Adcoin

INTEGRATION MANUAL

November 2017

Table of contents

Table of contents 2

1. What you need to know before starting 3
   1.1 What is Adcoin 3
   1.2 Why integrate 3
   1.3 Adcoin ad delivery and verification process overview 3
   1.4 Key optional features 4
   1.5 Which method works for me? 6
   1.6 Before you start 7
   1.7 Preparations for the integration 7

2. Code Snippet method 8
   2.1 Connections and languages 8
   2.2 Snippet format and customization 8
   2.3 Detailed installation instructions 10
   2.4 Advanced Ad Space Configuration 11

3. API Method 13
   3.1 Connections and languages 13
   3.2 API call URL and JSON Object formats 13
   3.3 Detailed installation instructions 22

4. Custom method 23

5. Testing and troubleshooting 23
   5.1 Testing your implementation 23
   5.2 Common errors and troubleshooting 23

Appendix A – list of supported languages 25

1. What you need to know before starting

1.1 What is Adcoin

Adcoin platform allows publishers to monetise digital content by using completion ads. Publishers can place their content behind an access gateway (such as paywall), and offer Adcoin ads as a method to unlock the content. When Adcoin is selected as the unlock method, a completion ad is displayed to a user, who needs to complete the task given to unlock access. After a successful completion, the user is shown a follow-on ad and a button to access the unlocked content.

1.2 Why integrate

All publishers looking to use Adcoin for ad delivery need to take steps to integrate their content management system with Adcoin platform; delivering ads and managing user responses requires seamless communication between the two systems. This is achieved by integrating the two platforms for efficient data exchange.

There are 3 integration options available that range from a simple copy and paste code method that require little to no development work, to using a flexible REST API, to a fully customized integration.

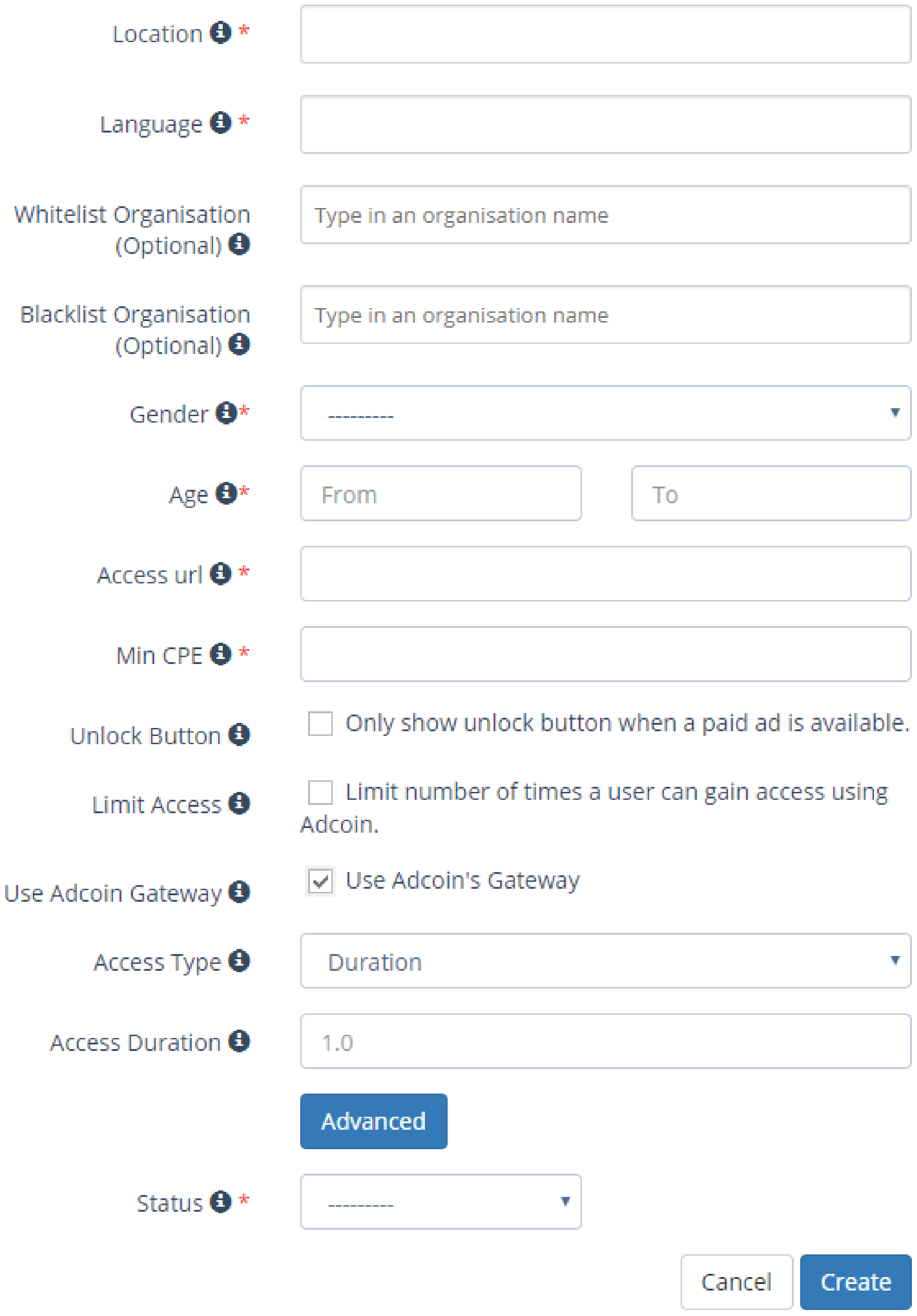
1.3 Adcoin ad delivery and verification process overview

