

Generic Trace Generator (GTG)

0.1

Generated by Doxygen 1.8.7

Mon Jun 2 2014 10:59:53

Contents

1	The GTG library	1
1.1	Presentation	1
2	Module Index	3
2.1	Modules	3
3	Data Structure Index	5
3.1	Data Structures	5
4	File Index	7
4.1	File List	7
5	Module Documentation	9
5.1	Trace type handler	9
5.1.1	Detailed Description	9
5.1.2	Enumeration Type Documentation	9
5.1.2.1	traceType	9
5.1.3	Function Documentation	10
5.1.3.1	bufferedModeActivated	10
5.1.3.2	getName	10
5.1.3.3	getTraceType	10
5.1.3.4	setTraceType	10
5.2	To init the generated trace file(s)	11
5.2.1	Detailed Description	11
5.2.2	Function Documentation	11
5.2.2.1	addEntityValue	11
5.2.2.2	endTrace	11
5.2.2.3	initTrace	11
5.2.2.4	setCompress	12
5.3	Functions related to the containers	13
5.3.1	Detailed Description	13
5.3.2	Function Documentation	13
5.3.2.1	AddComment	13

5.3.2.2	addContainer	13
5.3.2.3	addContType	13
5.3.2.4	destroyContainer	14
5.4	Functions related to the states	15
5.4.1	Detailed Description	15
5.4.2	Function Documentation	15
5.4.2.1	addStateType	15
5.4.2.2	popState	15
5.4.2.3	pushState	15
5.4.2.4	setState	16
5.5	Functions related to the events	17
5.5.1	Detailed Description	17
5.5.2	Function Documentation	17
5.5.2.1	addEvent	17
5.5.2.2	addEventType	17
5.6	Functions related to links	18
5.6.1	Detailed Description	18
5.6.2	Function Documentation	18
5.6.2.1	addLinkType	18
5.6.2.2	endLink	18
5.6.2.3	startLink	19
5.7	Functions related to variables	20
5.7.1	Detailed Description	20
5.7.2	Function Documentation	20
5.7.2.1	addVar	20
5.7.2.2	addVarType	20
5.7.2.3	setVar	20
5.7.2.4	subVar	21
5.8	Defined colors for GTG	22
5.8.1	Detailed Description	23
5.8.2	Function Documentation	23
5.8.2.1	GTG_COLOR_GET_BLUE	23
5.8.2.2	GTG_COLOR_GET_GREEN	23
5.8.2.3	GTG_COLOR_GET_RED	23
5.8.2.4	GTG_COLOR_SET_COLOR	23
5.8.3	Variable Documentation	23
5.8.3.1	GTG_BLACK	23
5.8.3.2	GTG_BLUE	23
5.8.3.3	GTG_BROWN	23
5.8.3.4	GTG_DARKBLUE	24

5.8.3.5	GTG_DARKGREY	24
5.8.3.6	GTG_DARKPINK	24
5.8.3.7	GTG_GREEN	24
5.8.3.8	GTG_GRENAT	24
5.8.3.9	GTG_KAKI	24
5.8.3.10	GTG_LIGHTBROWN	24
5.8.3.11	GTG_LIGHTGREY	24
5.8.3.12	GTG_LIGHTPINK	24
5.8.3.13	GTG_MAUVE	24
5.8.3.14	GTG_ORANGE	24
5.8.3.15	GTG_PINK	24
5.8.3.16	GTG_PURPLE	25
5.8.3.17	GTG_RED	25
5.8.3.18	GTG_REDBLOOD	25
5.8.3.19	GTG_SEABLUE	25
5.8.3.20	GTG_TEAL	25
5.8.3.21	GTG_WHITE	25
5.8.3.22	GTG_YELLOW	25
5.9	Memory management	26
5.9.1	Detailed Description	26
5.9.2	Function Documentation	26
5.9.2.1	gtg_block_free	26
5.9.2.2	gtg_block_malloc	26
5.9.2.3	gtg_block_memory_init	26
5.10	OTF interface in C of the traceGeneratorBasic API	27
5.10.1	Detailed Description	28
5.10.2	Function Documentation	28
5.10.2.1	OTF_get_color	28
5.10.2.2	OTFAddComment	28
5.10.2.3	OTFAddContType	28
5.10.2.4	OTFAddEntityValue	28
5.10.2.5	OTFAddEvent	29
5.10.2.6	OTFAddEventType	29
5.10.2.7	OTFAddLinkType	29
5.10.2.8	OTFAddStateType	30
5.10.2.9	OTFAddVar	30
5.10.2.10	OTFAddVarType	30
5.10.2.11	OTFDestroyContainer	30
5.10.2.12	OTFEndLink	31
5.10.2.13	OTFEndTrace	31

5.10.2.14 OTFInitTrace	31
5.10.2.15 OTFPopState	31
5.10.2.16 OTFPushState	32
5.10.2.17 OTFSetCompress	32
5.10.2.18 OTFSetState	32
5.10.2.19 OTFSetVar	33
5.10.2.20 OTFStartContainer	34
5.10.2.21 OTFStartLink	34
5.10.2.22 OTFSubVar	34
5.11 Page interface in C of the GTGBasic1 API	36
5.11.1 Detailed Description	37
5.11.2 Function Documentation	37
5.11.2.1 Page_get_color	37
5.11.2.2 pageAddComment	37
5.11.2.3 pageAddContainer	37
5.11.2.4 pageAddContType	38
5.11.2.5 pageAddEntityValue	38
5.11.2.6 pageAddEvent	38
5.11.2.7 pageAddEventType	39
5.11.2.8 pageAddLinkType	40
5.11.2.9 pageAddStateType	40
5.11.2.10 pageAddVar	40
5.11.2.11 pageAddVarType	41
5.11.2.12 pageDestroyContainer	41
5.11.2.13 pageEndLink	41
5.11.2.14 pageEndTrace	42
5.11.2.15 pageGetName	42
5.11.2.16 pageInitTrace	42
5.11.2.17 pagePopState	42
5.11.2.18 pagePushState	43
5.11.2.19 pageSeqAddContainer	43
5.11.2.20 pageSetCompress	43
5.11.2.21 pageSetState	43
5.11.2.22 pageSetVar	44
5.11.2.23 pageStartLink	44
5.11.2.24 pageSubVar	44
5.11.2.25 viteEndTrace	45
5.12 Functions for postponing event-processing function calls	46
5.12.1 Detailed Description	46
5.12.2 Function Documentation	46

5.12.2.1	gtg_record	46
5.12.2.2	gtg_write_events	46
5.13	Types used	47
5.13.1	Detailed Description	47
5.13.2	Typedef Documentation	47
5.13.2.1	varPrec	47
5.13.3	Enumeration Type Documentation	47
5.13.3.1	trace_return_t	47
6	Data Structure Documentation	49
6.1	__OTF2_String Struct Reference	49
6.1.1	Field Documentation	49
6.1.1.1	id	49
6.1.1.2	name	49
6.2	Container Struct Reference	49
6.2.1	Detailed Description	50
6.2.2	Field Documentation	50
6.2.2.1	alias	50
6.2.2.2	alias	50
6.2.2.3	ctType	50
6.2.2.4	ctType	50
6.2.2.5	evt_writer	50
6.2.2.6	id	50
6.2.2.7	id	50
6.2.2.8	name	50
6.2.2.9	name	50
6.2.2.10	parent	50
6.2.2.11	state_stack	50
6.2.2.12	token	50
6.3	ContainerType Struct Reference	50
6.3.1	Field Documentation	51
6.3.1.1	alias	51
6.3.1.2	alias	51
6.3.1.3	id	51
6.3.1.4	id	51
6.3.1.5	name	51
6.3.1.6	name	51
6.3.1.7	parent_type	51
6.3.1.8	token	51
6.4	EntityValue Struct Reference	51

6.4.1	Detailed Description	51
6.4.2	Field Documentation	51
6.4.2.1	alias	51
6.4.2.2	alias	51
6.4.2.3	groupId	51
6.4.2.4	id	51
6.4.2.5	id	52
6.4.2.6	name	52
6.4.2.7	name	52
6.4.2.8	token	52
6.5	EventType Struct Reference	52
6.5.1	Detailed Description	52
6.5.2	Field Documentation	52
6.5.2.1	alias	52
6.5.2.2	alias	52
6.5.2.3	contType	52
6.5.2.4	contType	52
6.5.2.5	id	52
6.5.2.6	id	52
6.5.2.7	name	52
6.5.2.8	name	52
6.5.2.9	token	53
6.6	gtg_color Struct Reference	53
6.6.1	Detailed Description	53
6.6.2	Field Documentation	53
6.6.2.1	color_name	53
6.6.2.2	rgb	53
6.7	gtg_list Struct Reference	53
6.7.1	Field Documentation	53
6.7.1.1	next	53
6.7.1.2	prev	54
6.8	gtg_memory Struct Reference	54
6.8.1	Field Documentation	54
6.8.1.1	block_len	54
6.8.1.2	current_mem	54
6.8.1.3	first_free	54
6.8.1.4	first_mem	54
6.8.1.5	first_new	54
6.8.1.6	mem_len	54
6.8.1.7	nb_allocated	54

6.9	gtg_paje_edp_s Struct Reference	54
6.9.1	Field Documentation	55
6.9.1.1	name	55
6.9.1.2	next	55
6.9.1.3	type	55
6.10	gtg_paje_eventdef_s Struct Reference	55
6.10.1	Field Documentation	55
6.10.1.1	first	55
6.10.1.2	id	55
6.10.1.3	last	55
6.10.1.4	name	55
6.11	Link Struct Reference	55
6.11.1	Field Documentation	55
6.11.1.1	src	55
6.11.1.2	time	56
6.12	LinkType Struct Reference	56
6.12.1	Detailed Description	56
6.12.2	Field Documentation	56
6.12.2.1	alias	56
6.12.2.2	alias	56
6.12.2.3	contType	56
6.12.2.4	contType	56
6.12.2.5	destType	56
6.12.2.6	destType	56
6.12.2.7	id	56
6.12.2.8	id	56
6.12.2.9	name	56
6.12.2.10	name	56
6.12.2.11	srcType	57
6.12.2.12	srcType	57
6.12.2.13	token	57
6.13	otf_color Struct Reference	57
6.13.1	Field Documentation	57
6.13.1.1	blue	57
6.13.1.2	colorID	57
6.13.1.3	green	57
6.13.1.4	red	57
6.14	State Struct Reference	57
6.14.1	Detailed Description	58
6.14.2	Field Documentation	58

6.14.2.1	cont	58
6.14.2.2	cont	58
6.14.2.3	stateType	58
6.14.2.4	stateType	58
6.14.2.5	token	58
6.14.2.6	value	58
6.14.2.7	value	58
6.15	StateType Struct Reference	58
6.15.1	Detailed Description	58
6.15.2	Field Documentation	58
6.15.2.1	alias	58
6.15.2.2	alias	58
6.15.2.3	groupId	58
6.15.2.4	id	59
6.15.2.5	id	59
6.15.2.6	name	59
6.15.2.7	name	59
6.15.2.8	token	59
6.16	Variable Struct Reference	59
6.16.1	Field Documentation	59
6.16.1.1	id	59
6.16.1.2	id	59
6.16.1.3	parent	59
6.16.1.4	parent	59
6.16.1.5	token	59
6.16.1.6	type	59
6.16.1.7	type	59
6.16.1.8	value	59
6.17	VariableType Struct Reference	60
6.17.1	Detailed Description	60
6.17.2	Field Documentation	60
6.17.2.1	alias	60
6.17.2.2	alias	60
6.17.2.3	contType	60
6.17.2.4	contType	60
6.17.2.5	id	60
6.17.2.6	id	60
6.17.2.7	name	60
6.17.2.8	name	60
6.17.2.9	token	60

7	File Documentation	61
7.1	GTG.h File Reference	61
7.1.1	Detailed Description	61
7.2	GTG_EZTrace.h File Reference	61
7.2.1	Macro Definition Documentation	63
7.2.1.1	GTG_FLAG_NONE	63
7.2.1.2	GTG_FLAG_NOTBUF	63
7.2.1.3	GTG_FLAG_OUTOFORDER	63
7.2.1.4	GTG_FLAG_USE_MPI	63
7.2.2	Typedef Documentation	63
7.2.2.1	gtg_flag_t	63
7.2.2.2	traceType_t	63
7.2.3	Function Documentation	63
7.2.3.1	gtg_addContainer	63
7.2.3.2	gtg_addContType	63
7.2.3.3	gtg_addEntityValue	63
7.2.3.4	gtg_addEvent	63
7.2.3.5	gtg_addEventType	63
7.2.3.6	gtg_addLinkType	63
7.2.3.7	gtg_addStateType	63
7.2.3.8	gtg_addVarType	63
7.2.3.9	gtg_bufferedModeActivated	63
7.2.3.10	gtg_destroyContainer	63
7.2.3.11	gtg_endLink	63
7.2.3.12	gtg_endTrace	64
7.2.3.13	gtg_getName	64
7.2.3.14	gtg_getTraceType	64
7.2.3.15	gtg_initTrace	64
7.2.3.16	gtg_popState	64
7.2.3.17	gtg_pushState	64
7.2.3.18	gtg_setCompress	64
7.2.3.19	gtg_setState	64
7.2.3.20	gtg_setTraceType	64
7.2.3.21	gtg_setVar	64
7.2.3.22	gtg_startLink	64
7.3	GTGBasic.h File Reference	64
7.3.1	Detailed Description	66
7.3.2	Macro Definition Documentation	66
7.3.2.1	GTG_FLAG_NONE	66
7.3.2.2	GTG_FLAG_NOTBUF	66

7.3.2.3	GTG_FLAG_OUTOFORDER	66
7.3.2.4	GTG_FLAG_USE_MPI	66
7.3.3	Typedef Documentation	66
7.3.3.1	gtg_flag_t	66
7.3.3.2	traceType_t	67
7.3.4	Enumeration Type Documentation	67
7.3.4.1	traceType	67
7.3.5	Function Documentation	67
7.3.5.1	AddComment	67
7.3.5.2	addVar	67
7.3.5.3	subVar	67
7.4	GTGColor.h File Reference	67
7.4.1	Detailed Description	69
7.4.2	Macro Definition Documentation	69
7.4.2.1	GTG_COLOR_BLUE_MASK	69
7.4.2.2	GTG_COLOR_BLUE_POS	69
7.4.2.3	GTG_COLOR_GREEN_MASK	69
7.4.2.4	GTG_COLOR_GREEN_POS	69
7.4.2.5	GTG_COLOR_RED_MASK	69
7.4.2.6	GTG_COLOR_RED_POS	69
7.4.3	Typedef Documentation	69
7.4.3.1	gtg_color_t	69
7.4.3.2	gtg_rgb_color_t	69
7.4.4	Function Documentation	69
7.4.4.1	gtg_color_create	69
7.4.4.2	gtg_color_exit	69
7.4.4.3	gtg_color_free	69
7.4.4.4	gtg_color_init	69
7.5	GTGCompress.h File Reference	69
7.5.1	Function Documentation	70
7.5.1.1	gtg_compress_f2f	70
7.5.1.2	gtg_compress_f2m	70
7.5.1.3	gtg_compress_init	70
7.5.1.4	gtg_compress_m2f	70
7.5.1.5	gtg_compress_m2m	70
7.5.1.6	gtg_decompress_f2f	70
7.5.1.7	gtg_decompress_f2m	70
7.5.1.8	gtg_decompress_init	70
7.5.1.9	gtg_decompress_m2f	70
7.5.1.10	gtg_decompress_m2m	70

7.6	GTGList.h File Reference	70
7.6.1	Macro Definition Documentation	71
7.6.1.1	GTG_LIST	71
7.6.1.2	gtg_list_entry	71
7.6.1.3	gtg_list_for_each	71
7.6.1.4	gtg_list_for_each_entry	71
7.6.1.5	gtg_list_for_each_entry_safe	72
7.6.1.6	gtg_list_for_each_reverse	72
7.6.1.7	gtg_list_for_each_safe	72
7.6.1.8	GTG_LIST_INIT	72
7.6.2	Typedef Documentation	73
7.6.2.1	gtg_list_t	73
7.6.3	Function Documentation	73
7.6.3.1	__gtg_list_add	73
7.6.3.2	__gtg_list_del	73
7.6.3.3	gtg_list_add	73
7.6.3.4	gtg_list_add_tail	73
7.6.3.5	gtg_list_del	73
7.6.3.6	gtg_list_size	73
7.7	GTGMemory.h File Reference	73
7.7.1	Detailed Description	74
7.7.2	Typedef Documentation	74
7.7.2.1	gtg_memory_t	74
7.8	GTGOTF.h File Reference	74
7.8.1	Detailed Description	74
7.9	GTGOTF2_Structs.h File Reference	75
7.9.1	Macro Definition Documentation	76
7.9.1.1	alloc_init_struct	76
7.9.1.2	alloc_State	76
7.9.1.3	alloc_struct	76
7.9.1.4	alloc_Variable	77
7.9.1.5	Container_NIL	77
7.9.1.6	ContainerType_NIL	77
7.9.1.7	EntityValue_NIL	77
7.9.1.8	EventType_NIL	77
7.9.1.9	free_struct	77
7.9.1.10	init_Container	77
7.9.1.11	init_ContainerType	77
7.9.1.12	init_EntityValue	78
7.9.1.13	init_EventType	78

