Primiview

COLLABORATORS			
	TITLE : Primiview		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		April 12, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Primiview

1.1 main

```
                    Primiview 3.1 ($VER: 39.15)

        Copyright 1993-1995 Jarkko Lindblad
            1995-1998 Arno Hollosi
```

Primiview stands for primitive viewer - it's purpose is to be a simple
SGF viewer/editor. If you want fancy graphics or a graphical user interface
with millions of buttons, this program is not for you.
Primiview is a freak program! It's powerful but has a simple interface, with
which you can work efficiently, if you get used to all the short-cuts.

ToDo-list
Note:

PLEASE mail me your suggestions, wish-lists, bug-reports, criticisms, ...

Especially I'm interested in BUGS. I don't discover all of these nasty
creatures myself, so I need your help to make Primiview more useable.

Send email to: <Arno Hollosi> hollosi@sbox.tu-graz.ac.at

1.2 about

What is it:

Primiview is a SGF file viewer/editor. SGF stands for 'Smart Game Format'
and is THE standard format for Go game records on the internet.
Primiview is NOT a computer Go player.

About the Authors

Arno Hollosi:

Email: hollosi@sbox.tu-graz.ac.at
This will change sometime during the 90's.

Snail mail address:
Arno Hollosi
Oberndorf 313
A-6322 Kirchbichl
AUSTRIA
Europe

I'm improving Primiview since V39.5, as Jarkko hasn't got the time
to continue his great work.
So if you've got any wishes, suggestions, bug-reports or anything
else, then email me.

My current account on IGS & NNGS is "dada"

I would like to thank the following people:

* Bob Carter for suggestions, beta-testing and for
proof-reading this text.

* Agi, Mike and the whole Amiga-Graz bunch for
moral support and being good friends.

Jarkko Lindblad:

Email: lindblad@kruuna.helsinki.fi
This will probably change sometime during the 90's.

Snail mail address:
Rälssintie 16 D 60
00720 Helsinki
Finland
Europe
This will also probably change sometime during the 90's.

1.3 disclaimer

Legal Mush

1.4 starting

Installing

Just drag the Primiview drawer to the desired place on your harddisk.
That's it. No assignments or environment-variables have to be set.

If you are short of diskspace you can omit every file but Primiview.
All that is needed is the program itself - 1 file. But I suggest that you
copy the manual too. Might be useful :o)

Starting Primiview

Primiview can be launched both from WB and CLI.

For CLI users: Primiview needs some stack-space. The maximum stack depends
on the longest tree in a SGF-file (not on the number of variations). Be sure
to set the stack size of the CLI you're using to a proper value. I suggest
you set the stack size to 20000 as I've never had problems with this size.
Once this is done you can enter at the CLI prompt: primiview [game-file]

WB users should just double-click the icon and the program starts.
The stack size is already preset to 20000 in the icon info.

Primiview searches for two files in the program directory:

"Primiview.prefs" - The preference file with all settings that
        may be customized. Select 'Save options' from the
        'Settings' menu to generate this file.

"Primiview.guide" - The Amigaguide documentation of Primiview
        (If Primiview can't find this file you don't have
        access to online help)

Programs Requirements

Primiview runs on Amiga computers with AmigaOS 3.0 (V39) or better.
There are no plans to modify the program to run on older versions of the OS.

1.5 menu_usage

                Menus:
======

    Project

  New   - Opens a new board window

        Open   -   Opens a file requester and loads the
                                game-file the user chooses into the
        current (last active) board-window

  Save   - Save game-file of the current board-window
  Save As   - Opens file requester, saves game-file

        About   -   Copyrights, PubScreen and ARexx port name

  Close   - Closes the current board-window (if the last
        board window is closed the program exits)
        Quit   -   Exit program

    Buffer

    (If you load more than one file into one window, they are stored in
     different buffers. Actions take place in the current [last active]
     board window.)

```
Clear       - Clears all buffers of the current window
List        - Not implemented

Next        - Switches to next buffer in window
Prev        - Switches to previous buffer in window
First       - Switches to first buffer in window

Game info     -   Lists information about the current game

Score mode      - Activate
                score mode
                 Show points     - Searches the comment text for points (e.g.
      'R16', 'C4', ..) and marks them on the board

Set file comment    - Sets the AmigaDOS file comment string
      (especially added for Bob :)

  Edit        - For a complete description of these items
  ----      have a look at the
               edit menu commands
              .


  Settings
  --------
      Screen Mode         -   The screen mode the program uses can be
                              changed with this menu item

      Text Font           -   Allows the user to change the font used on
                              comment, information and variation windows
      (Hint: try 'helvetica.font' height: 13)
      Board Font          -   Changes the font used in the board window
      Title Font          -   Changes the font used in the window and
                              screen titles
Edit Font     - Changes the font used for editing text

Variations
  Board position  - Variation-move lookahead;
        shows first move of each variation
  As Children - Show variations as children of current node
  As Siblings - Show variations as siblings of current
        node (XGoban style)

Quick gameinfo      - Print gameinfo into comment window and
      print result at last node of game

Snap1 <-> Snap2     - Primiview remembers two positions/sizes of
      comment-, variation- and board-window.
      Choosing this menu item chooses between
      the two snapshots of these windows.
      I use the two snapshots for:
      a) big board (e.g. pro-game, no comments)
      b) normal board (IGS/NNGS games with kibitz)

Take snapshot     - Snapshots the positions of all OPEN windows
                              (board windows: only the current [last
```

```
        active] board window)

        Save options      -   Saves all the information one can change
        from the settings menu. Also the current
        directory (the last one from which one
        loaded a file) and the file-pattern
        is saved.
```

