Technical Specifications for Preparing HTML5 Creatives in AOcean

www.gemius.com

The purpose of this document is to outline the general rules and requirements for the preparation of HTML5 creatives delivered via AdOcean servers.

General Information

1. Creatives on websites

Creatives are embedded in a webpage inside the <iframe> tag. All parameters are passed to the creative via the URI fragment[1] in the source of the <iframe> tag. They take the following form: key=value, and are separated by the ampersand symbol ('&').

Important: *Parameters are attached to the URI of the creative automatically by the emitter server. The URI fragment should not be used in the code of the creative in any other way than described in this manual.*

In order to have access to the parameters (including access to the redirect variable), the following code has to be implemented at the beginning of the <body> of the creative:

```
<script type='text/javascript'>
  var parsed = (document.location.href.split('#')[1]||'').split('&');
  var params = parsed.reduce(function (params, param) {
    var param = param.split('=');
    params[param[0]] = decodeURIComponent(param.slice(1).join('='));
    return params;
  }, {});
</script>
```

2. Creatives in mobile apps

In the case of creatives in mobile apps, it is necessary to follow the MRAID standard. gemius SDK supports a basic set of commands of the official MRAID standard (https://www.iab.net/media/file/IAB_MRAID_v2_FINAL.pdf). For more information, please refer to gemiusSDK documentation.

Creative Assets

An HTML creative should be built with as few elements as possible. The fewer creative assets, the faster the creative loads and is visible for the user.

Relative paths should be used for all requests for elements (excluding elements hosted on external servers).

[1] URI fragment is the part of the URI string after the # symbol.

Gemius SA
18 B Postepu Street
02-676 Warsaw, Poland

Phone: + 48 22 390 90 90
+ 48 22 378 30 50
Fax: + 48 22 874 41 01

contact@gemius.com
www.gemius.com

CSS and JavaScript can be included inside the main HTML file, which will decrease the number of elements in the creative.

Graphic elements can also be included in the HTML file (by encoding with Base64 algorithm). It will increase the size of the HTML file, but it will not be necessary to load image files. It will also ensure that all elements are available in the creative at the same time.

If a campaign is delivered on an HTTPS website, all requests going from the creative to the HTTP content will be blocked by modern browsers. This can cause a failure to deliver the creative. Thus, it needs to be ensured that an HTML creative can be also delivered on an HTTPS website and that all links to external assets are working correctly.

Uploading Creatives

There are two ways to upload an HTML5 creative into our systems. You can inline the whole creative into one HTML file or you can upload a .zip file.

a) single file creative

To upload an HTML5 creative as a single file, you need to embed all CSS and JS files into an HTML file. You should also inline images using the Base64 algorithm or host images on an external server and link them inside the HTML file.

b) zip file creative

It is also possible to upload a .zip file containing all assets. All relative paths will be preserved. The whole creative will be uploaded to the same folder on our emitter server. In order to upload a creative in the .zip format, the file must meet the following requirements:

- must contain at least one HTML file,
- should contain **index.html** file (it is especially important if inside the .zip file there is more than one HTML file),
- main HTML file must be at the top level or in a root folder,
- can have root folder,
- must not have multiple root folders,
- can contain multiple HTML files.

Example of a .zip file structure with a root folder:

```
+ creative.zip
+-+ /rootfolder
  +-+ index.html
    + image.png
    + style.css
```

Gemius SA
18 B Postepu Street
02-676 Warsaw, Poland

Phone: + 48 22 390 90 90
+ 48 22 378 30 50
Fax: + 48 22 874 41 01

contact@gemius.com
www.gemius.com

Example of a .zip file structure without a root folder:

```
+ creative.zip
+-+ index.html
  + image.png
  + style.css
```

Example of an **invalid** .zip file structure with multiple root folders:

```
+ creative.zip
+-+ /rootfolder1
| +-+ index.html
|   + image.png
|   + style.css
|
+-+ /rootfolder2
  +-+ index.html
    + image2.png
    + style2.css
```

Click Tags

1. creative with a click placeholder

This type of a creative should include a variable to which the redirect link can be assigned. After the code described in the General Information part is executed, the click tag will become available in the `params.clickTag` variable.

