An Automated Sprite Rendering System using Blender

Kenan Bölükbaşı

2014-04-03 Thu

An Automated Sprite Rendering System using Blender

The point of the talk is not really to introduce the system, but to show how easily such a system can be put together on top of F/LOSS stack in a short amount of time, by example.

What is it?

- It is an in-house automated sprite rendering system using Blender. No name yet.
- It makes the job of our designers much easier by automating everything possible and by making visual revisions on all graphics easier and more interactive.
- Works completely on Blender, except cli written as shell script.
- Written in Python and Bash. ~500 loc.
- The base system is written in 1 month by one person, me.
- The atlas generator is written by İsmail Döner, who also wrote the whole in-house game engine.
- Not released yet. Plan is to release it as F/LOSS at some point.
- We would love to hear if there is interest in such software.

It briefly

- Uses 3D models generated in the design department as Collada exchange files.
- Afterwards the system automatically:
  - imports the model
  - adds
    - shading,
    - texturing,
    - lighting,
    - cameras,
    - environment,
    - effects
  - adjust settings and renders sprite
  - generates atlas
  - compiles/runs the game demo

Who am I?

- Kenan Bölükbaşı
- İstanbul, Turkey.
- CG Generalist
- Hobbyist programmer and designer
- Fully F/LOSS software stack for 6 years as a professional
- Lead of Graphics Department at Ekseriya Studios.
- I heavily use Blender, Gimp, Inkscape and ImageMagick on Arch Linux.

Who are we?

- Ekseriya Studios
- We are a game development studio located in İstanbul.
- Completely F/LOSS technology stack on development.
- Fully in-house production infrastructure and game-engine.
- Small, very ambitious team.
- 7 people working on the project. (development: 3 & graphics: 4)

What is the project?

- An online multi-player cross-platform (mobile & desktop) strategy game.
- Project uses sprite graphics.
- An interactive development process was requested.

Why such system?

- Process was hard and problematic to maintain manually.
  - Too many assets, easy to make mistakes.
  - Varying settings:
    - Grid size
    - Directions
  - So much repetitive work.
  - Concept revisions are very common.
  - Materials and lighting are due to change.
  - Models should be rendered with varying number of directions.
  - Sprites should be managed properly for atlas generation.

