created for each Game, and all changes to Candidates or Group Options are written to this file. Information displayed at Level 1 (see above) is written to this file.
3	Additional data is written to the monitor file

The default setting is 0. You can change the level by touching the number that corresponds to the level you want. This level is remembered and reinstated when you next load the app.

If you use Monitor level 2 or above, you *must* use the "Clear Logs" menu item periodically to clear out the log file folder.

2.1.8 Status Line

A status bar at the bottom of the screen displays a number of aspects of the game's progress.

Figure 11

2.1.8.1 Excluded (E:nnn)

This shows the number of Candidate options currently excluded. The number starts at zero, and when it reaches 648 the game is completely solved.

2.1.8.2 Solved (S:nn)

This shows the number of Cells for which the value is known. It includes those that are preset in an instance of a SuDoku game. When this count reaches 81, the game is solved.

2.1.8.3 Groups (G:nnn)

This shows the number of Groups created during the setup and analysis of the game.

2.1.8.4 Level (L:n)

When the app is asked to solve a game, it progresses through a number of levels, with a number of increasingly powerful but esoteric processes being introduced at each level. The current level, in the range 1 to 4, is shown here, and thus effectively shows the complexity of the game.

2.1.8.5 Solution Exists (S)

Where the definition of a game includes its solution, a capital "S" is displayed to the right of the "Level" indicator. You can't see the solution, but if you try to eliminate a Candidate that is actually the solution for a given Cell, the app will warn you and refuse to make that change. Thus if there is a known solution, you won't be able to mess up the game. If there isn't, you will be able to make a mess of it and may, at some stage, receive an error message which tells you how messed up it is.

2.1.8.6 Game Name

The right-most part of the status bar shows the name of the last loaded game, or “null” if none has yet been loaded.

2.1.8.7 Progress Bar

The right-most part of the status line is a progress indicator that shows the increase in the number of excluded Candidates from 0 to 648.

2.1.9 Number Pad

A set of number buttons is displayed on the side of the form, underneath the Control Buttons. This enables you to enter values into the grid by touching one of these buttons as well using a finger or stylus.

Figure 12

The Number Pad also offers the facility, when solving a game, to highlight all the Cells that contain a particular Candidate value. Thus, pressing “3” will highlight all the Cells for which 3 is still an available Candidate.

Tne Number Pad includes three other buttons, used as follows.

- **OK**, which is used to complete an operation such as specifying the sum in a Group
- **0**, which doubles as the ¬ button when this is needed. 0 is used when defining a SuDoku game to remove a value which you have added in error. ¬ (the *not* sign) is used before a numeric string to indicate that the following digits should be *removed* from the Candidates in one or more Cells
- **C,** which is used to “backspace” when entering a numeric value, and to undo a number of other functions

2.1.10 Information Area

On the other side of the form, underneath the Group Definitions Area, is an area used to display information about the current progress of the game.

- When a Cell, or set of Cells is selected, it displays the list of Cells
- When a House or set of Houses is selected, it displays the names of these Houses
- When a Group is selected, it displays the Group name, the Cells that comprise it, and other information including the current Options for that Group, and Pairs and any Mandatories for it
- When a play is made, it displays the number of Candidates eliminated as a result of that play.

Figure 13

2.2 DEFINING OR LOADING A GAME

Killer SuDoku and SuDoku games are published in various newspapers (including The Times) and books, and can be found on the internet, for example at killersudoku.com. You can enter the definitions of these games into the app, and store them for future analysis.

Any such games will have a copyright owner, and it is your responsibility to recognize and record the owner of any game that you store in this way.

You can start recording the details of a game you have discovered from one of these sources by touching the **New Killer** or **New SuDoku** buttons. Alternatively, you can reload a game you have previously stored by touching the **Select Game** button.

3 SUDOKU

3.1 GAME CONCEPTS

In SuDoku, a game is defined by the presence of fixed values in some of the Cells in the Grid. The aim is to fill in the other Cells with values so that each digit in the range 1 to 9 occurs once and once only in each Row, Column or Box.

3.2 DEFINING A SUDOKU GAME

If you touch **New SuDoku**, the Grid is cleared and each Cell is set to contain all the possible Candidates 1 through 9. These are shown in small text at the bottom of each Cell.

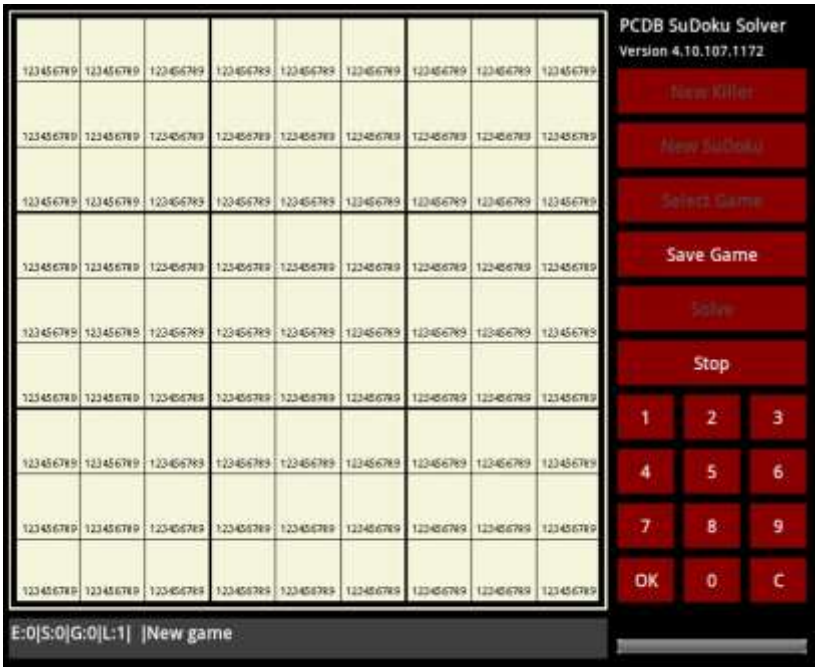

Figure 14

For each fixed digit in the printed game, touch the Cell in the Grid. The Cell turns red. Then touch the appropriate digit in the Number Pad. The digit you touched is then shown in the Cell, in black.

For example, touch Cell (1,2), then touch “6”. The Grid is updated as shown here.

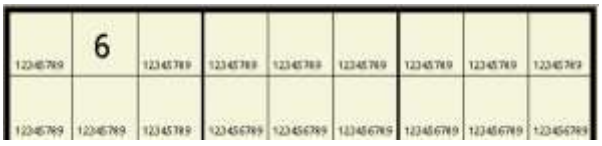

Figure 15

Note that Candidate {6} has been removed from every other Cell in Row 1, in Column 2 and in Box 1 (Cells (1,1) to (3,3)). Every time you specify a fixed digit in a Cell, it is removed from the Candidates in every affected Cell.

Continue selecting Cells and their contents until you have completed the definition of the game, for example as shown here.

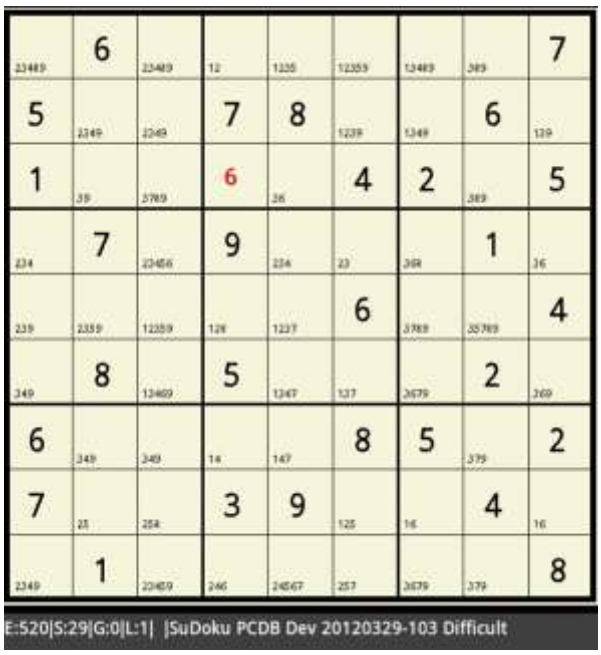

Figure 16

This screen shot shows the progress area at the bottom of the Grid. It shows that the game includes 29 specified Cells, and that 520 of the total of 648 unwanted Candidates have already been eliminated. The aim now is to eliminate another 128 Candidates until there is only one Candidate left per Cell.

Before you can play the game, though, you have to save it, as described in Section 4.4 below.

With practice, you can define a SuDoku game in this way without any trouble. At first, you may make mistakes such as entering the wrong value in a Cell. For example, if you tried to specify a value of 6 in Cell (3,1), rather than the 5 shown in the example above, it displays the following transient error message.

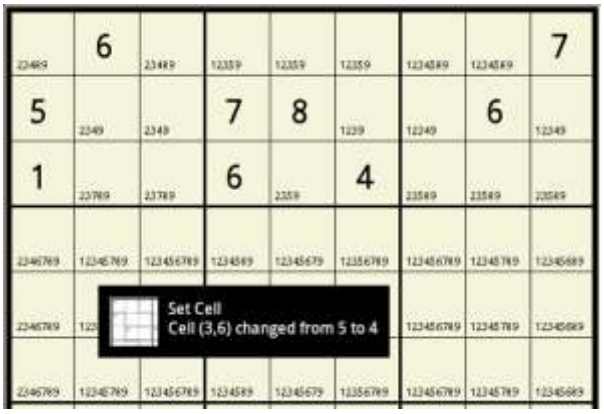

Figure 17

If you realize that you have entered the wrong value in a Cell, go back to it and type in the correct value. For example, if you enter **5** in Cell (3,6) instead of 4, go back to the Cell and type **4**. The Grid is updated and you will receive the following message:

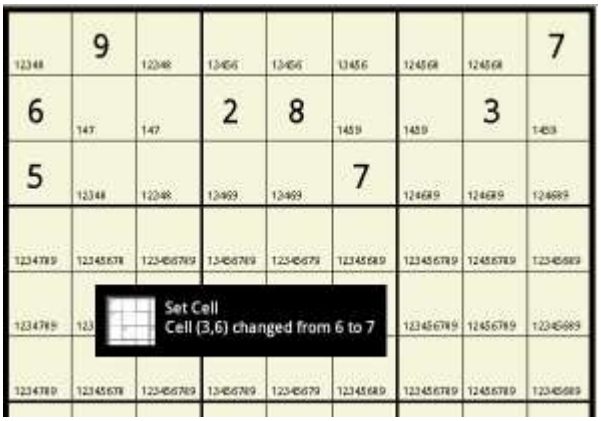

Figure 18

If you set a value into the wrong Cell, you first need to remove it from that Cell. Go back to the Cell, and touch **0**. Then select the correct Cell and enter the value there.