Ad delivery process consists of several steps of data exchange. When a user arrives to the publisher’s website, the following sequence of exchange between the publisher and Adcoin ad delivery platform occurs:

1. Publishing platform contacts Adcoin and passes known user attributes
2. Adcoin platform performs ad selection and returns an ad unit with a task
3. User input is sent back to Adcoin for verification and a result is returned
4. On success, the follow-on ad and reward URL is sent from Adcoin

1.4 Key optional features

Adcoin platform has several key features that the publisher may either choose to use, choose to ignore and implement them using their own systems, or choose not to implement the features at all.



Location *	
Language *	
Whitelist Organisation (Optional)	Type in an organisation name
Blacklist Organisation (Optional)	Type in an organisation name
Gender *	---------
Age *	From / To
Access url *	
Min CPE *	
Unlock Button	☐ Only show unlock button when a paid ad is available.
Limit Access	☐ Limit number of times a user can gain access using Adcoin.
Use Adcoin Gateway	☑ Use Adcoin's Gateway
Access Type	Duration
Access Duration	1.0
	Advanced
Status *	---------

1.4.1 Unlock button

By using unlock the button feature, you can check, if ads are available before you offer the Adcoin option to the user. This way you can show Adcoin option to users when there is an ad available for them, and hide the option when there are no ads available.

1.4.2 Limit access

This feature may be used if the publisher wants to restrict the number of times or the duration a single user can gain access to content using Adcoin. Limit access feature is Ad space specific and can be used independently of access gateway feature. To use this feature, publisher must use a unique identifier for each user. This identifier can be some form of advertising ID, an ID used on publisher's platform or specifically set for Adcoin. Publisher can specify that any users without identifiers are rejected. Otherwise, these users will always get content without any access limits.

EXAMPLE: Publisher wants to restrict access to content so that each user can only access content 5 times within a period of a week. Publisher sets "number of times accessible" to 5, and "timeout duration" to 7 days. Now the user can access content 5 times, after which the user is unable to access content again using Adcoin until 7 days has passed. The duration is access specific, which means that if user accesses content two times on day 1 and three times on day 2, the user cannot access any content between days 3 and 7, but then the user can access content again two times on day 8, and three more times on day 9.

NOTE: limiting user access may reduce publisher income from an Ad space

1.4.3 Access gateway

When this setting is checked, all content after successful completion is delivered through Adcoin's gateway; Adcoin platform acts as a content proxy. The gateway can be configured to restrict access to the content by a limited number of visits or by limiting the duration when access is granted. When this feature is used, upon completion of an engagement, a reusable URL will be generated by Adcoin where the content will be accessible for the user (access URL).

1.5 Which method works for me?

A) **Code snippet** (section 2 in this manual) integration method is a very straight forward option, and can be implemented with little to no development. After you log into Adcoin platform and create an Ad space, you are provided with a code snippet. All you have to do is to copy and paste it onto your site. Everything else is handled by Adcoin. With the easy implementation, there are few considerations:

- No development necessary
- Not intended for native mobile app (e.g. iOS or Android)
- Adcoin will manage delivery of the ads, answer verification, and reward delivery
- Look and feel is controlled via the Adcoin UI and cannot be set dynamically
- You can control certain targeting parameters dynamically (requires JavaScript development)

B) Using the **REST API** (section 3 in this manual) option will require some development work, but does give you much more flexibility. If you want to integrate into native mobile platforms or want greater control over data exchange, this is the way to go. To start using the API method, make sure you read and understand the process and the available API calls detailed in the later sections of this manual.

- You can embed ads to be part of your apps and get freedom how to present them
- You can manage the reward delivery
- Select targeting parameters dynamically
- Suitable for native mobile apps

C) **Custom** (section 4 in this manual) integration is for those publishers who require even further flexibility, additional functionality and control over the integration options, or want to integrate several assets or platforms at one go. Work with our technology team to integrate our platform the way you need. With total freedom comes the possibility to maximise revenue by passing any parameters to achieve the highest levels of targeting and attract highest bids from advertisers.