7.9.1.14	init_LinkType	78
7.9.1.15	init_OTF2_String	78
7.9.1.16	init_State	78
7.9.1.17	init_StateType	79
7.9.1.18	init_Variable	79
7.9.1.19	init_VariableType	79
7.9.1.20	LinkType_NIL	79
7.9.1.21	MAX_PROCESS	79
7.9.1.22	State_NIL	79
7.9.1.23	StateType_NIL	79
7.9.1.24	Variable_NIL	79
7.9.1.25	VariableType_NIL	79
7.9.2	Typedef Documentation	79
7.9.2.1	Container_t	79
7.9.2.2	ContainerType_t	79
7.9.2.3	EntityValue_t	79
7.9.2.4	EventType_t	80
7.9.2.5	Link_t	80
7.9.2.6	LinkType_t	80
7.9.2.7	otf_color_t	80
7.9.2.8	State_t	80
7.9.2.9	StateType_t	80
7.9.2.10	uint32_t	80
7.9.2.11	Variable_t	80
7.9.2.12	VariableType_t	80
7.10	GTGOTF_Basic.h File Reference	80
7.10.1	Detailed Description	81
7.10.2	Function Documentation	82
7.10.2.1	OTFDefineContainer	82
7.11	GTGOTF_Structs.h File Reference	82
7.11.1	Detailed Description	83
7.11.2	Macro Definition Documentation	83
7.11.2.1	alloc_init_struct	83
7.11.2.2	alloc_State	84
7.11.2.3	alloc_struct	84
7.11.2.4	alloc_Variable	84
7.11.2.5	Container_NIL	84
7.11.2.6	ContainerType_NIL	84
7.11.2.7	EntityValue_NIL	84
7.11.2.8	EventType_NIL	84

7.11.2.9	free_struct	84
7.11.2.10	init_Container	85
7.11.2.11	init_ContainerType	85
7.11.2.12	init_EntityValue	85
7.11.2.13	init_EventType	85
7.11.2.14	init_LinkType	85
7.11.2.15	init_State	86
7.11.2.16	init_StateType	86
7.11.2.17	init_Variable	86
7.11.2.18	init_VariableType	86
7.11.2.19	LinkType_NIL	86
7.11.2.20	MAX_PROCESS	86
7.11.2.21	State_NIL	86
7.11.2.22	StateType_NIL	86
7.11.2.23	Variable_NIL	86
7.11.2.24	VariableType_NIL	86
7.11.3	Typedef Documentation	87
7.11.3.1	Container_t	87
7.11.3.2	ContainerType_t	87
7.11.3.3	EntityValue_t	87
7.11.3.4	EventType_t	87
7.11.3.5	Link_t	87
7.11.3.6	LinkType_t	87
7.11.3.7	off_color_t	87
7.11.3.8	State_t	87
7.11.3.9	StateType_t	87
7.11.3.10	Variable_t	87
7.11.3.11	VariableType_t	87
7.12	GTGPaje.h File Reference	87
7.12.1	Detailed Description	88
7.12.2	Typedef Documentation	88
7.12.2.1	paje_color_t	88
7.13	GTGPaje_Basic.h File Reference	88
7.13.1	Detailed Description	90
7.13.2	Macro Definition Documentation	90
7.13.2.1	FMT_PAJE	90
7.13.2.2	FMT_VITE	90
7.13.3	Typedef Documentation	90
7.13.3.1	gtg_paje_edp_t	90
7.13.3.2	gtg_paje_eventdef_t	90

7.13.4	Enumeration Type Documentation	90
7.13.4.1	gtg_paje_evtdef_e	91
7.13.4.2	gtg_paje_fieldtype_e	91
7.13.5	Function Documentation	91
7.13.5.1	pajeEventDefAddParam	91
7.13.6	Variable Documentation	91
7.13.6.1	paje_eventdefs	91
7.14	GTGReplay.h File Reference	91
7.14.1	Detailed Description	92
7.14.2	Enumeration Type Documentation	92
7.14.2.1	event_type_t	92
7.15	GTGStack.h File Reference	92
7.15.1	Macro Definition Documentation	93
7.15.1.1	GTG_STACK	93
7.15.1.2	gtg_stack_entry	93
7.15.1.3	GTG_STACK_INIT	93
7.15.2	Typedef Documentation	93
7.15.2.1	gtg_stack	93
7.15.2.2	gtg_stack_t	93
7.15.3	Function Documentation	93
7.15.3.1	gtg_stack_empty	93
7.15.3.2	gtg_stack_pop	93
7.15.3.3	gtg_stack_push	93
7.15.3.4	gtg_stack_top	93
7.16	GTGTypes.h File Reference	93
7.16.1	Typedef Documentation	94
7.16.1.1	trace_return_t	94

Chapter 1

The GTG library

(V)
(*~*)
(")(")

1.1 Presentation

The GTG library provides a low level library to generate traces in various formats (Paje, OTF).

The use of the library is simple, you just need to include the [GTG.h](#) header and then you can use the library as you wish.

Some simple examples are available in the test directory.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Trace type handler	9
To init the generated trace file(s)	11
Functions related to the containers	13
Functions related to the states	15
Functions related to the events	17
Functions related to links	18
Functions related to variables	20
Defined colors for GTG	22
Memory management	26
OTF interface in C of the traceGeneratorBasic API	27
Paje interface in C of the GTGBasic1 API	36
Functions for postponing event-processing function calls	46
Types used	47

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

__OTF2_String	49
Container	49
ContainerType	50
EntityValue	51
EventType	52
gtg_color	
This structure defines a color that can be used by GTG	53
gtg_list	53
gtg_memory	54
gtg_paje_edp_s	54
gtg_paje_eventdef_s	55
Link	55
LinkType	56
otf_color	57
State	57
StateType	58
Variable	59
VariableType	60

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

GTG.h	Generic header to include	61
GTG_EZTrace.h	61
GTGBasic.h	GTGBasic is a basic interface to generate trace in various formats	64
GTGColor.h	This file defines some useful colors to use in entity values for GTG	67
GTGCompress.h	69
GTGList.h	70
GTGMemory.h	This file defines a fast allocator for fixed-size blocks	73
GTGOTF.h	OTF is the global file for gtg interface using OTF	74
GTGOTF2_Structs.h	75
GTGOTF_Basic.h	OTF_GTGBasic1 is the OTF implementation of the basic interface to generate traces (GTG↔Basic1)	80
GTGOTF_Structs.h	OTF_Structs gives the global types and functions needed to have the OTF implementation	82
GTGPaje.h	PajeColor is a file that defines function that manipulate colors	87
GTGPaje_Basic.h	Paje_GTGBasic1 is the Paje implementation of the basic interface to generate traces (GTG↔Basic1)	88
GTGReplay.h	This file defines functions for postponing event-processing function calls	91
GTGStack.h	92
GTGTypes.h	93

Chapter 5

Module Documentation

5.1 Trace type handler

Enumerations

- enum `traceType` {
 PAJE, VITE, OTF, TAU,
 PAJE, VITE, OTF, TAU }
 The type of the output trace.

Functions

- void `setTraceType` (`traceType_t` type)
 Set the type of output trace.
- `traceType_t` `getTraceType` ()
 Get the type of the output trace.
- char * `getName` (int procRk)
 To get the name of the file to give to the addCont function for processors.
- int `bufferedModeActivated` ()
 Check whether the buffered-mode is activated.

5.1.1 Detailed Description

5.1.2 Enumeration Type Documentation

5.1.2.1 enum `traceType`

The type of the output trace.

Enumerator

- PAJE** Paje trace format.
- VITE** ViTE-specific trace format.
- OTF** OTF trace format.
- TAU** TAU Trace format.
- PAJE** Paje trace format.
- VITE** ViTE-specific trace format.
- OTF** OTF trace format.
- TAU** TAU Trace format.

5.1.3 Function Documentation

5.1.3.1 `int bufferedModeActivated ()`

Check wether the buffered-mode is activated.

Returns

1 is the buffered-mode is activate.
0 otherwise.

5.1.3.2 `traceType_t getName (int procRk)`

To get the name of the file to give to the addCont function for processors.

Parameters

<i>procRk</i>	Rank of the proc to get the file containing it
---------------	--

Returns

The name of the file to give for a proc

5.1.3.3 `traceType_t getTraceType ()`

Get the type of the output trace.

Returns

The type of the trace

5.1.3.4 `void setTraceType (traceType_t type)`

Set the type of output trace.

Parameters

<i>type</i>	Type of trace to generate
-------------	---------------------------

5.2 To init the generated trace file(s)

Functions

- `trace_return_t initTrace` (const char *filename, int rank, `gtg_flag_t` flags)
Initialize a trace.
- `trace_return_t endTrace` ()
Finalize a trace.
- `trace_return_t setCompress` (int val)
Enable trace compression (only available for OTF traces).
- `trace_return_t addEntityValue` (const char *alias, const char *entType, const char *name, `gtg_color_t` p_color)
Add an Entity Value.

5.2.1 Detailed Description

5.2.2 Function Documentation

5.2.2.1 `trace_return_t addEntityValue` (const char * alias, const char * entType, const char * name, `gtg_color_t` p_color)

Add an Entity Value.

Parameters

<i>alias</i>	Alias on the entity value
<i>entType</i>	Type of the entity that can have the value
<i>name</i>	Alternative name of the variable type
<i>p_color</i>	Color of the entity

Returns

TRACE_SUCCESS on success
An error code otherwise

5.2.2.2 `trace_return_t endTrace` ()

Finalize a trace.

Returns

TRACE_SUCCESS on success
An error code otherwise

5.2.2.3 `int initTrace` (const char * filename, int rank, `gtg_flag_t` flags)

Initialize a trace.

Parameters

<i>filename</i>	Root name of the file to create
<i>rank</i>	Process number of the file to create
<i>flags</i>	One of GTG_FLAG_NONE, GTG_FLAG_USE_MPI, GTG_FLAG_NOTBUF.

Returns

TRACE_SUCCESS on success
An error code otherwise

5.2.2.4 trace_return_t setCompress (int val)

Enable trace compression (only available for OTF traces).

Parameters

<i>val</i>	0 means no compression, otherwise the output files will be compressed
------------	---

Returns

TRACE_SUCCESS on success
An error code otherwise

5.3 Functions related to the containers

Functions

- [trace_return_t addContType](#) (const char *alias, const char *contType, const char *name)
Add a [Container](#) Type.
- [trace_return_t addContainer](#) (varPrec time, const char *alias, const char *type, const char *container, const char *name, const char *file)
Add a [Container](#).
- [trace_return_t destroyContainer](#) (varPrec time, const char *name, const char *type)
Destroy a [Container](#).
- [trace_return_t AddComment](#) (const char *comment)
Add some Comment in Trace file.

5.3.1 Detailed Description

5.3.2 Function Documentation

5.3.2.1 trace_return_t AddComment (const char * comment)

Add some Comment in Trace file.

Parameters

<i>comment</i>	Comment to be added
----------------	---------------------

Returns

TRACE_SUCCESS on success
An error code otherwise

5.3.2.2 trace_return_t addContainer (varPrec time, const char * alias, const char * type, const char * container, const char * name, const char * file)

Add a [Container](#).

Parameters

<i>time</i>	Time at which the container is added
<i>alias</i>	Alias of the new container
<i>type</i>	Type of the new container
<i>container</i>	Container parent
<i>name</i>	Alternative name of the variable type
<i>file</i>	File containing the container for vite format. Use "0" or "" chains for other formats.

Returns

TRACE_SUCCESS on success
An error code otherwise

5.3.2.3 trace_return_t addContType (const char * alias, const char * contType, const char * name)

Add a [Container](#) Type.

Parameters

<i>alias</i>	Alias on the container added
<i>contType</i>	Type of the parent container
<i>name</i>	Alternative name of the new container type

Returns

TRACE_SUCCESS on success
An error code otherwise

5.3.2.4 `trace_return_t destroyContainer (varPrec time, const char * name, const char * type)`

Destroy a [Container](#).

Parameters

<i>time</i>	Time at which the container is destroyed
<i>name</i>	Name of the container
<i>type</i>	Type of the container

Returns

TRACE_SUCCESS on success
An error code otherwise

5.4 Functions related to the states

Functions

- `trace_return_t addStateType` (const char *alias, const char *contType, const char *name)
Add a *State* Type.
- `trace_return_t setState` (varPrec time, const char *type, const char *cont, const char *val)
Set the *State* of a *Container*.
- `trace_return_t pushState` (varPrec time, const char *type, const char *cont, const char *val)
Save the current *State* on a stack and change the *State* of a *Container*.
- `trace_return_t popState` (varPrec time, const char *type, const char *cont)
Revert the *State* of a *Container* to its previous value.

5.4.1 Detailed Description

5.4.2 Function Documentation

5.4.2.1 `trace_return_t addStateType (const char * alias, const char * contType, const char * name)`

Add a *State* Type.

Parameters

<i>alias</i>	Alias on the state type added
<i>contType</i>	Type of container of these states
<i>name</i>	Alternative name of the state type

Returns

TRACE_SUCCESS on success
An error code otherwise

5.4.2.2 `trace_return_t popState (varPrec time, const char * type, const char * cont)`

Revert the *State* of a *Container* to its previous value.

Parameters

<i>time</i>	Time the state changes
<i>type</i>	Type of the state
<i>cont</i>	<i>Container</i> whose state changes

Returns

TRACE_SUCCESS on success
An error code otherwise

5.4.2.3 `trace_return_t pushState (varPrec time, const char * type, const char * cont, const char * val)`

Save the current *State* on a stack and change the *State* of a *Container*.

Parameters

<i>time</i>	Time the state changes
<i>type</i>	Type of the state
<i>cont</i>	Container whose state changes
<i>val</i>	Value of state of container

Returns

TRACE_SUCCESS on success
An error code otherwise

5.4.2.4 `trace_return_t setState (varPrec time, const char * type, const char * cont, const char * val)`

Set the [State](#) of a [Container](#).

Parameters

<i>time</i>	Time the state changes
<i>type</i>	Type of the state
<i>cont</i>	Container whose state changes
<i>val</i>	Value of new state of container

Returns

TRACE_SUCCESS on success
An error code otherwise

5.5 Functions related to the events

Functions

- [trace_return_t addEventType](#) (const char *alias, const char *contType, const char *name)
Add an Event Type.
- [trace_return_t addEvent](#) (varPrec time, const char *type, const char *cont, const char *val)
Add an Event.

5.5.1 Detailed Description

5.5.2 Function Documentation

5.5.2.1 [trace_return_t addEvent](#) (varPrec time, const char * type, const char * cont, const char * val)

Add an Event.

Parameters

<i>time</i>	Time the event happens
<i>type</i>	Type of the event
<i>cont</i>	Container that produced the event
<i>val</i>	Value of the new event

Returns

TRACE_SUCCESS on success
An error code otherwise

5.5.2.2 [trace_return_t addEventType](#) (const char * alias, const char * contType, const char * name)

Add an Event Type.

Parameters

<i>alias</i>	Alias on the event type
<i>contType</i>	Type of container of these events
<i>name</i>	Alternative name of the event type

Returns

TRACE_SUCCESS on success
An error code otherwise

5.6 Functions related to links

Functions

- [trace_return_t addLinkType](#) (const char *alias, const char *name, const char *contType, const char *srcContType, const char *destContType)
Add a [Link](#) Type.
- [trace_return_t startLink](#) (varPrec time, const char *type, const char *cont, const char *src, const char *dest, const char *val, const char *key)
Start a [Link](#).
- [trace_return_t endLink](#) (varPrec time, const char *type, const char *cont, const char *src, const char *dest, const char *val, const char *key)
End a [Link](#).

5.6.1 Detailed Description

5.6.2 Function Documentation

5.6.2.1 [trace_return_t addLinkType](#) (const char * *alias*, const char * *name*, const char * *contType*, const char * *srcContType*, const char * *destContType*)

Add a [Link](#) Type.

Parameters

<i>alias</i>	Alias on the link type
<i>name</i>	Alternative name of the link type
<i>contType</i>	Type of common ancestral container
<i>srcContType</i>	Type of the source container
<i>destContType</i>	Type of the destination container

Returns

TRACE_SUCCESS on success
An error code otherwise

5.6.2.2 [trace_return_t endLink](#) (varPrec *time*, const char * *type*, const char * *cont*, const char * *src*, const char * *dest*, const char * *val*, const char * *key*)

End a [Link](#).

Parameters

<i>time</i>	Time the link ends
<i>type</i>	Type of the link
<i>cont</i>	Container containing the link (an ancestor of source and destination container)
<i>src</i>	Source container
<i>dest</i>	Destination container
<i>val</i>	Value of the link
<i>key</i>	Key to match the start link

Returns

TRACE_SUCCESS on success
An error code otherwise

5.6.2.3 `trace_return_t startLink (varPrec time, const char * type, const char * cont, const char * src, const char * dest, const char * val, const char * key)`

Start a [Link](#).

Parameters

<i>time</i>	Time the link starts
<i>type</i>	Type of the link
<i>cont</i>	Container containing the link (an ancestor of source and destination container)
<i>src</i>	Source container
<i>dest</i>	Destination container
<i>val</i>	Value of the link
<i>key</i>	Key to match the end link

Returns

TRACE_SUCCESS on success
An error code otherwise

5.7 Functions related to variables

Functions

- `trace_return_t addVarType` (const char *alias, const char *name, const char *contType)
Add a [Variable](#) Type.
- `trace_return_t setVar` (varPrec time, const char *type, const char *cont, varPrec val)
Set a [Variable](#) value.
- `trace_return_t addVar` (varPrec time, const char *type, const char *cont, varPrec val)
Add a value to a [Variable](#).
- `trace_return_t subVar` (varPrec time, const char *type, const char *cont, varPrec val)
Subtract a value from a [Variable](#).

5.7.1 Detailed Description

5.7.2 Function Documentation

5.7.2.1 `trace_return_t addVar (varPrec time, const char * type, const char * cont, varPrec val)`

Add a value to a [Variable](#).

Parameters

<i>time</i>	Time the variable is incremented
<i>type</i>	Type of the variable
<i>cont</i>	Container containing the variable
<i>val</i>	Value added

Returns

TRACE_SUCCESS on success
An error code otherwise

5.7.2.2 `trace_return_t addVarType (const char * alias, const char * name, const char * contType)`

Add a [Variable](#) Type.

