

# facebook

## Facebook广告API 入门到精通？

Yu LI  
**Solutions Engineer**  
**APAC**

# 了解广告API

<https://developers.facebook.com/docs/marketing-apis>

# technology to help



Automate

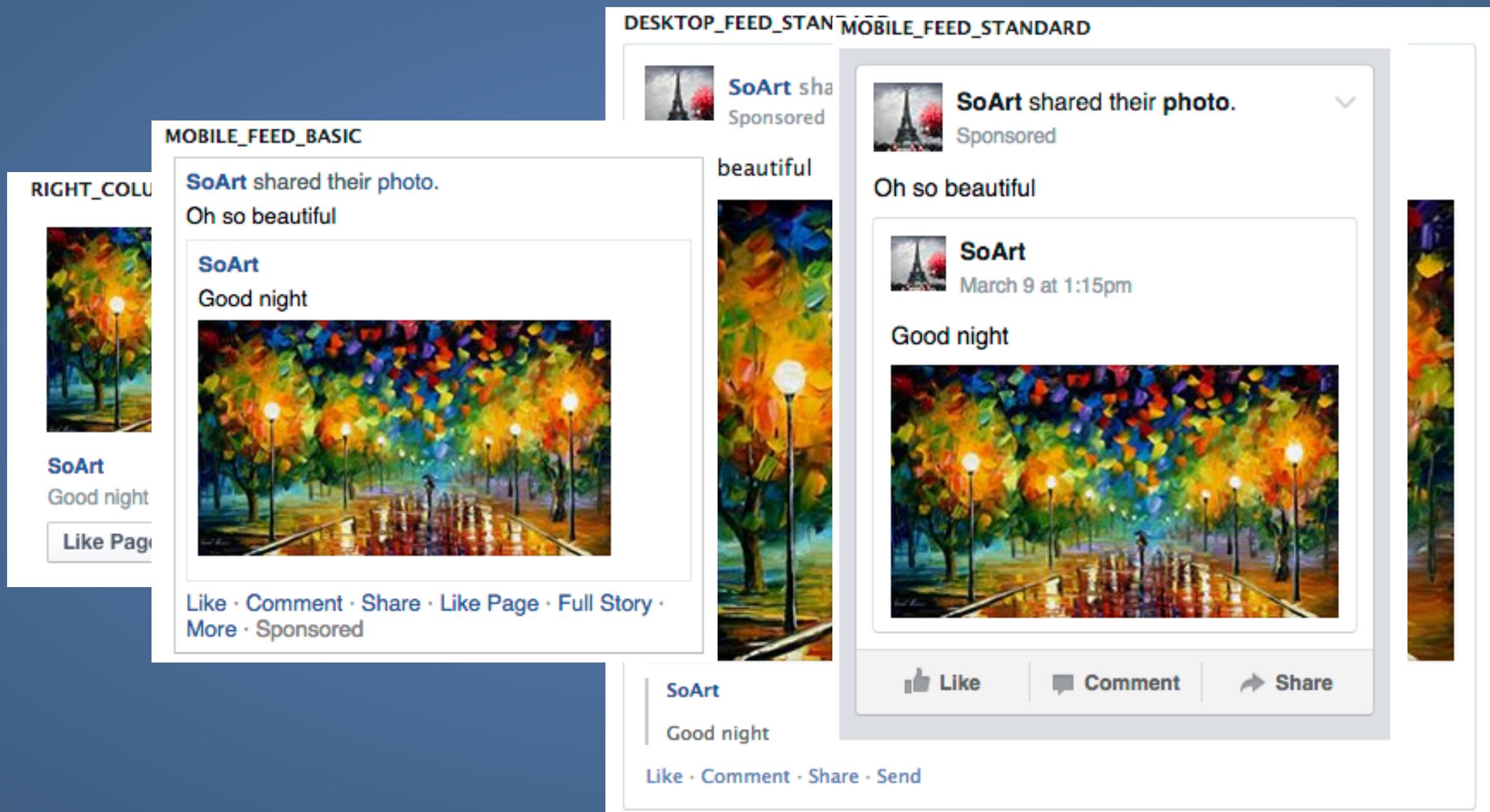


Integrate



Scale

# Facebook广告系统：跨设备平台，面向真实用户



# Facebook广告系统：丰富的营销目标与表现形式

选择营销活动目标

帮助：选择目标

- 推广帖子
- 推广主页
- 吸引更多网站访客
- 增加网站的转化量
- 增加应用安装量
- 提高应用使用率
- 覆盖业务周边的人群
- 增加活动参与人数
- 吸引用户领取优惠
- 获得视频观看量

## Available example ads:

- Page Likes – Inline Fan Creative
- Page Likes – Inline Fan Creative (Large Format)
- Page Likes – Shared Page Story
- Page Likes – Creative Optional Flag (Uni)
- Page Likes – Creative Optional Flag 2
- Page Likes – Creative Optional Flag 3
- Post Engagement – Status Update Story
- Post Engagement – Photo Post Story
- Post Engagement – Link Post Story
- Post Engagement – Multi-Product Ad Post
- Event Responses – Inline RSVP Creative
- Event Responses – Event Creation Story
- Event Responses – Shared Event Story
- Website Clicks/Conversions – Standard Creative
- Website Clicks/Conversions – Standard Creative (With Page)
- Website Clicks/Conversions – Shared Link Story
- Canvas/Mobile App Installs/Engagement – Shared App Story
- Video Plays – Shared Video Story
- Invalid – Object ID Not App For Platform Context Creative
- Inline Image – Inline Fan Creative Using Image URL
- Type 1 Ad – Image URL
- Adgroup ID Preview – Creative Optional Flag
- Ad ID Preview – Event Creation Story
- Adgroup ID Preview – Event Creation Story
- Creative ID Preview – Event Creation Story
- Website Clicks/Conversions – Standard Creative With Cropping
- Website Clicks/Conversions – Standard (With Page) With Cropping
- Inline Image – Inline Fan Creative Using Image URL – With Crop
- Adgroup ID preview – With cropping
- Creative ID preview – With cropping
- Ad ID preview – With cropping2
- Adgroup ID preview – With cropping2
- Creative ID preview – With cropping2
- Creative image url crops
- Unsupported Creative Type 9 (Ad ID)
- Type 27 with post data
- Product Item ID Preview
- Retailer ID Preview

Render Previews

# Facebook广告系统：基于真实用户投放系统

你希望广告如何呈现？

**单个图片和链接**  
广告一次仅显示一张图片。你可以最多上传 6 张图片进行测试，选出最适合的一张。

**一个广告使用多张图片**  
同时显示 5 张图片，而无需多付任何费用。用户可滚动查看所有图片。详细了解

**设置广告受众**

帮助：选择受众

选择不同图片创建多图片广告

添加最多6张图片。你可以上传新图片、使用图库的图片或从...  
创建多个广告  
每添加一张图片

**设置广告预算**

预算 每天 10.00 \$ USD  
排程 从今天起长期投放广告系列  
设置开始和结束日期

优化目的 帖文互动

价格 竞价将为增加主页帖文互动率而优化。你在每次广告得到展示时支付费用。  
以最优价格获得最多的帖子互动 - 你将为展示次数付费  
为单次帖文互动设置你愿意支付的金额

