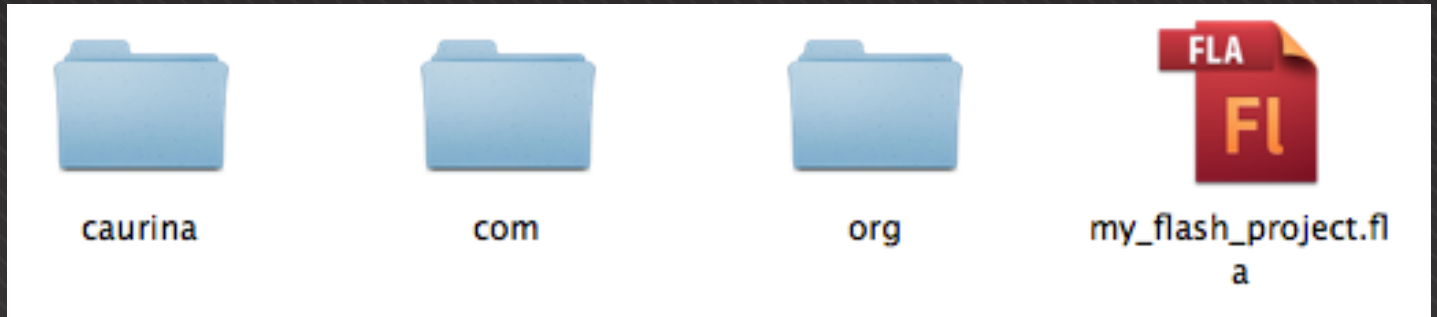


Chapter 1: TILES hello world

for this tutorial, please refer to the file "tutorial1.fla", into the "tutorials" folder

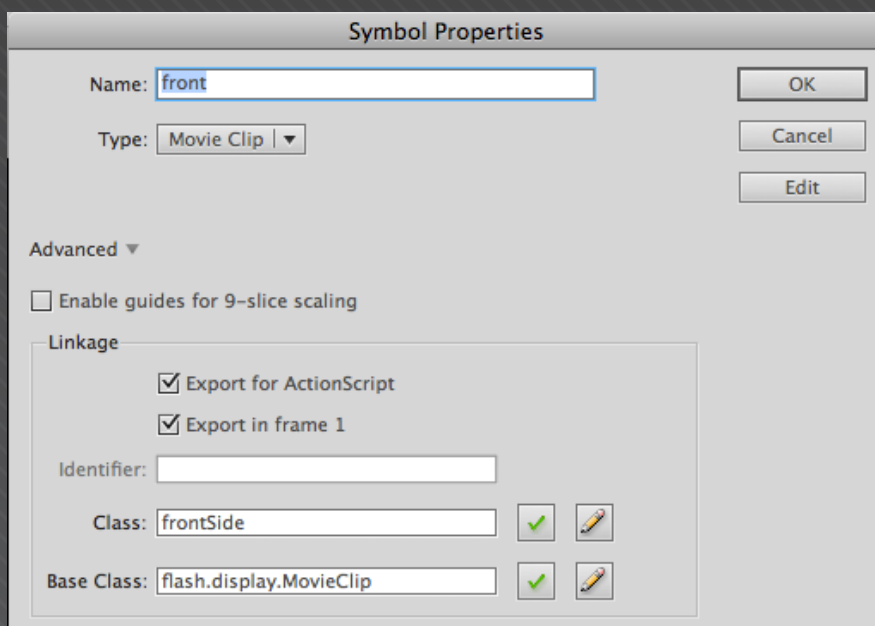
1. First of all, lets be sure to copy the TILES source code into the root of our flash project. (three folders: 'caurina', 'com' and 'org'); Alternatively, you might choose to copy the TILES component into your library. (for reference, TILES quick start guide)



2. Lets open a new flash document. It needs to be an ACTIONSCRIPT 3.0 project, since TILES doesnt support actionscript 2. Set the stage size to 800x600, with a black colour as back-ground, just to make it look a little better :)



- 3.1 Create a new movieclip: draw a simple box, and set the size to 400 pixels width and 300 pixels height. Set a nice color, and add some text to make it look like the picture here :). Label the movieclip you just created 'front'.



