
ADVANCE
DEVELOPER'S MANUAL

DEVELOPER AGREEMENT

RIGHTS, RESPONSIBILITIES, DUTIES

You are the exclusive owner and copyright holder of the missions, themes and phrasebooks produced by you. All responsibility relative to them rests entirely with you. You can distribute them under any license of your choice, except that if you make a profit out of them you must own legally a non-demo copy of **BOH**. You can redistribute this manual only at no charge and in unmodified form.

DISCLAIMER

THIS MANUAL IS PROVIDED "AS-IS", WITHOUT ANY WARRANTY. TO THE FULLEST EXTENT ALLOWED BY LAW, THE AUTHOR CANNOT BE HELD LIABLE FOR ANY DAMAGE ARISING OUT OF THE (INABILITY OF MAKING) USE OF IT. USE AT YOUR OWN RISK.

GOVERNING LAW AND JURISDICTION

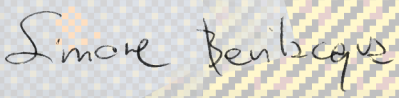
This agreement shall be governed by the laws of the country of residence of the author at the time of the dispute. The author reserves the right to appoint the venue for the dispute.

HELCOHE

BOH can be expanded by means of new missions, themes and phrasebooks. The game program has been written so that developing additional content requires no custom tool.

This manual illustrates all there is to know to create your own expansions. It assumes that you are very familiar with the game and borrows concepts and terminology from the user's manual.

The power of making your and other players' experience even more exciting is in your hands!



MISSIONS

To create missions, a copy of **BOH** and a plain text editor[1] are sufficient, but it is recommended to use pencil and paper as well.

DESCRIPTORS

Mission phases are defined by descriptors, ISO 8859-1 text files[2] stored in the directory **`missions`**[3] according to these naming rules:

- ASCII characters only;
- maximum length: 39 characters;
- first phase: **`<mission name>.bpd`** (e.g. **`Geometry.bpd`**);
- other phases: **`<mission name>-<phase number>.bpd`** (e.g. **`Geometry-2.bpd`**).

Errors in descriptors are written to **`BOH-log.txt`**[4].

HEADER

Every descriptor begins with a header that defines the phase characteristics through statements in the format **`key[=value(s)]`**.

```
BPDA
DEPENDENCIES                    = <mis.>[§<mis.>…]
BRIEF.START[.<LANGUAGE>]        = <text>[§<text>…]
BRIEF.SUCCESS[.<LANGUAGE>]      = <text>[§<text>…]
BRIEF.FAILURE[.<LANGUAGE>]      = <text>[§<text>…]
MESSAGE_POINT.0[.<LANGUAGE>]    = <text>[§<text>…]
MESSAGE_POINT.0.DISABLINGS      = <i>[<SPACE><j>…]
  …
MESSAGE_POINT.N[.<LANGUAGE>]    = <text>[§<text>…]
BATTLE_LIGHTING.0               = <R G B>
BATTLE_LIGHTING.1               = <R G B>
RECHARGE                        = <value>
TIME_LIMIT                      = <MM:SS>
```

---

1. Editors capable of column and block editing are highly recommended.

2. To make your files cheat-proof, submit them for encryption to **`contact@retream.com`**.

3. In the program directory (and/or where **Windows** relocated it; e.g. **`%LOCALAPPDATA%\VirtualStore\Program Files (x86)\BOH`**).

4. In **`T:`** on **AmigaOS/AROS** or in **`%TMP%`** on **Windows**.

```
DIFFICULTY                    = <value>
CREDITS                       = <credit>[§<credit>…]
DATE                          = <YYYYMMDD HH:MM>
MAP
```

- **BPDA**[5]: abbreviation of **BOH Phase Descriptor (type) A**
- **DEPENDENCIES**[6 7]: name(s) of the other mission(s) to complete in order to unlock this mission
- **BRIEF.*[.<LANGUAGE>]**[6 8]: text(s) shown[9] when a phase is started (**.START**), completed (**.SUCCESS**) or failed (**.FAILURE**)
- **MESSAGE_POINT.X[.<LANGUAGE>]**[8]: text(s) shown[10] by the **X**th[11] message point
- **MESSAGE_POINT.X.DISABLINGS**[6]: index(es) of the message point(s) disabled when the **X**th message point is accessed
- **BATTLE_LIGHTING.0** and **BATTLE_LIGHTING.1**[12]: colors the ambient illumination fades between while the EM attacks directly
- **RECHARGE**[6 13]: recharge awarded upon completion of the phase
- **TIME_LIMIT**[6]: time the phase must be completed within
- **DIFFICULTY**[7]: degree of difficulty (**0** = training; **1** = very low; **2** = low; **3** = average; **4** = high; **5** = very high)
- **CREDITS**[7]: author(s) credits
- **DATE**[7]: last modification date and time
- **MAP**[14]: end of header / beginning of map

---

5. No characters allowed before.

6. Optional.

7. Only for first-phase descriptors.

8. The **<LANGUAGE>** text is used if the configuration specifies such language. **<LANGUAGE>** must be written in English (e.g. **BRIEF.START.DUTCH**). If no text is defined for the language specified by the configuration, the default one is used. The default text must always be defined and written in English.

9. Each text is printed on a single line (squeezed, if necessary).

10. Each text is printed on multiple lines when it does not fit on a single line. It is best to keep texts as short as possible so that they do not cover too big an area of the screen and the player does not need too long to read them.

11. **X** can be at most 255.

12. Mandatory if the summoning point is defined.

13. Only for phases other than the last.

14. Must be the last statement.

EXAMPLE

```
BPDA
DEPENDENCIES                = Caves of Steel§Naked Sun
BRIEF.START                 = Get out of here.§Now!
BRIEF.START.ITALIAN         = Esci fuori di qui.§Ora!
BRIEF.SUCCESS               = Not that bad, outside.
BRIEF.SUCCESS.ITALIAN       = Non è malaccio, fuori.
BRIEF.FAILURE               = :(
MESSAGE_POINT.0             = Inside the womb.
MESSAGE_POINT.0.ITALIAN     = Dentro il ventre.
BATTLE_LIGHTING.0           = 000 000 000
BATTLE_LIGHTING.1           = 255 000 000
RECHARGE                    = 40
TIME_LIMIT                  = 5:30
DIFFICULTY                  = 3
CREDITS                     = Elijah Baley§Daneel Olivaw
DATE                        = 20160923 14:16
MAP
```

