HED Documentation

Release 8.0.0

HED Working Group

Dec 01, 2021

CONTENTS:

A PDF version of this document can be found here

CHAPTER

ONE

1. OVERVIEW

This document contains the specification for third generation HED or HED-3G. It is meant for the implementers and users of HED tools. Other tutorials and tagging guides are available to researchers using HED to annotate their data. This document contains the specification for the first official release of HED-3G (HED versions 8.0.0-xxx and above.) **When the term HED is used in this document, it refers to third generation (HED-3G) unless explicitly stated otherwise.**

The aspects of HED that are described in this document are supported or will soon be supported by validators and other tools and are available for immediate use by annotators. The schema vocabulary can be viewed using an expandable schema viewer.

All HED-related source and documentation repositories are housed on the HED-standard organization GitHub site, https://github.com/hed-standard, which is maintained by the HED Working Group. HED development is open-source and community-based. Also see the official HED website https://www.hedtags.org for a list of additional resources.

The HED Working Group invites those interested in HED to contribute to the development process. Users are encouraged to use the *Issues* mechanism of the `hed-specification` repository on the GitHub `hed-standard` working group website: https://github.com/hed-standard/hed-specification/issues to ask for help or make suggestions. The HED discussion forum https://github.com/hed-standard/hed-specification/discussions is maintained for in depth discussions of HED issues and evolution.

Several other aspects of HED annotation are being planned, but their specification has not been fully determined. These aspects are not contained in this specification document, but rather are contained in ancillary working documents which are open for discussion. These ancillary specifications include the HED working document on spatial annotation and the HED working document on task annotation.

1.1 1.1. Scope of HED

HED (an acronym for Hierarchical Event Descriptors) is an evolving framework that facilitates the description and formal annotation of events identified in time series data, together with tools for validation and for using HED annotations in data search, extraction, and analysis. This specification describes the official release of third generation of HED or HED-3G, which is HED version 8.0.0.

Third generation HED represents a significant advance in documenting the content and intent of experiments in a format that enables large-scale cross-study analysis of time-series behavioral and neuroimaging data, including but not limited to EEG, MEG, iEEG, eye-tracking, motion-capture, EKG, and audiovisual recording. In principle, third generation HED might be extended or adapted to annotate events in any other type of ordered or time series data.

Specifically, the goal of HED is to allow researchers to annotate what happened during an experiment, including experimental stimuli and other sensory events, participant responses and actions, experimental design, the role of events in the task, and the temporal structure of the experiment. The resulting annotation is machine-actionable, meaning that it can be used as input to algorithms without manual intervention. HED facilitates detailed comparisons of data across studies.

HED annotations may be included in BIDS (Brain Imaging Data Structure) datasets https://bids.neuroimaging.io as described in Chapter 7: HED in BIDS.

1.2 1.2. Brief history of HED

HED was originally proposed by Nima Bigdely-Shamlo in 2010 to support annotation in HeadIT and early public repository for EEG data hosted by the Swartz Center for Computational Neuroscience, UCSD (Bigdely-Shamlo et al. 2013). HED-1G was partially based on CogPO (Turner and Laird 2012).

Event annotation in HED-1G was organized around a single hierarchy whose root was the *Time-Locked Event*. Users could extend the HED-1G hierarchy at its deepest (leaf) nodes. First generation HED (HED-1G, versions < 5.0.0) attempted to describe events using a strictly hierarchical vocabulary.

HED-1G was oriented toward annotating stimuli and responses, but its lack of orthogonality in vocabulary design presented major difficulties. If *Red/Triangle* and *Green/Triangle* are terms in a hierarchy, one is also likely to need *Red/Square* and Green/Square* as well as other color and shape combinations.

HED-2G (versions 5.0.0 - 7.x.x) introduced a more orthogonal vocabulary, meaning that independent terms were in different subtrees of the vocabulary tree. Separating independent concepts such as shapes and colors into separate hierarchies, eliminates an exponential vocabulary growth due to term duplication in different branches of the hierarchy.

Parentheses were introduced so that terms could be grouped. Tools for validation and epoching based on HED tags were built, and large-scale cross-study "mega-analyses" were performed. However, as more complicated and varied datasets were annotated using HED-2G, the vocabulary started to become less manageable as HED tried to adapt to more complex annotation demands.

In 2019, work began on a rethinking of the HED vocabulary design, resulting in the release of the third generation of HED (HED-3G) in August 2021. HED-3G represents a dramatic increase in annotation capacity, but also a significant simplification of the user experience.

New in HED (versions 8.0.0+).

1. Improved vocabulary structure
2. Short-form annotation
3. Library schema
4. Definitions
5. Temporal scope
6. Encoding of experimental design

Following basic design principles, the HED Working Group redesigned the HED vocabulary tree to be organized in a balanced hierarchy with a limited number of subcategories at each node. (See the expandable schema browser to browser the vocabulary and explore the overall organization. Chapter2:Terminology defines some important HED tags and terminology used in HED.)

A major improvement in vocabulary design was the adoption of the requirement that individual nodes or terms in the HED vocabulary must be unique. This allows users to use individual node names (short form) rather than the full paths to the schema root during annotation, resulting in substantially simpler, more readable annotations.

To enable and regulate the extension process, the root HED-3G head schema specified here includes, for the first time, *HED library schema* to extend the HED vocabulary to include terms and concepts of importance to individual user communities – for example researchers who design and perform experiments to study brain and language, brain and music, or brain dynamics in natural or virtual reality environments. The HED library schema concept may also be used to extend HED annotation to encompass specialized vocabularies used in clinical research and practice.

HED-3G also introduced a number of advanced tagging concepts that allow users to represent events with temporal duration, as well as annotations that represent experimental design.

1.3 1.2. Goals of HED

Event annotation documents the things happening during data recording regardless of relevance to data analysis and interpretation. Commonly recorded events in electrophysiological data collection include the initiation, termination, or other features of **sensory presentations** and **participant actions**. Other events may be **unplanned environmental events** (for example, sudden onset of noise and vibration from construction work unrelated to the experiment, or a laboratory device malfunction), events recording **changes in experiment control** parameters as well as **data feature events** and control **mishap events** that cause operation to fall outside of normal experiment parameters. The goals of HED are to provide a standardized annotation and supporting infrastructure.

Goals of HED.

1. **Document the exact nature of events** (sensory, behavioral, environmental, and other) that occur during recorded time series data in order to inform data analysis and interpretation.
2. **Describe the design of the experiment** including participant task(s).
3. **Relate event occurrences** both to the experiment design and to participant tasks and experience.
4. **Provide basic infrastructure** for building and using machine-actionable tools to systematically analyze data associated with recorded events in and across data sets, studies, paradigms, and modalities.

Current systems in neuroimaging experiments do not record events beyond simple numerical (3) or text (Event type Target) labels whose more complete and precise meanings are known only to the experimenter(s).

A central goal of HED is to enable building of archives of brain imaging data in a form amenable to new forms of larger scale analysis, both within and across studies. Such event-related analysis requires that the nature(s) of the recorded events be specified in a common language. The HED project seeks to formalize the development of this language, to develop and distribute tools that maximize its ease of use, and to inform new and existing researchers of its purpose and value.

