H&D2 RevEngEd 2

User Guide

by Sebastian 'Ikaros' Rizzo
& Michele 'Mikiller' Endrici
[*http://malgolan.altervista.org*](http://malgolan.altervista.org)

Summary

Disclaimer

RevEngEd2 is a double purpose editor. First, it integrates almost all the functionality provided by DC|ED, making you able you to modify H&D2 files in likely ways. Second, it's a much more user friendly editor for new occupation missions.

This guide contains several levels of information. Some paragraphs are intended for expert modders only and can be skipped by a casual user who just want to use RevEngEd 2. Therefore, **advanced information is written in blue**.

Also, some images could be out of date, slightly changed after the release of this document, but you should be able to identify the information they contain with ease.

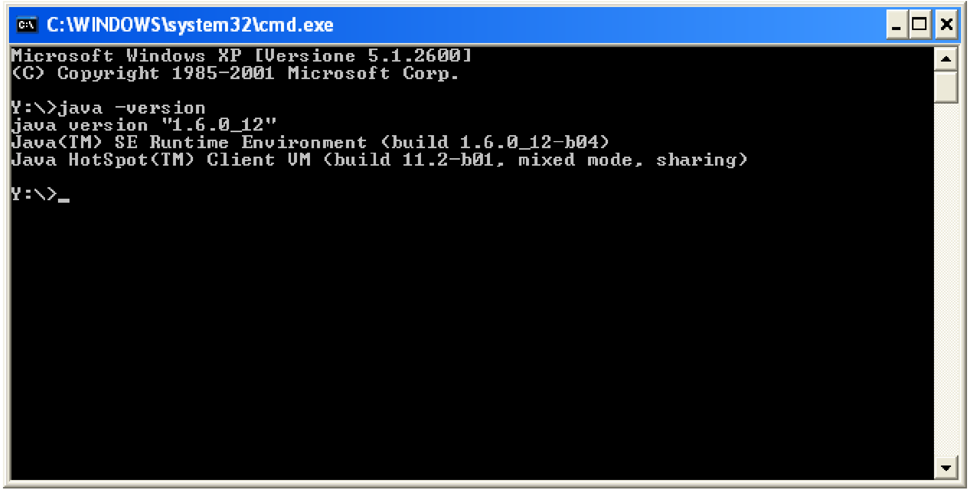
Glossary

- **Archive**: a set of directories extracted from .DTA files of hidden, which you should already have. If you're not familiar with this concept, please see http://malgolan.altervista.org/Guide/index.php?page=EB15-extractor.php.
- **Block type**: the number associated to objects (specification and definition) inside .bin files. Specification types are (for example) 6 = dummies, 9 = models, 11 = volumes. Definitions types are (for example) 3 = soldiers, 4 = vehicles, 37 = zones.
- **Definition**: inside .bin files, the object specific information stored inside the second part of the file (the block starting with 0x21 0xAE for those who used to edit .bin files with hex editors)
- **G-Map** (or ***game map***): the three-dimensional terrain where you can walk around while in play.
- **H&D main folder**: the directory where your Hidden & Dangerous 2 is installed. Usually, "C:\ C:\Program Files\Illusion Softworks\Hidden & Dangerous 2\".
- **Library**: the folder inside RevEngEd2 that contains a list of binary fragments, along with some icons.
- **M-Map**: the aerial map you can see by holding M key while in play, usually in a drawn style, showing your position, location of flags, etc.
- **Maps archive**: the folder where you've extracted all .BMP, .TGA, etc. from MAPS.DTA
- **Mission archive**: the folder where you've extracted all mission sub-folder from MISSION.DTA or SABRE.DTA. Each of them contains several files, like actors.bin, scene2.bin, scene.4ds, map.4ds, etc.
- **Mission**: a playable multiplayer map, be it occupation, objectives, co-op or deathmatch.
- **Mission folder**: the directory representing a map, containing several files, like actors.bin, scene2.bin, scene.4ds, map.4ds, etc.
- **Occupation mission**: a mission where two human teams fight for control of several *zones*, identified by *flags*.
- **Specification**: inside .bin files, the object information stored in the first part of file, such position, rotation, model, sector, etc. (the block starting with 0x10 0x40 for those who used to edit .bin files with hex editors).

Requirements

- **Java JRE 1.6 or above**.

You can check your version of java by opening a *cmd* console and typing *java –version*:



```
Microsoft Windows XP [Versione 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

Y:\>java -version
java version "1.6.0_12"
Java(TM) SE Runtime Environment (build 1.6.0_12-b04)
Java HotSpot(TM) Client VM (build 11.2-b01, mixed mode, sharing)

Y:\>_
```

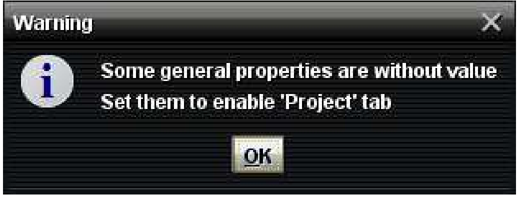
Configuration

Download “H&D2 RevEngEd 2.rar” from *http://malgolan.altervista.org* and un-rar it wherever you want.

Copy the new MODELS and MAPS provided (you find them inside the */New Components/* folder) into your H&D2 main folder.

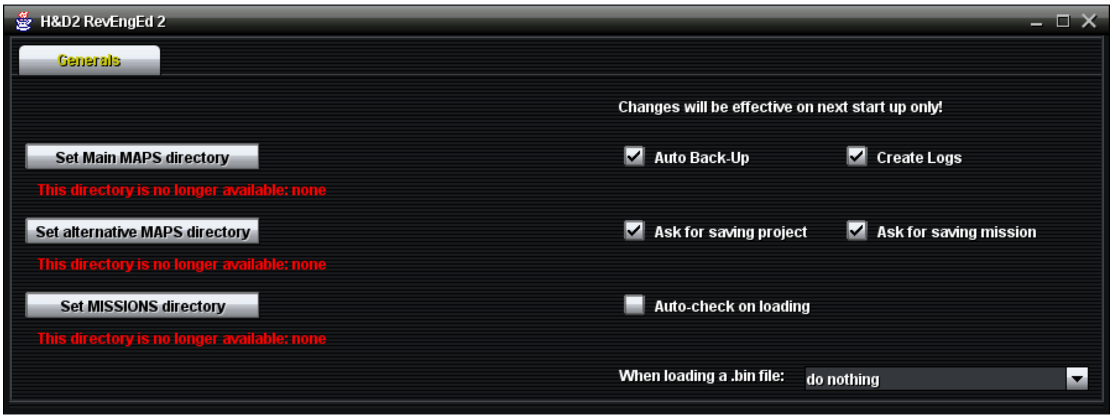
Make sure that “Start” link points to “*your installation folder*\HeD2-Revenged2\dist\HeD2-Revenged2.jar”. If not so, please set it properly and launch “Start”.

At first start, you’ll be prompted a warning messages[1]:



Some general properties are without value
Set them to enable 'Project' tab

Press ok and let’s have a look a the next frame:



Changes will be effective on next start up only!

This directory is no longer available: none

This directory is no longer available: none

This directory is no longer available: none

☑ Auto Back-Up ☑ Create Logs

☑ Ask for saving project ☑ Ask for saving mission

☐ Auto-check on loading

When loading a .bin file: do nothing

This is the *Generals* tab and red values have to be set before continue. Click on relative buttons to chose a folder for each of them.

- ***Main MAPS directory***: this is where .BMP are retrieved, when loaded as textures. Usually, you’ll set this folder as the “MAPS” folder inside the H&D2 main directory. For example: “C:\Program Files\Illusion Softworks\Hidden & Dangerous 2\MAPS”.
- ***Alternative MAPS directory***: if a .BMP is not found in main MAPS directory, is then searched within this one. Usually, is set to the MAPS archive extracted from MAPS.DTA[2].
- ***MISSION directory***: this is where all mission folders loaded by H&D are stored. Usually, is set to the “MISSIONS” folder inside the H&D2 main directory. For example: “C:\Program Files\Illusion Softworks\Hidden & Dangerous 2\MISSIONS”.

When these three directories are properly set, a new tab appears, called “Project”. You have to set these folders only the first time you start RevEngEd2.

---

[1] If not so, you could miss the java requirement or .jar files could be not associated with java.exe on your machine.

[2] If you’re not familiar with this concept, please have a look at http://malgolan.altervista.org/Guide/index.php?page=EB15-extractor.php.

Before going on to *Projects* chapter, go back to “Generals” tab and have a look at the other values. These are customizable options, and modifying them will take effect ***only on the next start-up*** of RevEngEd2. They are:

- ***Auto Back-up***: if set, whenever RevEngEd 2 is started, a new back up is made inside the “*mission*\Back-ups” folder. A folder is created, called “*year month day – hour minute*”, and *actors.bin*, *scene2.bin* and *map.4ds* (the only files RevEngEd 2 modifies) are copied into it.
- ***Create Logs***: if set, whenever RevEngEd 2 is started, some plain-text files (*actors.bin.txt*, *scene2.bin.txt* and *map.4ds.txt*) are created inside the mission folder. You can open those files with a text editor, to see the content of relative files, displayed in a user-friendlier way than binary files.
- ***Ask for saving project***: if set, whenever RevEngEd 2 is closed, you’re asked if you want to save the project (see *Projects* chapter).
- ***Ask for saving mission***: if set, whenever RevEngEd 2 is closed, you’re asked if you want to save current mission files (*actors.bin*, *scene2.bin* and *map.4ds*)
- ***Auto-check on loading***: if set, whenever a mission is loaded, *occupation* checks are automatically launched (see *Occupation Checks* chapter)
- ***When loading a .bin file***: if set to “order blocks alphabetically”, whenever a mission is loaded, its specification and definitions blocks (objects) are ordered according to a few criteria:
    - First, by block type.
    - Then, *zone* have precedence over *respawns*.
    - Then, alphabetically.

Projects

To any modified mission is associated a *project*. *Projects* are files handled by RevEngEd 2 and used to store information that can’t be contained inside mission files. This information is provided by the user only when working on a occupation mission (see *Main Tab* chapter for further information).

Yet, to work on any kind of mission, you’ve to create a proper project first.

Set up

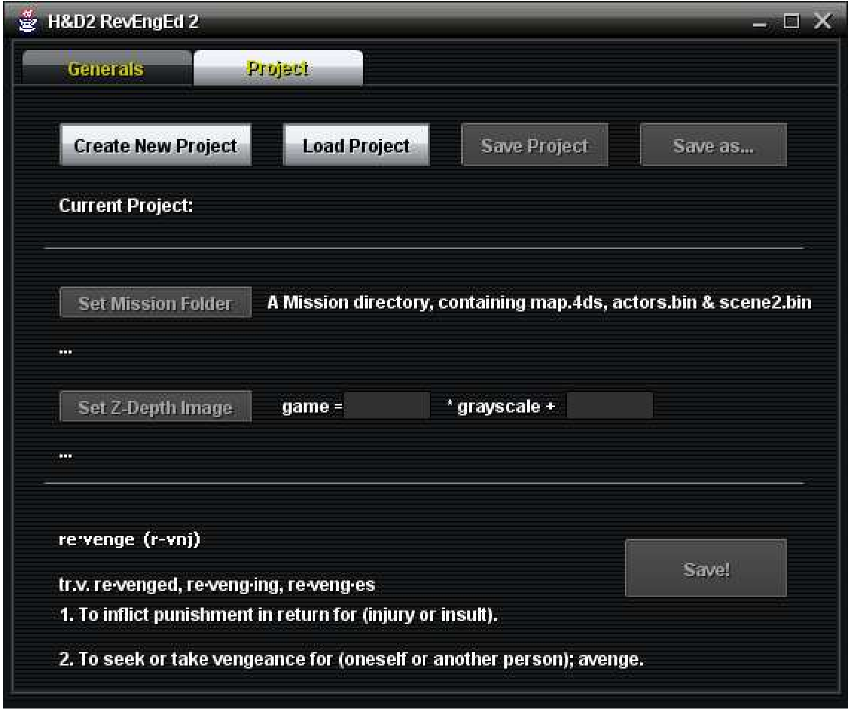
Choose a mission from your archive and copy it inside the MISSION folder of the H&D2 main directory. You can also create sub-folders inside the MISSION folder. For this example, I'll take *co_libye1* (thanks to *hdmaster* for extracting data from *Sabre Squadron* .DTAs) from my archive and copy its content inside "*H&D2 main folder*\MISSIONS\MIKA\LIBYE_A\".

Then, overwrite *map.4ds* and *loader.4ds* with the ones provided along with RevEngEd 2. This is a crucial operation, but only if you plan to use the advanced functionality of this editor (Main Tab chapter). If you want to use the DC|ED-like functionality and nothing more, you can leave whatever version of files are inside the mission folder.

You may also want to retrieve the M-Map picture and copy it to the main MAPS folder. If the picture is made by 2 or more pieces, you have to create a new single .BMP image from them.

Creating a new project

Go to *Projects* tab and have a look:



A Mission directory, containing map.4ds, actors.bin & scene2.bin

re·venge (r-vnj)

tr.v. re·venged, re·veng·ing, re·veng·es

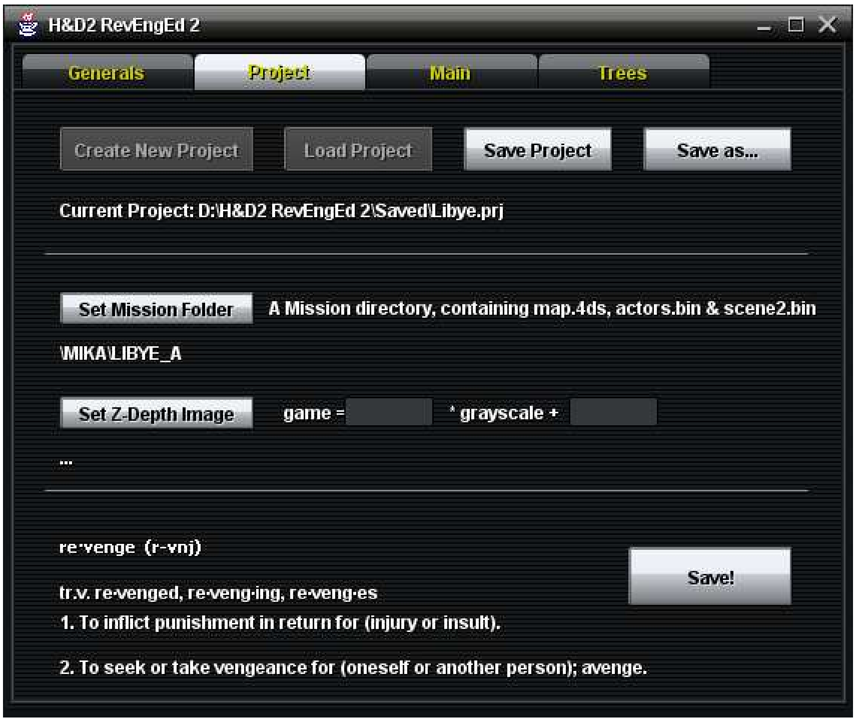
1. To inflict punishment in return for (injury or insult).