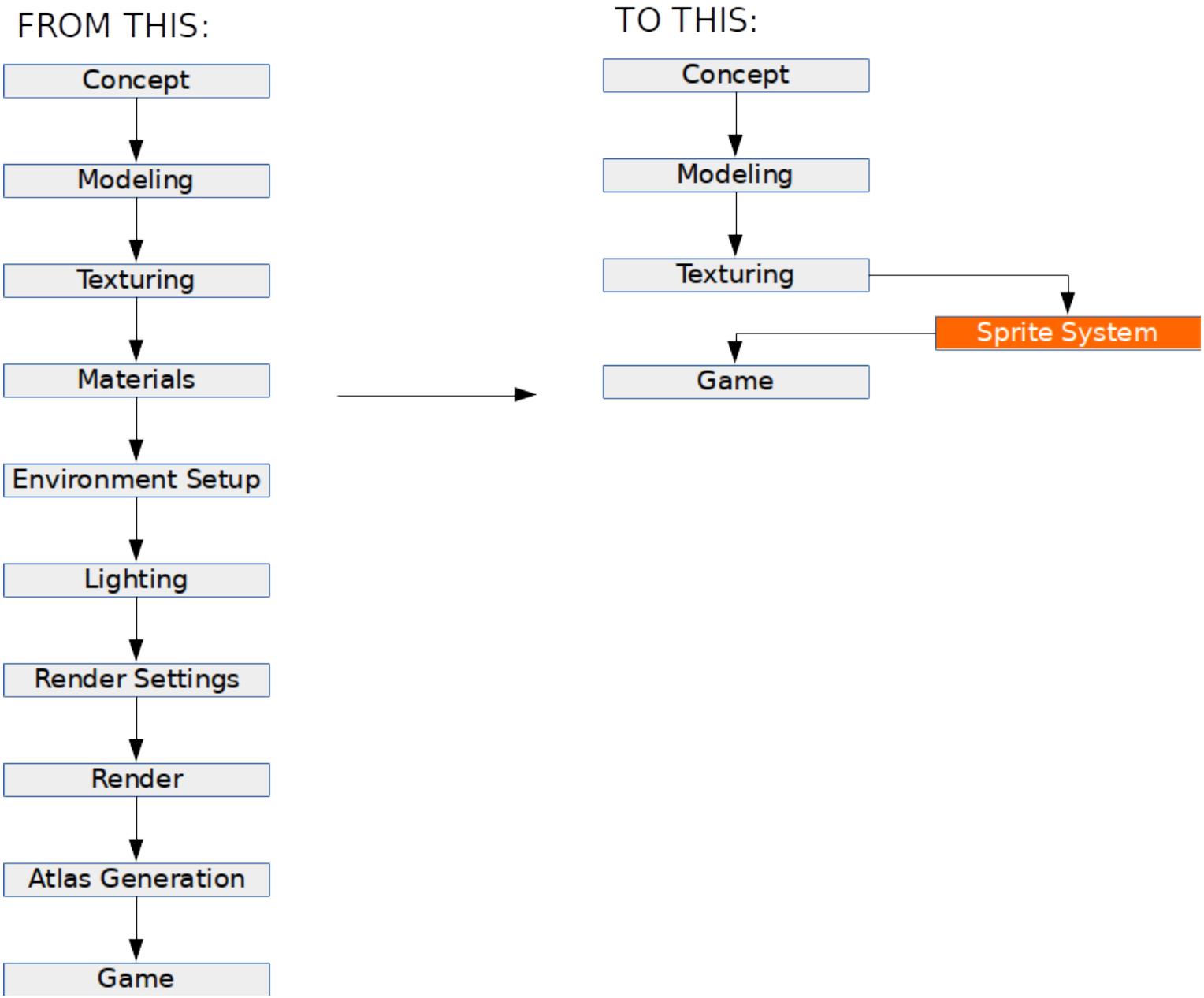
To sum up

- So much repetitive, variation.
- Simply library linking didn't solve many problems in this case.
- Almost everything will drastically change.
- Almost everything will drastically change again.
- We need to keep up with requests and constantly push new graphics in-game.

I realized

- I need some kind of a system to optimize labor.
- I need a centralized mechanism of control.
- I never done Python before, but it is easy!