- Ability to integrate substantial number of assets/applications/platforms at once
- Custom data and parameters can be included in the data exchange
- Use your own data points and user knowledge to optimise revenue dynamically
- Can perform any type of dynamic integration

1.6 Before you start

Code snippet method requires only basic understanding of web technologies, however, if custom parameters are to be used, understanding of JavaScript language is required. Understanding the technologies involved is required for REST API and Custom integration methods. Make sure you read this manual before starting, and if you have any concerns or run into issues, contact Adcoin support at support@adcoin.com.

There are only few basic prerequisites you must have in place to start the integration:
- Support HTML standard
- Have access to your HTML / Application code
- Support for HTTPS/SSL
- Internet connection and access to port 443
- DNS server access

1.7 Preparations for the integration

Make sure you go through these steps before you start the integration work:
- Login to Adcoin platform (adcoin.com/login)
- Fill in your business profile details
- Create a Platform
- Create an Ad space and make it inactive
- Turn on sandbox mode for the Ad space and make it active
- Perform and test integration

After the integration work is complete
- Turn Ad space inactive
- Turn sandbox mode off
- Turn Ad space active

2. Code Snippet method

2.1 Connections and languages

Observe the following connectivity limitations for security and technical reasons.

- All connections must utilise Secure HTTP (HTTPS/SSL)
- Connection port is standard HTTPS/SSL port: 443
- Connection to Adcoin requires DNS to resolve hostname to an IP address
- There are no limitations for the number of calls made
- Our snippet can be controlled with JavaScript

2.2 Snippet format and customization

Our snippet is ready as is and the snippet itself cannot be modified. It will work as is, using the targeting options set on the Adcoin portal for that specific Ad space.
Example code snippet:

```
<script type="text/javascript"src="https://adserve.adcoin.com/display/snippet/19/315">
</script>
```

If you wish to pass targeting parameters dynamically, you can do that by adding a JavaScript tag which includes the targeting parameters before the snippet. If you pass invalid variable data, these will simply be ignored and the Ad space default values will be used instead. You can pass one or more variables at a time.

Example snippet with targeting variables:

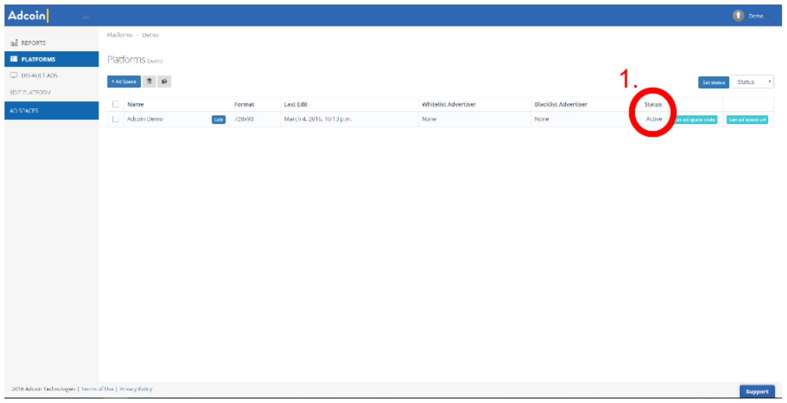
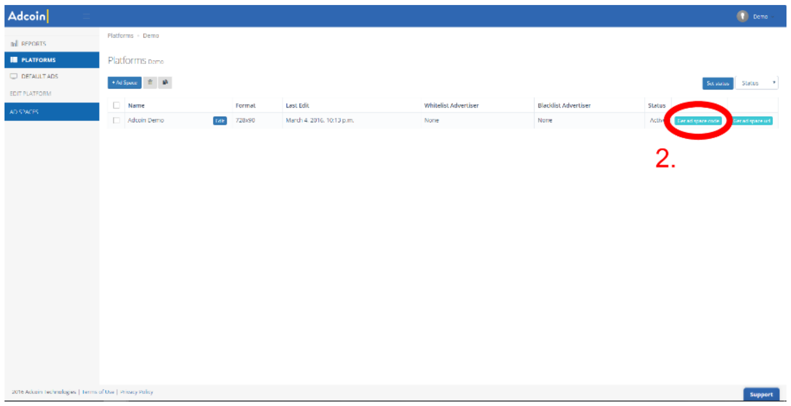
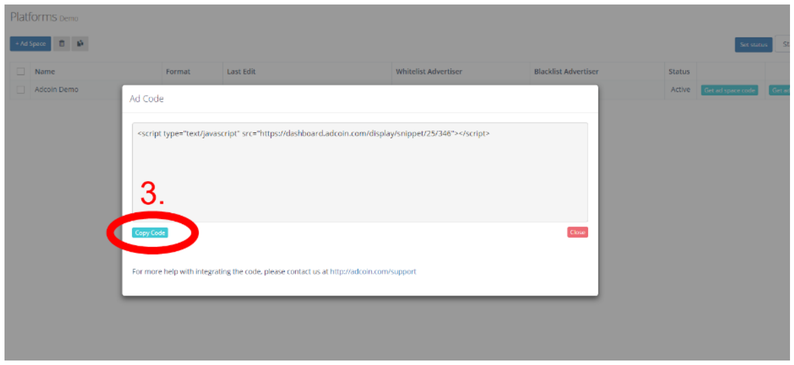
```
<script>
profiledata = {
   pd_age:26,
   pd_gender:'f'
}
</script>

<script
type="text/javascript"src="https://adserve.adcoin.com/display/snippet/19/315"></script>
```

