mediaSleeve

version 0.1 — 1st June '05

mediaSleeve has been conceived as a basic **service discovery protocol** for annotations and artwork from internet radio streams. It is intended for implementation by streaming providers, radio hosts, and playback clients. It can be used to deliver information relating to the stream to the listener, however the design is not specific to this use and may be extended in other ways.

Current status

This is a work-in-progress planned for trial implemention in both a desktop client (OnDeck – OS X) and an internet radio stream (Atmosphere).

Further information

Feedback, suggestions and greater participation are welcomed! Please visit the **mediaSleeve Google group**.

The author

Jacob Jay
< jjay [at] verse [dot] org >

Contents

Terminology

- **Content** refers to a stream or the end-function of a URL.
- **Server** refers to an ISP or server from which content is served or hosted (a service provider).
- **Author** refers to the originator of content (a radio DJ).
- The **Resource** refers to the mediaSleeve function as implemented on a server.
- The **Service** refers to the data provided by the resource.

Service Discovery

A common endpoint

http://example.com**/mediasleeve/**

This **resource** must be implmented at the domain (or site) of the **author** and should be implemented at the domain of the **server** (if different).

> The URL may be aliased using an HTTP redirect (i.e. to another URL such as http://example.com/api/sleeve/). Client implementations must respect and follow HTTP redirects. The mediaSleeve URL is lower-case.

The common endpoint provides the ability to efficiently discover the mediaSleeve **resource**. If the **server** supports the **service**, retrieval of the resource will return an **HTTP 200** (or redirect) and the **service** response (see below), if the service does not support mediaSleeve, it will return an **HTTP 404**.

> HTTP GET may not be used for any other purpose than service discovery, however HEAD is recommended for discovery when not combined with a query.

Where the mediaSleeve **resource** cannot be provided by the **server**, the following methods may be provided by the **author**.

Streams

The domain or path of the author's site should be written in the stream name as the last element following a -- delimiter. Clients should parse the name for this token and then retrive the resource at that address.

> My Streaming Radio: 24/7 Electronica **-- example.com**

Other methods

Clients may define their own methods to handle content not suppored by the above discovery methods, such as a lookup directory mapping **content** URLs to **author** domains.

Protocol

mediaSleeve uses **HTTP** on port **80** and employs a **REST**/**RSS** model in which queries are **POST**ed to the resource as **application/x-www-form-urlencoded** and responses returned as **XML**.

> Linebreaks are expected to be UNIX-style using the LF character (ASCII 10). Text endcoding is presumed to be UTF-8.

Standard response

The XML response format to be returned by the core methods when a protocol does not otherwise specify format (i.e. REST), is as follows:

> The XML identifier is optional, and the entire block may be placed in a comment if desired (ideally in the head of the document). Every element of the method response(s) must appear on individual lines and no changes may be made to their spacing or the ordering of attributes (elements can be reordered). This structure is used to ease parsing with substrings and tokens in non-XML capable clients. This response may be embedded in HTML inside a comment if necessary.

Queries

> Custom parameters may be included in queries, however these should not alter the core functionality provided by the method. Parameter names starting 'm-' are reserved. Custom elements may be included in responses but the names of these may only start with 'x-', all other names are reserved. Custom attributes may not be included.

Query

A client will post the following fields to the resource.

- **url** – the URL of the content
- **ask** – an optional list of data type to be returned

Response

The resource will return XML:

```
<?xml version="1.0"?>
<mediasleeve>
<notify name="Untitled" id="123" href="http://example.com/author" />
<content url="rtsp://example.com/example.pls" date="UTC" name="example"
href="http://example.com" refresh="UTC" description="" />
<artwork url="http://example.com/artwork.jpg" href="http://amazon.com/" />
<album name="Untitled" creator="Unknown" href="http://amazon.com/" />
<track name="Untitled" creator="Unknown" href="http://amazon.com/" />
<programme name="Untitled" creator="Unknown" href="http://example.com/author" />
</content>
</mediasleeve>
```

- **refresh** – number of seconds in which to refresh the data
- **href** – a URL which can be provided as a link for the respective data
- **creator** – the artist, author or other originator of the content
- **programme** – used in cases where the content is a segment of a show or no other d
- **notify** – a message to be broadcast to users, id is any arbitrary number (prevents users from getting the message repeatedly)
- **ath** – a list of supported authentication identifiers (optional)
- **cip** – a list of supported encryption identifiers (optional)
- **key** – a session token for authentication (optional)
- **doc** – the path or URL to API documentation for humans (optional)

Function

A service should return an **act** value of NO in all cases except when it recieves a valid login query (**username** parameter is present), in which case it should return OK, or ERR as appropriate. Under special circumstances (outages, disabled account) a service may return an **act** of URL and with a specific URL for the user to visit for more details in place of the usual **res** value.

A client application may carry out an initial login with its preferred compatability options, or with none specified (i.e. the defaults of REST protocol, ST authentication, NO encryption) and only parse the service compatability values to retry the login appropriately when the **act** value returned is NO. If the **act** returned is URL the client should display that URL to the user and ask them if they wish to open that URL, otherwise the **act** should be treated as ERR.