2. To seek or take vengeance for (oneself or another person); avenge.

Many buttons are disables, except for *create new* and *load project* ones. And since you have never created one by now, hit *create new* and enter a name of your choice. The first time you'll save the project, a file called *your_project.prj* will be created inside the *\Saved* folder of the editor. You'll be able to load it whenever you want by the *load project* button.

Now, you have to associate a mission folder to this project. To do this, hit the *Set Mission Folder* and select the previously created LIBYA_A directory. Note that the directory thus selected must be placed inside the MISSION folder set in *Generals* tab.

Say *yes* to the pop-up (which basically asks if you want to reload all mission files). Two more tabs will appear and buttons will now open up:



Current Project: D:\H&D2 RevEngEd 2\Saved\Libye.prj

A Mission directory, containing map.4ds, actors.bin & scene2.bin

\MIKA\LIBYE_A

game = * grayscale +

...

re·venge (r-vnj)

tr.v. re·venged, re·veng·ing, re·veng·es
1. To inflict punishment in return for (injury or insult).

2. To seek or take vengeance for (oneself or another person); avenge.

You can't, from now on, create or load other projects. To do so, you'll have to restart RevEngEd 2. Instead, you can do several other things:

- ***Save Project***: will save current projects parameters to project (*.prj*) file.
- ***Save as***…: will save current project to a different named project file.
- ***Save!***: will save to file all modifies done to current mission (*actors.bin*, *scene2.bin* and *map.4ds*)
- ***Set Z-Depth Image***: see *Z-Depth* chapter.
- ***Main Tab***: see *Main Tab* chapter.
- ***Trees Tab***: see *Trees Tab* chapter.

Z-Depth

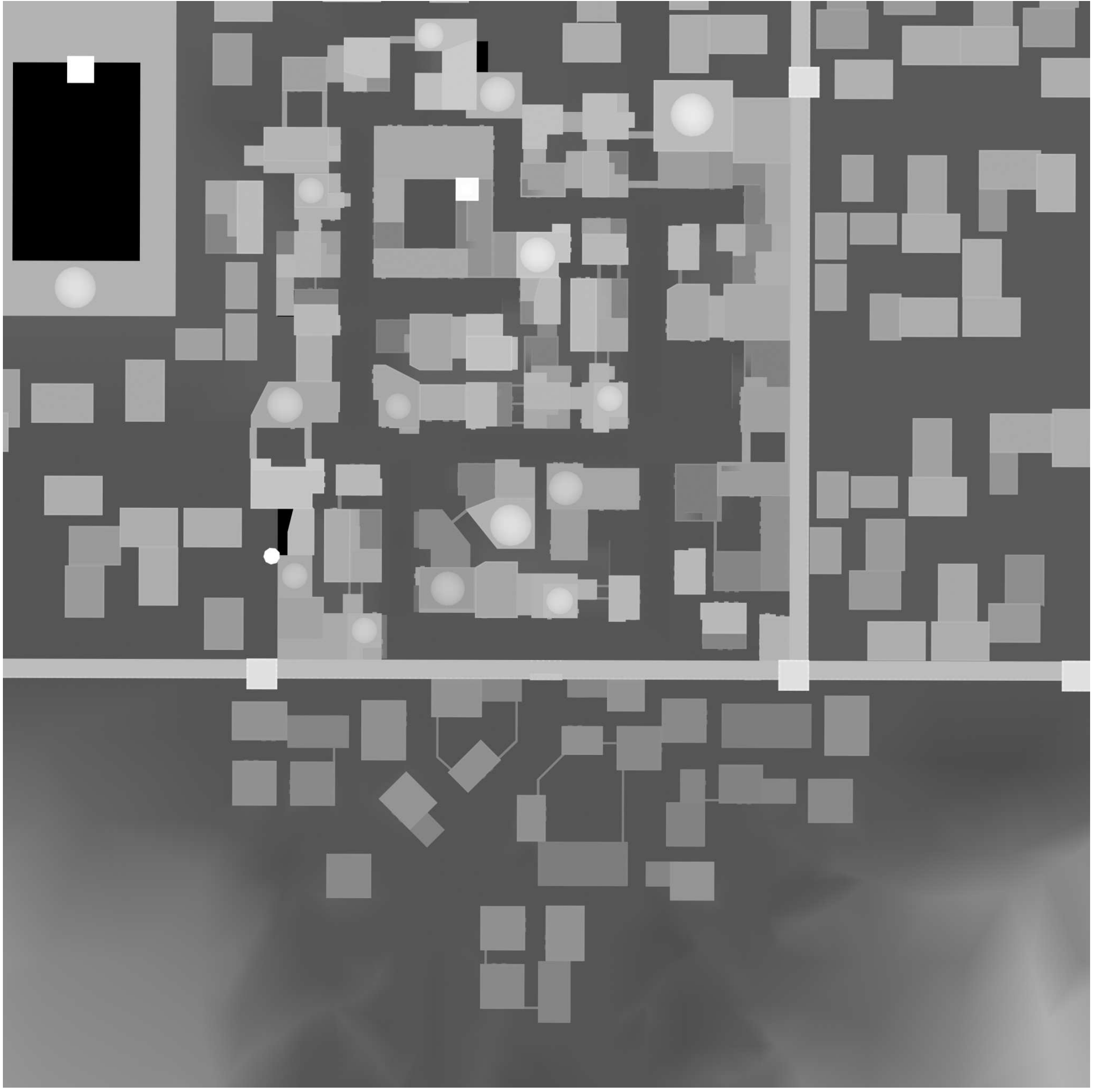
This functionality is one of the most complex, and needs 3ds max to work. It's not a necessary step to use RevEngEd 2, but it will ease you a lot of works!

First, load the current mission *scene.4ds* into 3ds max by using hdmaster great script[3]. Create a *target camera* and place it so that it frames the needed portion of map from high above. Be sure to change its property from *perspective* to *orthogonal*.

In rendering window (Rendering > Render > Common), set width and height with the same ratio as the .BMP used as background (see *M-Map* chapter). For example, if the .BMP is 400x300, you can set rendering width and height to 1200x900.

Add a new *z-depth* rendering element (Rendering > Render > Render Elements). Set Z-min and Z-max so that they wrap the area of the scene you want to use. For example, if the camera is at a height of 100, and the scenery of this map ranges from 0 (ground) to 20 (highest building), you can set Z-min to 80 (distance between the camera and nearest building) and Z-max to 110 (10 below ground, an edge to tell ground apart from holes).

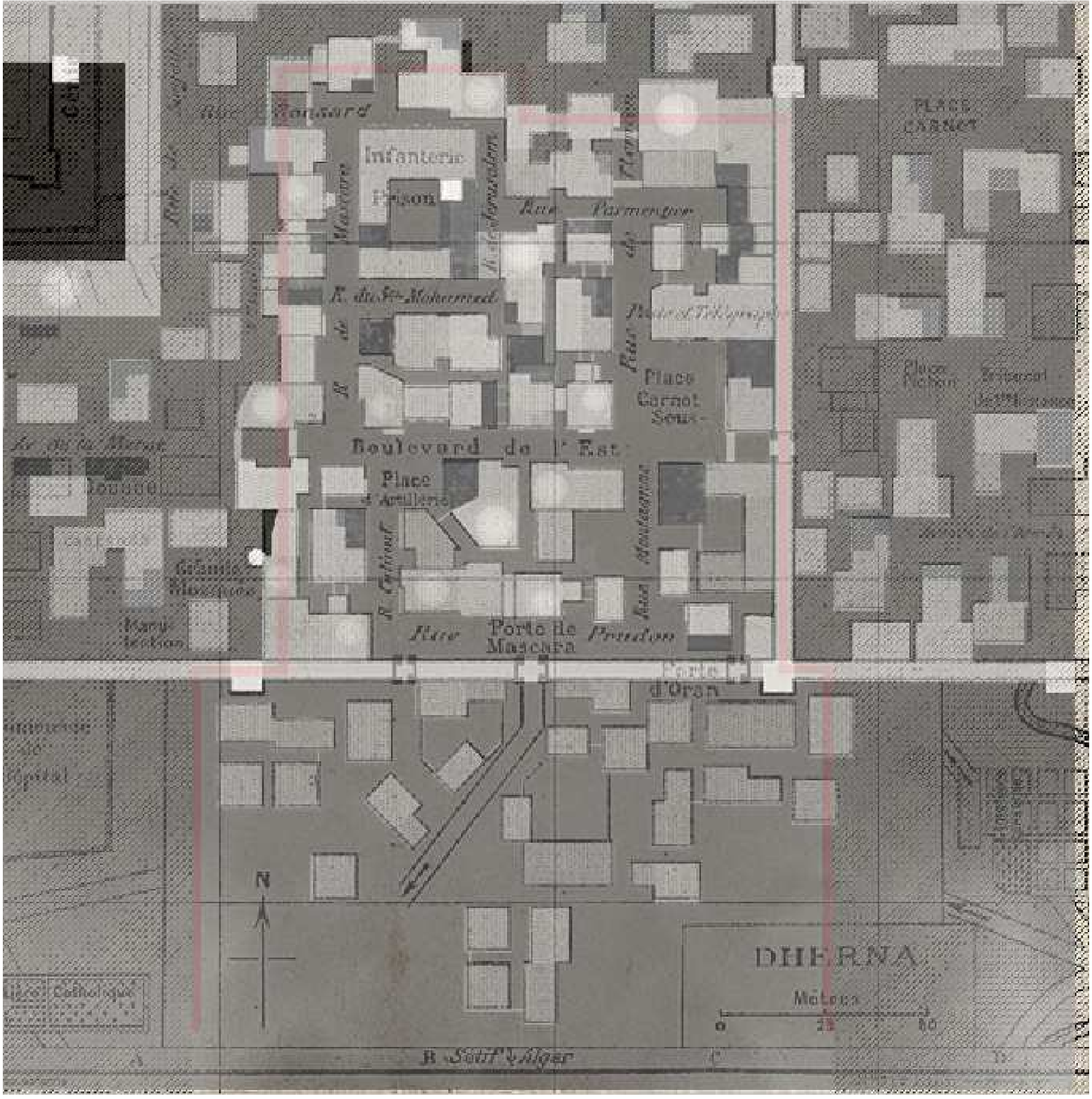
Render the scene from camera. You should obtain something like this:



---

[3] http://www.hidden-and-dangerous.net/board/index.php/topic,2003.0.html.

With a graphic software, crop and translate this image to overlap perfectly the background .BMP. Here I've removed opacity to the above layer so to make clear what task is required in this step:



Save the image as .jpg wherever you want, then click on *Set Z-Depth Image* button and load it. A file named after your project, but with extension .jpg, will be copied inside the */Saved/* directory of the editor.

Now open the *"Z-Depth calculator.xls"* provided with */New Components/*. With the graphic software, measure the grey value (0 – 255) of a few locations, and measure the height of the same location in 3ds max.

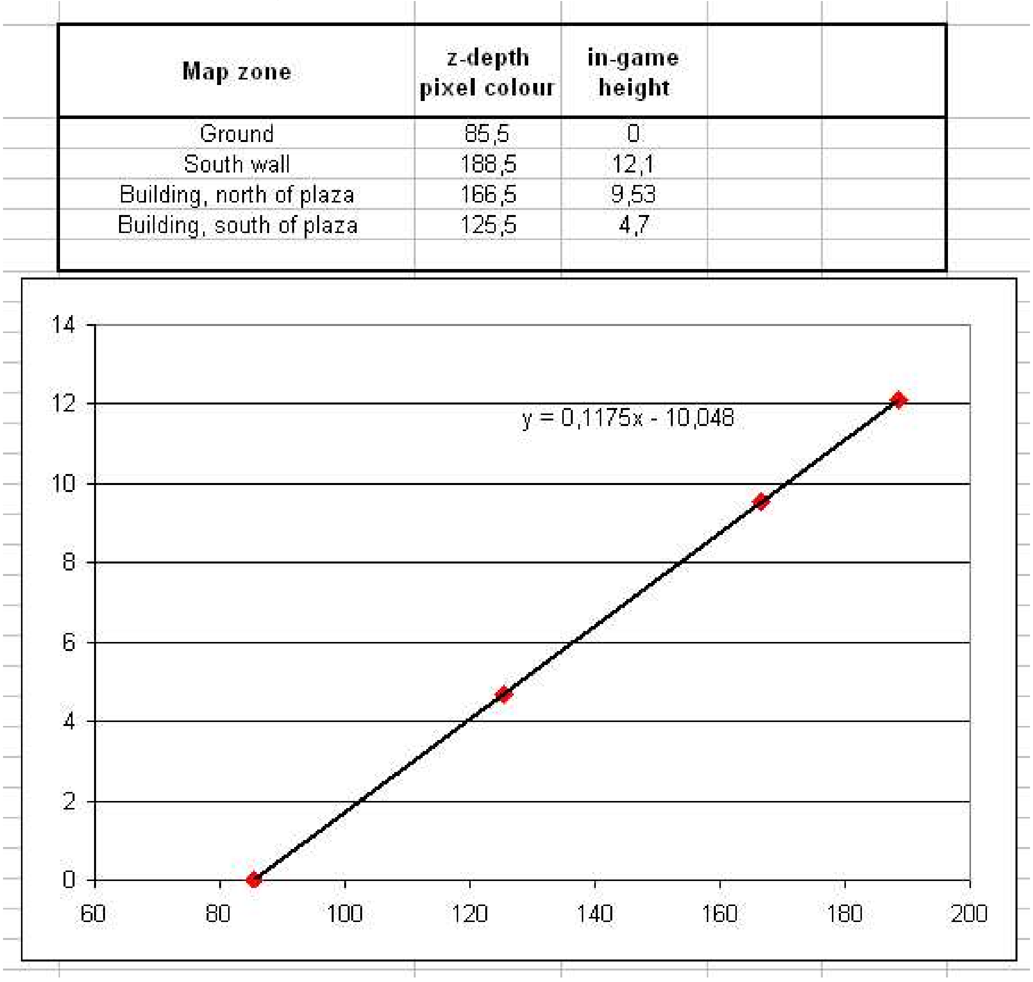
To achieve this second task, you can follow these steps (in 3ds max):