3.3 SAVING A GAME DEFINITION

When you have specified all the fixed values in the game, touch **Save Game**. You are asked to specify some details of the game. These are recorded in the Game Definition File, and are also used to set up its default name. You can accept this name, or overwrite it if you want. An Android Virtual Keyboard is displayed to help you enter text. (This may look different on your device.)

Figure 19

You can dismiss the keyboard (the appearance of this may differ according to your device) to show the complete form thus.

Figure 20

Note that a number of fields on this form are followed by a red button with a caption of"+". These provide access to a drop-down list of options, for example of levels of difficulty. A number of values are preloaded for each of these properties, and you can display the list by touching the field where the existing value is displayed. For example, if you touch the "Source" field, a list of game sources is displayed thus:

Figure 21

Touch the entry you want to select.

If none of the entries in this list are relevant to your game, you can add a new entry to the list by touching the original text field and then touching the red "+" button to the right of the field. The display changes thus:

Figure 22

Touch the field to bring up the virtual keyboard, and then type the name of the new entry. Then touch "Next" or "Done" on the virtual keyboard. The new entry is stored and appears as the selected entry. For example if you type "The Daily Bugle" and then touch **Next**, it is shown thus:

Figure 23

Note that the new entry has been incorporated into the Game's name.

The fields on this form are used as follows.

- *Source* - the origin of the game you are saving
- *Copyright* - the Copyright owner of the game you have defined, if known
- *Date* - the date the game was printed, in the format yyyymmdd
- *Number* - the publisher's number for the game
- *Difficulty* – this enables you to define the level of difficulty for the game, as specified by its publisher. The initial examples given are mostly those used by The Times, and there are different values for Killer SuDoku and SuDoku games
- *Minutes* – if given by the publisher, you can enter the number of minutes which it is assumed are needed to solve the game
- *Game Name* – this field contains the actual name with which the game will be stored. By default, it is left free for you to type in whatever name you want. However, if you want the name to be made up of a concatenation of most of the above entries, click on the green Check Box to the right of the name. A tick appears in the box, and is remembered for future calls on Save Game (though it is set "off" if you load a game whose name does not appear to be constructed from these properties). The content of the *Game Name* field is updated, and in future will change automatically as you update the other fields on the screen. You can still customise the game name, but **don't modify any of the other fields afterwards, or the Game Name will be reset**. Touch "Done" when you have finished.

  Here is what happens if you touch the "Auto Update" check box.

Figure 24

- *Folder* - allows you to specify the Folder that the Game is to be saved in. When you install PCDB SuDoku Solver, it creates two folders called **Killer** and **SuDoku**. You can choose one of these (the one you choose should be obvious) or create a new folder by touching the + key to the right of the field an then typing in the name of the new folder

  Note that any Folder you specify here is remembered and selected, by default, the next time you save or select a game. (Similarly, if you change the default folder in **Select Game** it will be displayed by default when you next save a game.)

- **Buttons** – are provided to confirm the Filename and save the Game with this name in the chosen folder (OK), or to cancel the save operation (Cancel). Note that if you touch **Cancel**, any new Sources, Copyrights, Difficulty levels or Folders you have set up remain.

3.4 LOADING A GAME

You can reload the game's Game Definition File immediately. After you save a game, the **Load** Game button text changes to **Reload**. Click on this button to reload the game you have just saved.

Figure 25

Alternatively, click on the **Stop** button. The **Reload** button changes back to **Select Game.** Touch this button. The screen displays the currently active folder, and a list of all the games in it, as here.

Figure 26

You can touch a game to load it, of touch-and-hold a name for more options:

Figure 27

All of these options do what they say. Note that the **Rename Game** option loads the game and then invokes the **Save Game** function so that you can specify not just the new game but the game properties

(source, copyright and so on). You can also specify a different folder. The game is saved with its new name (and folder, if relevant) and then the old game is deleted.

Once you have loaded a Game Definition File, you can either solve it yourself, with help from the app, or direct the app to carry out a full analysis of the game.

3.5 MODIFYING A GAME

Sometimes, attempting to solve a game throws up errors which indicate that you have set it up incorrectly. You can modify a saved SuDoku game, so long as you do so immediately after loading it, and you have not previously saved it with an embedded solution. Load the game. The **Save Game** button changes to say "Modify", as here.

Figure 28

If you touch this button, the game reverts to the state it was in immediately before you saved it. You can remove values from the Grid, or modify the value in a fixed Cell, as in Section 3.2 above. You can then re-save the Game at any point.

3.6 SOLVING A SUDOKU GAME

If you touch **Solve**, the app attempts to reduce the set of Candidates to one per Cell. As each Candidate is eliminated, it can display the change being made and the reason for it. You have to reset the Monitor Level 0 1 to enable this, otherwise the app just charges ahead and solves what it can before stopping.

In the example below, there is only one Candidate ({6}) left for Cell (3,4). So the app begins to eliminate all other 6s from the Candidates in the same Row, Col or Block as Cell (3,4). Thus Cell (3,5), whose Candidates were initially set to {36}, has them reduced to {3} and (because there is only one candidate left) this is now displayed in red. Similar eliminations take place in the remainder of Col 4, in Row 3 and in Box 2.

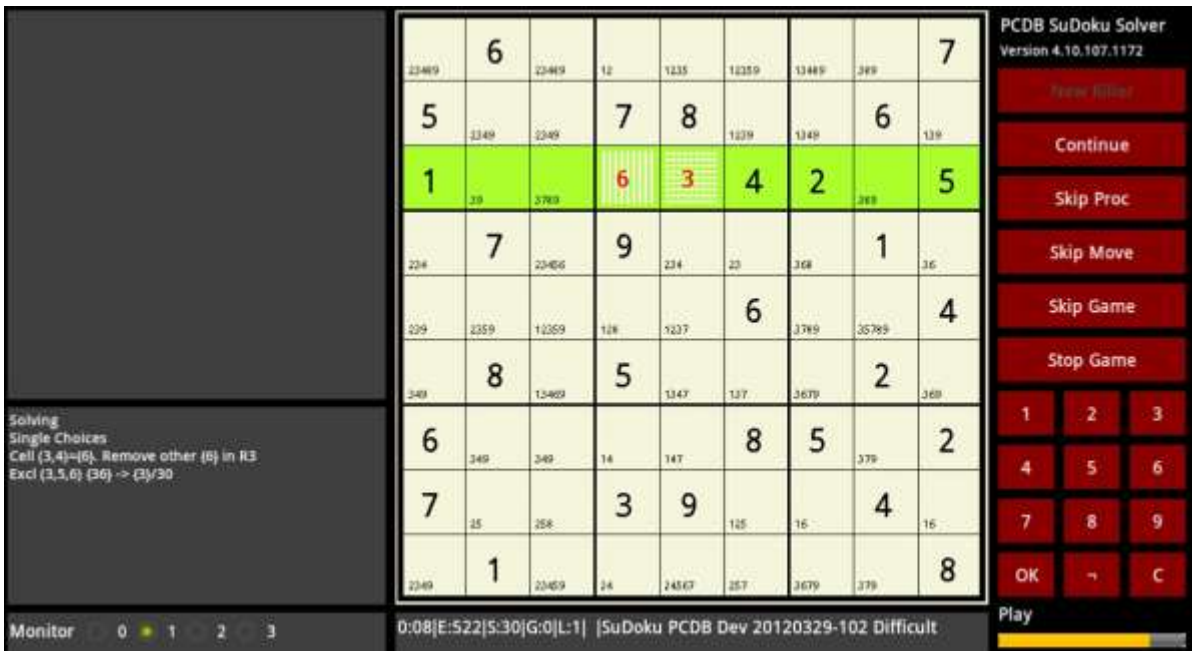

Figure 29

Note that:

- The scope of the change in this example is a House, Row 3 4. This is displayed with a green background, the conventional colour used to highlight a House.
- The Cell (3,4) that gives rise to the elimination has the background colour hatched with vertical white stripes.
- The Cell (3,5) at which the elimination actually takes place has the green background overlaid with small white squares

You have five options at this stage.

- Touch **Continue** and the app proceeds to and displays the next elimination.
- Touch **Skip Proc** and the app silently processes other eliminations made by this procedure (Single Choices) and stops when the first elimination is made by another procedure.
- Touch **Skip Move** and the app silently processes other eliminations until a Cell is reduced to only a single Candidate, which must be the solution in that Cell.
- Touch **Skip Game** and the app silently processes other eliminations until all Cells are solved and the game is over, or it is unable to make any mor eliminations.
- Touch **Stop Game** to stop the analysis. You can continue to solve the game manually, as described below.

As the analysis proceeds, the board is steadily filled in with the solved values, which are displayed in red. Eliminated Candidates are removed from the Grid.

When the game is solved, click on **Rewrite** to save the solution in the Game Definition File.

3.7 PLAYING A SUDOKU GAME

If you want to play the game yourself, you can specify the solution to individual Cells, or eliminate Candidates from Cells.

3.7.1 Selecting and Deselecting Cells

To eliminate Candidates, you need to be able to select an individual Cell, a set of Cells, or one or more adjacent Houses.

- You can select one or more Cells by touching them one at a time.
- You can select a rectangle of Cells by dragging a finger or stylus from one corner Cell to the opposite corner. All the selected Cells in the rectangle are selected and are coloured red
- You can select a House, or set of adjacent Houses, by dragging from one corner to the opposite corner of the set of Houses. (This is an extension of the technique above.) The selected Houses are coloured Green
- If you select the wrong Cell or House, touch **C** on the Number Pad on the screen to deselect it

3.7.2 Specifying Cell Values

If a Cell has only one Candidate left, that value is coloured in red, as here.

