

admazing AG - Technische Spezifikation

Inhaltsverzeichnis

1. Allgemeine Vorgaben für alle Werbemittel	2
2. Formattabelle.....	2
3. Flash Spezifikationen	2
a. „clickTag“ Implementierung	2
4. Multilink.....	3
5. Expandable Ad	3
6. Floating Ad (Layer Ad).....	3
7. Streaming Banner.....	4
8. Interaction Tracking	4
9. Codebeispiele	4
a. clickTAG-Implementierung in ActionScript 2	4
b. clickTAG-Implementierung in ActionScript 3	5
c. Multilink-Implementierung in ActionScript 2	6
d. Multilink-Implementierung in ActionScript 3	7
e. Expandable/Floating Ad Implementierung in ActionScript 2	8
f. Expandable/Floating Ad Implementierung in ActionScript 3	9

1. Allgemeine Vorgaben für alle Werbemittel

- Werbemittel müssen in der vereinbarten Pixel-Grösse angeliefert werden.
- Das maximale Gewicht der Formate darf nicht überschritten werden.
- Die Variable „clickTAG“ muss für die Klickbarkeit des Flash-Werbemittels in exakt dieser Gross-/Kleinschreibweise verwendet werden.
- Die Zielurl muss mit den Werbemitteln separat angeliefert werden (Zielurl darf nicht fix in den Flash-Banner eingesetzt werden).
- Die Werbemittel-Anlieferung sollte spätestens 4 Werkzeuge vor Kampagnenbeginn erfolgen.
- HTML-Werbemittel müssen als *.txt-File oder als *.html-File in einer *.zip-Datei versendet werden, nicht als E-Mail-Inhalt.
- Senden Sie alle Werbemittel mit den notwendigen Angaben an Ihren Ansprechpartner.

2. Formattabelle

Format	Max. Grösse (Pixel)	Max. Gewicht (kB)
Fullbanner	468x60	20
Leaderboard	728x90	40
Half Page Ad	300x600	60
Maxiboard	994x118	40
Monsterboard	468x400	60
Rectangle	300x250	40
Skyscraper	120x600 / 160x600	40
Sticky Monster Sky	245x770	60

3. Flash Spezifikationen

- Die Anlieferung besteht aus einer SWF-Datei und einem alternativen Werbemittel als GIF.
- Die Gestaltung der Werbemittel steht Ihnen frei.
- Ein Flash-Werbemittel mit ActionScript 2 darf maximal für den Flash Player 8 exportiert werden (ansonsten wird der Popup-Blocker im Internet Explorer aktiv).
- Für ein Flash-Werbemittel mit ActionScript 3 muss die Spezifikation beachtet werden ([siehe Codebeispiele](#)).
- Achtung: Bei Verwendung der Funktion „navigateToURL“ (ActionScript 3) erscheint der Popup-Blocker des Internet Explorers beim Klick auf das Werbemittel.

a. „clickTag“ Implementierung

- Damit der AdServer einen Klick zählen kann, wird statt der Zielurl die Variable „clickTAG“ in den ActionScript-Code des Flash-Werbemittels integriert.
- Diese Variable gewährleistet eine Änderung der Zielurl während der laufenden Kampagne ohne dass das Werbemittel verändert werden muss.
- Codebeispiele in [ActionScript 2](#) und [ActionScript 3](#)
- Codebeispiele zum Download für [ActionScript 2](#) und [ActionScript 3](#)

4. Multilink

- Falls das Werbemittel mehrere klickbare Bereiche mit verschiedenen Zielurls enthält, kann die „clickTAG“-Variable für sämtliche Zielurls um eine Nummer erweitert werden.
- Beispiel:
„clickTAG1“ entspricht der Zielurl „http://www.zielseite.ch/unterseite_1/“
„clickTAG2“ entspricht der Zielurl „http://www.zielseite.ch/unterseite_2/“
„clickTAG3“ entspricht der Zielurl „http://www.zielseite.ch/unterseite_3.html“
- Codebeispiele in [ActionScript 2](#) und [ActionScript 3](#)
- Codebeispiele zum Download für [ActionScript 2](#) und [ActionScript 3](#)