- Create a new object and call it *rayP*.
- Move *rayP* above the location you want to measure
- Execute the following max script:

```
select $*
plant = ray [$rayP.pos.x, $rayP.pos.y, $rayP.pos.z][0,0,-1]
for obj in selection do (
    if(obj != $rayP)then(
        intersection = intersectray obj plant
        if intersection != undefined then (
            print intersection.pos.z
        )
    )
)
deselect $*
```

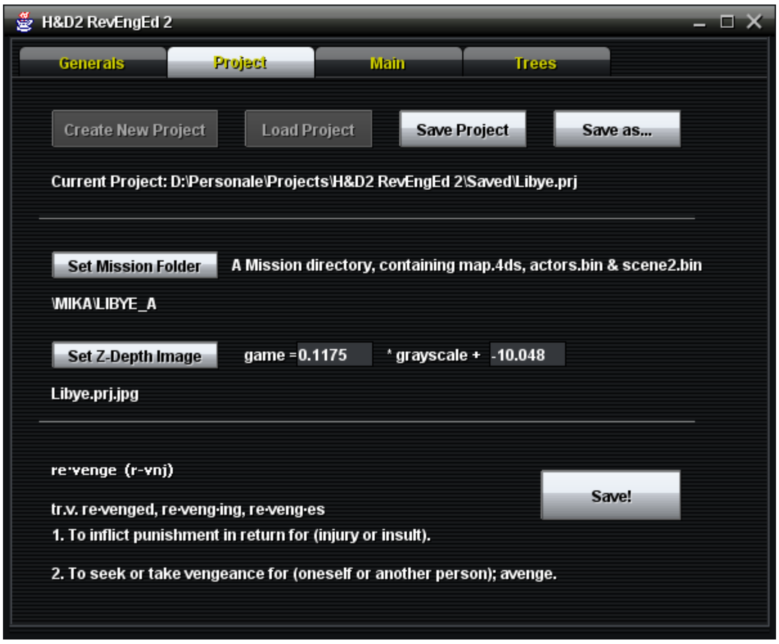
All the intersections found by a ray shot from the centre of *rayP*, directed downwards will be prompted.

Report some pair of heights (pixel colour and output of script) to the excel sheet, as I did:



Map zone	z-depth pixel colour	in-game height
Ground	85,5	0
South wall	188,5	12,1
Building, north of plaza	166,5	9,53
Building, south of plaza	125,5	4,7

If you have been precise, all red squares will be aligned. Also, the line equation will be displayed. In my example, 0.1175 is the gradient and -10.048 is the intercept. Copy these values in the *Project* tab:



Current Project: D:\Personale\Projects\H&D2 RevEngEd 2\Saved\Libye.prj

A Mission directory, containing map.4ds, actors.bin & scene2.bin

\MIKA\LIBYE_A

game =0.1175 * grayscale + -10.048

Libye.prj.jpg

re·venge (r-vnj)

tr.v. re·venged, re·veng·ing, re·veng·es

1. To inflict punishment in return for (injury or insult).

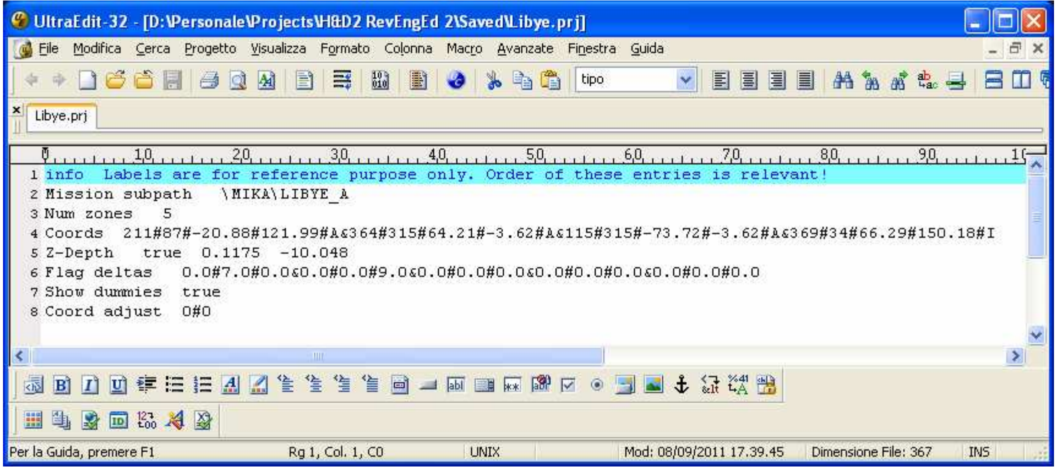
2. To seek or take vengeance for (oneself or another person); avenge.

If you’ve done all these steps accurately, you’ll see great benefits in many layers of the *Main Tab* chapter.

Project file structure

Let’s now look inside a .prj file and see what specific information is stored there and how it’s composed. It will be particularly handful when you decide to copy a part of a project to another one. By now, these concept could be obscure to you, but you can return to this chapter after you’ve achieved some familiarity with the editor.

Project *.prj* files are plain-text, and you can open them with any text editor. This is how looks my *Libye1.prj* after a little working on the mission:


UltraEdit-32 - [D:\Personale\Projects\H&D2 RevEngEd 2\Saved\Libye.prj]
File Modifica Cerca Progetto Visualizza Formato Colonna Macro Avanzate Finestra Guida
tipo
Libye.prj
Per la Guida, premere F1
Rg 1, Col. 1, C0
UNIX
Mod: 08/09/2011 17.39.45
Dimensione File: 367
INS

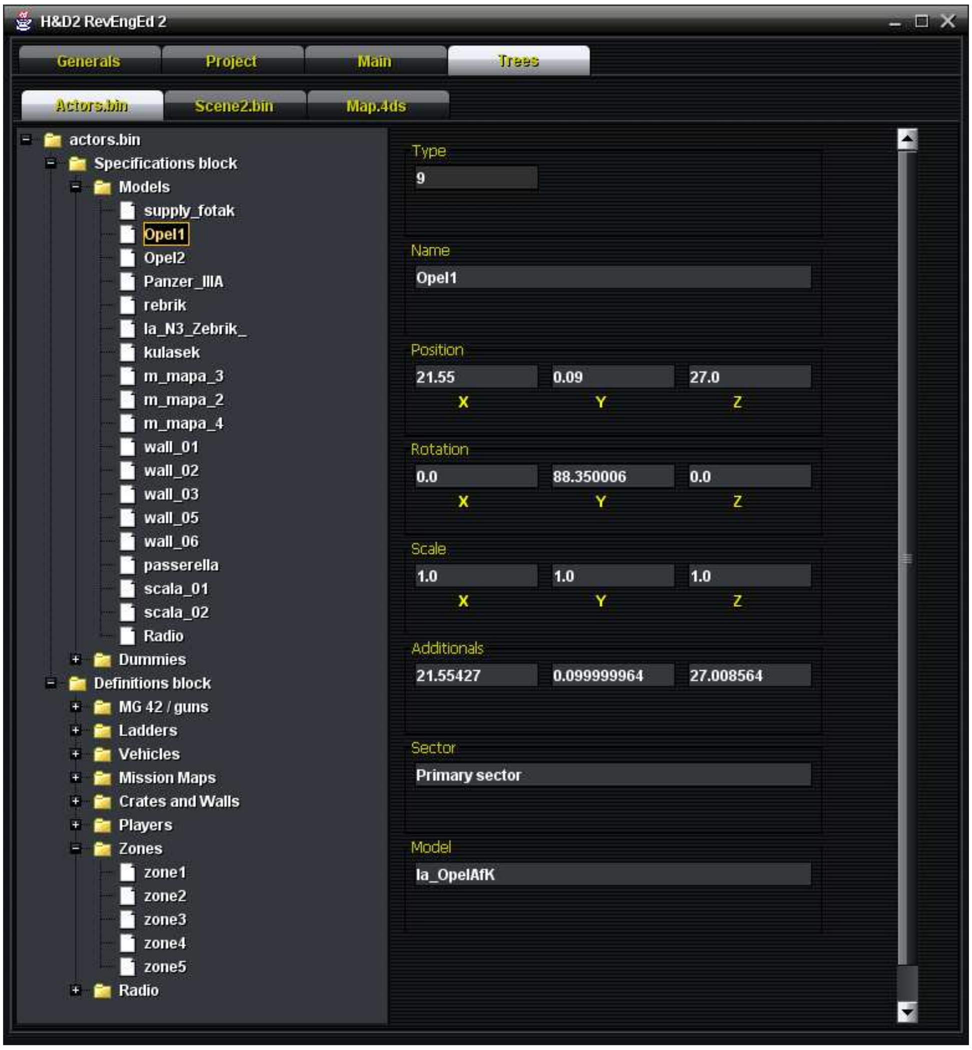
```
1 info  Labels are for reference purpose only. Order of these entries is relevant!
2 Mission subpath   \MIKA\LIBYE_A
3 Num zones   5
4 Coords  211#87#-20.88#121.99#A&364#315#64.21#-3.62#A&115#315#-73.72#-3.62#A&369#34#66.29#150.18#I
5 Z-Depth   true  0.1175  -10.048
6 Flag deltas   0.0#7.0#0.0&0.0#0.0#9.0&0.0#0.0#0.0&0.0#0.0#0.0&0.0#0.0#0.0
7 Show dummies  true
8 Coord adjust  0#0
```

First line is just a header, and you can ignore it. Following lines contains various information, and their order is significant:

- **Line 2**: the mission sub-path, relative to the MISSION folder set in *Generals* tab, which contains *actors.bin*, *scene2.bin* and *map.4ds* (along with all others files).
- **Line 3**: the last saved number of *zones* automatically created by the *Occupation Checks* Layer.
- **Line 4**: it’s a list of pairs of pairs, which links several game coordinates with pixel coordinates. Each pair is separated by an ‘&’ and it’s built like this: pixel coordinate X # pixel coordinate Y # game coordinate X # game coordinate Z # A (for active) or I (for inactive). These coordinate are generated by the *Coordinates* layer and are used to display three-dimensional objects over a bi-dimensional image (map) in proper position.
- **Line 5**: *true* tells that a *z-depth* map has been loaded, and is followed by two numbers representing the *gradient* and *intercept* of the equation used to convert pixel color into game coordinates height.
- **Line 6**: it’s a list of triplets, each separated by a ‘&’. Each triplet contains 3 values, which represents the shift (along x, y and z axis) of the flagpole relative to the correspondent *zone*. It’s used to allow a flagpole to be moved away from the very position of a *zone*. In this example, the flagpole for *zone1* is 7.0 higher than the *zone* itself, while second flagpole is 9.0 away along z axis.
- **Line 7**: if true, dummy models of soldiers are displayed in place of respawns, to help the user to check their positions (otherwise invisible).
- **Line 8**: the user can adjusts (pixel by pixel) the coordinates system calculated by *Coordinates* layer. User variations are stored inside these two values.

Trees Tab

DC|ED modders should be familiar with this tab:



Type: 9

Name: Opel1

Position: X 21.55, Y 0.09, Z 27.0

Rotation: X 0.0, Y 88.350006, Z 0.0

Scale: X 1.0, Y 1.0, Z 1.0

Additionals: 21.55427, 0.099999964, 27.008564

Sector: Primary sector

Model: la_OpelAfK

I'm not going to explain it in detail, and it's quite un-useful to those who wants to develop new *occupation* missions. You can right-click the tree, see which options are displayed and modify the values contained in the textboxes on the left as you like. Additionally, you can type CTRL+D to delete selected object.

I want to point out just one important thing, here: this tab can be used to delete those objects present in the starting mission (i.e. the one from the archive) which may cause some malfunctioning to occupation, like wrong named zones, players or so. Identifying these objects is not an easy task and won't be covered by this guide. You can learn the proper occupation structure in the *Occupation Checks* chapter and then check and correct anomalies using the *Trees Tab*.

In addition to all DC|ED functionality, RevEngEd 2 trees can handle correctly the definitions. If you export a specification, it's correspondent definition is saved as well (to a .def file in the same directory). And when you import a specification, a definition is searched in the same directory (and automatically loaded if found). Similarly, if you delete a specification, its correspondent definition is also deleted, and so on.

Moreover, RevEngEd 2 trees handle correctly the byte counts and rotations (which DC|ED didn't).

Main Tab

This is the core of the editor and allows you to manipulate *occupation* mission in a user-friendlier way. It’s composed of several parts, named *layers*, which can be activated by icons on the right. At first, just the *Occupation Checks* is enabled, because the other ones rely on non-granted information that are checked by the first layer. The layers are:

- Occupation Checks
- M-Map
- Coordinates
- Black/Blue areas
- Zones
- Respawns
- Actors

Occupation Checks layer

This layer analyze *actors.bin*, *scene2.bin* and *map.4ds* files and try to fix them, so that they're suitable for an occupation mission. It uses also several naming conventions which are not needed by H&D2 but are essential to this editor.

Just select a number X of *zones* for your mission, then click *Start* button to launch several checks (remember to overwrite map.4ds and loader.4ds as explained in *Set Up* chapter). Manual operations requested to the user are displayed in red.

Actors.bin

- Warning about definitions with missing specification.
- Delete all *soldier* objects.
- If a *player* object does not exists, creates it.
    - Only the first time checks are done for a certain mission, all *player* objects should be deleted manually with *Tree Tab*, before doing the checks.
- Makes sure that all *zone* objects from *zone01* to *zone0X* exist and delete all *zones* from *zoneX+1* to *zone30*. Checks also that every *zoneN* has N-1 as capture order byte.
    - Check with *Tree Tab* that other *zone* objects do not exist. If you find some, delete them. For example, a zone called *Zone* (capital letter) would survive the checks and cause troubles.

