# AdOcean and gDE+ technical specification for preparing ad creatives (international)

## I. GENERAL REQUIREMENTS

Preparing ad creatives:

1. **You are required to submit the complete set of creatives (for the whole campaign period) at least 3 workdays before the campaign starts.** Otherwise we cannot guarantee the materials will be verified on time.

2. Creatives **must have dimensions and file size** allowed for a particular AdOcean (AO) or gemiusDirectEffect Plus (gDE+) client ordering the creatives. The limitations of creative size and file weight may be exceeded only when it is clearly defined by the campaign's assumptions and it has been accepted by AdOcean or a local Gemius subsidiary.

3. SWF (Flash) creatives prepared for AO/gDE+ **should have supplementary creatives in GIF or JPG format.** Supplementary files are not required for the following creative forms: toplayer, brandmark, interstitial. For expanding forms, the supplementary creatives should have the size of the "closed" state.

4. **A hexadecimal colour code of the background** should be also delivered with the complete set of creatives. It is not necessary if the background is white (#FFFFFF), since this is the default value in AO and gDE+ ad servers. Of course, this requirement may also be neglected when a creative has a layer type of form with transparent background. Neither AdOcean nor any local Gemius subsidiary can be held responsible for any changes in the creative's appearance resulting from negligence of the requirements included in this point.

5. **An active URL to which the creatives are to be linked** also has to be delivered with the complete set of creatives, **(it is the so called 'landing page URL').** If creatives use customized creative codes (in special cases), these must be sent as well. The landing page URL cannot be longer than 200 characters.

6. **The submitted advertisement cannot overload the processor of a standard computer by 25% or more.** A standard computer is assumed to have the following hardware and software components: Intel Celeron 1.7 GHz or AMD Duron 1.7 GHz processor with Flash Player 8 installed on the machine.

7. **SWF creatives can be made in any Flash version up to Flash 10, but AdOcean cannot assist in checking proper implementation of required functions if ActionScript 3 is used. The creative agency is always responsible for proper implementation of the functions specified in this document!** In case of ActionScript 3 being used, if checking by AdOcean is required then the creative agency has to send AdOcean either the FLA source or – more commonly – the appropriate parts of the ActionScript only.

8. It is not permitted to cover creatives containing transparent areas with invisible clickable buttons – only parts of the creative that are visible to the user may redirect to the advertiser's website (landing page) upon clicking.

9. It is forbidden to use scripts that change the browser window (its size, position, etc.). The only exceptions are creatives that assume actions of this type.

10. It is forbidden to use scripts to monitor user's interaction (e-tuning).

11. Ad creatives or creative codes causing errors or warnings during an advertisement's delivery will not be accepted.

12. **AdOcean does not engage in preparation of creative codes for advertisements.** We kindly request you to send us your custom creative codes with the set of materials.

13. SWF creatives may have any size, provided they comply with the maximum allowed creative size for a given website and the restrictions defined in the relevant AO or gDE+ client's (who orders the creatives for a campaign) contract.

14. If a creative utilizes **more than one clickTag ('multi-click' creative),** those clickTags should follow the following naming convention (**pay special attention to the character case**):

Landing URL 1 -> **_root.clickTag1**
Landing URL 2 -> **_root.clickTag2**, etc

15. When creatives are not served by AO/gDE+, the technical specification of the given third-party ad server should be used. This happens usually **when communicators or instant messengers are used** (e.g. MSN Messenger) or if the planner of the campaign makes such decision.

**Important notice!** Third-party ad servers have their own restrictions concerning such factors as, the size of a creative, number of frame rates per second and the functions that operate the creatives.

## II. THE COMPLETE SET OF MATERIALS AND NAMING CONVENTIONS

By the complete set of materials we mean:
- ✓ The complete set of creatives fulfilling the specification requirements,
- ✓ URL address(es) of active landing page(s),
- ✓ Custom creative codes, if they are required.

Naming conventions of files:
- ✓ It is recommended to use file names following this pattern: *creativeform_size_version*;
- ✓ You should only use lowercase letters of the English alphabet, underscores and numbers; it is recommended to avoid spaces in file names.

Landing page URLs:
- ✓ If more than one landing page is used in a campaign, you should always provide information about which URL a particular creative should be assigned to.

## III. GENERAL REQUIREMENTS FOR CREATIVE FORMS

### :: IMAGES ::

- ✓ Creatives must have the same size as defined in the media plan.
- ✓ Images must have the same file name as SWF files (differing only in file name extension) if they are meant to be used as alternative versions of Flash creatives.

