Package: rotio (via r-universe)

June 20, 2026

Type Package

Title Pure-R 'OpenTimelineIO' Document Model

Version 0.1.0

Date 2026-06-12

Description A dependency-light 'OpenTimelineIO' (OTIO) <https://github.com/AcademySoftwareFoundation/OpenTimelineIO> document layer in pure R. Provides constructors for the OTIO object model (timelines, tracks, clips, gaps, media references, rational times and time ranges), functional builders that return new objects, and readers and writers for canonical '.otio' files through 'jsonlite'. The optional 'RcppOTIO' package validates output against the real 'OpenTimelineIO' C++ library. No compiled code.

License Apache License (== 2.0)

URL https://github.com/cornball-ai/rotio

BugReports https://github.com/cornball-ai/rotio/issues

Depends R (>= 4.4.0)

Imports jsonlite

Additional_repositories https://cornball-ai.github.io/drat

Suggests tinytest, RcppOTIO

Encoding UTF-8

RoxygenNote 7.3.2

Repository https://cornball-ai.r-universe.dev

Date/Publication 2026-06-12 22:51:29 UTC

RemoteUrl https://github.com/cornball-ai/rotio

RemoteRef HEAD

RemoteSha 057eabe166225d714cf96b0bc6b47389f6cb2a20

Contents

`active_media_reference_key` *Active media reference key of a clip*

Description

Active media reference key of a clip

Usage

```
active_media_reference_key(x)

active_media_reference_key(x) <- value
```

Arguments

`x`	A `Clip`.
`value`	New active key.

Value

The active media reference key, a character string.

Examples

```
cl <- Clip("a", ExternalReference("a.mp4"))
active_media_reference_key(cl)
```

---

add_child *Append a child, functionally (clone + append)*

Description

Returns a new composition: a clone of `parent` with a clone of `child` appended. Neither input is mutated (cf. `append_child`, which mutates in place).

Usage

```
add_child(parent, child)
```

Arguments

`parent`	A composition (Track/Stack) or collection.
`child`	An OTIO object.

Value

A new composition.

Examples

```
v <- add_child(Track("V1"), Clip("a", ExternalReference("a.mp4")))
```

---

add_effect *Append an effect (functional: returns a new object)*

Description

Returns a clone of `x` with `effect` appended to its `effects` list; the input is unchanged.

Usage

```
add_effect(x, effect)
```

Arguments

x	An item or composition (anything with an `effects` list).
effect	An `Effect`.

Value

A new object of the same class.

Examples

```
clip <- Clip("a", ExternalReference("a.mp4"))
clip <- add_effect(clip, LinearTimeWarp(effect_name = "LinearTimeWarp",
                                        time_scalar = 2))
length(effects(clip))
```

---

add_track *Append a track to a timeline, functionally*

Description

Returns a new timeline (clone) with a clone of `track` appended to its stack. Neither input is mutated.

Usage

```
add_track(timeline, track)
```

Arguments

timeline	A `Timeline`.
track	A `Track`.

Value

A new timeline.

Examples

```
tl <- add_track(Timeline("demo"), Track("V1", kind = "Video"))
```

almost_equal *Are two RationalTimes almost equal?*

Description

Rescales a to b's rate and compares to b's value within `delta` (in b's rate units), matching opentime.

Usage

```
almost_equal(a, b, delta = 0)
```

Arguments

`delta`	Tolerance in b's rate units (default 0).
a, b	`RationalTime`s.

Value

Logical scalar.

Examples

```
almost_equal(RationalTime(24, 24), RationalTime(48, 48))
```

append_child *Append a child to a composition (in place)*

Description

Mutates `x`, attaching `child` and setting its parent. Errors if `child` already has a parent (use `remove_child` first, or the functional `add_child`). Returns `x` invisibly.

Usage

```
append_child(x, child)
```

Arguments

`x`	A composition or collection.
`child`	An OTIO object with no parent.

Value

`x`, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a", ExternalReference("a.mp4")))
length(children(trk))
```

`available_range` *Available range of a media reference or clip*

Description

For a media reference, the stored available range. For a clip, the active media reference's available range (errors if none is set, matching OTIO).

Usage

```
available_range(x)

available_range(x) <- value
```

Arguments

`x`	A media reference or clip.
`value`	A `TimeRange` or `NULL`.

Value

A `TimeRange` (`NULL` for a media reference with none set); the setter returns `x`.

Examples

```
ref <- ExternalReference("a.mp4",
    available_range = TimeRange(RationalTime(0, 30), RationalTime(90, 30)))
available_range(ref)
```

`children` *Children of a composition or collection*

Description

Children of a composition or collection

Usage

```
children(x)
```

Arguments

x	A composition (Track/Stack) or collection.

Value

A list of child OTIO objects, in order.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a"))
children(trk)
```

---

clamped *Clamp a time or range into a bounding TimeRange*

Description

`tr` is the bounding range; `other` is clamped into it (opentime `TimeRange::clamped`).

Usage

```
clamped(tr, other)
```

Arguments

tr	The bounding `TimeRange`.
other	A `RationalTime` or `TimeRange` to clamp.

Value

A clamped `RationalTime` or `TimeRange`, matching `other`.

Examples

```
tr <- TimeRange(RationalTime(0, 24), RationalTime(48, 24))
clamped(tr, RationalTime(96, 24))
```

clear_children *Remove all children (in place)*

Description

Remove all children (in place)

Usage

```
clear_children(x)
```

Arguments

x	A composition or collection.

Value

x, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a"))
clear_children(trk)
length(children(trk))
```

Clip *Construct a Clip*

Description

A media reference plus a source_range. The reference is stored under the DEFAULT_MEDIA key (OTIO's multi-reference model).

Usage

```
Clip(name, media_reference = MissingReference(), source_range = NULL,
     metadata = NULL)
```

Arguments

name	Clip name.
media_reference	A media reference (default MissingReference).
source_range	Optional TimeRange.
metadata	Named list of metadata.

Value

A `Clip`.

Examples

```
Clip("a", ExternalReference("a.mp4"),
     source_range = TimeRange(RationalTime(0, 30), RationalTime(90, 30)))
```

---

clone *Deep-clone an OTIO object*

Description

Returns a deep copy of `x`. The clone's parent is reset to `NULL` and its descendants' parent pointers are rewired inside the clone, so it is a fully detached, internally consistent subtree.

Usage

```
clone(x)
```

Arguments

`x`	An OTIO object.

Value

A deep copy.

Examples

```
tl <- add_track(Timeline("a"), Track("V1"))
tl2 <- clone(tl)
```

---

color *Get or set an item's display color*

Description

Get or set an item's display color

Usage

```
color(x)

color(x) <- value
```

Arguments

x	An item or composition.
value	A color value (or NULL).

Value

The item's color value, or NULL if unset.

Examples

```
cl <- Clip("a")
color(cl) <- "RED"
color(cl)
```

---

comment *Comment of a Marker*

Description

Comment of a Marker

Usage

