Tools Programming

Asset integration pipeline 1

Class 6

Index

- State of the art
- Asset integration pipeline
  - Structure
  - Workflow
  - Components

- Exporting asset data
  - Read data from 3DSMax
  - Write data to export
  - Export meshes

State of the art

Asset integration is built-in present in all of the commercial engines out in the market.

Built-in features such as:

- Input read FBX
- Map editor
- Asset configuration tools
- Import/Export settings.
- ...

State of the art

- The dark age: (1980-1996)
  - Videogame references: platforms, roguelikes (sega, atari…)
  - Engines: Proprietary, each game used his own engine.
  - Internet (WWW): 1995
  - Almost no documentation on videogames
  - **Private industry.**

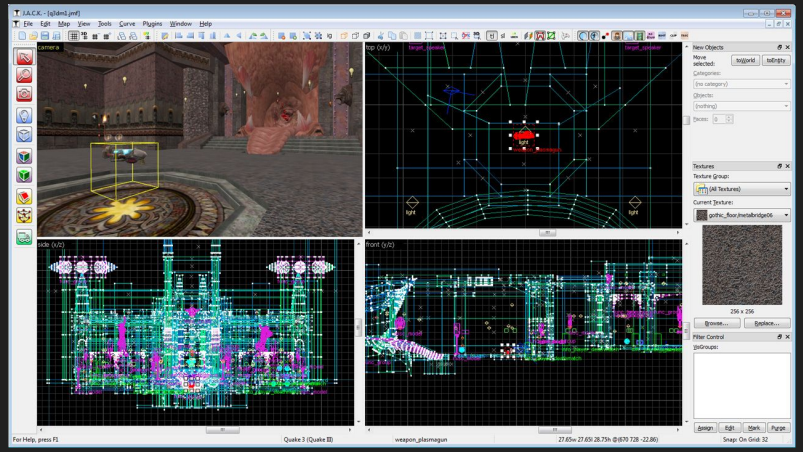
Link: http://fabiensanglard.net/

State of the art

Link: https://www.youtube.com/watch?v=ZWQ0591PAxM

State of the art

- The launch: (1996-2008)
  - Videogames: Quake, Half life...
  - Engine: Valve editor (Hammer), id tech, Unity launch, Unreal engine
  - Modelling tools included within the editors (bsp).
  - The industry begins to grow a **community** around the games itself.

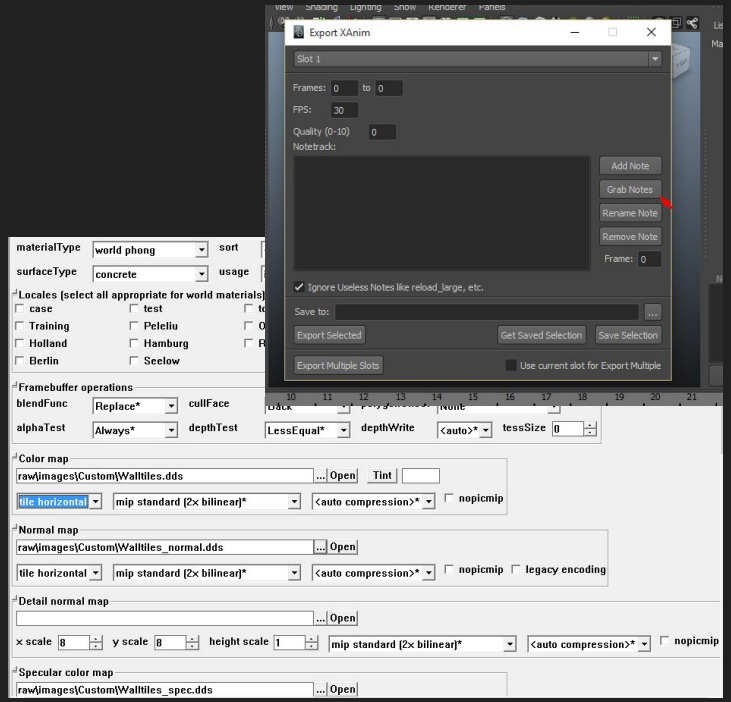


For the first time community grows around engine tools

VIDEOGAME MODDING.

