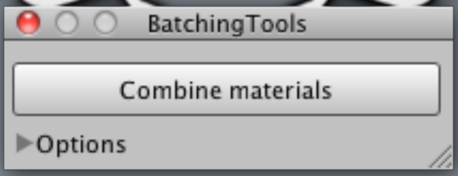
BATCHING TOOLS



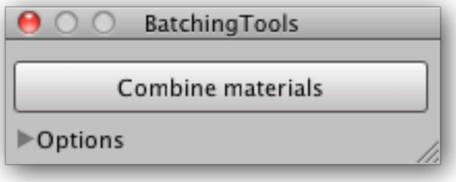
USER MANUAL v1.4

-Table of contents-

Introduction

Hi, thanks for your interest in Batching tools !

Batching tools is a Unity Editor script that helps you create shared materials for use with Unity built-in batching system, or with BTStatic, a custom batching method.



Batching is the process of combining draw calls. Using it can make your games faster and more energy - efficient.
To use batching, you should create shared materials i.e. a material that is shared by more than one objects.
Batching tools helps you to create those shared materials with one click.

One click functionality is the strong point of Batching Tools. Batching tools is a productivity tool. It helps you to develop fast games, fast. A lot of operations that otherwise would require jumping to other applications and doing a bunch of work there, are done in the Unity Editor, with one click.

Batching Tools does a lot of 3d modeling, uv mapping, texture atlasing behind the scenes. It currently supports 16 shaders, lightmaps, saves automatically the prefabs in the project, allows texture export and mesh export for further processing.

It also provides a custom batching mode, BTStatic, that batches non-moving gameobjects, available for every Unity license. BTStatic can work as a substitute for Unity built-in static batching in Unity Free or Unity Mobile Basic. It can also work in parallel with Unity built-in static batching for even faster results.

You should note that batching is not a magic thing, be sure to check the guidelines for batching in the Unity site and the rest of the documentation before you purchase this product. Having said that, if you plan to use batching, this script can prove very helpful.

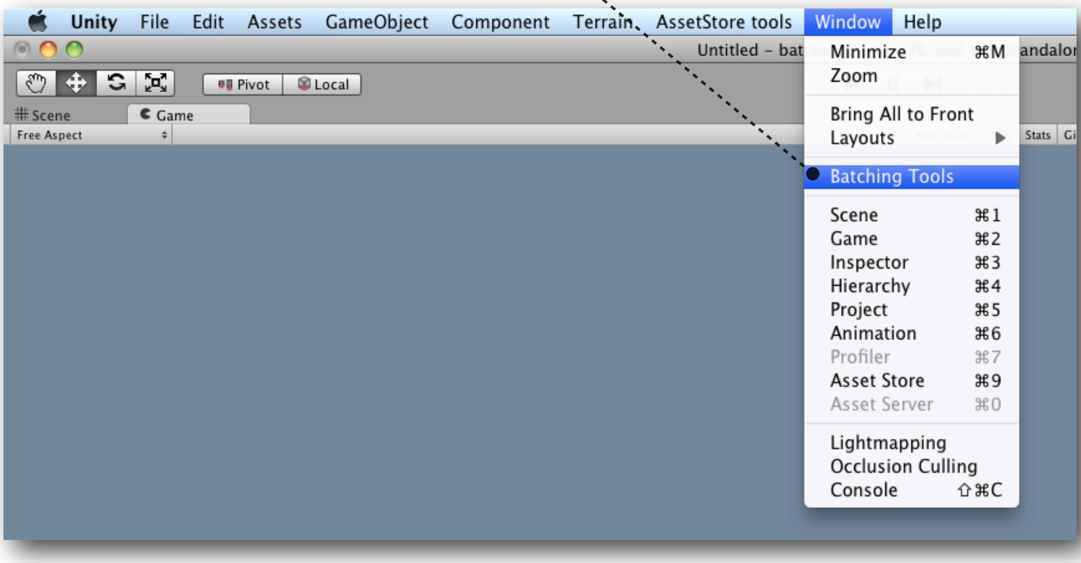
Batching Tools is available at the Unity Asset Store.

