HED JavaScript

Release 3.10.0

HED Working Group

Oct 18, 2023

CONTENTS:

_static/images/croppedWideLogo.png

Links

- PDF docs
- Source code

Note: this is a work in progress. More information is coming.

CHAPTER

ONE

BIDS

Bids functionality

1.1 Schema

Intentionally Blank

1.2 tsvParse

parseTSV(*contents*)

Parse a TSV file.

Arguments

- **contents** (*string*) – The contents of a TSV file.

Returns

Object – The parsed contents of the TSV file.

1.3 types

class BidsData()

Note: Deprecated: Will be removed in v4.0.0.

Base class for BIDS data.

BidsData.**definitions**

Note: Deprecated: Will be removed in v4.0.0.

type: Map.<string, ParsedHedGroup>

A Mapping from definition names to their associated ParsedHedGroup objects.

BidsData.**hedIssues**

Note: Deprecated: Will be removed in v4.0.0.

type: Array.<Issue>

A list of HED validation issues. This will be converted to BidsIssue objects later on.

BidsData.**parsedStringMapping**

Note: Deprecated: Will be removed in v4.0.0.

type: Map.<string, ParsedHedString>

A mapping from unparsed HED strings to ParsedHedString objects.

class BidsFile(*name*, *file*)

A BIDS file.

BidsFile.**file**

type: object

The file object representing this file data. This is used to generate BidsIssue objects.

BidsFile.**name**

type: string

The name of this file.

class BidsJsonFile()

A BIDS JSON file.

BidsJsonFile.**BidsJsonFile**

BidsJsonFile.**jsonData**

type: object

This file's JSON data.

class BidsTsvFile(*name*, *tsvData*, *file*, *potentialSidecars*, *mergedDictionary*)

A BIDS TSV file.

Constructor.

Arguments

- **name** (*string*) – The name of the TSV file.
- **tsvData** (*Object|string*) – This file's TSV data.
- **file** (*object*) – The file object representing this file.
- **potentialSidecars** (*Array.<string>*) – The list of potential JSON sidecars.
- **mergedDictionary** (*object*) – The merged sidecar data.

BidsTsvFile.**BidsTsvFile**

Constructor.

BidsTsvFile.**hedColumnHedStrings**

type: Array.<string>

HED strings in the "HED" column of the TSV data.

BidsTsvFile.**mergedSidecar**

type: BidsSidecar

The pseudo-sidecar object representing the merged sidecar data.

BidsTsvFile.**parsedTsv**

type: Object

This file's parsed TSV data.

BidsTsvFile.**potentialSidecars**

type: Array.<string>

The list of potential JSON sidecars.

BidsTsvFile.**sidecarHedData**

type: Map.<string, (string|Object.<string, string>)>

The extracted HED data for the merged pseudo-sidecar.

class BidsEventFile(*name*, *potentialSidecars*, *mergedDictionary*, *tsvData*, *file*)

A BIDS events.tsv file.

Constructor.

Arguments

- **name** (*string*) – The name of the event TSV file.
- **potentialSidecars** (*Array.<string>*) – The list of potential JSON sidecars.
- **mergedDictionary** (*object*) – The merged sidecar data.
- **tsvData** (*Object|string*) – This file's TSV data.
- **file** (*object*) – The file object representing this file.

BidsEventFile.**BidsEventFile**

Constructor.

class BidsTabularFile(*name*, *potentialSidecars*, *mergedDictionary*, *tsvData*, *file*)

A BIDS TSV file other than an events.tsv file.

Constructor.

Arguments

- **name** (*string*) – The name of the TSV file.
- **potentialSidecars** (*Array.<string>*) – The list of potential JSON sidecars.
- **mergedDictionary** (*object*) – The merged sidecar data.
- **tsvData** (*Object|string*) – This file's TSV data.
- **file** (*object*) – The file object representing this file.

BidsTabularFile.**BidsTabularFile**

Constructor.

class BidsSidecar(*name*, *sidecarData*, *file*)

Constructor.

Arguments

- **name** (*string*) – The name of the sidecar file.
- **sidecarData** (*Object*) – The raw JSON data.
- **file** (*Object*) – The file object representing this file.

1.4 utils

sidecarValueHasHed(*sidecarValue*)

Determine whether a sidecar value has HED data.

Arguments

- **sidecarValue** (*object*) – A BIDS sidecar value.

Returns

boolean – Whether the sidecar value has HED data.

1.5 validate

sidecarValueHasHed(*sidecarValue*)

Determine whether a sidecar value has HED data.

Arguments

- **sidecarValue** (*object*) – A BIDS sidecar value.

Returns

boolean – Whether the sidecar value has HED data.

validateBidsDataset(*dataset*, *schemaDefinition*)

Validate a BIDS dataset.

Arguments

- **dataset** (*BidsDataset*) – The BIDS dataset.
- **schemaDefinition** (*object*) – The version spec for the schema to be loaded.

Returns

Promise.<Array.<BidsIssue>> – Any issues found.

validateFullDataset(*dataset*, *hedSchemas*)

Validate a full BIDS dataset using a HED schema collection.

Arguments

- **dataset** (*BidsDataset*) – A BIDS dataset.
- **hedSchemas** (Schemas) – A HED schema collection.

Returns

Promise.<Array.<BidsIssue>>|Promise.<never> – Any issues found.

validateBidsTsvFile(*tsvFileData*, *hedSchemas*)

Validate a BIDS TSV file.

Arguments

- **tsvFileData** (BidsTsvFile) – A BIDS TSV file.
- **hedSchemas** (Schemas) – A HED schema collection.

Returns

Array.<BidsIssue> – Any issues found.

validateSidecars(*sidecarData*, *hedSchemas*)

Validate a collection of BIDS sidecars.

Arguments

- **sidecarData** (*Array.<BidsSidecar>*) – A collection of BIDS sidecars.
- **hedSchemas** (Schemas) – A HED schema collection.

Returns

validateHedColumn(*eventData*, *hedSchemas*)

Validate the HED columns of a collection of BIDS event TSV files.

Arguments

- **eventData** (*Array.<BidsEventFile>*) – A collection of BIDS event TSV files.
- **hedSchemas** (Schemas) – A HED schema collection.

Returns

parseTsvHed(*tsvFileData*)

Combine the BIDS sidecar HED data into a BIDS TSV file's HED data.

Arguments

- **tsvFileData** (BidsTsvFile) – A BIDS TSV file.

Returns

validateCombinedDataset(*hedStrings*, *hedSchemas*, *tsvFileData*)

Validate the HED data in a combined event TSV file/sidecar BIDS data collection.

Arguments

- **hedStrings** (*Array.<string>*) – The HED strings in the data collection.
- **hedSchemas** (Schemas) – The HED schema collection to validate against.
- **tsvFileData** (BidsTsvFile) – The BIDS event TSV file being validated.

Returns

Array.<BidsHedIssue> – Any issues found.

validateStrings(*hedStrings*, *hedSchemas*, *fileObject*, *settings*)

Validate a set of HED strings.

Arguments

- **hedStrings** (*Array.<string>*) – The HED strings to validate.
- **hedSchemas** (Schemas) – The HED schema collection to validate against.

- **fileObject** (*Object*) – A BIDS-format file object used to generate {@link BidsHedIssue} objects.
- **settings** (*Object*) – Options to pass to {@link validateHedString}.

Returns

Array.<BidsHedIssue> – Any issues found.

convertHedIssuesToBidsIssues(*hedIssues*, *file*)

Convert one or more HED issues into BIDS-compatible issues.

Arguments

- **hedIssues** (*IssueError|Array.<Issue>*) – One or more HED-format issues.
- **file** (*Object*) – A BIDS-format file object used to generate {@link BidsHedIssue} objects.

Returns

Array.<BidsHedIssue> – The passed issue(s) in BIDS-compatible format.

CHAPTER

TWO

COMMON

2.1 schema

2.1.1 types

class Schema(*xmlData*)

An imported HED schema object.

Constructor.