### :: FLASH CREATIVES ::

- ✓ Creatives must have the same size as defined in the media plan.
- ✓ If a creative is not transparent, please provide the hexadecimal colour code to be used for the background.
- ✓ Regardless of their form, all SWF creatives must include the **_root.clickTag** parameter to enable the ad server to track clicks. Here is how to do it:

#### Actionscript 2

1. Create a new layer **on top** of a prepared animation.
2. Draw a rectangle on it, then transform it into a **button type symbol**.
3. Set this button to be **completely transparent**.
4. **Assign** the following **function call** to the button's **on (release)** event:

```
on (release)
{
    getURL(_root.clickTag, "_blank");
}
```

**IMPORTANT NOTICE:** please pay attention to the character case in the **_root.clickTag** variable!

### Actionscript 3

1. Create a directory with the name '*adocean'* in the same directory where you have saved your FLA file.
2. In this new directory, create a file by the name *'URLUtil.as'*.
3. Please paste the following piece of code in this file:

```
package adocean {
    import flash.external.ExternalInterface;
    public class URLUtil {
        protected static const WINDOW_OPEN_FUNCTION:String="window.open";
        public static function openWindow(url:String, window:String="_blank",
features:String=""):void {
            ExternalInterface.call(WINDOW_OPEN_FUNCTION, url, window, features);
            }
    }
}
```

Definition of the clickTag parameter:

1. Create a layer called *'URL'* and add a transparent button to it.
2. Rename this button to *'target_button'*.
3. Add the following piece of code to its first frame:

```
function targetUrlHandler(event:MouseEvent) {

      import adocean.URLUtil;

      var keyStr:String = 'clickTag';
      var valueStr:String;

      var paramObj:Object = LoaderInfo(this.root.loaderInfo).parameters;
      valueStr = String(paramObj[keyStr]);
      var url:String = valueStr;
      URLUtil.openWindow(url);
}
target_button.addEventListener(MouseEvent.CLICK, targetUrlHandler);
```

## IV. REQUIREMENTS FOR ADVANCED CREATIVE FORMS

### :: BRANDMARK ::

a) Your creative should have a closing cross with the following function call attached:

#### Actionscript 2

```
on (release)
{
	getURL("javascript:onFinishedPlaying();","_self");
}
```

#### Actionscript 3

```
function AdoceanClickHandler(event:MouseEvent) {
	var url:String = "javascript: onFinishedPlaying();";
	var request:URLRequest = new URLRequest(url);
	navigateToURL(request,"_self");
}
close_button.addEventListener(MouseEvent.CLICK, AdoceanClickHandler);
```

By the 'closing cross' we mean a rectangular button with a size at least 14x14 pixels designed to close the creative.

b) The creative should have a button that minimizes the advertisement to a small bar with the following function call:

#### Actionscript 2

```
on (release)
{
	getURL("javascript:bm_min();","_self");
}
```

#### Actionscript 3

```
function AdoceanClickHandler(event:MouseEvent){
	var url:String = "javascript: bm_min();";
	var request:URLRequest = new URLRequest(url);
	navigateToURL(request,"_self");
}
min_button.addEventListener(MouseEvent.CLICK, AdoceanClickHandler);
```

The bar cannot block the web page outside the bar's dimensions in any way. The size of the visible area in the minimized state is defined with numerical parameters in the *bm_min()* function in the creative code (set in the ad server by campaign managers). It is forbidden to let the transparent area of the standard sized advertisement be clickable or to make it impossible to click on links in the web page content.

**CAUTION:** in this type of creatives, the **button symbol cannot cover the bar's area**, it should not take users to the landing page when clicked!

c) The creative should have a button that resets the advertisement to its standard size with the following function call:

### Actionscript 2

```
on (release)
{
	getURL("javascript:bm_max();","_self");
}
```

### Actionscript 3

```
function AdoceanClickHandler(event:MouseEvent){
	var url:String = "javascript: bm_max();";
	var request:URLRequest = new URLRequest(url);
	navigateToURL(request,"_self");
}
max_button.addEventListener(MouseEvent.CLICK, AdoceanClickHandler);
```

The function is responsible for expanding the visible area of the standard size. This size should be set with numerical values in *bm_max()* function in the creative code (done in the ad server by campaign managers).