There is a discussion thread on Batching Tools at the Unity Forums :
http://forum.unity3d.com/threads/113508-Batching-Tools
Video tutorial :
http://youtu.be/HAt8W_MVG5Y

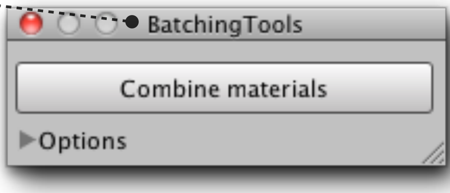
Feel free to post questions and suggestions there. A big part of the current Batching Tools functionality is the result of people’s requests and suggestions, your opinion can help shape a better tool.

Launching the script

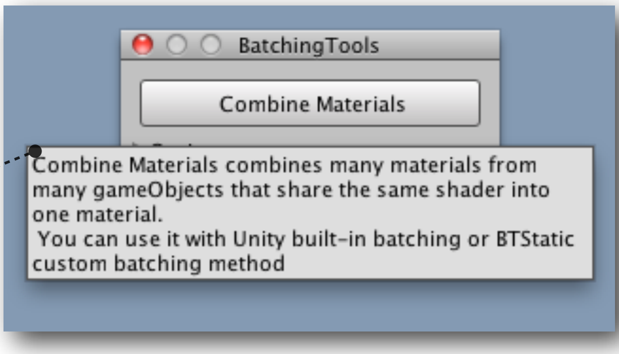
You can launch Batching tools by selecting
Menu -> Window -> Batching Tools.



Once you have launched
it, the following window
appears.



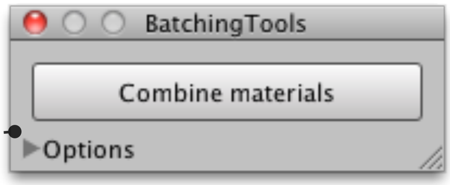
The combine materials
button combines the
selected materials. Try to
hover the mouse cursor
above the gui, you will
notice that a **tooltip help**
appears. You will see
there a brief explanation
on what each gui element
does.



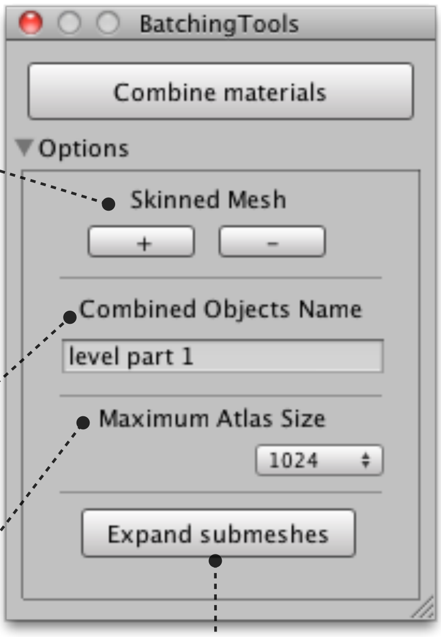
Combine Materials combines many materials from many gameObjects that share the same shader into one material.
You can use it with Unity built-in batching or BTStatic custom batching method

Options at a Glance

Click the little triangle on the left of the Options to open them



Add / Remove to Skinned Mesh
Adds or removes meshes from the skinned mesh to be created (see more in the BTDynamic section)



Combined Objects Name
You can name the objects that you create as you wish, by filling this field before you combine the objects. This helps to organize better the assets in your project. Just fill in something meaningful instead of “combined objects” which is there by default