1.6 key_usage

```
                Related:
                 Edit commands
                Keyboard Commands:
------------------

    All move-commands stepping down the tree (moving forward) choose
    variation 'A,' if there are any variations.

    Cursor right            -   Moves one node forward
    Cursor left             -   Moves one node backward
    Cursor up       -   Moves to start of current variation
    Cursor down       -   Moves to end of current variation or
                              to next variation start in the tree

    Space                   -   Moves one node forward
    Shift + Space     - Moves one node backward
    Alt + Space       - Same as Cursor down
    Shift + Alt + Space     - Same as Cursor up

    'a'-'z'                 - Chooses a variation

    Del         -   Switches to next variation-tree
    Shift + Del       - Switches to previous variation-tree
    Alt + Del       - Switches to variation tree 'A'
    Shift + Alt + Del     - Switches to root path (i.e. where you first
        took a variation instead of the 'A' path)

    Shift + CSR left      - Step ten nodes backward
    Shift + CSR right     - Step ten nodes forward

    Shift + CSR down      - Move to next comment
    Shift + CSR up      - Move to previous comment

    Ctrl + CSR up     - Move to root node of game
    Ctrl + CSR down     - Move to last node of current variation

    Shift + 'G'       - Goto node number #

    1,2,3...8,9,0     - Goto position 1-10
        (pressing again jumps to previous position)
    Shift + 1-0       - Current node is stored as position 1-10

    TAB         - Switch to next buffer (in the active window)
    Ctrl + TAB        - Step five buffers forward
```

```
    Shift + TAB       - Switch to previous buffer
    Shift + Ctrl + TAB      - Step five buffers backward
    Alt + TAB       - Switch to first buffer in active window

    Help                    -   Opens the AmigaGuide help file
    Shift + Help      - Opens the AmigaGuide help file (MAIN node)

    Ctrl + R        - Redraw Board (if GFX gets messed up)

    ESC         - Close all unnecessary windows
        (information window / requester window)


Mouse Commands:
---------------

    Left Mouse Button (LMB) - Moves one node forward (If 'show variation
        board positions' is ON, clicking on a
        variation mark chooses that variation)
    LMB + Shift       - Move one node backward

    LMB + ALT       - Moves to end of current variation or
                              to next variation start in the tree
    LMB + Shift + ALT     - Moves to start of current variation

    LMB + CTRL        - On empty intersection: go down tree and
        stop where this point gets occupied.
        On stone: go up tree and stop where this
        stone gets set.
```

1.7 score_mode

```
Score Mode
----------

The score mode is a powerful tool for scoring games.

It's based on the idea of influence: every stone influences its direct
neighbourhood (up to 3 points in each direction.) The algorithm sums up
all influences radiated and calculates the color of a given point.

Scoring can even be done during fuseki. But don't expect wonders :o)
It works best in the late middlegame and endgame.

You can interact and support this process by mouse.

Clicking the left mouse button (LMB) on

  an alive group -> marks this group as dead. Note that neighbour
        groups might change their status too. Click
        on these neighbour groups to achieve your goal.

  a dead group -> marks this group as alive.

  black or white territory -> remove the territory. The algorithm
```

```
      detects only trival cases of dame points.

  an empty point -> LMB: add black territory
        Shift + LMB: add white territory
        This can be useful to fill any holes - the
        algorithm automatically fills any completely
        surrounded area.

Clicking again on the same point or group undos your previous click.

The result is shown in the comment window. There are two values given for
komi - the left one is used for calculation, the right one (in parenthesis)
is the komi text as given in the file. If the values differ please go to
the root node and correct the komi value.

Press 'ESC' to leave the score mode.
```

1.8 edit_mode

```
                Since version 3.0 (39.11) Primiview is able to edit and save SGF- ↩
                   files.


Press <RETURN> (or <ENTER>) in order to enter or leave the edit mode.



                 Edit Menu

                 Key short cuts

                 Mouse commands

                 Editing text
                About SGF:
----------

                 General concepts

                 Restrictions

                 Style

                 Game info entries
                If edit mode is active, the edit menu will be enabled and a status ↩
                    line
displayed in the variation window.


Status line:  # Move #  Markup: filled square    * Add *
------------  ---------------------------------------------

# Move #  Is printed either in black or white and shows the
    color used for adding a new move
```

```
Markup: xxx Shows the currently active board markup

* Add *   Is printed either in black or white and shows the
    color used for adding stones to the board
```

1.9 sgf_concept

```
General Concepts:
-----------------

SGF is a text-only tree based file format.
It consists of nodes which are structured in a tree, i.e. a node has
exactly one predecessor called parent, but may have one OR MORE successors
called children. Thus SGF can store game records (a list of moves) and
variations of the actual line of play.

A node is the smallest unit visible to the user, i.e. the user steps through
the tree node-wise (forward [down the tree], backward [up the tree], etc.).

A node consists of properties. These properties contain a certain kind
of information, e.g. the 'B[]' property describes a black move made,
the 'C[]' property contains a comment text.

For example: if you step forward and see a new move on the board and a
comment in the comment window plus some markup on the board THEN all this
information is represented by different properties which are parts of the
SAME node.

Thus editing is done in two levels: adding/deleting nodes and adding
or deleting properties.

To make it clear: a move is part of a node and not the node part of the
move. A move is represented by ONE property but a node may contain more
than one property.


'Make a Move' vs. 'Place a Stone on the Board':
-----------------------------------------------

SGF provides two ways to add new stones to the board:

a) make a move   (SHIFT+LMB)
b) place a stone (CTRL +LMB)

Making a move is like making a move in a real game, i.e. you can only make
moves on empty intersections, you can only make one move per turn
(here: per node) and you may take some prisoners by making a move.

Primiview marks a move with a blue square.

Primiview doesn't check if a move is correct, i.e. you can make a move
which recaptures a ko.

Placing a stone on the board is like setting up a position, e.g.
```

handicap stones, setting up a problem or analysis of positions ("this
would work if the position over there would look like this...").
Thus one can place MORE than one stone, stones of different colors,
remove stones, replace stones with that of the opposite color
all in one node.
BUT: there are no prisoners made as these are not regular moves!

1.10 sgf_restrictions

Restrictions:

It is good style (and will be required in future versions of SGF) to
distinguish between a move and the position arrived at by this move.

Primiview therefor DOESN'T ALLOW mixing setup properties and move
properties within the same node.

Move properties are properties such as a black or white move or
annotations (bad move, interesting move, etc.).
Setup properties are properties used to set up or describe a position
such as place black/white stones on the board or annotations like
'black to play'.

Whenever Primiview denies you setting stones, making a move or adding
an annotation you're most likely to try to mix setup and move properties.

There are some SGF files available which contain nodes where mixing
these properties occurs. Primiview denies you to edit these nodes,
unless you delete either the setup or the move properties in these nodes.

1.11 sgf_style

Style:
------

* The first branch (variation 'A') is the main branch.
  Variation 'A' should always follow the real game.
  Consider yourself viewing a game where you have to press 'b' then 'c'
  and then 'b' again just to follow the real game - disturbing.

* Length of labels: Primiview is able to display labels of any size.
  The current file format allows only 4 chars - that's why Primiview
  doesn't allow the creation of labels longer than 4 chars.
  Attention: some applications have problems displaying labels longer
  than 2 chars. Use long labels with care!

