A Step-by-Step Guide to Setting Up clickTAG Tracking in Flash Banner Ads

1. What is a clickTAG, anyways?

A clickTAG is really nothing more than a placeholder for a link.

Flash advertising is great: it allows clients to use video, animation and interaction in their ad campaigns, and it is an easy-to-use media platform on which many people know how to design, develop, and program. But Flash advertising campaigns are not as easy to distribute or manage as static JPG or GIF campaigns.

Unlike a static image, which can easily be surrounded by a standard HTML link, Flash banners must be specifically designed to include a button (a place where the user will be able to click on the ad) and a link (which will tell the user's browser where to go, once the ad has been clicked.)

So, we just create a button in Flash, toss a link in there, and we're good to go, right?

Not quite. What if a client needed to update every link in an entire campaign? Or what if a variable needed to be added to a link, for tracking purposes?

That's where clickTAG comes in. Rather than linking each ad unit to, say, *my-great-surf-company.com*, each ad references the *clickTAG* variable. Then, the ad distributor - in this case, we here at Surfline - go behind the scenes and connect the ads to the right places.

2. OK, I get it now. How do I set up a clickTAG in my Flash ad units?

The short answer is "hire a good Flash designer/developer, and they'll know what to do."

But that's not always realistic. With more and more business of all shapes and sizes entering the world of online marketing, hiring a dedicated Flash developer is not always an option.

In light of that, Surfline wrote this brief guide to help you add a button and a clickTAG link to your Flash ad unit.

If you're using ActionScript 2.0 , keep reading.

If you're using ActionScript 3.0, check out PAGE 6 first.

Adding clickTAG with ActionScript 2.0

Many banner ads today are still programmed in ActionScript 2.0 This is for two reasons:

1. There's a portion of the consumer market that still uses Flash 8.0 or below. Only newer versions of Flash can support ActionScript 3.0. As a result, some ad networks have requirements such as "Flash 8 or below" or "Flash 9 and below". (Note: These guidelines are frequently changing as Flash's newer versions are adopted by more users. Check with Surfline to make sure you're exporting to the appropriate Flash version.)

2. It's generally considered easier to program in ActionScript 2.0 than ActionScript 3.0.

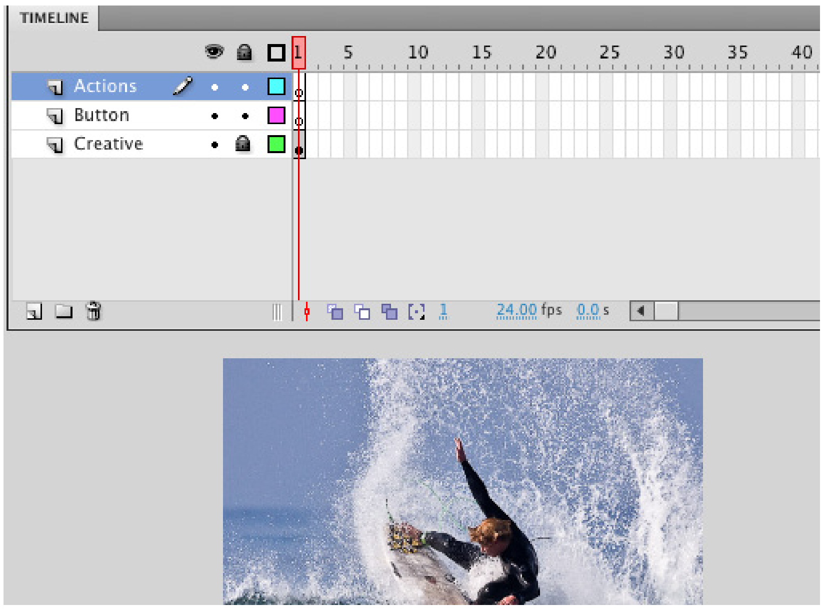
Adding a clickTAG in ActionScript 2.0 is easy. Let's get started.

1. Don't mess up your hard work.

First things first. Open up your .FLA file in Flash, and lock the layers that contain your advertising message - animation, photos, artwork, text, etc.

Now create two new layers. Name one "Actions," and name the other "Button."

