Fish Fillets 2 Editor Getting Started Guide

This document will be teaching you many of the basic features included in the Fish Fillets 2 Editor. Some of the features included are "Basic room file manipulation", inserting special objects and beams, and creating new objects. However, advanced topics such as scripting are not included in this manual. If you are interested in learning to write scripts, the best way is to go through some scripts of standard game rooms, which can be found in SCRIPTS/ROOMS folder.

Technical notes

Be sure to have Microsoft .NET Framework 1.1 Redistributable installed! You need this particular version; otherwise the editor will not work properly.
The editor has to be installed into the same folder as your game!
You might encounter problems running the editor from network drive; install both the game and the editor locally!

Abbreviations used in this manual

LMB – left mouse button
RMB – right mouse button
MMB – middle mouse button

Basic room file manipulation

Individual rooms are stored in files. All the common editor actions with these files can be done using the File menu. You will find self-explaining options there, such as Open room and Save room (for proper file naming and placing please see Folder structure and file naming conventions).
Re-open room action allows you to quickly get rooms to the state when it was saved for the last time.
New rooms can be created using the New mission command. A window will appear after selecting this option, where you can type mission and campaign name and all the basic parameters of the room. If you leave campaign name blank, only a single mission will be created (this is probably what you are looking for at the beginning of your work with the editor). Basic room parameters are *room size* in fields and *size of one game field* in pixels. Size of one game field is altered automatically so that room will fit into the window, when it is opened in the game. This size therefore matters only when a room is opened from the editor. Note that you can change all these room parameters later.

Dynamic properties of selected object

As you have probably noticed there is one more window present apart from the main editor window. This window is called *dynamic properties* and its contents change according to what type of object is currently selected in the main editor window for editing. We will provide a detailed overview of individual dynamic properties in following chapters of this manual. Four different sets of properties might be displayed.
First set is called *room properties* and is completely different from the other sets. It contains properties that are associated with whole room such as maximum number of moves, script file name or name of the image file that will be displayed behind used interface when a room is opened from

the game. The room properties are displayed when no other object in the room is selected (to deselect an object, click RMB on a spot, where there is no object).
The second set is called *room object properties* and it is a subset of the third set. It contains properties that are associated to single objects' in the room such as its weight, name for the scripts, a material. Room object properties are displayed whenever some *non-critter object* is selected (to select an object, click LMB on it).
The third set is called *critter properties* and contains all the room object properties plus some extras such as critter look direction. The critter properties are displayed whenever a critter is selected.
The last set is called *beam properties* and contains properties connected to the beams; the only changeable property is *beam direction,* at the moment. Beam properties are displayed when beam tool is selected (using Beam tool from the Editing tools menu) and you click on some beam emitter.

Using Mighty editing tool to create game objects

Now we are getting to the part where you will learn how to create objects in the game. There is a tool for manipulating objects and their shapes. This tool is the default one when the editor is turned on, or it can be selected using Mighty tool command from the Editing tools menu.
You can *create new object* by clicking the LMB on a free place where no object is present. Newly created object becomes object selected for editing.
After new object is created, you will probably want to *alter* its *shape*. You can do this by clicking the LMB on a free field. This will add the field to object's shape. Removing field from the object's shape is done using the RMB on fields filled with the shape. You can add larger amount of game fields at once by using LMB with Shift or Ctrl. LMB with Shift can be used to add whole rectangles (rectangle is drawn from last clicked game field as one corner to currently clicked game field as second corner). LMB with Ctrl can be used to add whole line (line is drawn from last clicked game field as one point to currently clicked game field as second point). This works rather similarly for RMB and removing.
When the last field is removed from the object, the whole object is deleted. You can *delete objects* also by pressing Del key when an object is selected. Please note that there is a special object in every room that can't be deleted. It is the static part of the room that is called the wall.
Mighty tool allows you to edit not only the object's shape, but also its position. This is be done by dragging the object to a different place in the room. When object is dragged, its position changes, but when you drag the object on some place where it doesn't fit, only a red outline is drawn.
You can forbid further shape and position editing by selecting Locked shape or Locked position in the room object properties.
Editing of the object's shape and position is just one part of designing the room. Now, that you have your objects, you will need to change their types. This can be done using Weight in the room object properties. You can set an object to be *ice*, *seaweed*, *normal object* or *steel pipe*. Other types available in combo box are not valid for common objects.
Another interesting feature is the possibility to set an object as *dangerous*. This can be done using Dangerous in the room object properties. Dangerous objects can only be touched with the turtle. The other critters are not allowed to touch them.

