



ADS API #TECHTALK

AGENDA



Introductions



Setup & Getting Started



Workshop



PARTNER ENGINEERING



@hwz



@jaakkosf



@brandonmblack



TEAM LOCATIONS

Partner Engineering, Ads API



WHAT WE DO

API DESIGN

Implementation review for all API features.

TECHNICAL COMMS

Newsletters and all technical communications.

COMMUNITY & PARTNER SUPPORT

Forums, inquiries and office hours events.

DOCS, CODE SAMPLES AND SDKs

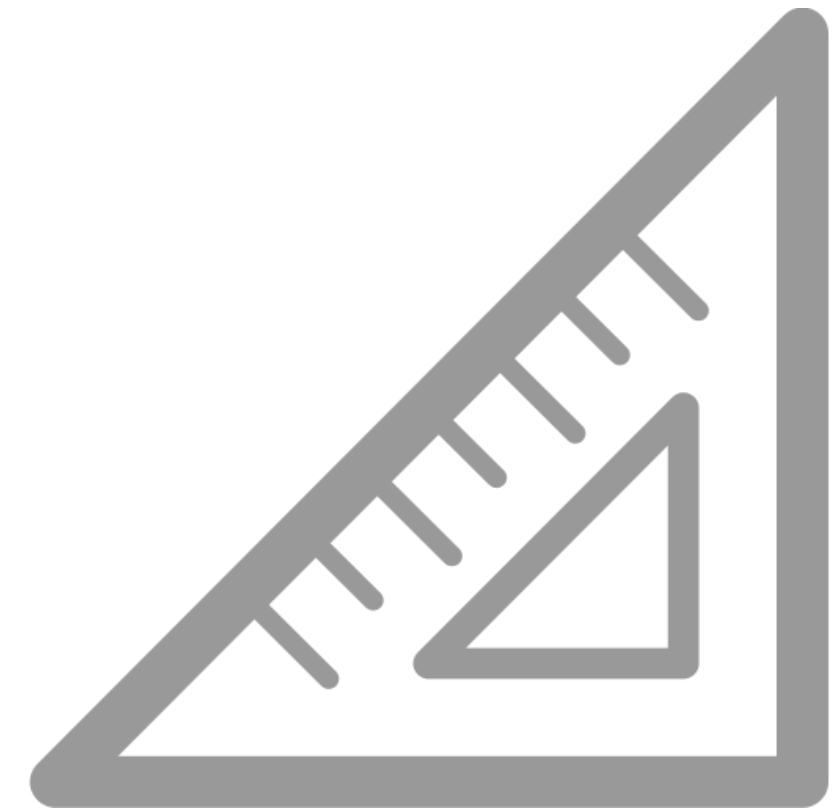
Guides, tutorials, examples.

INTEGRATION

On-boarding and consulting.

PRODUCT FEEDBACK

Internal advocacy and product feedback.



WHAT WE DO

API DESIGN

Implementation review for all API features.

TECHNICAL COMMS

Newsletters and all technical communications.

COMMUNITY & PARTNER SUPPORT

Forums, inquiries and office hours events.

DOCS, CODE SAMPLES AND SDKs

Guides, tutorials, examples.

INTEGRATION

On-boarding and consulting.

PRODUCT FEEDBACK

Internal advocacy and product feedback.



WHAT WE DO

API DESIGN

Implementation review for all API features.

TECHNICAL COMMS

Newsletters and all technical communications.

COMMUNITY & PARTNER SUPPORT

Forums, inquiries and office hours events.

DOCS, CODE SAMPLES AND SDKs

Guides, tutorials, examples.

INTEGRATION

On-boarding and consulting.

PRODUCT FEEDBACK

Internal advocacy and product feedback.



WHAT WE DO

API DESIGN

Implementation review for all API features.

TECHNICAL COMMS

Newsletters and all technical communications.

COMMUNITY & PARTNER SUPPORT

Forums, inquiries and office hours events.

DOCS, CODE SAMPLES AND SDKs

Guides, tutorials, examples.

INTEGRATION

On-boarding and consulting.

PRODUCT FEEDBACK

Internal advocacy and product feedback.



WHAT WE DO

API DESIGN

Implementation review for all API features.

