

Word Game Solver Manual

Robert Gamble

January 2, 2014

version 1.0

Contents

| | | |
|----------|---|-----------|
| 1 | General Overview | 4 |
| 2 | Functional Summary | 4 |
| 3 | Command-line Tool | 5 |
| 3.1 | Board Generation | 5 |
| 3.2 | Board Scoring | 6 |
| 3.3 | Board Solving | 6 |
| 3.4 | Board Analysis | 9 |
| 3.5 | Board Validation and Word Checker | 10 |
| 4 | Other | 11 |
| 4.1 | Scoring Details | 11 |
| 4.2 | Board Notation | 11 |
| 4.3 | Board Coordinates | 12 |
| 5 | Configuration | 14 |
| 5.1 | Grids | 14 |
| 5.2 | Scoring Rules | 15 |
| 5.3 | Dictionaries | 18 |
| 5.4 | LetterDistributions | 18 |
| 5.5 | Preferences | 20 |
| 5.6 | Game Rules | 22 |
| 6 | Graphical Interface | 23 |
| 6.1 | Graphical Solver | 23 |
| 6.1.1 | Board | 23 |
| 6.1.2 | Solution Grid | 24 |
| 6.1.3 | Solution Explorer | 24 |
| 6.1.4 | Board Entry | 24 |
| 6.1.5 | Random Board Generation | 24 |
| 6.1.6 | Patterns | 24 |
| 6.2 | Dictionary Configuration | 25 |
| 6.3 | Scoring Rules Configuration | 26 |
| 6.4 | Game Rules Configuration | 27 |
| 6.5 | Letter Distribution Configuration | 28 |
| 6.6 | Grid Designer | 29 |
| 7 | Performance and Benchmarks | 30 |
| 7.1 | Benchmark Details | 30 |
| 7.2 | Random Board Generation | 30 |
| 7.3 | Scoring and Solving | 30 |
| 7.4 | Board Validation | 31 |

| | |
|---|-----------|
| 7.5 Word Checker | 31 |
| 8 GNU Free Documentation License | 33 |

Copyright 2014 Robert Gamble.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

1 General Overview

Word Game Solver (WGS) is a command-line and GUI-based tool for generating, solving, and analyzing anagram and grid-based word search puzzles. The default configuration provides out of the box support for numerous versions of Boggle® including several 4x4 and 5x5 versions as well as the more recent 6x6 Super Big Boggle®, Zynga's Scramble with Friends, Word Hero, and the anagram-based games Quarrel and 7 Little Words. The provided configuration file is easily modified to support new games by specifying game rules, scoring options, letter distributions, and grid layouts.

2 Functional Summary

- Support for dice-based games (like Boggle®), letter propensity games (ala Scrabble®), and pre-defined word lists such as is used by many anagram-based games.
- Support for grid-based games of arbitrary layouts (4x4, 5x5, etc., as well as non-rectangular layouts) as well as anagram-based games.
- Fast random generation of games for all game types.
- Generation of high-scoring games for most game types.
- Simple JSON-based configuration file.
- Fast game solver provides customizable information about solved game boards including:
 - Details of every word found including the word, the positions on the board where each letter in the word was found, and scoring information for the word.
 - Board statistics including the number of unique words found, total board score, word count and scoring data for words using letters at particular positions, and count and scoring information for n-letter and n+-letter words.
 - Ability to report the best (highest-scoring) instance of duplicate words or to report all instances of duplicate words.
- Customizable game grid for grid-based games.
- Customizable scoring options including letter-based scoring, minimum word length, word length bonuses, letter and word multipliers, etc.
- Fast Board validator to determine if a given board is valid.
- Word checker that can determine if there exists a board on which it is possible to form a particular word.

3 Command-line Tool

The command-line tool is called `wgs` and performs various tasks depending upon the arguments provided on the command-line:

```
wgs cfg-file create game-name [boards [min-words [min-points [minimize]]]]
wgs cfg-file score game-name
wgs cfg-file solve game-name [format [prefix-string [suffix-string]]]
wgs cfg-file solve-dups game-name [format [prefix-string [suffix-string]]]
wgs cfg-file analyze game-type [format]
wgs cfg-file check-board game-type
wgs cfg-file check-word game-type [stats | verbose]
```

3.1 Board Generation

The `create` command requests the generation of one or more random boards which are printed to `stdout`. Each board is terminated by a newline. Board generation is governed by the letter distribution rules defined for the specified game.

For `WordList` games, a random entry is selected using a one-pass algorithm through the configured word list file and letters are shuffled if `ShuffleLetters` is set to `true`.

For `LetterPropensity` games, letters are randomly selected from the configured pool of letters and the `SampleWithoutReplacement` setting is honored. For `Dice` games, each die is rolled and the result is shuffled if `ShuffleDice` is set to `true`.

The resulting board string is then printed. This operation is performed as many times as necessary to generate the desired number of boards, the default is to generate a single board. Boards are not scored or analyzed in any other way and the dictionary file is not used (and does not need to be defined) for this operation. Basic board generation is always a fast operation.

If there are not enough letters or dice configured to fill the board then the result will be limited to the number of letters or dice that are configured. If more dice are configured than required the behavior is dependent upon the value of `ShuffleDice`: if set to `false`, the first `n` configured dice are used where `n` is the number of needed tiles, otherwise `n` randomly selected dice are used.

In addition to basic board generation, optimized board generation may be requested by setting a target number of words and/or a target score. In this case, an optimizer will take a randomly generated board and perform various manipulation operations in an attempt to reach the target criteria. For example, on a dice-based board, the operations include re-rolling individual die and swapping die positions in an attempt to increase the value of the board. The result is always a valid board. If the optimizer determines that the target is not likely to be met after a number of operations, it will return the best board it is able to create based on the starting board. Optimized board generation takes considerably longer than basic board generation as each change to the board requires the entire board to be re-scored which may occur thousands of times while trying to reach the target. The amount of time needed is largely a function of the target criteria and the starting board. The output of this operation is the generated board, the number of words, and the board score, all separated by a space and terminated with a newline. This operation is supported for `Dice` and `LetterPropensity` games.

In addition to requesting boards that meet or exceed certain scoring criteria, it is possible to specify boards where scores should be minimized by adding the `minimize` keyword on the command line. In this case, the board will be manipulated to minimize the board score, down to the target threshold.

If `min-words` and `min-points` are both zero or not specified and the `minimize` option is not provided then basic board generation will take place. Otherwise, if `minimize` is not specified then the optimizer will attempt to prepare a board that meets or exceeds both the `min-words` and the `min-points` criteria. If `minimize` is specified then the optimizer will attempt to create a board that does not exceed either criteria.

EXAMPLES

```

$ wgs wgs.json create "Boggle (New)"
AEVIHETAIHOOYIIO

$ wgs wgs.json create "Boggle (New)" 3
CAIIXOTITVIYRWEN
XKGHTHONASNOTYTE
NESWEJDHSTTVPDIR

$ wgs wgs.json create "Boggle (New)" 1 0 0
GONFTERHTUGVOSEV

$ wgs wgs.json create "Boggle (New)" 3 500
OSEBIRAHETLWDESS 506 1059
SIBHTANERERTEWSI 512 1349
UREBHARDSEIOTSNS 500 1222

$ wgs wgs.json create "Boggle (New)" 3 500 2000
EDLRNIESTAMCSERA 762 2063
ESLCMIEAANTRRESO 771 2118
STNDTEIELARSMRTO 706 2090

$ wgs wgs.json create "Boggle (New)" 3 0 0 minimize
RPGTDWMTLHHRJIHR 0 0
SPWRBGNSCHXQOVUY 0 0
HXGTSPCRHNHNSNLRB 0 0

```

3.2 Board Scoring

The **score** operation reads one game board per line from **stdin** and outputs the unique word count and board score, separated by a space. The **analyze** command can also be used to provide this information, along with lots of other information, but the **score** operation is somewhat more efficient if the capabilities provided by **analyze** are not needed.

If fewer letters are entered than used by the game type, the remaining will be blank and will not be used for scoring. If more letters are provided than used by the game type, the extra letters will be ignored.

EXAMPLES

```

$ wgs wgs.json score "Boggle (New)"
Enter letters (empty to quit): ABCDEFGHIJKLMNOP
18 20
STNDTEIELARSMRTO
706 2090

```

3.3 Board Solving

The **solve** and **solve-dups** commands read one game board per line from **stdin** and output the specified information for every word found in the board. The **solve-dups** command will output a solution entry for every word including duplicate spellings of the same word that consist of a unique path through the board (which may involve identical tiles as a previous solution). The **solve** command only outputs unique words, if the same word is found multiple times on the board, only the best (highest-scoring) is reported.

