

# SoundCloud Map

A Web 2.0 Mashup with  
SoundCloud and Google Maps



SoundCloud Map and this document have been created by Johan Uhle during the seminar „Webprogrammierung / Web 2.0 Technologien“ under the supervision of Christian Willems in the summer term 2009 at the Hasso Plattner Institute in Potsdam / Germany.

A **live version** of SoundCloud Map can be found here

<http://soundcloudmap.appspot.com>

The **source code** has been released under Apache 2 License on Github here

<http://github.com/freenerd/SoundCloud-Map/>

**Hasso-Plattner-Institute**

<http://www.hpi-web.de>

**Johan Uhle**

<http://www.freenerd.de>

<http://www.twitter.com/freenerd>

*SoundCloud Map has not been affiliated neither with SoundCloud Ltd. nor Google Inc.*

*SoundCloud is a trademark of Soundcloud Ltd. / Berlin - Germany*

*Google Maps and Google App Engine are trademarks of Google Inc. / Mountain View - USA*

# Introduction

SoundCloud Map displays the latest music uploaded to SoundCloud<sup>1</sup> on a Google Maps<sup>2</sup> map. It features track playback, genre-based filtering and an easy-to-use user interface.

The target audience are Soundcloud users as well as non-users. SoundCloud Map can be used to get a new and playful perspective on the tracks uploaded to SoundCloud. In addition it shows that SoundCloud is used frequently and world-wide.

SoundCloud Map has been implemented by using Python<sup>3</sup> on Google App Engine<sup>4</sup>. The front-end uses jQuery<sup>5</sup> as well as the SoundCloud Javascript Player<sup>6</sup> and Soundmanager 2<sup>7</sup>.

## SoundCloud and Google Maps

SoundCloud is called to be "Flickr for Music"<sup>8</sup>. It's an online platform for audio professionals to share music, both in public with fans or in private with labels, collaborators or distributors. Various audio file formats are supported and there is no file size limit. SoundCloud is a so-called "Freemium" service. There are free accounts available, that have certain limits regarding available features and number of tracks. For more professional usage three levels of pro accounts are to be purchased. SoundCloud offers a powerful free-to-use API<sup>9</sup>.

Google Maps is a web service by Google providing world-wide map information. The maps feature cartographical and satellite view as well as an easy-to-use user interface. It's possible to embed Google Maps in web pages via the Google Maps API<sup>10</sup>. Custom content can be layered on top of the embedded map. The usage of Google Maps within certain quota<sup>11</sup> is free of charge.

---

<sup>1</sup> <http://www.soundcloud.com>

<sup>2</sup> <http://maps.google.com>

<sup>3</sup> <http://www.python.org>

<sup>4</sup> <http://code.google.com/appengine/>

<sup>5</sup> <http://www.jquery.com/>

<sup>6</sup> <http://wiki.github.com/soundcloud/api/javascript-player>

<sup>7</sup> <http://www.schillmania.com/projects/soundmanager2/>

<sup>8</sup> <http://soundcloud.com/press/releases/2008-12-15-soundcloud-midem.pdf>

<sup>9</sup> <http://www.soundcloud.com/api>

<sup>10</sup> <http://code.google.com/apis/maps/>

<sup>11</sup> <http://www.google.com/enterprise/maps/>

# Functionality

SoundCloud Map was designed to behave like a desktop application in order to supply an incessant and rich user-experience.

When the user opens SoundCloud Map, a window-filling map is displayed. Several markers appear on the map, each representing a track uploaded to SoundCloud. No more than 100 markers are displayed on the map to assure performance even on slow systems. The map features the easy navigation provided by Google Maps. Users are able to move the map by dragging, zoom in and out by using the zoom tool or switch to map, satellite and hybrid view.

By clicking on a marker a "bubble" is opened revealing information about the track such as track title, artist name, artist picture and creation time. Furthermore a link is provided for playing the track. On click a player is opened at the bottom of the screen and the track starts to play automatically. The player provides artist name, track name, link to the track on SoundCloud, play/pause control, loading information, time information and a waveform. A click on the waveform starts playback at the particular track position. During track playback a user may continue navigating the map and open new bubbles without interrupting the playing track.

