

# **TicTacTic User's Manual**

---

For version 1.2.1, 1 November 2000

**Gianluca Moro**

---

Copyright © 2000 Gianluca Moro

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions, except that this permission notice may be stated in a translation approved by the Free Software Foundation.

# 1 Introduction

This manual documents version 1.2.1 of the *TicTacTic* game.

The game is a variant of the classical Tris or TicTacToe game implemented using the Caml language and the Tk graphical toolkit.

One of the aim of the author was to test the flexibility of this developing environment (and this test has been succesfully accomplished), the other was to find a way to spend a week closed at home for an accident at the foot!

And, last but not the least, give a simple but challenging game to the Linux community. I have not seen an implementation of this game for Linux: if I am wrong, let me know at `giammy@artemide.dei.unipd.it`



## 2 Game

### 2.1 Main idea of the game

The game is a variant of the standard Tris or TicTacToe game, with the addition of the concept of “Aging cells”, that is in a board 3x3 you can have at most 3 cells occupied: when placing another cell, the oldest one is automatically freed.

In this way, while playing, you must be carefull both at the positions and at the sequence which brought to that position.

This allow to eliminate the typical draw situation of the TicTacToe standard game, and obtaining a something different game: at first its much more interesting, at least until you find out how to defeat the computer: if you are the first to play, you always can do it (in a 3x3 board: I did not check bigger ones).

The idea was taken by the author from a gamepad he saw somewhere.

### 2.2 The rules

The game is played on a 3x3 board: each player occupies a free cell, continuing to do so until one of the player has 3 cells aligned both horizontally, vertically or diagonally.

Each player can occupy at most 3 cells: whenever a fourth cell is occupied, the oldest one is freed.

The White player always starts the game.

That’s all folks: it’s simpler to play than to read the rules, but it’s not so simple to keep up with the computer!

One of the main tricks, is to occupy a cell which gives 2 pieces (our 2 younger ones) aligned with the older enemy piece: in this way, the next move, whichever cell the enemy occupies, he frees his oldest one and let us to align our 3 cells.

### 2.3 The heuristic

The move evaluation is accomplished recursively, applying the alpha-beta pruning algorithm.

The main idea is to evaluate a move such as its value is:

- (Maximum - recursion level) if that move M is winning
- Medium if the maximum recursion depth has been reached
- the value of the best move that the enemy can do after my move M, changed of sign.

Each analysis branch is abandoned whenever the condition of apha-beta pruning are satisfied. For more information see: *D. Knuth, An Analysis of Alpha-Beta pruning, Artificial Intelligence 6(1975) 293-326.*

This algorithm is implemented in the file `game.ml`: modifying it, or adding other algorithms should involve the modification of this file only, in particular the function `search`.

## 2.4 The code

The specific game code is kept in the `ttt` directory, while the support library functions are kept in the `bmlib` directory. The functions in this directory are a stripped down version of a major and more flexible library developed for another project. By now I cannot publish the original library under GPL (it's office work...), so I rewritten the parts I needed for my game, keeping the same function calling interface.

The bigger part of the original library missing here is the Thcheme interpreter, substituted by a much more simpler, but less flexible configuration file interpreter, (`configuration.ml`).

Anyway the main ideas (most of them coming from Matteo Frigo) are there, and rewriting the code helped me to understand various passages I did miss in current library usage! I hope that the whole library will be published in the near future.

## 3 Menu

This chapter describe the menu items present in the game: the “Quit” and “Help” entry are not explained, if you do not understand them, don’t play this game: it is not for you.

As a matter of fact it never happened to me to read a manual of a game to play it: especially for a game of this type. If you have been so patient to read this manual, my compliments!

Anyway the game is intended to be self-explanatory: if not, I should have projected it in a better way.

### 3.1 New Game

To start a new game: this entry clears the board and starts a new game: all the preferences menu choices will be active in the game which is starting. You can also use this entry to abort the current game, and start again.

### 3.2 Players

You can select which player is managed by the computer, White or Black: the difference is that the White always starts the game. You can also play with a friend of your: you can note that this board game is quite difficult to play without a computer for the cumbersome task of remembering which is the cell to free for each move.

Note that cell colors can be changed: in this manual I use the term “White” to indicate the player who starts the game.

### 3.3 Levels

The game presents a series of level, named with some fancy names: if you like them, lucky you. For the most curious, the difficulty level just says how many levels of look ahead are performed, starting from 0 for the lower level. Anyway, the source is there ...

In my computer (Pentium II 400) the maximum computer thinking time at the highest level (14 levels deep recursion) is 3 seconds on a 3x3 board (see [Section 2.3 \[The heuristic\]](#), [page 3](#)).

### 3.4 Language

Select the language used by the game, just English or Italiano: note that everything changes on the fly when you change it. (By the way, any volunteer to translate the manual to italian?)