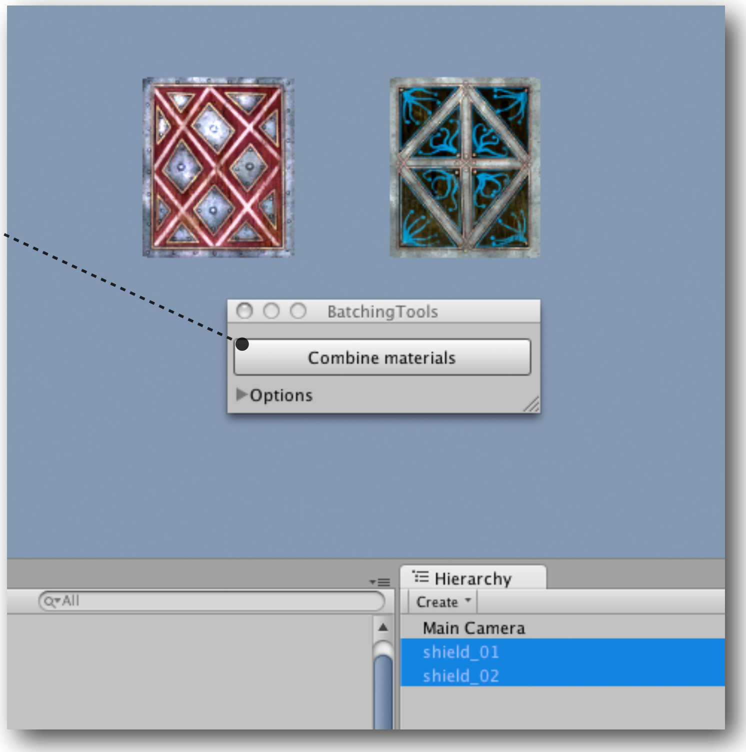
Maximum Atlas Size
The maximum size of the width or the height of the new texture atlas.

Expand submeshes
In unity editor, a mesh with submeshes is represented as a gameObject with one mesh renderer and many materials. To batch them, you should first “expand” them, by selecting them and clicking this button. It creates many meshes with one material each from one mesh with many materials (submeshes).

First run

To showcase Batching Tools I used a model pack offered for free from Mr Necturus at the Unity Asset Store, called “ 9 squares wooden shields ”. In this example, I used a Mobile Diffuse shader.

Import two shields prefabs in the scene. Select them and hit the **Combine materials** button



And that was it. With one click, Batching Tools :

- Deactivated all the gameObjects you choose to combine and placed them in a new gameObject called Original Objects (all operations are non-destructive), so you can easily revert back to them.

- Created a copy of the original gameObjects and placed it in a new gameObject called Combined Objects.

- Created a new material and a texture atlas and assigned it to all the Combined Objects, after it modified their uv accordingly.

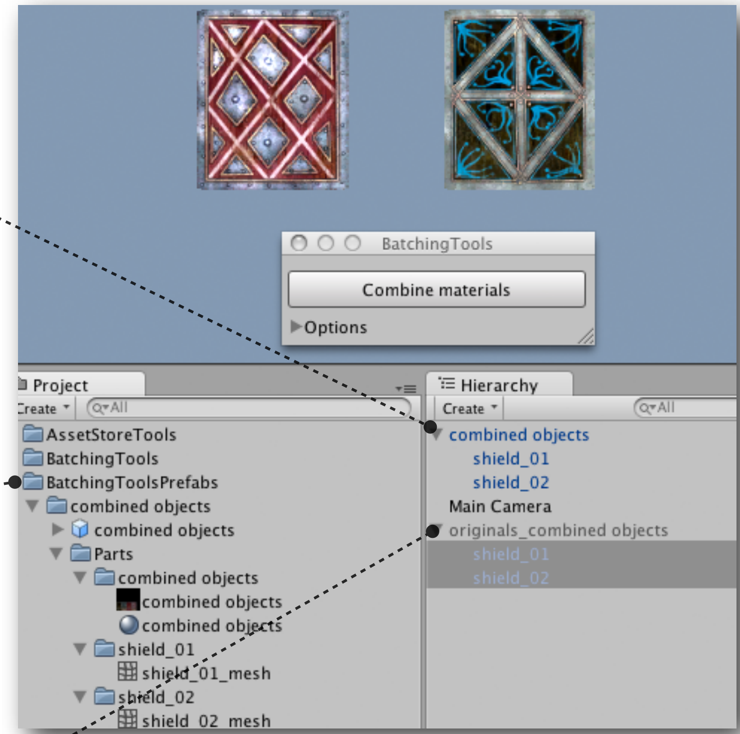
- Created a new combined mesh from the meshes of Combined Objects, assigned the shared material to it.

- Saved the new assets in a new Folder called BatchingToolsPrefabs.