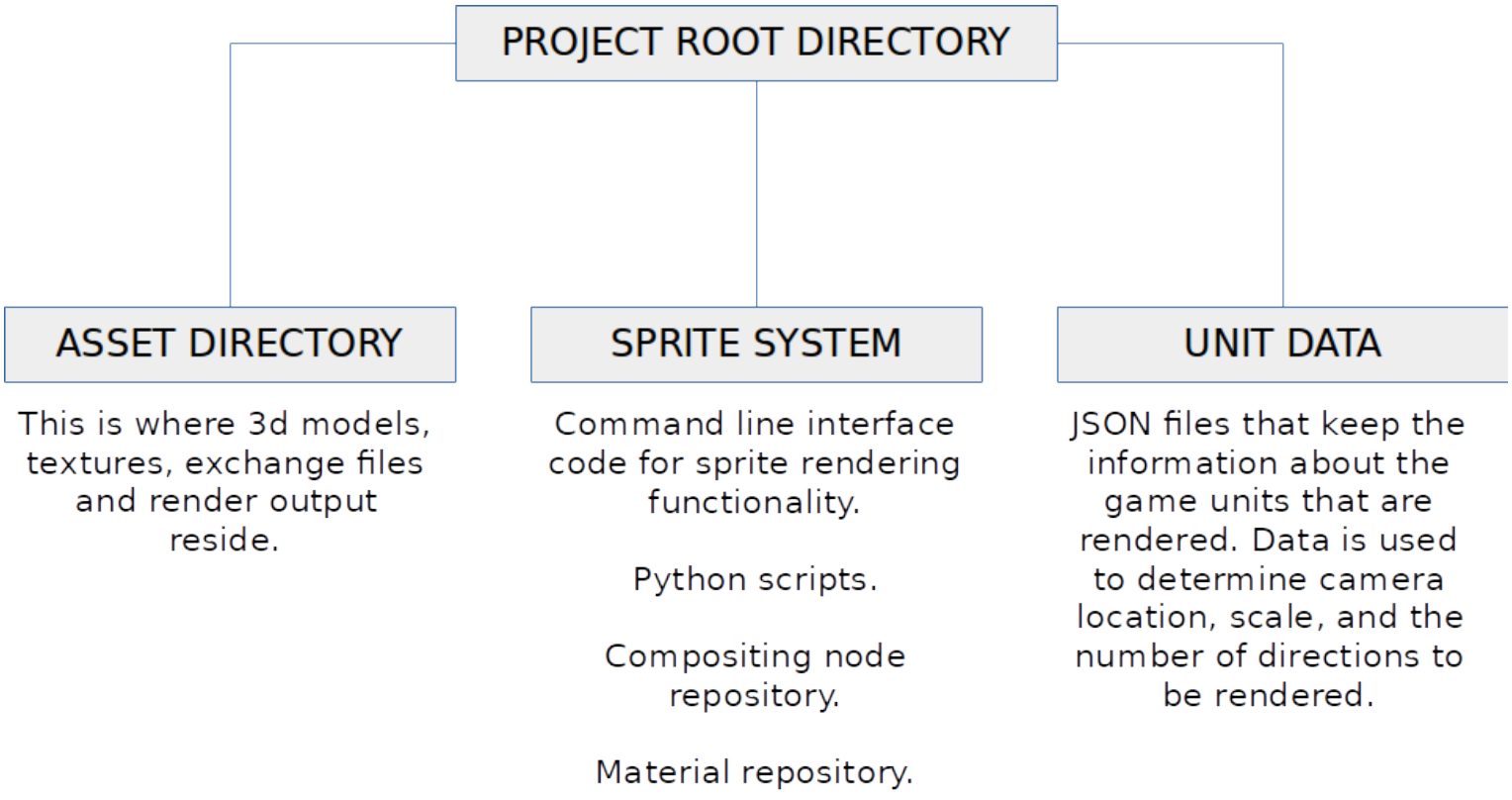
Aim


FROM THIS:
Concept
Modeling
Texturing
Materials
Environment Setup
Lighting
Render Settings
Render
Atlas Generation
Game
TO THIS:
Concept
Modeling
Texturing
Sprite System
Game

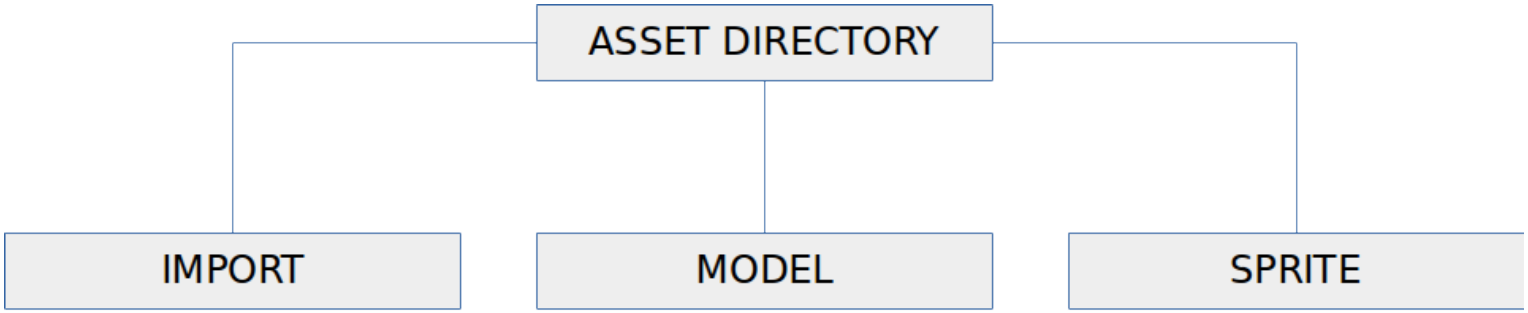
Later Additions

- Framestep
- Proper handling of imports for easy modification in Blender.
- Collada support to make it modeling software agnostic.
- Several Animation Directions
- Explosion Particles
- VFX
- Compositing
- Atlas Mipmap
- GPU Supported Texture Compression
- Cygwin Compability
- Distributed rendering, almost.

How: Project Root Directory



How: Asset Directory



Collada exchange files and textures.

Blend files consisting of lots of scenes, each containing a specific animation at a specific level of a game unit.