Parameters

<i>alias</i>	Alias on the variable type
<i>contType</i>	Type of container
<i>name</i>	Alternative name of the variable type

Returns

TRACE_SUCCESS on success
An error code otherwise

5.7.2.3 `trace_return_t setVar (varPrec time, const char * type, const char * cont, varPrec val)`

Set a [Variable](#) value.

Parameters

<i>time</i>	Time the variable changes
<i>type</i>	Type of the variable
<i>cont</i>	Container containing the variable
<i>val</i>	New value of the variable

Returns

TRACE_SUCCESS on success
An error code otherwise

5.7.2.4 `trace_return_t subVar (varPrec time, const char * type, const char * cont, varPrec val)`

Substract a value from a [Variable](#).

Parameters

<i>time</i>	Time the variable is incremented
<i>type</i>	Type of the variable
<i>cont</i>	Container containing the variable
<i>val</i>	Value substracted

Returns

TRACE_SUCCESS on success
An error code otherwise

5.8 Defined colors for GTG

Data Structures

- struct `gtg_color`

This structure defines a color that can be used by GTG.

Functions

- static `uint8_t GTG_COLOR_GET_BLUE (gtg_rgb_color_t rgb)`
Return the 1-byte value of the blue component of a rgb color.
- static `uint8_t GTG_COLOR_GET_GREEN (gtg_rgb_color_t rgb)`
Return the 1-byte value of the green component of a rgb color.
- static `uint8_t GTG_COLOR_GET_RED (gtg_rgb_color_t rgb)`
Return the 1-byte value of the red component of a rgb color.
- static `gtg_rgb_color_t GTG_COLOR_SET_COLOR (uint8_t r, uint8_t g, uint8_t b)`
Return the 4-bytes RGB color from 3 1-byte components.

Variables

- `gtg_color_t GTG_BLACK`
Default black color. (R,G,B) = (0, 0, 0)
- `gtg_color_t GTG_RED`
Default red color. (R,G,B) = (255, 0, 0)
- `gtg_color_t GTG_GREEN`
Default green color. (R,G,B) = (0, 255, 0)
- `gtg_color_t GTG_BLUE`
Default blue color. (R,G,B) = (0, 0, 255)
- `gtg_color_t GTG_WHITE`
Default white color. (R,G,B) = (255, 255, 255)
- `gtg_color_t GTG_TEAL`
Default teal color. (R,G,B) = (0, 255, 255)
- `gtg_color_t GTG_DARKGREY`
Default dark grey color. (R,G,B) = (85, 85, 85)
- `gtg_color_t GTG_YELLOW`
Default yellow color. (R,G,B) = (255, 255, 0)
- `gtg_color_t GTG_PURPLE`
Default purple color. (R,G,B) = (153, 25, 230)
- `gtg_color_t GTG_LIGHTBROWN`
Default light brown color. (R,G,B) = (170, 130, 130)
- `gtg_color_t GTG_LIGHTGREY`
Default light grey color. (R,G,B) = (200, 200, 200)
- `gtg_color_t GTG_DARKBLUE`
Default dark blue color. (R,G,B) = (0, 0, 80)
- `gtg_color_t GTG_PINK`
Default pink color. (R,G,B) = (255, 0, 255)
- `gtg_color_t GTG_DARKPINK`
Default dark pink color. (R,G,B) = (180, 80, 180)
- `gtg_color_t GTG_SEABLUE`
Default sea blue color. (R,G,B) = (25, 128, 200)

- `gtg_color_t GTG_KAKI`
Default kaki color. (R,G,B) = (80, 100, 25)
- `gtg_color_t GTG_REDBLOOD`
Default red blood color. (R,G,B) = (200, 25, 25)
- `gtg_color_t GTG_BROWN`
Default brown color. (R,G,B) = (100, 25, 25)
- `gtg_color_t GTG_GRENAT`
Default grenat color. (R,G,B) = (100, 0, 80)
- `gtg_color_t GTG_ORANGE`
Default orange color. (R,G,B) = (255, 160, 0)
- `gtg_color_t GTG_MAUVE`
Default mauve color. (R,G,B) = (128, 0, 255)
- `gtg_color_t GTG_LIGHTPINK`
Default light pink color. (R,G,B) = (255, 128, 255)

5.8.1 Detailed Description

5.8.2 Function Documentation

5.8.2.1 GTG_COLOR_GET_BLUE (`gtg_rgb_color_t rgb`) [inline],[static]

Return the 1-byte value of the blue component of a rgb color.

5.8.2.2 GTG_COLOR_GET_GREEN (`gtg_rgb_color_t rgb`) [inline],[static]

Return the 1-byte value of the green component of a rgb color.

5.8.2.3 GTG_COLOR_GET_RED (`gtg_rgb_color_t rgb`) [inline],[static]

Return the 1-byte value of the red component of a rgb color.

5.8.2.4 GTG_COLOR_SET_COLOR (`uint8_t r, uint8_t g, uint8_t b`) [inline],[static]

Return the 4-bytes RGB color from 3 1-byte components.

5.8.3 Variable Documentation

5.8.3.1 GTG_BLACK

Default black color. (R,G,B) = (0, 0, 0)

5.8.3.2 GTG_BLUE

Default blue color. (R,G,B) = (0, 0, 255)

5.8.3.3 GTG_BROWN

Default brown color. (R,G,B) = (100, 25, 25)

5.8.3.4 GTG_DARKBLUE

Default dark blue color. (R,G,B) = (0, 0, 80)

5.8.3.5 GTG_DARKGREY

Default dark grey color. (R,G,B) = (85, 85, 85)

5.8.3.6 GTG_DARKPINK

Default dark pink color. (R,G,B) = (180, 80, 180)

5.8.3.7 GTG_GREEN

Default green color. (R,G,B) = (0, 255, 0)

5.8.3.8 GTG_GRENAT

Default grenat color. (R,G,B) = (100, 0, 80)

5.8.3.9 GTG_KAKI

Default kaki color. (R,G,B) = (80, 100, 25)

5.8.3.10 GTG_LIGHTBROWN

Default light brown color. (R,G,B) = (170, 130, 130)

5.8.3.11 GTG_LIGHTGREY

Default light grey color. (R,G,B) = (200, 200, 200)

5.8.3.12 GTG_LIGHTPINK

Default light pink color. (R,G,B) = (255, 128, 255)

5.8.3.13 GTG_MAUVE

Default mauve color. (R,G,B) = (128, 0, 255)

5.8.3.14 GTG_ORANGE

Default orange color. (R,G,B) = (255, 160, 0)

5.8.3.15 GTG_PINK

Default pink color. (R,G,B) = (255, 0, 255)

5.8.3.16 GTG_PURPLE

Default purple color. (R,G,B) = (153, 25, 230)

5.8.3.17 GTG_RED

Default red color. (R,G,B) = (255, 0, 0)

5.8.3.18 GTG_REDBLOOD

Default red blood color. (R,G,B) = (200, 25, 25)

5.8.3.19 GTG_SEABLUE

Default sea blue color. (R,G,B) = (25, 128, 200)

5.8.3.20 GTG_TEAL

Default teal color. (R,G,B) = (0, 255, 255)

5.8.3.21 GTG_WHITE

Default white color. (R,G,B) = (255, 255, 255)

5.8.3.22 GTG_YELLOW

Default yellow color. (R,G,B) = (255, 255, 0)

5.9 Memory management

Functions

- void `gtg_block_memory_init` (`gtg_memory_t` *memory, `size_t` block_size, long initial_block_number)
Initialize the allocator.
- void * `gtg_block_malloc` (`gtg_memory_t` memory)
Allocate a block of data.
- void `gtg_block_free` (`gtg_memory_t` memory, void *ptr)
Free a block of data.

5.9.1 Detailed Description

5.9.2 Function Documentation

5.9.2.1 void `gtg_block_free` (`gtg_memory_t` memory, void * ptr)

Free a block of data.

Parameters

<i>memory</i>	The memory describer
<i>ptr</i>	The block of data to free

5.9.2.2 void * `gtg_block_malloc` (`gtg_memory_t` memory)

Allocate a block of data.

Parameters

<i>memory</i>	The memory describer
---------------	----------------------

Returns

A pointer to a block or NULL if allocation failed

5.9.2.3 void `gtg_block_memory_init` (`gtg_memory_t` * memory, `size_t` block_size, long initial_block_number)

Initialize the allocator.

Parameters

<i>memory</i>	A memory describer
<i>block_size</i>	The block size to be allocated when malloc is called
<i>initial_block_↔ number</i>	The number of blocks to allocate initially

5.10 OTF interface in C of the traceGeneratorBasic API

Functions

- `const otf_color_t OTF_get_color (gtg_color_t color)`
Converts a GTG color into a OTF color.
- `trace_return_t OTFInitTrace (const char *filename, gtg_flag_t flags)`
Initialize an OTF trace.
- `trace_return_t OTFSetCompress (int val)`
Enable trace compression.
- `trace_return_t OTFAddContType (const char *alias, const char *contType, const char *name)`
Add a [Container](#) Type.
- `trace_return_t OTFAddStateType (const char *alias, const char *contType, const char *name)`
Add a [State](#) Type.
- `trace_return_t OTFAddEventType (const char *alias, const char *contType, const char *name)`
Add an [Event](#) Type.
- `trace_return_t OTFAddLinkType (const char *alias, const char *name, const char *contType, const char *srcContType, const char *destContType)`
Add a [Link](#) Type.
- `trace_return_t OTFAddVarType (const char *alias, const char *name, const char *contType)`
Add a [Variable](#) Type.
- `trace_return_t OTFAddEntityValue (const char *alias, const char *entType, const char *name, const otf_color_t color)`
Add an [Entity](#) Value.
- `trace_return_t OTFStartContainer (varPrec time, const char *alias, const char *type, const char *container, const char *name, const char *file)`
Start a [Container](#).
- `trace_return_t OTFDestroyContainer (varPrec time, const char *name, const char *type)`
Destroy a [Container](#).
- `trace_return_t OTFSetState (varPrec time, const char *type, const char *cont, const char *val)`
Set the [State](#) of a [Container](#).
- `trace_return_t OTFPushState (varPrec time, const char *type, const char *cont, const char *val)`
Save the current [State](#) on a stack and change the [State](#) of a [Container](#).
- `trace_return_t OTFPopState (varPrec time, const char *type, const char *cont)`
Revert the [State](#) of a [Container](#) to its previous value.
- `trace_return_t OTFAddEvent (varPrec time, const char *type, const char *cont, const char *val)`
Add an [Event](#).
- `trace_return_t OTFStartLink (varPrec time, const char *type, const char *src, const char *dest, const char *val, const char *key)`
Start a [Link](#).
- `trace_return_t OTFEndLink (varPrec time, const char *type, const char *src, const char *dest, const char *val, const char *key)`
End a [Link](#).
- `trace_return_t OTFSetVar (varPrec time, const char *type, const char *cont, varPrec val)`
Set a [Variable](#) value.
- `trace_return_t OTFAddVar (varPrec time, const char *type, const char *cont, varPrec val)`
Add a value to a [Variable](#).
- `trace_return_t OTFSubVar (varPrec time, const char *type, const char *cont, varPrec val)`
Subtract a value from a [Variable](#).
- `trace_return_t OTFAddComment (const char *comment)`
Add some [Comment](#) in [Trace](#) file.
- `trace_return_t OTFEndTrace ()`
Finalize an OTF trace.

5.10.1 Detailed Description

5.10.2 Function Documentation

5.10.2.1 `const char * OTF_get_color (gtg_color_t color)`

Converts a GTG color into a OTF color.

Parameters

<i>color</i>	GTG color to convert
--------------	----------------------

Returns

The OTF color

5.10.2.2 `trace_return_t OTFAddComment (const char * comment)`

Add some Comment in Trace file.

Parameters

<i>comment</i>	Comment to be added
----------------	---------------------

Returns

TRACE_SUCCESS on success
An error code otherwise

5.10.2.3 `trace_return_t OTFAddContType (const char * alias, const char * contType, const char * name)`

Add a [Container](#) Type.

Parameters

<i>alias</i>	Alias on the container
<i>contType</i>	Type of container
<i>name</i>	Name of the container type

Returns

0 if success
An error code otherwise

5.10.2.4 `trace_return_t OTFAddEntityValue (const char * alias, const char * entType, const char * name, const otf_color_t color)`

Add an Entity Value.

Parameters

<i>alias</i>	Alias on the entity value
<i>entType</i>	Type of the entity

<i>name</i>	Name of the variable type
<i>color</i>	Color of the entity

Returns

0 if success
An error code otherwise

5.10.2.5 `trace_return_t OTFAddEvent (varPrec time, const char * type, const char * cont, const char * val)`

Add an Event.

Parameters

<i>time</i>	Time at which the event happens
<i>type</i>	Type of the event
<i>cont</i>	Container in this event
<i>val</i>	Entity value of the event of the container

Returns

0 if success
An error code otherwise

5.10.2.6 `trace_return_t OTFAddEventType (const char * alias, const char * contType, const char * name)`

Add an Event Type.

Parameters

<i>alias</i>	Alias on the event type
<i>contType</i>	Type of container
<i>name</i>	Name of the event type

Returns

0 if success
An error code otherwise

5.10.2.7 `trace_return_t OTFAddLinkType (const char * alias, const char * name, const char * contType, const char * srcContType, const char * destContType)`

Add a [Link](#) Type.

Parameters

<i>alias</i>	Alias on the link type
<i>name</i>	Name of the link type
<i>contType</i>	Type of container
<i>srcContType</i>	Type of the source container
<i>destContType</i>	Type of the destination container

Returns

0 if success
An error code otherwise

5.10.2.8 `trace_return_t` OTFAddStateType (`const char *` *alias*, `const char *` *contType*, `const char *` *name*)

Add a [State](#) Type.

Parameters

<i>alias</i>	Alias on the state type
<i>contType</i>	Type of container
<i>name</i>	Name of the state type

Returns

0 if success
An error code otherwise

5.10.2.9 `trace_return_t OTFAddVar (varPrec time, const char * type, const char * cont, varPrec val)`

Add a value to a [Variable](#).

Parameters

<i>time</i>	Time at which the variable is incremented
<i>type</i>	Type of the variable
<i>cont</i>	Container containing the variable
<i>val</i>	Value added

Returns

0 if success
An error code otherwise

5.10.2.10 `trace_return_t OTFAddVarType (const char * alias, const char * contType, const char * name)`

Add a [Variable](#) Type.

Parameters

<i>alias</i>	Alias on the variable type
<i>contType</i>	Type of container
<i>name</i>	Name of the variable type

Returns

0 if success
An error code otherwise

5.10.2.11 `trace_return_t OTFDestroyContainer (varPrec time, const char * name, const char * type)`

Destroy a [Container](#).

Parameters

<i>time</i>	Time at which the container is destroyed
<i>name</i>	Name of the container
<i>type</i>	Type of the container

Returns

0 if success
An error code otherwise

5.10.2.12 `trace_return_t` OTFEndLink (`varPrec` *time*, `const char *` *type*, `const char *` *cont*, `const char *` *dest*, `const char *` *val*, `const char *` *key*)

End a [Link](#).

Parameters

<i>time</i>	Time at which the link ends
<i>type</i>	Type of the link
<i>cont</i>	Container containing the link
<i>dest</i>	Container destination
<i>val</i>	Entity value of the link
<i>key</i>	Key to identify the link

Returns

0 if success
An error code otherwise

5.10.2.13 OTFEndTrace ()

Finalize an OTF trace.

Returns

0 if success
An error code otherwise

5.10.2.14 trace_return_t OTFInitTrace (const char * filename, gtg_flag_t flags)

Initialize an OTF trace.

Parameters

<i>filename</i>	Root name of the file to create
<i>flags</i>	One of GTG_FLAG_NONE, GTG_FLAG_USE_MPI, GTG_FLAG_NOTBUF.

Returns

0 if success An error code otherwise

5.10.2.15 trace_return_t OTFPopState (varPrec time, const char * type, const char * cont)

Revert the [State](#) of a [Container](#) to its previous value.

Parameters

<i>time</i>	Time at which the state is popped
<i>type</i>	Type of the state
<i>cont</i>	Container in this state

Returns

0 if success
An error code otherwise

5.10.2.16 trace_return_t OTFPushState (varPrec time, const char * type, const char * cont, const char * val)

Save the current [State](#) on a stack and change the [State](#) of a [Container](#).

Parameters

<i>time</i>	Time at which the state is pushed
<i>type</i>	Type of the state
<i>cont</i>	Container in this state
<i>val</i>	Entity value of the state of the container

Returns

0 if success
An error code otherwise

5.10.2.17 `trace_return_t OTFSetCompress (int val)`

Enable trace compression.

Parameters

<i>val</i>	0 means no compression, otherwise the output files will be compressed.
------------	--

Returns

0 if success
An error code otherwise

5.10.2.18 `trace_return_t OTFSetState (varPrec time, const char * type, const char * cont, const char * val)`

Set the [State](#) of a [Container](#).

Parameters

<i>time</i>	Time at which the state is set
<i>type</i>	Type of the state
<i>cont</i>	Container in this state
<i>val</i>	Entity value of the state of the container

Returns

0 if success
An error code otherwise

5.10.2.19 `trace_return_t OTFSetVar (varPrec time, const char * type, const char * cont, varPrec val)`

Set a [Variable](#) value.

Parameters

<i>time</i>	Time at which the variable is set
<i>type</i>	Type of the variable
<i>cont</i>	Container containing the variable
<i>val</i>	Value of the variable

Returns

0 if success
An error code otherwise

5.10.2.20 `trace_return_t OTFStartContainer (varPrec time, const char * alias, const char * type, const char * container,
const char * name, const char * file)`

Start a [Container](#).