Figure 30

If you touch that Cell, the remaining value is written into the Cell, in blue, and the implications of that move are handled. Thus, that value is removed as a Candidate from all other Cells in the selected Cell's Row, Col or Box. Here is an example. Note that Cell (3,5) now has only a single Candidate left.

Figure 31

If the Cell has more than one Candidate left, touching it will colour the Cell in red, as here.

Figure 32

You can specify the solution for that Cell by touching the relevant button on the Number Pad, and then touch **OK**. For example, you can set this Cell to **9** as here.

Figure 33

The value you specify is displayed in the Cell in blue, and the consequences of the move are followed through in Cells in the same Row, Col or Block.

3.7.3 Eliminating Candidates

You can also eliminate some but not all Candidates from a Cell. In this example, the two {59} Cells in Box 3 mean that {59} can't appear in any other Cell in that Box.

Figure 34

So you can eliminate **6** from Cell (9,7). Select the Cell, type **¬6** on the Number Pad and touch OK. Note that **Play ¬6** appears in the area below the Number Pad, as shown here.

Figure 35

The Candidates for that Cell are reduced to {379}.

Figure 36

Note that the app won't object if you eliminate values that have already been removed from a Cell. So you could have selected Cell (9,8) as well as (9,7) before typing "-6". It *will* object if the game definition has a solution saved with it, and you try to eliminate the solution value from a Cell.

3.7.4 Saving a Game's Solution

When you have completely solved a game, or have used the **Solve** button to get the app to do it for you, press the **Rewrite** button to save the solution with the game. Then, if you try to play the game later and make an error, the app can detect it straight away.

If you load a game that has an embedded solution, a letter "S" appears in the progress area, as here.

Figure 37

3.8 IDENTIFYING CANDIDATES

You can use the Number Pad to visually identify all the Cells that still retain a given numeric value as one of their Candidates. So long as no Cell, Group or House is selected, touching (for example) **4** will highlight all the Cells for which 4 is still a viable candidate. Those for which the value is already set are striped. Here is an example.

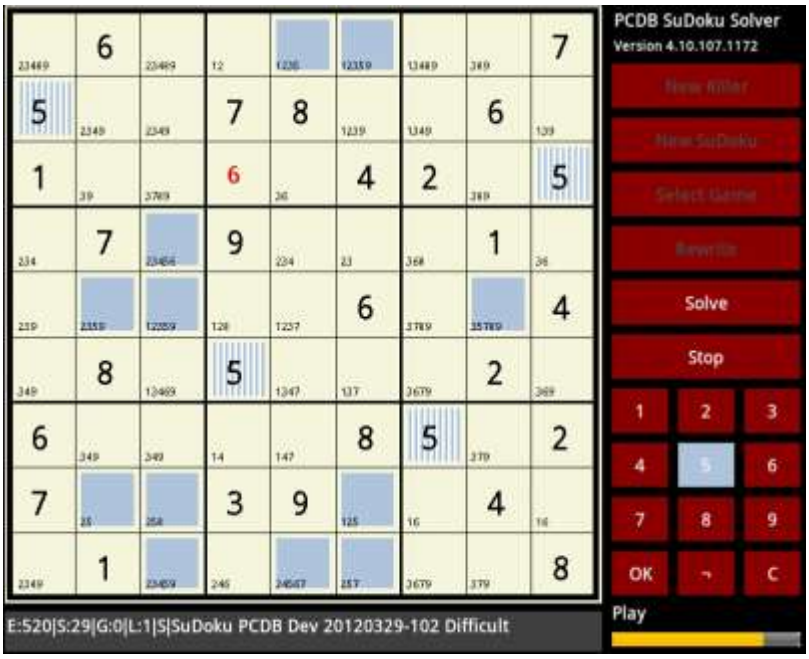

Figure 38

The colour of the highlighted Cells varies according to the numeric value selected. The background colour for selected button itself is set to the same colour as the highlighted Cells.

In this example, it is immediately obvious that the solution for Cell (5,8) is 4, as 4 is only a Candidate in that Cell within Box 6. You can set that value by touching the Cell, then touching the **4** button, then the **OK** button. Note that when you select a Cell that is highlighted with a button colour, the normal Cell selection colour (red) is shown at the outside of the Cell, with the button colour on the inside.

Once you have set that value, it then becomes clear that 4 is also the solution for Cell (6,3).

If you click on a different number key, a different set of Cells will be highlighted. If you click on the numeric key again, or click on the **C** key, the highlighting is suppressed,

3.9 FURTHER GAME ANALYSIS HINTS

The following processes are carried out when the app is solving a game, but can't yet be invoked manually when you are attempting to solve it yourself.

A number of techniques can be used to solve a SuDoku game.

3.9.1 Singlets

If a Cell only has one Candidate left, that must be the solution to that Cell.

If a Candidate only occurs in one Cell in a Row, Col or Box, then it must be the solution to that Cell.

3.9.2 Doublets

If the same two Candidates occur in two Cells within the same Row, Col or Box, then those must be the solutions to those two Cells. If there are no other Candidates in those two Cells, this is called a *Naked Doublet* and is fairly easy to spot. Any occurrences of those two Candidates elsewhere in the House can be eliminated.

If there are other Candidates in the two Cells in question, it is called a *Hidden Doublet* and may be more difficult to find. But any other Candidates in those Cells can be eliminated once you identify the doublet.

3.9.3 Triplets

If the same three Candidates occur in three Cells within the same Row, Col or Box, then those must be the solutions to those three Cells. Any other Candidates in those Cells can be eliminated, and any occurrences of those three Candidates elsewhere in the Row, Col or Box can also be eliminated.

Again, the Triplet may be *Naked* or *Hidden* depending on whether there are any other Candidates in any of the three Cells.

3.9.4 Quads

If the same four Candidates occur in four Cells within the same Row, Col or Box, then those must be the solutions to those four Cells. Any other Candidates in those Cells can be eliminated, and any occurrences of those four Candidates elsewhere in the Row, Col or Box can also be eliminated.

Again, the Quad may be *Naked* or *Hidden* depending on whether there are any other Candidates in any of the four Cells.

3.9.5 Quins

The same principle applies to five Candidates in five Cells, though this is extremely difficult to detect.

3.9.6 Box Constrainers 1

If all the Options for any particular value within a Box are all in the same Row or Col, then any other Options for that value in the same Row or Col but different Boxes can be elminiated.

Using the Number Pad keys to identify all the Cells which have a particular Candidate left can help you to visually identify instances where this rule applies.

3.9.7 Box Constrainers 2

Examine each group of three Boxes. If all the Options for any particular Candidate within two of the Boxes are all in the same two Rows or Cols, then that Candidate must be in the third Row or Col in the third Box.

Again, using the Number Pad keys to identify all the Cells which have a particular Candidate left can help you to visually identify instances where this rule applies.

3.9.8 Swordfish

3.9.9 X-Wings

A value can occur only once in each Row, Col or Box. If a given Candidate only exists in two possible Cells in a given Row, then it must be assigned to one of those two Cells. If a game has two Rows where the same Candidate is restricted to exactly the same two Cols (and no more than two Cols), then the Candidate *must* occur in those two Cols in those Rows, and can be eliminated from any other Cells in the two selected Cols.

3.9.10 XY-Wings

tbs

3.9.11 Forcing Chains

tbs

3.9.12 Conjugate Chains

Look for instances where two Candidates are in only two Cells in a given House. These have a "conjugate relationship"; one or other of the Cells must contain the value.

Try to follow a chain of such conjugate links, "colouring" the Cells alternately with two colours. One colour marks the "true" Cells, which are assumed to hold the value. The other marks the "false" Cells that don't. If a case is found where two Cells in such a chain in the same House have the same colour, then that must be the "false" colour and the value represented by that colour can be eliminated from the Cells marked with that colour.

4 KILLER SUDOKU

4.1 GAME CONCEPTS

A Killer SuDoku game differs from a SuDoku one in that there are no predefined Cell values. Instead, the 9 x 9 Grid is partitioned into a set of Groups, each containing a set of contiguous Cells. Within a Group, a digit cannot be repeated. The digits that go into each Cell in a Group must add up to the value shown at the top left of the top leftmost Cell in the Grid. Note that the total of the sums of all the Groups must be 405. Here is an example of a Killer SuDoku game.

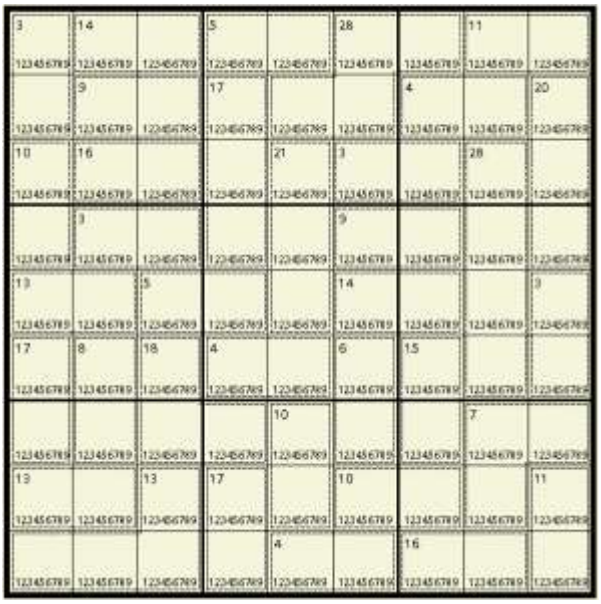

Figure 39

Here is the same grid with the solution filled in.

