Engine22 tools - World Editor

1. Background Story:

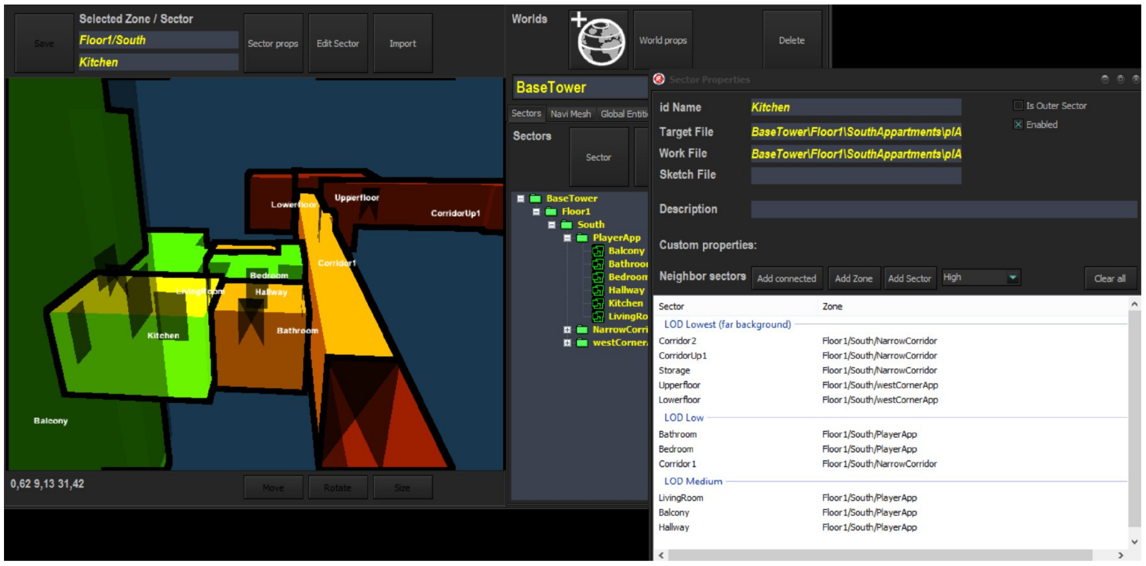
http://tower22.blogspot.nl/2015/07/play-with-duplo.html

2. Summary:

Engine22 does "mapping" –making the game world- on two levels:

- A global World Map
  - Not necessarily made by (3D) artists
  - Does not contain any geometry details, lights, scripts or entities
  - Just primitives (such as cubes) that form sector boundaries
  - Mainly meant to make large parts of the world quickly - "design phase"
  - Includes the connections between sectors

- Detailed pieces of the World Map, called Sectors
  - Actual game content; geometry, lights, entities, scripts, physics, audio, …
  - Maps are split in sectors; typically rooms, corridors, areas, or a X^2 meters patch
  - Engine22 only draws the sectors that are in sight
  - Engine22 only loads "nearby" sectors (roaming) as told in the World Editor (see screenshot below)



Selected Zone / Sector: Floor1/South, Kitchen

0,62 9,13 31,42

BaseTower

id Name	Kitchen
Target File	BaseTower\Floor1\SouthAppartments\plA
Work File	BaseTower\Floor1\SouthAppartments\plA
Sketch File	
Description	

☐ Is Outer Sector
☒ Enabled

Custom properties:

Neighbor sectors: High

Sector	Zone
LOD Lowest (far background)	
Corridor2	Floor1/South/NarrowCorridor
CorridorUp1	Floor1/South/NarrowCorridor
Storage	Floor1/South/NarrowCorridor
Upperfloor	Floor1/South/westCornerApp
Lowerfloor	Floor1/South/westCornerApp
LOD Low	
Bathroom	Floor1/South/PlayerApp
Bedroom	Floor1/South/PlayerApp
Corridor1	Floor1/South/NarrowCorridor
LOD Medium	
LivingRoom	Floor1/South/PlayerApp
Balcony	Floor1/South/PlayerApp
Hallway	Floor1/South/PlayerApp

Screenshot of "WorldEd"

3. Motivation:

Many games use large maps –called "Worlds" in Engine22- nowadays. Some type of (roaming) games even only use a single huge world. Grand Theft Auto, Fallout, Tower22…

This introduces several problems. For one thing, such worlds are so large that it cannot be loaded or rendered all at once. We need to split up into smaller parts –called "Sectors" in Engine22. Second, in terms of workflow, such a huge world can't be made at once, and in case a team of multiple artists is making the "Sectors", there needs to be a referene, a red line.

WorldEd (World Editor) has been made to give a helping hand here. It's not really a tool for production artists, but merely for those who do the global map design; the Designer(s) make a global a layout of the world made in cubes and primitive shapes. The environment Artist(s) then work out the sectors as planned by the Designer(s).