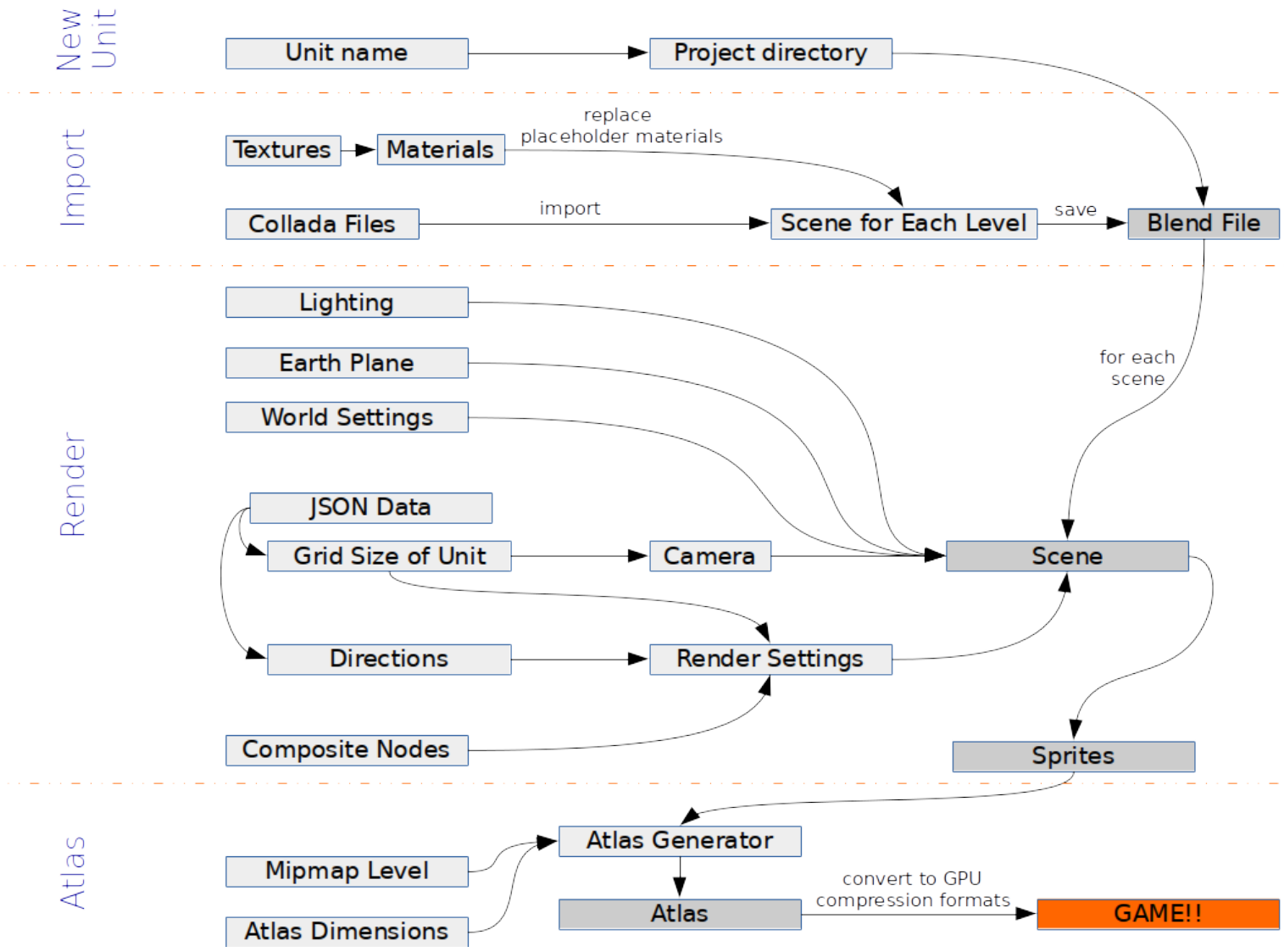
Symlinks to material and compositing node repositories.

Resulting sprites of unit animations as PNG sequences.

Static sprites e.g. game map, map props, environment elements, UI assets.

Supported atlas conversions of all assets, e.g. PNG, DDS, PVR.

