

TimeSide

un framework, un web service et un player open source
d'analyse et d'annotation sonore

Guillaume Pellerin, WAM team, Dpt Innovation et Moyens de la Recherche, IRCAM, Paris

Séminaire Archives Sonores et humanités numériques - 04/05/2023 - Corte, France



Outline

- Introduction
- History
- Core
- Server
- Player
- Perspectives

TimeSide

Context

- **digization** > big data > platforms > algorithms > **listeners**
- more **data sets**, databases, formats, algorithm versions, open DSP libraries & **communities**
- difficulties to **port and scale some algorithms** on production platforms (DSP at scale)
- **computer science** needs human data, **digital humanities** need computers
- **collaborative workflows, citizen sciences** through the web
- **temporal description** of sound and musical events

Constraints

- sustainable digital archiving
- reproducible research
- copyrights, dissemination
- open source, open standards, Web, format evolution and abstraction
- continuous development, continuous integration
- version everything, access everywhere

Use cases

- MIR + archiving = MIRchiving
- digitization and media packaging services
- metadata enhanced streaming services
- music data based applications
- machine learning based applications

TimeSide

History

- 2007 : [Telemeta](#) developed for the sound archives of the CNRS / Musée de l'Homme
- 2010 : TimeSide separation as a autonomous library and then a framework with a plugin oriented architecture
- 2011 : Telemeta v1.0 released and <https://archives.crem-cnrs.fr>
- 2013 : DIADEMS project (ANR) : external plugins
- 2015 : TimeSide server and RESTful API prototypes
- 2015 : KAMoulox (ANR), DaCaRyh (Labex) projects
- 2016 : WASABI (ANR), CREM-NYU-Parisson projects
- 2019-2023 : Fully embed into IRCAM WAM innovation team

Telemeta - CREM

 Search

Welcome, Thomas Fillion | Profile | Help | Sign out

Desk Archives Geo Navigator Advanced search Users

Item : TROMPE ET TAMBOUR :41-15

Title	TROMPE ET TAMBOUR
Original title / translation	OLU BOY
Collector	
Collection	CNRSMH_I_1970_012
Recording date	Nov. 1, 1960 - Nov. 30, 1960
Last modification	Oct. 24, 2012, 5:39 p.m. (admin)

Geographic and cultural informations

Location	Mali, Afrique occidentale, Afrique
Location details	SANGA
Population / social group	DOGON
Ethnographic context	

Musical informations

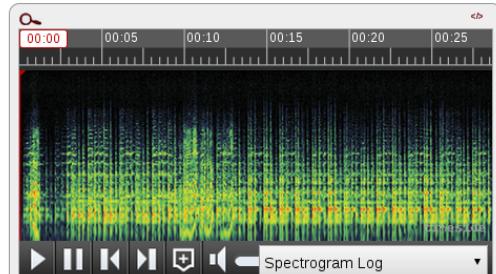
Number	Composition	Vernacular name	Interprets
1	Trompe	KAKELE	
1	Tambour frappé	BOYDUNULE	

Archiving data

Code	CNRSMH_I_1970_012_041_15
Original code	BM.1970.012.001/46:41-15

Edit Copy Add to playlist Previous

Next



Analysis

Property	Value	Unit
Channels	1	
Duration	0:00:28.73	s
Max level	-3.816	dBFS
Mean DC shift	-0	%
MIME type	audio/x-wav	
Resolution	24	bits
RMS level	-19.12	dBFS
Samplerate	48000	Hz

<https://archives.crem-cnrs.fr>

Telemeta - CREM

Desk Archives **Geo Navigator** Advanced search Users

 Geographic Navigator

Map | List

Map Satellite



A world map with numerous red location pins placed across all continents, indicating active users or data points. The pins are densely clustered in Europe, Africa, and Asia, with a more sparse distribution in the Americas and Oceania. The map includes labels for major countries and oceans.

North Pacific Ocean

North Atlantic Ocean

South Pacific Ocean

South Atlantic Ocean

Indian Ocean

Google

Terms of Use Report a map error

6 / 40

Telemeta / TimeSide integration

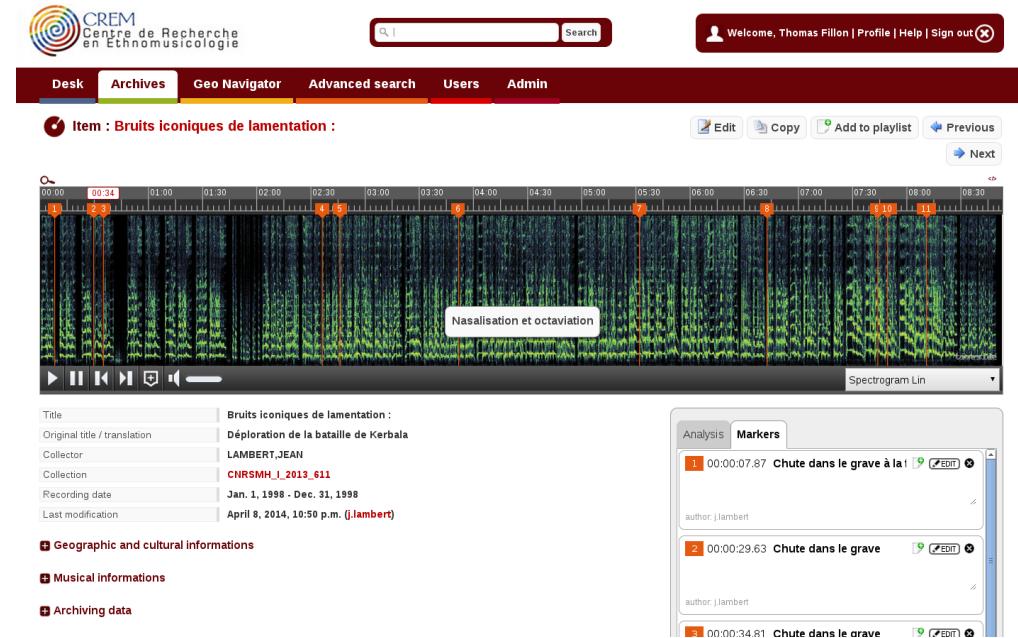


Collaborative multimedia asset management system

MIR + Musicology + Archiving = MIRchiving

Ecosystem

- 20 public partners
- 15 historical developers (6000 commits)
- 500 users (CREM)
- and thousands of developers! (open source community)
- mutualized development model