Case Study: Quake Toolset

- Export:
  - Export plugin scripts for:
    - Exporting models
    - Exporting animations
- Import:
  - Asset Manager: External modding tool, the same toolset that developers used to develop the videogame during production.
    - Generate encrypted assets by providing raw source assets.
    - Allow us to adjust and set parameters to our datasets.



Link: https://znation.nl/tutorials/cod2/MCh2207Cz_Tutorial/Modeling_Part2.htm
Link: https://github.com/promod/CoD4-Mod-Tools

Case Strudy: Quake Toolset

- GTK Radiant edition tools for id tech videogames.
- Open source level design editor created by the community.
- Modding focused tool

Github: https://github.com/TTimo/GtkRadiant

State of the art

- Today: (2008-Now)
  - Videogames: INDIES!!!
  - Engines: Unreal, Unity, Godot….
  - Tools fully integrated within the engine
  - All functionality is public (Free*)
  - Many tutorials and documentation on the net.
  - Indie community becomes solid.
  - Industry is **public**

Case Study: Unity

- Multiplatform engine.
- Contains all tools in a single product pack (except modelling)
- Asset integration can be done instantly without having a wide knowledge of the toolset.
- Huge community support.
- Free to use and distribute.

Class Tip

When building a Custom Engine have a reference engine to focus your development based on that ref.

We will use Unity as reference.

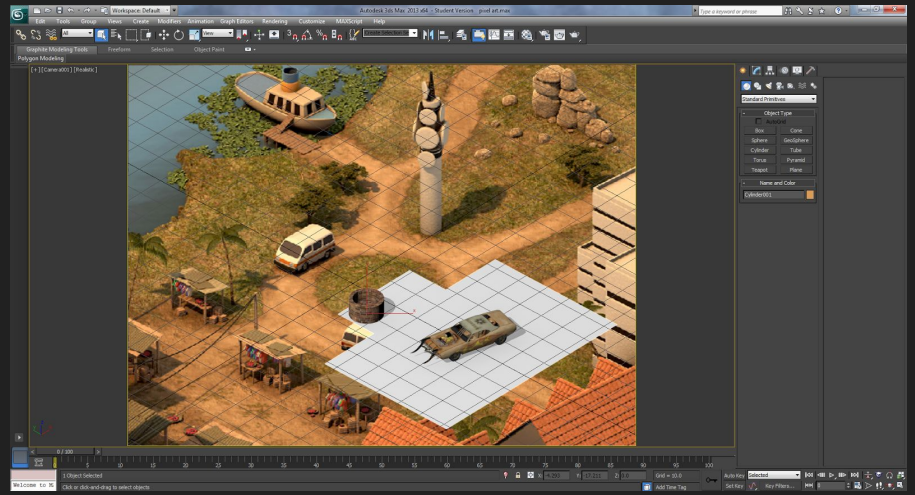
TODAY

Asset Integration Pipeline

3DSMax as Toolset

- We have seen that a scene editor provides:
  - Edition saving (Persistence)
  - Object transform edition (Modification)
  - Object property edition (Modification)
  - Object modelling edition (not always)
  - Utilities to speed up process (not always)

- Every point previously denoted is provided by Max.
- Building our own custom editor in scene is very expensive.
- Loot of debugging needed



3DSMax as Toolset

How do we export all the scene data into our custom engine?

- Maxscript to build an exporting tool
- Artist pipeline work consists in (among others):
  - Generate the concept art.
  - Generate the assets out of concept art (models, textures…)
  - **Export the assets**
  - Lighting setup...

- Tools must be user friendly (Good UI, Good documentation)

Artist will use our tools, not programmers!