The **solve** commands take a format argument that specifies the information to be output for each found word. The format consists of literal characters and format specifiers that begin with a percent sign. The

format specifiers are replaced with the corresponding data for each solution and the result is printed.

| Format Specifier | Description |
|------------------|---|
| %w | The word in all uppercase letters. |
| %s | The overall score for the word including letter points, word multipliers, and length bonus. |
| %l | The letter points associated with the word. Includes letter multiplier bonuses but not word multipliers or length bonuses. |
| %m | The word multiplier for this word. If the word is made up of multiple letters that have word multipliers, the value is the product of all corresponding word multipliers. |
| %b | The length bonus for this word. |
| %p | The list of positions used to for this word. The %p specifier must be followed by the character used to separate the positions. For example, %p, indicates that positions be separated by commas. |
| %(...) | Expands to the literal text inside the parenthesis for all but the last solution printed. The same backslash escape sequences are expanded and \) can be used to include a literal closing parenthesis. |
| %% | A literal percentage sign. |

In addition, the following escape sequences are supported:

| Escape Sequence | Meaning |
|-----------------|-------------------|
| \t | Horizontal tab |
| \n | Newline sequence |
| \\ | Literal backslash |

Note: The solutions do not end with a newline unless one is specified in the format.

Two additional optional arguments are accepted, a prefix-string and a suffix-string which are printed before each set of solutions and after each set of solutions, respectively.

EXAMPLES

The below example prints each word and the point value in parenthesis with entries separated by commas, the %() specifier is used to print a comma and space after each solution except the last one:

```
$ wgs wgs.json solve "Scramble" "%w (%s)%(", )" "" "\n"
Enter letters (empty to quit): ABCDEFGHIJKLMNOP
AB (1), AE (1), BA (1), BE (1), EF (1), FA (1), FAB (9), FE (1), FI (1), FIE (6),
FIN (7), FINK (12), FINO (8), GLOP (10), IF (1), IN (1), INK (8), JIN (13),
JINK (18), JO (1), KNIFE (16), KNOP (12), KOJI (17), KOP (10), LO (1), LOP (7),
MI (1), MINK (12), NIM (7), NO (1), ON (1), OP (1), PLONK (17), POL (7)
```

In the following example, each word is printed out with the number of points and the positions of the grid tiles that are used to form the word.

```
$ wgs wgs.json solve "Scramble" "%w - %spts (%p,)\n"
Enter letters (empty to quit): ABCDEFGHIJKLMNOP
AB - 1pts (1,2)
AE - 1pts (1,5)
BA - 1pts (2,1)
BE - 1pts (2,5)
EF - 1pts (5,6)
FA - 1pts (6,1)
FAB - 9pts (6,1,2)
FE - 1pts (6,5)
```

```

FI - 1pts (6,9)
FIE - 6pts (6,9,5)
FIN - 7pts (6,9,14)
FINK - 12pts (6,9,14,11)
FINO - 8pts (6,9,14,15)
GLOP - 10pts (7,12,15,16)
IF - 1pts (9,6)
IN - 1pts (9,14)
INK - 8pts (9,14,11)
JIN - 13pts (10,9,14)
JINK - 18pts (10,9,14,11)
JO - 1pts (10,15)
KNIFE - 16pts (11,14,9,6,5)
KNOP - 12pts (11,14,15,16)
KOJI - 17pts (11,15,10,9)
KOP - 10pts (11,15,16)
LO - 1pts (12,15)
LOP - 7pts (12,15,16)
MI - 1pts (13,9)
MINK - 12pts (13,9,14,11)
NIM - 7pts (14,9,13)
NO - 1pts (14,15)
ON - 1pts (15,14)
OP - 1pts (15,16)
PLONK - 17pts (16,12,15,14,11)
POL - 7pts (16,15,12)

```

This example prints details of each solution in a JSON element notation:

```

$ wgs wgs.json solve "Scramble" '{ "word" : "%w",\n "letter-points": "%l",\n
"word-multiplier" : "%m",\n "length-bonus" : "%b",\n "total-points" : "%s",\n
"positions" : "[%p,]\n}%(\,)\n'
Enter letters (empty to quit): ABCDEFGHIJKLMNOP
{"word" : "AB",
"letter-points": "1",
"word-multiplier" : "1",
"length-bonus" : "0",
"total-points" : "1",
"positions" : [1,2]
},
{"word" : "PLONK",
"letter-points": "14",
"word-multiplier" : "1",
"length-bonus" : "3",
"total-points" : "17",
"positions" : [16,12,15,14,11]
},
...
{"word" : "POL",
"letter-points": "7",
"word-multiplier" : "1",
"length-bonus" : "0",
"total-points" : "7",
"positions" : [16,15,12]
}

```


3.4 Board Analysis

The `analyze` command reads one board per line from `stdin` and provides aggregate information about each board using a format string similar to that used by the `solve` command.

| Format Specifier | Description |
|-------------------|--|
| <code>%B</code> | The board as provided. |
| <code>%nW</code> | Count of distinct words that use the tile at position n . |
| <code>%nS</code> | Score of distinct words that use the tile at position n . |
| <code>%nC</code> | Count of distinct n -letter words. |
| <code>%nP</code> | Sum of points for all distinct n -letter words. |
| <code>%n+C</code> | Count of all distinct words with n or more letters. |
| <code>%n+P</code> | Sum of points for all distinct words with n or more letters. |
| <code>%nX</code> | Highest scoring n -letter word. |
| <code>%nY</code> | Score of highest scoring n -letter word. |

In addition, the following escape sequences are supported:

| Escape Sequence | Meaning |
|-----------------|-------------------|
| <code>\t</code> | Horizontal tab |
| <code>\n</code> | Newline sequence |
| <code>\\</code> | Literal backslash |

Note: Output does not end with a newline unless one is specified in the format.

Each of the format specifiers, except for `%B`, accept an optional integer n ; if n is not provided then the behavior is the same as if 0 were provided. `%W`, `%C`, and `%+C` all provide a count of the number of distinct word. `%S`, `%P`, and `%+P` all provide the sum of the scores of all distinct words, the highest-scoring word is used for each set of duplicates. `%X` and `%Y` yield the best scoring word on the board and the score of that word, respectively.

For `%nW` and `%nS`, only distinct words are considered but if a word can be spelled using different sets of tiles, each tile that can be used to spell the word is counted once for each tile. For example, in the board below:

| | | |
|---|---|---|
| S | T | R |
| X | P | I |
| X | X | P |

The word STRIP can be spelled two different ways, one using the P at position 5 and the other using the P as position 9. Both tiles 5 and 9 would include this word when calculating the value for `%5W`, `%9W`, `%5S`, and `%9S`. The tiles at positions 1, 2, and 3 would only have this word counted once as each word is only credited toward a tile one time.

EXAMPLES

Print each board with the number of words, the total board score, the best word on the board, and the score of the best word.

```
$ wgs wgs.json analyze "Boggle (New)" "%B: %C words, %S points. Best word is %X (%Y points)\n"
```

```
Enter letters (empty to quit): STNDTEIELARSMRTO
```

```
STNDTEIELARSMRTO: 908 words, 2632 points. Best word is ARENITES (11 points)
```

Number of 3 letter words and points, ... 8+ letter words and points:

```
$ wgs wgs.json analyze "Boggle (New)"
"3-letter words: %3C (%3P pts)\n4-letter words: %4C (%4P pts)\n
5-letter words: %5C (%5P pts)\n6-letter words: %6C (%6P pts)\n
7-letter words: %7C (%7P pts)\n8+-letter words: %8+C (%8+P pts)\n"
```

```

Enter letters (empty to quit): STNDTEIELARSMRTO
3-letter words: 97 (97 pts)
4-letter words: 208 (208 pts)
5-letter words: 236 (472 pts)
6-letter words: 194 (582 pts)
7-letter words: 105 (525 pts)
8+-letter words: 68 (748 pts)

```

3.5 Board Validation and Word Checker

The **check-board** command runs the board validator and the **check-word** command runs the word checker. Both commands read data from **stdin** where each line should contain either a candidate board or word. **check-board** determines if the provided board is a valid configuration based on the game's defined letter distribution and **check-word** determines if it is possible for the provided word to appear on some valid board configuration described by the game's letter distribution. Both operations are supported only for **Dice** and **LetterPropensity** games, **WordList** games are not supported.