1.4 1.3. HED design principles

The near decade-long effort to develop effective event annotation for neurophysiological and behavioral data, culminating to date in HED-3G, has revealed the importance of four principles (aka the PASS principles), all of which have roots in other fields:

The PASS principles for HED design.

1. **Preserve orthogonality** of concepts in specifying vocabularies.
2. **Abstract functionality** into layers (e.g., more general vs. more specific).
3. **Separate content** from presentation.
4. **Separate implementation** from the interface (for flexibility).

Orthogonality, the notion of keeping independently applicable concepts in separate hierarchies (1 above), has long been recognized as a fundamental principle in reusable software design, distilled in the design rule: *Favor composition over inheritance* (Gamma et al. 1994).

Abstraction of functionality into layers (2) and separation of content from presentation (3) are well-known principles in user-interface and graphics design that allow tools to maintain a single internal representation of needed information while emphasizing different aspects of the information when presenting it to users.

Similarly, making validation and analysis code independent of the HEDschema (4) allows redesign of the schema without having to re-implement the annotation tools. A well-specified and stable API (application program interface) empowers tool developers.

CHAPTER

TWO

2. DIRECT ANNOTATION

This document contains the specification for third generation HED or HED-3G. It is meant for the implementers and users of HED tools. Other tutorials and tagging guides are available to researchers using HED to annotate their data. This document contains the specification for the first official release of HED-3G (HED versions 8.0.0-xxx and above.) **When the term HED is used in this document, it refers to third generation (HED-3G) unless explicitly stated otherwise.**

The aspects of HED that are described in this document are supported or will soon be supported by validators and other tools and are available for immediate use by annotators. The schema vocabulary can be viewed using an expandable schema viewer.

All HED-related source and documentation repositories are housed on the HED-standard organization GitHub site, https://github.com/hed-standard, which is maintained by the HED Working Group. HED development is open-source and community-based. Also see the official HED website https://www.hedtags.org for a list of additional resources.

The HED Working Group invites those interested in HED to contribute to the development process. Users are encouraged to use the *Issues* mechanism of the `hed-specification` repository on the GitHub `hed-standard` working group website: https://github.com/hed-standard/hed-specification/issues to ask for help or make suggestions. The HED discussion forum https://github.com/hed-standard/hed-specification/discussions is maintained for in depth discussions of HED issues and evolution.

Several other aspects of HED annotation are being planned, but their specification has not been fully determined. These aspects are not contained in this specification document, but rather are contained in ancillary working documents which are open for discussion. These ancillary specifications include the HED working document on spatial annotation and the HED working document on task annotation.

2.1 1.1. Scope of HED

HED (an acronym for Hierarchical Event Descriptors) is an evolving framework that facilitates the description and formal annotation of events identified in time series data, together with tools for validation and for using HED annotations in data search, extraction, and analysis. This specification describes the official release of third generation of HED or HED-3G, which is HED version 8.0.0.

Third generation HED represents a significant advance in documenting the content and intent of experiments in a format that enables large-scale cross-study analysis of time-series behavioral and neuroimaging data, including but not limited to EEG, MEG, iEEG, eye-tracking, motion-capture, EKG, and audiovisual recording. In principle, third generation HED might be extended or adapted to annotate events in any other type of ordered or time series data.

Specifically, the goal of HED is to allow researchers to annotate what happened during an experiment, including experimental stimuli and other sensory events, participant responses and actions, experimental design, the role of events in the task, and the temporal structure of the experiment. The resulting annotation is machine-actionable, meaning that it can be used as input to algorithms without manual intervention. HED facilitates detailed comparisons of data across studies.

HED annotations may be included in BIDS (Brain Imaging Data Structure) datasets https://bids.neuroimaging.io as described in Chapter 7: HED in BIDS.

2.2 1.2. Brief history of HED

HED was originally proposed by Nima Bigdely-Shamlo in 2010 to support annotation in HeadIT and early public repository for EEG data hosted by the Swartz Center for Computational Neuroscience, UCSD (Bigdely-Shamlo et al. 2013). HED-1G was partially based on CogPO (Turner and Laird 2012).

Event annotation in HED-1G was organized around a single hierarchy whose root was the *Time-Locked Event*. Users could extend the HED-1G hierarchy at its deepest (leaf) nodes. First generation HED (HED-1G, versions < 5.0.0) attempted to describe events using a strictly hierarchical vocabulary.

HED-1G was oriented toward annotating stimuli and responses, but its lack of orthogonality in vocabulary design presented major difficulties. If *Red/Triangle* and *Green/Triangle* are terms in a hierarchy, one is also likely to need *Red/Square* and Green/Square* as well as other color and shape combinations.

HED-2G (versions 5.0.0 - 7.x.x) introduced a more orthogonal vocabulary, meaning that independent terms were in different subtrees of the vocabulary tree. Separating independent concepts such as shapes and colors into separate hierarchies, eliminates an exponential vocabulary growth due to term duplication in different branches of the hierarchy.

Parentheses were introduced so that terms could be grouped. Tools for validation and epoching based on HED tags were built, and large-scale cross-study "mega-analyses" were performed. However, as more complicated and varied datasets were annotated using HED-2G, the vocabulary started to become less manageable as HED tried to adapt to more complex annotation demands.

In 2019, work began on a rethinking of the HED vocabulary design, resulting in the release of the third generation of HED (HED-3G) in August 2021. HED-3G represents a dramatic increase in annotation capacity, but also a significant simplification of the user experience.

New in HED (versions 8.0.0+).

1. Improved vocabulary structure
2. Short-form annotation
3. Library schema
4. Definitions
5. Temporal scope
6. Encoding of experimental design

Following basic design principles, the HED Working Group redesigned the HED vocabulary tree to be organized in a balanced hierarchy with a limited number of subcategories at each node. (See the expandable schema browser to browser the vocabulary and explore the overall organization. Chapter2:Terminology defines some important HED tags and terminology used in HED.)

A major improvement in vocabulary design was the adoption of the requirement that individual nodes or terms in the HED vocabulary must be unique. This allows users to use individual node names (short form) rather than the full paths to the schema root during annotation, resulting in substantially simpler, more readable annotations.

To enable and regulate the extension process, the root HED-3G head schema specified here includes, for the first time, *HED library schema* to extend the HED vocabulary to include terms and concepts of importance to individual user communities – for example researchers who design and perform experiments to study brain and language, brain and music, or brain dynamics in natural or virtual reality environments. The HED library schema concept may also be used to extend HED annotation to encompass specialized vocabularies used in clinical research and practice.

HED-3G also introduced a number of advanced tagging concepts that allow users to represent events with temporal duration, as well as annotations that represent experimental design.

2.3 1.2. Goals of HED

Event annotation documents the things happening during data recording regardless of relevance to data analysis and interpretation. Commonly recorded events in electrophysiological data collection include the initiation, termination, or other features of **sensory presentations** and **participant actions**. Other events may be **unplanned environmental events** (for example, sudden onset of noise and vibration from construction work unrelated to the experiment, or a laboratory device malfunction), events recording **changes in experiment control** parameters as well as **data feature events** and control **mishap events** that cause operation to fall outside of normal experiment parameters. The goals of HED are to provide a standardized annotation and supporting infrastructure.

