



Revision Date: Jun 27<sup>th</sup>, 2014

# **Adhese API documentation**

# table of contents

table of contents .....	2
introduction.....	3
getting access to the API .....	4
response formats .....	5
part I: planning and inventory .....	6
introduction.....	6
glossary.....	6
slots data.....	7
part II: order creation and management.....	9
introduction.....	9
glossary.....	9
list orders .....	10
insert new order / update existing order .....	11
get order .....	13
list bookings for orders .....	15
Insert new booking / update existing booking.....	16
part III: reporting .....	18
order report .....	18
booking report .....	20
creative report .....	22
periodical report .....	24
slot report per order .....	26
slot report per booking .....	27
part IV: uploading creatives into the hotfolder.....	28
What is a hotfolder.....	28
Uploading ads in the hotfolder.....	28
Processing the uploaded creatives.....	28
Reports .....	29

2/29

# introduction

The adhese API provides access to all orders and inventory related data in the adhese database. This data is available through a series of calls described in this document.

The complete workflow can be accessed through the API. This workflow is typically split up in three steps:

- before an order: planning and inventory queries
- creating the order: editing of campaigns, bookings and creatives
- during and after the order: delivery reports, optimization

This document will describe each element of the workflow in terms of available data and how to access it through the API.

3/29

# getting access to the API

To use the API, you will need a key and a base url, both tied to a specific adhese account.

With these two elements you will be able to form the specific url's described here for each part of the service.

Authentication can be done on a key basis by adding the key parameter to the request and providing a valid key based on a string containing the date of today as yyyy-MM-dd and followed by your unique key as provided by your sales rep.

The resulting string should then be encrypted with sha-1 and added as the "key" parameter to the url. The result looks like this:

<https://subdomain.adhese.com/ip/getOrder.do?json=1&key=mySha1Key&...>

4/29

# response formats

Response from the service are available in two formats: json and introspective xml.

To get the response as JSON data, simply add the parameter `json=1` to the url you are calling (the value of the parameter doesn't matter, as soon as it is present, JSON will be responded). To get the response as xml data, simply add the parameter `xml=1` to the url you are calling. If both parameters are present, JSON will be the response.

5/29

# part I: planning and inventory

## introduction

Part I of this document contains the different methods you can use to get reporting concerning the daily inventory per location, format or slot as well as reports on availability per position.

The queries are driven by values that are available in the adhese UI as well, and are determined by each account owner.

## glossary

A simple glossary on the terms used in this part.

**slots:** The actual integration of an advert on a page or application. In the adhese UI you can find these under the main tab “Positions”. Also known as “zones”, “ad spots”, ... in other software.

**locations:** The content section a slot belongs to. A location has a url (string identifying it through the tag integration) and a code (a foreign key to refer to this location). In the adhese UI you can find these in Inventory > Locations.

**templates:** A definition of an advert format. Each template has a unique code used in the tag integration to identify the requested dimensions and format. In the adhese UI you find these in Inventory > Formats.

## slots data

For each defined adhere slot you can query daily inventory reports. The result of these reports are the actual numbers for each day in the requested period. This data is updated nightly (in the time zone of your adhere application).

### URL

/ip/reports/slot/data.do?json=1

### request parameters

The request can contain one or more parameters. The parameters `templateCode` and `productUrl` can be left out, or occur several times with different values.

start: the start date for the report you are accessing (yyyymmdd)  
end: the end date  
templateCode: the template TAG code for the position(s) you are accessing  
productUrl: the product url for the position(s) you are accessing

### response

The response contains an object of type `ReportData` that contains totals and an array of `ReportDayData` objects that contain daily numbers.

The `ReportData` object contains these properties:

name: the display name of the element reported  
key: the unique key identifying this slot (cf. the code used in the tag integration - string)  
contacts: number of unique daily contacts  
impressions: number of impressions  
start: date of first day of report  
end: date of last day of report  
data: an array of `ReportDayData` objects, one for each day of the report  
The `ReportDayData` object contains these properties:  
date: date of the report day  
contacts: unique contacts reached (integer)  
impressions: number of impressions (integer)  
dateS: formatted date of the report day  
contactsS: number of contacts as formatted string (string)  
impressionsS: number of impressions as formatted string (string)

## HTTP responses

400: when no start or no end date was sent in the request or start or end date wasn't in a valid format, a 400 status will be send back to the client with some information.

404: when no data was found, a 404 status will be send back to the client instead of an empty array.