In addition, WorldEd contains tools to plot routes, lines, reference photo's and description labels in the global 3D world. This won't appear in the actual game, but it can help designers to create their worlds, as well as showing the plans to other departments of the team (concept artists, environment artists, audio composers, writers, …).

4. The Connection between WorldEd and MapEd:

The arguments above make sense, but how to make a logical click between the design and actual mapping phase? One can make doodles on paper, prototype 3D models or description documents. But none of that really falls under any typical game-engine tool.

Engine22 ensures WorldEd will be used by properly, by enforcing the link; you simply can't make sectors without a global world. The files produces by WorldEd, are loaded into your game, as they tell:

- Boundaries of sectors → Needed to quickly pinpoint locations (where am I?)–more about this later
- Load-lists → So the engine knows what other sectors to load when standing in sector
- Portals between sectors → So the engine knows what (not) to draw at any given point
- Global properties → For custom usage. Loading screens, intro texts, level-rules, et cetera

We felt we shouldn't bother an environment artist with such things, so that helped making a split between global World editing and Sector editing. Environment artists should do what they are best at: making cool environments. The other details are for the game-designers and eventually programmers.

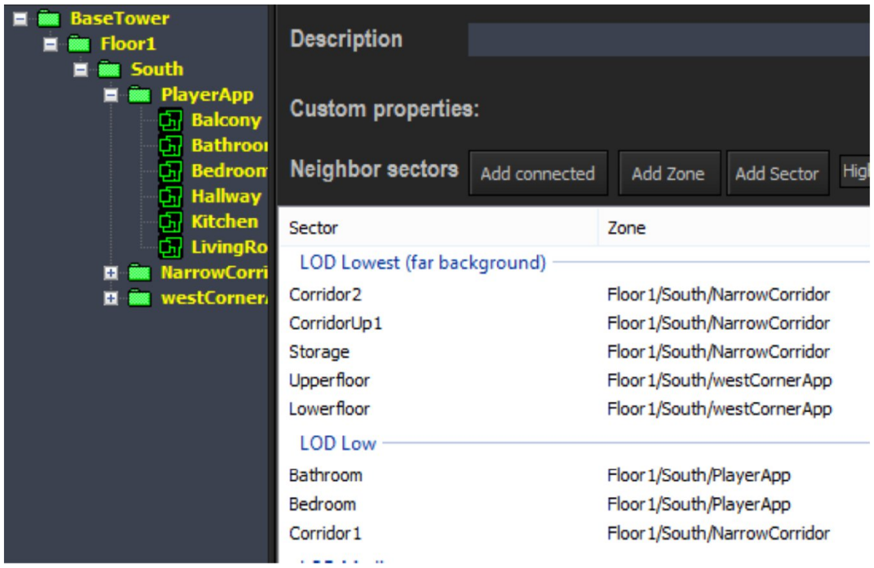
5. Workflow:

Design phase (designers)

1. Split up the entire game into 1 or more worlds.
   We could go for a single huge roaming world, or we split up in more traditional "levels".
2. In a 3D modeller program, make your World(s)
   We can make 1 huge file per World, or use multiple files. We export to OBJ or LWO (Lightwave) for now. Most programs support OBJ (Blender, Max, Maya, Lightwave, ...)

3. Use primitives such as cubes to *lay out the boundaries of each sector*. A cubic shaped room would be a single cube. An L shaped room would be made out of 2 or 3 cubes. Less is better here, but try to avoid overlaps with neighbour sectors or covering empty "vacuum" parts of the world. *The boundaries need to be somewhat tight, as the artists will use this to model within later on*!
4. Name the primitives as follow:
   <PrimitiveType><primitiveNumber>.<(local)zone><SectorName>
   AB1.Tower22/Floor1/South/Apartment2/Kitchen
   AB2.Tower22/Floor1/South/Apartment2/Kitchen
   AB1.Tower22/Floor1/South/Apartment2/Bathroom
   "AB" stands for "Axis Aligned Bounding Box" (thus a non-rotated cube)

   So the importer knows for each polygon to what primitive, and what sector it belongs. The same naming methods will re-appear in making Sectors and Objects as well by the way.
5. When importing such a file, the named paths and sectors will be generated (if not done before) into the "Tree":



Sector	Zone
LOD Lowest (far background)	
Corridor2	Floor1/South/NarrowCorridor
CorridorUp1	Floor1/South/NarrowCorridor
Storage	Floor1/South/NarrowCorridor
Upperfloor	Floor1/South/westCornerApp
Lowerfloor	Floor1/South/westCornerApp
LOD Low	
Bathroom	Floor1/South/PlayerApp
Bedroom	Floor1/South/PlayerApp
Corridor1	Floor1/South/NarrowCorridor

   Note that we can use multiple files for the world. Tower22 for example could have a file for each floor. In that case the "zone" folder can be localized. For example, if the imported sector primitives were named "ABx.PlayerApp/Bedroom", and being imported into zone "BaseTower/Floor1/South/", the full sector path becomes "BaseTower/Floor1/South/PlayerApp/Bedroom".