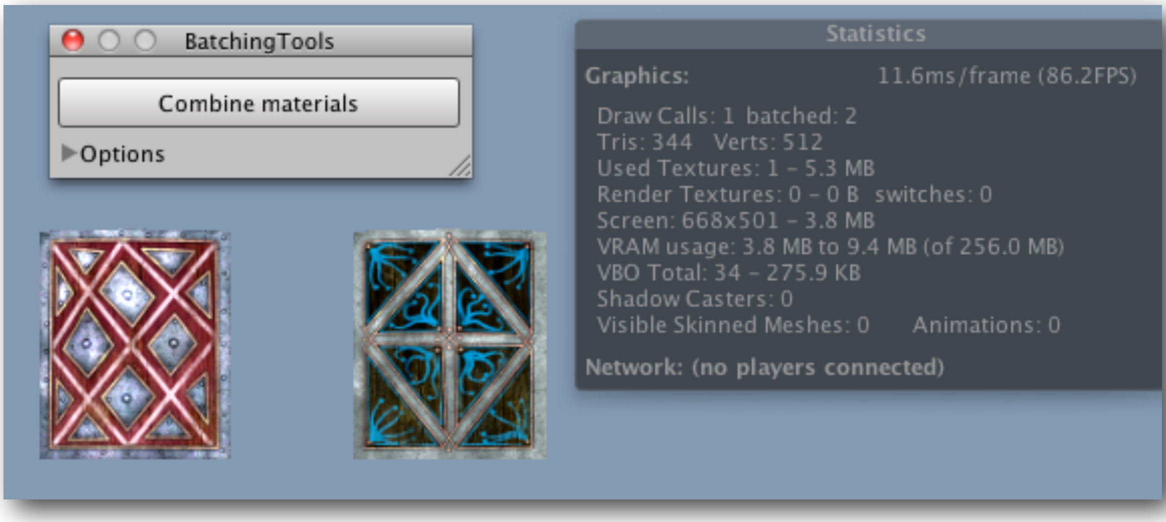
Batching Tools creates a new gameObject called **Combined Objects** which contains the new meshes with modified uv's that share the same material.

In the Project panel a new folder appears named **Batching Tools Prefabs.** It contains an automatically created prefab of the combined objects, and its parts.

Batching Tools places the original objects you selected in originals_combined objects and deactivates them (**Original Objects** in text)



Now the combined objects are ready to be batched. Check the **Draw Calls**.



```
Graphics:                    11.6ms/frame (86.2FPS)
  Draw Calls: 1  batched: 2
  Tris: 344   Verts: 512
  Used Textures: 1 - 5.3 MB
  Render Textures: 0 - 0 B  switches: 0
  Screen: 668x501 - 3.8 MB
  VRAM usage: 3.8 MB to 9.4 MB (of 256.0 MB)
  VBO Total: 34 - 275.9 KB
  Shadow Casters: 0
  Visible Skinned Meshes: 0      Animations: 0
Network: (no players connected)
```

The thing to notice is that it took just one click to perform all these operations :)

About Batching

Indie game developers must often wear many hats (take on many roles) due to lack of funds. One day they are 3d artists, next day they create textures in Photoshop and after that they are programmers. Or they can team-up, making small teams of artists and a programmers. Batching involves extra work from both artists and programmers. Time is money and the creation of a game is vary time-consuming. While many developers are aware of the benefits of batching, they do not use it because it requires much labor. Not any more.

Batching Tools frees up time for the Unity developers, permitting them to focus more on the overall “feel” of a scene instead of losing time and energy changing uv, creating texture atlases and combining models. Those repetitious tasks can be done automatically with this tool.

While there are some guidelines for batching, intuition can fail. Only testing shows the most efficient combination for each platform. With Batching Tools you can create fast different combinations to select the most efficient one.

Batching is the process of combining draw calls. The reasoning is that each draw call has an overhead, which can be saved by combining them. Unity supports static and dynamic batching. For those batching modes to work, developer must create shared materials, i.e a material that is shared by many gameObjects. This is done by Batching Tools. Further, Batching Tools provides two new batching modes, one called BTStatic, which is suitable for non-moving gameObjects and another called BTDynamic, suitable for moving objects.

