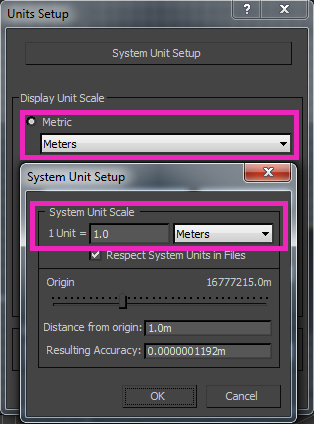
3D MODELLING FOR GAMES

- PRE-EXPORT CHECKS-

* **Have you used your poly budget well? Have you used your material/texture budget well?**
  + Your Art Lead has fought for your budgets, and has given considerable thought to how the budgets are to be divided and allocated. Your Art Lead will be unhappy if you have not used the budgets sensibly. Budgets typically give you an idea of the amount of detail your Art Lead is expecting to see in your assets. For example, a castle might be constructed from a texture with alpha mapped onto a single quad; conversely, you might use a million polys with fifty materials constructed from diffuse colour, specular, normal, and even parallax and incandescence maps. It is not your place to decide – you do not have enough information to see the bigger picture. Work with the budgets you have been allocated.
* **Are you working to a consistent scale?**
  + As a general rule, you should be working in meters. Under **Customize** > **Units Setup**, ensure that you are displaying **Meters**. This information is not exported, but does allow you to measure things correctly. The exported information is stored in **System Unit Setup**. Open this dialogue box and set the **System Unit Scale** to **Meters**. In this way, every asset will be exported to a consistent scale.
* **Are your objects appropriately named?**
  + There is always a vast amount of files to control in the development of a game. Name every file appropriately, adhering to prescribed naming conventions.
* **Are you using only legitimate polys?**
  + Working with quads and tris allows you to control your geometry in a neat and efficient way, and with minimal fuss in the game engine. Use **Selection > By Numeric** to identify polys which have more than four sides, or less than three sides.
* **Have you reset the xForm?**
  + The xForm contains invisible transform information (translation, scale, rotation). This information builds up the more you detach and attach meshes, and can have catastrophic effects on a mesh once exported to a game engine. Reset the xForm in the **Utilities** tab, and then collapse your stack and check for adverse effects.
* **Have you reset the pivot point?**
  + Your Art Lead will tell you what is required from the pivot point. The function of the pivot point varies from game to game, asset to asset, and engine to engine. For example, when exporting from Max to Unity, the y axis should be rotated to the up position. When constructing game props, the pivot point is typically set to the centre of the base of the object, and is used when transforming props in the game engine.
* **Have you tidied your scene?**
  + By default, everything in your scene will be exported. If you are using a three-point lighting rig to light your assets for render, ensure that the lights are removed prior to export. Keep your scenes tidy as you work, and ensure they are spotless before you export. If in doubt, **Unhide All** and open the item lister.
* **Have you tidied your material editor?**
  + Not only will you have errors on export if your file is attempting to find obsolete textures, but you run the risk of exporting rogue files. Furthermore, consider the poor artist who has been asked to go into your file to make changes. They will not be pleased if they have to spend thirty minutes searching for your textures, or trying to figure out which materials are referenced by the objects in the scene. Open the Material Editor and drag clean materials onto obsolete materials, or drag empty map buttons onto obsolete texture maps. Remember to name all active materials.
* **Have you set the correct path for your textures?**
  + max_dropper.jpgAll textures referenced by a scene should be kept together in a folder. It is then a simple task setting up a path to this folder, and the Material Editor can be repopulated in a single action. Should you ever need to repopulate the Material Editor by hand, you can use the dropper to pick materials from the scene.
* **Are all textures powers of two?**
  + This is the common language used by the software and hardware you will be working with. By speaking their language, your files will be efficient. MIPMAPs are also generated using the power of two rule.
* **Are all textures appropriately named?**
  + Your Art Lead will provide you with naming conventions. These vary, along with file formats, from company to company, and will be documented in the master Design Document or, occasionally, a Style Guide. For the duration of your degree, unless otherwise stated, I would like you to populate your Material Editor with **tga** files. You should name your files using **lower case** letters, and with **underscores** replacing spaces. You are welcome to include a number to denote a variation. Your suffix should denote the file’s function: I suggest you use **col** for diffuse colour, **spec** for specular, and **norm** for normal. As such, you might create a file named **brick\_wall\_01\_col.tga**

You should also keep a master **psd** file, containing organised folders of **unflattened** layers. This will allow you, or another team member, to effortlessly make changes to your maps at any time.

