XMMS2 Collections

Sébastien Cevey <seb@cine7.net>
http://bytes.inso.cc/

Metaweb presentation (October 22, 2009)

Swiss Federal Institute of Technology, Lausanne

XMMS2 Music Player

Playlouder MSP

Outline

1. Introduction
2. Concept
3. Usage
4. Future
5. Conclusion

1. Introduction

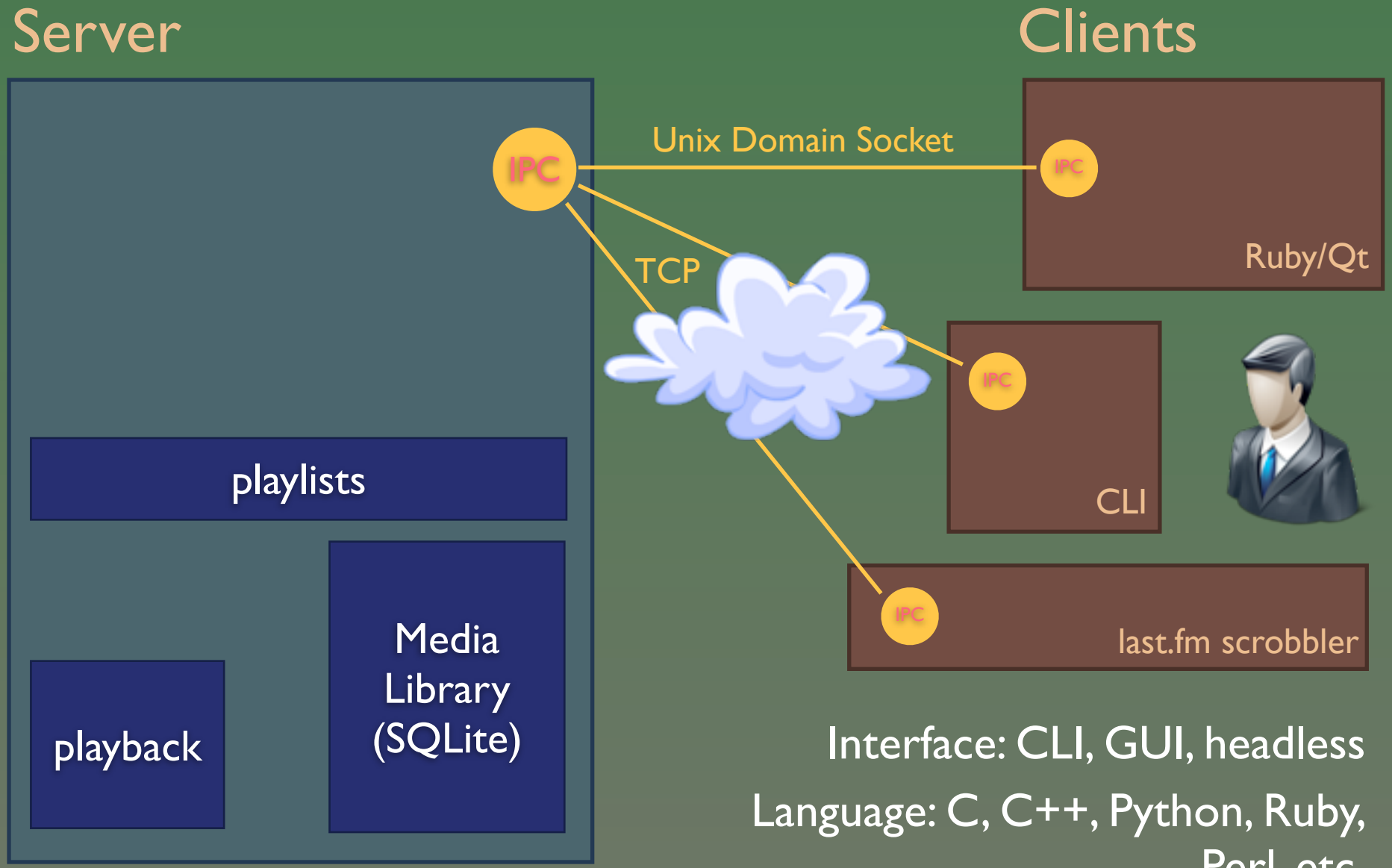
"why all the fuss?"

Personal Interests

The project initially comes from personal interest in:

- Software engineering
  design smart abstractions to build powerful things
- Information architecture
  exploit computers to manage large collections of structure information

XMMS2 Land



Interface: CLI, GUI, headless

Language: C, C++, Python, Ruby, Perl, etc.

XMMS2 Media Library

- SQLite database
- Store of objects ("media", typically songs) with arbitrary properties
- Denormalized schema, with unique (id, key)

id	key	value
42	artist	Britney Spears
42	title	Toxic
42	duration	245497
43	artist	Beyonce

- SQL queries require lots of JOINs

UI Requirements

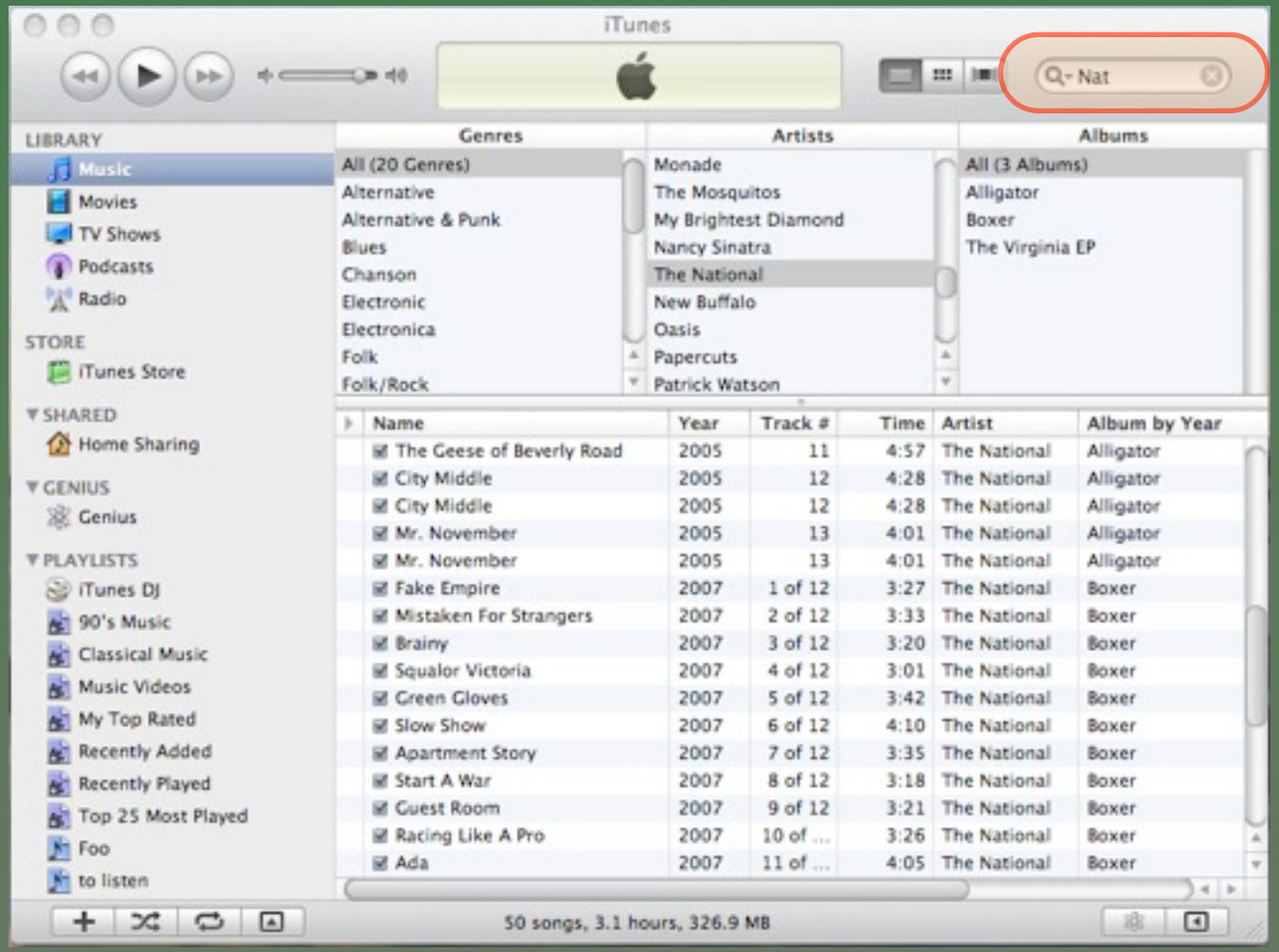
- Browse
- Search
- Organize

UI Requirements

- Browse
- Search
- Organize

UI Requirements

- Browse
- Search
- Organize



UI Requirements

- Browse
- Search
- Organize

UI Requirements

- Browse
- Search
- Organize

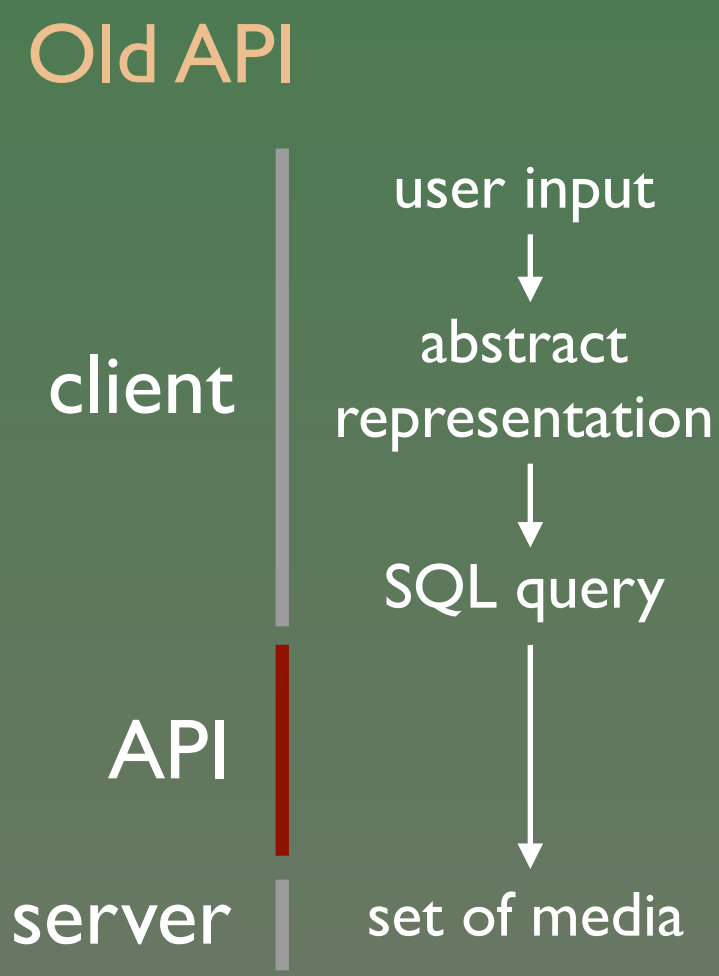
i.e.

