# Table of Contents

**SCREEN LAYOUT** .................................................. 3

**EDITOR TOOLS** .................................................. 3

- **PLACE TILE/OBJECT** .......................................... 3
- **EDIT** .............................................................. 4
- **CREATE/EDIT EXIT** ............................................. 4
- **MONSTER SPAWNER** ............................................ 5
  - **EDIT SPAWNER DIALOG** .................................... 5
- **EDIT PATH** ........................................................ 6
- **FILL TILE** .......................................................... 7
- **NPC PLACE/EDIT** ............................................... 7
- **INCREASE MAP SIZE** .......................................... 7

**HOT KEYS** .......................................................... 7

**DESIGNER TOOL REPORTS** ...................................... 8

**ADDING YOUR LEVELS** .......................................... 8

- **CREATING A LEV** ............................................... 8
- **DESIGN TIPS** ..................................................... 9
  - **MODIFYING THE GAME** ..................................... 9

**FILE FORMATS & DESCRIPTIONS** ............................... 10

- **.LEV** .................................................................. 10
- **.DAT** .................................................................. 10
- **.BMP** .................................................................. 19
- **.WAV** .................................................................. 19
- **DIRECTORY STRUCTURE** ....................................... 19
Screen Layout

Editor Tools

Place Tile/Object

The Place tool used to place tiles or objects on the map.

- To place an Object
  Left click on the object icon to select it for placement. Now when the mouse is in the viewport area, simply hold the left mouse button down until you position the object for exactly where you want it. Release the mouse button to place it. If you don't want to place an object down, after you have
pressed the mouse button, you may either hold down the SHIFT key and release the mouse button, or you may simply move your mouse outside the view area and release the mouse button.

- **To place a tile:**
  Left click on the tile icon to select a tile as primary placement, or right click on the tile for secondary placement. While the mouse is over the viewport you may either left click to place the primary tile or right mouse click to place the secondary tile selection. To reset the secondary tile to the "default tile" you may right click in the Tile Swatch mini-window (below the Tile Palette Selection Window).

**Edit**

![Tile Icon]

The Edit tool is used to edit containers, gold piles, view object properties, and delete objects. To stop editing press the ESC key.

- **To Edit an object:**
  Use the mouse to drag a selection box around an object(s). The object's image (the bitmap rectangle) must be at least a third inside the selection box to be selected. Release the mouse button and the selected objects will be highlighted. A small purple square will be displayed in the center of the current object. To choose a different object to edit press the TAB key.

- **To Delete an object:**
  Once the object you want to delete is highlighted, press the DELETE key or you may hold the SHIFT key and press the right mouse button.

- **To view object Properties:**
  Once the object you want is highlighted, press the ENTER key or you may right click (No SHIFT KEY). The Z-Depth field of the item properties box is used to ensure the object is always displayed on top of the objects behind it. Currently only values of 0 or 1 are used. The properties dialog of an object is determined by the type of object being selected. Thus if a container object is selected, a dialog will be displayed allowing you to edit the contents of that container (a side note, containers can not contain other containers). Editing a door allows you to lock/unlock the door and select the key required to do so. And likewise for other types of editable objects.

**Create/Edit Exit**

![Exit Icon]

The Exit tool is used to place a map exit or shop entrance on the map. When the player enters the invisible rectangle in the game the level file specified will be loaded. Left click and drag the exit rectangle to the size you need. When prompted to enter the exit information you may type your own level name or select one from the drop down list. If the exit is for an actual level the name must have a .lev extension. The name can be the same map if you simply want to reposition the player on the map (as if using a teleporter). If the exit is actually a shop entrance then the exit map name should be one of the pre-defined shop names in the game engine code. The exit coordinates for a shop entrance are actually the coordinates that the player will return to when exiting the shop. To edit an existing exit double click inside the exit rectangle.
If you examine some of the game levels you would eventually find yet another type of exit. Late in the game’s development exits were also used to trigger hidden NPC scripts for additional plot development (when entering a room a script would appear, or after defeating a main boss). Unfortunately, for various reasons, the exit names were hardcoded and they can only be triggered once per new game.

**Monster Spawner**

The Monster tool is used to place a monster spawner on the map. A default spawner can be set so that subsequent spawners that are placed will use your defined settings. This is helpful if you are placing numerous spawners on the map that are going to use the same settings.