# part II: order creation and management

## introduction

Part II describes the various ways of creating and editing orders, bookings and creatives.

9/29

There are two ways of accessing order data. One, by using your own keys, stored as foreign keys in the adhere dbase (fi. a salesforce id stored as externalKey of the adhere order).

Or two, by getting a list of orders for a specific period, and starting from the list, getting more info through the adhere order id.

## glossary

order: an order is an entity that has an id and a name, and can optionally have one or more advertiser parties (i.e.. clients). An alternative name for an order could be a “campaign”.

booking: a booking is one element of an order. It has a start and end, a specific slot and optional profile targets. A booking is also known as ad-line, flight, ...

## list orders

This method allows you to query all orders. It returns an array of objects that contain some extra information as well as the unique id of the order. Further queries can be based upon this list.

You can query through a set of pre-made queries, or by defining a start and end period.

### URL

/ip/listOrders.do?json=1

### request parameters

searchString: this parameter can contain one of strings explained below  
 runToday: all live orders  
 endYesterday: orders ended in the last 7 days  
 startSoon: orders starting in the coming 7 days  
 endSoon: orders ending on the coming 7 days  
 select: when using this value, you must include the start and end parameter. The result will be a list of orders running in the selected period  
 start: the start date of your selection (yyyymmdd)  
 end: the end date of your selection (yyyymmdd)

### response

An array of OrderSearchSqlBean objects is returned. These contain the following properties

user: the name of the user who created this order  
 publication: the display label of the position(s) of this order  
 reservation: the status (0=reservation, 1=confirmed order)  
 id: the unique adhere id of this order  
 externalKey: the foreign key of this order  
 name: the display name of this order  
 priority: the priority of this order as an integer (0=highest, 3=lowest, integer)  
 s: the start date and hour of the first booking of this order (yyyy-MM-dd hh:mm:ss)  
 e: the end date and hour of the last booking of this order (yyyy-MM-dd hh:mm:ss)

### HTTP responses

400: If the searchString is select and no start or end date is defined or start or enddate was in an invalid format, a 400 status will be send back.  
 404: when no data was found, a 404 status will be send back to the client instead of an empty array.

## insert new order / update existing order [deprecated]

An order is a container for one or more bookings, related to the same advertiser.

### URL

/ip/updateOrder.do?json=1

### request parameters

11/29

id: integer, the unique identifier of an order (0 for new orders)  
priority: integer determining the priority of an order (0=highest priority, 2=lowest priority)  
publisherId: integer, identifying the publisher to which this order belongs (should be predetermined fixed value)  
userId: integer, identifying the user who created this order (should be predetermined fixed value)  
advertiserCompanyIdExt: string identifying the advertiser party (optional)  
advertiserCompanyInvoiceIdExt: string identifying the invoicing party (optional)  
advertiserCompanyIntermediaryIdExt: string identifying the creative agency (optional)  
reservation: integer that determines if an order is just a reservation (0) or confirmed order (1)  
campaignType: integer setting the order type (for profile based orders use the fixed value 5)  
name: string containing name of order

### response

The response contains the full JSON object of the order just created. These are the most important properties:

adspaceCount: number of bookings for this order  
adspace: array of bookings  
advertiserCompanyBranch: meta data concerning the advertiser  
advertiserCompanyContact: name of contact person  
advertiserCompanyContactId: dbase id of contact person  
advertiserCompanyId: id of advertiser party  
advertiserCompanyIdExt: external key of advertiser party  
advertiserCompanyIntermediaryId: id of intermediary (sales house, buyer, ...)  
advertiserCompanyIntermediaryIdExt: external key of intermediary (sales house, buyer, ...)  
advertiserCompanyInvoiceContact: id of invoice party contact person  
advertiserCompanyInvoiceContactId: external key of invoice party contact person  
advertiserCompanyInvoiceId: id of invoice party  
advertiserCompanyInvoiceIdExt: external key of invoice party  
advertiserCompanyInvoiceName: company name of invoice party

advertiserCompanyMediaId: id of agency/creative party  
advertiserCompanyMediaIdExt: external key of agency/creative party  
advertiserCompanyMediaName: company name of agency/creative party  
advertiserCompanyName: company name of advertiser party  
agencyCommission: optional overall commission on order, in percentage  
agencyCommissionS: string representation of this commission  
budget: total budget for this order (optional)  
budgetString: string rep. of budget  
campaignType: integer  
closed: 1 = order is ready for archiving  
comment1: visible comment field  
comment2: internal comment field  
creation: timestamp of creation  
edited: timestamp of last modification  
priority: priority for this order, is a fixed set  
reservation: integer, switch for creating a reservation or confirmed order [0 = option, 1 = order]

