atlasEditor documentation

Overview

AtlasEditor is a unity utility which allow you to create atlases from several images, result is a new Texture2d which contains all texture which you added to the canvas, and metadata where you can find position for all of your elements.

Abbreviation

atlas - is a large image containing a collection of sub-images, or "atlas" which contains many smaller sub-images, each of which is a texture for some part of a 3D object.

Why to use atlases? because it reduce drawcalls, Example, if you have 10 object on scene and all of them have a different materials you will receive 10 drawcals, but if you have 10 objects and have one shared material for all of them you will receive just 1 drawcal, all of you need to do is use one material for all of you object which pointed to your atlas texture and modify mesh.uv coordinates for you objects.

Description

Installation

To install this package, you can download latest version from git repository, or install it through asset store.

When you will import package to the project, you can choose which folders you want to include in your project.

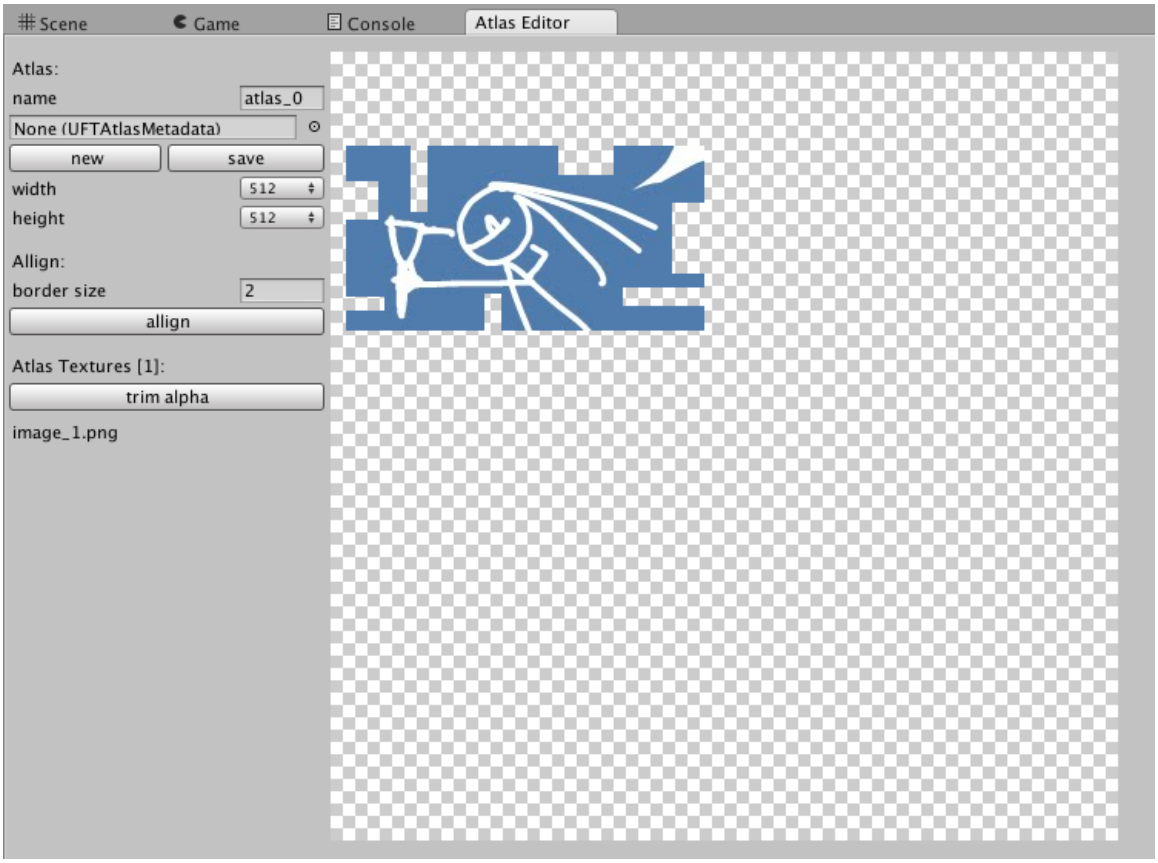
If you know how to works with atlases, you need to import just **UFTAtlasEditor** folder with no **UFTAtlasEditor -> Demo** folder.