* Omit extra pass plays and empty nodes at the end of the game.
  The last node of the game should contain the last move on the board.
  Do not put game information such as 'Black wins and connects Ko' into
  the comment field, rather add another move to the game which connects
  the Ko.

```
* Use annotations for standard situations (e.g bad move).
  Annotations are represented by extra SGF-properties and these
  property don't contain the text displayed by Primiview, but the meaning,
  i.e. a multilingual viewer could display these annotations in any
  language requested, or computer programs can use this information for
  their calculations.
```

1.12 edit_key

```
                Related:
                 Edit mouse commands

                 Edit menu commands

                 Navigation commands
                General Concept of Keyboard Shortcuts:
--------------------------------------

Qualifier Shift: node manipulations and MOVE properties
Qualifier Ctrl:  SETUP properties
Qualifier Alt:   MARKUP properties


Edit keyboard commands:
-----------------------

    Shift + 'A'     - Insert an empty new node as a child of the
      current node. If you start in an empty board
      window (created by the 'New' menu item) you have
      to add the root node first in order to edit e.g.
      gameinfo values!
    Shift + 'V'     - Insert an empty new node as variation
      (may be child or sibling depending on variation mode)
    Shift + 'N'     - Delete the current node.
      The root node can't be deleted!
    Shift + 'T'     - Delete the whole subtree starting with the current
      node. The root node can't be deleted!
    Shift + 'R'     - Reorder variations:
      step 1: press 'a'-'z' to select a variation
      step 2: press 'a'-'z' for variation AFTER which
        the selected variation should be inserted
      If an error occurs (or if you abort the operation
      by doing anything else than pressing these two keys)
      the display beeps.

    Shift + 'C'     - Enter
                    edit text
                    mode.
      You may edit the node-name and the comment text.
    Shift + 'I'     - Enter
                    edit text
                    mode.
    Shift + 'M'   You may edit almost all game info entries.
      Remember to move into the ROOT node (CTRL+CSR up)
```

```
      as game info properties are usually stored there.
      Primiview beeps the display if you're not in the root
      node. Some properties have a
                 required format
                .

    Shift + 'B'      - Set the 'move color' to BLACK, i.e. the next move
      inserted will be black
    Shift + 'W'      - Set the 'move color' to WHITE, i.e. the next move
      inserted will be white
    Shift + 'P'      - Add a pass move (current color).
      Adding is done like adding a normal move
      (see
                 mouse commands
                ).
    Shift + 'D'      - Delete all move properties
      (move annotations get deleted too)

    Ctrl + 'B'       - The color for adding stones is set to BLACK
    Ctrl + 'W'       - The color for adding stones is set to WHITE
    Ctrl + 'E'       - The color for adding stones is set to EMPTY
      (delete stones)
    Ctrl + 'D'       - Delete all setup properties
      (position annotations get deleted too)

    Alt + 'T'      - Markup: triangle - this should be used on stones only
    Alt + 'X'      - Markup: crosses  - this should be used on
             empty intersections only
    Alt + 'C'      - Markup: circle - this should be used on stones only
    Alt + 'S'      - Markup: square - this should be used on stones only
    Alt + 'F'      - Markup: filled square (SGF: selected points)
    Alt + 'A'      - Markup: labels
      A text requester pops up; this is the text inserted
      when you press ALT+LMB. Have a look at
                 style
                 for
      restrictions (currently 4 chars).
    Alt + 'L'      - Markup: lowercase letters from a-z
    Alt + 'N'      - Markup: numbers from 1-99
    Alt + 'B'      - Markup: black territory
    Alt + 'W'      - Markup: white territory
    Alt + 'D'      - Delete all markup properties of the current node
```

1.13 edit_mouse

```
                Related:
                 Edit key commands

                 Edit menu commands
                Edit Mouse Commands:
--------------------

    LMB        - Move one node forward or takes variation
      (just like in non-edit mode)
```

```
    Shift + LMB     - Make a move (the color is alternated)
    or      Clicking again on the same move deletes it
    CapsLock + LMB  If the current node doesn't contain a move
      the move is inserted.
      If the current node DOES contain a move a new child
      node (or variation!) is created (for easier input
      of game-records/variations.)
      Note that it is forbidden to make a move in the
      root node (as this is bad style.)

    Ctrl + LMB      - Place stones on the board (for setting up positions).
      Clicking again undos the placing.

    Alt + LMB     - Mark the current position with the chosen markup type.
      Clicking again removes the markup.

    Alt + Shift + LMB - Same as Alt+LMB, but uses a fill algorithm (only
      orthogonal connections - no diagonal connections).
      Using this function you can easily mark a whole
      group of stones or empty territory.
```

1.14 edit_menu

```
                Related:
                 Edit key commands

                 Edit mouse commands

                 Menu commands
                Edit Menu Commands:
-------------------

    Set goban size  - sets size of goban, allowed range: 1-19
      Attention: deletes all SGF data in buffer!
      Action takes place in the (last) active board window.
      Hint: you may use this function to clear a buffer.

    Node      - See
                keyboard commands
                    Color     - See
                keyboard commands
                    Add Stones      - See
                keyboard commands
                    Markup      - See
                keyboard commands
                    Annotation

    Annotations are used for making remarks on the current position or
    the move made (e.g. good or bad move). General annotations and move
    annotations may be emphasized.
    In this case Primiview displays '!!' after the text, e.g.
    'Good for Black!!' i.e. 'Very good for Black.'
```

```
    You can toggle emphasize by selecting the same menu item again.

  Delete        - Remove all annotations from the current node

  Good for Black      - General remark (something good for black)
  Good for White      - General remark (something good for white)
  Even position     - General remark
  Unclear position    - General remark

  Tesuji        - Move annotation (tesuji == locally good move)
  Bad move      - Move annotation
  Doubtful move     - Move annotation
  Interesting move    - Move annotation

  Black to play     - Position annotation
  White to play     - Position annotation
```

1.15 edit_text

```
Editing Text:
-------------

Editing text is done in the comment window using the edit font, which
is a fixed-width font that may be set via the "Edit font" menu item.
The text editor is a very basic one (no advanced features).

As it is possible to edit more than one value at a time, each value
(e.g. node name, comment text) is given a caption.

You can't edit the caption. If the cursor is over a caption its
color is set to blue. If the cursor is over a editable value its
color is set to red.

Functions:
----------

Cursor keys - To move around the cursor
Backspace - Delete the character to the left of the cursor
Del   - Delete the character under the cursor

CTRL + D  - Delete current value
CTRL + R  - Force a redraw of the text display

ESC   - Leave text edit mode and save changes    (OK)
Shift + ESC - Leave text edit mode and DISCARD changes (cancel)

If you select a menu item, activate a board window, resize a window
or anything similar Primiview exits text edit mode and saves the changes!

SGF defines that some values may hold multiple lines, others must not.
In the latter case Primiview ignores <Enter> and <Return>.

The text editor isn't able to scroll text.
If the text doesn't fit into the window you can't edit it!
```

```
If you delete lines the text that is outside the window is NOT scrolled
into the window. You have to force a redraw with CTRL+R.

The text buffer is currently 10 KBytes large.
Any text larger than this gets cut off!
```