Arguments

- **xmlData** (*object*) – The schema XML data.

Schema.**Schema**

Constructor.

Schema.**generation**

type: Number

The HED generation of this schema.

Schema.**library**

type: string

The HED library schema name.

Schema.**prefix**

type: string

This schema's prefix in the active schema set.

Schema.**version**

type: string

The HED schema version.

Schema.**xmlData**

type: Object

The schema XML data.

Schema.**tagHasAttribute**(*tag*, *tagAttribute*)

Determine if a HED tag has a particular attribute in this schema.

Arguments

- **tag** (*string*) – The HED tag to check.
- **tagAttribute** (*string*) – The attribute to check for.

Returns
boolean – Whether this tag has this attribute.

`class Hed2Schema(xmlData, attributes)`

Hed2Schema class

Constructor.

Arguments

- **xmlData** (*object*) – The schema XML data.
- **attributes** (SchemaAttributes) – A description of tag attributes.

`Hed2Schema.Hed2Schema`

Constructor.

`Hed2Schema.attributes`

type: SchemaAttributes

The description of tag attributes.

`Hed2Schema.tagHasAttribute(tag, tagAttribute)`

Determine if a HED tag has a particular attribute in this schema.

Arguments

- **tag** (*string*) – The HED tag to check.
- **tagAttribute** (*string*) – The attribute to check for.

Returns
boolean – Whether this tag has this attribute.

`class Hed3Schema(xmlData, entries, mapping)`

Hed3Schema class

Constructor.

Arguments

- **xmlData** (*object*) – The schema XML data.
- **entries** (SchemaEntries) – A collection of schema entries.
- **mapping** (Mapping) – A mapping between short and long tags.

`Hed3Schema.Hed3Schema`

Constructor.

`Hed3Schema.entries`

type: SchemaEntries

The collection of schema entries.

`Hed3Schema.mapping`

type: Mapping

The mapping between short and long tags.

Hed3Schema.**tagHasAttribute**(*tag*, *tagAttribute*)

Determine if a HED tag has a particular attribute in this schema.

Arguments

- **tag** (*string*) – The HED tag to check.
- **tagAttribute** (*string*) – The attribute to check for.

Returns

boolean – Whether this tag has this attribute.

class Schemas(*schemas*)

The collection of active HED schemas.

Constructor.

Arguments

- **schemas** (*Schema|Map.<string, Schema>|null*) – The imported HED schemas.

Schemas.**Schemas**

Constructor.

Schemas.**baseSchema**

The base schema, i.e. the schema with no nickname, if one is defined.

Schemas.**generation**

type: Number

The HED generation of this schema.

If baseSchema is null, generation is set to 0.

Schemas.**isHed3**

Whether this schema collection comprises HED 3 schemas.

Schemas.**isSyntaxOnly**

Whether this schema collection is for syntactic validation only.

Schemas.**librarySchemas**

The library schemas, i.e. the schema with nicknames, if any are defined.

Schemas.**schemas**

type: Map.<string, Schema>|null

The imported HED schemas.

The empty string key ("") corresponds to the schema with no nickname, while other keys correspond to the respective nicknames.

This field is null for syntax-only validation.

Schemas.**standardSchema**

The standard schema, i.e. primary schema implementing the HED standard, if one is defined.

Schemas.**getSchema**(*schemaName*)

Return the schema with the given nickname.

Arguments

- **schemaName** (*string*) – A nickname in the schema set.

Returns
Schema – The schema object corresponding to that nickname.

class SchemaSpec(*nickname*, *version*, *library*, *localPath*)

A schema version specification.

Constructor.

Arguments

- **nickname** (*string*) – The nickname of this schema.
- **version** (*string*) – The version of this schema.
- **library** (*string*) – The library name of this schema.
- **localPath** (*string*) – The local path for this schema.

SchemaSpec.**SchemaSpec**

Constructor.

SchemaSpec.**library**

type: string

The library name of this schema.

SchemaSpec.**localName**

Compute the name for the bundled copy of this schema.

SchemaSpec.**localPath**

type: string

The local path for this schema.

SchemaSpec.**nickname**

type: string

The nickname of this schema.

SchemaSpec.**path**

Alias to old name of localPath.

SchemaSpec.**version**

type: string

The version of this schema.

class SchemasSpec()

A specification mapping schema nicknames to SchemaSpec objects.

Constructor.

SchemasSpec.**SchemasSpec**

Constructor.

SchemasSpec.**data**

type: Map.<string, SchemaSpec>

The specification mapping data.

SchemasSpec.**addSchemaSpec**(*schemaSpec*)

Add a schema to this specification.

Arguments

- **schemaSpec** (SchemaSpec) – A schema specification.

Returns

SchemasSpec|map – This object.

SchemasSpec.**isDuplicate**(*schemaSpec*)

Determine whether this specification already has a schema with the given nickname.

Arguments

- **schemaSpec** (SchemaSpec) – A schema specification with a nickname.

Returns

boolean – Whether the nickname exists in this specification.

2.1.2 loader

loadSchema(*schemaDef*, *useFallback*, *reportNoFallbackError*)

Load schema XML data from a schema version or path description.

Arguments

- **schemaDef** (SchemaSpec) – The description of which schema to use.
- **useFallback** (*boolean*) – Whether to use a bundled fallback schema if the requested schema cannot be loaded.
- **reportNoFallbackError** (*boolean*) – Whether to report an error on a failed schema load when no fallback was used.

Returns

loadSchemaFromSpec(*schemaDef*)

Load schema XML data from a schema version or path description.

Arguments

- **schemaDef** (SchemaSpec) – The description of which schema to use.

Returns

loadPromise(*schemaDef*)

Choose the schema Promise from a schema version or path description.

Arguments

- **schemaDef** (SchemaSpec) – The description of which schema to use.

Returns

Promise.<object> – The schema XML data or an error.

loadRemoteSchema(*schemaDef*)

Load schema XML data from the HED GitHub repository.

Arguments

- **schemaDef** (SchemaSpec) – The standard schema version to load.

Returns
Promise.<object> – The schema XML data.

`loadLocalSchema`(*path*)

Load schema XML data from a local file.

Arguments

- **path** (*string*) – The path to the schema XML data.

Returns
Promise.<object> – The schema XML data.

`loadBundledSchema`(*schemaDef*)

Load schema XML data from a bundled file.

Arguments

- **schemaDef** (SchemaSpec) – The description of which schema to use.

Returns
Promise.<object> – The schema XML data.

`loadSchemaFile`(*xmlDataPromise*, *issueCode*, *issueArgs*)

Actually load the schema XML file.

Arguments

- **xmlDataPromise** (*Promise.<string>*) – The Promise containing the unparsed XML data.
- **issueCode** (*string*) – The issue code.
- **issueArgs** (*Object.<string, string>*) – The issue arguments passed from the calling function.

Returns
Promise.<object> – The parsed schema XML data.

`parseSchemaXML`(*data*)

Parse the schema XML data.

Arguments

- **data** (*string*) – The XML data.

Returns
Promise.<object> – The schema XML data.

2.1.3 config

Intentionally blank for now

2.2 issues

2.2.1 issues

class Issue(*internalCode*, *hedCode*, *level*, *message*)

A HED validation error or warning.

Constructor.

Arguments

- **internalCode** (*string*) – The internal error code.
- **hedCode** (*string*) – The HED 3 error code.
- **level** (*string*) – The issue level (error or warning).
- **message** (*string*) – The detailed error message.

Issue.**Issue**

Constructor.

Issue.**code**

Note: Deprecated.

type: string

Also the internal error code.

TODO: This is kept for backward compatibility until the next major version bump.

Issue.**hedCode**

type: string

The HED 3 error code.

Issue.**internalCode**

type: string

The internal error code.

Issue.**level**

type: string

The issue level (error or warning).

Issue.**message**

type: string

The detailed error message.

Issue.**toString**()

Override of {@link Object.prototype.toString}.

Returns

string – This issue's message.

generateIssue(*internalCode*, *parameters*)

Generate a new issue object.