The available parameters are:

Parameter	Description
pd_gender	The gender of the user, can be **m** for male or **f** for female. Must be in lowercase
pd_age	The age in years of the user (e.g. ‘**29**’)
pd_dob	This is the date of birth of the user, which can be a four digit year or a four digit year and a two digit month or a four digit year and a two digit month and a two digit date. For example: 1990 is valid, 199006 is valid, 19900623 is valid. 1990623 is not valid and 199011 will be interpreted as November 1990 not the first of January 1990.
pd_min_age	Must be used together and these specify the minimum and maximum age of the user. Values allowed are between 1 and 99 These parameters can be used in cases, where the exact age of the user is not known, but instead their age bracket is (e.g. from surveys ’25-30’)
pd_max_age	
pd_identifier	This is a unique identifier for the user, it can be an Apple advertiser ID or similar other. Must be used in conjunction with pd_identifier_type
pd_identifier_type	This specifies the type of identifier. Value can be: Google, Apple, Microsoft or another identifier. The only requirement is that the pd_identifier and pd_identifier_type combination is unique.
pd_location	This is the country name (in English) that is used to select the ads for this Ad space. Alternatively, this can be user’s location as latitude and longitude, in decimal format with a comma as the separator.
pd_categories	This is a comma separated list of categories that will be used to select the ads. This must match the categories found in Adcoin platform. Before using this parameter, it must be encoded using the JavaScript function encodeURI or similar.
pd_language	This is the language of the user. See Appendix A for a list of supported languages.
mocid	This is the content id of the reward the user will get, passed by Adcoin to your server, see 2.4.

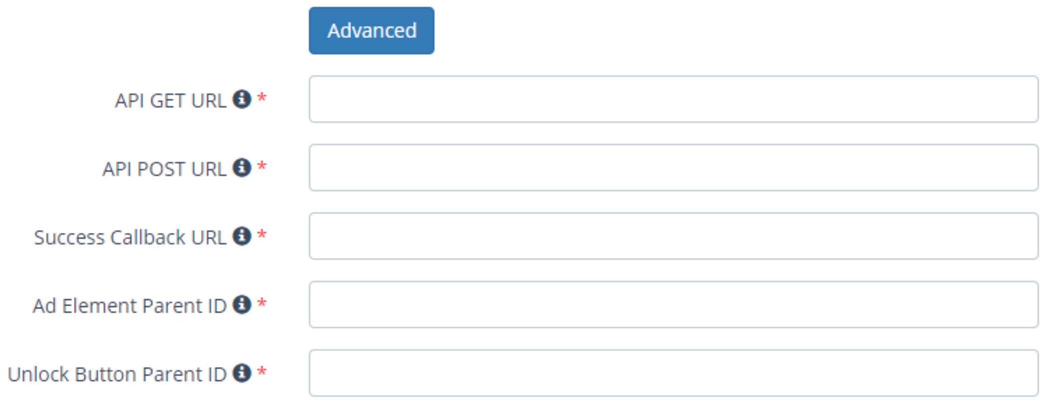
2.3 Detailed installation instructions

1.	 Adcoin REPORTS PLATFORMS DEFAULT ADS EDIT PLATFORM AD SPACES Platforms › Demo Platforms Demo + Ad Space Set status Status 1. Support Name: Adcoin Demo · Format: 728x90 · Last Edit: March 4, 2016, 10:13 p.m. · Whitelist Advertiser: None · Blacklist Advertiser: None · Status: Active · Get ad space code · Get ad space url	Login to Adcoin platform and ensure you have a platform and an Ad space ready and in active mode (1).
2.	 Adcoin REPORTS PLATFORMS DEFAULT ADS EDIT PLATFORM AD SPACES Platforms › Demo Platforms Demo + Ad Space Set status Status 2. Support Name: Adcoin Demo · Format: 728x90 · Last Edit: March 4, 2016, 10:13 p.m. · Whitelist Advertiser: None · Blacklist Advertiser: None · Status: Active · Get ad space code · Get ad space url	The creation of an Ad space will automatically create the snippet, click “Get Ad space code” to open a new window with your snippet code in it (2).
3.	 `<script type="text/javascript" src="https://dashboard.adcoin.com/display/snippet/25/346"></script>` For more help with integrating the code, please contact us at http://adcoin.com/support	Copy and paste the snippet on your site and you are ready. <u>OPTIONAL!</u> You may design your page to dynamically add JavaScript targeting variables for better targeting and higher revenue. Refer to section 2.2.