6. Inside the same model, we can model portals as quads; a portal connects 2 sectors. So for example, a door/window/hole/opening between 2 sectors. The polygons that make up a portal shall be named simply"PORTAL".

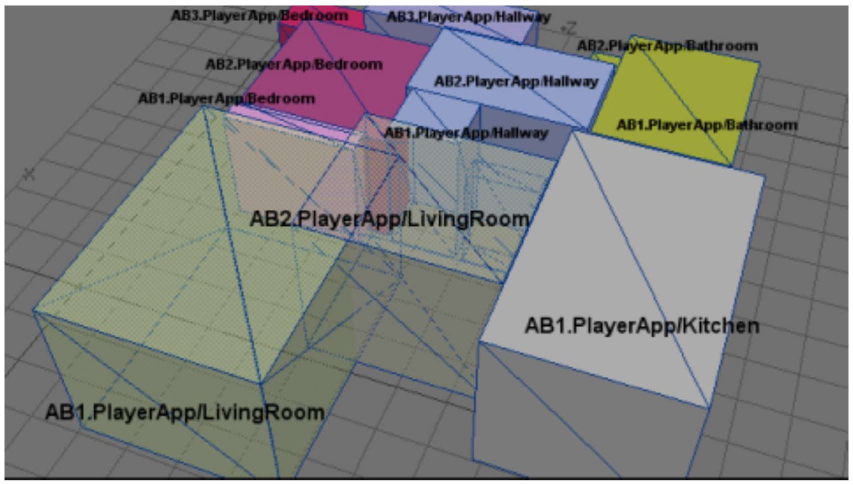
7. For each imported sector, we fill in:
    a. Which other sectors should be loaded into the memory, and at which level of detail (LOD), see list in the picture above
    b. Other global properties, such as a description or game-specific custom properties.
8. Save the World file

Sector mapping – Detail phase (artists)

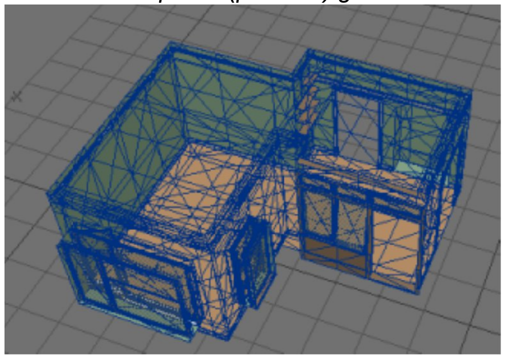
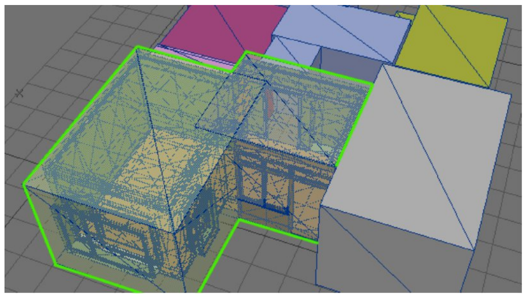
1. Use the global worldfile (or a piece of it, depending on what you'll be making) as a reference; the *detailed geometry must be modelled WITHIN the primitive boundaries* (anything outside will be clipped away, or belongs to the neighbour sector if there is one).

    If the global boundaries do not fit (too small, too big, different shape), we'll have to adjust the global World file(s) and reimport them as well!

    Note that you can make multiple sectors in a single file. In fact, if you prefer, you can model the entire detailed world into a single file as well.



Here an example of (piece of) global World model

And here the "LivingRoom" in detail – the actual geometry as shown in the game. The detailed geometry is covered by the global World boundaries.

2. For modelling how-to information, see the "Sector Modelling" document.

3. Save your files as OBJ or LWO (Lightwave). Possibly other formats like FBX or Collada may follow, but we chose to start with OBJ as this is a simply and widely (properly) implemented format.

4. Start MapEd (Sector Map Editor), or WorldEd and select the zone you'd like to make imports and launch MapEd from there.
5. In MapEd, move the camera and go stand inside the specific sector you'd like to import.
6. Click the "Import" function and select your file

7. All polygons that fall within the current Sector boundaries will be imported.

Now you understand why those boundaries have to fit. And not only for the modelling process, also the engine uses it for locating purposes (pathfinding, insertion of entities, checking where we are, what to draw, et cetera).

6. Conclusion

So, to conclude, as a designer remember you'll have to provide the artists a global model of the world. No details required, yet the boundaries should represent the shapes of your rooms, corridors, area's, et cetera. The "WorldEd" program can be used (or make your own program, or just edit text files) to create World files & to visualize your world globally, which can make it easier to show & explain to others.

As an artist, remember you'll have to work with the given boundaries. They will tell you the global shapes of the environments, and the placement relative to other neighbour sectors. In other words, *draw within the lines!*