ddf.minim

Interface Playable

All Known Implementing Classes:

AudioPlayer, AudioSnippet

`public interface` **`Playable`**

`Playable` defines functionality that you would expect from a tapedeck or CD player. Implementing classes are usually playing an audio file.

Author:

Damien Di Fede

Method Summary

`void`	**`cue`**`(int millis)` Sets the position to `millis` milliseconds from the beginning.
`AudioMetaData`	**`getMetaData`**`()` Returns the meta data for this.
`boolean`	**`isLooping`**`()` Returns true if this is currently playing and has more than one loop left to play.
`boolean`	**`isPlaying`**`()` Returns true if this currently playing.
`int`	**`length`**`()` Returns the length of the sound in milliseconds.
`void`	**`loop`**`()` Sets looping to continuous.
`void`	**`loop`**`(int num)` Sets this to loop `num` times.
`int`	**`loopCount`**`()` Returns the number of loops left to do.
`void`	**`pause`**`()` Pauses playback.
`void`	**`play`**`()` Starts playback from the current position.
`void`	**`play`**`(int millis)` Starts playback `millis` from the beginning.
`int`	**`position`**`()` Returns the current position of the "playhead" (ie how much of the sound has already been played)

`void`	`rewind()` Rewinds to the beginning.
`void`	`setLoopPoints(int start, int stop)` Sets the loop points used when looping.
`void`	`skip(int millis)` Skips `millis` from the current position.

Method Detail

play

`void play()`

Starts playback from the current position. If this was previous set to loop, looping will be disabled.

---

play

`void play(int millis)`

Starts playback `millis` from the beginning. If this was previous set to loop, looping will be disabled.

Parameters:
`millis` -

---

isPlaying

`boolean isPlaying()`

Returns true if this currently playing.

Returns:
true if this is currently playing

---

loop

`void loop()`

Sets looping to continuous. If this is already playing, the position *will not* be reset to the beginning. If this is not playing, it will start playing.

---

loop

`void loop(int num)`

Sets this to loop `num` times. If this is already playing, the position *will not* be reset to the beginning. If this is not playing, it will start playing.

Parameters:
`num` - the number of times to loop

---

isLooping

```
boolean isLooping()
```

Returns true if this is currently playing and has more than one loop left to play.

Returns:
true if this is looping

---

loopCount

```
int loopCount()
```

Returns the number of loops left to do.

Returns:
the number of loops left

---

setLoopPoints

```
void setLoopPoints(int start,
                   int stop)
```

Sets the loop points used when looping.

Parameters:
`start` - the start of the loop in milliseconds
`stop` - the end of the loop in milliseconds

---

pause

```
void pause()
```

Pauses playback.

---

cue

```
void cue(int millis)
```

Sets the position to `millis` milliseconds from the beginning. This will not change the playstate. If an error occurs while trying to cue, the position will not change. If you try to cue to a negative position or try to a position that is greater than `length()`, the amount will be clamped to zero or `length()`.

Parameters:
`millis` - the position to place the "playhead"

---

skip

`void **skip**(int millis)`

Skips `millis` from the current position. `millis` can be negative, which will make this skip backwards. If the skip amount would result in a negative position or a position that is greater than `length()`, the new position will be clamped to zero or `length()`.

Parameters:
`millis` - how many milliseconds to skip, sign indicates direction

---

rewind

`void **rewind**()`

Rewinds to the beginning. This *does not* stop playback.

---

position

`int **position**()`

Returns the current position of the "playhead" (ie how much of the sound has already been played)

Returns:
the current position of the "playhead"

---

length

`int **length**()`

Returns the length of the sound in milliseconds. If for any reason the length could not be determined, this will return -1. However, an unknown length should not impact playback.

Returns:
the length of the sound in milliseconds

---

getMetaData

`AudioMetaData **getMetaData**()`

Returns the meta data for this.