隐藏高级选项

将了解你业务的人群作为目标受众  
你可以创建自定义受众，将广告展示给联系人、网站访客或应用用户。创建自定义受众。

地区 美国 整个美国境内  
添加国家/地区、州/省、城市、邮编或地址

预计每日预算  
1,800-4,800  
这只是根据广告定位估算出来的

这个地区的所有人  
年龄 18 - 65+  
性别 所有 男性 女性  
语言 输入语言...  
更多定位条件

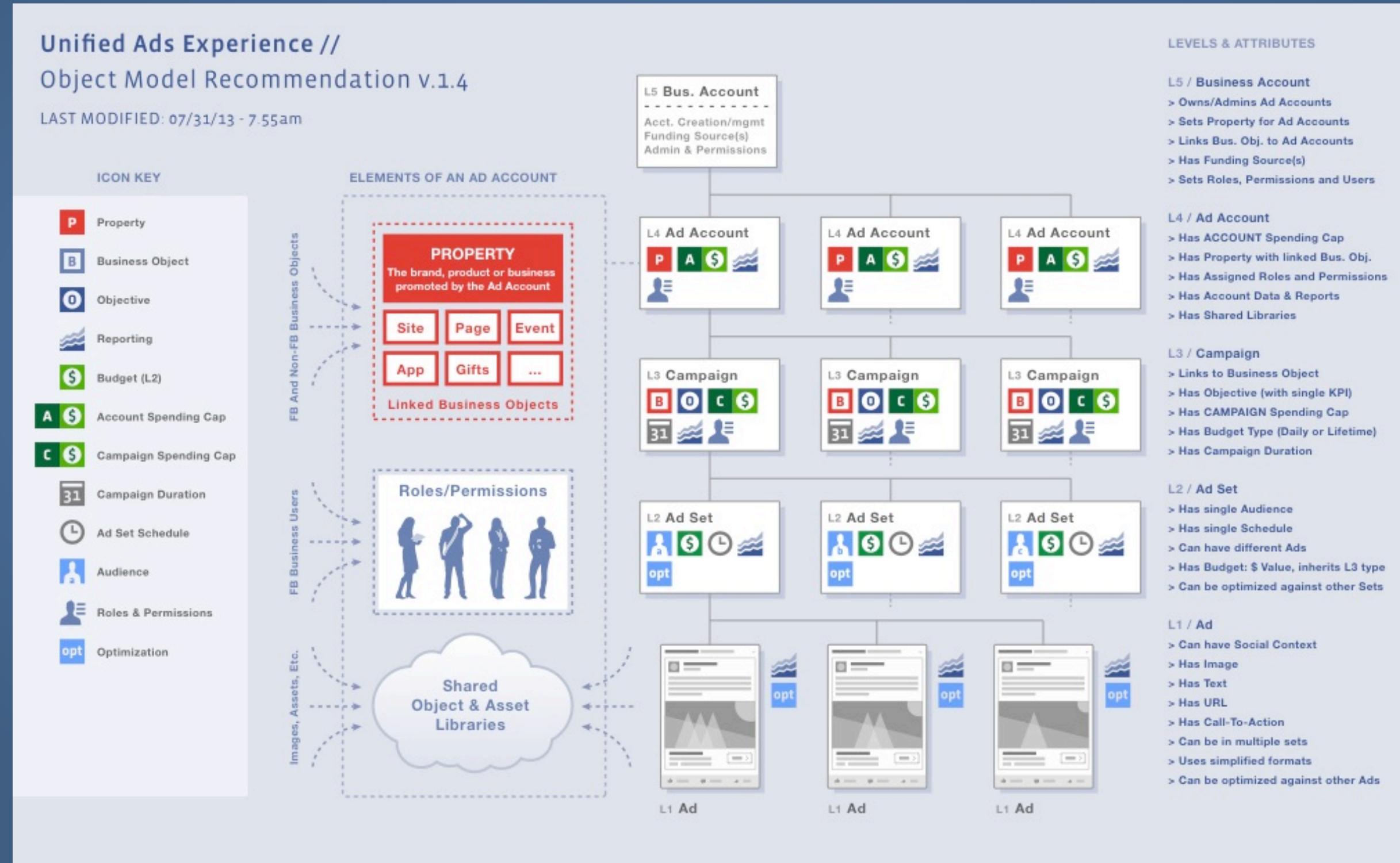
广告受众详情

广告受众过广  
具体 广泛

受众详情：  
地点：美国  
年龄：18 - 65+

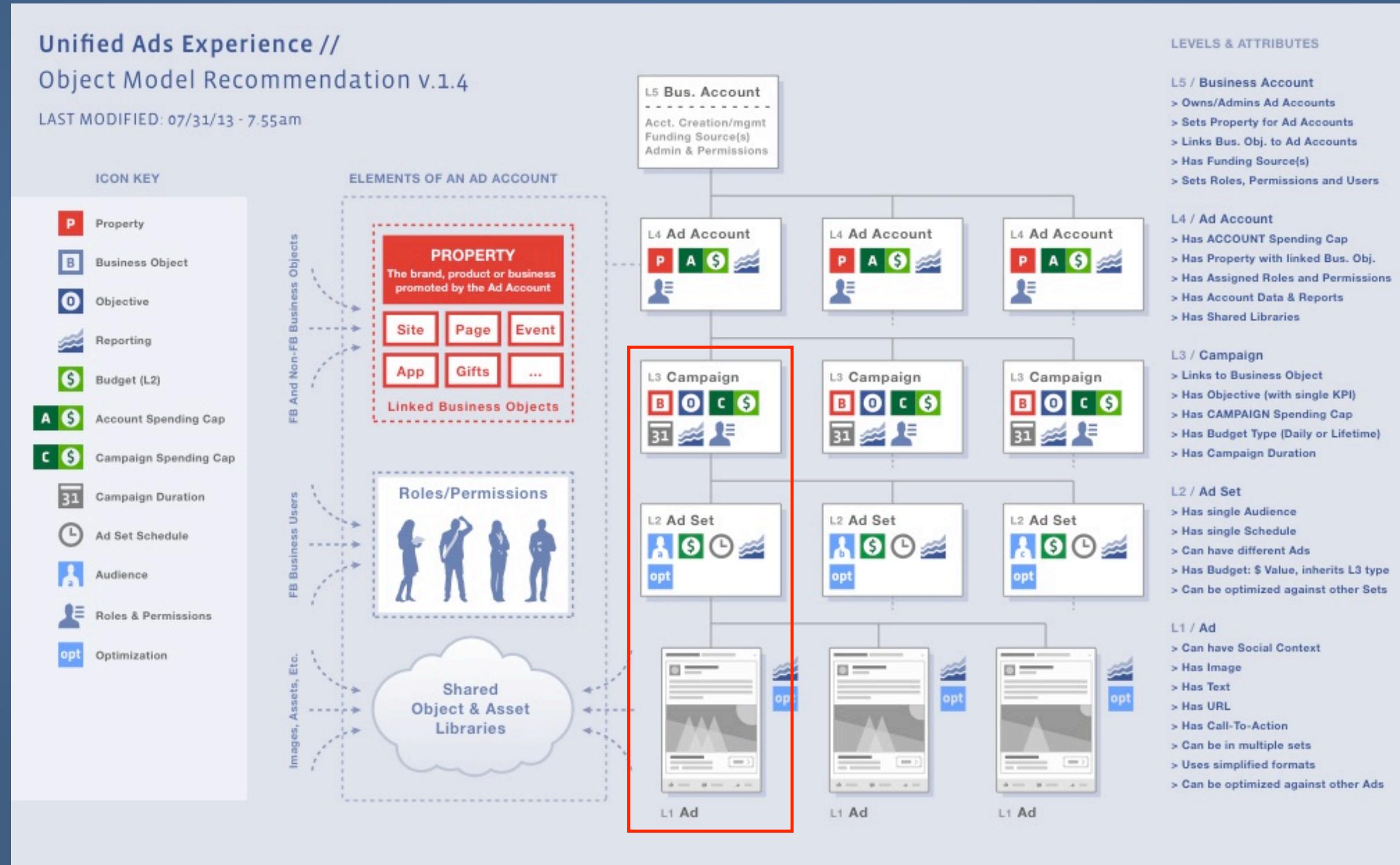
预估覆盖：180,000,000位用户

# Facebook广告系统的架构



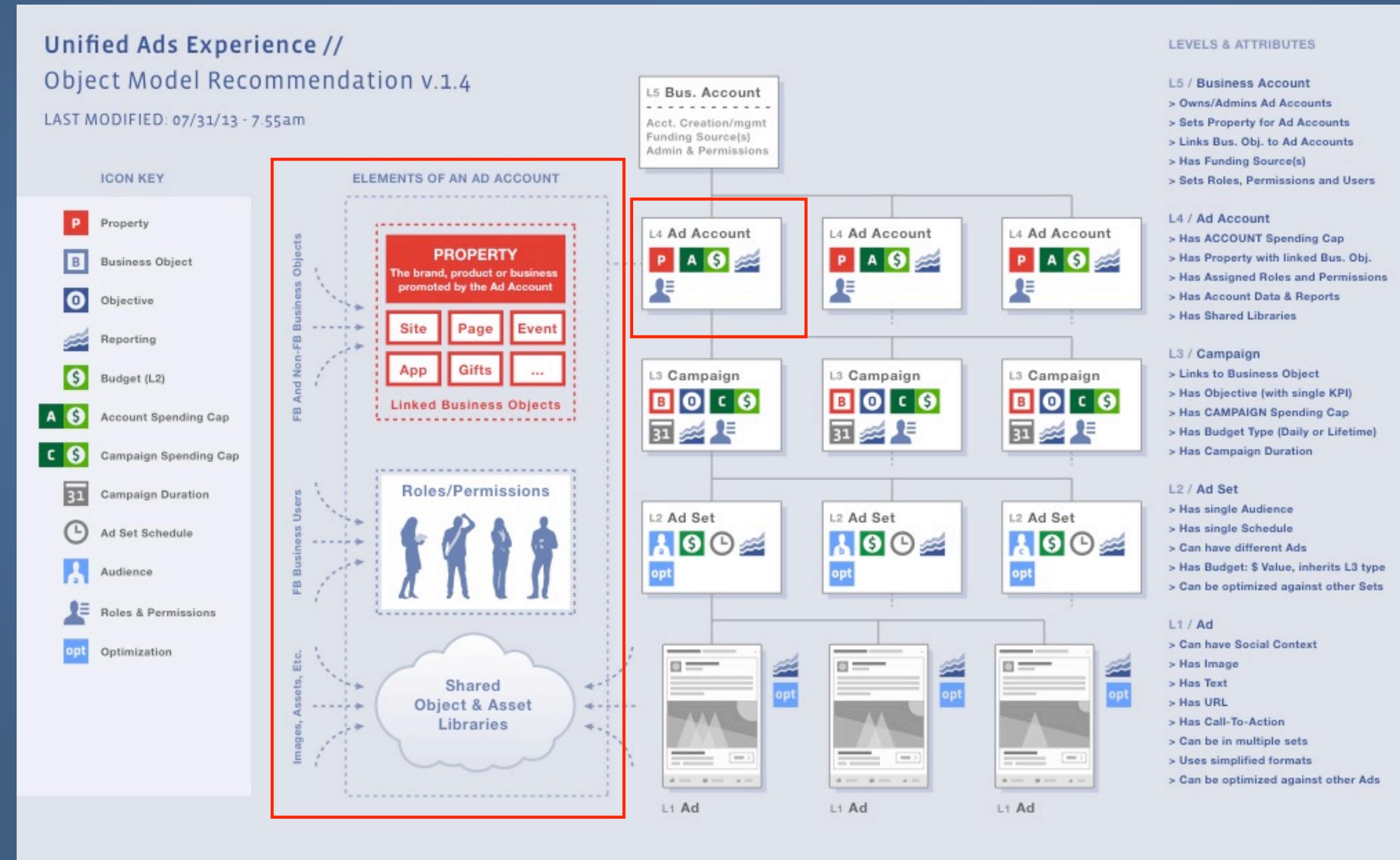