* **Have you collapsed your modifier stack?**
  + Your Art Lead might ask you to export a file with a set of modifiers arranged in an uncollapsed stack. These modifiers, often proprietary, are accessed by the code and may have all manner of functions assigned to them. However, unless otherwise instructed, you should collapse your modifier stack prior to export.
* **Are your verts appropriately welded?**
  + Think of your meshes as hollow, water-tight skins. Vertices may also be used for all manner of functions; they may, for example, be coloured in order to denote a certain property, such as a physical weight or elasticity. They contain normal (vector) information which is averaged from the connected edges, and is used in the calculation of smoothing information. Use **xView** to check for **Overlapping Vertices**. Use **Weld** to eliminate overlapping verts.

* **Are your polys facing in the correct direction?**
  + Is Max lying to you? All might seem well in the Max viewports; be sure to turn the **Backface Culling** on under the **Display** tab. You can also use **xView** to check **Face Orientation.** Fix backfacing polys using the **Flip** button in the poly sub-object menu.
* **Are there any co-planar polys?**
  + When two or more polys occupy the same surface area, they battle for dominance in the rendering engine. This results in a flickering effect called z-fightingwhich is highly undesirable. You are also likely to weld these polys together as you perform your welding routine, making them difficult to spot, and sometimes very difficult to clean up. Use **xView** to check for **Overlapping Faces**.
* **Have you ensured that all UV’s are contained within the 0-1 boundaries?**
  + UV’s mapped outside of the 0-1 boundaries need to be dealt with with specific code, which will chop up and reorganize these UV’s. This is highly inefficient, and unacceptable if this code is not in place. The Code Lead will not thank you if you extend beyond these boundaries by even a single pixel. As a rule of thumb, always leave a border of a couple of pixels inside the perimeter to ensure that your UV’s are legal, and to allow for the pixel bleed which occurs as textures are downsampled in the engine.
* **Have you maximized the use of your texture page?**
  + When laying out your UV’s, your goal is to maximize use of the texture page. In this way, you will get the most amount of detail from the available pixels. Avoid very small, or very large, UV islands where possible. Scale all UV’s uniformly to maintain a consistent texel (textured pixel) resolution.
* **Are your UV faces discrete?**
  + Large environment assets such as high-poly buildings and terrain are typically coated in a light map. As such, a separate set of UV’s are generated for laying out the components of the light map. However, small and non-environment assets will use the base set of UV’s to lay out baked textures such as AO maps and normal maps. As such, the UV faces must not overlap. You can use **xView** to check for **Overlapping UVW Faces**.
* **Are your edges well-turned?**
  + When arranging the interior edges (those that do not define the shape of your object), the rule of thumb is to turn these edges such that they travel the shortest distance. In this way, your topology is neat and efficient. (And neat and efficient topology is your goal!) You must, though, use your own judgement.
  + A game engine will interpret your meshes as triangles. Where warped quads persist, you will need to triangulate the quads to control the form. Similarly, you will need to triangulate quads that contain t-junctions, otherwise you run the risk of presenting extremely thin triangles (referred to as slivers) which buckle when rendered, or even triangles with no surface area. The **T-Vertices** xView check is unreliable and you must find t-junctions by eye. Triangles with no surface area are identified by the **Multiple Edges** xView check. Where these persist, you will need to delete the neighbouring faces and then reconstruct them.
* **Are your smoothing groups working as intended?**
  + Turn your textures off and change the viewport mode to **Shaded** and check your mesh in its naked, solid form. In many cases, you can rely on Max’s **Clear All** and **Auto Smooth** buttons to assign your smoothing groups. Occasionally, you will need to adjust the angle tolerance; on rarer occasions, you will have to go in and manually assign smoothing groups.