TECHNICAL COMMS

Newsletters and all technical communications.

COMMUNITY & PARTNER SUPPORT

Forums, inquiries and office hours events.

DOCS, CODE SAMPLES AND SDKs

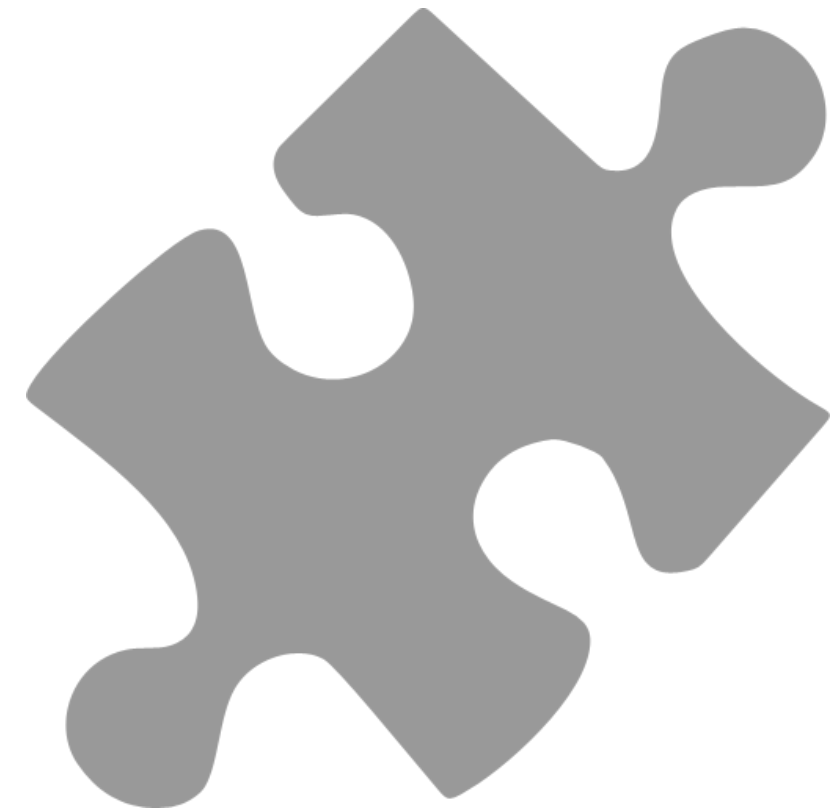
Guides, tutorials, examples.

INTEGRATION

On-boarding and consulting.

PRODUCT FEEDBACK

Internal advocacy and product feedback.



WHAT WE DO

API DESIGN

Implementation review for all API features.

TECHNICAL COMMS

Newsletters and all technical communications.

COMMUNITY & PARTNER SUPPORT

Forums, inquiries and office hours events.

DOCS, CODE SAMPLES AND SDKs

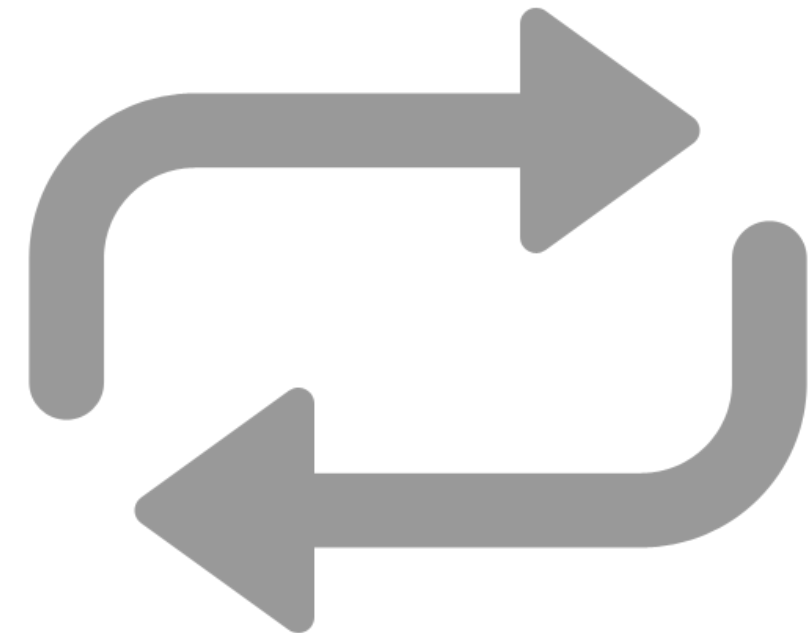
Guides, tutorials, examples.