<https://github.com/Parisson/Telemeta>

timeside.core

TimeSide

Python audio processing framework and server made for the web

<https://github.com/Ircam-WAM/TimeSide>

Goals

- **Process** audio fast and asynchronous with **Python**,
- **Decode** audio frames from *any* audio or video media format into **Numpy arrays**,
- **Analyze** audio content with some **state-of-the-art** audio feature extraction libraries like **Librosa**, **Aubio**, **Essentia**, **Yaafe**, **VAMP**, **IRCAM descriptors** and pure python processors
- **Visualize** audio data with various fancy waveforms, spectrograms and other cool graphers,
- **Transcode** audio data in various media formats and stream them through web apps,
- **Serialize** feature analysis data through various portable formats (XML, JSON, HDF5)
- **Playback and interact on demand** through a smart high-level **HTML5 extensible player**,
- **Index, tag and annotate** audio archives with **cultural and semantic metadata**,
- **Deploy and scale** your own audio processing engine flawlessly through any infrastructure with **Docker**

TimeSide

Python audio processing framework and server made for the web

<https://github.com/Ircam-WAM/TimeSide>

Use cases

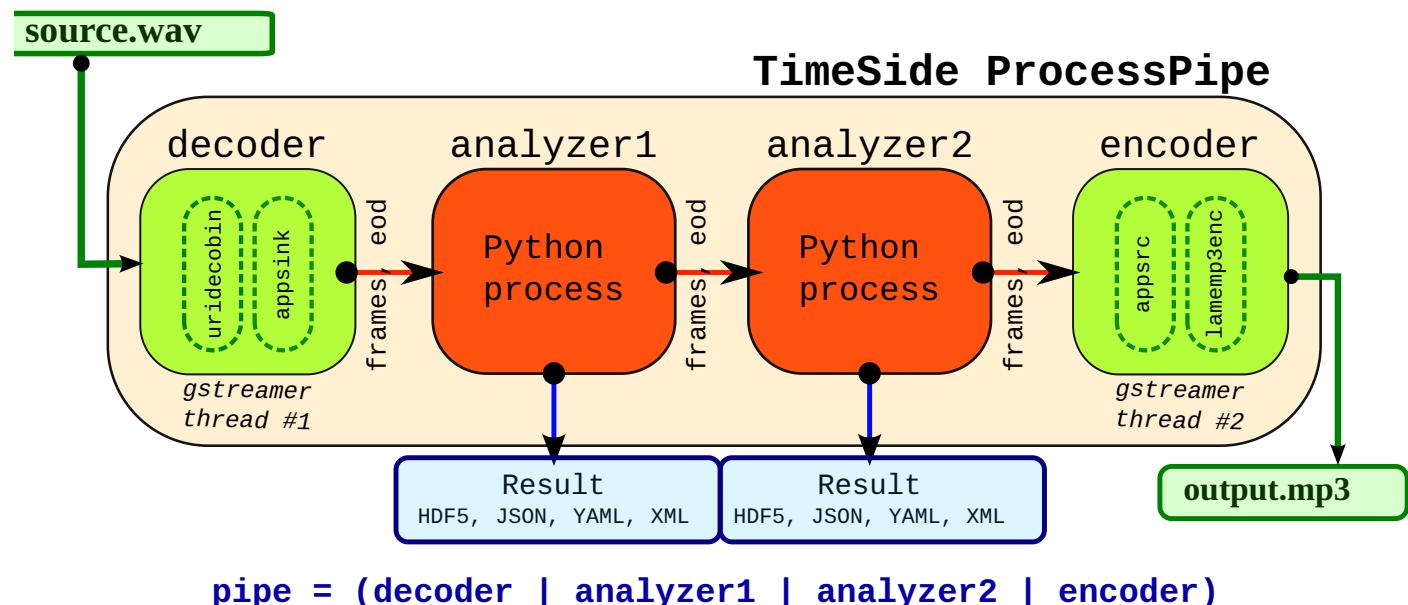
- Scaled audio processing (filtering, transcoding, machine learning, etc...)
- Audio process prototyping
- Audio dynamic visualization
- Automatic segmentation and labelling synchronized with audio events
- Collaborative annotation
- Audio web services

License: AGPL v2

timeside.core

API & architecture

- streaming oriented core engine
- data persistence



API & architecture

- streaming oriented core engine
- data persistence
- processing API
- plugin architecture
- namespace

```
class DummyAnalyzer(Analyzer):
    """A dummy analyzer returning random samples from audio frames"""
    implements(IAnalyzer)

    @interfacedoc
    def setup(self, channels=None, samplerate=None, blocksize=None, totalfram
        super(DummyAnalyzer, self).setup(channels, samplerate, blocksize, tot
        self.values = numpy.array([0])

    @staticmethod
    @interfacedoc
    def id():
        return "dummy"

    @staticmethod
    @interfacedoc
    def name():
        return "Dummy analyzer"

    @staticmethod
    @interfacedoc
    def unit():
        return "None"

    def process(self, frames, eod=False):
        size = frames.size
        if size:
            index = numpy.random.randint(0, size, 1)
            self.values = numpy.append(self.values, frames[index])
        return frames, eod

    def post_process(self):
        result = self.new_result(data_mode='value', time_mode='global')
        result.data_object.value = self.values
```

timeside.core

API & architecture

- streaming oriented core engine
- data persistence
- processing API
- plugin architecture
- namespace
- ~500 unit tests
- docker packaged (Linux, OSX, Windows)

```
import timeside.core
from timeside.core import get_processor
from timeside.core.tools.test_samples import samples

wavfile = samples['sweep.wav']
decoder = get_processor('file_decoder')(wavfile)
grapher = get_processor('spectrogram')()
analyzer = get_processor('level')()
encoder = get_processor('vorbis_encoder')('sweep.ogg')

pipe = (decoder | grapher | analyzer | encoder)
pipe.run()

grapher.render(output='spectrogram.png')
print('Level:', analyzer.results)
Level: {'level.max': AnalyzerResult(...), 'level.rms': AnalyzerResult(...)}
```

```
$ git clone --recursive https://github.com/Ircam-WAM/TimeSide.git
$ docker-compose run app python3 manage.py shell
$ docker-compose up
```

<https://timeside.readthedocs.io/en/latest/index.html>

timeside.core

Plugin examples

- FileDecoder, ArrayDecoder, LiveDecoder, AubioDecoder
- VorbisEncoder, WavEncoder, Mp3Encoder, FlacEncoder, OpusEncoder, etc.
- WaveformAnalyzer, SpectrogramAnalyzer
- AubioTemporal, AubioPitch, etc.
- Yaafe wrapper (graph oriented)
- librosa (function oriented)
- VampPyHost
- Essentia bridge
- ...
- Speech detection
- Music detection
- Singing voice detection
- Monophony / Polyphony
- Dissonance
- Timbre Toolbox

<https://github.com/DIADEMS/timeside-diadems>

timeside.core

Roadmaps

0.9 > 1.0 (2019-2021)

- 674 commits, 7 contributors
- Python 2.7 to 3.7
- Drop GStreamer for Aubio as default decoder and encoder
- Regroup all dependencies on pip requirements, drop Conda
- Add `Provider` object
- Add Provider plugins `deezer-preview`, `deezer-complete` and `youtube`
- Add loggers
- Improve Docker packaging and image building

timeside.core

Roadmaps

1.0 > 1.1 (2021 - 2023)