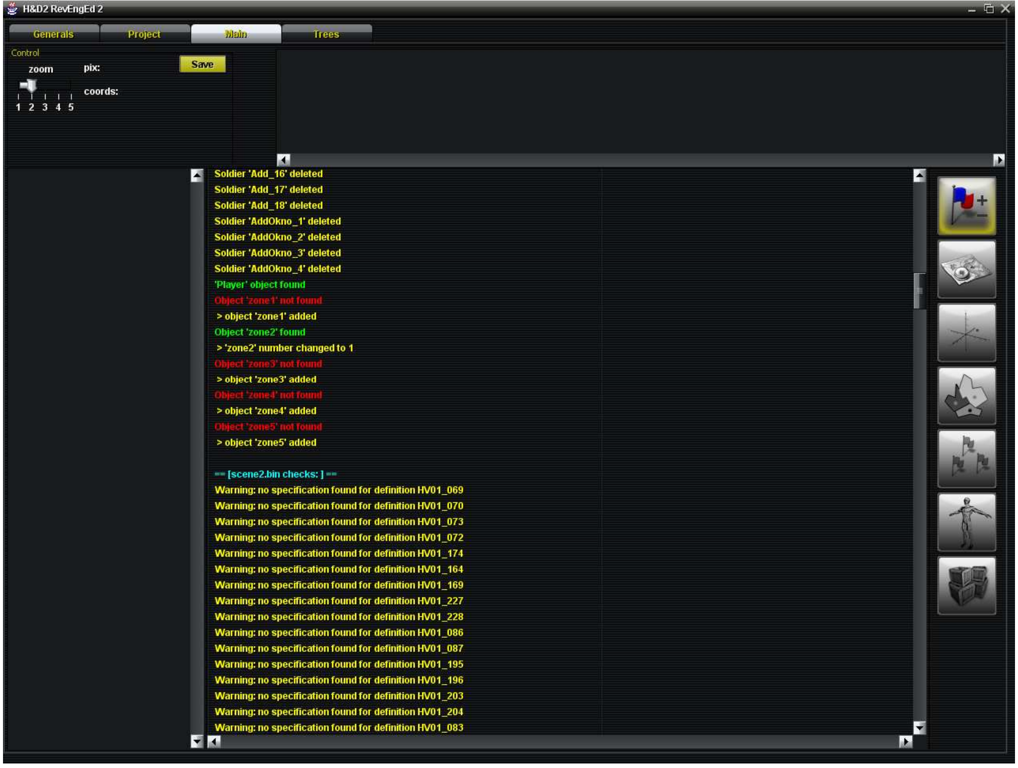
Scene2.bin

- Warning about definitions with missing specification.
- Makes sure that all models from *D_flagpole_01* to *D_flagpole_0X* exist and delete all models from *D_flagpole_X+1* to *D_flagpole_30*.
- Makes sure that all models from *d_gerflag_01* to *d_gerflag_0X* exist and delete all models from *d_gerflag_X+1* to *d_gerflag_30*.
- Makes sure that all models from *d_gbflag_01* to *d_gbflag_0X* exist and delete all models from *d_gbflag_X+1* to *d_gbflag_30*.
- Makes sure that all models from *d_neutralflag01* to *d_neutralflag0X* exist and delete all models from *d_neutralflagX+1* to *d_neutralflag30*.
- Makes sure that all dummies from *zone1spawn* to *zoneXspawn* exist and delete all models from *zoneX+1spawn* to *zone30spawn*.
- For each *zoneXspawn*, makes sure that all dummies from *z0X_01* to *z0X_16* exist and checks that they have sector equal to *zoneXspawn*. Also deleted all dummies from zX+1_01 to z30_16.
    - Check with Tree Tab that other respawn dummies do not exist. If you find some, delete them. Check also that every respawn is **after** its relative spawn zone. If you find respawns before their relative spawn zone, delete them and **re-launch checks**.
- For each valid respawn, creates a soldier model which will be used to show respawns positions in game (see *Respawns* chapter). Deletes also soldier models associated to deleted spawn zones.

Map.4ds

- Checks that a *selectzone* object exists at level zero.
- Checks that a *center* object exists at level zero.
- Checks that a *mapa* object exists as child of *center*.
- Makes sure that all objects from *zone1* to *zoneX* exist as child of *mapa*. If some are missing, it creates them, along with their three children: *alliedX*, *axisX* and *zone_disabledX*.
- Delete all objects from *zoneX+1* to *zone30* (along with their children).

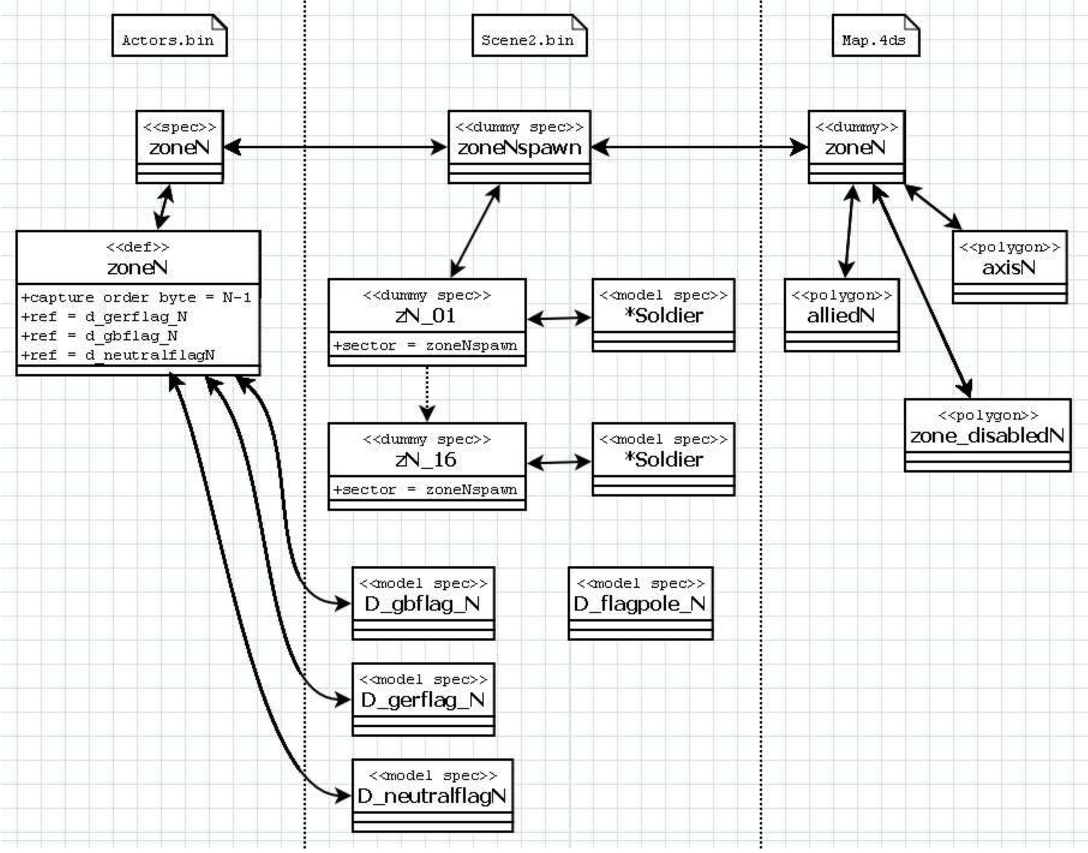
You can see a log of done operations and other layers are now active:



```
Soldier 'Add_16' deleted
Soldier 'Add_17' deleted
Soldier 'Add_18' deleted
Soldier 'AddOkno_1' deleted
Soldier 'AddOkno_2' deleted
Soldier 'AddOkno_3' deleted
Soldier 'AddOkno_4' deleted
'Player' object found
Object 'zone1' not found
 > object 'zone1' added
Object 'zone2' found
 > 'zone2' number changed to 1
Object 'zone3' not found
 > object 'zone3' added
Object 'zone4' not found
 > object 'zone4' added
Object 'zone5' not found
 > object 'zone5' added

== [scene2.bin checks: ] ==
Warning: no specification found for definition HV01_069
Warning: no specification found for definition HV01_070
Warning: no specification found for definition HV01_073
Warning: no specification found for definition HV01_072
Warning: no specification found for definition HV01_174
Warning: no specification found for definition HV01_164
Warning: no specification found for definition HV01_169
Warning: no specification found for definition HV01_227
Warning: no specification found for definition HV01_228
Warning: no specification found for definition HV01_086
Warning: no specification found for definition HV01_087
Warning: no specification found for definition HV01_195
Warning: no specification found for definition HV01_196
Warning: no specification found for definition HV01_203
Warning: no specification found for definition HV01_204
Warning: no specification found for definition HV01_083
```

For each *zone* N, these objects should exist; moreover, they have to be precisely named and they have to be in these relationships with each other:

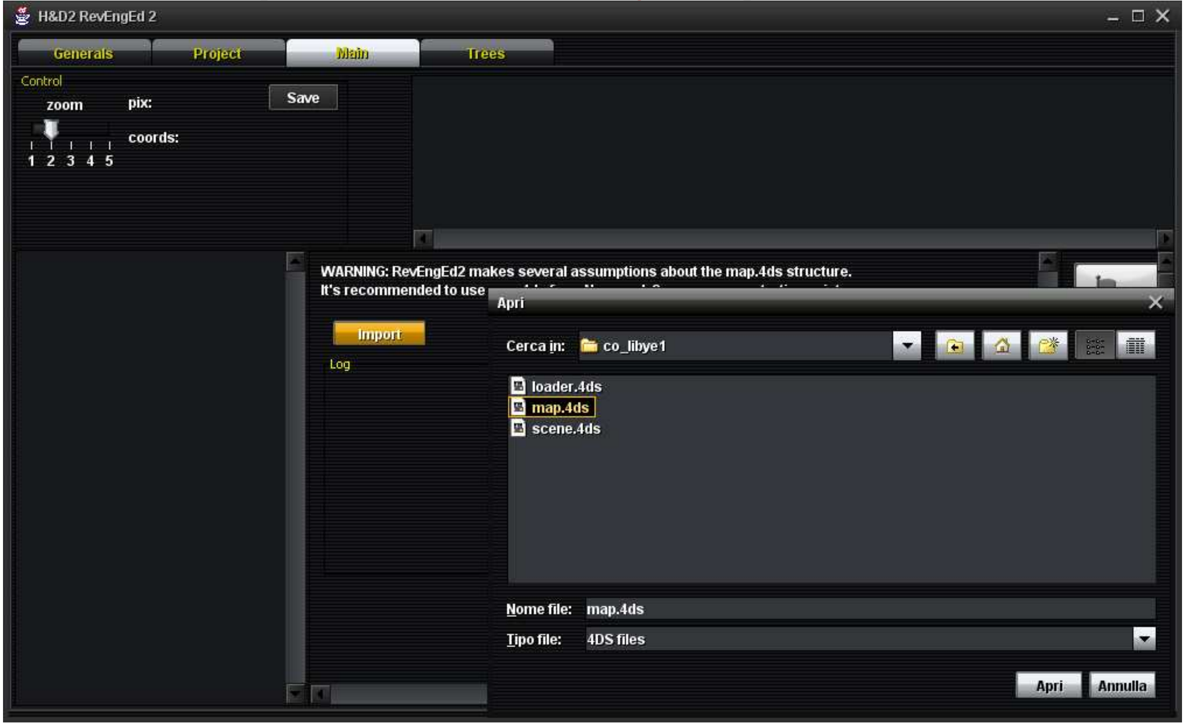


Every similar object which does not correspond to one of the *zones* should be deleted.

M-Map layer

This layer is used to adapt the m-map contained in the map.4ds provided with the editor (the one you used to overwrite the original map.4ds).

First, click on *Import* button and select the original map.4ds:



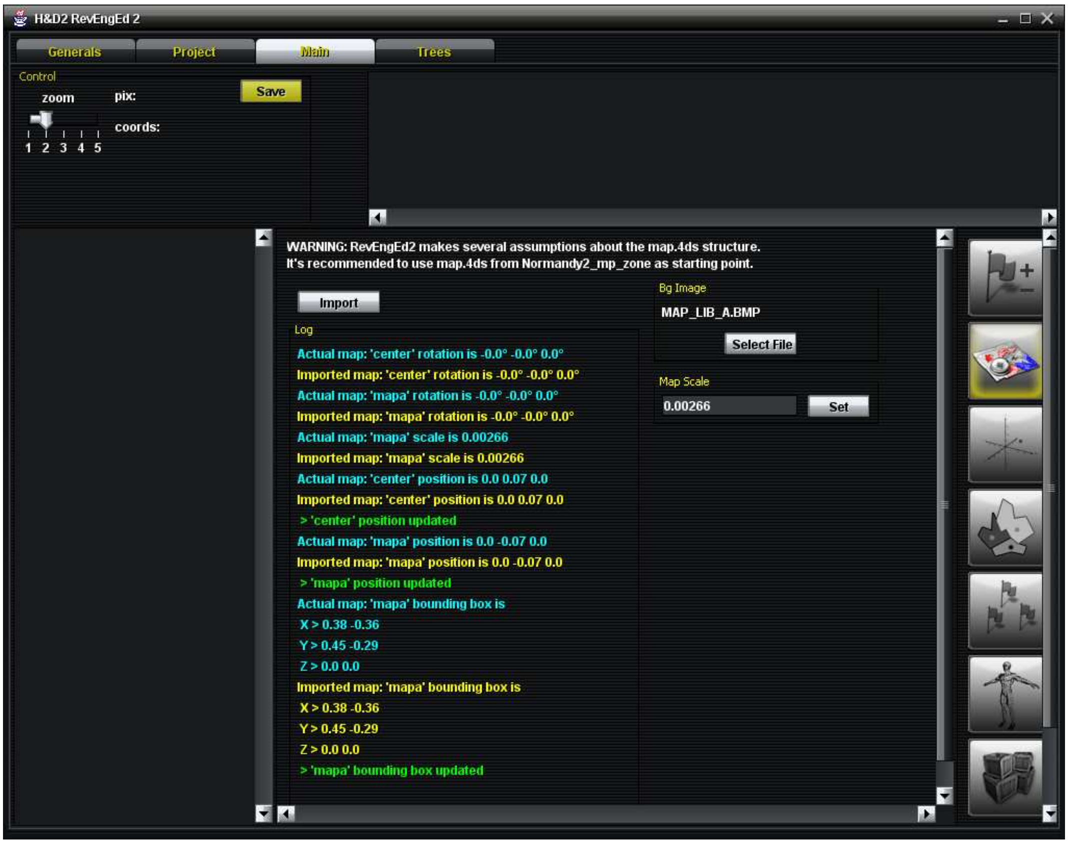
WARNING: RevEngEd2 makes several assumptions about the map.4ds structure.
It's recommended to use

Cerca in: co_libye1

loader.4ds
map.4ds
scene.4ds

Nome file: map.4ds
Tipo file: 4DS files

RevEngEd 2 will import scale, position, vertices, etc. from the original map.4ds to the map.4ds located in the current mission folder. This is necessary to synchronize the m-map with the three-dimensional in game environment. You can see a log of changes then:



WARNING: RevEngEd2 makes several assumptions about the map.4ds structure.
It's recommended to use map.4ds from Normandy2_mp_zone as starting point.