Goals of HED.

1. **Document the exact nature of events** (sensory, behavioral, environmental, and other) that occur during recorded time series data in order to inform data analysis and interpretation.
2. **Describe the design of the experiment** including participant task(s).
3. **Relate event occurrences** both to the experiment design and to participant tasks and experience.
4. **Provide basic infrastructure** for building and using machine-actionable tools to systematically analyze data associated with recorded events in and across data sets, studies, paradigms, and modalities.

Current systems in neuroimaging experiments do not record events beyond simple numerical (3) or text (Event type Target) labels whose more complete and precise meanings are known only to the experimenter(s).

A central goal of HED is to enable building of archives of brain imaging data in a form amenable to new forms of larger scale analysis, both within and across studies. Such event-related analysis requires that the nature(s) of the recorded events be specified in a common language. The HED project seeks to formalize the development of this language, to develop and distribute tools that maximize its ease of use, and to inform new and existing researchers of its purpose and value.

2.4 1.3. HED design principles

The near decade-long effort to develop effective event annotation for neurophysiological and behavioral data, culminating to date in HED-3G, has revealed the importance of four principles (aka the PASS principles), all of which have roots in other fields:

The PASS principles for HED design.

1. **Preserve orthogonality** of concepts in specifying vocabularies.
2. **Abstract functionality** into layers (e.g., more general vs. more specific).
3. **Separate content** from presentation.
4. **Separate implementation** from the interface (for flexibility).

Orthogonality, the notion of keeping independently applicable concepts in separate hierarchies (1 above), has long been recognized as a fundamental principle in reusable software design, distilled in the design rule: *Favor composition over inheritance* (Gamma et al. 1994).

Abstraction of functionality into layers (2) and separation of content from presentation (3) are well-known principles in user-interface and graphics design that allow tools to maintain a single internal representation of needed information while emphasizing different aspects of the information when presenting it to users.

Similarly, making validation and analysis code independent of the HEDschema (4) allows redesign of the schema without having to re-implement the annotation tools. A well-specified and stable API (application program interface) empowers tool developers.

CHAPTER

THREE

1. INTRODUCTION TO HED

This document contains the specification for third generation HED or HED-3G. It is meant for the implementers and users of HED tools. Other tutorials and tagging guides are available to researchers using HED to annotate their data. This document contains the specification for the first official release of HED-3G (HED versions 8.0.0-xxx and above.) **When the term HED is used in this document, it refers to third generation (HED-3G) unless explicitly stated otherwise.**

The aspects of HED that are described in this document are supported or will soon be supported by validators and other tools and are available for immediate use by annotators. The schema vocabulary can be viewed using an expandable schema viewer.

All HED-related source and documentation repositories are housed on the HED-standard organization GitHub site, https://github.com/hed-standard, which is maintained by the HED Working Group. HED development is open-source and community-based. Also see the official HED website https://www.hedtags.org for a list of additional resources.

The HED Working Group invites those interested in HED to contribute to the development process. Users are encouraged to use the *Issues* mechanism of the `hed-specification` repository on the GitHub `hed-standard` working group website: https://github.com/hed-standard/hed-specification/issues to ask for help or make suggestions. The HED discussion forum https://github.com/hed-standard/hed-specification/discussions is maintained for in depth discussions of HED issues and evolution.

Several other aspects of HED annotation are being planned, but their specification has not been fully determined. These aspects are not contained in this specification document, but rather are contained in ancillary working documents which are open for discussion. These ancillary specifications include the HED working document on spatial annotation and the HED working document on task annotation.

3.1 1.1. Scope of HED

HED (an acronym for Hierarchical Event Descriptors) is an evolving framework that facilitates the description and formal annotation of events identified in time series data, together with tools for validation and for using HED annotations in data search, extraction, and analysis. This specification describes the official release of third generation of HED or HED-3G, which is HED version 8.0.0.

Third generation HED represents a significant advance in documenting the content and intent of experiments in a format that enables large-scale cross-study analysis of time-series behavioral and neuroimaging data, including but not limited to EEG, MEG, iEEG, eye-tracking, motion-capture, EKG, and audiovisual recording. In principle, third generation HED might be extended or adapted to annotate events in any other type of ordered or time series data.

Specifically, the goal of HED is to allow researchers to annotate what happened during an experiment, including experimental stimuli and other sensory events, participant responses and actions, experimental design, the role of events in the task, and the temporal structure of the experiment. The resulting annotation is machine-actionable, meaning that it can be used as input to algorithms without manual intervention. HED facilitates detailed comparisons of data across studies.

HED annotations may be included in BIDS (Brain Imaging Data Structure) datasets https://bids.neuroimaging.io as described in Chapter 7: HED in BIDS.

3.2 1.2. Brief history of HED

HED was originally proposed by Nima Bigdely-Shamlo in 2010 to support annotation in HeadIT and early public repository for EEG data hosted by the Swartz Center for Computational Neuroscience, UCSD (Bigdely-Shamlo et al. 2013). HED-1G was partially based on CogPO (Turner and Laird 2012).

Event annotation in HED-1G was organized around a single hierarchy whose root was the *Time-Locked Event*. Users could extend the HED-1G hierarchy at its deepest (leaf) nodes. First generation HED (HED-1G, versions < 5.0.0) attempted to describe events using a strictly hierarchical vocabulary.

HED-1G was oriented toward annotating stimuli and responses, but its lack of orthogonality in vocabulary design presented major difficulties. If *Red/Triangle* and *Green/Triangle* are terms in a hierarchy, one is also likely to need *Red/Square* and Green/Square* as well as other color and shape combinations.

HED-2G (versions 5.0.0 - 7.x.x) introduced a more orthogonal vocabulary, meaning that independent terms were in different subtrees of the vocabulary tree. Separating independent concepts such as shapes and colors into separate hierarchies, eliminates an exponential vocabulary growth due to term duplication in different branches of the hierarchy.

Parentheses were introduced so that terms could be grouped. Tools for validation and epoching based on HED tags were built, and large-scale cross-study "mega-analyses" were performed. However, as more complicated and varied datasets were annotated using HED-2G, the vocabulary started to become less manageable as HED tried to adapt to more complex annotation demands.

In 2019, work began on a rethinking of the HED vocabulary design, resulting in the release of the third generation of HED (HED-3G) in August 2021. HED-3G represents a dramatic increase in annotation capacity, but also a significant simplification of the user experience.

New in HED (versions 8.0.0+).

1. Improved vocabulary structure
2. Short-form annotation
3. Library schema
4. Definitions
5. Temporal scope
6. Encoding of experimental design

Following basic design principles, the HED Working Group redesigned the HED vocabulary tree to be organized in a balanced hierarchy with a limited number of subcategories at each node. (See the expandable schema browser to browser the vocabulary and explore the overall organization. Chapter2:Terminology defines some important HED tags and terminology used in HED.)

A major improvement in vocabulary design was the adoption of the requirement that individual nodes or terms in the HED vocabulary must be unique. This allows users to use individual node names (short form) rather than the full paths to the schema root during annotation, resulting in substantially simpler, more readable annotations.