INTEGRATION

On-boarding and consulting.

PRODUCT FEEDBACK

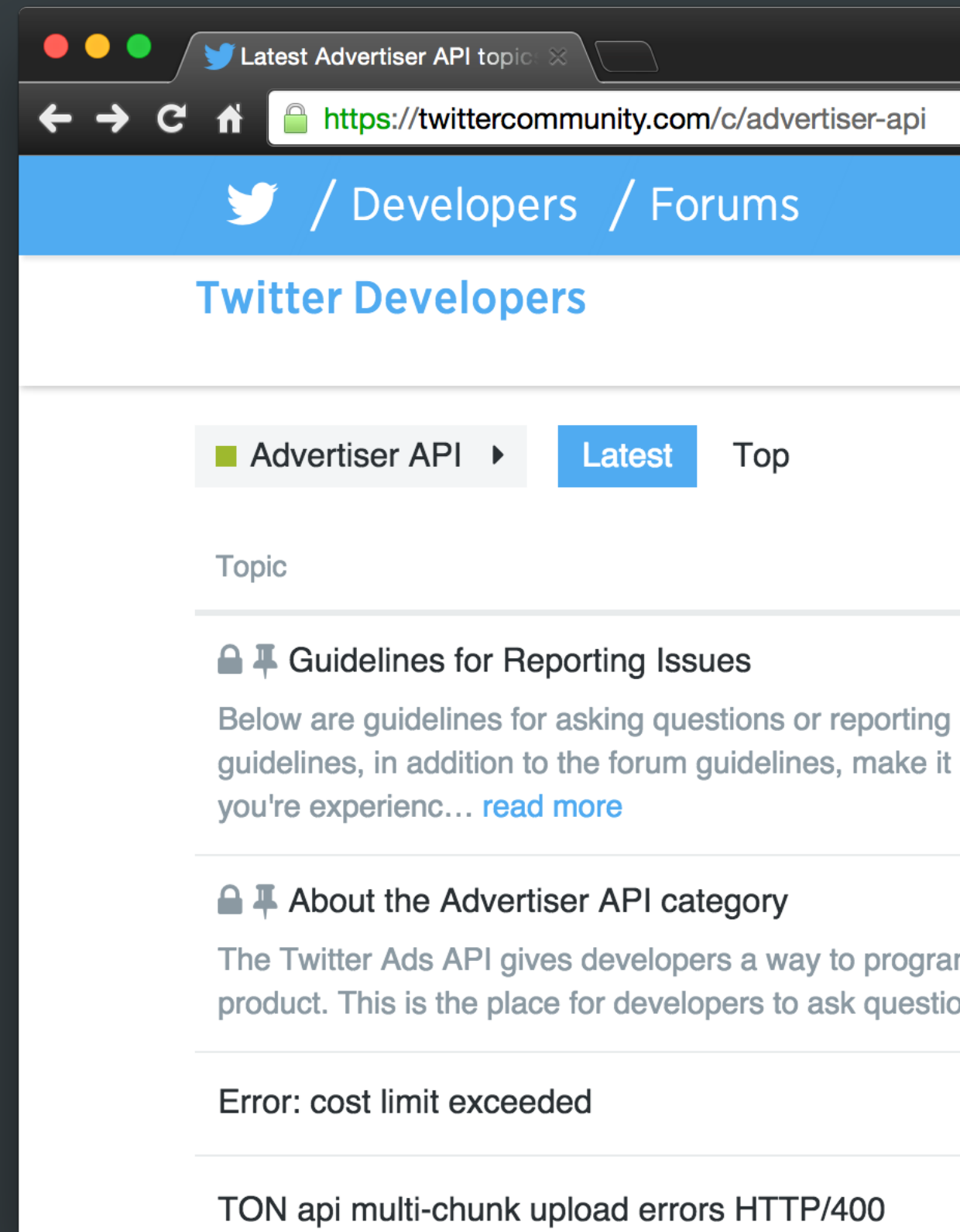
Internal advocacy and product feedback.