Bg Image: MAP_LIB_A.BMP

Map Scale: 0.00266

```
Actual map: 'center' rotation is -0.0° -0.0° 0.0°
Imported map: 'center' rotation is -0.0° -0.0° 0.0°
Actual map: 'mapa' rotation is -0.0° -0.0° 0.0°
Imported map: 'mapa' rotation is -0.0° -0.0° 0.0°
Actual map: 'mapa' scale is 0.00266
Imported map: 'mapa' scale is 0.00266
Actual map: 'center' position is 0.0 0.07 0.0
Imported map: 'center' position is 0.0 0.07 0.0
 > 'center' position updated
Actual map: 'mapa' position is 0.0 -0.07 0.0
Imported map: 'mapa' position is 0.0 -0.07 0.0
 > 'mapa' position updated
Actual map: 'mapa' bounding box is
 X > 0.38 -0.36
 Y > 0.45 -0.29
 Z > 0.0 0.0
Imported map: 'mapa' bounding box is
 X > 0.38 -0.36
 Y > 0.45 -0.29
 Z > 0.0 0.0
 > 'mapa' bounding box updated
```

Then, search the original .BMP used as background for the m-map, and copy it from your archive to the main MAPS folder. If the background is split into 2 or more files, reassemble them into a unique .BMP. For example, if the background is 2 side-by-side 400x600 .BMPs, you'll have to produce a single 800x600 BMP.

In this example, the background for libye1 co-op is mapa_l1.bmp. I've copied it into main MAPS folder and renamed to MAP_LIB_A.BMP. Yes, you can rename it, and it's a useful thing to do if you plan to alter the background image and want to keep the original one.

Select the new .BMP by the *Select File* button.

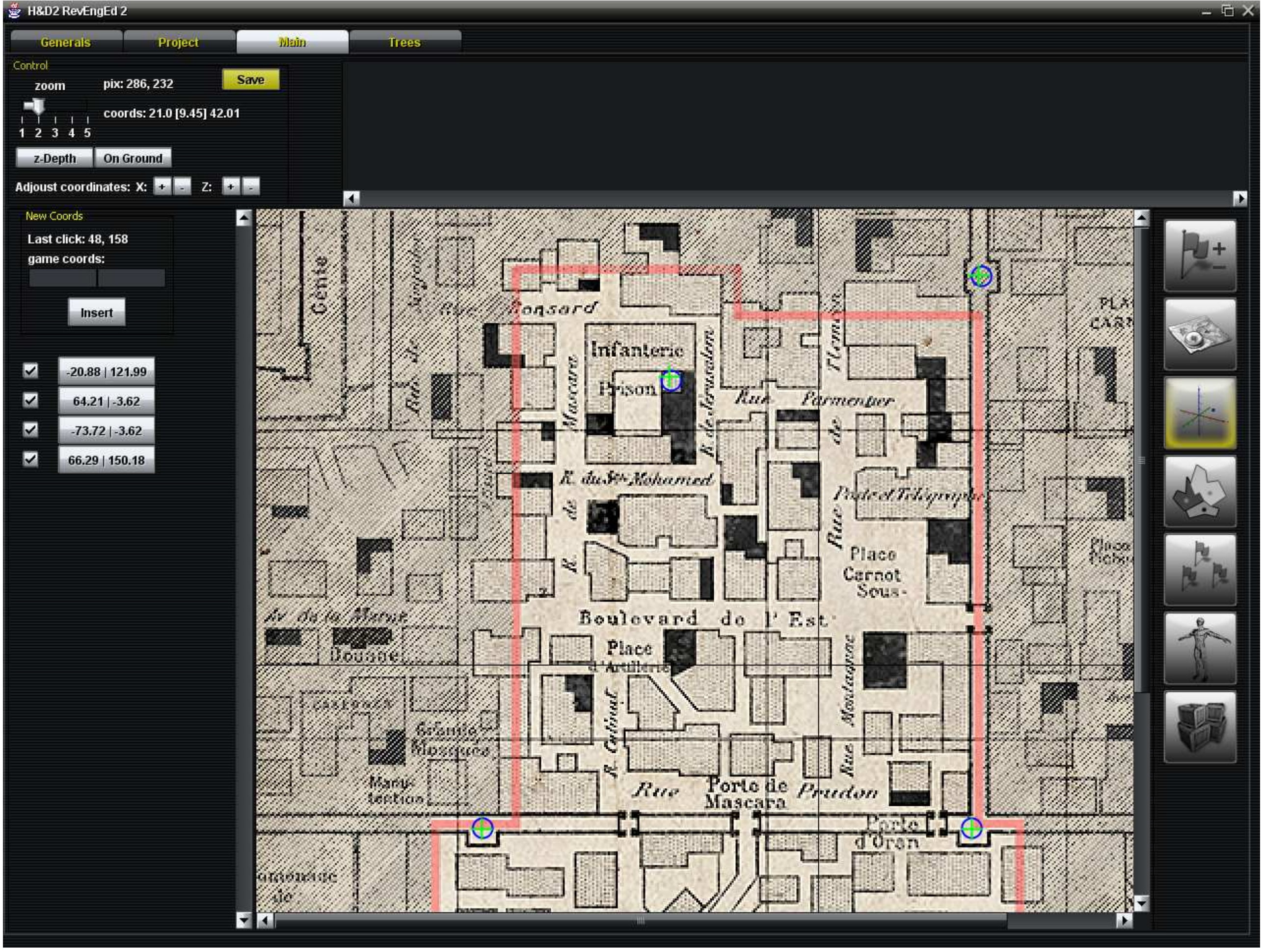
Coordinates layer

If all went well in M-Map layers, you should see a background realized by the chosen .BMP. In this layer, you have to tell RevEngEd 2 how to translate game coordinates to pixel coordinates.

Follow these steps:

- Click on the map: *last click* on the left will be set to the location of your click, in pixels.
- Type the correspondent in game values in the *Game coords* text boxes[4].
- Hit *Insert*.

You need to enter more than one coordinate; more coordinates equal a more accurate output. In my example, I've inserted four of them:



pix: 286, 232

coords: 21.0 [9.45] 42.01

Adjoust coordinates: X: + - Z: + -

Last click: 48, 158

game coords:

☑ -20.88 | 121.99
☑ 64.21 | -3.62
☑ -73.72 | -3.62
☑ 66.29 | 150.18

If the number of coordinates inserted is sufficient, those green crosses will appear. The blue circles represent the location you've inserted. The green crosses represent those points, re-projected using the calculated grid. Optimum is when all crosses are perfectly centred on circles.

You can delete a pair of coordinate by clicking the relative button on the right. Or, you can disable one by un-checking it. A disabled pair is ignored by all grid calculations, but is not lost and you can re-activate it whenever you want.

How to know in game coordinates?

This is probably the question rose to you mind when trying to follow the three-step insert. You've basically three ways.

---

[4] See below, to learn how to know these values.

1) **Old school**: by using the *Trees Tab* you can create a new, visible, object at a precise location. For example, a ladder with scaling Y = 10.0 (very high in the sky) at game coords = 0.0 0.0. Then, enter in H&D2 and load this mission. Walk around to search the ladder and look at the m-map to determine where it is. Now go back to RevEngEd 2 and you have a pair of coordinates to set. Click on the map where you saw the big ladder and type 0.0 0.0. This method is complex and inaccurate, and should be avoided if others options are available.
2) **3ds max**: load the *scene.4ds* file of this mission into 3ds max, by using the script realized by *hdmaster*[5]. 3ds max coordinates are equals to game coordinates, so you have only to look at scene loaded and transpose some easily identifiable point.
3) **Reusing**: find some one who already did the work, and copy&paste the right row directly into the project file. See *Project file structure* chapter. I mean to publish good coordinate strings on my web site (http://malgolan.altervista.org), for this very purpose.

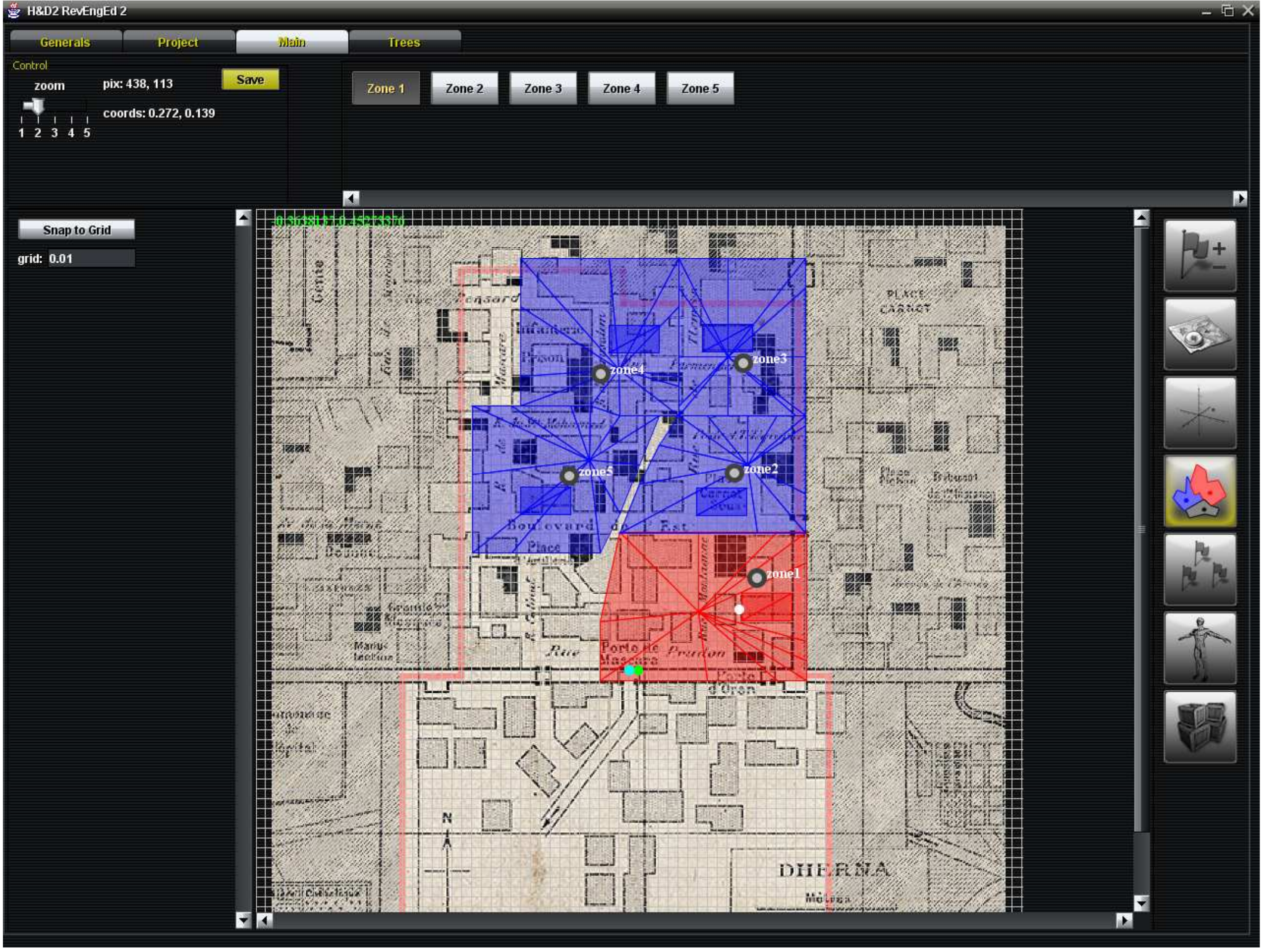
The next steps will be a lot easier if this one is done with accuracy!

---

[5] http://www.hidden-and-dangerous.net/board/index.php/topic,2003.0.html.

Black/Blue areas layer

Here you can adjust those black and blue areas visible on m-map, which change colour when a zone changes owner:



pix: 438, 113

coords: 0.272, 0.139

grid: 0.01

At the top, choose the *zone* to want to work on: only the vertices of that *zone* will be movable. The active *zone* is displayed in red, while other ones in blue.

You can drag vertices one by one, or all of them at once by dragging the white circle. If you drag one of the corners of the little rectangle, all the rectangle will be moved (it's the surface where the texture of the little flag is applied).

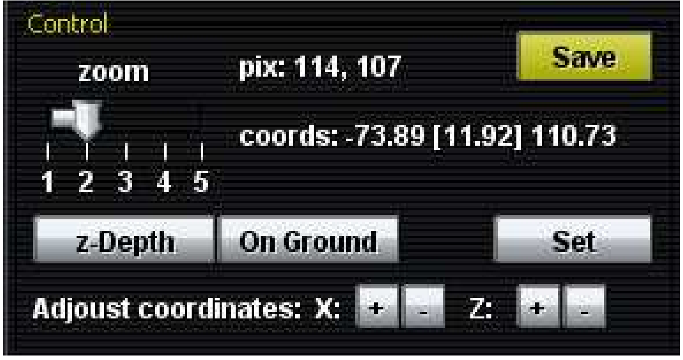
You can snap vertices to the grid, by activating the *Snap to Grid* button. You can also change the grid size by inserting a new value under that button (minimum is 0.01).

Zone circles (which you click so select respawn zone) are automatically moved by the *Zones* layer.

The green and cyan circles are useless to you. The green one indicates the position of the *center* object, while the cyan one the position of the *mapa* object.