To enable and regulate the extension process, the root HED-3G head schema specified here includes, for the first time, *HED library schema* to extend the HED vocabulary to include terms and concepts of importance to individual user communities – for example researchers who design and perform experiments to study brain and language, brain and music, or brain dynamics in natural or virtual reality environments. The HED library schema concept may also be used to extend HED annotation to encompass specialized vocabularies used in clinical research and practice.

HED-3G also introduced a number of advanced tagging concepts that allow users to represent events with temporal duration, as well as annotations that represent experimental design.

3.3 1.2. Goals of HED

Event annotation documents the things happening during data recording regardless of relevance to data analysis and interpretation. Commonly recorded events in electrophysiological data collection include the initiation, termination, or other features of **sensory presentations** and **participant actions**. Other events may be **unplanned environmental events** (for example, sudden onset of noise and vibration from construction work unrelated to the experiment, or a laboratory device malfunction), events recording **changes in experiment control** parameters as well as **data feature events** and control **mishap events** that cause operation to fall outside of normal experiment parameters. The goals of HED are to provide a standardized annotation and supporting infrastructure.

Goals of HED.

1. **Document the exact nature of events** (sensory, behavioral, environmental, and other) that occur during recorded time series data in order to inform data analysis and interpretation.
2. **Describe the design of the experiment** including participant task(s).
3. **Relate event occurrences** both to the experiment design and to participant tasks and experience.
4. **Provide basic infrastructure** for building and using machine-actionable tools to systematically analyze data associated with recorded events in and across data sets, studies, paradigms, and modalities.

Current systems in neuroimaging experiments do not record events beyond simple numerical (3) or text (Event type Target) labels whose more complete and precise meanings are known only to the experimenter(s).

A central goal of HED is to enable building of archives of brain imaging data in a form amenable to new forms of larger scale analysis, both within and across studies. Such event-related analysis requires that the nature(s) of the recorded events be specified in a common language. The HED project seeks to formalize the development of this language, to develop and distribute tools that maximize its ease of use, and to inform new and existing researchers of its purpose and value.

3.4 1.3. HED design principles

The near decade-long effort to develop effective event annotation for neurophysiological and behavioral data, culminating to date in HED-3G, has revealed the importance of four principles (aka the PASS principles), all of which have roots in other fields:

The PASS principles for HED design.

1. **Preserve orthogonality** of concepts in specifying vocabularies.
2. **Abstract functionality** into layers (e.g., more general vs. more specific).
3. **Separate content** from presentation.
4. **Separate implementation** from the interface (for flexibility).

Orthogonality, the notion of keeping independently applicable concepts in separate hierarchies (1 above), has long been recognized as a fundamental principle in reusable software design, distilled in the design rule: *Favor composition over inheritance* (Gamma et al. 1994).

Abstraction of functionality into layers (2) and separation of content from presentation (3) are well-known principles in user-interface and graphics design that allow tools to maintain a single internal representation of needed information while emphasizing different aspects of the information when presenting it to users.

Similarly, making validation and analysis code independent of the HEDschema (4) allows redesign of the schema without having to re-implement the annotation tools. A well-specified and stable API (application program interface) empowers tool developers.

CHAPTER

FOUR

1. USING DEFINITIONS

This document contains the specification for third generation HED or HED-3G. It is meant for the implementers and users of HED tools. Other tutorials and tagging guides are available to researchers using HED to annotate their data. This document contains the specification for the first official release of HED-3G (HED versions 8.0.0-xxx and above.) **When the term HED is used in this document, it refers to third generation (HED-3G) unless explicitly stated otherwise.**

The aspects of HED that are described in this document are supported or will soon be supported by validators and other tools and are available for immediate use by annotators. The schema vocabulary can be viewed using an expandable schema viewer.

All HED-related source and documentation repositories are housed on the HED-standard organization GitHub site, https://github.com/hed-standard, which is maintained by the HED Working Group. HED development is open-source and community-based. Also see the official HED website https://www.hedtags.org for a list of additional resources.

The HED Working Group invites those interested in HED to contribute to the development process. Users are encouraged to use the *Issues* mechanism of the `hed-specification` repository on the GitHub `hed-standard` working group website: https://github.com/hed-standard/hed-specification/issues to ask for help or make suggestions. The HED discussion forum https://github.com/hed-standard/hed-specification/discussions is maintained for in depth discussions of HED issues and evolution.

Several other aspects of HED annotation are being planned, but their specification has not been fully determined. These aspects are not contained in this specification document, but rather are contained in ancillary working documents which are open for discussion. These ancillary specifications include the HED working document on spatial annotation and the HED working document on task annotation.

4.1 1.1. Scope of HED

HED (an acronym for Hierarchical Event Descriptors) is an evolving framework that facilitates the description and formal annotation of events identified in time series data, together with tools for validation and for using HED annotations in data search, extraction, and analysis. This specification describes the official release of third generation of HED or HED-3G, which is HED version 8.0.0.

Third generation HED represents a significant advance in documenting the content and intent of experiments in a format that enables large-scale cross-study analysis of time-series behavioral and neuroimaging data, including but not limited to EEG, MEG, iEEG, eye-tracking, motion-capture, EKG, and audiovisual recording. In principle, third generation HED might be extended or adapted to annotate events in any other type of ordered or time series data.

Specifically, the goal of HED is to allow researchers to annotate what happened during an experiment, including experimental stimuli and other sensory events, participant responses and actions, experimental design, the role of events in the task, and the temporal structure of the experiment. The resulting annotation is machine-actionable, meaning that it can be used as input to algorithms without manual intervention. HED facilitates detailed comparisons of data across studies.

HED annotations may be included in BIDS (Brain Imaging Data Structure) datasets https://bids.neuroimaging.io as described in Chapter 7: HED in BIDS.

4.2 1.2. Brief history of HED

HED was originally proposed by Nima Bigdely-Shamlo in 2010 to support annotation in HeadIT and early public repository for EEG data hosted by the Swartz Center for Computational Neuroscience, UCSD (Bigdely-Shamlo et al. 2013). HED-1G was partially based on CogPO (Turner and Laird 2012).

Event annotation in HED-1G was organized around a single hierarchy whose root was the *Time-Locked Event*. Users could extend the HED-1G hierarchy at its deepest (leaf) nodes. First generation HED (HED-1G, versions < 5.0.0) attempted to describe events using a strictly hierarchical vocabulary.

HED-1G was oriented toward annotating stimuli and responses, but its lack of orthogonality in vocabulary design presented major difficulties. If *Red/Triangle* and *Green/Triangle* are terms in a hierarchy, one is also likely to need *Red/Square* and Green/Square* as well as other color and shape combinations.

HED-2G (versions 5.0.0 - 7.x.x) introduced a more orthogonal vocabulary, meaning that independent terms were in different subtrees of the vocabulary tree. Separating independent concepts such as shapes and colors into separate hierarchies, eliminates an exponential vocabulary growth due to term duplication in different branches of the hierarchy.

Parentheses were introduced so that terms could be grouped. Tools for validation and epoching based on HED tags were built, and large-scale cross-study "mega-analyses" were performed. However, as more complicated and varied datasets were annotated using HED-2G, the vocabulary started to become less manageable as HED tried to adapt to more complex annotation demands.

In 2019, work began on a rethinking of the HED vocabulary design, resulting in the release of the third generation of HED (HED-3G) in August 2021. HED-3G represents a dramatic increase in annotation capacity, but also a significant simplification of the user experience.