## get order

### URL

/ip/getOrder.do?json=1

### request parameters

id: integer, the unique identifier of an order (0 for new orders)  
externalKey: a string identifying an order

note: both parameters can be used together but will only succeed if they point to the same order

### response

The response contains the full JSON object of the order just created. These are the most important properties:

adspaceCount: number of bookings for this order  
adspaces: array of bookings  
advertiserCompanyBranch: meta data concerning the advertiser  
advertiserCompanyContact: name of contact person  
advertiserCompanyContactId: dbase id of contact person  
advertiserCompanyId: id of advertiser party  
advertiserCompanyIdExt: external key of advertiser party  
advertiserCompanyIntermediaryId: id of intermediary (sales house, buyer, ...)  
advertiserCompanyIntermediaryIdExt: external key of intermediary (sales house, buyer, ...)  
advertiserCompanyInvoiceContact: id of invoice party contact person  
advertiserCompanyInvoiceContactId: external key of invoice party contact person  
advertiserCompanyInvoiceId: id of invoice party  
advertiserCompanyInvoiceIdExt: external key of invoice party  
advertiserCompanyInvoiceName: company name of invoice party  
advertiserCompanyMediaId: id of agency/creative party  
advertiserCompanyMediaIdExt: external key of agency/creative party  
advertiserCompanyMediaName: company name of agency/creative party  
advertiserCompanyName: company name of advertiser party  
agencyCommission: optional overall commission on order, in percentage  
agencyCommissionS: string representation of this commission  
budget: total budget for this order (optional)  
budgetString: string rep. of budget  
campaignType: integer  
closed: 1 = order is ready for archiving  
comment1: visible comment field  
comment2: internal comment field

creation: timestamp of creation

edited: timestamp of last modification

priority: priority for this order, is a fixed set

reservation: integer, switch for creating a reservation or confirmed order [0 = option, 1 = order]

## HTTP responses

14/29

400: If no orderId and no external key was sent in the request, a 400 status will be send back.

404: when no data was found, a 404 status will be send back to the client instead of an empty array.

## list bookings for orders

For each order you can obtain a list of all its bookings.

### URL

/ip/listAdspaces.do?json=1

### request parameters

id: integer, the unique identifier of the order

### response

The response is an array of Adspace objects containing the following properties:

id: the unique identifier of the booking (integer)  
 slotId: the id of the ad position (integer)  
 templateId: the id of the format (integer)  
 active: the status of the booking: 1 = active, 0 = paused, -1 = stopped  
 orderId: the id of the order to which this booking belongs  
 start: the start date of the booking  
 startTime: the start time of the booking  
 end: the end date of the booking  
 endTime: the end time of the booking  
 pricingId: the id of the pricing type  
 impressions: the volume to be reached  
 rate: the rate for the unit of the given pricing type (CPM: rate per thousand impressions, CPC: rate per click, ...)  
 externalKey: the key (string) attached to this booking

### HTTP responses

400: if no id was sent in the request, a 400 status will be send back.  
 404: when no data was found, a 404 status will be send back to the client instead of an empty array.

## get booking

### URL

ip/getAdspace.do?json=1

### request parameters

id: integer, the unique identifier of a booking  
externalKey: a string identifying a booking

*notes: both parameters should be used separately  
if the booking is not found an empty response will be send*

### response

The response contains the JSON object of the booking asked for. These are the most important properties:

id: the unique identifier of the booking (integer)  
slotId: the id of the ad position (integer)  
templateId: the id of the format (integer)  
active: the status of the booking: 1 = active, 0 = paused, -1 = stopped  
orderId: the id of the order to which this booking belongs  
start: the start date of the booking  
startTime: the start time of the booking  
end: the end date of the booking  
endTime: the end time of the booking  
pricingId: the id of the pricing type  
impressions: the volume to be reached  
rate: the rate for the unit of the given pricing type (CPM: rate per thousand impressions, CPC: rate per click, ...)  
externalKey: the key (string) attached to this booking



## Insert new booking / update existing booking [deprecated]

A booking is a part of a order. Each booking contains one slotId (i.e.. position in adhese interface) and can contain several profile selections. A booking has a start and end date/hour, as well as a pricing type, rate and to reach volume (depending on the pricing type).