Each word or board read is echoed back to the screen with a leading plus or minus sign indicating success or failure, respectively. Non-alphabetic characters are ignored by the **check-word** command when performing the check although they will be printed back out in the result. In addition to alphabetic characters, the wildcard symbol (?) is accepted by the **check-board** command. Boards should be written using the notation described in the "Board Notation" section.

The number of tiles provided for the **check-board** command may be more or less than the number specified by the game's grid size and still be considered a valid board by this command which only determines if the letters or dice provided in the game's configuration can be used to form the given board. Likewise, the **check-word** command will consider words to be valid even if they use more tiles than would normally appear on a valid board. Both commands do require that the board or word be able to be created with the set of configured dice or tiles though and no tiles or dice will be used multiple times to attempt to form the word (except for **LetterDistribution** games when **SampleWithoutReplacement** is false).

Game rules are not consulted to determine whether a word is valid when using the **check-word** command. For example, if the game rules define a minimum word length of 5 letters, a 4-letter word is still considered valid by this command as long as it can be formed on a valid board. Additionally, the checked word need not be in the game's configured dictionary to be valid.

If the **stats** option is specified, statistics about the algorithms used will be output to **stderr** at the end of the session. If the **verbose** option is provided, details about the status of each request will be printed to **stderr**.

EXAMPLES

```

$ wgs wgs.json check-board "Boggle (New)"
Enter word to check (empty to quit): ABCDEFGHIJKLMNOP
-ABCDEFGHIJKLMNOP
QADRICENTENNIALS
+QADRICENTENNIALS

```

The below example demonstrates that while there are several 17-letter words that can be spelled using a valid Boggle configuration, some common, shorter words cannot.

```

$ wgs wgs.json check-word "Boggle (New)"
Enter word to check (empty to quit): Inconsequentially
+Inconsequentially
Quadricentennials
+Quadricentennials

```

Sesquicentennials
+Sesquicentennials
Baby
-Baby
Waffle
-Waffle

17-letter words can be spelled using the "Qu" tile. While there are two "B"s and two "F"s on a set of Boggle dice, both "B"s are on the same die and both "F"s on another so no word with more than one B or F can be spelled in a game of Boggle.

4 Other

4.1 Scoring Details

Scoring is performed on all valid words found on a board and is used by the **score**, **solve**, and **analyze** commands as well as during optimized board generation. There are several aspects to scoring and the parameters that control them that are discussed below.

When a dictionary word is found, it is submitted for scoring which consists of the following steps:

1. Obtain the letter values associated with each tile used to spell the word as specified in the **LetterValues** parameter. A tile may contain multiple letters but typically consists of a single letter or "Qu". If the tile is a wild-card, letter points are assigned only if the **WildCardPoints** is set to true.
2. The letter value of each tile is then multiplied by any letter multipliers associated with the tile.
3. The product of all word multipliers for the tiles used in the word are calculated.
4. The base score is then calculated as the sum of the letter values (including letter multipliers) multiplied by the product of the word multipliers. For games that don't use letter-based scoring, this value will always be zero.
5. If the word is a short word (as defined by the **ShortWordLength** parameter), the score is the value associated with **ShortWordPoints**. If **ShortWordMultiplier** is set to true then this value is multiplied by the word multiplier, if any.
6. If the word is not a short word, the length bonus is calculated from the length of the word using the values from the **LengthBonuses** parameter. The word length is almost always equivalent to the number of letters in the word but can be less if **QIsQu** is true and **QuLength** is set to 1.
7. The length bonus is then added to the word score or, if **MultiplyLengthBonus** is set to true, it is multiplied by the word score. The resulting score is rounded down (truncated) to the nearest integer unless **RoundBonusUp** is set to true.

Some tiles may contain multiple letters. In such cases, the letter value of the tile is the sum of the values of all of the letters and any letter multipliers for the tile operate on this sum. The only exception is a word formed using a "Qu" tile when **QIsQu** is set to true in which case the value of the letter "Q" is used, not the value of the letters "Q" and "U".

It is possible for a multi-letter tile to begin with a wild-card, e.g. "?h" can be used as "Th", "Sh", etc. In this case, the individual letters are score separately so the wild-card letter may not receive points (depending on the value of **WildCardPoints**) but the remaining letters are eligible for points.

4.2 Board Notation

There is a specific notation used by WGS to express a given board as well as denoting multi-letter tiles, wildcard tiles, and letter and word multipliers. This notation is used when defining the dice for **Dice** based games, the letter pool for **LetterPropensity** games, and the entries in the **WordList** for word

list games as well as in generated boards and boards provided to the solver. This section describes that notation.

The board notation uses the uppercase letters A-Z, the lowercase letters a-z, the ? character for wildcards, a dot to represent an empty board position, and the colon and semi-colon characters for letter and word multipliers respectively. An uppercase letter or ? always starts a new board position whereas a lowercase letter always indicates a letter to be appended to the current tile. Tiles may consist of any number of letters but all of the letters must be used, in the order provided, to form a word with that tile. A multi-letter tile may start with a wildcard but a wildcard cannot be embedded inside of a multi-letter tile nor can a single tile have multiple wildcards. For example, “T” is a single-letter tile whereas “Th” is a multi-letter tile. “TH” would be interpreted as two, single-letter, tiles. “t” is not valid by itself because it is not preceded by an uppercase letter or a wildcard. “?at” represents a multi-letter tile that starts with any letter followed by “at”. Case is relevant only for determining which letters exist on which tiles, word finding is done case-insensitively.

It is also possible to attach letter and word multipliers to a tile such that scoring bonuses are realized for words utilizing such tiles. This is done by preceding the tile letter(s) with one or more colons and/or one or more semi-colons. Each colon increases the letter multiplier by one and each semi-colon increases the word multiplier by one. Each tile starts with a letter and word multiplier of one so one colon would make the tile a double-scoring tile. For example, “C:H::A;;T” consists of 4 tiles, a “C” tile with no multipliers, an “H” tile with a double-letter score, an “A” with a triple letter score, and a “T” with a triple word score. If C and H are worth 4 points each and A and T are each worth one point then the word CHAT would have a base score of $3 * (4 + (2 * 4) + (3 * 1) + 1) = 48$ points.

4.3 Board Coordinates

There are two coordinate systems used by WGS: an absolute coordinate system used for specifying the squares that constitute the board, and a board coordinate system used to specify the tiles of a specific grid.

The absolute coordinate system is only used in the Grids section of the configuration file. Absolute coordinates consist of an x,y pair starting with 1,1 at the top-left.

Board coordinates are used with several of the format specifiers for the **analyze** command to obtain statistics related to specific board locations. Board coordinates are also used when providing the word path via the %p format specifier of the **solve** command.

Board coordinates start at the number 1 and continue up to the number of positions defined for a Grid based game or to **MaxAnagramLetters** for anagram-based games. Position 1 starts at the top, left-most square on the board and positions progress left to right within a row. At the end of a row, the next position begins at the left-most square in the next row. Progression continues until the bottom right-most square is reached.

EXAMPLES

On a simple 4x4 grid, the board coordinates would be as follows:

| | | | |
|----|----|----|----|
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

Board coordinates for a non-rectangular board:

| | | | | | |
|----|----|----|----|----|----|
| | | 1 | 2 | | |
| | 3 | 4 | 5 | 6 | |
| 7 | 8 | | | 9 | 10 |
| 11 | 12 | | | 13 | 14 |
| | 15 | 16 | 17 | 18 | |
| | | 19 | 20 | | |

Absolute coordinates for the above board with tile coordinates that would be specified in the board geometry of the Grids section:

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----|-----|-----|-----|-----|-----|-----|---|---|---|----|
| 1 | | | 3,1 | 4,1 | | | | | | |
| 2 | | 2,2 | 3,2 | 4,2 | 5,2 | | | | | |
| 3 | 1,3 | 2,3 | | | 5,3 | 6,3 | | | | |
| 4 | 1,4 | 2,4 | | | 5,4 | 6,4 | | | | |
| 5 | | 2,5 | 3,5 | 4,5 | 5,5 | | | | | |
| 6 | | | 3,6 | 4,6 | | | | | | |
| 7 | | | | | | | | | | |
| 8 | | | | | | | | | | |
| 9 | | | | | | | | | | |
| 10 | | | | | | | | | | |

5 Configuration

The WGS configuration file contains all of the information necessary to support a specific game including the type of game, the board layout, the letter distribution, the scoring rules, and the dictionary to use. The file uses standard JSON format and can be modified using a text editor or the provided GUI configuration tool.