New in HED (versions 8.0.0+).

1. Improved vocabulary structure
2. Short-form annotation
3. Library schema
4. Definitions
5. Temporal scope
6. Encoding of experimental design

Following basic design principles, the HED Working Group redesigned the HED vocabulary tree to be organized in a balanced hierarchy with a limited number of subcategories at each node. (See the expandable schema browser to browser the vocabulary and explore the overall organization. Chapter2:Terminology defines some important HED tags and terminology used in HED.)

A major improvement in vocabulary design was the adoption of the requirement that individual nodes or terms in the HED vocabulary must be unique. This allows users to use individual node names (short form) rather than the full paths to the schema root during annotation, resulting in substantially simpler, more readable annotations.

To enable and regulate the extension process, the root HED-3G head schema specified here includes, for the first time, *HED library schema* to extend the HED vocabulary to include terms and concepts of importance to individual user communities – for example researchers who design and perform experiments to study brain and language, brain and music, or brain dynamics in natural or virtual reality environments. The HED library schema concept may also be used to extend HED annotation to encompass specialized vocabularies used in clinical research and practice.

HED-3G also introduced a number of advanced tagging concepts that allow users to represent events with temporal duration, as well as annotations that represent experimental design.

4.3 1.2. Goals of HED

Event annotation documents the things happening during data recording regardless of relevance to data analysis and interpretation. Commonly recorded events in electrophysiological data collection include the initiation, termination, or other features of **sensory presentations** and **participant actions**. Other events may be **unplanned environmental events** (for example, sudden onset of noise and vibration from construction work unrelated to the experiment, or a laboratory device malfunction), events recording **changes in experiment control** parameters as well as **data feature events** and control **mishap events** that cause operation to fall outside of normal experiment parameters. The goals of HED are to provide a standardized annotation and supporting infrastructure.

Goals of HED.

1. **Document the exact nature of events** (sensory, behavioral, environmental, and other) that occur during recorded time series data in order to inform data analysis and interpretation.
2. **Describe the design of the experiment** including participant task(s).
3. **Relate event occurrences** both to the experiment design and to participant tasks and experience.
4. **Provide basic infrastructure** for building and using machine-actionable tools to systematically analyze data associated with recorded events in and across data sets, studies, paradigms, and modalities.

Current systems in neuroimaging experiments do not record events beyond simple numerical (3) or text (Event type Target) labels whose more complete and precise meanings are known only to the experimenter(s).

A central goal of HED is to enable building of archives of brain imaging data in a form amenable to new forms of larger scale analysis, both within and across studies. Such event-related analysis requires that the nature(s) of the recorded events be specified in a common language. The HED project seeks to formalize the development of this language, to develop and distribute tools that maximize its ease of use, and to inform new and existing researchers of its purpose and value.

4.4 1.3. HED design principles

The near decade-long effort to develop effective event annotation for neurophysiological and behavioral data, culminating to date in HED-3G, has revealed the importance of four principles (aka the PASS principles), all of which have roots in other fields:

The PASS principles for HED design.

1. **Preserve orthogonality** of concepts in specifying vocabularies.
2. **Abstract functionality** into layers (e.g., more general vs. more specific).
3. **Separate content** from presentation.
4. **Separate implementation** from the interface (for flexibility).

Orthogonality, the notion of keeping independently applicable concepts in separate hierarchies (1 above), has long been recognized as a fundamental principle in reusable software design, distilled in the design rule: *Favor composition over inheritance* (Gamma et al. 1994).

Abstraction of functionality into layers (2) and separation of content from presentation (3) are well-known principles in user-interface and graphics design that allow tools to maintain a single internal representation of needed information while emphasizing different aspects of the information when presenting it to users.

Similarly, making validation and analysis code independent of the HEDschema (4) allows redesign of the schema without having to re-implement the annotation tools. A well-specified and stable API (application program interface) empowers tool developers.

CHAPTER

FIVE

5. COMPLEX CONDITIONS

This document contains the specification for third generation HED or HED-3G. It is meant for the implementers and users of HED tools. Other tutorials and tagging guides are available to researchers using HED to annotate their data. This document contains the specification for the first official release of HED-3G (HED versions 8.0.0-xxx and above.) **When the term HED is used in this document, it refers to third generation (HED-3G) unless explicitly stated otherwise.**

The aspects of HED that are described in this document are supported or will soon be supported by validators and other tools and are available for immediate use by annotators. The schema vocabulary can be viewed using an expandable schema viewer.

All HED-related source and documentation repositories are housed on the HED-standard organization GitHub site, https://github.com/hed-standard, which is maintained by the HED Working Group. HED development is open-source and community-based. Also see the official HED website https://www.hedtags.org for a list of additional resources.

The HED Working Group invites those interested in HED to contribute to the development process. Users are encouraged to use the *Issues* mechanism of the `hed-specification` repository on the GitHub `hed-standard` working group website: https://github.com/hed-standard/hed-specification/issues to ask for help or make suggestions. The HED discussion forum https://github.com/hed-standard/hed-specification/discussions is maintained for in depth discussions of HED issues and evolution.

Several other aspects of HED annotation are being planned, but their specification has not been fully determined. These aspects are not contained in this specification document, but rather are contained in ancillary working documents which are open for discussion. These ancillary specifications include the HED working document on spatial annotation and the HED working document on task annotation.

5.1 1.1. Scope of HED

HED (an acronym for Hierarchical Event Descriptors) is an evolving framework that facilitates the description and formal annotation of events identified in time series data, together with tools for validation and for using HED annotations in data search, extraction, and analysis. This specification describes the official release of third generation of HED or HED-3G, which is HED version 8.0.0.

Third generation HED represents a significant advance in documenting the content and intent of experiments in a format that enables large-scale cross-study analysis of time-series behavioral and neuroimaging data, including but not limited to EEG, MEG, iEEG, eye-tracking, motion-capture, EKG, and audiovisual recording. In principle, third generation HED might be extended or adapted to annotate events in any other type of ordered or time series data.

Specifically, the goal of HED is to allow researchers to annotate what happened during an experiment, including experimental stimuli and other sensory events, participant responses and actions, experimental design, the role of events in the task, and the temporal structure of the experiment. The resulting annotation is machine-actionable, meaning that it can be used as input to algorithms without manual intervention. HED facilitates detailed comparisons of data across studies.

HED annotations may be included in BIDS (Brain Imaging Data Structure) datasets https://bids.neuroimaging.io as described in Chapter 7: HED in BIDS.

5.2 1.2. Brief history of HED

HED was originally proposed by Nima Bigdely-Shamlo in 2010 to support annotation in HeadIT and early public repository for EEG data hosted by the Swartz Center for Computational Neuroscience, UCSD (Bigdely-Shamlo et al. 2013). HED-1G was partially based on CogPO (Turner and Laird 2012).

Event annotation in HED-1G was organized around a single hierarchy whose root was the *Time-Locked Event*. Users could extend the HED-1G hierarchy at its deepest (leaf) nodes. First generation HED (HED-1G, versions < 5.0.0) attempted to describe events using a strictly hierarchical vocabulary.

