Navicast Output API Documentation

iPhones
iPods & iPads

Web players

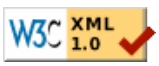
Podcasts & iTunes

Mobile phones

RSS-feeds

Custom CMS
Custom players
Custom integration

Output API



JSON &
JSONP

Table of Contents

Introduction 3

Output API Error Codes 3

podcast.php 4

- XML/Podcast (sample) 4
- XML/Podcast (mode=1) (sample) 5
- Custom RSS/Podcast elements 5

XSPF.php 6

- XSPF (sample) 6

XMLMovieList.php, JSONMovieList.php, JSONPMovieList.php 7

- Password protected movies 7
- Theft protected movies 8
  - Theft protection and security 8
- XMLMovieList (sample) 8
- JSONMovieList (sample) 9
- JSONPMovieList (sample) 9

XMLMovieInfo.php, JSONMovieInfo.php, JSONPMovieInfo.php 10

getXMLtkt.php, getJSONtkt.php, getJSONPtkt.php 10

- XMLtkt (sample) 11
- JSONtkt (sample) 11
- JSONPtkt (sample) 11
- The “secret” channel key 11

getMovie.php 11

getMoviePlaylist.php 12

MoviePlaylist (sample) **12**

Javascript, AJAX and CORS support **13**

Introduction

The Navicast Output API consists of a number of methods that return RSS/Podcasts feeds, XSPF playlists or XML, JSON/JSONP formatted data structures. In contrast to the Input API the Output API is focusing on providing small standalone portions of the Navicast data structure at a time. To view a movie we do not need to know the whole Navicast data structure, we only need to know about the movie in question.

The Output API is designed to follow standards, be straight forward and simple to use for people that are familiar with RSS/Podcast feeds, XSPF playlists or javascript and XML (AJAX) as well as JSON. However, when dealing with theft protected and password protected movies it will require some more knowledge about how Navicast works.

As of Navicast 4 we should be able to set limits on the total number of movies a channel can show in any 30 day period. This limit is set in Navicast studio. This affects the **Output API**. If a viewer tries to watch a movie that would exceed the 30 day hitlimit then the movie would not be shown, instead a message “Maximum viewer quota exceeded, please return again later”.

```
<\?xml version="1.0" encoding="UTF-8"?\>
<Result id="601"”>
        Request Rejected! Maximum viewer impressions exceeded.
</Result>
```

Output API Error Codes

Error codes from 1-199 are reserved to Navicast Internal.
Error codes from 200-599 are reserved to the Input API.
Error codes from 600-999 are reserved to the Output API.
Error codes from 1000 and above are general free.

- Error 0 “Success” **(no error)**
- Error 1 “Server Error! Could not connect to MySQL.”
- Error 2 “Server Error! Could not connect to the database.”
- Error 600 “Rejected! Missing required parameter.”
- Error 601 "Request Rejected! Maximum viewer impressions exceeded.”
- Error 602 “Unauthorized! Channel and/or category password not valid.”
- Error 603 “Unauthorized! Category password not valid.”
- Error 604 “ Not found.”
- Error 605 “ Unauthorized! Key not valid.”

podcast.php

This document returns a combined RSS/Podcast feed. Theft protected and password protected movies will not be shown in the RSS/Podcast list.

http://en.wikipedia.org/wiki/RSS
http://en.wikipedia.org/wiki/Podcast

Required GET parameters
- ch
  The channel ID.

Optional GET parameters
- cat
  The category ID.
- limit
  Limits the result list to the given number (If not set defaults to 50).
- speed
  Bandwidth indicator, 1=low, 2=medium, 3=high (2 is default)
- mode
  if set to 1 gives simplified/alternative formating without link to player.

Optional POST parameters
- ch_pwd
  The channel password in case the channel is password protected.
- cat_pwd
  The category password in case the category is password protected.

XML/Podcast (sample)

<?xml version="1.0" encoding="UTF-8" ?>
<rss xmlns:itunes="http://www.itunes.com/dtds/podcast-1.0.dtd" xmlns:atom="http://www.w3.org/2005/Atom" version="2.0">
<channel>
<title>First Channel</title>
<description>Created with eZender, modified by studio</description>
<link>http://x/player.php?ch=00001</link>
<atom:link href="http://x/podcast.php?ch=1&limit=1" rel="self" type="application/rss+xml" /><language>en-us</language>
<itunes:category text="TV & Film"/>
<itunes:explicit>clean</itunes:explicit>
<itunes:owner><itunes:email>info@barsark.com</itunes:email></itunes:owner>
<item>
<title>Win mp4 with audio</title>
<description>