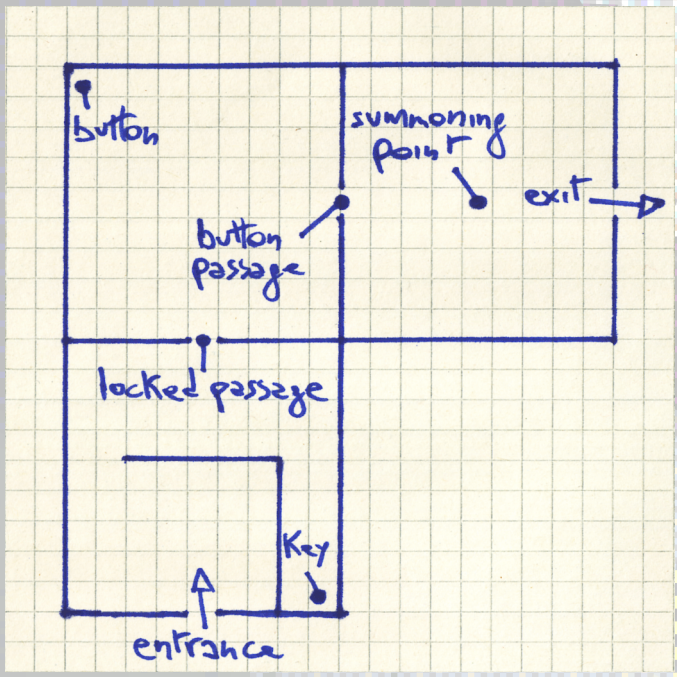
- **DEPENDENCIES**: the mission is unlocked only if the missions **Caves of Steel** and **Naked Sun** have been completed
- **BRIEF.START[.ITALIAN]**: texts shown[15] (on two lines) before the phase starts
- **BRIEF.SUCCESS[.ITALIAN]**: text shown[15] after the phase is completed successfully
- **BRIEF.FAILURE**: text shown after the phase ends in a failure
- **MESSAGE_POINT.0[.ITALIAN]**: text shown[15] when the RM looks at the first (and only) message point in the map
- **BATTLE_LIGHTING.X**: while the EM attacks directly, the ambient illumination fades between black and pure red
- **RECHARGE**: the shield gets recharged by 40% if the player completes the phase
- **TIME_LIMIT**: the phase must be completed within 5 min. and 30 sec.
- **DIFFICULTY**: the mission difficulty is average
- **CREDITS**: the mission was created by **Elijah Baley** and **Daneel Olivaw**
- **DATE**: the descriptor was last modified on Sep. 23, 2016, at 14:16

15. The Italian text is used if the configuration specifies the Italian language.

MAP

The map follows the header and defines the battlefield. Each of its characters represents a tile.

To illustrate the details of maps, this section shows how to create one (and the related header) starting from this sketch[16]:



16. For simplicity, barriers have been represented with lines, but for more complex maps it is best to use a square for each tile.

STEP 1 - defining one of the three areas[17] as a 9x9 tiles box:

```
#########
#.......#
#.......#
#.......#
#.......#
#.......#
#.......#
#.......#
#########
```

# = barrier
. = ordinary floor 0

[18]

STEP 2 - completing the basic layout by copying & pasting the area and adding the barrier in the BLA[19]:

```
###################
#.......#.......#
#.......#.......#
#.......#.......#
#.......#.......#
#.......#.......#
#.......#.......#
#.......#.......#
###################
#.......#
#.......#
#.#####.#
#.....#.#
#.....#.#
#.....#.#
#.....#.#
#########
```

# = barrier
. = ordinary floor 0

[20]

17. They have not be called "rooms" because what they actually look like depends on the theme.

18. The white area represents the text editor view.

19. For convenience: TLA = Top-Left Area; TRA = Top-Right Area; BLA = Bottom-Left Area.

20. The lines can (and should) be trimmed at the last non-**<SPACE>** character (they will be padded with **<SPACE>** upon loading).

STEP 3 - adding the passages:

```
##################
#.......#.......#
#.......#.......#
#.......#.......#
#.......0.......E
#.......#.......#
#.......#.......#
#.......#.......#
####\#############
#.......#
#.......#
#.######.#
#.....#.#
#.....#.#
#.....#.#
#.....#.#
####e####
```

# = barrier
. = ordinary floor O
e = entrance
E = exit
**** = locked passage
0 = passage A

The extra line at the bottom was added because the entrance needs a void tile attached to its outward side[21]. That would not have been necessary if the bottom had been something like this:

```
#.....#.#
#.##e####
###
```

# = barrier
. = ordinary floor O
e = entrance

21. Only **#**, **E** and **<SPACE>** can be used on the boundaries of the editing area.

STEP 4 - adding a key[22] for the locked passage, the button A[22] for the passage A and the summoning point[22][23]:

```
##################
#0.....#.......#
#.......#.......#
#.......#.......#
#.......0...S...E
#.......#.......#
#.......#.......#
#.......#.......#
####\#############
#.......#
#.......#
#.#####.#
#.....#.#
#.....#.#
#.....#.#
#.....#k#
####e####
```

# = barrier
. = ordinary floor 0
e = entrance
E = exit
**** = locked passage
0 = passage A
0 = button of passage A
k = key
S = summoning point

STEP 5 - definining the environment features by adding at the top the line `1 64 64 16 Main`[24], where:

- `1` places the map one level[25] above the ground;
- `64 64 16` is the RGB triple that defines the color (dark yellow) of the ambient illumination;
- `Main` is the label[26] that the game will display in the HUD beside the level number.

22. Without them, the program would have rejected the map, as there would have been no way to open the passages.

23. The exit key can replace the summoning point in non-final phases maps .

24. Had such line been omitted, the level would have defaulted to `0`, the ambient illumination to black and the label to an empty string. The line will be replaced with a row of void tiles upon loading.

25. "Floor" has not been used to avoid confusion with floor tiles. The level number must be in `[-120, 127]`.

26. The label consists of any text until the end of the line. It is recommendable to keep it short. It is optional.

STEP 6 - adding a (minimal) header to obtain a complete descriptor, so that the map can be tested already:

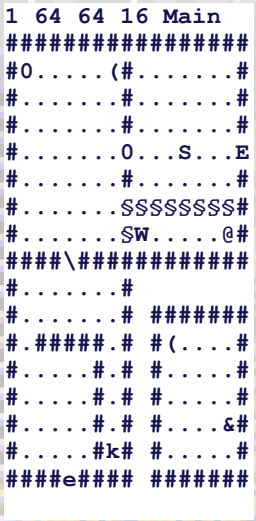
```
BPDA
BATTLE_LIGHTING.0 = 255 000 255
BATTLE_LIGHTING.1 = 000 255 000
DIFFICULTY        = 4
CREDITS           = Simone Bevilacqua
DATE              = 20160922 17:45
MAP
1 64 64 16 Main
##################
#0......#.......#
#.......#.......#
#.......#.......#
#.......0...S...E
#.......#.......#
#.......#.......#
#.......#.......#
####\############
#.......#
#.......#
#.#####.#
#.....#.#
#.....#.#
#.....#.#
#.....#k#
####e####
```

STEP 7 - making the map more interesting by:

- dividing the TRA in two parts with a transparent barrier;
- placing a Class A weapon[27] inside the smaller part of the TRA (note how the player can see the weapon also from the TLA);
- adding an area (Bottom-Right Area = BRA) connected to the rest of the map by means of teleporters[28];
- adding staircases to connect the BRA and the TRA part where the weapon is to some other place.

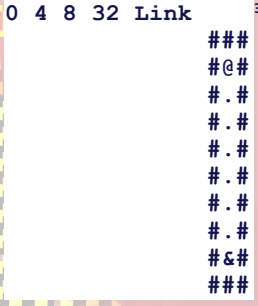
27. This affects radically the difficulty rating: before the mission was hard, despite the simplicity of the map, because the lack of a powerful weapon (as well as of other power-ups) made fighting against the EM very difficult.