Arguments

- **internalCode** (*string*) – The internal error code.
- **parameters** (*Object.<string, (string|Array.<number>)>*) – The error string parameters.

Returns

Issue – An object representing the issue.

2.2.2 data

Intentionally blank for now

CHAPTER

THREE

CONVERTER

3.1 types

class TagEntry(*shortTag*, *longTag*)

A tag dictionary entry.

Constructor.

Arguments

- **shortTag** (*string*) – The short version of the tag.
- **longTag** (*string*) – The long version of the tag.

TagEntry.**TagEntry**

Constructor.

TagEntry.**longFormattedTag**

type: string

The formatted long version of the tag.

TagEntry.**longTag**

type: string

The long version of the tag.

TagEntry.**shortTag**

type: string

The short version of the tag.

TagEntry.**takesValue**

type: boolean

Whether this tag takes a value.

class Mapping(*mappingData*)

A short-to-long mapping.

Constructor.

Arguments

- **mappingData** (*Map.<string, TagEntry>*) – A dictionary mapping forms to TagEntry instances.

Mapping.**Mapping**

Constructor.

Mapping.**mappingData**

type: Map.<string, TagEntry>

A dictionary mapping forms to TagEntry instances.

3.2 schema

buildMappingObject(*entries*)

Build a short-long mapping object from schema XML data.

Arguments

- **entries** (SchemaEntries) – The schema XML data.

Returns

Mapping – The mapping object.

schema.**buildSchema**(*schemaDef*)

Note: Deprecated.

Build a schema container object containing a short-long mapping from a base schema version or path description.

Arguments

- **schemaDef** (*Object*) – The description of which schema to use.

Returns

Promise.<never>|Promise.<Schemas> – The schema container object or an error.

3.3 issues

Intentionally blank for now

3.4 splitHedString

splitHedString(*hedString*)

Split a HED string into delimiters and tags.

Arguments

- **hedString** (*string*) – The HED string to split.

Returns

Array.<Array> – A list of string parts. The boolean is true if the part is a tag and false if it is a delimiter. The numbers are the bounds of the part.

3.5 converter

removeSlashesAndSpaces(*hedString*)

Remove extra slashes and spaces from a HED string.

Arguments

- **hedString** (*string*) – The HED string to clean.

Returns

string – The cleaned HED string.

convertTagToLong(*schema*, *hedTag*, *hedString*, *offset*)

Convert a HED tag to long form.

The seemingly redundant code for duplicate tag entries (which are errored out on for HED 3 schemas) allow for similar HED 2 validation with minimal code duplication.

Arguments

- **schema** (Schema) – The schema object containing a short-to-long mapping.
- **hedTag** (*string*) – The HED tag to convert.
- **hedString** (*string*) – The full HED string (for error messages).
- **offset** (*number*) – The offset of this tag within the HED string.

Returns

convertTagToShort(*schema*, *hedTag*, *hedString*, *offset*)

Convert a HED tag to short form.

Arguments

- **schema** (Schema) – The schema object containing a short-to-long mapping.
- **hedTag** (*string*) – The HED tag to convert.
- **hedString** (*string*) – The full HED string (for error messages).
- **offset** (*number*) – The offset of this tag within the HED string.

Returns

convertPartialHedStringToLong(*schema*, *partialHedString*, *fullHedString*, *offset*)

Convert a partial HED string to long form.

This is for the internal string parsing for the validation side.

Arguments

- **schema** (Schema) – The schema object containing a short-to-long mapping.
- **partialHedString** (*string*) – The partial HED string to convert to long form.
- **fullHedString** (*string*) – The full HED string.
- **offset** (*number*) – The offset of the partial HED string within the full string.

Returns

convertHedStringToLong(*schemas*, *hedString*)

Note: Deprecated.

Convert a HED string to long form.

Arguments

- **schemas** (Schemas) – The schema container object containing short-to-long mappings.
- **hedString** (*string*) – The HED tag to convert.

Returns

convertHedStringToShort(*schemas*, *hedString*)

Note: Deprecated.

Convert a HED string to short form.

Arguments

- **schemas** (Schemas) – The schema container object containing short-to-long mappings.
- **hedString** (*string*) – The HED tag to convert.

Returns

CHAPTER

FOUR

UTILS

4.1 xml2js

setParent(*node*, *parent*)

Handle top level of parent-setting recursion before passing to setNodeParent.

Arguments

- **node** (*object*) – The child node.
- **parent** (*object*) – The parent node.

4.2 map

Intentionally blank for now # .. js:autofunction:: filterNonEqualDuplicates

4.3 types

class MemoizerMixin()

Mix-in/superclass for property memoization until we can get away with private fields.

class Memoizer()

Memoizer class

4.4 files

readFile(*fileName*)

Read a local file.

Arguments

- **fileName** (*string*) – The file path.

Returns

Promise.<unknown> – A promise with the file contents.

`readHTTPSFile`(*url*)

Read a remote file using HTTPS.

Arguments

- **url** (*string*) – The remote URL.

Returns

Promise.<string> – A promise with the file contents.

4.5 xpath

`find`(*element*, *query*)

Execute an XPath query on an xml2js object.

Arguments

- **element** (*object*) – An xml2js element.
- **query** (*string*) – An XPath query.

Returns

Array.<object> – An array of xml2js elements matching the query.

`parseXPath`(*query*)

Parse an XPath query.

This is a minimal parser only suitable for this package.

Arguments

- **query** (*string*) – An XPath query.

Returns

object – The parsed search parameters.

`search`(*element*, *elementName*, *attributeName*)

Search for children of an element with a given name and attribute.

Arguments

- **element** (*object*) – An xml2js element.
- **elementName** (*string*) – The element name.
- **attributeName** (*string*) – The attribute name.

Returns

Array.<object> – An array of xml2js elements with the given name and attribute.

4.6 string

stringIsEmpty(*string*)

Check if a string is empty or only whitespace.

Arguments

- **string** (*string*) – The string to check.

Returns

boolean – Whether the string is empty.

getCharacterCount(*string*, *characterToCount*)

Get number of instances of an character in a string.

Arguments

- **string** (*string*) – The string to search.
- **characterToCount** (*string*) – The character to search for.

Returns

number – The number of instances of the character in the string.

capitalizeString(*string*)

Get a copy of a string with the first letter capitalized.

Arguments

- **string** (*string*) – The string to capitalize.

Returns

string – The capitalized string.

isClockFaceTime(*timeString*)

Determine if a string is a valid clock face time.

Arguments

- **timeString** (*string*) – The string to check.

Returns

boolean – Whether the string is a valid clock face time.

isDateTime(*dateTimeString*)

Determine if a string is a valid date-time.

Arguments

- **dateTimeString** (*string*) – The string to check.

Returns

boolean – Whether the string is a valid date-time.

isNumber(*numericString*)

Determine if a string is a valid number.

Arguments

- **numericString** (*string*) – The string to check.

Returns

boolean – Whether the string is a valid number.

stringTemplate(*strings*, *keys*)

Parse a template literal string.

Copied from https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template_literals.

Arguments

- **strings** (*Array.<string>*) – The literal parts of the template string.
- **keys** (*number|string*) – The keys of the closure arguments.

Returns

function – A closure to fill the string template.

4.7 hedData

getGenerationForSchemaVersion(*version*)

Determine the HED generation for a base schema version number.

Arguments

- **version** (*string*) – A HED base schema version number.

Returns

number – The HED generation the base schema belongs to.

getParsedParentTags(*hedSchemas*, *shortTag*)

Get the parent tag objects for a given short tag.

Arguments

- **hedSchemas** (Schemas) – The HED schema collection.
- **shortTag** (*string*) – A short-form HED 3 tag.

Returns

Map.<Schema, ParsedHedTag> – A Map mapping a {@link Schema} to a {@link ParsedHedTag} object representing the full tag.

4.8 array

getElementCount(*array*, *elementToCount*)

Get number of instances of an element in an array.