```
comment(x)

comment(x) <- value
```

Arguments

x	A Marker.
value	New comment.

Value

The comment string; the setter returns x.

Examples

```
m <- Marker("note", comment = "check color")
comment(m)
```

contains *Does a TimeRange contain a time or range?*

Description

For a RationalTime: start <= t < end_exclusive. For a TimeRange: the other range lies strictly within on both ends (tolerance epsilon_s).

Usage

```
contains(tr, other, epsilon_s = .EPS)
```

Arguments

tr	A TimeRange.
other	A RationalTime or TimeRange.
epsilon_s	Tolerance in seconds.

Value

Logical scalar.

Examples

```
tr <- TimeRange(RationalTime(0, 24), RationalTime(48, 24))
contains(tr, RationalTime(12, 24))
```

default_media_key *The default media reference key ("DEFAULT_MEDIA")*

Description

The default media reference key ("DEFAULT_MEDIA")

Usage

```
default_media_key()
```

Value

The string "DEFAULT_MEDIA".

Examples

```
default_media_key()
```

`Effect` *Construct an OTIO effect*

Description

`Effect` is a generic `effect_name` plus metadata parameters. `LinearTimeWarp` is OTIO's linear speed change (`time_scalar`).

Usage

```
Effect(name = "", effect_name = "", enabled = TRUE, metadata = NULL)

LinearTimeWarp(name = "", effect_name = "", time_scalar = 1, enabled = TRUE,
               metadata = NULL)

TimeEffect(name = "", effect_name = "", metadata = NULL)

FreezeFrame(name = "", metadata = NULL)
```

Arguments

`name`	Effect instance name (default empty).
`effect_name`	Effect kind label (default empty, matching OTIO).
`enabled`	Whether the effect is active (default `TRUE`).
`metadata`	Named list of parameters.
`time_scalar`	Linear time scale (playback rate multiplier).

Value

An `Effect`.

Examples

```
Effect("blur", "GaussianBlur", metadata = list(size = 4))

LinearTimeWarp("speed", effect_name = "LinearTimeWarp", time_scalar = 2)
```

effect_name *Effect kind label*

Description

Effect kind label

Usage

```
effect_name(x)

effect_name(x) <- value
```

Arguments

x	An Effect.
value	New effect_name.

Value

The effect kind string; the setter returns x.

Examples

```
e <- Effect("blur", "GaussianBlur")
effect_name(e)
```

effects *Effects of an item or composition*

Description

Effects of an item or composition

Usage

```
effects(x)
```

Arguments

x	An item or composition.

Value

A list of Effect objects.

Examples

```
clip <- add_effect(Clip("a"), FreezeFrame())
effects(clip)
```

`enabled` *Whether an item, composition, or effect is enabled*

Description

A disabled clip/track is muted; a disabled effect is bypassed.

Usage

```
enabled(x)

enabled(x) <- value
```

Arguments

`x`	An object with an `enabled` field.
`value`	`TRUE` or `FALSE`.

Value

`TRUE` if the object is enabled, else `FALSE`.

Examples

```
cl <- Clip("a")
enabled(cl) <- FALSE
enabled(cl)
```

`end_frame` *Last frame number of an ImageSequenceReference*

Description

Last frame number of an ImageSequenceReference

Usage

```
end_frame(x)
```

Arguments

`x`	An `ImageSequenceReference`.

Value

Integer frame number of the last image in the sequence.

Examples

```
isr <- ImageSequenceReference("file:///frames", "frame.", ".exr", rate = 24,
    available_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
end_frame(isr)
```

---

end_time_exclusive *Exclusive / inclusive end of a TimeRange*

Description

`end_time_exclusive` = start + duration. `end_time_inclusive` is the last whole frame; for a span of one frame or less it is the start time (matching opentime).

Usage

```
end_time_exclusive(x)

end_time_inclusive(x)
```

Arguments

`x`	A `TimeRange`.

Value

A `RationalTime`.

Examples

```
tr <- TimeRange(RationalTime(0, 24), RationalTime(48, 24))
end_time_exclusive(tr)
end_time_inclusive(tr)
```

extended_by *Smallest TimeRange covering two ranges*

Description

Smallest TimeRange covering two ranges

Usage

```
extended_by(tr, other)
```

Arguments

tr, other	TimeRanges.

Value

A TimeRange.

Examples

```
a <- TimeRange(RationalTime(0, 24), RationalTime(48, 24))
b <- TimeRange(RationalTime(24, 24), RationalTime(48, 24))
extended_by(a, b)
```

ExternalReference *Construct a media reference*

Description

ExternalReference points a clip at media by URL. MissingReference is a placeholder for a clip with no resolvable media.

Usage

```
ExternalReference(target_url, available_range = NULL, metadata = NULL)

MissingReference(name = "", available_range = NULL, metadata = NULL)
```

Arguments

target_url	Media URL for an ExternalReference.
available_range	Optional TimeRange of available media.
metadata	Named list of metadata.
name	Reference name (default empty).

Value

A media reference object.

Examples

```
ExternalReference("media/clip01.mp4")
MissingReference()
```

`fill` *Fill a gap with an item (3/4-point edit)*

Description

Replaces the gap covering `track_time` with `item`. The `reference_point` controls the transform: `"Source"` uses the clip's own duration, `"Sequence"` clamps to the gap, `"Fit"` time-warps the clip to fill the gap exactly. Mutates `track` in place.

Usage

```
fill(item, track, track_time, reference_point = "Source")
```

Arguments

`item`	The item to place (usually a clip).
`track`	A `Track`.
`track_time`	A `RationalTime` inside the gap to fill.
`reference_point`	One of `"Source"`, `"Sequence"`, `"Fit"`.

Value

`track`, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Gap(RationalTime(48, 24)))
cl <- Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                         RationalTime(24, 24)))
fill(cl, trk, RationalTime(0, 24))
name(children(trk)[[1]])
```

find_clips *All clips within an object (recursive)*

Description

All clips within an object (recursive)

Usage

```
find_clips(x)
```

Arguments

x	A Timeline, composition, or collection.