28. The use of the same character is what connects the teleporters.



# = barrier
. = ordinary floor 0
e = entrance
E = exit
\ = locked passage
0 = passage A
0 = button of passage A
k = key
S = summoning point
§ = transparent barrier
W = class A weapon
@ = staircase
& = staircase
(= teleporter

Such map would not validate because the staircases lead nowhere. Since staircases always lead to a different level, a solution is adding a lower[29] level[30] that both the staircases connect to[31]:



[32]

# = barrier
. = ordinary floor 0
@ = staircase
& = staircase

29. The level chosen is the one immediately below, but it could have been any other in the allowed range.

30. It is a good practice to always have levels match spatially as in the example, although the program does not perform any check in such regard.

31. The use of the same character is what connects the staircases.

32. The different ambient illumination distinguishes the levels further.

STEP 8 - refining the map: the map allows the RM to fight the EM without stumbling upon the class A weapon first; however, while that is generally a legitimate design choice, here the idea is to make sure that the player finds the weapon - one of the many ways to obtain that is this:

```
1 64 64 16 Main
##################
#0..?..(#.......#
#...#...#.......#
#####...#.......#
#.......0...S...E
#.......#.......#
#.......§§§§§§§§#
#.......§C.....@#
####\############
#.......#
#.......# #######
#.#####.# #(....#
#.....#.# #.....#
#.....#.# #.....#
#.....#.# #....&#
#.....#k# #.....#
####e#### #######

0 4 8 32 Link
                ###
                #@#
                #.#
                #W#
                #.#
                #.#
                #.#
                #.#
                #&#
                ###
```

# = barrier
. = ordinary floor 0
e = entrance
E = exit
**** = locked passage
0 = passage A
0 = button of passage A
k = key
S = summoning point
§ = transparent barrier
W = class A weapon
@ = staircase
& = staircase
(= teleporter
? = hidden passage
C = remote control

Now the player will come across the class A weapon on the way to the remote control, which is strictly necessary to reach the summoning point (remote control → hidden passage → button of passage A → passage A → summoning point).

STEP 9 - putting the finishing touches to the upper level:

- a normal automapper;
- a 360° viewer;
- a partial recharge;
- a few obstacles;
- some merely aesthetic changes.

```
1 64 64 16 Main
##################
#0..?,¶(#......®#
#..r#,¶,#.%.....#
#####,¶,#.......#
#,,,,,,,0...S...E
#,¶¶,,,,#.......#
#,,¶,,,,§§§§§§§§#
#¶m¶,,,,§C.....@#
####\#############
#.V.....#
#....%..# #######
#.#####.# #(...%#
#;;;;;#.# #.::::.#
#;;;;;#.# #.::::.#
#;;;;;#.# #.::::&#
#;;;;;#k# #.....#
####e#### #######
```

```
# = barrier
. = ordinary floor 0
e = entrance
E = exit
\ = locked passage
0 = passage A
0 = button of passage A
k = key
S = summoning point
§ = transparent barrier
W = class A weapon
@ = staircase
& = staircase
( = teleporter
? = hidden passage
C = remote control
m = normal automapper
V = 360° viewer
r = partial recharge
, = ordinary floor 1
: = ordinary floor 2
; = ordinary floor 3
¶ = block
% = explosive object
® = rotating lamp
```

The automapper, the 360° viewer, the partial recharge and the class A weapon added earlier affect radically the difficulty of the mission, which now is very low.

STEP 10 - adding a briefing message and putting eveything together into the final descriptor:

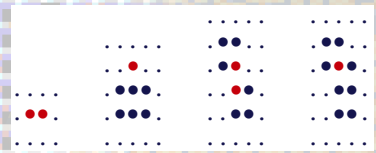
```
BPDA
BRIEF.START          = There's a lot to learn, here.
BRIEF.START.ITALIAN = C'è tanto da imparare, qui.
BATTLE_LIGHTING.0    = 255 000 255
BATTLE_LIGHTING.1    = 000 255 000
DIFFICULTY           = 1
CREDITS              = Simone Bevilacqua
DATE                 = 20161108 10:58
MAP
1 64 64 16 Main
##################
#0..?,¶(#......®#
#..r#,¶,#.%.....#
#####,¶,#.......#
#,,,,,,,O...S...E
#,¶¶,,,,#.......#
#,,¶,,,,§§§§§§§§#
#¶m¶,,,,§C.....@#
####\############
#.V.....#
#....%..# #######
#.#####.# #(...%#
#;;;;;#.# #.::::.#
#;;;;;#.# #.::::.#
#;;;;;#.# #.::::&#
#;;;;;#k# #.....#
####e#### #######

0 4 8 32 Link
             ###
             #@#
             #.#
             #W#
             #.#
             #.#
             #.#
             #.#
             #&#
             ###
```

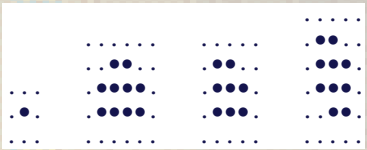
FRAME COMPOSITION RULES

Some tiles must be placed according to these rules:

- isolated tiles are allowed;
- sub-clusters must be rectangular and at least 2x2;
- any two adjacent/overlapping rectangular sub-clusters must adjoin/overlap by two or more tiles.



illegal layouts



legal layouts

MAP CHARACTERS TABLE

character(s)	tile kind	notes
r	partial recharge	
R	total recharge	
w	class B weapon	
W	class A weapon	
A	aiming device	
l	mid light	
L	big light	
V	360° viewer	
m	normal automapper	
M	advanced automapper	
d	short-range detector	
D	long-range detector	

C	remote control	
k	key	
K	exit key	at most one per phase; cannot co-exist with a summoning point; cannot be used in final phases[33]
0 1 2 3 4 5 6 7 8 9 ₁ ₂ ₃ ₒ ₒ ₐ	buttons or button passages	0 = button (of) passage(s) A(x), 1 = button (of) passage(s) B(x), etc.; if specified between two opposite barriers and two opposite tiles of another kind, a character defines a passage; otherwise, it represents a button connected to the passage(s) defined with the same character
.	ordinary floors 0	main floors
,	ordinary floors 1	alternative floors
:	ordinary floors 2	alternative floors; obey the frame composition rules
;	ordinary floors 3	alternative floors; obey the frame composition rules
^ v < >	moving floors	the direction is the one that the character visually points to
_	enemy fields	obey the frame composition rules
~	traps	obey the frame composition rules
¡	crumbly floors	to avoid inconsistent graphics, should not be used beside pits
!	pits	obey the frame composition rules
#	barriers	
§	transparent barriers	
¶	blocks	obey the frame composition rules

33. In such case, the program replaces it with a summoning point.

®	rotating lamps	
©	fixed lamps	
%	explosive objects	
( ) [ ] { } " '	teleporters	each character defines a teleporter; teleporters defined with the same character are chained together (the order the RM travels through them depends on the positions of the characters in the map, according to a left-to-right, top-to-bottom scan order)
@ & + − * / ± × ÷ ¬	staircases	each character defines a staircase; a staircase must be connected to another single one specified with the same character; there must be at least a tile in the surroundings that gives access to the staircase
=	timed passages	must be placed between two opposite barriers
\	locked passages	must be placed between two opposite barriers
?	hidden passages	must be placed between barriers and/or other hidden passages
|	automatic passages	must be placed between two opposite barriers
« »	one-way passages	must be placed between two opposite barriers; direction they can be passed through, depending on the orientation of the surrounding barriers: « = left/up; » = right/down
e	entrance	must be placed between two opposite barriers and a void tile
E	exit	must be placed between two opposite barriers