view & manipulate subsets of the media library

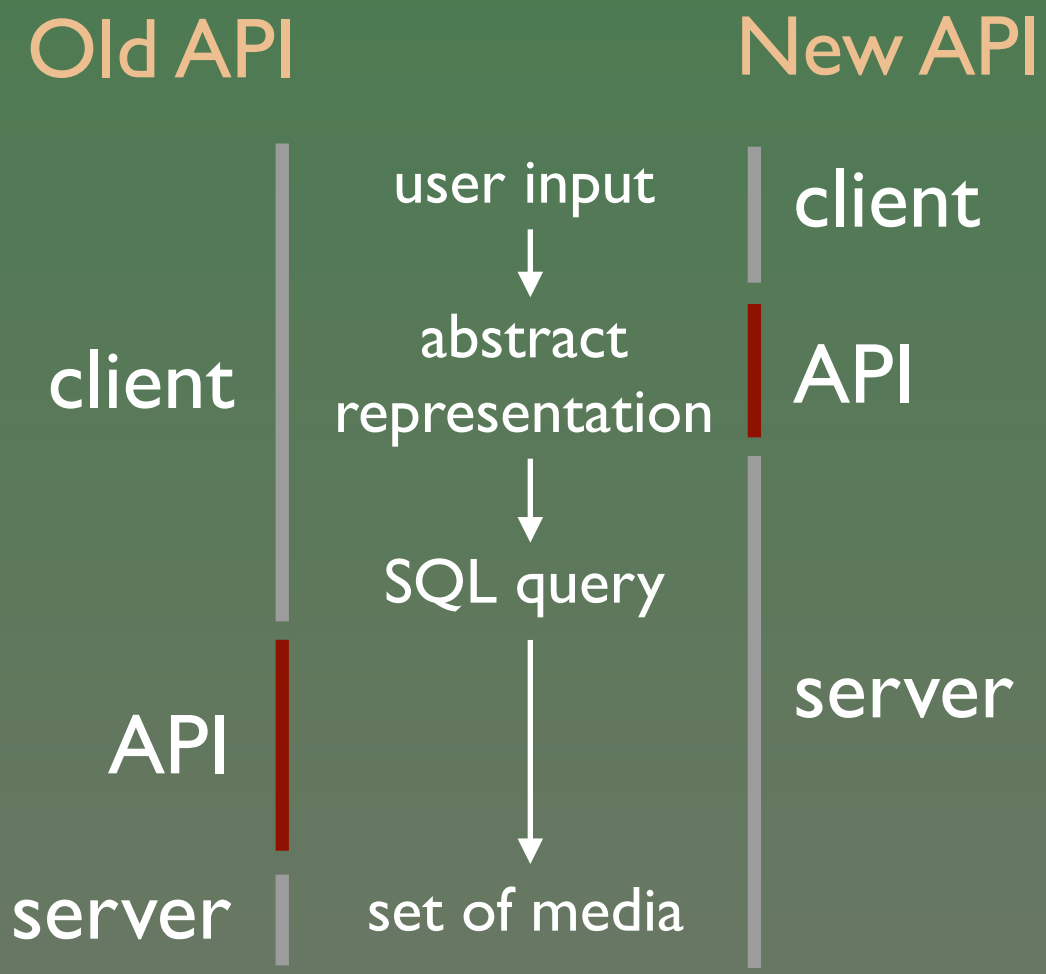
The Problem

- Abstract the structure of the media library
- Simplify querying of subsets of the media library
- Share abstraction of such subsets among clients
- Allow refining, editing, composing the subsets

The Problem

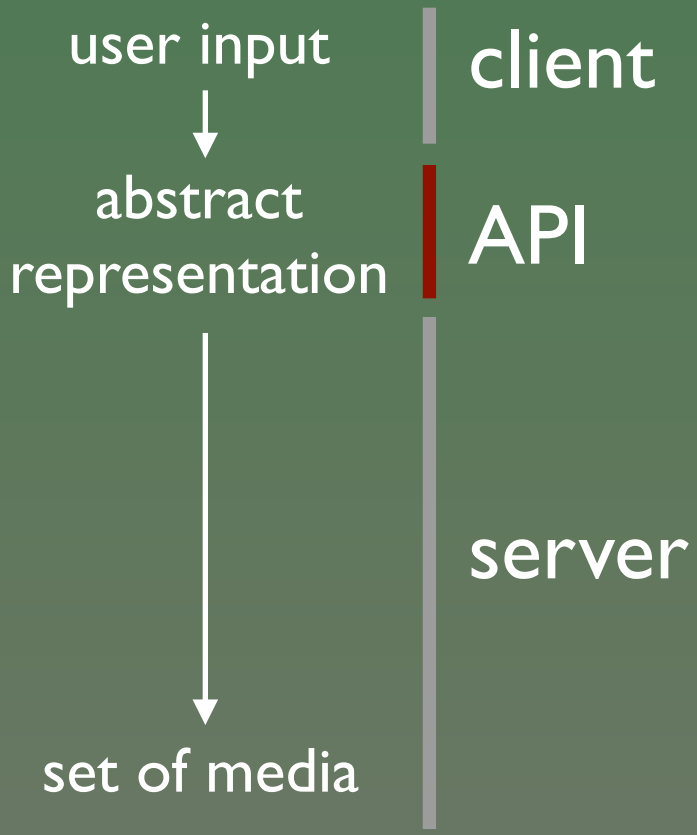

Old API
client
API
server
user input
abstract representation
SQL query
set of media

The Solution



The Solution

Collections =



2. Concept

"what are you on about?"

Collection: Definition

"A collection is a representation of a subset of the media library, structured as a Directed Acyclic Graph (DAG) of operators."

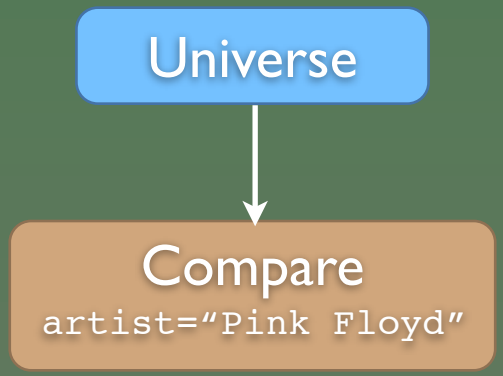
Collection: Definition

"A collection is a representation of a subset of the media library, structured as a Directed Acyclic Graph (DAG) of operators."

Universe

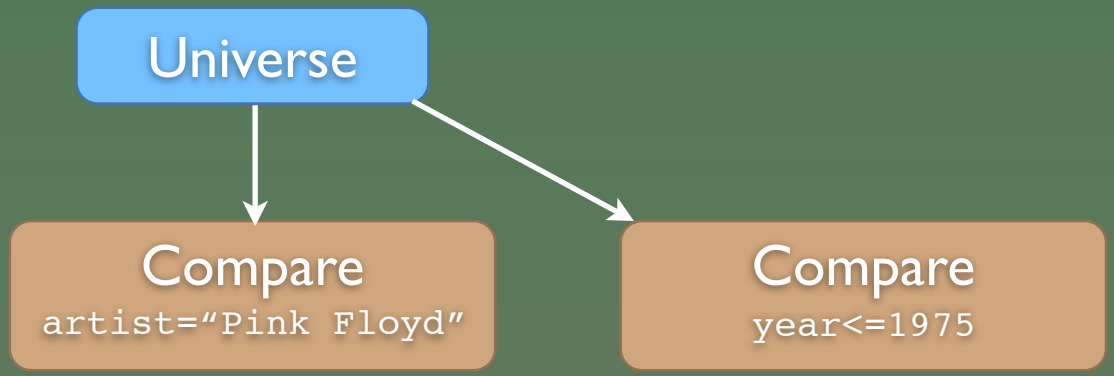
Collection: Definition

"A collection is a representation of a subset of the media library, structured as a Directed Acyclic Graph (DAG) of operators."



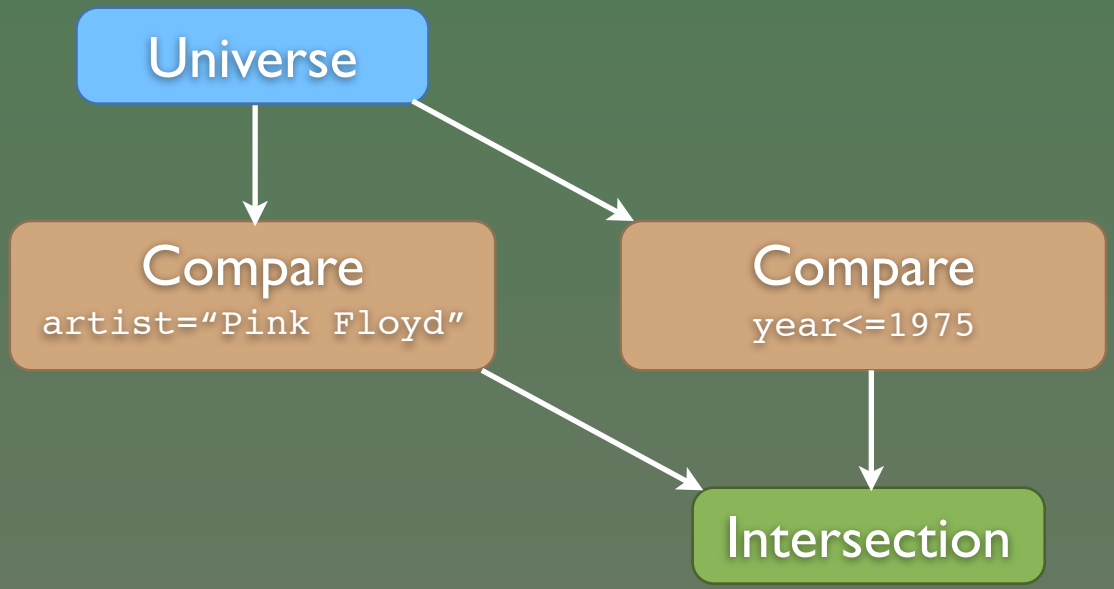
Collection: Definition

"A collection is a representation of a subset of the media library, structured as a Directed Acyclic Graph (DAG) of operators."



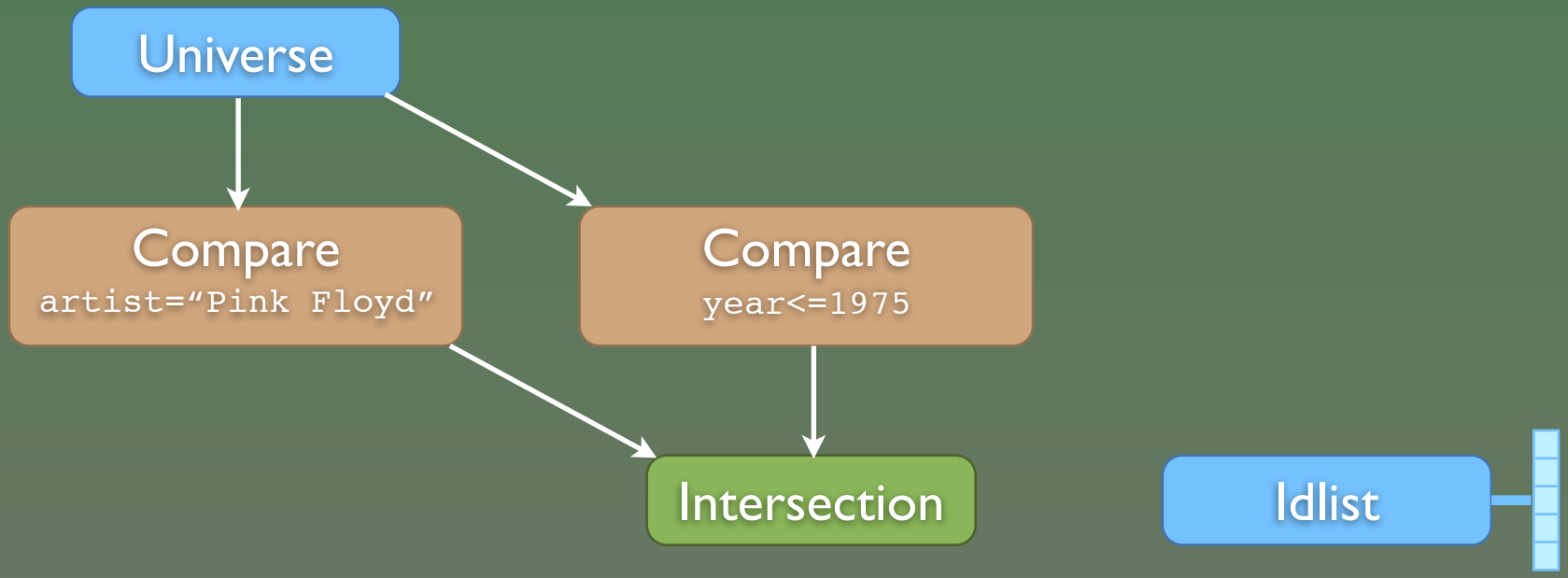
Collection: Definition

"A collection is a representation of a subset of the media library, structured as a Directed Acyclic Graph (DAG) of operators."