Arguments

- **array** (*Array*) – The array to search.
- **elementToCount** (*) – The element to search for.

Returns

number – The number of instances of the element in the array.

asArray(*array*)

Return a scalar as a singleton array and an array as-is.

Arguments

- **array** (*T|Array.<T>*) – An array or scalar.

Returns
Array.<T> – The original array or a singleton array of the scalar.

recursiveMap(*fn*, *array*)

Apply a function recursively to an array.

Arguments

- **fn** (*function*) – The function to apply.
- **array** (*Array.<T>*) – The array to map.

Returns
Array.<U> – The mapped array.

4.9 hedStrings

replaceTagNameWithPound()

Replace the end of a HED tag with a pound sign.

.. js:autofunction:: getTagSlashIndices

getTagLevels(*tag*)

Get the levels of a tag.

Arguments

- **tag** (*string*) – A HED tag string.

Returns
Array.<string> – The levels of this tag.

hedStrings.**getTagName**(*tag*, *character*)

Get the last part of a HED tag.

Arguments

- **tag** (*string*) – A HED tag
- **character** (*string*) – The character to use as a separator.

Returns
string – The last part of the tag using the given separator.

hedStrings.**getParentTag**()

Get the HED tag prefix (up to the last slash).

hedStringIsAGroup(*hedString*)

Determine whether a HED string is a group (surrounded by parentheses).

Arguments

- **hedString** (*string*) – A HED string.

loadSchemaFile(*xmlDataPromise*, *issueCode*, *issueArgs*)

Actually load the schema XML file.

Arguments

- **xmlDataPromise** (*Promise.<string>*) – The Promise containing the unparsed XML data.
- **issueCode** (*string*) – The issue code.

- **issueArgs** (*Object.<string, string>*) – The issue arguments passed from the calling function.

Returns

Promise.<object> – The parsed schema XML data.

removeGroupParentheses(*tagGroup*)

Return a copy of a group tag with the surrounding parentheses removed.

Arguments

- **tagGroup** (*string*) – A tag group string.

CHAPTER

FIVE

VALIDATOR

5.1 dataset

parseDefinitions(*parsedHedStrings*)

Parse the dataset's definitions and evaluate labels in the dataset.

Arguments

- **parsedHedStrings** (*Array.<ParsedHedString>*) – The dataset's parsed HED strings.

Returns

checkGroupForTemporalOrder(*parsedGroup*, *activeScopes*)

Check a parsed HED group for its onset and offset ordering.

Arguments

- **parsedGroup** (ParsedHedGroup) – A parsed HED group.
- **activeScopes** (*Set.<string>*) – The active duration scopes, represented by the groups' canonical Def tags.

Returns

Array.<Issue> – Any issues found.

validateTemporalOrder(*hedStrings*, *hedSchemas*)

Validate onset and offset ordering.

Arguments

- **hedStrings** (*Array.<ParsedHedString>*) – The dataset's HED strings.
- **hedSchemas** (Schemas) – The HED schema container object.

Returns

Array.<Issue> – Any issues found.

validateDataset(*definitions*, *hedStrings*, *hedSchemas*)

Perform dataset-level validation on a HED dataset.

Arguments

- **definitions** (*Definitions*) – The parsed dataset definitions.
- **hedStrings** (*Array.<ParsedHedString>*) – The dataset's HED strings.
- **hedSchemas** (Schemas) – The HED schema container object.

Returns

Array.<Issue> – Whether the HED dataset is valid and any issues found.

validateHedEvents(*parsedHedStrings*, *hedSchemas*, *definitions*, *settings*)

Validate a group of HED strings.

Arguments

- **parsedHedStrings** (*Array.<string>|Array.<ParsedHedString>*) – The dataset's parsed HED strings.
- **hedSchemas** (Schemas) – The HED schema container object.
- **definitions** (*Map.<string, ParsedHedGroup>*) – The dataset's parsed definitions.
- **settings** (*Object*) – The configuration settings for validation.

Returns

validateHedDataset(*hedStrings*, *hedSchemas*, *checkForWarnings*)

Validate a HED dataset.

Arguments

- **hedStrings** (*Array.<string>*) – The dataset's HED strings.
- **hedSchemas** (Schemas) – The HED schema container object.
- **checkForWarnings** (*boolean*) – Whether to check for warnings or only errors.

Returns

validateHedDatasetWithContext(*hedStrings*, *contextHedStrings*, *hedSchemas*, *checkForWarnings*)

Validate a HED dataset with additional context.

Arguments

- **hedStrings** (*Array.<string>*) – The dataset's HED strings.
- **contextHedStrings** (*Array.<string>*) – The dataset's context HED strings.
- **hedSchemas** (Schemas) – The HED schema container object.
- **checkForWarnings** (*boolean*) – Whether to check for warnings or only errors.

Returns

5.2 schema

5.2.1 init

isHed3Schema(*xmlData*)

Determine whether a HED schema is based on the HED 3 spec.

Arguments

- **xmlData** (*object*) – HED XML data.

Returns

boolean – Whether the schema is a HED 3 schema.

buildSchemaAttributesObject(*xmlData*)

Build a schema attributes object from schema XML data.

Arguments

- **xmlData** (*object*) – The schema XML data.

Returns
SchemaAttributes|SchemaEntries – The schema attributes object.

buildSchemaObject(*xmlData*)

Build a single schema container object from a base schema version or path description.

Arguments

- **xmlData** (*object*) – The schema's XML data

Returns
Schema – The HED schema object.

init.**buildSchema**(*schemaDef*, *useFallback*)

Note: Deprecated.

Build a schema collection object from a schema specification.

Arguments

- **schemaDef** (*Object*) – The description of which schemas to use.
- **useFallback** (*boolean*) – Whether to use a bundled fallback schema if the requested schema cannot be loaded.

Returns
Promise.<never>|Promise.<Schemas> – The schema container object or an error.

buildSchemas(*schemaSpecs*)

Build a schema collection object from a schema specification.

Arguments

- **schemaSpecs** (*Map.<string, SchemaSpec>|SchemasSpec*) – The description of which schemas to use.

Returns

5.2.2 types

class SchemaEntries(*schemaParser*)

SchemaEntries class

Constructor.

Arguments

- **schemaParser** (*Hed3SchemaParser*) – A constructed schema parser.

SchemaEntries.**SchemaEntries**

Constructor.

SchemaEntries.**SIUnitModifiers**

Get the schema's SI unit modifiers.

SchemaEntries.**SIUnitSymbolModifiers**

Get the schema's SI unit symbol modifiers.

SchemaEntries.**allUnits**

Get a map of all of this schema's units.

SchemaEntries.**attributes**

type: SchemaEntryManager

The schema's attributes.

SchemaEntries.**definitions**

type: Map.<string, SchemaEntryManager>

The schema's definitions.

SchemaEntries.**properties**

type: SchemaEntryManager

The schema's properties.

SchemaEntries.**unitClassMap**

Get the schema's unit classes.

SchemaEntries.**tagHasAttribute**(*tag*, *tagAttribute*)

Determine if a HED tag has a particular attribute in this schema.

Arguments

- **tag** (*string*) – The HED tag to check.
- **tagAttribute** (*string*) – The attribute to check for.

Returns

boolean – Whether this tag has this attribute.

class SchemaEntryManager(*definitions*)

A manager of {@link SchemaEntry} objects.

Constructor.

Arguments

- **definitions** (*Map.<string, T>*) – A map of schema entry definitions.

SchemaEntryManager.**SchemaEntryManager**

Constructor.

SchemaEntryManager.**_definitions**

type: Map.<string, T>

The definitions managed by this entry manager.

SchemaEntryManager.**keys**()

Iterator over the entry manager's keys.

Returns

IterableIterator.<string> –

SchemaEntryManager.**values**()

Iterator over the entry manager's keys.

Returns

IterableIterator.<T> –

class SchemaEntry()

SchemaEntry class

SchemaEntry.**SchemaEntry**

SchemaEntry.**_name**

type: string

The name of this schema entry.