The configuration file contains several sections:

- Grids
- ScoringRules
- Dictionaries
- LetterDistributions
- GameRules
- Preferences

Each section contains zero or more named entries that define concrete aspects of a game. The **Grids** section is used to define the grid geometry, **ScoringRules** specify how words are scored, **Dictionaries** associates names to word lists, and **LetterDistributions** is where details of how boards are generated is stored. **Preferences** specify user-preferences such as the default format specifiers used by the **solve** and **analyze** commands as well as GUI customizations. **GameRules** tie everything together by associating a name with a combination of the other components. The layout allows aspects to be reused to easily create new games with minor variations on existing ones without having to specify everything from scratch. For example, a single entry for a 4x4 game grid can be used by all games that use such a grid.

5.1 Grids

The **Grids** section specifies board geometries. Each grid entry has two elements, **Tiles** and **Adjacency**.

The value for the **Tiles** parameter is an array of two-element arrays that correspond to the positions used on a 10x10 graph where the top left-hand corner is 1,1 and the bottom right-hand corner is 10,10. The first number is the x coordinate and the second number is the y coordinate. The order in which tiles are specified is immaterial.

The value for the **Adjacency** parameter is a string value that specifies the adjacency of tiles used to determine which paths along the board form valid words. The supported values are "Full", "Diagonal", and "Straight". "Full" means that every tile is considered adjacent to every other tile and is typically used for anagram games. Straight indicates that tiles that are touching (up/down/left/right) are considered adjacent. Diagonal indicates that any immediately surrounding tile (up/down/left/right/diagonal) is adjacent.

EXAMPLES

The example below defines a 4x4 grid with diagonal adjacency such as used by Boggle®.

```
"Grids" : {
  "4x4" : {
    "Tiles" : [
      [1,1], [1,2], [1,3], [1,4],
      [2,1], [2,2], [2,3], [2,4],
      [3,1], [3,2], [3,3], [3,4],
      [4,1], [4,2], [4,3], [4,4]],
    "Adjacency" : "Diagonal"
  }
}
```

The below example defines an 8-letter grid with full adjacency for an anagram game.

```
"Grids" : {
  "anagram" : {
    "Tiles" : [
      [1,1], [1,2], [1,3], [1,4], [1,5], [1,6], [1,7], [1,8]]
  }
}
```

```
        "Adjacency" : "Full"  
    }  
}
```

5.2 Scoring Rules

The ScoringRules section provides the details of how words are scored and consists of the following elements:

| Parameter | Value Type | Description | Default |
|---------------------|------------|--|---------|
| LetterValues | Object | An object containing <i>letter:value</i> pairs where <i>letter</i> is a letter and <i>value</i> is an integer value between 0 and 999 representing the letter's value. | None |
| LengthBonuses | Object | An object containing <i>length:value</i> pairs where length is an integer representing the length of the formed word and value is a decimal value representing the bonus associated with words of the given length. | None |
| QIsQu | Boolean | If true, any "Q" tile encountered on a board is automatically converted to "Qu". | True |
| ShortWordLength | Integer | The length of "short" words. Short words are valid words but may be scored differently (see below). | None |
| ShortWordPoints | Integer | The number of points assigned to short words. This value is used instead of summing up the values of each letter. Short words are not eligible for length bonuses. | 0 |
| ShortWordMultiplier | Boolean | Determines if short words are eligible for letter and word multipliers. | False |
| MinWordLength | Integer | The minimum word length. Words shorter than this value of not considered valid. | 1 |
| QuLength | Integer | The length of "Qu" when used in a word. Valid values are 1 and 2. Only applicable for words formed using a "Q" tile when QIsQu is true. Affects the length bonus for the word and whether the minimum word length is met. For example, the word QUIT formed using "Qu", "I", and "T" is considered 3 letters if QuLength is 1 and 4 letters if QuLength is 2. Only affects letters formed using a "Qu" tile, QUIT formed using the tiles "Q", "U", "I", and "T" is always considered 4 letters. | None |
| WildCardPoints | Boolean | Determines how a wild-card tile, represented by "?", is scored. If WildCardPoints is true then the value associated with the letter used for the wild-card in the formed word is used, otherwise no letter points are granted for the wild-card although it still counts toward the length of the word. | False |
| MultiplyLengthBonus | Boolean | If true, the corresponding length bonus is multiplied by the word score, otherwise the length bonus is added to the score. | False |
| RoundBonusUp | Boolean | If true, the length bonus is rounded up to the nearest integer, otherwise the value is rounded down. | False |
| RandomBoardSize | Integer | Specifies the maximum number of tiles to use when generating a random board. Random board generation typically fills the configured grid but this is not always desired. For example, if the grid contains 16 tiles but only 7 of the should be filled when requesting a random board (think Scrabble), this parameter can be set to 7. | |

EXAMPLES

Below is an example of the scoring rules for Boggle. There is no need for a **LetterValues** section (although one could be included) since Boggle does not use letter points, score is based solely on word length. This example uses **QIsQu** so that when the user enters a board tile of "Q", it is automatically converted to "Qu", otherwise "Qu" would have to be used to represent this value. There are no "short" words in Boggle so **ShortWordLength** and **ShortWordPoints** are set to zero, alternatively they could be removed from the configuration. Words must be at least 3 letters long which is enforced with **MinWordLength**. **QuLength** is set to 2 as Boggle counts "Qu" as a two letters.

```
"ScoringRules" : {
  "Boggle" : {
    "LengthBonuses" : {
      "1" : 0,
      "2" : 0,
      "3" : 1,
      "4" : 1,
      "5" : 2,
      "6" : 3,
      "7" : 5,
      "8" : 11,
      "9" : 11,
      "10" : 11,
      "11" : 11,
      "12" : 11,
      "13" : 11,
      "14" : 11,
      "15" : 11,
      "16" : 11,
      "17" : 11,
      "18" : 11,
      "19" : 11,
      "20" : 11
    },
    "QIsQu" : true,
    "ShortWordLength" : 0,
    "ShortWordPoints" : 0,
    "ShortWordMultiplier" : false,
    "MinWordLength" : 3,
    "QuLength": 2
  }
}
```

The below example provides scoring rules for Quarrel, an anagram finding game. Scoring in Quarrel is based solely on the values of the letters used. **QIsQu** is set to false as a "Q" tile cannot be used to mean "QU". There is no minimum word size in Quarrel so **MinWordLength** is set to 1.

```
"ScoringRules" : {
  "Quarrel" : {
    "LetterValues" : {
      "A" : 1,
      "B" : 5,
      "C" : 2,
      "D" : 3,
      "E" : 1,
      "F" : 5,
      "G" : 4,
      "H" : 4,
      "I" : 1,
      "J" : 15,
      "K" : 6,
      "L" : 2,
```

```

        "M" : 4,
        "N" : 1,
        "O" : 1,
        "P" : 3,
        "Q" : 15,
        "R" : 2,
        "S" : 1,
        "T" : 1,
        "U" : 3,
        "V" : 6,
        "W" : 5,
        "X" : 10,
        "Y" : 5,
        "Z" : 12
    },
    "RandomBoardSize" : 8,
    "QIsQu" : false,
    "MinWordLength" : 1
}
}

```

5.3 Dictionaries

A dictionary consists of a single name:path pair where *name* a string containing the name given to the dictionary and *path* is a string representing the location of the dictionary file. The dictionary file itself should consist of a series of valid words separated by whitespace (such as newlines). The dictionary file need not be sorted and words are read case-insensitively.

EXAMPLE

The below example defines a dictionary named “Common” at the path /usr/share/dict/words.

```

"Dictionaries " : {
    "Common" : "/usr/share/dict/words",
}

```

The full pathname should typically be specified, relative pathnames are relative to the directory in which the application is started.