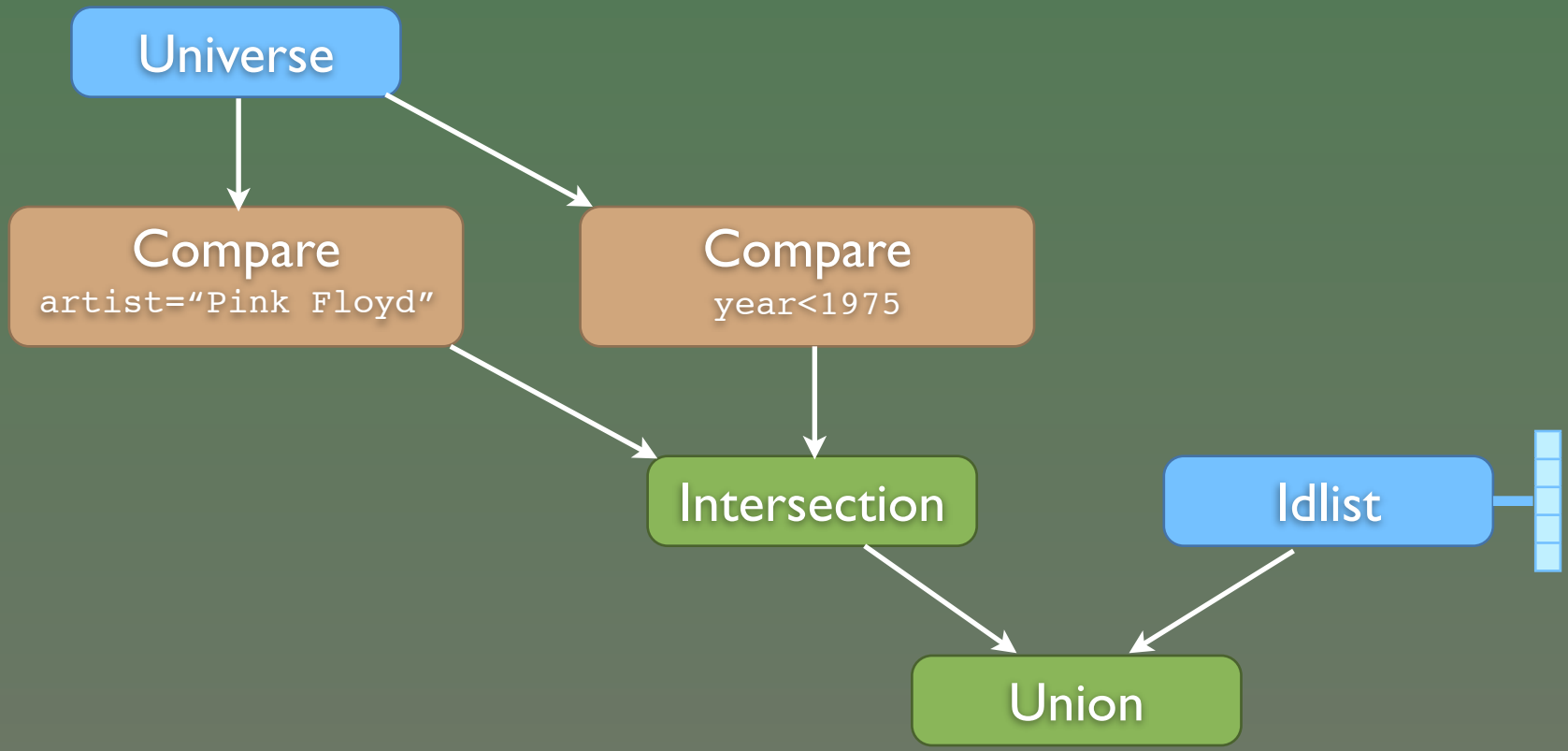
Collection: Definition

"A collection is a representation of a subset of the media library, structured as a Directed Acyclic Graph (DAG) of operators."



Collection: Definition

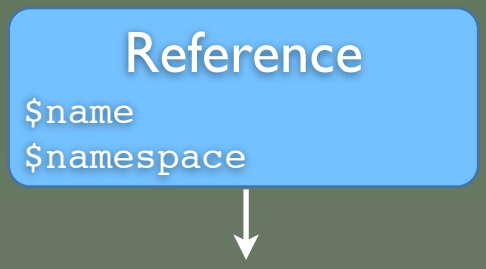
"A collection is a representation of a subset of the media library, structured as a Directed Acyclic Graph (DAG) of operators."



Source Operators

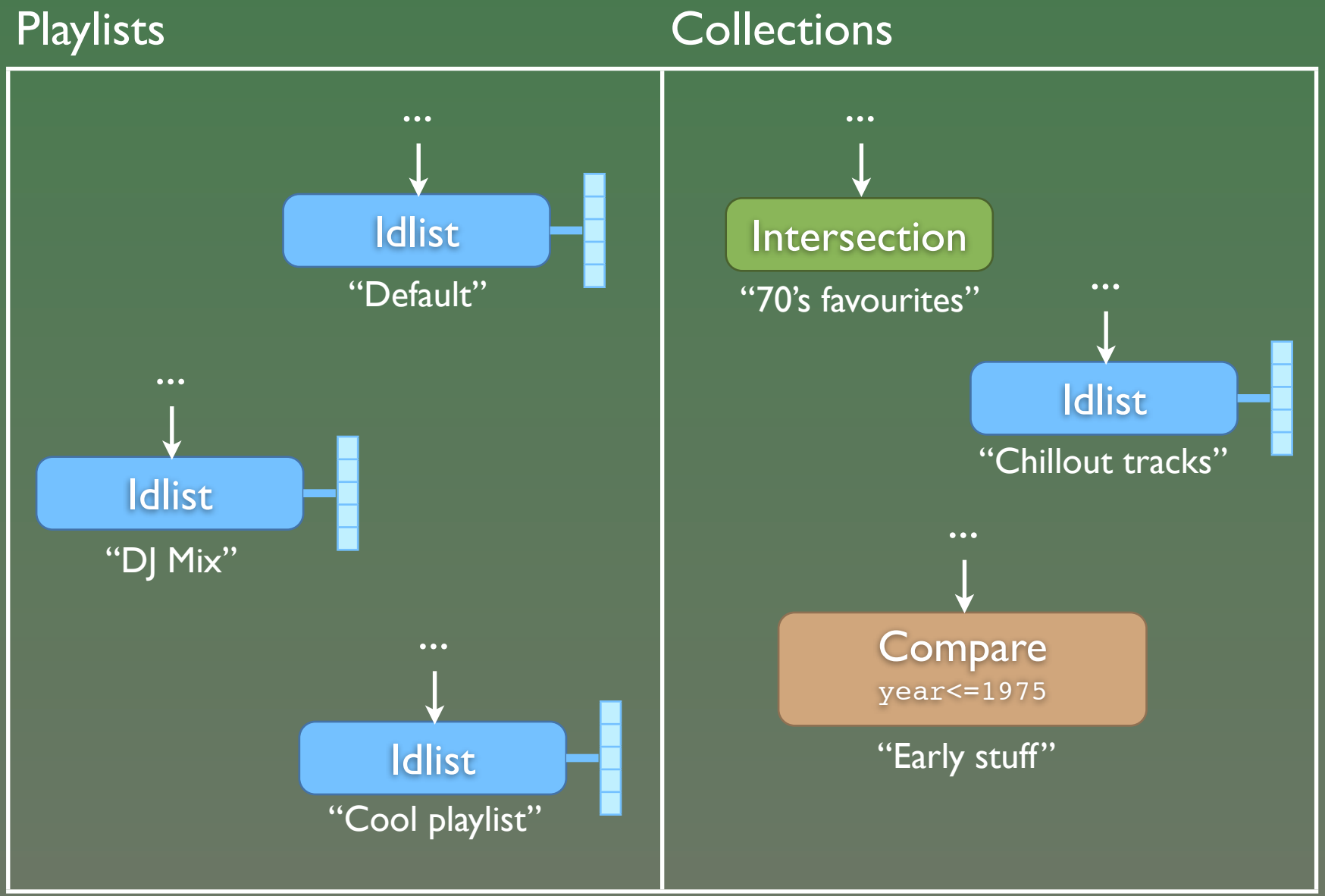
All media in the Media Library

Static list of media (by id)

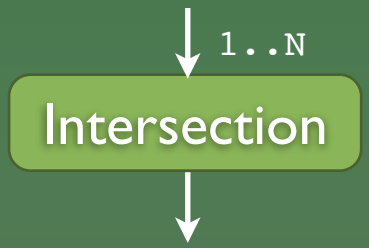


All media in the saved collection `$name` in `$namespace`

Saved Collections

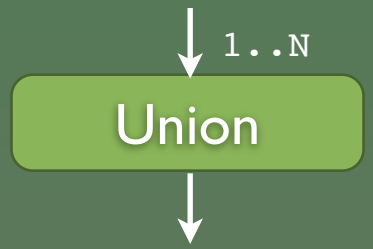


Set Operators



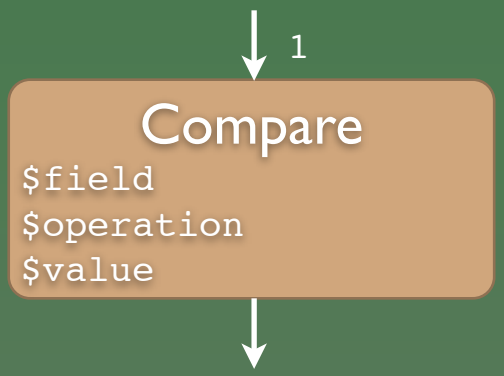
All media present in all the operands

All media present in any of the operands



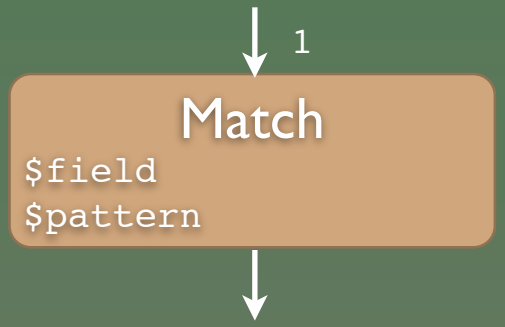
All media not present in the operand

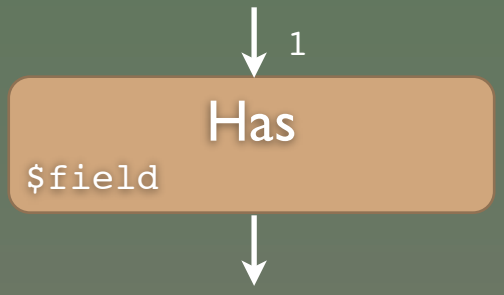
Filter Operators



All media where `$field` satisfies the comparison `$operation` with `$value`
(`$operation`: `<`, `<=`, `=`, `=>`, `>`)