Below is an example with a variable to measure clicks in the creative, which in the code is referred to as `click`, and in the interface as *clickTag*.

```
<script>
  ...
  //click='http://www.gemius.pl'; // constant URL without tracking
  click=params.clickTag;  // assign redirect variable read from parameters
  ...
</script>
```

If you set a different name for a click tag in the AO interface or you use multiple click tags, you have to use appropriate variable names, for example:

Gemius SA
18 B Postepu Street
02-676 Warsaw, Poland

Phone: + 48 22 390 90 90
+ 48 22 378 30 50
Fax: + 48 22 874 41 01

contact@gemius.com
www.gemius.com

```
params.clickthrough;
params.clickTAG;
params.clickTag0;
params.clickTag1;
```

2. creative without a click placeholder

If a creative does not include a click variable and you do not have a place to assign an appropriate parameter, you can try to figure out which element in the creative should be clickable and surround it with <a> tags. For instance, if you have an HTML5 creative containing an image only, then the <img> tag should be surrounded with <a> tags. If there are tags, you should choose the most outer one.

You can set the “href” attribute of the <a> tag using this script. Remember that this script has to be executed after the measured element.

```
<script>
  // setting href of an <A> tag with ID: "ID_OF_AN_<A>_TAG"
  // with redirect link.
  var creativeLink = document.getElementById("ID_OF_AN_<A>_TAG");
  creativeLink.href = params.clickTag;
</script>
```

Example of the full code:

Pure creative without codes

```
<!DOCTYPE html> <!-- EXAMPLE CREATIVE -->
<html>
<head>
</head>
<body>
  <img src="...">
</body>
</html
```

To properly measure clicks for this type of a creative, you need to make changes as in the example below:

Creative with added tracking codes

```
<!DOCTYPE html> <!-- EXAMPLE CREATIVE -->
<html>
<head>
</head>
<body>
  <script type='text/javascript'>
    var parsed = (document.location.href.split('#')[1]||'').split('&');
    var params = parsed.reduce(function (params, param) {
      var param = param.split('=');
      params[param[0]] = decodeURIComponent(param.slice(1).join('='));
      return params;
    }, {});
  </script>

  <a id='creativelink' target='_blank'>
    <img src='...'>
  </a>

  <script>
    var creativeLink = document.getElementById('creativelink');
    creativeLink.href = params.clickTag;
  </script>
</body>
</html
```

Important: *Please note that before the* `params` *variable is called, the code from the General Information part has to be executed. Assigning a redirect link before executing that code will generate an error.*

a. Click tracking according the MRAID standard

   To properly measure clicks for this type of a creative, use mraid.open() method as described in the MRAID specification:

```
<a onClick="mraid.open(' creativelink ')"><img src="..."></a>
```

Interactions

Creatives with more advanced behavior (e.g. Expand, Toplayer) require integration with template codes responsible for changing size, closing the creative, etc. Names of interactions are passed to the creative through parameters in the URI fragment (the same way as in the case of clickTag). In the Expand template, there are **doexpand** and **dolittle** parameters, and in the Toplayer template there is the **onCrossClick** parameter.

The code responsible for the execution of an appropriate function looks as follows:

```
// replace PARAMETER_NAME with proper name (e.g. doexpand, onCrossClick etc.)
window.parent.postMessage(params.PARAMETER_NAME, '*');
```

The fragment of the code that detects when mouse is over or out of the BODY of the banner looks as follows:

```
<script type='text/javascript'>
  ...
  // Adding expanding function when mouse will be over BODY of the banner
  document.body.addEventListener('mouseenter', function() {
    window.parent.postMessage(params.doexpand, '*');
  });
  // Adding reduce function when mouse will be out of BODY of the banner
  document.body.addEventListener('mouseleave', function() {
    window.parent.postMessage(params.dolittle, '*');
  });
  ...
</script>
```

Important: *To be able to use those functions as described above, you need to select the „Manual expand" field when adding a creative in the interface. Without this field selected, expanding functions are added automatically to the body of the HTML5 creative.*

Example of the code of the Toplayer creative containing an element with an ID equal to „closeButton". After pressing this element, the banner should close immediately.

```
<script type='text/javascript'>
  ...
  // Adding function, which closes banner, after click
  // element with id='closeButton'
  document.getElementById('closeButton').
    addEventListener('click', function() {
      window.parent.postMessage(params.onCrossClick, '*');
    });
  ...
</script>
```

Important: *The above functions are responsible for the behavior of the parent element, inside which the creative is embedded. Additional actions (e.g. change in graphic elements) related to this behavior (e.g. resizing banner) have to be done inside the creative.*