Parameters

<i>time</i>	Time at which the container is added
<i>alias</i>	Alias of the new container
<i>type</i>	Type of the container
<i>container</i>	Container parent
<i>name</i>	Name of the variable type
<i>file</i>	File containing the container trace

Returns

0 if success
An error code otherwise

5.10.2.21 `trace_return_t OTFStartLink (varPrec time, const char * type, const char * cont, const char * src, const char * val, const char * key)`

Start a [Link](#).

Parameters

<i>time</i>	Time at which the link starts
<i>type</i>	Type of the link
<i>cont</i>	Container containing the link
<i>src</i>	Container source
<i>val</i>	Entity value of the link
<i>key</i>	Key to identify the link

Returns

0 if success
An error code otherwise

5.10.2.22 `trace_return_t OTFSubVar (varPrec time, const char * type, const char * cont, varPrec val)`

Subtract a value from a [Variable](#).

Parameters

<i>time</i>	Time at which the variable is incremented
<i>type</i>	Type of the variable
<i>cont</i>	Container containing the variable
<i>val</i>	Value subtracted

Returns

0 if success
An error code otherwise

5.11 Paje interface in C of the GTGBasic1 API

Functions

- `paje_color_t Paje_get_color (gtg_color_t p_color)`
Converts a GTG color into a PAJE color.
- `trace_return_t pajeInitTrace (const char *filename, int rank, gtg_flag_t flags, int fmt)`
*Initialize a VITE trace (*.ept)*
- `char * pajeGetName (int rk)`
Function to get the name of the file containing all the data for the proc of rank rk.
- `trace_return_t pajeSetCompress (int val)`
Enable trace compression.
- `trace_return_t pajeAddContType (const char *alias, const char *contType, const char *name)`
Add a Container Type.
- `trace_return_t pajeAddStateType (const char *alias, const char *contType, const char *name)`
Add a State Type.
- `trace_return_t pajeAddEventType (const char *alias, const char *contType, const char *name)`
Add an Event Type.
- `trace_return_t pajeAddLinkType (const char *alias, const char *name, const char *contType, const char *srcContType, const char *destContType)`
Add a Link Type.
- `trace_return_t pajeAddVarType (const char *alias, const char *contType, const char *name)`
Add a Variable Type.
- `trace_return_t pajeAddEntityValue (const char *alias, const char *entType, const char *name, const char *color)`
Add an Entity Value.
- `trace_return_t pajeAddContainer (varPrec time, const char *alias, const char *type, const char *container, const char *name, const char *file)`
Add a Container (VITE format).
- `trace_return_t pajeSeqAddContainer (varPrec time, const char *alias, const char *type, const char *container, const char *name)`
Add a Container (PAJE format).
- `trace_return_t pajeDestroyContainer (varPrec time, const char *name, const char *type)`
Destroy a Container.
- `trace_return_t pajeSetState (varPrec time, const char *type, const char *cont, const char *val)`
Set the State of a Container.
- `trace_return_t pajePushState (varPrec time, const char *type, const char *cont, const char *val)`
Save the current State on a stack and change the State of a Container.
- `trace_return_t pajePopState (varPrec time, const char *type, const char *cont)`
Revert the State of a Container to its previous value.
- `trace_return_t pajeAddEvent (varPrec time, const char *type, const char *cont, const char *val)`
Add an Event.
- `trace_return_t pajeStartLink (varPrec time, const char *type, const char *cont, const char *src, const char *val, const char *key)`
Start a link.
- `trace_return_t pajeEndLink (varPrec time, const char *type, const char *cont, const char *dest, const char *val, const char *key)`
Start a link.
- `trace_return_t pajeSetVar (varPrec time, const char *type, const char *cont, varPrec val)`
Set a Variable value.
- `trace_return_t pajeAddVar (varPrec time, const char *type, const char *cont, varPrec val)`

Add a value to a [Variable](#).

- [trace_return_t pajeSubVar](#) ([varPrec](#) time, const char *type, const char *cont, [varPrec](#) val)

Subtract a value from a [Variable](#).

- [trace_return_t pajeAddComment](#) (const char *comment)

Add some Comment in Trace file.

- [trace_return_t pajeEndTrace](#) ()

Finalize a PAJE trace.

- [trace_return_t viteEndTrace](#) ()

Finalize a VITE trace.

5.11.1 Detailed Description

5.11.2 Function Documentation

5.11.2.1 `const paje_color_t Paje_get_color (gtg_color_t color)`

Converts a GTG color into a PAJE color.

Parameters

<i>color</i>	GTG color to convert
--------------	----------------------

Returns

The PAJE color

5.11.2.2 `trace_return_t pajeAddComment (const char * comment)`

Add some Comment in Trace file.

Parameters

<i>comment</i>	Comment to be added
----------------	---------------------

Returns

TRACE_SUCCESS on success
An error code otherwise

5.11.2.3 `trace_return_t pajeAddContainer (varPrec time, const char * alias, const char * type, const char * container, const char * name, const char * file)`

Add a [Container](#) (VITE format).

Parameters

<i>time</i>	Time at which the container is added
<i>alias</i>	Alias on the new container
<i>type</i>	Type of the container
<i>container</i>	Container parent
<i>name</i>	Name of the variable type

<i>file</i>	File containing the container trace
-------------	-------------------------------------

Returns

0 if success
An error code otherwise

5.11.2.4 `trace_return_t` `pajeAddContType` (`const char * alias`, `const char * contType`, `const char * name`)

Add a [Container](#) Type.

Parameters

<i>alias</i>	Alias on the container
<i>contType</i>	Type of container
<i>name</i>	Name of the container type

Returns

0 if success
An error code otherwise

5.11.2.5 `trace_return_t` `pajeAddEntityValue` (`const char * alias`, `const char * entType`, `const char * name`, `const char * color`)

Add an Entity Value.

Parameters

<i>alias</i>	Alias on the entity value
<i>entType</i>	Type of the entity
<i>name</i>	Name of the variable type
<i>color</i>	Color of the entity

Returns

0 if success
An error code otherwise

5.11.2.6 `trace_return_t` `pajeAddEvent` (`varPrec time`, `const char * type`, `const char * cont`, `const char * val`)

Add an Event.

Parameters

<i>time</i>	Time at which the event happens
<i>type</i>	Type of the event
<i>cont</i>	Container in this event
<i>val</i>	Entity value of the event of the container

Returns

0 if success
An error code otherwise

5.11.2.7 `trace_return_t` `pajeAddEventType` (`const char *` *alias*, `const char *` *contType*, `const char *` *name*)

Add an Event Type.

Parameters

<i>alias</i>	Alias on the event type
<i>contType</i>	Type of container
<i>name</i>	Name of the event type

Returns

0 if success
An error code otherwise

5.11.2.8 `trace_return_t` `pajeAddLinkType (const char * alias, const char * name, const char * contType, const char * srcContType, const char * destContType)`

Add a [Link](#) Type.

Parameters

<i>alias</i>	Alias on the link type
<i>name</i>	Name of the link type
<i>contType</i>	Type of container
<i>srcContType</i>	Type of the source container
<i>destContType</i>	Type of the destination container

Returns

0 if success
An error code otherwise

5.11.2.9 `trace_return_t` `pajeAddStateType (const char * alias, const char * contType, const char * name)`

Add a [State](#) Type.

Parameters

<i>alias</i>	Alias on the state type
<i>contType</i>	Type of container
<i>name</i>	Name of the state type

Returns

0 if success
An error code otherwise

5.11.2.10 `trace_return_t` `pajeAddVar (varPrec time, const char * type, const char * cont, varPrec val)`

Add a value to a [Variable](#).

Parameters

<i>time</i>	Time at which the variable is incremented
<i>type</i>	Type of the variable

<i>cont</i>	Container containing the variable
<i>val</i>	Value added

Returns

0 if success
An error code otherwise

5.11.2.11 `trace_return_t pajeAddVarType (const char * alias, const char * contType, const char * name)`

Add a Variable Type.

Parameters

<i>alias</i>	Alias on the variable type
<i>contType</i>	Type of container
<i>name</i>	Name of the variable type

Returns

0 if success
An error code otherwise

5.11.2.12 `trace_return_t pajeDestroyContainer (varPrec time, const char * name, const char * type)`

Destroy a Container.

Parameters

<i>time</i>	Time at which the container is destroyed
<i>name</i>	Name on the container to destroy
<i>type</i>	Type of the container

Returns

0 if success
An error code otherwise

5.11.2.13 `trace_return_t pajeEndLink (varPrec time, const char * type, const char * cont, const char * dest, const char * val, const char * key)`

Start a link.

Parameters

<i>time</i>	Time at which the link starts
<i>type</i>	Type of the link
<i>cont</i>	Container parent of the source and destination containers containing the link
<i>dest</i>	Source container
<i>val</i>	Value of the link
<i>key</i>	Key used to match start link with end link

Returns

0 if success
An error code otherwise

5.11.2.14 `pajeEndTrace ()`

Finalize a PAJE trace.

Returns

0 if success
An error code otherwise

5.11.2.15 `char * pajeGetName (int rk)`

Function to get the name of the file containing all the data for the proc of rank rk.

Parameters

<i>rk</i>	Rank of the proc you want the filename containing it
-----------	--

Returns

Name of the file.

5.11.2.16 `trace_return_t pajeInitTrace (const char * filename, int rank, gtg_flag_t flags, int fmt)`

Initialize a VITE trace (*.ept)

Parameters

<i>filename</i>	Root name of the file to create
<i>rank</i>	Rank of the processor
<i>flags</i>	One of GTG_FLAG_NONE, GTG_FLAG_USE_MPI, GTG_FLAG_NOTBUF.
<i>fmt</i>	Format, paje or vite

Returns

0 if success An error code otherwise

5.11.2.17 `trace_return_t pajePopState (varPrec time, const char * type, const char * cont)`

Revert the [State](#) of a [Container](#) to its previous value.

Parameters

<i>time</i>	Time at which the state is popped
<i>type</i>	Type of the state
<i>cont</i>	Container in this state

Returns

0 if success
An error code otherwise

5.11.2.18 `trace_return_t pajePushState (varPrec time, const char * type, const char * cont, const char * val)`

Save the current [State](#) on a stack and change the [State](#) of a [Container](#).

Parameters

<i>time</i>	Time at which the state is pushed
<i>type</i>	Type of the state
<i>cont</i>	Container in this state
<i>val</i>	Entity value of the state of the container

Returns

0 if success
An error code otherwise

5.11.2.19 `trace_return_t pajeSeqAddContainer (varPrec time, const char * alias, const char * type, const char * container, const char * name)`

Add a [Container](#) (PAJE format).

Parameters

<i>time</i>	Time at which the container is added
<i>alias</i>	Alias on the new container
<i>type</i>	Type of the container
<i>container</i>	Container parent
<i>name</i>	Name of the variable type

Returns

0 if success
An error code otherwise

5.11.2.20 `trace_return_t pajeSetCompress (int val)`

Enable trace compression.

Parameters

<i>val</i>	0 means no compression, otherwise the output files will be compressed.
------------	--

Returns

0 if success
An error code otherwise

5.11.2.21 `trace_return_t pajeSetState (varPrec time, const char * type, const char * cont, const char * val)`

Set the [State](#) of a [Container](#).

Parameters

<i>time</i>	Time at which the state is set
<i>type</i>	Type of the state
<i>cont</i>	Container in this state
<i>val</i>	Entity value of the state of the container

Returns

0 if success
An error code otherwise

5.11.2.22 `trace_return_t` `pajeSetVar (varPrec time, const char * type, const char * cont, varPrec val)`

Set a [Variable](#) value.

Parameters

<i>time</i>	Time at which the variable is set
<i>type</i>	Type of the variable
<i>cont</i>	Container containing the variable
<i>val</i>	Value of the variable

Returns

0 if success
An error code otherwise

5.11.2.23 `trace_return_t pajeStartLink (varPrec time, const char * type, const char * cont, const char * src, const char * val, const char * key)`

Start a link.

Parameters

<i>time</i>	Time at which the link starts
<i>type</i>	Type of the link
<i>cont</i>	Container parent of the source and destination containers containing the link
<i>src</i>	Source container
<i>val</i>	Value of the link
<i>key</i>	Key used to match start link with end link

Returns

0 if success
An error code otherwise

5.11.2.24 `trace_return_t pajeSubVar (varPrec time, const char * type, const char * cont, varPrec val)`

Subtract a value from a [Variable](#).

Parameters

<i>time</i>	Time at which the variable is incremented
<i>type</i>	Type of the variable
<i>cont</i>	Container containing the variable
<i>val</i>	Value subtracted

Returns

0 if success
An error code otherwise

5.11.2.25 `viteEndTrace ()`

Finalize a VITE trace.

Returns

0 if success
An error code otherwise

5.12 Functions for postponing event-processing function calls

Functions

- void `gtg_record` (enum `event_type_t` type, `varPrec` time,...)
postpone the recording of an event
- void `gtg_write_events` (long nb_events_to_write)
run the first nb_events_to_write events

5.12.1 Detailed Description

5.12.2 Function Documentation

5.12.2.1 void `gtg_record` (enum `event_type_t` type, `varPrec` time, ...)

postpone the recording of an event

Parameters

<i>type</i>	The type of function to postpone
<i>time</i>	The time at which the event happens

5.12.2.2 void `gtg_write_events` (long nb_events_to_write)

run the first nb_events_to_write events

Parameters

<i>nb_events_to_write</i>	The number of functions to process (-1 for all functions)
---------------------------	---

5.13 Types used

Typedefs

- typedef double `varPrec`

Use the double precision type for time and value.

Enumerations

- enum `trace_return_t` {
 `TRACE_SUCCESS`, `TRACE_ERR_OPEN`, `TRACE_ERR_CLOSE`, `TRACE_ERR_WRITE`,
 `TRACE_ERR_NOT_IMPL` }

Define various return values.

5.13.1 Detailed Description

5.13.2 Typedef Documentation

5.13.2.1 typedef double `varPrec`

Use the double precision type for time and value.

5.13.3 Enumeration Type Documentation

5.13.3.1 enum `trace_return_t`

Define various return values.

Enumerator

`TRACE_SUCCESS` Success of the call.

`TRACE_ERR_OPEN` Failed to open files to write.

`TRACE_ERR_CLOSE` Failed to close file.

`TRACE_ERR_WRITE` Failed to write trace.

`TRACE_ERR_NOT_IMPL` Function not impleteneted.

Chapter 6

Data Structure Documentation

6.1 __OTF2_String Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- [uint32_t id](#)
- [char * name](#)

6.1.1 Field Documentation

6.1.1.1 [uint32_t __OTF2_String::id](#)

6.1.1.2 [char* __OTF2_String::name](#)

The documentation for this struct was generated from the following file:

- [GTGOTF2_Structs.h](#)

6.2 Container Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- [struct __OTF2_String name](#)
- [struct __OTF2_String alias](#)
- [ContainerType_t * ctType](#)
- [otf2_id_t id](#)
- [struct Container * parent](#)
- [struct gtg_list token](#)
- [State_t state_stack](#)
- [OTF2_EvtWriter * evt_writer](#)
- [char * name](#)
- [char * alias](#)
- [int ctType](#)
- [int id](#)

6.2.1 Detailed Description

Containers

6.2.2 Field Documentation

6.2.2.1 `char* Container::alias`

6.2.2.2 `struct __OTF2_String Container::alias`

6.2.2.3 `int Container::ctType`

6.2.2.4 `ContainerType_t* Container::ctType`

6.2.2.5 `OTF2_EvtWriter* Container::evt_writer`

6.2.2.6 `int Container::id`

6.2.2.7 `otf2_id_t Container::id`

6.2.2.8 `char* Container::name`

6.2.2.9 `struct __OTF2_String Container::name`

6.2.2.10 `struct Container* Container::parent`

6.2.2.11 `State_t Container::state_stack`

6.2.2.12 `struct gtg_list Container::token`

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF2_Structs.h](#)

6.3 ContainerType Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- `struct __OTF2_String name`
- `struct __OTF2_String alias`
- `otf2_id_t id`
- `struct ContainerType * parent_type`
- `struct gtg_list token`
- `char * name`
- `char * alias`
- `int id`

6.3.1 Field Documentation

6.3.1.1 `char* ContainerType::alias`

6.3.1.2 `struct __OTF2_String ContainerType::alias`

6.3.1.3 `int ContainerType::id`

6.3.1.4 `otf2_id_t ContainerType::id`

6.3.1.5 `char* ContainerType::name`

6.3.1.6 `struct __OTF2_String ContainerType::name`

6.3.1.7 `struct ContainerType* ContainerType::parent_type`

6.3.1.8 `struct gtg_list ContainerType::token`

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF2_Structs.h](#)

6.4 EntityValue Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- `struct __OTF2_String name`
- `struct __OTF2_String alias`
- `int groupId`
- `otf2_id_t id`
- `struct gtg_list token`
- `char * name`
- `char * alias`
- `int id`

6.4.1 Detailed Description

[EntityValue](#), contains the name of the functions/states

6.4.2 Field Documentation

6.4.2.1 `char* EntityValue::alias`

6.4.2.2 `struct __OTF2_String EntityValue::alias`

6.4.2.3 `int EntityValue::groupId`

6.4.2.4 `int EntityValue::id`

6.4.2.5 `otf2_id_t EntityValue::id`

6.4.2.6 `char* EntityValue::name`

6.4.2.7 `struct __OTF2_String EntityValue::name`

6.4.2.8 `struct gtg_list EntityValue::token`

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF_Structs.h](#)

6.5 EventType Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- `struct __OTF2_String name`
- `struct __OTF2_String alias`
- `ContainerType_t * contType`
- `otf2_id_t id`
- `struct gtg_list token`
- `char * name`
- `char * alias`
- `int contType`
- `int id`

6.5.1 Detailed Description

Events/Markers

6.5.2 Field Documentation

6.5.2.1 `char* EventType::alias`

6.5.2.2 `struct __OTF2_String EventType::alias`

6.5.2.3 `int EventType::contType`

6.5.2.4 `ContainerType_t* EventType::contType`

6.5.2.5 `int EventType::id`

6.5.2.6 `otf2_id_t EventType::id`

6.5.2.7 `char* EventType::name`

6.5.2.8 `struct __OTF2_String EventType::name`

6.5.2.9 struct gtg_list EventType::token

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF_Structs.h](#)

6.6 gtg_color Struct Reference

This structure defines a color that can be used by GTG.

```
#include <GTGColor.h>
```

Data Fields

- char * [color_name](#)
- [gtg_rgb_color_t](#) rgb

6.6.1 Detailed Description

This structure defines a color that can be used by GTG.