SoundCloud Map also offers a rudimentary genre filtering accessible in the upper left by the four links "All", "Rock", "Techno" and "House". When clicked, all markers currently displayed on the map are removed and new markers are added according to the selected filter. Since this is done asynchronously, the map does not need to be reloaded and track playback is not interrupted.

Compatibility to modern browsers has been insured by the use of jQuery. It has been tested with Firefox 3.0, Safari 4.0, Opera 9.64 and Internet Explorer 8.0.

# Realisation

The application is basically split into two parts. The back-end fetches the latest tracks from SoundCloud and writes them to a cache. The front-end takes the tracks from the cache and displays them on the map.

## The back-end

The back-end routine is called on a regular schedule. At the moment it is executed every three minutes. The routine queries the SoundCloud API to fetch all new tracks having been uploaded within the last three minutes. The returned tracks are filtered for streamability. For every track the SoundCloud API is queried again to fetch the complete user information which also includes the user-entered city and country. This information is used to geocode the track via the Google Maps API<sup>12</sup>. After that every geocoded track is written to the cache.

---

<sup>12</sup> <http://code.google.com/intl/en-EN/apis/maps/documentation/geocoding/>

As a result the cache is never more than three minutes late and therefore up-to-date with SoundCloud. Another scheduled script deletes tracks older than 24 hours since we are only interested in recent tracks.

## The front-end

When SoundCloud Map is opened the first object to be loaded is Google Maps. After the map is completely loaded an asynchronous call to a script within SoundCloud Map is made. This script returns the latest tracks from the cache formatted as JSON<sup>13</sup>. For every received track, a marker on the map and a corresponding bubble are created. An event is attached to every marker to open the bubble on click. Every bubble includes a "Click to play"-link having an event attached that is loading the player.

The player has been realised with the help of the SoundCloud Javascript Player, which is using SoundManager 2 for audio playback. SoundManager 2 uses Adobe Flash<sup>14</sup> to play the music but all the controls are accessible via Javascript thus making it easy to integrate.

Before loading a new player old players are removed. Next the new player is created, outfitted with the right waveform url and a permalink to identify the track. The track information is fetched from the SoundCloud API and the audio stream is loaded from SoundCloud. When pre-caching has been completed the track automatically starts to play.

## Reasons for the separation between front-end and back-end

The alternative to the given separation would have been to fetch the tracks in the users browser via Javascript. To display the latest 100 tracks, 2 queries to the SoundCloud API for the 100 latest tracks<sup>15</sup>, 100 queries to the SoundCloud API for complete user data and 100 queries to the Google Maps API for geocoding would have been needed, resulting in a total 202 queries on every start-up of SoundCloud Map. Experience shows, that querying remote APIs is slow and unstable. The total loading time would dramatically increase resulting in a reduced user experience. Furthermore this approach may not scale well by causing high traffic on the SoundCloud API and Google Maps API. Another downside is that filtering would only be possible over the loaded tracks in the browser and not over the latest tracks of the last 24 hours.

# Problems

## Updating the cache without exceeding the runtime limit

Updating the cache is a fairly easy-to-program task. The problem lies in making it robust and fast.

Robustness is a problem, because the update relies on remote systems which may behave unpredictable. Therefore error handling and consistency checks are to be implemented to cover the behaviour of the remote system but still keeping the cache update working.

---

<sup>13</sup> <http://www.json.org/>

<sup>14</sup> <http://www.adobe.com/products/flashplayer/>

<sup>15</sup> The SoundCloud API only returns 50 tracks per query, so 2 queries are necessary to fetch 100 tracks

The logical limit for a cache update task is to be finished before the next cache update starts. But Google App Engine imposes a 30 second runtime limit on every request and this also applies to scheduled tasks<sup>16</sup>. To cope with this behaviour the cache update has been streamlined to work as fast as possible. A major improvement has been achieved by not getting the latest tracks in general<sup>17</sup> from SoundCloud but to fetch only to latest tracks since the last cache update<sup>18</sup>. This lead to a significant speed improvement (peak API request runtime from 6 seconds to under 1 second). The cache update flow has been optimised to reduce API calls by discarding all tracks that do not meet certain restrictions (not streamable, no location data) as early as possible to not cause useless API calls.