# 广告的数据结构类似于树

Campaign x Ad Set x Ad, 以ID相连



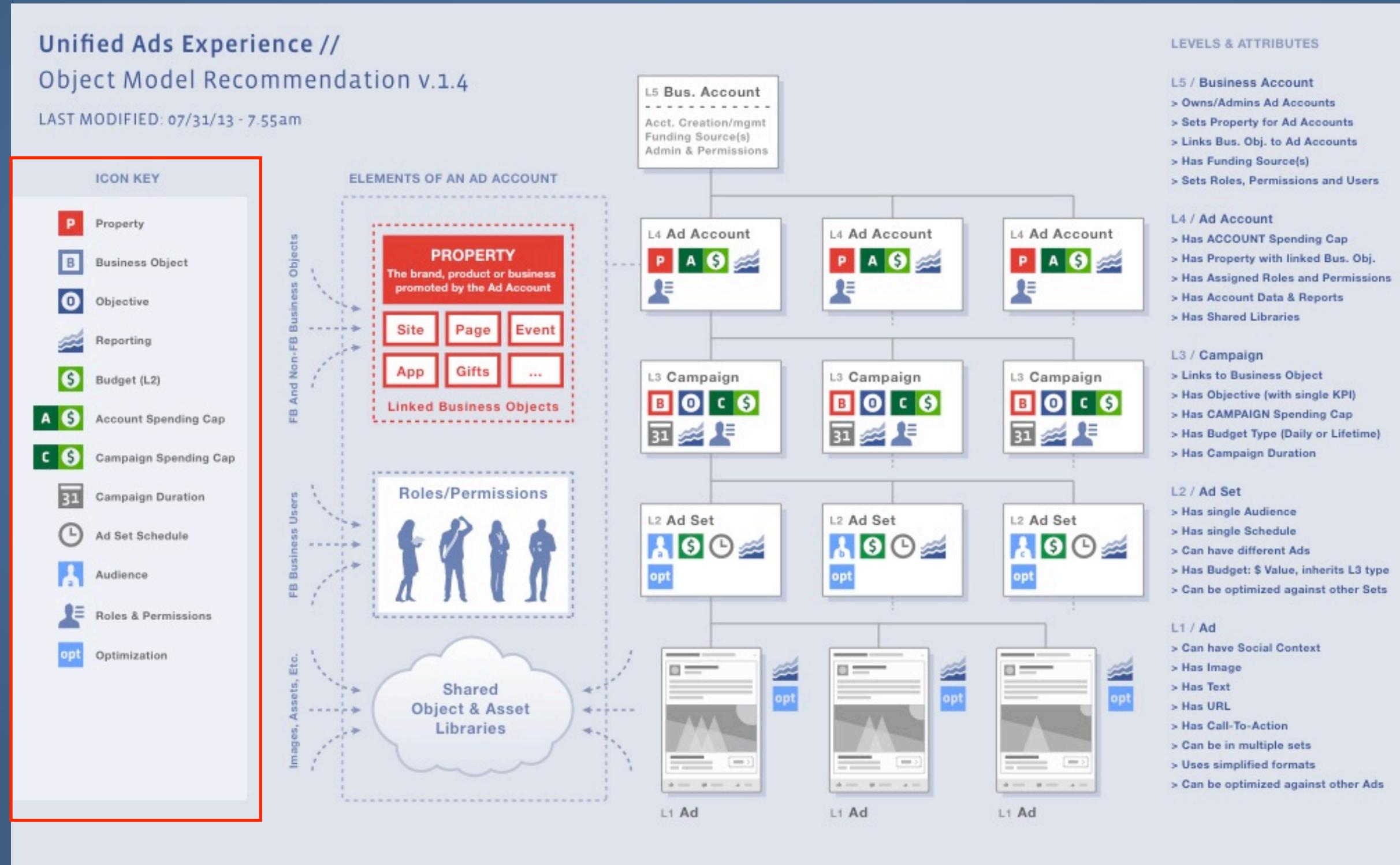
# 位于中心地位的广告账户

拥有图片等各类资产，关联Page, App, Events, 为FB用户指定角色



# 注意各类要素存在的层级

Report? Budget? Objective? Audience?



# Facebook的策略：API优先

- 我们是行动优先， API优先 (mobile-first, API first) .
  - 产品开发是按照如下的顺序
    - 首先是API
    - 然后是Power Editor
    - 最后是Ads Manager

"我可以等待Power Editor"