HED-1G was oriented toward annotating stimuli and responses, but its lack of orthogonality in vocabulary design presented major difficulties. If *Red/Triangle* and *Green/Triangle* are terms in a hierarchy, one is also likely to need *Red/Square* and Green/Square* as well as other color and shape combinations.

HED-2G (versions 5.0.0 - 7.x.x) introduced a more orthogonal vocabulary, meaning that independent terms were in different subtrees of the vocabulary tree. Separating independent concepts such as shapes and colors into separate hierarchies, eliminates an exponential vocabulary growth due to term duplication in different branches of the hierarchy.

Parentheses were introduced so that terms could be grouped. Tools for validation and epoching based on HED tags were built, and large-scale cross-study "mega-analyses" were performed. However, as more complicated and varied datasets were annotated using HED-2G, the vocabulary started to become less manageable as HED tried to adapt to more complex annotation demands.

In 2019, work began on a rethinking of the HED vocabulary design, resulting in the release of the third generation of HED (HED-3G) in August 2021. HED-3G represents a dramatic increase in annotation capacity, but also a significant simplification of the user experience.

New in HED (versions 8.0.0+).

1. Improved vocabulary structure
2. Short-form annotation
3. Library schema
4. Definitions
5. Temporal scope
6. Encoding of experimental design

Following basic design principles, the HED Working Group redesigned the HED vocabulary tree to be organized in a balanced hierarchy with a limited number of subcategories at each node. (See the expandable schema browser to browser the vocabulary and explore the overall organization. Chapter2:Terminology defines some important HED tags and terminology used in HED.)

A major improvement in vocabulary design was the adoption of the requirement that individual nodes or terms in the HED vocabulary must be unique. This allows users to use individual node names (short form) rather than the full paths to the schema root during annotation, resulting in substantially simpler, more readable annotations.

To enable and regulate the extension process, the root HED-3G head schema specified here includes, for the first time, *HED library schema* to extend the HED vocabulary to include terms and concepts of importance to individual user communities – for example researchers who design and perform experiments to study brain and language, brain and music, or brain dynamics in natural or virtual reality environments. The HED library schema concept may also be used to extend HED annotation to encompass specialized vocabularies used in clinical research and practice.

HED-3G also introduced a number of advanced tagging concepts that allow users to represent events with temporal duration, as well as annotations that represent experimental design.

5.3 1.2. Goals of HED

Event annotation documents the things happening during data recording regardless of relevance to data analysis and interpretation. Commonly recorded events in electrophysiological data collection include the initiation, termination, or other features of **sensory presentations** and **participant actions**. Other events may be **unplanned environmental events** (for example, sudden onset of noise and vibration from construction work unrelated to the experiment, or a laboratory device malfunction), events recording **changes in experiment control** parameters as well as **data feature events** and control **mishap events** that cause operation to fall outside of normal experiment parameters. The goals of HED are to provide a standardized annotation and supporting infrastructure.

Goals of HED.

1. **Document the exact nature of events** (sensory, behavioral, environmental, and other) that occur during recorded time series data in order to inform data analysis and interpretation.
2. **Describe the design of the experiment** including participant task(s).
3. **Relate event occurrences** both to the experiment design and to participant tasks and experience.
4. **Provide basic infrastructure** for building and using machine-actionable tools to systematically analyze data associated with recorded events in and across data sets, studies, paradigms, and modalities.

Current systems in neuroimaging experiments do not record events beyond simple numerical (3) or text (Event type Target) labels whose more complete and precise meanings are known only to the experimenter(s).

A central goal of HED is to enable building of archives of brain imaging data in a form amenable to new forms of larger scale analysis, both within and across studies. Such event-related analysis requires that the nature(s) of the recorded events be specified in a common language. The HED project seeks to formalize the development of this language, to develop and distribute tools that maximize its ease of use, and to inform new and existing researchers of its purpose and value.

5.4 1.3. HED design principles

The near decade-long effort to develop effective event annotation for neurophysiological and behavioral data, culminating to date in HED-3G, has revealed the importance of four principles (aka the PASS principles), all of which have roots in other fields:

The PASS principles for HED design.

1. **Preserve orthogonality** of concepts in specifying vocabularies.
2. **Abstract functionality** into layers (e.g., more general vs. more specific).
3. **Separate content** from presentation.
4. **Separate implementation** from the interface (for flexibility).

Orthogonality, the notion of keeping independently applicable concepts in separate hierarchies (1 above), has long been recognized as a fundamental principle in reusable software design, distilled in the design rule: *Favor composition over inheritance* (Gamma et al. 1994).

Abstraction of functionality into layers (2) and separation of content from presentation (3) are well-known principles in user-interface and graphics design that allow tools to maintain a single internal representation of needed information while emphasizing different aspects of the information when presenting it to users.

Similarly, making validation and analysis code independent of the HEDschema (4) allows redesign of the schema without having to re-implement the annotation tools. A well-specified and stable API (application program interface) empowers tool developers.

CHAPTER

SIX

DOCUMENTATION

6.1 1. HED publications

Explanation of the history, development, and motivation for third generation HED:

> Robbins, K., Truong, D., Jones, A., Callanan, I., & Makeig, S. (2020, August 1). Building FAIR functionality: Annotating events in time series data using Hierarchical Event Descriptors (HED). https://doi.org/10.31219/osf.io/5fg73

Detailed case study in using HED-3G for tagging:

> Robbins, K., Truong, D., Appelhoff, S., Delorme, A., & Makeig, S. (2021, May 7). Capturing the nature of events and event context using Hierarchical Event Descriptors (HED). BioRxiv, 2021.05.06.442841. https://doi.org/10.1101/2021.05.06.442841

6.2 2. Working documents

Mapping of HED terms and their descriptions to known ontologies is:

> HED-3G Working Document on Ontology mapping https://drive.google.com/file/d/13y17OwwNBlHdhB7hguSmOBdxn0Uk4hsI/view?usp=sharing

Two other working documents hold portions of the HED-3G specification that are under development and will not be finalized for Release 1:

> HED-3G Working Document on Spatial Annotation https://docs.google.com/document/d/1jpSASpWQwOKtan15iQeiYHVewvEeefcBUn1xipNH5-8/view?usp=sharing

> HED-3G Working Document on Task Annotation https://docs.google.com/document/d/1eGRI_gkYutmwmAl524ezwkX7VwikrLTQa9t8PocQMlU/view?usp=sharing

6.3 3. Schema viewers

The HED schema is usually developed in `.mediawiki` format and converted to XML for use by tools. However, researchers wishing to tag datasets will find both of these views hard to read. For this reason, we provide links to three versions of the schema. The expandable HTML viewer is easier to navigate. Annotators can also use CTAGGER, which includes a schema viewer and tagging hints.

Table 1: HED web-based schema vocabulary viewers.

Viewer	Link
Expandable HTML	https://www.hedtags.org/display_hed.html?version=8.0.0
Mediawiki	https://github.com/hed-standard/hed-specification/blob/master/hedwiki/HED8.0.0.mediawiki
XML	https://github.com/hed-standard/hed-specification/blob/master/hedxml/HED8.0.0.xml

6.4 4. HED Websites

The following is a summary of the HED-related websites

Table 2: HED websites.