NOTE: A single platform can be used for a single website. If you have multiple websites, it is advisable to create multiple platforms.

2.4 Advanced Ad Space Configuration

These fields allow for greater control of how the snippet is used in both on your website and how it interacts with your backend.



API GET URL *	
API POST URL *	
Success Callback URL *	
Ad Element Parent ID *	
Unlock Button Parent ID *	

2.4.1 API GET URL

In this field specify where the snippet will make the call to get an Adcoin ad. This can be used if you are trying to provide access to Adcoin ads behind a firewall, such as for granting access to a WiFi network.

2.4.2 API POST URL

In this field specify where the snippet will make the call to answer an Adcoin ad. This can be used if you are trying to provide access to Adcoin ads behind a firewall, such as for granting access to a WiFi network.

2.4.3 Success Callback URL

This is called by ad server on a successful completion of an ad. The communication takes place between the Adcoin ad server and the publishers web server. The adserver will add the ‘mocid’, the ‘engagement id’ and the value of the ad as URL parameters to the URL defined in this field. The resulting call will look something like this:

```
www.publishering-company.com/reward-
endpoint?val=<engagement_value>&eid=<engagement id>&mocid=<your content id>
```

The return from this call must look like this, replace the reward-here.com with the desired reward URL:

```
{
        “url”:”http://reward-here.com”
}

The url returned from this call will be used as the reward URL for the user.
```

2.4.4 Ad Element Parent ID

This is the ID where the snippet will be written to on your web page. For example if on your page there is a DIV with the id of adcoin_ad and you specify adcoin_ad in that field, the snippet will write the Adcoin ad inside that DIV.

2.4.5 Unlock Button Parent ID

This is the ID where the unlock button will be written to on your web page.

3. API Method

3.1 Connections and languages

Observe the following connectivity limitations for security and technical reasons.

- All API calls must utilise Secure HTTP (HTTPS/SSL)
- Connection to Adcoin require DNS to resolve hostname to an IP address
- There are no limitations for the number of calls
- Connection port is standard HTTPS/SSL port: 443
- The API return object is JSON only

3.2 API call URL and JSON Object formats

3.2.1 Step 1: Ad request (Publisher to Adcoin)

The standard API URL is in format:

```
<adcoin adserver url>/display/ad_block/<ADSPACEID>/?api_key=<APIKEY>
```

To pass targeting variables dynamically, you can modify the URL as with the variables in a table below. You can pass one or more variables at a time.

Example API URL with targeting variables:

```
https://adserve.adcoin.com/display/ad_block/122/?api_key=gHCgudieF88neKFQx38gpJfeK
1BLEldis69nxzUV&pd_gender=f&pd_age=26
```

The available parameters are:

Parameter	**Description**
pd_gender	The gender of the user, can be m for male or f for female. Must be in lowercase.
pd_age	The age in years of the user.
pd_dob	This is the date of birth of the user, can be a four digit year or a four digit year and a two digit month or a four digit year and a two digit month and a two digit date. For example: 1990 is valid, 199006 is valid, 19900623 is valid. 1990623 is not valid and 199011 will be interpreted as November 1990 not the first of January 1990.
pd_min_age	Must be used together and these specify the minimum and maximum age of the user. Values allowed are between 1 and 99
pd_max_age	
pd_identifier	This is a unique identifier for the user, it can be an Apple advertiser ID or similar. Must be used in conjunction with pd_identifier_type.
pd_identifier_type	This specifies the type of identifier. Value can be: Google, Apple, Microsoft or any other identifier. The only requirement is that the pd_identifier and pd_identifier_type combination is unique.
pd_location	This is the country name (in English) that is used to select the ads for this Ad space. Alternatively, this can be users location as latitude and longitude, in decimal format with a comma as the separator.
pd_categories	This is a comma separated list of categories that will be used to select the ads. This must match the categories found in Adcoin platform. Before using this parameter, it must be encoded using the JavaScript function encodeURI or similar.
pd_language	The primary language of the user, see Appendix A for a list of supported languages.
ids	Exclude certain Ad IDs from showing to user. Use this to skip ads, for example in a situation when user has repeatedly given a wrong answer to an ad, and you want to request a new ad and want to make sure the previous ad is not shown to the user. Use the Ad ID number as the variable.