SchemaEntry.**name**

The name of this schema entry.

SchemaEntry.**hasAttributeName**(*attributeName*)

Whether this schema entry has this attribute (by name).

This method is a stub to be overridden in {@link SchemaEntryWithAttributes}.

Arguments

- **attributeName** (*string*) – The attribute to check for.

Returns

boolean – Whether this schema entry has this attribute.

class SchemaProperty()

A schema property.

SchemaProperty.**SchemaProperty**

SchemaProperty.**_propertyType**

type: string

The type of the property.

SchemaProperty.**isCategoryProperty**

Whether this property describes a schema category.

SchemaProperty.**isRoleProperty**

Whether this property describes a role.

SchemaProperty.**isTypeProperty**

Whether this property describes a data type.

class SchemaAttribute(*name*, *properties*)

A schema attribute.

Constructor.

Arguments

- **name** (*string*) – The name of the schema attribute.
- **properties** (*Array.<SchemaProperty>*) – The properties assigned to this schema attribute.

SchemaAttribute.**SchemaAttribute**

Constructor.

SchemaAttribute.**_categoryProperties**

type: Set.<SchemaProperty>

The categories of elements this schema attribute applies to.

SchemaAttribute.**_roleProperties**

type: Set.<SchemaProperty>

The set of role properties for this schema attribute.

SchemaAttribute.**_typeProperty**

type: SchemaProperty

The data type of this schema attribute.

SchemaAttribute.**categoryProperty**

The categories of elements this schema attribute applies to.

SchemaAttribute.**roleProperties**

The set of role properties for this schema attribute.

SchemaAttribute.**typeProperty**

The data type property of this schema attribute.

class SchemaEntryWithAttributes(*name*, *booleanAttributes*, *valueAttributes*)

SchemaEntryWithAttributes class

SchemaEntryWithAttributes.**booleanAttributeNames**

type: Set.<string>

The set of boolean attribute names this schema entry has.

SchemaEntryWithAttributes.**booleanAttributes**

type: Set.<SchemaAttribute>

The set of boolean attributes this schema entry has.

SchemaEntryWithAttributes.**valueAttributeNames**

type: Map.<string, *>

The collection of value attribute names this schema entry has.

SchemaEntryWithAttributes.**valueAttributes**

type: Map.<SchemaAttribute, *>

The collection of value attributes this schema entry has.

SchemaEntryWithAttributes.**getAttributeValue**(*attribute*, *alwaysReturnArray=false*)

Retrieve the value of an attribute on this schema entry.

Arguments

- **attribute** (SchemaAttribute) – The attribute whose value should be returned.
- **alwaysReturnArray** (*boolean*) – Whether to return a singleton array instead of a scalar value.

Returns

* – The value of the attribute.

SchemaEntryWithAttributes.**getNamedAttributeValue**(*attributeName*, *alwaysReturnArray=false*)

Retrieve the value of an attribute (by name) on this schema entry.

Arguments

- **attributeName** (*string*) – The attribute whose value should be returned.

- **alwaysReturnArray** (*boolean*) – Whether to return a singleton array instead of a scalar value.

Returns
***** – The value of the attribute.

SchemaEntryWithAttributes.**hasAttribute**(*attribute*)

Whether this schema entry has this attribute.

Arguments

- **attribute** (SchemaAttribute) – The attribute to check for.

Returns
boolean – Whether this schema entry has this attribute.

SchemaEntryWithAttributes.**hasAttributeName**(*attributeName*)

Whether this schema entry has this attribute (by name).

Arguments

- **attributeName** (*string*) – The attribute to check for.

Returns
boolean – Whether this schema entry has this attribute.

class SchemaUnit()

SchemaUnit class

SchemaUnit.**SchemaUnit**

SchemaUnit.**_derivativeUnits**

type: Array.<string>

The legal derivatives of this unit.

class SchemaUnitClass(*name*, *booleanAttributes*, *valueAttributes*, *units*)

SchemaUnitClass class

Constructor.

Arguments

- **name** (*string*) – The name of this unit class.
- **booleanAttributes** (*Set.<SchemaAttribute>*) – The boolean attributes for this unit class.
- **valueAttributes** (*Map.<SchemaAttribute, *>*) – The value attributes for this unit class.
- **units** (*Map.<string, SchemaUnit>*) – The units for this unit class.

SchemaUnitClass.**SchemaUnitClass**

Constructor.

SchemaUnitClass.**_units**

type: Map.<string, SchemaUnit>

The units for this unit class.

SchemaUnitClass.**defaultUnit**

Get the default unit for this unit class.

SchemaUnitClass.**units**

Get the units for this unit class.

class SchemaUnitModifier()

SchemaUnitModifier class

SchemaUnitModifier.**SchemaUnitModifier**

class SchemaValueClass()

SchemaValueClass class

SchemaValueClass.**SchemaValueClass**

class SchemaTag(*name*, *booleanAttributes*, *valueAttributes*, *unitClasses*)

A tag in a HED schema.

Constructor.

Arguments

- **name** (*string*) – The name of this tag.
- **booleanAttributes** (*Set.<SchemaAttribute>*) – The boolean attributes for this tag.
- **valueAttributes** (*Map.<SchemaAttribute, *>*) – The value attributes for this tag.
- **unitClasses** (*Map.<string, SchemaUnit>*) – The unit classes for this tag.

SchemaTag.**SchemaTag**

Constructor.

SchemaTag.**_parent**

type: SchemaTag

This tag's parent tag.

SchemaTag.**_unitClasses**

type: Array.<SchemaUnitClass>

This tag's unit classes.

SchemaTag.**hasUnitClasses**

Whether this tag has any unit classes.

SchemaTag.**parent**

type: SchemaTag

This tag's parent tag.

SchemaTag.**unitClasses**

type: Array.<SchemaUnitClass>

This tag's unit classes.

5.2.3 parser

class SchemaParser()

SchemaParser class

SchemaParser.**SchemaParser**

SchemaParser.**getElementTagName**(*element*)

Extract the name of an XML element.

NOTE: This method cannot be merged into {@link getElementTagValue} because it is used as a first-class object.

Arguments

- **element** (*object*) – An XML element.

Returns

string – The name of the element.

SchemaParser.**getElementTagValue**(*element*, *tagName*)

Extract a value from an XML element.

Arguments

- **element** (*object*) – An XML element.
- **tagName** (*string*) – The tag value to extract.

Returns

string – The value of the tag in the element.

5.2.4 hed3

Intentionally blank for now

5.3 event

5.3.1 init

initiallyValidateHedString(*hedString*, *hedSchemas*, *options*, *definitions=null*)

Perform initial validation on a HED string and parse it so further validation can be performed.

Arguments

- **hedString** (*string|ParsedHedString*) – The HED string to validate.
- **hedSchemas** (Schemas) – The HED schemas to validate against.
- **options** (*Object.<string, boolean>*) – Any validation options passed in.
- **definitions** (*Map.<string, ParsedHedGroup>*) – The definitions for this HED dataset.

Returns

validateHedString(*hedString*, *hedSchemas*, *checkForWarnings*, *expectValuePlaceholderString*)

Note: Deprecated.

Validate a HED string.

Arguments

- **hedString** (*string|ParsedHedString*) – The HED string to validate.
- **hedSchemas** (Schemas) – The HED schemas to validate against.
- **checkForWarnings** (*boolean*) – Whether to check for warnings or only errors.
- **expectValuePlaceholderString** (*boolean*) – Whether this string is expected to have a '#' placeholder representing a value.

Returns

validateHedEvent(*hedString*, *hedSchemas*, *checkForWarnings*)

Note: Deprecated.

Validate a HED event string.

Arguments

- **hedString** (*string|ParsedHedString*) – The HED event string to validate.
- **hedSchemas** (Schemas) – The HED schemas to validate against.
- **checkForWarnings** (*boolean*) – Whether to check for warnings or only errors.