- Use the libav based aubio decoder by default (fastest audio to numpy array converter on the planet!)
- Add a VAMP based Analyzer and a few plugins like `VampFlatness`, `VampCrest`, `VampTempo`, `VampTuning`, `VampSpectralCentroid`, `VampSpectralKurtosis` and `VampSpectralSlope`
- Updated documentation <https://timeside.ircam.fr/docs/>

timeside.server

timeside.server

RESTful API built on TimeSide

👉 <https://timeside.ircam.fr/api/>

Use cases

- **Upload** audio tracks
- **Retrieve** audio tracks from remote providers
- **Stream** original or transcoded sources
- **Run** on-demand analysis
- **Customize** processors parameters
- **Collect** track's annotation to build audio corpora
- **Deliver** and share several types of results:
 - transcoded audio
 - serialized analysis as JSON or image
- **Export/import** an audio analysis dataset

Api Root

The default basic root view for DefaultRouter

GET /timeside/api/

HTTP 200 OK

Allow: GET, HEAD, OPTIONS

Content-Type: application/json

Vary: Accept

```
{  
    "users": "https://sandbox.wasabi.telemeta.org/timeside/api/users/",  
    "items": "https://sandbox.wasabi.telemeta.org/timeside/api/items/",  
    "providers": "https://sandbox.wasabi.telemeta.org/timeside/api/providers/",  
    "selections": "https://sandbox.wasabi.telemeta.org/timeside/api/selections/",  
    "processors": "https://sandbox.wasabi.telemeta.org/timeside/api/processors/",  
    "subprocessors": "https://sandbox.wasabi.telemeta.org/timeside/api/subprocessors/",  
    "presets": "https://sandbox.wasabi.telemeta.org/timeside/api/presets/",  
    "experiences": "https://sandbox.wasabi.telemeta.org/timeside/api/experiences/",  
    "tasks": "https://sandbox.wasabi.telemeta.org/timeside/api/tasks/",  
    "analysis": "https://sandbox.wasabi.telemeta.org/timeside/api/analysis/",  
    "analysis_tracks": "https://sandbox.wasabi.telemeta.org/timeside/api/analysis_tracks/",  
    "annotation_tracks": "https://sandbox.wasabi.telemeta.org/timeside/api/annotation_tracks/",  
    "annotations": "https://sandbox.wasabi.telemeta.org/timeside/api/annotations/",  
    "results": "https://sandbox.wasabi.telemeta.org/timeside/api/results/"  
}
```

OPTIONS GET ▾

timeside.server

RESTful API built on TimeSide

👉 <https://timeside.ircam.fr/api/>

Server design

- Based on Django REST Framework (DRF)
- **Interoperability** between other servers or frontends and TimeSide instance and its data
- Object-relational **database** in order to store music tracks and processing results
- **Models**: define essential fields and behaviors of stored data
- **queue-worker architecture** enables to run tasks asynchronously

Api Root

The default basic root view for DefaultRouter

GET /timeside/api/

HTTP 200 OK

Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

```
{  
    "users": "https://sandbox.wasabi.telemeta.org/timeside/api/users/",  
    "items": "https://sandbox.wasabi.telemeta.org/timeside/api/items/",  
    "providers": "https://sandbox.wasabi.telemeta.org/timeside/api/providers/",  
    "selections": "https://sandbox.wasabi.telemeta.org/timeside/api/selections/",  
    "processors": "https://sandbox.wasabi.telemeta.org/timeside/api/processors/",  
    "subprocessors": "https://sandbox.wasabi.telemeta.org/timeside/api/subprocessors/",  
    "presets": "https://sandbox.wasabi.telemeta.org/timeside/api/presets/",  
    "experiences": "https://sandbox.wasabi.telemeta.org/timeside/api/experiences/",  
    "tasks": "https://sandbox.wasabi.telemeta.org/timeside/api/tasks/",  
    "analysis": "https://sandbox.wasabi.telemeta.org/timeside/api/analysis/",  
    "analysis_tracks": "https://sandbox.wasabi.telemeta.org/timeside/api/analysis_tracks/",  
    "annotation_tracks": "https://sandbox.wasabi.telemeta.org/timeside/api/annotation_tracks/",  
    "annotations": "https://sandbox.wasabi.telemeta.org/timeside/api/annotations/",  
    "results": "https://sandbox.wasabi.telemeta.org/timeside/api/results/"  
}
```

OPTIONS GET ▾

timeside.server

RESTful API built on TimeSide

👉 <https://timeside.ircam.fr/api/>

What's new on server?

- Add audio providers (Deezer, Youtube)
- Switch from MySQL to PostgreSQL
- Add a JWT authentication
- Make the API follow the OpenAPI specification
- Build a TypeScript SDK on the REST API
- Add several tools, views, models and serializers
- Improve server unit testing
- Fix few bugs
- Python, Django, DRF and Celery upgrades

Api Root

The default basic root view for DefaultRouter

GET /timeside/api/

HTTP 200 OK

Allow: GET, HEAD, OPTIONS

Content-Type: application/json

Vary: Accept

```
{  
    "users": "https://sandbox.wasabi.telemeta.org/timeside/api/users/",  
    "items": "https://sandbox.wasabi.telemeta.org/timeside/api/items/",  
    "providers": "https://sandbox.wasabi.telemeta.org/timeside/api/providers/",  
    "selections": "https://sandbox.wasabi.telemeta.org/timeside/api/selections/",  
    "processors": "https://sandbox.wasabi.telemeta.org/timeside/api/processors/",  
    "subprocessors": "https://sandbox.wasabi.telemeta.org/timeside/api/subprocessors/",  
    "presets": "https://sandbox.wasabi.telemeta.org/timeside/api/presets/",  
    "experiences": "https://sandbox.wasabi.telemeta.org/timeside/api/experiences/",  
    "tasks": "https://sandbox.wasabi.telemeta.org/timeside/api/tasks/",  
    "analysis": "https://sandbox.wasabi.telemeta.org/timeside/api/analysis/",  
    "analysis_tracks": "https://sandbox.wasabi.telemeta.org/timeside/api/analysis_tracks/",  
    "annotation_tracks": "https://sandbox.wasabi.telemeta.org/timeside/api/annotation_tracks/",  
    "annotations": "https://sandbox.wasabi.telemeta.org/timeside/api/annotations/",  
    "results": "https://sandbox.wasabi.telemeta.org/timeside/api/results/"  
}
```

OPTIONS

GET ▾

timeside.server

RESTful API built on TimeSide

👉👉 <https://timeside.ircam.fr/api/>

Models

- Item: audio content and metadata (external id)
- Provider: provide audio from a given platform
- Selection: list of items (corpus)
- Processor: plugins with version and default parameters
- Preset: processor and a set of parameters
- Experience: list of presets forming a pipe (reproducible)
- Task: an experience and a selection
- Result: transcoded audio or numerical outputs (hdf5 file)
- Annotation: label audio file on a given time or segment