3.2.2 Step 2: Ad request return JSON object (Adcoin to Publisher)

In return you will either receive a JSON object with the following content and format, or an error message 204 or 404 if there is no inventory or default ads available for the request:

```
{
        “id”: “b75977f7-b23e-4f04-b338-a64d92b539ed”,
        “ad”: 7,
        “wi”: 300,
        “he”: 250,
        “ei”: “Engagement Instructions”,
        “ao”:”image”,
        “bi”: “banner_image_url.png”,
        “sa”:1,
        “an”:”42”,
        “uo”: true,
        “ve”: “video events url”,
        “bv”: “banner_video_url.mp4”,
        “ue”: “unlock event url”,
        “da”: false,
        “ca”: 1,
        “cl”: 5,
        “az”:””,
        “bh”: “banner html string”
}
```

Description of the content:

Parameter	**Type**	**Description**
id	String	This is the unique ID that is generated for each new request. Required to tie subsequent messages to a single end to end ad transaction.
ad	Integer	This is the ID of the ad that is returned from the request. Required to post answer.
wi	Integer	This is the width of the ad image.
he	Integer	This is the height of the ad image.

ei	String	This is the hint/question for the user to answer.
ao	String	This is the banner ad type. Value can either be image or multimedia.
bi	String	This is the URL of the image to show (this will be empty if a video ad).
sa	Integer	Show answer. Can be either 1 or 2. If 1, then show the answer to the user on the initial load if 2, then show the answer on 2nd attempt
an	String	The correct answer to the question
uo	Boolean	Show unlock with Adcoin button. The type of ad available can be either: “True” if there is inventory available for this request “False” if there is no paid inventory available, but a default ad is available
ve	String	Video events server url, used to post video events to see video events message
bv	String	The url of the banner video
ue	String	When a user clicks the ‘Unlock Button’, make a HTTP post to the URL within this parameter. Do not include any content with this post.
da	Boolean	Default Ad flag If true, this ad is a Default Ad. If false, it is a paid ad
ca	Integer	The number of correct answers a user has answered. This is for Ad spaces that limit the number of accesses via Adcoin.
cl	Integer	The total number of times a user can access content using Adcoin. This is for Ad spaces that limit the number of accesses via Adcoin.

az	String/JSON	This contains all the different answer choices for a multi choice ad. **Example:** “az”:”{ "type":"1"," ”answers":[{"answer":"An answer", correctness":1}, {"answer":"An answer 2","correctness":0}] }”, **Where:** **Parameter** | **Type** | **Description** Type | String | “1” for an answer the user types in, “2” for a multi choice where the options are displayed to the user (there should be a button per answer that the user can push). Answers | Array | An array containing all the possible answers. Answer | String | The answer to the ad instruction
bh	String	This contains the HTML code of the ad. This will be empty if these is something in bv or bi

TOP TIP!
For the best user experience, Adcoin strongly suggests you do not show an “Unlock content” button to the user in case error 204 or 404 is returned. Only show the option to use Adcoin as an access method if there is an ad available.

3.2.3 Step 3: Post answer (Publisher to Adcoin)

Post the answer as a JSON object to the Ad space url:

```
https://adserve.adcoin.com/display/ad_block/122/?api_key=gHCgudieF88neKFQx38gpJfeK
1BLEldis69nxzUV
```

The JSON object needs to have the following data

```
{
        “id”: “b75977f7-b23e-4f04-b338-a64d92b539ed”,
        “ad”: 7,
        “an”: “answer”,
        “st”: 1455901112.897,
        “et”:  1455901115.897
       “cid”: “1”
        “kp”:
'1494927047.791,1494927048.064,1494927048.318,1494927049.055,1494927049.599,14
94927050.375,1494927050.559,1494927050.767'
}
```

Description of the content:

Parameter	**Type**	**Description**
id	String	This is the unique ID that is generated for each new request. Required to tie subsequent messages to a single end to end ad transaction.
ad	Integer	This is the ID of the ad that is returned from the request. Required to post answer.
an	String	This is the answer to the engagement instructions.
st	Integer	This is the start time the ad was initially loaded. Time must be in seconds.
et	Integer	This is the end time or the time when the answer was submitted. Time must be in seconds.
cid	String	Optional. This parameter is send back to the publishers server to retrieve the reward URL, see section 2.4
kp	String	Optional. The time of each key press the user did to answer the ad, comma separated string. If the number of keypresses doesn’t match the number of characters in the answer then the answer is rejected.