1.16 edit_gameinfo

```
                Related:
                 Edit text mode
                Format of Game Info Entries:
----------------------------

Some game info values have a mandatory format, but Primiview doesn't check
the input data for correctness.
So it's up to you, to enter correct values. PLEASE TAKE CARE!


Black/white name: Name of the player who played black/white
      (no recommended format)

Black/white rank: Strength of the player who played black/white
      recommended format:
      "10k" or "10 kyu" for a kyu player
      "3d"  or "3 dan"  for a dan player
      Go servers may add a '*' or '?,' e.g. "10k*"

Black/white team: Name of the team (for games played in team events)
      (no recommended format)

Result:   Final result of the game
    MANDATORY FORMAT:
    "0" (zero) for a draw (jigo)
    "B+score" for a black win and
    "W+score" for a white win, e.g. "B+2.5" or "W+64"
    For half point results use "B+0.5" or "W+0.5"
    "B+R" or "B+Resign" and "W+R" or "W+Resign" for a win by
    resignation. You MUST NOT write "Black resigns"
    "B+T" or "B+Time" and "W+T" or "W+Time" for a win on time
    "B+F" or "B+Forfeit" and "W+F" or W+Forfeit" for a win by
    forfeit
    "?" for an unknown result

Komi:   Score adjustment (points added to White's score)
    MANDATORY FORMAT:
    Float value, e.g. "5.5" or "0" or "0.5" or "-10," etc.
    NOT: e.g. "5 points" <-- Error

Handicap: Number of handicap stones
    MANDATORY FORMAT:
    Integer value greater zero, e.g. "1" or "5" or "9"

Time:   Regular playing time for each side
    MANDATORY FORMAT:
```

```
    Time is given in seconds as a float value,
    e.g. "4600" or "300"
    NOT: "1 hour" <-- Error

Date:   Date when game was played
    MANDATORY FORMAT:
    Use the ISO-standard format "YYYY-MM-DD" because
    dates in this format can be sorted alphanumerically.
    Do not use other separators such as "/" or " " or ".".
    For games that last more than one day, following format has
    to be used:
    "YYYY-MM-DD,DD" if the month is the same
    "YYYY-MM-DD,MM-DD" if the month is different
    "YYYY-MM-DD,YYYY-MM-DD" if the year is different


For the following entries there's no recommended format:

Event:    Name of event (e.g. tournament name)
Round:    Number of tournament round
Place:    Name of place (e.g. city, country) where game took place
Rules:    Name of rule set used (e.g. Japanese, Chinese, AGA, GOE, etc.)

Game name:  Name of the game
Game ID:  ID for game (may become obsolete)
Opening:  Describes the opening played (e.g. san-ren-sei)
Game comment: General comment about the game

Source:   Name of the source (e.g. book, journal, etc.)
User:   Name of user (or program) who entered the game record
Annotation: Name of the person who made the annotations
Copyright:  Any copyright information
```

1.17 about_go

```
About Go
--------

Go (or WeiQi as it's called in China) is an ancient chinese board game.
It's played using black and white stones, which are set alternately on
the intersections of the board. The aim of the game is to surround as
much territory (empty intersections) as possible. Placed stones must
not be moved. Stones may get captured though.

For more information have a look at the FAQ posted on 'rec.games.go' or

have a look at one of the following:

* Mindy McAdams' Go page: http://www.well.com/user/mmcadams/gointro.html

* Jan van der Steen's pages: http://www.cwi.nl/~jansteen/go/go-0.html
  (thousands of stored professional games)

* Ken Warkentyne's Go Web Index: http://nngs.cosmic.org/~kenw/go/golinks.html
```

* The Go Teaching Ladder: http://www.iicm.edu/GTL

Go Definitions

Here's a list of Japanese Go terms and their explanation.

It was compiled by Fletch Holmquist and Bill Taylor.
(included by permission of Bill Taylor)

The original file can be found at: ftp://igs.nuri.net/Go/info/definitions.Z

Ten very common Go terms, for which there is no exact English word:

Atari	Dame	Go	Gote	Hane
Ko	Komi	Moyo	Seki	Sente

Ten more commonly used Go terms:

Dango	Fuseki	Joseki	Kakari	Miai
Ponnuki	Shimari	Tenuki	Tesuji	

Five common Go tournament terms:

Byo-yomi	Dan	Komi	Kyu	Nigiri

Aji (taste):
  Latent threats or possibilities existing in a situation.

Ajikeshi (aji erasure):
  A play which removes aji.

Amarigatachi:
  Play where one feels he has made good moves, when in fact he has
  accomplished little.

Ate, Atari:
  An immediate threat to capture; a single liberty remains.

Boshi (hat):
  A capping move.

Byo-yomi:
  Extra count-down time after regular clock time has elapsed.

Chuban[sen]:
  The middle game.

Dame (useless):
  A neutral point, territory for neither; a liberty.

Damezumari:
  Shortage of liberties.

Dan: Advanced grade.

Dango (dumpling shape):
A solid mass of stones; a very inefficient shape.

De (go between):
A move which pushes between two enemy stones.

Degiri: A sequence of two moves which push and cut.

Furikawari:
Exchange (of territories).

Fuseki: The opening moves of the game where influence and territory outlines are formed. (literally: 'no stones')

Geta (clog, as the shoe):
A method of capturing a enemy stone; a net trap.

Gote: Defensive play, loss of initiative. (Literally: 'lower hand')

Hamete: A trap.

Hane: A diagonal move played in contact with an enemy stone.
Hane-komi: Hane between two stones

Hanami ko ('flower-viewing ko'):
Ko where one player stands to lose a lot, but the other only a tiny amount.

Hasami (pincer play):
A play that attacks by preventing the opponent's extension down either side.

Hiki: Draw back.

Hiraki: 3rd or 4th line extension.

Honte: The proper move.

Horikomi (throw-in):
A single stone played as a sacrifice.

Hoshi: Star point, 4-4 point.

Igo: An alternative name for Go.

Ikken-tobi:
One point extension.

Insei: Student professional.

Ishi-no-shita:
Under the stones; a tesuji.

Ishi: Stone.

Jigo: Drawn game (by equal territory).

Joseki (established stones):
Known sequences of moves near the corner which result in near-equal positions for white and black.

Kakari (approach):
A move that attacks a single enemy corner stone.

Katatsuki (shoulder hit):
A play on a diagonal of the opponent's stone.

Kake: Press down.

Kaketsugi (hanging connection):
A open connection. An example is three stones surrounding an empty point. Promise for eye shape, but can be attacked.