Using Insert tool to insert special object into the room

Now you are familiar with ways of creating game objects, but as you noticed, you still don't know the way that allows you to add special objects (*critters* or *stars*) into the room. And this is exactly what Insert tool from Editing tools menu can be used for.
When this tool is activated, a small window containing names of all the critters and a star appears. As you select any of these, an image of the special object will appear in the room and you can move

your mouse to find a suitable location for it. This object can be placed using LMB. By RMB, you will cancel inserting and default editing tool (Mighty tool) is automatically selected.

Using Beam tool to insert beams onto the room objects

Let me shortly introduce you to the topic of beams. Beams are thin lines that occupy one line of game fields and that are able to kill any critter except the turtle. To manipulate beams, you need to select Beam tool from the Editing tools menu.
To *add beam* to some part of the game field (four beams can be placed to a single game field occupied by some object), use LMB to click close to the side of the game field where you want the beam to be placed.
Removing of beams is done similarly; just instead of LMB you will use RMB.
You can alter the direction where the beam points. This is done by selecting a beam emitter by clicking on it and changing the Direction in beam properties.
Beams change their length automatically as you move objects, alter their shape or change some parameters of the room.

Using Area tool to define goal-related or graphics-related areas in the room

The next tool you are going to meet is used for defining free-shaped areas bordered by the connected lines. These areas can be used to define *goal area* of the room or some specific areas to generate graphics into. Area tool can be activated from the Editing tools menu.
To *create new area*, click on Create new button in the window that appeared. New unnamed selection will be created and you can add points to it. Use LMB to place points into the room. These points will be connected by a line so that point will form enclosed area of space. You can finish adding points to the new area either by clicking again on the first point or by clicking RMB. Both these actions will close unfinished area.
You can *add new points* to the existing area by selecting it from the main list of areas and using RMB to click on the lines.
Moving point forming some existing area can is done by selecting the area from the main list of areas and dragging and dropping the points using LMB.
Of course, you can also *remove points* from the area borders. This can be done by selecting the area from the main list of areas and clicking on points using RMB.
You can also *move whole area* by dragging and dropping it using LMB.
To remove an area, select it from the main list of areas first and then click the Remove button.
You need to distinguish between multiple areas so you can *change name of the area* without any problems too. Just select the area to rename from the main list of areas and type a new name in a text box below then press Enter.
To visualize areas, you can use Show border or Show fill check boxes at the bottom of the window. This visualization is available only in the editor; while in the game areas are not visualized.
There is one exception to this rule however. You can define special *goal area* by creating an area and naming it “goal”. This area will be visualized by rotating stars during the game. It has also special meaning when you select “Get items on place” as a Goal from room properties.

Using Graphics tool to assign graphical representation to the object

So far you have enough information to create every game mechanism that this game supports. Now, you will learn how to make the game look nicer so that player will see some beautiful graphics instead of the dummy squares. To assign some graphics to the objects, start the Graphics tool from