Description	Site
Information and documentation	
HED organization website	https://www.hedtags.org
HED organization github	https://github.com/hed-standard
HED specification repository	https://github.com/hed-standard/hed-specification
Examples of HED annotation	https://github.com/hed-standard/hed-examples
HED documentation website	https://github.com/hed-standard/hed-standard.github.io
HED Python resources	
Python code repository	https://github.com/hed-standard/hed-python
Python validator and tools	https://github.com/hed-standard/hed-python/tree/master/hedtools
HED JavaScript resources	
HED JavaScript code	https://github.com/hed-standard/hed-javascript
BIDS validator	https://github.com/bids-standard/bids-validator
HED Matlab resources	
Matlab source code	https://github.com/hed-standard/hed-matlab
Annotator resources	
CTAGGER executable jar	https://github.com/hed-standard/hed-java/raw/master/ctagger.jar
CTAGGER repository	https://github.com/hed-standard/CTagger
Java repository	https://github.com/hed-standard/hed-java
Online HED tools	
Online website	https://hedtools.ucsd.edu/hed
Docker deployment	https://github.com/hed-standard/hed-python/tree/master/webtools/deploy_hed

CHAPTER

SEVEN

TOOLS AND SERVICES

7.1 1. CTagger for annotation

The CTagger tool for annotating data provides a graphical user interface (GUI) to assist HED users in the annotation process. CTagger can be run as a standalone application (https://github.com/hed-standard/hed-java/raw/master/ctagger.jar) or using the HEDTools plug-in in EEGLAB.

The tool is designed to ease the process of constructing HED strings, with features including tag search, an expandable schema-browser view, and free-form formatting. The interchangeability between long-short forms introduced in HED-3G is fully supported. CTagger is also compatible with BIDS, allowing users to import BIDS `events.tsv` and `events.json` files to extract the event structure. Once finished tagging, users can export their HED annotation into a json file compatible with BIDS events.json. See CTagger GitHub repository for more details, guides, and tutorials.

7.2 2. Web-based tools

The web-based tools are summarized in this section. All tools are available from the main access point https://hedtools.ucsd.edu/hed. The services are implemented in a Docker module and can be deployed locally provided that Docker is installed on.

Event files are BIDS style tab-separated value files. The first line is always a header line giving the names of the columns, which are used as keys to metadata in accompanying JSON sidecars. The online tools for BIDS events file are designed to help users debug their HED annotations for BIDS datasets before using the BIDS validator.

Web tools for BIDS-style events.tsv files

Assemble events: Assemble HED annotation of a BIDS-style events file.

- Upload the event file and an optional JSON sidecar file.
- Specify the HED version.
- Select the `Assemble` processing option and whether to expand definitions.
- Click the `Process` button.
- The tool assembles a new two-column event file.
- If there are errors, a downloadable file of error messages is returned.
- If no errors, the new event file is returned.

Validate events: Validate a BIDS-style events file with optional JSON sidecar.

- Upload the event file and an optional JSON sidecar file.

- Specify the HED version.
- Select the `Validate` processing option.
- Click the `Process` button.
- The tool first validates the sidecar if present.
- If the sidecar contains no errors, the tool validates the events file with the sidecar.
- If there are any errors, the tool returns a downloadable file of error messages.

Notes:

1. If the HED version number is given, the tool downloads a standard version from GitHub. Otherwise, the user must upload a local HED schema.
2. The `Assemble` creates a two-column event file. The first column is the event onset time and the second column is the full event-level HED annotation.

HED schema tools are designed to assist HED schema developers and library schema developers in making sure that the schema has the correct form and to provide easy conversion between schema formats.

Web tools for HED schema files.

Convert schema: Convert a HED schema between XML and MEDIAWIKI format.

- Upload a HED schema file or gives a URL pointing to a schema file.
- Select the `Convert` option.
- Press the `Process` button.
- The tool returns a downloadable converted file (XML input is converted to MEDIAWIKI and vice versa).
- Errors are reported as message at the bottom of the screen.

Validate schema: Validate a HED schema between XML and MEDIAWIKI format.

- Upload a HED schema file or gives a URL pointing to a schema file.
- Selects the `Validate` option.
- Press the `Process` button.
- The tool returns a downloadable file of error messages if the schema is invalid.

BIDS JSON sidecars have file names ending in `_events.json`. These JSON files contain metadata and HED tags applicable to associated events files.

Web tools for BIDS style JSON sidecar.

Convert to long: Convert the HED tags in a BIDS-style events JSON sidecar to long form.

- Upload the JSON sidecar file.
- Specify the HED version of HED.
- Select the `Convert to long` option.
- Press the `Process` button.

- The tool first validates the sidecar and returns an error file if errors.
- Otherwise, the tool returns a JSON sidecar with the HED tags converted to full-paths.

Convert to short: Convert the HED tags in a BIDS-style events JSON sidecar to short form.

- Upload the JSON sidecar file.
- Specify the HED version.
- Select the `Convert to short` option.
- Press the `Process` button.
- The tool first validates the sidecar and returns an error file if errors.
- Otherwise, the tool returns a JSON sidecar with the HED tags converted to short-form.

Validate sidecar: Validate a single BIDS-style events JSON sidecar.

- Upload the JSON sidecar file.
- Specify the HED version.
- Select the `Validate` option.
- Press the `Process` button.
- The tool validates the sidecar and returns an error file if there are errors.

Notes:

1. If the HED version number is given, the tool downloads a standard version from GitHub. Otherwise, the user must upload a local HED schema.

Spreadsheets (either in Excel or tab-separated-value format) are convenient for organizing tags.

Web tools for spreadsheets of HED tags.

Convert to long: Convert the HED tags in tag spreadsheet to long form.

- Upload the spreadsheet file, indicating whether the first row is a header.
- Select a worksheet if the spreadsheet is an Excel file with multiple worksheets.
- Specify the HED version.
- Select the columns containing HED tags or are prefix columns.
- Select the `Convert to long` option.
- Press the `Process` button.
- The tool first validates the spreadsheet and returns an error file if errors.
- Otherwise, the tool returns a spreadsheet with the HED tags converted to full-paths.

Convert to short: Convert the HED tags in a spreadsheet to short form.

- Upload the spreadsheet file, indicating whether the first row is a header.
- Select a worksheet if the spreadsheet is an Excel file with multiple worksheets.
- Specify the HED version.
- Select the columns containing HED tags or are prefix columns.

- Select the `Convert to short` option._ Press the `Process` button.
- The tool first validates the spreadsheet and returns an error file if errors.
- Otherwise, the tool returns a spreadsheet with the HED tags converted to short-form.

Validate: Validate the HED tags in a spreadsheet.

- Upload the spreadsheet file, indicating whether the first row is a header.
- Selects a worksheet if the spreadsheet is an Excel file with multiple worksheets.
- Specify the HED version.
- Select the columns containing HED tags or are prefix columns.
- Select the `Validate` option.
- Press the `Process` button.
- The tool validates the spreadsheet and returns an error file if the spreadsheet is invalid.

Notes:

1. If the HED version number is given, the tool downloads a standard version from GitHub. Otherwise, the user must upload a local HED schema.

Online validation of HED strings.

Web tools for processing strings

Convert to long: Convert a HED string to long form.