Katachi:
The shape of the stones.

Sabaki: Quick development, light shape.

Keima: Knight's move extension.

Keshi: Erasure.

Kikashi:
A forcing forcing move, usually made outside the main flow of play. Often answered, then ignored; to be used later in the game.

Kiri: Cut.

Ko: Repetitive capture. (Literally: 'eternity')

Ko threat:
Intervening move (that one hopes will force a reply) before a ko can be recaptured.

Komi: Score adjustment usually penalizing black for playing first. Often 5.5 points.

Komoku: 3-4 point.

Korigatachi (frozen shape):
Inefficient or ugly shape.

Kosumi: A diagonal play next to one's own stone.

Kyu: Learner grade.

Magari (turn):
A play which turns a group, forming a corner.

Mane Go:

Mirror go. White playing symmetrically opposite black.

Miai: Two points which accomplish the same result; if deprived of one, the other must be played.

Mokuhazushi:
3-5 point.

Moyo: Large potential territory.

Nadare: Avalanche joseki.

Nakade: Unsettled eye shape.

Nidan bane (double hane):
Two sucessive hane plays by one player.

Nigiri: Equivalent of coin-toss to decide who starts. One grabs a handful of stones; the other guesses odd or even.

Nihon kiin:
Japanese go association.

Ni ren sei:
Fuseki with two adjacent star points.

Nobi (Stretch):
An extension away from an opponent's tsuke, cross-cut, etc.

Nozoki: A peeping move which threatens to cut.

Oba: Large fuseki point.

Ogeima (large knight's move):
Three across and one vertically (or vice versa).

Osae: A blocking move which prevents extension along a line.

Oyose: Large end-game plays.

Ozaru: Monkey jump.

Ponnuki:
Space between four stones after capture.

Sabaki: Light play; disposable stones.

Sagari: To descend straight toward the edge of the board.

San ren sei:
Fuseki with three adjacent star points.

San-san:
3-3 point.

Seki: A situation where neither player may place the other in ate without placing himself in ate. Stalemate, with no

territory awarded.

Semeai: Race to capture.

Sente: Threat forcing direct response, creates initiative.
The right to choose where to play next.
Opposite to gote. (Literally: ‘upper hand’)

Shibori:
Squeeze play.

Shicho: Ladder play.

Shicho-atari:
Ladder breaker. A stone played in the path of a potential shicho, threatening to make it fail.

Shimari (corner enclosure):
A two-stone corner formation. May not secure the corner, but attacker is at a disadvantage. Opposite of kakari.

Kogeima shimari (small knight’s enclosure):
The 3-4 and 5-3 points.
Ikken shimari (one-point enclosure):
The 3-4 and 5-4 points.
Ogeima shimari (large knight’s enclosure):
The 3-4 and 6-3 points.

Shin fuseki:
A revolutionary 1930’s strategy. Now blended with traditional strategy to form the modern style.

Shinogi:
Eye forming sequence; to rescue a group under attack.

Suji: Style; skillfulness.

Tachi: Extension adjacent to centre.

Taisha: A joseki arising from an ignored low kakari to 4-3 point.

Takamoku: 4-5 point.

Tengen: The centre point of the board.

Tenuki: Ignoring opponent’s last move to play elsewhere.

Tesuji: The best play in a local position; skillful tactical move.

Tewari diagram:
Analysing by removing irrelevant stones.

Tobi: Jump.

Tsugi: Connection.

Tsuke: Attatch. A play made in contact with an enemy stone,

```
        but not in contact with any friendly stones.
  Tsuke-atari:  Bang against (head-on).
  Tsuke-nobi:   Attatch and extend (handicap joseki).

Tsume:  Extension preventing an enemy extension.

Tsume-go:
  Life and death problems.

Uchikomi:
  Playing to invade enemy territory.

Warikomi:
  Wedge between two stones.

Wariuchi:
  A wedging move which has room for expansion in either direction.

Watari: To connect underneath.

Wei Chi:
  The Chinese name for Go. (Literally: 'game of encirclement')

Yose:   End game.

Yose-ko:
  A ko of little value.

Yosu miru:
        Probe; to see opponent's response. May be sacrificed.
```

1.18 arexx

```
                AREXX
-----

Since V39.15 Primiview has an AREXX interface. Now you can use Primiview
in combination with a database program to manage your SGF files.

The documentation isn't as good as it should be. When I've more time
I'll improve it.


                 Configuring the ARexx menu

                 Supplied scripts

                 The supplied database file
                Command interface
-----------------

  Only a minimal core right now. Just barely enough to do what
  it was designed for in the first place: work together with a
  database program. Other commands will follow in later versions.
```

TYPE AREXX means that this is a unique arexx command

TYPE INTERFACE means that this command is exactly the same
as the user-interface function. The action taken is
the same as if the user had selected this function
manually.

Every ARexx command returns '0' in case of success and a non-zero number in case of an error.
'RESULT -' therefore means that no extra data is returned. Just the return code (in variable RC) for succesfull completion or error.

GETSCREENNAME
- get public screen name

LISTWINDOWS
- get window id, status and name

LISTBUFFERS
- get buffer status and names

OPEN
- load a file into current window

CLOSE
- close current window

GOTONODE
- goto specified node

GETNODE
- get node- and move-number; prisoners

GETPROPVALUE
- get property values

LISTCHILDREN
- list children of current node

GETBOARD
- get ASCII board

SAVEOPTIONS
- save current options

SNAPSHOT
- take snapshot

SWAPSNAPSHOT
- swap snapshots

1.19 rx_menu

```
Configuring the ARexx menu
--------------------------

- select 'Save options' from the menu to get the latest version of
  'Primiview.prefs'.

- start a text editor and load 'Primiview.prefs'.
  The first line should be: >V: 3901501< (or a higher number)

- goto line 53 and you'll see ten lines with '-x-', where x is a number,
  separated by newlines.

- the '-x-' lines are the text that is displayed in the ARexx menu.
  Change these lines accordingly.

- every line immidiatly following an '-x-' line contains the name of
  the ARexx script which should be executed. Specify the FULL path!

- A preference file might look like this:
  -1-

  -2-

  -3-

  -4-

  -5-

  Add Record
  Go:arexx/add_record.arexx
  -7-

  -8-

  Start DB (GTL)
  Go:arexx/start_db.arexx Go:Go_Teaching_Ladder/DB
  Start DB (pro)
  Go:arexx/start_db.arexx Go:pro/DB

- don't mess with other lines or add new lines!

- save the file and quit and restart Primiview to load the new preferences.
```