### 3.5 Tear-off menus

Menu tear off: enable the menu detaching from the menu bar to keep it at hand. To actually detach the menu, just click over the upper part of the menu, where you see the dashed line, and you'll have a floating menu window.

### 3.6 Save Preferences

This entry is to save the current preferences: the user choices are kept in a configuration file, named `~/.tttrc`, which, if present, is loaded at each program start. It's a plain ASCII configuration file, whose values are assigned with a Scheme-like syntax. It should be self-explanatory, if not, it's a good reason to start studying Scheme.



## 4 Installation and Customization

This chapter describes the installation and customization of *TicTacTic*, the latest version of which may be downloaded from SourceForge, which hosts the official TicTacTic home page (<http://tictactic.sourceforge.net>), or from the author's home page (<http://www.giammy.com>)

The distributed program has been compiled and tested in a linux box, running RedHat 6.0, using the `ocaml` compiler, version 2.02, and the Tk toolkit 8.0.

The program has been written using the Caml language.

In the package there is a precompiled executable named `ttt`.

### 4.1 Installation on Unix

To compile the package you need the Caml environment: if it is installed, the package comes with the standard GNU configuration tools: just type `./configure` and `make`.

You will get the executable called `ttt.opt` in the `ttt` directory: just copy it in a directory in the PATH.

If you do not want to compile it, just copy the precompiled executable named `ttt`, in a directory present in the PATH.

### 4.2 Installation on non-Unix Systems

The *TicTacTic* game can be compiled for Windows boxes, using the Cygnus GCC and OCAML cross compiler for Linux. This is the procedure I have followed to compile for Windows: as a matter of fact the developing environment (gcc and caml compilers) can be found native for Windows too, but I never tested them.

### 4.3 Customizing the game

For the customization, you just configure the game as you prefer using the Preferences menu, and then use the "Save Preferences" menu entry. That's all.

By the way, if you change manually the configuration file `~/.tttrc` you can access some feature not actually configurable from the menu, such as the board dimension: just change the line (`define-property 'board-size 3`). The game, at least for my experience, is not very interesting in the 4x4 board, and for bigger boards, it is not playable.

If you get in trouble doing this, just remove the configuration file and start again.

You can also change the cell colors, the main supported colors are: `gray60 gray70 gray80 gray85 gray90 gray95 AntiqueWhite1 AntiqueWhite2 AntiqueWhite3 AntiqueWhite4 bisque1 bisque2 bisque3 bisque4 PeachPuff1 PeachPuff2 PeachPuff3 PeachPuff4`

NavajoWhite1 NavajoWhite2 NavajoWhite3 NavajoWhite4 LemonChiffon1 LemonChiffon2  
 LemonChiffon3 LemonChiffon4 cornsilk1 cornsilk2 cornsilk3 cornsilk4 ivory1 ivory2  
 ivory3 ivory4 LavenderBlush1 LavenderBlush2 LavenderBlush3 LavenderBlush4 MistyRose1  
 MistyRose2 MistyRose3 MistyRose4 azure1 azure2 azure3 azure4 SlateBlue1 SlateBlue2  
 SlateBlue3 SlateBlue4 RoyalBlue1 RoyalBlue2 RoyalBlue3 RoyalBlue4 blue1 blue2  
 blue3 blue4 DodgerBlue1 DodgerBlue2 DodgerBlue3 DodgerBlue4 DeepSkyBlue1 DeepSkyBlue2  
 DeepSkyBlue3 DeepSkyBlue4 SkyBlue1 SkyBlue2 SkyBlue3 SkyBlue4 LightSkyBlue1  
 LightSkyBlue2 LightSkyBlue3 LightSkyBlue4 SlateGray1 SlateGray2 SlateGray3  
 SlateGray4 LightBlue1 LightBlue2 LightBlue3 LightBlue4 LightCyan1 LightCyan2 LightCyan3  
 LightCyan4 PaleTurquoise1 PaleTurquoise2 PaleTurquoise3 PaleTurquoise4 turquoise1  
 turquoise2 turquoise3 turquoise4 cyan1 cyan2 cyan3 cyan4 DarkSlateGray1 DarkSlateGray2  
 DarkSlateGray3 DarkSlateGray4 aquamarine1 aquamarine2 aquamarine3 aquamarine4  
 DarkSeaGreen1 DarkSeaGreen2 DarkSeaGreen3 DarkSeaGreen4 SeaGreen1 SeaGreen2  
 SeaGreen3 SeaGreen4 PaleGreen1 PaleGreen2 PaleGreen3 PaleGreen4 green1 green2  
 green3 green4 OliveDrab1 OliveDrab2 OliveDrab3 OliveDrab4 khaki1 khaki2 khaki3 khaki4  
 LightYellow1 LightYellow2 LightYellow3 LightYellow4 yellow1 yellow2 yellow3 yellow4  
 gold1 gold2 gold3 gold4 goldenrod1 goldenrod2 goldenrod3 goldenrod4 RosyBrown1  
 RosyBrown2 RosyBrown3 RosyBrown4 IndianRed1 IndianRed2 IndianRed3 IndianRed4 sienna1  
 sienna2 sienna3 sienna4 burlywood1 burlywood2 burlywood3 burlywood4 wheat1 wheat2  
 wheat3 wheat4 tan1 tan2 tan3 tan4 chocolate1 chocolate2 chocolate3 chocolate4 brown1  
 brown2 brown3 brown4 salmon1 salmon2 salmon3 salmon4 LightSalmon1 LightSalmon2  
 LightSalmon3 LightSalmon4 orange1 orange2 orange3 orange4 coral1 coral2 coral3 coral4  
 tomato1 tomato2 tomato3 tomato4 OrangeRed1 OrangeRed2 OrangeRed3 OrangeRed4  
 red1 red2 red3 red4 DeepPink1 DeepPink2 DeepPink3 DeepPink4 HotPink1 HotPink2  
 HotPink3 HotPink4 pink1 pink2 pink3 pink4 PaleVioletRed1 PaleVioletRed2 PaleVioletRed3  
 PaleVioletRed4 maroon1 maroon2 maroon3 maroon4 VioletRed1 VioletRed2 VioletRed3  
 VioletRed4 magenta1 magenta2 magenta3 magenta4 orchid1 orchid2 orchid3 orchid4  
 plum1 plum2 plum3 plum4 MediumOrchid1 MediumOrchid2 MediumOrchid3 MediumOrchid4  
 DarkOrchid1 DarkOrchid2 DarkOrchid3 DarkOrchid4 purple1 purple2 purple3 purple4  
 MediumPurple1 MediumPurple2 MediumPurple3 MediumPurple4 thistle1 thistle2 thistle3 thistle4