Api Root

The default basic root view for DefaultRouter

GET /timeside/api/

HTTP 200 OK

Allow: GET, HEAD, OPTIONS

Content-Type: application/json

Vary: Accept

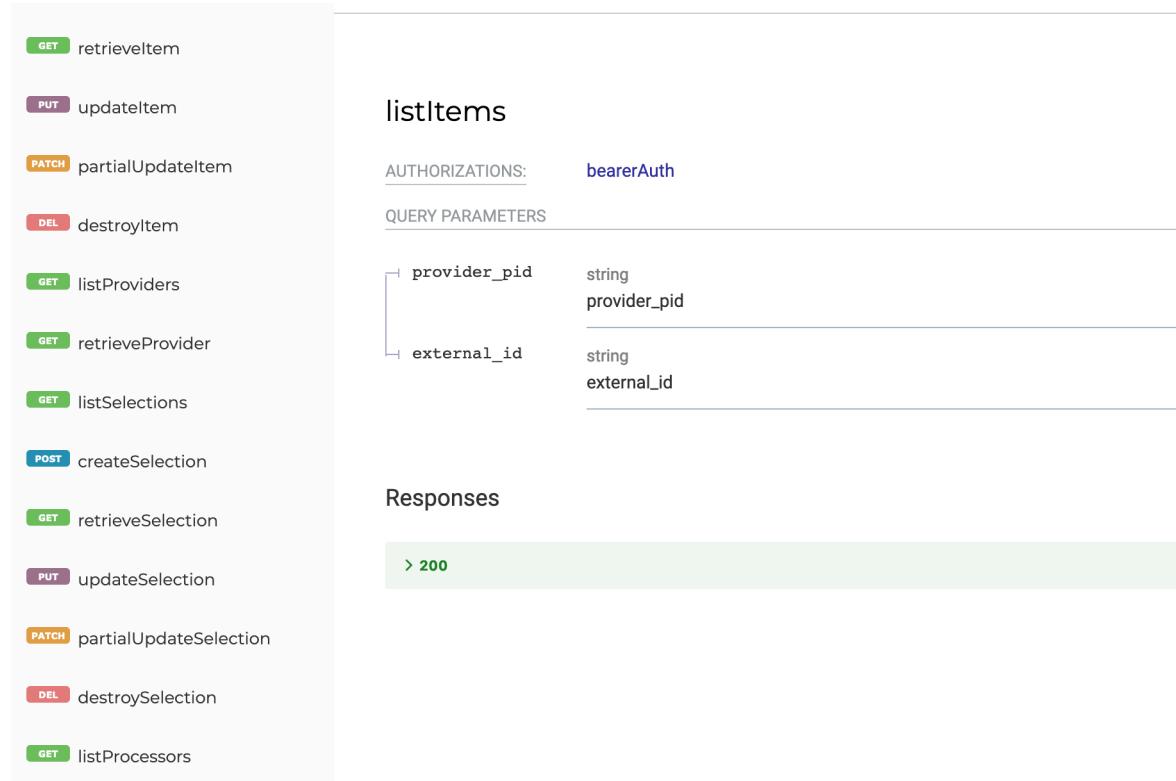
```
{  
    "users": "https://sandbox.wasabi.telemeta.org/timeside/api/users/",  
    "items": "https://sandbox.wasabi.telemeta.org/timeside/api/items/",  
    "providers": "https://sandbox.wasabi.telemeta.org/timeside/api/providers/",  
    "selections": "https://sandbox.wasabi.telemeta.org/timeside/api/selections/",  
    "processors": "https://sandbox.wasabi.telemeta.org/timeside/api/processors/",  
    "subprocessors": "https://sandbox.wasabi.telemeta.org/timeside/api/subprocessors/",  
    "presets": "https://sandbox.wasabi.telemeta.org/timeside/api/presets/",  
    "experiences": "https://sandbox.wasabi.telemeta.org/timeside/api/experiences/",  
    "tasks": "https://sandbox.wasabi.telemeta.org/timeside/api/tasks/",  
    "analysis": "https://sandbox.wasabi.telemeta.org/timeside/api/analysis/",  
    "analysis_tracks": "https://sandbox.wasabi.telemeta.org/timeside/api/analysis_tracks/",  
    "annotation_tracks": "https://sandbox.wasabi.telemeta.org/timeside/api/annotation_tracks/",  
    "annotations": "https://sandbox.wasabi.telemeta.org/timeside/api/annotations/",  
    "results": "https://sandbox.wasabi.telemeta.org/timeside/api/results/"  
}
```

OPTIONS GET ▾

timeside.server

RESTful API documentation

auto-generated thanks to ReDoc on top of OpenAPI  <https://timeside.ircam.fr/api/docs/>

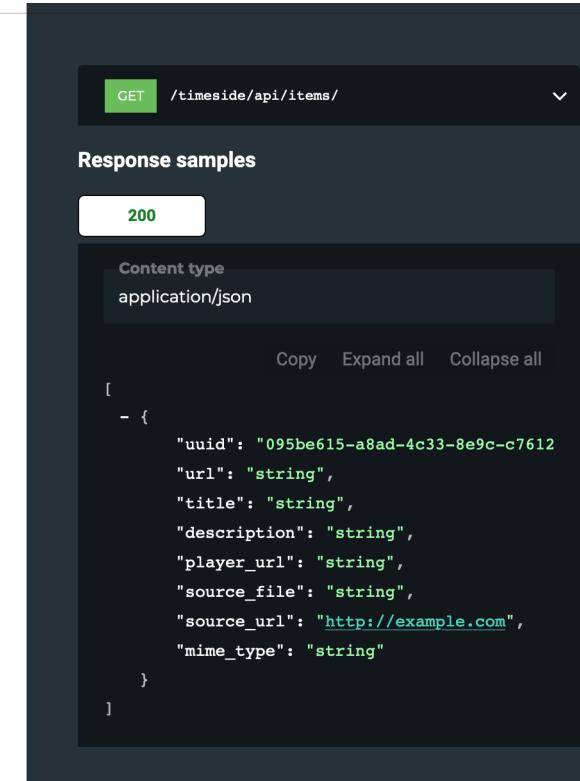


The screenshot shows the ReDoc interface for the timeside.server API. On the left, a sidebar lists various endpoints with their HTTP methods and descriptions:

- GET retrieveItem
- PUT updateItem
- PATCH partialUpdateItem
- DELETE destroyItem
- GET listProviders
- GET retrieveProvider
- GET listSelections
- POST createSelection
- GET retrieveSelection
- PUT updateSelection
- PATCH partialUpdateSelection
- DELETE destroySelection
- GET listProcessors

The main content area is focused on the **listItems** endpoint. It includes:

- A sidebar with **AUTHORIZATIONS:** `bearerAuth`.
- QUERY PARAMETERS:**
 - `provider_pid`: string
 - `external_id`: string
- Responses:** A section showing the **> 200** response.