• 最终不是所有的API都有UI支持  
• 早起的鸟儿有食吃

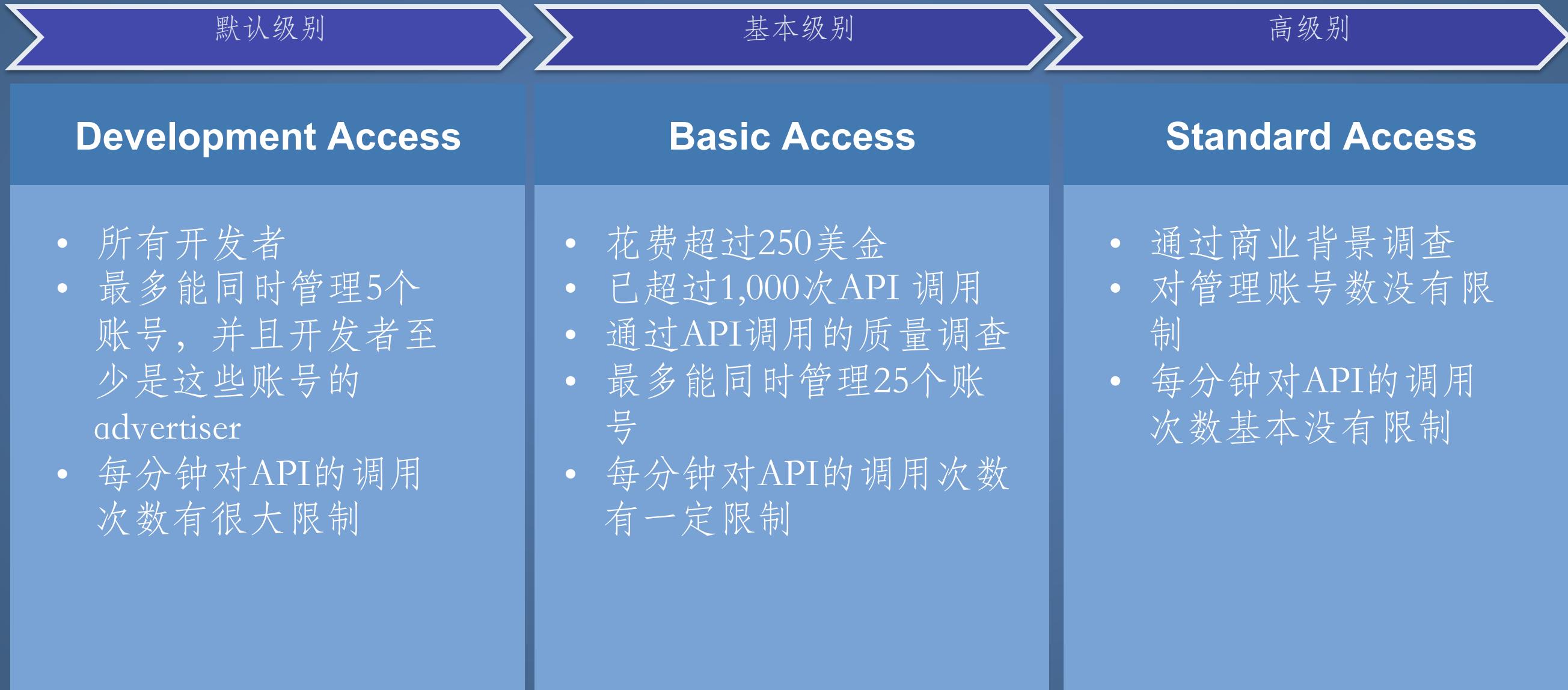
# API能做什么？

- API能做到任何UI能做的事情
  - 创建广告，更改竞价，调整受众等等
- API还能完成已发布但是未出现在UI中的功能
  - 例如DPA for mobile广告，例如精细化受众定义
- 需要使用API的情况
  - 大规模自动化的广告购买与调整
  - 精细化的广告策略
  - 使用新的广告产品，抢占先机
  - 程序化获取广告报表数据（insights）

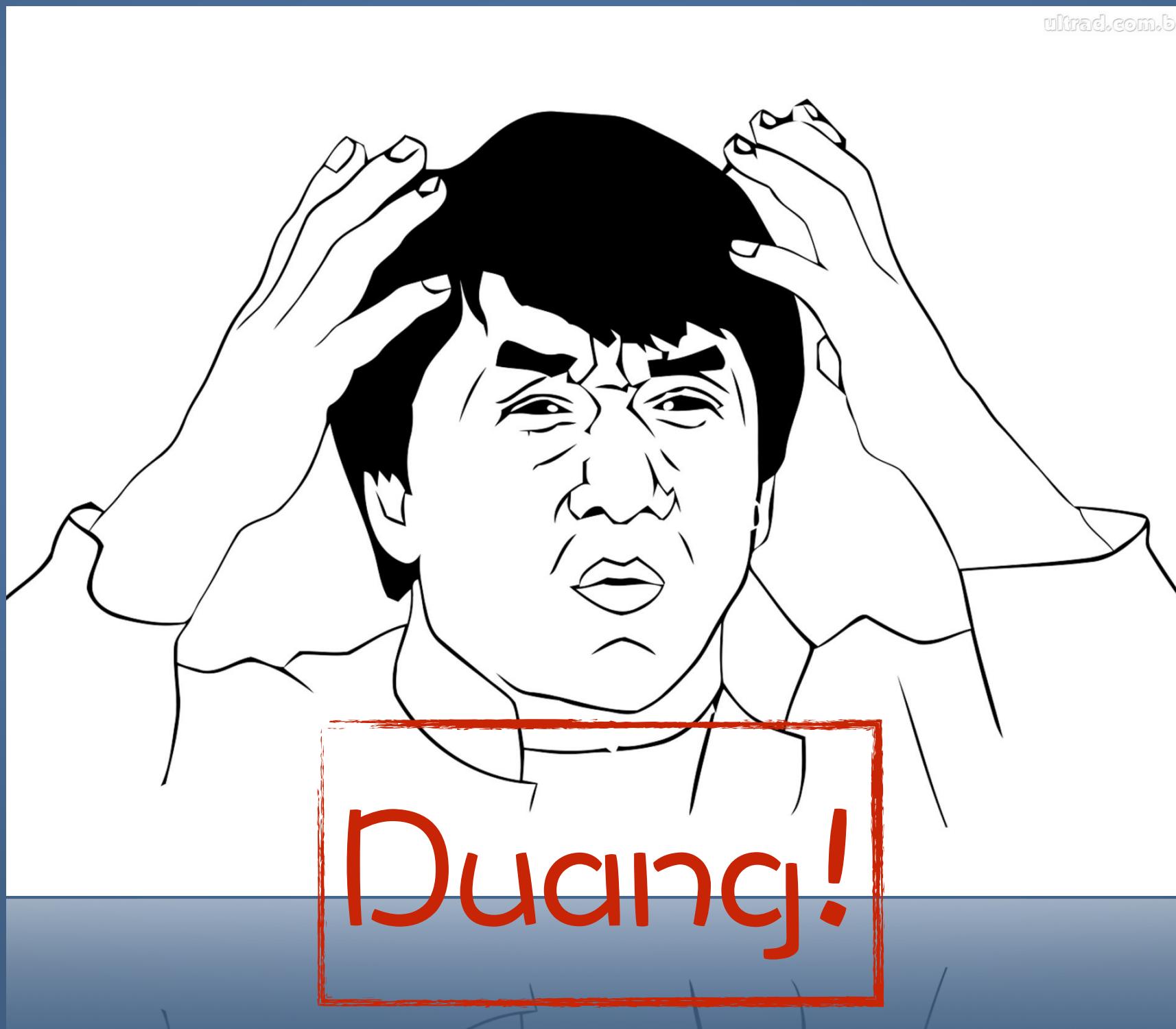
# 获得广告API的使用权限

<https://developers.facebook.com/docs/marketing-api/using-the-api>

# Facebook广告API现已开放



# 我们需要立刻升级到最高级么？



# 答案是：没有必要

- Developer级别已经足够开发使用
  - 广告账户的限制5个
  - 调用频率限制足够工具类使用
  - API调用频率限制对双方都是很好的保护措施
- 升级到Basic Access?
  - 开发阶段没有必要
  - 需要提交人工Review，提供清晰的描述与UI
- 升级到Standard Access
  - 需要提交申请等待审核
  - <https://www.facebook.com/business/standardadsapi>

# 准备条件

- 广告账户 (Ad Account)
- Facebook App
- Facebook个人账号
- 授权
  - Facebook个人账号至少成为广告账户的Advertiser, Facebook App的Developer
  - Facebook App关联广告账户

The screenshot shows the 'Roles' tab in the Facebook Developers interface for an app named 'PhpSDKTest'. It lists three categories: Administrators, Developers, and Testers. Under Administrators, there is one entry for 'YU LI' (Facebook). Under Developers, there is one entry for 'Minhua Lin' (University of York, UK). Under Testers, it is noted that 'This app has no Testers.'

The screenshot shows the 'Apps' section in the Facebook Business Manager for a page named 'Didiads'. It lists three associated Facebook apps: 'DiDiAds' (selected), 'DiDiAdsAPITest', and 'DiDiAdsMarketingAPITest'. For 'DiDiAds', it shows 'App ID: 334467960089822' and buttons for 'Assign Ad Accounts' and 'Assign Agency'. Below the app list, it shows 'Authorized Ad Accounts (1)' and 'Assigned Agencies (0)'. A user profile 'YU LI - Didiads' is also visible at the bottom.

# 立即尝试广告API

The screenshot shows the Facebook Graph API Explorer interface. At the top, there's a navigation bar with links for Developers, My Apps, Products, Docs, Tools & Support, and News. A search bar labeled "Search in Docs" is also present. Below the navigation bar, the "Graph API Explorer" section is visible. It includes fields for "Application" (set to "PhpSDKTest"), "Locale" (set to "English (US)"), "API Version" (set to "v2.4"), and an "Access Token" input field containing a long string of characters. Two specific buttons are highlighted with red boxes: "Debug" and "Get Token". The main area displays a JSON response for the query `GET /v2.4/act_10206805680808922/insights`. The response shows various engagement metrics like likes, link clicks, mentions, photo views, posts, post likes, page engagements, and post engagements.