Similar to Compare, but takes a `$pattern` with wildcards (*, *?*) instead of an exact value



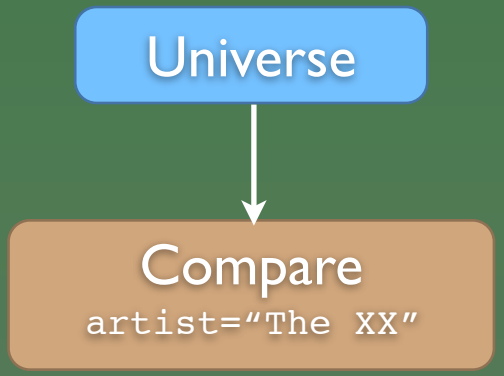


All media that have `$field` set

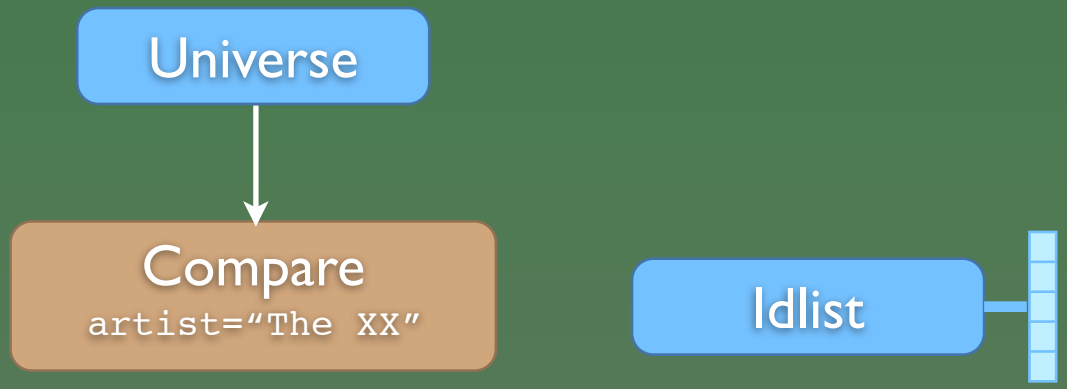
Example Structures

Universe

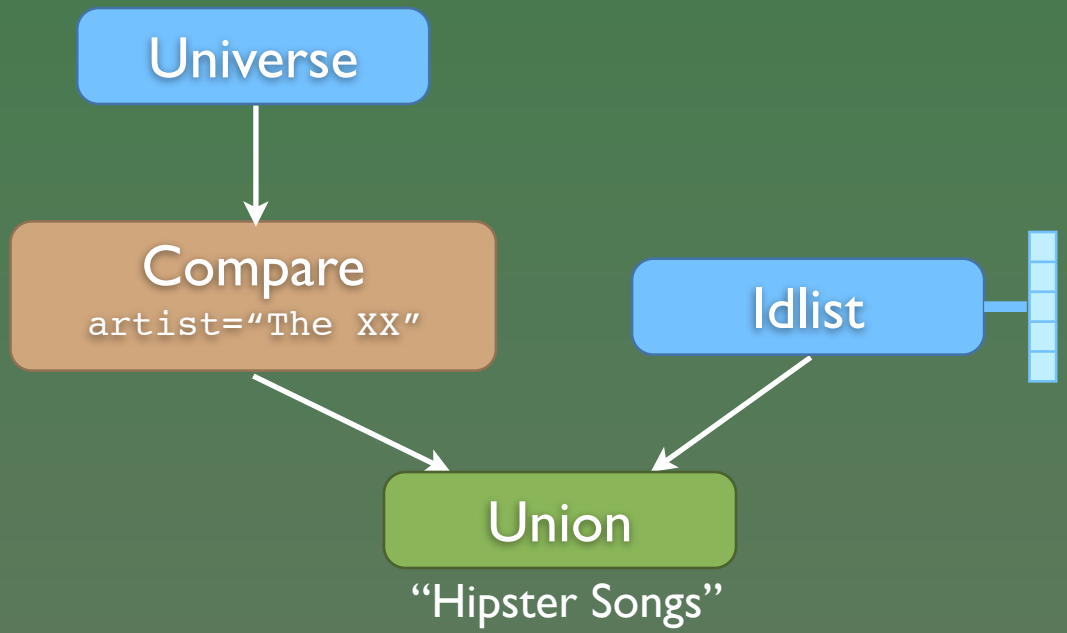
Example Structures



Example Structures

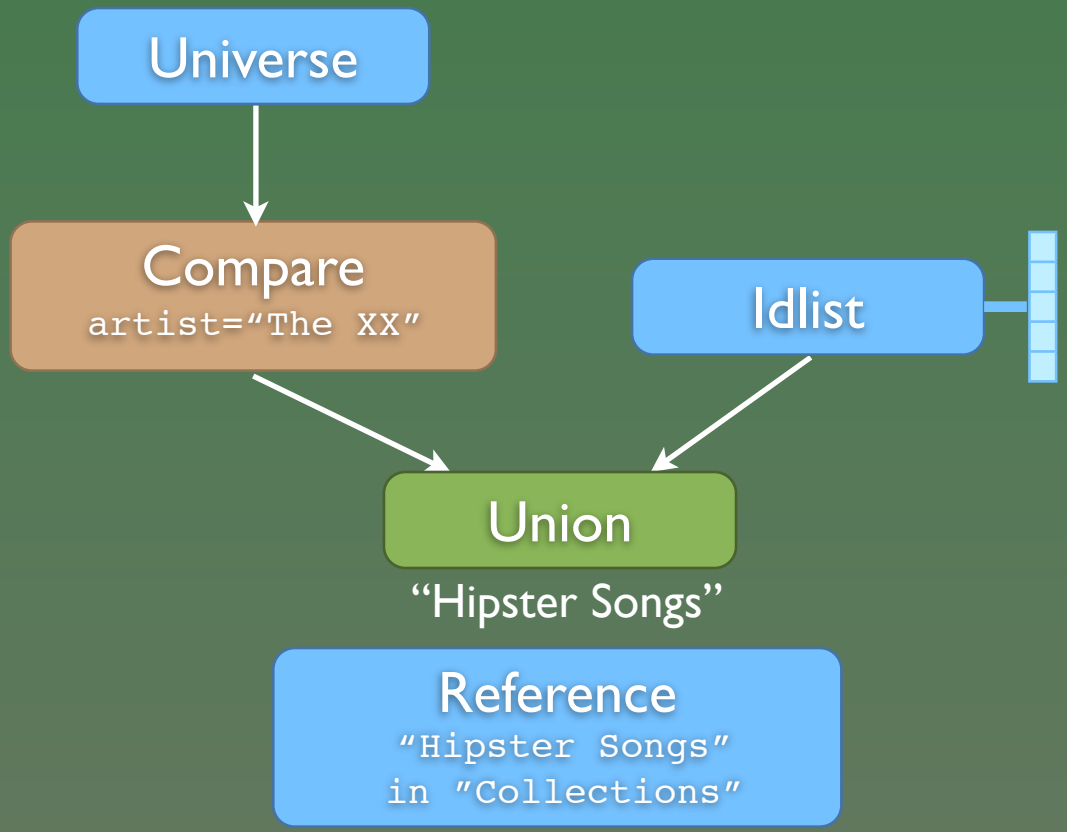


Example Structures

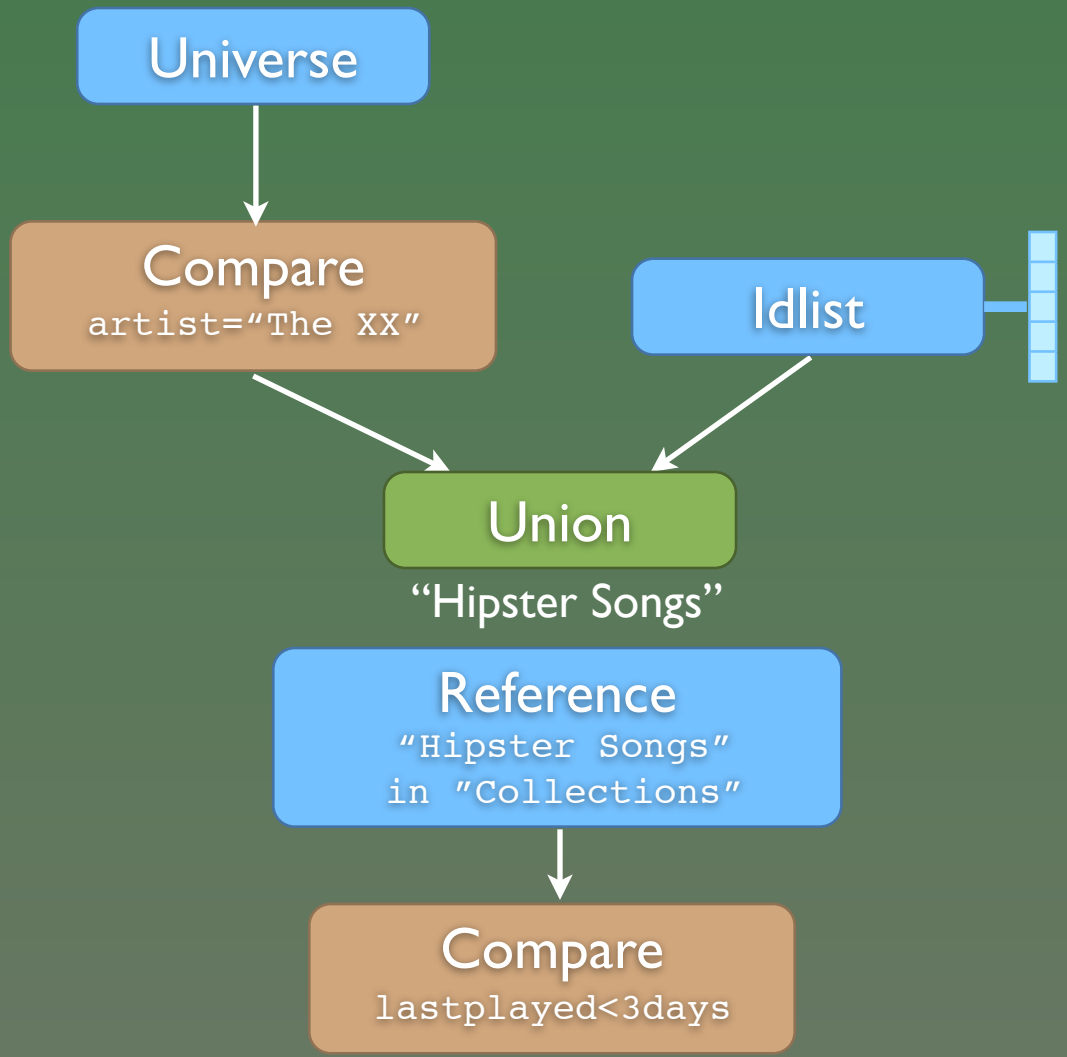


Organize

Example Structures

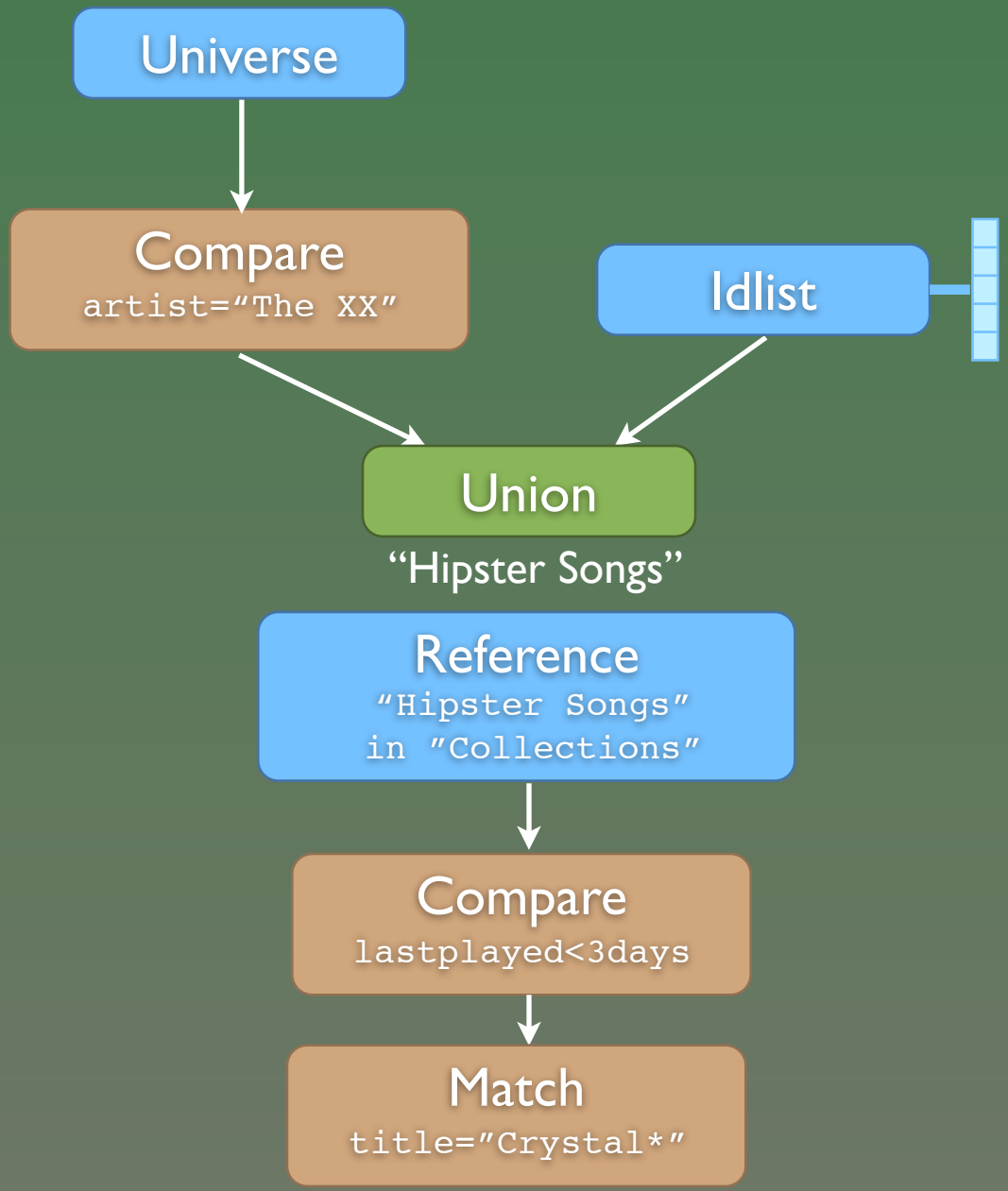


Example Structures



Browse

Example Structures



Search

Remarks

- Used to organize, browse, search, or any combination of those
- Can use different UI approaches to combine them, but all using the collection API
- Dynamic sets but allows manual edits

Collections 2.0

Google Summer of Code 2008 project by Erik Massop (nesciens)

- Support for sources:

id	key	value
42	artist	Britney Spears
42	title	Toxic
42	url	file:///home/...

- Support "medialists" (ordered, with duplicates, as opposed to "mediasets") in the DAG
- Token matching operator
- Fix, improve and optimize querying (more later)

Collections 2.0

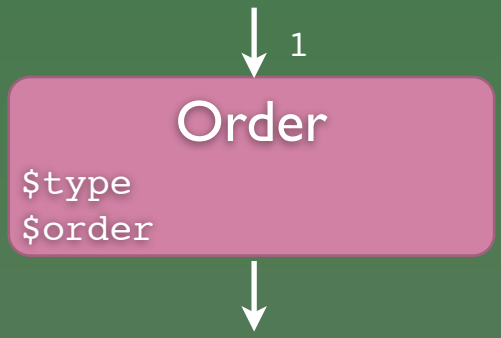
Google Summer of Code 2008 project by Erik Massop (nesciens)

- Support for sources:

id	key	source	value
42	artist	plugins/id3v2	Britny Spaers
42	artist	clients/nycli	Britney Spears
42	title	plugins/id3v2	Toxic
42	url	server	file:///home/...

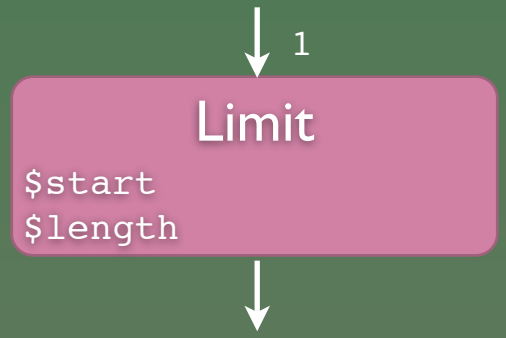
- Support "medialists" (ordered, with duplicates, as opposed to "mediasets") in the DAG
- Token matching operator
- Fix, improve and optimize querying (more later)

Collections 2.0 Operators



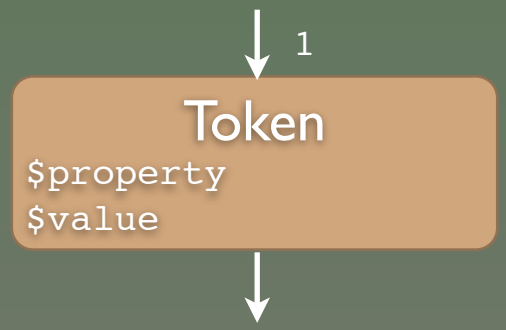
Returns a medialist ordered by `$order`

Returns `$length` entries from the operand, starting at `$start`



Returns the mediaset of the operand (no ordering, no duplicate)

All media where `$property` token-matches `$value` (e.g. "Floyd" matches "Pink Floyd")



3. Usage

"how, then?"

Command-line Interface

Collections are exposed in *nycli* (the new XMMS2 CLI) through

collection patterns