The screenshot shows a detailed view of a response sample for the `GET /timeside/api/items/` endpoint. The response is shown in JSON format:

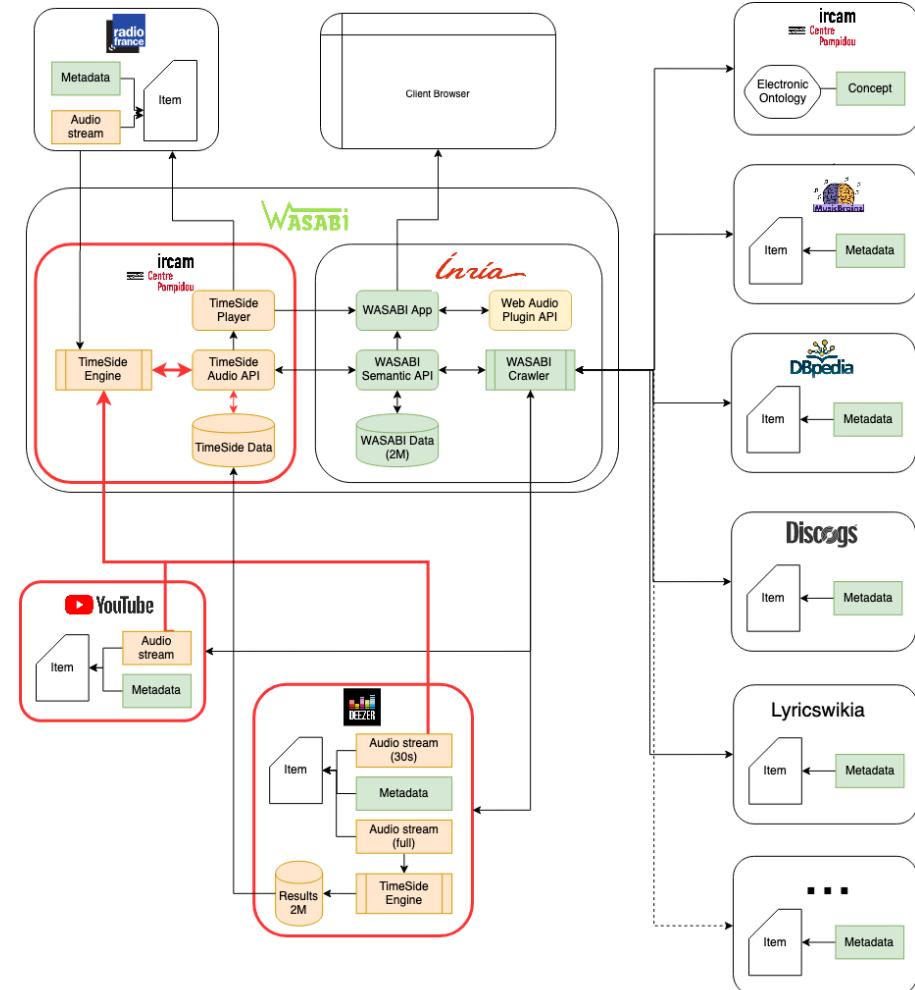
```
[  
  - {  
      "uuid": "095be615-a8ad-4c33-8e9c-c7612",  
      "url": "string",  
      "title": "string",  
      "description": "string",  
      "player_url": "string",  
      "source_file": "string",  
      "source_url": "http://example.com",  
      "mime_type": "string"  
    }  
]
```

timeside.server - wasabi project

workflow examples in the WASABI platform

with providers

- Youtube
 - based on [youtube-dl](#)
 - must be adaptable to YouTube's changes
- Deezer 30 seconds long preview
 - consuming Deezer's API
 - find another solution to full contents



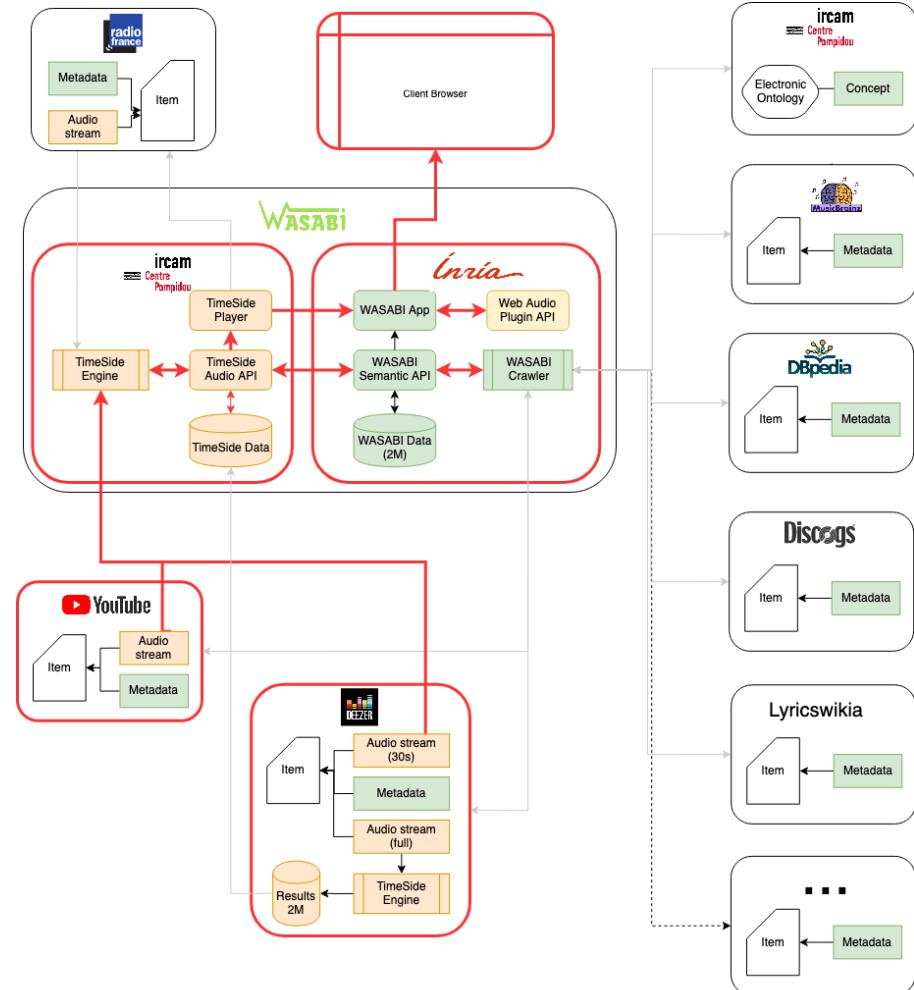
timeside.server - wasabi project

workflow examples in the WASABI platform

with server

POC of a webservice

- delivers audio analysis to another remote server
- enhance its musical metadata with results