µ	message points	the numbering which determines the correspondence to the texts defined in the header depends on the positions of the characters in the map, according to a left-to-right, top-to-bottom scan order; there must be at least a tile in the surroundings that gives access to the message point
S	summoning point	at most one per phase; must be present in final phases; cannot co-exist with an exit key
<SPACE>	void tile	empty space; blocks everything

TIPS

- Make sure that there is always a way to complete the mission.
- Make sure there are no ways other than the intended one(s) to complete the mission.
- Test the mission thoroughly throughout the development (especially the final version, from start to finish).
- Define the main structure of the map as first thing.
- Add the message points and make the aesthetic changes only when the structure of the map is not going to change anymore.
- Scatter clues here and there.
- Evaluate the difficulty of a phase considering:
  - how complicated figuring out the right path(s) is;
  - the number of real and dummy passages;
  - whether the button passages are in sequence;
  - the map structure (size, complexity, levels, pits, etc.);
  - the number of recharges;
  - the available devices and how difficult finding them is;
  - the time limit;
  - the ambient lighting;
  - the flashing during the direct attack of the EM;
  - how it compares to the official missions phases.
- Set the difficulty of a multi-phase mission according to the difficulty of the hardest of its phases.

THEMES

The **BOH**-specific part of creating themes is not that difficult, as it mostly consists in arranging some data according to a few (simple) rules; the demanding part is creating the graphics, sound effects and music, which requires artistic skills. For a basic theme you need (the demo of) **BOH** and a plain text editor; for custom graphics, sound effects and music you need also the appropriate editing tools.

GENERAL INFORMATION

- A theme is a set of text, graphic and sound files in a directory named after the theme and located in the directory **`themes`**[34].
- Text files use the ISO 8859-1 charset.
- Graphics:
  - can have any color depth up to 24 bit[35] and an alphachannel;
  - use the RGB triple **`<128 128 129>`** as colorkey;
  - are stored as BMP files[36] according to the following: the color data is stored in files with **`.c.bmp`** extension; the alphachannel data is stored in 8 bit grayscale files with **`.a.bmp`** extension (if the **`.c.bmp`** and the **`.a.bmp`** files are both provided, the color and alphachannel data get combined[37]; if only the **`.c.bmp`** file is provided, the transparency is dictated by the colorkey; if only the **`.a.bmp`** file is provided, the color data is considered white).
- Sound effects are stored as OGG files[38] (16 bit; mono or stereo <= 44100 Hz for the menu; mono <= 22050 Hz for the game).
- Music tracks are stored as OGG files (16 bit stereo <= 44100 Hz).
- The missing files are loaded from the theme **Sci-Fi** [39].

In case of errors, the program falls back to the theme **Sci-Fi** and outputs a message to **`BOH-log.txt`**[4].

34. In the program directory.

35. 4/8 bit indexed images are recommended to keep the size down.

36. Uncompressed, without color space information.

37. The width and height of the images must match.

38. To reduce memory usage, keep sound effects short.

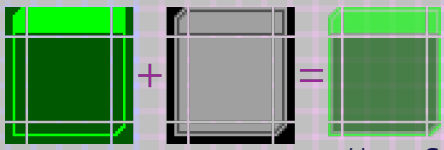
39. This allows testing incomplete themes and keeps the size down when some data would just be replicated.

WALLPAPER

The menu background graphics are stored in **`wallpaper.*.bmp`**. The size can be any[40].

TEMPLATE

The menu areas graphics are stored in **`template.*.bmp`**. They are conceptually divided into nine sections so that they can be stretched/squeezed to cover areas of various sizes by repeating and/or clipping the non-corner sections.



`template.a.bmp` + **`template.c.bmp`** (theme: **Sci-Fi**)

section	**dimensions** (pixels)
top-left corner	WxH: 8 x <font height> + 4
top side	H: <font height> + 4
top-right corner	WxH: 8 x <font height> + 4
left side	W: 8
right side	W: 8
bottom-left corner	WxH: 8x8
bottom side	H: 8
bottom-right corner	WxH: 8x8

- The central section occupies the remaining area of the image (recommended size: 32x32 pixels).
- The horizontal/vertical sides inherit the width/height from the central section.
- The top side is where menu titles are vertically centered.

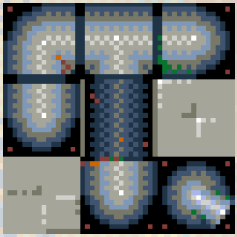
40. Graphics get clipped if they overshoot the screen boundaries due to their size and/or position.

TILES

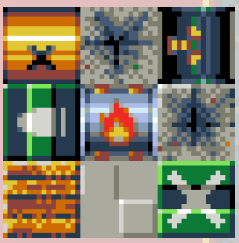
The battlefield tiles graphics are stored in **`tiles.c.bmp`** according to these rules:

- the image size is 256x880 pixels;
- the tile size is 16x16 pixels;
- tiles are arranged in 55 rows of 16 tiles each, without any space.

The tiles should be drawn with a directionally neutral lighting in order not to clash with the dynamic lighting of the game[41]. To give depth to graphics, it is best to use a vertical lighting - for example:



That said, such lighting still is not really suitable to convey depth and tends to produce "pillow-shaded" graphics. Therefore, it is acceptable to use a minimally directional lighting as in these tiles:



Another thing to keep in mind is that some tiles (like those above) may cast shadows on the adjacent ones[42].

41. For example, if a block were drawn with a strong light coming from the right, it would look very bad when illuminated from the left.

42. The pipes in the example picture lay on black plates so that they do not look brighter than the adjacent tiles when the latter are shadowed.

TILES INDEXES TABLE

row(s)	tile kind	**indexes** (from left to right)
0	partial recharge[43]	looping animation: 0-14 empty tile: 15
1	total recharge[43]	looping animation: 0-14 empty tile: 15
2	class B weapon[43]	looping animation: 0-14 empty tile: 15
3	class A weapon[43]	looping animation: 0-14 empty tile: 15
4	aiming device[43]	looping animation: 0-14 empty tile: 15
5	mid light[43]	looping animation: 0-14 empty tile: 15
6	big light[43]	looping animation: 0-14 empty tile: 15
7	360° viewer[43]	looping animation: 0-14 empty tile: 15
8	normal automapper[43]	looping animation: 0-14 empty tile: 15
9	advanced automapper[43]	looping animation: 0-14 empty tile: 15
10	short-range detector[43]	looping animation: 0-14 empty tile: 15
11	long-range detector[43]	looping animation: 0-14 empty tile: 15
12	remote control[43]	looping animation: 0-14 empty tile: 15
13	passage key[43]	looping animation: 0-14 empty tile: 15