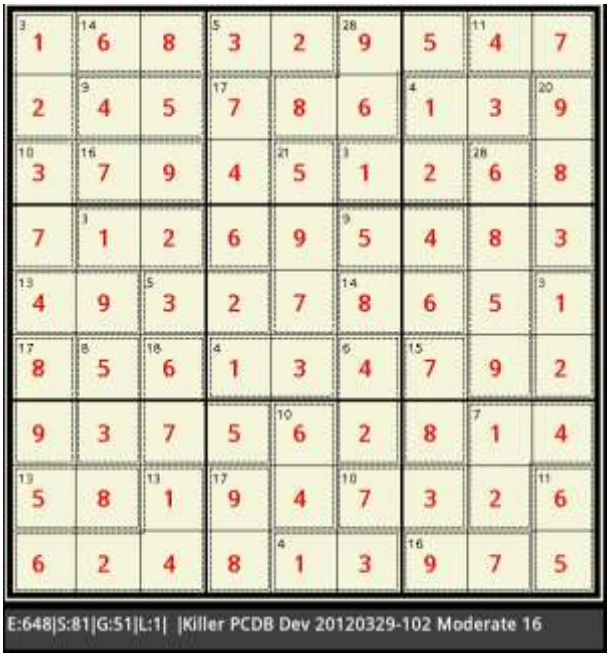

Figure 40

You solve a Killer SuDoku game by calculating the set of digits that can occupy each Group and House.

The most important concept to use is that, as each Row, Col or Box must contain the digits 1 through 9, then the sum of the digits in each Row, Col or Box must be 45. By extension, the digits in two adjacent Rows or Cols must sum to 90, and so on.

There are a number of techniques, of varying degrees of complexity, which can be used to solve a Killer SuDoku game.

4.1.1 Innies and Outies

If a House contains a number of Groups so that only one Cell in one of the Groups protrudes outside the House, then it is easy to calculate the value that goes into that Cell. In the example below, the 10(2) Group is partly in Box 1 and partly out of it. The portion *within* the House is called an *Innie*. The portion *outside* the House is called an *Outie*.

Figure 41

Now:

- The sum of the Groups in this House, excluding the 10(2) Group, is 16 + 9 + 3 + 14 = 42. So as the sum of all the Cells in the House must be 45, the value of Cell (3,1) must be 3.
- Ths sum of the Groups in the House, including the 10(2) Group, is 10 + 16 + 9 + 3 + 14 = 52. As the sum of all the Cells in the House must be 45, the value of the Cell *outside* the House must be 52 – 45 = 7.

Of course, in this simple case, having solved one of the Cells in the 10(2) Group the other can also be found by subtracting it from the sum of the Group!

In more complex cases, there may be multiple Innies and Outies, as in this example.

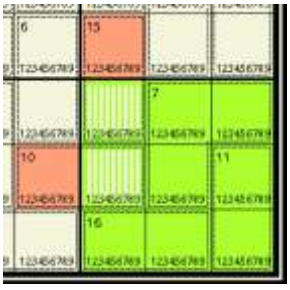

Figure 42

From this, we can deduce that the sum of the Innies (Cells (7,7)(8,7)) must be 11, and the sum of the Outies (Cells (6,7)(8,6)) must be 14. To help in exploiting this knowledge, we can define two new Groups, an 11(2) covering (7,7)(8,7) and a 14(2) covering (6,7)(8,6). Note that as the latter Group contains Cells that are not in the same Row, Col or Box, it is entirely possible that it contains two 7s!

4.1.2 Group Options

When a Group is defined, it has two basic parameters: its size, and its Sum. The value of all the Cells in the Group must add up to its Sum. Depending on the size and sum of the Group, this may be possible in a number of different ways. But there are some cases where the number of possibilities is limited, and these can help you reduce the solution to manageable proportions. For example, a 2-Cell Group with a sum of 16 *must* have the solutions 7 and 9. (Values of 8 and 8 would violate the rule that all the digits in a Group must be distinct.) When such a Group is found, these two values can then be eliminated from other Cells that are entirely within the same House as the 16(2) Group.

Similarly, a 3(2) Group can only contain 1 and 2. A 22(3) Group can contain either 6, 7 and 9 or 5, 8 and 9.

These possibilities are called the Group's Options, and are conventionally represented by a number of equal-length digit strings in square brackets, thus:

- 17(2) [89]
- 3(2) [12]
- 22(3) [589,679]

As you solve a Killer SuDoku game, the values that you work out can limit the Options in a Group. Thus, if you have a 22(3) Group and identify that one of the Cells in it holds the value 7, then its Options reduce from [589,679] to [679].

Similarly, if a 22(3) Group is entirely contained within a single House, and you deduce that one of the Cells in the House but outside the Group has the value 5, then the [589] Option is no longer valid for the Group and can be eliminated.

One of the aims of the app, or of you as the solver, is to reduce each Group to a single Option string, so that you know the exact digits that must go into each Group. The remaining problem then is to determine which digit goes into which Cell!

4.1.3 Signed Groups

A crucial part of solving a Killer game is to add Group definitions for the Innies and Outies associated with any House Set. There is a more subtle way to approach this, which is to example combinations of Innies and Outies. Consider the following from the same game:

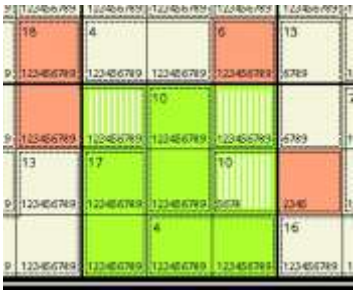

Figure 43

Now it is possible to define Groups covering both the Innies ((7,4)(7,6)(8,6)) and the Outies ((6,3)(6,6)(7,3)(8,7)). However consider a Group comprising (7,4)(8,6) and (6,6). It can be deduced that the sum of the first two Cells, minus the third, must be 8. This can be represented as a Group +8(3), where the "+" indicates that the Group is signed. The Cells which comprise it are represented as (7,4)(8,6)-(6,6).

4.1.4 Group Patterns

Given the processes above, you can reduce a Group to a single Option string, but still not know which digit goes into which Cell within the House. To make this deduction, we use *Group Patterns*. This is a process that tries to make all the possible solutions for the Group based upon its Options and the Candidates for the Cells that comprise the Group. It notes which Candidates from which Cells are used in any of the successful solutions, and eliminates any Candidates that are not used in any of these solutions. It also eliminates any Option that cannot be made using the available Candidates.

For example, consider a 19(3) Group that has remaining Options [379,568]. The three Cells have candidates:

- C1 {35678}
- C2 {356}
- C3 {59}

There are only two possible solutions to this:

- [379]: (C2,C1,C3), and
- [568]: (C3,C2,C1)

So the Candidates {356} in C1 and {5} in C2 cannot be used in any achievable solution and so can be eliminated, increasing the number of excluded Candidates by 4.

4.1.5 Overlapped Houses

This technique is attributed to Jean-Cristophe Godart.

4.1.6 Split Groups

4.2 CREATING A KILLER SUDOKU GAME

If you touch **New Killer**, the grid is cleared and each Cell is coloured light blue, as shown here.

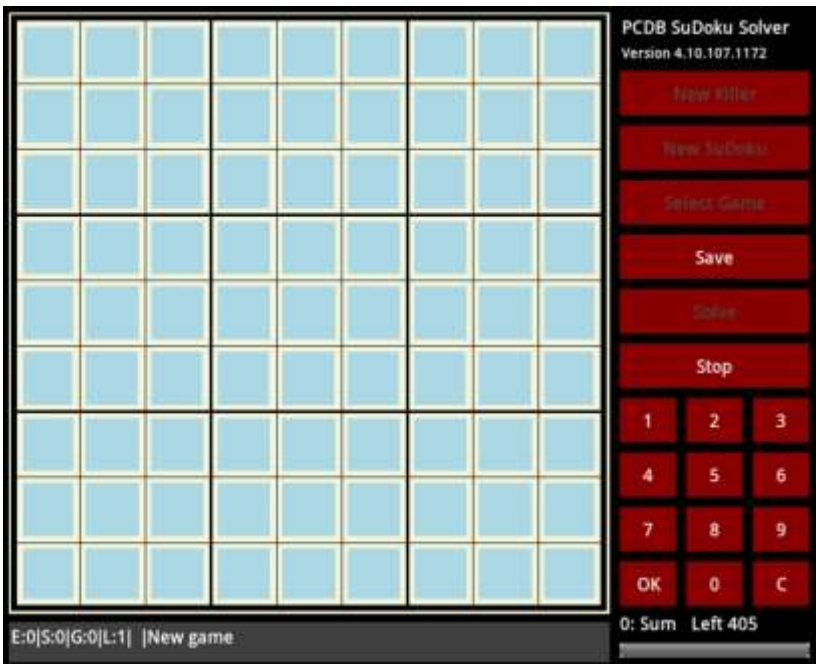

Figure 44

You need to identify a set of Groups which between them cover every Cell in the grid, and specify the sum of the Cells within each Group. To specify the Cells that are to comprise a Group, touch the Cells one after the other, or drag your finger or a stylus over them. These Cells must be contiguous, either horizontally or vertically. As you touch a Cell, or drag over it, it changes to red, as shown here.

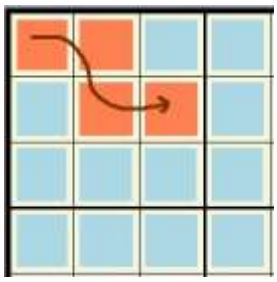

Figure 45

After the first, each Cell that you select must be adjacent to (beside, above or below) an already selected Cell.

If you select a Cell in error, touch it again and it will revert back to light blue. If you drag over a Cell that you don't want to select, lift your finger or stylus and then touch the Cell to deselect it. (If you drag the cursor over a Cell that is already part of another Group, it is ignored. If you lift your finger or stylus while on such a Cell, you will receive an error message.)

After you have selected all the Cells that are to form the Group, you need to specify the sum for that Group, using the numeric buttons on the screen. If you make a mistake, touch the **C** (Clear) button. When the sum is complete, touch the **OK** button, or begin to select a new Group.

The number of cells selected appears under the Number Pad, as shown here. As you type the sum, it appears after that. Here, a sum of 15 is specified.

Figure 46

The background of the selected Cells changes to white, a dotted line is drawn around them, and the Group's sum is displayed in the top leftmost Cell, as shown here.

Figure 47

Continue to select the Cells for each Group, and specify its sum. For example, the second Group in this game may be a 17(3) Group as shown here.