You should end up with something like what you see on the right.



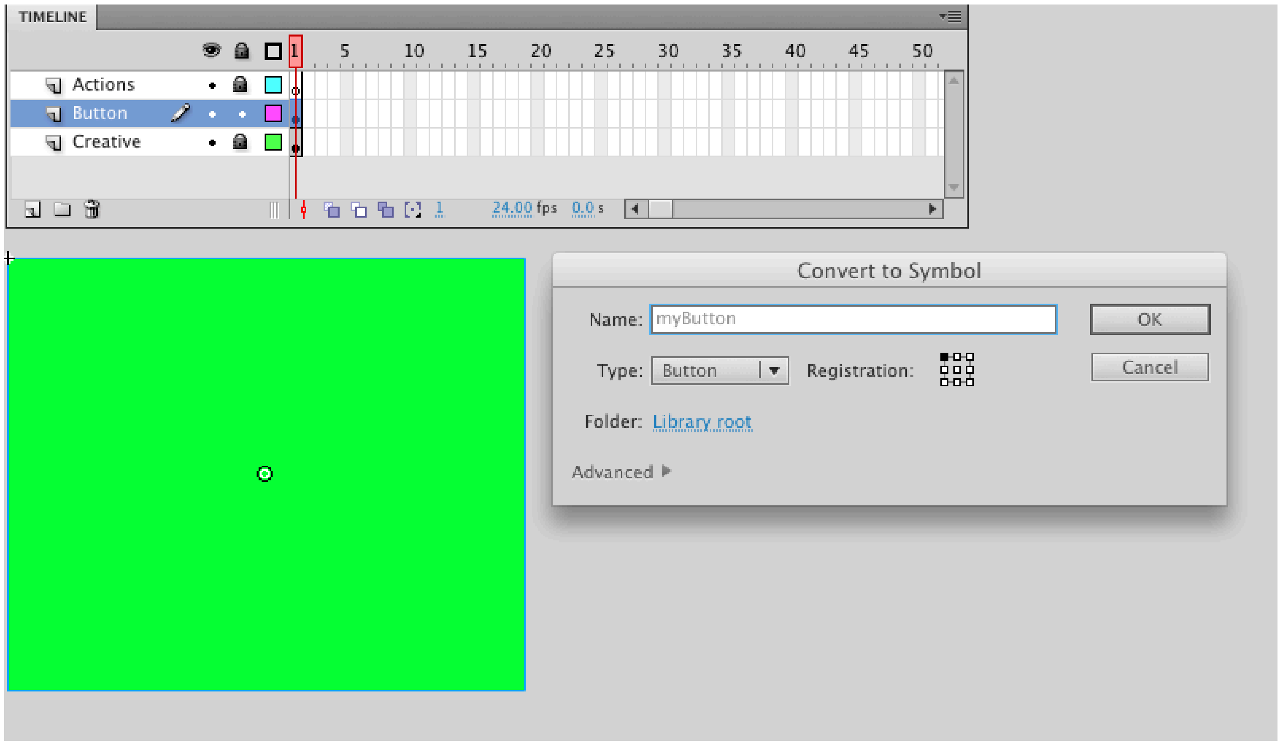
2. Create a button.

Select your newly created "Button" layer. Using the rectangle tool, draw a rectangle large enough to cover the entire stage. It's fine to go over the edges of the stage. It just needs to cover the entire area of the ad unit. This is the area that users will be able to click on. It's OK if the rectangle is covering up your ad and making it impossible to see anything. We'll be fixing that in a moment.

Select the rectangle you just drew, go to the "Modify... > Convert to Symbol."

Under "Type", select "Button". In the "Name" field, call it "myButton."

You can see the end result on the next page, in Figure 1.2.