The Control Panel

In the top left corner of the *Main Tab* there is a little display, labelled *Control*. Here you can find general purpose functionalities, which can be hidden or shown according to the current layer. When all of them are displayed, you see something like this:



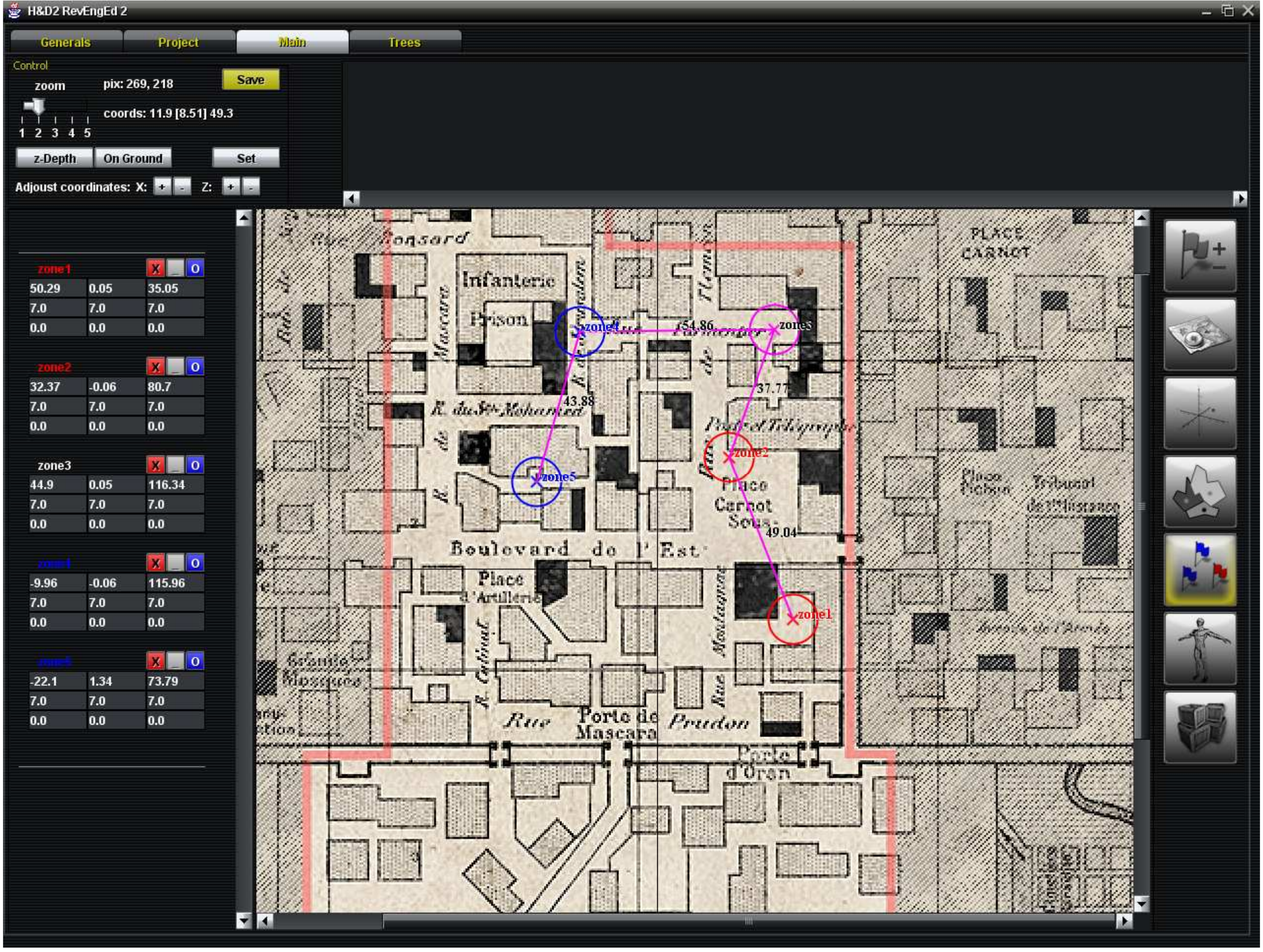
pix: 114, 107

coords: -73.89 [11.92] 110.73

- **Zoom**: change the zoom of map. Note that the *blue/black areas* layer uses a different coordinate system, so zoom can affect it differently.
- **Pix**: pixel coordinate of current cursor position. These coordinates refer to the background .BMP and are zoom independent.
- **Coords**: if *blue/black areas* layer is selected, these coordinates refer to the cursor position relatively to the object of map.4ds. If other layers are selected, these coordinates refer to the cursor position relatively to in game coordinates. If a z-depth map has been loaded, also height is shown, within brackets.
- **Save**: this button save the information of *actors.bin*, *scene2.bin* and *map.4ds* to file.
- **Z-Depth**: this button enables/disables the view of the z-depth image loaded in *Projects* tab.
- **On Ground**: if this toggle button is activated, when an object is dragged in map, its height is automatically set according to the z-depth image loaded. If a valid z-depth map has not been loaded, is impossible to activate this button.
- **Set**: the changes you make to the values displayed in the left column text boxes are not validated until this button is pressed. If you modify them and then change layer or drag an object without hitting set, those changes are lost.
- **Adjust coordinates**: it may happen that the grid calculated by the *Coordinates* layer is not so precise. If you feel that objects should be displayed a little more to the left rather than upwards, you can adjust their position by hitting the plus and minus buttons. You'll see immediately the effects on map.

Zones layer

Here you can adjust *conquest zones*, the core of *occupation* game-play:



pix: 269, 218

coords: 11.9 [8.51] 49.3

zone1

50.29	0.05	35.05
7.0	7.0	7.0
0.0	0.0	0.0

zone2

32.37	-0.06	80.7
7.0	7.0	7.0
0.0	0.0	0.0

zone3

44.9	0.05	116.34
7.0	7.0	7.0
0.0	0.0	0.0

zone4

-9.96	-0.06	115.96
7.0	7.0	7.0
0.0	0.0	0.0

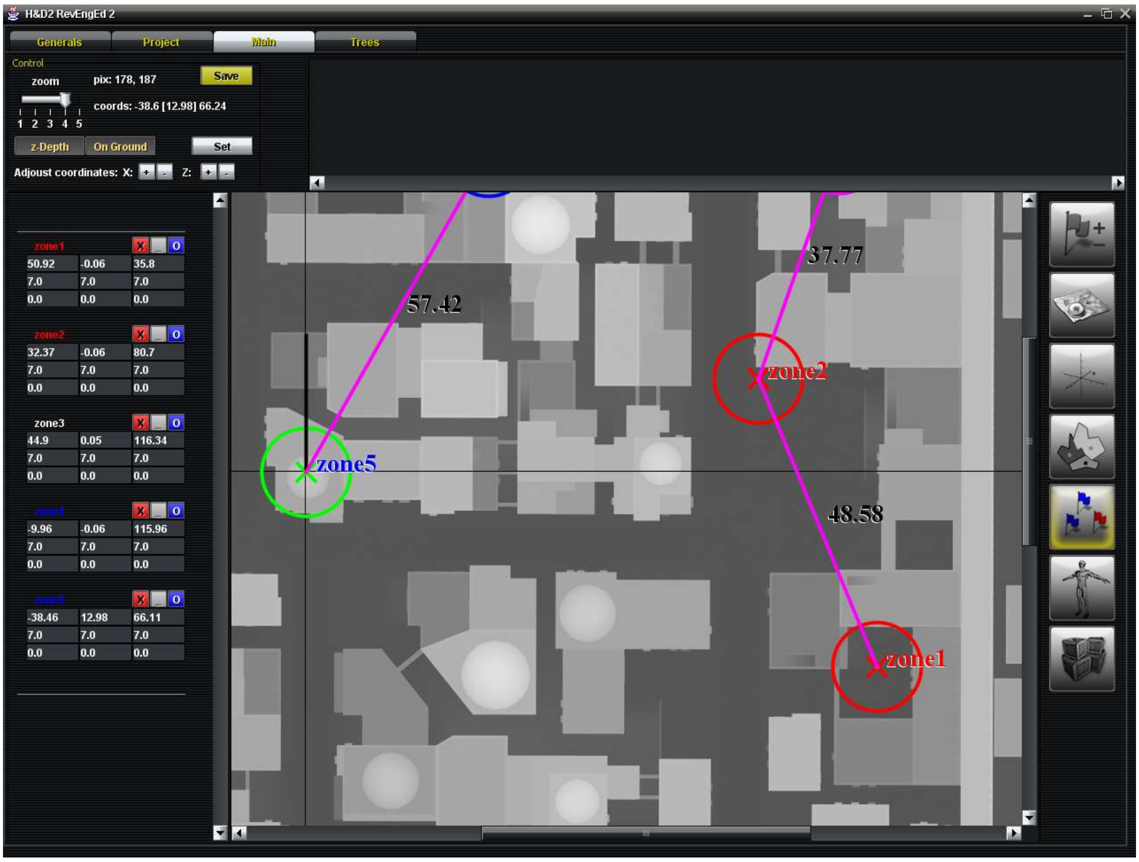
zone5

-22.1	1.34	73.79
7.0	7.0	7.0
0.0	0.0	0.0

You can simply drag the circle where you want to place the zone of conquest. *Zone*, *flagpole*, *flags* will be all automatically moved. You can also set a precise position for it by using the text boxes on the left. If so, remember to hit the *Set* button when you are done, or these changes will be lost.

Height of *zones* is displayed as a vertical black line and can be set in two ways: you can digit it manually in the text boxes on the left or you can activate the *On ground* button (for this second option, a valid z-depth map is required).

Try to activate both *Z-Depth* and *On Ground* button and drag a circle around the map. You’ll see that its height will vary according to the tone of z-depth map:



pix: 178, 187

coords: -38.6 [12.98] 66.24

zone1

50.92	-0.06	35.8
7.0	7.0	7.0
0.0	0.0	0.0

zone2

32.37	-0.06	80.7
7.0	7.0	7.0
0.0	0.0	0.0

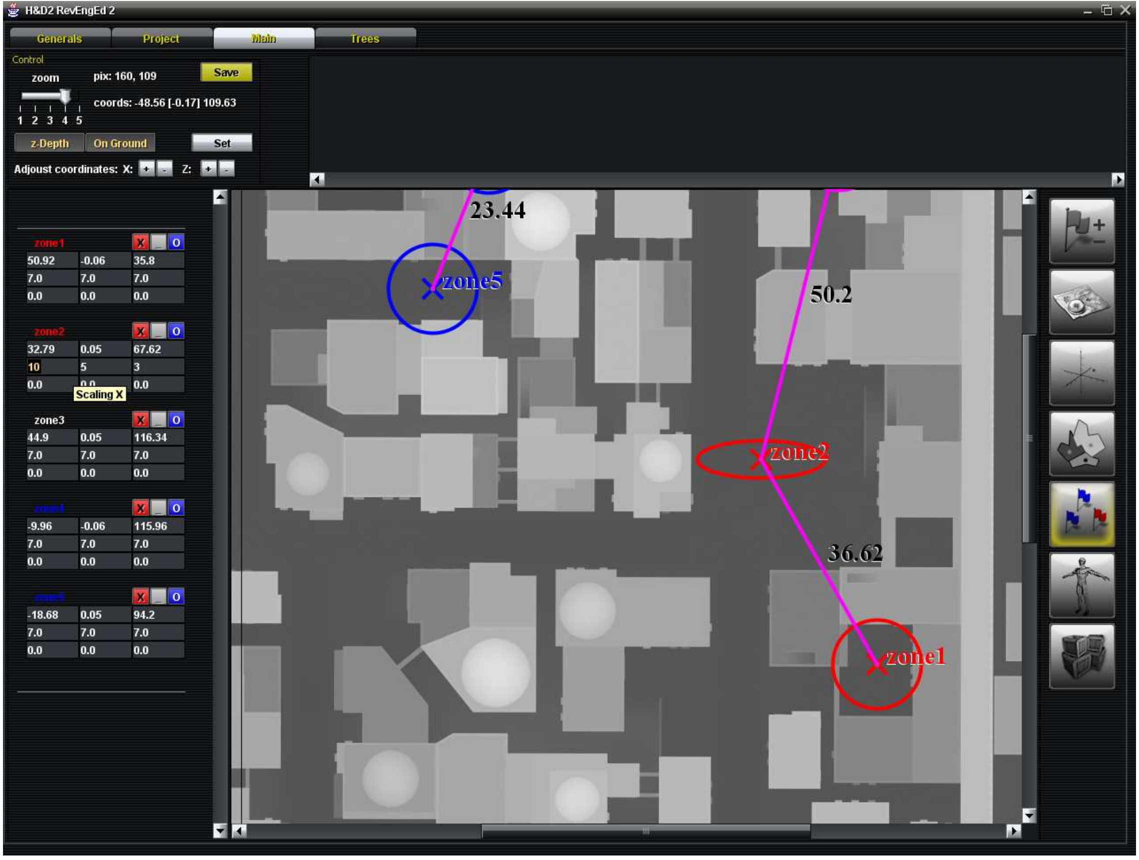
zone3

44.9	0.05	116.34
7.0	7.0	7.0
0.0	0.0	0.0

-9.96	-0.06	115.96
7.0	7.0	7.0
0.0	0.0	0.0

-38.46	12.98	66.11
7.0	7.0	7.0
0.0	0.0	0.0

Radius of *capture zones* is also displayed; it corresponds to actual in game zone of capture. You can adjust it by editing the second line of text boxes on the left (and hitting *set* button as always):