Figure 48

Note that you don't actually have to press OK to complete the definition of a Group. Once you have a non-zero value in the Sum text box, touching another Cell will complete the definition of that Group and start a new Group. You do have to click on the **OK** button to complete the last Group on the Grid though.

As the game definition progresses, the app keeps a running count of the number that the remaining Groups must sum to. After those two Groups, it is 373 as shown here:

Figure 49

When you have set up the entire board, the app sets up the Candidates in each Cell. At this stage, Candidates in each Cell are {123456789}. Figure 34 above shows an example, and here is the full screen with this game defined.

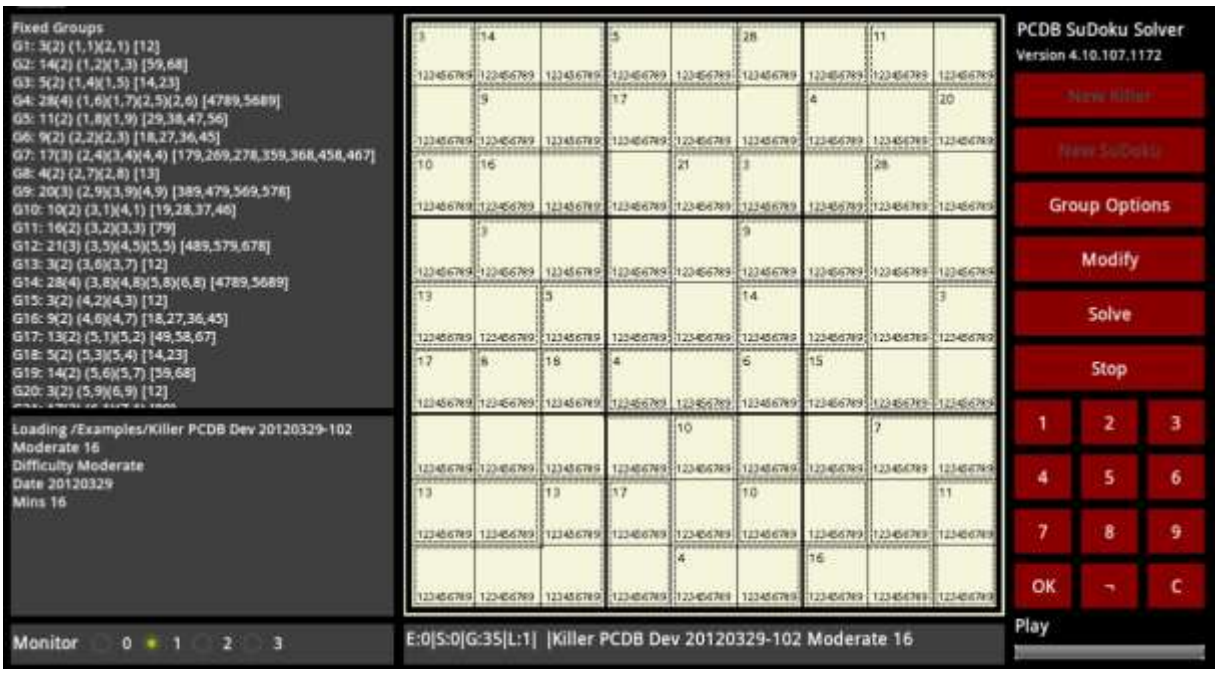

Figure 50

Note that the Groups area on the left contains details of all the Groups you have defined. The format of each Group definition, such as for example G14: 28(4) (3,8)(4,8)(5,8)(6,8) [4789,5689], is as follows.

- G14 is the Group number
- 28(4) is the size of the Group (4 Cells) and its sum (28)
- (3,8)(4,8)(5,8)(6,8) identify the four Cells that comprise the Group. A Cell is identified by its Row and Column in that order so that (3,8) represents Row 3 Column 8
- [4789,5689] is the initial set of Options for this Group, based purely upon its size and sum. If the set of Options is too large to be displayed, the first and last Options are given and those in between are represented by their number, so that {1234,.(15).,6789] represents 17 Options of which the first is [1234] and the last [6789]

As Groups are added, their description can also include further information, for example that the Group is formed of the Outies relating to a particular House.

4.3 SAVING A GAME

When you have finished the definition of a game, you can save it in a Game Definition File so that you can reload and solve it at your leisure. You can't play or solve the Game until you have saved and reloaded it!

You save a Killer SuDoku game in the same way as a SuDoku game. See Section 3.3 above.

4.4 LOADING A GAME

You can reload the game's Game Definition File immediaterly. After you save a game, the **Load** Game button text changes to **Reload**, as with SuDoku. Touch this button to reload the game you have just saved.

As with SuDoku, you can modify and resave the game if there are any errors in it.

As with SuDoku, you can touch **Stop** and then **Select Game** so that you can select another saved Game.

4.5 SOLVING A GAME

Once you have loaded a Game Definition File, you can either solve it yourself, with help from the app, or direct the app to carry out a full analysis of the game.

If you touch **Solve**, the app attempts to reduce the set of Candidates to one per Cell. It attempts to reduce the set of Options to one per Group, and then use this single Option to eliminate Candidates that don't confirm to it. As each Cell is reduced to a single Candidate, it is removed from any Group to which it belongs, and when all the Cells in a Group are reduced to a single Candidate the Group itself is removed.

In the example below, the app has looked at the Options for a 3(2) Group. These are fairly simple: [12]. Thus all Candidates except {12} can be eliminated from both the Cells in this Group.

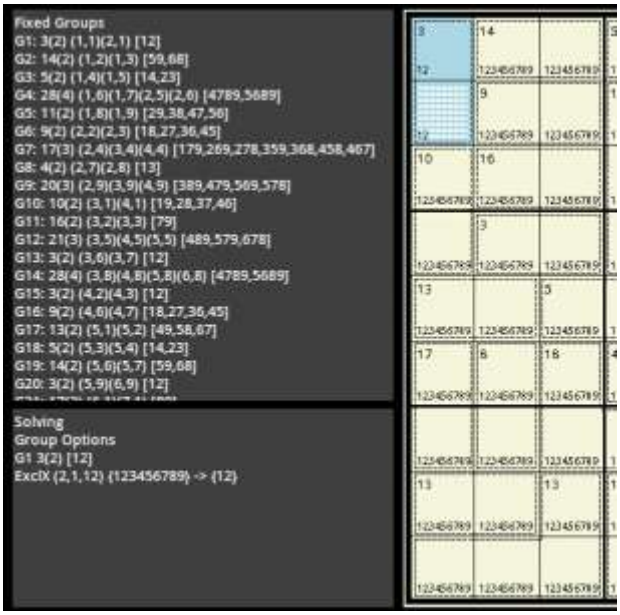

Figure 51

You have five options at this stage.

- Touch Continu and the app proceeds to the next elimination.
- Touch **Skip Proc** and the app silently processes other eliminations made by this procedure (Single Choices) and stops when the first elimination is made by another procedure.
- Touch **Skip Move** and the app silently processes other eliminations until one Cell is left with only a single Candidate, which must be the solution in that Cell.
- Touch **Skip Game** and the app silently processes other eliminations until all Cells are solved.
- Touch **Stop Game** to stop the analysis. You can continue to solve the game manually, as described below.

As the analysis proceeds, the board is steadily filled in with the solved values (shown in red) and eliminated Candidates are removed from the Grid.

When the game is solved, touch **Rewrite** to save the solution in the game file.

4.6 PLAYING A GAME

If you want to play the game yourself, you can specify the solution to individual Cells, or eliminate Candidates from Cells.

4.6.1 Selecting and Deselecting Cells, Houses and Groups

To eliminate Candidates, you need to be able to select an individual Cell, a set of Cells, a Group, or one or more adjacent Houses.

- You can select one or more Cells by touching them one at a time, or by dragging a finger or stylus from one Cell to another. All the selected Cells are coloured red.
- You can select a Group by long-touching[5] on any Cell within it, apart from a Cell whose value has already been entered and is shown in blue; these are removed from the Group definition. The Cells in the Group are then coloured light blue. If the Cell appears in more than one Group (for example because it is part of an Innie or Outie Group), then repeatedly tapping on the same Cell will iterate through all the Groups that the Cell belongs to.
- If you have selected a Group, you can select one or more Cells within it (other than the Cell that you touched to select the Group), for example prior to invoking the **Split Group** tool (see below). The Cells you select are coloured in Beige. If you select a Cell *outside* the Group, the selection of the Group is cancelled and the newly selected Cell is shown in red.
- You can select a House, or set of adjacent Houses, by dragging from one corner to the opposite corner of the set of Houses. The selected Houses are coloured Green. As an aid to solving, the display identifies the Innies and Outies relating to the selected Houses, as shown in the example below.
- If you select the wrong Cell, House or Group, touch the **C** button on the Number Pad to deselect it.

4.6.2 Specifying Cell Values

If you touch a Cell that has only one Candidate left (and thus displays that Candidate in red), the Candidate is automatically set as the value in the Cell.

To specify the solution to a Cell that has more than one Candidate left, touch the Cell and type the value using the Number Pad. Touch **OK** to confirm the value.

In both cases, the value that you specified appears in the Cell in blue, to differentiate it from the values that form part of the game definition. This value is removed from all other Cells in the same Row, Col or Box.

In this example, it is clear that the value in the 3(2) Group Cell at (7.9) must be 2 because the 4(2) Group in Row 6 must have the values 1 and 3, which means that Cell (6,9) can't be 1. Touch Cell (6,9).

[5] *Touch the Cell and hold for at least half a second before lifting the finger or stylus.*

Figure 52

then touch **2** and then **OK.**

Figure 53

The value 2 appears in the Cell, coloured blue, and all the previous Candidates shown in that Cell are removed. The app then eliminates that value from other Cells in the same Row, Col or Box, according to normal SuDoku rules. It also eliminates it from the Cell's fixed Group, and any other Constrained Groups that the Cell belongs to. This leaves Cell (5,9) with only one Candidate (1), so that Candidate is displayed in red. If you then touch that Cell, the Candidate 1 turns blue and is removed from all other Cells in the same Row, Col, Box and Groups.