Value

A list of `Clip` objects.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a"))
length(find_clips(trk))
```

flatten_stack *Flatten a stack of tracks into a single track*

Description

Composites top-down: starts from the topmost track and fills its holes (gaps and disabled items) with content from the tracks below, recursing downward. Transitions are preserved; a lower transition cut by a hole boundary errors with "Cannot trim in the middle of a transition" (matching OTIO).

Usage

```
flatten_stack(x)
```

Arguments

x	A `Stack` or a list of `Track`s (bottom-to-top).

Value

A flattened `Track`.

Examples

```
trk <- add_child(Track("V1"), Clip("a",
    source_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24))))
flatten_stack(list(trk))
```

---

frame_for_time *Frame for a time in an ImageSequenceReference*

Description

Errors if `time` falls outside the available range (matching OTIO).

Usage

```
frame_for_time(x, time)
```

Arguments

`x`	An `ImageSequenceReference`.
`time`	A `RationalTime`.

Value

Integer frame number for `time`.

Examples

```
isr <- ImageSequenceReference("file:///frames", "frame.", ".exr", rate = 24,
    available_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
frame_for_time(isr, RationalTime(10, 24))
```

---

from_frames *Construct a RationalTime from a frame number at a rate*

Description

Construct a RationalTime from a frame number at a rate

Usage

```
from_frames(frame, rate)
```

Arguments

`frame`	Integer frame number.
`rate`	Rate (fps).

Value

A `RationalTime`.

Examples

```
from_frames(48, 24)
```

`from_json_file` *Read an OTIO JSON file into the object model*

Description

Read an OTIO JSON file into the object model

Usage

```
from_json_file(file_name)
```

Arguments

`file_name`	Path to a `.otio` JSON file.

Value

The reconstructed OTIO object, with parent pointers wired.

Examples

```
f <- tempfile(fileext = ".otio")
to_json_file(Timeline("demo"), f)
name(from_json_file(f))
unlink(f)
```

`from_json_string` *Parse an OTIO JSON string into the object model*

Description

Parse an OTIO JSON string into the object model

Usage

```
from_json_string(input)
```

Arguments

`input`	An OTIO JSON string.

Value

The reconstructed OTIO object (typically a Timeline), with parent pointers wired.

Examples

```
tl <- Timeline("demo")
identical(name(from_json_string(to_json_string(tl))), "demo")
```

---

from_seconds *Construct a RationalTime from seconds at a rate*

Description

Construct a RationalTime from seconds at a rate

Usage

```
from_seconds(seconds, rate = 1)
```

Arguments

seconds	Numeric seconds.
rate	Rate (fps).

Value

A RationalTime.

Examples

```
from_seconds(1.5, 24)
```

---

from_time_string *RationalTime from a time string ("HH:MM:SS.sss") at a rate*

Description

Preserves the fractional value (no rounding to a frame boundary).

Usage

```
from_time_string(time_string, rate)
```

Arguments

time_string	A time string.
rate	Rate (fps).

Value

A RationalTime.

Examples

```
from_time_string("00:00:03.75", 24)
```

---

from_timecode *RationalTime from a SMPTE timecode*

Description

A ; frame separator is treated as drop-frame.

Usage

```
from_timecode(timecode, rate)
```

Arguments

timecode	A "HH:MM:SS:FF" (or ";FF" for drop-frame) string.
rate	Timecode rate.

Value

A RationalTime.

Examples

```
from_timecode("01:00:00:00", 24)
```

Gap *Construct a Gap*

Description

Empty space on a track of a given duration.

Usage

```
Gap(duration, name = "", metadata = NULL)
```

Arguments

duration	A RationalTime.
name	Gap name.
metadata	Named list of metadata.

Value

A Gap.

Examples

```
Gap(RationalTime(15, 30))
```

`generator_kind` *Generator kind of a GeneratorReference*

Description

Generator kind of a GeneratorReference

Usage

```
generator_kind(x)

generator_kind(x) <- value
```

Arguments

`x`	A `GeneratorReference`.
`value`	New generator kind.

Value

The generator kind string; the setter returns `x`.

Examples

```
g <- GeneratorReference("bars", generator_kind = "SMPTEBars")
generator_kind(g)
```

GeneratorReference *Construct a GeneratorReference*

Description

Procedurally-generated media (color bars, solids, ...).

Usage

```
GeneratorReference(name = "", generator_kind = "", available_range = NULL,
                   parameters = NULL, metadata = NULL)
```

Arguments

name	Reference name.
generator_kind	Generator kind (e.g. "SMPTEBars").
available_range	Optional TimeRange.
parameters	Named list of generator parameters.
metadata	Named list of metadata.

Value

A GeneratorReference.

Examples

```
GeneratorReference("bars", generator_kind = "SMPTEBars")
```

global_start_time *Global start time of a timeline*

Description

Global start time of a timeline

Usage

```
global_start_time(x)

global_start_time(x) <- value
```

Arguments

x	A Timeline.
value	A RationalTime or NULL.

Value

The timeline's global start time (a RationalTime or NULL); the setter returns x.

Examples

```
tl <- Timeline("demo")
global_start_time(tl) <- RationalTime(0, 24)
global_start_time(tl)
```

---

has_child *Does a composition directly contain a child?*

Description

Does a composition directly contain a child?

Usage

```
has_child(x, child)
```

Arguments

x	A composition or collection.
child	An OTIO object.

Value

TRUE if child is a direct child of x, else FALSE.

Examples

```
trk <- Track("V1")
cl <- Clip("a")
append_child(trk, cl)
has_child(trk, cl)
```

has_clips *Does an object contain any clips (recursive)?*

Description

Does an object contain any clips (recursive)?

Usage

```
has_clips(x)
```

Arguments

x	A Timeline, composition, or collection.

Value

TRUE if x contains at least one clip, else FALSE.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a"))
has_clips(trk)
```

ImageSequenceReference *Construct an ImageSequenceReference*

Description

A numbered image sequence (e.g. frame.0001.exr).

Usage

```
ImageSequenceReference(target_url_base = "", name_prefix = "",
                       name_suffix = "", start_frame = 1L, frame_step = 1L,
                       rate = 1, frame_zero_padding = 0L,
                       missing_frame_policy = "error", available_range = NULL,
                       metadata = NULL)
```

Arguments

`target_url_base`	Directory URL.
`start_frame`	First frame number (default 1).
`frame_step`	Frames between images (default 1).
`rate`	Frame rate (default 1).
`frame_zero_padding`	Zero-padding width for the frame number (default 0).
`missing_frame_policy`	One of `"error"`, `"hold"`, `"black"`.
`available_range`	Optional `TimeRange`.
`metadata`	Named list of metadata.
`name_prefix`, `name_suffix`	Filename prefix/suffix around the frame number.

Value

An `ImageSequenceReference`.

Examples

```
ImageSequenceReference("file:///frames", "frame.", ".exr",
                       frame_zero_padding = 4, rate = 24)
```

`index_of_child` *1-based position of a child, or NA*

Description

1-based position of a child, or NA

Usage

```
index_of_child(x, child)
```

Arguments

`x`	A composition or collection.
`child`	An OTIO object.

Value

The 1-based integer position of `child` in `x`, or `NA_integer_` if it is not a direct child.

Examples

```
trk <- Track("V1")
cl <- Clip("a")
append_child(trk, cl)
index_of_child(trk, cl)
```

`insert` *Insert an item at a time, splitting the item it lands in*

Description

Inserts `item` at `time`, splitting whatever item spans that point; before the start it prepends, past the end it appends (filling any gap). Mutates `composition` in place.

Usage

```
insert(item, composition, time, remove_transitions = TRUE, fill_template = NULL)
```

Arguments

`item`	The item to insert (usually a clip).
`composition`	A `Track` (or composition).
`time`	A `RationalTime` to insert at.
`remove_transitions`	Remove transitions at `time` (default TRUE).
`fill_template`	Optional gap template when appending past the end.