Returns

validateHedEventWithDefinitions(*hedString*, *hedSchemas*, *definitions*, *checkForWarnings*)

Validate a HED event string.

Arguments

- **hedString** (*string|ParsedHedString*) – The HED event string to validate.
- **hedSchemas** (Schemas) – The HED schemas to validate against.
- **definitions** (*Map.<string, ParsedHedGroup>*) – The dataset's parsed definitions.
- **checkForWarnings** (*boolean*) – Whether to check for warnings or only errors.

Returns

5.3.2 validator

class HedValidator(*parsedString*, *hedSchemas*, *options*)

HedValidator class

Constructor.

Arguments

- **parsedString** (ParsedHedString) – The parsed HED string to be validated.

- **hedSchemas** (Schemas) – The collection of HED schemas.
- **options** (*Object.<String, boolean>*) – The validation options.

HedValidator.**HedValidator**

Constructor.

HedValidator.**hedSchemas**

type: Schemas

The collection of HED schemas.

HedValidator.**issues**

type: Array.<Issue>

The running issue list.

HedValidator.**options**

type: Object.<string, boolean>

The validation options.

HedValidator.**parsedString**

type: ParsedHedString

The parsed HED string to be validated.

HedValidator.**_checkForTagAttribute**(*attribute*, *fn*)

Validation check based on a tag attribute.

Arguments

- **attribute** (*string*) – The name of the attribute.
- **fn** (*function*) – The actual validation code.

HedValidator.**checkForDuplicateTags**(*tagList*)

Check for duplicate tags at the top level or within a single group.

HedValidator.**checkForMultipleUniqueTags**(*tagList*)

Check for multiple instances of a unique tag.

HedValidator.**checkForRequiredTags**()

Check that all required tags are present.

HedValidator.**checkIfTagIsValid**(*tag*, *previousTag*)

Check if an individual HED tag is in the schema or is an allowed extension.

HedValidator.**checkIfTagRequiresChild**(*tag*)

Check if a tag is missing a required child.

Arguments

- **tag** (ParsedHedTag) – The HED tag to be checked.

HedValidator.**checkIfTagUnitClassUnitsAreValid**(*tag*)

Check that the unit is valid for the tag's unit class.

Arguments

- **tag** (ParsedHedTag) – A HED tag.

HedValidator.**checkValueTagSyntax**(*tag*)

Check the syntax of tag values.

Arguments

- **tag** (ParsedHedTag) – A HED tag.

HedValidator.**pushIssue**(*internalCode*, *parameters*)

Generate a new issue object and push it to the end of the issues array.

Arguments

- **internalCode** (*string*) – The internal error code.
- **parameters** (*Object.<string, (string|Array.<number>)>*) – The error string parameters.

HedValidator.**validateHedTagGroup**(*parsedTagGroup*)

Validate a HED tag group.

HedValidator.**validateHedTagGroups**()

Validate the HED tag groups in a parsed HED string.

HedValidator.**validateHedTagLevel**(*tagList*)

Validate a HED tag level.

HedValidator.**validateHedTagLevels**()

Validate the HED tag levels in a parsed HED string object.

HedValidator.**validateIndividualHedTag**(*tag*, *previousTag*)

Validate an individual HED tag.

HedValidator.**validateIndividualHedTags**()

Validate the individual HED tags in a parsed HED string object.

HedValidator.**validateTopLevelTags**()

Validate the top-level HED tags in a parsed HED string.

5.3.3 hed3

class Hed3Validator(*parsedString*, *hedSchemas*, *definitions*, *options*)

Hed3Validator class

Constructor.

Arguments

- **parsedString** (ParsedHedString) – The parsed HED string to be validated.
- **hedSchemas** (Schemas) – The collection of HED schemas.
- **definitions** (*Map.<string, ParsedHedGroup>*) – The parsed definitions.
- **options** (*Object.<string, boolean>*) – The validation options.

Hed3Validator.**Hed3Validator**

Constructor.

Hed3Validator.**definitions**

type: Map.<string, ParsedHedGroup>

The parsed definitions.

Hed3Validator.**checkDefinitionGroupSyntax**(*tagGroup*)

Check the syntax of HED 3 definitions.

Arguments

- **tagGroup** (ParsedHedGroup) – The tag group.

Hed3Validator.**checkDefinitionStringSyntax**()

Check full-string Definition syntax.

Hed3Validator.**checkForInvalidTopLevelTagGroupTags**()

Check for tags marked with the topLevelTagGroup attribute that are not in top-level tag groups.

Hed3Validator.**checkForInvalidTopLevelTags**()

Check for invalid top-level tags.

Hed3Validator.**checkForMissingDefinitions**(*tag*, *defShortTag="Def"*)

Check for missing HED 3 definitions.

Arguments

- **tag** (ParsedHedTag) – The HED tag.
- **defShortTag** (*string*) – The short tag to check for.

Hed3Validator.**checkIfTagIsValid**(*tag*, *previousTag*)

Check if an individual HED tag is in the schema or is an allowed extension.

Hed3Validator.**checkIfTagUnitClassUnitsAreValid**(*tag*)

Check that the unit is valid for the tag's unit class.

Arguments

- **tag** (ParsedHed3Tag) – A HED tag.

Hed3Validator.**checkPlaceholderStringSyntax**()

Check full-string placeholder syntax.

Hed3Validator.**checkPlaceholderTagSyntax**(*tag*)

Check basic placeholder tag syntax.

Arguments

- **tag** (ParsedHedTag) – A HED tag.

Hed3Validator.**checkTemporalSyntax**(*tagGroup*)

Check the syntax of HED 3 onsets and offsets.

Arguments

- **tagGroup** (ParsedHedGroup) – The tag group.

Hed3Validator.**checkValueTagSyntax**(*tag*)

Check the syntax of tag values.

Arguments

- **tag** (ParsedHed3Tag) – A HED tag.

Hed3Validator.**validateFullParsedHedString**()

Validate the full parsed HED string.

Hed3Validator.**validateHedTagGroup**(*parsedTagGroup*)

Validate a HED tag group.

Hed3Validator.**validateIndividualHedTag**(*tag*, *previousTag*)

Validate an individual HED tag.

Hed3Validator.**validateTopLevelTagGroups**()

Validate the top-level HED tag groups in a parsed HED string.

Hed3Validator.**validateTopLevelTags**()

Validate the top-level HED tags in a parsed HED string.

Hed3Validator.**validateUnits**(*tag*)

Validate a unit and strip it from the value.

Arguments

- **tag** (ParsedHed3Tag) – A HED tag.

Returns

Hed3Validator.**validateValue**(*value*, *isNumeric*)

Determine if a stripped value is valid.

Arguments

- **value** (*string*) – The stripped value.
- **isNumeric** (*boolean*) – Whether the tag is numeric.

5.4 parser

5.4.1 parsedHedTag

class ParsedHedTag(*originalTag*, *hedString*, *originalBounds*, *hedSchemas*, *schemaName*)

A parsed HED tag.

Constructor.

Arguments

- **originalTag** (*string*) – The original HED tag.
- **hedString** (*string*) – The original HED string.
- **originalBounds** (*Array.<number>*) – The bounds of the HED tag in the original HED string.
- **hedSchemas** (Schemas) – The collection of HED schemas.
- **schemaName** (*string*) – The label of this tag's schema in the dataset's schema spec.

ParsedHedTag.**ParsedHedTag**

Constructor.

ParsedHedTag.**allowsExtensions**

Check if any level of this HED tag allows extensions.

ParsedHedTag.**canonicalTag**

type: string

The canonical form of the HED tag.

ParsedHedTag.**conversionIssues**

type: Array.<Issue>

Any issues encountered during tag conversion.

ParsedHedTag.**formattedTag**

type: string

The formatted canonical version of the HED tag.

ParsedHedTag.**schema**

type: Schema

The HED schema this tag belongs to.

ParsedHedTag.**_convertTag**(*hedString*, *hedSchemas*, *schemaName*)

Convert this tag to long form.