 <img src="http://x/getThumbnailImage.php?ch=00001&aid=781" width="160"></description>
<pubDate>Mon, 27 Apr 2009 21:47:58 +0200</pubDate>
<enclosure url="http://x/movies/781.mp4" length="0" type="video/mp4" />
<link>http://x/player.php?ch=00001&id=781</link>
<guid>http://x/movies/781.mp4</guid>
<itunes:image href="http://x/getThumbnailImage.php?ch=00001&aid=781" />
<itunes:subtitle> - </itunes:subtitle>
<itunes:summary></itunes:summary>
<itunes:duration>00:00:12</itunes:duration>
</item>

```
        </channel>
</rss>
```

XML/Podcast (mode=1) (sample)

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss xmlns:itunes="http://www.itunes.com/dtds/podcast-1.0.dtd" xmlns:atom="http://www.w3.org/2005/Atom"
version="2.0">
        <channel>
                <title>First Channel</title>
                <description>Created with eZender, modified by studio</description>
                <link>http://x/player.php?ch=00001</link>
                <atom:link href="http://x/podcast.php?ch=1&amp;limit=1" rel="self" type="application/rss
+xml" /><language>en-us</language>
<itunes:category text="TV &amp; Film"/>
<itunes:explicit>clean</itunes:explicit>
<itunes:owner><itunes:email>info@barsark.com</itunes:email></itunes:owner>
                <item>
                        <title>Win mp4 with audio</title>
                        <description>&lt;br&gt;&lt;br&gt; &lt;img src="http://x/getThumbnailImage.php?
ch=00001&amp;aid=781" width="160"&gt;</description>
                        <pubDate>Mon, 27 Apr 2009 21:47:58 +0200</pubDate>
                        <enclosure url="http://x/getThumbnailImage.php?ch=00001&amp;aid=781"
length="0" type="image/jpeg" />
                        <link>http://x/movies/781.mp4</link>
                        <itunes:image href="http://x/getThumbnailImage.php?ch=00001&amp;aid=781" />
                        <itunes:subtitle> - </itunes:subtitle>
                        <itunes:summary></itunes:summary>
                        <itunes:duration>00:00:12</itunes:duration>
                </item>
        </channel>
</rss>
```

Custom RSS/Podcast elements

In the channel Comment field in Navicast studio you may enter custom elements that will be added to the RSS/Podcast channel element. To do this you need to first enter “XmL” and then your properly formatted custom xml elements. An example could look like this.

```
XmL<language>en-us</language>
<itunes:category text="TV &amp; Film"/>
<itunes:explicit>clean</itunes:explicit>
<itunes:owner>
        <itunes:email>email@xyz.com</itunes:email>
</itunes:owner>
```

For more information on customizing a podcast visit:
http://www.apple.com/itunes/whatson/podcasts/specs.html#rss

XSPF.php

This document returns a XSPF (Spiff) play list. Theft protected and password protected movies will not be shown in the XSPF list.

http://en.wikipedia.org/wiki/XSPF

Required GET parameters
- ch
  The channel ID.

Optional GET parameters
- cat
  The category ID.
- limit
  Limits the result list to the given number (If not set defaults to 50).
- start
  Where in the list to start returning movies. Useful for paging.
- find
  URL encoded string. Finds and returns a list of matching movies.
- speed
  Bandwidth indicator, 1=low, 2=medium, 3=high.

Optional POST parameters
- ch_pwd
  The channel password in case the channel is password protected.
- cat_pwd
  The category password in case the category is password protected.

XSPF (sample)