- **To set a default spawner:**
  Click on the "Egg" button and set the spawner settings.

- **To place a spawner:**
  Left click in the viewport. To toggle a spawner formation, prior to placement, press the TAB key. You will only see the spawner shape change if you are placing a spawner with more than one monster per spawn.

- **To edit a spawner:**
  Right click in the placed spawner rectangle (the monster tool must be the current tool). Each spawner has red spawn detection sectors which trigger the spawn timer. To resize the radius formation press and hold the right mouse button plus the CTRL (control) key inside of a spawner. Then simply drag to a new size. It may help to press the mouse button first then press the CTRL key to keep the map view from scrolling. You may also edit the radius size directly (type a new number) by editing the spawner (right click the spawner).

**Edit Spawner Dialog**

When editing a spawner (or setting a default spawner) this dialog is displayed. Most of the entries are self explanatory. However, some are not. The Path presets simply allow you to quickly choose a geometric shape that the monster will traverse. More often than not, this is primarily used to set the number of nodes desired and then the path is reshaped to simulate “walking around.” The Patrol setting will cause the monster to follow its path until the player comes within its detection radius. If the player runs away the monster will return to its patrol path. The Guard setting will simply have the monster stay stationary until the player comes within its detection radius. The monster will then attack the player. Again, if the player runs away the monster will return to its guard position. The wander setting is for monsters to simply “wander about”. The path shape used for wandering monsters will affect how realistic the wandering appears. At each path node the monster will have a random chance of mirroring the path shape so that it heads off in a new direction.

A monster (or group of monsters depending upon your settings) is spawned whenever the timer has expired (the number of msec between spawns). The current frequency can be used to offset the timer for its first spawn. For example a spawner with a msec setting of 1000 and a current frequency of 400 would have the first spawn at 600 milliseconds after the player walked into the detection radius. Then every successive
spawn would occur at 1000 milliseconds. The infinite check box simply means that the number of spawns is set to 32000 (that is a lot of monsters to spawn and as such was considered “infinite” for all intents and purposes.

**Edit Path**

The Edit Path tool is used to edit the walking path for a spawned creature or NPC.

- **To reposition a node**
  Left click on the node and drag to a new location.

- **To scale the path size**
  Left click inside of the spawner rectangle, then while holding the mouse button down and pressing the CTRL key drag a new size. Be careful not to oversize the path! Paths are only shown if the spawner is in the viewport, so if you need to edit a node it may not be visible for you to edit it. You can always scale the path back down first, if needed. You may also mirror the path quickly by moving the mouse pointer over the spawner whose path you wish to mirror and then pressing the TAB key repeatedly.

  If you wish to reset a spawner’s path to the preset shape and size you must first edit the spawner and select a different shape, press okay, then edit the spawner again and choose the shape you truly desire. This was done to prevent accidental changing of the spawner’s current path shape. You edit
NPC paths in a similar manner, only their paths always use the same shape. A final note, you are not able to change the starting path node location for a spawner or NPC.

Fill Tile

The Fill tool is used to place tiles in a large area quickly.

- **To fill an area**
  To fill with the primary tile (ensure a tile is highlighted in the tile selection window, not an object from the object window) drag a selection box with the left mouse. Whichever tile locations the selection box intersects with will be filled with the tile. To fill with the secondary tile selection simply perform the same steps as the primary accept use the right mouse button. If you hold the SHIFT key down while you release the mouse button you will not fill.

  **TIP:** You can use the fill tool with a "wall" tile to build a long wall quickly. You may also use the fill tool to place a single tile by not dragging a selection box (or a very small one).

NPC Place/Edit

The NPC tool is used to place an Non-Player Character on the map. Simply left click to place the NPC, or right click on an existing NPC to edit it. Be Careful of which NPC's you place. The NPC must have a script sequence (or have a generic NPC ID in the data file). Otherwise the main game engine will not be able to load the NPC file. Certain NPC’s are “invisible.” These NPC’s are part of the exit trigger/script code for Arkoss. Certain exits (with special names) were are capable of triggering the script sequence from one of these invisible PCs.

Increase Map Size

Pressing this button will allow you to increase the size of the map after you have created it. Unfortunately there is no way to decrease the size of the map.