Here is a table that shows the different batching modes available :

Unity static	Unity dynamic	BTStatic	BTDynamic
non-moving objects	moving objects	non-moving objects	moving objects
more than 300 vertices	up to 300 vertices	more than 300 vertices	more than 300 vertices
Pro only	all licences	all licences	all licences

More information about batching in Unity can be found at :

http://unity3d.com/support/documentation/Manual/iphone-DrawCall-Batching.html

http://unity3d.com/support/documentation/Manual/iPhone%20Optimizing%20Graphics%20Performance.html

Choosing what to combine

Supported shaders

Batching Tools can combine objects that use the same shader.
The supported shaders are :

Desktop	Mobile	Mobile/ Particles
Diffuse [1]	Mobile/Diffuse	Mobile/Particles/ Additive
Transparent/ Diffuse[1]	Mobile/Unlit (Supports Lightmap)	Mobile/Particles/ Alpha Blended
Unlit/Texture	Mobile/Bumped Diffuse [2]	Mobile/Particles/ Multiply
Unlit/Transparent	Mobile/Bumped Specular [2,3]	Mobile/Particles/ VertexLit
Bumped Diffuse [1,2]	Mobile/Bumped Specular (1 Directional Light)[2,3]	Mobile/Particles/ VertexLit (Only Directional Lights)
Bumped Specular [1,2,3]		

[1] Those shaders after material combine operation retain main Color results (the tint) but does not allow further modification of them. They do not batch using Unity's built-in dynamic batching, but can be used in Unity static and BTStatic mode.

[2] Those shaders work only if the normal texture size has equal dimensions with main Texture.

[3] Those shaders after material combine operation do not provide per gameObject control of specular color or amount of Shininess.

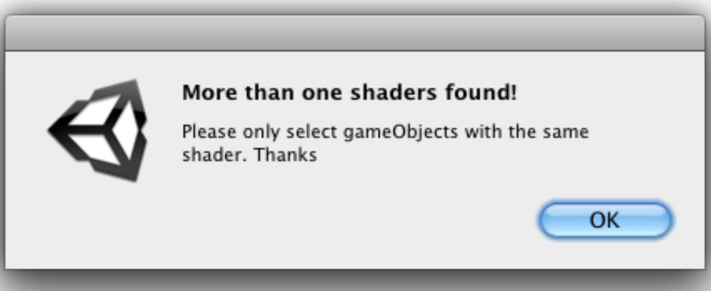
Only choose objects that use one of the above shaders. Please pay attention to the shaders limitations, since they are not subject to change.

Combine gameObjects that use the same shader

To create a shared material, it is necessary that all the gameObjects you try to combine use the same shader.

If by mistake you include objects with unsupported shaders, you will be prompted with an error message :

Click ok and make sure that you use Batching Tools with one of the supported shaders only. Also, make sure that all the gameObjects you choose to combine their materials share the same unique shader. You cannot combine gameObjects that use more than one shaders.

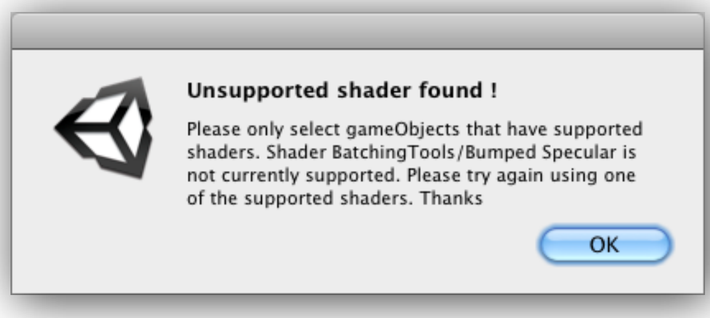


More than one shaders found!

Please only select gameObjects with the same shader. Thanks

If you accidentally choose to combine materials of gameObject with more than shader, you will be prompted with an error message too:

Click ok and make sure that you use Batching Tools with one of the supported shaders only (i.e not more than one of the supported shaders).

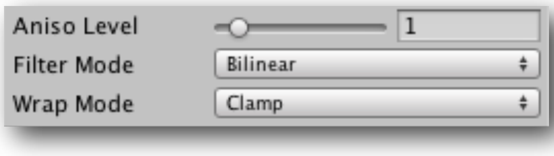


Unsupported shader found !