e.g.:

- `Idioteque`
- `Pink Floyd Echoes`
- `artist:“Sufjan Stevens” l:Illinois`
- `(genre:Rock OR genre:Pop/Rock) date<1980`
- `title~usa url:*ogg +tag.favourite in:Awesome`

Collection Patterns

- API helper function to parse patterns into collection structures
- Supported in multiple *nycli* commands:
  - `search <pattern>`
  - `add <pattern>`
  - `jump <pattern>`
  - `coll create <name> <pattern>`
  - `etc.`
- More generally: share a common syntax across clients (GUI, CLI, etc)

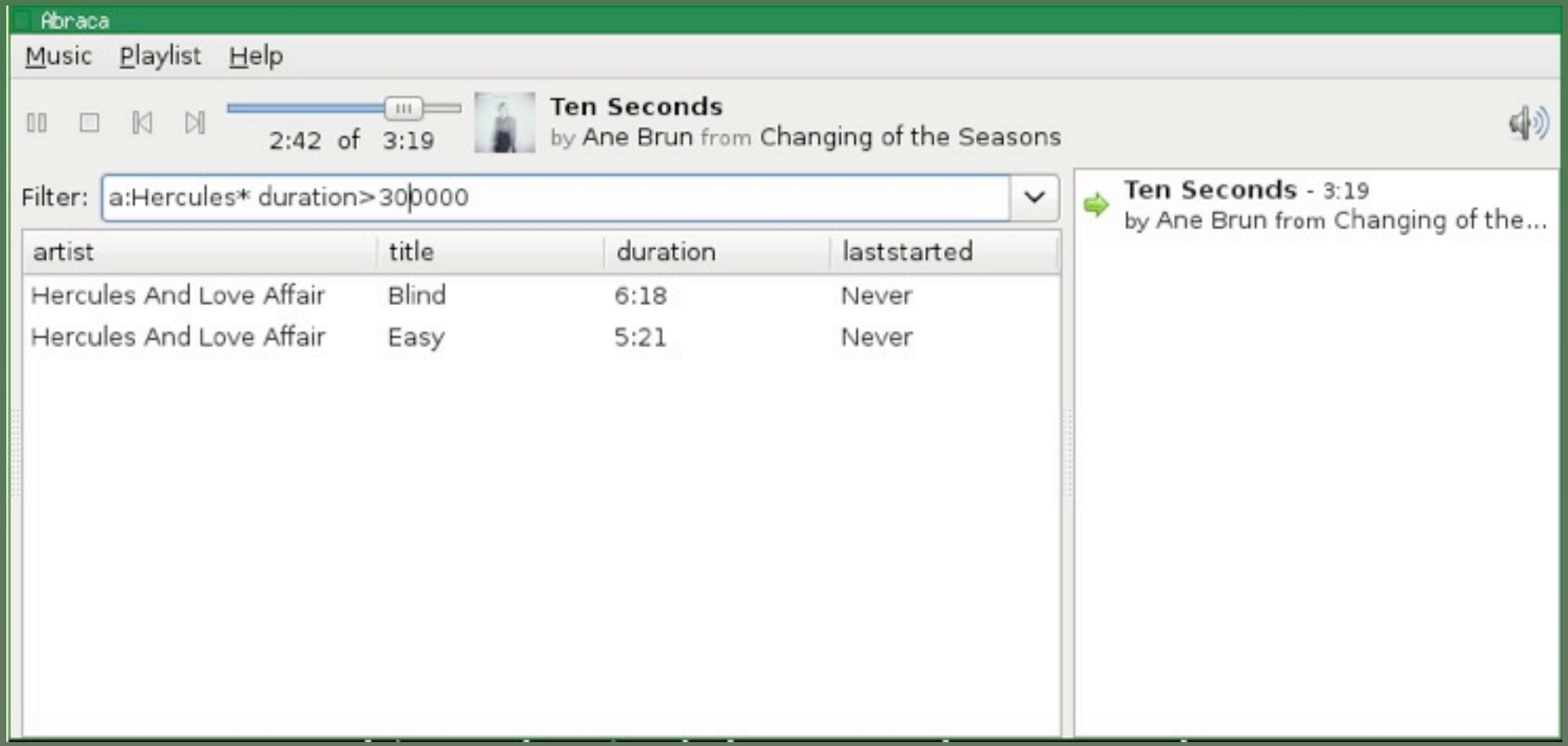
Note: NOT a bijection with collection structures

Collection Patterns

```
seb@crookshanks:~
seb@crookshanks ~ $ nyxmms2 search a:Hercules* duration\>300000 # i.e. >5min
--[Result]-------------------------------------------------------------------
id   | artist                    | album                     | title
1577 | Hercules And Love Affair | Hercules And Love Affair | Blind
1579 | Hercules And Love Affair | Hercules And Love Affair | Easy
-----------------------------------------------------------[Count:     2]--
seb@crookshanks ~ $
```

nycli (new official CLI)
http://xmms2.sf.net/

Collection Patterns



abraca

http://abraca.xmms.se/

Collection Structure API

```
all = coll_universe()
m_floyd = coll_match(all, field="artist",
                     operation="="
                     value="Pink Floyd")
m_early = coll_match(m_floyd, field="year",
                     operation="<",
                     value="1975")
m_static = coll_idlist([42,1337,666])
m_join = coll_union(m_early, m_static)
```

Collection (1.0) Query API

```
coll_query_ids(conn, coll, order[],
               limit_start, limit_len);
```

=> List of object ids matched by `coll`, ordered by the list of properties `order`, optionally restricted to a range.

```
coll_query_infos(conn, coll, order[],
                 limit_start, limit_len,
                 fetch[], group[]);
```

=> Dict of properties (selected by `fetch`) of objects matched by `coll`, ordered by the list of properties `order`, optionally restricted to a range and grouped.