Convert to Symbol

Name: myButton

Type: Button

Registration:

Folder: Library root

Advanced

OK

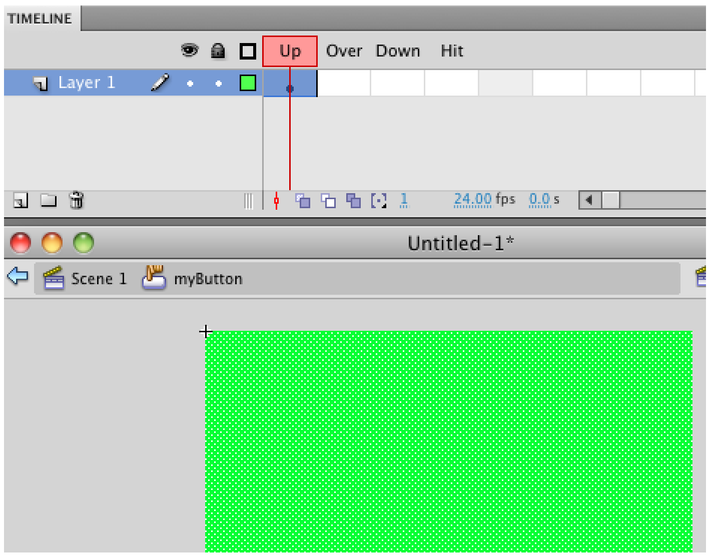
Cancel

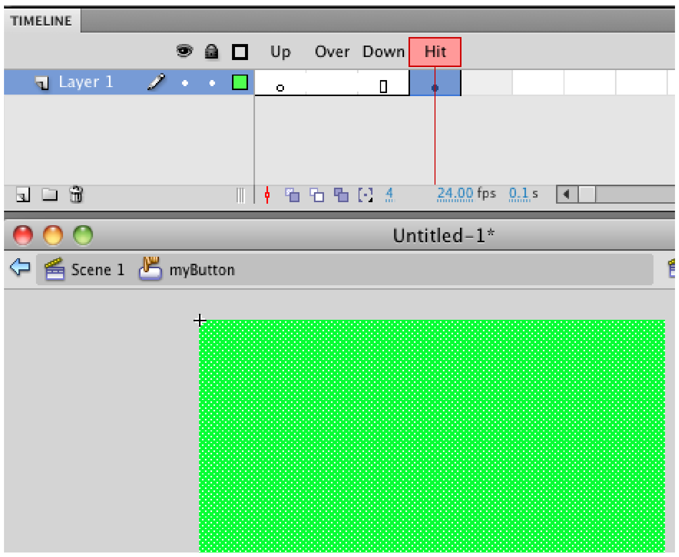
3. Making your button see-through.

Double click on the big rectangular button that you just made. This will take you into a new area of the timeline, where you can edit four different "Button States".

Right now, there should be only one keyframe in your button timeline, in the "Up" state. Click on that key frame and drag it to the last frame of the Button States, called "Hit" state. This should leave three blank keyframes in the "Up", "Over", and "Down" states.

Here's what it should look like before and after:





4. Giving your button an “instance name”

Leave the button editor and return to the main timeline of your Flash document.

You’ll notice that the button is now slighty transparent, with a turqoise hue. That’s because your button’s “active area” is now only visible in the “Hit” state. It is, essentially, an invisible button covering your entire ad.

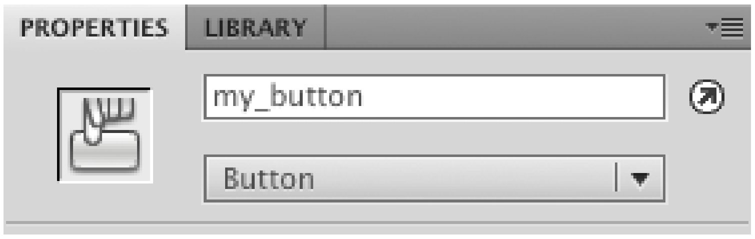
Click on your button, and find the “Properties” window. If the “Properties” window is not visible, go to the “Window...” menu and select “Properties”.

In the Properties window, the first thing you should see is a text field, containing the words *<Instance Name>*. Replace *<Instance Name>* with this:

my_button <--- ***Don’t forget: Capitilization and case matters!!***

An instance name is nothing more than a way of referring to your button later - a sort of nickname. (You might think that we named it already, when we typed “myButton” into the “Convert to Symbol” dialog box. But that was just a name for the Flash library object. Since you could have fifty copies of “myButton”, we need to name each instance of them individually to be able to reference it in code.)