Arguments

- **hedString** (*string*) – The original HED string.
- **hedSchemas** (Schemas) – The collection of HED schemas.
- **schemaName** (*string*) – The label of this tag's schema in the dataset's schema spec.

ParsedHedTag.**_formatTag**()

Format this HED tag by removing newlines, double quotes, and slashes.

ParsedHedTag.**toString**()

Override of {@link Object.prototype.toString}.

Returns

string – The original form of this HED tag.

static ParsedHedTag.**ancestorIterator**(*tagString*)

Iterate through a tag's ancestor tag strings.

Arguments

- **tagString** (*string*) – A tag string.

static ParsedHedTag.**getParentTag**(*tagString*)

Get the HED tag prefix (up to the last slash).

static ParsedHedTag.**getTagName**(*tagString*)

Get the last part of a HED tag.

Arguments

- **tagString** (*string*) – A HED tag.

Returns

string – The last part of the tag using the given separator.

class ParsedHed3Tag()

A parsed HED3 tag.

ParsedHed3Tag.**defaultUnit**

Get the default unit for this HED tag.

ParsedHed3Tag.**existsInSchema**

Determine if this HED tag is in the schema.

ParsedHed3Tag.**hasUnitClass**

Checks if this HED tag has the 'unitClass' attribute.

ParsedHed3Tag.**takesValue**

Checks if this HED tag has the 'takesValue' attribute.

ParsedHed3Tag.**takesValueFormattedTag**

Determine value-taking form of this tag.

ParsedHed3Tag.**takesValueTag**

Get the schema tag object for this tag's value-taking form.

ParsedHed3Tag.**unitClasses**

Get the unit classes for this HED tag.

ParsedHed3Tag.**validUnits**

Get the legal units for a particular HED tag.

ParsedHed3Tag.**_convertTag**(*hedString*, *hedSchemas*, *schemaName*)

Convert this tag to long form.

Arguments

- **hedString** (*string*) – The original HED string.
- **hedSchemas** (Schemas) – The collection of HED schemas.
- **schemaName** (*string*) – The label of this tag's schema in the dataset's schema spec.

5.4.2 splitHedString

class HedStringTokenizer(*hedString*)

Class for tokenizing hed strings

HedStringTokenizer.**tokenize**()

Split the HED string into delimiters and tags.

Returns

checkForInvalidCharacters(*hedString*, *tagSpecs*)

Check the split HED tags for invalid characters

Arguments

- **hedString** (*string*) – The HED string to be split.
- **tagSpecs** (*Array.<TagSpec>*) – The tag specifications.

Returns

Object.<string, Array.<Issue>> – Any issues found.

createParsedTags(*hedString*, *hedSchemas*, *tagSpecs*, *groupSpecs*)

Create the parsed HED tag and group objects.

Arguments

- **hedString** (*string*) – The HED string to be split.
- **hedSchemas** (Schemas) – The collection of HED schemas.
- **tagSpecs** (*Array.<TagSpec>*) – The tag specifications.
- **groupSpecs** (*GroupSpec*) – The bounds of the tag groups.

Returns

splitHedString(*hedString*)

Split a HED string into delimiters and tags.

Arguments

- **hedString** (*string*) – The HED string to split.

Returns

Array.<Array> – A list of string parts. The boolean is true if the part is a tag and false if it is a delimiter. The numbers are the bounds of the part.

5.4.3 parsedHedString

class ParsedHedString(*hedString*, *parsedTags*)

A parsed HED string.

Constructor.

Arguments

- **hedString** (*string*) – The original HED string.
- **parsedTags** (*Array.<ParsedHedSubstring>*) – The nested list of parsed HED tags and groups.

ParsedHedString.**ParsedHedString**

Constructor.

ParsedHedString.**definitionGroups**

type: Array.<ParsedHedGroup>

The definition tag groups in the string.

ParsedHedString.**hedString**

type: string

The original HED string.

ParsedHedString.**tagGroups**

type: Array.<ParsedHedGroup>

The tag groups in the string.

ParsedHedString.**tags**

type: Array.<ParsedHedTag>

All the tags in the string.

ParsedHedString.**topLevelTagGroups**

type: Array.<Array.<ParsedHedTag>>

The top-level tag groups in the string, split into arrays.

ParsedHedString.**topLevelTags**

type: Array.<ParsedHedTag>

All the top-level tags in the string.

5.4.4 parsedHedSubstring

class ParsedHedSubstring(*originalTag*, *originalBounds*)

A parsed HED substring.

Constructor.

Arguments

- **originalTag** (*string*) – The original HED tag.
- **originalBounds** (*Array.<number>*) – The bounds of the HED tag in the original HED string.

ParsedHedSubstring.**ParsedHedSubstring**

Constructor.

ParsedHedSubstring.**originalBounds**

type: Array.<int>

The bounds of the HED tag in the original HED string.

ParsedHedSubstring.**originalTag**

type: string

The original pre-parsed version of the HED tag.

ParsedHedSubstring.**toString**()

Override of {@link Object.prototype.toString}.

Returns

string – The original form of this HED substring.

5.4.5 parsedHedGroup

class ParsedHedGroup(*parsedHedTags*, *hedSchemas*, *hedString*, *originalBounds*)

A parsed HED tag group.

Constructor.

Arguments

- **hedSchemas** (Schemas) – The collection of HED schemas.
- **hedString** (*string*) – The original HED string.
- **originalBounds** (*Array.<number>*) – The bounds of the HED tag in the original HED string.

ParsedHedGroup.**ParsedHedGroup**

Constructor.

ParsedHedGroup.**defCount**

Determine the number of {@code Def} and {@code Def-expand} tag/tag groups included in this group.

ParsedHedGroup.**defExpandChildren**

type: Array.<ParsedHedGroup>

The top-level child subgroups containing Def-expand tags.

ParsedHedGroup.**defExpandName**

Determine the name of this group's definition.

ParsedHedGroup.**defExpandNameAndValue**

Determine the name and value of this group's definition.

ParsedHedGroup.**defExpandTag**

Find what should be the sole {@code Def-expand} tag, or throw an error if more than one is found.

ParsedHedGroup.**defExpandTags**

The {@code Def-expand} tags associated with this HED tag group.

ParsedHedGroup.**defExpandValue**

Determine the value of this group's definition.

ParsedHedGroup.**defName**

Determine the name(s) of this group's definition.

ParsedHedGroup.**defNameAndValue**

Determine the name and value of this group's {@code Def} or {@code Def-expand}.

ParsedHedGroup.**defTags**

The {@code Def} tags associated with this HED tag group.

ParsedHedGroup.**defValue**

Determine the name of this group's definition.

ParsedHedGroup.**definitionGroup**

Determine the value of this group's definition.

ParsedHedGroup.**definitionName**

Determine the name of this group's definition.

ParsedHedGroup.**definitionNameAndValue**

Determine the name and value of this group's definition.

ParsedHedGroup.**definitionTag**

Find what should be the sole definition tag, or throw an error if more than one is found.

ParsedHedGroup.**definitionTags**

The {@code Definition} tags associated with this HED tag group.

ParsedHedGroup.**definitionValue**

Determine the value of this group's definition.

ParsedHedGroup.**hasDefExpandChildren**

type: boolean

Whether this HED tag group has child groups with a Def-expand tag.

ParsedHedGroup.**isDefExpandGroup**

Whether this HED tag group has a {@code Def-expand} tag.

ParsedHedGroup.**isDefGroup**

Whether this HED tag group has a {@code Def} tag.

ParsedHedGroup.**isDefinitionGroup**

Whether this HED tag group is a definition group.

ParsedHedGroup.**isInsetGroup**

Whether this HED tag group is an inset group.

ParsedHedGroup.**isOffsetGroup**

Whether this HED tag group is an offset group.

ParsedHedGroup.**isOnsetGroup**

Whether this HED tag group is an onset group.

ParsedHedGroup.**isTemporalGroup**

Whether this HED tag group is an onset, offset, or inset group.