- Paste a string into the input text box.
- Specify the HED version.
- Select the `Convert to long` option.
- Press the `Process` button.
- The tool first validates the string and returns errors in the lower text box if any.
- Otherwise the tool returns the full path version of the strings in the lower text box.

Convert to short: Convert a HED tag string to short form.

- Paste a string into the input text box.
- Specify the HED version.
- Selects the `Convert to short` option.
- Press the `Process` button.
- The tool first validates the string and returns errors in the lower text box if any.
- Otherwise the tool returns the short-form versions of the tag strings in the lower text box.

Validate: Validate a HED string.

- Paste a string into the input text box.
- Specify the HED version.
- Select the `Validate` option and presses the `Process` button.

- The tool validates the string and returns any error messages in the lower text box.

Notes:

1. If the HED version number is given, the tool downloads a standard version from GitHub. Otherwise, the user must upload a local HED schema.

7.3 3. Web-based services

HED supports a number of web-based tools for HED validation, schema conversion and validation, JSON dictionary validation (as for a BIDS JSON sidecar for events), and validation of a single BIDS event file with supporting JSON sidecar.

Additional web-based tools are planned for various analysis and conversion tasks. In addition, a HED web service interface is available for accessing many of the tools programmatically, including from MATLAB and Python programs. The following table summarizes the location of the relevant URLs for online deployments of HED web-based tools and services.

Table 1: URLs for HED online services.

Service	URL
Online HED tools	https://hedtools.ucsd.edu/hed
CSRF token access	https://hedtools.ucsd.edu/hed/services
Service request	https://hedtools.ucsd.edu/hed/services_submit

HED services are accessed by passing a JSON dictionary of parameters in a request to the online server. All requests include a `service` name and additional parameters. The parameters are explained in a subsequent table. Parameter values listed in square brackets (e,g, [`a`, `b`]) indicate that only one of `a` or `b` should be provided.

Table 2: Summary of HED ReST services

Service	Parameters	Descriptions
get_services	none	Returns a list of available services.
events_assemble	events_string,json_string,[schema_version,schema_string],expand_defs	Returns an error file or a string of assembled events.
events_validate	events_string,json_string,[schema_version,schema_string],check_for_warnings	Returns an error file if errors.
sidecar_to_long	json_string,[schema_version,schema_string]	Returns an error file or converted file.
sidecar_to_short	json_string,[schema_version,schema_string]	Returns an error file or a long form JSON file.
sidecar_validate	json_string,[schema_version,schema_string],check_for_warnings	Returns an error file if errors.
spreadsheet_validate	spreadsheet_string,[schema_version,schema_string],check_for_warnings	Returns an error file if errors.
strings_to_long	string_list,[schema_version,schema_string]	Returns strings or a list of strings to long form.
strings_to_short	string_list,[schema_version,schema_string]	Converts strings or a list of short-form strings.
strings_validate	hed_strings,[schema_version,schema_string]	Validates a string of hed strings and returns a list of errors.

The following table gives an explanation of the parameters used for various services.

Table 3: Parameters for web services.

Key value	Type	Description
check_for_warnings	boolean	If true, check for warnings when validating.
defs_expand	boolean	If true assembly replaces *def/XXX* with *def-expand/XXX*.
events_string	string	Events tsv file with header passed as a string.
hed_columns	list of numbers	A list of HED string column numbers (starting with 1).
hed_schema_string	string	HED schema in XML format as a string.
hed_strings	list of strings	A list containing HED strings.
json_string	string	BIDS-style JSON events sidecar as a string.
json_strings	string	A list of BIDS-style JSON sidecars as strings.
schema_string	string	A HED schema file as a string.
schema_version	string	Version of HED to be accessed if relevant.
service	string	The name of the requested service.
spreadsheet_string	string	A spreadsheet tsv as a string.

The web-services always return a JSON dictionary with four keys: `service`, `results`, `error_type`, and `error_msg`. If `error_type` and `error_msg` are not empty, the operation failed, while if these fields are empty, the operation completed. Completed operations always return their results in the `results` dictionary. Keys in the `results` dictionary return as part of a HED web service response.

Table 4: The results dictionary.

Key	Type	Description
command	string	Command executed in response to the service request.
data	string	A list of errors or the processed result.
schema_version	string	The version of the HED schema used in the processing.
msg_category	string	One of success, warning, or failure depending on the result.
msg	string	Explanation of the result of service processing.

The `hedweb/examples/matlab` directory of the hed-python repository gives running MATLAB examples of how to call these services in MATLAB.

7.4 4. Python tools

The python code for validation is in the `hedtools` project located in the `hed-python` repository https://github.com/hed-standard/hed-python. You can install the tools using `pip` if you have downloaded the `hed-python` repository:

```
pip install <hedtools-local-path>
```

The validation functions are in the `hed.validator` module. The data representations for various items such as dictionaries or event files can be found in the `hed.models` module. The `hed_input.py` module reads in a spreadsheet and possibly a dictionary and creates a `HedInput` object representing the spreadsheet. The `hed-validator.py` module creates a `HedValidator` object that takes a `HedSchema` object to use in subsequent validation. The `validate_input` method of `HedValidator` validates HED input in various formats and returns a list of issues.

7.5 5. JavaScript tools

The JavaScript code for HED validation is in the validation directory of the `hed-javascript` repository located at https://github.com/hed-standard/hed-javascript.

7.5.1 5.1 Installation

You can install the validator using `npm`:

```
npm install hed-validator
```

7.5.2 5.2 Package organization

This package contains two sub-packages.

`hedValidator.validator` validates HED strings and contains the functions:

> `buildSchema` imports a HED schema and returns a JavaScript Promise object.`validateHedString` validates a single HED string using the returned schema object.

`hedValidator.converter` converts HED strings between short and long forms and contains the following functions:

> `buildSchema` behaves similarly to the `buildSchema` function in `hedValidator.validator` except that it does not work with attributes.
>
> `convertHedStringToShort` converts HED strings from long form to short form.
>
> `convertHedStringToLong` converts HED strings from short form to long form.

7.5.3 5.3 Programmatic interface

The programmatic interface to the HED JavaScript `buildSchema` must be modified to accommodate a base HED schema and arbitrary library schemas. The BIDS validator will require additional changes to locate the relevant HED schemas from the specification given by `"HEDVersion"` in `dataset_description.json`.

The programmatic interface is similar to the JSON specification of the proposed BIDS implementation except that the `"fileName"` key has been replaced by a `"path"` key to emphasize that callers must replace filenames with full paths before calling `buildSchema`.

Example: JSON passed to buildSchema.

```
{
    "path": "/data/wonderful/code/mylocal.xml",
    "libraries": {
        "la": {
            "libraryName": "libraryA",
            "version": "1.0.2"
        },
        "lb": {
            "libraryName": "libraryB",
            "path": "/data/wonderful/code/HED_libraryB_0.5.3.xml"
        }
```

(continues on next page)

(continued from previous page)

```
    }
}
```

NOTE: This interface is proposed and is awaiting resolution of BIDS PR #820 on file passing to BIDS.

7.6 6. MATLAB tools

HED validation can be done using the online web-services from MATLAB as shown in the `./examples/matlab` directory of the hedweb project in the hed-python repository.