5.4 LetterDistributions

This section specifies how letters are distributed when generating a random board and validating an entered board. It is also used for using the Check Word feature which determines if a word can ever exist on any valid board.

| Parameter | Value Type | Description | Default |
|--------------------------|------------|--|---------|
| GenerationMethod | String | One of "Dice", "LetterPropensity", or "WordList". | None |
| DiceLetters | String | Used only for "Dice" games. A string consisting of dice specifications delimited by commas. Each dice specification includes one or more letters. | None |
| ShuffleDice | Boolean | Indicates whether dice are shuffled during random board generation. If false, the first die defined in DiceLetters will always be placed in the top-left-most location on the board, etc. | true |
| PropensityLetters | String | Used only for "LetterPropensity" games. Specifies the pool of letters from which to draw for random board creation. | None |
| SampleWithoutReplacement | Boolean | Specifies whether letters are available for re-use. If true, letters may not be reused (unless the same letter appears multiple times), otherwise the letter is available to be drawn again. | true |
| WordListFile | String | The name of the file containing the word list from which to generate random games. | None |
| ShuffleLetters | Boolean | Specifies whether letters are scrambled in WordList games. If false, the letters are loaded in the order in which they appear in the WordListFile. | true |

EXAMPLE

The below example contains configuration for several games. For Scrabble, the **GenerationMethod** specifies that the letters come from a pool and the **SampleWithoutReplacement** parameters specifies that the each letter is removed from the pool as it is used. Two versions of boggle dice are represented. Lastly, Quarrel does not use a letter pool or dice but rather pulls random words from a list of 8-letter words so its **GenerationMethod** is set to **WordList** and a **WordListFile** is provided.

```

"LetterDistributions" : {
  "Scrabble" : {
    "GenerationMethod" : "LetterPropensity",
    "PropensityLetters" : "AAAAAAAAABCCDDDEEEEEEEEEEEFFGGGHH
                          IIIIIIIJKLLLLMMNNNNNNNOOOOOOOPPPQ
                          RRRRRRSSSTTTTTUUUVVWWXXYYZZ??",
    "SampleWithoutReplacement": true
  },
  "Boggle (Old)" : {
    "GenerationMethod" : "Dice",
    "DiceLetters" : "AACIOT, AHMORS, EGKLUY, ABILTY, ACDEMP, EGINTV,
                    GILRUW, ELPSTU, DENOSW, ACELRS, ABJMOQ, EEFHIY,
                    EHINPS, DKNOTU, ADENVZ, BIFORX",
    "ShuffleDice" : true
  },
  "Boggle (New)" : {
    "GenerationMethod" : "Dice",
    "DiceLetters" : "AAEEGN, ELRTTY, AOOTTW, ABBJOO, EHRTVW, CIMOTU,
                    DISTTY, EIOSST, DELRVY, ACHOPS, HIMNQU, EEINSU,
                    EEGHNW, AFFKPS, HLNNRZ, DEILRX",
    "ShuffleDice" : true
  },
},

```

```
"Quarrel" : {  
  "GenerationMethod" : "WordList",  
  "WordListFile" : "/home/rgamble/projects/wgs/anagrams.txt",  
  "ShuffleLetters" : true  
}  
}
```

5.5 Preferences

The preferences section contains user-preferences including defaults for optional command parameters and GUI customization. All parameters take a string value.

| Parameter | Description | Default |
|----------------------|--|--|
| BoardSummaryFormat1 | The label that appears <i>above</i> the grid in the GUI solver. May contain the same format specifiers described in "Board Analysis" as well as the backslash escape sequences described there. | 2:%2C 3:%3C 4:%4C 5:%5C 6:%6C 7:%7C 8+:%8+C |
| BoardSummaryFormat2 | The label that appears <i>below</i> the grid in the GUI solver. May contain the same format specifiers described in "Board Analysis" as well as the backslash escape sequences described there. | %W words, %S points |
| ShowPattern | Determines whether the regular expression pattern entry box is visible in the GUI. | true |
| ShowRandom | Determines whether the random board generation button and related options are visible in the GUI. | true |
| ShowOptions | Determines whether the "Show Solutions" and "Show Duplicates" checkboxes are visible in the GUI. | true |
| ShowValid | Determines whether the valid board indicator is visible in the GUI. | true |
| SolutionFormat | The default format for <code>solve</code> and <code>solve-dups</code> command. | %w:\t%s points\n |
| AnalysisFormat | The default format for the <code>analyze</code> command. | %B: %W words, %S points\n |
| SolutionPrefix | The default solution prefix for the <code>solve</code> and <code>solve-dups</code> command. | None |
| SolutionSuffix | The default solution suffix for the <code>solve</code> and <code>solve-dups</code> command. | None |
| ToolTipFormat | The message that appears when hovering over a grid tile in the GUI solver. May contain the same format specifiers as used by the <code>analyze</code> command. In addition, an asterisk may be used in place of the tile position for the %W and %S format specifiers which will calculate and display the appropriate values for each tile. | /*W words, /*S points |
| SolutionDetailFormat | The text that appears in the solution explorer section of the GUI below the grid when a particular solution is selected. May contain the format specifiers used by the <code>solve</code> command as well as html markup. | None |
| TileSize | The pixel width and height of the grid tiles in the GUI tool. | 37 |
| Font | Change the font used in the GUI tool. | None |

In addition to regularly-named preference sets, there may be a *default* preference set with a name of Default. If such a default preference set exists, any parameters not supplied by the preference set associated with a given game rules will be implicitly set by the values in the default preference set.

EXAMPLES

Below is an example value for the `SolutionDetailFormat` which displays details of the solution in a table format.

```

<table width=\"150\">
  <tr>
    <td><strong>%s</strong></td>
    <td align=\"right\"><strong>%w</strong></td>
  </tr>
  <tr>
    <td>%l</td>
    <td align=\"right\">Letter Points</td>
  </tr>
  <tr>
    <td>x%m</td>
    <td align=\"right\">Word Multiplier</td>
  </tr>
  <tr>
    <td>%b</td>
    <td align=\"right\">Length Bonus</td>
  </tr>
</table>

```

The backslashes before the quotes are necessary to distinguish them from the quotes that end the JSON string. The result looks something like this:

| | |
|-----------|------------------|
| 34 | TRAPLINES |
| 14 | Letter Points |
| x1 | Word Multiplier |
| 20 | Length Bonus |

5.6 Game Rules

The Game Rules section defines a specific game type by tying together entries from the other sections. Each parameter in the game rules section takes a string value which corresponds to the named entry of the associated section. There are no defaults for any of the parameters. It is not required that all parameters are present but functionality may be limited by the lack of certain components. For example, it will not be possible to generate or validate boards without a **LetterDistribution** entry although it would still be possible to score and solve user-provided boards.

| Parameter | Description |
|--------------------|--|
| GridDesign | The name of the GridDesign entry applicable to this game type. |
| ScoringRules | The name of the ScoringRules entry associated with this game type. |
| LetterDistribution | The LetterDistribution for this game type. |
| Dictionary | The Dictionary to use for this game type. |
| Preferences | The Preferences to use for this game type. |

EXAMPLE

```

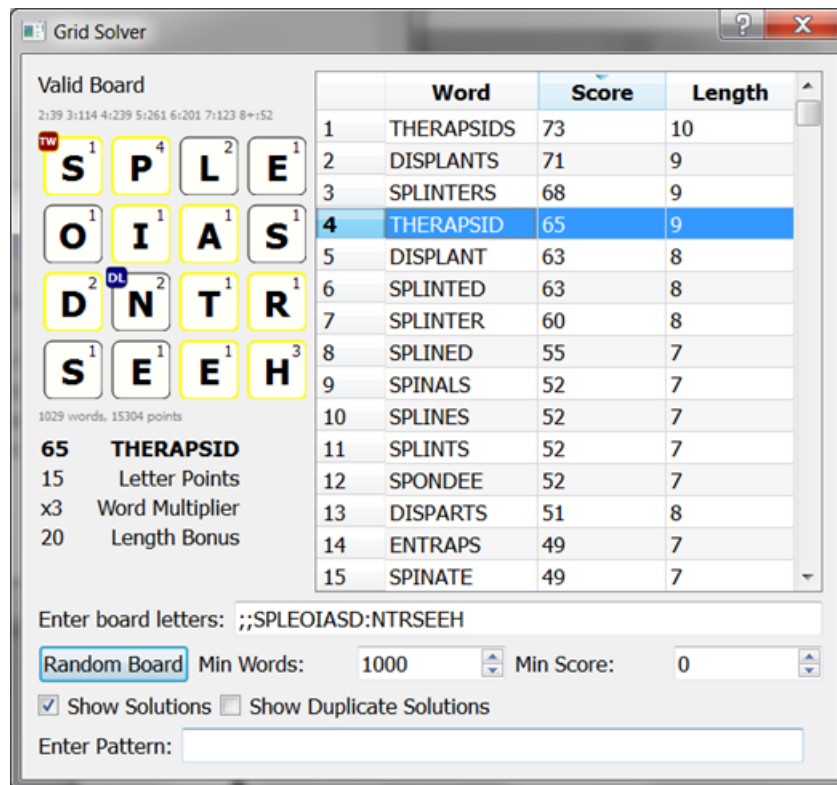
"GameRules" : {
  "Boggle (New)" : {
    "GridDesign" : "4x4",
    "ScoringRules" : "Boggle",
    "LetterDistribution" : "Boggle (New)",
    "Dictionary" : "SOWPODS"
  }
}

```

6 Graphical Interface

The graphical interface provides interactive configuration and solving functionality.