## Geocoding

Geocoding is done via the Google Maps Geocode API which limits requests by applying quotas. Google Maps sets these quota per-ip and not per-app. Since Google App Engine uses Proxies for fetching remote urls, all requests share only a limited number of IPs. Thereby the quota is reached on Google Maps side every evening. This problem has been solved by the Google Maps team by raising the quota for the App Engine Proxies<sup>19</sup>. A change to per-app-key quota may happen in later versions of the Google Maps API.

## Abstracting data

The obvious way to pass data from the cache to the front-end in order to display it to the user is looping over in array while rendering the template. This is easy to do and obvious, but does not separate data from design enough. In the end, a different approach has been taken. The cached tracks are provided JSON-formatted by a dedicated script and loaded dynamically by the browser. The disadvantage of this approach is increased complexity and decreased speed, both on client side (one extra url to be loaded and computed) and server side (one extra script to be computed), but these are compensated by the advantages of a cleaner and better maintainable code as well as added flexibility. The implemented routine allows adding new markers without having to reload the main website. At the moment this is also used for filtering tracks by genre. More filters like "show tracks of a certain time period" or "only show tracks with an artwork" could easily be implemented as well. Another functionality using this routine for is updating the front-end with the most recent tracks from the back-end, so that the also the application already loaded in the browser stays up-to-date with the latest tracks.

# Outlook

## Improving the player

Several aspects about the player are to be improved. In the current version every time a new track is loaded, the player is replaced by a new one, but the corresponding SoundManager Javascript object remains existent resulting in an ever-growing memory demand. It would be better to keep only one SoundManager object and just replace the

---

<sup>16</sup> [http://code.google.com/intl/de-DE/appengine/docs/python/runtime.html#The\\_Request\\_Timer](http://code.google.com/intl/de-DE/appengine/docs/python/runtime.html#The_Request_Timer)

<sup>17</sup> <http://api.soundcloud.com/tracks.json?latest>

<sup>18</sup> [http://api.soundcloud.com/tracks.json?created\\_at\[from\]=2009-06-29T01:12:52.702786](http://api.soundcloud.com/tracks.json?created_at[from]=2009-06-29T01:12:52.702786)

<sup>19</sup> [http://groups.google.com/group/Google-Maps-API/browse\\_thread/thread/5b2382ee1324e9f6/](http://groups.google.com/group/Google-Maps-API/browse_thread/thread/5b2382ee1324e9f6/)

stream url. Another improvement would be adding a volume control. The user experience would also benefit from an „auto-play next track“-feature. After a song has finished playing, another song should be randomly chosen from the map and be played.

All these features are available in SoundManager 2 but have to be implemented in the Soundcloud Javascript Player and the front-end.

### **Track list**

An additional scrollable track list on the side displaying the tracks in descending creation time order would allow a different perspective on the tracks. This list could be coupled with the map. Clicking a track in the list would result in showing the track on the map and vice versa.

### **Multiple tracks per bubble**

At the moment there is only one track per marker position possible. If one position does have multiple tracks to it, only the most recent track is displayed. Especially big cities do have a large variety of tracks that should be made accessible therewith enabling the user to listen to the "sound of a city".

### **Improving the user interface and the design**

There has been no effort put into the graphical design yet. As a result SoundCloud Map is not appealing to users. Furthermore it does not match the SoundCloud corporate design and does not promote the connection to SoundCloud. By separating data from design the applied architecture makes it easy even for a web designer without deep Javascript or Python knowledge to customise the looks.

### **Log-in for SoundCloud user**

People could log-in with their SoundCloud account. Thus it would be possible to show friends, followers and favourite tracks or even people who favourited the users tracks on the map. This could be extended with sharing features like "show other people my favourite tracks" adding a social component which could make SoundCloud Map and in the end SoundCloud more known.

## **Conclusion**

For building SoundCloud Map latest technology has been used to create a rich web application that expands the view onto the music uploaded to SoundCloud. It has been designed with a solid architecture to be robust and easy to extend.

## **References**

„jQuery in Action“ by Bear Bibeault and Yehuda Katz, 2008, Manning Pubn.  
The footnotes within this document may be used as reference.