### URL

/ip/updateAdspace.do?json=1

### request parameters

id: integer, the unique identifier of a booking (0 for new)  
slotId: optional, integer identifying the ad position for this booking  
templateId: integer identifying the ad format for this booking  
active: integer, switch for making a booking active / inactive (1= active, 0=inactive)  
orderId: integer, order to which this booking belongs  
start: start date and hour of a order (YYYYMMdd hh:mm)  
end: end date and hour of a order (YYYYMMDD hh:mm)  
startTime: daily start hour of a booking (hh:mm)  
endTime: daily end hour of a booking (hh:mm)  
pricingId: integer identifying the type of spreading applied to this booking  
impressions: volume to be reached  
language: 2 char iso code (lower case; if more then one is sent, all combinations are created for this booking; ex. language=nl&language=fr&language=de)  
country: two char iso code (capitals; if more then one is sent, all combinations are created for this booking)  
region: 3 char fips code (if more then one is sent, all combinations are created for this booking)  
sex: MALE | FEMALE  
minAge: integer  
maxAge: integer (is counted as including)  
creatives: (optional) plain text string containing the name and variables of an ADVAR template; more than one creative can be included in the request. each creative will be saved and trafficked to this booking; each creative will receive an equal share in the ad server

17/29

# part III: reporting

Through the reporting services you can access reports based on specific time periods, or by order id's.

## order report

Reports for each order and booking can be requested. The numbers returned are updated daily.

### URL

/ip/reports/campaign/data.do?json=1

### request parameters

orderId: integer, the id of the order to be reported

### response

The response contains an object of type ReportData that contains totals and an array of ReportDayData objects that contain daily numbers.

The ReportData object contains these properties:

name: the display name of the element reported  
key: the unique key identifying this element (string)  
contacts: number of unique daily contacts  
impressions: number of impressions  
clicks: number of clicks reached  
start: date of first day of report  
end: date of last day of report  
data: an array of ReportDayData objects, one for each day of the report  
The ReportDayData object contains these properties:  
date: date of the report day  
contacts: unique contacts reached (integer)  
clicks: number of clicks (integer)  
impressions: number of impressions (integer)  
clickRate: relation between clicks and impressions (0-1) (float)  
clickRateC: relation between clicks and contacts (0-1) (float)  
dateS: formatted date of the report day  
contactsS: number of contacts as formatted string (string)  
clicksS: number of clicks as formatted string (string)  
impressionsS: number of impressions as formatted string (string)  
clickRateS: relation between clicks and impressions as a percentage (string)  
clickRateCS: relation between clicks and contacts as a percentage (string)

impressionsShareS: share of this day in the overall sum of impressions as percentage (string)  
budgetSpent: amount of the total budget spent on this day

## HTTP responses

400: if no orderId was sent in the request, a 400 status will be send back.  
404: when no data was found, a 404 status will be send back to the client instead of an empty object.

19/29

## booking report

Reports for each booking can be requested. The numbers returned are updated daily.

### URL

/ip/reports/campaign/data.do?json=1

### request parameters

adspaceId: integer, the id of the booking to be reported

### response

The response contains an object of type ReportData that contains totals and an array of ReportDayData objects that contain daily numbers.

The ReportData object contains these properties:

name: the display name of the element reported  
key: the unique key identifying this element (string)  
contacts: number of unique daily contacts  
impressions: number of impressions  
clicks: number of clicks reached  
start: date of first day of report  
end: date of last day of report  
data: an array of ReportDayData objects, one for each day of the report  
The ReportDayData object contains these properties:  
date: date of the report day  
contacts: unique contacts reached (integer)  
clicks: number of clicks (integer)  
impressions: number of impressions (integer)  
clickRate: relation between clicks and impressions (0-1) (float)  
clickRateC: relation between clicks and contacts (0-1) (float)  
dateS: formatted date of the report day  
contactsS: number of contacts as formatted string (string)  
clicksS: number of clicks as formatted string (string)  
impressionsS: number of impressions as formatted string (string)  
clickRateS: relation between clicks and impressions as a percentage (string)  
clickRateCS: relation between clicks and contacts as a percentage (string)  
impressionsShareS: share of this day in the overall sum of impressions as percentage (string)  
budgetSpent: amount of the total budget spent on this day

## HTTP responses

400: if no adspaceId was sent in the request, a 400 status will be send back.

404: when no data was found, a 404 status will be send back to the client instead of an empty object.

## creative report

This method reports the daily numbers for one or more creatives. The creative id's can be obtained from the report campaign menu.