If you don't know how to spend your time, give them a try.

There's a flag to enable the showing of a cell age, but it makes the game quite boring.

I don't think that the other fields can give interesting variation if changed.

## 5 Acknowledgments

The project of *TicTacTic* was accomplished on a Linux box, using the various tools from the GNU project including texinfo, GCC and OCAML compiler, the Tk toolkit and others.

The OCAML compiler has been developed by Institut National de Recherche en Informatique et en Automatique (INRIA): for more information, please check the CAML home page (<http://pauillac.inria.fr/caml>) at [pauillac.inria.fr/caml](http://pauillac.inria.fr/caml) or [ftp.inria.fr](http://ftp.inria.fr).

I used the icon `ttt.xpm`, which I found associated to `kreversi` by Mario Weilguni.

The original library interface to Tk for the various widgets has been projected and developed mainly by Matteo Frigo, while working for an other project. Matteo Frigo gave me some very useful hints for the game optimization (even if I have not yet implemented them all), and before this, he introduced me to the world of functional programming.

The project is hosted and uses the facilities of the SourceForge site.



## 6 License and Copyright

*TicTacTic* is copyright © 2000 Gianluca Moro

*TicTacTic* is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA. You can also find the GPL on the GNU web site (<http://www.gnu.org/copyleft/gpl.html>).

For more informations, contact the author ([giammy@artemide.dei.unipd.it](mailto:giammy@artemide.dei.unipd.it))



## 7 Concept Index

### A

alpha-beta pruning ..... 3

### C

compiling ..... 7

configuration ..... 4, 6, 7

customize ..... 7

### G

game ..... 3

gcc ..... 7, 9

### I

installing ..... 7

### L

language ..... 5

levels ..... 5

Linux ..... 7

### N

newgame ..... 5

### O

ocaml ..... 7, 9

### P

players ..... 5

precompiled ..... 7

### R

Redhat ..... 7

### T

tear-off ..... 6

tk ..... 1, 7, 9

### W

Windows ..... 7





# Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>1</b>
<b>2</b>	<b>Game .....</b>	<b>3</b>
2.1	Main idea of the game .....	3
2.2	The rules .....	3
2.3	The heuristic .....	3
2.4	The code .....	4
<b>3</b>	<b>Menu .....</b>	<b>5</b>
3.1	New Game .....	5
3.2	Players .....	5
3.3	Levels .....	5
3.4	Language .....	5
3.5	Tear-off menus .....	6
3.6	Save Preferences .....	6
<b>4</b>	<b>Installation and Customization .....</b>	<b>7</b>
4.1	Installation on Unix .....	7
4.2	Installation on non-Unix Systems .....	7
4.3	Customizing the game .....	7
<b>5</b>	<b>Acknowledgments .....</b>	<b>9</b>
<b>6</b>	<b>License and Copyright .....</b>	<b>11</b>
<b>7</b>	<b>Concept Index .....</b>	<b>13</b>