- 3.2 open the library panel into flash, right click over the 'front' movieclip, and select 'properties' from the menu. The simbol properties panel will open. In the 'linkage' box, select 'Export for ActionScript'. Set the Class to 'frontSide'. The base class is supposed to be 'flash.display.MovieClip.' Click 'ok' to close the panel.



A definition for this class could not be found in the classpath, so one will be automatically generated in the SWF file upon export.

Don't show again.

Cancel

OK

3.3 You might get a warning like this one. just ignore it clicking ok.



rear side

4.1 Create another new movieclip, draw a simple box, and set the size to 400 pixels width and 300 pixels height. Label the movieclip you just created 'rear'.

Name: rear

Type: Movie Clip | ▾

Advanced ▾


Enable guides for 9-slice scaling

Linkage

Export for ActionScript

Export in frame 1

Identifier:

Class: rearSide 

Base Class: flash.display.MovieClip 

4.2 open the library panel into flash, and right click over the 'rear' movieclip, and select properties from the menu.

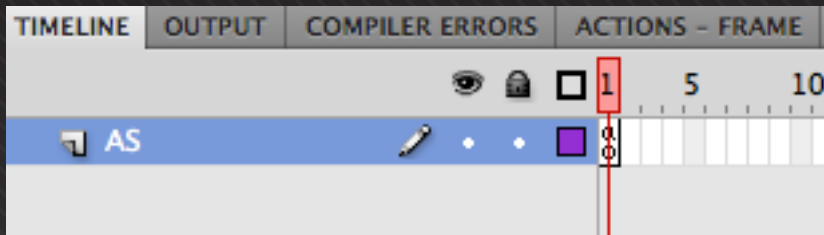
The simbol properties panel will open.

In the 'linkage' box, select 'Export for ActionScript'.

Set the Class to 'rearSide'. The base class is supposed to be 'flash.display.MovieClip.'

Click 'ok' to close the panel.

5. make sure that your stage area is clear. if you have any movieclip on it, remove them.



6. Ok, its time to create some Actionscript code. Lets open the timeline palette, select the first frame of the first layer, and lets open the Actions palette.

6.1 First of all, lets create an instance of the 'frontSide' movieclip and one of the 'rearSide' movieclip. The frontSide instance, 'frontMc' will be rendered by TILES at the startup. The rearSide instance, 'rearMc', its the target of our tweening.

```
var frontMc:frontSide = new frontSide();  
  
var rearMc:rearSide = new rearSide();
```

6.2 Lets import the TILES class and lets create and instance of it. The TILES constructor accept 2 parameters, width and height. I am setting those to 400px for the width and 300px for the height, *accordingly to the size of the two movieclips i have been created before.*

```
import com.chrometaphore.transitions.tiles.Tiles;  
  
var t:Tiles = new Tiles( 400, 300 );
```

6.3 Now its time to add the TILES instance ("t") to the stage. im also going to center it vertically and horizontally.

```
addChild(t);  
  
t.x = this.stage.stageWidth / 2 - t.width / 2;  
t.y = this.stage.stageHeight / 2 - t.height / 2;
```



6.4 Call the `init()` to start rendering the `frontSide` movieclip into the TILES viewport at the startup.

```
t.init(frontMc);
```