6.6.2 Field Documentation

6.6.2.1 char* gtg_color::color_name

The name of the color (ie. "RED" or "Black",...)

6.6.2.2 gtg_rgb_color_t gtg_color::rgb

RGB code of the color. It should be obtained by calling GTG_COLOR_SET_COLOR(r, g, b).

The documentation for this struct was generated from the following file:

- [GTGColor.h](#)

6.7 gtg_list Struct Reference

```
#include <GTGList.h>
```

Data Fields

- struct [gtg_list](#) * [prev](#)
- struct [gtg_list](#) * [next](#)

6.7.1 Field Documentation

6.7.1.1 struct gtg_list* gtg_list::next

6.7.1.2 struct `gtg_list`* `gtg_list::prev`

The documentation for this struct was generated from the following file:

- [GTGList.h](#)

6.8 `gtg_memory` Struct Reference

```
#include <GTGMemory.h>
```

Data Fields

- void * [first_mem](#)
- void * [current_mem](#)
- size_t [block_len](#)
- long [mem_len](#)
- void * [first_free](#)
- long [first_new](#)
- long [nb_allocated](#)

6.8.1 Field Documentation

6.8.1.1 size_t `gtg_memory::block_len`

6.8.1.2 void* `gtg_memory::current_mem`

6.8.1.3 void* `gtg_memory::first_free`

6.8.1.4 void* `gtg_memory::first_mem`

6.8.1.5 long `gtg_memory::first_new`

6.8.1.6 long `gtg_memory::mem_len`

6.8.1.7 long `gtg_memory::nb_allocated`

The documentation for this struct was generated from the following file:

- [GTGMemory.h](#)

6.9 `gtg_paje_edp_s` Struct Reference

```
#include <GTGPaje_Basic.h>
```

Data Fields

- struct `gtg_paje_edp_s` * [next](#)
- char * [name](#)
- enum [gtg_paje_fieldtype_e](#) [type](#)

6.9.1 Field Documentation

6.9.1.1 `char* gtg_paje_edp_s::name`

6.9.1.2 `struct gtg_paje_edp_s* gtg_paje_edp_s::next`

6.9.1.3 `enum gtg_paje_fieldtype_e gtg_paje_edp_s::type`

The documentation for this struct was generated from the following file:

- [GTGPaje_Basic.h](#)

6.10 gtg_paje_eventdef_s Struct Reference

```
#include <GTGPaje_Basic.h>
```

Data Fields

- `char * name`
- `int id`
- `gtg_paje_edp_t * first`
- `gtg_paje_edp_t * last`

6.10.1 Field Documentation

6.10.1.1 `gtg_paje_edp_t* gtg_paje_eventdef_s::first`

6.10.1.2 `int gtg_paje_eventdef_s::id`

6.10.1.3 `gtg_paje_edp_t* gtg_paje_eventdef_s::last`

6.10.1.4 `char* gtg_paje_eventdef_s::name`

The documentation for this struct was generated from the following file:

- [GTGPaje_Basic.h](#)

6.11 Link Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- `varPrec time`
- `int src`

6.11.1 Field Documentation

6.11.1.1 `int Link::src`

6.11.1.2 varPrec Link::time

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF2_Structs.h](#)

6.12 LinkType Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- [struct __OTF2_String](#) name
- [struct __OTF2_String](#) alias
- [ContainerType_t](#) * contType
- [ContainerType_t](#) * srcType
- [ContainerType_t](#) * destType
- [otf2_id_t](#) id
- [struct gtg_list](#) token
- [char](#) * name
- [char](#) * alias
- [int](#) contType
- [int](#) srcType
- [int](#) destType
- [int](#) id

6.12.1 Detailed Description

Links/Messages

6.12.2 Field Documentation

6.12.2.1 [char*](#) LinkType::alias

6.12.2.2 [struct __OTF2_String](#) LinkType::alias

6.12.2.3 [int](#) LinkType::contType

6.12.2.4 [ContainerType_t*](#) LinkType::contType

6.12.2.5 [int](#) LinkType::destType

6.12.2.6 [ContainerType_t*](#) LinkType::destType

6.12.2.7 [int](#) LinkType::id

6.12.2.8 [otf2_id_t](#) LinkType::id

6.12.2.9 [char*](#) LinkType::name

6.12.2.10 [struct __OTF2_String](#) LinkType::name

6.12.2.11 int LinkType::srcType

6.12.2.12 ContainerType_t* LinkType::srcType

6.12.2.13 struct gtg_list LinkType::token

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF_Structs.h](#)

6.13 otf_color Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- char * [colorID](#)
- uint8_t [red](#)
- uint8_t [green](#)
- uint8_t [blue](#)

6.13.1 Field Documentation

6.13.1.1 uint8_t otf_color::blue

6.13.1.2 char * otf_color::colorID

6.13.1.3 uint8_t otf_color::green

6.13.1.4 uint8_t otf_color::red

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF_Structs.h](#)

6.14 State Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- [EntityValue_t](#) * [value](#)
- [Container_t](#) * [cont](#)
- [StateType_t](#) * [stateType](#)
- [gtg_stack](#) token
- int [value](#)
- int [cont](#)
- int [stateType](#)

6.14.1 Detailed Description

States

6.14.2 Field Documentation

6.14.2.1 `int State::cont`

6.14.2.2 `Container_t* State::cont`

6.14.2.3 `int State::stateType`

6.14.2.4 `StateType_t* State::stateType`

6.14.2.5 `gtg_stack State::token`

6.14.2.6 `int State::value`

6.14.2.7 `EntityValue_t* State::value`

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF_Structs.h](#)

6.15 StateType Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- struct [__OTF2_String](#) name
- struct [__OTF2_String](#) alias
- int [groupId](#)
- [otf2_id_t](#) id
- struct [gtg_list](#) token
- char * [name](#)
- char * [alias](#)
- int [id](#)

6.15.1 Detailed Description

StateTypes

6.15.2 Field Documentation

6.15.2.1 `struct __OTF2_String StateType::alias`

6.15.2.2 `char* StateType::alias`

6.15.2.3 `int StateType::groupId`

6.15.2.4 `int StateType::id`

6.15.2.5 `otf2_id_t StateType::id`

6.15.2.6 `char* StateType::name`

6.15.2.7 `struct __OTF2_String StateType::name`

6.15.2.8 `struct gtg_list StateType::token`

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF_Structs.h](#)

6.16 Variable Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- [Container_t](#) * `parent`
- [VariableType_t](#) * `type`
- [uint64_t](#) `value`
- [otf2_id_t](#) `id`
- [struct gtg_list](#) `token`
- [int](#) `parent`
- [int](#) `type`
- [int](#) `id`

6.16.1 Field Documentation

6.16.1.1 `int Variable::id`

6.16.1.2 `otf2_id_t Variable::id`

6.16.1.3 `int Variable::parent`

6.16.1.4 `Container_t* Variable::parent`

6.16.1.5 `struct gtg_list Variable::token`

6.16.1.6 `int Variable::type`

6.16.1.7 `VariableType_t* Variable::type`

6.16.1.8 `uint64_t Variable::value`

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF_Structs.h](#)

6.17 VariableType Struct Reference

```
#include <GTGOTF2_Structs.h>
```

Data Fields

- struct [__OTF2_String](#) name
- struct [__OTF2_String](#) alias
- [ContainerType_t](#) * contType
- [otf2_id_t](#) id
- struct [gtg_list](#) token
- char * [name](#)
- char * [alias](#)
- int [contType](#)
- int [id](#)

6.17.1 Detailed Description

Variables/Counters

6.17.2 Field Documentation

6.17.2.1 char* VariableType::alias

6.17.2.2 struct [__OTF2_String](#) VariableType::alias

6.17.2.3 int VariableType::contType

6.17.2.4 [ContainerType_t](#)* VariableType::contType

6.17.2.5 int VariableType::id

6.17.2.6 [otf2_id_t](#) VariableType::id

6.17.2.7 char* VariableType::name

6.17.2.8 struct [__OTF2_String](#) VariableType::name

6.17.2.9 struct [gtg_list](#) VariableType::token

The documentation for this struct was generated from the following files:

- [GTGOTF2_Structs.h](#)
- [GTGOTF_Structs.h](#)

Chapter 7

File Documentation

7.1 GTG.h File Reference

Generic header to include.

```
#include <stdint.h>
#include "GTGTypes.h"
#include "GTGColor.h"
#include "GTGBasic.h"
```

7.1.1 Detailed Description

Generic header to include.

Authors

Developpers are :

- Francois Rue - francois.rue@labri.fr
- Francois Trahay - francois.trahay@labri.fr
- Johnny Jazeix - jazeix@enseirb-matmeca.fr
- Kevin Coulomb - kevin.coulomb@gmail.com
- Mathieu Faverge - faverge@labri.fr
- Olivier Lagrasse - lagrasse@enseirb-matmeca.fr

7.2 GTG_EZTrace.h File Reference

```
#include <stdlib.h>
#include <string.h>
#include "GTGColor.h"
#include "GTGTypes.h"
```

Macros

- `#define GTG_FLAG_NONE 0`
No flag specified.
- `#define GTG_FLAG_USE_MPI 1`
Several MPI processes are currently using GTG.

- `#define GTG_FLAG_NOTBUF 2`
For writing the traces in a non-buffered mode.
- `#define GTG_FLAG_OUTOFORDER 4`
Allow the application to record events out of order.

Typedefs

- `typedef uint8_t gtag_flag_t`
Flags that can be specified to GTG.
- `typedef enum traceType traceType_t`

Enumerations

- `enum traceType {`
 PAJE, VITE, OTF, TAU,
 PAJE, VITE, OTF, TAU
The type of the output trace.

Functions

- `void gtag_setTraceType (traceType_t type)`
- `traceType_t gtag_getTraceType ()`
- `char * gtag_getName (int procRk)`
- `int gtag_bufferedModeActivated ()`
- `trace_return_t gtag_initTrace (const char *filename, int rank, gtag_flag_t flags)`
- `trace_return_t gtag_endTrace ()`
- `trace_return_t gtag_setCompress (int val)`
- `trace_return_t gtag_addContType (const char *alias, const char *contType, const char *name)`
- `trace_return_t gtag_addStateType (const char *alias, const char *contType, const char *name)`
- `trace_return_t gtag_addEventType (const char *alias, const char *contType, const char *name)`
- `trace_return_t gtag_addLinkType (const char *alias, const char *name, const char *contType, const char *srcContType, const char *destContType)`
- `trace_return_t gtag_addVarType (const char *alias, const char *name, const char *contType)`
- `trace_return_t gtag_addEntityValue (const char *alias, const char *entType, const char *name, gtag_color_t p_color)`
- `trace_return_t gtag_addContainer (varPrec time, const char *alias, const char *type, const char *container, const char *name, const char *file)`
- `trace_return_t gtag_destroyContainer (varPrec time, const char *name, const char *type)`
- `trace_return_t gtag_setState (varPrec time, const char *type, const char *cont, const char *val)`
- `trace_return_t gtag_pushState (varPrec time, const char *type, const char *cont, const char *val)`
- `trace_return_t gtag_popState (varPrec time, const char *type, const char *cont)`
- `trace_return_t gtag_addEvent (varPrec time, const char *type, const char *cont, const char *val)`
- `trace_return_t gtag_startLink (varPrec time, const char *type, const char *cont, const char *src, const char *dest, const char *val, const char *key)`
- `trace_return_t gtag_endLink (varPrec time, const char *type, const char *cont, const char *src, const char *dest, const char *val, const char *key)`
- `trace_return_t gtag_setVar (varPrec time, const char *type, const char *cont, varPrec val)`
- `trace_return_t addVar (varPrec time, const char *type, const char *cont, varPrec val)`
Add a value to a Variable.
- `trace_return_t subVar (varPrec time, const char *type, const char *cont, varPrec val)`
Subtract a value from a Variable.
- `trace_return_t AddComment (const char *comment)`
Add some Comment in Trace file.

7.2.1 Macro Definition Documentation

7.2.1.1 #define GTG_FLAG_NONE 0

No flag specified.

7.2.1.2 #define GTG_FLAG_NOTBUF 2

For writing the traces in a non-buffered mode.

7.2.1.3 #define GTG_FLAG_OUTOFORDER 4

Allow the application to record events out of order.

7.2.1.4 #define GTG_FLAG_USE_MPI 1

Several MPI processes are currently using GTG.

7.2.2 Typedef Documentation

7.2.2.1 typedef uint8_t gtg_flag_t

Flags that can be specified to GTG.

7.2.2.2 typedef enum traceType traceType_t

7.2.3 Function Documentation

7.2.3.1 `trace_return_t gtg_addContainer (varPrec time, const char * alias, const char * type, const char * container, const char * name, const char * file)`

7.2.3.2 `trace_return_t gtg_addContType (const char * alias, const char * contType, const char * name)`

7.2.3.3 `trace_return_t gtg_addEntityValue (const char * alias, const char * entType, const char * name, gtg_color_t p_color)`

7.2.3.4 `trace_return_t gtg_addEvent (varPrec time, const char * type, const char * cont, const char * val)`

7.2.3.5 `trace_return_t gtg_addEventType (const char * alias, const char * contType, const char * name)`

7.2.3.6 `trace_return_t gtg_addLinkType (const char * alias, const char * name, const char * contType, const char * srcContType, const char * destContType)`

7.2.3.7 `trace_return_t gtg_addStateType (const char * alias, const char * contType, const char * name)`

7.2.3.8 `trace_return_t gtg_addVarType (const char * alias, const char * name, const char * contType)`

7.2.3.9 `int gtg_bufferedModeActivated ()`

7.2.3.10 `trace_return_t gtg_destroyContainer (varPrec time, const char * name, const char * type)`

7.2.3.11 `trace_return_t gtg_endLink (varPrec time, const char * type, const char * cont, const char * src, const char * dest, const char * val, const char * key)`

- 7.2.3.12 `trace_return_t` `gtg_endTrace ()`
- 7.2.3.13 `char*` `gtg_getName (int procRk)`
- 7.2.3.14 `traceType_t` `gtg_getTraceType ()`
- 7.2.3.15 `trace_return_t` `gtg_initTrace (const char * filename, int rank, gtg_flag_t flags)`
- 7.2.3.16 `trace_return_t` `gtg_popState (varPrec time, const char * type, const char * cont)`
- 7.2.3.17 `trace_return_t` `gtg_pushState (varPrec time, const char * type, const char * cont, const char * val)`
- 7.2.3.18 `trace_return_t` `gtg_setCompress (int val)`
- 7.2.3.19 `trace_return_t` `gtg_setState (varPrec time, const char * type, const char * cont, const char * val)`
- 7.2.3.20 `void` `gtg_setTraceType (traceType_t type)`
- 7.2.3.21 `trace_return_t` `gtg_setVar (varPrec time, const char * type, const char * cont, varPrec val)`
- 7.2.3.22 `trace_return_t` `gtg_startLink (varPrec time, const char * type, const char * cont, const char * src, const char * dest, const char * val, const char * key)`

7.3 GTGBasic.h File Reference

GTGBasic is a basic interface to generate trace in various formats.

```
#include <stdlib.h>
#include <string.h>
#include "GTGColor.h"
#include "GTGTypes.h"
```

Macros

- `#define` `GTG_FLAG_NONE` 0
No flag specified.
- `#define` `GTG_FLAG_USE_MPI` 1
Several MPI processes are currently using GTG.
- `#define` `GTG_FLAG_NOTBUF` 2
For writing the traces in a non-buffered mode.
- `#define` `GTG_FLAG_OUTOFORDER` 4
Allow the application to record events out of order.

Typedefs

- `typedef` `uint8_t` `gtg_flag_t`
Flags that can be specified to GTG.
- `typedef` `enum` `traceType` `traceType_t`

Enumerations

- enum `traceType` {
`PAJE`, `VITE`, `OTF`, `TAU`,
`PAJE`, `VITE`, `OTF`, `TAU` }

Functions

- void `setTraceType` (`traceType_t` type)
Set the type of output trace.
- `traceType_t` `getTraceType` ()
Get the type of the output trace.
- char * `getName` (int procRk)
To get the name of the file to give to the `addCont` function for processors.
- int `bufferedModeActivated` ()
Check whether the buffered-mode is activated.
- `trace_return_t` `initTrace` (const char *filename, int rank, `gtg_flag_t` flags)
Initialize a trace.
- `trace_return_t` `endTrace` ()
Finalize a trace.
- `trace_return_t` `setCompress` (int val)
Enable trace compression (only available for OTF traces).
- `trace_return_t` `addContType` (const char *alias, const char *contType, const char *name)
Add a *Container* Type.
- `trace_return_t` `addStateType` (const char *alias, const char *contType, const char *name)
Add a *State* Type.
- `trace_return_t` `addEventType` (const char *alias, const char *contType, const char *name)
Add an *Event* Type.
- `trace_return_t` `addLinkType` (const char *alias, const char *name, const char *contType, const char *src↔
ContType, const char *destContType)
Add a *Link* Type.
- `trace_return_t` `addVarType` (const char *alias, const char *name, const char *contType)
Add a *Variable* Type.
- `trace_return_t` `addEntityValue` (const char *alias, const char *entType, const char *name, `gtg_color_t` p_↔
color)
Add an *Entity Value*.
- `trace_return_t` `addContainer` (`varPrec` time, const char *alias, const char *type, const char *container, const
char *name, const char *file)
Add a *Container*.
- `trace_return_t` `destroyContainer` (`varPrec` time, const char *name, const char *type)
Destroy a *Container*.
- `trace_return_t` `setState` (`varPrec` time, const char *type, const char *cont, const char *val)
Set the *State* of a *Container*.
- `trace_return_t` `pushState` (`varPrec` time, const char *type, const char *cont, const char *val)
Save the current *State* on a stack and change the *State* of a *Container*.
- `trace_return_t` `popState` (`varPrec` time, const char *type, const char *cont)
Revert the *State* of a *Container* to its previous value.
- `trace_return_t` `addEvent` (`varPrec` time, const char *type, const char *cont, const char *val)
Add an *Event*.
- `trace_return_t` `startLink` (`varPrec` time, const char *type, const char *cont, const char *src, const char *dest,
const char *val, const char *key)

Start a *Link*.