pix: 160, 109

coords: -48.56 [-0.17] 109.63

zone1

50.92	-0.06	35.8
7.0	7.0	7.0
0.0	0.0	0.0

zone2

32.79	0.05	67.62
10	5	3
0.0	0.0	0.0

zone3

44.9	0.05	116.34
7.0	7.0	7.0
0.0	0.0	0.0

-9.96	-0.06	115.96
7.0	7.0	7.0
0.0	0.0	0.0

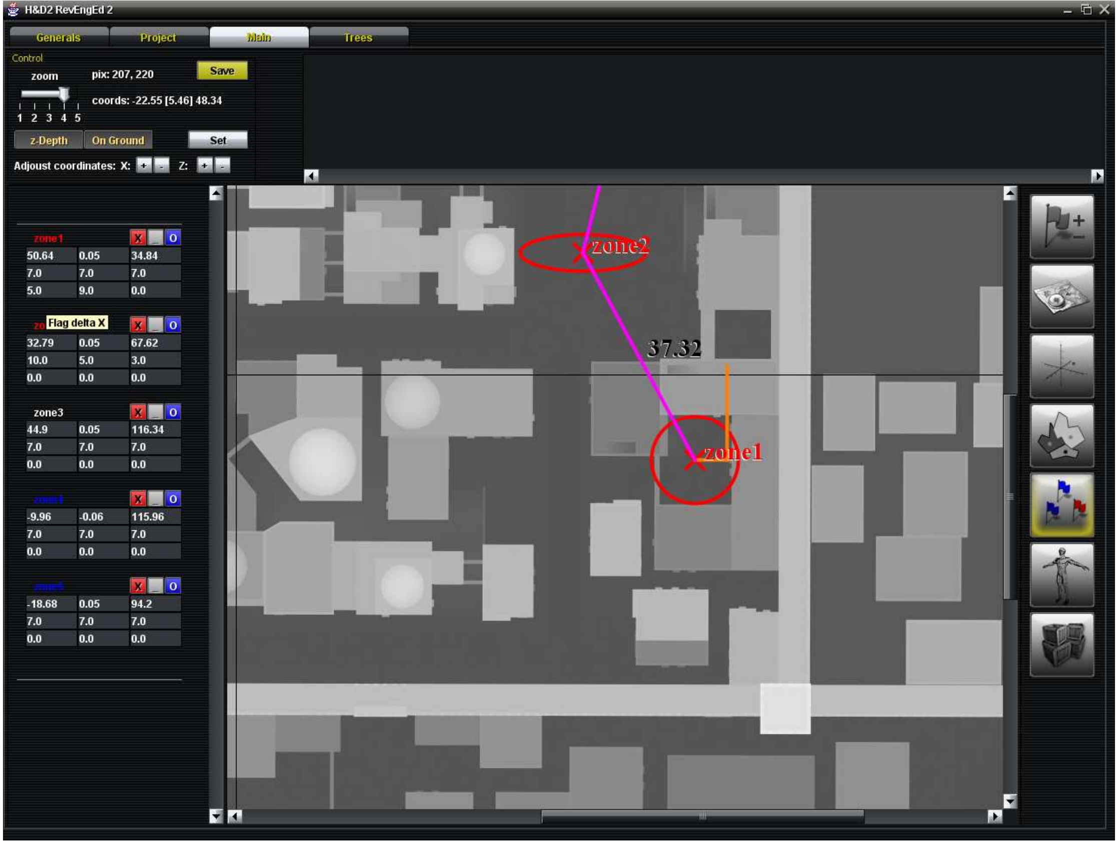
-18.68	0.05	94.2
7.0	7.0	7.0
0.0	0.0	0.0

Distances between subsequent zones of capture are automatically displayed. This should help you judge how far flagpoles are. Subsequent zones in capture order are also linked by a line, to help visualizing the path from one headquarter to the other.

You can simply change the initial owner of a certain zone, by clicking the respective *X*, _ and *O* buttons on the left.

In some situations, you would want to have flagpole far away from the actual zone of capture. For example, when the flag is set higher, or think at Normandy2, where the allied home flag is leaning over the balcony of the bank.

The third line of text boxes on the left expresses a delta (on X, Y and Z) that will be added to the flagpole (and its flags) from the zone position. In this example, the flagpole of zone1 will be 5.0 eastern and 9.0 higher from the zone1 of capture (a hint line is also drawn):



pix: 207, 220

coords: -22.55 [5.46] 48.34

zone1 ☒ _ O

50.64	0.05	34.84
7.0	7.0	7.0
5.0	9.0	0.0

☒ _ O

32.79	0.05	67.62
10.0	5.0	3.0
0.0	0.0	0.0

zone3 ☒ _ O

44.9	0.05	116.34
7.0	7.0	7.0
0.0	0.0	0.0

☒ _ O

-9.96	-0.06	115.96
7.0	7.0	7.0
0.0	0.0	0.0

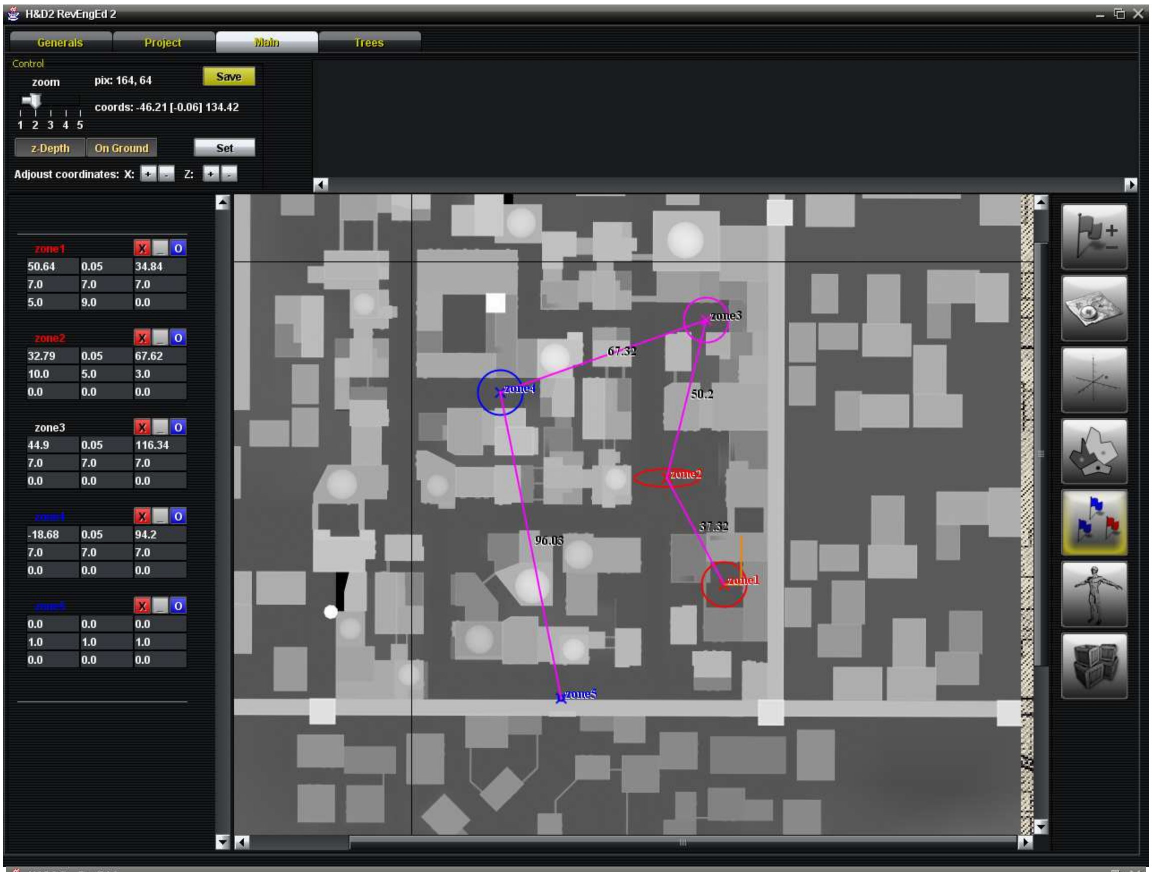
☒ _ O

-18.68	0.05	94.2
7.0	7.0	7.0
0.0	0.0	0.0

Finally, by right-clicking a zone in map, you can **delete it or swap** it for another zone. When you do such operations, all involved objects and parameters are automatically updated!

Instead, if you want to **add one or more zone** of capture, you just have to go back to *Occupation Checks* layer, set a different number of zones (from example from 5 to 7) and hit *Start*. All existent objects will be left untouched, while new objects will be created, accordingly to the occupation structure.

In this example, I've deleted zone4 and then started the checks with 5 zones again:



zoom 1 2 3 4 5

pix: 164, 64

coords: -46.21 [-0.06] 134.42

zone1

50.64	0.05	34.84
7.0	7.0	7.0
5.0	9.0	0.0

zone2

32.79	0.05	67.62
10.0	5.0	3.0
0.0	0.0	0.0

zone3

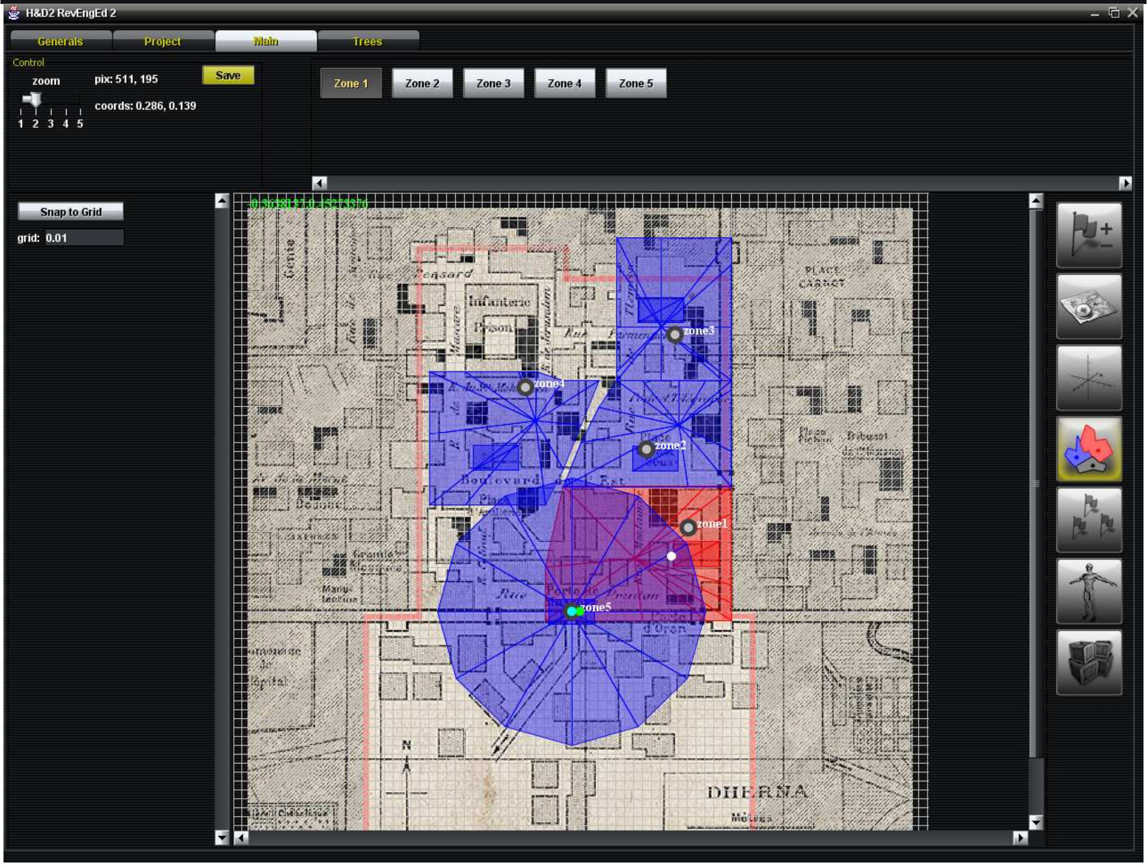
44.9	0.05	116.34
7.0	7.0	7.0
0.0	0.0	0.0

zone4

-18.68	0.05	94.2
7.0	7.0	7.0
0.0	0.0	0.0

zone5

0.0	0.0	0.0
1.0	1.0	1.0
0.0	0.0	0.0



zoom 1 2 3 4 5

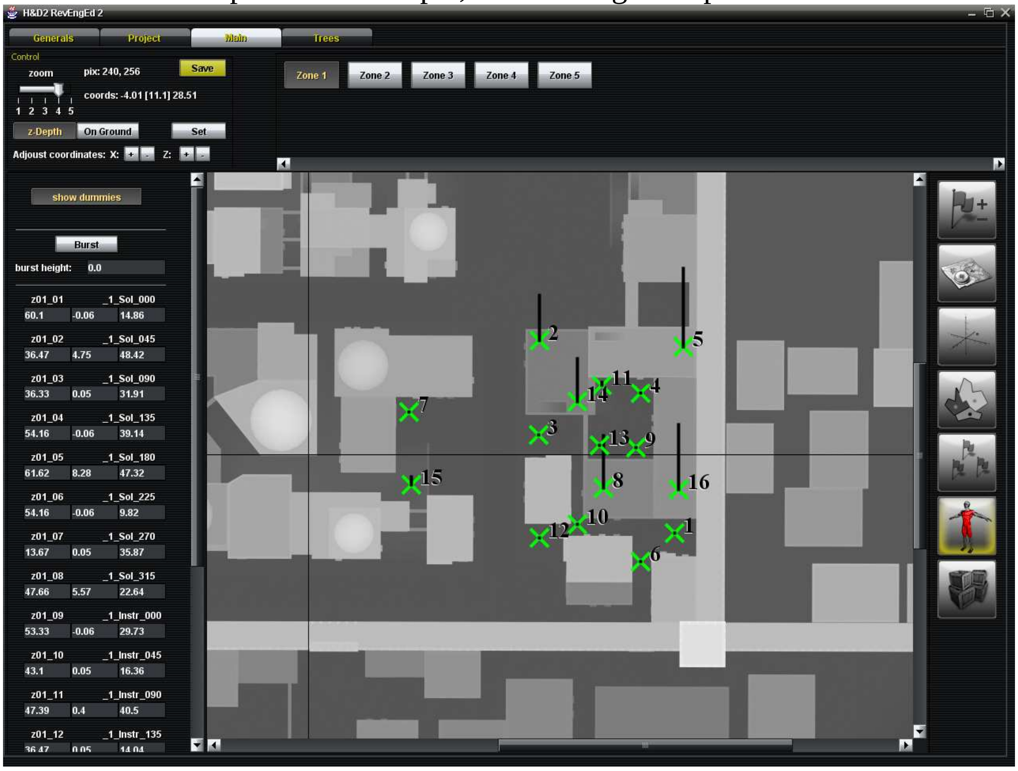
pix: 511, 195

coords: 0.286, 0.139

grid: 0.01

Respawns layer

Number of respawns per zone is fixed (16) and you can set them with this layer. You can visualize respawns for only one zone at a time, and you can switch between various zones by clicking the buttons at the top. In this example, I'm looking at respawns for zone1:



pix: 240, 256

coords: -4.01 [11.1] 28.51

burst height: 0.0

z01_01	_1_Sol_000		
60.1	-0.06	14.86	
z01_02	_1_Sol_045		
36.47	4.75	48.42	
z01_03	_1_Sol_090		
36.33	0.05	31.91	
z01_04	_1_Sol_135		
54.16	-0.06	39.14	
z01_05	_1_Sol_180		
61.62	8.28	47.32	
z01_06	_1_Sol_225		
54.16	-0.06	9.82	
z01_07	_1_Sol_270		
13.67	0.05	35.87	
z01_08	_1_Sol_315		
47.66	5.57	22.64	
z01_09	_1_Instr_000		
53.33	-0.06	29.73	
z01_10	_1_Instr_045		
43.1	0.05	16.36	
z01_11	_1_Instr_090		
47.39	0.4	40.5	
z01_12	_1_Instr_135		
36.47	0.05	14.04	

Like for zones, X and Z position is displayed directly on map, while Y position is represented by a black vertical line.