ParsedHedGroup.**specialTags**

type: Map.<string, Array.<ParsedHedTag>>

Any HED tags with special handling.

ParsedHedGroup.**tags**

The parsed HED tags in the HED tag group.

ParsedHedGroup.**temporalGroupName**

Whether this HED tag group is an onset, offset, or inset group.

ParsedHedGroup.**nestedGroups**()

The deeply nested array of parsed tags.

Returns
Array.<ParsedHedTag> –

ParsedHedGroup.**subGroupArrayIterator**()

Iterator over the full HED groups and subgroups in this HED tag group.

ParsedHedGroup.**subParsedGroupIterator**()

Iterator over the ParsedHedGroup objects in this HED tag group.

ParsedHedGroup.**tagIterator**()

Iterator over the parsed HED tags in this HED tag group.

ParsedHedGroup.**topLevelGroupIterator**()

Iterator over the top-level parsed HED groups in this HED tag group.

static ParsedHedGroup.**findDefinitionName**(*canonicalTag*, *definitionBase*)

Determine the name of this group's definition.

static ParsedHedGroup.**findGroupTags**(*group*, *hedSchemas*, *shortTag*)

Determine a parsed HED tag group's special tags.

Arguments

- **group** (ParsedHedGroup) – The parsed HED tag group.

- **hedSchemas** (Schemas) – The collection of HED schemas.
- **shortTag** (*string*) – The short tag to search for.

Returns
null|Array.<ParsedHedTag> – The tag(s) matching the short tag.

static ParsedHedGroup.**getDefinitionTagValue**(*tag*, *parentTag*)

Extract the value from a definition tag.

Arguments

- **tag** (ParsedHedTag) – A definition-type tag.
- **parentTag** (*string*) – The expected parent of the tag.

Returns
string – The parameterized value of the definition, or an empty string if no value was found.

5.4.6 main

substituteCharacters()

Substitute certain illegal characters and report warnings when found.

countTagGroupParentheses()

Check if group parentheses match. Pushes an issue if they don't match.

isCommaMissingAfterClosingParenthesis()

Check if a comma is missing after an opening parenthesis.

findDelimiterIssuesInHedString()

Check for delimiter issues in a HED string (e.g. missing commas adjacent to groups, extra commas or tildes).

validateFullUnparsedHedString(*hedString*)

Validate the full unparsed HED string.

Arguments

- **hedString** (*string*) – The unparsed HED string.

Returns
Object.<string, Array.<Issue>> – String substitution issues and other issues.

parseHedString(*hedString*, *hedSchemas*)

Parse a full HED string into an object of tag types.

Arguments

- **hedString** (*string*) – The full HED string to parse.
- **hedSchemas** (Schemas) – The collection of HED schemas.

Returns

parseHedStrings(*hedStrings*, *hedSchemas*)

Parse a set of HED strings.

Arguments

- **hedStrings** (*Array.<string>*) – A set of HED strings.
- **hedSchemas** (Schemas) – The collection of HED schemas.

Returns

5.5 hed2

5.5.1 schema

5.5.1.1 schemaAttributes

class SchemaAttributes(*schemaParser*)

A description of a HED schema's attributes.

Constructor.

Arguments

- **schemaParser** (Hed2SchemaParser) – A constructed schema parser.

SchemaAttributes.**SchemaAttributes**

Constructor.

SchemaAttributes.**hasUnitClasses**

type: boolean

Whether the schema has unit classes.

SchemaAttributes.**hasUnitModifiers**

type: boolean

Whether the schema has unit modifiers.

SchemaAttributes.**tagAttributes**

type: Object.<string, Object.<string, (boolean|string|Array.<string>)>>

The mapping from attributes to tags to values.

SchemaAttributes.**tagUnitClasses**

type: Object.<string, Array.<string>>

The mapping from tags to their unit classes.

SchemaAttributes.**tags**

type: Array.<string>

The list of all (formatted) tags.

SchemaAttributes.**unitAttributes**

type: Object.<string, Object.<string, (boolean|string|Array.<string>)>>

The mapping from units to their attributes.

SchemaAttributes.**unitClassAttributes**

type: Object.<string, Object.<string, (boolean|string|Array.<string>)>>

The mapping from unit classes to their attributes.

SchemaAttributes.**unitClasses**

type: Object.<string, Array.<string>>

The mapping from unit classes to their units.

SchemaAttributes.**unitModifiers**

type: Object.<string, Array.<string>>

The mapping from unit modifier types to unit modifiers.

SchemaAttributes.**tagHasAttribute**(*tag*, *tagAttribute*)

Determine if a HED tag has a particular attribute in this schema.

Arguments

- **tag** (*string*) – The HED tag to check.
- **tagAttribute** (*string*) – The attribute to check for.

Returns

boolean|null – Whether this tag has this attribute, or null if the attribute doesn't exist.

5.5.1.2 hed2SchemaParser

class Hed2SchemaParser()

Hed2SchemaParser class

5.5.2 event

5.5.2.1 hed2Validator

class Hed2Validator()

Hed2Validator class

Hed2Validator.**Hed2Validator**

Hed2Validator.**checkIfTagUnitClassUnitsAreValid**(*tag*)

Check that the unit is valid for the tag's unit class.

Arguments

- **tag** (ParsedHed2Tag) – A HED tag.

Hed2Validator.**checkValueTagSyntax**(*tag*)

Check the syntax of tag values.

Arguments

- **tag** (ParsedHed2Tag) – A HED tag.

Hed2Validator.**validateValue**(*value*, *isNumeric*)

Determine if a stripped value is valid.

Arguments

- **value** (*string*) – The stripped value.
- **isNumeric** (*boolean*) – Whether the tag is numeric.

Returns

boolean – Whether the stripped value is valid.

5.5.2.2 units

units.**validateUnits**(*originalTagUnitValue*, *tagUnitClassUnits*, *hedSchemaAttributes*)

Validate a unit and strip it from the value.

Arguments

- **originalTagUnitValue** (*string*) – The unformatted version of the value.
- **tagUnitClassUnits** (*Array.<string>*) – The list of valid units for this tag.
- **hedSchemaAttributes** (SchemaAttributes) – The collection of schema attributes.

Returns

isPrefixUnit(*unit*, *hedSchemaAttributes*)

Determine whether a unit is a valid prefix unit.

Arguments

- **unit** (*string*) – A unit string.
- **hedSchemaAttributes** (SchemaAttributes) – The collection of schema attributes.

Returns

boolean – Whether the unit is a valid prefix unit.

getValidDerivativeUnits(*unit*, *hedSchemaAttributes*)

Get the list of valid derivatives of a unit.

Arguments

- **unit** (*string*) – A unit string.
- **hedSchemaAttributes** (SchemaAttributes) – The collection of schema attributes.

Returns

Array.<string> – The list of valid derivative units.

getAllUnits()

Get the legal units for a particular HED tag.

5.5.3 parser

5.5.3.1 parsedHed2Tag

class ParsedHed2Tag()

ParsedHedTag class

ParsedHed2Tag.**defaultUnit**

Get the default unit for this HED tag.

ParsedHed2Tag.**existsInSchema**

Determine if this HED tag is in the schema.

ParsedHed2Tag.**hasUnitClass**

Checks if this HED tag has the 'unitClass' attribute.

ParsedHed2Tag.**takesValue**

Checks if this HED tag has the 'takesValue' attribute.

ParsedHed2Tag.**takesValueFormattedTag**

Determine value-taking form of this tag.

ParsedHed2Tag.**unitClasses**

Get the unit classes for this HED tag.

ParsedHed2Tag.**validUnits**

Get the legal units for a particular HED tag.

ParsedHed2Tag.**_convertTag**(*hedString*, *hedSchemas*, *schemaName*)

Convert this tag to long form.

Arguments

- **hedString** (*string*) – The original HED string.
- **hedSchemas** (Schemas) – The collection of HED schemas.
- **schemaName** (*string*) – The label of this tag's schema in the dataset's schema spec.

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

INDEX

I

L

M

P

R

S

T

U

V