- `trace_return_t endLink` (`varPrec` time, const char *type, const char *cont, const char *src, const char *dest, const char *val, const char *key)

End a *Link*.

- `trace_return_t setVar` (`varPrec` time, const char *type, const char *cont, `varPrec` val)

Set a *Variable* value.

- `trace_return_t addVar` (`varPrec` time, const char *type, const char *cont, `varPrec` val)
- `trace_return_t subVar` (`varPrec` time, const char *type, const char *cont, `varPrec` val)
- `trace_return_t AddComment` (const char *comment)

7.3.1 Detailed Description

GTGBasic is a basic interface to generate trace in various formats.

Version

0.1

Authors

Developers are :

Francois Rue - `francois.rue@labri.fr`

Francois Trahay - `francois.trahay@labri.fr`

Johnny Jazeix - `jazeix@enseirb-matmeca.fr`

Kevin Coulomb - `kevin.coulomb@gmail.com`

Mathieu Faverge - `faverge@labri.fr`

Olivier Lagrasse - `lagrasse@enseirb-matmeca.fr`

It has been initiated in 2010 by *eztrace* and *ViTE* projects that both needs a good library to generate traces.

7.3.2 Macro Definition Documentation

7.3.2.1 `#define GTG_FLAG_NONE 0`

No flag specified.

7.3.2.2 `#define GTG_FLAG_NOTBUF 2`

For writing the traces in a non-buffered mode.

7.3.2.3 `#define GTG_FLAG_OUTOFORDER 4`

Allow the application to record events out of order.

7.3.2.4 `#define GTG_FLAG_USE_MPI 1`

Several MPI processes are currently using GTG.

7.3.3 Typedef Documentation

7.3.3.1 `typedef uint8_t gtg_flag_t`

Flags that can be specified to GTG.

7.3.3.2 typedef enum traceType traceType_t

7.3.4 Enumeration Type Documentation

7.3.4.1 enum traceType

Enumerator

- PAJE** Paje trace format.
- VITE** ViTE-specific trace format.
- OTF** OTF trace format.
- TAU** TAU Trace format.
- PAJE** Paje trace format.
- VITE** ViTE-specific trace format.
- OTF** OTF trace format.
- TAU** TAU Trace format.

7.3.5 Function Documentation

7.3.5.1 trace_return_t AddComment (const char * *comment*)7.3.5.2 trace_return_t addVar (varPrec *time*, const char * *type*, const char * *cont*, varPrec *val*)7.3.5.3 trace_return_t subVar (varPrec *time*, const char * *type*, const char * *cont*, varPrec *val*)

7.4 GTGColor.h File Reference

This file defines some useful colors to use in entity values for GTG.

```
#include <stdint.h>
```

Data Structures

- struct [gtg_color](#)

This structure defines a color that can be used by GTG.

Macros

- #define [GTG_COLOR_BLUE_POS](#) 0
- #define [GTG_COLOR_GREEN_POS](#) 8
- #define [GTG_COLOR_RED_POS](#) 16
- #define [GTG_COLOR_BLUE_MASK](#) (0x000000ff << GTG_COLOR_BLUE_POS)
- #define [GTG_COLOR_GREEN_MASK](#) (0x000000ff << GTG_COLOR_GREEN_POS)
- #define [GTG_COLOR_RED_MASK](#) (0x000000ff << GTG_COLOR_RED_POS)

Typedefs

- typedef [uint32_t](#) [gtg_rgb_color_t](#)
- typedef struct [gtg_color](#) * [gtg_color_t](#)

Functions

- static uint8_t [GTG_COLOR_GET_BLUE](#) ([gtg_rgb_color_t](#) rgb)
Return the 1-byte value of the blue component of a rgb color.
- static uint8_t [GTG_COLOR_GET_GREEN](#) ([gtg_rgb_color_t](#) rgb)
Return the 1-byte value of the green component of a rgb color.
- static uint8_t [GTG_COLOR_GET_RED](#) ([gtg_rgb_color_t](#) rgb)
Return the 1-byte value of the red component of a rgb color.
- static [gtg_rgb_color_t](#) [GTG_COLOR_SET_COLOR](#) (uint8_t r, uint8_t g, uint8_t b)
Return the 4-bytes RGB color from 3 1-byte components.
- void [gtg_color_init](#) ()
- void [gtg_color_exit](#) ()
- [gtg_color_t](#) [gtg_color_create](#) (const char *name, uint8_t r, uint8_t g, uint8_t b)
- void [gtg_color_free](#) ([gtg_color_t](#) color)

Variables

- [gtg_color_t](#) [GTG_BLACK](#)
Default black color. (R,G,B) = (0, 0, 0)
- [gtg_color_t](#) [GTG_RED](#)
Default red color. (R,G,B) = (255, 0, 0)
- [gtg_color_t](#) [GTG_GREEN](#)
Default green color. (R,G,B) = (0, 255, 0)
- [gtg_color_t](#) [GTG_BLUE](#)
Default blue color. (R,G,B) = (0, 0, 255)
- [gtg_color_t](#) [GTG_WHITE](#)
Default white color. (R,G,B) = (255, 255, 255)
- [gtg_color_t](#) [GTG_TEAL](#)
Default teal color. (R,G,B) = (0, 255, 255)
- [gtg_color_t](#) [GTG_DARKGREY](#)
Default dark grey color. (R,G,B) = (85, 85, 85)
- [gtg_color_t](#) [GTG_YELLOW](#)
Default yellow color. (R,G,B) = (255, 255, 0)
- [gtg_color_t](#) [GTG_PURPLE](#)
Default purple color. (R,G,B) = (153, 25, 230)
- [gtg_color_t](#) [GTG_LIGHTBROWN](#)
Default light brown color. (R,G,B) = (170, 130, 130)
- [gtg_color_t](#) [GTG_LIGHTGREY](#)
Default light grey color. (R,G,B) = (200, 200, 200)
- [gtg_color_t](#) [GTG_DARKBLUE](#)
Default dark blue color. (R,G,B) = (0, 0, 80)
- [gtg_color_t](#) [GTG_PINK](#)
Default pink color. (R,G,B) = (255, 0, 255)
- [gtg_color_t](#) [GTG_DARKPINK](#)
Default dark pink color. (R,G,B) = (180, 80, 180)
- [gtg_color_t](#) [GTG_SEABLUE](#)
Default sea blue color. (R,G,B) = (25, 128, 200)
- [gtg_color_t](#) [GTG_KAKI](#)
Default kaki color. (R,G,B) = (80, 100, 25)
- [gtg_color_t](#) [GTG_REDBLOOD](#)
Default red blood color. (R,G,B) = (200, 25, 25)

- [gtg_color_t GTG_BROWN](#)
Default brown color. $(R,G,B) = (100, 25, 25)$
- [gtg_color_t GTG_GRENAT](#)
Default grenat color. $(R,G,B) = (100, 0, 80)$
- [gtg_color_t GTG_ORANGE](#)
Default orange color. $(R,G,B) = (255, 160, 0)$
- [gtg_color_t GTG_MAUVE](#)
Default mauve color. $(R,G,B) = (128, 0, 255)$
- [gtg_color_t GTG_LIGHTPINK](#)
Default light pink color. $(R,G,B) = (255, 128, 255)$

7.4.1 Detailed Description

This file defines some useful colors to use in entity values for GTG.

Version

0.1

7.4.2 Macro Definition Documentation

7.4.2.1 `#define GTG_COLOR_BLUE_MASK (0x000000ff << GTG_COLOR_BLUE_POS)`

7.4.2.2 `#define GTG_COLOR_BLUE_POS 0`

7.4.2.3 `#define GTG_COLOR_GREEN_MASK (0x000000ff << GTG_COLOR_GREEN_POS)`

7.4.2.4 `#define GTG_COLOR_GREEN_POS 8`

7.4.2.5 `#define GTG_COLOR_RED_MASK (0x000000ff << GTG_COLOR_RED_POS)`

7.4.2.6 `#define GTG_COLOR_RED_POS 16`

7.4.3 Typedef Documentation

7.4.3.1 `typedef struct gtg_color* gtg_color_t`

7.4.3.2 `typedef uint32_t gtg_rgb_color_t`

7.4.4 Function Documentation

7.4.4.1 `gtg_color_t gtg_color_create (const char * name, uint8_t r, uint8_t g, uint8_t b)`

7.4.4.2 `void gtg_color_exit ()`

7.4.4.3 `void gtg_color_free (gtg_color_t color)`

7.4.4.4 `void gtg_color_init ()`

7.5 GTGCompress.h File Reference

```
#include <stdint.h>
#include <stdio.h>
#include <zlib.h>
```

Functions

- int [gtg_compress_m2m](#) (z_stream *z, void *in_buf, [uint32_t](#) len, void *out_buf, [uint32_t](#) out_max_len)
- int [gtg_compress_m2f](#) (z_stream *z, void *in_buf, [uint32_t](#) len, FILE *file_out)
- int [gtg_compress_f2m](#) (z_stream *z, FILE *file_in, void *out_buf, [uint32_t](#) out_max_len)
- int [gtg_compress_f2f](#) (z_stream *z, FILE *file_in, FILE *file_out)
- int [gtg_decompress_m2m](#) (z_stream *z, void *in_buf, [uint32_t](#) len, void *out_buf, [uint32_t](#) out_max_len)
- int [gtg_decompress_m2f](#) (z_stream *z, void *in_buf, [uint32_t](#) len, FILE *file_out)
- int [gtg_decompress_f2m](#) (z_stream *z, FILE *file_in, void *out_buf, [uint32_t](#) out_max_len)
- int [gtg_decompress_f2f](#) (z_stream *z, FILE *file_in, FILE *file_out)
- int [gtg_compress_init](#) (z_stream *z, int compression_ratio)
- int [gtg_decompress_init](#) (z_stream *z)

7.5.1 Function Documentation

7.5.1.1 int [gtg_compress_f2f](#) (z_stream * z, FILE * *file_in*, FILE * *file_out*)

7.5.1.2 int [gtg_compress_f2m](#) (z_stream * z, FILE * *file_in*, void * *out_buf*, [uint32_t](#) *out_max_len*)

7.5.1.3 int [gtg_compress_init](#) (z_stream * z, int *compression_ratio*)

7.5.1.4 int [gtg_compress_m2f](#) (z_stream * z, void * *in_buf*, [uint32_t](#) *len*, FILE * *file_out*)

7.5.1.5 int [gtg_compress_m2m](#) (z_stream * z, void * *in_buf*, [uint32_t](#) *len*, void * *out_buf*, [uint32_t](#) *out_max_len*)

7.5.1.6 int [gtg_decompress_f2f](#) (z_stream * z, FILE * *file_in*, FILE * *file_out*)

7.5.1.7 int [gtg_decompress_f2m](#) (z_stream * z, FILE * *file_in*, void * *out_buf*, [uint32_t](#) *out_max_len*)

7.5.1.8 int [gtg_decompress_init](#) (z_stream * z)

7.5.1.9 int [gtg_decompress_m2f](#) (z_stream * z, void * *in_buf*, [uint32_t](#) *len*, FILE * *file_out*)

7.5.1.10 int [gtg_decompress_m2m](#) (z_stream * z, void * *in_buf*, [uint32_t](#) *len*, void * *out_buf*, [uint32_t](#) *out_max_len*)

7.6 GTGList.h File Reference

Data Structures

- struct [gtg_list](#)

Macros

- #define [GTG_LIST_INIT](#)(ptr)
initialize a list.
- #define [GTG_LIST](#)(name)
declare and initialize a list.
- #define [gtg_list_entry](#)(ptr, type, member) ((type *)((char *)(ptr) - (char *)&((type *)0)->member)))
get the structure corresponding to a list entry
- #define [gtg_list_for_each](#)(pos, head) for (pos = (head)->next; pos != (head); pos = pos->next)

- #define `gtg_list_for_each_reverse`(pos, head) for (pos = (head)->prev; pos != (head); pos = pos->prev)
- #define `gtg_list_for_each_safe`(pos, n, head)
- #define `gtg_list_for_each_entry`(pos, head, member)
iterate over list of given type
- #define `gtg_list_for_each_entry_safe`(pos, n, head, member)
iterate over list of given type safe against removal of list entry

Typedefs

- typedef struct `gtg_list` * `gtg_list_t`

Functions

- static void `__gtg_list_add` (`gtg_list_t` lnew, `gtg_list_t` prev, `gtg_list_t` next)
- static void `gtg_list_add` (`gtg_list_t` lnew, `gtg_list_t` head)
Insert a new entry after the specified head.
- static void `gtg_list_add_tail` (`gtg_list_t` lnew, `gtg_list_t` head)
Insert a new entry before the specified head (ie. at the tail of the list).
- static void `__gtg_list_del` (`gtg_list_t` prev, `gtg_list_t` next)
- static void `gtg_list_del` (`gtg_list_t` entry)
delete an entry from its list and reinitialize it.
- static int `gtg_list_size` (`gtg_list_t` l)

7.6.1 Macro Definition Documentation

7.6.1.1 GTG_LIST(name)

Value:

```
struct gtg_list name; \
    GTG_LIST_INIT (&name)
```

declare and initialize a list.

Parameters

<i>name</i>	Name of the variable
-------------	----------------------

7.6.1.2 gtg_list_entry(ptr, type, member) ((type *)((char *) (ptr) - (char *)&((type *)0)->member)))

get the structure corresponding to a list entry

Parameters

<i>ptr</i>	pointer to the list entry (<code>gtg_list_t</code>)
<i>type</i>	the type of the struct this is embedded in.
<i>member</i>	the name of the struct <code>gtg_list</code> member within the struct.

7.6.1.3 #define gtg_list_for_each(pos, head) for (pos = (head)->next; pos != (head); pos = pos->next)

7.6.1.4 #define gtg_list_for_each_entry(pos, head, member)

Value:

```
for (pos = gtg_list_entry((head)->next, typeof(*pos), member);
    &pos->member != (head);
    pos = gtg_list_entry(pos->member.next, typeof(*pos), member)) \
    \
```

iterate over list of given type

[gtg_list_for_each_entry\(pos, head, member\)](#)

Parameters

<i>pos</i>	the type * to use as a loop counter.
<i>head</i>	the head for the list.
<i>member</i>	the name of the struct gtg_list member within the struct.

7.6.1.5 #define gtg_list_for_each_entry_safe(*pos*, *n*, *head*, *member*)

Value:

```
for (pos = gtg_list_entry((head)->next, typeof(*pos), member),
    n = gtg_list_entry(pos->member.next, typeof(*pos), member);
    &pos->member != (head);
    pos = n, n = gtg_list_entry(n->member.next, typeof(*n), member)) \
    \
```

iterate over list of given type safe against removal of list entry

[gtg_list_for_each_entry_safe\(pos, n, head, member\)](#)

Parameters

<i>pos</i>	the type * to use as a loop counter.
<i>n</i>	another type * to use as temporary storage
<i>head</i>	the head for the list.
<i>member</i>	the name of the struct gtg_list member within the struct.

7.6.1.6 #define gtg_list_for_each_reverse(*pos*, *head*) for (pos = (head)->prev; pos != (head); pos = pos->prev)

7.6.1.7 #define gtg_list_for_each_safe(*pos*, *n*, *head*)

Value:

```
for (pos = (head)->next, n = pos->next; pos != (head); \
    pos = n, n = pos->next)
```

7.6.1.8 GTG_LIST_INIT(*ptr*)

Value:

```
do {
    (ptr)->prev = (ptr);
    (ptr)->next = (ptr);
} while (0) \
\
```

initialize a list.

Parameters

<i>ptr</i>	pointer to the list (<code>gtg_list_t</code>).
------------	--

7.6.2 Typedef Documentation

7.6.2.1 typedef struct `gtg_list*` `gtg_list_t`

7.6.3 Function Documentation

7.6.3.1 static void `__gtg_list_add (gtg_list_t lnew, gtg_list_t prev, gtg_list_t next)` `[inline]`, `[static]`

7.6.3.2 static void `__gtg_list_del (gtg_list_t prev, gtg_list_t next)` `[inline]`, `[static]`

Delete a list entry by making the prev/next entries point to each other.

This is only for internal list manipulation where we know the prev/next entries already!

7.6.3.3 void `gtg_list_add (gtg_list_t lnew, gtg_list_t head)` `[inline]`, `[static]`

Insert a new entry after the specified head.

Parameters

<i>lnew</i>	new entry to be added
<i>head</i>	list head to add it after

7.6.3.4 void `gtg_list_add_tail (gtg_list_t lnew, gtg_list_t head)` `[inline]`, `[static]`

Insert a new entry before the specified head (ie. at the tail of the list).