Value

`composition`, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                                     RationalTime(48, 24))))
cl <- Clip("b", source_range = TimeRange(RationalTime(0, 24),
                                         RationalTime(12, 24)))
insert(cl, trk, RationalTime(24, 24))
length(children(trk))
```

insert_child *Insert a child at a 1-based position (in place)*

Description

Insert a child at a 1-based position (in place)

Usage

```
insert_child(x, index, child)
```

Arguments

x	A composition or collection.
index	1-based position.
child	An OTIO object with no parent.

Value

x, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("b"))
insert_child(trk, 1, Clip("a"))
name(children(trk)[[1]])
```

intersects *Do two TimeRanges intersect?*

Description

Do two TimeRanges intersect?

Usage

```
intersects(tr, other, epsilon_s = .EPS)
```

Arguments

epsilon_s	Tolerance in seconds.
tr, other	TimeRanges.

Value

Logical scalar.

Examples

```
a <- TimeRange(RationalTime(0, 24), RationalTime(48, 24))
b <- TimeRange(RationalTime(24, 24), RationalTime(48, 24))
intersects(a, b)
```

---

is_effect *Is x an Effect?*

Description

Is x an Effect?

Usage

```
is_effect(x)
```

Arguments

x	Object to test.

Value

TRUE if x is an Effect, else FALSE.

Examples

```
is_effect(Effect("blur", "GaussianBlur"))
```

---

is_equivalent_to *Are two OTIO objects equivalent?*

Description

Structural equality, compared via canonical OTIO JSON.

Usage

```
is_equivalent_to(x, other)
```

Arguments

x, other	OTIO objects.

Value

Logical scalar.

Examples

```
is_equivalent_to(Timeline("a"), Timeline("a"))
```

---

is_missing_reference *Is x a MissingReference?*

Description

Is x a MissingReference?

Usage

```
is_missing_reference(x)
```

Arguments

x	Object to test.

Value

TRUE if x is a MissingReference, else FALSE.

Examples

```
is_missing_reference(MissingReference())
```

---

is_otio *Type predicates for OTIO objects*

Description

Type predicates for OTIO objects

Usage

```
is_otio(x)

is_timeline(x)

is_composition(x)

is_media_reference(x)
```

Arguments

x	Object to test.

Value

TRUE if x is of the corresponding class, else FALSE.

Examples

```
is_otio(Timeline("demo"))
is_timeline(Track("V1"))
```

---

is_parent_of *Is x an ancestor of other?*

Description

Is x an ancestor of other?

Usage

```
is_parent_of(x, other)
```

Arguments

x	A composition or collection.
other	An OTIO object.

Value

TRUE if x appears anywhere in other's parent chain, else FALSE.

Examples

```
trk <- Track("V1")
cl <- Clip("a")
append_child(trk, cl)
is_parent_of(trk, cl)
```

is_rational_time *Is x a RationalTime / TimeRange?*

Description

Is x a RationalTime / TimeRange?

Usage

```
is_rational_time(x)

is_time_range(x)
```

Arguments

x	Object to test.

Value

Logical scalar.

Examples

```
is_rational_time(RationalTime(24, 24))
is_time_range(TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
```

is_unknown_schema *Is an object's schema unknown?*

Description

TRUE when the object's schema type is not a recognised OTIO type (e.g. parsed from JSON written by a newer or third-party schema).

Usage

```
is_unknown_schema(x)
```

Arguments

x	An OTIO object.

Value

A logical scalar.

Examples

```
is_unknown_schema(Timeline("demo"))
```

Item *Construct a base Item*

Description

A generic OTIO `Item` (the base class of clips, gaps, and compositions). Mainly used by the `"Fit"` reference point of `fill`, which wraps media in a plain item carrying a time warp.

Usage

```
Item(name = "", source_range = NULL, effects = NULL, markers = NULL,
     enabled = TRUE, metadata = NULL)
```

Arguments

`name`	Item name.
`source_range`	Optional `TimeRange`.
`effects`	List of `Effect`s.
`markers`	List of `Marker`s.
`enabled`	Whether the item is enabled (default `TRUE`).
`metadata`	Named list of metadata.

Value

An `Item`.

Examples

```
Item("warp", source_range = TimeRange(RationalTime(0, 30), RationalTime(60, 30)))
```

kind *Get or set a track's kind*

Description

Get or set a track's kind

Usage

```
kind(x)

kind(x) <- value
```

Arguments

x	A Track.
value	"Video" or "Audio".

Value

The track kind, "Video" or "Audio".

Examples

```
trk <- Track("A1", kind = "Audio")
kind(trk)
```

---

marked_range *Marked range of a Marker*

Description

Marked range of a Marker

Usage

```
marked_range(x)

marked_range(x) <- value
```

Arguments

x	A Marker.
value	A TimeRange.

Value

The marker's TimeRange; the setter returns x.

Examples

```
m <- Marker("note", TimeRange(RationalTime(0, 30), RationalTime(30, 30)))
marked_range(m)
```

Marker *Construct a Marker*

Description

Annotates a time range on an item.

Usage

```
Marker(name = "", marked_range = NULL, color = "GREEN", comment = "",
       metadata = NULL)
```

Arguments

name	Marker name.
marked_range	A TimeRange.
color	Marker color (default "GREEN").
comment	Free-text comment.
metadata	Named list of metadata.

Value

A Marker.

Examples

```
Marker("note", TimeRange(RationalTime(0, 30), RationalTime(30, 30)),
       comment = "check color")
```

media_reference *Active media reference of a clip*

Description

Active media reference of a clip

Usage

```
media_reference(x)

media_reference(x) <- value
```

Arguments

x	A Clip.
value	A media reference.

Value

The clip's active media reference object.

Examples

```
cl <- Clip("a", ExternalReference("a.mp4"))
media_reference(cl)
```

---

media_references *Media references of a clip*

Description

Media references of a clip

Usage

```
media_references(x)
```

Arguments

x	A Clip.

Value

The clip's named list of media references.

Examples

```
cl <- Clip("a", ExternalReference("a.mp4"))
names(media_references(cl))
```

MediaReference *Construct a generic MediaReference*

Description

Construct a generic MediaReference

Usage

```
MediaReference(name = "", available_range = NULL, metadata = NULL)
```

Arguments

name	Reference name.
available_range	Optional TimeRange.
metadata	Named list of metadata.

Value

A MediaReference.

Examples

```
MediaReference("ref")
```

---

metadata *Get or set object metadata*

Description

Get or set object metadata

Usage

```
metadata(x)

metadata(x) <- value
```

Arguments

x	An OTIO object.
value	A named list.

Value

The object's metadata as a named list (possibly empty).

Examples

```
cl <- Clip("a")
metadata(cl) <- list(note = "take 2")
metadata(cl)
```

name *Get or set the name of an OTIO object*

Description

Get or set the name of an OTIO object

Usage

```
name(x)

name(x) <- value
```

Arguments

x	An OTIO object.
value	New name.

Value

The object's name, a character string.