43. Interactive items must occupy an area with a diameter of 9 pixels or less at the center of tiles.

14	exit key[43]	looping animation: 0-14 empty tile: 15
15	button[43]	looping animation: 0-14 pressed state: 15
16-17	ordinary floors 0[44]	scatterable tiles[45]: 0-15 extra-scatterable tiles[46]: 16-31
18-19	ordinary floors 1	scatterable tiles[45]: 0-15 extra-scatterable tiles[46]: 16-31
20-22	ordinary floors 2	contextual variants[47]: 0, 3, 6-7, 9, 11-14 inverse corners[48]: 1-2, 4-5 alternative tiles[49]: 16-23, 25, 27-30 scatterable inner tiles[45]: 32-47
23-25	ordinary floors 3	contextual variants[47]: 0, 3, 6-7, 9, 11-14 inverse corners[48]: 1-2, 4-5 alternative tiles[49]: 16-23, 25, 27-30 scatterable inner tiles[45]: 32-47
26	moving floors	northward animation: 0-3 southward animation: 4-7 eastward animation: 8-11 westward animation: 12-15

44. All the other tiles must be drawn to nicely border and mix with these ones.

45. The index of the tile that will actually appear in the game is extracted randomly from the given range.

46. If the tile 15 is extracted from the scatterable tiles, another extraction is performed to choose the final tile from [15, 31] (practically, the tiles 15-31 appear much less frequently than the tiles 0-14).

47. Contextual variants are used when the aspect of a tile is affected by adjacent tiles of the same kind. Here the frame composition rules apply. The adjacent tiles positions and the indexes match as follows (each tile is indicated with a letter relative to its position: N = North, S = South, E = East, W = West): <no tile>-0, NE-3, SE-6, NSE-7, NW-9, NWE-11, SW-12, NSW-13, SWE-14, NSWE-15 (e.g. tile 7 is for each tile with tiles of the same kind North, South and East of it).

48. The frame composition rules allow/require inverse (concave) corners. The corner positions and the indexes match as follows: NE-1, SE-2, NW-4, SW-5 (e.g. tile 1 is the North-East corner tile).

49. Alternative tiles are substitutes of the tiles in the image upper row and are extracted randomly with a 50% probability.

27-29	enemy fields	contextual variants[47]: 0, 3, 6-7, 9, 11-14 inverse corners[48]: 1-2, 4-5 alternative tiles[49]: 16-23, 25, 27-30 scattered looping animation[50]: 32-47
30-32	traps	contextual variants[47]: 0, 3, 6-7, 9, 11-14 inverse corners[48]: 1-2, 4-5 alternative tiles[49]: 16-23, 25, 27-30 scattered looping animation[50]: 32-47
33	crumbly floors	type 0 crack steps: 0-2 type 0 crumble animation: 3-4 type 1 crack steps: 5-7 type 1 crumble animation: 8-9 type 2 crack steps: 10-12 type 2 crumble animation: 13-14 pit: 15
34-35	pits[51]	contextual variants[47 52]: 0, 3, 6-7, 9, 11-15 inverse corners[48]: 1-2, 4-5 alternative tiles[49]: 16-23, 25, 27-31
36-37	barriers	contextual variants[53]: 0-15 alternative tiles: 16-31
38-39	transparent barriers	contextual variants[53]: 0-15 alternative tiles: 16-31
40-41	blocks	contextual variants[53]: 0-15 alternative tiles: 16-31
42	rotating lamps	looping clockwise animation[54]: 0-15
43	fixed lamps	looping animation: 0-15

50. The inner tiles are animated according to the given range, with the initial index extracted randomly.

51. The size of borders must be less than 5 pixels (preferably 3 or 4).

52. When the map is rendered, barriers and transparent barriers are treated as if they were pits, so that pits border nicely with them.

53. Here the frame composition rules do not apply. The adjacent tiles positions and the indexes match as follows: <no tile>-0, N-1, E-2, NE-3, S-4, NS-5, SE-6, NSE-7, W-8, NW-9, WE-10, NWE-11, SW-12, NSW-13, SWE-14, NSWE-15 (e.g. tile 1 is for each tile with a tile of the same kind North of it only).

54. Starting from 0°.

44	explosive objects (causing chain explosions)	type 0, intact: 0 type 0, destruction animation: 1-4 type 1, intact: 5 type 1, destruction animation: 6-9 type 2, intact: 10 type 2, destruction animation: 11-14 empty tile: 15
45	explosive objects (not causing chain explosions)	type 0, intact: 0 type 0, destruction animation: 1-4 type 1, intact: 5 type 1, destruction animation: 6-9 type 2, intact: 10 type 2, destruction animation: 11-14 empty tile: 15
46	teleporters[55]	inactive state animation: 0-5 active state animation: 6-15
47	staircases	going downstairs, eastward: 0 going downstairs, northward: 1 going downstairs, westward: 2 going downstairs, southward: 3 going upstairs, eastward: 4 going upstairs, northward: 5 going upstairs, westward: 6 going upstairs, southward: 7
48	timed passages	horizontal, closed: 0 horizontal, opening animation: 1-7 vertical, closed: 8 vertical, opening animation: 9-15
49	locked passages	horizontal, closed: 0 horizontal, opening animation: 1-7 vertical, closed: 8 vertical, opening animation: 9-15
50	hidden passages	horizontal, closed: 0 horizontal, opening animation: 1-7 vertical, closed: 8 vertical, opening animation: 9-15

55. The active area must be centered and have a diameter of 11 pixels.

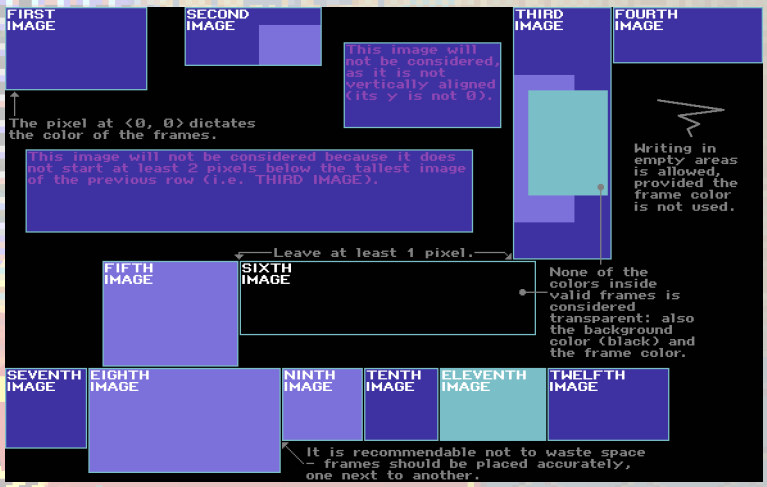
51	button passages	horizontal, closed: 0 horizontal, opening animation: 1-7 vertical, closed: 8 vertical, opening animation: 9-15
52	automatic passages	horizontal, open: 0 horizontal, closing animation: 1-7 vertical, open: 8 vertical, closing animation: 9-15
54-54	one-way passages	up, closed: 0 up, opening animation: 1-7 left, closed: 8 left, opening animation: 9-15 down, closed: 16 down, opening animation: 17-23 right, closed: 24 right, opening animation: 25-31
55	entrance	horizontal, open: 0 horizontal, closing animation: 1-7 vertical, open: 8 vertical, closing animation: 9-15
56	exit	horizontal, closed: 0 horizontal, opening animation: 1-7 vertical, closed: 8 vertical, opening animation: 9-15
57	message points	facing westward: 0 facing southward: 1 facing estward: 2 facing northward: 3
58	summoning point[56]	inactive state animation: 0-7 active state animation: 8-15