6.5 The interesting thing about TILES is the power to customize the tweening action. Lets have a look to all the tweening parameters in detail here.

```
//this object will contain all the custom parameters
var tweenParams:Object = {};

//number of TILES on the x-axis
tweenParams.xTiles = 5;

//number of TILES on the y-axis
tweenParams.yTiles = 5;

//set TILES rotationAxis: 'y' or 'x'
tweenParams.rotationAxis = 'y';

//single tile rotation transition type, default: 'easeOutSine'.
//(caurina tweener, for reference: http://hosted.zeh.com.br/tweener/docs/en-us/ )
tweenParams.rotationTween = 'easeOutSine';

//single tile rotation time (milliseconds)
tweenParams.rotationTime = 800;

//pause before starting of the next tile rotation (milliseconds)
tweenParams.rotationPause = 100;

set Tiles rotation order: "linear-x", "linear-y", "random", "diagonal", default: 'linear-x'
tweenParams.rotationOrder = 'diagonal';

set how many rotation loops every stripe will perform, default: 1
tweenParams.rotationloops = 1;

set Tiles (cubes) depth, default: 10 ( 0=flat 2d planes )
tweenParams.tilesDepth = 20;

set Tiles (cubes) side color, default: 0x000000 (black)
tweenParams.sideColor = 0x333333;

set Tiles (cubes) next target polygon face: "opposite", "adjoining"
tweenParams.targetPolyFace = "adjoining";

set Tiles explosion range on both x and y axis, default: 0px
tweenParams.explosionRange = 20;
```



```
set Tiles explosion tween type (caurina Tweener), default: 'easeOutSine'.  
//(caurina tweener, for reference: http://hosted.zeh.com.br/tweener/docs/en-us/ )  
tweenParams.exlosionTween = 'easeOutSine';
```

```
//set Tiles implosion tween type (caurina Tweener), default: 'easeOutSine'.  
//(caurina tweener, for reference: http://hosted.zeh.com.br/tweener/docs/en-us/ )  
tweenParams.implosionTween = 'easeOutSine';
```

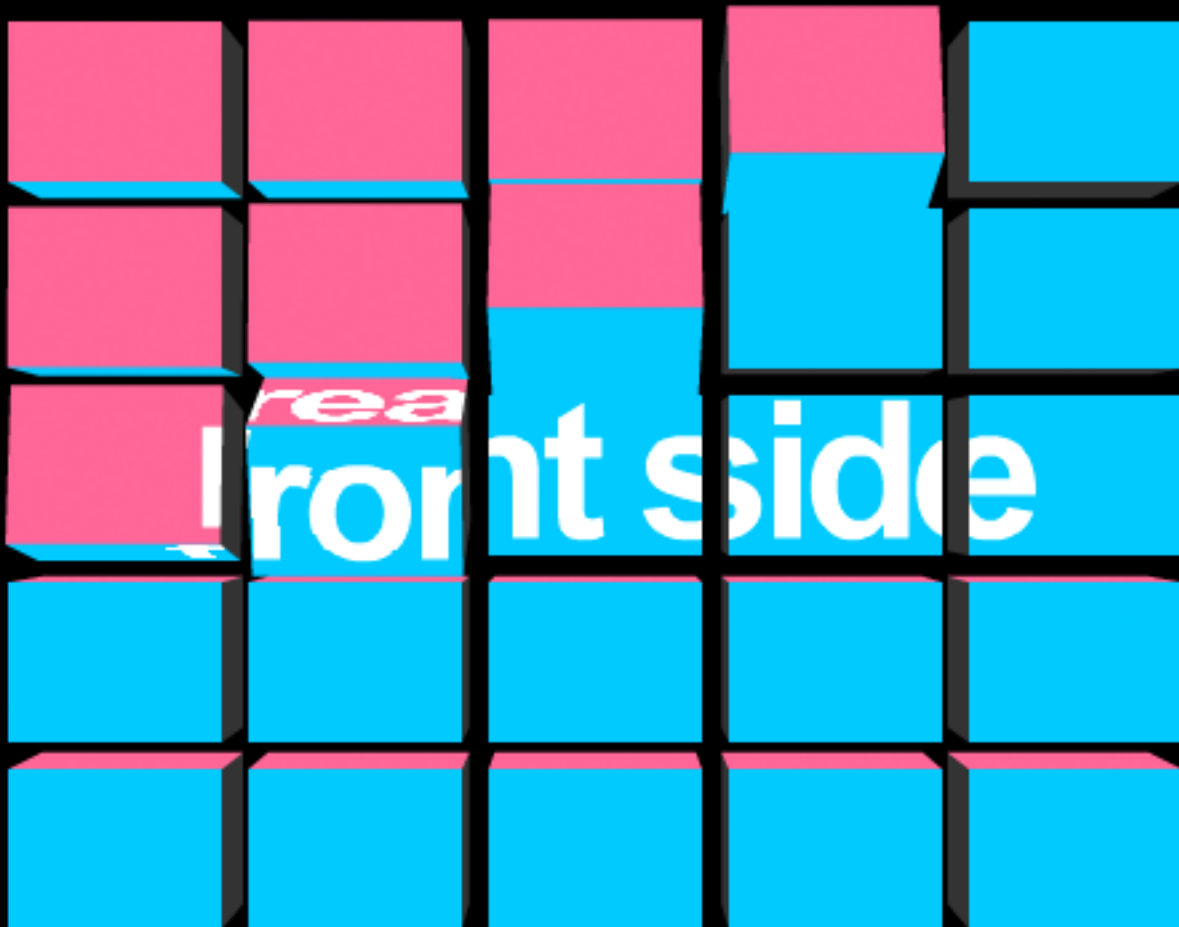
```
//set Tiles explosion time (milliseconds), default: 1000  
tweenParams.exlosionTime = 700;
```

```
//set Tiles implosion time (milliseconds), default: 1000  
tweenParams.implosionTime = 700;
```