6.1 Graphical Solver



The graphical solver is divided into several parts.

6.1.1 Board

The board on the upper-left maintains a visual representation of the game and includes the letters present on each tile, the tile scores, and the presence of letter and word multipliers. Immediately above and below the board are board analysis summary details which are configurable by setting the `BoardSummary1` and `BoardSummary2` preferences. The data presented in the board summaries is not effected by *starts with* or *contains* constraints or pattern constraints. A valid board indicator is displayed above the board if the game is configured with `PropensityLetters` or `DiceLetters`.

Hovering over a board tile will provide details about that tile which is configurable via the `HoverText` preference, by default the number or words that can be spelled with the tile and the total score of those words is displayed. The data presented in the hover text is not effected by *starts with* or *contains* constraints or pattern constraints.

Board tiles can also be clicked on. Left clicking a tile causes it to cycle between the modes of *normal*, *start with*, and *contains*. In the *normal* mode, the tile letters are black, in the *starts with* mode the letters are red, and in the *contains* mode the letters are green. When a tile is in the *starts with* mode, only solutions that begin with this tile are shown in the solution list. When a tile is in the *contains* mode, only solutions that contain this tile are shown. If multiple tiles are in the *contains* mode then only words that contain all of the corresponding tiles are shown. If multiple tiles are in the *starts with* mode then solutions starting with any of the corresponding tiles is shown. If the board contains tiles in both the *starts with* and *contains* modes, only solutions that satisfy both sets of constraints are shown. For example, in Figure 2, only solutions starting with the S in the top left hand corner and containing P, L, and N are shown.

The boarder of the tiles is normally black but when a solution is selected from the solution grid, the tiles that correspond to that solution are highlighted as shown in Fig 1.

6.1.2 Solution Grid

The solution grid on the right side contains all of the words that can be spelled using the current game board. The set of solutions shown can be constrained by the "starts with" and "contains" tiles as well as a user-provided pattern.

For each solution, the word, score, and length is provided. The solutions may be sorted by any of these attributes by left-clicking the appropriate header, clicking again will sort in opposite direction.

Clicking on a solution will select it and update the board to highlight the corresponding tiles as well as the solution explorer.

The solution grid can be temporarily hidden by unchecking the "Show Solutions" checkbox. Checking the "Show Duplicate Solutions" will show duplicate solutions that have a unique path through the board.

6.1.3 Solution Explorer

The solution explorer provides details about the selected solution. The default is a table that contains the word, points, and a breakdown detailing how the points are calculated. The information provided in the solution explorer is completely configurable via the `SolutionDetailFormat` preference.

6.1.4 Board Entry

This is where boards are manually entered. As described in the "Board Notation" section, tiles are populated left-to-right, top-to-bottom and may begin with a combination of colons and semi-colons to represent letter and word multipliers respectively. Uppercase letters start a new tile, lowercase letters continue a tile. A dot (period) can be used to represent an empty tile.

6.1.5 Random Board Generation

Clicking on the "Random Board" button will cause a random board to replace the existing board if this operation is supported by the select game type. If Min Words or Min Score are non-zero, an attempt will be made to find a board that meets these criteria.

6.1.6 Patterns

A regular expression can be entered into the pattern box at the bottom of the solver to limit the solutions shown to those that match the pattern. All matches are performed case-insensitively and are not anchored by default (use `^` and `$` to anchor expressions). The current implementation uses Qt's regular expression engine which is very similar to Perl regular expressions. Notable deviations from Perl include:

- `^` and `$` always indicates the beginning or end of a string when used outside of a character class.
- Non-greedy matches are not supported.
- Comments are not supported within expressions.
- Look-behind assertions and conditional expressions are not supported.

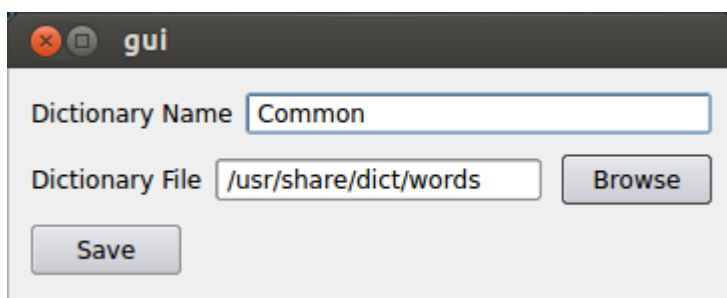
The tables below provide an overview of the basic functionality provided by regular expression patterns and some useful examples, for more details refer to <http://qt-project.org/doc/qt-4.8/qregexp.html>.

| Metacharacter | Description |
|---------------|---------------------------|
| \ | Escape next metacharacter |
| ^ | Match beginning of string |
| \$ | Match end of string |
| . | Match any character |
| | Pattern alternation |
| () | Grouping |
| [] | Character class |

| Quantifier | Description |
|------------|-----------------------------------|
| * | Match 0 or more times |
| + | Match 1 or more times |
| ? | Match 0 or 1 time |
| {n} | Match n times |
| {n,} | Match n or more times |
| {n,m} | Match at least n times, up to m |
| {,m} | Match zero or more times, up to m |

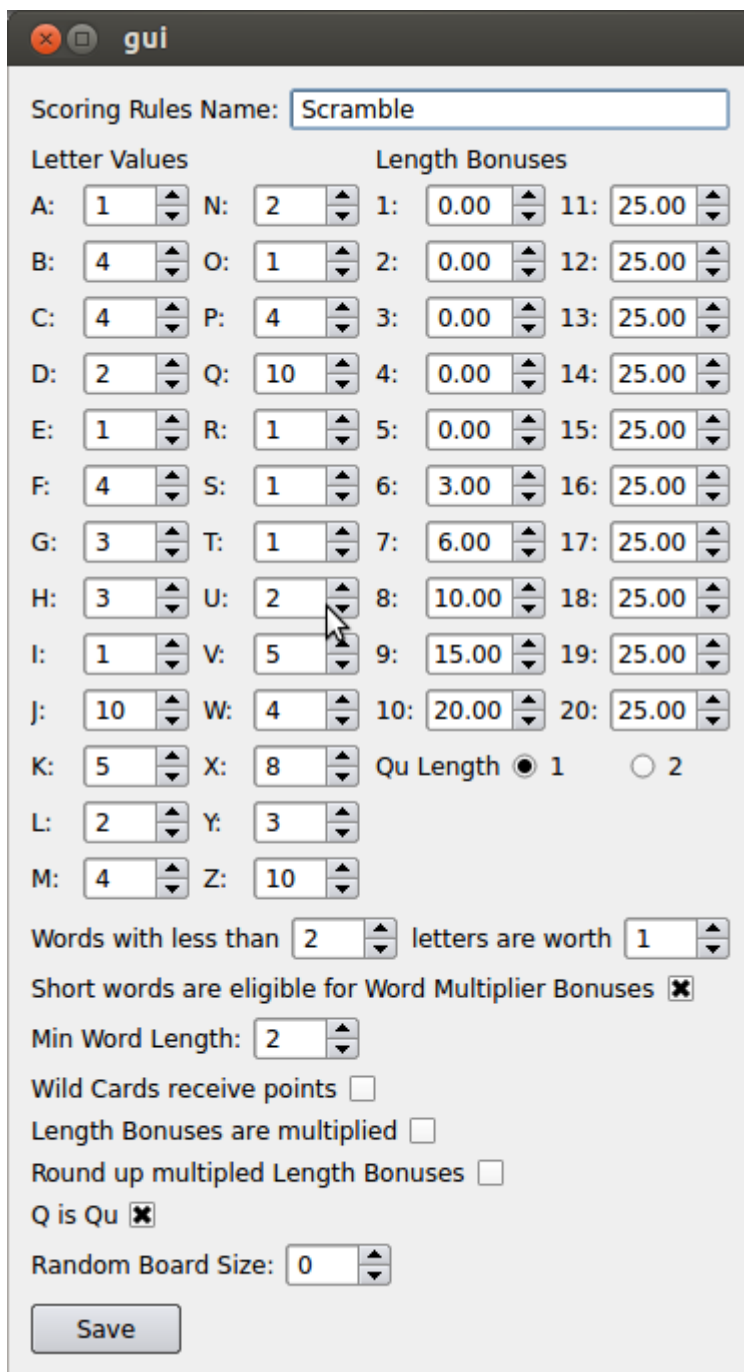
| Pattern | Description |
|-----------------|--|
| cat | Match words that contain the sequence <code>cat</code> . |
| ^cat | Match words that begin with <code>cat</code> . |
| cat\$ | Match words that end in <code>cat</code> . |
| | Match words with at least 5 letters. |
| .{5} | Match words with at least 5 letters. |
| ^. {5}\$ | Match words with exactly 5 letters. |
| ^(...t g...m)\$ | Match 5 letter words that end with <code>t</code> or that start with <code>g</code> and end with <code>m</code> . |
| ^tr.?[scde]+\$ | Match words that start with <code>tr</code> possibly followed any one letter and ending in a combination or one or more of the letters <code>s</code> , <code>c</code> , <code>d</code> , and <code>e</code> . |

6.2 Dictionary Configuration



The dictionary configuration dialog allows the creation and modification of dictionaries for use in game rule definitions. Select a name for the dictionary and browse to an existing file or enter a file path manually and click Save to update or add the dictionary entry.