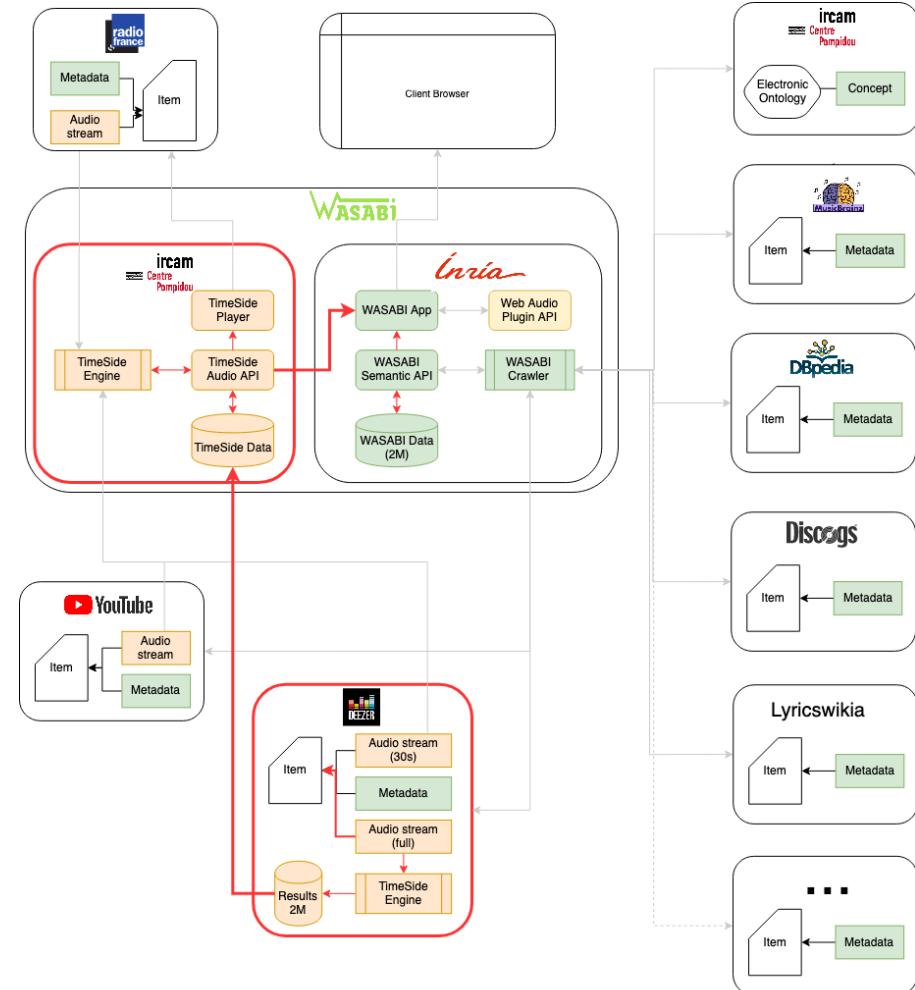
timeside.server - wasabi project

workflow examples in the WASABI platform

Import/export of a run on Deezer's infrastructure

POC of a sharing system of an audio analysis datasets

- easy deployment thanks to docker
- audio does not have to be shared



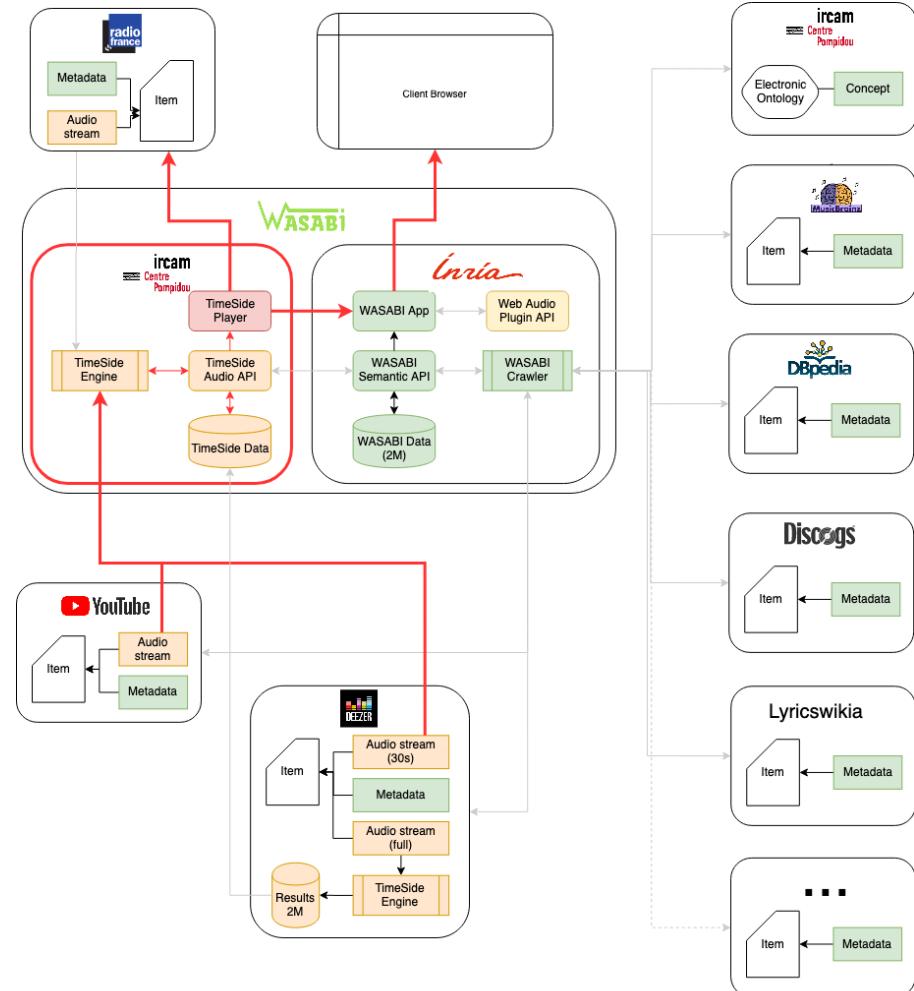
timeside.server - wasabi project

workflow examples in the WASABI platform

with a frontend player

POC of a JavaScript app that consume the Rest API

- serialize data as JSON or image
- deliver analysis track (spectrogram, waveform)
- deliver or get annotations on audio track



timeside.server - wasabi project ([link](#))