Examples

```
cl <- Clip("a")
name(cl) <- "b"
name(cl)
```

number_of_images_in_sequence *Number of images in an ImageSequenceReference*

Description

Number of images in an ImageSequenceReference

Usage

```
number_of_images_in_sequence(x)
```

Arguments

x	An ImageSequenceReference.

Value

Integer count of images (0 if no available range is set).

Examples

```
isr <- ImageSequenceReference("file:///frames", "frame.", ".exr", rate = 24,
    available_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
number_of_images_in_sequence(isr)
```

`overlapping` *Does an item overlap its neighbours?*

Description

Only transitions overlap (they span a cut); everything else returns `FALSE`.

Usage

```
overlapping(x)
```

Arguments

`x`	An item.

Value

Logical scalar.

Examples

```
overlapping(Clip("a", ExternalReference("a.mp4")))
```

`overlaps` *Do two TimeRanges overlap (cross without containment)?*

Description

Do two TimeRanges overlap (cross without containment)?

Usage

```
overlaps(tr, other, epsilon_s = .EPS)
```

Arguments

`epsilon_s`	Tolerance in seconds.
`tr`, `other`	`TimeRanges`.

Value

Logical scalar.

Examples

```
a <- TimeRange(RationalTime(0, 24), RationalTime(48, 24))
b <- TimeRange(RationalTime(24, 24), RationalTime(48, 24))
overlaps(a, b)
```

---

overwrite *Overwrite a span of a composition with an item*

Description

Places `item` over `range` (in composition coordinates), partitioning or removing the items it covers and filling any hole before/after with a gap (or `fill_template`). Mutates `composition` in place.

Usage

```
overwrite(item, composition, range, remove_transitions = TRUE,
          fill_template = NULL)
```

Arguments

`item`	The item to place (usually a clip).
`composition`	A `Track` (or composition).
`range`	A `TimeRange` to overwrite.
`remove_transitions`	Remove transitions within `range` (default TRUE).
`fill_template`	Optional gap template for holes.

Value

`composition`, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Gap(RationalTime(48, 24)))
cl <- Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                         RationalTime(12, 24)))
overwrite(cl, trk, TimeRange(RationalTime(12, 24), RationalTime(12, 24)))
length(children(trk))
```

parameters *Parameters of a GeneratorReference*

Description

Parameters of a GeneratorReference

Usage

```
parameters(x)

parameters(x) <- value
```

Arguments

x	A GeneratorReference.
value	A named list of parameters.

Value

A named list of generator parameters; the setter returns x.

Examples

```
g <- GeneratorReference("bars", parameters = list(width = 1920))
parameters(g)
```

parent *Parent of an OTIO object*

Description

The containing composition/collection, or NULL. A Timeline's root track Stack is parentless.

Usage

```
parent(x)
```

Arguments

x	An OTIO object.

Value

The parent object or NULL.

Examples

```
trk <- Track("V1")
cl <- Clip("a")
append_child(trk, cl)
name(parent(cl))
```

`presentation_time_for_image_number`
Presentation time of the nth image (1-based)

Description

Presentation time of the nth image (1-based)

Usage

```
presentation_time_for_image_number(x, image_number)
```

Arguments

`x`	An `ImageSequenceReference`.
`image_number`	1-based image index.

Value

A `RationalTime` at the reference's rate.

Examples

```
isr <- ImageSequenceReference("file:///frames", "frame.", ".exr", rate = 24,
    available_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
presentation_time_for_image_number(isr, 2)
```

`range_from_start_end_time`
Construct a TimeRange from start and exclusive end times

Description

Duration is computed in the start time's rate (opentime convention).

Usage

```
range_from_start_end_time(start_time, end_time_exclusive)
```

Arguments

start_time	A RationalTime.
end_time_exclusive	A RationalTime.

Value

A TimeRange.

Examples

```
range_from_start_end_time(RationalTime(0, 24), RationalTime(48, 24))
```

---

range_in_parent *Range of an item within its parent composition*

Description

Computed from the durations of the preceding siblings (gaps included). The duration is the item's trimmed duration.

Usage

```
range_in_parent(x)
```

Arguments

x	An item with a parent.

Value

A TimeRange in the parent's coordinates.

Examples

```
trk <- Track("V1")
cl <- Clip("a",
    source_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
append_child(trk, cl)
range_in_parent(cl)
```

RationalTime *Construct a RationalTime*

Description

A `value` / `rate` pair measured in the rate's units (frame number / fps for frame-based work).

Usage

```
RationalTime(value, rate = 1)
```

Arguments

`value`	Time value. Coerced to double.
`rate`	Rate (fps); must be > 0. Default 1.

Value

A `RationalTime`.

Examples

```
RationalTime(180, 30)
```

read_otiod *Read an OTIOD bundle*

Description

Reads `<dir>/content.otio` into the object model. Media references stay relative to the bundle directory.

Usage

```
read_otiod(dir)
```

Arguments

`dir`	Bundle directory.

Value

The bundle's OTIO object (typically a `Timeline`).

Examples

```
d <- tempfile("bundle")
write_otiod(Timeline("demo"), d)
name(read_otiod(d))
unlink(d, recursive = TRUE)
```

`register_downgrade_function` *Register a schema downgrade function*

Description

`fn` receives the field list of a `schema_name` object at `version_to_downgrade_from` and returns the field list one version lower. Stored for use when writing older schema versions.

Usage

```
register_downgrade_function(schema_name, version_to_downgrade_from, fn)
```

Arguments

`schema_name`	Schema type name. Must be a known OTIO schema.
`version_to_downgrade_from`	Source version (integer).
`fn`	Function of one argument (the field list) returning a field list.

Value

`TRUE` if registered; `FALSE` for an unknown schema or a duplicate `(schema, version)` pair (matching RcppOTIO).

Examples

```
register_downgrade_function("Effect", 2, function(fields) fields)
```

register_upgrade_function *Register a schema upgrade function*

Description

`fn` receives the parsed field list of a `schema_name` object at the version below `version_to_upgrade_to` and returns the upgraded list. It is applied automatically by `from_json_string` when an older version is read.

Usage

```
register_upgrade_function(schema_name, version_to_upgrade_to, fn)
```

Arguments

`schema_name`	Schema type name (e.g. `"Marker"`). Must be a known OTIO schema.
`version_to_upgrade_to`	Target version (integer).
`fn`	Function of one argument (the field list) returning a field list.

Value

`TRUE` if registered; `FALSE` for an unknown schema or a duplicate `(schema, version)` pair (matching RcppOTIO).

Examples

```
register_upgrade_function("Effect", 2, function(fields) fields)
```

remove *Remove the item at a time, optionally leaving a gap*

Description

Removes the item covering `time`; if `fill`, replaces it with a gap (or `fill_template`) of the same range, otherwise the neighbours concatenate. Mutates `composition` in place.

Usage

```
remove(composition, time, fill = TRUE, fill_template = NULL)
```

Arguments

composition	A Track (or composition).
time	A RationalTime within the item to remove.
fill	Fill the hole with a gap/template (default TRUE).
fill_template	Optional item to fill with (default a gap).

Value

composition, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                                     RationalTime(48, 24))))
remove(trk, RationalTime(0, 24))
class(children(trk)[[1]])
```

---

remove_child *Remove the child at a position (in place)*

Description

Remove the child at a position (in place)

Usage

```
remove_child(x, index)
```

Arguments

x	A composition or collection.
index	1-based position.