3.2.4 Step 4: Answer verification

With a wrong answer, a ‘406’ error will be returned.
With the right answer, the return will be a JSON object with the follow-on ad.

```
{
        “ad": 7,
        “fo": "image",
        “fi": “follow_on_image_url.png”,
        “fm": “follow-on message",
        “fu": “follow-on url",
        “fh”: “follow-on html”,
        “ru":"reward url",
        “fc”: “follow-on click url”,
        “fv”: “follow-on video url”,
        “ro”: False
}
```

Parameter	**Type**	**Description**
ad	Integer	This is the same Ad ID as in previous steps.
fo	String	This is the follow-on ad type. Value can either be image or multimedia.
fi	String	Where image is selected this is the follow-on ad image URL.
fm	String	This is the follow-on message to be displayed for the user with the follow-on ad.
fh	String	Where html is selected this is the HTML content for the follow-on ad.
fu	String	This is the follow-on URL for the image ad (empty if video is used).
ru	String	This is the reward URL.
fc	String	This is the URL used when the Follow-on Ad is clicked on.
fv	String	Follow on video URL (empty if image ad is used).
ro	Bool	Used to tell if the Ad space is routing the reward through Adcoin or not. If false, the reward URL will be the url in the Ad space, If true, the URL will be the Adcoin Reward API documented below, 3.2.6.

3.2.5 Video event messages (only if video ads are used)

Video event messages are sent from publisher to Adcoin to record event during video playback either for the video ads or the follow-on video ads. All video status messages will always be returned with a code 200 to signify that the message has been received.

```
{
        “id”: “b75977f7-b23e-4f04-b338-a64d92b539ed”
        “state”: 1,
        “video”: “completion”,
}
```

Parameter	**Type**	**Description**
id	String	This is the unique ID that is generated for each new request. Required to tie subsequent messages to a single end to end ad transaction
state	Integer	this can be 1 for playing 2 for pause 3 for ended
video	String	This can be “completion” for the completion video or “follow-on” for a follow-on video

3.2.6 Claim Reward (only if using Adcoin access gateway feature)

This is used to claim the reward from Adcoin, if the Ad space is configured to route the rewards through Adcoin. This is a HTTP GET request. If a HTTP status of 204 is returned, it indicates that the reward has expired. A HTTP status of 404 indicates an error in one of the parameters passed in. A HTTP status of 200 indicates a successful call and the below json will be in the response.

```
{
        “content”: “<html><body>Some Content</body></html>”,\
        “uses”: 1,
        “total_uses”: 4,
        “duration”: 0,
        “total_duration”: 300
 }
```

Parameter	**Type**	**Description**
content	String	This is the content of the page at the URL of the reward specified in the Ad space
uses	Integer	This is the number of times this reward has been used.
total_uses	Integer	This is the total number of times this reward can be used, 0 if this reward has a duration.
duration	Integer	The number of seconds the reward has been valid for.
total_duration	Integer	The total number of seconds that this reward will be valid for.

3.3 Detailed installation instructions

Step 1:
Login to Adcoin platform and ensure you have an existing platform and an Ad space ready. Creation of the platform will automatically create the necessary keys, and the creation of an Ad space will automatically create a URL to refer to. A single platform can be used for single website and also a mobile app. If you have multiple websites/apps, you must create multiple platforms.

JavaScript Key - this is used for web based Ad spaces. Using the provided Website API URL automatically includes the correct key. If you are building and formatting your own ad user interface, use this key as your api_key in the URL structure. You can access your JavaScript Key via the “edit platform” screen.

Client ID - Use this as the api_key when creating a mobile application to use Ad spaces. Using the provided Mobile API URL automatically includes the correct key. If you are building your own mobile app or custom Ad snippet use the Client ID in your URL structure. You can access your Client ID via the “edit platform” screen.

Step 2:
As the user requests an Ad, make an API call using the ad request URL for the Ad space you want. You can pass one or more user related attributes based on what you know of the user, or not pass any attributes at all to use the Ad space default values (set via Adcoin UI). See section 3.2.1 for the format of the URL.

Step 3:
In return you will receive a JSON object that includes the details about the Ad, the task the user must complete. See section 3.2.2 for details. Deliver these to the end user. User will now submit an answer to the task.

Step 4:
Pass the user’s answer to Adcoin. Refer to section 3.2.3 for the format of the URL to use. As a response, you will either get; the same data you posted, along with an error code 406, that signifies that the answer was incorrect, or a new JSON object which signifies that the answer was correct.

If the user repeatedly answers wrong (3 or more times), it is recommended that you discard the ad, and request a new ad for the user. To do this, return to step 2. Consider also using the “ids” parameter to ensure that the user is not served the same ad again (Refer to section 3.2.1 for details).

Step 5:
Pass the Follow-on Ad to the user, and make the reward available for them.