Please only select gameObjects that have supported shaders. Shader BatchingTools/Bumped Specular is not currently supported. Please try again using one of the supported shaders. Thanks

Use models with non-overlapping uv

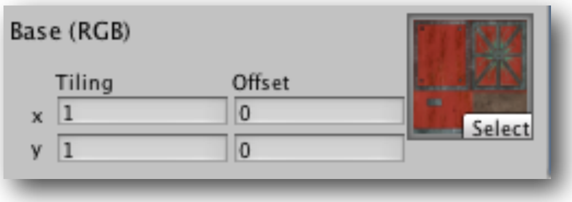
Models with overlapping uv are not supported. If you try to combine them, you will observe artifacts. To check if a model has overlapping uv select its main texture and in the inspector set its wrap mode in Clamp. Now inspect the 3d model in the Editor. If it visually appears ok, you can use Batching Tools with this model. Else, you (or the 3d artist) should modify the model uv in a 3d modeling app.



Aniso Level 1
Filter Mode Bilinear
Wrap Mode Clamp

Do not use offset and tiling

The offset and tiling of each material to combine should be set to 0 and 1 respectively. If you use tiling or offset to define the appearance of your models, in order to combine them you should modify them first in a 3d modeling app by adding geometry for tiling and moving vertices for offset. I.e say that you have a texture that tiles 4 times
(Tiling x = 2, y = 2)
on a plane. You could create four planes instead, place and scale them accordingly and combine them to have the same results. I plan a small utility to facilitate this cumbersome process.


Base (RGB)

	Tiling	Offset
x	1	0
y	1	0

Do not use non-power of two textures

The use of non power of two textures is not recommended because they cannot be mipmapped without artifacts. Use power of two textures instead (e.g 512x512, 256x1024). Be aware though that in IOS non square power of two textures cannot be compressed (i.e 256x1024 cannot be compressed in PVRTC, while 1024x1024 can be compressed).

How to plan ahead a good atlas
General information

Batching tools automatically creates a texture atlas from the textures of the objects you wish to combine. In order to combine the textures of the objects efficiently, you can plan ahead in which resolution they will be combined. “Efficiently” means that you utilize as much surface of the texture atlas as possible (ideally all).

Ideally you should combine 4 materials of size KxK in a texture atlas that is K^2xK^2, where K is a power of two.
Or 4^n materials of size KxK in a texture atlas that is K^{n+1}xK^{n+1}.

Texture Atlas size	Textures nr and dimensions
2048x2048	-4 textures 1024x1024 each -16 textures 512x512 each -64 textures 256x256 each -2 texture 1024x1024 each **and** 8 textures 512x512 each -1 texture 1024x1024 **and** 8 textures 512x512 each **and** 16 textures 256x256 each
1024x1024	-4 textures 512x512 each -16 textures 256x256 each -2 textures 512x512 each **and** 8 textures 256x256 each.

Many more combinations are also valid, check the table below for some examples. The above combinations ensure that in the produced texture Atlas you will not encounter loss of quality. If you try to fit four 4096x4096 textures in a 1024x1024 texture, you have to shrink them to fit. It is done automatically and results in quality loss.

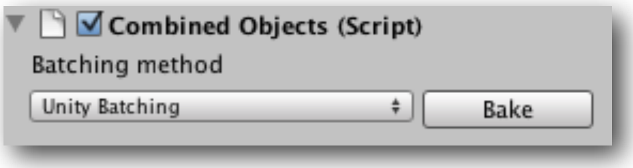
The new texture atlas size is selectable by clicking options, in the field Maximum atlas size.

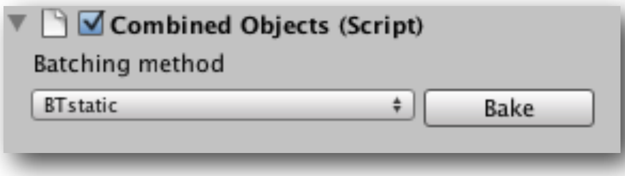
A 1024x1024 texture atlas can contain four 512x512 textures, or sixteen 256x256 textures without shrinking them.