S

Led Zeppelin

SHOW RDF



Led Zeppelin

W                 

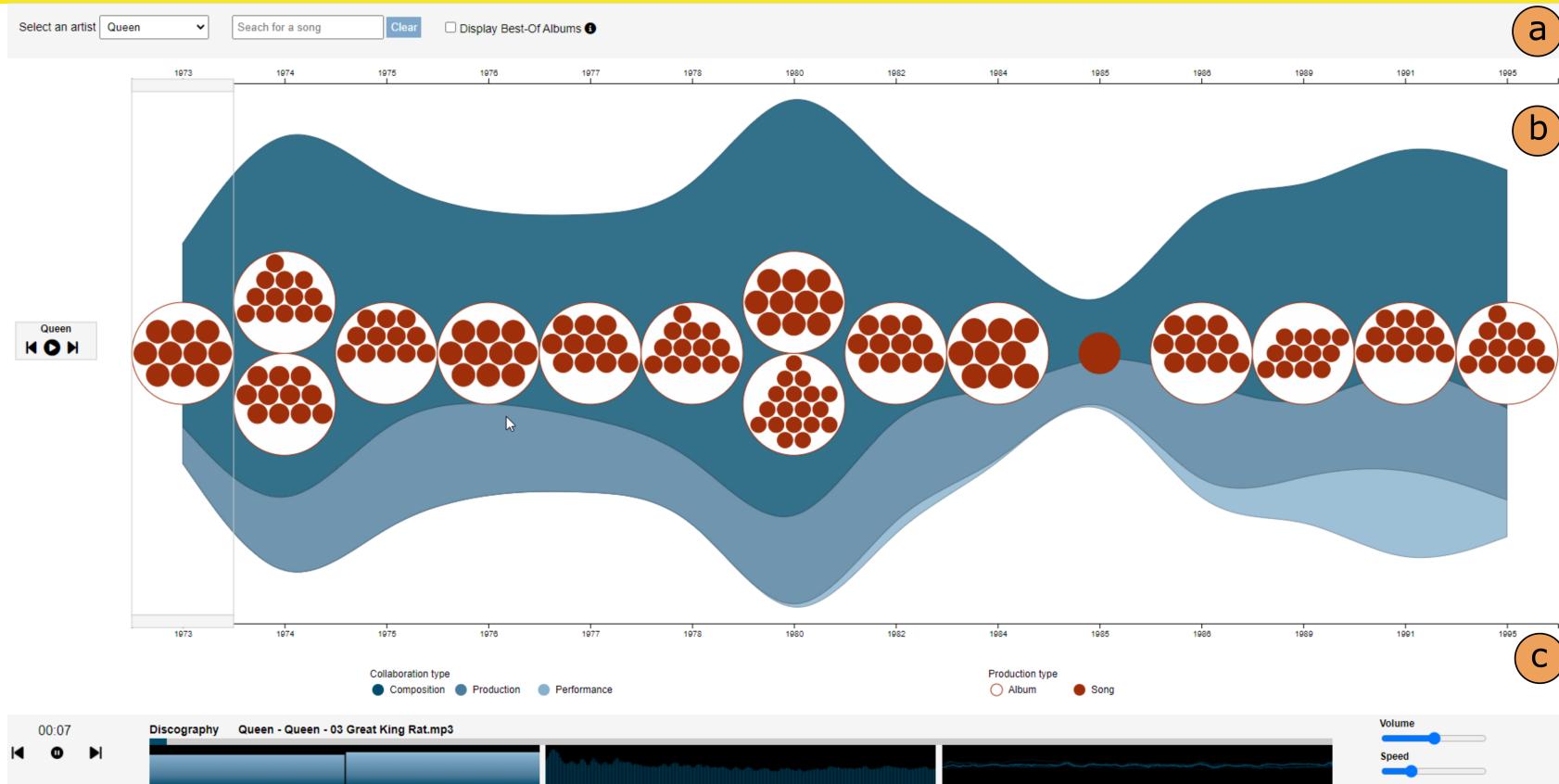
Led Zeppelin were an English rock band formed in London in 1968. The group consisted of guitarist Jimmy Page, singer Robert Plant, bassist and keyboardist John Paul Jones, and drummer John Bonham. The band's heavy, guitar-driven sound, rooted in blues and psychedelia on their early albums, has earned them recognition as one of the progenitors of heavy metal, though their unique style drew from a wide variety of influences, including folk music and blues. After changing their name from the New Yardbirds, Led Zeppelin signed a deal with Atlantic Records that afforded them considerable artistic freedom. Although the group was initially unpopular with critics, they achieved significant commercial success with albums such as *Led Zeppelin* (1969), *Led Zeppelin II* (1969), *Led Zeppelin III* (1970), their untitled fourth album (1971), *Houses of the Holy* (1973), and *Physical Graffiti* (1975). Their fourth album, which features the track "Stairway to Heaven", is among the most popular and influential works in rock music, and it helped to secure the group's popularity. Page wrote most of Led Zeppelin's music, particularly early in their career, while Plant generally supplied the lyrics. Jones' keyboard-based compositions later became central to the group's catalogue, which featured increasing experimentation. The latter half of their career saw a series of record-breaking tours that earned the group a reputation for excess and debauchery. Although they remained commercially and critically successful, their output and touring schedule were limited during the late 1970s, and the group disbanded following Bonham's death from alcohol-related asphyxia in 1980. In the decades that followed, the surviving members sporadically collaborated and participated in one-off Led Zeppelin reunions. The most successful of these was the 2007 Ahmet Ertegun Tribute Concert in London, with Jason Bonham taking his late father's place behind the drums. Led Zeppelin are widely considered one of the most successful, innovative, and influential rock groups in history. They are one of the best-selling music artists in the history of audio recordings.

timeside.server - wasabi project ([link](#))

The screenshot shows a web application interface with a blue header bar. The header includes a menu icon (three horizontal lines), a search bar with the text "Led Zeppelin", a microphone icon for voice search, and a user profile icon (a small letter 'e'). Below the header is a navigation menu with links: Home, DiscoveryHub, QMUL/chords, IRCAM/Timeside, WebAudio tools, WebAudio plugins, and Multitracks. A prominent blue button labeled "<>SHOW RDF" is visible. The main content area displays a search result for "Led Zeppelin". On the left is a thumbnail of the band's photo. In the center, the title "Whole Lotta Love" is displayed above a video player showing a black and white photo of Robert Plant. To the right is another thumbnail for "Led Zeppelin II". At the bottom of the result card are sharing icons for WhatsApp (W), YouTube (play button), Dailymotion (DM), and others.