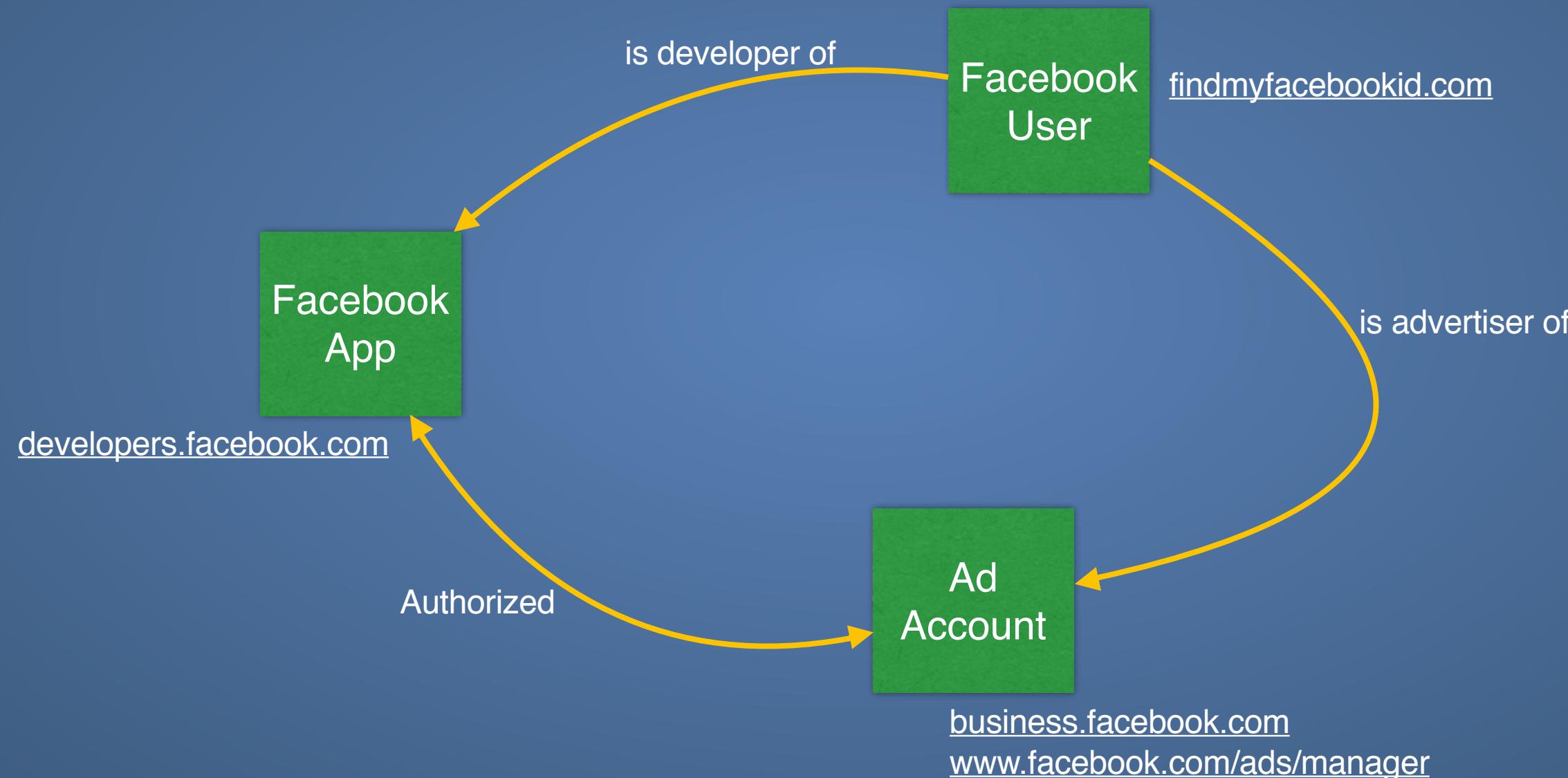
```
{  
  "data": [  
    {  
      "account_id": "10206805680808922",  
      "actions": [  
        {  
          "action_type": "like",  
          "value": 514  
        },  
        {  
          "action_type": "link_click",  
          "value": 68  
        },  
        {  
          "action_type": "mention",  
          "value": 1  
        },  
        {  
          "action_type": "photo_view",  
          "value": 174  
        },  
        {  
          "action_type": "post",  
          "value": 16  
        },  
        {  
          "action_type": "post_like",  
          "value": 189  
        },  
        {  
          "action_type": "page_engagement",  
          "value": 962  
        },  
        {  
          "action_type": "post_engagement",  
          "value": 447  
        }  
      ]  
    }  
  ]  
}
```

- 注意使用正确的FB账户
- 注意选择您的FB APP
- 注意Get Token时获取Ads Manage和 Ads Read权限

The screenshot shows the "Select Permissions" dialog box. It has two tabs: "User Data Permissions" and "Extended Permissions". Under "User Data Permissions", the checkbox for "ads\_management" is checked. Under "Extended Permissions", the checkboxes for "ads\_read" and "email" are checked. Other options like "manage\_notifications", "manage\_pages", "publish\_actions", etc., are unchecked. At the bottom, there's a note about "Public profile included by default." and three buttons: "Get Access Token", "Clear", and "Cancel".

# OAuth error? Can not find ad account error?

- 检查授权关系



# 访问API的途径



<https://developers.facebook.com/docs/marketing-api/reference>



<https://github.com/facebook/facebook-php-ads-sdk>



<https://github.com/facebook/facebook-python-ads-sdk>

# 广告API入门

<https://developers.facebook.com/docs/marketing-api/using-the-api>

欲速则不达

# Facebook Graph API使用入门

<https://developers.facebook.com/docs/graph-api/using-graph-api>

# Facebook Graph API的结构

- Graph API是基于HTTP/HTTPS协议的API
- 版本号 vX.X
  - <https://developers.facebook.com/docs/apps/changelog>
- 组成要素
  - 节点node: <http://graph.facebook.com/v2.3/<fbid>>
  - 边Edge: <http://graph.facebook.com/v2.3/<fbid>/feed>
  - 域Field: <http://graph.facebook.com/v2.3/<fbid>?fields=birthday>
- 返回值
  - JSON文档
  - 错误也是JSON文档形式

# Access Token

- <https://developers.facebook.com/docs/facebook-login/access-tokens>
- 每一次API访问都必须提交Access Token/User Access Token
- 怎样获得?
  - 测试? 使用 <https://developers.facebook.com/tools/explorer/> 是最快的办法
    - 注意选择正确的App
  - 程序使用
    - 通过Javascript SDK与Facebook SDK见 <https://developers.facebook.com/docs/facebook-login/access-tokens#usertokens>
    - 通过http request

<http://www.mywebsite.com>发起请求

[https://www.facebook.com/dialog/oauth?  
client\\_id=<appid>&redirect\\_uri=http://www.mywebsite.com/fbauth](https://www.facebook.com/dialog/oauth?client_id=<appid>&redirect_uri=http://www.mywebsite.com/fbauth)

客户端

从结果中获得code(as c), 发起请求

[https://graph.facebook.com/v<X.X>/oauth/access\\_token?  
client\\_id=<appid>&redirect\\_uri=http://www.mywebsite.com/  
fbauth&client\\_secret=<appsecret>&code=<c>](https://graph.facebook.com/v<X.X>/oauth/access_token?client_id=<appid>&redirect_uri=http://www.mywebsite.com/fbauth&client_secret=<appsecret>&code=<c>)

服务器端

从结果中获得access token

服务器端

## 案例：第1步，发起请求

```
var FBLogin = React.createClass({
  render: function() {
    return (
      <div>
        <ButtonToolbar>
          <Button bsStyle='primary' bsSize='large' onClick={this.login}>
            Login into your Facebook Account
          </Button>
        </ButtonToolbar>
      </div>
    );
  },
  login: function() {
    window.open('https://www.facebook.com/dialog/oauth?' +
      'client_id=' + this.props.appid +
      '&redirect_uri=' + window.location.origin + '/fblogin-cb',
      'FB Login', 'width=600,height=400');
  }
});
```

# 案例：第2-3步，获得Code后发起请求Access Token

```
@route('/fblogin-cb')
def fblogin_cb():
    if not request.params.code:
        return 'Ooops, do not find code in callback URL, please close this.'
    import urllib2
    url = fblogin_at_url_tpl.format(setting['appid'], fblogin_cb_url,
                                     setting['appsecret'], request.params.code)
    resp = urllib2.urlopen(url)
    respcontent = resp.read()
    with open('frontend/oauth_ok.tpl', 'r') as f:
        oauth_ok_page_tpl = f.read()
    return template(oauth_ok_page_tpl, respcontent=respcontent)
```

```
<!DOCTYPE html>
<html><head>
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
    <p>FB Auth OK, will go back soon ...</p>
    <script>
setTimeout(function() {
    window.opener.loginOk('{{!respcontent}}');
    window.close();
}, 1000);
</script>
</body></html>
```