Maxscript Structure

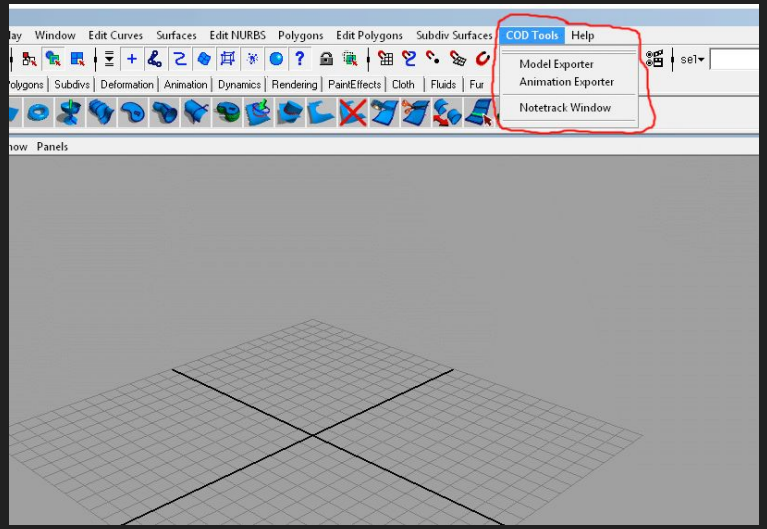
- All maxscript code must be in your project inside Tools folder
- Tools folder should contain the following files
  - MVD Components export tools
  - MVD Geometry export tools
  - MVD Json library exporter
  - MVD Scene exporter
  - MVD Utility tools
  - MVD Utils

QuickAccess

- Tools must be user friendly
- Tools must be very accessible

Solution:

- We can use macroscripts to define a UI entry for our tools
- Should launch on startup!
- Seen on previous class!
- Remember to add documentation to the tools so artist will understand!



Macroscripts

- Macroscripts are scripts that associate functionality with the internal 3DSMax user interface.
- Can also be used as an event handler to process inputs from different menus and action bars.

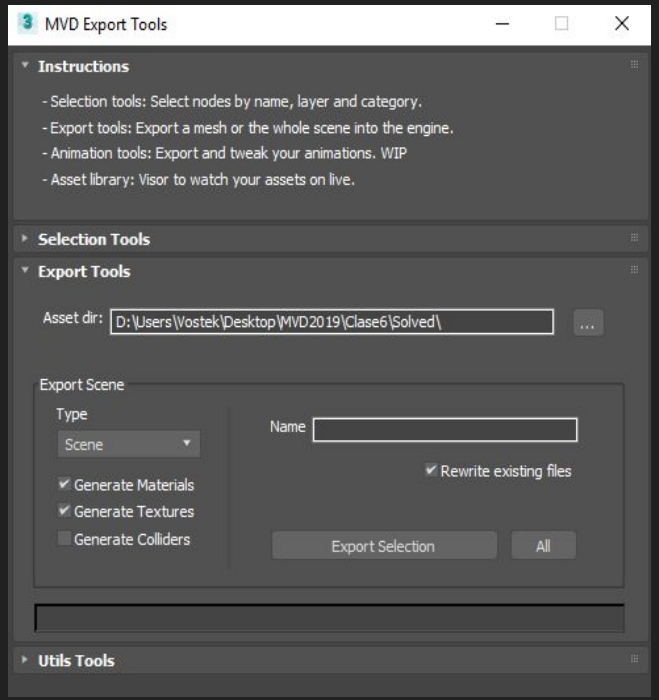
```
-- Remove macro function to avoid stacking up macros on toolbar
function removeMenu m_name =
(
	del_menu = menuMan.findMenu m_name
	if del_menu != undefined do
		menuMan.unregistermenu del_menu
)

macroScript LaunchMenu category:"MVD" --macroscript menu
(
	create_object_test()
)

--Adding menu macro process with menuMan
removeMenu "MVD Tools"
theMainMenu = menuMan.getMainMenuBar() --get the main menu bar
theMenu = menuMan.createMenu "MVD Tools" --create a menu called Forum Help
theSubMenu = menuMan.createSubMenuItem "Launch tool" theMenu --create a
SubMenuItem
theMainMenu.addItem theSubMenu (theMainMenu.numItems()+1) --add the
SubMenu to the Main Menu
theAction = menuMan.createActionItem "LaunchMenu" "MVD" --create an
ActionItem from the MacroScript
theMenu.addItem theAction (theMenu.numItems()+1) --add the ActionItem to the
menu
menuMan.updateMenuBar() --update the menu bar
```