6.6 Ok, now that everything is set correctly, lets call the tween function. the first parameter of the function asks for the tweening target (in our case, the rearSide instance, 'rearMc'), the second (optional) parameter is the object containing all the tweening parameteres.

```
t.tween( rearMc, tweenParams );
```

6.7 Test your movie! (Control -> Test Movie). The movieclip should smoothly tweening from the frontSide mc to the rearSide mc.

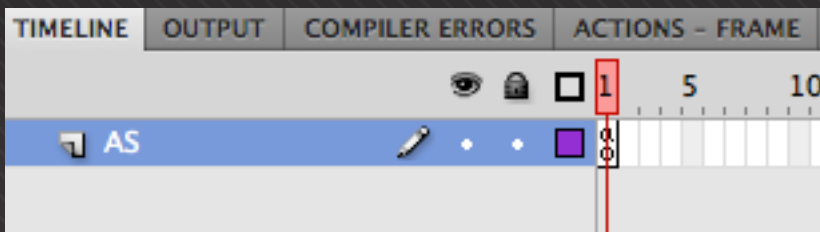


Chapter 2: working with events.

for this tutorial, please refer to the file "tutorial2 fla", into the "tutorials" folder. Also, 'tutorial3 fla' shows how to use a Mouse Event to trigger a TILES tweening.

2.1.1 Ok, lets open the flash file we have created in the chapter one. The purpose of this chapter is to learn how to wok with TILES events, to create a continuous looping animation.

2.1.2 On the timeline, select the AS layer on the first frame, where we previously created all the code required for the tweening. Open the Actions palette.



2.1.2 At the end of our previously created code, just before the last line that activate the tween() function, lets import the Tiles events class, TweeningEngineEvent.

```
import com.chrometaphore.transitions.TweeningEngineEvent;
```

```
82
83 //import the TweeningEngineEvent class
84 import com.chrometaphore.transitions.TweeningEngineEvent;
85
86 //start tweening!
87 t.tween( rearMc, tweenParams );
```

2.1.3 Let's add a variable to register our current mc, we will use this var a little later. Since our first movieclip is the frontSide instance, i will set the currentmc var to 'front'.

```
var currentMc:String = "front";
```

2.1.4 The TweeningEngineEvent class includes 3 available events:

START_TWEEN triggered at the beginning of a tweening.

END_TWEEN triggered at the end of a tweening.

PARTIAL_TWEEN triggered at the beginning of every single tile rotation.

2.1.5 lets use the addEventListener function to start listening to the 'END_TWEEN' event. So, when a tweening animation is completed, the onTweeningComplete function will be called automatically.

```
t.addEventListener( TweeningEngineEvent.END_TWEEN, onTweeningComplete );
```



2.1.6 the onTweeningComplete() function, triggered at the end of every tweening, switch the next target of the tweening according to the currentMc var. The result is an infinite looping animation.

```
function onTweeningComplete( e:TweeningEngineEvent )
{
    if ( currentMc == "front" )
    {
        currentMc = "rear";
        t.tween( frontMc, tweenParams );
    }
    else
    {
        currentMc = "front";
        t.tween( rearMc, tweenParams );
    }
}
```