### :: TOPLAYER ::

a) Your creative should have a closing cross with the following function call attached:

### Actionscript 2

```
on (release)
{
	getURL("javascript:onFinishedPlayingCross();","_self");
}
```

### Actionscript 3

```
function AdoceanClickHandler(event:MouseEvent) {
      var url:String = "javascript: onFinishedPlayingCross();";
      var request:URLRequest = new URLRequest(url);
      navigateToURL(request,"_self");
}
close_button.addEventListener(MouseEvent.CLICK, AdoceanClickHandler);
```

By the 'closing cross' we mean a rectangular button with a size at least 14x14 pixels designed to close a creative.

b) In the last empty frame of the flash animation you have to add:

### Actionscript 2

```
getURL("javascript:onFinishedPlaying();","_self");
```

### Actionscript 3

```
var url:String = "javascript: onFinishedPlaying();";
var request:URLRequest = new URLRequest(url);
navigateToURL(request,"_self");
```

### :: EXPANDING ::

Your creative should have an active area over its whole surface and throughout the complete animation sequence (both in folded and unfolded state) with the following function calls attached:

### Actionscript 2

```
on (rollOver)
{
      getURL("javascript:doexpand();","_self");
}

on (rollOut)
{
      getURL("javascript:dolittle();","_self");
}
```

### Actionscript 3

```
function targetUrlHandler(event:MouseEvent) {

      import adocean.URLUtil;

      var keyStr:String = 'clickTag';
      var valueStr:String;

      var paramObj:Object =LoaderInfo(this.root.loaderInfo).parameters;
      valueStr = String(paramObj[keyStr]);
      var url:String = valueStr;
      URLUtil.openWindow(url);
}
target_button.addEventListener(MouseEvent.CLICK, targetUrlHandler);

function MouseRollOver(event:MouseEvent) {
      var url:String = "javascript:doexpand();";
      var request:URLRequest = new URLRequest(url);
      navigateToURL(request,"_self");
}

function MouseRollOut(event:MouseEvent) {
      var url:String = "javascript:dolittle();";
      var request:URLRequest = new URLRequest(url);
      navigateToURL(request,"_self");
}
target_button.addEventListener(MouseEvent.ROLL_OVER, MouseRollOver);
target_button.addEventListener(MouseEvent.ROLL_OUT, MouseRollOut);
```

The size of the visible area both in the folded and unfolded state is defined by the numerical parameters in *dolittle()* and *doexpand()* functions in the creative code (set in the ad server by campaign managers). When moving the mouse pointer over the advertisement, it should unfold; after moving the cursor away from the advertisement's area, the banner should fold back to its original state and size. Be aware that some publishers only allow expansion on a click event. Campaign managers can always provide such information if needed.

If the creative is supposed to expand automatically only on the first occasion, this or a similar solution can be used:

### Actionscript 2

```
// AD_ID: a unique value that we associate with a given creative, e.g. „Mobile_240x120_publisher1”

so = SharedObject.getLocal (AD_ID);
if (so.data.num_of_disp == 1) {
      getURL("javascript:dolittle();","_self");
} else {
      var so:SharedObject = SharedObject.getLocal(AD_ID);
      so.data.num_of_disp = 1;
      so.flush();
      getURL("javascript:doexpand();","_self");
}
```

The above solution uses Local Shared Objects in Flash. These objects provide a functionality similar to cookies and makes it possible to store certain variables on a user's computer. This way we are able to store information about a creative, whether it is being displayed for the first time or not. The above code's minimum requirements are Flash 6 and ActionScript 1.

To ensure that automatic expansion takes place once per day, follow the same procedure, i.e. use Local Shared Objects. However, in this case we will store the date when the creative has been delivered and match it next time with the current date (the functions *new Date()* and *getDate()* can be used).

**CAUTION:** in case of expand skyscrapers please ensure that they unfold to the right, left, or in both directions. Flash creatives should follow suit (alignment of a small size animation with the expanded one is crucial). Resizing of a DIV containing a creative is done in the creative code in AO/gDE+ ad server, at the moment when it receives the proper function calls (*dolittle()* or *doexpand()* (in case of expanding forms) from a Flash banner.

**CAUTION:** expanding creatives will not work when placed inside an iframe by the publisher! Please consult the publisher about possible placement of a creative before the campaign starts.

### :: PUSH ::

Push creatives have to be displayed in the minimized state when they get delivered to a web page. When fully downloaded, a SWF creative calls the *dopushlock()* function. It automatically expands the banner's visible area (the DIV that contains the banner) to the maximized size. After 5 seconds the *dopushunlock()* function is called, starting to gradually roll up the banner's DIV container and resetting it to its minimized state. After this process has been completed, the *dopushon()* and *dopushoff()* functions (which operate the same way

as the ones described above) are unblocked; their call is initiated by a user's actions (moving the cursor over/off of a creative). The animation effects should be blocked in the SWF creative while the functions get executed and the DIV container changes its size.

a) The entire surface of your creative should be active throughout the complete animation sequence (both in the default and pushed down state), with the following function calls attached:

### Actionscript 2

```
on (rollOver)
{
	getURL("javascript:dopushon();", "_self");
}

on (rollOut)
{
	getURL("javascript:dopushoff();", "_self");
}
```