Help?

Join the conversation in the
Twitter Community forums

Follow [@AdsAPI](https://twitter.com/AdsAPI) for platform updates

A screenshot of a web browser displaying the Twitter Developers forum page. The browser's address bar shows the URL 'https://twittercommunity.com/c/advertiser-api'. The page has a blue header with the Twitter logo and navigation links for 'Developers' and 'Forums'. Below the header, the title 'Twitter Developers' is displayed. A filter bar shows 'Advertiser API' selected, with 'Latest' and 'Top' sorting options. The main content area lists forum topics, including 'Guidelines for Reporting Issues' and 'About the Advertiser API category'. The bottom of the screenshot shows two specific error messages: 'Error: cost limit exceeded' and 'TON api multi-chunk upload errors HTTP/400'.

Latest Advertiser API topic

← → ↻ 🏠 <https://twittercommunity.com/c/advertiser-api>

🐦 / Developers / Forums

Twitter Developers

■ Advertiser API ▶ Latest Top

Topic

🔒📌 Guidelines for Reporting Issues

Below are guidelines for asking questions or reporting guidelines, in addition to the forum guidelines, make it you're experienc... [read more](#)

🔒📌 About the Advertiser API category

The Twitter Ads API gives developers a way to program product. This is the place for developers to ask question

Error: cost limit exceeded

TON api multi-chunk upload errors HTTP/400

GETTING STARTED

SETUP

RUBY VERSION

```
$ ruby -v  
ruby 2.0.0p643 (2015-02-25 revision 49749) [x86_64-darwin14.1.0]
```

INSTALLING TOOLS & DEPENDENCIES

```
$ gem install pry jsonpretty twurl
```

Note: Depending on how your Ruby installation is setup, you may need to run the above “gem install” commands with “sudo”.



AUTHORIZATION

```
$ twurl authorize --consumer-key key --consumer-secret secret
```



AUTHORIZATION



OAUTH 1.0A

The Ads API uses OAuth 1.0a for authorization and implements the 3-legged OAuth flow.

More info at: <https://dev.twitter.com/oauth>



AUTHORIZATION

API key and secret are available under the “Keys and Access Tokens” tab.

Application Management

My Ads API App

Test OAuth

Details

Settings

Keys and Access Tokens

Permissions

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

| | |
|------------------------------|---|
| Consumer Key (API Key) | 0Khhjf5jrb9niakb6dGTeAVID |
| Consumer Secret (API Secret) | gENTjg6VUJyIBvA1COyPbSefNPofpZP8BfeJ8TZhNmFVugBrYp |
| Access Level | Read and write (modify app permissions) |
| Owner | brandonmblack |
| Owner ID | 15678855 |

Application Actions

Regenerate Consumer Key and Secret

Change App Permissions

CONSUMER KEY & SECRET

Your consumer key is a publicly visible identifier for your app.

You should never share your consumer secret.



SETUP

AUTHORIZATION

```
$ twurl authorize --consumer-key key --consumer-secret secret
```

VALIDATE SETUP

```
$ cat ~/.twurlrc
```

TRY IT OUT

```
$ twurl -H ads-api-sandbox.twitter.com "/0/accounts" | jsonpretty
```