2.1.7 Test your file. (Control -> Test Movie). The movieclip should keep on tweening from one movieclip to another.

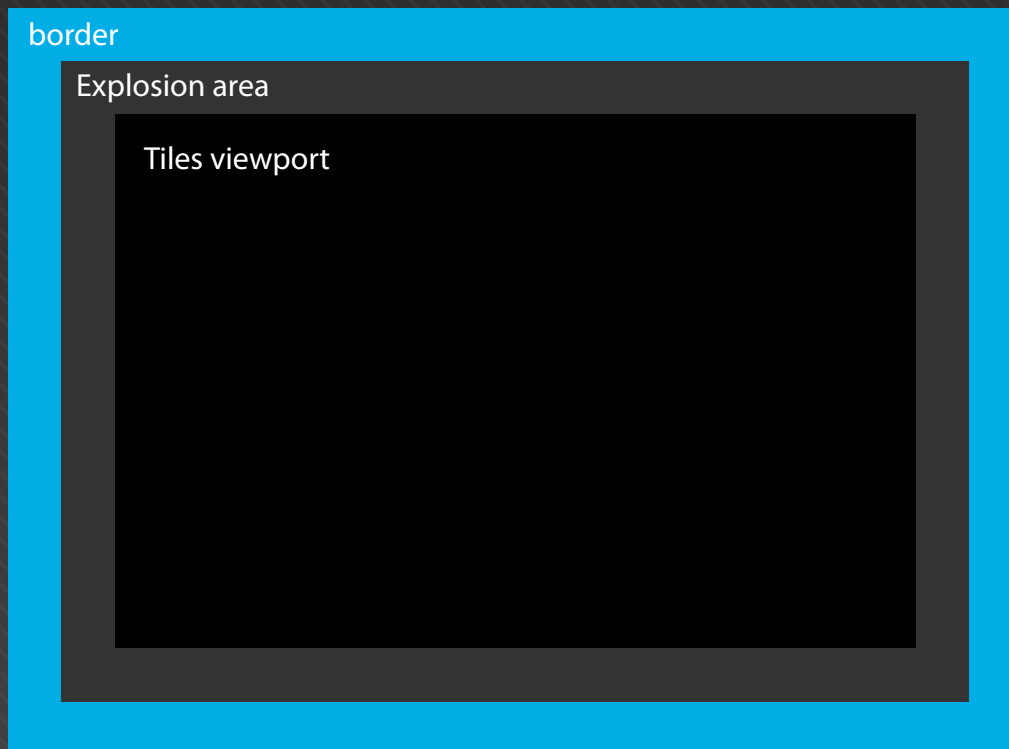


Chapter 3: some little extra things.

3.1 TILES borders

When you create a new TILES instance, there is an extra parameter that you have to keep in mind: the *border* parameter. This parameter has a default value of 50px. The border value is strictly connected to the tiles *explosion* value. F.i, if you set an explosion value to 100px, you will have to set the border value to >100px, otherwise your explosion might result cropped.

Please handle the border value with care. Large border values might affect seriously the overall movie performance, since the CPU has to manage a large 3D area.



F.i the line of code below sets the border value to 150pixels.

```
var t:Tiles = new Tiles( 400, 300, 150 );
```

3.2 TILES tweening property

Sometimes its useful to know whether a tweening is happening at the moment or not.

```
trace(t.tweening) //print 'true' if tiles is currently tweening , or 'false'
```

3.2 TILES material segments

During the tiles rotation the movieclip is divided into segments. The default value is 1 segment. More segments equals to a better quality of the surface during the rotation, but it also heavily affects the performance. I suggest to keep the value set to 1, but the value can be changed in this way:

```
t.view3D.matSegments = 1;
```



3.3 TILES is a Sprite itself!

TILES inherits all the Sprite class properties, so you can modify properties like Alpha, or even add effects like blur, glow, or drop shadow.

to change the alpha of a TILES instance, you just need to add something like this:

```
t.alpha = 0.75;
```

and, to add a simple drop shadow filter to a Tiles instance:

```
var dsf:DropShadowFilter = new DropShadowFilter( 8, 45, 0x000000, 1, 6, 6, 0.5 );  
t.filters = [ dsf ];
```

*..Thats pretty much everything! I suggest you to have a close look to all the examples available into the “examples” folder, and to read the as3 documentation into the “docs” folder.
If you get stucked, you can also send an email to info@chrometaphore.com.*

Thank you, and have fun with TILES!