How: The System



Example

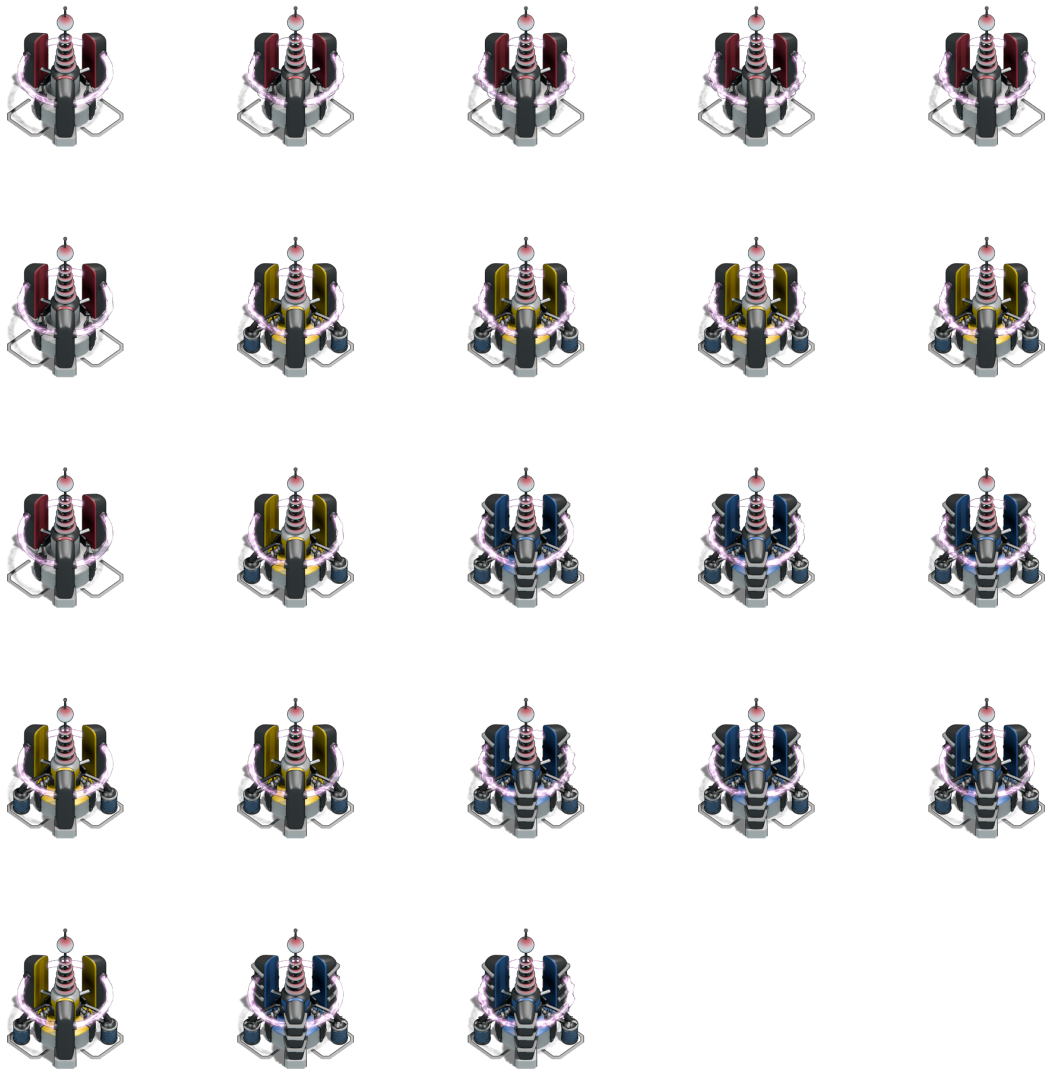
$ sprite -nira beamer helix air-missiles hadron-collider tesla-coil battle-cruiser marauder

Results

Results

Results

Results



Why using F/LOSS?

You know why.

Contact

- Twitter: @kenanbolukbasi
- Mail: kenanbolukbasi@gmail.com
- Website: kenanb.com
- github: github.com/kenanb
- freenode: kenanb
- Ekseriya: ekseriya.com

Thanks

- Thanks to F/LOSS community.
- Special thanks to Libre Graphics Community.
- Even more special thanks to Blender Foundation and community.
- Thanks to Emacs, Org-Mode, Beamer, LibreOffice, GIMP for presentation tools.