BTStatic

BTStatic is a custom batching mode for objects that do not move. It is available for every Unity license and can run along with Unity static batching. What BTStatic does is create a mesh with one material from many meshes with many materials.

In 3d modeling terms it is equal to take many meshes, create the texture atlas from their textures, uv map them to the texture atlas, and combine them to one mesh.





To use it, just select the combined objects. In the Inspector, there is a pop-up menu called Batching Method. Select BTStatic. (Hint : check the fps difference, in some cases BTStatic is very effective)
If you are happy with the results, press Bake. If Unity Batching is selected, it will destroy the BTStatic created mesh. Otherwise, if BTStatic is selected, it will destroy the mesh renderers and mesh filters components of the game objects, leaving only the BTStatic mesh.
The main benefit of this approach is speed of development. Batching is tricky, some things work as expected, others not. You can try many different combinations fast to see which one works best with your setup.

Important ! You should bake your desired batching mode before building the scene !

BTDynamic

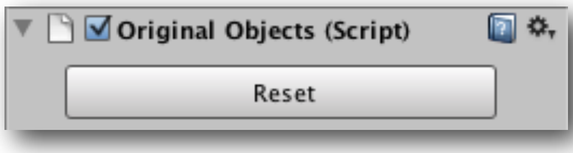
BTDynamic is a custom batching mode for objects that do move. It is available for every Unity license. What BTDynamic does is create a skinned mesh with one material from many meshes with many materials. The transform of each gameObject that is included is turned into a bone.

In 3d modeling terms it is equal to take many meshes, create the texture atlas from their textures, uv map them to the texture atlas, combine them to one mesh and rig a bone for each one.

To use it, first select the gameObjects you wish to combine and click the + button. This will include the gameObjects selected and their children in the skinned mesh. If you accidentally included some extra objects, simply select them and click the – button. After that, click combine objects button. In the combined objects select BTDynamic and press the Bake button. That’s it !

Revert to the original objects

If you wish for some reason to revert the gameObjects back to how they were before the combination took place is simple. Select the original_combined objects. In the inspector you will see a big button called Reset. Click it and the original objects will be restored back to the place they were initially.



Using names to organise your project

You can name the objects that you create as you wish, by filling the Combined objects name field before you combine the objects. This helps to organize better the assets in your project. Just fill in something meaningful instead of “level part 1” which is there by default.

Expanding submeshes button

In unity editor, a mesh with submeshes is represented as a gameObject with one mesh renderer and many materials. If one of the object you selected to combine has submeshes, you will get a similar message to the following :

Click ok, select the mesh with the submeshes and click Expand Submeshes button. This creates some new gameObjects, each one having one material and one mesh. Those can be further processed for batching with the above options.

Lightmaps

Batching tools supports lightmaps. Bake your lightmaps before you combine the materials, not afterwards, for best results.

Exporting the textures

It is possible to export the textures. Select the texture, then choose
Menu -> Assets -> Export Textures. A dialog appears where you choose where you wanna save the texture. Thanks to Eric Haines for providing the script in the wiki.

Exporting the mesh

It is possible to export the mesh you created. Select the gameObject that contains the mesh you wish to export, then choose
Menu -> Assets -> Export Mesh from GameObject. A dialog appears where you choose where you wanna save the texture. Thanks to Keli Hlodversson for providing the ObjExporter script in the wiki.

Hierarchy

In order to preserve the scene hierarchy, i.e child - parent objects relations, always select the top hierarchy object. not the children. Please do not choose objects that are inactive or have inactive children, this cannot be handled by batching tools and will give errors.

The progress bar doesn't go away !

If an unexpected error is encountered, the progress bar stays on screen. To solve this, simply click on the little arrow on the left of the Options and it will disappear. Please notify me, describing what happened and if possible, include a minimal scene where the error can be reproduced.

Many thanks to...

Neil Carter, Michael Garforth, Eric Haines, Keli Hlodversson, Tenebrous, fholm, airship, duckets, Tak, LittleAngel and all the other people who helped me over time to create this script.

Thanks for your support !
Ippokratis Bournellis