5. Expandable Ad

- Die Flash-Spezifikationen sind hier ebenfalls zu beachten ([siehe Abschnitt „Flash Spezifikationen“](#)).
- Im Expandable Ad müssen zwingend die JavaScript-Funktionen für das Öffnen und das Schliessen des Banners verwendet werden.
- Das Öffnen geschieht entweder per Mouse-Over, direkt bei der Anzeige oder durch Klick z.B. auf einen Button.
- Wenn der Banner mit einem Button oder direkt bei der Anzeige geöffnet wird, muss zwingend ein Close-Button vorhanden sein, auch wenn ein automatisches Schliessen nach einer bestimmten Dauer eingesetzt wird.
- Das Schliessen des Banners wird per Mouse-Out oder durch einen Klick auf den Close-Button ausgelöst.
- Die automatische Schliessen-Funktion soll nach ca. 7 bis 10 Sekunden aufgerufen werden. Banner die durch Mausaktionen öffnen und schliessen sollen nicht durch die automatische Schliessen-Funktion geschlossen werden.
- Der Aufruf der JavaScript-Funktion zum Öffnen des visuellen Bereichs des Banner wird vor der Animation ausgeführt und lautet „adm_expand_layer();“ ([siehe Codebeispiel](#)).
- Die JavaScript-Funktion zum Schliessen dieses Bereichs wird nach der Schliessen-Animation des Banners aufgerufen und lautet „adm_collapse_layer();“ ([siehe Codebeispiel](#)).
- Für den Close-Button muss nach dem Klick folgende Funktion ausgeführt werden: „adm_close_layer();“ ([siehe Codebeispiel](#)).
- In Frame 1 der Datei darf kein JavaScript-Befehl aufgerufen werden. Dies kann zu Problemen bei der Darstellung der Webseite führen, auf welcher der Banner geschaltet wird. Die Steuerung des Anfangszustandes (normal oder ausgeklappt) übernimmt das HTML-Template (es darf am Anfang also weder die Expand-, Collapse- noch die Close-Funktion aufgerufen werden).
- Da die Werbemittel mit dem Parameter „wmode=transparent“ ausgeliefert werden, ist sicherzustellen, dass ein sichtbarer Hintergrund als Objekt im Flash eingebunden ist, wo einer vorhanden sein muss.
- Codebeispiele in [ActionScript 2](#) und [ActionScript 3](#)
- Codebeispiele zum Download für [ActionScript 2](#) und [ActionScript 3](#)

6. Floating Ad (Layer Ad)

- Die Flash-Spezifikationen sind hier ebenfalls zu beachten ([siehe Abschnitt „Flash Spezifikationen“](#)).
- Floating Ads (Layer Ads) erfordern zwingend einen gut sichtbaren Close-Button.
- Für den Close-Button muss nach dem Klick folgende Funktion ausgeführt werden: „adm_close_layer();“ ([siehe Codebeispiel](#)).
- Die automatische Schliessen-Funktion muss nach ca. 7 bis 10 Sekunden aufgerufen werden.

- In Frame 1 der Datei darf kein JavaScript-Befehl aufgerufen werden. Dies kann zu Problemen bei der Darstellung der Webseite führen, auf welcher der Banner geschaltet wird.
- Da die Werbemittel mit dem Parameter „wmode=transparent“ ausgeliefert werden, ist sicherzustellen, dass ein sichtbarer Hintergrund als Objekt im Flash eingebunden ist, wo einer vorhanden sein muss.
- Codebeispiele in [ActionScript 2](#) und [ActionScript 3](#)
- Codebeispiele zum Download für [ActionScript 2](#) und [ActionScript 3](#)

7. Streaming Banner

- Die Flash-Spezifikationen sind hier ebenfalls zu beachten ([siehe Abschnitt „Flash Spezifikationen“](#)).
- Ein Streaming Banner besteht aus einem SWF-File als Player und einem FLV-File als Video- / Audio-Material.
- Der Banner (SWF-File, Player) darf kein Video- / Audio-Material beinhalten.
- Das Video (können je nach Grösse auch mehrere Videos sein) wird zu einem FLV umgewandelt.
- Verweise (Video-URL, Links oder Bilder die gestreamt werden) dürfen im Banner (SWF-File, Player) nie fix definiert sein. Alle Ressourcen werden vom AdServer als Variablen an das Flash-File des Banners übergeben.
- Der AdServer übergibt die URL zu der FLV-Datei über den Parameter „flvfile“.
- Der Sound darf bei der Anzeige des Banners nicht sofort abgespielt werden. Der Benutzer kann über einen Sound-Button den Sound aktivieren.
- Anmerkung: Je nach Webseite kann es Spezialregelungen geben.

