Events

Events are the primary means of organizing structs data. They have two forms: **instants**, referring to a **single moment in time**, and **intervals**, referring to a **range of time**. The goal of events is to make it easy to find, compare, and trend data.

Fields

Unlike most structs databases, event databases may include as many custom fields as required, so long as they do not conflict with the required standard fields:

UEID

Universally unique event identifier (UUID). Intended to permanently, globally specify each event. Should be generated at the creation of the event to ensure consistency even if data is reprocessed.

Event ID

Optional numeric reference to an event definition (also see below). If not provided, defaults to `0`.

Type

Indicates how the event should be viewed and interpreted. The options are defined by XINA.

Level

Indicates how the event should be viewed and interpreted. The options are defined by XINA.

Label

Required plain text description of the event. Limited to to 128 bytes for indexing.

Content

Optional plain text, HTML, or JSON of unlimited length.

Meta

Optional JSON object of arbitrary additional content.

Types

XINA defines a fixed set of standard event types, each with an associated numeric code. The type is stored as the code in the database for performance reasons; for practical purposes most actions can use the type name directly, unless interacting directly with the API.

Standard Types

Code	Name	Ins	Int	Description
0	message	?	?	Basic event, ID optional
1	marker	?	?	Organized event, ID required
2	alert	?	?	Organized event, ID required, level (severity) required
2000	test		?	Discrete test period, may not overlap other tests, ID optional
2001	activity		?	Discrete activity period, may not overlap other activities, ID optional
2002	phase		?	Discrete phase period, may not overlap other phases, ID optional
2010	filter		?	Filter state
3000	data	?	?	General purpose data set
3001	spectrum	?	?	General purpose spectrum data

Additional types will be added in the future as needed, with codes based on this chart:

Standard Type Code Ranges

code	ins	int	description
0-999	?	?	General types for instants and intervals
1000-1999	?		General types for instants only
2000-2999		?	General types for intervals only
3000-3999	?	?	Data set types for instants and intervals
4000-4999	?		Data set types for instants only
5000-5999		?	Data set types for intervals only

Definitions

As with mnemonics, events may be identified with event definitions. However, unlike mnemonics, not every event requires a definition. The event ID field associates an event with an event definition. Each event ID is associated with a unique name, describing the definition. These work similarly to mnemonic names for purposes for definition creation. If an event is inserted with an unrecognized name, a new definition will be created for that name and assigned a new event ID.

Context

An event database may either be a child of a model or pipe group. A model event database is essentially like any other XINA database, but with support for the STRUCT EVENT API actions.

Pipe event databases are more restrictive. The events must be embedded in the data set of the pipe, and cannot be inserted manually. Each event is associated with the archive in which it starts. Events may cross archive boundaries by initially being inserted as `open` events (having a start time but no end time) and later using the `$event.close` operation to assign an end time. Additionally, pipe event databases have an associated event change database to track manually applied updates outside of the source data set. This allows changes to be preserved if data is remined from the archive files. The intention is that these accurately reflect the archived data, so fields must opt-in to being editable by this operation.

Data Formats

The `data` event type indicates a basic data set. This is typically used with the single file per event database structure, in which case the file will contain the data set. For event databases without files, the data is expected to be stored in the `content` field. This is only recommended for small datasets (less than 1MB).

Files must be either ASCII or UTF-8 encoded. New lines will be interpretted from either `\n` or `\r\n`. The `conf` object may define other customization of the format:

Conf Definition

Key	Value	Default	Description
`delimiter`	`string`	auto detect (`','`, `'\t'`, `';'`)	value delimiter
`quoteChar`	`character`	`"` (double quote character)	value quote character
`ignoreLines`	`number`	`0`	number of lines to skip before the header
`invalid`	`null`, `'NaN'`, `number`	`null`	preferred interpretation of invalid literal
`nan`	`null`, `'NaN'`, `number`	`null`	preferred interpretation of `'Nan'` literal
`pInfinity`	`null`, `'Inf'`, `number`	`null`	preferred interpretation of positive `'Infinity'` literal
`nInfinity`	`null`, `'Inf'`, `number`	`null`	preferred interpretation of negative `'Infinity'` literal
`utc`	`boolean`	`false`	if `true`, interpret all unzoned timestamps as UTC

Starting after the number provided for `ignoreLines`, the content must include a header for each column, with a name and optional unit in parentheses. Special standard unit names may be used to indicate time types, which will apply different processing to the column:

Unit	Description
`ts`	text timestamp, interpretted in local browser timezone (absent explicit zone)
`ts_utc`	text timestamp, interpretted as UTC timezone (absent explicit zone)
`unix_s`	Unix time in seconds
`unix_ms`	Unix time in milliseconds

Unit	Description
`unix_us`	Unix time in microseconds

Revision #49
Created 27 July 2022 18:13:41 by Nick Dobson
Updated 16 September 2024 20:44:11 by Bradley Tse