4.6.3 Eliminating Candidates

You can also eliminate some but not all Candidates from a Cell or Group. In our example, the 4(2) Group in Row 4 can only contain the values [13]. You can specify that all the Cells in the Group are limited to these Candidates. Touch-and-hold on either Cell in the Group. Its members are highlighted in light blue, thus:

Figure 54

Touch **1** and then **3** and then **OK.** This specifies that the Candidates for the selected Cells can only be taken from the values 1 and 3. The Candidates for the both Cells in the Group are reduced to {13}.

Figure 55

Now, because this 4(2) Goup is entirely within Row 6, the values 1 and 3 can't appear as Candidates anywhere else in Row 6. Touch all the relevant Cells, one after the other, then touch ¬, **1** and **3** in that order. ¬**13** means remove the Candidates 1 and 3 from the selected Cells.

Figure 56

Touch **OK**. The candidates 1 and 3 are removed from these Cells.

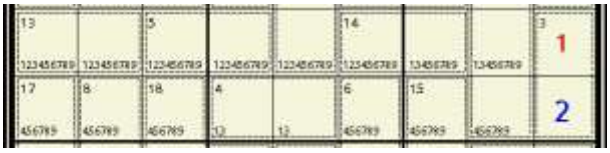

Figure 57

Note that the app won't object if you eliminate values that have already been removed from a Cell. It *will* object if the Game Definition File has a solution saved with it, and you try to eliminate a value that is the proper solution for a Cell.

If you inadvertently remove a Candidate that you later decide needs to stay in contention, you can reinstate it. For example, if the Candidates for a Cell were {1479} and you inadvertently reduced these to {179}, you can select the Cell, type "+4" and touch **OK**. The app will check that you wish to restore the Candidates to {1479} (i.e. the merger of the existing Candidates and those following the "+"), and then make that change. Note that if you have made any changes to a Group's Options or Patterns as a consequence of the erroneous Candidates, these will remain and may cause problems later in the game.

4.6.4 Using the Solving Tools

There are number of tools that you can use to help play a game. At present, these are only relevant for Killer SuDoku games.

4.6.4.1 Tools for Groups

The Group Tools can be used on a specific Group, or on all Groups. Double-tap or touch-and-hold a Cell within a Group to select it. Touch **C** to remove any selections if you want the tool to apply to the entire game.

There is a difference between using one of the Group tools from the menu buttons, and using the **Solve** facility. The menu buttons apply the tool once, either to a specific Group or to all Groups. The **Solve** function repeatedly applies all of these tools, and a number of others, until the game is solved or no further progress can be made.

Some of the app's buttons give the names of these tools, thus:

Figure 58

4.6.4.1.1 Split Groups

This process tries to split the selected Group into one or more pairs of Subgroups. Each Subgroup is a Child of the selected Group, and each Subgroup in a pair is a Sibling of the other. Each pair of Subgroups will have Sum ranges that complement each other and are derived from the sum of the Father Group. That is, the maximum of one plus the minimum of the other must add up to the Sum of the Father Group, and vice versa. For example, if a 23(4) Group is split into two Subgroups, each of these will have an initial specification of 7..17(2), whereas the basic range of a two-Cell Group is 3..17(2). If any of the Cells in the Parent Group have had their Candidates reduced, the sum ranges of the Subgroups could be further constrained.

If one of the potential Subgroups only has one Cell in it, no Subgroup is created but the sum range for this potential Subgroup is used to constrain the Candidates for that Cell.

If you select **Split Groups** with just a Group selected (as in the example below), the app finds any instance where some of the Cells in the selected Group are in one House and some in another. There may be various ways to do this.

Figure 59

In this example, the app creates a Subgroup comprising Cells (1,7)(1,8)(1,9), which you can display by long-clicking on Cell (1,7) and then touching it again..

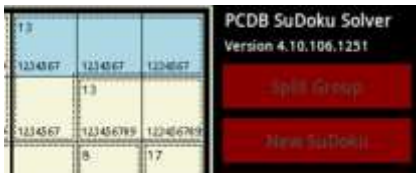

Figure 60

Alternately, once you have selected the Group, you can then select some of the Cells within it. Then, if you touch **Split Group**, the app attempts to create a Subgroup comprising the selected Cells, and another comprising the residue. If you select the same 13(4) Group, and then select Cells (1,8)(1,9) as shown below, and then touch **Split Group**, the app creates two new 3..10(2) Subgroups, one (G37) comprising (1,8)(1,9) and the other (G38) comprising (1,7)(2,7).

Figure 61

Their properties are shown in the Groups area thus:

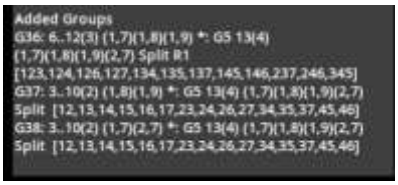

Figure 62

4.6.4.1.2 Group Options

Touching this button causes the app to review all the Options associated with the Group, or all Groups, in the light of the Candidates currently left for the Group's Cells. This is a multi-stage operation that carries out a number of processes including the following.

- *Identify Group Cells*. Identify the Cells that comprise the Group. Check any signed Cells, and adjust the SumMin and SumMax values accordingly. Build up a list of all the Candidate values found in any +ve Cell in the Group.
- *Subtraction Combo***.** If there are no –ve Cells, and the sum of the Group's Candidates is greater than the Group's Sum, then we can try to eliminate one or more Candidates that add up to the difference between the Group Sum and the Candidate Sum. Elimination is possible if there is only one set of Candidates that adds to this difference. If any such Candidates can be found, eliminate them from all Cells in the Group.
- *Copy Congruent Unsigned Group Options*. If the Group is Signed, then check to see if there is an Unsigned Group comprising solely the +ve Cells. If so, copy the Options and other information from that Group into the Signed one under consideration.
- *Option Candidate Checks*. Check each +ve Cell's Candidates against each of the Group's Options. At least one of the Candidate values should appear in the Option. If not, then that Option is not feasible and can be removed from the Group.
- *Candidate Option Checks*. We have previously calculated the union of all the Candidates that appear in the Group. If any value is missing from his list, then we can remove any Option that contains values not in the union. Thus, suppose we have a Group 15(3) in Cells (1,1)(1,2)(1,3) with Options [159,168,249,258,267,357,456]. These Cells have Candidates as follows:

  - (1,1) {125}
  - (1,2) {1267}
  - (1,3) {2358}

  The union of these Candidates is {1235678}, and hence any Option that contains either 4 or 9 can be eliminated, leaving [168,258,267,357] and reducing the Option count by three.

- *Father's Options Check*. Check the Group's Peers and their Fathers to see whether any of the Group's Options can't be contained within any of the Father's Options. If so, remove that Option from the Group.

- *Children's Options Check*. The converse of the above. Check the Group's Children to see whether any of the Child's Options can't be contained within any of the Father's Options. If so, remove that Option from the Father Group.
- *Sibling Mandatory Check*. If the Group (or one of its Peers) has a Sibling with any Mandatories, then remove any Options that contain any of these Mandatories. Because the Group and its Sibling are both in the same House or Constrained Group, any Mandatories of one Group can't appear as Candidates or Options in the other, so eliminate them as Candidates in the Group's Cells as well as from its Options.
- *Sibling Range Check*. If the Group has a value range, check its Peers and their Sibling to see whether its range values can be further limited. This will be the case if the range for any Sibling is smaller than this Group's range. Also, check the sums of the Options for each Sibling, and if any value in the Range isn't achievable in the Sibling, remove any Options with the complementary sums in this Group. Note that Signed Groups don't have any Siblings, and nor do Groups where the Sibling, if it existed, would only comprise one Cell. So the only checks needed to trigger this process are that the Groups .SumMin and .SumMax values differe and there is a Father.
- *House Residue Check*. If the +ve Cells in a Group are entirely within a single House, look through all Cells in the House that are not in the Group. If any of the Group's Options contains all of the Candidates in that Cell, then the Option cannot be achieved and can be removed.
- *Group Pairs Check*. If the Group is entirely contained within a single House, look for other Groups which are entirely within the same House. If such a Group has any Pairs, use these to remove Options which conflict with those from the current Group.
- *Negative Cell Options Check*. If the Group is Signed, with a single –ve Cell, check each value in the –ve Cell's Candidates against the Group's +ve Cells' Options. Remove any Option which does not deliver the complementary sum to one of the –ve Cell's Candidates. Also eliminate Candidates that don't complement any Option.
- *Excluded Candidates Check*. Eliminate any Candidate that doesn't appear in one of the Group's Options.
- *Group Mandatories Check*. If there are any Mandatories in the Group (i.e. values that occur in every Option), and if the Group is entirely within a single House, then those Candidates can be removed from Cells in the House that aren't in the Group.
- *Negative Cell Check*. If the Group is Signed, identify the only possible Candidates for the –vev Cell, and eliminate any others.
- *Multiple Negative Cells Options Check*. If the Group has multiple –ve Cells, check each –ve Cell against the Group's +ve Cells' Options. Remove any Option which does not deliver the complementary sum to one of the –ve Cell Options. Also eliminate Candidates that don't complement any Option.
- *Two-Cell Group Complements Check*. For each Constrained two-Cell Group, remove Candidates from one Cell that complement Candidates already removed from the other. For each Signed two-Cell Group, remove Options from one Cell that correspond, given the difference between them, to Options already removed from the other.
- *Persistent Duplicates Check*. Where a Group is entirely constrained within a House, then each value in the Group must be unique. Where an added Group overlaps more than one House, then it may have duplicated values. This process looks for any persistent duplicates in the Group's Options, i.e. cases where every Option has at least one duplicated value. If there is one Cell that is

not in the same House as all the others, it can only have these duplicates as its Candidates, so eliminate any other Candidates in that Cell

- *Single Cell Groups*. If there is only one Cell left in the Group, then remove any Candidates except those indicated by its remaining Options.
- *Replicate to Peers*. Finally, replicate any changes to all the Peers of this Group.

For example, if we invoke **Group Options** on the 16(2) Group selected in the example above, the Options for that Group are set to [79], and Candidates {79} are removed from other Cells in the same Row and Box, as here.