8. Interaction Tracking

- Über das Interaction Tracking können verschiedene Events (zum Beispiel: Sound-On, Sound-Off, Play, Stop, Ende des Videos, usw.) innerhalb des Banners gemessen werden.
- Um dies Messen zu können, muss eine JavaScript-Funktion aufgerufen werden. Diese lautet „adm_track_event([event_name]);“. Der Platzhalter [event_name] muss hier mit dem entsprechenden Namen des Events ersetzt werden.
- In Frame 1 der Datei darf kein JavaScript-Befehl aufgerufen werden. Dies kann zu Problemen bei der Darstellung der Webseite führen, auf welcher der Banner geschaltet wird.
- Die zu trackenden Events müssen zusammen mit dem entsprechenden Kampagnen-Berater definiert werden.

9. Codebeispiele

a. clickTAG-Implementierung in ActionScript 2

```
/**
 * define onRelease event on a clickable button instance
 */
_root.clickarea.onRelease = function() {
    getURL(_root.clickTAG, '_blank');
};
```

b. clickTAG-Implementierung in ActionScript 3

```

/**
 * searches for a clickTAG, case-insensitive search
 */
function clickTagGet():String {
    for (var key:String in
LoaderInfo(this.root.loaderInfo).parameters) {
        if (key.toLowerCase()=='clicktag') {
            return
LoaderInfo(this.root.loaderInfo).parameters[key];
        }
    }
    return null;
}

/**
 * searches for a clickTARGET, case-insensitive search
 */
function clickTargetGet():String {
    for (var key:String in
LoaderInfo(this.root.loaderInfo).parameters) {
        if (key.toLowerCase()=='clicktarget') {
            return
LoaderInfo(this.root.loaderInfo).parameters[key];
        }
    }
    return null;
}

/**
 * Opens the clickTAG and prevents to be blocked as a popup
 */
function clickTagOpen(url:String=null,
target:String=null):void {
    if (! target) {
        target=clickTargetGet();
        if (! target) {
            target='_blank';// default: to new window
        }
    }
    if (! url) {
        url=clickTagGet();
    }
    if (url) {
        var req:URLRequest=new URLRequest(url);
        if (! ExternalInterface.available) {
            navigateToURL(req, target);
        } else {
            var
strUserAgent:String=String(ExternalInterface.call("function()
{ return navigator.userAgent; }")).toLowerCase();
            if (strUserAgent.indexOf("firefox") != -1 ||
(strUserAgent.indexOf("msie") != -1 &&
uint(strUserAgent.substr(strUserAgent.indexOf("msie") + 5, 3))
>= 6)) {
                ExternalInterface.call("window.open",
req.url, target);
            } else {
                navigateToURL(req, target);
            }
        }
    }
}

```

```

    } else {
        trace('clickTAG was not found.');
```

c. Multilink-Implementierung in ActionScript 2

```

/**
 * searches for the clickTAG, case-insensitive search
 */
function clickTagGet(clickNumber:String) {
    var key:String = '';
    for (key in _root) {
        if (typeof (_root[key]) == 'string' &&
key.toLowerCase() == 'clicktag' + clickNumber) {
            return _root[key];
        }
    }
    return null;
}

/**
 * searches for the clickTARGET, case-insensitive search
 */
function clickTargetGet(clickNumber:String) {
    var key:String = '';
    for (key in _root) {
        if (typeof (_root[key]) == 'string' &&
key.toLowerCase() == 'clicktarget' + clickNumber) {
            return _root[key];
        }
    }
    return null;
}

/**
 * opens the clickTAG, parameters (url and target) are
optional
 */
function clickTagOpen(clickNumber:String, url:String,
target:String) {
    if (!target) {
        target = clickTargetGet(clickNumber);
        if (!target) {
            target = '_blank'; // default: to new window
        }
    }
    if (!url) {
        url = clickTagGet(clickNumber);
    }
    if (url) {
        getURL( url, target );
    } else {
        trace('clickTAG' + clickNumber + ' was not found.');
```

```

}

/**
 * define onRelease event on a clickable button instance
 */
_root.clickarea1.onRelease = function() {
    clickTagOpen("1");
};
_root.clickarea2.onRelease = function() {
    clickTagOpen("2");
}