## 案例：获得Access Token之后使用Graph API

```
curl -G -d  
"access_token=CAAPPhxzSarUcBAB42GF4y5qGMUygy2QNCGZAqMHWZBN0Y5  
SUYORDKgl4wTlVOnQwnW2jsSKAW8NN4npAhrkMJZAAf0IxWp4aIpzUOdBH7CX  
Qd8d2IMU2QufdCeG5I5sz3YZA7uqDK6YlxMRIs8Cu0cX2ZA09N1VH1MUUi6K  
S1NH1lzV3XiqKFCouZBeJA93418GPOxEaOu514oCk8Vd3mmoA" -s  
https://graph.facebook.com/v2.3/me?fields=id,name,adaccounts
```

# 使用广告API

<http://www.mywebsite.com>发起请求

`https://www.facebook.com/dialog/oauth?  
client_id=<appid>&redirect_uri=http://www.mywebsite.com/  
fbauth&scope=ads_management,ads_read`

客户端

从结果中获得code(as c), 发起请求

`https://graph.facebook.com/v<X.X>/oauth/access_token?  
client_id=<appid>&redirect_uri=http://www.mywebsite.com/  
fbauth&client_secret=<appsecret>&code=<c>`

服务器端

从结果中获得access token

服务器端

## 获得Access Token之后使用Ads API

```
curl -G -d  
"access_token=CAAPPhxzSarUcBAB42GF4y5qGMUygy2QNCGZAqMHWZBN0Y5SUYORDKg14  
wTlVOnQwnW2jsKAW8NN4npAhrkJZAAf0IxWp4aIpzUOdBH7CXQd8d2IMU2QufdCeG5I5s  
z3YZA7uqDK6YlxMRIs8Cu0cX2ZA09N1VH1MUUi6KS1NH1lzV3XiqKFCouZBeJA93418GPO  
xEaOu514oCk8Vd3mmoA" -s  
https://graph.facebook.com/v2.3/act\_10206805676768821
```

诀窍

<https://graph.facebook.com/v2.4/<id>?metadata=1>

# 怎样使用SDK

```
use FacebookAds\Api;
use FacebookAds\Object\AdAccount;
use FacebookAds\Object\AdAccountFields;

Api::init($app_id, $app_secret, $access_token);

$account = new AdAccount('act_<ACCOUNT_ID>');
$fields = array(
    AdAccountFields::NAME,
    AdAccountFields::BALANCE,
);

$params = array(
    AdCampaignFields::STATUS => array(AdCampaign::STATUS_PAUSED),
);

$account->read($fields);

// Output account name
echo $account->{AdAccountFields::NAME}.PHP_EOL;

// Output account balance
echo $account->{AdAccountFields::BALANCE}.PHP_EOL;
```



```
from facebookads.api import FacebookAdsApi
from facebookads.objects import AdAccount

FacebookAdsApi.init(
    app_id,
    app_secret,
    access_token
)

account = AdAccount('act_<ACCOUNT_ID>')

account.remote_read(fields=[
    AdAccount.Field.name,
    AdAccount.Field.balance
])
print(account[AdAccount.Field.name])
print(account[AdAccount.Field.balance])
```



# 创建广告四部曲

## 1.2.3.4

1. 创建campaign

<https://developers.facebook.com/docs/marketing-api/adcampaign>

2. 创建ad set

<https://developers.facebook.com/docs/marketing-api/adset>

3. 创建ad creative

<https://developers.facebook.com/docs/marketing-api/adcreative>

4. 创建ad

<https://developers.facebook.com/docs/marketing-api/adgroup>

# 步骤0: 上传image, 创建page post, etc

- 例如上传image

- <https://developers.facebook.com/docs/marketing-api/adimage>

```
curl \
  -F "test.zip=@test.zip" \
  -F "access_token=<ACCESS_TOKEN>" \
"https://graph.facebook.com/<API_VERSION>/act_<AD_ACCOUNT_ID>/adimages"
```

- 创建page post

- 多图广告 (Carousal Ads) 必须
  - <https://developers.facebook.com/docs/marketing-api/guides/carousel-ads/v2.3#pageFirst>

```
def create_campaign(self, accountid, name):
    campaign = AdCampaign(parent_id=accountid)
    campaign[AdCampaign.Field.name] = name
    campaign[AdCampaign.Field.objective] = (
        AdCampaign.Objective.mobile_app_engagement
    )
    campaign.remote_create()
    return campaign[AdCampaign.Field.id]
```

```
def create_ad_set(self, accountid, name, dailybudget, campaignid,
                  bidinfo, targeting, appinfo):
    logger.debug('bidinfo: ' + str(bidinfo))
    pdata = {
        AdSet.Field.name: name,
        AdSet.Field.bid_type: AdSet.BidType.cpc,
        AdSet.Field.bid_info: bidinfo,
        AdSet.Field.status: AdSet.Status.active,
        AdSet.Field.daily_budget: dailybudget,
        AdSet.Field.campaign_group_id: campaignid,
        AdSet.Field.promoted_object: {
            'application_id': appinfo['fbapplication_id'],
            'object_store_url': appinfo['appstore_link']
        },
        AdSet.Field.targeting: targeting,
    }
    adset = AdSet(parent_id=accountid)
    adset.remote_create(params=pdata)
    return adset[AdSet.Field.id]
```

```
def create_creative(self, accountid, name, imagehash, message, appinfo):
    link_data = LinkData()
    link_data[LinkData.Field.link] = appinfo['appstore_link']
    link_data[LinkData.Field.message] = message
    link_data[LinkData.Field.image_hash] = imagehash
    call_to_action = {'type': 'USE_MOBILE_APP'}
    call_to_action['value'] = {
        'link': appinfo['appstore_link'],
        'app_link': appinfo['app_deep_link'],
        'application': appinfo['fbapplication_id'],
        'link_title': appinfo['app_name']
    }
    link_data[LinkData.Field.call_to_action] = call_to_action
    object_story_spec = ObjectStorySpec()
    object_story_spec[ObjectStorySpec.Field.page_id] = appinfo['fbpage_id']
    object_story_spec[ObjectStorySpec.Field.link_data] = link_data
    creative = AdCreative(parent_id=accountid)
    creative[AdCreative.Field.name] = name
    creative[AdCreative.Field.object_story_spec] = object_story_spec
    creative.remote_create()
    return creative[AdCreative.Field.id]
```

```

def create_ad(self, accountid, name, adsetid, creativeid, appinfo):
    adgroup = AdGroup(parent_id=accountid)
    adgroup[AdGroup.Field.name] = name
    adgroup[AdGroup.Field.campaign_id] = adsetid
    adgroup[AdGroup.Field.creative] = {'creative_id': str(creativeid)}
    adgroup[AdGroup.Field.objective] = \
        AdGroup.Objective.mobile_app_engagement
    tracking_specs = [
        {'action.type': ['mobile_app_install'],
         'application': [appinfo['fbapplication_id']]},
        {'action.type': ['app_custom_event'],
         'application': [appinfo['fbapplication_id']]}
    ]
    if not (appinfo['fboffsitepixel_id'].strip() == 'null'):
        tracking_specs.append({
            'action.type': ['offsite_conversion'],
            'offsite_pixel': [appinfo['fboffsitepixel_id']]})
    adgroup[AdGroup.Field.tracking_specs] = tracking_specs
    adgroup[AdGroup.Field.status] = AdGroup.Status.active
    adgroup.remote_create()
return adgroup[AdGroup.Field.id]

```

# object\_story\_spec

<https://developers.facebook.com/docs/marketing-api/adcreative>

object_story_spec	object	The page id and the content to create a new unpublished page post specified using one of <a href="#">link_data</a> , <a href="#">photo_data</a> , <a href="#">video_data</a> , <a href="#">offer_data</a> , <a href="#">text_data</a> or <a href="#">template_data</a> .
page_id	int	ID of a Facebook page. An unpublished page post will be created on this page. User must have <a href="#">Admin</a> or <a href="#">Editor</a> role for this page.
link_data	object	Content for creating a link page post or a multi product ad
photo_data	object	Content for creating a photo page post.
video_data	object	Content for creating a video page post.
offer_data	object	Content for creating an offer page post
text_data	object	Content for creating a status page post.
template_data	object	Content for creating a dynamic ad template creative