the Editing tools menu.
You can use this tool in two different ways. You can add an appearance to the object automatically using generation scripts or the database, or you can add the appearance manually.
We will start with the first way (to use the *database* or prepared *generation scripts)* because it is easier to use.
By pressing the Database button, you will open a new window containing all the available object's graphical representations created to that moment and marked as shared, that fit by their shape to the object that is currently selected. By clicking on any of these objects, you will assign that appearance to current object. Database window can be closed by pressing the OK button. You have also possibility to *rebuild whole database* by pressing Rebuild button. This will rebuild the database, taking object's appearances from all the shared objects located in the ROOMS folder. Resulting database file is stored in ROOMS/_GR_REPRES_DB.TXT. Before rebuilding the database, be sure to backup this file!
Another way how to set the appearance of an object is to use generation scripts. Press Generated button to open window containing names of some prepared generation scripts. Select the name of the script and apply script by pressing the OK button.
Now we will go through manual creation of the graphics. To understand this, you should be more familiar with the concept of assigning graphics to objects, so let me explain some basics. You can specify static bitmap graphics to the objects using *sprites*. Sprites can be added using the Add sprite button. Static graphics is suitable only for some objects, so there is another possibility - to assign dynamic graphics using *sprite sets*. This is done by the Add sprite set button. If you want an object to be created from more static, dynamic or even generated parts, *container* is what you are looking for. You can add it using the Add container button. Containers can contain more sprites, sprite sets or even other containers. To remove any of these, use the Remove button. To change a structure in that individual sprites, sprite sets and containers are arranged in order to form single graphical representation, you can used To container (creates new container, places it on the place of the currently selected part and moves the part into the new container), Up (moves currently selected part up in the container) and Down (moves currently selected part down in the container) buttons.
Containers are special not only because they can contain another elements, but also because they can contain *construction commands*. These construction commands are script commands that can instruct game engine to create generated graphics exactly as you want. To edit construction commands, select a container and click on the Construction commands tab. Short edit box allows you to write construction commands. To help you with editing, we have added some help tools available through the RMB. You can for example add some prepared script name to the code using Add script name command under RMB menu. You can also add sprite or sprite set name, so you do not need to enter it manually. Whenever you want to open a script file or an image file, just move cursor on the file name, click with LMB and then click RMB and choose Open. For advanced placing of sprites and sprite sets on exact coordinates, *code generation tool* is prepared for you. To activate it, press the Open code gen. tool button on the Construction commands tab. This will open the tool and if there is already some previously generated code with it, it will be used to initialise the tool so that you will be able to change the code in a comfortable way. Upper part of the tool contains file browser. You can see all the files that can be used there (if you want to insert a sprite set, select its first picture and after the picture is inserted, check is sprite set check box). If some pictures are too small or too big, + and - labelled buttons in the upper right corner of the tool can make pictures bigger or smaller. So when you find a picture that is suitable for you, click on it to select it. Now, you can click anywhere into the main editor window to add the picture to its place. The picture's position can be altered by drag and drop using LMB in the main editor window (pictures are snapping on the game field's grid, this can be turned on or off using Snap to grid check box). To change picture's size, enter it into the text fields above the Change size button and then press the button (size, as well as position that is written in the square brackets next to the picture

name in the lower part of the tool, is in the percent of game field size). To remove inserted picture form the tool, use the Remove button. You can change a z-ordering of the pictures by Up and Down buttons. The check box Use sorted list instructs the tool to generate a code that will use special inserting method that is useful in combination with some generation scripts, for example scrap yard. To disable the visualization of pictures to be inserted in the main window, use Draw inserted elements check box. After all the sprites and sprite sets are exactly where you want them, you can let the tool generate code that represents whole layout of the elements. Click into the text box on Construction commands tab on the place where you want to insert the generated code and press Paste form code gen. tool button. To apply all the changes in construction commands, do not forget to press the Use button.

All the graphical representations you create can be marked as *shared* using the Shared check box. Shared representations will be included into the database during its rebuild.

An appearance of the object is not defined only by the sprites and sprite sets it consists of, but part of it is also the colour of the subtitles used when the object speaks. You can specify this colour using Subtitle colour.

Whenever you want to display dummy graphics instead of normal one, just select the Dummy graphics check box from the room properties or on the main tool bar.

Using Background tool to define room background

When you look at your room now, it is almost perfect. Interesting room design, nice look of the objects, but there is still something missing. Yes, it is the background. And this is the moment where the Background tool from the Editing tools menu comes into play.

The background of the room is defined as a set of rectangles of one of three different types. We will look at these types further below. To *create new rectangle*, press the Create button. This new rectangle will appear in the top left corner of the main window. To *move the rectangle*, just drag and drop it using LMB. As you see, this rectangle has second smaller rectangle in its lower right corner. To *change the size of the rectangle*, drag and drop this lower right corner. Z-order of the rectangles can be changed using the Up and Down buttons. Rectangles you don't need any more can be deleted using the Delete button. If you want to move or resize the rectangle smoothly, disable Snap to grid check box. If you want to turn off the visualization of the rectangles, disable Draw rectangles check box.