If you want to check how it works, you can import the whole package, this package will contains demo floder with two scenes.

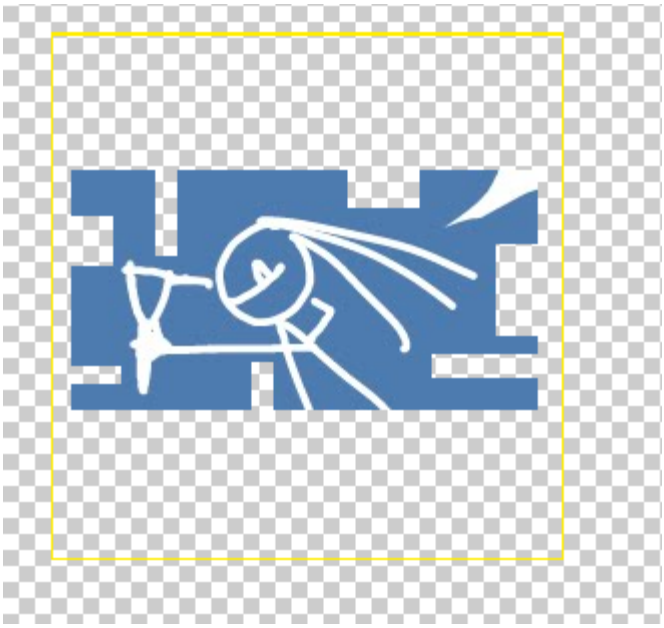
Main Editor window

New menu available at **Window -> UFT Atlas Editor**

When you click on this menu, new window will be opened attached to the screen view



- **Name** atlas name, when you will save you atlas, texture and metadata will have this name, by default it will be atlas_[0,1,2....] depends on whether you have atlas_* objects in Asset folder
- **UFTAtlasMetada** object chooser, click on it if you want to choose and modify existing atlas. leave it **None** for a new atlas.
- **New** create new atlas
- **Save** save/update atlas
- **width** atlas width
- **height** atlas height
- **Align**
  - **border size** minimal gap between textures on your atlas
  - **allign** tile you textures on atlases (used standard Texture2D.PackTextures function to tile textures in to atlas)
- **Atlas Textures** indicates textures names used in current atlas
- **Trim alpha** if your texture has a border with alpha=0 this function will remove it
  e.g. this texture has an empty borders (Yellow border indicate original texture size)



after trimming it will looks like

- Drag textures from unity navigator to the atlas, to add them to current atlas
- Select texture and press **Delete** to remove one atlas from current atlas