DEVELOPER RATE LIMITS

READS

25
REQUESTS

PER MINS / ENDPOINT

WRITES

100
REQUESTS

PER MIN / CATEGORY

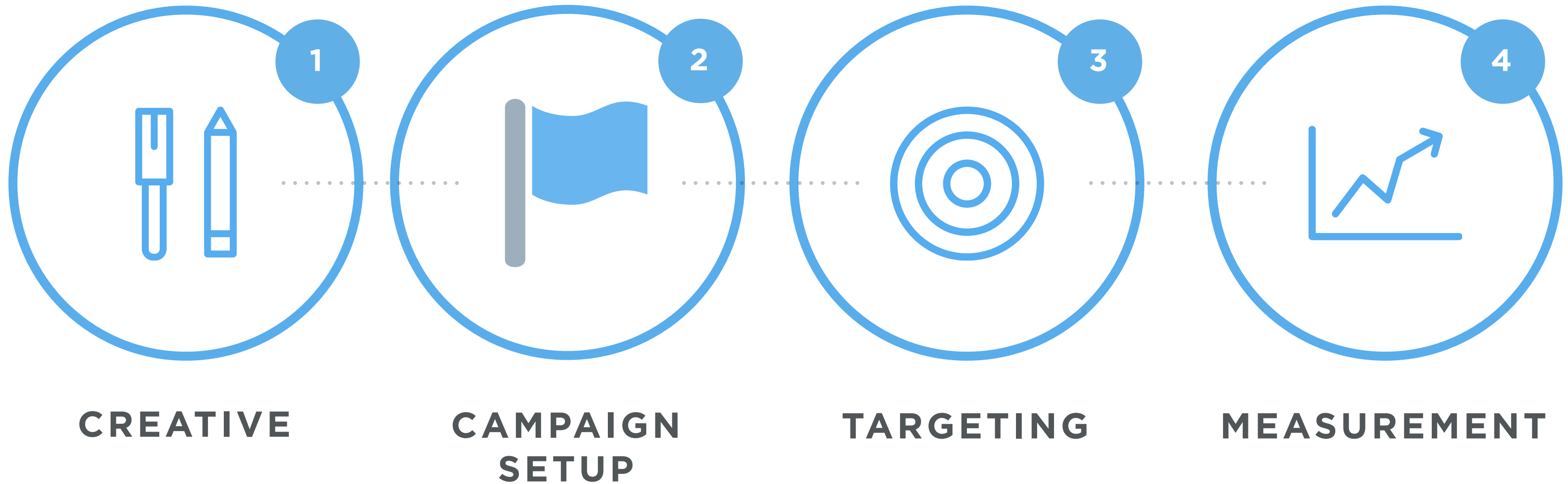
More info at: <https://dev.twitter.com/ads/basics/rate-limiting>



The background is a solid blue color with a subtle, light blue grid pattern. Scattered across the grid are several geometric elements: small circles of varying sizes, larger circles with concentric outlines, and plus signs. These elements are positioned at various points, some aligned with the grid intersections and others slightly offset, creating a modern, minimalist aesthetic.

WORKSHOP

WHAT WE'LL LEARN TODAY



SETUP

INSTALLING THE RUBY SDK

```
$ gem install twitter-ads
```

START AN INTERACTIVE SESSION

```
$ twitter-ads
```

Note: Depending on how your Ruby installation is setup, you may need to run the above “gem install” commands with “sudo”.



CLIENT

```
require 'twitter-ads'

# enable sandbox mode
CLIENT.options[:sandbox] = true

# load up the account instance
account = CLIENT.accounts.first
```



CAMPAIGN

```
# create your campaign
campaign = TwitterAds::Campaign.new(account)
campaign.funding_instrument_id = account.funding_instruments.first.id
campaign.daily_budget_amount_local_micro = 1_000_000
campaign.name = 'my first campaign'
campaign.paused = true
campaign.start_time = Time.now.utc
campaign.save
```