```
<?xml version="1.0" encoding="UTF-8" ?>
<playlist version="1" xmlns="http://xspf.org/ns/0/">
        <title>First Channel</title>
        <trackList>
                <track>
                        <meta rel="type">video</meta>
                        <identifier>781</identifier>
                        <title>Win mp4 with audio</title>
                        <creator>1</creator>
                        <annotation> </annotation>
                        <info>http://x/player.php?ch=00001&amp;id=781</info>
                        <duration>12</duration>
                        <image>http://x/getThumbnailImage.php?ch=00001&amp;aid=781</image>
                        <location>http://x/getMovie.php?mid=781&amp;br=256</location>
                </track>
        </trackList>
</playlist>
```

XMLMovieList.php, JSONMovieList.php, JSONPMovieList.php

These documents return an XML, JSON or JSONP list of all the available movies in a channel, category or search. All movies, including theft protected and password protected, are returned hence you will need to handle both getting and providing tickets for theft protected movies and providing correct passwords before the theft and password protected movies can be played.

By default the returned list is limited to listing 10 movies, but you can specify the number of movies with the “limit” parameter. The maximum number of movies that will be included in the list is 50.
If you need to browse hundreds or thousands of movies you should use some kind of pagination using the “limit” and “start” parameters.

The default ordering of the movies are by sortorder, as set in Navicast Studio, and publication date. If no sortorder is set in Navicast Studio the movies will be sorted with the most resent published movie on top of the list, the second most resent as number two in the list and so on.

Required GET parameters
- ch
  The channel ID.

Optional GET parameters
- cat
  The category ID.
- sort
  Controls how the movies in the list are sorted.
  0 = ordered by publication date and time.
  1 = By random order.
  2 = Hottest yesterday (will not return movies that were not viewed yesterday).
  3 = Hottest today (will not return movies that have not been viewed today).
  4 = By Sortorder, as set in Navicast studio, and publication date.
  5 = Order by all time hits.
- limit
  Limits the result list to the given number, 1...50.
- start
  Where in the list to start returning movies. Useful for paging.
- find
  URL encoded string. Finds and returns a list of matching movies.
- speed
  Bandwidth indicator, 1=low, 2=medium, 3=high.
- callback*
  The name of the callback function in JSONP. Will be callback if not set.
- **FullInfo**
  Returns extra information about the channel and available categories within the channel. Useful to get initial information when first loading a player.

Optional POST parameters
- ch_pwd
  If a channel is password protected you will need to pass the correct channel password here.
- cat_pwd (GET or POST)
  If a category is password protected you will need to pass the correct category password here.

Password protected movies

The movie element in the MovieList holds information about the password protection state of the movie.
If the movie is password protected the movie element tag “m_pwd” will be “1” else “0”. If a movie is password protected in Navicast studio you will not be able to request or view the movie without passing the valid

password at the end of the URL to the movie.
To request, load and play a password protected movie over http you will need to pass the correct password to the getMovie.php document. The password is sent in a GET variable named “m_pwd” and needs to be Base64 encoded when used with getMovie.php. To load and play a movie over rtsp you will need to pass the correct password to the getXMLtkt.php or getJSONtkt.php or getJSONPtkt.php document but here it is not to be Base64 encoded.

Theft protected movies

“Theft protection” is a Navicast feature that is implemented to prevent unauthorized highjacking and hot linking of movies as well as raising the security level regarding movie delivery in general.

If a movie is set to be theft protected in Navicast studio the movie list element tag “tkt” will be “1” else “0”. If a movie is set to be theft protected in Navicast studio you will not be able to request or view the movie without first getting a valid ticket. In this case you will need to get a valid ticket using the getXMLtkt.php document and add that ticket to the http or rtsp movie requests. If you load a movie through http you will need to add a get variable “&tkt=theticket” with the ticket to the end of the url. If you load a movie through rtsp you will need to add the ticket directly to the end of the url. For further information on getting valid tickets see getXMLtkt.php, getJSONtkt.php, getJSONPtkt.php below.

Theft protection and security

The “Theft protection” feature in Navicast does not in any way guarantee that the movie can not be stolen or in other ways distributed without proper authorization. The feature does however take some basic protective measures to make it more difficult to access the movie without permission. Theft protection in combination with password protection will give you a fair security level. The “Theft protection” feature also makes it possible for you to add further security and in that way add and control your own security level.

XMLMovieList (sample)

```
<?xml version="1.0" encoding="UTF-8" ?>
<Navicast type="OutputXMLMovieList" version="0.8" docs="http://www.navicast.net/">
	<Result id="0">Success</Result>
	<Channel id="00001">
		<ChannelName>First Channel</ChannelName>
		<Description>Some text</Description>
		<Reviews active="1"/>
		<Intro active="0"/>
		<Categories>
			<Category id="1" name="Some Category" cat_pwd="0"/>
		</Categories>
		<Movies from="1" to="1" of="59">
			<Movie id="744" m_pwd="0" tkt="1">
				<ExternalID></ExternalID>
				<Hits>110</Hits>
				<Title>Movie Title</Title>
				<Heading>Something</Heading>
				<SubHeading>Else</SubHeading>
				<Body>Some text</Body>
				<Duration>00:01:00</Duration>
				<PublishDate>2009-04-20 10:23:38</PublishDate>
				<PublishBy>5</PublishBy>
				<ThumbNail url="http://x/getThumbnailImage.php?ch=00001&amp;aid=74"/>
				<HTTPURL>http://x/getMovie.php?mid=74&amp;br=256</HTTPURL>
				<RTSPURL>rtsp://stream.barsark.net/4.navicast.net.secure/</RTSPURL>
			</Movie>
		</Movies>
```