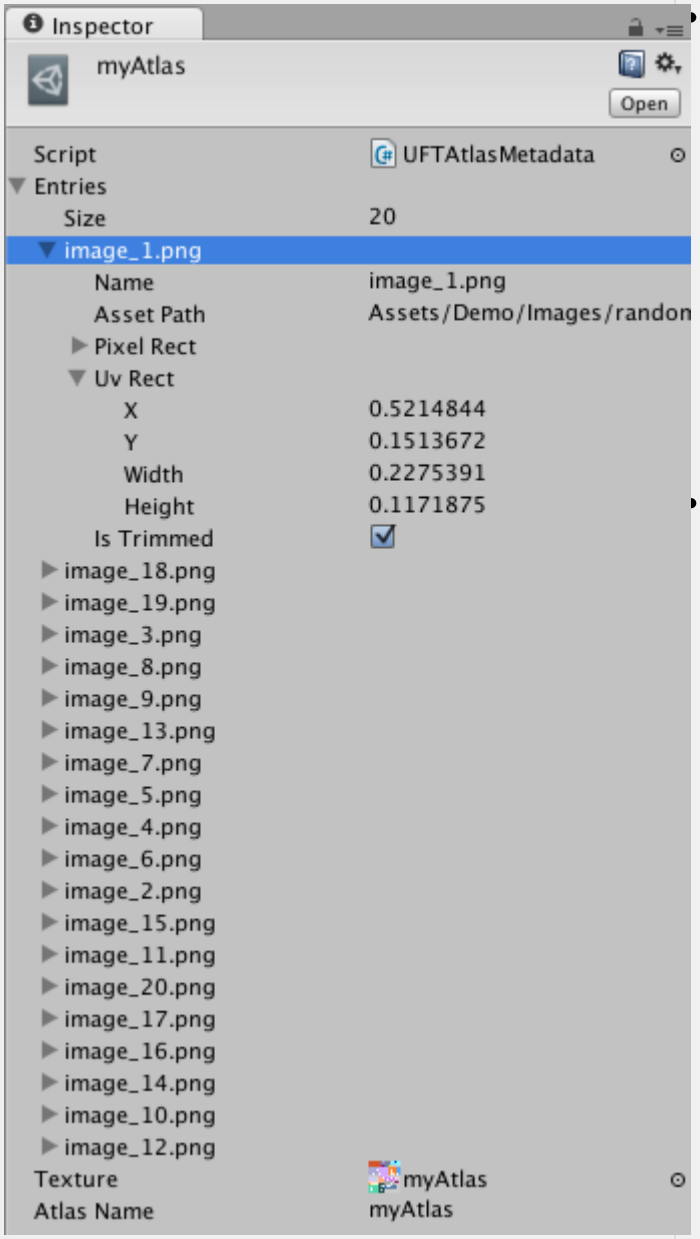
Supported Features

- drag'n'drop textures within atlas
- yellow color border when mouse cursor within atlas
- Undo/Redo
- When you will open existing atlas, if any of your texture has been moved to another location, this textures won't show

MetaData format

If you will forget atlas structure, just open metadata object, which has the same name as an atlas, in inspector you will se the structure.

As well as you can open cs files in **UFTAtlasEditor->Resources->Scripts->Core->Atlas**



- MetadataStructure
  - texture (Texture2D) - texture which you will use in your material
  - atlasName (string) - atlas name
  - entries (Array) - each image on atlas has own detail
    - name (string) - original texture name
    - assetPath (string) - original texture path
    - pixelRect (Rect) - position on atlas in pi xels (0,0) is left upper corner
    - uvRect (Rect) - position which you will use in uv (0,0) is left down corner
    - isTrimmed (Bool) - indicate was this tex ture trimmed or not
- Example

C#

```
public void updateUV(){
  Rect
rect=atlasMetadata.entries[textu
reIndex].uvRect;
  Mesh mesh=getObjectMesh();
  Vector2[] uvs=new
Vector2[mesh.uv2.Length];
  for (int i=0; i<uvs.Length;
i++){
   uvs[i].x = mesh.uv2[i].x *
rect.width + rect.x;
   uvs[i].y = mesh.uv2[i].y *
rect.height + rect.y;
  }
  mesh.uv=uvs;
 }
```

Examples

All examples use **UFTAtlasEditor -> Resources -> Scripts -> UFTSelectTextureFromAtlas** script.

This script at first execution check, whether object has uv2 coordinates or not, if this array = 0 or =null, it will copy original uv array to uv2.

Then when you change texture index in atlas it multiply original uv coordinates (which already stored in uv2) to uvRect from atlas metadta.