4. Custom method

Where the functionality of the API doesn't meet your requirements, you want to integrate your backend systems (such as customer database) or you have significant number of sites or assets or mobile apps to cover, we can create a custom integration. Work with our technology team to integrate our platform the way you need. The work will be based on the standard API, but we can create any type of additional functionality you need to run efficiently. Note that there may be a cost associated with a custom integration, which will be determined based on the complexity and the type of work required.

Contact Adcoin Sales team (sales@adcoin.com) or support team (support@adcoin.com) to learn more about custom integrations.

5. Testing and troubleshooting

5.1 Testing your implementation

On the Adcoin platform, each Ad space can be set to a "sandbox mode". To test the Adcoin API and other features, set your Ad space in sandbox mode. No earnings will be recorded whilst the Ad space is in this mode. Sandbox mode can only be turned on when the Ad space is inactive. Once testing has been completed, turn the sandbox mode off.

5.2 Common errors and troubleshooting

5.2.1 Error codes

There are 5 error codes you may encounter:

- Code 200: This is not an error, but a code to signify that video status message has been received successfully
- Error 204: This error is returned if there are no ads available, either inventory or default ads that match the criteria on the request.
- Error code 404: This error signifies an error in format of a post back to Adcoin, and can be caused by a number of issues. Usually it is due to error in the syntax, or an invalid ID or key being used.
- Error code 406: This is a specific error code that you will receive when then answer that user has provided to a task is incorrect. This is a soft error, after which user should be prompted to try again.
- Error code 500: There is an issue on Adcoin side, please contact Adcoin support.

5.2.2 Troubleshooting Code Snippet method

If you experience errors during the use of Code Snippet, and you have provided custom parameters as detailed in section 2.2, please check the syntax of the JavaScript variables. If you continue to experience issues, or haven't added any custom parameters to the code snippet, please copy and paste the full snippet code again. Ensure the code snippet is not altered.

5.2.3 Troubleshooting REST API

If you experience issues with the API, see the following table for steps:

Problem	**Resolution**
No response from API	- Check that you initiate a call with HTTPS - Check that the URL hostname is correct - Copy URL from your Ad space and try again - Check that you have Internet access - Check DNS resolution is working - Check that you have access to port 443
API returns an Error code 404	- Check the syntax of your URL, including any additional parameters - Check that you are using the correct Key (Mobile key or JavaScript key) as your api_key - Check that you use a valid unique ID - Check that you use a valid Ad space ID - Check that you use a valid Ad ID
API returns an error code 406	- User gave an incorrect answer for the task in an ad. Prompt for a new try.
API returns an error code of 205	- The current ad has expired, load a new ad.
API returns an error code 204	- There are no paid or default ads available - Adcoin recommends enabling default ads that can be served to the users in case there are no paid ads available
API returns an error code 500	- Contact our support team at support@adcoin.com

If these troubleshooting steps do not rectify the issue, please contact Adcoin support team at support@adcoin.com

Appendix A – list of supported languages

Language	**Code**
Afrikaans	af
Arabic	ar
Asturian	ast
Azerbaijani	az
Bulgarian	bg
Belarusian	be
Bengali	bn
Breton	br
Bosnian	bs
Catalan	ca
Czech	cs
Welsh	cy
Danish	da
German	de
Greek	el
English	en
Australian English	en-au
British English	en-gb
Esperanto	eo
Spanish	es
Argentinian Spanish	es-ar
Mexican Spanish	es-mx
Nicaraguan Spanish	es-ni
Venezuelan Spanish	es-ve
Estonian	et
Basque	eu
Persian	fa
Finnish	fi
French	fr
Frisian	fy
Irish	ga
Galician	gl
Hebrew	he
Hindi	hi
Croatian	hr
Hungarian	hu
Interlingua	ia
Indonesian	id
Ido	io
Icelandic	is
Italian	it
Japanese	ja
Georgian	ka
Kazakh	kk
Khmer	km
Kannada	kn
Korean	ko
Luxembourgish	lb
Lithuanian	lt
Latvian	lv
Macedonian	mk
Malayalam	ml
Mongolian	mn
Marathi	mr
Burmese	my
Norwegian Bokmal	nb
Nepali	ne
Dutch	nl
Norwegian Nynorsk	nn
Ossetic	os
Punjabi	pa
Polish	pl
Portuguese	pt
Brazilian Portuguese	pt-br
Romanian	ro
Russian	ru
Slovak	sk
Slovenian	sl
Albanian	sq
Serbian	sr
Serbian Latin	sr-latn
Swedish	sv
Swahili	sw
Tamil	ta
Telugu	te
Thai	th
Turkish	tr
Tatar	tt
Udmurt	udm
Ukrainian	uk
Urdu	ur
Vietnamese	vi
Simplified Chinese	zh-cn
Simplified Chinese	zh-hans
Traditional Chinese	zh-hant
Traditional Chinese	zh-tw