### URL

/ip/reports/campaign/creative/data.do?json=1

### request parameters

liblds: one or more adhere id's of creatives  
from: optional start date of report (yyyymmdd)  
till: optional end date of report (yyyymmdd)

### response

The response contains an object of type ReportData that contains totals and an array of ReportDayData objects that contain daily numbers.

The ReportData object contains these properties:

name: the display name of the element reported  
key: the unique key identifying this element (string)  
contacts: number of unique daily contacts  
impressions: number of impressions  
clicks: number of clicks reached  
start: date of first day of report  
end: date of last day of report  
data: an array of ReportDayData objects, one for each day of the report  
The ReportDayData object contains these properties:  
date: date of the report day  
contacts: unique contacts reached (integer)  
clicks: number of clicks (integer)  
impressions: number of impressions (integer)  
clickRate: relation between clicks and impressions (0-1) (float)  
clickRateC: relation between clicks and contacts (0-1) (float)  
dateS: formatted date of the report day  
contactsS: number of contacts as formatted string (string)  
clicksS: number of clicks as formatted string (string)  
impressionsS: number of impressions as formatted string (string)  
clickRateS: relation between clicks and impressions as a percentage (string)  
clickRateCS: relation between clicks and contacts as a percentage (string)  
impressionsShareS: share of this day in the overall sum of impressions as percentage (string)

budgetSpent: amount of the total budget spent on this day

## HTTP responses

400: if no libIds was sent in the request or if the optional start or end date was in an invalid format, a 400 status will be send back.

404: when no data was found, a 404 status will be send back to the client instead of an empty object.

23/29

## periodical report

Reports for a specific start and end date, independent of a specific order. The numbers returned are daily.

### URL

/ip/reports/campaign/daily.do?json=1

### request parameters

start: the start day of the report (YYYYMMDD)  
 end: the end day of the report (YYYYMMDD)  
 pricingId: optional, id of specific booking pricing type to filter orders  
 slotId: optional, id of specific ad position, to filter orders

### response

The response of the periodical report is an array of BookingDayData objects (grouping the results per day and per booking). These extend the ReportDayData object and add a few fields of their own. Each object contains numbers for one day and one booking.

inherited from ReportDayData properties:

date: date of the report day  
 contacts: unique contacts reached (integer)  
 clicks: number of clicks (integer)  
 impressions: number of impressions (integer)  
 clickRate: relation between clicks and impressions (0-1) (float)  
 clickRateC: relation between clicks and contacts (0-1) (float)  
 dateS: formatted date of the report day  
 contactsS: number of contacts as formatted string (string)  
 clicksS: number of clicks as formatted string (string)  
 impressionsS: number of impressions as formatted string (string)  
 clickRateS: relation between clicks and impressions as a percentage (string)  
 clickRateCS: relation between clicks and contacts as a percentage (string)  
 impressionsShareS: share of this day in the overall sum of impressions as percentage (string)  
 budgetSpent: amount of the total budget spent on this day

BookingDayData properties:

externalKey: the foreign key for this booking  
 id: the adhese id for the booking  
 orderExternalKey: the foreign key of the order  
 orderId: the adhese id of the order



pricingId: the pricing type if the booking

## HTTP responses

400: when no start or end date was sent in the request or when the start or end date was in an invalid format, a 400 status will be send back to the client with some information.

404: when no data was found, a 404 status will be send back to the client instead of an empty array.

25/29

## slot report per order

Reports the distribution of impressions per slot and per reative for a specific order.

### URL

/ip/reports/campaign/slot/data.do?json=1

### request parameters

orderId: the id of the order to report

### response

The response of this request is an array of AdSlotReportData objects containing an object for each combination of a creative and a slot. Each AdSlotReportData object contains an array of ReportDayData objects (as the property "data") for each day to report.

AdSlotReportData properties:

publication: the name of the publication  
 publicationId: the dbase id of the publication  
 product: the name of the produyct to which the slot belongs  
 productId: the dbase id of the product  
 slot: the name of the slot  
 slotId: the id of the slot  
 templateId: the dbase id of the format  
 template: the name of the format

ReportDayData properties:

date: date of the report day  
 contacts: unique contacts reached (integer)  
 impressions: number of impressions (integer)  
 clicks: number of clicks (integer)

### HTTP responses

400: if no orderId was sent in the request, a 400 status will be send back.  
 404: when no data was found, a 404 status will be send back to the client instead of an empty array.

## slot report per booking

Reports the distribution of impressions per slot and per reative for a specific booking.