```

d. Multilink-Implementierung in ActionScript 3

```

/**
 * searches for a clickTAG, case-insensitive search
 */
function clickTagGet(clickNumber:String):String {
    for (var key:String in
LoaderInfo(this.root.loaderInfo).parameters) {
        if (key.toLowerCase()=='clicktag' + clickNumber) {
            return
LoaderInfo(this.root.loaderInfo).parameters[key];
        }
    }
    return null;
}

/**
 * searches for a clickTARGET, case-insensitive search
 */
function clickTargetGet(clickNumber:String):String {
    for (var key:String in
LoaderInfo(this.root.loaderInfo).parameters) {
        if (key.toLowerCase()=='clicktarget' + clickNumber) {
            return
LoaderInfo(this.root.loaderInfo).parameters[key];
        }
    }
    return null;
}

/**
 * Opens the clickTAG and prevents to be blocked as a popup
 */
function clickTagOpen(clickNumber:String, url:String=null,
target:String=null):void {
    if (! target) {
        target=clickTargetGet(clickNumber);
        if (! target) {
            target='_blank';// default: to new window
        }
    }
    if (! url) {
        url=clickTagGet(clickNumber);
    }
    if (url) {
        var req:URLRequest=new URLRequest(url);
        if (! ExternalInterface.available) {
            navigateToURL(req, target);
        } else {

```

```

        var
strUserAgent:String=String(ExternalInterface.call("function()
{ return navigator.userAgent; }")).toLowerCase();
        if (strUserAgent.indexOf("firefox") != -1 ||
(strUserAgent.indexOf("msie") != -1 &&
uint(strUserAgent.substr(strUserAgent.indexOf("msie") + 5, 3))
>= 6)) {
                ExternalInterface.call("window.open",
req.url, target);
        } else {
                navigateToURL(req, target);
        }
    }
} else {
    trace('clickTAG' + clickNumber + ' was not found.');
```

e. Expandable/Floating Ad Implementierung in ActionScript 2

```

/**
 * expand: use the getURL function after the mouse-over event
 just before the expand animation of the banner starts
 */
_root.actionarea.onMouseOver = function() {
    getURL('javascript:adm_expand_layer();', '_self');
    do_expand_animation();
};

/* ----- */
/* ----- */

/**
 * collapse: use the getURL function after the collapse
 animation of the banner ends
 */
getURL('javascript:adm_collapse_layer();', '_self');

/* ----- */
/* ----- */

/**
 * close: use the getURL function on the onRelease-event of
 the close-button or after the close animation
 */
_root.closebutton.onRelease = function() {
    getURL('javascript:adm_close_layer();', '_self');
```

```
};
```

f. Expandable/Floating Ad Implementierung in ActionScript 3

```
/**
 * imports are necessary for the movieclip- and mouse-events
 */
import flash.display.MovieClip;
import flash.events.MouseEvent;

/**
 * expand: define an action area on which the mouseover-event
 will be attached. after the expand-function play the expand-
 animation of the banner.
 */
actionarea.addEventListener(MouseEvent.ROLL_OVER,
onRollOverHandler);

function onRollOverHandler(myEvent:MouseEvent) {
    var jscommand:String='adm_expand_layer(';
    var url:URLRequest=new URLRequest('javascript:'+jscommand);
    navigateToURL(url, '_self');

    gotoAndPlay('expand');
}

/* -----
----- */

/**
 * collapse: define an action area on which the mouseout-event
 will be attached. play the collapse-animation first.
 */
actionarea.addEventListener(MouseEvent.ROLL_OUT,
onRollOutHandler);
function onRollOutHandler(myEvent:MouseEvent) {
    gotoAndPlay('collapse');
}

/**
 * collapse: at the end of the collapse-animation use the
 collapse-function.
 */
var jscommand:String='adm_collapse_layer(';
var url:URLRequest=new URLRequest('javascript:'+jscommand);
navigateToURL(url, '_self');

/* -----
----- */

/**
 * close: define a close-button with the mouseclick-event and
 execute the close-function after the click
 */
closebutton.addEventListener(MouseEvent.CLICK,
onClickCloseHandler);
```

```
function onClickCloseHandler(myEvent:MouseEvent) {  
    var jscommand:String='adm_close_layer();'  
    var url:URLRequest=new URLRequest('javascript:'+jscommand);  
    navigateToURL(url, '_self');  
}
```