Collection Query Example

```
coll_query_ids(conn, m_join,
                ['artist','album','tracknr']);
=> [412, 413, 414, 323, 5454, 4234, ...]


coll_query_infos(conn, m_join,
                  ['artist','album','tracknr'],
                  0, 50,
                 ['artist', 'title']);
=> [{artist: 'Pink Floyd', title: 'Eclipse'},
    {artist: 'Pink Floyd', title: 'Echoes'},
    {artist: 'Britney Spears', title: 'Toxic'},
    ...]
```

Collection Query Example

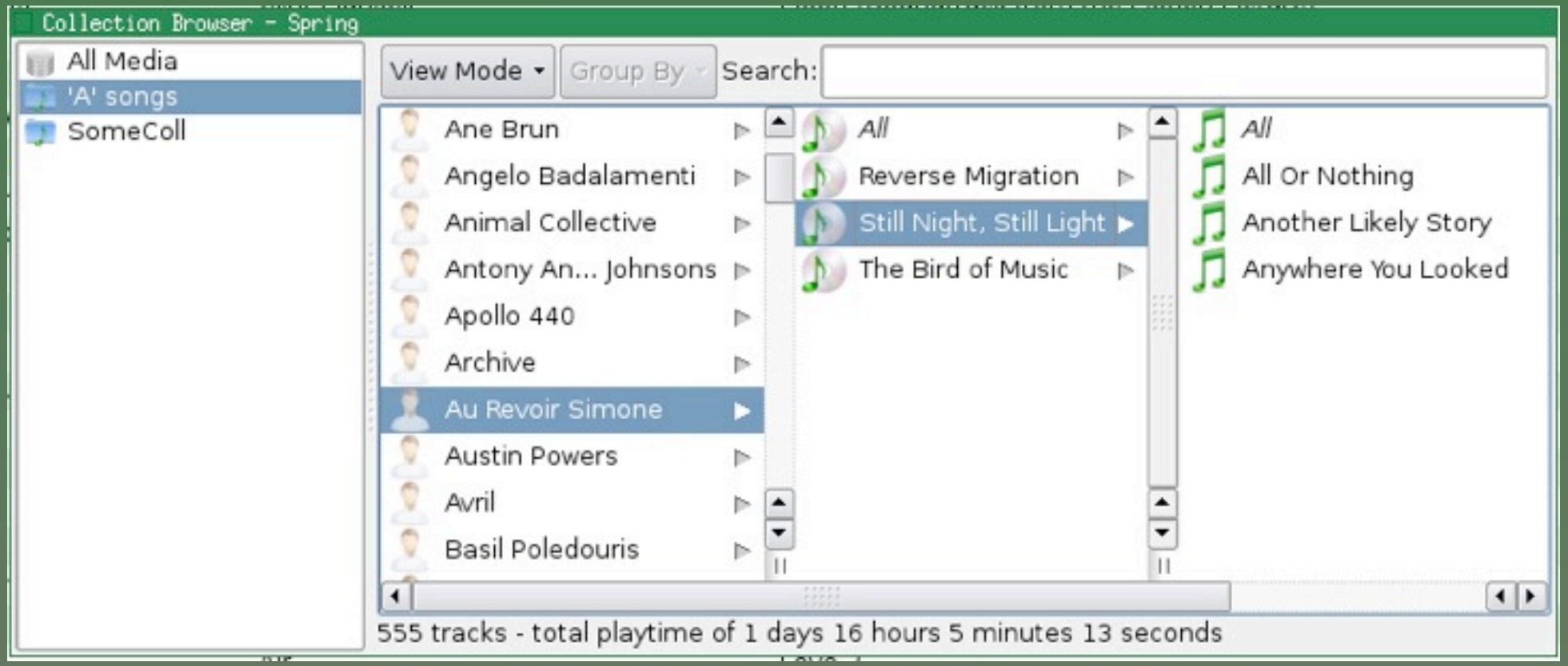
```
coll_query_infos(conn, m_join,
                 ['artist','album'],
                 0, 50,
                ['artist','album'], ['album']);
=> [{artist: 'Pink Floyd', album: 'Meddle'},
    {artist: 'Britney Spears', album: 'Foo'},
    ...]
```

Collection Server API

Save collections in the XMMS2 server:

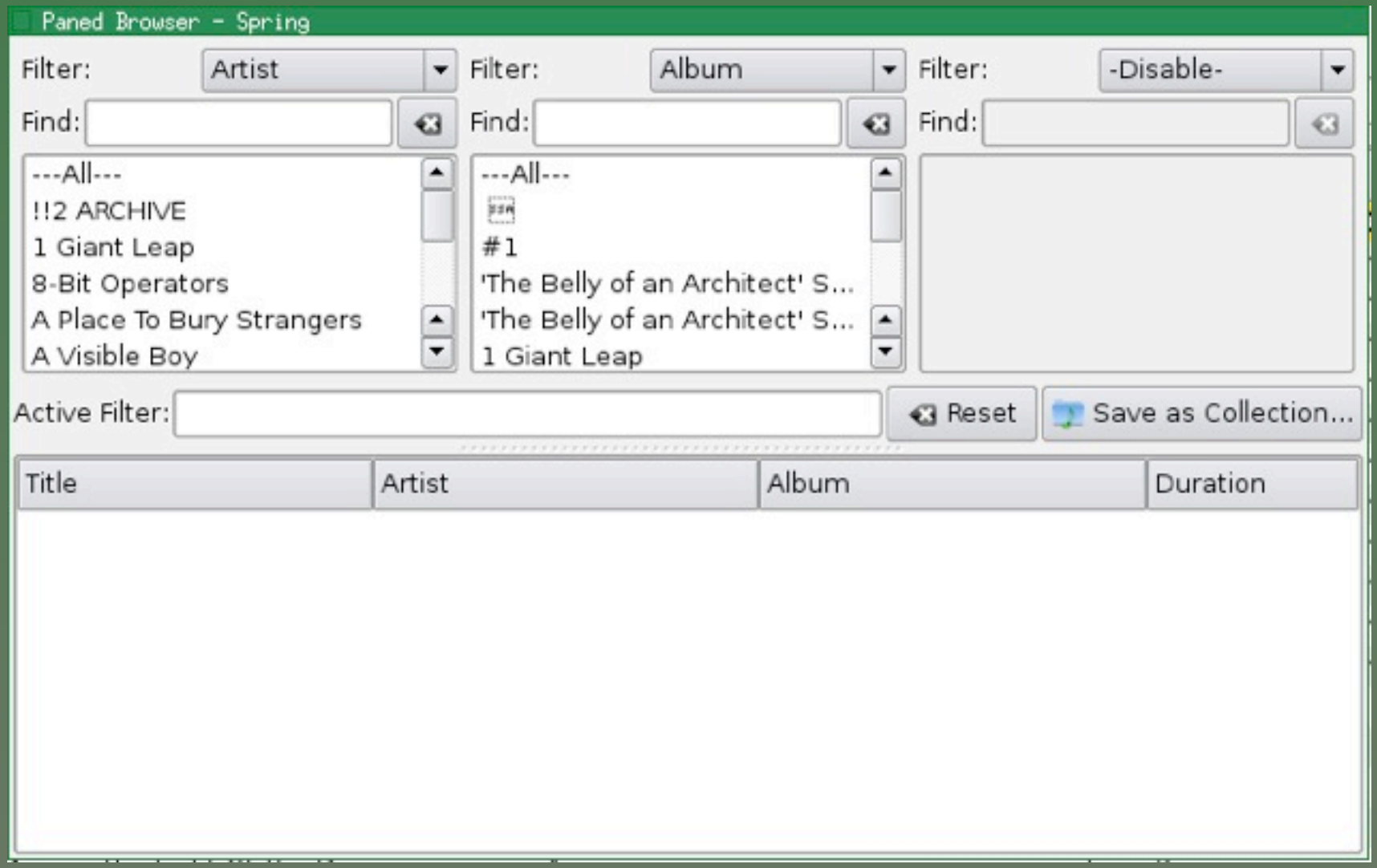
```
xmmsc_coll_save(conn, coll, name, namespace);
xmmsc_coll_get(conn, name, namespace);
xmmsc_coll_list(conn, namespace);
xmmsc_coll_rename(conn, name, to, namespace);
xmmsc_coll_remove(conn, name, namespace);

xmmsc_coll_find(conn, media_id, namespace);
```