1.20 rx_scripts

```
Supplied scripts:
-----------------

The scripts supplied with Primiview are to be used in combination with
the database program called DB. DB was written by David Ekholm and Marcin
Orlowski. You can find the program on AMINET. Download version 3.5 or later.
```

```
start_db.arexx
  This script is used to start DB on Primiview's public screen.
  Correct the path of DB in the script to suit your needs.
  This script should be added to your Primiview Arexx menu.

add_record.arexx
  This script adds a new record to the currently open database.
  If you've multiple copies of DB running, the record is added
  to the first DB with ARexx port 'DB.1'.
  Note that the script works only with the supplied database
  file or copies thereof - see database file below.
  This script should be added to your Primiview Arexx menu.

open.arexx
  This script is called by DB, if you double-click on the file
  gadget. It loads the file specified in the currently active
  window of Primiview.
  If you've multiple copies of Primiview running, the file is
  loaded into the first Primiview with ARexx port 'RX_Primiview'

index.arexx
  This script is used to add a whole directory (including all
  sub-directories) to the database. Note that you've to have both
  Primiview and DB running when you execute this script. You should
  have loaded the supplied database file or a copy thereof into DB
  before you start the script.
  Open a shell and type 'rx index.arexx DIRECTORY' where DIRECTORY
  is the !complete! path of the directory you want to index.
  Wait and see.
  Hints:  - to get rid of non-SGF files sort the database by filenames
      and delete all empty records at the beginning. You can
      search for files not ending with '.sgf' or '.mgt' too.
    - minimize the goban window and the comment window of
      Primiview so that updating the windows after loading a
      file doesn't take too long.
  Speed: I indexed my collection of ~2000 professional games in about
         20 minutes. I know that this isn't fast, but I hadn't time
         to add code to suppress Primiview's output.
```

1.21 rx_db

```
The supplied database file 'DB':
--------------------------------

The database file supplied with Primiview is to be used in combination with
the database program called DB. DB was written by David Ekholm and Marcin
Orlowski. You can find the program on AMINET. Download version 3.5 or later.

'DB' is an empty database file. It supplies two views for the data: a
standard one and a compact one. When you double-click on the filename gadget
the script 'open.arexx' is called and the file is loaded into Primiview.

Make copies of this file for every new database you want to create.
Don't change the order of items in the first and second line or the scripts
```

won’t work anymore.

You’ve to change the path to ‘open.arexx’ though. Load the file into a
text editor which !preserves! TAB characters.
Go to the end of the second line. Change the RXFILE entry according to
your installation.

You can type your own comments into the ‘Comment’ field and search for them
afterwards. If this isn’t enough, you can add new items - refer to the RFF
file format description in the documentation of DB.
Attention: only add new items at the tail of the current item list.
     Otherwise the scripts won’t work correctly.

You may change the views with the view editor of DB.
No limitations here :o)

1.22 rx_close

```
 NAME
CLOSE

 TYPE
INTERFACE

 SYNOPSIS
CLOSE FORCE/S,QUIET/S

If QUIET is not specified, a save-requester opens up if a file has
been changed, but not saved yet. If QUIET is specified then no
requester opens and in case of an unsaved file, the function fails
(RC is set to 5.)
FORCE should only be used in combination with QUIET and closes the
current window, no matter if the files have been saved or not.

 RESULTS
-

RC is set to 5, if QUIET was specified (but not FORCE) and there
were unsaved files. See LISTBUFFERS for status of buffers in the
current window.

 EXAMPLE

options results
address 'RX_Primiview'

close quiet
if rc=5 then say "there are unsaved files";
    else say "window closed";
```

1.23 rx_getboard

```
 NAME
GETBOARD

 TYPE
AREXX

 SYNOPSIS
GETBOARD

 RESULTS
BOARD/M

Returns an ASCII version of the current board position.

'.' ... empty point
'+' ... empty star point (hoshi)
'O' ... white stone
'#' ... black stone

 EXAMPLE

options results
address 'RX_Primiview'

getboard stem b.
do i=0 to b.board.count-1
    say b.board.i
    end i

 EXAMPLE OUTPUT

    A B C D E F G H J K L M N O P Q R S T
19 |. . . . . . . . . . . . . . . . . . .| 19
18 |. . . . . . . . . . . . . . . . . . .| 18
17 |. . . O . . O . . . . . . . . . . . .| 17
16 |. . . + . . . . . + . . . O . # . . .| 16
15 |. . # . . # . . . . . . . . . . . . .| 15
14 |. . . . . . . . . . . . . . . # . . .| 14
13 |. . . . . . . . . . . . . . . . . . .| 13
12 |. . O . . . . . . . . . . . . . . . .| 12
11 |. . . . . . . . . . . . . . . . . . .| 11
10 |. . . + . . . . . + . . . . . # . . .| 10
 9 |. . . . . . . . . . . . . . . . . . .|  9
 8 |. . . . . . . . . . . . . . . . . . .|  8
 7 |. . . . . . . . . . . . . . . . . . .|  7
 6 |. . . . . . . . . . . . . . . . . . .|  6
 5 |. . . . . . . . . . . . . # . . . . .|  5
 4 |. . . O . . . . . # . . . . # # # . .|  4
 3 |. . . . . . . . . . . . . . O O O O .|  3
 2 |. . . . . . . . . . . . . O . . . . .|  2
 1 |. . . . . . . . . . . . . . . . . . .|  1
    A B C D E F G H J K L M N O P Q R S T
```

1.24 rx_getnode

```
 NAME
GETNODE

 TYPE
AREXX

 SYNOPSIS
GETNODE

 RESULTS
NUMBER/N,MOVE/N,BPRISONER/N,WPRISONER/N

number ... node number (starts with 0)
     If the buffer is empty, number is set to -1
move ... move number (starts with 1)
   If the buffer is empty, move is set to -1
bprisoner ... number of captured black stones
wprisoner ... number of captured white stones

 EXAMPLE

options results
address 'RX_Primiview'

getnode stem b.
say "Node #" b.number
say "Move #" b.move
say "B prisoners:" b.bprisoner
say "W prisoners:" b.wprisoner

 EXAMPLE OUTPUT

Node # 152
Move # 152
B prisoners: 3
W prisoners: 8
```