Value

x, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a"))
remove_child(trk, 1)
length(children(trk))
```

rescaled_to *Rescale a RationalTime to a new rate*

Description

Rescale a RationalTime to a new rate

Usage

```
rescaled_to(x, new_rate)
```

Arguments

x	A RationalTime.
new_rate	Target rate (fps).

Value

A RationalTime at new_rate.

Examples

```
rescaled_to(RationalTime(24, 24), 48)
```

ripple *Ripple an item's source range*

Description

Adjusts the item's source_range start by delta_in and exclusive end by delta_out without moving any other item. Mutates in place.

Usage

```
ripple(item, delta_in, delta_out)
```

Arguments

item	An item (usually a clip).
delta_in, delta_out	RationalTime adjustments to the start and exclusive end.

Value

item, invisibly.

Examples

```
cl <- Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                         RationalTime(48, 24)))
ripple(cl, RationalTime(0, 24), RationalTime(12, 24))
source_range(cl)
```

`roll` *Roll an item, adjusting adjacent items to fit*

Description

Adjusts the item's start by `delta_in` and exclusive end by `delta_out`, absorbing the change into the neighbouring items' source ranges (no new items are created); clamps to media available ranges. Mutates in place.

Usage

```
roll(item, delta_in, delta_out)
```

Arguments

`item`	An item (usually a clip).
`delta_in`, `delta_out`	`RationalTime` adjustments.

Value

`item`, invisibly.

Examples

```
trk <- Track("V1")
a <- Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                        RationalTime(24, 24)))
b <- Clip("b", source_range = TimeRange(RationalTime(24, 24),
                                        RationalTime(24, 24)))
append_child(trk, a)
append_child(trk, b)
roll(b, RationalTime(-6, 24), RationalTime(0, 24))
```

schema_name *Schema name of an OTIO object*

Description

The schema type (without the version suffix); unrecognised types report `"UnknownSchema"` (matching OTIO).

Usage

```
schema_name(x)
```

Arguments

x	An OTIO object.

Value

A character scalar.

Examples

```
schema_name(Clip("shot1"))
```

schema_version *Schema version of an OTIO object*

Description

Schema version of an OTIO object

Usage

```
schema_version(x)
```

Arguments

x	An OTIO object.

Value

An integer.

Examples

```
schema_version(Clip("shot1"))
```

SerializableCollection *Construct a SerializableCollection*

Description

A flat, named collection of OTIO objects (not a composition).

Usage

```
SerializableCollection(name = "", children = list(), metadata = NULL)
```

Arguments

name	Collection name.
children	A list of OTIO objects.
metadata	Named list of metadata.

Value

A SerializableCollection.

Examples

```
col <- SerializableCollection("shots", list(Clip("a"), Clip("b")))
length(children(col))
```

set_child *Replace the child at a position (in place)*

Description

Detaches the replaced child and attaches the new one. Returns x invisibly.

Usage

```
set_child(x, index, child)
```

Arguments

x	A composition or collection.
index	1-based position.
child	An OTIO object with no parent.

Value

x, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a"))
set_child(trk, 1, Clip("b"))
name(children(trk)[[1]])
```

---

set_children *Replace all children (in place)*

Description

Available as `set_children(x, kids)` and `set_children(x) <- kids`.

Usage

```
set_children(x, children)

set_children(x) <- value
```

Arguments

x	A composition or collection.
children, value	A list of parentless OTIO objects.

Value

x (invisibly from the function form).

Examples

```
trk <- Track("V1")
set_children(trk, list(Clip("a"), Clip("b")))
length(children(trk))
```

set_media_references *Replace the media references of a clip (in place)*

Description

Replace the media references of a clip (in place)

Usage

```
set_media_references(x, media_references, new_active_key = NULL)
```

Arguments

x	A Clip.
media_references	A named list of media references.
new_active_key	Optional new active key.

Value

x, invisibly.

Examples

```
cl <- Clip("a")
set_media_references(cl, list(DEFAULT_MEDIA = ExternalReference("a.mp4")))
target_url(cl)
```

slice *Slice an item in two at a time*

Description

Splits the item covering time into two adjacent items at that point. Mutates composition in place. A slice exactly at an item boundary is a no-op; slicing through a transition removes it (or errors if remove_transitions is FALSE).

Usage

```
slice(composition, time, remove_transitions = TRUE)
```

Arguments

composition	A Track (or composition).
time	A RationalTime to slice at.
remove_transitions	Remove transitions at time (default TRUE).

Value

composition, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                                     RationalTime(48, 24))))
slice(trk, RationalTime(24, 24))
length(children(trk))
```

---

slide *Slide an item, adjusting the previous item's duration*

Description

Moves the item's start by delta by changing the previous item's duration; the item's own source range is unchanged. No-op for the first item. Mutates in place.

Usage

```
slide(item, delta)
```

Arguments

item	An item (usually a clip).
delta	A RationalTime to slide by.

Value

item, invisibly.

Examples

```
trk <- Track("V1")
append_child(trk, Gap(RationalTime(24, 24)))
cl <- Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                         RationalTime(48, 24)))
append_child(trk, cl)
slide(cl, RationalTime(-6, 24))
```

slip *Slip an item's source range*

Description

Shifts the item's source_range start by delta without changing its duration or any surrounding item; clamps to the media available range when present. Mutates item in place.

Usage

```
slip(item, delta)
```

Arguments

item	An item (usually a clip).
delta	A RationalTime to slip by.

Value

item, invisibly.

Examples

```
cl <- Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                         RationalTime(48, 24)))
slip(cl, RationalTime(6, 24))
source_range(cl)
```

source_range *Get or set an item's source range*

Description

Get or set an item's source range

Usage

```
source_range(x)

source_range(x) <- value
```

Arguments

x	An item (clip, gap, track).
value	A TimeRange.

Value

The item's TimeRange, or NULL if unset.

Examples

```
cl <- Clip("a")
source_range(cl) <- TimeRange(RationalTime(0, 24), RationalTime(48, 24))
source_range(cl)
```

Stack *Construct a Stack*

Description

Parallel tracks (a Timeline's tracks is a Stack).

Usage

```
Stack(name = "tracks", source_range = NULL, metadata = NULL)
```

Arguments

name	Stack name (default "tracks").
source_range	Optional TimeRange.
metadata	Named list of metadata.

Value

A Stack.

Examples

```
Stack()
```

start_time *TimeRange start_time and duration*

Description

TimeRange start_time and duration

Usage

```
start_time(x)

duration(x)
```

Arguments

x	A TimeRange.

Value

A RationalTime.

Examples

```
tr <- TimeRange(RationalTime(0, 24), RationalTime(48, 24))
start_time(tr)
duration(tr)
```

target_url *Target URL of a clip or external reference*

Description

Target URL of a clip or external reference

Usage