6.3 Scoring Rules Configuration



The image shows a software window titled "gui" with a "Scoring Rules Name" field containing the text "Scramble". Below this, there are two main sections: "Letter Values" and "Length Bonuses".

Letter Values: This section contains a grid of input fields for letters A through Z. Each letter has a corresponding numerical value. For example, A is 1, B is 4, C is 4, D is 2, E is 1, F is 4, G is 3, H is 3, I is 1, J is 10, K is 5, L is 2, M is 4, N is 2, O is 1, P is 4, Q is 10, R is 1, S is 1, T is 1, U is 2, V is 5, W is 4, X is 8, Y is 3, and Z is 10. Each input field has up and down arrows for adjustment.

Length Bonuses: This section contains a grid of input fields for word lengths from 1 to 20. Each length has a corresponding numerical value. For example, 1 is 0.00, 2 is 0.00, 3 is 0.00, 4 is 0.00, 5 is 0.00, 6 is 3.00, 7 is 6.00, 8 is 10.00, 9 is 15.00, 10 is 20.00, 11 is 25.00, 12 is 25.00, 13 is 25.00, 14 is 25.00, 15 is 25.00, 16 is 25.00, 17 is 25.00, 18 is 25.00, 19 is 25.00, and 20 is 25.00. Each input field has up and down arrows for adjustment.

Below the length bonuses, there is a "Qu Length" section with two radio buttons: "1" (selected) and "2".

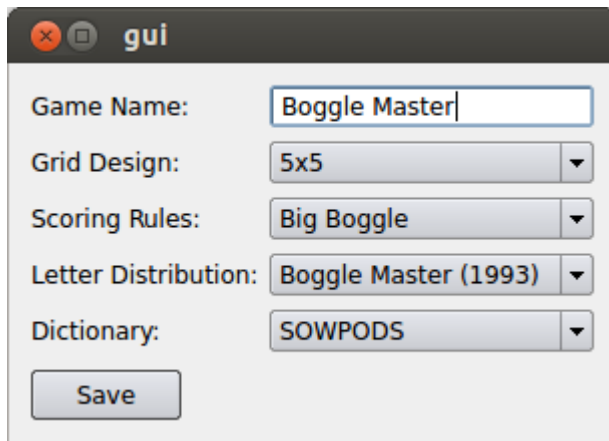
At the bottom of the dialog, there are several options and a "Save" button:

- "Words with less than 2 letters are worth 1" (with a dropdown for the number 2 and an input field for the value 1).
- "Short words are eligible for Word Multiplier Bonuses" (checked with an 'x' icon).
- "Min Word Length: 2" (with a dropdown for the number 2).
- "Wild Cards receive points" (unchecked checkbox).
- "Length Bonuses are multiplied" (unchecked checkbox).
- "Round up multiplied Length Bonuses" (unchecked checkbox).
- "Q is Qu" (checked with an 'x' icon).
- "Random Board Size: 0" (with a dropdown for the number 0).
- A "Save" button.

The scoring rules configuration dialog is used to create or edit scoring rules to use when defining new games. The "Letter Values" specify the points earned for using each letter or zero for games that do not use letter values. "Length Bonuses" are used to associate points with word lengths, games that do not use points based on word-length should have zero values. Decimal values are supported for length bonuses, this is intended to be used with the "Length Bonuses are multiplied" option to support games like Word Hero. "Qu Length" indicates whether a Q tile counts as 1 or 2 letters for length bonus purposes when using the "Q is Qu" option.

The remaining options specify detailed rules for scoring specific cases, "Random Board Size" is used to specify a random board size smaller than the configured board geometry.

6.4 Game Rules Configuration



The screenshot shows a window titled "gui" with a dark header bar. Inside, there are five configuration options, each with a label and a control element:

- Game Name:** A text input field containing "Boggle Master".
- Grid Design:** A dropdown menu showing "5x5".
- Scoring Rules:** A dropdown menu showing "Big Boggle".
- Letter Distribution:** A dropdown menu showing "Boggle Master (1993)".
- Dictionary:** A dropdown menu showing "SOWPODS".

At the bottom left of the dialog is a "Save" button.

The Game Rules dialog is used to define a new game. Use the drop-down buttons to associate the combination of a grid design, the scoring rules, letter distribution, and dictionary to a game name.

6.5 Letter Distribution Configuration

gui

Letter Distribution Name:

Generation Method:

Dice Letters:

AEEEEEM, ABDEIO, ...EIO, HOPRST, CEIPST,
AFIRSY, AAEEOO, ADENNN, IPRSY, BBJKXZ,
DHLNOR, EHILRS, OOOTTU, AEILMN,
JKQuWXZ, DHHLOR, AEEGMU, CEIITT, EILPST,
EIILST, GORRVW, NOOTUW, AAEEEE, AAFIRS,
AEINO, AnErHeInThQu, DHHNOW, ENSSSU,
CDDLNN, AEGMNN, EOMTTT, CCENST,
AAAFRS, HIRSTV, CFGNUY, DDHNOT

Shuffle Dice ☒

Propensity Letters:

Sample without Replacement ☒

Word List File:

Shuffle Letters ☐

The Letter Distribution Dialog is used to specify the source of randomly generated letters for a game. A Generation Method must be selected, available options are "Dice", "LetterPropensity", and "WordList".

For Dice based games, the faces of each die should be provided with dice separated with commas. There may be more or less dice defined than the number of tiles in the grid design. Each die may have any number of sides. The "Shuffle Dice" checkbox indicates whether randomly generated boards present the dice in the configured order or mix them up.

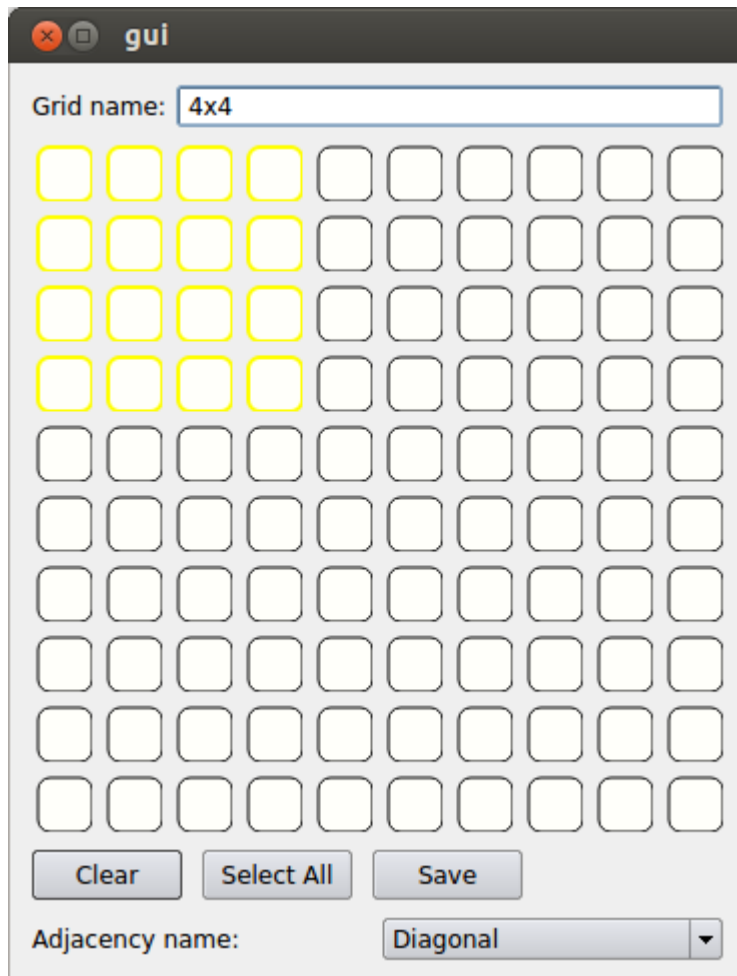
For LetterPropensity games, a pool of letters is provided. All letters should be upper-case except for faces that contain multiple letters in which case the first letter should be upper-case and the remainder should be lower-case. There may be more or less dice defined than the number of tiles in the grid design. If the "Sample without Replacement" box is checked then each letter will not be used more than the number of times it appears in the list when generating a random board.