1.25 rx_getpropvalue

```
 NAME
GETPROPVALUE

 TYPE
AREXX

 SYNOPSIS
GETPROPVALUE PROPERTY/A

 RESULTS
VALUE/M

 EXAMPLE
```

```
options results
address 'RX_Primiview'

gotonode root
getpropvalue stem b. PW
if rc=0 then say "Player white:" b.value.0;
getpropvalue stem b. WR
if rc=0 then say "White rank:" b.value.0;
getpropvalue stem b. PB
if rc=0 then say "Player black:" b.value.0;
getpropvalue stem b. BR
if rc=0 then say "Black rank:" b.value.0;
say ""
getpropvalue stem b. KM
if rc=0 then say "Komi:" b.value.0;
getpropvalue stem b. HA
if rc=0 then say "Handicap:" b.value.0;
getpropvalue stem b. RE
if rc=0 then say "Result:" b.value.0;
say ""
getpropvalue stem b. DT
if rc=0 then say "Date:" b.value.0;
getpropvalue stem b. PC
if rc=0 then say "Place:" b.value.0;
getpropvalue stem b. EV
if rc=0 then say "Event:" b.value.0;
getpropvalue stem b. RO
if rc=0 then say "Round:" b.value.0;
getpropvalue stem b. GN
if rc=0 then say "Game name:" b.value.0;

 EXAMPLE OUTPUT

Player white: Go Seigen
White rank: 9 dan
Player black: Kitani Minoru
Black rank: 9 dan

Komi: 0
Result: W+Resign

Date: April 14-15, 1957
Place: Tokyo, Japan
Event: Go Seigen vs Kitani Minoru
Game name: 1st Japan's Strongest Deciding Matches
```

1.26 rx_getscreenname

```
 NAME
GETSCREENNAME

 TYPE
AREXX
```

```
 SYNOPSIS
GETSCREENNAME

 RESULTS
PUBSCRNAME

Result is a simple string, which contains the public screen name
of Primiview. This can be used to open other applications windows
on Primiview's screen.

 EXAMPLE

options results
address 'RX_Primiview'

getscreenname
say "Screenname" result

 EXAMPLE OUTPUT

Screenname Primiview
```

1.27 rx_gotonode

```
 NAME
GOTONODE

 TYPE
INTERFACE

 SYNOPSIS
GOTONODE ACTION/A

ACTION can be one of  xxx  any integer; goto node number xxx
          'DOWN'  move down tree
          'DOWN_VAR'  move down to start of next var
          'DOWN_COM'  move down tree to next comment
          'UP'  move up tree
          'UP_VAR'  move up until start of variation
          'UP_COM'  move up tree to previous comment
          'A' - 'Z'   select variation (may move down)
          'VAR_NEXT'  move to next variation (sibling)
          'VAR_PREV'  move to previous var (sibling)
          'VAR_A'   move to variation a
          'VAR_ROOT'  move to root path
          'ROOT'  move to root node
          'END'   move to end of tree

 RESULTS
OLDNODE/N

RC is set to 5 if the action failed. E.g. specified node or
variation doesn't exist.

 EXAMPLE
```

```
options results
address 'RX_Primiview'

gotonode DOWN_VAR
gotonode B
```

1.28 rx_listbuffers

```
 NAME
LISTBUFFERS

 TYPE
AREXX

 SYNOPSIS
LISTBUFFERS

 RESULTS
BUFFERLIST/M

Bufferlist is given as  "ccxrr ca nnnnnnnnnnnnnnnnnnnnnn" with

  'ccxrr' gobansize, 'cc' is number of columns, 'rr' is
    number of rows.

  'c'    '-': buffer has not been changed
    '*': buffer has been changed and not saved yet

  'a'    'a': active buffer
    'i'; inactive buffer

  'nnnnnnnnnn'  filename

 EXAMPLE

options results
address 'RX_Primiview'

listwindows stem w.
do i=0 to w.windowlist.count-1
    say w.windowlist.i
    if substr(w.windowlist.i, 7, 1) ~= "-" then do
  id = Left(w.windowlist.i, 4);
  listbuffers stem b. id
  if rc = 5 then
      say "goban window" id "not found"
  else
      do j=0 to b.bufferlist.count-1
    say b.bufferlist.j
    end j
  say ""
  end
    end i
```

```
 EXAMPLE OUTPUT

0000 V- Node-info & variations
0001 C- Comment Window
0100 Bi 1: ram:game1.sgf
19x19 -a ram:game1.sgf
19x19 *i ram:girodias1.sgf

0102 Bi Go:Pro/misc/9x9/game1
09x09 -a Go:Pro/misc/9x9/game1

0103 Ba 3: titles/meijin/1997/game-3.sgf
19x19 -i Go:Pro/japan/titles/meijin/1997/game-1.sgf
19x19 -i Go:Pro/japan/titles/meijin/1997/game-2.sgf
19x19 -a Go:Pro/japan/titles/meijin/1997/game-3.sgf
19x19 -i Go:Pro/japan/titles/meijin/1997/game-4.sgf
19x19 -i Go:Pro/japan/titles/meijin/1997/game-5.sgf
19x19 -i Go:Pro/japan/titles/meijin/1997/game-6.sgf
```

1.29 rx_listchildren

```
 NAME
LISTCHILDREN

 TYPE
AREXX

 SYNOPSIS
LISTCHILDREN

 RESULTS
NODELIST/N/M

RC is set to 5 if the current node has no children.
RC is set to 7 if the buffer is empty

 EXAMPLE

options results
address 'RX_Primiview'

say "CHILDREN:"
listchildren stem b.
if rc<5 then
    do i=0 to b.nodelist.count-1
  say b.nodelist.i
  end i
else
    if rc=5 then say "no children"
    else say "empty buffer"

 EXAMPLE OUTPUT

CHILDREN:
101
```

```
460
465
468
```

1.30 rx_listwindows

```
 NAME
LISTWINDOWS

 TYPE
AREXX

 SYNOPSIS
LISTWINDOWS

 RESULTS
WINDOWLIST/M

Windowlist is given as  "xxxx yz nnnnnnnnnnnnnnnnnnnnnn" with

  "xxxx" window number (used to identify window)

  "y"  window type: V ... variation window
      C ... comment window
      I ... gameinfo window
      B ... board window
      R ... arexx "window"

  "z"  window status: a ... activated
      l ... locked and activated
      i ... inactive
      - ... none of above applies (variation-,
           comment- and gameinfo-window)
  "nnnn" window name

 EXAMPLE

options results
address 'RX_Primiview'

listwindows stem b.
do i=0 to b.windowlist.count-1
    say b.windowlist.i
    end i

 EXAMPLE OUTPUT

0000 V- Node-info & variations
0001 C- Comment Window
0002 I- Information on the game
0100 Bi masters/goseigen/2/014_047.mgt
0101 Ba Pro/masters/shusai/game-6.SGF
```

1.31 rx_open

```
 NAME
OPEN

 TYPE
INTERFACE

 SYNOPSIS
OPEN FILENAME/A,FORCE/S,QUIET/S

 RESULTS
-

RC is set to 5 if there were unsaved files (and open not executed)
RC is set to 7 if there was an error during loading (note that
    in this case all buffers in the winow have been
    deleted; the window contains an empty buffer)

 EXAMPLE

options results
address 'RX_Primiview'

open force go:pro/japan/titles/meijin/1987/game-2.sgf
```