### Actionscript 3

```
function targetUrlHandler(event:MouseEvent) {

	import adocean.URLUtil;

	var keyStr:String = 'clickTag';
	var valueStr:String;

	var paramObj:Object =LoaderInfo(this.root.loaderInfo).parameters;
	valueStr = String(paramObj[keyStr]);
	var url:String = valueStr;
	URLUtil.openWindow(url);
}
target_button.addEventListener(MouseEvent.CLICK, targetUrlHandler);

function MouseRollOver(event:MouseEvent) {
	var url:String = "javascript:dopushon();";
	var request:URLRequest = new URLRequest(url);
	navigateToURL(request,"_self");
}
```

```
function MouseRollOut(event:MouseEvent) {
      var url:String = "javascript:dopushoff();";
      var request:URLRequest = new URLRequest(url);
      navigateToURL(request,"_self");
}
target_button.addEventListener(MouseEvent.ROLL_OVER, MouseRollOver);
target_button.addEventListener(MouseEvent.ROLL_OUT, MouseRollOut);
```

b) When fully downloaded, the creative should call the following function:

**Actionscript 2**

```
getURL("javascript:dopushlock();", "_self");
```

**Actionscript 3**

```
var url:String = "javascript:dopushlock();";
var request:URLRequest = new URLRequest(url);
navigateToURL(request,"_self");
```

c) 5 seconds after starting its animation for the first time, the creative should call the following function:

**Actionscript 2**

```
getURL("javascript:dopushunlock();", "_self");
```

**Actionscript 3**

```
var url:String = "javascript:dopushunlock();";
var request:URLRequest = new URLRequest(url);
navigateToURL(request,"_self");
```

**CAUTION:** push creatives will not work when placed inside an iframe by the publisher! Please consult the publisher about possible placement of the creative before the campaign starts.

## :: PEEL AWAY ::

The entire surface of your creative should be active throughout the complete animation sequence (both in the default and pushed down state), with the following function calls attached:

### Actionscript 2

```
on (rollOver)
{
	getURL("javascript:doexpand();","_self");
}

on (rollOut)
{
	getURL("javascript:dolittle();","_self");
}
```

### Actionscript 3

```
function targetUrlHandler(event:MouseEvent) {

	import adocean.URLUtil;

	var keyStr:String = 'clickTag';
	var valueStr:String;

	var paramObj:Object =LoaderInfo(this.root.loaderInfo).parameters;
	valueStr = String(paramObj[keyStr]);
	var url:String = valueStr;
	URLUtil.openWindow(url);
}
target_button.addEventListener(MouseEvent.CLICK, targetUrlHandler);

function MouseRollOver(event:MouseEvent) {
	var url:String = "javascript:doexpand();";
	var request:URLRequest = new URLRequest(url);
	navigateToURL(request,"_self");
}
```

```
function MouseRollOut(event:MouseEvent) {
      var url:String = "javascript:dolittle();";
      var request:URLRequest = new URLRequest(url);
      navigateToURL(request,"_self");
}
target_button.addEventListener(MouseEvent.ROLL_OVER, MouseRollOver);
target_button.addEventListener(MouseEvent.ROLL_OUT, MouseRollOut);
```

The size of the visible area in either the folded and unfolded state is defined with numerical parameters in *dolittle()* and *doexpand()* functions in the creative code (this is done in the ad server by a campaign manager). When moving the mouse pointer over an advertisement, it should unfold, and after moving it away the advertisement's area, the banner should fold back to the original state.

**:: SCROLLER / SCROLL FOOTER ::**

a) In case of SWF creatives, the scrolling should be carried into effect in the creative.
b) In case of IMAGE type creatives, the image should be static – the scrolling will be carried into effect by special functions in the creative code.

**:: POP-UP & POP-UNDER ::**

We advise that pop-up and pop-under creatives should be prepared in at least Flash 8 version. In earlier Flash versions these creative forms do not work properly in Internet Explorer browsers because they activate a built-in pop-up blocker instead of simply opening the landing page in a new tab or window.

## MORE THINGS WORTH KNOWING

### How to check whether an .swf file appropriately contains _root.clickTag to count the number of clicks?

Copy the URL of the given SWF file from the ad server into your browser's location bar, or if you do not have access to the ad server, then just upload it for testing to any web server's public directory. Add the following string at the end without additional spaces: **?clickTag=** and an arbitrary URL that is different from the campaign's landing page URL, e.g. http://www.gemius.com. Then click on the creative. If the newly opened window loads the web page that you pasted before at the end of the URL address, then everything works fine.

*Last update: 27th January, 2010*