Your Properties window should look like this:



5. Setting up the link to clickTAG (finally)

We’re done building the button - now let’s add a bit of ActionScript 2.0 to make it a link, and we’ll be done.

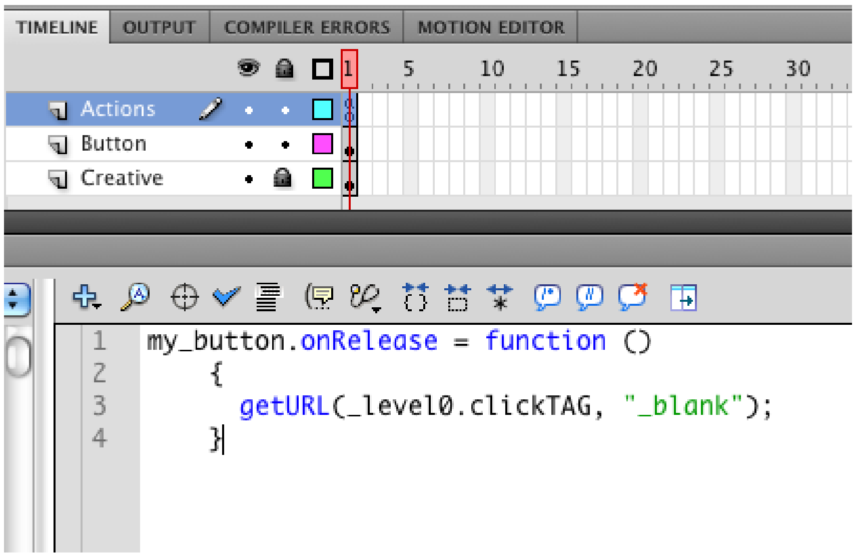
Return to your main timeline and lock the “Button” layer.

Right click on the first frame of the “Actions” layer, and select “Actions” from the dropdown menu. This should bring up a window with a text area in which to enter code.

Enter this exact code into the text area:

```
my_button.onRelease = function ()
   {getURL(_level0.clickTAG, "_blank");}
```

For reference, this is exactly how it should look:



```
my_button.onRelease = function ()
    {
      getURL(_level0.clickTAG, "_blank");
    }
```

We won't get into all the code, but basically, those four lines say *"Hey - you know that button called* my_button*? If someone clicks on it, you say 'clickTAG' to the ad server, and we'll take it from there."*

We're almost done.

Go to "File... > Publish Settings".

Under the "Flash" tab, make sure the "Script" option is set to "ActionScript 2.0" and the Flash Player is set to the appropriate version.

Click "Publish" at the bottom of the Publish Settings window to publish your SWF.

You're all done.

If you'd like to validate that your clickTAG function is working properly, you can use this tool:

http://dclkhelp.appspot.com/validator/

Adding clickTAG with ActionScript 3.0

1. Follow Steps 1-4 in the ActionScript 2.0 directions

The process of setting up a button, making it transparent, and giving it an instance name are the same in ActionScript 2.0 and ActionScript 3.0.

The major difference is the ActionScript code used to define what the button does.

Follow steps 1-4 in the ActionScript 2.0 directions, and then refer back here.

2. The ActionScript 3.0 code

After opening the Actions editor (the first part of Step 5 in the AS 2.0 instructions above), paste the following code into the Actions panel:

```
var _url:String = "";

if (LoaderInfo(root.loaderInfo).parameters.clickTAG)
       {  _url = LoaderInfo(root.loaderInfo).parameters.clickTAG;
          my_button.addEventListener(MouseEvent.MOUSE_UP, handleMouse);
       }

function handleMouse(event:MouseEvent):void
       {  navigateToURL(new URLRequest(_url), "_blank"); }
```

3. Publish

Go to “File... > Publish Settings”.

Under the “Flash” tab, make sure the “Script” option is set to “ActionScript 3.0” and the Flash Player is set to the appropriate version.

Click “Publish” at the bottom of the Publish Settings window to publish your SWF.

You’re all done.

If you’d like to validate that your clickTAG is working properly, you can use this tool:

http://dclkhelp.appspot.com/validator/