```
target_url(x)

target_url(x) <- value
```

Arguments

x	A Clip or ExternalReference.
value	New URL.

Value

The target URL string, or `NULL` if the active reference has no URL.

Examples

```
cl <- Clip("a", ExternalReference("a.mp4"))
target_url(cl) <- "b.mp4"
target_url(cl)
```

`target_url_base` *ImageSequenceReference fields*

Description

ImageSequenceReference fields

Usage

```
target_url_base(x)

target_url_base(x) <- value

name_prefix(x)

name_suffix(x)

start_frame(x)

frame_step(x)

frame_zero_padding(x)
```

Arguments

`x`	An `ImageSequenceReference`.
`value`	New value.

Value

The stored field value; the setter returns `x`.

Examples

```
isr <- ImageSequenceReference("file:///frames", "frame.", ".exr",
                              frame_zero_padding = 4, rate = 24)
target_url_base(isr)
start_frame(isr)
```

target_url_for_image_number *Target URL of the nth image (1-based)*

Description

Target URL of the nth image (1-based)

Usage

```
target_url_for_image_number(x, image_number)
```

Arguments

x	An ImageSequenceReference.
image_number	1-based image index.

Value

The image URL as a string.

Examples

```
isr <- ImageSequenceReference("file:///frames", "frame.", ".exr",
    frame_zero_padding = 4, rate = 24,
    available_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
target_url_for_image_number(isr, 1)
```

time_scalar *Time scalar of a LinearTimeWarp*

Description

Time scalar of a LinearTimeWarp

Usage

```
time_scalar(x)

time_scalar(x) <- value
```

Arguments

x	A LinearTimeWarp.
value	New time scalar.

Value

The numeric time scalar; the setter returns `x`.

Examples

```
w <- LinearTimeWarp(effect_name = "LinearTimeWarp", time_scalar = 2)
time_scalar(w)
```

`Timeline` *Construct a Timeline*

Description

Wraps a `Stack` of tracks. Add tracks with `add_track`.

Usage

```
Timeline(name = "", global_start_time = NULL, metadata = NULL)
```

Arguments

`name`	Timeline name.
`global_start_time`	Optional `RationalTime`.
`metadata`	Named list of metadata.

Value

A `Timeline`.

Examples

```
Timeline("demo")
```

TimeRange *Construct a TimeRange*

Description

A `start_time` plus a `duration`, both `RationalTime`.

Usage

```
TimeRange(start_time, duration)
```

Arguments

`start_time`	A `RationalTime`.
`duration`	A `RationalTime`.

Value

A `TimeRange`.

Examples

```
TimeRange(RationalTime(0, 30), RationalTime(180, 30))
```

TimeTransform *Construct a TimeTransform*

Description

An offset/scale/rate transform (OTIO `TimeTransform`). Note: in RcppOTIO `TimeTransform` is a plain opentime value type that does not serialize to JSON; rotio gives it an `OTIO_SCHEMA` for its own (de)serialization, which is not verified against RcppOTIO JSON.

Usage

```
TimeTransform(offset = RationalTime(0, 1), scale = 1, rate = -1)
```

Arguments

`offset`	A `RationalTime` offset (default 0).
`scale`	Time scale (default 1).
`rate`	Target rate, or -1 to preserve (default -1).

Value

A `TimeTransform`.

Examples

```
TimeTransform(RationalTime(12, 24), scale = 2)
```

---

to_frames *Frame number of a RationalTime*

Description

Integer frame number. With no `rate`, truncates at the time's own rate; with a `rate`, rescales first. Truncates toward zero (opentime `int(value_rescaled_to(rate))`).

Usage

```
to_frames(x, rate = NULL)
```

Arguments

`x`	A `RationalTime`.
`rate`	Optional target rate to rescale to first.

Value

Integer scalar, the frame number.

Examples

```
to_frames(from_seconds(1.5, 24))
to_frames(RationalTime(48, 24), 48)
```

---

to_json_file *Write an OTIO object to a JSON file*

Description

Write an OTIO object to a JSON file

Usage

```
to_json_file(x, file_name, indent = 2, target_schema_versions = NULL)
```

Arguments

`x`	An OTIO object (typically a `Timeline`).
`file_name`	Output path (conventionally `content.otio`).
`indent`	Indent width (default 2).
`target_schema_versions`	Optional schema downgrade targets; see `to_json_string`.

Value

`file_name`, invisibly.

Examples

```
f <- tempfile(fileext = ".otio")
to_json_file(Timeline("demo"), f)
unlink(f)
```

`to_json_string` *Serialize an OTIO object to a JSON string*

Description

Emits canonical OpenTimelineIO JSON. Each object carries its own `OTIO_SCHEMA`, so the output is the standard `.otio` format and parses in any OTIO reader (verify with `validate_with_RcppOTIO`).

Usage

```
to_json_string(x, indent = 2, target_schema_versions = NULL)
```

Arguments

`x`	An OTIO object (typically a `Timeline`).
`indent`	Indent width for pretty-printing (default 2). Use 0 for compact.
`target_schema_versions`	Optional named integer vector mapping schema type to a target version; objects above that version are downgraded using the registered downgrade functions (e.g. `c(Clip = 1L)`).

Value

A JSON string.

Examples

```
to_json_string(Timeline("demo"))
```

to_seconds *Convert a RationalTime to seconds*

Description

Convert a RationalTime to seconds

Usage

```
to_seconds(x)
```

Arguments

x	A RationalTime.

Value

Numeric scalar, the time in seconds.

Examples

```
to_seconds(RationalTime(48, 24))
```

to_time_string *Time string ("HH:MM:SS.sss") for a RationalTime*

Description

Time string ("HH:MM:SS.sss") for a RationalTime

Usage

```
to_time_string(x)
```

Arguments

x	A RationalTime.

Value

Character scalar, the time as "HH:MM:SS.sss".

Examples

```
to_time_string(RationalTime(90, 24))
```

to_timecode *SMPTE timecode for a RationalTime*

Description

SMPTE timecode for a RationalTime

Usage

```
to_timecode(x, rate = NULL, drop_frame = NULL)
```

Arguments

x	A RationalTime.
rate	Timecode rate (default x's rate).
drop_frame	Drop-frame timecode. NULL (default) infers it from the rate (on for 30000/1001 and 60000/1001), matching opentime's InferFromRate; pass TRUE/FALSE to force.

Value

Character scalar, the timecode as "HH:MM:SS:FF" (or ";FF" for drop-frame).

Examples

```
to_timecode(RationalTime(86400, 24))
```

Track *Construct a Track*

Description

An ordered sequence of items. Add children with add_child (or the mutating append_child).

Usage

```
Track(name, kind = "Video", source_range = NULL, metadata = NULL)
```

Arguments

name	Track name.
kind	"Video" (default) or "Audio".
source_range	Optional TimeRange.
metadata	Named list of metadata.

Value

A Track.

Examples