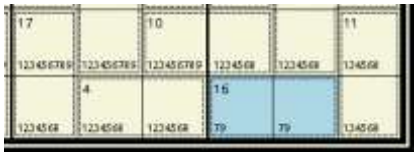

Figure 63

4.6.4.1.3 Group Patterns

This process tries to make all the possible solutions for the Group's Cells based upon its Options and the Candidates for the Cells that comprise the Group. The process is described, with an example, in Section 4.1.4 above. It notes which Candidates from which Cells are used in any of the successful solutions, and eliminates any Candidates that are not used in any of these solutions. It also eliminates any Option that cannot be made using the available Candidates.

4.6.4.1.4 Group Mandatories

Where a Group is entirely within a House, and that Group has one or more Mandatory values (i.e. values that occur in every one of the Options for the Group), then those values cannot appear anywhere else in the House and can be eliminated from the Cells that are in the House but not in the Group.

As a simple example, consider a 16(2) Group, which can only have the values 7 and 9. These are thus its Mandatories, and if the Group is entirely within a House then these Candidates can be eliminated from all other Cells in the House.

4.6.4.2 Tools for Houses

Select a House or House Set by dragging the cursor from one corner of it to the opposite corner. In the example below, the cursor is dragged from Cell (1,1) to (3,3), as shown by the red arrow. Note that the Innies and Outies relating to these Columns are identified. Outies are shaded in salmon pink, and Innies are green (as with the rest of the House), but with white horizontal stripes.

If a House (or House Set) is selected, then tools related to Houses are enabled.

Figure 64

4.6.4.2.1 Find Innies & Outies

This tool (the only one currently available) examines every Group that has some Cells in, and some outside, the selected Houses. It processes Innies, Outies, and then Innie/Outie combinations. For a single Innie or Outie it can calculate the value that is to go in the Cell. If there are multiple Innies or Outies, but less than a configurable limit which is currently set to 6, it sets up a new Added Group that contains the set of Innies or Outies. The sum of this Group is the sum of the Innies or Outies as calculated.

As shown in the diagram below, in this example the sum of the Outies, Cells (6,1)(6,2)(6,3)(7,4), must be 24. (17 + 8 + 18 + 13 + 13 = 69; 69 – 45 = 24.) So a new 24(4) Group is created comprising these Cells.

Simarly, a new 19(3) Group is created comprising the Innies, Cells (7,1)(7,2)(7,3).

These new Groups appear in the Groups area and can be seen if you scroll this area down, thus:

Figure 65

Note that there are also five Innie/Outie Groups created by this use of the tool.

4.6.4.2.2 Find Subgroups

The **Find Subgroups** menu item is enabled if you select just a single House (i.e. one Row, or one Col, or one Box). It looks for any single Group in that House, and tries to form another Group with the Cells that aren't in that Group. These are called the *residual cells*.

4.6.5 Identifying Residual Candidate Values

As with the SuDoku game, you can use the Number Pad to identify all the Cells that contain a given numeric value as one of their residual Candidates. Make sure that no Cell, Group or House is selected, and then press on the key for the value you are interested in. All Cells for which this is a remaining Candidate are highlighted. The highlight colour depends on the numeric value. The number key's background is set to the same colour.

Clicking on a different key causes a different set of Cells to be highlighted. Clicking on the same key again, or on the "C" key, suppresses highlighting.

4.7 SAVING A GAME'S SOLUTION

When you have completely solved a game, or have used the **Solve** button to get the app to do it for you, touch the **Rewrite** button to save the solution with the game definition. Then, if you try to play the game later and make an error, the app can detect it straight away.

If you load a game that has an embedded solution, a letter "S" appears in the status area, as here.

Figure 66

4.8 FURTHER GAME ANALYSIS HINTS

All the techniques that can be used to solve a SuDoku game also apply to Killer Games, though you will need to elimate a goodly number of Candidates before most of them become applicable.

In addition, there are a number of processes specific to Killer games that are carried out when the app is solving a game, but can't yet be invoked manually when you are attempting to solve it yourself.

4.8.1 Group Doublets

If any Group contains two Cells with identical two-number Candidates, then those Candidates can be eliminated from other Cells within the same Group.

4.8.2 Complex Elimination

If a Group can be treated like a Doublet, then check it against other Doublets in the same House to see whether there are any Candidates that can be eliminated. A Group can be treated like a Doublet if it has any Pairs set.

4.8.3 Conjugate Chains

Look for instances where two Candidates are in only two Cells in a given House. These have a "conjugate relationship"; one or other of the Cells must contain the value.

Try to follow a chain of such conjugate links, "colouring" the Cells alternately with two colours. One colour marks the "true" Cells, which are assumed to hold the value. The other marks the "false" Cells that don't If a case is found where two Cells in such a chain in the same House have the same colour, then that must be the "false" colour and the value represented by that colour can be eliminated from the Cells marked with that colour.

4.8.4 Conjugate Groups

Look for any House which is solely occupied by two fixed Groups, where one of these Groups has Cells outside the House. If any single Cell in one Group is outside the House, then the Candidates in that Cell

must be in the other (Conjugate) Group *within* the House. If only one Cell in the Conjugate Group has any of these Candidates in it, then remove any other Candidates in the Cell.

4.8.5 Complex Houses

If a Group lies entirely within a House, try to find any Doublet, Triplet, Quad or Quin that can be made with the Group's residue within the House. In doing this, assume that the Group's Candidates are the merger of all the Group's Cells' Candidates.

If any Doublet (etc) is found, it will not be possible to restrict the Candidates within the Doublet (etc), but the Doublet (etc) values can safely be eliminated form the Cells that do not form part of the Doublet (etc).

4.8.6 Complementary Groups

If two Groups are both restricted to the same two Houses, and have any Mandatories in common, those Mandatories can be removed from the rest of Cells in these Houses.

5 FILE FORMATS

Game definitions are stored in files whose names are formed from the game name suffixed with "`.txt`". Different formats are used for SuDoku and Killer SuDoku games. You can find these files in the **My Files** app, under the Android/data/killersudoku.com/files/games folder, as shown here.

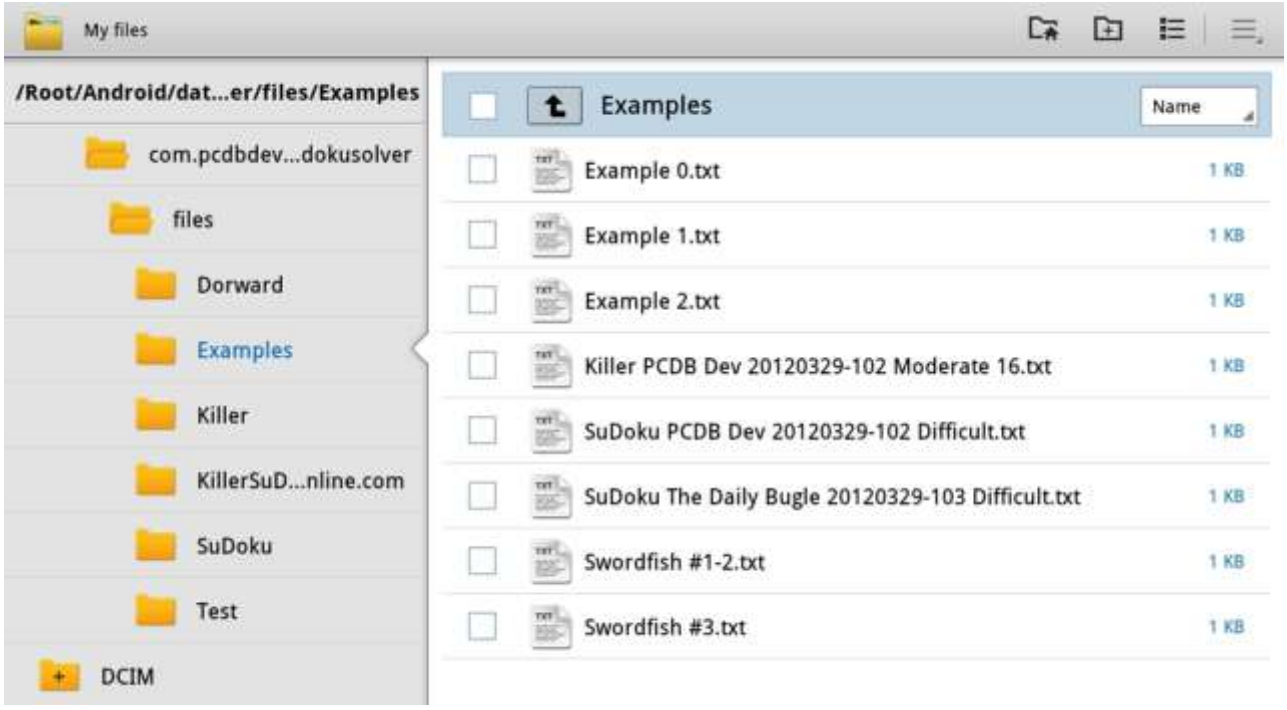

Figure 67

5.1 KILLER SUDOKU DEFINITION FILES

Here is an example of a Killer SuDoku Game Definition File, with line number added.

```
01: Killer SuDoku V3.18.101.1120 27/07/2011 11:17:45
02: Difficulty Tough
03: Date 20110616
04: Number 2044
05: Mins 43
06: Group 1,13,2,(1,1)(2,1)
07: Group 2,8,2,(1,2)(2,2)
08: Group 3,20,4,(1,3)(1,4)(2,3)(2,4)
09: Group 4,14,2,(1,5)(1,6)
10: Group 5,22,4,(1,7)(1,8)(2,7)(3,7)
11: Group 6,11,3,(1,9)(2,8)(2,9)
12: Group 7,10,2,(2,5)(3,5)
13: Group 8,10,2,(2,6)(3,6)
14: Group 9,3,2,(3,1)(3,2)
15: Group 10,15,3,(3,3)(3,4)(4,4)
16: Group 11,12,2,(3,8)(3,9)
17: Group 12,13,2,(4,1)(4,2)
18: Group 13,11,2,(4,3)(5,3)
19: Group 14,15,2,(4,5)(4,6)
20: Group 15,8,3,(4,7)(4,8)(4,9)
21: Group 16,19,4,(5,1)(5,2)(6,1)(6,2)
22: Group 17,18,4,(5,4)(6,3)(6,4)(7,3)
23: Group 18,10,4,(5,5)(5,6)(5,7)(6,5)
24: Group 19,17,2,(5,8)(5,9)
25: Group 20,28,4,(6,6)(7,5)(7,6)(8,5)
26: Group 21,13,2,(6,7)(7,7)
27: Group 22,7,2,(6,8)(7,8)
28: Group 23,22,4,(6,9)(7,9)(8,9)(9,9)
29: Group 24,6,2,(7,1)(7,2)
30: Group 25,9,2,(7,4)(8,4)
```

```
31: Group 26,22,3,(8,1)(9,1)(9,2)
32: Group 27,13,2,(8,2)(8,3)
33: Group 28,18,4,(8,6)(8,7)(8,8)(9,8)
34: Group 29,10,2,(9,3)(9,4)
35: Group 30,8,3,(9,5)(9,6)(9,7)
36: Solution
37: 738259416
38: 659174823
39: 214836975
40: 946387251
41: 175642398
42: 382915647
43: 421568739
44: 567493182
45: 893721564
46: End
```