Hot Keys

- **A** Press to scroll the tile selection window up one page (8 tiles).
- **Z** Press to scroll the tile selection window down one page (8 tiles).
- **S** Press to scroll the object selection window down one page (8 objects).
- **X** Press to scroll the object selection window down one page (8 objects).
CTRL + Mousemove Will scroll the viewport around your map.
SHIFT + MouseBtn This will disable the current command. Currently only for Fill, and Object Placement.
SHIFT + RMouseBtn If editing deletes the current edit object.
MINI-VIEW The miniview displays the size of the current map. To quickly jump to a new location of your map, simply left click in one of the map sectors.
ZOOM Features: Full view means to view the entire map in one screen size. Other sizes will zoom the map from your current map location into the viewport. Exits are highlighted yellow, doors are purple, spawners are green, and all other objects are blue.

NOTE: The layer control menu will also control which items and tiles are displayed in the zoom view. Whenever a menu option is changed the zoom option must be selected again to redraw the view. To continue editing left click in the viewport area and you will be returned to normal view size (you are not able to edit the map in a zoomed view). The Zoom-BackLit feature will highlight any non-walkable tiles red and the background will be gray (not black) to aid in viewing darker tile sets.

**Designer Tool Reports**

There are a few tools to help you design levels. A report is simply a list view box that pops up and displays the data (it is not a printout or saved data file.) The List Keys report is used to display the total key number of keys used and or placed on the current map. Use this to verify you have enough keys to open up the locked objects you put on the map! The List Exits report is used to summarize the current map exits. Use this to ensure you have assigned valid data to each of the exits. You can open up another instance of the editor with another map and check the map exits in one editor against the destinations in the other editor. BE SURE TO TURN OFF THE AUTO_SAVE FEATURE IN ONE OF THE EDITORS when two or more instances are open (just to be on the safe side). The List NPCs report is used to summarize the NPCs actually placed on the current map. You can use this to verify a town has the correct NPCs in it. Or if you need to find where you placed an NPC you can simply get the placement coordinates from this report and then use the Display Coords option to hunt it down. The last report, and perhaps the most important, is the Monster Counts report. This report is used to balance a level for playability. The report will list the total number of monsters that can be spawned and the range of experience points that can be awarded if all the monsters are killed. You can use this to determine how many levels a player can potentially gain on that map, and the monsters they can fight.

**Adding Your Levels**

**Creating a LEV**

The editor uses a slightly different file than the game engine. In order to use your new map in the game engine, you must first compile your .MAP file into a .LEV file for the engine. Simply click on Tools from the menu bar and then select Compile Level, then select Text Mode. Note that the Binary mode was a feature that was never utilized, and thus not necessary. The .LEV file will be placed in the maps directory from where you opened it. Simply copy this file over to the sub-directory called Master, where the game was installed. The batch compile option will compile the entire directory of .MAP files to .LEV files. If you have a lot of maps to do, this is handy. NOTE that you should save your current map before using the batch compile feature.
In order to link the various maps together you must be sure to place an exit on the map that leads you from one to the other. In other words, if you want to link one of your new maps to the existing game maps then you must be sure to place an exit from the existing game map to your map and vice versa.

Once you have copied the .LEV files from your working directory (the directory where the .MAP was saved) over to the Master sub-directory of the game engine, you will need to modify the data file called FILELIST.DAT in the sub-directory called Master from where the game was installed. Simply open it up in a text editor (like NotePad) and type in the names of the .LEV files you want included with the game. The first name at the top is the level you will start on.

If you want to change the coordinates of where the player starts on the first map then there is one more file to change. Open up the PLAYER.DAT file from the Master sub-directory in a text editor (like NotePad). Each of the characters has two variables called [startX] and [startY], the numbers immediately following these labels are their world coordinates (820, 364). Simply change these for each of the characters to the coordinates you desire and you are done!! One final note, these changes will not affect your old saved games. Any new game or new saved games will have your new levels, though.

**Design Tips**

Here are a couple of tips on how to make levels that don't hinder game performance. We only tested map sizes up to 20 sections by 20 sections. If you have more than 25 to 30 active monsters on the screen at one time (running on the recommended computer system) you will probably experience some game lag. If you are planning on running your maps on a system with only a 4 MB video card then don't put more than 3 or 4 different monster types on the same map, otherwise the images may be stored in system memory and the frame rate may drop. The frame counter in the game engine is simply a reference gauge. We notice a loss in gameplay when the counter drops below 40. If the counter is running at about 130 you will have excellent gameplay. So far the fastest we've seen is 528 (PIII 450, NT4, TNT2) :)

When saving a map there is an option to save only the tiles. This was originally used for testing purposes but later became a “feature” of sorts. With this feature the editor can essentially become a tile editor for any game engine. You can load in any tile palette and place the tiles then save just the tiles. This file could then be loaded by another tile-based game engine.