```
        </Channel>
</Navicast>
```

JSONMovieList (sample)

{"type":"OutputJSONPMovieList", "version":"0.8", "docs":"http://www.navicast.net/",
"Result":{"id":"0", "Message":"Success"},

        "Channel":
        {
                "id":"00001", "ChannelName":"First Channel",
                "Description":"Created with eZender, modified by studio",
                "Reviews":{"active":"1"},
                "Intro": {"active":"0"},
                "Categories":
                [
                        {"id":"1", "name":"Slideshows", "cat_pwd":"0"}
                ],
                "MovieRange":{"from":1, "to":1, "of":3},
                "Movies":
                [
                        {"id":"744",
                        "m_pwd":0,
                        "tkt":1,
                        "ExternalID":"",
                        "Hits":110,
                        "Title":"1MinutDV.mov",
                        "Heading":"",
                        "SubHeading":"",
                        "Body":"",
                        "Duration":"00:01:00",
                        "PublishDate":"2009-04-20 10:23:38",
                        "PublishBy":"5",
                        "ThumbNail":{"url":"http://x/getThumbnailImage.php%3Fch=00001%26aid=744"},
                        "HTTPURL":"http://x/getMovie.php%3Fmid=744%26br=256",
                        "RTSPURL":"rtsp://stream.barsark.net/x.secure/"}
                ]
        }
}

JSONPMovieList (sample)

callback({"type":"OutputJSONPMovieList", "version":"0.8", "docs":"http://www.navicast.net/",
"Result":{"id":"0", "Message":"Success"},

        "Channel":
        {
                "id":"00001", "ChannelName":"First Channel",
                "Description":"Created with eZender, modified by studio",
                "Reviews":{"active":"1"},
                "Intro": {"active":"0"},
                "Categories":
                [
                        {"id":"1", "name":"Slideshows", "cat_pwd":"0"}
                ],
                "MovieRange":{"from":1, "to":1, "of":3},
                "Movies":
                [

```
                        {"id":"744",
                        "m_pwd":0,
                        "tkt":1,
                        "ExternalID":"",
                        "Hits":110,
                        "Title":"1MinutDV.mov",
                        "Heading":"",
                        "SubHeading":"",
                        "Body":"",
                        "Duration":"00:01:00",
                        "PublishDate":"2009-04-20 10:23:38",
                        "PublishBy":"5",
                        "ThumbNail":{"url":"http://x/getThumbnailImage.php%3Fch=00001%26aid=744"},
                        "HTTPURL":"http://x/getMovie.php%3Fmid=744%26br=256",
                        "RTSPURL":"rtsp://stream.barsark.net/x.secure/"}
                ]
        }
})
```


XMLMovieInfo.php, JSONMovieInfo.php, JSONPMovieInfo.php

These documents return an XML, JSON or JSONP list of a single movie in a channel, category. The data is structured in the same way as in XMLMovieList.php, JSONMovieList.php, JSONPMovieList.php but will only hold information about a single movie. All movies, including theft protected and password protected, are returned hence you will need to handle both getting and providing tickets for theft protected movies and providing correct passwords before the theft and password protected movies can be played.

Required GET parameters
- ch
  The ID of the Navicast channel the Movie is in. Available in Navicast studio.
- mid
  The ID of the Movie. Available in Navicast studio.

Optional GET parameters
- ch_pwd
  Required if the channel is password protected.
- speed
  Bandwidth indicator, 1=low, 2=medium, 3=high.
- FullInfo
  If set to 1, returns extra information about the channel and available categories within the channel. Useful to get initial information when first loading a player.

getXMLtkt.php, getJSONtkt.php, getJSONPtkt.php

If a movie is “Theft protected” in Navicast studio you will need a valid ticket to be able to request and view the movie. This document will let you request a vlid ticket. The ticket is valid for a short period of time after it is requested, do not count on it being there for more than 1 minute. If the ticket is not used within a minute you will need to request a new ticket before you can access the movie.

Required GET parameters
- ch
  Channel ID

- id
  Movie ID

Optional GET parameters
- rtsp
  If set to “1” you will get a key for an rtsp streaming movie otherwise you will get a ticket for a http progressive download movie.
- speed
  Bandwidth indicator, 1=low, 2=medium, 3=high (if not set defaults to 2)

Optional/required GET or POST parameters
- key (required if the channel key is set (POST or GET))
  String that exactly matches the channels key (set in Navicast studio)
- m_pwd (required for password protected movies and rtsp streaming movies)
  The movies password

XMLtkt (sample)