Now for rectangle types. You can choose from thee different types – *normal rectangles*, *glow rectangles* and *moving rectangles* using the Type drop-down box.

Normal rectangles are static pictures that can be masked, coloured, small cuts can be taken from them and ripple effect can be applied to them. Mask effect allows you to create various strange effects in combination with glow rectangles. Mask has to be of the same size as the original image and its red channel is used to mask the original image (it is cut out where red colour is present). When normal rectangle is masked, it is drawn twice. First time it is drawn as the lowest backgrounds without the mask, than all glow rectangles are drawn and at the end, normal rectangle is drawn with the use of the mask. So the mask defines where the glow rectangles should be visible. Normal rectangles can also be coloured. This simply means that the image will be drawn using given colour. Not whole source image has to be used to fill a rectangle; you can use only part of the source image. Four text fields are available to insert information about what part of the source image to use. You can change both Source offset (upper left corner of the drawn part) and Source size (width and height of the drawn part). A number should be entered in percent of the size of one game field counted for the selected image (for example, if you choose image that is placed somewhere on the path containing X30 folder and the image is 90x90 pixels in size, entering offset 100x100 and size 100x100 will choose central 30x30 pixels from the image). If you enter wrong numbers (for example coordinates for part of the image that is larger than the image itself) they will

be ignored. Another feature you can use for normal rectangles is a ripple effect that makes the rectangle look like it is distorted by the water. Be careful with using this effect, it consumes a lot of processor time so it can be problematic on slower computers when it is not turned off in the game options.
Another type of rectangle is the glow rectangle. These rectangles are used to simulate water caustics effects. You do not need to set their image when no image is set a default one is used. You can set Glow density (how often an image is repeated in the rectangle), the lower the value is the denser the glow will be (real number is expected). Another parameter you can set is Glow size (how big the image will be – its scale factor), the lower the value is the smaller the images will be (real number is expected). Next parameter is Glow speed (how fast the images will move), the lower the value, the slower the images moves (real number is expected). The last glow specific parameter is Glow flatten (how much will the images be non-proportionally scaled), the higher the value is, the narrower the images will be (integer number is expected). Glow rectangles can be coloured as well as normal rectangles.
The last type of rectangle is a moving rectangle. This rectangle type is the right if you want to create for example clouds on the sky. You can set Movement speed for them. The lower the value is, the slower the movement will be (real number is expected). Another parameter is the Direction this rectangle will move in. And finally, moving rectangles can be coloured as well as normal rectangles.

Concept of editing the room

While you are designing your own room, you need some possibility to play and test it as you are creating it. This is why the editor offers two basic modes – *edit mode* (can be turned on using Edit mode in Edit menu) and *game mode*. Game mode is further divided into *running game mode* (can be turned on using Running game mode in menu Edit->Game mode) and *paused game mode* (can be turned on using Paused game mode in menu Edit->Game mode).
Edit mode is used for creating the starting state of the room. Whenever someone starts the room from the game, it starts in this state.
Running game mode is used for testing the room. In this mode, the editor works almost like the game, so you can play the room like a normal player.
The paused game mode can be used for editing the room while testing. You should always remember that when you edit your room in this mode any changes will not appear in the starting state of the room. You can use this mode for example in the situation during the testing when you discover that a few more squares should be useful for the small fish to be able to pick up some object but you do not want to play the whole room from the beginning.
You can freely switch between these modes as you wish. Of course, when you start testing your new room, than you switch back to the edit mode, add edit shape of objects, after switching back to the game mode, changes in the starting state of the room will not project here. You can alter also the partially tested room using the paused game mode or you can restart testing and play the room from the beginning using the Restart game mode command in menu Edit.