Use a combination of the spawn detection radii and timer frequencies to set “traps” for the players. The subterranean maps have several examples of such spawn traps.

- **To enable the debug/cheat commands**

  Open up the file CONTROLS.DAT in the Master sub-directory of where you installed the game. The [Options] entry should have a 1 under it (a value of 0 disables the cheats). Now when playing the game, press the ALT and F1 keys at the same time. A dialog box will appear allowing you to change various settings in the game so that you can more efficiently test your levels (or simply cheat >:) ).

**Modifying the Game**

So making levels wasn’t enough to satisfy your taste for designing. Well that’s no problem. The quickest and easiest method of modifying the game is to change the sounds or artwork. All the artwork is
stored in 24 bit color Windows Uncompressed Bitmaps. Just open up your favorite paint program and go at it.

The other method of modifying the game can be done through creating more objects or modifying existing objects. Check out the File Formats section to find the type of file you would like to modify. Also take heed of the warning listed in the File Formats section.

**File Formats & Descriptions**

Please note that while some of the file layouts are listed below, it is not recommended that you edit them. Erroneous or incorrect values could cause the game to not operate properly. If the affected files are not capable of being repaired then a reinstallation of the game would be necessary.

However for those of you who feel confident and have the burning desire to customize your own game then these file listings are the way to go about doing it.

**.LEV**

The .LEV files, level files, store the placement data and current state data for a particular map. This includes the following information:

- Support Data Filenames
- Tile Data
- Placed Object Data
- Money Data
- Exit Data
- Spawner Data
- NPC Data

**.DAT**

The .DAT files, data files, store the master list type information for maps, NPCs, items, etc. The data file may either be level specific or may be for all levels in the game.

**All game levels:**

**Player.dat:**

- Player Name (do not change, it controls read in functions)
- PlayerHeight
- PlayerWidth
- Absorption Modifier
- Armor Class
- Accuracy
- BaseSpeed
- Speed
- DamageMax
- DamageMin
- DamageMod
- Dexterity
- Strength
- DodgeMod
Encumbrance
Firing Rate
Gold
Current Hit Points
Maximum Hit Points
Current Level
Experience Points
Current Magic Proficiency
Maximum Magic Proficiency
Current Melee Proficiency
Maximum Melee Proficiency
Current Missile Weapon Proficiency
Maximum Missile Weapon Proficiency
Current Technology Proficiency
Maximum Technology Proficiency
Data File Label Text [startX]
Starting X Coordinate
Data File Label Text [startY]
Starting Y Coordinate
Destination Rectangle for player Legs sprite (Left, Top, Right, Bottom)
FileName for Leg Sprite Data
Destination Rectangle for player Torso sprite (Left, Top, Right, Bottom)
FileName for Torso Sprite Data
FileName for Paperdoll Sprite Data
Player Type Id (should be 1)
StateFlags (should be 0)
BattleFlags (should be 0)
KeyFlags (bitfield for which keys have been found, should be 0)
NumHealPotions (number of type 1 heal potions for player, should be 0)
NumHealPotions (number of type 2 heal potions for player, should be 0)
Number of Keys (for the 8 types of keys, all should be 0)
Width Collision box modifier (value is subtracted)
Height Collision box modifier (value is subtracted)
Number of monsters killed by player
Number of potions consumed by player
Amount of Gold player has acquired.
Direction of Sprite to face
Current Main Book Page Text (must be 10 lines, no blank lines)
Default Weapon Id
Current Weapon Id
Default Armor Id
Current Armor Id
MasterObj.dat: This is the data file which contains information for all the game objects that are or can be used on different levels. These are primarily inventory types objects, weapons, keys, potions, etc. The actual data that is stored is dependent upon the type of object to be read in.

The following short lists are the various flag values that can be set in some of the object data fields.

Collision Flag Type:
- 1 Can Walk On
- 2 Can Pick Up
- 4 Can See/Shoot Over
These values are added together for multiple effects.