"Whole Lotta Love" is a song by English hard rock band Led Zeppelin. It is the opening track on the band's second album, *Led Zeppelin II*, and was released in the United States and Japan as a single. The US release became their first hit single; it was certified Gold on 13 April 1970, having sold one million copies. As with other Led Zeppelin songs, no single was released in the United Kingdom, but singles were released in Germany (where it reached number one), the Netherlands (where it reached number four), Belgium and France. In 2004, the song was ranked number 75 on Rolling Stone magazine's list of the 500 Greatest Songs of All Time, and in March 2005, Q magazine placed "Whole Lotta Love" at number three in its list of the 100 Greatest Guitar Tracks. It was placed 11 on a similar list by Rolling Stone. In 2009 it was named the third greatest hard rock song of all time by VH1. Already part of their live repertoire, "Whole Lotta Love" saw its first official release on the LP *Led Zeppelin II* on 22

timeside.server - wasabi project ([link](#))



Aline Menin, Michel Buffa, Maroua Tikat, Benjamin Molinet, Guillaume Pellerin, Laurent Pottier, Franck Michel, & Marco Winckler. (2022, June 28). Incremental and multimodal visualization of discographies: exploring the WASABI music knowledge base. Web Audio Conference 2022 (WAC 2022), Cannes, France.
<https://doi.org/10.5281/zenodo.6767530>

timeside.player

timeside.player - v1

Features

- full HTML through SoundManager2
- on demand processing and display
- simple marker annotation
- bitmap image cache only



timeside.player - v2

Constraints

- Handling multiple hours audio files
- Multiple user annotations and analysis tracks
- Vectorized
- Zooming
- Scaling



timeside.player - v2

API SDK (client library)

- Timeside API: 75 routes
- openapi-generator
 - Typescript
 - Fetch
 - OpenAPI v3 Schemas
- Improve schema support on DRF (PR)
 - Components
 - Customize default names
- Glue code
 - Authentication
 - Initialization on Browser / Node
- Documentation
- SDK: <https://github.com/Ircam-WAM/timeside-sdk-js>
- Scripts: <https://github.com/Ircam-WAM/timeside-scripts>

```
/timeside/api/analysis/:  
  get:  
    operationId: listAnalysis  
    parameters: []  
    responses:  
      '200':  
        content:  
          application/json:  
            schema:  
              type: array  
              items:  
                $ref: '#/components/schemas/Analysis'
```

Opportunity: `openapi-generator` also supports Python, C/C++, Ruby, Go, Rust etc...

timeside.player - v2

HTML Player

Technologies

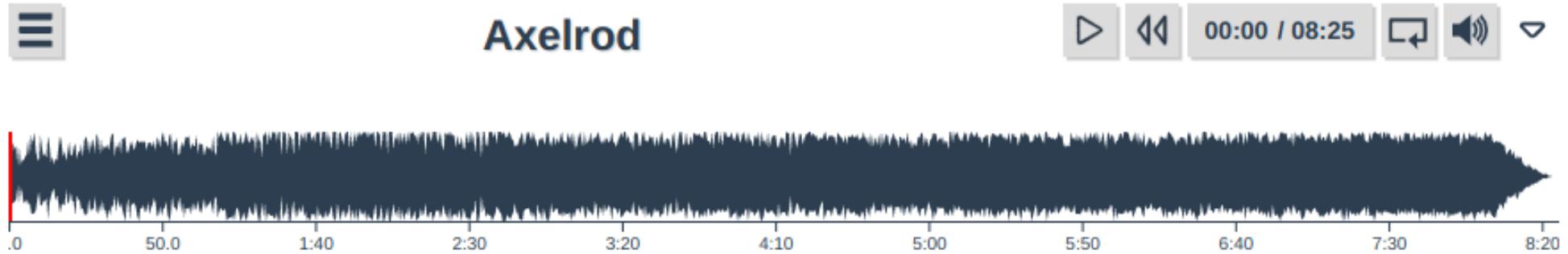
- Vue (composition-api): DOM Manipulation, Data reactivity
- D3 (SVG): Render waveform / Analysis
- HTML5 Audio / Web Audio
- Web Animations API
- Websockets
- Resize Observer
- GitHub Action for continuous test & deployment (npm, gh-page)
- <https://github.com/Ircam-WAM/timeside-player>

Usage

- Standalone app
- Web library
 - React
 - Vue
 - HTML

Demo time!

timeside - embed player



timeside - music explorer (AI4Media project)

Logout

Music Explorer

Attribute search Similarity search

Select attributes

[instrumental x] [happy x] [jazz x]

instrumental: 23 happy: 50 jazz: 33

Select tempo range

90 160 BPM

UPDATE

FILE	INFO	RESULTS
	Artist: instrumental: 94% happy: 94% jazz: 94% Album: tempo: 181 BPM Title: ItemTest00	search similar music
instrumental: 94% sung: 94% drama: 94% background: 94% fun: 94% ambience: 94% energetic: 94% calm: 94% emotional: 94% dark: 94% happy: 94% love: 94% party: 94% sad: 94% game: 94% horror: 94% metal_and_hard: 94% rock: 94% blues: 94% classical: 94% jazz: 94% folk: 94% pop: 94% electro: 94% rap: 94% reggae_and_ska: 94% world: 94%		

rows per page: 25 ▾ 1-1 of 1 < 1 >

Logout

Music Explorer

Attribute search Similarity search

Choisir un fichier hoedic_p...seaux.wav enter a youtube url

UPLOAD

Title: hoedic_panorama_hold2_oiseaux.wav

Select musical dimension

[genre x] [instrumentation x] [era x]

UPDATE

FILE	INFO	RESULTS
	ItemTest00	Distance to the reference: 5.00 search similar music
genre: 5.00 mood: 5.00 instrumentation: 5.00 era: 5.00 tempo: 5.00 key: 5.00		

rows per page: 25 ▾ 1-1 of 1 < 1 >

TimeSide - roadmaps

1.0 > 1.1 (2021)

- composition
 - base image upgrades
- core
 - lib upgrades
 - VampAnalyzer
 - graphers are back
- server
 - upgrade deps
 - stabilization (worker)
 - API doc
- worker
 - full asynchronous
 - better YouTube provider handling
- player
 - embed design
 - better annotation editor
 - listing

1.1 > 1.2 (2023)

- core
 - parameters
 - switch to poetry (python package manager)
- server
 - extended API
- worker
 - websocket based monitoring
 - test Deezer provider at scale
- player
 - uploader
 - better listing
 - looping

Perspectives

Audio processing and annotation web service (SaaS)

TODO list

- vertical scales
- clustering and orchestration
- synchronous task status polling
- more plugins (VAMP host, source separation)
- test some JS MIR librairies (Essentia.js)
- easy upload
- audio cross comparison functions
- more embed experiences
- re-use of archives in contemporary music

Dual licencing?

- open source community release of the core framework (AGPL)
- proprietary (entreprise) release with specific plugins

New partnerships

- IRCAM Amplify
- BNF ?
- +CNRS ?
- LISA ?

Thank you and kudos to all contributors!

timeside.ircam.fr

guillaume.pellerin@ircam.fr / [@yomguy](https://twitter.com/yomguy)