56. The active area must be centered and have a diameter of 13 pixels.

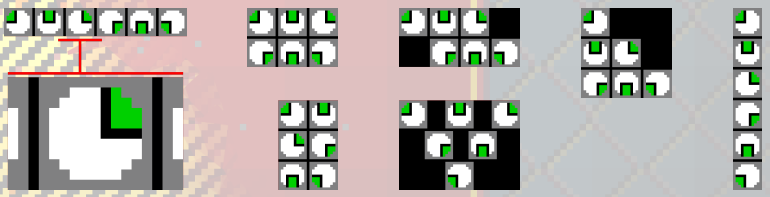
GRAPHICS ARRANGEMENT

Some images sets are stored in an image according to these rules:

- the images must be enclosed in 1-pixel-thick frames;
- the frames must be all in the same color (which can be any);
- the frame of the first image must be located at <0, 0>;
- the frames on the same row[57] must be spaced by at least 1 pixel;
- a row of frames must be spaced from the bottommost frame in the previous row by at least 1 pixel;
- the images order is left-to-right, top-to-bottom.



For readability, it is best to use the colorkey as both background color and frame color - like in these examples, which illustrate some equivalent ways of defining the same set of images:



57. Frames belong to the same row if their top sides are vertically aligned.

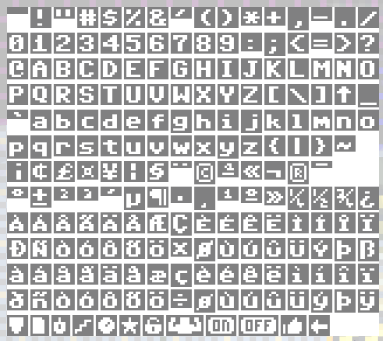
FONTS

The font base graphics are stored in **`font.*.bmp`** according to these rules:

- the charset is ISO 8859-1 (characters 32-255);
- the graphics obey the graphics arrangement rules;
- the width[58] and height[59] of each character can be any;
- the height of the font is given by the tallest character;
- the last row must contain these icons (in this order): shield, key, remote control, level, time, star, unlocked mission, training mission, "on", "off", "thumbs-up", "back".
- the empty columns of pixels around characters determine the horizontal spacing[60];
- vertical alignment is relative to the top of characters.

EXAMPLE

ISO 8859-1 charset



`font.c.bmp` (theme: **C64**)

- The colorkey is used for the frames and the background.
- The first character is a **<SPACE>**: it consists of an empty row of pixels[61] just to define the width.

58. It is best not to make characters (much) wider than those in the example.

59. It is best to keep the height under 12 pixels. Leaving no empty rows of pixels under characters makes for a little speed and memory optimization.

60. When printing, the program adds automatically a spacing of 1 pixel. Generally, variable-width fonts do not embed spacing in the graphics.

61. A fancy font might well have a visible **<SPACE>**.

FONT RECOLORING

Even though the characters in the previous example are white, in the game they appear in the classic **C64** cyan and light blue: that is because the image is just the base the actual fonts are derived from through recoloring.

Recoloring depends on three parameters (one per color channel): `RP`, `GP`, `BP`[62]. Each parameter indicates a percentage variation in `[-100, 100]` of the associated color channel: for example, setting `RP` to `50` strenghtens the red channel of the derived font by 50% with respect to the font base, so that, for instance, if a pixel of the font base has a red value of 64, the corresponding red value in the derived font is 96 (i.e. 64 + 50%). Since the biggest variation possible is +100% (i.e. at most values can be doubled, up to 255), it is best to start from a white base and to use negative parameters.

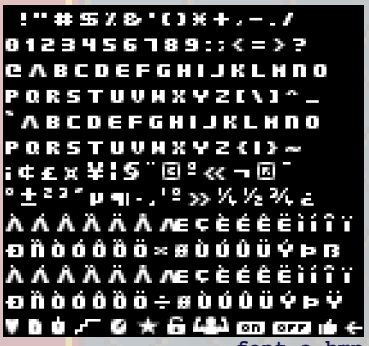
For example, to have the characters from the theme **C64** printed in orange, one could set `RP` to preserve the red channel, `GP` to halve the green channel and `BP` to turn off the blue channel entirely:

```
color given by RP=0 GP=-50 BP=-100
```

FONT ALPHACHANNEL

To create a smooth and/or a partially transparent font, it is possible to add an alphachannel file:

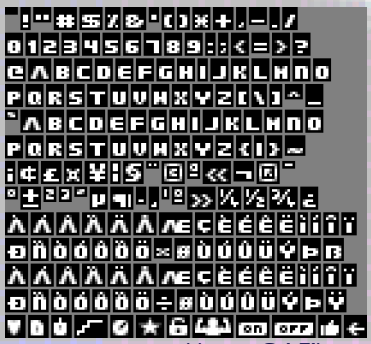
font.c.bmp



font.a.bmp

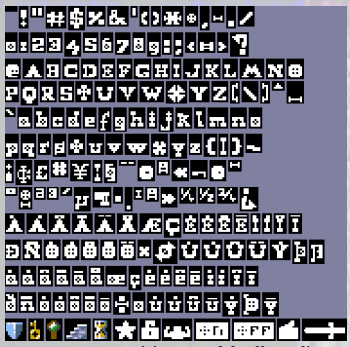
62. Specified in the descriptor (see **DESCRIPTOR**, p. 36).

Although such solution works, the possibility of providing just an alphachannel file allows to get rid of `font.c.bmp` altogether - all that is needed is making the alphachannel image obey the graphics arrangement rules:



`font.a.bmp` (theme: **Sci-Fi**)

Multicolored characters are allowed as well - for example, like some of the icons in these images:



`font.c.bmp` (theme: **Medieval**)

`font.a.bmp` (theme: **Medieval**)

When drawing multicolored characters, it is important to keep in mind that recoloring affects them as well.

SPRITES

The RM and enemies graphics are stored in **`sprites.*.bmp`** according to the graphics arrangement rules. The hot-spot of sprites is their center. For perfect centering, width and height must be odd.

SPRITES TABLE

index(es)	assignment and notes	animation[63]
0	RM doing nothing	-
1	empty image	-
2-9	RM dying because of shield implosion	one-shot
10-13	RM falling into a pit	one-shot
14-23	RM waiting	loop
24-28	RM opening a passage	one-shot
29-32	RM shooting; main part; last frame shown until release or next shot	one-shot
33-36	RM shooting; shooting part; close to target; last frame shown until release or next shot	one-shot
37-40	RM shooting; shooting part; far from target; last frame shown until release or next shot	one-shot
41-48	RM moving forward; main part[64]	loop
49-56	RM moving forward; shooting part	loop
57-64	RM moving backward; main part[64]	loop
65-72	RM moving backward; shooting part	loop
73-76	RM turning left; main part	loop
77-80	RM turning left; shooting part	loop
81-84	RM turning right; main part	loop

63. "-" = not animated; "one-shot" = animation played once from the first to the last frame; "loop" = animation played continuously.