Potion Effects:
- 0x0001 Current Hitpoints Increased by 1 - 10
- 0x0002 Strength Increased
- 0x0004 Dexterity Increased
- 0x0008 Maximum Hitpoints Increased
- 0x0010 Damage Modifier Increased
- 0x0020 Damage Absorption Increased
- 0x0040 Dodge Value Decreased
- 0x0080 Current Hitpoints Increased by 1
- 0x????0000 Amount to adjustment by
These values are added together for multiple effects.
example: 0x00010004 would increase the player’s dexterity by 1 point

Weapon Attack Resistances: used to indicate which resistance applies to this weapon.
- 0x01 Sharp
- 0x02 Blunt
- 0x04 Force
- 0x08 Electrical
- 0x10 Fire
- 0x20 Water
These values are added together for multiple effects.

Weapon Attack Type: used to indicate what kind of weapon it is
- 0x0001 Melee (the weapon shot is created immediately in front of the player)
- 0x0002 Missile (the shot is projected forward from the player)
- 0x0004 Explosion (the shot expands outwards from where it is created)
- 0x0008 Summon (no longer used)
- 0x0010 Ranged Loss (does weapon shot lose energy based on distance)
- 0x0020 Target Loss (does weapon shot lose energy after hitting each target)
- 0x0040 Multiple Attacks (can weapon damage multiple targets before stopping)
- 0x0080 Can Damage Player
- 0x0100 Can Damage Monster
- 0x0200 Lightning (specific case)
These values are added together for multiple effects.
Weapon Proficiency Identifiers: used to determine which proficiency is required

0x0001 Melee
0x0002 Missile
0x0004 Magic
0x0008 Technology

Containers: Denoted by the [container] tag. These are objects which can contain other objects. Do not place container objects within other container objects.

# of Different Container Objects (only listed once)
Collision Flag Type
Collision Box Adjust (Left, Top, Right, Bottom)
Lock Value: value of the key required to unlock it (should be 0 for a master file)
Image Rect: (Left, Top, Right(width), Bottom(height))
Sound Value (unused)
Name (max 28 characters)

Money: denoted by the [money] tag, this defines objects that increase the player’s gold

# of Different objects (only listed once)
Collision Flag Type
Collision Box Adjust (Left, Top, Right, Bottom)
Image Rect (Left, Top, Right(width), Bottom(height))
Minimum Value of object
Maximum Value of object
Name of Object

Armor: denoted by the [armor] tag, objects worn by the player

# of Different objects (only listed once)
Collision Flag Type
Collision Box Adjust (Left, Top, Right, Bottom)
Absorption Value (# of hitpoints absorbed by the armor per attack)
Dodge Value (amount to reduce players ability to dodge an attack)
Encumbrance (amount of weight of object, affects amount inventory)
Cost (cost of buying the item)
Speed Mod (unused)
Image Rect (Left, Top, Right(width), Bottom(height))
Buyable (1 if object can be purchased in a store)
Item ID (unique integer identifier)
Name

Key: denoted by the [key] tag. These can be any object that unlocks another object. An example, other than a literal key, is the map used to “open” the mountain pass rock.

# of Different objects (only listed once)
Collision Flag Type
Collision Box Adjust (Left, Top, Right, Bottom)
Lock Value (the “Unlocking” value of this key, when an object is locked with this value then the matching key is used to unlock it)
Image Rect (Left, Top, Right(width), Bottom(height))
Name

**Potion:** denoted by the [potion] tag. These are objects which can influence various player statistics. Examples are magic rings, healing potions, and single use potions

# of Different objects (only listed once)
Collision Flag Type
Collision Box Adjust (Left, Top, Right, Bottom)
Item’s Effect (hexadecimal)
Image Rect (Left, Top, Right(width), Bottom(height))
Name

**Weapon:** denoted by the [weapon] tag. These are objects that the player can arm in order to attack other creatures. This object type is unique in that each weapon may have a subset of data corresponding to a sub weapon. For example a fireball is a projectile which when reaching its destination changes into a different type of weapon, an explosion. There is no fixed number of sub weapons for a given main weapon. Also note that all weapons create a “shot” when the player attacks. This shot is what is updated every frame. For melee weapons this shot is simply invisible.
The weapons can be a bit overwhelming, a proven technique to creating new weapons is to copy and paste existing weapons that you would like to mimic and then modify only those values you wish to change (don’t forget to update the # of total objects just under the [weapon] file tag).