1.32 rx_saveoptions

```
 NAME
SAVEOPTIONS

 TYPE
INTERFACE

 SYNOPSIS
SAVEOPTIONS QUIET/S

If QUIET is not specified, a requester pops up in case of an error
during writing the preference file.

 RESULTS
-

 EXAMPLE

options results
address 'RX_Primiview'

saveoptions
```

1.33 rx_snapshot

```
 NAME
SNAPSHOT

 TYPE
INTERFACE

 SYNOPSIS
SNAPSHOT

 RESULTS
-

 EXAMPLE

options results
address 'RX_Primiview'

snapshot
```

1.34 rx_swapsnapshot

```
 NAME
SWAPSNAPSHOT

 TYPE
INTERFACE

 SYNOPSIS
SWAPSNAPSHOT

 RESULTS
-

 EXAMPLE

options results
address 'RX_Primiview'

swapsnapshot
```

1.35 history

```
                History
-------


V39.15 (Jan 17th 1998)

  - NEW: AREXX interface and scripts for interaction with DB
  - NEW: Score mode
  - NEW: quick gameinfo (shown in comment window)
```

```
  - Primiview searches the program directory (instead of the
    current directory) for '.prefs' and '.guide' now.
  - NEW: 'Show Points' - search the comment for points and
    mark them on the board
  - starting editing is easier now: if there is no node in the
    buffer, adding a stone by clicking adds the necessary (two)
    nodes automatically
  - adding moves to root nodes forbidden now (this is bad style)
  - fixed some bugs (check for OS, CTRL+D)


V39.13 (Mar 1st 1997)

  - Edit / Save functionality added (finally!!)
  - Online help (AmigaGuide)
  - Loading multiple files into a window possible
    (a window may contain more than one SGF buffer)
  - New variation style: variations shown as siblings (XGoban style)
  - New mouse navigation functions (see
                Commands
                )
  - New navigation command: "goto root path"
  - Some functions have a new
                 key short-cut
                  - Capture of suicide moves added
  - Removed "Strip Whitespace" menu item: it is done automatically now
  - And bug fixes (as always :)


V39.10 (July 6th 1996)

  - Primiview is able to keep several SGF files open simultaneously.
    (New menu-items: New / Close)
  - Board windows are resizable now!! (as in version 0.1 :-)
  - Added support for different goban sizes (9x9, 13x13, etc.)
  - Final reorganization of internal structure: properties of a node
    are stored as a list instead of a single string.
  - Many new properties recognized. The most important are:
    WL/BL (w/b time left), TW/TB (w/b territory), LB (board label)
  - New commands:
    Shift + Help = Previous variation
    Alt + Help   = Goto variation 'A'
    Shift + Del  = Goto end of current variation
    F-keys :  Pressing twice jumps back to previous position
        (for easier comparison of two positions)
    (Shift+)Alt+Space = Same as CSR up / CSR down)
    'Strip whitespace' - See 'usage' for more info
    'Snap1 <-> Snap2' - See 'usage' for more info
  - Primiview is no longer compiled with STACKEXTEND-option
    as this SAS-routine is buggy. Be sure to increase the stack
    to a proper size (e.g. 20000) if started from CLI.
  - Bug fixes, minor enhancements, etc.


V39.8 (Mar 31st 1996)

  - Completely recoded SGF-parsing
```

```
    Now Primiview builds up a real TREE-structure!
  - New commands: CSR down, Shift+Space, LMB, next-variation (Help),
           redraw (<Return>), bookmarks (F-keys),
      goto ('G'), next/prev-comment (Shift+CSR down/up),
      10 nodes forward/backward (Shift + CSR right/left),
      close unnecessary windows (ESC)
  - Placing stones on board added (Shift+LMB)
  - Output in comment & info-window is word-wrapped now
  - Info-window displays more info on a game-file
  - Variation window is kept open and contains:
    move-description (number, place), number of black and
    white prisoners and the node number
  - Again: several bug-fixes, some of them were really tough
         to hunt down - but I got'em ;-)
  - And of course a lot of other minor enhancements, too many to
    list all of them :)


V39.6 (Dec 2nd 1995)

  - Board-window title shows filename
  - Junk in front of the first "( ;" is skipped
    (no need to remove mail-headers any longer)
  - Variation handling fixed; is now 100% ok (hopefully ;-)
  - CRSR UP command implemented
  - Nodenames in variation window shown
  - Variation-move look ahead
    (shows first move of each variation)
  - Documentation now in GUIDE-format
  - Tons of bug fixes; compiled with STRICT option; made Primiview
    a lot safer; Primiview runs without Enforcer-hits (at least
    on my machine)


V39.4 (May 30th 1995)

  - Made variation window size itself automatically
  - Fixed the hopefully last enforcer hits and the screen
    scrolling bug. Primiview _should_ no longer guru and now has
    at least one goto-instruction in source. :-)
  - Fixed the variations-bug. Now variations _should_ work all
    the time.


V39.0 (May 11th 1995)

  - Primiview TNG aka Primiview 2.0 alpha was released after a
    couple days of hard work. :-)
          97% of the code was written from scratch - I only copied and
    modified a couple routines from the old Primiview.


V36.0 (Late 1993-early 1994)

  - The first generation primiview was being coded and released.
    God it was awfully primitive.
```

1.36 limitations

```
Limitations
-----------

 - If a node has lots of comments some of them won't fit into the comment
   window and won't be shown. Usually this is no problem, but if you
   review a game with lots of kibitzes some of them may not show up.
   I have NO plans to include any kind of feature to go around this.
   Primiview is primitive. You get what you pay for.
   If this is a problem use larger screen and smaller font. :-)

 - Don't really know what Primiview does if it gets low on memory.
   Better not to find out. :o)

 - Primiview expects a valid, syntactically correct SGF-file.
   Some errors in a SGF-file don't disturb Primiview, others do.
   It may even be possible that an incorrect SGF-file crashes Primiview!


Known Bugs
----------

Currently I'm not aware of any problems using Primiview.
If you discover a bug please don't hesitate to contact me at:

Arno Hollosi <hollosi@sbox.tu-graz.ac.at>

Please don't forget to mention the version of Primiview you're using.
If possible supply the SGF-file and indicate where the error occurs.
Thanks!
```

1.37 todo

```
ToDo
----

o Support of FF[4] (the latest SGF standard)

o An improved (error-tolerant) SGF parser

o More sophisticated edit functions

o enhance AREXX support

o Improve the look of Primiview (nice goban & stone graphics)

o Database functions (joseki / fuseki / pattern)
```