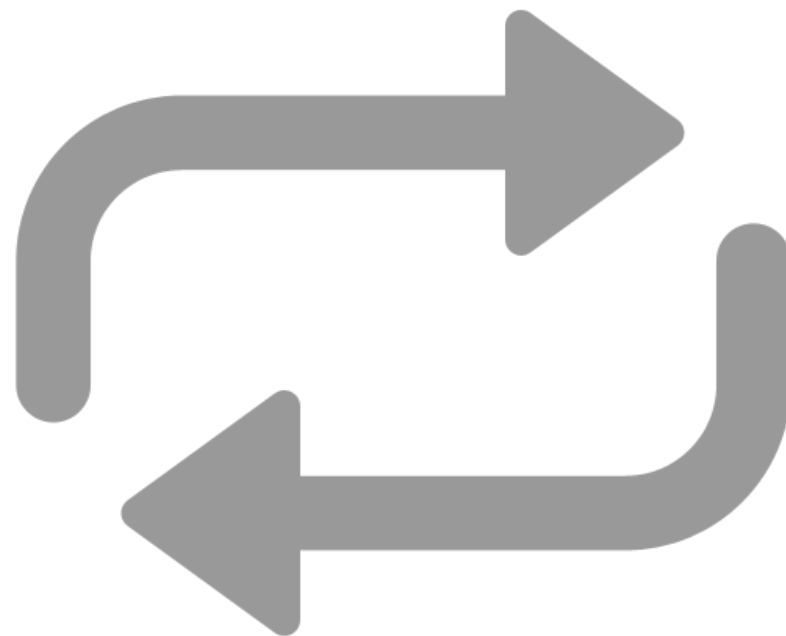
LINE ITEM

```
# create a line item for the campaign
line_item = TwitterAds::LineItem.new(account)
line_item.campaign_id      = campaign.id
line_item.name              = 'my first ad'
line_item.product_type     = TwitterAds::Product::PROMOTED_TWEETS
line_item.placements       = [TwitterAds::Placement::ALL_ON_TWITTER]
line_item.objective        = TwitterAds::Objective::TWEET_ENGAGEMENTS
line_item.bid_amount_local_micro = 10_000
line_item.paused           = true
line_item.save
```

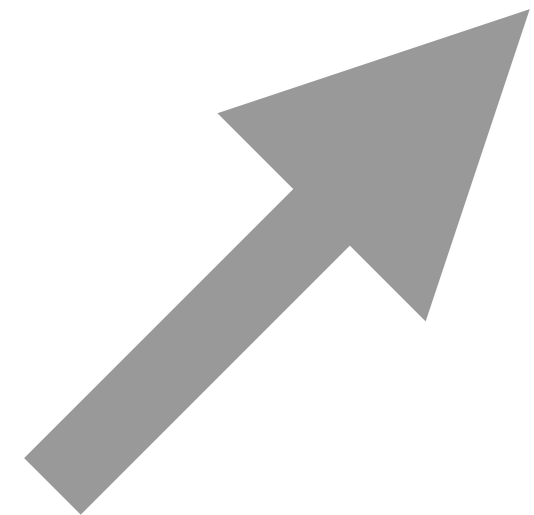




TWEET



RETWEET



PROMOTED TWEET



PROMOTED TWEET

```
# resource url for tweet creation
resource = "/0/accounts/#{account.id}/tweet"

# create request for a simple null-casted tweet
tweet_params = { status: 'Hello @AdsAPI!' }
request = TwitterAds::Request.new(CLIENT, :post, resource, params: tweet_params)
tweet = request.perform

# promote the tweet using our line item
promoted_tweet = TwitterAds::Creative::PromotedTweet.new(account)
promoted_tweet.line_item_id = line_item.id
promoted_tweet.tweet_id     = tweet.body[:data][:id]
promoted_tweet.save
```



PROMOTED TWEET (CARD)

```
# create request for a null-casted tweet with a website card
website_card = TwitterAds::Creative::WebsiteCard.all(account).first
tweet_params = { status: "Hello @AdsAPI #{website_card.preview_url}" }
request = TwitterAds::Request.new(client, :post, resource, params: tweet_params)
tweet = request.perform
```



TARGETING

```
# fetching targeting criteria values
resource = '/0/targeting_criteria/locations'
params   = { location_type: 'CITY', q: 'port' }
request  = TwitterAds::Request.new(CLIENT, :get, resource, params: params)
cursor   = TwitterAds::Cursor.new(nil, request)
```

```
# add targeting criteria
targeting_criteria = TwitterAds::TargetingCriteria.new(account)
targeting_criteria.line_item_id = line_item.id
targeting_criteria.targeting_type = 'LOCATION'
targeting_criteria.targeting_value = '00a8b25e420adc94'
targeting_criteria.save
```



```
# limit request count and grab the first 10 line items from TwitterAds::Cursor
line_items = account.line_items(nil, count: 10)[0..9]

# the list of metrics we want to fetch
metrics = [:billed_engagements, :billed_follows]

# fetching stats on the instance
line_items.first.stats(metrics)

# fetching stats for multiple line items
ids = line_items.map(&:id)
TwitterAds::LineItem.stats(account, ids, metrics)
```



Q&A



#THANKYOU