Parameters

<i>lnew</i>	new entry to be added
<i>head</i>	list head to add it after

7.6.3.5 void `gtg_list_del (gtg_list_t entry)` `[inline]`, `[static]`

delete an entry from its list and reinitialize it.

Parameters

<i>entry</i>	the element to delete from the list.
--------------	--------------------------------------

7.6.3.6 static int `gtg_list_size (gtg_list_t /)` `[inline]`, `[static]`

7.7 GTGMemory.h File Reference

This file defines a fast allocator for fixed-size blocks.

```
#include <stdlib.h>
```

Data Structures

- struct [gtg_memory](#)

Typedefs

- typedef struct [gtg_memory](#) * [gtg_memory_t](#)

Functions

- void [gtg_block_memory_init](#) ([gtg_memory_t](#) *memory, size_t block_size, long initial_block_number)
Initialize the allocator.
- void * [gtg_block_malloc](#) ([gtg_memory_t](#) memory)
Allocate a block of data.
- void [gtg_block_free](#) ([gtg_memory_t](#) memory, void *ptr)
Free a block of data.

7.7.1 Detailed Description

This file defines a fast allocator for fixed-size blocks.

Version

0.1

7.7.2 Typedef Documentation

7.7.2.1 typedef struct [gtg_memory](#)* [gtg_memory_t](#)

7.8 GTGOTF.h File Reference

OTF is the global file for gtg interface using OTF.

```
#include <stdint.h>
#include "GTGOTF_Structs.h"
#include "GTGOTF_Basic.h"
```

7.8.1 Detailed Description

OTF is the global file for gtg interface using OTF.

Version

0.1

Authors

Developers are :
Francois Rue - francois.rue@labri.fr
Francois Trahay - francois.trahay@labri.fr
Johnny Jazeix - jazeix@enseirb-matmeca.fr
Kevin Coulomb - kevin.coulomb@gmail.com
Mathieu Faverge - faverge@labri.fr
Olivier Lagrasse - lagrasse@enseirb-matmeca.fr

7.9 GTGOTF2_Structs.h File Reference

```
#include <stdint.h>
#include "GTGList.h"
#include "GTGStack.h"
```

Data Structures

- struct [__OTF2_String](#)
- struct [StateType](#)
- struct [State](#)
- struct [ContainerType](#)
- struct [Container](#)
- struct [EntityValue](#)
- struct [EventType](#)
- struct [LinkType](#)
- struct [Link](#)
- struct [VariableType](#)
- struct [Variable](#)
- struct [otf_color](#)

Macros

- #define [MAX_PROCESS](#) 64
- #define [ContainerType_NIL](#) 0
- #define [Container_NIL](#) 0
- #define [StateType_NIL](#) 0
- #define [State_NIL](#) 0
- #define [EntityValue_NIL](#) 0
- #define [EventType_NIL](#) 0
- #define [LinkType_NIL](#) 0
- #define [VariableType_NIL](#) 0
- #define [Variable_NIL](#) 0
- #define [init_OTF2_String](#)(var)
- #define [init_StateType](#)(var)
- #define [init_State](#)(var)
- #define [init_ContainerType](#)(var)
- #define [init_Container](#)(var)
- #define [init_EntityValue](#)(var)
- #define [init_EventType](#)(var)
- #define [init_LinkType](#)(var)
- #define [init_VariableType](#)(var)
- #define [init_Variable](#)(var)
- #define [alloc_struct](#)(ptr, type, list_head)
- #define [alloc_init_struct](#)(type, ptr, list_head, _name_, _alias_)
- #define [alloc_Variable](#)(_ptr_, _id_, _parent_, _type_, _value_)
- #define [alloc_State](#)(_ptr_, _value_, _cont_, _stateType_)
- #define [free_struct](#)(_type_, _list_head_)

Typedefs

- typedef `otf2_id_t` `uint32_t`
- typedef struct `StateType` `StateType_t`
- typedef struct `State` `State_t`
- typedef struct `ContainerType` `ContainerType_t`
- typedef struct `Container` `Container_t`
- typedef struct `EntityValue` `EntityValue_t`
- typedef struct `EventType` `EventType_t`
- typedef struct `LinkType` `LinkType_t`
- typedef struct `Link` `Link_t`
- typedef struct `VariableType` `VariableType_t`
- typedef struct `Variable` `Variable_t`
- typedef struct `otf_color` * `otf_color_t`

7.9.1 Macro Definition Documentation

7.9.1.1 #define alloc_init_struct(type, ptr, list_head, _name_, _alias_)

Value:

```
do {
    alloc_struct(ptr, type, list_head);
    (ptr)->name = (char *)malloc(sizeof(char)*(strlen(_name_)+1));
    strcpy((ptr)->name, _name_);
    (ptr)->alias = (char *)malloc(sizeof(char)*(strlen(_alias_)+1));
    strcpy((ptr)->alias, _alias_);
}while(0)
```

7.9.1.2 #define alloc_State(_ptr_, _value_, _cont_, _stateType_)

Value:

```
do {
    _ptr_ = (State_t*) malloc(sizeof(State_t));
    init_State(*(_ptr_));
    (_ptr_)->value = _value_;
    (_ptr_)->cont = _cont_;
    (_ptr_)->stateType = _stateType_;
}while(0)
```

7.9.1.3 #define alloc_struct(ptr, type, list_head)

Value:

```
do {
    ptr = (type*) malloc(sizeof(type));
    GTG_LIST_INIT(&(ptr->token));
    ptr->id = (gtg_list_entry)((list_head)->prev, type, token->id) + 1;
    gtg_list_add_tail(&(ptr->token), list_head);
}while(0)
```

7.9.1.4 #define alloc_Variable(_ptr_, _id_, _parent_, _type_, _value_)

Value:

```
do {
    (_ptr_) = (Variable_t*) malloc(sizeof(Variable_t));
    init_Variable(*(_ptr_));
    (_ptr_)->id = _id_;
    (_ptr_)->parent = _parent_;
    (_ptr_)->type = _type_;
    (_ptr_)->value = _value_;
}while(0)
```

7.9.1.5 #define Container_NIL 0

7.9.1.6 #define ContainerType_NIL 0

7.9.1.7 #define EntityValue_NIL 0

7.9.1.8 #define EventType_NIL 0

7.9.1.9 #define free_struct(_type_, _list_head_)

Value:

```
do{
    _type_ *ptr, *tmp;
    gtg_list_for_each_entry_safe(ptr, tmp, &(_list_head_.token, token) {
        gtg_list_del(&(ptr->token));
        free(ptr->name.name);
        free(ptr->alias);
        free(ptr);
    }
}while(0)
```

7.9.1.10 #define init_Container(var)

Value:

```
do {
    init_OTF2_String((var).name);
    init_OTF2_String((var).alias);
    (var).ctType = NULL;
    (var).id = Container_NIL;
    (var).parent = NULL;
    (var).ev_writer = NULL;
    GTG_LIST_INIT(&(var).token);
    GTG_STACK_INIT(&(var).state_stack.token);
}while(0)
```

7.9.1.11 #define init_ContainerType(var)

Value:

```
do {
    init_OTF2_String((var).name);
    init_OTF2_String((var).alias);
    (var).id = ContainerType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.9.1.12 #define init_EntityValue(var)

Value:

```
do {
    init_OTF2_String((var).name);
    init_OTF2_String((var).alias);
    (var).groupId = 0;
    (var).id      = EntityValue_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.9.1.13 #define init_EventType(var)

Value:

```
do {
    init_OTF2_String((var).name);
    init_OTF2_String((var).alias);
    (var).contType = NULL;
    (var).id       = EventType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.9.1.14 #define init_LinkType(var)

Value:

```
do {
    init_OTF2_String((var).name);
    init_OTF2_String((var).alias);
    (var).contType = NULL;
    (var).srcType  = NULL;
    (var).destType = NULL;
    (var).id       = LinkType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.9.1.15 #define init_OTF2_String(var)

Value:

```
do {
    (var).id  = -1;
    (var).name = NULL;
}while(0)
```

7.9.1.16 #define init_State(var)

Value:

```
do {
    (var).value      = EntityValue_NIL;
    (var).cont       = NULL;
    (var).stateType  = NULL;
    GTG_STACK_INIT(&(var).token);
}while(0)
```


7.9.1.17 #define init_StateType(var)

Value:

```
do {
    init_OTF2_String((var).name);
    init_OTF2_String((var).alias);
    (var).groupId = 0;
    (var).id      = StateType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.9.1.18 #define init_Variable(var)

Value:

```
do {
    (var).parent = NULL;
    (var).type   = NULL;
    (var).value  = 0;
    (var).id     = Variable_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.9.1.19 #define init_VariableType(var)

Value:

```
do {
    init_OTF2_String((var).name);
    init_OTF2_String((var).alias);
    (var).contType = NULL;
    (var).id       = VariableType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.9.1.20 #define LinkType_NIL 0

7.9.1.21 #define MAX_PROCESS 64

7.9.1.22 #define State_NIL 0

7.9.1.23 #define StateType_NIL 0

7.9.1.24 #define Variable_NIL 0

7.9.1.25 #define VariableType_NIL 0

7.9.2 Typedef Documentation

7.9.2.1 typedef struct Container Container_t

Containers

7.9.2.2 typedef struct ContainerType ContainerType_t

7.9.2.3 typedef struct EntityValue EntityValue_t

[EntityValue](#), contains the name of the functions/states

7.9.2.4 typedef struct EventType EventType_t

Events/Markers

7.9.2.5 typedef struct Link Link_t

7.9.2.6 typedef struct LinkType LinkType_t

Links/Messages

7.9.2.7 typedef struct otf_color* otf_color_t

7.9.2.8 typedef struct State State_t

States

7.9.2.9 typedef struct StateType StateType_t

StateTypes

7.9.2.10 typedef otf_id_t uint32_t

7.9.2.11 typedef struct Variable Variable_t

7.9.2.12 typedef struct VariableType VariableType_t

Variables/Counters

7.10 GTGOTF_Basic.h File Reference

OTF_GTGBasic1 is the OTF implementation of the basic interface to generate traces (GTGBasic1).

```
#include "GTGTypes.h"
#include "GTGBasic.h"
#include "GTGOTF_Structs.h"
```

Functions

- `const otf_color_t OTF_get_color (gtg_color_t color)`
Converts a GTG color into a OTF color.
- `trace_return_t OTFInitTrace (const char *filename, gtg_flag_t flags)`
Initialize an OTF trace.
- `trace_return_t OTFSetCompress (int val)`
Enable trace compression.
- `trace_return_t OTFAddContType (const char *alias, const char *contType, const char *name)`
*Add a *Container* Type.*
- `trace_return_t OTFAddStateType (const char *alias, const char *contType, const char *name)`
*Add a *State* Type.*
- `trace_return_t OTFAddEventType (const char *alias, const char *contType, const char *name)`
*Add an *Event* Type.*

- [trace_return_t OTFAddLinkType](#) (const char *alias, const char *name, const char *contType, const char *srcContType, const char *destContType)
Add a [Link](#) Type.
- [trace_return_t OTFAddVarType](#) (const char *alias, const char *name, const char *contType)
Add a [Variable](#) Type.
- [trace_return_t OTFAddEntityValue](#) (const char *alias, const char *entType, const char *name, const [otf_color_t](#) color)
Add an [Entity Value](#).
- [trace_return_t OTFDefineContainer](#) (const char *alias, const char *type, const char *container, const char *name, const char *file)
- [trace_return_t OTFStartContainer](#) ([varPrec](#) time, const char *alias, const char *type, const char *container, const char *name, const char *file)
Start a [Container](#).
- [trace_return_t OTFDestroyContainer](#) ([varPrec](#) time, const char *name, const char *type)
Destroy a [Container](#).
- [trace_return_t OTFSetState](#) ([varPrec](#) time, const char *type, const char *cont, const char *val)
Set the [State](#) of a [Container](#).
- [trace_return_t OTFPushState](#) ([varPrec](#) time, const char *type, const char *cont, const char *val)
Save the current [State](#) on a stack and change the [State](#) of a [Container](#).
- [trace_return_t OTFPopState](#) ([varPrec](#) time, const char *type, const char *cont)
Revert the [State](#) of a [Container](#) to its previous value.
- [trace_return_t OTFAddEvent](#) ([varPrec](#) time, const char *type, const char *cont, const char *val)
Add an [Event](#).
- [trace_return_t OTFStartLink](#) ([varPrec](#) time, const char *type, const char *src, const char *dest, const char *val, const char *key)
Start a [Link](#).
- [trace_return_t OTFEndLink](#) ([varPrec](#) time, const char *type, const char *src, const char *dest, const char *val, const char *key)
End a [Link](#).
- [trace_return_t OTFSetVar](#) ([varPrec](#) time, const char *type, const char *cont, [varPrec](#) val)
Set a [Variable](#) value.
- [trace_return_t OTFAddVar](#) ([varPrec](#) time, const char *type, const char *cont, [varPrec](#) val)
Add a value to a [Variable](#).
- [trace_return_t OTFSubVar](#) ([varPrec](#) time, const char *type, const char *cont, [varPrec](#) val)
Subtract a value from a [Variable](#).
- [trace_return_t OTFAddComment](#) (const char *comment)
Add some [Comment](#) in [Trace file](#).
- [trace_return_t OTFEndTrace](#) ()
Finalize an [OTF trace](#).

7.10.1 Detailed Description

OTF_GTGBasic1 is the OTF implementation of the basic interface to generate traces (GTGBasic1).

Version

0.1

Authors

Developers are :
 Francois Rue - francois.rue@labri.fr
 Francois Trahay - francois.trahay@labri.fr
 Johnny Jazeix - jazeix@enseirb-matmeca.fr
 Kevin Coulomb - kevin.coulomb@gmail.com
 Mathieu Faverge - faverge@labri.fr
 Olivier Lagrasse - lagrasse@enseirb-matmeca.fr

7.10.2 Function Documentation

7.10.2.1 `trace_return_t OTFDefineContainer (const char * alias, const char * type, const char * container, const char * name, const char * file)`

7.11 GTGOTF_Structs.h File Reference

OTF_Structs gives the global types and functions needed to have the OTF implementation.

```
#include <stdint.h>
#include "GTGList.h"
#include "GTGStack.h"
```

Data Structures

- struct [StateType](#)
- struct [State](#)
- struct [ContainerType](#)
- struct [Container](#)
- struct [EntityValue](#)
- struct [EventType](#)
- struct [LinkType](#)
- struct [Link](#)
- struct [VariableType](#)
- struct [Variable](#)
- struct [otf_color](#)

Macros

- `#define MAX_PROCESS 64`
- `#define ContainerType_NIL 0`
- `#define Container_NIL 0`
- `#define StateType_NIL 0`
- `#define State_NIL 0`
- `#define EntityValue_NIL 0`
- `#define EventType_NIL 0`
- `#define LinkType_NIL 0`
- `#define VariableType_NIL 0`
- `#define Variable_NIL 0`
- `#define init_ContainerType(var)`
- `#define init_Container(var)`
- `#define init_StateType(var)`

- `#define init_EntityValue(var)`
- `#define init_EventType(var)`
- `#define init_LinkType(var)`
- `#define init_VariableType(var)`
- `#define init_Variable(var)`
- `#define init_State(var)`
- `#define alloc_struct(ptr, type, list_head)`
- `#define alloc_init_struct(type, ptr, list_head, _name_, _alias_)`
- `#define alloc_Variable(_ptr_, _id_, _parent_, _type_, _value_)`
- `#define alloc_State(_ptr_, _value_, _cont_, _stateType_)`
- `#define free_struct(_type_, _list_head_)`

Typedefs

- `typedef struct StateType StateType_t`
- `typedef struct State State_t`
- `typedef struct ContainerType ContainerType_t`
- `typedef struct Container Container_t`
- `typedef struct EntityValue EntityValue_t`
- `typedef struct EventType EventType_t`
- `typedef struct LinkType LinkType_t`
- `typedef struct Link Link_t`
- `typedef struct VariableType VariableType_t`
- `typedef struct Variable Variable_t`
- `typedef struct otf_color * otf_color_t`

7.11.1 Detailed Description

OTF_Structs gives the global types and functions needed to have the OTF implementation.

Version

0.1

Authors

Developers are :

Francois Rue - francois.rue@labri.fr

Francois Trahay - francois.trahay@labri.fr

Johnny Jazeix - jazeix@enseirb-matmeca.fr

Kevin Coulomb - kevin.coulomb@gmail.com

Mathieu Faverge - faverge@labri.fr

Olivier Lagrasse - lagrasse@enseirb-matmeca.fr

7.11.2 Macro Definition Documentation

7.11.2.1 `#define alloc_init_struct(type, ptr, list_head, _name_, _alias_)`

Value:

```
do {
    alloc_struct(ptr, type, list_head);
    (ptr)->name = (char *)malloc(sizeof(char)*(strlen(_name_)+1));
    strcpy((ptr)->name, _name_);
    (ptr)->alias = (char *)malloc(sizeof(char)*(strlen(_alias_)+1));
    strcpy((ptr)->alias, _alias_);
}while(0)
```

7.11.2.2 #define alloc_State(_ptr_, _value_, _cont_, _stateType_)

Value:

```
do {
    _ptr_ = (State_t*) malloc(sizeof(State_t));
    \
    init_State(*(_ptr_));
    \
    (_ptr_)->value = _value_;
    \
    (_ptr_)->cont = _cont_;
    \
    (_ptr_)->stateType = _stateType_;
    \
} while (0)
```

7.11.2.3 #define alloc_struct(ptr, type, list_head)

Value:

