Developer API

The Developer API gives access to the metadata available for all families served by Google Fonts. It allows the creation of dynamic apps by being able to query Google Fonts and get an accurate list of the families currently available. The REST API gives access to the data in the JSON format and includes for each family a list of the styles available as well as a list of scripts (called subsets in Google Fonts) supported. The API gives an option to sort the list of families alphabetically, by date added, by number of styles, by trend or by popularity. More data might be added over time.

Audience

This document is intended for web and application developers; using the Developer API requires knowledge of JavaScript.

A quick example

To retrieve the dynamic list of fonts offered by the Google Fonts service, send the following request:

```
https://www.googleapis.com/webfonts/v1/webfonts?key=YOUR-API-KEY
```

A sample of the result would look like this:

```
{
 "kind": "webfonts#webfontList",
 "items": [



[...]



{
   "kind": "webfonts#webfont",
   "family": "Anonymous Pro",
   "variants": [
    "regular",
    "italic",
    "700",
    "700italic"
   ],
   "subsets": [
    "greek",
    "greek-ext",
    "cyrillic-ext",
    "latin-ext",
    "latin",
    "cyrillic"
   ],
   "version": "v3",
   "lastModified": "2012-07-25",
   "files": {
    "regular": "http://themes.googleusercontent.com/static/fonts/anonymouspro
    "italic": "http://themes.googleusercontent.com/static/fonts/anonymouspro/
    "700": "http://themes.googleusercontent.com/static/fonts/anonymouspro/v3/
    "700italic": "http://themes.googleusercontent.com/static/fonts/anonymousp
   }
  },
  {
   "kind": "webfonts#webfont",
   "family": "Antic",
   "variants": [
    "regular"
   ],
   "subsets": [
    "latin"
   ],
   "version": "v4",
   "lastModified": "2012-07-25",
   "files": {
    "regular": "http://themes.googleusercontent.com/static/fonts/antic/v4/hEa
   }
  },


[...]


]
}
```

Identifying your application to Google

Your application needs to identify itself every time it sends a request to the Google Fonts

Developer API, by including an API key (https://developers.google.com/console/help/generating-dev-keys) with each request.

Acquiring and using an API key

To acquire an API key:

1. Open the Credentials page (https://console.developers.google.com/apis/credentials?project=_).
2. Click **Add credentials > API key** and select the appropriate key type:

Server keys

Create and use a **server key** if your application runs on a server. Do not use this key outside of your server code. For example, do not embed it in a web page. To prevent quota theft, restrict your key so that requests are only allowed from your servers' source IP addresses.

Browser keys

Create and use a **browser key** if your application runs on a client, such as a web browser. To prevent your key from being used on unauthorized sites, only allow referrals from domains you administer.

iOS keys

Create and use an **iOS key** if your application runs on iOS devices. Google verifies that each request originates from an iOS application that matches one of the bundle identifiers you specify. An app's `.plist` file contains its bundle identifier. Example: `com.example.MyApp`

Android keys

Create and use an **Android key** if your application runs on Android devices. To do so, you need to specify the SHA1 fingerprints and package names of the application using that key.

a. In the **Package name** field, enter your Android app's package name

(//developer.android.com/guide/topics/manifest/manifest-element.html#package)

.

b. In a terminal, run the Keytool utility (//developer.android.com/studio/publish/app-signing.html#signing-manually) to get the SHA1 fingerprint for your digitally signed `.apk` file's public certificate.

```
keytool -exportcert -alias androiddebugkey -keystore path-to-debug-c
```

Note: For the `debug.keystore`, the password is `android`. For Eclipse, the debug keystore is typically located at `~/.android/debug.keystore`.

The Keytool prints the fingerprint to the shell. For example:

```
$ keytool -exportcert -alias androiddebugkey -keystore ~/.android/de
Enter keystore password: Type "android" if using debug.keystore
Alias name: androiddebugkey
Creation date: Aug 27, 2012
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=Android Debug, O=Android, C=US
Issuer: CN=Android Debug, O=Android, C=US
Serial number: 503bd581
Valid from: Mon Aug 27 13:16:01 PDT 2012 until: Wed Aug 20 13:16:01
Certificate fingerprints:
   MD5:  1B:2B:2D:37:E1:CE:06:8B:A0:F0:73:05:3C:A3:63:DD
   SHA1: D8:AA:43:97:59:EE:C5:95:26:6A:07:EE:1C:37:8E:F4:F0:C8:05:C8
   SHA256: F3:6F:98:51:9A:DF:C3:15:4E:48:4B:0F:91:E3:3C:6A:A0:97:DC:
   Signature algorithm name: SHA1withRSA
   Version: 3
```

Copy the SHA1 fingerprint, which is highlighted in the example above.

Important: When you prepare to release your app to your users, follow these steps again and create a new OAuth 2.0 client ID for your production app. For production apps, use your own private key to sign the production app's `.apk` file. For more information, see Signing your applications (//developer.android.com/tools/publishing/app-signing.html).

c. Paste the SHA1 fingerprint into the form where requested.

d. Click **Create**.

To keep your API keys secure, follow the best practices for securely using API keys (https://developers.google.com/console/help/api-key-best-practices).

After you have an API key, your application can append the query parameter `key=yourAPIKey` to all request URLs.

The API key is safe for embedding in URLs; it doesn't need any encoding.

Details

The JSON response (refer to sample above) is composed of an array named "items" which contains objects with information about each font family.

A family object is composed of 4 fields:

- kind: The kind of object, a webfont object
- family: The name of the family
- subsets: A list of scripts supported by the family
- variants: The different styles available for the family
- version: The font family version.
- lastModified: The date (format "yyyy-MM-dd") the font family was modified for the last time.
- files: The font family files (with all supported scripts) for each one of the available variants.

By combining the information for each family it is easy to create a Fonts API request. For example assuming we have a reference to the family object for Anonymous Pro:

```
[...]

var apiUrl = [];
apiUrl.push('//fonts.googleapis.com/css?family=');
apiUrl.push(anonymousPro.family.replace(/ /g, '+'));
if (contains('italic', anonymousPro.variants)) {
  apiUrl.push(':');
  apiUrl.push('italic');
}
if (contains('greek', anonymousPro.subsets)) {
  apiUrl.push('&subset=');
  apiUrl.push('greek');
}
```

```
// url: '//fonts.googleapis.com/css?family=Anonymous+Pro:italic&subset=greek'
var url = apiUrl.join('');

[...]
```

The list of families is returned in no particular order by default. It is possible however to sort the list using the sort parameter:

```
https://www.googleapis.com/webfonts/v1/webfonts?sort=popularity
```

The possible sorting values are:

- alpha: Sort the list alphabetically
- date: Sort the list by date added (most recent font added or updated first)
- popularity: Sort the list by popularity (most popular family first)
- style: Sort the list by number of styles available (family with most styles first)
- trending: Sort the list by families seeing growth in usage (family seeing the most growth first)

Last updated June 14, 2016.