Gemius SA
18 B Postepu Street
02-676 Warsaw, Poland

Phone: + 48 22 390 90 90
+ 48 22 378 30 50
Fax: + 48 22 874 41 01

contact@gemius.com
www.gemius.com

Examples

Below there are two examples with correctly implemented redirect links.

a) Creative that uses <a> tags

The simplest way to redirect a creative is to surround the right element with <a> tags.

```
<!doctype html>
<html>
<head></head>
<body>
  <script>
    // here we read all parameters and store them in params variable
    var parsed = (document.location.href.split('#')[1]||'').split('&');
    var params = parsed.reduce(function (params, param) {
      var param = param.split('=');
      params[param[0]] = decodeURIComponent(param.slice(1).join('='));
      return params;
    }, {});
  </script>

  <a href='#' target='_blank' id='creativelink'>
    <img src='... ' />
  </a>

  <script>
    // here we set up proper redirection of an <a> tag
    // which has id = 'creativelink'
    document.getElementById('creativelink').href = params.clickTag;
  </script>
</body>
</html>
```

b) Example of HTML5 MRAID compliant creative

Gemius SA
18 B Postepu Street
02-676 Warsaw, Poland

Phone: + 48 22 390 90 90
+ 48 22 378 30 50
Fax: + 48 22 874 41 01

contact@gemius.com
www.gemius.com

```
<html>
<head>
  <script src="mraid.js"></script>
</head>
<body style="background:transparent;">
  <div style="text-align:center">
    <div style="position:relative;margin:auto;">
    <a onClick="mraid.open(' creativelink ')"><img src="..."></a>
    </div>
    <script
type="text/javascript">checkForMraid=function(){'undefined'!=typeof
mraid&&('loading'===mraid.getState()?mraid.addEventListener('ready',mr
aidReady):mraidReady())},mraidReady=function(){mraid.removeEventListen
er('ready',mraidReady)},checkForMraid();</script>
  </div>
</body>
</html>
```

c) Swiffy creative

Swiffy allows the conversion from Flash to HTML. The conversion includes also all parameters passed to Flash (FlashVars). The below code is an example of the correct implementation of appropriate scripts. All manual insertions to the code are colored blue. Please note that all FlashVars parameters are passed in one command, this is why there is no need to execute the code described in the General Information part. All parameters of the creative are included in just one line of the code. Please note that the original .swf file has to be properly defined. All Flash functions should be defined according to our AdOcean technical specifications for preparing Flash creatives. For example, if it is a banner, then a clickTag variable should be included. In case a different name is used (e.g. clickTAG, clickthrough), changes should also be made in the interface.

If the converted creative is Expand or Toplayer, please make sure that correct functions have been defined for the Flash banner (e.g. dolittle, doexpand, onCrossClick).

```
<!doctype html>
<html>
<head>
  <meta charset='utf-8'>
  <meta http-equiv='X-UA-Compatible' content='IE=edge'>
  <title>Swiffy Output</title>
  <script type='text/javascript'
src='https://www.gstatic.com/swiffy/v7.3.0/runtime.js'></script>
  <script>
    swiffyobject = {'as3':true, 'frameRate':12,...};
    // here is a long swiffy code
  </script>
  <style>html, body {width: 100%; height: 100%}</style>
</head>
<body style='margin: 0; overflow: hidden; '>
  <div id='swiffycontainer' style='width: 750px; height: 300px'>
  </div>
  <script>

    // orginal code
    var stage = new swiffy.Stage(document.getElementById('swiffycontainer'),
      swiffyobject, {});

    // you can set up transparent background color by this command
    // below line is optional
    stage.setBackground(null);

    // forwarding all parameters to swiffy
    stage.setFlashVars(document.location.hash.split('#')[1]);

    // original code
    stage.start();

  </script>
</body>
</html>
```

Gemius SA
18 B Postepu Street
02-676 Warsaw, Poland

Phone: + 48 22 390 90 90
+ 48 22 378 30 50
Fax: + 48 22 874 41 01

contact@gemius.com
www.gemius.com

Gemius SA

18B Postepu Street

02-676 Warsaw, Poland

Phone: + 48 22 390 90 90

+ 48 22 378 30 50

Fax: + 48 22 874 41 01

contact@gemius.com

www.gemius.com