64. The movement sound effects (see **SOUND EFFECTS TABLE**, p. 34) are played when the 3rd and 7th frames are shown.

85-88	RM turning right; shooting part	loop
89-92	class 6 enemy moving	loop
93-95	class 6 enemy hit by class A/B/C weapon	-
96-99	class 6 enemy dying	one-shot
100-103	class 5 enemy moving	loop
104-106	class 5 enemy hit by class A/B/C weapon	-
107-110	class 5 enemy dying	one-shot
111-114	class 4 enemy flying	loop
115-117	unused - define as empty areas	-
118-121	class 4 enemy dying	one-shot
122-125	class 3 enemy moving	loop
126-128	class 3 enemy hit by class A/B/C weapon	-
129-132	class 3 enemy dying	one-shot
133-136	class 2 enemy moving	loop
137-139	class 2 enemy hit by class A/B/C weapon	-
140-143	class 2 enemy dying	one-shot
144-147	class 1 enemy moving	loop
148-150	class 1 enemy hit by class A/B/C weapon	one-shot
151-154	class 1 enemy dying	one-shot
155-158	EM walking	loop
159-161	EM hit by class A/B/C weapon	-
162-165	EM dying	one-shot

Since the RM can shoot both while standing still and moving, to avoid a big amount of frames the sprites are divided in two parts: the part that shoots ("shooting") and the rest of the body ("main"). These parts should be drawn in a way that they always combine nicely.

The shooting part frames are laid over the main part ones, as shown in the following example[65]:

The shooting animation depends on the distance of the target:

The animation on the left is used when a target is very close, the one on the right when the target is sufficiently distant. The numerical values indicate the distance in pixels of the shooting point[66] from the center of the sprite (including the center pixel itself). The shooting point must be centered horizontally.

The core of sprites should cover a roughly circular area[67]:

sprites	**diameter** (pixels)	**example**
RM class 6 enemy	5	
class 5-3 enemies	7	
class 2 enemy	9	
class 1 enemy EM	11	

65. The sprites areas are designed to obtain horizontal and vertical centering.

66. This is where the calculations of the shooting trajectory begin from.

67. It is not strictly necessary to cover the areas precisely, but accuracy is highly preferable.

SOUND EFFECTS AND MUSIC TRACKS

Each sound effect and music track is stored in an own file.

SOUND EFFECTS TABLE

file(s)[68] (`.ogg` omitted)	played when
`ambient`	the game is being played (environmental noises)[69]
`button`	a passage button is pressed
`confirmation`	a menu selection is confirmed
`death-enemy-1…6/EM`	an enemy dies
`death-RM-pit`	a pit kills the RM
`death-RM-shield`	a shield implosion kills the RM
`detector`	a detector reports an enemy
`explosive_object-0…5`	an explosive object blows up
`floor-crumbly-crack`	crumbly floors crack
`floor-crumbly-crumble`	crumbly floors crumble
`floor-enemy_field`	enemy fields are audible[69]
`floor-moving`	moving floors are audible[69]
`floor-trap`	traps are audible[69]
`hit`	a shot hits something
`hit-enemy-1/2/3/5/6/EM`	an enemy is hit
`movement`	the RM moves over something other than what listed below
`movement-moving_floor`	the RM moves over moving floors
`movement-ordinary_floor-1…3`	the RM moves over ordinary floors
`movement-teleporter`	the RM moves over teleporters

68. Indexes indicate the types of objects the sound effects are relative to.

69. Loops continuously.

`navigation`	a menu item is selected
`passage-automatic`	an automatic passage closes
`passage-button`	a button passage opens
`passage-entrance`	the entrance closes
`passage-exit`	the exit opens
`passage-hidden`	a hidden passage opens
`passage-locked`	a locked passage opens
`passage-one-way`	a one-way passage opens/closes
`passage-timed`	a timed passage opens/closes
`picking_up`	the RM picks up an object
`shield`	the shield absorbs a hit
`teleporter`	a teleporter operates
`tick`	the time left is a multiple of 60 sec. or, if the time left is less than 1 min., once per sec.
`weapon-A...C`	a weapon shoots

When designing the sound effects, keep in mind that the volume they are played back

- ambient: the larger the area the RM is in, the louder;
- movements: the quicker, the louder;
- enemy fields/traps/moving floors: the closer/bigger, the louder;
- events: the closer, the louder.

MUSIC TRACKS TABLE

file(s) (`.ogg` omitted)	**played in a loop when**
`failure`	failure is reported
`menu`	the menu is active
`success`	success is reported

DESCRIPTOR

The characteristics of a theme are defined in **descriptor.txt**[70] through statements in the format **key=value(s)**.

```
WALLPAPER                       = <X Y>
GRADIENT                        = <Rt Gt Bt Rb Gb Bb>
AREA.PAGE                       = <X Y W H>
AREA.INFORMATION                = <X Y W H>
AREA.FULL_SCREEN                = <X Y W H>
AREA.REQUESTER                  = <X Y W H>
TEXTS_ALIGNMENT                 = left|center|right
CURSOR                          = <R G B>
FONT.TITLE                      = <RP GP BP>
FONT.BODY                       = <RP GP BP>
FONT.SELECTED_ITEM              = <RP GP BP>
FONT.UNSELECTED_ITEM            = <RP GP BP>
FONT.IN-GAME                    = <RP GP BP>
SHIELD                          = <R G B>
WEAPON.CLASS_A                  = <R G B>
WEAPON.CLASS_B                  = <R G B>
WEAPON.CLASS_C                  = <R G B>
LIGHT.SMALL                     = <R G B>
LIGHT.MID                       = <R G B>
LIGHT.BIG                       = <R G B>
LAMP.ROTATING                   = <R G B>
LAMP.FIXED                      = <R G B>
EXPLOSIVE_OBJECT.0              = <R G B>
EXPLOSIVE_OBJECT.1              = <R G B>
EXPLOSIVE_OBJECT.2              = <R G B>
EXPLOSIVE_OBJECT.3              = <R G B>
EXPLOSIVE_OBJECT.4              = <R G B>
EXPLOSIVE_OBJECT.5              = <R G B>
AUTOMAPPER.OBJECT               = <R G B>
AUTOMAPPER.BUTTON               = <R G B>
AUTOMAPPER.FLOOR                = <R G B>
AUTOMAPPER.DANGER               = <R G B>
AUTOMAPPER.BARRIER              = <R G B>
AUTOMAPPER.TRANSPARENT_BARRIER  = <R G B>
AUTOMAPPER.OBSTACLE             = <R G B>
AUTOMAPPER.EXPLOSIVE_OBJECT     = <R G B>
```

70. Each theme should have an own descriptor.

```
AUTOMAPPER.CONNECTION          = <R G B>
AUTOMAPPER.PASSAGE.GENERIC     = <R G B>
AUTOMAPPER.PASSAGE.HIDDEN      = <R G B>
AUTOMAPPER.EXIT                = <R G B>
AUTOMAPPER.SUMMONING_POINT     = <R G B>
CREDITS                        = <cred.>[§<cred.>...]
DATE                           = <YYYYMMDD HH:MM>
```