Example GUIs



Spring
http://my-trac.assembla.com/spring/

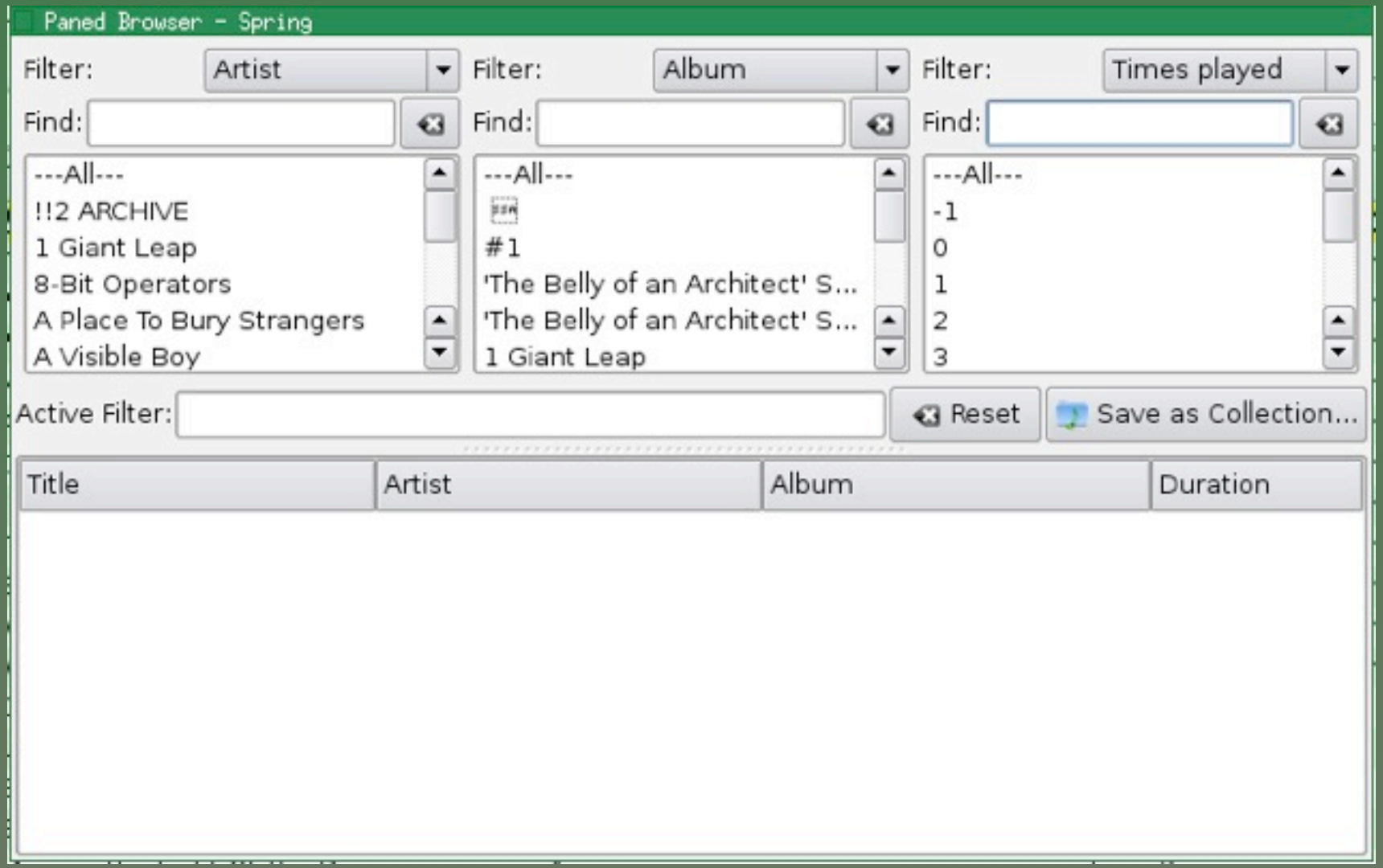
Example GUIs



Spring

http://my-trac.assembla.com/spring/

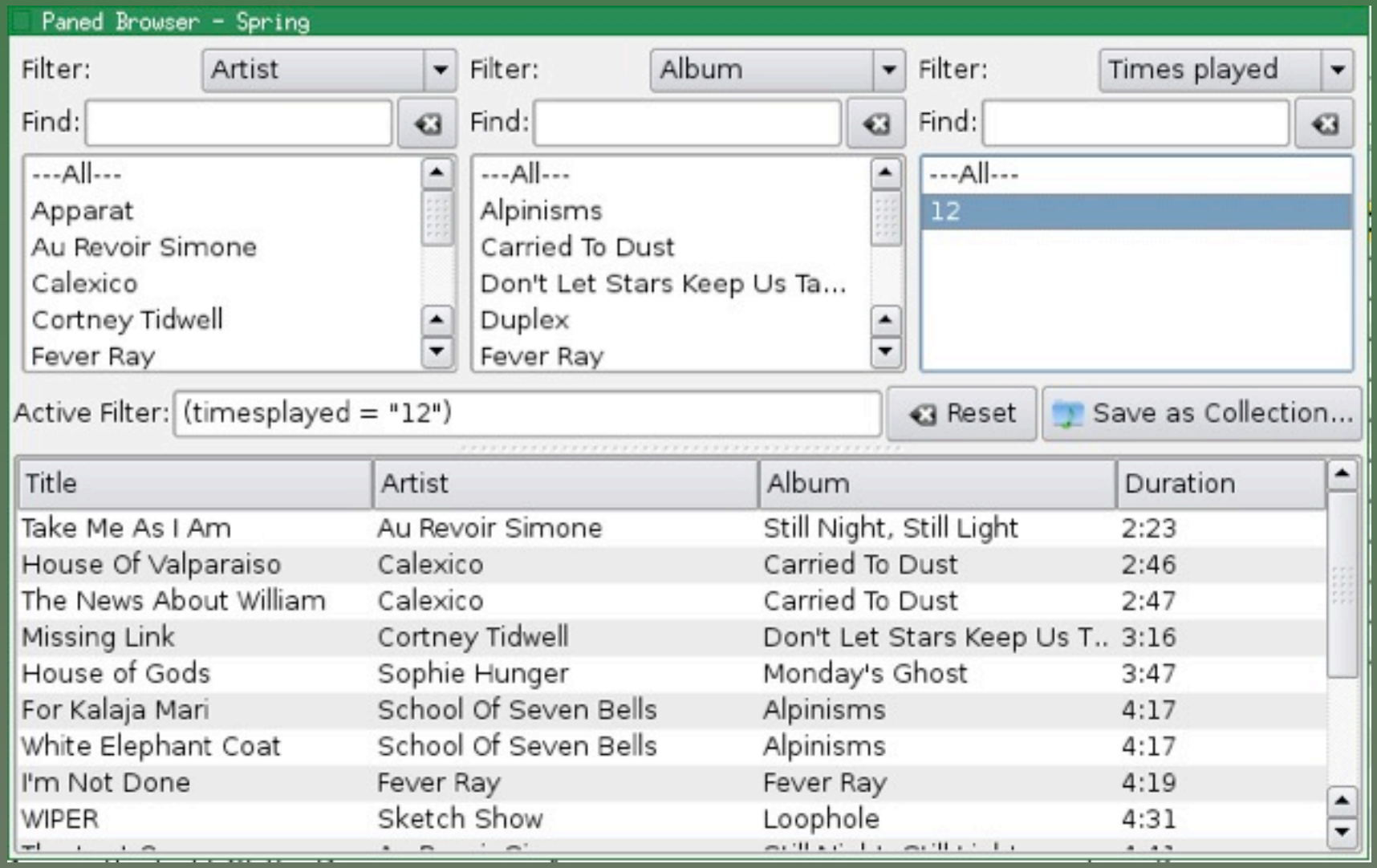
Example GUIs



Spring

http://my-trac.assembla.com/spring/

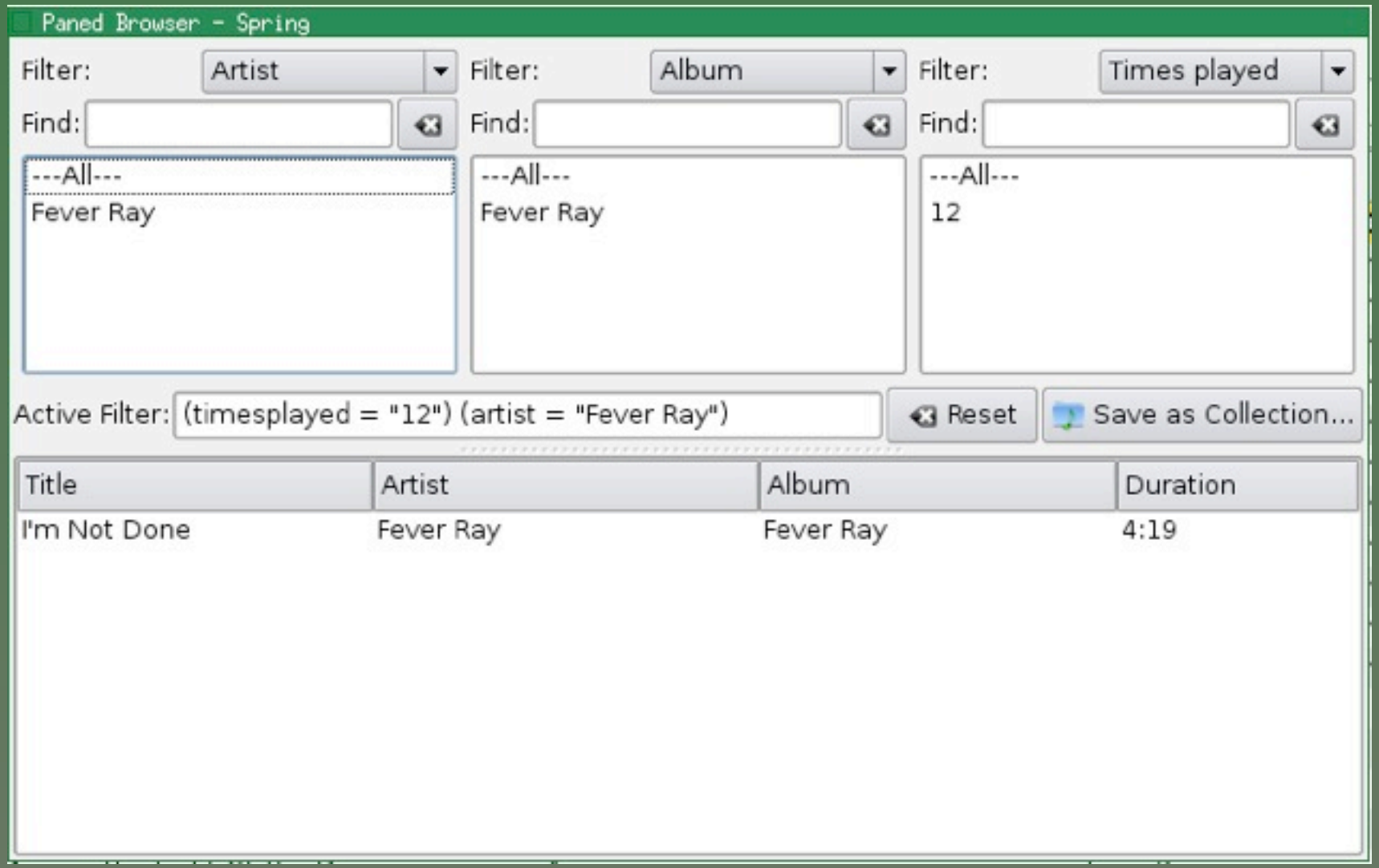
Example GUIs



Spring

http://my-trac.assembla.com/spring/

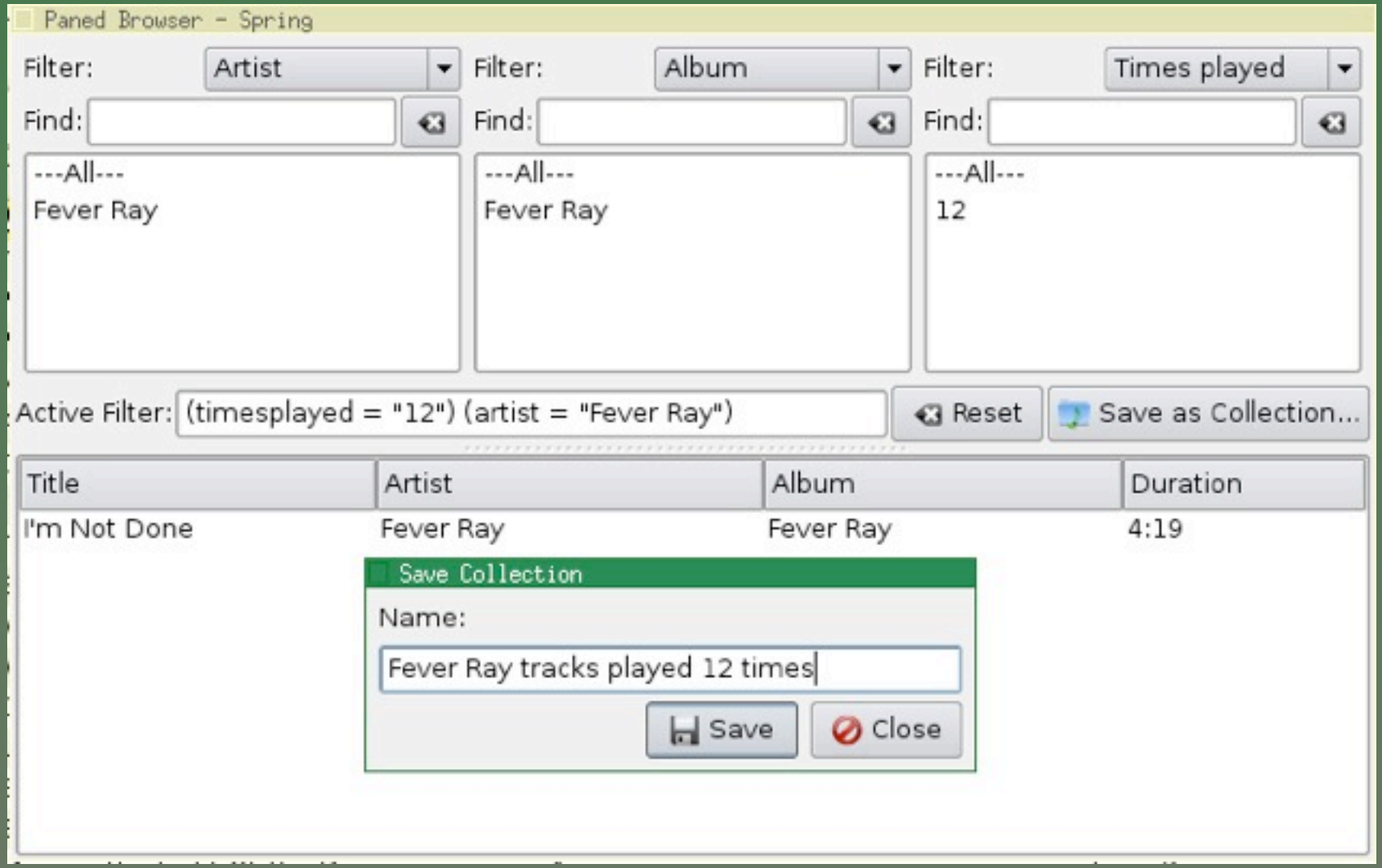
Example GUIs



Spring

http://my-trac.assembla.com/spring/

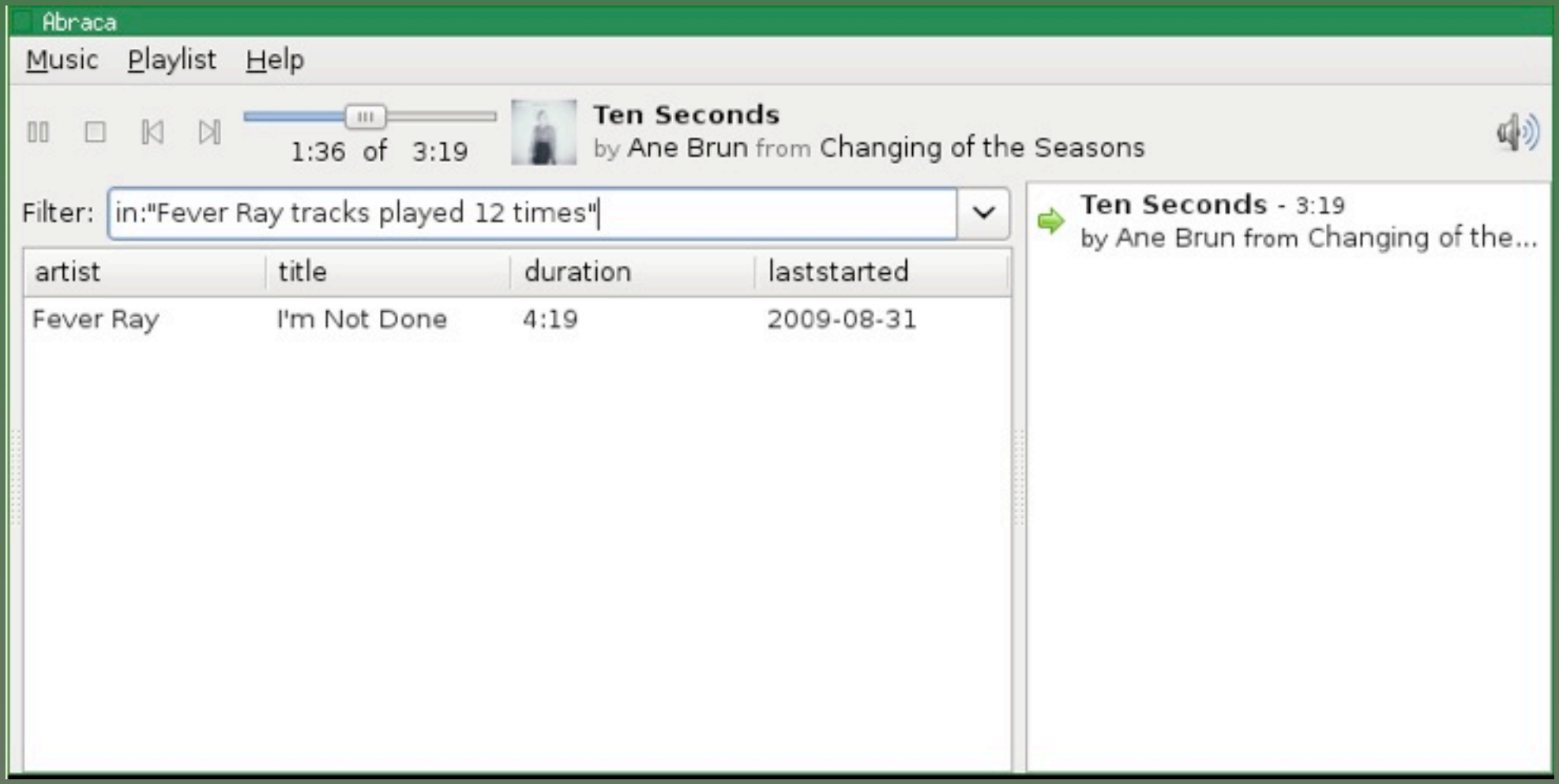
Example GUIs



Spring

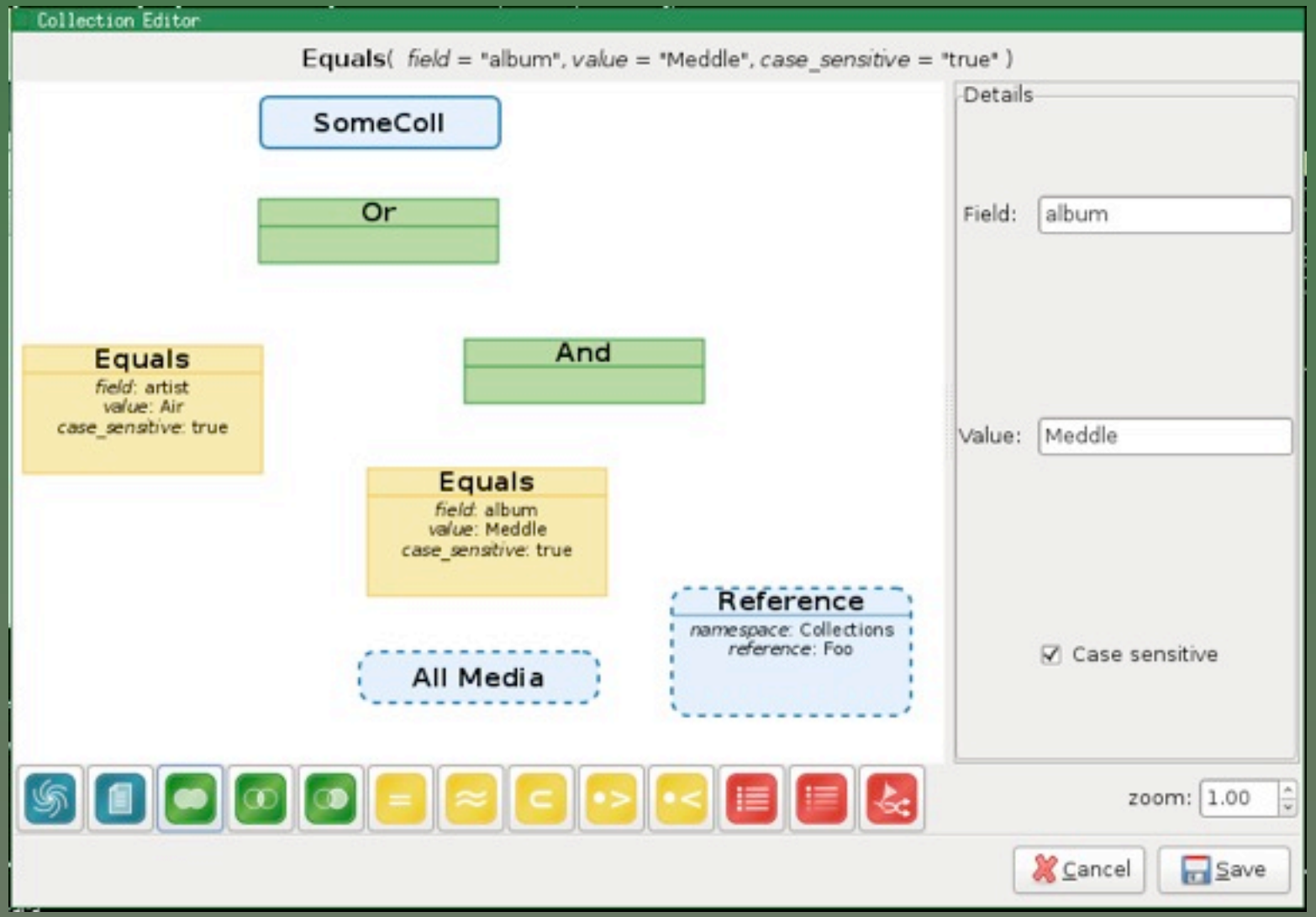
http://my-trac.assembla.com/spring/

Example GUIs



abraca
http://abraca.xmms.se/

Example GUIs



Etude

http://code.google.com/p/etude-music-player/

4. Future

"fair enough, what's next?"

Collections 2.0

Finish & Merge:

- New querying API
- Finalize support for sources
- Performance optimization
- More tests

Coll 2.0: Advanced Queries

Current querying too limited:

- No count
- No custom aggregation function
- No variable in operators (lastplayed<NOW-1day)
- Simplistic grouping, always returns flat tuples

=> Collections 2.0 will allow specifying the "cluster structure" to return.

Not unlike Freebase querying, API still under works.

New Uses/GUI Front-Ends

- Save filtered views, load and edit them
- Organize using "shelf analogy":
  - spatial
  - drag entities as filters (e.g. artist, tag) or atomically (e.g. selection of tracks)
- New & custom views, e.g. on a timeline, by properties, etc.
- Share across XMMS2 server instances (à la XSPF)

S4

(Storage System for Short Strings)

- New XMMS2 media library backend, optimized for our data model
- Uses collections as native query language
- Doesn't support all operators, only works with sets (selection of properties, grouping, ordering done externally)

=> brings collections to the world, as a generic interface for browsing, searching & organizing collections of objects!

5. Conclusion

"is that it?"

Conclusion

- Simple common abstraction of searching, browsing and organization
- Flexible: can be modified, refined, composed
- Powerful API for building UIs to manipulate a large database of information
- Works well for XMMS2
- Could be applied to other collections of objects

Questions?

http://xmms2.sf.net/