```
Track("V1", kind = "Video")
```

`track_trimmed_to_range` *Trim a track to a time range*

Description

Returns a new track containing each child trimmed to the portion that falls within `trim_range` (in track coordinates); children fully outside are dropped.

Usage

```
track_trimmed_to_range(in_track, trim_range)
```

Arguments

`in_track`	A Track.
`trim_range`	A TimeRange in track coordinates.

Value

A new Track.

Examples

```
trk <- add_child(Track("V1"), Clip("a",
    source_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24))))
track_trimmed_to_range(trk,
    TimeRange(RationalTime(0, 24), RationalTime(24, 24)))
```

tracks *The track stack of a timeline*

Description

The track stack of a timeline

Usage

```
tracks(x)

tracks(x) <- value
```

Arguments

x	A Timeline.
value	A Stack.

Value

The timeline's Stack.

Examples

```
tl <- add_track(Timeline("demo"), Track("V1", kind = "Video"))
length(children(tracks(tl)))
```

Transition *Construct a Transition*

Description

A transition (e.g. a dissolve) between adjacent items on a track.

Usage

```
Transition(name = "", transition_type = "", in_offset = RationalTime(0, 1),
           out_offset = RationalTime(0, 1), metadata = NULL)
```

Arguments

name	Transition name.
transition_type	Transition type (e.g. "SMPTE_Dissolve").
metadata	Named list of metadata.
in_offset, out_offset	RationalTime offsets into the neighbouring clips.

Value

A Transition.

Examples

```
Transition(transition_type = "SMPTE_Dissolve",
           in_offset = RationalTime(5, 30), out_offset = RationalTime(5, 30))
```

`transition_type` *Transition type / offsets*

Description

Transition type / offsets

Usage

```
transition_type(x)

transition_type(x) <- value

in_offset(x)

in_offset(x) <- value

out_offset(x)

out_offset(x) <- value
```

Arguments

x	A Transition.
value	New value.

Value

transition_type returns a string, in_offset and out_offset a RationalTime; setters return x.

Examples

```
tr <- Transition(transition_type = "SMPTE_Dissolve",
                 in_offset = RationalTime(5, 30), out_offset = RationalTime(5, 30))
transition_type(tr)
in_offset(tr)
```

trim *Trim an item, filling the freed time with gap*

Description

Adjusts the item's start by `delta_in` (also extending the previous item) and its exclusive end by `delta_out`, filling now-empty time with a gap (or `fill_template`); an adjacent gap is grown instead. Mutates in place.

Usage

```
trim(item, delta_in, delta_out, fill_template = NULL)
```

Arguments

`item`	An item (usually a clip).
`fill_template`	Optional item to fill freed time with (default a gap).
`delta_in`, `delta_out`	`RationalTime` adjustments.

Value

`item`, invisibly.

Examples

```
trk <- Track("V1")
cl <- Clip("a", source_range = TimeRange(RationalTime(0, 24),
                                         RationalTime(48, 24)))
append_child(trk, cl)
trim(cl, RationalTime(6, 24), RationalTime(0, 24))
source_range(cl)
```

trimmed_range *Trimmed range of an item*

Description

The item's `source_range` if set, else (for a clip) the active media reference's available range.

Usage

```
trimmed_range(x)
```

Arguments

`x`	An item (clip, gap).

Value

A TimeRange.

Examples

```
cl <- Clip("a",
    source_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
trimmed_range(cl)
```

`trimmed_range_in_parent` *Range of an item within its parent, trimmed by the parent's source range*

Description

Range of an item within its parent, trimmed by the parent's source range

Usage

```
trimmed_range_in_parent(x)
```

Arguments

x	An item with a parent.

Value

A TimeRange.

Examples

```
trk <- Track("V1",
    source_range = TimeRange(RationalTime(12, 24), RationalTime(24, 24)))
cl <- Clip("a",
    source_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
append_child(trk, cl)
trimmed_range_in_parent(cl)
```

type_version_map *Current OTIO schema versions*

Description

A named integer vector mapping each schema type to its current version, matching OpenTimelineIO 0.18.1.

Usage

```
type_version_map()
```

Value

A named integer vector.

Examples

```
type_version_map()["Clip"]
```

validate_with_RcppOTIO *Validate an OTIO object against the real OpenTimelineIO library*

Description

Serializes `x` to OTIO JSON, parses it with `RcppOTIO` (the exact libopentimelineio binding), and re-serializes through RcppOTIO. Confirms the emitted JSON is accepted by real OTIO. Requires the optional `RcppOTIO` package.

Usage

```
validate_with_RcppOTIO(x)
```

Arguments

`x`	An OTIO object (typically a `Timeline`).

Value

Invisibly, a list with `status` (`"valid"` or `"unverified"`) and, when validated, the RcppOTIO-normalized JSON.

Examples

```
validate_with_RcppOTIO(Timeline("demo"))
```

value *RationalTime value and rate*

Description

RationalTime value and rate

Usage

```
value(x)

rate(x)
```

Arguments

x	A RationalTime.

Value

Numeric scalar, the time's value or rate.

Examples

```
value(RationalTime(180, 30))
rate(RationalTime(180, 30))
```

video_tracks *Video / audio tracks of a timeline*

Description

Video / audio tracks of a timeline

Usage

```
video_tracks(x)

audio_tracks(x)
```

Arguments

x	A Timeline.

Value

A list of Tracks of the matching kind.

Examples

```
tl <- add_track(Timeline("demo"), Track("V1", kind = "Video"))
video_tracks(tl)
audio_tracks(tl)
```

`visible` *Is an item visible?*

Description

A Gap is never visible; a Transition always is; any other item is visible when enabled (matching OTIO).

Usage

```
visible(x)
```

Arguments

x	An item.

Value

Logical scalar.

Examples

```
visible(Gap(RationalTime(24, 24)))
visible(Clip("a", ExternalReference("a.mp4")))
```

`visible_range` *Visible range of an item (including adjacent transitions)*

Description

Visible range of an item (including adjacent transitions)

Usage

```
visible_range(x)
```

Arguments

x	An item with a parent.

Value

A TimeRange.

Examples

```
trk <- Track("V1")
cl <- Clip("a",
    source_range = TimeRange(RationalTime(0, 24), RationalTime(48, 24)))
append_child(trk, cl)
visible_range(cl)
```

`write_otiod` *Write an OTIOD bundle*

Description

Writes `timeline` to `<dir>/content.otio` and ensures a `media/` subdirectory exists. Optionally copies `media` files into it. Media references inside `timeline` should already be relative paths (e.g. `"media/clip01.mp4"`); this writer does not rewrite them.

Usage

```
write_otiod(timeline, dir, media = NULL, indent = 2)
```

Arguments

`timeline`	A Timeline.
`dir`	Bundle directory (created if needed).
`media`	Optional character vector of media files to copy into `<dir>/media/`.
`indent`	Indent width for content.otio (default 2).

Value

The path to the written `content.otio`, invisibly.

Examples

```
d <- tempfile("bundle")
write_otiod(Timeline("demo"), d)
dir(d)
unlink(d, recursive = TRUE)
```

Index