

---

## Airplay

build unknown build unknown



Airplay attempts to be compatible with the latest AppleTV firmware but I'd like to add compatibility to other servers.

### Contribute

You can contribute with code, bugs or feature requests.

The development of the gem takes time and there's a lot of research and hardware tests to make all of this. If you want to contribute please consider donating as much as you want in: Paypal or Gumroad

### Table of Contents

- [Contribute](#)
- [Installation](#)

- 
- Usage
    - CLI
    - Library
  - Testing
  - Documentation
  - Contributors

## Installation

### Library

```
gem install airplay
```

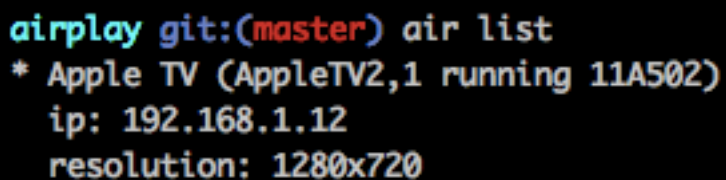
### CLI

```
gem install airplay-cli
```

## Usage

### CLI

**View devices** `air list`



```
airplay git:(master) air list
* Apple TV (AppleTV2,1 running 11A502)
  ip: 192.168.1.12
  resolution: 1280x720
```

```
1 * Apple TV (AppleTV2,1 running 11A502)
2   ip: 192.168.1.12
3   resolution: 1280x720
```

**Play a video** `air play [url to video or local file]`



```
airplay git:(master) air play ~/Downloads/ifrankenstein-tlr_h480p.mov
Playing /Users/elcuervo/Downloads/ifrankenstein-tlr_h480p.mov
time: 00:00:20 [=====] 12% Apple TV
```

---

```
1 Playing http://movietrailers.apple.com/movies/universal/rush/rush-
  tlr3_480p.mov?width=848&height=352
2 Time: 00:00:13 [=====] 7%
  Apple TV
```

## Show images

```
air view [url to image or image folder]
```

## Library

### Configuration

```
1 Airplay.configure do |config|
2   config.log_level      # Log4r levels (Default: Log4r::ERROR)
3   config.autodiscover   # Allows to search for nodes (Default: true)
4   config.host           # In which host to bind the server (Default:
    0.0.0.0)
5   config.port           # In which port to bind the server (Default:
    will find one)
6   config.output         # Where to log (Default: Log4r::Outputter.
    stdout)
7 end
```

### Devices

```
1 require "airplay"
2
3 Airplay.devices.each do |device|
4   puts device.name
5 end
```

### Accessing and Grouping

```
1 # You can access a known device easily
2 device = Airplay["Apple TV"]
3
4 # And add the password of the device if needed
5 device.password = "my super secret password"
6
7 # Or you can group known devices to have them do a given action
  together
8 Airplay.group["Backyard"] << Airplay["Apple TV"]
9 Airplay.group["Backyard"] << Airplay["Room TV"]
10
11 # The groups can later do some actions like:
12 Airplay.group["Backyard"].play("video")
```

---

### Images

```
1 require "airplay"
2
3 apple_tv = Airplay["Apple TV"]
4
5 # You can send local files
6 apple_tv.view("my_image.png")
7
8 # Or use remote files
9 apple_tv.view("https://github.com/elcuervo/airplay/raw/master/doc/img/
    logo.png")
10
11 # And define a transition
12 apple_tv.view("url_to_the_image", transition: "Dissolve")
13
14 # View all transitions
15 apple_tv.transitions
```

### Video

```
1 require "airplay"
2
3 apple_tv = Airplay["Apple TV"]
4 trailer = "http://movietrailers.apple.com/movies/dreamworks/
    needforspeed/needforspeed-tlr1xxzszs2_480p.mov"
5
6 player = apple_tv.play(trailer)
```

### Playlist

```
1 # You can also add videos to a playlist and let the library handle them
2 player.playlist << "video_url"
3 player.playlist << "video_path"
4 player.play
5
6 # Or control it yourself
7 player.next
8 player.previous
9
10 # Or if you prefer you can have several playlists
11 player = apple_tv.player
12 player.playlists["Star Wars Classic"] << "Star Wars Episode IV: A New
    Hope"
13 player.playlists["Star Wars Classic"] << "Star Wars Episode V: The
    Empire Strikes Back"
14 player.playlists["Star Wars Classic"] << "Star Wars Episode VI: Return
    of the Jedi"
15
16 player.playlists["Star Wars"] << "Star Wars Episode I: The Phantom
    Menace"
17 player.playlists["Star Wars"] << "Star Wars Episode II: Attack of the
    Clones"
```

---

```
18 player.playlists["Star Wars"] << "Star Wars Episode III: Revenge of the
    Sith"
19
20 player.use("Star Wars Classic")
21 player.play
22 player.wait
```

### **Player**

```
1 # Wait until the video is finished
2 player.wait
3
4 # Actions
5 player.pause
6 player.resume
7 player.stop
8 player.scrub
9 player.info
10 player.seek
11
12 # Access the playback time per second
13 player.progress -> progress {
14   puts "I'm viewing #{progress.position} of #{progress.duration}"
15 }
```

## **Testing**

Now there are two types of tests: Regular unit tests and integration tests. Thanks to the magic of the internet and a raspberry pi there are integration tests with a real Apple TV that is currently accessible.



The Apple TV is password protected to avoid issues with the tests but is configured in Travis CI. For that reason you won't be able to run those tests if you don't have an Apple TV.

Run unit tests with: `rake test:unit` and integration ones with: `rake test:integration`  
You can run all of them together with: `rake test:all`

## Documentation

All the documentation of the README can be found in the `doc` folder. To generate an updated README based on the contents of `doc` please use `rake doc:generate`

## Contributors

Last but not least a special thanks to all the contributors