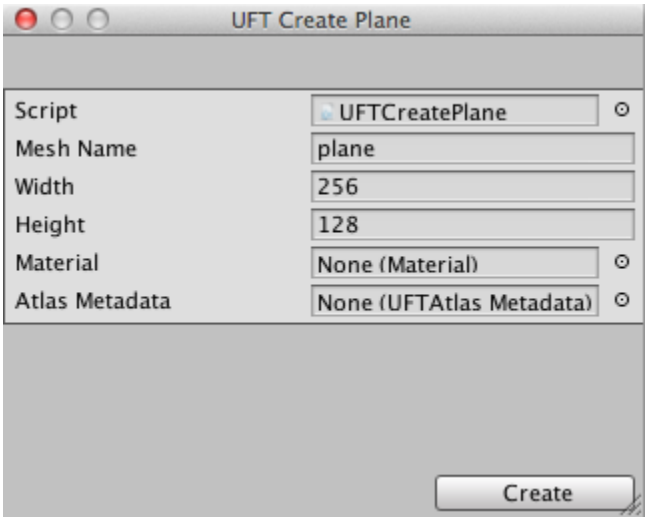
I didn't include original test images in to the build, if you would like to download them, you can simply download this package

Editor scene

Editor scene available at this path **UFTAtlasEditor -> Demo -> EditorDemo**

For editor, I've created simple script which create plane and attach all necesary scripts to it (MeshRenderer, Material and UFTSelectTextureFromAtlas),

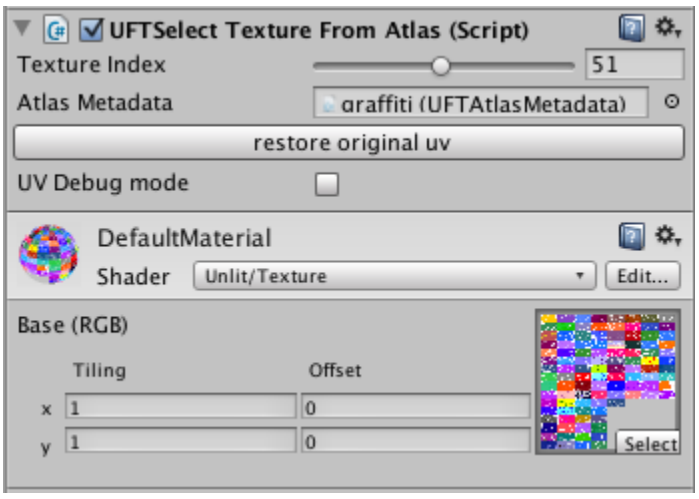
you can find it at **Game Object -> UFT Create Plane**



you can create material which use your atlas texture and use it wit metadata in this script.

At editor scene there is just one plane pointed to the camera

This plane has following parameters



As you can see, here is a defaultMaterial which pointed to atlas texture,

And you can change TextureIndex, and appropriate part of atlas will be showed.

When you will Use this script it will change original uv position permanently

> This script use uv2 coordinates to store original uv it means that your original model must not use uv2 coordinates. If it does, just implement your own script which will store original uv, somwhere else.

After you modify original mesh uv, you no longer need attached script **UFTSelectTextureFromAtlas** on to it.

If you want to return original uv coordinates, use **restore original uv** button and after that you can remove **UFTSelectTextureFromAtlas** script

If you interesting what stored in uv and uv2 array of you mesh, use **UV Debug mode**

p.s. I didn't included all grafiti small pictures, just generated atlas, if you would like to have it, get it out from git repository or download this package

Game scene

Game scene located at **UFTAtlasEditor -> Demo -> GameScene**

It has just one camera and one script which generate planes with textures from the same atlas. as result when you will click play you will see moving planes

As you can see it has just one drawcall and 102 batched textures. it happens because all of planes use the same material, the different's is just in uv coordinates

Script which generate planes use the same script which change original uv as in EditorScene

Source code

Atlas Editor distributed under BSD license.

Source code available here http://git.nicloay.com/atlaseditor

Latest build is here http://git.nicloay.com/atlaseditor/downloads

Feedback

Tweet to @nicloay

unity3d forum thread

Known Issues

1. After trimming some textures leaking, so on scene save you will see following notification (it's not an error it's just show that unity remove leaked objects

```
Cleaning up leaked objects in scene since no game object, component or manager
is referencing them
Texture2D has been leaked 1 times
```

2. Delete button doesn't work if editor attached to scene, if it in floating window it works well.