# bid\_info

[https://  
developers.facebook.com/  
docs/marketing-api/adset](https://developers.facebook.com/docs/marketing-api/adset)

bid_info	JSON object	A dictionary of {objective}: {value} that you place on your bid, based on the <b>bid_type</b> . Values are defined in your currency's minimum denomination: For CPM, use <code>{'IMPRESSIONS': &lt;value&gt;}</code> For CPC, use <code>{'CLICKS': &lt;value&gt;}</code> For ABSOLUTE_OCPM, use <code>{'ACTIONS':&lt;value&gt;, 'REACH':&lt;value&gt;, 'CLICKS':&lt;value&gt;, 'SOCIAL':&lt;value&gt;}</code> For CPA, use <code>{'ACTIONS': &lt;value&gt;}</code> Impressions min value: 1c Reach min value: 2c Clicks min value: 1c Action min value: 1c Social min value: 1c	Yes if <b>is_autobid</b> not true
bid_type	string	The bid type of the set. All ad groups of this set use this bid type once it is set. Possible values: CPC CPM MULTI_PREMIUM ABSOLUTE_OCPM CPA	Yes

# 定义受众

# Targeting Specs

<https://developers.facebook.com/docs/marketing-api/targeting-specs>

# Facebook广告的特色功能：Targeting

- 不同于关键字广告，Facebook广告使用Targeting来定义受众
  - 是人的行为而不是关键词
- Facebook Targeting的能力
  - <https://developers.facebook.com/docs/marketing-api/targeting>
  - 人口学 Demographic
  - 用户兴趣 Interest
  - 用户行为 Behaviour
  - 社交联系 Connection

# Targeting Spec

- <https://developers.facebook.com/docs/marketing-api/targeting-specs>
- JSON格式

```
curl \
-X POST \
-F "name=My AdSet" \
-F "bid_type=CPC" \
-F "bid_info={'CLICKS': 150}" \
-F "campaign_status=ACTIVE" \
-F "daily_budget=2000" \
-F "campaign_group_id=<CAMPAIGN_GROUP_ID>" \
-F "targeting= { \
    'geo_locations': { \
        'countries':[ 'US' ], \
    }, \
    'behaviors':[ \
        {'id':6004386044572,'name':'Android Owners (All)'}, \
        {'id':6007101597783,'name':'Business Travelers'}, \
    ], \
}" \
-F "access_token=<ACCESS_TOKEN>" \
"https://graph.facebook.com/<API_VERSION>/act_<AD_ACCOUNT_ID>/adcampaigns"
```

# 怎样熟悉Targeting Spec?

- 练习，多加练习
- 创建广告时使用 adgroup\_status = PAUSED
- 使用reach estimate
  - <https://developers.facebook.com/docs/marketing-api/reachestimate>

```
curl -G \
-d "currency=USD" \
-d "targeting_spec=____" \
-d "access_token=____" \
"https://graph.facebook.com/<API_VERSION>/act_<AD_ACCOUNT_ID>/reachestimate"
```
- 广告预览
  - <https://developers.facebook.com/docs/marketing-api/ad-preview>
  - <https://developers.facebook.com/docs/marketing-api/ad-preview-plugin>

# 案例分析

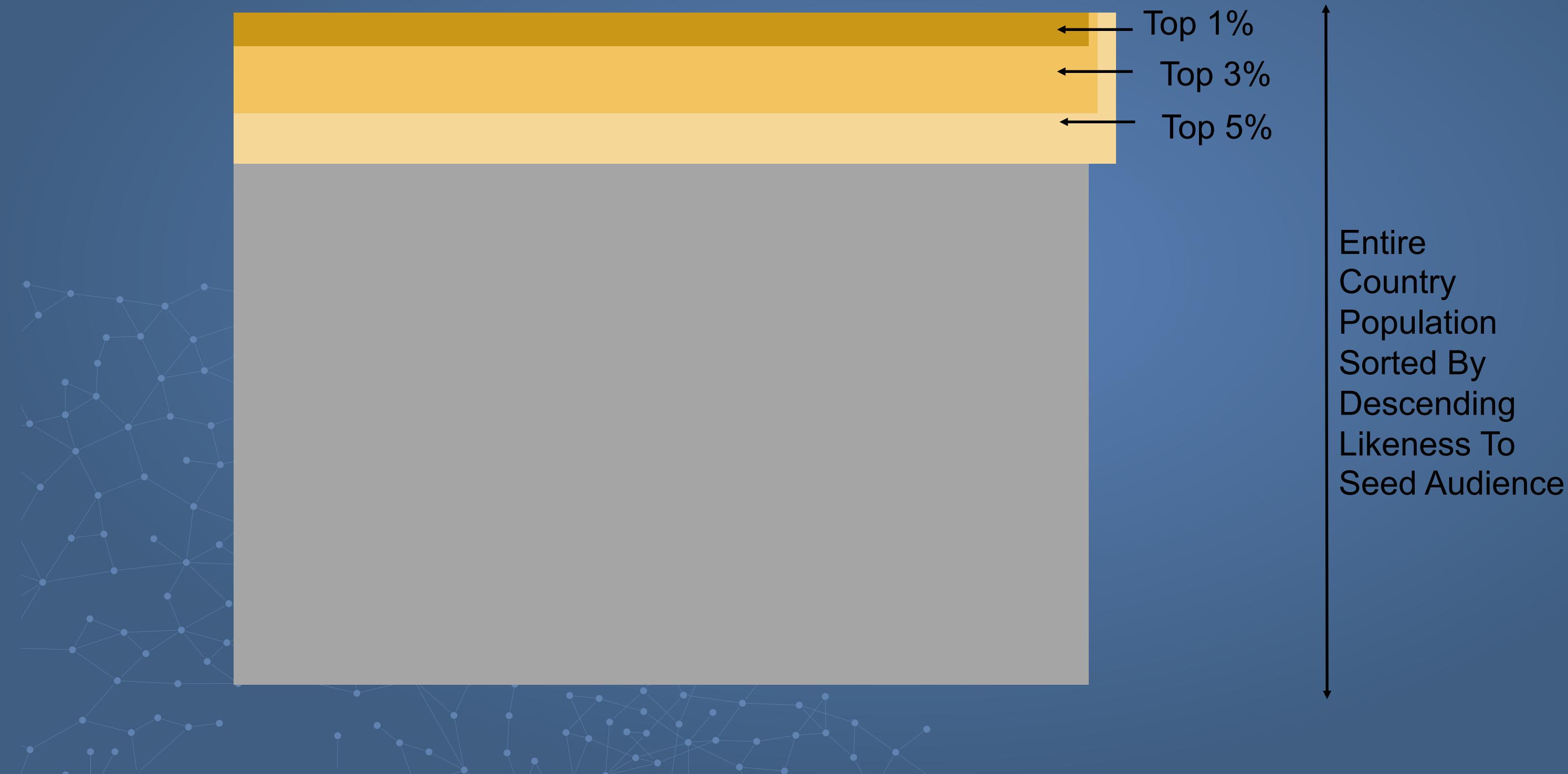
<http://a.url.can/go?here>



Scale While Maintaining CPI

# Increasing Reach While Maintaining CPI/ROI

## Tiered Lookalike Strategy



# Manual Steps

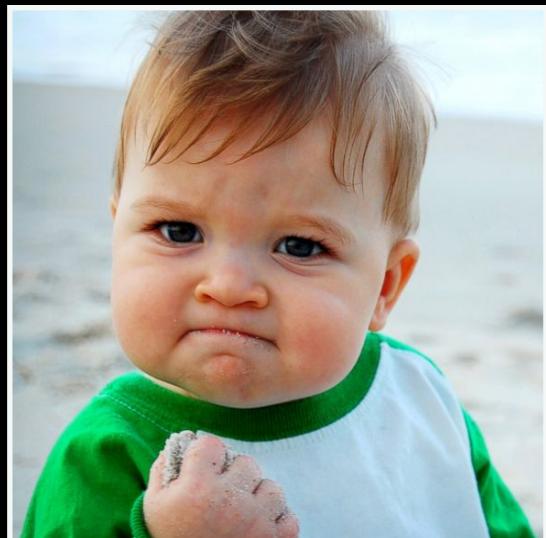
1. Create a custom audience
2. Create 1% lookalike
3. Repeat for 3% and 5%
4. Create adset using 1% lookalike
5. Create adset using 3% lookalike, exclude 1% lookalike, bid lower
6. Create adset using 5% lookalike, exclude 3% lookalike, bid lower
7. Don't forget cross-device tracking!
8. Repeat 1- 7 for other seed audiences
9. Repeat 1- 8 for other countries

# Create Tired LAL

```
for tier in range(1, tiers + 1):
    lookalike = CustomAudience(parent_id=accountid)
    lookalike[LookalikeAudience.Field.name] = \
        '{0} LAL {1}'.format(name, tier)
    lookalike[LookalikeAudience.Field.origin_audience_id] = seed_id
    lookalike[LookalikeAudience.Field.lookalike_spec] = {
        'ratio': '{:.2f}'.format(tier / 100.0),
        LookalikeAudience.Field.LookalikeSpec.country: country,
    }
    lookalike.remote_create()
```