Like for zones, you can simply drag respawn where you want them, and have their height automatically set if *On Ground* button is active.

You can also set their coordinates manually, by using the text boxes on the left. In this case, make sure to hit the *Set* button after you're done, or changes will be lost when you change layer or drag the correspondent item in map.

If you click on *Burst* button, then each click on map will place a respawn, cycling from 1 to 16. If *On Ground* is active, each respawn is placed at height determined by z-depth map, otherwise, each respawn is placed at height typed inside the *burst height* text box.

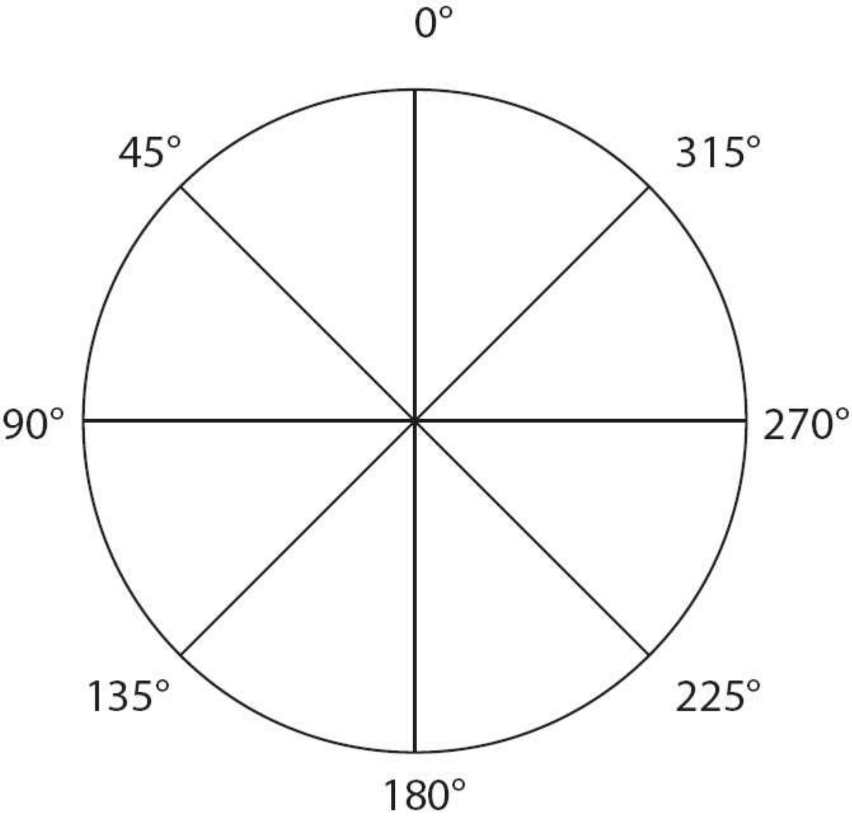
The soldier dummies

On the left, you can see a name associated to each respawn. It's the name of the soldier models automatically added by RevEngEd 2. For example, respawn *z01_02* is associated to solder model *_1_Sol_045*. When you move a respawn, its associated soldier is also automatically moved.

You can show/hide these models in game, by activating the *Show dummies* toggle button. If this button is active, ghost soldier models are shown exactly where you have placed respawns, so you can check in game that everything is ok (i.e. that there are no floating respawns or overlapping with buildings). If you deactivate it, soldier models are placed 100.0 below ground.

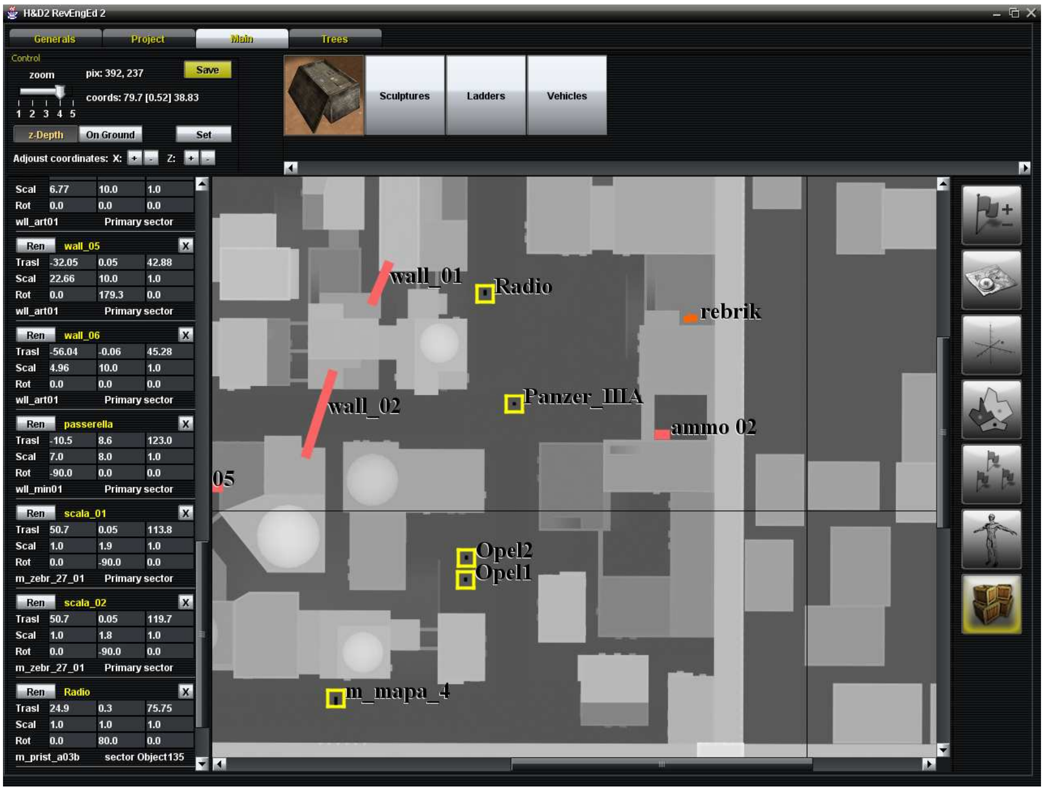
Every soldier model is also rotated by a precise angle, to ease in game identification. For example, *_1_Sol_045* is rotated 45°, while *_1_Instr_135* is rotated 135°. Moreover, *Sol* are brown dressed, while *Instr* are yellow/green dressed.

That’s how angles are related to map:



Actors layer

This layer is used to add, move and remove all other actors, like ammo crates, walls, ladders, vehicles and so on. Look at it:



pix: 392, 237

coords: 79.7 [0.52] 38.83

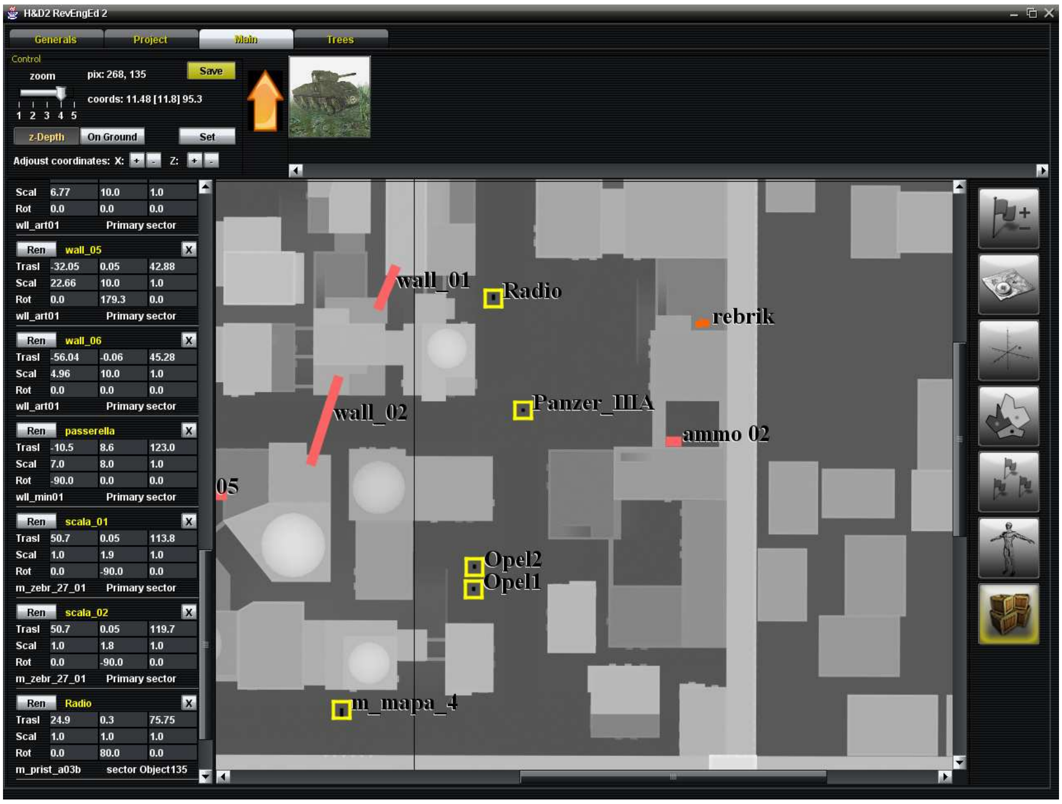
Adjoust coordinates: X: + - Z: + -

Scal	6.77	10.0	1.0
Rot	0.0	0.0	0.0
wll_art01	Primary sector		
wall_05			
Trasl	-32.05	0.05	42.88
Scal	22.66	10.0	1.0
Rot	0.0	179.3	0.0
wll_art01	Primary sector		
wall_06			
Trasl	-56.04	-0.06	45.28
Scal	4.96	10.0	1.0
Rot	0.0	0.0	0.0
wll_art01	Primary sector		
passerella			
Trasl	-10.5	8.6	123.0
Scal	7.0	8.0	1.0
Rot	-90.0	0.0	0.0
wll_min01	Primary sector		
scala_01			
Trasl	50.7	0.05	113.8
Scal	1.0	1.9	1.0
Rot	0.0	-90.0	0.0
m_zebr_27_01	Primary sector		
scala_02			
Trasl	50.7	0.05	119.7
Scal	1.0	1.8	1.0
Rot	0.0	-90.0	0.0
m_zebr_27_01	Primary sector		
Radio			
Trasl	24.9	0.3	75.75
Scal	1.0	1.0	1.0
Rot	0.0	80.0	0.0
m_prist_a03b	sector Object135		

Dragging of objects works exactly like for *zones* and *respawns*. At the left you have usual text boxes to manually insert values (remember to hit *Set* when done), and here you have translation (first line), scaling (second line) and eulerian rotation (third line). You have also a button to rename (*Ren*) and a button to delete (*X*).

At the top, instead, you have a several buttons, each corresponding to a definition type (categories). Some might have an icon (if you hold your cursor over, a hint text will be displayed), some might have a text description.

If you click one of them, they will be replaced by a list of actors:



pix: 268, 135

coords: 11.48 [11.8] 95.3

Scal	6.77	10.0	1.0
Rot	0.0	0.0	0.0
wll_art01	Primary sector		

wall_05

Trasl	-32.05	0.05	42.88
Scal	22.66	10.0	1.0
Rot	0.0	179.3	0.0
wll_art01	Primary sector		

wall_06

Trasl	-56.04	-0.06	45.28
Scal	4.96	10.0	1.0
Rot	0.0	0.0	0.0
wll_art01	Primary sector		

passerella

Trasl	-10.5	8.6	123.0
Scal	7.0	8.0	1.0
Rot	-90.0	0.0	0.0
wll_min01	Primary sector		

scala_01

Trasl	50.7	0.05	113.8
Scal	1.0	1.9	1.0
Rot	0.0	-90.0	0.0
m_zebr_27_01	Primary sector		

scala_02

Trasl	50.7	0.05	119.7
Scal	1.0	1.8	1.0
Rot	0.0	-90.0	0.0
m_zebr_27_01	Primary sector		

Radio

Trasl	24.9	0.3	75.75
Scal	1.0	1.0	1.0
Rot	0.0	80.0	0.0
m_prist_a03b	sector Object135		

You can go back to categories by clicking the orange arrow.

These actors are loaded directly from the */Library/* folder of RevEngEd 2. If you put a specification and a definition inside this folder (and optionally an image), it is automatically categorized and displayed above (you'll have to restart RevEngEd 2 tough). Specification and definition must have matching names, and their extensions must be respectively .dnc and .def. If an icon is present, must be 100x100, have matching name too, and .jpg extension.

Trees Tab export automatically creates suitable files for Library.

To add a new actor, just click on its icon and click in map. You'll be asked a name and the actor (along with its definition) will be automatically created. Only if you add a wall[6], 2 clicks are required, and the wall will extends exactly from the first click to the second.

To determine if an actor is a wall, RevEngEd 2 looks at its model. If it starts with *wll_* then it's a wall, otherwise not. Keep this in mind, if you plan to add new wall models to library.

---

[6] Walls are nothing but empty stretched crates with new textures.

Errors and Exceptions

Unexpected events are not perfectly handled, so, if an error message pops ups, it’s better to exit without saving to avoid files corruption!

Deterministic errors happen because of at least two reasons:

1) **Bugs**: if you find some, please notify me on http://malgolan.altervista.org
2) **Wrong occupation structure**: for example, if you mess up with the Trees Tab after checks are done, and you delete important pieces, RevEngEd 2 could act as those pieces are present and results of unhandled errors.