### URL

/ip/reports/campaign/booking/slot/data.do?json=1

### request parameters

adspaceId: the id of the order to report  
 start: the start date of the report you are querying (optional), (yyyy-mm-dd)  
 end: the end date of the report you are querying (optional), (yyyy-mm-dd)

### response

The response of this request is an array of AdSlotReportData objects containing an object for each combination of a creative and a slot. Each AdSlotReportData object contains an array of ReportDayData objects (as the property "data") for each day to report.

AdSlotReportData properties:

publisher: the name of the publisher  
 publisherId: the id of the publisher  
 publication: the name of the publication  
 publicationId: the dbase id of the publication  
 product: the name of the product to which the slot belongs  
 productId: the dbase id of the product  
 slot: the name of the slot  
 slotId: the id of the slot  
 templateId: the dbase id of the format  
 template: the name of the format  
 pricingCalculationCode: a value indicating the budget calculation basis (cpm, cpc, cpd, cpp, cpl, cpa)  
 clicks, impressions, contacts as numbers and formatted as strings  
 revenue as number and as formatted string

ReportDayData properties:

date: date of the report day  
 contacts: unique contacts reached (integer)  
 impressions: number of impressions (integer)  
 clicks: number of clicks (integer)  
 revenue as number and as formatted string

## HTTP responses

404: when no data was found, a 404 status will be send back to the client instead of an empty array.

# part IV: uploading creatives into the hotfolder

## What is a hotfolder

A hotfolder is a folder on the Adhese server where you can upload your creatives (=ads).

The uploaded creatives follow a specific name structure. Every uploaded file is verified and, if correct, attached to the corresponding booking. When a file is processed a report is produced.

The hotfolder can be started on demand or can be scheduled to automate the process.

## Uploading ads in the hotfolder

To connect with the ftp server you need an adhese ftp account. If you did not receive an account yet, please contact us.

When you log in with an ftp program you arrive at the correct folder on the adhese server. That folder is empty or contains the latest files waiting to be processed.

You can drag and drop all your creatives into this folder. After they have been processed your hotfolder will be empty again. A rapport is produced for every creative processed.

## Processing the uploaded creatives

After you uploaded your files you can process them immediately. Log in to your adhese account and click the tab 'Hotfolder'. It brings you to an overview of the latest reports.

To process the hotfolder, click on the 'process' link at the top of the page. After this operation a new report appears at the top of the list of reports. Click on it and a new screen comes into view. In addition you receive an email with a link to the report.

## Give a name to the uploaded creatives

Every file in the hotfolder must have a specific name structure so that it can be identified and processed.

### Basics: creative ID

The key to connect a file to a booking is the creative ID that can be added when a campaign is booked. The separation sign used in name giving is an \_ (underscore).

There are three ways to add a number to the file's name:

as the full name of a file: 123456A8.swf

as the last part of the name of the file, preceded by an \_ (underscore) and followed by a dot and an extension: leaderboard-coca-cola\_123456A8.swf

as part of the name, preceded and followed by an \_ (underscore): leaderboard-coca-cola\_123456A8\_version-mon-2-jan-2008.swf

### Link url's of uploaded creatives

With every uploaded creative you can upload a second file. The extension is "-LINK". It follows the same name giving rules as above:

as the full name of a file: 123456A8-LINK.txt

as the last part of the name of the file, preceded by an \_ (underscore) and followed by a dot and an extension: leaderboard-coca-cola\_123456A8-LINK.txt

as part of the name, preceded and followed by an \_ (underscore): leaderboard-coca-cola\_123456A8-LINK\_version-mon-2-jan-2008.txt

### Extra files to be processed with the main file

For certain creatives you will need to upload several files (fi. Expandable ads, alternative versions of the creative, ...). To upload these, simply add -X2, -X3 or -X4 to the key.

A valid filename for an extra file will look like this: leaderboard-coca-cola\_123456A8-X2.swf

## Reports

### Processing report

A general report is produced of every process made. This report is a listing of processed files and non-processed files. It shows whether a file was processed correct or incorrect.

You can click on the name of the file to get a more detailed report.

### Detail report and preview page

Every processed file has its own detail report. The report contains the possibly detected order, a description of the file properties of the creative, and error reports (if there are any).

The report also shows a preview of the creative how it will appear online. You can test whether the link is working by clicking on the creative.

The detailed report can be saved from your browser or can be send by email to use it as a proof. each report is available online, and can be sent by email after processing.