# of Different weapon objects including # of sub stats (listed only once)
# sub stats for this weapon
Unique Weapon Id
Collision Flag Type
Collision Box Adjust (Left, Top, Right, Bottom)
Weapon Velocity (used for updating weapon shot)
Master Object Image Rect (Left, Top, Right(width), Bottom(height))
Maximum Energy (controls damage and update distance)
Maximum Distance (controls update distance)
Energy Loss per Update
Proficiency Identifier (hexadecimal)
Maximum Damage (per update)
Minimum Damage (per update)
Required Proficiency Rating (amount needed for 1:1 use of weapon)
Firing Rate
Resistance Type (hexadecimal)
Attack Type (hexadecimal)
Number of animation frames
Animation Frame Rects (Left, Top, Right, Bottom) must match number of frames
Collision Rect Adjustment (Left, Top, Right, Bottom) must match number of frames
Encumbrance
Cost
Buyable (1 if object can be purchased in a store)
Sound File IDs (hexadecimal, see sound.dat explanation)
Animation Rate (* 10msec)
Name (also used to identify weapon image file name, ie. FIREBALL.BMP)

SubWeapon Data will follow immediately if indicated otherwise continue with next weapon.....

NPCList.dat: This is the data file which contains the information for all the NPC’s in the game. The current design does not incorporate a large number of NPC’s, therefore there was no need to separate the NPC’s into data files by level. The general format for the file is the following:

# of NPCs in data file
NPC id# (250 if its a generic NPC, no script only a 1 liner response)
NPC Name
AIStateIndex (0x0 stand still, 0x3 walking)
Current HitPoints
AC (0-18)
KeyFlag should be 0- (unused)
StatusFlags (0x8 NPC has an item to give to the player)
TreasureSet (the set which is dropped when the player succeeds with quest or conversation)
ImageSourceRect(Left, Top, Right(width), Bottom(height))
ImageDestinationRect(left) should be 0
ImageDestinationRect(top) should be 0
ImageDestinationRect(width)
ImageDestinationRect(height)
Last Location x should be 0
Last Location y should be 0
Number of path nodes should be 8
Current path node should be 0
(8 pairs)
Path Node x should be 0
Path Node y should be 0
AI Path id# default is 4
Image Surface Index (0: still images, 1-3 female walking, 4-6 male walking)

Script.dat: This is the data file which contains information for all the NPC script sequences in the game. The general format is the following:

Total # of NPCs to read
NPC Name
NPC id#
# of Script Sets (max 2)
# of scripts to read for this NPC
ScriptId
SpecialFlags
UserResponsesIDs (bit field representing which responses to display, eg. 0x11 displays first and fifth response
Paragraph

# of Player responses for this NPC
Response id# for this NPC
Paragraph

HDLevel.dat: The monster hit die levels are stored here. The corresponding individual monster statistics from the Monster.dat file are not used. Twenty levels must be specified, the level is determined by the order during file read in.

Minimum Hit points
Maximum Hit points
Minimum Damage
Maximum Damage
Experience Points Minimum
Experience Points Maximum

Opener.dat: This is the data file which controls the character description text that is displayed below the character portrait screens, as well as the introduction text after selecting a character. The @ indicates a paragraph. The * indicates an end of the story. The story names are coded specifically, so do not change the names.

Endings.dat: This is the data file which handles the various game ending stories for the characters. It follows the same format as Opener.dat.

Treasure.dat: This is the data file which controls the various treasure sets for creatures or NPCs. When a creature is killed depending upon which treasure set they have, there is a chance items from that treasure set will be dropped.

# of Treasure Sets to read in (listed once)
% chance Armor Appear
% chance Money Appear
Number of Potions
% chance for Potion to Appear
% chance for Weapon to Appear
Armor Id #
Key Lock Value
Money Name
Minimum Money Amount
Maximum Money Amount
Potion Name
WeaponId

**Monster.dat:** This data file contains all the data necessary for the various monsters in the game, this includes monster data and monster weapon data.