- **WALLPAPER**: pixel coordinates to center the menu wallpaper at
- **GRADIENT**[71]: colors of the topmost line and bottommost line of the menu background gradient
- **AREA.***: top-left corner pixel coordinates and dimensions of the menu areas (**.PAGE**: items page; **.INFORMATION**: additional information for selected item; **.FULL_SCREEN**: full screen information; **.REQUESTER**: string requester[72])
- **TEXTS_ALIGNMENT**: horizontal alignment of the menu texts
- **CURSOR**: color of the cursor in string requesters
- **FONT.*** recoloring parameters[73] for the menu fonts (**.TITLE**, **.BODY**, **.SELECTED_ITEM**, **.UNSELECTED_ITEM**) and for the in-game font (**.IN-GAME**)
- **SHIELD**: color the RM is recolored in when the shield gets hit
- **WEAPON.CLASS_***: colors of the weapons sparks, explosions and aiming lines
- **LIGHT.***: colors of the lights
- **LAMP.***: colors of the lamps
- **EXPLOSIVE_OBJECT.***: colors of the explosive objects explosions
- **AUTOMAPPER.***: colors of the tiles in the automapper maps
- **CREDITS**: author(s) credits
- **DATE**: last modification date and time

71. Can be omitted when the wallpaper covers the whole screen.

72. Automatically resized and centered, if needed.

73. See **FONTS**, p. 28.

PHRASEBOOKS

BOH allows to localize all the texts. However, the linguistical engine is quite simple, so characters are limited to the ISO 8859-1 charset (see **FONTS**, p. 28) and are printed only from left to right.

Translations are stored in text files located in the directory **`phrasebooks`**[34]. These files must carry the English names of their respective languages with **`.txt`** as extension (e.g. **`Polish.txt`**) and follow this scheme:

```
<language name>
<credit>[§<credit>…]
YYYYMMDD HH:MM
<first string>
…
<last string>
```

The lines represent, respectively:

- the native name of the language (e.g. **`Français`**);
- the author(s) credits;
- the last modification date and time;
- the localized strings.

Translations must:

- be faithful to the English one;
- respect the order of the lines;
- respect the order of the C placeholders (**`%s`**, **`%u`**, etc.);
- be grammatically correct;
- use neutral terms (because things look different in different themes - e.g. **`light`**, not **`flashlight`**);
- be consistent (synonyms to refer to the same thing in different places are not allowed);
- respect the case of characters (if possible);
- be as concise as possible.

CONTACTS

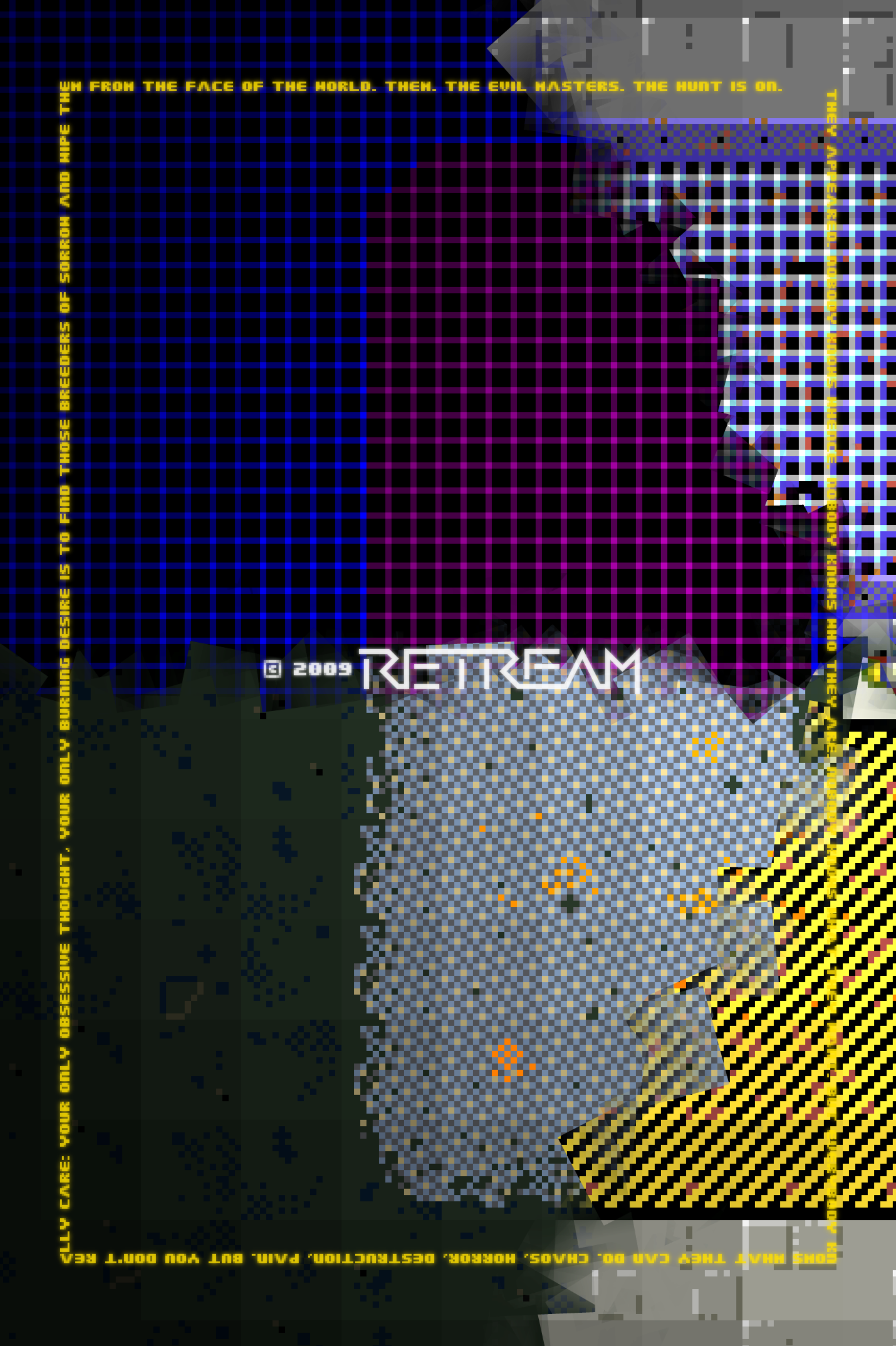
- email: **contact@retream.com**
- website: **www.retream.com/BOH**

ACKNOWLEDGEMENTS

Texts, graphics, editing:
Simone Bevilacqua

This font:
Sansation © 2008 **Bernd Montag**

Many thanks to:
Jesus / Davide Allegra / James Monkman / John Scolieri / Mark Ashley / customers / supporters


THEY APPEARED. NOBODY KNOWS WHENCE. NOBODY KNOWS WHO THEY ARE. NOBODY KNOWS WHY THEY ARE HERE. NOBODY KNOWS WHAT THEY CAN DO. CHAOS, HORROR, DESTRUCTION, PAIN. BUT YOU DON'T REALLY CARE: YOUR ONLY OBSESSIVE THOUGHT, YOUR ONLY BURNING DESIRE IS TO FIND THOSE BREEDERS OF SORROW AND WIPE THEM FROM THE FACE OF THE WORLD. THEN, THE EVIL MASTERS. THE HUNT IS ON.
© 2009 RETREAM