Building the UI

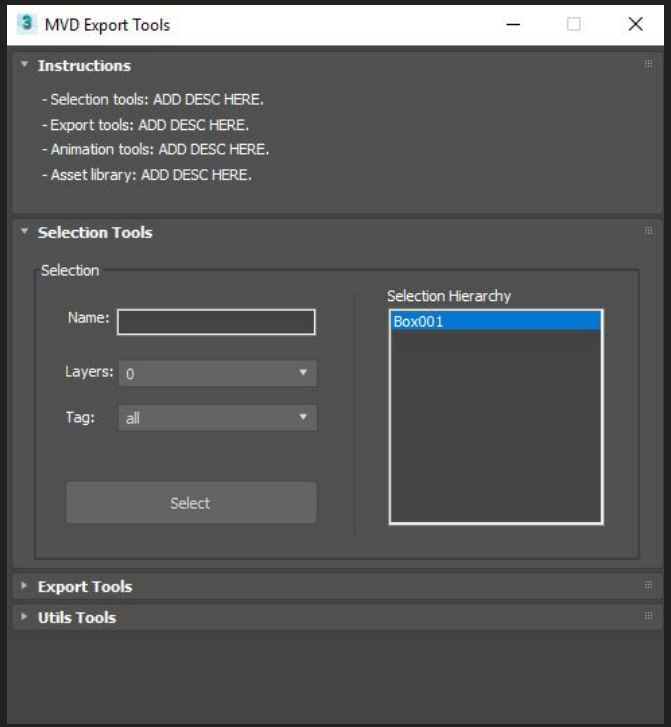
- We can use floated panels to iterate faster
- The UI should have the following structure:
  - Instructions (optional)
  - Utility tools
    - Selection tools
    - Misc
  - Exporting tools
    - Select folder where you want to export the data
    - Export scene, prefab, spline tool
  - Utils tools
    - Export components
    - Generate colliders..



Utilities

- Make our life easier and happier
- Specially focused on avoid repetition.
- We will work with selections as exporting containers
- Case study: Selection tools
  - Input string as name selection
  - Layers list from the scene as filter
  - Tag list from the scene as filter
  - Dropdownlist with filtering results

- Layers: physics distinction
- Tags: category distinction



Sample

- Finish the utility tools by creating:
  - Method to retrieve all the existent layers
  - Method to retrieve all the matches given the selected filters
  - Method to generate a selection of the given objects.

TO-DO

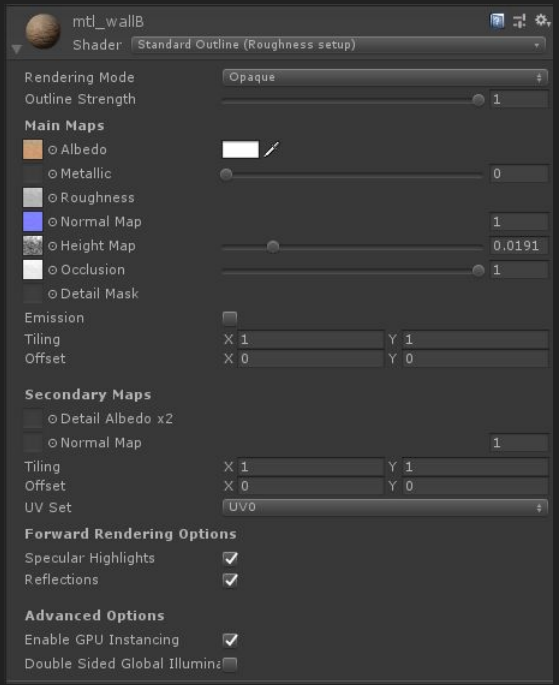
Export geometry

- Geometry data must be exported in external files.
- Files must have a format, in our case .obj
  - Build a function under MVD_Geometry class to export a selected object or the whole scene into a .obj file
  - Allow support to other formats in the future.

TO-DO

Export material data

- We are working under phong
  - Colormap, specular, normal maps..
- Working under pbr:
  - albedo, metallic, normal, roughness, occlusion…
  - 3DSMax doesn't have support for pbr.
  - We determine a series of inputs for each of the maps.



Export material data

- Utils functions
  - Create a function to retrieve materials used by mesh
  - Retrieve all the maps and print them into file
  - Retrieve all maps and export them to proper textures folder under assets directory
  - Retrieve other material information and print it into file
  - Apply support to multimaterial data (in the future)
  - Add other necessary functions to the set.

```
{
        "tech": "pbr.tech",
        "textures": {
                "diffuse": "data/assets/textures/block_teal.tga"
                "specular": "data/assets/textures/block_teal.tga"
        }
}
```