For Word List games, specify the name of a file that contains one set of letters, per line. A random

entry from this file will be used to populate the board when random board generation is requested. If "Shuffle Letters" is checked, the positions of the letters from the entry will be shuffled, otherwise the letters will be placed onto the board in the order in which they appear in the file.

For all game types, the letters provided for "Dice Letters", "Propensity Letters", or in the word file should be upper-case except for die faces or tiles that contain multiple letters in which case the first letter should be upper-case and the remainder should be lower-case. A dot can be used to indicate a face or tile is empty and cannot be used in the formation of a word. A question mark can be used to indicate a wild-card. Other characters will be ignored.

6.6 Grid Designer



The Grid Designer is used to specify the board geometry used by a game. Any game whose geometry can be specified by selecting tiles on a 10x10 grid is supported. Click on a tile to select it as part of the board, click again to deselect. Selected tiles will be highlighted. Empty rows and columns will be elided from the board during analysis and will not show up in the solver. Tiles do not need to form a rectangular shape. The "Adjacency" is one of "Diagonal", "Straight", or "Full". "Full" means that every tile is considered adjacent to every other tile and is typically used for anagram games. "Straight" indicates that tiles that are touching (up/down/left/right) are considered adjacent. "Diagonal" indicates that any immediately surrounding tile (up/down/left/right/diagonal) is adjacent.

7 Performance and Benchmarks

One of the goals of WGS is high-performance without sacrificing functionality, customizability, or code maintainability. The use of sensible algorithms and profile-based optimizations help ensure a reasonably high level of performance. Performance considerations for various operations as well as benchmarks are provided below.

7.1 Benchmark Details

The benchmark numbers provided below are based on the following configuration:

| | |
|------------------|--------------------------------|
| Operating System | Windows 7 SP1 |
| Memory | 8GB RAM |
| Processor | Intel Core i7-3720QM @ 2.60GHz |
| Compiler | GCC 4.6.2 (MinGW) |

7.2 Random Board Generation

Basic random board generation is always a very fast operation as there is little work to be done. Optimized board generation performs various manipulations on randomly generated board utilizing simulated annealing in an attempt to improve the board. These operations are non-deterministic with the amount of work being a function of the board size, the target criteria, the letter distribution, the size of the dictionary, and the initial board generated. Use of wildcards often increases scoring times, sometimes by an order of magnitude or more.

The below table shows the number of boards per second that are generated using the Boggle (Old) configuration with the TWL06 dictionary. The average random Boggle board contains about 93 words with only 3% of boards containing 200 or more words and 0.15% of boards containing 300 or more words.

| Min Words | Boards per second |
|-----------|----------------------|
| 0 | 56,034 |
| 1 | 1,771 |
| 100 | 690 |
| 200 | 83 |
| 300 | 49 |
| 400 | 22 |
| 500 | 9 |
| 600 | 3.5 |

Generating boards without a scoring criteria is very fast because there is no board solving involved, solving is required only when a non-zero criteria in order to determine if the criteria has been met. A criteria of one word is provided as a baseline, 99.9% of boards contain at least one word. The time taken slightly more than doubles as the criteria increases by 100. 100 words is an outlier which is due to the fact that most boards contain around 100 points to begin with and those that don't generally don't take much work to get them to 100 words. Optimized board generation is the area that has probably received the least amount of attention with regards to speed optimization, the algorithm can probably be improved significantly but being able to generate 10 boards/sec that lie outside of the 99.999% mark of random board generation is not too shabby.

7.3 Scoring and Solving

The `score`, `solve`, and `analyze` commands are both relatively fast operations with the time spent depending largely on the board size and word density of the board, boards with more words will naturally take longer to solve than boards with few words or more paths that can be pruned early on. Scoring is a slightly faster operation than solving and both are faster than analyzing because the analysis command

first solves and then instantiates a class that studies the solution set to generate aggregate analysis information. Boards with multiple wildcards can also quickly increase the set of valid paths and the solution size exponentially.

To produce the data in the table below, the solve and analyze operations were performed on six sets of boards. Two games were used: Boggle (New), a 4x4 dice-based grid game, and Big Boggle (1979), it's 5x5 cousin. A file of random boards were created for each game. A file of low-scoring boards and high-scoring boards were also created. The low scoring boards contain, on average, less than 10 words per board. The high-scoring boards contain about 2.5 times as many words per board as the random boards. The SOWPODS dictionary was use for both games. An empty format string was used for both operations to demonstrate the speed of the solver.

| Game Type | Avg Words per Board | Solving Boards per Second | Analyzing Boards per Second |
|----------------------------|--------------------------------|--------------------------------------|--|
| Boggle - Random Boards | 134 | 2500 | 2250 |
| Boggle - Low Boards | 8 | 20000 | 17500 |
| Boggle - High Boards | 322 | 900 | 800 |
| Big Boggle - Random Boards | 256 | 950 | 850 |
| Big Boggle - Low Boards | 9 | 7150 | 6750 |
| Big Boggle - High Boards | 642 | 450 | 390 |

7.4 Board Validation

Board validation is always solved with a fast, deterministic algorithm with a time complexity dependent solely on the number of dice/letters in the candidate set and the number of tiles in the candidate board.

Validating random Boggle Boards Validating random Big Boggle Boards Validating random Super Big Boggle Boards Validating random Scrabble Boards

7.5 Word Checker

The word checker uses what is probably the most complex set of algorithms employed by WGS. For games that employ only single-letter tiles, this is always a fast operation using the same algorithm as board validation. For games that utilize multi-letter tiles, such as “Qu”, “Th”, etc., the problem turns into a set cover constraint satisfaction problem which is NP-Complete. The algorithm needed to solve this type of problem is non-deterministic and can take a significant amount of time to solve for certain word and board combinations. For this reason, certain optimizations are employed such as attempting to determine if the word can be spelled using the single-letter tiles first and falling back to using the multi-letter tiles only if a solution cannot be found using the faster method and at least one of the multi-letter tiles exists in the candidate word. The length of the candidate word is also checked to ensure that it doesn't exceed the maximum word length of the available letter/dice pool before employing the multi-letter algorithm. Even with these optimizations, it is still possible to construct combinations of letter distributions and words that will take extraordinary amounts of time to solve although in practice this is rarely an issue (as indicated by the benchmarks below).

The below table shows how long it takes, in seconds, to check every word in several dictionaries against several game layouts.

| | TWL06 | SOWPODS | ENABLE1 |
|-------------------|----------------------|----------------------|----------------------|
| | 178,691 words | 267,651 words | 172,819 words |
| Boggle (Old) | 11.0 | 17.3 | 11.0 |
| Boggle (New) | 11.0 | 16.6 | 10.8 |
| Boggle Master | 13.8 | 20.8 | 13.5 |
| Big Boggle (2011) | 14.0 | 21.0 | 15.8 |
| Super Big Boggle | 17.1 | 25.7 | 17.1 |
| Scramble | 11.3 | 16.6 | 10.8 |
| Scrabble | 9.3 | 13.8 | 9.3 |

As an example of a difficult word/board combination, the ENABLE1 word list contains the word ETHYLENEDIAMINETETRAACETATE. It is quickly determined that this word can be spelled with the Super Big Boggle board without using the multi-letter die and that this word cannot be spelled with the 4x4 grid games. This word is 27 letters long which is one letter longer than the longest word that can be spelled using a Big Boggle (2011) board which contains 24 one-letter dice and one 2-letter die. Without employing this check, it takes hours to determine that this word cannot be spelled on a Big Boggle (2011) board. A similar situation can be created by adding a 26th die to the Big Boggle configuration.

8 GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.

`<http://fsf.org/>`

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “**Document**”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “**you**”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “**Modified Version**” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “**Secondary Section**” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “**Invariant Sections**” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “**Cover Texts**” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “**Transparent**” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise

Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “**Opaque**”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “**Title Page**” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “**publisher**” means any person or entity that distributes copies of the Document to the public.

A section “**Entitled XYZ**” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “**Acknowledgements**”, “**Dedications**”, “**Endorsements**”, or “**History**”.) To “**Preserve the Title**” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any

one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no

Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with ... Texts.” line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.