- Line 01. This defines the type of game (Killer SuDoku), the Version of the app which wrote the file (`V3.18.101.1120`) and the date and time at which it was written (27/07/2011 11:17:45).
- Line 02. This defines the difficulty level as defined by the publisher (in this case Tough)
- Line 03. This defines the date on which the game was published, in the format yyyymmdd.
- Line 04. This defines the number given by the publisher, in this case 2044.
- Line 05. This defines the number of minutes set as a guideline by the publisher within which the game should be solvable manually; in this case 43.
- Lines 05-35. This consists of a number of lines each defining one Fixed Group within the game. Each line has the format "Group no,sum,#cells,cells" where "no" is a unique consecutive number given to the Group, "sum" is the sum of the Cells in the Group and "#cells" is the number of Cells. "cells" is a specification of each Cell forming part of the Group, in the format "(r,c)", where "r" is the Row and "c" the Column.
- Lines 36-45. This is optional, and only exists if the game has been solved, either by the app or by the user, and then the Game Definition File rewritten to include the solution. It consists of the word "`Solution`" followed, on nine consecutive lines, by nine digits each defining the contents of the Cells in one Row of the solved game. The first line represents Row 1, the second Row 2 and so on.
- Line 48. This consists solely of the word "End" and signifies the end of the game definition.

5.2 SuDoku Game Definition Files

Here is an example of a SuDoku Game Definition File, with line numbers added.

```
01: SuDoku V3.17.106.1120 20/07/2011 10:23:20
02: Difficulty Difficult
03: Date 20110516
04: Number 4032
05: Hash 0100000023002400105000067030208000500000010060403000701000043072009800600500000004
06: Game
07: 090000007
08: 600280030
09: 500007401
10: 070400010
11: 000005003
12: 060800090
13: 900003104
14: 400670050
15: 050000006
16: Solution
17: 291354867
18: 647281539
19: 538967421
```

```
20: 375496218
21: 829715643
22: 164832795
23: 986523174
24: 413678952
25: 752149386
26: End
```

Points to note are as follows.

- Line 01. This defines the type of game (SuDoku), the Version of the app which wrote the file (V3.17.106.1120) and the date and time at which it was written (20/07/2011 10:23:20).
- Line 02. This defines the difficulty level as defined by the publisher (in this case Difficult)
- Line 04. This defines the date on which the game was published, in the format yyyymmdd.
- Line 05. This defines the number given by the publisher, in thi case 4032.
- Line 05. This defines a hash code, or 81-character numeric string which is unique amongst all inversions and reflections of this game.
- Lines 06-15. This contains the word "Game" followed, on nine consecutive lines, by nine digits each defining the content of the Cells in one Row of the game. The first line represents Row 1, the second Row 2 and so on. A value of "0" shows that the Cell is empty.
- Lines 16-25. This is optional, and only exists if the game has been solved, either by the app or by the user, and then rewritten to include the solution. It consists of the word "Solution" followed, on nine consecutive lines, by nine digits each defining the contents of the Cells in one Row of the solved game. The first line represents Row 1, the second Row 2 and so on.
- Line 26. This consists solely of the word "End" and signifies the end of the game definition.

5.3 LOG FILES

If you set the monitor level to 2 or above, then each time you load a game the app creates a log file in the Android/data/com.pcdbdev.pcdbdevsolver/files/logs folder. The name of the file is the same as the name of the game which it logs.

6 ERROR MESSAGES

A number of error messages can occur while defining, solving or playing a SuDoku game. They are listed here.

6.1 ERRORS WHILE DEFINING A GAME

Message	***Title***	***Meaning***
Max number of Cells in a Group is nine	Select Cell	
This Cell is already part of a different Group	Select Cell	You clicked on a Cell that is already defined as part of a different Group
Select a Cell next to an already selected Cell	Select Cell	You clicked on a Cell this is not immediately above, below or to the side of an already selected Cell
You selected more than 9 Cells	Select Cells	You selected more than nine Cells
Group must be numeric	*Set Group Mon*	*You specified a non-numeric value for the Group to monitor*
Mon must be in the range 0 to 3		*You specified a negative value, or one greater than 3, for the monitor level*
Please enter a numeric value	Set Group Sum	You tried to enter a non-numeric value for the sum of a Group
Please select the Cells that comprise this Group	Set Group Sum	You tried to enter a Group sum without first selecting one or more Cells
Sum (*n*) is too small for a Group of this size (*m*)	Set Group Sum	You entered a sum value that is too small for the number of Cells selected. For example, a 4-Cell Group must have a minimum sum of 10
Sum (*n*) is too large for a Group of this size (*m*)	Set Group Sum	You entered a sum value that is too large for the number of Cells selected. For example, a 3-Cell Group must have a maximum sum of 24
Sum of Group sums must be 405; this is *nnn*. Please modify one or more Group sums.	Set Group Sum	At the end of the definition phase for a Killer SuDoku game, the sums of all the Groups defined don't add up to 405
Cell (*r*,*c*) has excluded solution value *nn*	Set Cell	You tried to eliminate a Candidate that is actually the proper solution for this Cell
This move (*r*,*c*:*n*) is not allowed	Set Cell	You tried to enter a solution value *n* that isn't one of the remaining Candidates for this Cell
Value must be between 1 and 9 (inclusive)	Set Cell	You tried to enter a Cell value outside the range 1 to 9

6.2 ERRORS WHILE LOADING A GAME

Message	***Title***	***Meaning***
Invalid file header: *line*	Load Game	
Line: Group *nn* does not exist	Load Game	
Game incomplete, Cells=*nn*, Sum=*mm*	Load Game	

6.3 ERRORS WHILE PLAYING A GAME

Message	*Title*	*Meaning*
Please select one or more adjacent Houses	Find Innies and Outies	
Please select a Group	Group Mandatories Group Options Group Patterns	
Group must be numeric	*Set Group Mon*	*You specified a non-numeric value for the Group to monitor*
Mon must be in the range 0 to 3		*You specified a negative value, or one greater than 3, for the monitor level*
You can't enter the same value in multiple Cells	Play (r,c:n)	You tried to enter a single value in more than one selected Cell
You can't enter a single value for a Group	Play	You tried to enter a single value after selecting a Group
You can't enter a single value for a House	Play	You tried to enter a single value after selecting a House
Cell (*r*,*c*) has excluded solution value *nn*	Set Cell	You tried to eliminate a Candidate that is actually the proper solution for this Cell
This move (*r*,*c*:*n*) is not allowed	Set Cell	You tried to enter a solution value *n* that isn't one of the remaining Candidates for this Cell
Cell (*r*,*c*) already set to n	Set Cell	You tried to enter a solution value in a Cell that already has a solution value *n*
Value must be between 1 and 9 (inclusive)	Set Cell	You tried to enter a solution value outside the range 1 to 9

Any of the following error messages is an indication either that the game you are trying to solve is unsolveable, or that you have gone wrong at some stage.

Message	*Title*	*Meaning*
No options left for G*nn*	Group Options	At some stage during the processing of a Group's Options, there are no Options left for the Group
Remaining Value not in 1..9	Reset Groups for(*r*,*c*:*n*)	As a result of the given move, a Group which it is in is now reduced to a single Cell, but the value for this Cell is outside the range 1 to 9
Only –ve Cell left in G*nn*	Group Options	
Options {*candsvals*} does not contain value *n*	Group Options	Subtraction Combo on the Group leads to a situation in which the sum of the Candidate values does not enable elimination of a value *n*
Cell (*n*,*m*) Sum < 1	Group Options	A Child Group differs from its Father by only one Cell, but the sum that this delivers for the residual Cell is less than 1
Cell (*n*,*m*) Sum > 9	Group Options	A Child Group differs from its Father by only one Cell, but the sum that this delivers for the residual Cell is greater than 9
G*gg* Max *mm* < Min *nn*	Group Options	
G*gg* Cell Vals <> *mm*		
Option is missing for Group *gg*	Group Patterns	
Option [*option*] Length > No of Cells *nn*		

No Candidates left for Cell (*r*,*c*)		

6.4 SYSTEM ERRORS

You may encounter any of the following error messages while playing a game.

Message	***Title***	***Meaning***
Sibling *nn* specified but no father	Various (Add Group)	
Peer SumMin > SumMax	Various (Add Group)	
Something wrong with (*cells*)	Various (Add Group)	
No of Cells in (*cells*) different from iCells (*n*)	Various (Add Group)	
Sibling *nn* father different from designated father *mm*	Various (Add Group)	
New SumMin *n* Greater than SumMax *mm*	Various (Add Group)	
SumMin < 1	Various	
SumMax > 9	Various	
G*gg* Pairs don't match	Group Options	Having processed the matching pairs in a two-Cell Group, the lengths of the remaining Candidates in the two Cells differ
House Range *nn*..*mm* >9	Find House Subgroups	
New Group SumMin *nn* Greater than SumMax *mm*	Find Subgroups	
Houses < 0	Set Group Bounds	
.Unconstrained < 0	Set Group Bounds	
NHouses < 0	Set Group Bounds	
.NUnconstrained < 0	Set Group Bounds	