```
do {
    ptr = (type*) malloc(sizeof(type));
    \
    GTG_LIST_INIT(&(ptr->token));
    \
    ptr->id = (gtg_list_entry)((list_head)->prev, type, token->id) + 1;
    \
    gtg_list_add_tail(&(ptr->token), list_head);
    \
} while (0)
```

7.11.2.4 #define alloc_Variable(_ptr_, _id_, _parent_, _type_, _value_)

Value:

```
do {
    (_ptr_) = (Variable_t*) malloc(sizeof(Variable_t));
    \
    init_Variable(*(_ptr_));
    \
    (_ptr_)->id = _id_;
    \
    (_ptr_)->parent = _parent_;
    \
    (_ptr_)->type = _type_;
    \
    (_ptr_)->value = _value_;
    \
} while (0)
```

7.11.2.5 #define Container_NIL 0

7.11.2.6 #define ContainerType_NIL 0

7.11.2.7 #define EntityValue_NIL 0

7.11.2.8 #define EventType_NIL 0

7.11.2.9 #define free_struct(_type_, _list_head_)

Value:

```
do {
    \
    _type_ *ptr, *tmp;
    \
    gtg_list_for_each_entry_safe(ptr, tmp, &(_list_head_).token, token) {
    \
    gtg_list_del(&(ptr->token));
    \
    free(ptr->name);
    \
    free(ptr->alias);
    \
    free(ptr);
    \
    }
    \
} while (0)
```

7.11.2.10 #define init_Container(var)

Value:

```
do {
    (var).name    = NULL;
    (var).alias   = NULL;
    (var).ctType  = ContainerType_NIL;
    (var).id      = Container_NIL;
    GTG_LIST_INIT(&(var).token);
    GTG_STACK_INIT(&(var).state_stack.token);
}while(0)
```

7.11.2.11 #define init_ContainerType(var)

Value:

```
do {
    (var).name    = NULL;
    (var).alias   = NULL;
    (var).id      = ContainerType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.11.2.12 #define init_EntityValue(var)

Value:

```
do {
    (var).name    = NULL;
    (var).alias   = NULL;
    (var).groupId = 0;
    (var).id      = EntityValue_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.11.2.13 #define init_EventType(var)

Value:

```
do {
    (var).name    = NULL;
    (var).alias   = NULL;
    (var).contType = ContainerType_NIL;
    (var).id      = EventType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.11.2.14 #define init_LinkType(var)

Value:

```
do {
    (var).name    = NULL;
    (var).alias   = NULL;
    (var).contType = ContainerType_NIL;
    (var).srcType  = ContainerType_NIL;
    (var).destType = ContainerType_NIL;
    (var).id      = LinkType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.11.2.15 #define init_State(var)

Value:

```
do {
    (var).value   = EntityValue_NIL;
    (var).cont    = Container_NIL;
    (var).stateType = StateType_NIL;
    GTG_STACK_INIT(&(var).token);
}while(0)
```

7.11.2.16 #define init_StateType(var)

Value:

```
do {
    (var).name     = NULL;
    (var).alias    = NULL;
    (var).groupId  = 0;
    (var).id       = StateType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.11.2.17 #define init_Variable(var)

Value:

```
do {
    (var).parent = Container_NIL;
    (var).parent = VariableType_NIL;
    (var).value  = 0;
    (var).id     = Variable_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.11.2.18 #define init_VariableType(var)

Value:

```
do {
    (var).name     = NULL;
    (var).alias    = NULL;
    (var).contType = ContainerType_NIL;
    (var).id       = VariableType_NIL;
    GTG_LIST_INIT(&(var).token);
}while(0)
```

7.11.2.19 #define LinkType_NIL 0

7.11.2.20 #define MAX_PROCESS 64

7.11.2.21 #define State_NIL 0

7.11.2.22 #define StateType_NIL 0

7.11.2.23 #define Variable_NIL 0

7.11.2.24 #define VariableType_NIL 0

7.11.3 Typedef Documentation

7.11.3.1 typedef struct Container Container_t

Containers

7.11.3.2 typedef struct ContainerType ContainerType_t

7.11.3.3 typedef struct EntityValue EntityValue_t

[EntityValue](#), contains the name of the functions/states

7.11.3.4 typedef struct EventType EventType_t

Events/Markers

7.11.3.5 typedef struct Link Link_t

7.11.3.6 typedef struct LinkType LinkType_t

Links/Messages

7.11.3.7 typedef struct otf_color* otf_color_t

7.11.3.8 typedef struct State State_t

States

7.11.3.9 typedef struct StateType StateType_t

StateTypes

7.11.3.10 typedef struct Variable Variable_t

7.11.3.11 typedef struct VariableType VariableType_t

Variables/Counters

7.12 GTGPaje.h File Reference

pajeColor is a file that defines function that manipulate colors.

```
#include "GTGPaje_Basic.h"
```

Typedefs

- typedef char * [paje_color_t](#)

Functions

- [paje_color_t Paje_get_color \(gtg_color_t p_color\)](#)
Converts a GTG color into a PAJE color.

7.12.1 Detailed Description

pajeColor is a file that defines function that manipulate colors.

Version

0.1

7.12.2 Typedef Documentation

7.12.2.1 typedef char* paje_color_t

7.13 GTGPaje_Basic.h File Reference

paje_GTGBasic1 is the Paje implementation of the basic interface to generate traces (GTGBasic1).

```
#include "GTGBasic.h"
```

Data Structures

- struct [gtg_paje_edp_s](#)
- struct [gtg_paje_eventdef_s](#)

Macros

- #define [FMT_PAJE](#) 0
Constant to create a paje trace.
- #define [FMT_VITE](#) 1
Constant to create a vite trace.

Typedefs

- typedef struct [gtg_paje_edp_s](#) [gtg_paje_edp_t](#)
- typedef struct [gtg_paje_eventdef_s](#) [gtg_paje_eventdef_t](#)

Enumerations

- enum [gtg_paje_evtdef_e](#) {
[GTG_PAJE_EVTDEF_DefineContainerType](#), [GTG_PAJE_EVTDEF_DefineStateType](#), [GTG_PAJE_EVTDEF_DefineEventType](#), [GTG_PAJE_EVTDEF_DefineVariableType](#),
[GTG_PAJE_EVTDEF_DefineLinkType](#), [GTG_PAJE_EVTDEF_DefineEntityValue](#), [GTG_PAJE_EVTDEF_CreateContainer](#), [GTG_PAJE_EVTDEF_DestroyContainer](#),
[GTG_PAJE_EVTDEF_SetState](#), [GTG_PAJE_EVTDEF_PushState](#), [GTG_PAJE_EVTDEF_PopState](#), [GTG_PAJE_EVTDEF_ResetState](#),
[GTG_PAJE_EVTDEF_NewEvent](#), [GTG_PAJE_EVTDEF_SetVariable](#), [GTG_PAJE_EVTDEF_AddVariable](#),
[GTG_PAJE_EVTDEF_SubVariable](#),
[GTG_PAJE_EVTDEF_StartLink](#), [GTG_PAJE_EVTDEF_EndLink](#), [GTG_PAJE_EVTDEF_NBR](#) }

- enum [gtg_paje_fielddtype_e](#) {
[GTG_PAJE_FIELDDTYPE_Int](#), [GTG_PAJE_FIELDDTYPE_Hex](#), [GTG_PAJE_FIELDDTYPE_Date](#), [GTG_PAJE_FIELDDTYPE_Double](#),
[GTG_PAJE_FIELDDTYPE_String](#), [GTG_PAJE_FIELDDTYPE_Color](#), [GTG_PAJE_FIELDDTYPE_NBR](#) }

Functions

- [trace_return_t pajeInitTrace](#) (const char *filename, int rank, [gtg_flag_t](#) flags, int fmt)
Initialize a VITE trace (*.ept)
- char * [pajeGetName](#) (int rk)
Function to get the name of the file containing all the data for the proc of rank rk.
- [trace_return_t pajeSetCompress](#) (int val)
Enable trace compression.
- [trace_return_t pajeAddContType](#) (const char *alias, const char *contType, const char *name)
Add a *Container* Type.
- [trace_return_t pajeAddStateType](#) (const char *alias, const char *contType, const char *name)
Add a *State* Type.
- [trace_return_t pajeAddEventType](#) (const char *alias, const char *contType, const char *name)
Add an *Event* Type.
- [trace_return_t pajeAddLinkType](#) (const char *alias, const char *name, const char *contType, const char *srcContType, const char *destContType)
Add a *Link* Type.
- [trace_return_t pajeAddVarType](#) (const char *alias, const char *contType, const char *name)
Add a *Variable* Type.
- [trace_return_t pajeAddEntityValue](#) (const char *alias, const char *entType, const char *name, const char *color)
Add an *Entity Value*.
- [trace_return_t pajeAddContainer](#) ([varPrec](#) time, const char *alias, const char *type, const char *container, const char *name, const char *file)
Add a *Container* (VITE format).
- [trace_return_t pajeSeqAddContainer](#) ([varPrec](#) time, const char *alias, const char *type, const char *container, const char *name)
Add a *Container* (PAJE format).
- [trace_return_t pajeDestroyContainer](#) ([varPrec](#) time, const char *name, const char *type)
Destroy a *Container*.
- [trace_return_t pajeSetState](#) ([varPrec](#) time, const char *type, const char *cont, const char *val)
Set the *State* of a *Container*.
- [trace_return_t pajePushState](#) ([varPrec](#) time, const char *type, const char *cont, const char *val)
Save the current *State* on a stack and change the *State* of a *Container*.
- [trace_return_t pajePopState](#) ([varPrec](#) time, const char *type, const char *cont)
Revert the *State* of a *Container* to its previous value.
- [trace_return_t pajeAddEvent](#) ([varPrec](#) time, const char *type, const char *cont, const char *val)
Add an *Event*.
- [trace_return_t pajeStartLink](#) ([varPrec](#) time, const char *type, const char *cont, const char *src, const char *val, const char *key)
Start a link.
- [trace_return_t pajeEndLink](#) ([varPrec](#) time, const char *type, const char *cont, const char *dest, const char *val, const char *key)
Start a link.
- [trace_return_t pajeSetVar](#) ([varPrec](#) time, const char *type, const char *cont, [varPrec](#) val)
Set a *Variable* value.
- [trace_return_t pajeAddVar](#) ([varPrec](#) time, const char *type, const char *cont, [varPrec](#) val)

- Add a value to a [Variable](#).*

 - [trace_return_t](#) [pajeSubVar](#) ([varPrec](#) time, const char *type, const char *cont, [varPrec](#) val)

Subtract a value from a [Variable](#).
- [trace_return_t](#) [pajeAddComment](#) (const char *comment)

Add some Comment in Trace file.
- [trace_return_t](#) [pajeEndTrace](#) ()

Finalize a PAJE trace.
- [trace_return_t](#) [viteEndTrace](#) ()

Finalize a VITE trace.
- void [pajeEventDefAddParam](#) (enum [gtg_paje_evtdef_e](#) event, const char *name, enum [gtg_paje_fieldtype_e](#) type)

Variables

- [gtg_paje_eventdef_t](#) [paje_eventdefs](#) [[GTG_PAJE_EVTDEF_NBR](#)]

7.13.1 Detailed Description

[paje_GTGBasic1](#) is the Paje implementation of the basic interface to generate traces ([GTGBasic1](#)).

Version

0.1

Authors

Developers are :
 Francois Rue - francois.rue@labri.fr
 Francois Trahay - francois.trahay@labri.fr
 Johnny Jazeix - jazeix@enseirb-matmeca.fr
 Kevin Coulomb - kevin.coulomb@gmail.com
 Mathieu Faverge - faverge@labri.fr
 Olivier Lagrasse - lagrasse@enseirb-matmeca.fr

7.13.2 Macro Definition Documentation

7.13.2.1 #define FMT_PAJE 0

Constant to create a paje trace.

7.13.2.2 #define FMT_VITE 1

Constant to create a vite trace.

7.13.3 Typedef Documentation

7.13.3.1 typedef struct [gtg_paje_edp_s](#) [gtg_paje_edp_t](#)

7.13.3.2 typedef struct [gtg_paje_eventdef_s](#) [gtg_paje_eventdef_t](#)

7.13.4 Enumeration Type Documentation

7.13.4.1 enum `gtg_paje_evtdef_e`

Enumerator

GTG_PAJE_EVTDEF_DefineContainerType
GTG_PAJE_EVTDEF_DefineStateType
GTG_PAJE_EVTDEF_DefineEventType
GTG_PAJE_EVTDEF_DefineVariableType
GTG_PAJE_EVTDEF_DefineLinkType
GTG_PAJE_EVTDEF_DefineEntityValue
GTG_PAJE_EVTDEF_CreateContainer
GTG_PAJE_EVTDEF_DestroyContainer
GTG_PAJE_EVTDEF_SetState
GTG_PAJE_EVTDEF_PushState
GTG_PAJE_EVTDEF_PopState
GTG_PAJE_EVTDEF_ResetState
GTG_PAJE_EVTDEF_NewEvent
GTG_PAJE_EVTDEF_SetVariable
GTG_PAJE_EVTDEF_AddVariable
GTG_PAJE_EVTDEF_SubVariable
GTG_PAJE_EVTDEF_StartLink
GTG_PAJE_EVTDEF_EndLink
GTG_PAJE_EVTDEF_NBR

7.13.4.2 enum `gtg_paje_fieldtype_e`

Enumerator

GTG_PAJE_FIELDTYPE_Int
GTG_PAJE_FIELDTYPE_Hex
GTG_PAJE_FIELDTYPE_Date
GTG_PAJE_FIELDTYPE_Double
GTG_PAJE_FIELDTYPE_String
GTG_PAJE_FIELDTYPE_Color
GTG_PAJE_FIELDTYPE_NBR

7.13.5 Function Documentation

7.13.5.1 void `pajeEventDefAddParam` (enum `gtg_paje_evtdef_e` *event*, const char * *name*, enum `gtg_paje_fieldtype_e` *type*)

7.13.6 Variable Documentation

7.13.6.1 `gtg_paje_eventdef_t` `paje_eventdefs`[`GTG_PAJE_EVTDEF_NBR`]

7.14 GTGReplay.h File Reference

This file defines functions for postponing event-processing function calls.

Enumerations

- enum [event_type_t](#) {
[event_addContainer](#), [event_destroyContainer](#), [event_setState](#), [event_pushState](#),
[event_popState](#), [event_addEvent](#), [event_startLink](#), [event_endLink](#),
[event_setVar](#), [event_addVar](#), [event_subVar](#) }

Functions

- void [gtg_record](#) (enum [event_type_t](#) type, [varPrec](#) time,...)
postpone the recording of an event
- void [gtg_write_events](#) (long nb_events_to_write)
run the first nb_events_to_write events

7.14.1 Detailed Description

This file defines functions for postponing event-processing function calls.

Version

0.1

7.14.2 Enumeration Type Documentation

7.14.2.1 enum [event_type_t](#)

Enumerator

[event_addContainer](#)
[event_destroyContainer](#)
[event_setState](#)
[event_pushState](#)
[event_popState](#)
[event_addEvent](#)
[event_startLink](#)
[event_endLink](#)
[event_setVar](#)
[event_addVar](#)
[event_subVar](#)

7.15 GTGStack.h File Reference

```
#include "GTGList.h"
```

Macros

- #define [GTG_STACK_INIT](#)(ptr) [GTG_LIST_INIT](#)(ptr)
- #define [GTG_STACK](#)(ptr) [GTG_LIST](#)(ptr)
- #define [gtg_stack_entry](#)(ptr, type, member) [gtg_list_entry](#)(ptr, type, member)

Typedefs

- typedef struct [gtg_list](#) [gtg_stack](#)
- typedef [gtg_stack](#) * [gtg_stack_t](#)

Functions

- static void [gtg_stack_push](#) ([gtg_stack_t](#) lnew, [gtg_stack_t](#) p_stack)
- static void [gtg_stack_pop](#) ([gtg_stack_t](#) p_stack)
- static [gtg_stack_t](#) [gtg_stack_top](#) ([gtg_stack_t](#) p_stack)
- static int [gtg_stack_empty](#) ([gtg_stack_t](#) p_stack)

7.15.1 Macro Definition Documentation

- 7.15.1.1 `#define GTG_STACK(ptr) GTG_LIST(ptr)`
- 7.15.1.2 `#define gtg_stack_entry(ptr, type, member) gtg_list_entry(ptr, type, member)`
- 7.15.1.3 `#define GTG_STACK_INIT(ptr) GTG_LIST_INIT(ptr)`

7.15.2 Typedef Documentation

- 7.15.2.1 typedef struct [gtg_list](#) [gtg_stack](#)
- 7.15.2.2 typedef [gtg_stack](#)* [gtg_stack_t](#)

7.15.3 Function Documentation

- 7.15.3.1 static int [gtg_stack_empty](#) ([gtg_stack_t](#) *p_stack*) `[inline], [static]`
- 7.15.3.2 static void [gtg_stack_pop](#) ([gtg_stack_t](#) *p_stack*) `[inline], [static]`
- 7.15.3.3 static void [gtg_stack_push](#) ([gtg_stack_t](#) *lnew*, [gtg_stack_t](#) *p_stack*) `[inline], [static]`
- 7.15.3.4 static [gtg_stack_t](#) [gtg_stack_top](#) ([gtg_stack_t](#) *p_stack*) `[inline], [static]`

7.16 GTGTypes.h File Reference

Typedefs

- typedef double [varPrec](#)
Use the double precision type for time and value.
- typedef enum [trace_return_t](#) [trace_return_t](#)

Enumerations

- enum [trace_return_t](#) {
TRACE_SUCCESS, TRACE_ERR_OPEN, TRACE_ERR_CLOSE, TRACE_ERR_WRITE,
TRACE_ERR_NOT_IMPL }
Define various return values.

7.16.1 Typedef Documentation

7.16.1.1 typedef enum trace_return_t trace_return_t