# Create Ad Set

```
for tier in range(1, tiers + 1):
    ad_set = AdSet(parent_id=accountid)
    ad_set[AdSet.Field.bid_info] = tiered_bidinfos[tier - 1]
    audience = tiered_lookalikes[tier - 1]
    exclude_audience = None
    if tier != 1:
        exclude_audience = tiered_lookalikes[tier - 2]
    targeting = {
        TargetingSpecsField.custom_audiences: [
            {
                'id': audience[CustomAudience.Field.id],
                'name': audience[CustomAudience.Field.name],
            }
        ]
    }
    if exclude_audience:
        targeting[TargetingSpecsField.excluded_custom_audiences] = [
            {
                'id': exclude_audience[CustomAudience.Field.id],
                'name': exclude_audience[CustomAudience.Field.name],
            }
        ]
    ad_set[AdSet.Field.targeting] = targeting
    ad_set.remote_create()
```





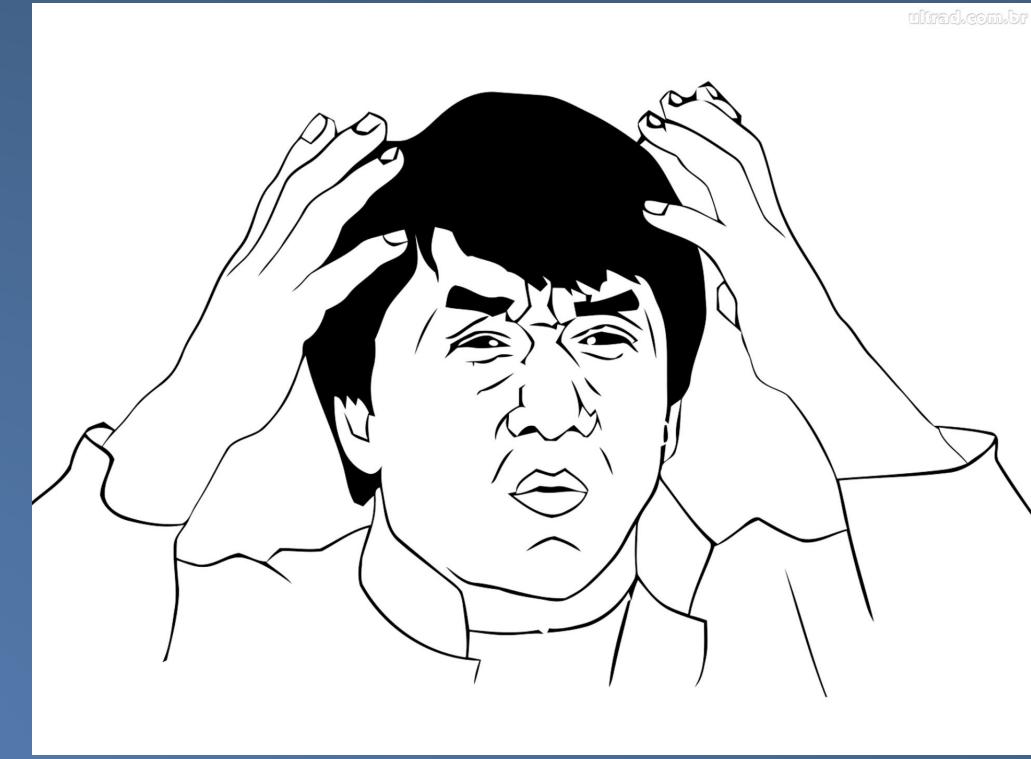
# Pulling Ad Insights Into Database/Excel

# Manual Steps

1. Open saved report in Facebook ads reports
2. Refresh data
3. Export to file
4. Open Excel
5. Import file
6. Save spreadsheet
7. Repeat daily
8. Everyone does this!

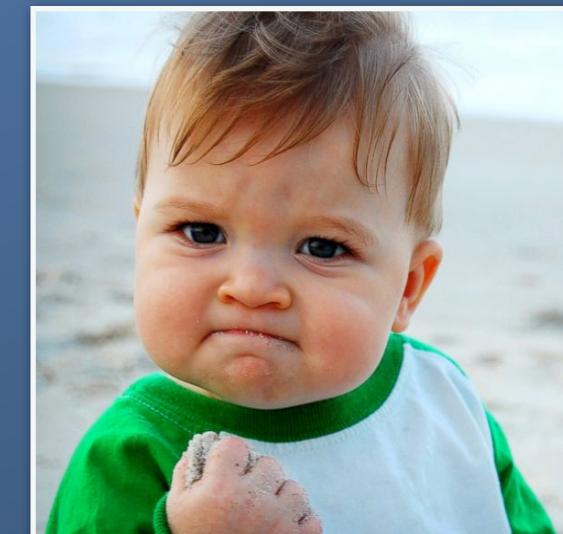
# Not finish yet

- Big team will have hierarchy reporting
  - Each team member takes care of his own
  - Someone takes care of summary
- This can go up
  - Different team in different product category takes care of its summary
  - Someone takes care of teams' summary
- Then what if any one got the number wrong? (typo, copy/paste missing)



# A Better Pipeline

1. ~~Open saved report in Facebook ads reports~~ One command to retrieve insights to mysql
2. ~~Refresh data~~
3. ~~Export to file~~
4. Open Excel
5. ~~Import file~~
6. ~~Save spreadsheet~~ Hit refresh button to refresh from mysql
7. Repeat daily
8. ~~Everyone~~ Build a robot to does this!



```

##### main #####
# get all ad campaign from an ad account
if not db_unix_socket == '':
    con = mdb.connect(host='localhost', user=db_username,
                      db=db_name, unix_socket=db_unix_socket, charset='utf8')
else:
    con = mdb.connect(host=db_host, port=db_port,
                      user=db_username, passwd=db_password,
                      db=db_name, charset='utf8')

with con:
    deleteAdSetsInsight(con, report_date)
    for my_ad_account_id in my_ad_account_ids:
        ad_account = AdAccount(my_ad_account_id)
        fields = [
            'campaign_group_name',
            'campaign_name',
            'campaign_id',
            'impressions',
            'clicks',
            'spend',
            'reach',
            'actions',
            'action_values'
        ]
        params = {
            'time_range': {
                'since': report_date,
                'until': report_date
            },
            'action_attribution_windows': ['28d_click'],
            'breakdowns': ['impression_device', 'placement'],
            'level': 'campaign',
            'limit': max_records
        }
        ad_insights = ad_account.get_insights(fields, params)
        print('Number of insight records:' + str(len(ad_insights)))
        count = 0
        for idx in range(0,min(max_records,len(ad_insights))):
            ad_insight = ad_insights[idx]
            print('Writing record number: ' + str(idx))
            writeAdInsight(ad_insight, con, report_date)

```

```

def writeAdInsight(ad_insight, con, report_date):
    impression_device = 0
    if 'impression_device' in ad_insight:
        impression_device = ad_insight['impression_device']

    action_device = 0
    if 'action_device' in ad_insight:
        action_device = ad_insight['action_device']

    key_value = {
        'start_date': report_date,
        'end_date': report_date,
        'campaign': ad_insight['campaign_group_name'],
        'adset': ad_insight['campaign_name'],
        'adset_id': ad_insight['campaign_id'],
        'impressions': ad_insight['impressions'],
        'clicks': ad_insight['clicks'],
        'spend': ad_insight['spend'],
        'reach': ad_insight['reach'],
        'impression_device': impression_device,
        #'conversion_device': action_device
    }
    if 'actions' in ad_insight:
        actions = ad_insight['actions']
        for action in actions:
            t = action['action_type']
            if t in action_type_columns:
                key_value[action_type_columns[t]] = action['28d_click']

    if 'action_values' in ad_insight:
        action_values = ad_insight['action_values']
        for action_value in action_values:
            t = action['action_type']
            if t in action_value_columns:
                key_value[action_value_columns[t]] = action['28d_click']

    key_str = ""
    value_str = ""
    count = 0
    for (key, value) in key_value.items():
        if count > 0:
            key_str += ", "
            value_str += ", "
        key_str += key
        value_str += "\\" + unicode(value) + "\\"
        count += 1

    stat = "INSERT INTO ad_set_insight (" + key_str + ") VALUES (" + value_str + ")";
    print stat

    cur = con.cursor()
    cur.execute(stat)

```

# SOLUTIONS+ENGINEERING

<https://developers.facebook.com/docs/marketing-api>