- # of monsters to Read in (listed once)
- Unique Monster Id#
- Monster Name
- Monster Image Filename
- Hit Points (unused)
- Experience Points (unused)
- Armor Class
- Minimum Damage (unused)
- Maximum Damage (unused)
- Resistance Type (hexadecimal, see weapon resistances)
- Speed (unused)
- Image Rect (Left, Top, Right(width), Bottom(height))
- X dimension Collision Box Adjust
- Y dimension Collision Box Adjust
- TileCollision Modifier (unused)
- Sound Id’s (see sound listing)
- Weapon ID (which weapon does this monster use.)

After the Monsters have all been listed in the data file a [weapon] file tag will appear which separates the monster data from the monster weapons data. The monster weapons data follows the same format as player weapons data.

**Sound.dat:** This data file should not be edited. The description that follows is simply to understand how to understand the file so that ID’s can be assigned to monsters and weapons. The file layout is as follows:

<table>
<thead>
<tr>
<th>Item name</th>
<th>Original wave name</th>
<th>ID Number</th>
<th>Number of Duplicate Buffers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weapon Sound ID’s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(files 1 -60)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0xAABBCCDD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A: Weapon Explosion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B: Weapon Miss</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C: Weapon Firing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D: Weapon Hit</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

So look at the fireball weapon id of 0x0E000F00

E in hex represents decimal 14 which is the fireball explosion sound effect and thus is placed in the upper two spots of the hex id number set. There is no weapon miss sound effect so these are 0’s. The sound effect for casting a fireball is decimal 15 (the same as the filename) and is the same as F hexadecimal. Thus F is placed in the location reserved for firing weapons. And finally since the fireball has no Hit sound effect this portion of the hex id number set is also left as 0’s.
Monster Sound ID’s (files 61-??)

0xAABBCD
A: Attack Sound
B: Ouch Sound
C: Death Sound
D: Spawn Sound

The monster sound id’s are composed in the same manner as the weapon id’s.

Door Sound ID’s
0x01 Dungeon Door
0x02 Metal Door
0x04 Normal Door
0x08 Jail Cell Door
0x10 Portcullis Gate

Level specific Data:

*obj.dat: This data file contains master list information for the game objects which are specific to a particular level. The format is specified by the type of object. Level specific objects are objects that can not be moved from one level to the next. The data is placed in a specific file, however this file can be loaded by multiple levels.

Static: objects that do not move

# of Different objects (only listed once)
Collision Flag Type
Collision Box Adjust (Left, Top, Right, Bottom)
Image Rect (Left, Top, Right(width), Bottom(height))
Name

Door: these are doors.

# of Different objects (only listed once)
Collision Flag Type
Collision Box Adjust (Left, Top, Right, Bottom)
Lock Value (should be 0)
Image Rect Open (unused, Left, Top, Right(width), Bottom(height))
Image Rect Closed Door (Left, Top, Right(width), Bottom(height))
Sound Id
Name of door.

*Tile.dat: These are editor specific data files which define the properties of the tiles used for the level. This data file is not used by the engine since this tile data is exported as part of the LEV file. Each Tile Image file must be 480 pixels wide using 48 x 48 pixel tiles. The data file has the tiles laid out going
across the width of the image first and then down. Each tile has a hexadecimal value associated with it. This value is comprised of 2 numbers added together. The lower digits are the collision type information (can walk on, can see over,...) and the top digits are reserved for the type of sound to play. The Tile sound id values are 1: dirt, 2: gravel, 3: leaves, 4: mud, 5: wood, 6: marble/stone. So a tile value of 0x20005 would mean the tile can be walked on and can be seen/shot over and the 2 portion indicates that the gravel sound effect will be played when the player steps on that type of tile.

.BMP

The .bmp files, bitmap files, store the graphical image data for the artwork used in the game. This includes tiles, objects, paperdolls, animation, screen art, and backgrounds. All images are saved in 24 bit color. The width of any image can not exceed 800 pixels; there is no restriction on the height.

.WAV

The .wav files, wave files, store the sound data for the sound effects used in the game. The sound files have been recorded in two different settings. Ambient sounds were recorded at 22kHz 8-bit stereo, while sound effects were recorded at 22kHz -16 bit mono.

Directory Structure

..\Arkoss
..\..\Current
..\..\Master
   Masterobj.dat
   NPCList.dat
   NPCScript.dat
   *.lev
   *.*.obj.dat
..\..\Art
   *.bmp
..\..\Sound
   *.wav
..\..\SaveGame1
..\..\SaveGame2
..\..\SaveGame3
..\..\SaveGame4