Changing parameters of the room

You can change the basic parameters of the room you entered at the time of creating the room as well as some advanced parameters using the Room parameters command from the Room menu. Upper part of the window that appears contains *water level* parameter of the room. You can enable it using the Water level is enabled check box. Position of the water level can be altered using Up and Down buttons.
In the middle part of the window, room size can be altered. By pressing + and - buttons, you can

add or remove game fields from all the sides of the room.
The lower part of the window contains two text fields, so that the game field size can be changed. For further details see part Basic room file manipulation.

Folder structure and file naming conventions

If you wish to create some custom contents using the editor, you should know where to place your files so that whole editor will work properly.
Basic decision you have to do is whether you want to create a single *mission* or whole *campaign*. If you wish to create just one room without any context, mission is the right choice for you. If you wish to create multiple rooms connected together, usually with some non-trivial story, the campaign is the right choice for you.
Every single mission should be stored in separate sub-folder (stored in CUSTOM/MISSIONS folder) named by the mission name. Let's call this mission sub-folder. File containing room information should be placed here and named ROOM.TXT. Dialog connected to the room should be placed in file named DIALOG.TXT and scripts should be in SCRIPT.LUA. Of course, all the other files that this single mission uses (graphics, sounds, music ...) should be also placed in this directory. For example, when you want to create custom mission called "MyMission", its room file should be placed in CUSTOM/MISSIONS/MYMYSSION/ROOM.TXT file.
When you want to create a whole campaign, the situation is a little bit more difficult. Your campaign should, like a mission, be stored in a separate sub-folder (stored in CUSTOM/CAMPAIGNS folder). Let's call this campaign sub-folder. Here, file MAP.TXT containing description of the whole campaign and a few more folders should be present. Folder ROOMS should contain files with rooms named as described in MAP.TXT. Folder SCRIPTS should contain script files for each room, with the same name as the room file, but with extension .LUA. The same holds for TEXTS folder and dialog files, just extension is .TXT again. For example, if I want to create a campaign called "MyCampaign" with two rooms called "Room1" and "Room2", its main map file should be named CUSTOM/CAMPAIGNS/MYCAMPAIGN /MAP.TXT. The Room1's room file should be named CUSTOM/CAMPAIGNS/MYCAMPAIGN/ROOMS/ROOM1.TXT. The dialog for it should be in CUSTOM/CAMPAIGNS/MYCAMPAIGN/TEXTS/ROOM1.TXT AND a script for it should be called CUSTOM/CAMPAIGNS/MYCAMPAIGN/SCRIPTS/ROOM1.LUA. Similarly to room two.

Managing room's scripts and dialogs

If your room is saved in the proper place as described in Folder structure and file naming conventions, or if the room has already assigned some dialog and script file, you are allowed to use Open room dialog and Open room script command from the Room menu. These commands open notepad editor with the file containing room dialog or room script. After changing any of these files, do not forget to use Reload scripts and dialogs command from the same menu. This will reflect all the changes made in mentioned files.
When a room has no dialog or script file assigned and is saved on proper place, files are created automatically when Open room dialog or Open room script command is used.
Another way how to assign dialog and script files to a room is to use Dialog file name and Script file name in room properties. Click LMB in the text area to assign file name, RMB to remove file name and MMB to open assigned file.

Creating mission or campaign packet

So, now things are in such a state where all the files necessary for a mission or a campaign to run (dialog files, script files, your own custom graphics, ...) are on its place (in one sub-folder under CUSTOM folder) and you want to create single file from it that can be uploaded on the internet or that you can send to your friend. This single file is called *mission packet* or *campaign packet*. I will describe everything for the mission, but creating campaign packet is quite similar. Choose Create mission packet from the File menu and window asking for the name of the directory will appear. Browse and select the directory containing your whole mission. For example, when your mission is called MYMISSION, you will choose directory CUSTOM/MISSIONS/MYMISSION from the folder that contains the editor. This will start build operation and after a while, build result will appear on the screen. When everything is OK, new file named MISSION_MYMISSION.F2P will be created in CUSTOM/CREATEDPACKETS folder. This is the resulting packet of the build operation. If you put this file into CUSTOM/DOWNLOADEDPACKETS folder, the game will automatically unpack it during its next start up to a proper place and the packet will be moved from CUSTOM/DOWNLOADEDPACKETS folder into CUSTOM/INSTALLEDPACKETS folder.