```
<?xml version="1.0" encoding="UTF-8" ?>
<Navicast type="OutputgetXMLtkt" version="0.8" docs="http://www.navicast.net/">
        <Result id="0">Success</Result>
        <Ticket>o5mgrs</Ticket>
</Navicast>
```

JSONtkt (sample)

```
{"type":"OutputJSONPMovieList", "version":"0.8", "docs":"http://www.navicast.net/",
"Result":{"id":"0", "Message":"Success"},
"Ticket":"b9nn4j"}
```

JSONPtkt (sample)

```
callback({"type":"OutputJSONPMovieList", "version":"0.8", "docs":"http://www.navicast.net/",
"Result":{"id":"0", "Message":"Success"},
"Ticket":"jnreiz"})
```

The “secret” channel key

Each channel in the Navicast system has a secret key that can be set in the channel admin area in Navicast studio. Anyone that has this key and knows how the Navicast system works may be able to request and get tickets and view movies on the Navicast system. By setting your own unique key and never exposing your channel key to anyone you should however obtain a high security level against unauthorized downloading and viewing of your movies.

getMovie.php

getMovie.php is used to request the actual movie files. You will need to pass the required parameters to get the movie.

Required GET parameters
- ch
  Channel ID

- mid
  Movie ID

Optional GET parameters
- speed
  Bandwidth indicator, 1=low, 2=medium, 3=high (if not set defaults to 2)
- tkt (a valid Navicast ticket)
- m_pwd (required for password protected and movies)
  The movies password (**Base64 encoded**)

getMoviePlaylist.php

getMoviePlaylist.php takes the same variables as getMovie.php but instead of returning the movie file it will return an XSPF playlist with a Navicast preroll and then the movie in it. Use this instead of getMovie.php when you want to display a Navicast preroll in front of the movie.

Here is how to use it with JW Player (http://www.longtailvideo.com/players/jw-flv-player/):
Instead of feeding the JW Player with file=getMovie.php?... you feed it with getMoviePlaylist.php?... using the same GET variables.
getMoviePlaylist.php uses all the same GET variables as getMovie.php so you only need to replace the file name from "getMovie" to "getMoviePlaylist".
Remember to remove the type=video flashvar in the JW Player embed tag.
Remember to add the repeat=list flashvar to the JW Player embed tag, otherwise only the intro movie will play.

Appart from the variables used by getMovie.php the getMoviePlaylist.php document also takes the category variable cat=categoryID.
You can just add &cat=x at the end of the other GET variables.
This makes it possible to play different intro movies depending on what category the movie is played from.

Required GET parameters
- ch
  Channel ID
- mid
  Movie ID

Optional GET parameters
- cat
  The category ID.
- speed
  Bandwidth indicator, 1=low, 2=medium, 3=high (if not set defaults to 2)
- tkt (a valid Navicast ticket if needed)
- m_pwd (required for password protected movies)
  The movies password (**Base64 encoded**)

MoviePlaylist (sample)

```
<?xml version="1.0" encoding="UTF-8"?>
<playlist version="1" xmlns="http://xspf.org/ns/0/">
        <trackList>
                <track>
                        <meta rel="type">video</meta>
                        <title>Preroll</title>
```

```
                <location>http://x/getIntroMovie.php?mid=2118&amp;ch=23&amp;br=128</location>
                <image>http://x/getThumbnailImage.php?aid=2125&amp;ch=23</image>
                <album>preroll</album>
            </track>
            <track>
                <meta rel="type">video</meta>
                <title>Movie</title>
                <location>http://x/getMovie.php?mid=2118&amp;br=128</location>
            </track>
        </trackList>
</playlist>
```


Javascript, AJAX and CORS support

The XMLMovieList.php, XMLMovieInfo.php and getXMLtkt.php API functions are very well suited for all kinds of AJAX[1] implementations. These XML based functions also implement CORS[2] to enable cross domain resource sharing. These functions will return the header *Access-Control-Allow-Origin: ** to enable full cross domain AJAX functionality.

This means that you can build your own Javascript or AJAX player, host it on your site and request all data from the Navicast server.

[1] AJAX is short for Asynchronous JavaScript and XML. For more info visit http://en.wikipedia.org/wiki/AJAX

[2] CORS is short for Cross-Origin Resource Sharing. For more info visit http://en.wikipedia.org/wiki/Cross-Origin_Resource_Sharing or http://hacks.mozilla.org/2009/07/cross-site-xmlhttprequest-with-cors/